

FflasFfpack

Generated on Wed Jul 19 2023 00:00:00 for FflasFfpack by Doxygen 1.9.7

Wed Jul 19 2023 00:00:00

1 FFLAS-FFPACK Documentation.	1
1.1 Introduction	1
1.2 Goals	1
1.3 Design	1
1.4 Using FFLAS-FFPACK.	1
1.5 Contributing to fflas-ffpack, getting assistance.	2
1.6 Copying and Licence	2
1.7 Tutorial	2
1.8 Configuring and Installing FFLAS-FFPACK	2
1.9 Architecture of the library.	2
2 Bug List	3
3 Bibliography	7
4 Todo List	9
5 Module Index	13
5.1 Modules	13
6 Namespace Index	15
6.1 Namespace List	15
7 Hierarchical Index	17
7.1 Class Hierarchy	17
8 Data Structure Index	25
8.1 Data Structures	25
9 File Index	33
9.1 File List	33
10 Module Documentation	41
10.1 CHECKER	41
10.2 FFLAS-FFPACK	41
10.2.1 Detailed Description	41
10.2.2 FFLAS	42
10.2.3 Interfaces	42
10.3 Matrix Multiplication Algorithms	42
10.3.1 Detailed Description	42
10.4 SIMD wrapper	42
10.5 FFPACK	43
10.5.1 Detailed Description	43
10.6 FFLAS-FFPACK fields	43
10.6.1 Detailed Description	43
10.7 RNS	43

11 Namespace Documentation	45
11.1 FFLAS Namespace Reference	45
11.1.1 Typedef Documentation	70
11.1.1.1 Checker_fgemm	70
11.1.1.2 Checker_ftrsm	70
11.1.1.3 ForceCheck_fgemm	70
11.1.1.4 ForceCheck_ftrsm	70
11.1.1.5 ZOSparseMatrix	70
11.1.1.6 NotZOSparseMatrix	70
11.1.1.7 SimdSparseMatrix	71
11.1.1.8 NoSimdSparseMatrix	71
11.1.1.9 MKLSparseMatrixFormat	71
11.1.1.10 NotMKLSparseMatrixFormat	71
11.1.1.11 has_plus	71
11.1.1.12 has_minus	71
11.1.1.13 has_equal	71
11.1.1.14 has_plus_eq	71
11.1.1.15 has_minus_eq	71
11.1.1.16 has_mul	72
11.1.1.17 has_mul_eq	72
11.1.1.18 Timer	72
11.1.1.19 BaseTimer	72
11.1.1.20 UserTimer	72
11.1.1.21 SysTimer	72
11.1.2 Enumeration Type Documentation	72
11.1.2.1 FFLAS_ORDER	72
11.1.2.2 FFLAS_TRANSPOSE	73
11.1.2.3 FFLAS_UPLO	73
11.1.2.4 FFLAS_DIAG	73
11.1.2.5 FFLAS_SIDE	73
11.1.2.6 FFLAS_BASE	74
11.1.2.7 number_kind	74
11.1.2.8 SparseMatrix_t	74
11.1.2.9 FFLAS_FORMAT	75
11.1.3 Function Documentation	75
11.1.3.1 InfNorm()	75
11.1.3.2 min3()	75
11.1.3.3 max3()	75
11.1.3.4 min4()	75
11.1.3.5 max4()	76
11.1.3.6 fadd() [1/8]	76
11.1.3.7 faddin() [1/4]	76

11.1.3.8 fsub() [1/4]	76
11.1.3.9 fsubin() [1/3]	76
11.1.3.10 fadd() [2/8]	77
11.1.3.11 pfadd()	77
11.1.3.12 psub()	77
11.1.3.13 pfaddin()	78
11.1.3.14 psubin()	78
11.1.3.15 fadd() [3/8]	78
11.1.3.16 fsub() [2/4]	79
11.1.3.17 faddin() [2/4]	79
11.1.3.18 fsubin() [2/3]	79
11.1.3.19 fadd() [4/8]	80
11.1.3.20 fassign() [1/10]	80
11.1.3.21 fassign() [2/10]	81
11.1.3.22 fassign() [3/10]	81
11.1.3.23 fassign() [4/10]	81
11.1.3.24 fassign() [5/10]	81
11.1.3.25 fassign() [6/10]	82
11.1.3.26 fassign() [7/10]	82
11.1.3.27 fassign() [8/10]	82
11.1.3.28 faxpy() [1/6]	82
11.1.3.29 faxpy() [2/6]	83
11.1.3.30 faxpy() [3/6]	83
11.1.3.31 faxpy() [4/6]	83
11.1.3.32 fdot() [1/11]	84
11.1.3.33 fdot() [2/11]	84
11.1.3.34 fdot() [3/11]	84
11.1.3.35 fdot() [4/11]	85
11.1.3.36 fdot() [5/11]	85
11.1.3.37 fdot() [6/11]	85
11.1.3.38 fdot() [7/11]	85
11.1.3.39 fdot() [8/11]	86
11.1.3.40 fgemm() [1/23]	86
11.1.3.41 fgemm() [2/23]	86
11.1.3.42 fgemm() [3/23]	87
11.1.3.43 fgemm() [4/23]	87
11.1.3.44 fgemm() [5/23]	88
11.1.3.45 fgemm() [6/23]	88
11.1.3.46 fsquare() [1/6]	89
11.1.3.47 fsquare() [2/6]	89
11.1.3.48 fsquare() [3/6]	90
11.1.3.49 fsquare() [4/6]	90

11.1.3.50 fsquare()	[5/6]	90
11.1.3.51 fgemm()	[7/23]	91
11.1.3.52 fgemm()	[8/23]	91
11.1.3.53 fgemm()	[9/23]	91
11.1.3.54 fgemm()	[10/23]	92
11.1.3.55 fgemm()	[11/23]	92
11.1.3.56 fgemm()	[12/23]	93
11.1.3.57 fgemm()	[13/23]	93
11.1.3.58 fgemm()	[14/23]	93
11.1.3.59 fgemm()	[15/23]	94
11.1.3.60 fgemm()	[16/23]	94
11.1.3.61 fgemm()	[17/23]	94
11.1.3.62 fgemm()	[18/23]	95
11.1.3.63 fgemv()	[1/19]	95
11.1.3.64 fgemv()	[2/19]	96
11.1.3.65 fgemv()	[3/19]	96
11.1.3.66 fgemv()	[4/19]	96
11.1.3.67 fgemv()	[5/19]	97
11.1.3.68 fgemv()	[6/19]	97
11.1.3.69 fgemv()	[7/19]	98
11.1.3.70 fgemv()	[8/19]	98
11.1.3.71 fgemv()	[9/19]	98
11.1.3.72 fgemv()	[10/19]	99
11.1.3.73 fgemv()	[11/19]	99
11.1.3.74 fgemv()	[12/19]	99
11.1.3.75 fgemv()	[13/19]	100
11.1.3.76 fgemv()	[14/19]	100
11.1.3.77 fgemv()	[15/19]	100
11.1.3.78 fgemv()	[16/19]	101
11.1.3.79 fger()	[1/12]	101
11.1.3.80 fger()	[2/12]	102
11.1.3.81 fger()	[3/12]	102
11.1.3.82 fger()	[4/12]	102
11.1.3.83 fger()	[5/12]	103
11.1.3.84 fger()	[6/12]	103
11.1.3.85 fger()	[7/12]	103
11.1.3.86 fger()	[8/12]	104
11.1.3.87 fger()	[9/12]	104
11.1.3.88 fger()	[10/12]	104
11.1.3.89 fger()	[11/12]	105
11.1.3.90 freduce()	[1/10]	105
11.1.3.91 freduce()	[2/10]	105

11.1.3.92 <code>freduce_constoverride()</code> [1/2]	106
11.1.3.93 <code>finit()</code> [1/8]	106
11.1.3.94 <code>finit()</code> [2/8]	106
11.1.3.95 <code>freduce()</code> [3/10]	107
11.1.3.96 <code>pfreduce()</code>	107
11.1.3.97 <code>freduce()</code> [4/10]	107
11.1.3.98 <code>freduce_constoverride()</code> [2/2]	108
11.1.3.99 <code>finit()</code> [3/8]	108
11.1.3.100 <code>finit()</code> [4/8]	108
11.1.3.101 <code>freduce()</code> [5/10]	108
11.1.3.102 <code>freduce()</code> [6/10]	109
11.1.3.103 <code>freivalds()</code>	109
11.1.3.104 <code>fscalin()</code> [1/10]	109
11.1.3.105 <code>fscal()</code> [1/10]	110
11.1.3.106 <code>fscal()</code> [2/10]	111
11.1.3.107 <code>fscal()</code> [3/10]	111
11.1.3.108 <code>fscalin()</code> [2/10]	111
11.1.3.109 <code>fscalin()</code> [3/10]	111
11.1.3.110 <code>fscalin()</code> [4/10]	111
11.1.3.111 <code>fscal()</code> [4/10]	112
11.1.3.112 <code>fscalin()</code> [5/10]	112
11.1.3.113 <code>fscal()</code> [5/10]	112
11.1.3.114 <code>fscalin()</code> [6/10]	113
11.1.3.115 <code>fscal()</code> [6/10]	113
11.1.3.116 <code>fscalin()</code> [7/10]	113
11.1.3.117 <code>fscal()</code> [7/10]	113
11.1.3.118 <code>fscalin()</code> [8/10]	114
11.1.3.119 <code>fscal()</code> [8/10]	114
11.1.3.120 <code>fsyr2k()</code>	114
11.1.3.121 <code>fsyrk()</code> [1/5]	115
11.1.3.122 <code>fsyrk()</code> [2/5]	116
11.1.3.123 <code>fsyrk()</code> [3/5]	116
11.1.3.124 <code>fsyrk()</code> [4/5]	117
11.1.3.125 <code>fsyrk()</code> [5/5]	117
11.1.3.126 <code>ftmm()</code> [1/3]	118
11.1.3.127 <code>ftmm()</code> [2/3]	119
11.1.3.128 <code>ftsm()</code> [1/9]	120
11.1.3.129 <code>ftsm()</code> [2/9]	120
11.1.3.130 <code>ftsm()</code> [3/9]	120
11.1.3.131 <code>ftsm()</code> [4/9]	121
11.1.3.132 <code>ftsm()</code> [5/9]	121
11.1.3.133 <code>cblas_imptrsm()</code>	121

11.1.3.134 ftrsv()	[1/2]	122
11.1.3.135 igemm_()		122
11.1.3.136 finit()	[5/8]	122
11.1.3.137 fconvert()	[1/3]	123
11.1.3.138 fnegin()	[1/4]	123
11.1.3.139 fneg()	[1/4]	124
11.1.3.140 fzero()	[1/4]	124
11.1.3.141 frand()	[1/2]	125
11.1.3.142 fiszero()	[1/4]	125
11.1.3.143 fequal()	[1/4]	125
11.1.3.144 faxpby()	[1/2]	126
11.1.3.145 fdot()	[9/11]	126
11.1.3.146 fswap()	[1/2]	127
11.1.3.147 fzero()	[2/4]	127
11.1.3.148 frand()	[2/2]	128
11.1.3.149 fequal()	[2/4]	128
11.1.3.150 fiszero()	[2/4]	129
11.1.3.151 fidentity()	[1/4]	129
11.1.3.152 fidentity()	[2/4]	129
11.1.3.153 finit()	[6/8]	129
11.1.3.154 fconvert()	[2/3]	130
11.1.3.155 fnegin()	[2/4]	130
11.1.3.156 fneg()	[2/4]	131
11.1.3.157 faxpby()	[2/2]	131
11.1.3.158 fmove()	[1/2]	132
11.1.3.159 bitsize()		132
11.1.3.160 bitsize< Givaro::ZRing< Givaro::Integer > >()		133
11.1.3.161 ftrmv()		133
11.1.3.162 ftrsm()	[6/9]	133
11.1.3.163 pfgemm()	[1/7]	134
11.1.3.164 pfgemm_1D_rec()		134
11.1.3.165 pfgemm_2D_rec()		135
11.1.3.166 pfgemm_3D_rec()		135
11.1.3.167 pfgemm_3D_rec2()		136
11.1.3.168 fgemm()	[19/23]	136
11.1.3.169 ftrsm()	[7/9]	136
11.1.3.170 ftrsm()	[8/9]	137
11.1.3.171 fspmv()	[1/2]	137
11.1.3.172 fspmm()		137
11.1.3.173 sparse_init()	[1/16]	138
11.1.3.174 sparse_init()	[2/16]	138
11.1.3.175 sparse_delete()	[1/12]	138

11.1.3.176 <code>sparse_delete()</code> [2/12]	138
11.1.3.177 <code>sparse_init()</code> [3/16]	138
11.1.3.178 <code>sparse_init()</code> [4/16]	139
11.1.3.179 <code>sparse_delete()</code> [3/12]	139
11.1.3.180 <code>sparse_delete()</code> [4/12]	139
11.1.3.181 <code>sparse_print()</code> [1/3]	139
11.1.3.182 <code>sparse_init()</code> [5/16]	139
11.1.3.183 <code>sparse_init()</code> [6/16]	140
11.1.3.184 <code>sparse_init()</code> [7/16]	140
11.1.3.185 <code>sparse_init()</code> [8/16]	140
11.1.3.186 <code>sparse_delete()</code> [5/12]	140
11.1.3.187 <code>sparse_init()</code> [9/16]	141
11.1.3.188 <code>sparse_init()</code> [10/16]	141
11.1.3.189 <code>sparse_init()</code> [11/16]	141
11.1.3.190 <code>sparse_delete()</code> [6/12]	141
11.1.3.191 <code>sparse_delete()</code> [7/12]	141
11.1.3.192 <code>sparse_init()</code> [12/16]	142
11.1.3.193 <code>sparse_init()</code> [13/16]	142
11.1.3.194 <code>sparse_delete()</code> [8/12]	142
11.1.3.195 <code>sparse_delete()</code> [9/12]	142
11.1.3.196 <code>sparse_print()</code> [2/3]	142
11.1.3.197 <code>sparse_delete()</code> [10/12]	142
11.1.3.198 <code>sparse_init()</code> [14/16]	143
11.1.3.199 <code>operator<<()</code>	143
11.1.3.200 <code>readSmsFormat()</code>	143
11.1.3.201 <code>readSprFormat()</code>	143
11.1.3.202 <code>getDataType()</code> [1/4]	143
11.1.3.203 <code>getDataType()</code> [2/4]	144
11.1.3.204 <code>getDataType()</code> [3/4]	144
11.1.3.205 <code>getDataType()</code> [4/4]	144
11.1.3.206 <code>readMachineType()</code>	144
11.1.3.207 <code>readDnsFormat()</code>	144
11.1.3.208 <code>writeDnsFormat()</code>	144
11.1.3.209 <code>fspmV()</code> [2/2]	145
11.1.3.210 <code>sparse_delete()</code> [11/12]	145
11.1.3.211 <code>sparse_delete()</code> [12/12]	145
11.1.3.212 <code>sparse_print()</code> [3/3]	145
11.1.3.213 <code>sparse_init()</code> [15/16]	145
11.1.3.214 <code>sparse_init()</code> [16/16]	145
11.1.3.215 <code>computeDeviation()</code>	146
11.1.3.216 <code>getStat()</code>	146
11.1.3.217 <code>fflas_delete()</code> [1/4]	146

11.1.3.218 fflas_delete() [2/4]	146
11.1.3.219 fflas_new() [1/7]	146
11.1.3.220 fflas_new() [2/7]	146
11.1.3.221 finit_rns() [1/2]	147
11.1.3.222 finit_trans_rns()	147
11.1.3.223 fconvert_rns() [1/2]	147
11.1.3.224 fconvert_trans_rns()	147
11.1.3.225 fflas_new() [3/7]	147
11.1.3.226 fflas_new() [4/7]	148
11.1.3.227 finit_rns() [2/2]	148
11.1.3.228 fconvert_rns() [2/2]	148
11.1.3.229 freduce() [7/10]	148
11.1.3.230 freduce() [8/10]	149
11.1.3.231 finit() [7/8]	149
11.1.3.232 fconvert() [3/3]	149
11.1.3.233 fnegin() [3/4]	150
11.1.3.234 fneg() [3/4]	150
11.1.3.235 fzero() [3/4]	151
11.1.3.236 fiszero() [3/4]	151
11.1.3.237 fequal() [3/4]	152
11.1.3.238 fassign() [9/10]	152
11.1.3.239 fscal() [9/10]	152
11.1.3.240 fscal() [9/10]	153
11.1.3.241 faxpy() [5/6]	154
11.1.3.242 fdot() [10/11]	154
11.1.3.243 fswap() [2/2]	155
11.1.3.244 fadd() [5/8]	155
11.1.3.245 fsub() [3/4]	155
11.1.3.246 faddin() [3/4]	156
11.1.3.247 fadd() [6/8]	156
11.1.3.248 fassign() [10/10]	156
11.1.3.249 fzero() [4/4]	157
11.1.3.250 fequal() [4/4]	157
11.1.3.251 fiszero() [4/4]	157
11.1.3.252 fidentity() [3/4]	158
11.1.3.253 fidentity() [4/4]	158
11.1.3.254 freduce() [9/10]	158
11.1.3.255 freduce() [10/10]	159
11.1.3.256 finit() [8/8]	159
11.1.3.257 fnegin() [4/4]	160
11.1.3.258 fneg() [4/4]	160
11.1.3.259 fscal() [10/10]	160

11.1.3.260 fscal() [10/10]	161
11.1.3.261 faxpy() [6/6]	161
11.1.3.262 fmove() [2/2]	162
11.1.3.263 fadd() [7/8]	163
11.1.3.264 fsub() [4/4]	163
11.1.3.265 fsubin() [3/3]	164
11.1.3.266 fadd() [8/8]	164
11.1.3.267 faddin() [4/4]	165
11.1.3.268 fgemv() [17/19]	165
11.1.3.269 fger() [12/12]	165
11.1.3.270 ftrsv() [2/2]	166
11.1.3.271 ftrsm() [9/9]	167
11.1.3.272 ftrmm() [3/3]	167
11.1.3.273 fgemm() [20/23]	168
11.1.3.274 fgemm() [21/23]	169
11.1.3.275 fgemm() [22/23]	169
11.1.3.276 fgemm() [23/23]	170
11.1.3.277 fsquare() [6/6]	170
11.1.3.278 BlockCuts() [1/2]	170
11.1.3.279 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()	171
11.1.3.280 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()	171
11.1.3.281 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()	171
11.1.3.282 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()	171
11.1.3.283 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()	172
11.1.3.284 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()	172
11.1.3.285 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()	172
11.1.3.286 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()	172
11.1.3.287 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()	172
11.1.3.288 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()	173
11.1.3.289 BlockCuts() [2/2]	173
11.1.3.290 pfzero()	173
11.1.3.291 pfrand()	173
11.1.3.292 fdot() [11/11]	174
11.1.3.293 pfgemm() [2/7]	174
11.1.3.294 pfgemm() [3/7]	174
11.1.3.295 pfgemm() [4/7]	175
11.1.3.296 pfgemm() [5/7]	175
11.1.3.297 pfgemm() [6/7]	175
11.1.3.298 pfgemm() [7/7]	176
11.1.3.299 fgemv() [18/19]	176
11.1.3.300 fgemv() [19/19]	176
11.1.3.301 parseArguments()	177

11.1.3.302 writeCommandString()	177
11.1.3.303 WriteMatrix() [1/2]	177
11.1.3.304 preamble()	178
11.1.3.305 ReadMatrix() [1/2]	178
11.1.3.306 ReadMatrix() [2/2]	178
11.1.3.307 WriteMatrix() [2/2]	179
11.1.3.308 WritePermutation()	179
11.1.3.309 alignable()	179
11.1.3.310 alignable< Givaro::Integer * >()	179
11.1.3.311 fflas_new() [5/7]	180
11.1.3.312 fflas_new() [6/7]	180
11.1.3.313 fflas_new() [7/7]	180
11.1.3.314 fflas_delete() [3/4]	180
11.1.3.315 fflas_delete() [4/4]	180
11.1.3.316 prefetch()	180
11.1.3.317 getTLBSize()	180
11.1.3.318 queryCacheSizes()	181
11.1.3.319 queryL1CacheSize()	181
11.1.3.320 queryTopLevelCacheSize()	181
11.1.3.321 getSeed()	181
11.2 FFLAS::BLAS3 Namespace Reference	181
11.2.1 Function Documentation	183
11.2.1.1 Bini()	183
11.2.1.2 WinoPar()	183
11.2.1.3 Winograd()	184
11.2.1.4 WinogradAcc_3_23()	184
11.2.1.5 WinogradAcc_3_21()	184
11.2.1.6 WinogradAcc_2_24()	185
11.2.1.7 WinogradAcc_2_27()	185
11.2.1.8 WinogradAcc_LR()	185
11.2.1.9 WinogradAcc_R_S()	186
11.2.1.10 WinogradAcc_L_S()	186
11.2.1.11 Winograd_LR_S()	187
11.2.1.12 Winograd_L_S()	187
11.2.1.13 Winograd_R_S()	187
11.3 FFLAS::csr_hyb_details Namespace Reference	188
11.4 FFLAS::CuttingStrategy Namespace Reference	188
11.4.1 Typedef Documentation	188
11.4.1.1 RNSModulus	188
11.5 FFLAS::details Namespace Reference	188
11.5.1 Function Documentation	190
11.5.1.1 fadd() [1/5]	190

11.5.1.2 fadd() [2/5]	190
11.5.1.3 fadd() [3/5]	190
11.5.1.4 fadd() [4/5]	191
11.5.1.5 fadd() [5/5]	191
11.5.1.6 freduce() [1/4]	191
11.5.1.7 freduce() [2/4]	191
11.5.1.8 freduce() [3/4]	192
11.5.1.9 freduce() [4/4]	192
11.5.1.10 fscaln() [1/2]	192
11.5.1.11 fscal() [1/2]	192
11.5.1.12 fscaln() [2/2]	193
11.5.1.13 fscal() [2/2]	193
11.5.1.14 igebb44()	193
11.5.1.15 igebb24()	193
11.5.1.16 igebb14()	194
11.5.1.17 igebb41()	194
11.5.1.18 igebb21()	194
11.5.1.19 igebb11()	194
11.5.1.20 igebp()	195
11.5.1.21 pack_lhs()	195
11.5.1.22 pack_rhs()	195
11.5.1.23 gebp()	196
11.5.1.24 BlockingFactor()	196
11.6 FFLAS::details_spmv Namespace Reference	196
11.7 FFLAS::ElementCategories Namespace Reference	196
11.8 FFLAS::FieldCategories Namespace Reference	196
11.8.1 Detailed Description	197
11.9 FFLAS::MMHelperAlgo Namespace Reference	197
11.10 FFLAS::ModeCategories Namespace Reference	197
11.10.1 Detailed Description	197
11.11 FFLAS::ParSeqHelper Namespace Reference	198
11.11.1 Detailed Description	198
11.12 FFLAS::Protected Namespace Reference	198
11.12.1 Function Documentation	201
11.12.1.1 computeFactorClassic() [1/3]	201
11.12.1.2 computeFactorClassic() [2/3]	201
11.12.1.3 computeFactorClassic() [3/3]	201
11.12.1.4 DotProdBoundClassic()	201
11.12.1.5 TRSMBound() [1/3]	201
11.12.1.6 TRSMBound() [2/3]	202
11.12.1.7 TRSMBound() [3/3]	202
11.12.1.8 fgemm_convert()	202

11.12.1.9 NeedPreAddReduction() [1/2]	203
11.12.1.10 NeedPreAddReduction() [2/2]	203
11.12.1.11 NeedPreSubReduction() [1/2]	203
11.12.1.12 NeedPreSubReduction() [2/2]	203
11.12.1.13 NeedDoublePreAddReduction() [1/2]	204
11.12.1.14 NeedDoublePreAddReduction() [2/2]	204
11.12.1.15 ScalAndReduce() [1/2]	204
11.12.1.16 ScalAndReduce() [2/2]	204
11.12.1.17 fsquareCommon()	205
11.12.1.18 WinogradThreshold() [1/4]	205
11.12.1.19 WinogradThreshold() [2/4]	205
11.12.1.20 WinogradThreshold() [3/4]	205
11.12.1.21 WinogradThreshold() [4/4]	205
11.12.1.22 WinogradSteps()	205
11.12.1.23 DynamicPeeling()	206
11.12.1.24 DynamicPeeling2()	206
11.12.1.25 WinogradCalc()	207
11.12.1.26 fgemv_convert()	207
11.12.1.27 fger_convert()	207
11.12.1.28 min_types() [1/7]	208
11.12.1.29 min_types() [2/7]	208
11.12.1.30 min_types() [3/7]	208
11.12.1.31 min_types() [4/7]	208
11.12.1.32 min_types() [5/7]	208
11.12.1.33 min_types() [6/7]	208
11.12.1.34 min_types() [7/7]	208
11.12.1.35 unfit() [1/4]	208
11.12.1.36 unfit() [2/4]	209
11.12.1.37 unfit() [3/4]	209
11.12.1.38 unfit() [4/4]	209
11.12.1.39 igemm_colmajor() [1/2]	209
11.12.1.40 igemm_colmajor() [2/2]	209
11.12.1.41 igemm()	210
11.12.1.42 MatF2MatD_Triangular()	210
11.12.1.43 MatF2MatFI_Triangular()	210
11.13 FFLAS::sell_details Namespace Reference	211
11.14 FFLAS::sparse_details Namespace Reference	211
11.14.1 Function Documentation	214
11.14.1.1 init_y() [1/2]	214
11.14.1.2 init_y() [2/2]	214
11.14.1.3 fspmv_dispatch() [1/2]	214
11.14.1.4 fspmv_dispatch() [2/2]	214

11.14.1.5 fspmv()	[1/12]	215
11.14.1.6 fspmv()	[2/12]	215
11.14.1.7 fspmv()	[3/12]	215
11.14.1.8 fspmv()	[4/12]	215
11.14.1.9 fspmv()	[5/12]	215
11.14.1.10 fspmv()	[6/12]	216
11.14.1.11 fspmv()	[7/12]	216
11.14.1.12 fspmv()	[8/12]	216
11.14.1.13 fspmv()	[9/12]	216
11.14.1.14 fspmm_dispatch()	[1/2]	217
11.14.1.15 fspmm_dispatch()	[2/2]	217
11.14.1.16 fspmm()	[1/9]	217
11.14.1.17 fspmm()	[2/9]	217
11.14.1.18 fspmm()	[3/9]	218
11.14.1.19 fspmm()	[4/9]	218
11.14.1.20 fspmm()	[5/9]	218
11.14.1.21 fspmm()	[6/9]	219
11.14.1.22 fspmm()	[7/9]	219
11.14.1.23 fspmm()	[8/9]	219
11.14.1.24 fspmm()	[9/9]	219
11.14.1.25 pfspmm_dispatch()	[1/2]	220
11.14.1.26 pfspmm_dispatch()	[2/2]	220
11.14.1.27 pfspmm()	[1/9]	220
11.14.1.28 pfspmm()	[2/9]	220
11.14.1.29 pfspmm()	[3/9]	221
11.14.1.30 pfspmm()	[4/9]	221
11.14.1.31 pfspmm()	[5/9]	221
11.14.1.32 pfspmm()	[6/9]	222
11.14.1.33 pfspmm()	[7/9]	222
11.14.1.34 pfspmm()	[8/9]	222
11.14.1.35 pfspmm()	[9/9]	222
11.14.1.36 pfspmv()	[1/6]	223
11.14.1.37 pfspmv()	[2/6]	223
11.14.1.38 pfspmv()	[3/6]	223
11.14.1.39 pfspmv()	[4/6]	223
11.14.1.40 pfspmv()	[5/6]	223
11.14.1.41 pfspmv()	[6/6]	224
11.14.1.42 fspmv()	[10/12]	224
11.14.1.43 fspmv()	[11/12]	224
11.14.1.44 fspmv()	[12/12]	224
11.15 FFLAS::sparse_details_impl Namespace Reference		225
11.15.1 Function Documentation		233

11.15.1.1 fspmm()	[1/15]	233
11.15.1.2 fspmm()	[2/15]	233
11.15.1.3 fspmm()	[3/15]	234
11.15.1.4 fspmm_simd_aligned()	[1/2]	234
11.15.1.5 fspmm_simd_unaligned()	[1/2]	234
11.15.1.6 fspmm_one()	[1/4]	234
11.15.1.7 fspmm_mone()	[1/4]	235
11.15.1.8 fspmm_one_simd_aligned()	[1/3]	235
11.15.1.9 fspmm_one_simd_unaligned()	[1/3]	235
11.15.1.10 fspmm_mone_simd_aligned()	[1/3]	235
11.15.1.11 fspmm_mone_simd_unaligned()	[1/3]	236
11.15.1.12 fspmv()	[1/21]	236
11.15.1.13 fspmv()	[2/21]	236
11.15.1.14 fspmv()	[3/21]	236
11.15.1.15 fspmv_one()	[1/10]	236
11.15.1.16 fspmv_mone()	[1/10]	237
11.15.1.17 fspmv_one()	[2/10]	237
11.15.1.18 fspmv_mone()	[2/10]	237
11.15.1.19 pfspmm()	[1/18]	237
11.15.1.20 pfspmm()	[2/18]	237
11.15.1.21 pfspmm()	[3/18]	238
11.15.1.22 pfspmm_one()	[1/2]	238
11.15.1.23 pfspmm_mone()	[1/2]	238
11.15.1.24 pfspmm_one()	[2/2]	238
11.15.1.25 pfspmm_mone()	[2/2]	239
11.15.1.26 pfspmv()	[1/18]	239
11.15.1.27 pfspmv_task()		239
11.15.1.28 pfspmv()	[2/18]	239
11.15.1.29 pfspmv()	[3/18]	239
11.15.1.30 pfspmv_one()	[1/8]	240
11.15.1.31 pfspmv_mone()	[1/8]	240
11.15.1.32 pfspmv_one()	[2/8]	240
11.15.1.33 pfspmv_mone()	[2/8]	240
11.15.1.34 fspmm()	[4/15]	240
11.15.1.35 fspmm()	[5/15]	241
11.15.1.36 fspmm_simd_aligned()	[2/2]	241
11.15.1.37 fspmm_simd_unaligned()	[2/2]	241
11.15.1.38 fspmm()	[6/15]	241
11.15.1.39 fspmm_one()	[2/4]	242
11.15.1.40 fspmm_mone()	[2/4]	242
11.15.1.41 fspmm_one_simd_aligned()	[2/3]	242
11.15.1.42 fspmm_one_simd_unaligned()	[2/3]	242

11.15.1.43 fspmm_mone_simd_aligned() [2/3]	243
11.15.1.44 fspmm_mone_simd_unaligned() [2/3]	243
11.15.1.45 fspmv() [4/21]	243
11.15.1.46 fspmv() [5/21]	243
11.15.1.47 fspmv() [6/21]	243
11.15.1.48 fspmv_one() [3/10]	244
11.15.1.49 fspmv_mone() [3/10]	244
11.15.1.50 fspmv_one() [4/10]	244
11.15.1.51 fspmv_mone() [4/10]	244
11.15.1.52 pfspmm() [4/18]	244
11.15.1.53 pfspmm() [5/18]	245
11.15.1.54 pfspmm() [6/18]	245
11.15.1.55 pfspmm() [7/18]	245
11.15.1.56 pfspmm() [8/18]	245
11.15.1.57 pfspmm() [9/18]	246
11.15.1.58 pfspmv() [4/18]	246
11.15.1.59 pfspmv() [5/18]	246
11.15.1.60 pfspmv() [6/18]	246
11.15.1.61 fspmm() [7/15]	246
11.15.1.62 fspmm() [8/15]	247
11.15.1.63 fspmm() [9/15]	247
11.15.1.64 fspmv() [7/21]	247
11.15.1.65 fspmv() [8/21]	247
11.15.1.66 fspmv() [9/21]	247
11.15.1.67 pfspmm() [10/18]	248
11.15.1.68 pfspmm() [11/18]	248
11.15.1.69 pfspmm() [12/18]	248
11.15.1.70 pfspmm() [13/18]	248
11.15.1.71 pfspmm() [14/18]	249
11.15.1.72 pfspmm() [15/18]	249
11.15.1.73 pfspmm_zo() [1/2]	249
11.15.1.74 pfspmm_zo() [2/2]	249
11.15.1.75 pfspmv() [7/18]	250
11.15.1.76 pfspmv() [8/18]	250
11.15.1.77 pfspmv() [9/18]	250
11.15.1.78 pfspmv_one() [3/8]	250
11.15.1.79 pfspmv_mone() [3/8]	250
11.15.1.80 pfspmv_one() [4/8]	251
11.15.1.81 pfspmv_mone() [4/8]	251
11.15.1.82 fspmm() [10/15]	251
11.15.1.83 fspmm() [11/15]	251
11.15.1.84 fspmm() [12/15]	252

11.15.1.85 fspmm_mone() [3/4]	252
11.15.1.86 fspmm_one() [3/4]	252
11.15.1.87 fspmm_mone() [4/4]	252
11.15.1.88 fspmm_one() [4/4]	253
11.15.1.89 fspmm_one_simd_aligned() [3/3]	253
11.15.1.90 fspmm_one_simd_unaligned() [3/3]	253
11.15.1.91 fspmm_mone_simd_aligned() [3/3]	253
11.15.1.92 fspmm_mone_simd_unaligned() [3/3]	254
11.15.1.93 fspmv() [10/21]	254
11.15.1.94 fspmv() [11/21]	254
11.15.1.95 fspmv() [12/21]	254
11.15.1.96 fspmv_one() [5/10]	254
11.15.1.97 fspmv_mone() [5/10]	255
11.15.1.98 fspmv_one() [6/10]	255
11.15.1.99 fspmv_mone() [6/10]	255
11.15.1.100 pfspmv() [10/18]	255
11.15.1.101 pfspmv() [11/18]	255
11.15.1.102 pfspmv() [12/18]	256
11.15.1.103 pfspmv_one() [5/8]	256
11.15.1.104 pfspmv_mone() [5/8]	256
11.15.1.105 pfspmv_one() [6/8]	256
11.15.1.106 pfspmv_mone() [6/8]	256
11.15.1.107 fspmv() [13/21]	257
11.15.1.108 fspmv_simd() [1/4]	257
11.15.1.109 fspmv() [14/21]	257
11.15.1.110 fspmv_simd() [2/4]	257
11.15.1.111 fspmv() [15/21]	257
11.15.1.112 fspmv_one() [7/10]	258
11.15.1.113 fspmv_mone() [7/10]	258
11.15.1.114 fspmv_one() [8/10]	258
11.15.1.115 fspmv_mone() [8/10]	258
11.15.1.116 fspmv_one_simd() [1/2]	258
11.15.1.117 fspmv_mone_simd() [1/2]	259
11.15.1.118 pfspmm() [16/18]	259
11.15.1.119 pfspmm() [17/18]	259
11.15.1.120 pfspmm() [18/18]	259
11.15.1.121 pfspmv() [13/18]	260
11.15.1.122 pfspmv() [14/18]	260
11.15.1.123 pfspmv() [15/18]	260
11.15.1.124 fspmm() [13/15]	260
11.15.1.125 fspmm() [14/15]	260
11.15.1.126 fspmm() [15/15]	261

11.15.1.127 fspmv() [16/21]	261
11.15.1.128 fspmv() [17/21]	261
11.15.1.129 fspmv() [18/21]	261
11.15.1.130 pfspmv() [16/18]	261
11.15.1.131 pfspmv() [17/18]	262
11.15.1.132 pfspmv() [18/18]	262
11.15.1.133 pfspmv_one() [7/8]	262
11.15.1.134 pfspmv_mone() [7/8]	262
11.15.1.135 pfspmv_one() [8/8]	262
11.15.1.136 pfspmv_mone() [8/8]	263
11.15.1.137 fspmv() [19/21]	263
11.15.1.138 fspmv_simd() [3/4]	263
11.15.1.139 fspmv() [20/21]	263
11.15.1.140 fspmv_simd() [4/4]	263
11.15.1.141 fspmv() [21/21]	264
11.15.1.142 fspmv_one() [9/10]	264
11.15.1.143 fspmv_mone() [9/10]	264
11.15.1.144 fspmv_one_simd() [2/2]	264
11.15.1.145 fspmv_mone_simd() [2/2]	264
11.15.1.146 fspmv_one() [10/10]	265
11.15.1.147 fspmv_mone() [10/10]	265
11.16 FFLAS::StrategyParameter Namespace Reference	265
11.17 FFLAS::StructureHelper Namespace Reference	265
11.17.1 Detailed Description	265
11.18 FFLAS::vectorised Namespace Reference	266
11.18.1 Function Documentation	267
11.18.1.1 VEC_ADD()	267
11.18.1.2 addp()	267
11.18.1.3 VEC_SUB()	267
11.18.1.4 subp()	268
11.18.1.5 add()	268
11.18.1.6 sub()	268
11.18.1.7 reduce() [1/9]	268
11.18.1.8 reduce() [2/9]	268
11.18.1.9 reduce() [3/9]	268
11.18.1.10 reduce() [4/9]	269
11.18.1.11 reduce() [5/9]	269
11.18.1.12 reduce() [6/9]	269
11.18.1.13 reduce() [7/9]	269
11.18.1.14 reduce() [8/9]	269
11.18.1.15 reduce() [9/9]	269
11.18.1.16 modp() [1/2]	270

11.18.1.17 modp() [2/2]	270
11.18.1.18 scalp() [1/2]	270
11.18.1.19 scalp() [2/2]	270
11.19 FFLAS::vectorised::unswitch Namespace Reference	270
11.19.1 Function Documentation	271
11.19.1.1 modp() [1/2]	271
11.19.1.2 modp() [2/2]	271
11.19.1.3 scalp() [1/2]	271
11.19.1.4 scalp() [2/2]	272
11.20 FFPACK Namespace Reference	272
11.20.1 Detailed Description	287
11.20.2 Typedef Documentation	287
11.20.2.1 Checker_PLUQ	287
11.20.2.2 Checker_Det	287
11.20.2.3 Checker_invert	288
11.20.2.4 Checker_charpoly	288
11.20.2.5 ForceCheck_PLUQ	288
11.20.2.6 ForceCheck_Det	288
11.20.2.7 ForceCheck_invert	288
11.20.2.8 ForceCheck_charpoly	288
11.20.3 Function Documentation	288
11.20.3.1 LAPACKPerm2MathPerm()	288
11.20.3.2 MathPerm2LAPACKPerm()	289
11.20.3.3 applyP() [1/4]	289
11.20.3.4 applyP() [2/4]	290
11.20.3.5 applyP() [3/4]	290
11.20.3.6 MonotonicApplyP()	290
11.20.3.7 fgetrs() [1/4]	291
11.20.3.8 fgetrs() [2/4]	292
11.20.3.9 fgesv() [1/4]	293
11.20.3.10 fgesv() [2/4]	293
11.20.3.11 ftrtri() [1/2]	294
11.20.3.12 trinv_left() [1/2]	294
11.20.3.13 ftrtrm() [1/2]	295
11.20.3.14 ftrstr()	295
11.20.3.15 ftrssyr2k()	296
11.20.3.16 fsytrf() [1/3]	296
11.20.3.17 fsytrf() [2/3]	297
11.20.3.18 fsytrf() [3/3]	297
11.20.3.19 fsytrf_nonunit() [1/3]	298
11.20.3.20 PLUQ() [1/6]	298
11.20.3.21 pPLUQ()	299

11.20.3.22 PLUQ() [2/6]	299
11.20.3.23 PLUQ() [3/6]	300
11.20.3.24 LUdivine() [1/4]	300
11.20.3.25 ColumnEchelonForm() [1/3]	301
11.20.3.26 pColumnEchelonForm()	301
11.20.3.27 ColumnEchelonForm() [2/3]	302
11.20.3.28 RowEchelonForm() [1/3]	302
11.20.3.29 pRowEchelonForm()	302
11.20.3.30 RowEchelonForm() [2/3]	303
11.20.3.31 ReducedColumnEchelonForm() [1/3]	303
11.20.3.32 pReducedColumnEchelonForm()	304
11.20.3.33 ReducedColumnEchelonForm() [2/3]	304
11.20.3.34 ReducedRowEchelonForm() [1/3]	304
11.20.3.35 pReducedRowEchelonForm()	305
11.20.3.36 ReducedRowEchelonForm() [2/3]	305
11.20.3.37 Invert() [1/4]	305
11.20.3.38 Invert() [2/4]	306
11.20.3.39 Invert2() [1/2]	306
11.20.3.40 CharPoly() [1/8]	307
11.20.3.41 CharPoly() [2/8]	308
11.20.3.42 CharPoly() [3/8]	308
11.20.3.43 MinPoly() [1/4]	309
11.20.3.44 MinPoly() [2/4]	309
11.20.3.45 MatVecMinPoly() [1/2]	310
11.20.3.46 Rank() [1/3]	310
11.20.3.47 pRank()	311
11.20.3.48 Rank() [2/3]	311
11.20.3.49 IsSingular() [1/2]	311
11.20.3.50 Det() [1/6]	312
11.20.3.51 pDet()	312
11.20.3.52 Det() [2/6]	313
11.20.3.53 Solve() [1/3]	313
11.20.3.54 Solve() [2/3]	313
11.20.3.55 pSolve()	314
11.20.3.56 RandomNullSpaceVector() [1/3]	314
11.20.3.57 NullSpaceBasis() [1/2]	315
11.20.3.58 RowRankProfile() [1/3]	315
11.20.3.59 pRowRankProfile()	316
11.20.3.60 RowRankProfile() [2/3]	316
11.20.3.61 ColumnRankProfile() [1/3]	316
11.20.3.62 pColumnRankProfile()	317
11.20.3.63 ColumnRankProfile() [2/3]	317

11.20.3.64 RankProfileFromLU()	317
11.20.3.65 LeadingSubmatrixRankProfiles()	318
11.20.3.66 RowRankProfileSubmatrixIndices() [1/2]	318
11.20.3.67 ColRankProfileSubmatrixIndices() [1/2]	319
11.20.3.68 RowRankProfileSubmatrix() [1/2]	320
11.20.3.69 ColRankProfileSubmatrix() [1/2]	320
11.20.3.70 getTriangular() [1/2]	321
11.20.3.71 getTriangular() [2/2]	322
11.20.3.72 getEchelonForm() [1/2]	322
11.20.3.73 getEchelonForm() [2/2]	323
11.20.3.74 getEchelonTransform()	324
11.20.3.75 getReducedEchelonForm() [1/2]	325
11.20.3.76 getReducedEchelonForm() [2/2]	325
11.20.3.77 getReducedEchelonTransform()	326
11.20.3.78 PLUQtoEchelonPermutation()	327
11.20.3.79 LQUPtoInverseOfFullRankMinor() [1/2]	327
11.20.3.80 RandomNullSpaceVector() [2/3]	328
11.20.3.81 solveLB() [1/2]	328
11.20.3.82 solveLB2() [1/2]	328
11.20.3.83 Danilevski()	329
11.20.3.84 buildMatrix()	329
11.20.3.85 CharPoly() [4/8]	329
11.20.3.86 CharPoly() [5/8]	330
11.20.3.87 Det() [3/6]	330
11.20.3.88 Det() [4/6]	330
11.20.3.89 fsytrf_BC_Crout()	330
11.20.3.90 fsytrf_BC_RL()	331
11.20.3.91 fsytrf_UP_RPM_BC_RL()	331
11.20.3.92 fsytrf_LOW_RPM_BC_Crout()	331
11.20.3.93 fsytrf_UP_RPM_BC_Crout()	331
11.20.3.94 fsytrf_UP_RPM()	332
11.20.3.95 fsytrf_nonunit() [2/3]	332
11.20.3.96 fsytrf_nonunit() [3/3]	332
11.20.3.97 fsytrf_RPM()	332
11.20.3.98 getTridiagonal()	333
11.20.3.99 LUdivine_gauss() [1/2]	333
11.20.3.100 LUdivine_small() [1/2]	333
11.20.3.101 LUdivine() [2/4]	333
11.20.3.102 LUdivine() [3/4]	334
11.20.3.103 MonotonicCompress()	334
11.20.3.104 MonotonicCompressMorePivots()	334
11.20.3.105 MonotonicCompressCycles()	335

11.20.3.106 MonotonicExpand()	335
11.20.3.107 applyP_block()	335
11.20.3.108 doApplyS()	335
11.20.3.109 MatrixApplyS() [1/3]	336
11.20.3.110 MatrixApplyS() [2/3]	336
11.20.3.111 MatrixApplyS() [3/3]	336
11.20.3.112 PermApplyS()	336
11.20.3.113 doApplyT()	337
11.20.3.114 MatrixApplyT() [1/3]	337
11.20.3.115 MatrixApplyT() [2/3]	337
11.20.3.116 MatrixApplyT() [3/3]	337
11.20.3.117 PermApplyT()	338
11.20.3.118 composePermutationsLLL()	338
11.20.3.119 composePermutationsLLM()	338
11.20.3.120 composePermutationsMLM()	339
11.20.3.121 cyclic_shift_mathPerm()	339
11.20.3.122 cyclic_shift_row_col() [1/2]	339
11.20.3.123 cyclic_shift_row() [1/3]	339
11.20.3.124 cyclic_shift_row() [2/3]	340
11.20.3.125 cyclic_shift_col() [1/3]	340
11.20.3.126 cyclic_shift_col() [2/3]	340
11.20.3.127 PLUQ_basecaseV3()	340
11.20.3.128 PLUQ_basecaseV2()	340
11.20.3.129 PLUQ_basecaseCrout()	341
11.20.3.130 _PLUQ()	341
11.20.3.131 PLUQ() [4/6]	341
11.20.3.132 threads_fgemm()	341
11.20.3.133 threads_ftrsm()	342
11.20.3.134 PLUQ() [5/6]	342
11.20.3.135 fflas_const_cast() [1/3]	342
11.20.3.136 fflas_const_cast() [2/3]	342
11.20.3.137 cyclic_shift_row_col() [2/2]	342
11.20.3.138 cyclic_shift_row() [3/3]	343
11.20.3.139 cyclic_shift_col() [3/3]	343
11.20.3.140 applyP() [4/4]	343
11.20.3.141 fgetrs() [3/4]	343
11.20.3.142 fgetrs() [4/4]	344
11.20.3.143 fgesv() [3/4]	344
11.20.3.144 fgesv() [4/4]	344
11.20.3.145 ftrtri() [2/2]	344
11.20.3.146 trinv_left() [2/2]	345
11.20.3.147 ftrtrm() [2/2]	345

11.20.3.148 PLUQ() [6/6]	345
11.20.3.149 LUdivine() [4/4]	345
11.20.3.150 LUdivine_small() [2/2]	346
11.20.3.151 LUdivine_gauss() [2/2]	346
11.20.3.152 RowEchelonForm() [3/3]	346
11.20.3.153 ReducedRowEchelonForm() [3/3]	346
11.20.3.154 ColumnEchelonForm() [3/3]	347
11.20.3.155 ReducedColumnEchelonForm() [3/3]	347
11.20.3.156 Invert() [3/4]	347
11.20.3.157 Invert() [4/4]	347
11.20.3.158 Invert2() [2/2]	347
11.20.3.159 CharPoly() [6/8]	348
11.20.3.160 CharPoly() [7/8]	348
11.20.3.161 CharPoly() [8/8]	348
11.20.3.162 MinPoly() [3/4]	348
11.20.3.163 MinPoly() [4/4]	348
11.20.3.164 MatVecMinPoly() [2/2]	349
11.20.3.165 KrylovElim()	349
11.20.3.166 SpecRankProfile()	349
11.20.3.167 Rank() [3/3]	349
11.20.3.168 IsSingular() [2/2]	349
11.20.3.169 Det() [5/6]	350
11.20.3.170 Det() [6/6]	350
11.20.3.171 Solve() [3/3]	350
11.20.3.172 solveLB() [2/2]	350
11.20.3.173 solveLB2() [2/2]	351
11.20.3.174 RandomNullSpaceVector() [3/3]	351
11.20.3.175 NullSpaceBasis() [2/2]	351
11.20.3.176 RowRankProfile() [3/3]	351
11.20.3.177 ColumnRankProfile() [3/3]	352
11.20.3.178 RowRankProfileSubmatrixIndices() [2/2]	352
11.20.3.179 ColRankProfileSubmatrixIndices() [2/2]	352
11.20.3.180 RowRankProfileSubmatrix() [2/2]	352
11.20.3.181 ColRankProfileSubmatrix() [2/2]	352
11.20.3.182 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [1/2]	353
11.20.3.183 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [2/2]	353
11.20.3.184 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2]	353
11.20.3.185 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2]	353
11.20.3.186 getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()	354
11.20.3.187 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2]	354
11.20.3.188 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2]	354
11.20.3.189 getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()	354

11.20.3.190 LQUPtoInverseOfFullRankMinor()	[2/2]	355
11.20.3.191 fflas_const_cast()	[3/3]	355
11.20.3.192 failure()		355
11.20.3.193 isOdd()	[1/3]	355
11.20.3.194 isOdd()	[2/3]	355
11.20.3.195 isOdd()	[3/3]	355
11.20.3.196 NonZeroRandomMatrix()	[1/2]	356
11.20.3.197 NonZeroRandomMatrix()	[2/2]	356
11.20.3.198 RandomMatrix()	[1/2]	357
11.20.3.199 RandomMatrix()	[2/2]	357
11.20.3.200 RandomTriangularMatrix()	[1/2]	358
11.20.3.201 RandomTriangularMatrix()	[2/2]	358
11.20.3.202 RandInt()		359
11.20.3.203 RandomSymmetricMatrix()		359
11.20.3.204 RandomMatrixWithRank()	[1/2]	360
11.20.3.205 RandomMatrixWithRank()	[2/2]	360
11.20.3.206 RandomIndexSubset()		361
11.20.3.207 RandomPermutation()		361
11.20.3.208 RandomRankProfileMatrix()		361
11.20.3.209 swapval()		362
11.20.3.210 RandomSymmetricRankProfileMatrix()		362
11.20.3.211 RandomMatrixWithRankandRPM()	[1/2]	362
11.20.3.212 RandomMatrixWithRankandRPM()	[2/2]	363
11.20.3.213 RandomSymmetricMatrixWithRankandRPM()	[1/2]	363
11.20.3.214 RandomSymmetricMatrixWithRankandRPM()	[2/2]	364
11.20.3.215 RandomMatrixWithRankandRandomRPM()	[1/2]	364
11.20.3.216 RandomMatrixWithRankandRandomRPM()	[2/2]	365
11.20.3.217 RandomSymmetricMatrixWithRankandRandomRPM()	[1/2]	366
11.20.3.218 RandomSymmetricMatrixWithRankandRandomRPM()	[2/2]	366
11.20.3.219 RandomMatrixWithDet()	[1/2]	367
11.20.3.220 RandomMatrixWithDet()	[2/2]	367
11.20.3.221 maxFieldElt()		368
11.20.3.222 maxFieldElt< Givaro::ZRing< Givaro::Integer > >()		368
11.20.3.223 chooseField()		368
11.20.3.224 chooseField< Givaro::ZRing< int32_t > >()		368
11.20.3.225 chooseField< Givaro::ZRing< int64_t > >()		368
11.20.3.226 chooseField< Givaro::ZRing< float > >()		368
11.20.3.227 chooseField< Givaro::ZRing< double > >()		369
11.21 FFPACK::Protected Namespace Reference		369
11.21.1 Function Documentation		370
11.21.1.1 LUdivine_construct()	[1/2]	370
11.21.1.2 GaussJordan()		371

11.21.1.3 KellerGehrig()	371
11.21.1.4 KGFast()	372
11.21.1.5 KGFast_generalized()	372
11.21.1.6 fgemv_kgf()	372
11.21.1.7 LUKrylov()	372
11.21.1.8 Danilevski()	372
11.21.1.9 RandomKrylovPrecond()	373
11.21.1.10 ArithProg()	373
11.21.1.11 LUKrylov_KGFast()	373
11.21.1.12 MatVecMinPoly()	373
11.21.1.13 Hybrid_KGF_LUK_MinPoly()	374
11.21.1.14 updateD()	374
11.21.1.15 newD()	374
11.21.1.16 CompressRows()	374
11.21.1.17 CompressRowsQK()	375
11.21.1.18 DeCompressRows()	375
11.21.1.19 DeCompressRowsQK()	375
11.21.1.20 CompressRowsQA()	375
11.21.1.21 DeCompressRowsQA()	376
11.21.1.22 LUdivine_construct() [2/2]	376
11.22 Givaro Namespace Reference	376
11.23 MKL_CONFIG Namespace Reference	376
11.24 Reclnt Namespace Reference	376
12 Data Structure Documentation	377
12.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference	377
12.1.1 Member Typedef Documentation	377
12.1.1.1 value	377
12.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference	377
12.2.1 Member Typedef Documentation	377
12.2.1.1 value	377
12.3 ArbitraryPreclntTag Struct Reference	377
12.3.1 Detailed Description	377
12.4 AreEqual< X, Y > Class Template Reference	378
12.4.1 Field Documentation	378
12.4.1.1 value	378
12.5 AreEqual< X, X > Class Template Reference	378
12.5.1 Field Documentation	378
12.5.1.1 value	378
12.6 Argument Struct Reference	378
12.6.1 Field Documentation	378
12.6.1.1 c	378

12.6.1.2 example	378
12.6.1.3 helpString	379
12.6.1.4 type	379
12.6.1.5 data	379
12.7 associatedDelayedField< Field > Struct Template Reference	379
12.7.1 Member Typedef Documentation	379
12.7.1.1 field	379
12.7.1.2 type	379
12.8 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference	379
12.8.1 Member Typedef Documentation	379
12.8.1.1 field	379
12.8.1.2 type	380
12.9 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference	380
12.9.1 Member Typedef Documentation	380
12.9.1.1 field	380
12.9.1.2 type	380
12.10 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference	380
12.10.1 Member Typedef Documentation	380
12.10.1.1 field	380
12.10.1.2 type	380
12.11 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference	381
12.11.1 Member Typedef Documentation	381
12.11.1.1 field	381
12.11.1.2 type	381
12.12 Auto Struct Reference	381
12.13 Bini Struct Reference	381
12.14 Block Struct Reference	381
12.15 callLUdivine_small< Element > Class Template Reference	381
12.15.1 Member Function Documentation	381
12.15.1.1 operator()()	381
12.16 callLUdivine_small< double > Class Reference	382
12.16.1 Member Function Documentation	382
12.16.1.1 operator()()	382
12.17 callLUdivine_small< float > Class Reference	382
12.17.1 Member Function Documentation	382
12.17.1.1 operator()()	382
12.18 CharpolyFailed Class Reference	383
12.19 Checker_Empty< Field > Struct Template Reference	383
12.19.1 Constructor & Destructor Documentation	383
12.19.1.1 Checker_Empty()	383
12.19.2 Member Function Documentation	383
12.19.2.1 check()	383

12.20 CheckerImplem_charpoly< Field, Polynomial > Class Template Reference	384
12.20.1 Constructor & Destructor Documentation	384
12.20.1.1 CheckerImplem_charpoly() [1/2]	384
12.20.1.2 CheckerImplem_charpoly() [2/2]	384
12.20.1.3 ~CheckerImplem_charpoly()	384
12.20.2 Member Function Documentation	384
12.20.2.1 check()	384
12.21 CheckerImplem_Det< Field > Class Template Reference	384
12.21.1 Constructor & Destructor Documentation	385
12.21.1.1 CheckerImplem_Det() [1/2]	385
12.21.1.2 CheckerImplem_Det() [2/2]	385
12.21.1.3 ~CheckerImplem_Det()	385
12.21.2 Member Function Documentation	385
12.21.2.1 check()	385
12.22 CheckerImplem_fgemm< Field > Class Template Reference	385
12.22.1 Constructor & Destructor Documentation	386
12.22.1.1 CheckerImplem_fgemm() [1/2]	386
12.22.1.2 CheckerImplem_fgemm() [2/2]	386
12.22.1.3 ~CheckerImplem_fgemm()	386
12.22.2 Member Function Documentation	386
12.22.2.1 check()	386
12.23 CheckerImplem_ftdsm< Field > Class Template Reference	386
12.23.1 Constructor & Destructor Documentation	387
12.23.1.1 CheckerImplem_ftdsm() [1/2]	387
12.23.1.2 CheckerImplem_ftdsm() [2/2]	387
12.23.1.3 ~CheckerImplem_ftdsm()	387
12.23.2 Member Function Documentation	387
12.23.2.1 check()	387
12.24 CheckerImplem_invert< Field > Class Template Reference	388
12.24.1 Constructor & Destructor Documentation	388
12.24.1.1 CheckerImplem_invert() [1/2]	388
12.24.1.2 CheckerImplem_invert() [2/2]	388
12.24.1.3 ~CheckerImplem_invert()	388
12.24.2 Member Function Documentation	388
12.24.2.1 check()	388
12.25 CheckerImplem_PLUQ< Field > Class Template Reference	388
12.25.1 Constructor & Destructor Documentation	389
12.25.1.1 CheckerImplem_PLUQ() [1/2]	389
12.25.1.2 CheckerImplem_PLUQ() [2/2]	389
12.25.1.3 ~CheckerImplem_PLUQ()	389
12.25.2 Member Function Documentation	389
12.25.2.1 check()	389

12.26 Classic Struct Reference	389
12.27 Column Struct Reference	390
12.28 CompactElement< Element > Struct Template Reference	390
12.28.1 Member Typedef Documentation	390
12.28.1.1 type	390
12.29 CompactElement< double > Struct Reference	390
12.29.1 Member Typedef Documentation	390
12.29.1.1 type	390
12.30 CompactElement< float > Struct Reference	390
12.30.1 Member Typedef Documentation	390
12.30.1.1 type	390
12.31 CompactElement< int16_t > Struct Reference	390
12.31.1 Member Typedef Documentation	391
12.31.1.1 type	391
12.32 CompactElement< int32_t > Struct Reference	391
12.32.1 Member Typedef Documentation	391
12.32.1.1 type	391
12.33 CompactElement< int64_t > Struct Reference	391
12.33.1 Member Typedef Documentation	391
12.33.1.1 type	391
12.34 compatible_data_type< Field > Struct Template Reference	391
12.34.1 Field Documentation	391
12.34.1.1 value	391
12.35 compatible_data_type< Givaro::ZRing< double > > Struct Reference	391
12.35.1 Field Documentation	392
12.35.1.1 value	392
12.36 compatible_data_type< Givaro::ZRing< float > > Struct Reference	392
12.36.1 Field Documentation	392
12.36.1.1 value	392
12.37 Compose< H1, H2 > Struct Template Reference	392
12.37.1 Constructor & Destructor Documentation	392
12.37.1.1 Compose() [1/5]	392
12.37.1.2 Compose() [2/5]	392
12.37.1.3 Compose() [3/5]	392
12.37.1.4 Compose() [4/5]	393
12.37.1.5 Compose() [5/5]	393
12.37.2 Member Function Documentation	393
12.37.2.1 first_component()	393
12.37.2.2 second_component()	393
12.37.3 Friends And Related Symbol Documentation	393
12.37.3.1 operator<<	393
12.38 Const_int_t< n > Class Template Reference	393

12.39 Const_uint_t< n > Class Template Reference	393
12.40 Simd128_impl< true, true, false, 2 >::Converter Union Reference	393
12.40.1 Field Documentation	394
12.40.1.1 v	394
12.40.1.2 t	394
12.41 Simd128_impl< true, true, false, 4 >::Converter Union Reference	394
12.41.1 Field Documentation	394
12.41.1.1 v	394
12.41.1.2 t	394
12.42 Simd128_impl< true, true, false, 8 >::Converter Union Reference	394
12.42.1 Field Documentation	394
12.42.1.1 v	394
12.42.1.2 t	394
12.43 Simd128_impl< true, true, true, 2 >::Converter Union Reference	394
12.43.1 Field Documentation	395
12.43.1.1 v	395
12.43.1.2 t	395
12.44 Simd128_impl< true, true, true, 4 >::Converter Union Reference	395
12.44.1 Field Documentation	395
12.44.1.1 v	395
12.44.1.2 t	395
12.45 Simd128_impl< true, true, true, 8 >::Converter Union Reference	395
12.45.1 Field Documentation	395
12.45.1.1 v	395
12.45.1.2 t	395
12.46 Simd256_impl< true, true, false, 2 >::Converter Union Reference	395
12.46.1 Field Documentation	396
12.46.1.1 v	396
12.46.1.2 t	396
12.47 Simd256_impl< true, true, false, 4 >::Converter Union Reference	396
12.47.1 Field Documentation	396
12.47.1.1 v	396
12.47.1.2 t	396
12.48 Simd256_impl< true, true, false, 8 >::Converter Union Reference	396
12.48.1 Field Documentation	396
12.48.1.1 v	396
12.48.1.2 t	396
12.49 Simd256_impl< true, true, true, 2 >::Converter Union Reference	396
12.49.1 Field Documentation	397
12.49.1.1 v	397
12.49.1.2 t	397
12.50 Simd256_impl< true, true, true, 4 >::Converter Union Reference	397

12.50.1 Field Documentation	397
12.50.1.1 v	397
12.50.1.2 t	397
12.51 Simd256_impl< true, true, true, 8 >::Converter Union Reference	397
12.51.1 Field Documentation	397
12.51.1.1 v	397
12.51.1.2 t	397
12.52 Simd512_impl< true, true, false, 8 >::Converter Union Reference	397
12.52.1 Field Documentation	398
12.52.1.1 v	398
12.52.1.2 t	398
12.53 Simd512_impl< true, true, true, 8 >::Converter Union Reference	398
12.53.1 Field Documentation	398
12.53.1.1 v	398
12.53.1.2 t	398
12.54 ConvertTo< T > Struct Template Reference	398
12.54.1 Detailed Description	398
12.55 Coo< ValT, IdxT > Struct Template Reference	398
12.55.1 Member Typedef Documentation	399
12.55.1.1 Self	399
12.55.2 Constructor & Destructor Documentation	399
12.55.2.1 Coo() [1/4]	399
12.55.2.2 Coo() [2/4]	399
12.55.2.3 Coo() [3/4]	399
12.55.2.4 Coo() [4/4]	399
12.55.3 Member Function Documentation	399
12.55.3.1 operator=() [1/2]	399
12.55.3.2 operator=() [2/2]	400
12.55.4 Field Documentation	400
12.55.4.1 val	400
12.55.4.2 row	400
12.55.4.3 col	400
12.56 Coo< Field > Struct Template Reference	400
12.56.1 Constructor & Destructor Documentation	400
12.56.1.1 Coo() [1/4]	400
12.56.1.2 Coo() [2/4]	400
12.56.1.3 Coo() [3/4]	401
12.56.1.4 Coo() [4/4]	401
12.56.2 Member Function Documentation	401
12.56.2.1 operator=() [1/2]	401
12.56.2.2 operator=() [2/2]	401
12.56.3 Field Documentation	401

12.56.3.1 val	401
12.56.3.2 col	401
12.56.3.3 row	401
12.56.3.4 deleted	401
12.57 Coo< ValT, IdxT > Struct Template Reference	401
12.57.1 Member Typedef Documentation	402
12.57.1.1 Self	402
12.57.2 Constructor & Destructor Documentation	402
12.57.2.1 Coo() [1/4]	402
12.57.2.2 Coo() [2/4]	402
12.57.2.3 Coo() [3/4]	402
12.57.2.4 Coo() [4/4]	402
12.57.3 Member Function Documentation	402
12.57.3.1 operator=() [1/2]	402
12.57.3.2 operator=() [2/2]	403
12.57.4 Field Documentation	403
12.57.4.1 val	403
12.57.4.2 row	403
12.57.4.3 col	403
12.58 CooMat< Field > Struct Template Reference	403
12.58.1 Field Documentation	403
12.58.1.1 _coo16	403
12.58.1.2 _coo32	403
12.58.1.3 _coo64	403
12.58.1.4 _coo16_zo	403
12.58.1.5 _coo32_zo	404
12.58.1.6 _coo64_zo	404
12.59 CsrMat< Field > Struct Template Reference	404
12.59.1 Field Documentation	404
12.59.1.1 _csr16	404
12.59.1.2 _csr32	404
12.59.1.3 _csr64	404
12.59.1.4 _csr16_zo	404
12.59.1.5 _csr32_zo	404
12.59.1.6 _csr64_zo	404
12.60 DefaultBoundedTag Struct Reference	405
12.60.1 Detailed Description	405
12.61 DefaultTag Struct Reference	405
12.61.1 Detailed Description	405
12.62 DelayedTag Struct Reference	405
12.62.1 Detailed Description	405
12.63 ElementTraits< Element > Struct Template Reference	405

12.63.1 Detailed Description	405
12.63.2 Member Typedef Documentation	405
12.63.2.1 value	405
12.64 ElementTraits< double > Struct Reference	406
12.64.1 Member Typedef Documentation	406
12.64.1.1 value	406
12.65 ElementTraits< FFPACK::rns_double_elt > Struct Reference	406
12.65.1 Member Typedef Documentation	406
12.65.1.1 value	406
12.66 ElementTraits< float > Struct Reference	406
12.66.1 Member Typedef Documentation	406
12.66.1.1 value	406
12.67 ElementTraits< Givaro::Integer > Struct Reference	406
12.67.1 Member Typedef Documentation	407
12.67.1.1 value	407
12.68 ElementTraits< int16_t > Struct Reference	407
12.68.1 Member Typedef Documentation	407
12.68.1.1 value	407
12.69 ElementTraits< int32_t > Struct Reference	407
12.69.1 Member Typedef Documentation	407
12.69.1.1 value	407
12.70 ElementTraits< int64_t > Struct Reference	407
12.70.1 Member Typedef Documentation	408
12.70.1.1 value	408
12.71 ElementTraits< int8_t > Struct Reference	408
12.71.1 Member Typedef Documentation	408
12.71.1.1 value	408
12.72 ElementTraits< RecInt::rint< K > > Struct Template Reference	408
12.72.1 Member Typedef Documentation	408
12.72.1.1 value	408
12.73 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference	408
12.73.1 Member Typedef Documentation	408
12.73.1.1 value	408
12.74 ElementTraits< RecInt::ruint< K > > Struct Template Reference	409
12.74.1 Member Typedef Documentation	409
12.74.1.1 value	409
12.75 ElementTraits< uint16_t > Struct Reference	409
12.75.1 Member Typedef Documentation	409
12.75.1.1 value	409
12.76 ElementTraits< uint32_t > Struct Reference	409
12.76.1 Member Typedef Documentation	409
12.76.1.1 value	409

12.77 ElementTraits< uint64_t > Struct Reference	409
12.77.1 Member Typedef Documentation	410
12.77.1.1 value	410
12.78 ElementTraits< uint8_t > Struct Reference	410
12.78.1 Member Typedef Documentation	410
12.78.1.1 value	410
12.79 EllMat< Field > Struct Template Reference	410
12.79.1 Field Documentation	410
12.79.1.1 _ell16	410
12.79.1.2 _ell32	410
12.79.1.3 _ell64	410
12.79.1.4 _ell16_zo	411
12.79.1.5 _ell32_zo	411
12.79.1.6 _ell64_zo	411
12.80 Failure Class Reference	411
12.80.1 Detailed Description	411
12.80.2 Constructor & Destructor Documentation	411
12.80.2.1 Failure()	411
12.80.3 Member Function Documentation	411
12.80.3.1 operator>() [1/2]	411
12.80.3.2 operator>() [2/2]	412
12.80.3.3 setErrorStream()	412
12.80.3.4 print()	412
12.80.4 Field Documentation	412
12.80.4.1 _errorStream	412
12.81 FailureCharpolyCheck Class Reference	412
12.82 FailureDetCheck Class Reference	413
12.83 FailureFgemmCheck Class Reference	413
12.84 FailureInvertCheck Class Reference	413
12.85 FailurePLUQCheck Class Reference	413
12.86 FailureTrsmCheck Class Reference	413
12.87 FieldSimd< _Field > Class Template Reference	413
12.87.1 Member Typedef Documentation	414
12.87.1.1 Field	414
12.87.1.2 Element	414
12.87.1.3 simd	414
12.87.1.4 vect_t	414
12.87.1.5 scalar_t	415
12.87.2 Constructor & Destructor Documentation	415
12.87.2.1 FieldSimd() [1/3]	415
12.87.2.2 FieldSimd() [2/3]	415
12.87.2.3 FieldSimd() [3/3]	415

12.87.3 Member Function Documentation	415
12.87.3.1 operator=() [1/2]	415
12.87.3.2 operator=() [2/2]	415
12.87.3.3 init() [1/2]	415
12.87.3.4 init() [2/2]	415
12.87.3.5 add() [1/2]	415
12.87.3.6 add() [2/2]	416
12.87.3.7 addin()	416
12.87.3.8 add_r() [1/2]	416
12.87.3.9 add_r() [2/2]	416
12.87.3.10 addin_r()	416
12.87.3.11 sub() [1/2]	416
12.87.3.12 sub() [2/2]	416
12.87.3.13 subin()	416
12.87.3.14 sub_r() [1/2]	417
12.87.3.15 sub_r() [2/2]	417
12.87.3.16 subin_r()	417
12.87.3.17 zero() [1/2]	417
12.87.3.18 zero() [2/2]	417
12.87.3.19 mod()	417
12.87.3.20 mul() [1/2]	417
12.87.3.21 mul() [2/2]	417
12.87.3.22 mulin()	417
12.87.3.23 mul_r() [1/2]	418
12.87.3.24 mul_r() [2/2]	418
12.87.3.25 axpy() [1/2]	418
12.87.3.26 axpy() [2/2]	418
12.87.3.27 axpyin()	418
12.87.3.28 axpy_r() [1/2]	418
12.87.3.29 axpy_r() [2/2]	418
12.87.3.30 axpyin_r()	419
12.87.3.31 maxpy() [1/2]	419
12.87.3.32 maxpy() [2/2]	419
12.87.3.33 maxpyin()	419
12.87.4 Field Documentation	419
12.87.4.1 vect_size	419
12.87.4.2 alignment	419
12.88 FieldTraits< Field > Struct Template Reference	419
12.88.1 Detailed Description	420
12.88.2 Member Typedef Documentation	420
12.88.2.1 category	420
12.88.3 Field Documentation	420

12.88.3.1 balanced	420
12.89 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference	420
12.89.1 Member Typedef Documentation	420
12.89.1.1 category	420
12.89.2 Field Documentation	420
12.89.2.1 balanced	420
12.90 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference	420
12.90.1 Member Typedef Documentation	421
12.90.1.1 category	421
12.90.2 Field Documentation	421
12.90.2.1 balanced	421
12.91 FieldTraits< Givaro::Modular< Element > > Struct Template Reference	421
12.91.1 Member Typedef Documentation	421
12.91.1.1 category	421
12.91.2 Field Documentation	421
12.91.2.1 balanced	421
12.92 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference	421
12.92.1 Member Typedef Documentation	422
12.92.1.1 category	422
12.92.2 Field Documentation	422
12.92.2.1 balanced	422
12.93 FieldTraits< Givaro::ZRing< double > > Struct Reference	422
12.93.1 Member Typedef Documentation	422
12.93.1.1 category	422
12.93.2 Field Documentation	422
12.93.2.1 balanced	422
12.94 FieldTraits< Givaro::ZRing< float > > Struct Reference	422
12.94.1 Member Typedef Documentation	423
12.94.1.1 category	423
12.94.2 Field Documentation	423
12.94.2.1 balanced	423
12.95 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference	423
12.95.1 Member Typedef Documentation	423
12.95.1.1 category	423
12.95.2 Field Documentation	423
12.95.2.1 balanced	423
12.96 FieldTraits< Givaro::ZRing< int16_t > > Struct Reference	423
12.96.1 Member Typedef Documentation	424
12.96.1.1 category	424
12.96.2 Field Documentation	424
12.96.2.1 balanced	424
12.97 FieldTraits< Givaro::ZRing< int32_t > > Struct Reference	424

12.97.1 Member Typedef Documentation	424
12.97.1.1 category	424
12.97.2 Field Documentation	424
12.97.2.1 balanced	424
12.98 FieldTraits< Givaro::ZRing< int64_t > > Struct Reference	424
12.98.1 Member Typedef Documentation	424
12.98.1.1 category	424
12.98.2 Field Documentation	425
12.98.2.1 balanced	425
12.99 FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > Struct Template Reference	425
12.99.1 Member Typedef Documentation	425
12.99.1.1 category	425
12.99.2 Field Documentation	425
12.99.2.1 balanced	425
12.100 FieldTraits< Givaro::ZRing< uint16_t > > Struct Reference	425
12.100.1 Member Typedef Documentation	425
12.100.1.1 category	425
12.100.2 Field Documentation	426
12.100.2.1 balanced	426
12.101 FieldTraits< Givaro::ZRing< uint32_t > > Struct Reference	426
12.101.1 Member Typedef Documentation	426
12.101.1.1 category	426
12.101.2 Field Documentation	426
12.101.2.1 balanced	426
12.102 FieldTraits< Givaro::ZRing< uint64_t > > Struct Reference	426
12.102.1 Member Typedef Documentation	426
12.102.1.1 category	426
12.102.2 Field Documentation	426
12.102.2.1 balanced	426
12.103 Fixed Struct Reference	427
12.104 FixedPreIntTag Struct Reference	427
12.104.1 Detailed Description	427
12.105 ForStrategy1D< blocksize_t, Cut, Param > Struct Template Reference	427
12.105.1 Constructor & Destructor Documentation	427
12.105.1.1 ForStrategy1D() [1/2]	427
12.105.1.2 ForStrategy1D() [2/2]	428
12.105.2 Member Function Documentation	428
12.105.2.1 build()	428
12.105.2.2 initialize()	428
12.105.2.3 isTerminated()	428
12.105.2.4 begin()	428
12.105.2.5 end()	428

12.105.2.6 numblocks()	428
12.105.2.7 blockindex()	428
12.105.2.8 operator++()	428
12.105.3 Field Documentation	429
12.105.3.1 ibeg	429
12.105.3.2 iend	429
12.105.3.3 current	429
12.105.3.4 firstBlockSize	429
12.105.3.5 lastBlockSize	429
12.105.3.6 changeBS	429
12.105.3.7 numBlock	429
12.106 ForStrategy2D< blockSize_t, Cut, Param > Struct Template Reference	429
12.106.1 Constructor & Destructor Documentation	430
12.106.1.1 ForStrategy2D()	430
12.106.2 Member Function Documentation	430
12.106.2.1 initialize()	430
12.106.2.2 isTerminated()	430
12.106.2.3 ibegin()	430
12.106.2.4 jbegin()	431
12.106.2.5 iend()	431
12.106.2.6 jend()	431
12.106.2.7 operator++()	431
12.106.2.8 rownumblocks()	431
12.106.2.9 colnumblocks()	431
12.106.2.10 blockindex()	431
12.106.2.11 rowblockindex()	431
12.106.2.12 colblockindex()	431
12.106.3 Friends And Related Symbol Documentation	431
12.106.3.1 operator<<	431
12.106.4 Field Documentation	432
12.106.4.1 _ibeg	432
12.106.4.2 _iend	432
12.106.4.3 _jbeg	432
12.106.4.4 _jend	432
12.106.4.5 rowBlockSize	432
12.106.4.6 colBlockSize	432
12.106.4.7 current	432
12.106.4.8 lastRBS	432
12.106.4.9 lastCBS	432
12.106.4.10 changeRBS	433
12.106.4.11 changeCBS	433
12.106.4.12 numRowBlock	433

12.106.4.13 numColBlock	433
12.106.4.14 BLOCKS	433
12.107 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference	433
12.108 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference	433
12.109 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference	433
12.110 ftrmmLeftLowerTransUnit< Element > Class Template Reference	433
12.111 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference	434
12.112 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference	434
12.113 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference	434
12.114 ftrmmLeftUpperTransUnit< Element > Class Template Reference	434
12.115 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference	434
12.116 ftrmmRightLowerNoTransUnit< Element > Class Template Reference	434
12.117 ftrmmRightLowerTransNonUnit< Element > Class Template Reference	434
12.118 ftrmmRightLowerTransUnit< Element > Class Template Reference	434
12.119 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference	435
12.120 ftrmmRightUpperNoTransUnit< Element > Class Template Reference	435
12.121 ftrmmRightUpperTransNonUnit< Element > Class Template Reference	435
12.122 ftrmmRightUpperTransUnit< Element > Class Template Reference	435
12.123 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference	435
12.124 ftrsmLeftLowerNoTransUnit< Element > Class Template Reference	435
12.125 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference	435
12.126 ftrsmLeftLowerTransUnit< Element > Class Template Reference	435
12.127 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference	436
12.127.1 Detailed Description	436
12.128 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference	436
12.129 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference	436
12.130 ftrsmLeftUpperTransUnit< Element > Class Template Reference	436
12.131 ftrsmRightLowerNoTransNonUnit< Element > Class Template Reference	436
12.132 ftrsmRightLowerNoTransUnit< Element > Class Template Reference	436
12.133 ftrsmRightLowerTransNonUnit< Element > Class Template Reference	437
12.134 ftrsmRightLowerTransUnit< Element > Class Template Reference	437
12.135 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference	437
12.136 ftrsmRightUpperNoTransUnit< Element > Class Template Reference	437
12.137 ftrsmRightUpperTransNonUnit< Element > Class Template Reference	437
12.138 ftrsmRightUpperTransUnit< Element > Class Template Reference	437
12.139 GenericTag Struct Reference	437
12.139.1 Detailed Description	437
12.140 GenericTag Struct Reference	438
12.140.1 Detailed Description	438
12.141 Grain Struct Reference	438
12.142 has_minus_eq_impl< C > Struct Template Reference	438
12.142.1 Field Documentation	438

12.142.1.1 value	438
12.143 has_minus_impl< C > Struct Template Reference	438
12.143.1 Field Documentation	438
12.143.1.1 value	438
12.144 has_mul_eq_impl< C > Struct Template Reference	438
12.144.1 Field Documentation	439
12.144.1.1 value	439
12.145 has_mul_impl< C > Struct Template Reference	439
12.145.1 Field Documentation	439
12.145.1.1 value	439
12.146 has_operation< T > Struct Template Reference	439
12.146.1 Field Documentation	439
12.146.1.1 value	439
12.147 has_plus_eq_impl< C > Struct Template Reference	439
12.147.1 Field Documentation	440
12.147.1.1 value	440
12.148 has_plus_impl< C > Struct Template Reference	440
12.148.1 Field Documentation	440
12.148.1.1 value	440
12.149 HelperFlag Struct Reference	440
12.149.1 Field Documentation	440
12.149.1.1 none	440
12.149.1.2 coo	440
12.149.1.3 csr	440
12.149.1.4 ell	440
12.149.1.5 aut	441
12.149.1.6 pm1	441
12.150 HelperMod< Field, ElementTraits > Struct Template Reference	441
12.151 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference	441
12.151.1 Constructor & Destructor Documentation	441
12.151.1.1 HelperMod() [1/2]	441
12.151.1.2 HelperMod() [2/2]	441
12.151.2 Field Documentation	441
12.151.2.1 p	441
12.151.2.2 invp	441
12.151.2.3 min	442
12.151.2.4 max	442
12.151.2.5 pow50rem	442
12.152 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference	442
12.152.1 Constructor & Destructor Documentation	442
12.152.1.1 HelperMod() [1/2]	442
12.152.1.2 HelperMod() [2/2]	442

12.152.2 Field Documentation	442
12.152.2.1 p	442
12.153 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference	442
12.153.1 Constructor & Destructor Documentation	443
12.153.1.1 HelperMod() [1/2]	443
12.153.1.2 HelperMod() [2/2]	443
12.153.2 Field Documentation	443
12.153.2.1 p	443
12.154 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference	443
12.154.1 Constructor & Destructor Documentation	443
12.154.1.1 HelperMod() [1/2]	443
12.154.1.2 HelperMod() [2/2]	443
12.154.2 Field Documentation	444
12.154.2.1 p	444
12.154.2.2 invp	444
12.154.2.3 min	444
12.154.2.4 max	444
12.155 Hybrid Struct Reference	444
12.156 Info Struct Reference	444
12.156.1 Constructor & Destructor Documentation	444
12.156.1.1 Info() [1/4]	444
12.156.1.2 Info() [2/4]	444
12.156.1.3 Info() [3/4]	445
12.156.1.4 Info() [4/4]	445
12.156.2 Member Function Documentation	445
12.156.2.1 operator=() [1/2]	445
12.156.2.2 operator=() [2/2]	445
12.156.3 Field Documentation	445
12.156.3.1 size	445
12.156.3.2 perm	445
12.156.3.3 begin	445
12.157 Info Struct Reference	445
12.157.1 Constructor & Destructor Documentation	446
12.157.1.1 Info() [1/4]	446
12.157.1.2 Info() [2/4]	446
12.157.1.3 Info() [3/4]	446
12.157.1.4 Info() [4/4]	446
12.157.2 Member Function Documentation	446
12.157.2.1 operator=() [1/2]	446
12.157.2.2 operator=() [2/2]	446
12.157.3 Field Documentation	446
12.157.3.1 size	446

12.157.3.2 perm	446
12.157.3.3 begin	446
12.158 is_simd< T > Struct Template Reference	446
12.158.1 Member Typedef Documentation	447
12.158.1.1 type	447
12.158.2 Field Documentation	447
12.158.2.1 value	447
12.159 isSparseMatrix< Field, M > Struct Template Reference	447
12.160 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > Struct Template Reference	447
12.161 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference	447
12.162 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > Struct Template Reference	448
12.163 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > Struct Template Reference	448
12.164 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference	448
12.165 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > Struct Template Reference	449
12.166 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > Struct Template Reference	449
12.167 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference	449
12.168 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference	450
12.169 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > Struct Template Reference	450
12.170 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > Struct Template Reference	450
12.171 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference	450
12.172 isSparseMatrixMKLFormat< F, M > Struct Template Reference	451
12.173 isSparseMatrixSimdFormat< F, M > Struct Template Reference	451
12.174 isZOSparseMatrix< F, M > Struct Template Reference	451
12.175 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference	452
12.176 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference	452
12.177 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference	452
12.178 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference	452
12.179 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference	453
12.180 Iterative Struct Reference	453
12.181 LazyTag Struct Reference	453
12.181.1 Detailed Description	453
12.182 limits< T > Struct Template Reference	453
12.183 limits< char > Struct Reference	453
12.183.1 Member Typedef Documentation	454
12.183.1.1 T	454
12.183.2 Member Function Documentation	454
12.183.2.1 max()	454
12.183.2.2 min()	454
12.183.2.3 digits()	454

12.184 limits< double > Struct Reference	454
12.184.1 Member Typedef Documentation	454
12.184.1.1 T	454
12.184.2 Member Function Documentation	454
12.184.2.1 max()	454
12.184.2.2 min()	455
12.184.2.3 digits()	455
12.185 limits< float > Struct Reference	455
12.185.1 Member Typedef Documentation	455
12.185.1.1 T	455
12.185.2 Member Function Documentation	455
12.185.2.1 max()	455
12.185.2.2 min()	455
12.185.2.3 digits()	455
12.186 limits< Givaro::Integer > Struct Reference	455
12.186.1 Member Typedef Documentation	456
12.186.1.1 T	456
12.186.2 Member Function Documentation	456
12.186.2.1 max()	456
12.186.2.2 min()	456
12.187 limits< int > Struct Reference	456
12.187.1 Member Typedef Documentation	456
12.187.1.1 T	456
12.187.2 Member Function Documentation	456
12.187.2.1 max()	456
12.187.2.2 min()	456
12.187.2.3 digits()	456
12.188 limits< long > Struct Reference	456
12.188.1 Member Typedef Documentation	457
12.188.1.1 T	457
12.188.2 Member Function Documentation	457
12.188.2.1 max()	457
12.188.2.2 min()	457
12.188.2.3 digits()	457
12.189 limits< long long > Struct Reference	457
12.189.1 Member Typedef Documentation	457
12.189.1.1 T	457
12.189.2 Member Function Documentation	457
12.189.2.1 max()	457
12.189.2.2 min()	458
12.189.2.3 digits()	458
12.190 limits< Reclnt::rint< K > > Struct Template Reference	458

12.190.1 Member Typedef Documentation	458
12.190.1.1 T	458
12.190.2 Member Function Documentation	458
12.190.2.1 max()	458
12.190.2.2 min()	458
12.191 limits< RecInt::ruint< K > > Struct Template Reference	458
12.191.1 Member Typedef Documentation	459
12.191.1.1 T	459
12.191.2 Member Function Documentation	459
12.191.2.1 max()	459
12.191.2.2 min()	459
12.192 limits< short int > Struct Reference	459
12.192.1 Member Typedef Documentation	459
12.192.1.1 T	459
12.192.2 Member Function Documentation	459
12.192.2.1 max()	459
12.192.2.2 min()	459
12.192.2.3 digits()	459
12.193 limits< signed char > Struct Reference	460
12.193.1 Member Typedef Documentation	460
12.193.1.1 T	460
12.193.2 Member Function Documentation	460
12.193.2.1 max()	460
12.193.2.2 min()	460
12.193.2.3 digits()	460
12.194 limits< unsigned char > Struct Reference	460
12.194.1 Member Typedef Documentation	460
12.194.1.1 T	460
12.194.2 Member Function Documentation	461
12.194.2.1 max()	461
12.194.2.2 min()	461
12.194.2.3 digits()	461
12.195 limits< unsigned int > Struct Reference	461
12.195.1 Member Typedef Documentation	461
12.195.1.1 T	461
12.195.2 Member Function Documentation	461
12.195.2.1 max()	461
12.195.2.2 min()	461
12.195.2.3 digits()	461
12.196 limits< unsigned long > Struct Reference	461
12.196.1 Member Typedef Documentation	462
12.196.1.1 T	462

12.196.2 Member Function Documentation	462
12.196.2.1 max()	462
12.196.2.2 min()	462
12.196.2.3 digits()	462
12.197 limits< unsigned long long > Struct Reference	462
12.197.1 Member Typedef Documentation	462
12.197.1.1 T	462
12.197.2 Member Function Documentation	462
12.197.2.1 max()	462
12.197.2.2 min()	462
12.197.2.3 digits()	463
12.198 limits< unsigned short int > Struct Reference	463
12.198.1 Member Typedef Documentation	463
12.198.1.1 T	463
12.198.2 Member Function Documentation	463
12.198.2.1 max()	463
12.198.2.2 min()	463
12.198.2.3 digits()	463
12.199 MachineFloatTag Struct Reference	463
12.199.1 Detailed Description	463
12.200 MachineIntTag Struct Reference	463
12.200.1 Detailed Description	464
12.201 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference	464
12.201.1 Member Typedef Documentation	465
12.201.1.1 Self_t	465
12.201.1.2 DelayedField_t	465
12.201.1.3 DelayedField	465
12.201.1.4 DFElt	465
12.201.2 Constructor & Destructor Documentation	465
12.201.2.1 MMHelper() [1/5]	465
12.201.2.2 MMHelper() [2/5]	465
12.201.2.3 MMHelper() [3/5]	465
12.201.2.4 MMHelper() [4/5]	465
12.201.2.5 MMHelper() [5/5]	466
12.201.3 Member Function Documentation	466
12.201.3.1 initC()	466
12.201.3.2 initA()	466
12.201.3.3 initB()	466
12.201.3.4 initOut()	466
12.201.3.5 MaxDelayedDim()	466
12.201.3.6 Aunfit()	466
12.201.3.7 Bunfit()	466

12.201.3.8 setOutBounds()	466
12.201.3.9 checkA()	467
12.201.3.10 checkB()	467
12.201.3.11 checkOut()	467
12.201.4 Friends And Related Symbol Documentation	467
12.201.4.1 operator<<	467
12.201.5 Field Documentation	467
12.201.5.1 recLevel	467
12.201.5.2 FieldMin	467
12.201.5.3 FieldMax	467
12.201.5.4 Amin	468
12.201.5.5 Amax	468
12.201.5.6 Bmin	468
12.201.5.7 Bmax	468
12.201.5.8 Cmin	468
12.201.5.9 Cmax	468
12.201.5.10 Outmin	468
12.201.5.11 Outmax	468
12.201.5.12 MaxStorableValue	468
12.201.5.13 delayedField	468
12.201.5.14 parseq	468
12.202 MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference	469
12.202.1 Member Typedef Documentation	469
12.202.1.1 Self_t	469
12.202.2 Constructor & Destructor Documentation	469
12.202.2.1 MMHelper() [1/5]	469
12.202.2.2 MMHelper() [2/5]	469
12.202.2.3 MMHelper() [3/5]	469
12.202.2.4 MMHelper() [4/5]	470
12.202.2.5 MMHelper() [5/5]	470
12.202.3 Member Function Documentation	470
12.202.3.1 setNorm()	470
12.202.4 Friends And Related Symbol Documentation	470
12.202.4.1 operator<<	470
12.202.5 Field Documentation	470
12.202.5.1 normA	470
12.202.5.2 normB	470
12.202.5.3 recLevel	470
12.202.5.4 parseq	470
12.203 MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeq↔ Trait > Struct Template Reference	471
12.203.1 Member Typedef Documentation	471

12.203.1.1 Self_t	471
12.203.2 Constructor & Destructor Documentation	471
12.203.2.1 MMHelper() [1/5]	471
12.203.2.2 MMHelper() [2/5]	471
12.203.2.3 MMHelper() [3/5]	471
12.203.2.4 MMHelper() [4/5]	472
12.203.2.5 MMHelper() [5/5]	472
12.203.3 Member Function Documentation	472
12.203.3.1 setNorm()	472
12.203.4 Friends And Related Symbol Documentation	472
12.203.4.1 operator<<	472
12.203.5 Field Documentation	472
12.203.5.1 normA	472
12.203.5.2 normB	472
12.203.5.3 recLevel	472
12.203.5.4 parseq	472
12.204 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > Struct Template Reference	473
12.204.1 Member Typedef Documentation	473
12.204.1.1 Self_t	473
12.204.2 Constructor & Destructor Documentation	473
12.204.2.1 MMHelper() [1/4]	473
12.204.2.2 MMHelper() [2/4]	473
12.204.2.3 MMHelper() [3/4]	473
12.204.2.4 MMHelper() [4/4]	474
12.204.3 Friends And Related Symbol Documentation	474
12.204.3.1 operator<<	474
12.204.4 Field Documentation	474
12.204.4.1 recLevel	474
12.204.4.2 parseq	474
12.205 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Struct Template Reference	474
12.205.1 Member Typedef Documentation	475
12.205.1.1 Self_t	475
12.205.2 Constructor & Destructor Documentation	475
12.205.2.1 MMHelper() [1/5]	475
12.205.2.2 MMHelper() [2/5]	475
12.205.2.3 MMHelper() [3/5]	475
12.205.2.4 MMHelper() [4/5]	475
12.205.2.5 MMHelper() [5/5]	475
12.205.3 Member Function Documentation	475
12.205.3.1 setNorm()	475
12.205.4 Friends And Related Symbol Documentation	476

12.205.4.1 operator<<	476
12.205.5 Field Documentation	476
12.205.5.1 normA	476
12.205.5.2 normB	476
12.205.5.3 recLevel	476
12.205.5.4 parseq	476
12.206 MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference	476
12.206.1 Detailed Description	477
12.206.2 Member Typedef Documentation	477
12.206.2.1 Self_t	477
12.206.3 Constructor & Destructor Documentation	477
12.206.3.1 MMHelper() [1/4]	477
12.206.3.2 MMHelper() [2/4]	477
12.206.3.3 MMHelper() [3/4]	477
12.206.3.4 MMHelper() [4/4]	477
12.206.4 Friends And Related Symbol Documentation	477
12.206.4.1 operator<<	477
12.206.5 Field Documentation	477
12.206.5.1 recLevel	477
12.206.5.2 parseq	478
12.207 ModeTraits< Field > Struct Template Reference	478
12.207.1 Detailed Description	478
12.207.2 Member Typedef Documentation	478
12.207.2.1 value	478
12.208 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference	478
12.208.1 Member Typedef Documentation	478
12.208.1.1 value	478
12.209 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference	478
12.209.1 Member Typedef Documentation	479
12.209.1.1 value	479
12.210 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference	479
12.210.1 Member Typedef Documentation	479
12.210.1.1 value	479
12.211 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference	479
12.211.1 Member Typedef Documentation	479
12.211.1.1 value	479
12.212 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference	479
12.212.1 Member Typedef Documentation	480
12.212.1.1 value	480
12.213 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference	480
12.213.1 Member Typedef Documentation	480
12.213.1.1 value	480

12.214 ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference	480
12.214.1 Member Typedef Documentation	480
12.214.1.1 value	480
12.215 ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference	480
12.215.1 Member Typedef Documentation	481
12.215.1.1 value	481
12.216 ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference	481
12.216.1 Member Typedef Documentation	481
12.216.1.1 value	481
12.217 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference	481
12.217.1 Member Typedef Documentation	481
12.217.1.1 value	481
12.218 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference	481
12.218.1 Member Typedef Documentation	482
12.218.1.1 value	482
12.219 ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference	482
12.219.1 Member Typedef Documentation	482
12.219.1.1 value	482
12.220 ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference	482
12.220.1 Member Typedef Documentation	482
12.220.1.1 value	482
12.221 ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference	482
12.221.1 Member Typedef Documentation	483
12.221.1.1 value	483
12.222 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference	483
12.222.1 Member Typedef Documentation	483
12.222.1.1 value	483
12.223 ModeTraits< Givaro::ZRing< double > > Struct Reference	483
12.223.1 Member Typedef Documentation	483
12.223.1.1 value	483
12.224 ModeTraits< Givaro::ZRing< float > > Struct Reference	483
12.224.1 Member Typedef Documentation	483
12.224.1.1 value	483
12.225 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference	484
12.225.1 Member Typedef Documentation	484
12.225.1.1 value	484
12.226 ModularBalanced< T > Class Template Reference	484
12.227 ModularTag Struct Reference	484
12.227.1 Detailed Description	484
12.228 Montgomery< T > Class Template Reference	484
12.229 need_field_characteristic< Field > Struct Template Reference	484
12.229.1 Field Documentation	484

12.229.1.1 value	484
12.230 need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference	485
12.230.1 Field Documentation	485
12.230.1.1 value	485
12.231 need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference	485
12.231.1 Field Documentation	485
12.231.1.1 value	485
12.232 NoSimd< T > Struct Template Reference	485
12.232.1 Member Typedef Documentation	486
12.232.1.1 vect_t	486
12.232.1.2 scalar_t	486
12.232.2 Member Function Documentation	486
12.232.2.1 type_string()	486
12.232.2.2 valid()	486
12.232.2.3 compliant()	486
12.232.3 Field Documentation	486
12.232.3.1 vect_size	486
12.233 Parallel< C, P > Struct Template Reference	486
12.233.1 Member Typedef Documentation	487
12.233.1.1 Cut	487
12.233.1.2 Param	487
12.233.2 Constructor & Destructor Documentation	487
12.233.2.1 Parallel()	487
12.233.3 Member Function Documentation	487
12.233.3.1 numthreads()	487
12.233.3.2 set_numthreads()	487
12.233.4 Friends And Related Symbol Documentation	487
12.233.4.1 operator<<	487
12.234 RNSInteger< RNS >::RandIter Class Reference	487
12.234.1 Constructor & Destructor Documentation	488
12.234.1.1 RandIter()	488
12.234.2 Member Function Documentation	488
12.234.2.1 random() [1/2]	488
12.234.2.2 random() [2/2]	488
12.234.2.3 operator>() [1/2]	488
12.234.2.4 operator>() [2/2]	488
12.234.2.5 ring()	488
12.235 RNSIntegerMod< RNS >::RandIter Class Reference	489
12.235.1 Constructor & Destructor Documentation	489
12.235.1.1 RandIter()	489
12.235.2 Member Function Documentation	489
12.235.2.1 random() [1/2]	489

12.235.2.2 random() [2/2]	489
12.235.2.3 operator>() [1/2]	489
12.235.2.4 operator>() [2/2]	489
12.235.2.5 ring()	489
12.236 readMyMachineType< Field, T > Struct Template Reference	490
12.236.1 Member Typedef Documentation	490
12.236.1.1 Element	490
12.236.1.2 Element_ptr	490
12.236.2 Member Function Documentation	490
12.236.2.1 operator>()	490
12.237 readMyMachineType< Field, mpz_t > Struct Template Reference	490
12.237.1 Member Typedef Documentation	491
12.237.1.1 Element	491
12.237.1.2 Element_ptr	491
12.237.2 Member Function Documentation	491
12.237.2.1 operator>()	491
12.238 Recursive Struct Reference	491
12.239 Recursive Struct Reference	491
12.240 rint< K > Class Template Reference	491
12.241 rns_double Struct Reference	491
12.241.1 Member Typedef Documentation	492
12.241.1.1 integer	492
12.241.1.2 ModField	492
12.241.1.3 BasisElement	492
12.241.1.4 Element	493
12.241.1.5 Element_ptr	493
12.241.1.6 ConstElement_ptr	493
12.241.2 Constructor & Destructor Documentation	493
12.241.2.1 rns_double() [1/4]	493
12.241.2.2 rns_double() [2/4]	493
12.241.2.3 rns_double() [3/4]	493
12.241.2.4 rns_double() [4/4]	493
12.241.3 Member Function Documentation	493
12.241.3.1 precompute_cst()	493
12.241.3.2 init() [1/3]	493
12.241.3.3 init() [2/3]	494
12.241.3.4 init_transpose()	494
12.241.3.5 convert() [1/2]	494
12.241.3.6 convert_transpose()	494
12.241.3.7 reduce()	494
12.241.3.8 init() [3/3]	495
12.241.3.9 convert() [2/2]	495

12.241.4 Field Documentation	495
12.241.4.1 <code>_basis</code>	495
12.241.4.2 <code>_basisMax</code>	495
12.241.4.3 <code>_negbasis</code>	495
12.241.4.4 <code>_invbasis</code>	495
12.241.4.5 <code>_field_rns</code>	495
12.241.4.6 <code>_M</code>	495
12.241.4.7 <code>_Mi</code>	495
12.241.4.8 <code>_MMi</code>	495
12.241.4.9 <code>_crt_in</code>	496
12.241.4.10 <code>_crt_out</code>	496
12.241.4.11 <code>_size</code>	496
12.241.4.12 <code>_pbits</code>	496
12.241.4.13 <code>_ldm</code>	496
12.241.4.14 <code>_mi_sum</code>	496
12.242 <code>rns_double_elt</code> Struct Reference	496
12.242.1 Constructor & Destructor Documentation	497
12.242.1.1 <code>rns_double_elt()</code> [1/3]	497
12.242.1.2 <code>~rns_double_elt()</code>	497
12.242.1.3 <code>rns_double_elt()</code> [2/3]	497
12.242.1.4 <code>rns_double_elt()</code> [3/3]	497
12.242.2 Member Function Documentation	497
12.242.2.1 <code>operator&()</code> [1/2]	497
12.242.2.2 <code>operator&()</code> [2/2]	497
12.242.3 Field Documentation	497
12.242.3.1 <code>_ptr</code>	497
12.242.3.2 <code>_stride</code>	497
12.242.3.3 <code>_alloc</code>	497
12.243 <code>rns_double_elt_cstptr</code> Struct Reference	497
12.243.1 Constructor & Destructor Documentation	498
12.243.1.1 <code>rns_double_elt_cstptr()</code> [1/5]	498
12.243.1.2 <code>rns_double_elt_cstptr()</code> [2/5]	498
12.243.1.3 <code>rns_double_elt_cstptr()</code> [3/5]	498
12.243.1.4 <code>rns_double_elt_cstptr()</code> [4/5]	498
12.243.1.5 <code>rns_double_elt_cstptr()</code> [5/5]	498
12.243.2 Member Function Documentation	499
12.243.2.1 <code>operator&()</code> [1/2]	499
12.243.2.2 <code>operator*()</code>	499
12.243.2.3 <code>operator[]()</code> [1/2]	499
12.243.2.4 <code>operator[]()</code> [2/2]	499
12.243.2.5 <code>operator++()</code>	499
12.243.2.6 <code>operator--()</code>	499

12.243.2.7 operator+()	499
12.243.2.8 operator-()	499
12.243.2.9 operator+=()	499
12.243.2.10 operator-=()	499
12.243.2.11 operator=()	499
12.243.2.12 operator<()	499
12.243.2.13 operator"!=(())	500
12.243.2.14 operator&() [2/2]	500
12.243.3 Field Documentation	500
12.243.3.1 other	500
12.243.3.2 _ptr	500
12.243.3.3 _stride	500
12.243.3.4 _alloc	500
12.244 rns_double_elt_ptr Struct Reference	500
12.244.1 Constructor & Destructor Documentation	501
12.244.1.1 rns_double_elt_ptr() [1/5]	501
12.244.1.2 rns_double_elt_ptr() [2/5]	501
12.244.1.3 rns_double_elt_ptr() [3/5]	501
12.244.1.4 rns_double_elt_ptr() [4/5]	501
12.244.1.5 rns_double_elt_ptr() [5/5]	501
12.244.2 Member Function Documentation	501
12.244.2.1 operator&() [1/2]	501
12.244.2.2 operator*()	501
12.244.2.3 operator[]() [1/2]	501
12.244.2.4 operator[]() [2/2]	501
12.244.2.5 operator++()	502
12.244.2.6 operator--()	502
12.244.2.7 operator+()	502
12.244.2.8 operator-()	502
12.244.2.9 operator+=()	502
12.244.2.10 operator-=()	502
12.244.2.11 operator=()	502
12.244.2.12 operator<()	502
12.244.2.13 operator"!=(())	502
12.244.2.14 operator&() [2/2]	502
12.244.3 Field Documentation	502
12.244.3.1 other	502
12.244.3.2 _ptr	502
12.244.3.3 _stride	502
12.244.3.4 _alloc	503
12.245 rns_double_extended Struct Reference	503
12.245.1 Member Typedef Documentation	503

12.245.1.1 integer	503
12.245.1.2 ModField	504
12.245.1.3 BasisElement	504
12.245.1.4 Element	504
12.245.1.5 Element_ptr	504
12.245.1.6 ConstElement_ptr	504
12.245.2 Constructor & Destructor Documentation	504
12.245.2.1 rns_double_extended() [1/3]	504
12.245.2.2 rns_double_extended() [2/3]	504
12.245.2.3 rns_double_extended() [3/3]	504
12.245.3 Member Function Documentation	504
12.245.3.1 precompute_cst()	504
12.245.3.2 init() [1/3]	504
12.245.3.3 init() [2/3]	505
12.245.3.4 convert() [1/2]	505
12.245.3.5 init() [3/3]	505
12.245.3.6 convert() [2/2]	505
12.245.3.7 reduce()	505
12.245.4 Field Documentation	505
12.245.4.1 _basis	505
12.245.4.2 _basisMax	505
12.245.4.3 _negbasis	506
12.245.4.4 _invbasis	506
12.245.4.5 _field_rns	506
12.245.4.6 _M	506
12.245.4.7 _Mi	506
12.245.4.8 _MMi	506
12.245.4.9 _crt_in	506
12.245.4.10 _crt_out	506
12.245.4.11 _size	506
12.245.4.12 _pbits	506
12.245.4.13 _ldm	506
12.246 RNSElementTag Struct Reference	506
12.246.1 Detailed Description	506
12.247 RNSInteger< RNS > Class Template Reference	507
12.247.1 Member Typedef Documentation	507
12.247.1.1 BasisElement	507
12.247.1.2 integer	508
12.247.1.3 Element	508
12.247.1.4 Element_ptr	508
12.247.1.5 ConstElement_ptr	508
12.247.2 Constructor & Destructor Documentation	508

12.247.2.1 RNSInteger() [1/2]	508
12.247.2.2 RNSInteger() [2/2]	508
12.247.3 Member Function Documentation	508
12.247.3.1 rns()	508
12.247.3.2 size()	508
12.247.3.3 isOne()	508
12.247.3.4 isMOne()	508
12.247.3.5 isZero()	509
12.247.3.6 characteristic()	509
12.247.3.7 cardinality()	509
12.247.3.8 init() [1/2]	509
12.247.3.9 init() [2/2]	509
12.247.3.10 reduce() [1/2]	509
12.247.3.11 reduce() [2/2]	509
12.247.3.12 convert()	509
12.247.3.13 assign()	509
12.247.3.14 write() [1/2]	510
12.247.3.15 write() [2/2]	510
12.247.4 Field Documentation	510
12.247.4.1 _rns	510
12.247.4.2 one	510
12.247.4.3 mOne	510
12.247.4.4 zero	510
12.248 RNSIntegerMod< RNS > Class Template Reference	510
12.248.1 Member Typedef Documentation	511
12.248.1.1 Element	511
12.248.1.2 Element_ptr	512
12.248.1.3 ConstElement_ptr	512
12.248.1.4 BasisElement	512
12.248.1.5 ModField	512
12.248.1.6 integer	512
12.248.2 Constructor & Destructor Documentation	512
12.248.2.1 RNSIntegerMod()	512
12.248.3 Member Function Documentation	512
12.248.3.1 rns()	512
12.248.3.2 delayed()	512
12.248.3.3 size()	512
12.248.3.4 isOne()	512
12.248.3.5 isMOne()	513
12.248.3.6 isZero()	513
12.248.3.7 characteristic() [1/2]	513
12.248.3.8 characteristic() [2/2]	513

12.248.3.9 cardinality() [1/2]	513
12.248.3.10 cardinality() [2/2]	513
12.248.3.11 minElement()	513
12.248.3.12 maxElement()	513
12.248.3.13 init() [1/3]	513
12.248.3.14 init() [2/3]	513
12.248.3.15 reduce() [1/2]	514
12.248.3.16 reduce() [2/2]	514
12.248.3.17 init() [3/3]	514
12.248.3.18 convert()	514
12.248.3.19 assign()	514
12.248.3.20 add()	514
12.248.3.21 sub()	514
12.248.3.22 neg()	514
12.248.3.23 mul()	515
12.248.3.24 axpyin()	515
12.248.3.25 inv()	515
12.248.3.26 areEqual()	515
12.248.3.27 write() [1/2]	515
12.248.3.28 write() [2/2]	515
12.248.3.29 reduce_modp() [1/2]	515
12.248.3.30 write_matrix()	515
12.248.3.31 write_matrix_long()	516
12.248.3.32 reduce_modp() [2/2]	516
12.248.3.33 reduce_modp_rnsmajor()	516
12.248.4 Field Documentation	516
12.248.4.1 _p	516
12.248.4.2 _Mi_modp_rns	516
12.248.4.3 _iM_modp_rns	516
12.248.4.4 _rns	516
12.248.4.5 _F	516
12.248.4.6 _RNSdelayed	516
12.248.4.7 one	517
12.248.4.8 mOne	517
12.248.4.9 zero	517
12.249 rnsRandIter< RNS > Class Template Reference	517
12.249.1 Constructor & Destructor Documentation	517
12.249.1.1 rnsRandIter()	517
12.249.2 Member Function Documentation	517
12.249.2.1 random() [1/2]	517
12.249.2.2 operator>() [1/2]	518
12.249.2.3 operator>() [2/2]	518

12.249.2.4 random() [2/2]	518
12.249.2.5 ring()	518
12.250 Row Struct Reference	518
12.251 rint< K > Class Template Reference	518
12.252 ScalFunctions< Element, Enable > Struct Template Reference	518
12.253 ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type > Struct Template Reference	518
12.253.1 Member Function Documentation	519
12.253.1.1 zero()	519
12.253.1.2 vand()	519
12.253.1.3 vor()	519
12.253.1.4 vxor()	519
12.253.1.5 vandnot()	519
12.253.1.6 ceil()	520
12.253.1.7 floor()	520
12.253.1.8 round()	520
12.253.1.9 add()	520
12.253.1.10 addin()	520
12.253.1.11 sub()	520
12.253.1.12 subin()	520
12.253.1.13 mul()	520
12.253.1.14 mulin()	520
12.253.1.15 div()	521
12.253.1.16 fmadd()	521
12.253.1.17 fmaddin()	521
12.253.1.18 fmsub()	521
12.253.1.19 fmsubin()	521
12.253.1.20 fnmadd()	521
12.253.1.21 fnmaddin()	521
12.253.1.22 lesser()	521
12.253.1.23 lesser_eq()	522
12.253.1.24 greater()	522
12.253.1.25 greater_eq()	522
12.253.1.26 eq()	522
12.254 ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type > Struct Template Reference	522
12.254.1 Member Function Documentation	523
12.254.1.1 zero()	523
12.254.1.2 round()	523
12.254.1.3 vand()	523
12.254.1.4 vor()	523
12.254.1.5 vxor()	523
12.254.1.6 vandnot()	524

12.254.1.7 add()	524
12.254.1.8 addin()	524
12.254.1.9 sub()	524
12.254.1.10 subin()	524
12.254.1.11 mul()	524
12.254.1.12 mullo()	524
12.254.1.13 mulhi()	524
12.254.1.14 mulx()	524
12.254.1.15 fmadd()	525
12.254.1.16 fmaddin()	525
12.254.1.17 fmaddx()	525
12.254.1.18 fmaddxin()	525
12.254.1.19 fmsub()	525
12.254.1.20 fmsubin()	525
12.254.1.21 fmsubx()	525
12.254.1.22 fmsubxin()	526
12.254.1.23 fnmadd()	526
12.254.1.24 fnmaddin()	526
12.254.1.25 fnmaddx()	526
12.254.1.26 fnmaddxin()	526
12.254.1.27 sra() [1/2]	526
12.254.1.28 sra() [2/2]	526
12.254.1.29 srl()	526
12.254.1.30 sll()	527
12.254.1.31 lesser()	527
12.254.1.32 lesser_eq()	527
12.254.1.33 greater()	527
12.254.1.34 greater_eq()	527
12.254.1.35 eq()	527
12.255 Sequential Struct Reference	527
12.255.1 Constructor & Destructor Documentation	528
12.255.1.1 Sequential() [1/3]	528
12.255.1.2 Sequential() [2/3]	528
12.255.1.3 Sequential() [3/3]	528
12.255.2 Member Function Documentation	528
12.255.2.1 numthreads()	528
12.255.3 Friends And Related Symbol Documentation	528
12.255.3.1 operator<<	528
12.256 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference	528
12.257 Simd128_impl< true, false, true, 4 > Struct Reference	528
12.257.1 Member Function Documentation	529
12.257.1.1 type_string()	529

12.258 Simd128_impl< true, false, true, 8 > Struct Reference	529
12.258.1 Member Function Documentation	529
12.258.1.1 type_string()	529
12.259 Simd128_impl< true, true, false, 2 > Struct Reference	529
12.259.1 Member Typedef Documentation	531
12.259.1.1 scalar_t	531
12.259.1.2 vect_t	531
12.259.2 Member Function Documentation	531
12.259.2.1 set1() [1/2]	531
12.259.2.2 set() [1/2]	531
12.259.2.3 gather() [1/2]	531
12.259.2.4 load() [1/2]	532
12.259.2.5 loadu() [1/2]	532
12.259.2.6 store() [1/2]	532
12.259.2.7 storeu() [1/2]	532
12.259.2.8 stream() [1/2]	532
12.259.2.9 sra()	532
12.259.2.10 greater()	532
12.259.2.11 lesser()	532
12.259.2.12 greater_eq()	532
12.259.2.13 lesser_eq()	532
12.259.2.14 mulhi()	533
12.259.2.15 mulx()	533
12.259.2.16 fmaddx()	533
12.259.2.17 fmaddxin()	533
12.259.2.18 fnmaddx()	533
12.259.2.19 fnmaddxin()	533
12.259.2.20 fmsubx()	533
12.259.2.21 fmsubxin()	533
12.259.2.22 hadd_to_scal()	533
12.259.2.23 valid()	534
12.259.2.24 compliant()	534
12.259.2.25 set1() [2/2]	534
12.259.2.26 set() [2/2]	534
12.259.2.27 gather() [2/2]	534
12.259.2.28 load() [2/2]	534
12.259.2.29 loadu() [2/2]	534
12.259.2.30 store() [2/2]	534
12.259.2.31 storeu() [2/2]	534
12.259.2.32 stream() [2/2]	535
12.259.2.33 sll()	535
12.259.2.34 srl()	535

12.259.2.35 shuffle()	535
12.259.2.36 unpacklo()	535
12.259.2.37 unpackhi()	535
12.259.2.38 blend()	535
12.259.2.39 add()	535
12.259.2.40 addin()	535
12.259.2.41 sub()	535
12.259.2.42 subin()	536
12.259.2.43 mullo()	536
12.259.2.44 mul()	536
12.259.2.45 fmadd()	536
12.259.2.46 fmaddin()	536
12.259.2.47 fnmadd()	536
12.259.2.48 fnmaddin()	536
12.259.2.49 fmsub()	536
12.259.2.50 fmsubin()	536
12.259.2.51 eq()	537
12.259.2.52 round()	537
12.259.2.53 mod()	537
12.259.2.54 type_string()	537
12.259.2.55 zero()	537
12.259.2.56 sll128()	537
12.259.2.57 srl128()	537
12.259.2.58 vand()	537
12.259.2.59 vor()	537
12.259.2.60 vxor()	537
12.259.2.61 vandnot()	538
12.259.3 Field Documentation	538
12.259.3.1 vect_size	538
12.259.3.2 alignment	538
12.260 Simd128_impl< true, true, false, 4 > Struct Reference	538
12.260.1 Member Typedef Documentation	540
12.260.1.1 scalar_t	540
12.260.1.2 vect_t	540
12.260.2 Member Function Documentation	540
12.260.2.1 set1() [1/2]	540
12.260.2.2 set() [1/2]	540
12.260.2.3 gather() [1/2]	540
12.260.2.4 load() [1/2]	540
12.260.2.5 loadu() [1/2]	540
12.260.2.6 store() [1/2]	540
12.260.2.7 storeu() [1/2]	540

12.260.2.8 stream() [1/2]	541
12.260.2.9 sra()	541
12.260.2.10 greater()	541
12.260.2.11 lesser()	541
12.260.2.12 greater_eq()	541
12.260.2.13 lesser_eq()	541
12.260.2.14 mulhi()	541
12.260.2.15 mulx()	541
12.260.2.16 fmaddx()	541
12.260.2.17 fmaddxin()	541
12.260.2.18 fnmaddx()	542
12.260.2.19 fnmaddxin()	542
12.260.2.20 fmsubx()	542
12.260.2.21 fmsubxin()	542
12.260.2.22 hadd_to_scal()	542
12.260.2.23 valid()	542
12.260.2.24 compliant()	542
12.260.2.25 set1() [2/2]	542
12.260.2.26 set() [2/2]	542
12.260.2.27 gather() [2/2]	543
12.260.2.28 load() [2/2]	543
12.260.2.29 loadu() [2/2]	543
12.260.2.30 store() [2/2]	543
12.260.2.31 storeu() [2/2]	543
12.260.2.32 stream() [2/2]	543
12.260.2.33 sll()	543
12.260.2.34 srl()	543
12.260.2.35 shuffle()	543
12.260.2.36 unpacklo()	543
12.260.2.37 unpackhi()	544
12.260.2.38 blend()	544
12.260.2.39 add()	544
12.260.2.40 addin()	544
12.260.2.41 sub()	544
12.260.2.42 subin()	544
12.260.2.43 mullo()	544
12.260.2.44 mul()	544
12.260.2.45 fmadd()	544
12.260.2.46 fmaddin()	545
12.260.2.47 fnmadd()	545
12.260.2.48 fnmaddin()	545
12.260.2.49 fmsub()	545

12.260.2.50 fmsubin()	545
12.260.2.51 eq()	545
12.260.2.52 round()	545
12.260.2.53 mod()	545
12.260.2.54 type_string()	546
12.260.2.55 zero()	546
12.260.2.56 sll128()	546
12.260.2.57 srl128()	546
12.260.2.58 vand()	546
12.260.2.59 vor()	546
12.260.2.60 vxor()	546
12.260.2.61 vandnot()	546
12.260.3 Field Documentation	546
12.260.3.1 vect_size	546
12.260.3.2 alignment	546
12.261 Simd128_impl< true, true, false, 8 > Struct Reference	547
12.261.1 Member Typedef Documentation	549
12.261.1.1 scalar_t	549
12.261.1.2 vect_t	549
12.261.2 Member Function Documentation	549
12.261.2.1 set1() [1/2]	549
12.261.2.2 set() [1/2]	549
12.261.2.3 gather() [1/2]	549
12.261.2.4 load() [1/2]	549
12.261.2.5 loadu() [1/2]	549
12.261.2.6 store() [1/2]	549
12.261.2.7 storeu() [1/2]	549
12.261.2.8 stream() [1/2]	549
12.261.2.9 sra()	550
12.261.2.10 greater()	550
12.261.2.11 lesser()	550
12.261.2.12 greater_eq()	550
12.261.2.13 lesser_eq()	550
12.261.2.14 mullo()	550
12.261.2.15 mulx()	550
12.261.2.16 fmaddx()	550
12.261.2.17 fmaddxin()	550
12.261.2.18 fnmaddx()	551
12.261.2.19 fnmaddxin()	551
12.261.2.20 fmsubx()	551
12.261.2.21 fmsubxin() [1/2]	551
12.261.2.22 hadd_to_scal()	551

12.261.2.23 valid()	551
12.261.2.24 compliant()	551
12.261.2.25 set1() [2/2]	551
12.261.2.26 set() [2/2]	551
12.261.2.27 gather() [2/2]	552
12.261.2.28 get()	552
12.261.2.29 load() [2/2]	552
12.261.2.30 loadu() [2/2]	552
12.261.2.31 store() [2/2]	552
12.261.2.32 storeu() [2/2]	552
12.261.2.33 stream() [2/2]	552
12.261.2.34 sll()	552
12.261.2.35 srl()	552
12.261.2.36 shuffle()	552
12.261.2.37 unpacklo()	553
12.261.2.38 unpackhi()	553
12.261.2.39 blend()	553
12.261.2.40 add()	553
12.261.2.41 addin()	553
12.261.2.42 sub()	553
12.261.2.43 subin()	553
12.261.2.44 mul()	553
12.261.2.45 fmadd()	553
12.261.2.46 fmaddin()	554
12.261.2.47 fnmadd()	554
12.261.2.48 fnmaddin()	554
12.261.2.49 fmsub()	554
12.261.2.50 fmsubin()	554
12.261.2.51 fmsubxin() [2/2]	554
12.261.2.52 eq()	554
12.261.2.53 round()	554
12.261.2.54 mask_high()	554
12.261.2.55 mulhi_fast()	555
12.261.2.56 mod()	555
12.261.2.57 signbits()	555
12.261.2.58 type_string()	555
12.261.2.59 zero()	555
12.261.2.60 sll128()	555
12.261.2.61 srl128()	555
12.261.2.62 vand()	555
12.261.2.63 vor()	555
12.261.2.64 vxor()	555

12.261.2.65 vandnot()	556
12.261.3 Field Documentation	556
12.261.3.1 vect_size	556
12.261.3.2 alignment	556
12.262 Simd128_impl< true, true, true, 2 > Struct Reference	556
12.262.1 Member Typedef Documentation	558
12.262.1.1 vect_t	558
12.262.1.2 scalar_t	558
12.262.2 Member Function Documentation	558
12.262.2.1 valid()	558
12.262.2.2 compliant()	558
12.262.2.3 set1()	558
12.262.2.4 set()	558
12.262.2.5 gather()	558
12.262.2.6 load()	558
12.262.2.7 loadu()	558
12.262.2.8 store()	559
12.262.2.9 storeu()	559
12.262.2.10 stream()	559
12.262.2.11 sll()	559
12.262.2.12 srl()	559
12.262.2.13 sra()	559
12.262.2.14 shuffle()	559
12.262.2.15 unpacklo()	559
12.262.2.16 unpackhi()	559
12.262.2.17 blend()	559
12.262.2.18 add()	560
12.262.2.19 addin()	560
12.262.2.20 sub()	560
12.262.2.21 subin()	560
12.262.2.22 mullo()	560
12.262.2.23 mul()	560
12.262.2.24 mulhi()	560
12.262.2.25 mulx()	560
12.262.2.26 fmadd()	560
12.262.2.27 fmaddin()	560
12.262.2.28 fmaddx()	561
12.262.2.29 fmaddxin()	561
12.262.2.30 fnmadd()	561
12.262.2.31 fnmaddin()	561
12.262.2.32 fnmaddx()	561
12.262.2.33 fnmaddxin()	561

12.262.2.34 fmsub()	561
12.262.2.35 fmsubin()	561
12.262.2.36 fmsubx()	561
12.262.2.37 fmsubxin()	562
12.262.2.38 eq()	562
12.262.2.39 greater()	562
12.262.2.40 lesser()	562
12.262.2.41 greater_eq()	562
12.262.2.42 lesser_eq()	562
12.262.2.43 hadd_to_scal()	562
12.262.2.44 round()	562
12.262.2.45 mod()	562
12.262.2.46 type_string()	563
12.262.2.47 zero()	563
12.262.2.48 sll128()	563
12.262.2.49 srl128()	563
12.262.2.50 vand()	563
12.262.2.51 vor()	563
12.262.2.52 vxor()	563
12.262.2.53 vandnot()	563
12.262.3 Field Documentation	563
12.262.3.1 vect_size	563
12.262.3.2 alignment	563
12.263 Simd128_impl< true, true, true, 4 > Struct Reference	564
12.263.1 Member Typedef Documentation	565
12.263.1.1 vect_t	565
12.263.1.2 scalar_t	565
12.263.2 Member Function Documentation	565
12.263.2.1 valid()	565
12.263.2.2 compliant()	566
12.263.2.3 set1()	566
12.263.2.4 set()	566
12.263.2.5 gather()	566
12.263.2.6 load()	566
12.263.2.7 loadu()	566
12.263.2.8 store()	566
12.263.2.9 storeu()	566
12.263.2.10 stream()	566
12.263.2.11 sll()	566
12.263.2.12 srl()	567
12.263.2.13 sra()	567
12.263.2.14 shuffle()	567

12.263.2.15 unpacklo()	567
12.263.2.16 unpackhi()	567
12.263.2.17 blend()	567
12.263.2.18 add()	567
12.263.2.19 addin()	567
12.263.2.20 sub()	567
12.263.2.21 subin()	567
12.263.2.22 mullo()	568
12.263.2.23 mul()	568
12.263.2.24 mulhi()	568
12.263.2.25 mulx()	568
12.263.2.26 fmadd()	568
12.263.2.27 fmaddin()	568
12.263.2.28 fmaddx()	568
12.263.2.29 fmaddxin()	568
12.263.2.30 fnmadd()	568
12.263.2.31 fnmaddin()	569
12.263.2.32 fnmaddx()	569
12.263.2.33 fnmaddxin()	569
12.263.2.34 fmsub()	569
12.263.2.35 fmsubin()	569
12.263.2.36 fmsubx()	569
12.263.2.37 fmsubxin()	569
12.263.2.38 eq()	569
12.263.2.39 greater()	569
12.263.2.40 lesser()	570
12.263.2.41 greater_eq()	570
12.263.2.42 lesser_eq()	570
12.263.2.43 hadd_to_scal()	570
12.263.2.44 round()	570
12.263.2.45 mod()	570
12.263.2.46 type_string()	570
12.263.2.47 zero()	570
12.263.2.48 sll128()	570
12.263.2.49 srl128()	570
12.263.2.50 vand()	571
12.263.2.51 vor()	571
12.263.2.52 vxor()	571
12.263.2.53 vandnot()	571
12.263.3 Field Documentation	571
12.263.3.1 vect_size	571
12.263.3.2 alignment	571

12.264 Simd128_impl< true, true, true, 8 > Struct Reference	571
12.264.1 Member Typedef Documentation	573
12.264.1.1 vect_t	573
12.264.1.2 scalar_t	573
12.264.2 Member Function Documentation	573
12.264.2.1 valid()	573
12.264.2.2 compliant()	573
12.264.2.3 set1()	573
12.264.2.4 set()	573
12.264.2.5 gather()	574
12.264.2.6 get()	574
12.264.2.7 load()	574
12.264.2.8 loadu()	574
12.264.2.9 store()	574
12.264.2.10 storeu()	574
12.264.2.11 stream()	574
12.264.2.12 sll()	574
12.264.2.13 srl()	574
12.264.2.14 sra()	574
12.264.2.15 shuffle()	575
12.264.2.16 unpacklo()	575
12.264.2.17 unpackhi()	575
12.264.2.18 blend()	575
12.264.2.19 add()	575
12.264.2.20 addin()	575
12.264.2.21 sub()	575
12.264.2.22 subin()	575
12.264.2.23 mullo()	575
12.264.2.24 mul()	575
12.264.2.25 mulx()	576
12.264.2.26 fmadd()	576
12.264.2.27 fmaddin()	576
12.264.2.28 fmaddx()	576
12.264.2.29 fmaddxin()	576
12.264.2.30 fnmadd()	576
12.264.2.31 fnmaddin()	576
12.264.2.32 fnmaddx()	576
12.264.2.33 fnmaddxin()	576
12.264.2.34 fmsub()	577
12.264.2.35 fmsubin()	577
12.264.2.36 fmsubx()	577
12.264.2.37 fmsubxin()	577

12.264.2.38 eq()	577
12.264.2.39 greater()	577
12.264.2.40 lesser()	577
12.264.2.41 greater_eq()	577
12.264.2.42 lesser_eq()	577
12.264.2.43 hadd_to_scal()	578
12.264.2.44 round()	578
12.264.2.45 mask_high()	578
12.264.2.46 mulhi_fast()	578
12.264.2.47 mod()	578
12.264.2.48 signbits()	578
12.264.2.49 type_string()	578
12.264.2.50 zero()	578
12.264.2.51 sll128()	578
12.264.2.52 srl128()	578
12.264.2.53 vand()	579
12.264.2.54 vor()	579
12.264.2.55 vxor()	579
12.264.2.56 vandnot()	579
12.264.3 Field Documentation	579
12.264.3.1 vect_size	579
12.264.3.2 alignment	579
12.265 Simd128fp_base Struct Reference	579
12.265.1 Member Function Documentation	579
12.265.1.1 type_string()	579
12.266 Simd128i_base Struct Reference	580
12.266.1 Member Typedef Documentation	580
12.266.1.1 vect_t	580
12.266.2 Member Function Documentation	580
12.266.2.1 type_string()	580
12.266.2.2 zero()	580
12.266.2.3 sll128()	580
12.266.2.4 srl128()	580
12.266.2.5 vand()	581
12.266.2.6 vor()	581
12.266.2.7 vxor()	581
12.266.2.8 vandnot()	581
12.267 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference	581
12.268 Simd256_impl< true, false, true, 4 > Struct Reference	581
12.269 Simd256_impl< true, false, true, 8 > Struct Reference	581
12.269.1 Member Typedef Documentation	583
12.269.1.1 vect_t	583

12.269.1.2 scalar_t	583
12.269.2 Member Function Documentation	583
12.269.2.1 valid()	583
12.269.2.2 compliant()	583
12.269.2.3 zero()	583
12.269.2.4 set1()	583
12.269.2.5 set()	583
12.269.2.6 gather()	583
12.269.2.7 load()	583
12.269.2.8 loadu()	584
12.269.2.9 store()	584
12.269.2.10 storeu()	584
12.269.2.11 stream()	584
12.269.2.12 unpacklo_twice()	584
12.269.2.13 unpackhi_twice()	584
12.269.2.14 blend()	584
12.269.2.15 blendv()	584
12.269.2.16 add()	584
12.269.2.17 addin()	584
12.269.2.18 sub()	585
12.269.2.19 subin()	585
12.269.2.20 mul()	585
12.269.2.21 mulin()	585
12.269.2.22 div()	585
12.269.2.23 fmadd()	585
12.269.2.24 fmaddin()	585
12.269.2.25 fnmadd()	585
12.269.2.26 fnmaddin()	585
12.269.2.27 fmsub()	586
12.269.2.28 fmsubin()	586
12.269.2.29 eq()	586
12.269.2.30 lesser()	586
12.269.2.31 lesser_eq()	586
12.269.2.32 greater()	586
12.269.2.33 greater_eq()	586
12.269.2.34 vand()	586
12.269.2.35 vor()	586
12.269.2.36 vxor()	586
12.269.2.37 vandnot()	587
12.269.2.38 floor()	587
12.269.2.39 ceil()	587
12.269.2.40 round()	587

12.269.2.41 hadd()	587
12.269.2.42 hadd_to_scal()	587
12.269.2.43 mod()	587
12.269.3 Field Documentation	587
12.269.3.1 vect_size	587
12.269.3.2 alignment	587
12.270 Simd256_impl< true, true, false, 2 > Struct Reference	588
12.270.1 Member Typedef Documentation	589
12.270.1.1 scalar_t	589
12.270.1.2 simdHalf	590
12.270.1.3 vect_t	590
12.270.1.4 half_t	590
12.270.2 Member Function Documentation	590
12.270.2.1 set1() [1/2]	590
12.270.2.2 set() [1/2]	590
12.270.2.3 gather() [1/2]	590
12.270.2.4 load() [1/2]	590
12.270.2.5 loadu() [1/2]	590
12.270.2.6 store() [1/2]	590
12.270.2.7 storeu() [1/2]	591
12.270.2.8 stream() [1/2]	591
12.270.2.9 sra()	591
12.270.2.10 greater()	591
12.270.2.11 lesser()	591
12.270.2.12 greater_eq()	591
12.270.2.13 lesser_eq()	591
12.270.2.14 mulhi()	591
12.270.2.15 mulx()	591
12.270.2.16 fmaddx()	591
12.270.2.17 fmaddxin()	592
12.270.2.18 fnmaddx()	592
12.270.2.19 fnmaddxin()	592
12.270.2.20 fmsubx()	592
12.270.2.21 fmsubxin()	592
12.270.2.22 hadd_to_scal()	592
12.270.2.23 valid()	592
12.270.2.24 compliant()	592
12.270.2.25 set1() [2/2]	592
12.270.2.26 set() [2/2]	593
12.270.2.27 gather() [2/2]	593
12.270.2.28 load() [2/2]	593
12.270.2.29 loadu() [2/2]	593

12.270.2.30 store() [2/2]	593
12.270.2.31 storeu() [2/2]	593
12.270.2.32 stream() [2/2]	593
12.270.2.33 sll()	593
12.270.2.34 srl()	594
12.270.2.35 shuffle()	594
12.270.2.36 unpacklo_twice()	594
12.270.2.37 unpackhi_twice()	594
12.270.2.38 unpacklo()	594
12.270.2.39 unpackhi()	594
12.270.2.40 unpacklohi()	594
12.270.2.41 blend_twice()	594
12.270.2.42 add()	594
12.270.2.43 addin()	595
12.270.2.44 sub()	595
12.270.2.45 subin()	595
12.270.2.46 mullo()	595
12.270.2.47 mul()	595
12.270.2.48 fmadd()	595
12.270.2.49 fmaddin()	595
12.270.2.50 fnmadd()	595
12.270.2.51 fnmaddin()	595
12.270.2.52 fmsub()	596
12.270.2.53 fmsubin()	596
12.270.2.54 eq()	596
12.270.2.55 round()	596
12.270.2.56 mod()	596
12.270.2.57 type_string()	596
12.270.2.58 zero()	596
12.270.3 Field Documentation	596
12.270.3.1 vect_size	596
12.270.3.2 alignment	596
12.271 Simd256_impl< true, true, false, 4 > Struct Reference	597
12.271.1 Member Typedef Documentation	600
12.271.1.1 scalar_t [1/2]	600
12.271.1.2 simdHalf [1/2]	600
12.271.1.3 scalar_t [2/2]	600
12.271.1.4 simdHalf [2/2]	600
12.271.1.5 vect_t [1/2]	600
12.271.1.6 vect_t [2/2]	600
12.271.1.7 half_t [1/2]	600
12.271.1.8 half_t [2/2]	600

12.271.2 Member Function Documentation	600
12.271.2.1 set1() [1/3]	600
12.271.2.2 set() [1/4]	600
12.271.2.3 gather() [1/3]	601
12.271.2.4 load() [1/3]	601
12.271.2.5 loadu() [1/3]	601
12.271.2.6 store() [1/3]	601
12.271.2.7 storeu() [1/3]	601
12.271.2.8 stream() [1/3]	601
12.271.2.9 sra() [1/2]	601
12.271.2.10 greater() [1/2]	601
12.271.2.11 lesser() [1/2]	601
12.271.2.12 greater_eq() [1/2]	601
12.271.2.13 lesser_eq() [1/2]	602
12.271.2.14 mulhi() [1/2]	602
12.271.2.15 mulx() [1/2]	602
12.271.2.16 fmaddx() [1/2]	602
12.271.2.17 fmaddxin() [1/2]	602
12.271.2.18 fnmaddx() [1/2]	602
12.271.2.19 fnmaddxin() [1/2]	602
12.271.2.20 fmsubx() [1/2]	602
12.271.2.21 fmsubxin() [1/2]	602
12.271.2.22 hadd_to_scal() [1/2]	603
12.271.2.23 set1() [2/3]	603
12.271.2.24 set() [2/4]	603
12.271.2.25 gather() [2/3]	603
12.271.2.26 load() [2/3]	603
12.271.2.27 loadu() [2/3]	603
12.271.2.28 store() [2/3]	603
12.271.2.29 storeu() [2/3]	603
12.271.2.30 stream() [2/3]	603
12.271.2.31 sra() [2/2]	604
12.271.2.32 greater() [2/2]	604
12.271.2.33 lesser() [2/2]	604
12.271.2.34 greater_eq() [2/2]	604
12.271.2.35 lesser_eq() [2/2]	604
12.271.2.36 mulhi() [2/2]	604
12.271.2.37 mulx() [2/2]	604
12.271.2.38 fmaddx() [2/2]	604
12.271.2.39 fmaddxin() [2/2]	604
12.271.2.40 fnmaddx() [2/2]	605
12.271.2.41 fnmaddxin() [2/2]	605

12.271.2.42 fmsubx() [2/2]	605
12.271.2.43 fmsubxin() [2/2]	605
12.271.2.44 hadd_to_scal() [2/2]	605
12.271.2.45 valid() [1/2]	605
12.271.2.46 valid() [2/2]	605
12.271.2.47 compliant() [1/2]	605
12.271.2.48 compliant() [2/2]	605
12.271.2.49 set1() [3/3]	605
12.271.2.50 set() [3/4]	606
12.271.2.51 set() [4/4]	606
12.271.2.52 gather() [3/3]	606
12.271.2.53 load() [3/3]	606
12.271.2.54 loadu() [3/3]	606
12.271.2.55 store() [3/3]	606
12.271.2.56 storeu() [3/3]	606
12.271.2.57 stream() [3/3]	607
12.271.2.58 sll() [1/2]	607
12.271.2.59 sll() [2/2]	607
12.271.2.60 srl() [1/2]	607
12.271.2.61 srl() [2/2]	607
12.271.2.62 shuffle_twice() [1/2]	607
12.271.2.63 shuffle_twice() [2/2]	607
12.271.2.64 shuffle() [1/2]	607
12.271.2.65 shuffle() [2/2]	607
12.271.2.66 unpacklo_twice()	607
12.271.2.67 unpackhi_twice()	608
12.271.2.68 unpacklo()	608
12.271.2.69 unpackhi()	608
12.271.2.70 unpacklohi()	608
12.271.2.71 blend()	608
12.271.2.72 add() [1/2]	608
12.271.2.73 add() [2/2]	608
12.271.2.74 addin() [1/2]	608
12.271.2.75 addin() [2/2]	608
12.271.2.76 sub() [1/2]	609
12.271.2.77 sub() [2/2]	609
12.271.2.78 subin() [1/2]	609
12.271.2.79 subin() [2/2]	609
12.271.2.80 mullo() [1/2]	609
12.271.2.81 mullo() [2/2]	609
12.271.2.82 mul() [1/2]	609
12.271.2.83 mul() [2/2]	609

12.271.2.84 fmadd() [1/2]	609
12.271.2.85 fmadd() [2/2]	609
12.271.2.86 fmaddin() [1/2]	610
12.271.2.87 fmaddin() [2/2]	610
12.271.2.88 fnmadd() [1/2]	610
12.271.2.89 fnmadd() [2/2]	610
12.271.2.90 fnmaddin() [1/2]	610
12.271.2.91 fnmaddin() [2/2]	610
12.271.2.92 fmsub() [1/2]	610
12.271.2.93 fmsub() [2/2]	610
12.271.2.94 fmsubin() [1/2]	610
12.271.2.95 fmsubin() [2/2]	611
12.271.2.96 eq() [1/2]	611
12.271.2.97 eq() [2/2]	611
12.271.2.98 round() [1/2]	611
12.271.2.99 round() [2/2]	611
12.271.2.100 mod() [1/2]	611
12.271.2.101 mod() [2/2]	611
12.271.2.102 type_string() [1/2]	611
12.271.2.103 type_string() [2/2]	612
12.271.2.104 zero() [1/2]	612
12.271.2.105 zero() [2/2]	612
12.271.2.106 vor()	612
12.271.2.107 vxor()	612
12.271.2.108 vand()	612
12.271.2.109 vandnot()	612
12.271.3 Field Documentation	612
12.271.3.1 vect_size	612
12.271.3.2 alignment	612
12.272 Simd256_impl< true, true, false, 8 > Struct Reference	612
12.272.1 Member Typedef Documentation	614
12.272.1.1 scalar_t	614
12.272.1.2 simdHalf	614
12.272.1.3 vect_t	615
12.272.1.4 half_t	615
12.272.2 Member Function Documentation	615
12.272.2.1 set1() [1/2]	615
12.272.2.2 set() [1/2]	615
12.272.2.3 gather() [1/2]	615
12.272.2.4 load() [1/2]	615
12.272.2.5 loadu() [1/2]	615
12.272.2.6 store() [1/2]	615

12.272.2.7 storeu() [1/2]	615
12.272.2.8 stream() [1/2]	615
12.272.2.9 sra()	616
12.272.2.10 greater()	616
12.272.2.11 lesser()	616
12.272.2.12 greater_eq()	616
12.272.2.13 lesser_eq()	616
12.272.2.14 mullo()	616
12.272.2.15 mulx()	616
12.272.2.16 fmaddx()	616
12.272.2.17 fmaddxin()	616
12.272.2.18 fnmaddx()	617
12.272.2.19 fnmaddxin()	617
12.272.2.20 fmsubx()	617
12.272.2.21 fmsubxin()	617
12.272.2.22 hadd_to_scal()	617
12.272.2.23 valid()	617
12.272.2.24 compliant()	617
12.272.2.25 set1() [2/2]	617
12.272.2.26 set() [2/2]	617
12.272.2.27 gather() [2/2]	618
12.272.2.28 get()	618
12.272.2.29 load() [2/2]	618
12.272.2.30 loadu() [2/2]	618
12.272.2.31 store() [2/2]	618
12.272.2.32 storeu() [2/2]	618
12.272.2.33 stream() [2/2]	618
12.272.2.34 sll()	618
12.272.2.35 srl()	618
12.272.2.36 shuffle()	618
12.272.2.37 unpacklo_twice()	619
12.272.2.38 unpackhi_twice()	619
12.272.2.39 unpacklo()	619
12.272.2.40 unpackhi()	619
12.272.2.41 unpacklohi()	619
12.272.2.42 blend()	619
12.272.2.43 add()	619
12.272.2.44 addin()	619
12.272.2.45 sub()	619
12.272.2.46 subin()	620
12.272.2.47 mul()	620
12.272.2.48 fmadd()	620

12.272.2.49 fmaddin()	620
12.272.2.50 fnmadd()	620
12.272.2.51 fnmaddin()	620
12.272.2.52 fmsub()	620
12.272.2.53 fmsubin()	620
12.272.2.54 eq()	620
12.272.2.55 round()	621
12.272.2.56 mask_high()	621
12.272.2.57 mulhi_fast()	621
12.272.2.58 mod()	621
12.272.2.59 signbits()	621
12.272.2.60 type_string()	621
12.272.2.61 zero()	621
12.272.3 Field Documentation	621
12.272.3.1 vect_size	621
12.272.3.2 alignment	621
12.273 Simd256_impl< true, true, true, 2 > Struct Reference	621
12.273.1 Member Typedef Documentation	623
12.273.1.1 vect_t	623
12.273.1.2 half_t	623
12.273.1.3 scalar_t	623
12.273.1.4 simdHalf	623
12.273.2 Member Function Documentation	623
12.273.2.1 valid()	623
12.273.2.2 compliant()	624
12.273.2.3 set1()	624
12.273.2.4 set()	624
12.273.2.5 gather()	624
12.273.2.6 load()	624
12.273.2.7 loadu()	624
12.273.2.8 store()	624
12.273.2.9 storeu()	624
12.273.2.10 stream()	625
12.273.2.11 sll()	625
12.273.2.12 srl()	625
12.273.2.13 sra()	625
12.273.2.14 shuffle()	625
12.273.2.15 unpacklo_twice()	625
12.273.2.16 unpackhi_twice()	625
12.273.2.17 unpacklo()	625
12.273.2.18 unpackhi()	625
12.273.2.19 unpacklohi()	625

12.273.2.20	blend_twice()	626
12.273.2.21	add()	626
12.273.2.22	addin()	626
12.273.2.23	sub()	626
12.273.2.24	subin()	626
12.273.2.25	mullo()	626
12.273.2.26	mul()	626
12.273.2.27	mulhi()	626
12.273.2.28	mulx()	626
12.273.2.29	fmadd()	626
12.273.2.30	fmaddin()	627
12.273.2.31	fmaddx()	627
12.273.2.32	fmaddxin()	627
12.273.2.33	fnmadd()	627
12.273.2.34	fnmaddin()	627
12.273.2.35	fnmaddx()	627
12.273.2.36	fnmaddxin()	627
12.273.2.37	fmsub()	627
12.273.2.38	fmsubin()	627
12.273.2.39	fmsubx()	628
12.273.2.40	fmsubxin()	628
12.273.2.41	eq()	628
12.273.2.42	greater()	628
12.273.2.43	lesser()	628
12.273.2.44	greater_eq()	628
12.273.2.45	lesser_eq()	628
12.273.2.46	hadd_to_scal()	628
12.273.2.47	round()	628
12.273.2.48	mod()	629
12.273.2.49	type_string()	629
12.273.2.50	zero()	629
12.273.3	Field Documentation	629
12.273.3.1	vect_size	629
12.273.3.2	alignment	629
12.274	Simd256_impl< true, true, true, 4 > Struct Reference	629
12.274.1	Member Typedef Documentation	632
12.274.1.1	vect_t [1/2]	632
12.274.1.2	half_t [1/2]	632
12.274.1.3	scalar_t [1/2]	632
12.274.1.4	simdHalf [1/2]	632
12.274.1.5	vect_t [2/2]	632
12.274.1.6	half_t [2/2]	632

12.274.1.7 scalar_t [2/2]	632
12.274.1.8 simdHalf [2/2]	632
12.274.2 Member Function Documentation	632
12.274.2.1 valid() [1/2]	632
12.274.2.2 compliant() [1/2]	633
12.274.2.3 set1() [1/2]	633
12.274.2.4 set() [1/2]	633
12.274.2.5 gather() [1/2]	633
12.274.2.6 load() [1/2]	633
12.274.2.7 loadu() [1/2]	633
12.274.2.8 store() [1/2]	633
12.274.2.9 storeu() [1/2]	633
12.274.2.10 stream() [1/2]	633
12.274.2.11 sll() [1/2]	634
12.274.2.12 srl() [1/2]	634
12.274.2.13 sra() [1/2]	634
12.274.2.14 shuffle_twice() [1/2]	634
12.274.2.15 shuffle() [1/2]	634
12.274.2.16 unpacklo_twice()	634
12.274.2.17 unpackhi_twice()	634
12.274.2.18 unpacklo()	634
12.274.2.19 unpackhi()	634
12.274.2.20 unpacklohi()	634
12.274.2.21 blend()	635
12.274.2.22 add() [1/2]	635
12.274.2.23 addin() [1/2]	635
12.274.2.24 sub() [1/2]	635
12.274.2.25 subin() [1/2]	635
12.274.2.26 mullo() [1/2]	635
12.274.2.27 mul() [1/2]	635
12.274.2.28 mulhi() [1/2]	635
12.274.2.29 mulx() [1/2]	635
12.274.2.30 fmadd() [1/2]	635
12.274.2.31 fmaddin() [1/2]	636
12.274.2.32 fmaddx() [1/2]	636
12.274.2.33 fmaddxin() [1/2]	636
12.274.2.34 fnmadd() [1/2]	636
12.274.2.35 fnmaddin() [1/2]	636
12.274.2.36 fnmaddx() [1/2]	636
12.274.2.37 fnmaddxin() [1/2]	636
12.274.2.38 fmsub() [1/2]	636
12.274.2.39 fmsubin() [1/2]	636

12.274.2.40 fmsubx() [1/2]	637
12.274.2.41 fmsubxin() [1/2]	637
12.274.2.42 eq() [1/2]	637
12.274.2.43 greater() [1/2]	637
12.274.2.44 lesser() [1/2]	637
12.274.2.45 greater_eq() [1/2]	637
12.274.2.46 lesser_eq() [1/2]	637
12.274.2.47 hadd_to_scal() [1/2]	637
12.274.2.48 round() [1/2]	637
12.274.2.49 mod() [1/2]	638
12.274.2.50 valid() [2/2]	638
12.274.2.51 compliant() [2/2]	638
12.274.2.52 set1() [2/2]	638
12.274.2.53 set() [2/2]	638
12.274.2.54 gather() [2/2]	638
12.274.2.55 load() [2/2]	638
12.274.2.56 loadu() [2/2]	639
12.274.2.57 store() [2/2]	639
12.274.2.58 storeu() [2/2]	639
12.274.2.59 stream() [2/2]	639
12.274.2.60 sll() [2/2]	639
12.274.2.61 srl() [2/2]	639
12.274.2.62 sra() [2/2]	639
12.274.2.63 shuffle_twice() [2/2]	639
12.274.2.64 shuffle() [2/2]	639
12.274.2.65 add() [2/2]	639
12.274.2.66 addin() [2/2]	640
12.274.2.67 sub() [2/2]	640
12.274.2.68 subin() [2/2]	640
12.274.2.69 mullo() [2/2]	640
12.274.2.70 mul() [2/2]	640
12.274.2.71 mulhi() [2/2]	640
12.274.2.72 mulx() [2/2]	640
12.274.2.73 fmadd() [2/2]	640
12.274.2.74 fmaddin() [2/2]	640
12.274.2.75 fmaddx() [2/2]	641
12.274.2.76 fmaddxin() [2/2]	641
12.274.2.77 fnmadd() [2/2]	641
12.274.2.78 fnmaddin() [2/2]	641
12.274.2.79 fnmaddx() [2/2]	641
12.274.2.80 fnmaddxin() [2/2]	641
12.274.2.81 fmsub() [2/2]	641

12.274.2.82 fmsubin() [2/2]	641
12.274.2.83 fmsubx() [2/2]	641
12.274.2.84 fmsubxin() [2/2]	642
12.274.2.85 eq() [2/2]	642
12.274.2.86 greater() [2/2]	642
12.274.2.87 lesser() [2/2]	642
12.274.2.88 greater_eq() [2/2]	642
12.274.2.89 lesser_eq() [2/2]	642
12.274.2.90 hadd_to_scal() [2/2]	642
12.274.2.91 round() [2/2]	642
12.274.2.92 mod() [2/2]	642
12.274.2.93 type_string() [1/2]	643
12.274.2.94 zero() [1/2]	643
12.274.2.95 type_string() [2/2]	643
12.274.2.96 zero() [2/2]	643
12.274.2.97 vor()	643
12.274.2.98 vxor()	643
12.274.2.99 vand()	643
12.274.2.100 vandnot()	643
12.274.3 Field Documentation	643
12.274.3.1 vect_size	643
12.274.3.2 alignment	643
12.275 Simd256_impl< true, true, true, 8 > Struct Reference	644
12.275.1 Member Typedef Documentation	645
12.275.1.1 vect_t	645
12.275.1.2 half_t	645
12.275.1.3 scalar_t	645
12.275.1.4 simdHalf	645
12.275.2 Member Function Documentation	646
12.275.2.1 valid()	646
12.275.2.2 compliant()	646
12.275.2.3 set1()	646
12.275.2.4 set()	646
12.275.2.5 gather()	646
12.275.2.6 get()	646
12.275.2.7 load()	646
12.275.2.8 loadu()	646
12.275.2.9 store()	646
12.275.2.10 storeu()	646
12.275.2.11 stream()	647
12.275.2.12 sll()	647
12.275.2.13 srl()	647

12.275.2.14 sra()	647
12.275.2.15 shuffle()	647
12.275.2.16 unpacklo_twice()	647
12.275.2.17 unpackhi_twice()	647
12.275.2.18 unpacklo()	647
12.275.2.19 unpackhi()	647
12.275.2.20 unpacklohi()	647
12.275.2.21 blend()	648
12.275.2.22 add()	648
12.275.2.23 addin()	648
12.275.2.24 sub()	648
12.275.2.25 subin()	648
12.275.2.26 mullo()	648
12.275.2.27 mul()	648
12.275.2.28 mulx()	648
12.275.2.29 fmadd()	648
12.275.2.30 fmaddin()	649
12.275.2.31 fmaddx()	649
12.275.2.32 fmaddxin()	649
12.275.2.33 fnmadd()	649
12.275.2.34 fnmaddin()	649
12.275.2.35 fnmaddx()	649
12.275.2.36 fnmaddxin()	649
12.275.2.37 fmsub()	649
12.275.2.38 fmsubin()	649
12.275.2.39 fmsubx()	650
12.275.2.40 fmsubxin()	650
12.275.2.41 eq()	650
12.275.2.42 greater()	650
12.275.2.43 lesser()	650
12.275.2.44 greater_eq()	650
12.275.2.45 lesser_eq()	650
12.275.2.46 hadd_to_scal()	650
12.275.2.47 round()	650
12.275.2.48 mask_high()	650
12.275.2.49 mulhi_fast()	651
12.275.2.50 mod()	651
12.275.2.51 signbits()	651
12.275.2.52 type_string()	651
12.275.2.53 zero()	651
12.275.3 Field Documentation	651
12.275.3.1 vect_size	651

12.275.3.2 alignment	651
12.276 Simd256fp_base Struct Reference	651
12.277 Simd256i_base Struct Reference	652
12.277.1 Member Typedef Documentation	652
12.277.1.1 vect_t	652
12.277.2 Member Function Documentation	652
12.277.2.1 type_string()	652
12.277.2.2 zero()	652
12.278 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference	652
12.279 Simd512_impl< true, false, true, 4 > Struct Reference	652
12.279.1 Member Function Documentation	653
12.279.1.1 type_string()	653
12.280 Simd512_impl< true, false, true, 8 > Struct Reference	653
12.280.1 Member Typedef Documentation	654
12.280.1.1 vect_t	654
12.280.1.2 scalar_t	654
12.280.2 Member Function Documentation	654
12.280.2.1 valid()	654
12.280.2.2 compliant()	654
12.280.2.3 zero()	654
12.280.2.4 set1()	654
12.280.2.5 set()	655
12.280.2.6 gather()	655
12.280.2.7 load()	655
12.280.2.8 loadu()	655
12.280.2.9 store()	655
12.280.2.10 storeu()	655
12.280.2.11 stream()	655
12.280.2.12 shuffle()	655
12.280.2.13 unpacklo_twice()	655
12.280.2.14 unpackhi_twice()	656
12.280.2.15 blend()	656
12.280.2.16 blendv()	656
12.280.2.17 add()	656
12.280.2.18 addin()	656
12.280.2.19 sub()	656
12.280.2.20 subin()	656
12.280.2.21 mul()	656
12.280.2.22 mulin()	656
12.280.2.23 div()	656
12.280.2.24 fmadd()	657
12.280.2.25 fmaddin()	657

12.280.2.26	fnmadd()	657
12.280.2.27	fnmaddin()	657
12.280.2.28	fmsub()	657
12.280.2.29	fmsubin()	657
12.280.2.30	eq()	657
12.280.2.31	lesser()	657
12.280.2.32	lesser_eq()	657
12.280.2.33	greater()	658
12.280.2.34	greater_eq()	658
12.280.2.35	floor()	658
12.280.2.36	ceil()	658
12.280.2.37	round()	658
12.280.2.38	hadd()	658
12.280.2.39	hadd_to_scal()	658
12.280.2.40	type_string()	658
12.280.3	Field Documentation	658
12.280.3.1	vect_size	658
12.280.3.2	alignment	658
12.281	Simd512_impl< true, true, false, 8 > Struct Reference	658
12.281.1	Member Typedef Documentation	661
12.281.1.1	scalar_t	661
12.281.1.2	simdHalf	661
12.281.1.3	vect_t	661
12.281.1.4	half_t	661
12.281.2	Member Function Documentation	661
12.281.2.1	set1() [1/2]	661
12.281.2.2	set() [1/3]	661
12.281.2.3	gather() [1/2]	661
12.281.2.4	load() [1/2]	661
12.281.2.5	loadu() [1/2]	661
12.281.2.6	store() [1/2]	661
12.281.2.7	maskstore() [1/2]	662
12.281.2.8	storeu() [1/2]	662
12.281.2.9	stream() [1/2]	662
12.281.2.10	sra()	662
12.281.2.11	greater()	662
12.281.2.12	lesser()	662
12.281.2.13	greater_eq()	662
12.281.2.14	lesser_eq()	662
12.281.2.15	mullo()	662
12.281.2.16	mulx()	662
12.281.2.17	fmaddx()	663

12.281.2.18 fmaddxin()	663
12.281.2.19 fnmaddx()	663
12.281.2.20 fnmaddxin()	663
12.281.2.21 fmsubx()	663
12.281.2.22 fmsubxin()	663
12.281.2.23 hadd_to_scal()	663
12.281.2.24 valid()	663
12.281.2.25 compliant()	663
12.281.2.26 set1() [2/2]	664
12.281.2.27 set() [2/3]	664
12.281.2.28 set() [3/3]	664
12.281.2.29 gather() [2/2]	664
12.281.2.30 load() [2/2]	664
12.281.2.31 loadu() [2/2]	664
12.281.2.32 store() [2/2]	664
12.281.2.33 maskstore() [2/2]	664
12.281.2.34 storeu() [2/2]	664
12.281.2.35 stream() [2/2]	665
12.281.2.36 sll()	665
12.281.2.37 srl()	665
12.281.2.38 shuffle()	665
12.281.2.39 unpacklo_twice()	665
12.281.2.40 unpackhi_twice()	665
12.281.2.41 unpacklo()	665
12.281.2.42 unpackhi()	665
12.281.2.43 unpacklohi()	665
12.281.2.44 blend()	666
12.281.2.45 add()	666
12.281.2.46 addin()	666
12.281.2.47 sub()	666
12.281.2.48 subin()	666
12.281.2.49 mul()	666
12.281.2.50 fmadd()	666
12.281.2.51 fmaddin()	666
12.281.2.52 fnmadd()	666
12.281.2.53 fnmaddin()	667
12.281.2.54 fmsub()	667
12.281.2.55 fmsubin()	667
12.281.2.56 eq()	667
12.281.2.57 round()	667
12.281.2.58 mask_high()	667
12.281.2.59 mulhi_fast()	667

12.281.2.60 mod()	667
12.281.2.61 signbits()	667
12.281.2.62 type_string()	668
12.281.2.63 zero()	668
12.281.2.64 vor()	668
12.281.2.65 vxor()	668
12.281.2.66 vand()	668
12.281.2.67 vandnot()	668
12.281.3 Field Documentation	668
12.281.3.1 vect_size	668
12.281.3.2 alignment	668
12.282 Simd512_impl< true, true, true, 8 > Struct Reference	668
12.282.1 Member Typedef Documentation	670
12.282.1.1 vect_t	670
12.282.1.2 half_t	670
12.282.1.3 scalar_t	670
12.282.1.4 simdHalf	670
12.282.2 Member Function Documentation	670
12.282.2.1 valid()	670
12.282.2.2 compliant()	671
12.282.2.3 set1()	671
12.282.2.4 set() [1/2]	671
12.282.2.5 set() [2/2]	671
12.282.2.6 gather()	671
12.282.2.7 load()	671
12.282.2.8 loadu()	671
12.282.2.9 store()	671
12.282.2.10 maskstore()	671
12.282.2.11 storeu()	672
12.282.2.12 stream()	672
12.282.2.13 sll()	672
12.282.2.14 srl()	672
12.282.2.15 sra()	672
12.282.2.16 shuffle()	672
12.282.2.17 unpacklo_twice()	672
12.282.2.18 unpackhi_twice()	672
12.282.2.19 unpacklo()	672
12.282.2.20 unpackhi()	672
12.282.2.21 unpacklohi()	673
12.282.2.22 blend()	673
12.282.2.23 add()	673
12.282.2.24 addin()	673

12.282.2.25 sub()	673
12.282.2.26 subin()	673
12.282.2.27 mullo()	673
12.282.2.28 mul()	673
12.282.2.29 mulx()	673
12.282.2.30 fmadd()	674
12.282.2.31 fmaddin()	674
12.282.2.32 fmaddx()	674
12.282.2.33 fmaddxin()	674
12.282.2.34 fnmadd()	674
12.282.2.35 fnmaddin()	674
12.282.2.36 fnmaddx()	674
12.282.2.37 fnmaddxin()	674
12.282.2.38 fmsub()	674
12.282.2.39 fmsubin()	675
12.282.2.40 fmsubx()	675
12.282.2.41 fmsubxin()	675
12.282.2.42 eq()	675
12.282.2.43 greater()	675
12.282.2.44 lesser()	675
12.282.2.45 greater_eq()	675
12.282.2.46 lesser_eq()	675
12.282.2.47 hadd_to_scal()	675
12.282.2.48 round()	676
12.282.2.49 mask_high()	676
12.282.2.50 mulhi_fast()	676
12.282.2.51 mod()	676
12.282.2.52 signbits()	676
12.282.2.53 type_string()	676
12.282.2.54 zero()	676
12.282.2.55 vor()	676
12.282.2.56 vxor()	676
12.282.2.57 vand()	676
12.282.2.58 vandnot()	677
12.282.3 Field Documentation	677
12.282.3.1 vect_size	677
12.282.3.2 alignment	677
12.283 Simd512fp_base Struct Reference	677
12.283.1 Member Function Documentation	677
12.283.1.1 type_string()	677
12.284 Simd512i_base Struct Reference	677
12.284.1 Member Typedef Documentation	678

12.284.1.1 vect_t	678
12.284.2 Member Function Documentation	678
12.284.2.1 type_string()	678
12.284.2.2 zero()	678
12.284.2.3 vor()	678
12.284.2.4 vxor()	678
12.284.2.5 vand()	678
12.284.2.6 vandnot()	678
12.285 SimdChooser< T, bool, bool > Struct Template Reference	678
12.286 SimdChooser< T, false, b > Struct Template Reference	679
12.286.1 Member Typedef Documentation	679
12.286.1.1 value	679
12.287 SimdChooser< T, true, false > Struct Template Reference	679
12.287.1 Member Typedef Documentation	679
12.287.1.1 value	679
12.288 SimdChooser< T, true, true > Struct Template Reference	679
12.288.1 Member Typedef Documentation	679
12.288.1.1 value	679
12.289 simdToType< T > Struct Template Reference	679
12.290 Single Struct Reference	680
12.291 Sparse< Field, SparseMatrix_t, IdxT, PtrT > Struct Template Reference	680
12.292 Sparse< _Field, SparseMatrix_t::COO > Struct Template Reference	680
12.292.1 Member Typedef Documentation	680
12.292.1.1 Field	680
12.292.2 Field Documentation	680
12.292.2.1 col	680
12.292.2.2 row	681
12.292.2.3 dat	681
12.292.2.4 delayed	681
12.292.2.5 kmax	681
12.292.2.6 m	681
12.292.2.7 n	681
12.292.2.8 nnz	681
12.292.2.9 nElements	681
12.292.2.10 maxrow	681
12.293 Sparse< _Field, SparseMatrix_t::COO_ZO > Struct Template Reference	681
12.293.1 Member Typedef Documentation	682
12.293.1.1 Field	682
12.293.2 Field Documentation	682
12.293.2.1 cst	682
12.293.2.2 col	682
12.293.2.3 row	682

12.293.2.4 dat	682
12.293.2.5 delayed	682
12.293.2.6 kmax	682
12.293.2.7 m	682
12.293.2.8 n	683
12.293.2.9 nnz	683
12.293.2.10 nElements	683
12.293.2.11 maxrow	683
12.294 Sparse< _Field, SparseMatrix_t::CSR > Struct Template Reference	683
12.294.1 Member Typedef Documentation	683
12.294.1.1 Field	683
12.294.2 Field Documentation	684
12.294.2.1 delayed	684
12.294.2.2 kmax	684
12.294.2.3 m	684
12.294.2.4 n	684
12.294.2.5 nnz	684
12.294.2.6 nElements	684
12.294.2.7 maxrow	684
12.294.2.8 col	684
12.294.2.9 st	684
12.294.2.10 stend	684
12.294.2.11 dat	684
12.295 Sparse< _Field, SparseMatrix_t::CSR_HYB > Struct Template Reference	685
12.295.1 Member Typedef Documentation	685
12.295.1.1 Field	685
12.295.2 Field Documentation	685
12.295.2.1 delayed	685
12.295.2.2 col	685
12.295.2.3 st	685
12.295.2.4 dat	685
12.295.2.5 kmax	685
12.295.2.6 m	686
12.295.2.7 n	686
12.295.2.8 nnz	686
12.295.2.9 nElements	686
12.295.2.10 maxrow	686
12.295.2.11 nOnes	686
12.295.2.12 nMOnes	686
12.295.2.13 nOthers	686
12.296 Sparse< _Field, SparseMatrix_t::CSR_ZO > Struct Template Reference	686
12.296.1 Member Typedef Documentation	687

12.296.1.1 Field	687
12.296.2 Field Documentation	687
12.296.2.1 cst	687
12.296.2.2 delayed	687
12.296.2.3 kmax	687
12.296.2.4 m	687
12.296.2.5 n	687
12.296.2.6 nnz	687
12.296.2.7 nElements	687
12.296.2.8 maxrow	688
12.296.2.9 col	688
12.296.2.10 st	688
12.296.2.11 stend	688
12.296.2.12 dat	688
12.297 Sparse< _Field, SparseMatrix_t::ELL > Struct Template Reference	688
12.297.1 Member Typedef Documentation	689
12.297.1.1 Field	689
12.297.2 Field Documentation	689
12.297.2.1 delayed	689
12.297.2.2 kmax	689
12.297.2.3 m	689
12.297.2.4 n	689
12.297.2.5 ld	689
12.297.2.6 nnz	689
12.297.2.7 nElements	689
12.297.2.8 maxrow	689
12.297.2.9 col	689
12.297.2.10 dat	689
12.298 Sparse< _Field, SparseMatrix_t::ELL_simd > Struct Template Reference	690
12.298.1 Field Documentation	690
12.298.1.1 delayed	690
12.298.1.2 chunk	690
12.298.1.3 m	690
12.298.1.4 n	690
12.298.1.5 ld	690
12.298.1.6 kmax	690
12.298.1.7 nnz	691
12.298.1.8 nElements	691
12.298.1.9 maxrow	691
12.298.1.10 nChunks	691
12.298.1.11 col	691
12.298.1.12 dat	691

12.299 Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > Struct Template Reference	691
12.299.1 Field Documentation	692
12.299.1.1 cst	692
12.299.1.2 delayed	692
12.299.1.3 chunk	692
12.299.1.4 m	692
12.299.1.5 n	692
12.299.1.6 ld	692
12.299.1.7 kmax	692
12.299.1.8 nnz	692
12.299.1.9 nElements	692
12.299.1.10 maxrow	692
12.299.1.11 nChunks	692
12.299.1.12 col	693
12.299.1.13 dat	693
12.300 Sparse< _Field, SparseMatrix_t::ELL_ZO > Struct Template Reference	693
12.300.1 Member Typedef Documentation	693
12.300.1.1 Field	693
12.300.2 Field Documentation	693
12.300.2.1 cst	693
12.300.2.2 delayed	694
12.300.2.3 kmax	694
12.300.2.4 m	694
12.300.2.5 n	694
12.300.2.6 ld	694
12.300.2.7 nnz	694
12.300.2.8 nElements	694
12.300.2.9 maxrow	694
12.300.2.10 col	694
12.300.2.11 dat	694
12.301 Sparse< _Field, SparseMatrix_t::HYB_ZO > Struct Template Reference	694
12.301.1 Member Typedef Documentation	695
12.301.1.1 Field	695
12.301.1.2 Self_t	695
12.301.2 Field Documentation	695
12.301.2.1 delayed	695
12.301.2.2 kmax	695
12.301.2.3 m	695
12.301.2.4 n	695
12.301.2.5 nnz	695
12.301.2.6 maxrow	695
12.301.2.7 nElements	696

12.301.2.8 dat	696
12.301.2.9 one	696
12.301.2.10 mone	696
12.302 Sparse< _Field, SparseMatrix_t::SELL > Struct Template Reference	696
12.302.1 Member Typedef Documentation	697
12.302.1.1 Field	697
12.302.2 Field Documentation	697
12.302.2.1 delayed	697
12.302.2.2 chunk	697
12.302.2.3 kmax	697
12.302.2.4 m	697
12.302.2.5 n	697
12.302.2.6 maxrow	697
12.302.2.7 sigma	697
12.302.2.8 nChunks	697
12.302.2.9 nnz	697
12.302.2.10 nElements	697
12.302.2.11 perm	698
12.302.2.12 st	698
12.302.2.13 chunkSize	698
12.302.2.14 col	698
12.302.2.15 dat	698
12.303 Sparse< _Field, SparseMatrix_t::SELL_ZO > Struct Template Reference	698
12.303.1 Member Typedef Documentation	699
12.303.1.1 Field	699
12.303.2 Field Documentation	699
12.303.2.1 cst	699
12.303.2.2 delayed	699
12.303.2.3 chunk	699
12.303.2.4 kmax	699
12.303.2.5 m	699
12.303.2.6 n	699
12.303.2.7 maxrow	699
12.303.2.8 sigma	699
12.303.2.9 nChunks	699
12.303.2.10 nnz	700
12.303.2.11 nElements	700
12.303.2.12 perm	700
12.303.2.13 st	700
12.303.2.14 chunkSize	700
12.303.2.15 col	700
12.303.2.16 dat	700

12.304 SpMat< Field, flag > Struct Template Reference	700
12.304.1 Field Documentation	700
12.304.1.1 _coo	700
12.304.1.2 _csr	700
12.304.1.3 _ell	701
12.305 Static_error_check< bool > Class Template Reference	701
12.305.1 Constructor & Destructor Documentation	701
12.305.1.1 Static_error_check()	701
12.306 Static_error_check< false > Class Reference	701
12.307 StatsMatrix Struct Reference	701
12.307.1 Field Documentation	702
12.307.1.1 rowdim	702
12.307.1.2 coldim	702
12.307.1.3 nOnes	702
12.307.1.4 nMOnes	702
12.307.1.5 nOthers	702
12.307.1.6 nnz	702
12.307.1.7 maxRow	702
12.307.1.8 minRow	702
12.307.1.9 averageRow	702
12.307.1.10 deviationRow	702
12.307.1.11 maxCol	702
12.307.1.12 minCol	703
12.307.1.13 averageCol	703
12.307.1.14 deviationCol	703
12.307.1.15 minColDifference	703
12.307.1.16 maxColDifference	703
12.307.1.17 averageColDifference	703
12.307.1.18 deviationColDifference	703
12.307.1.19 minRowDifference	703
12.307.1.20 maxRowDifference	703
12.307.1.21 averageRowDifference	703
12.307.1.22 deviationRowDifference	703
12.307.1.23 nDenseRows	703
12.307.1.24 nDenseCols	703
12.307.1.25 nEmptyRows	703
12.307.1.26 nEmptyCols	703
12.307.1.27 nEmptyColsEnd	704
12.307.1.28 denseRows	704
12.307.1.29 denseCols	704
12.308 support_fast_mod< T > Struct Template Reference	704
12.309 support_fast_mod< double > Struct Reference	704

12.310 support_fast_mod< float > Struct Reference	704
12.311 support_fast_mod< int64_t > Struct Reference	705
12.312 support_simd< T > Struct Template Reference	705
12.313 support_simd_add< T > Struct Template Reference	705
12.314 support_simd_mod< T > Struct Template Reference	705
12.315 tfn_minus Struct Reference	706
12.315.1 Member Function Documentation	706
12.315.1.1 operator>()	706
12.316 tfn_minus_eq Struct Reference	706
12.316.1 Member Function Documentation	706
12.316.1.1 operator>()	706
12.317 tfn_mul Struct Reference	706
12.317.1 Member Function Documentation	707
12.317.1.1 operator>()	707
12.318 tfn_mul_eq Struct Reference	707
12.318.1 Member Function Documentation	707
12.318.1.1 operator>()	707
12.319 tfn_plus Struct Reference	707
12.319.1 Member Function Documentation	707
12.319.1.1 operator>()	707
12.320 tfn_plus_eq Struct Reference	708
12.320.1 Member Function Documentation	708
12.320.1.1 operator>()	708
12.321 Threads Struct Reference	708
12.322 ThreeD Struct Reference	708
12.323 ThreeDAdaptive Struct Reference	708
12.324 ThreeDInPlace Struct Reference	708
12.325 TRSMHelper< ReclterTrait, ParSeqTrait > Struct Template Reference	708
12.325.1 Detailed Description	709
12.325.2 Constructor & Destructor Documentation	709
12.325.2.1 TRSMHelper() [1/3]	709
12.325.2.2 TRSMHelper() [2/3]	709
12.325.2.3 TRSMHelper() [3/3]	709
12.325.3 Member Function Documentation	709
12.325.3.1 pMMH() [1/2]	709
12.325.3.2 pMMH() [2/2]	710
12.325.4 Field Documentation	710
12.325.4.1 parseq	710
12.326 TwoD Struct Reference	710
12.327 TwoDAdaptive Struct Reference	710
12.328 UnparametricTag Struct Reference	710
12.328.1 Detailed Description	710

12.329 Winograd Struct Reference	710
12.330 WinogradPar Struct Reference	710
13 File Documentation	711
13.1 arithprog.C File Reference	711
13.1.1 Macro Definition Documentation	711
13.1.1.1 CUBE	711
13.1.1.2 GFOPS	711
13.1.2 Typedef Documentation	711
13.1.2.1 TTimer	711
13.1.3 Function Documentation	712
13.1.3.1 main()	712
13.2 fsyrk.C File Reference	712
13.2.1 Macro Definition Documentation	712
13.2.1.1 CUBE	712
13.2.1.2 GFOPS	712
13.2.2 Typedef Documentation	712
13.2.2.1 TTimer	712
13.2.3 Function Documentation	712
13.2.3.1 main()	712
13.3 fsytrf.C File Reference	713
13.3.1 Macro Definition Documentation	713
13.3.1.1 CUBE	713
13.3.1.2 GFOPS	713
13.3.2 Typedef Documentation	713
13.3.2.1 TTimer	713
13.3.3 Function Documentation	713
13.3.3.1 main()	713
13.4 ftrtri.C File Reference	713
13.4.1 Macro Definition Documentation	714
13.4.1.1 CUBE	714
13.4.1.2 GFOPS	714
13.4.2 Typedef Documentation	714
13.4.2.1 TTimer	714
13.4.3 Function Documentation	714
13.4.3.1 main()	714
13.5 winograd.C File Reference	714
13.5.1 Macro Definition Documentation	715
13.5.1.1 DOUBLE_TO_FLOAT_CROSSOVER	715
13.5.1.2 GFOPS	715
13.5.2 Typedef Documentation	715
13.5.2.1 TTimer	715

13.5.3 Function Documentation	715
13.5.3.1 balanced() [1/2]	715
13.5.3.2 balanced() [2/2]	715
13.5.3.3 main()	715
13.6 benchmark-charpoly-mp.C File Reference	715
13.6.1 Macro Definition Documentation	716
13.6.1.1 __FFLASFFPACK_FORCE_SEQ	716
13.6.2 Function Documentation	716
13.6.2.1 main()	716
13.7 benchmark-charpoly.C File Reference	716
13.7.1 Macro Definition Documentation	716
13.7.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	716
13.7.2 Function Documentation	716
13.7.2.1 run_with_field()	716
13.7.2.2 main()	717
13.8 benchmark-checkers.C File Reference	717
13.8.1 Macro Definition Documentation	717
13.8.1.1 ENABLE_ALL_CHECKINGS	717
13.8.1.2 _NR_TESTS	717
13.8.1.3 _MAX_SIZE_MATRICES	717
13.8.1.4 CUBE	717
13.8.2 Function Documentation	717
13.8.2.1 main()	717
13.9 benchmark-dgemm.C File Reference	718
13.9.1 Macro Definition Documentation	718
13.9.1.1 CBLAS_GEMM	718
13.9.2 Typedef Documentation	718
13.9.2.1 TTimer	718
13.9.2.2 Floats	718
13.9.3 Function Documentation	718
13.9.3.1 main()	718
13.10 benchmark-dgetrf.C File Reference	718
13.10.1 Macro Definition Documentation	719
13.10.1.1 __FFLASFFPACK_HAVE_DGETRF	719
13.10.2 Typedef Documentation	719
13.10.2.1 TTimer	719
13.10.3 Function Documentation	719
13.10.3.1 main()	719
13.11 benchmark-dgetri.C File Reference	719
13.11.1 Typedef Documentation	719
13.11.1.1 TTimer	719
13.11.2 Function Documentation	720

13.11.2.1 main()	720
13.12 benchmark-dsytrf.C File Reference	720
13.12.1 Macro Definition Documentation	720
13.12.1.1 EFGFF	720
13.12.2 Typedef Documentation	720
13.12.2.1 TTimer	720
13.12.3 Function Documentation	720
13.12.3.1 main()	720
13.13 benchmark-dtrsm.C File Reference	721
13.13.1 Typedef Documentation	721
13.13.1.1 TTimer	721
13.13.2 Function Documentation	721
13.13.2.1 main()	721
13.14 benchmark-dtrtri.C File Reference	721
13.14.1 Macro Definition Documentation	722
13.14.1.1 __FFLASFFPACK_HAVE_DTRTRI	722
13.14.2 Typedef Documentation	722
13.14.2.1 TTimer	722
13.14.3 Function Documentation	722
13.14.3.1 main()	722
13.15 benchmark-fadd-lvl2.C File Reference	722
13.15.1 Macro Definition Documentation	722
13.15.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	722
13.15.2 Function Documentation	722
13.15.2.1 main()	722
13.16 benchmark-fdot.C File Reference	722
13.16.1 Macro Definition Documentation	723
13.16.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	723
13.16.2 Function Documentation	723
13.16.2.1 run_with_field()	723
13.16.2.2 main()	723
13.17 benchmark-fgemm-mp.C File Reference	723
13.17.1 Macro Definition Documentation	724
13.17.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	724
13.17.1.2 MG_DEFAULT	724
13.17.1.3 STD_RECINT_SIZE	724
13.17.2 Function Documentation	724
13.17.2.1 tmain()	724
13.17.2.2 main()	724
13.18 benchmark-fgemm-rns.C File Reference	724
13.18.1 Macro Definition Documentation	725
13.18.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	725

13.18.2 Typedef Documentation	725
13.18.2.1 RNS	725
13.18.2.2 Field	725
13.18.2.3 Element_ptr	725
13.18.2.4 ConstElement_ptr	725
13.18.2.5 THREADS	725
13.18.2.6 GRAIN	725
13.18.2.7 TWOD	725
13.18.2.8 TWODA	725
13.18.2.9 THREED	725
13.18.2.10 THREEDA	725
13.18.2.11 THREEDIP	726
13.18.2.12 PSeq	726
13.18.3 Function Documentation	726
13.18.3.1 main()	726
13.19 benchmark-fgemm.C File Reference	726
13.19.1 Macro Definition Documentation	726
13.19.1.1 CLASSIC_HYBRID	726
13.19.2 Function Documentation	726
13.19.2.1 main()	726
13.20 benchmark-fgemv-mp.C File Reference	726
13.20.1 Macro Definition Documentation	727
13.20.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	727
13.20.1.2 MG_DEFAULT	727
13.20.1.3 STD_RECINT_SIZE	727
13.20.2 Function Documentation	727
13.20.2.1 write_matrix()	727
13.20.2.2 tmain()	727
13.20.2.3 main()	727
13.21 benchmark-fgemv.C File Reference	728
13.21.1 Macro Definition Documentation	728
13.21.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	728
13.21.2 Function Documentation	729
13.21.2.1 fill_value()	729
13.21.2.2 genData()	729
13.21.2.3 check_result()	729
13.21.2.4 benchmark_with_timer()	729
13.21.2.5 benchmark_disp()	730
13.21.2.6 benchmark_in_Field()	730
13.21.2.7 benchmark_with_field() [1/2]	730
13.21.2.8 benchmark_with_field() [2/2]	730
13.21.2.9 main()	731

13.22 benchmark-fgesv.C File Reference	731
13.22.1 Macro Definition Documentation	731
13.22.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	731
13.22.2 Function Documentation	731
13.22.2.1 main()	731
13.23 benchmark-fsyrk.C File Reference	731
13.23.1 Macro Definition Documentation	732
13.23.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	732
13.23.1.2 CUBE	732
13.23.2 Function Documentation	732
13.23.2.1 main()	732
13.24 benchmark-fsytrf.C File Reference	732
13.24.1 Macro Definition Documentation	732
13.24.1.1 __FFPACK_FSYTRF_BC_CROUT	732
13.24.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	732
13.24.1.3 CUBE	733
13.24.2 Function Documentation	733
13.24.2.1 main()	733
13.25 benchmark-ftsrm-mp.C File Reference	733
13.25.1 Macro Definition Documentation	733
13.25.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	733
13.25.2 Function Documentation	733
13.25.2.1 main()	733
13.26 benchmark-ftsrm.C File Reference	733
13.26.1 Macro Definition Documentation	734
13.26.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	734
13.26.2 Function Documentation	734
13.26.2.1 main()	734
13.27 benchmark-ftsrv.C File Reference	734
13.27.1 Macro Definition Documentation	734
13.27.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	734
13.27.2 Function Documentation	734
13.27.2.1 main()	734
13.28 benchmark-ftsrti.C File Reference	734
13.28.1 Macro Definition Documentation	735
13.28.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET	735
13.28.1.2 CUBE	735
13.28.2 Function Documentation	735
13.28.2.1 main()	735
13.29 benchmark-inverse.C File Reference	735
13.29.1 Macro Definition Documentation	735
13.29.1.1 CUBE	735

13.29.2 Function Documentation	736
13.29.2.1 main()	736
13.30 benchmark-lqup-mp.C File Reference	736
13.30.1 Function Documentation	736
13.30.1.1 main()	736
13.31 benchmark-lqup.C File Reference	736
13.31.1 Macro Definition Documentation	736
13.31.1.1 CUBE	736
13.31.2 Function Documentation	737
13.31.2.1 main()	737
13.32 benchmark-pluq.C File Reference	737
13.32.1 Macro Definition Documentation	737
13.32.1.1 __FflasFfpack_OPENBLAS_NT_ALREADY_SET	737
13.32.1.2 CUBE	737
13.32.2 Typedef Documentation	737
13.32.2.1 Field	737
13.32.3 Function Documentation	738
13.32.3.1 verification_PLUQ()	738
13.32.3.2 Rec_Initialize()	738
13.32.3.3 main()	738
13.33 benchmark-wino.C File Reference	738
13.33.1 Macro Definition Documentation	738
13.33.1.1 CUBE	738
13.33.2 Function Documentation	739
13.33.2.1 launch_wino()	739
13.33.2.2 main()	739
13.34 mainpage.doxy File Reference	739
13.35 charpoly.C File Reference	739
13.35.1 Macro Definition Documentation	739
13.35.1.1 CUBE	739
13.35.1.2 GFOPS	739
13.35.2 Typedef Documentation	740
13.35.2.1 TTimer	740
13.35.3 Function Documentation	740
13.35.3.1 main()	740
13.36 charpoly.C File Reference	740
13.36.1 Function Documentation	740
13.36.1.1 main()	740
13.37 det.C File Reference	740
13.37.1 Function Documentation	740
13.37.1.1 main()	740
13.38 matmul.C File Reference	741

13.38.1 Function Documentation	741
13.38.1.1 main()	741
13.39 pluq.C File Reference	741
13.39.1 Macro Definition Documentation	741
13.39.1.1 CUBE	741
13.39.1.2 GFOPS	742
13.39.2 Typedef Documentation	742
13.39.2.1 TTimer	742
13.39.3 Function Documentation	742
13.39.3.1 main()	742
13.40 pluq.C File Reference	742
13.40.1 Function Documentation	742
13.40.1.1 main()	742
13.41 rank.C File Reference	742
13.41.1 Function Documentation	743
13.41.1.1 main()	743
13.42 solve.C File Reference	743
13.42.1 Function Documentation	743
13.42.1.1 main()	743
13.43 checker_charpoly.inl File Reference	743
13.43.1 Macro Definition Documentation	743
13.43.1.1 __FFLASFFPACK_checker_charpoly_INL	743
13.44 checker_det.inl File Reference	743
13.44.1 Macro Definition Documentation	744
13.44.1.1 __FFLASFFPACK_checker_det_INL	744
13.45 checker_empty.h File Reference	744
13.46 checker_fgemm.inl File Reference	744
13.46.1 Macro Definition Documentation	744
13.46.1.1 __FFLASFFPACK_checker_fgemm_INL	744
13.47 checker_ftsm.inl File Reference	744
13.47.1 Macro Definition Documentation	745
13.47.1.1 __FFLASFFPACK_checker_ftsm_INL	745
13.48 checker_invert.inl File Reference	745
13.48.1 Macro Definition Documentation	745
13.48.1.1 __FFLASFFPACK_checker_invert_INL	745
13.49 checker_pluq.inl File Reference	745
13.49.1 Macro Definition Documentation	745
13.49.1.1 __FFLASFFPACK_checker_pluq_INL	745
13.50 checkers.doxy File Reference	746
13.51 checkers_fflas.h File Reference	746
13.52 checkers_fflas.inl File Reference	746
13.52.1 Macro Definition Documentation	746

13.52.1.1 FFLASFFPACK_checkers_fflas_inl_H	746
13.53 checkers_ffpack.h File Reference	747
13.54 checkers_ffpack.inl File Reference	747
13.54.1 Macro Definition Documentation	748
13.54.1.1 FFLASFFPACK_checkers_ffpack_inl_H	748
13.55 config-blas.h File Reference	748
13.55.1 Macro Definition Documentation	749
13.55.1.1 CBLAS_INT	749
13.55.1.2 CBLAS_ENUM_DEFINED_H	749
13.55.1.3 CBLAS_EXTERNALS	749
13.55.1.4 blas_enum	749
13.55.2 Enumeration Type Documentation	749
13.55.2.1 CBLAS_ORDER	749
13.55.2.2 CBLAS_TRANSPOSE	749
13.55.2.3 CBLAS_UPLO	749
13.55.2.4 CBLAS_DIAG	749
13.55.2.5 CBLAS_SIDE	750
13.55.3 Function Documentation	750
13.55.3.1 daxpy_()	750
13.55.3.2 saxpy_()	750
13.55.3.3 ddot_()	750
13.55.3.4 sdot_()	750
13.55.3.5 dasum_()	750
13.55.3.6 idamax_()	751
13.55.3.7 dnm2_()	751
13.55.3.8 dgemv_()	751
13.55.3.9 sgemv_()	751
13.55.3.10 dger_()	751
13.55.3.11 sger_()	752
13.55.3.12 dcopy_()	752
13.55.3.13 scopy_()	752
13.55.3.14 dscal_()	752
13.55.3.15 sscal_()	752
13.55.3.16 dtrsm_()	752
13.55.3.17 strsm_()	753
13.55.3.18 dtrmm_()	753
13.55.3.19 strmm_()	753
13.55.3.20 sgemm_()	753
13.55.3.21 dgemm_()	754
13.56 config.h File Reference	754
13.56.1 Macro Definition Documentation	755
13.56.1.1 HAVE_BLAS	755

13.56.1.2 HAVE_CBLAS	755
13.56.1.3 HAVE_CXX11	755
13.56.1.4 HAVE_DLFCN_H	755
13.56.1.5 HAVE_FLOAT_H	755
13.56.1.6 HAVE_INTPYPES_H	755
13.56.1.7 HAVE_LAPACK	755
13.56.1.8 HAVE_LIMITS_H	755
13.56.1.9 HAVE_LITTLE_ENDIAN	755
13.56.1.10 HAVE_PTHREAD_H	755
13.56.1.11 HAVE_STDDEF_H	755
13.56.1.12 HAVE_STDINT_H	755
13.56.1.13 HAVE_STDIO_H	756
13.56.1.14 HAVE_STDLIB_H	756
13.56.1.15 HAVE_STRINGS_H	756
13.56.1.16 HAVE_STRING_H	756
13.56.1.17 HAVE_SYS_STAT_H	756
13.56.1.18 HAVE_SYS_TIME_H	756
13.56.1.19 HAVE_SYS_TYPES_H	756
13.56.1.20 HAVE_UNISTD_H	756
13.56.1.21 LT_OBJDIR	756
13.56.1.22 OPENBLAS_NUM_THREADS	756
13.56.1.23 PACKAGE	756
13.56.1.24 PACKAGE_BUGREPORT	756
13.56.1.25 PACKAGE_NAME	756
13.56.1.26 PACKAGE_STRING	756
13.56.1.27 PACKAGE_TARNAME	756
13.56.1.28 PACKAGE_URL	757
13.56.1.29 PACKAGE_VERSION	757
13.56.1.30 SIZEOF_CHAR	757
13.56.1.31 SIZEOF_INT	757
13.56.1.32 SIZEOF_LONG	757
13.56.1.33 SIZEOF_LONG_LONG	757
13.56.1.34 SIZEOF_SHORT	757
13.56.1.35 SIZEOF__INT64	757
13.56.1.36 STDC_HEADERS	757
13.56.1.37 USE_OPENMP	757
13.56.1.38 VERSION	757
13.57 config.h File Reference	757
13.57.1 Macro Definition Documentation	758
13.57.1.1 __FFLASFFPACK_HAVE_BLAS	758
13.57.1.2 __FFLASFFPACK_HAVE_CBLAS	758
13.57.1.3 __FFLASFFPACK_HAVE_CXX11	758

13.57.1.4	__FFLASFFPACK_HAVE_DLFCN_H	758
13.57.1.5	__FFLASFFPACK_HAVE_FLOAT_H	758
13.57.1.6	__FFLASFFPACK_HAVE_INTPYPES_H	758
13.57.1.7	__FFLASFFPACK_HAVE_LAPACK	758
13.57.1.8	__FFLASFFPACK_HAVE_LIMITS_H	759
13.57.1.9	__FFLASFFPACK_HAVE_LITTLE_ENDIAN	759
13.57.1.10	__FFLASFFPACK_HAVE_PTHREAD_H	759
13.57.1.11	__FFLASFFPACK_HAVE_STDDEF_H	759
13.57.1.12	__FFLASFFPACK_HAVE_STDINT_H	759
13.57.1.13	__FFLASFFPACK_HAVE_STDIO_H	759
13.57.1.14	__FFLASFFPACK_HAVE_STDLIB_H	759
13.57.1.15	__FFLASFFPACK_HAVE_STRINGS_H	759
13.57.1.16	__FFLASFFPACK_HAVE_STRING_H	759
13.57.1.17	__FFLASFFPACK_HAVE_SYS_STAT_H	759
13.57.1.18	__FFLASFFPACK_HAVE_SYS_TIME_H	759
13.57.1.19	__FFLASFFPACK_HAVE_SYS_TYPES_H	759
13.57.1.20	__FFLASFFPACK_HAVE_UNISTD_H	759
13.57.1.21	__FFLASFFPACK_LT_OBJDIR	759
13.57.1.22	__FFLASFFPACK_OPENBLAS_NUM_THREADS	759
13.57.1.23	__FFLASFFPACK_PACKAGE	760
13.57.1.24	__FFLASFFPACK_PACKAGE_BUGREPORT	760
13.57.1.25	__FFLASFFPACK_PACKAGE_NAME	760
13.57.1.26	__FFLASFFPACK_PACKAGE_STRING	760
13.57.1.27	__FFLASFFPACK_PACKAGE_TARNAME	760
13.57.1.28	__FFLASFFPACK_PACKAGE_URL	760
13.57.1.29	__FFLASFFPACK_PACKAGE_VERSION	760
13.57.1.30	__FFLASFFPACK_SIZEOF_CHAR	760
13.57.1.31	__FFLASFFPACK_SIZEOF_INT	760
13.57.1.32	__FFLASFFPACK_SIZEOF_LONG	760
13.57.1.33	__FFLASFFPACK_SIZEOF_LONG_LONG	760
13.57.1.34	__FFLASFFPACK_SIZEOF_SHORT	760
13.57.1.35	__FFLASFFPACK_SIZEOF__INT64	760
13.57.1.36	__FFLASFFPACK_STDC_HEADERS	760
13.57.1.37	__FFLASFFPACK_USE_OPENMP	760
13.57.1.38	__FFLASFFPACK_VERSION	761
13.58	fflas-ffpack-config.h File Reference	761
13.58.1	Detailed Description	761
13.58.2	Macro Definition Documentation	761
13.58.2.1	GCC_VERSION	761
13.59	fflas-ffpack-default-thresholds.h File Reference	761
13.59.1	Macro Definition Documentation	761
13.59.1.1	__FFLASFFPACK_WINOTHRESHOLD	761

13.59.1.2	__FFLASFFPACK_WINOTHRESHOLD_FLT	761
13.59.1.3	__FFLASFFPACK_WINOTHRESHOLD_BAL	761
13.59.1.4	__FFLASFFPACK_WINOTHRESHOLD_BAL_FLT	762
13.59.1.5	__FFLASFFPACK_PLUQ_THRESHOLD	762
13.59.1.6	__FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD	762
13.59.1.7	__FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD	762
13.59.1.8	__FFLASFFPACK_ARITHPROG_THRESHOLD	762
13.59.1.9	__FFLASFFPACK_FTRTRI_THRESHOLD	762
13.59.1.10	__FFLASFFPACK_FSYTRF_THRESHOLD	762
13.59.1.11	__FFLASFFPACK_FSYRK_THRESHOLD	762
13.60	fflas-ffpack-thresholds.h File Reference	762
13.61	fflas-ffpack.doxy File Reference	762
13.62	fflas-ffpack.h File Reference	762
13.62.1	Detailed Description	762
13.63	fflas.doxy File Reference	762
13.64	fflas.h File Reference	762
13.64.1	Detailed Description	763
13.64.2	Macro Definition Documentation	763
13.64.2.1	WINOTHRESHOLD	763
13.64.2.2	DOUBLE_TO_FLOAT_CROSSOVER	764
13.65	fflas_bounds.inl File Reference	764
13.65.1	Macro Definition Documentation	764
13.65.1.1	__FFLASFFPACK_fflas_bounds_INL	764
13.65.1.2	FFLAS_INT_TYPE	764
13.66	fflas_enum.h File Reference	765
13.67	fflas_fadd.h File Reference	765
13.68	fflas_fadd.inl File Reference	767
13.68.1	Macro Definition Documentation	768
13.68.1.1	__FFLASFFPACK_fadd_INL	768
13.69	fflas_fassign.h File Reference	768
13.70	fflas_fassign.inl File Reference	768
13.70.1	Macro Definition Documentation	768
13.70.1.1	__FFLASFFPACK_fassign_INL	768
13.71	fflas_faxpy.inl File Reference	769
13.71.1	Macro Definition Documentation	769
13.71.1.1	__FFLASFFPACK_faxpy_INL	769
13.72	fflas_fdot.inl File Reference	769
13.72.1	Macro Definition Documentation	770
13.72.1.1	__FFLASFFPACK_fdot_INL	770
13.73	fflas_fgemm.inl File Reference	770
13.73.1	Macro Definition Documentation	772
13.73.1.1	__FFLASFFPACK_fgemm_INL	772

13.74 fgemm_classical.inl File Reference	772
13.75 fgemm_classical_mp.inl File Reference	772
13.75.1 Detailed Description	774
13.75.2 Macro Definition Documentation	774
13.75.2.1 __FFPACK_fgemm_classical_INL	774
13.76 fgemm_winograd.inl File Reference	774
13.76.1 Macro Definition Documentation	775
13.76.1.1 __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL	775
13.76.1.2 NEWWINO	775
13.77 matmul.doxy File Reference	775
13.78 schedule_bini.inl File Reference	775
13.78.1 Detailed Description	776
13.78.2 Macro Definition Documentation	776
13.78.2.1 __FFLASFFPACK_fgemm_bini_INL	776
13.79 schedule_winograd.inl File Reference	776
13.79.1 Macro Definition Documentation	776
13.79.1.1 __FFLASFFPACK_fgemm_winograd_INL	776
13.80 schedule_winograd_acc.inl File Reference	776
13.80.1 Macro Definition Documentation	777
13.80.1.1 __FFLASFFPACK_fgemm_winograd_acc_INL	777
13.81 schedule_winograd_acc_ip.inl File Reference	777
13.81.1 Macro Definition Documentation	778
13.81.1.1 __FFLASFFPACK_fgemm_winograd_acc_ip_INL	778
13.82 schedule_winograd_ip.inl File Reference	778
13.82.1 Macro Definition Documentation	778
13.82.1.1 __FFLASFFPACK_fgemm_winograd_ip_INL	778
13.83 fflas_fgemv.inl File Reference	779
13.83.1 Macro Definition Documentation	780
13.83.1.1 __FFLASFFPACK_fgemv_INL	780
13.84 fflas_fgemv_mp.inl File Reference	780
13.84.1 Macro Definition Documentation	781
13.84.1.1 __FFLASFFPACK_fgemv_mp_INL	781
13.85 fflas_fger.inl File Reference	781
13.85.1 Macro Definition Documentation	782
13.85.1.1 __FFLASFFPACK_fger_INL	782
13.86 fflas_fger_mp.inl File Reference	782
13.86.1 Macro Definition Documentation	783
13.86.1.1 __FFPACK_fger_mp_INL	783
13.87 fflas_freduce.h File Reference	783
13.88 fflas_freduce.inl File Reference	784
13.88.1 Macro Definition Documentation	785
13.88.1.1 __FFLASFFPACK_fflas_freduce_INL	785

13.88.1.2 FFLASFFPACK_COPY_REDUCE	785
13.89 fflas_freduce_mp.inl File Reference	785
13.89.1 Macro Definition Documentation	786
13.89.1.1 __FFLASFFPACK_fflas_freduce_mp_INL	786
13.90 fflas_freivalds.inl File Reference	786
13.90.1 Macro Definition Documentation	786
13.90.1.1 __FFLASFFPACK_freivalds_INL	786
13.91 fflas_fscal.h File Reference	786
13.92 fflas_fscal.inl File Reference	786
13.92.1 Macro Definition Documentation	788
13.92.1.1 __FFLASFFPACK_fscal_INL	788
13.93 fflas_fscal_mp.inl File Reference	788
13.93.1 Macro Definition Documentation	788
13.93.1.1 __FFLASFFPACK_fscal_mp_INL	788
13.94 fflas_fsyr2k.inl File Reference	788
13.94.1 Macro Definition Documentation	789
13.94.1.1 __FFLASFFPACK_fflas_fsyr2k_INL	789
13.95 fflas_fsyrk.inl File Reference	789
13.95.1 Macro Definition Documentation	790
13.95.1.1 __FFLASFFPACK_fflas_fsyrk_INL	790
13.96 fflas_ftrmm.inl File Reference	790
13.96.1 Macro Definition Documentation	790
13.96.1.1 __FFLASFFPACK_ftrmm_INL	790
13.97 fflas_ftrsm.inl File Reference	790
13.97.1 Macro Definition Documentation	791
13.97.1.1 __FFLASFFPACK_ftrsm_INL	791
13.98 fflas_ftrsm_mp.inl File Reference	791
13.98.1 Detailed Description	791
13.98.2 Macro Definition Documentation	791
13.98.2.1 __FFPACK_ftrsm_mp_INL	791
13.99 fflas_ftrsv.inl File Reference	792
13.99.1 Macro Definition Documentation	792
13.99.1.1 __FFLASFFPACK_ftrsv_INL	792
13.100 fflas_helpers.inl File Reference	792
13.100.1 Macro Definition Documentation	793
13.100.1.1 __FFLASFFPACK_fflas_fflas_mmhelper_INL	793
13.101 igemm.doxy File Reference	793
13.102 igemm.h File Reference	793
13.103 igemm.inl File Reference	794
13.103.1 Macro Definition Documentation	794
13.103.1.1 __FFLASFFPACK_fflas_igemm_igemm_INL	794
13.104 igemm_kernels.h File Reference	794

13.105 igemm_kernels.inl File Reference	795
13.105.1 Macro Definition Documentation	796
13.105.1.1 __FFLASFFPACK_fflas_igemm_igemm_kernels_INL	796
13.106 igemm_tools.h File Reference	796
13.107 igemm_tools.inl File Reference	796
13.107.1 Macro Definition Documentation	796
13.107.1.1 __FFLASFFPACK_fflas_igemm_igemm_tools_INL	796
13.108 fflas_level1.inl File Reference	796
13.108.1 Macro Definition Documentation	799
13.108.1.1 __FFLASFFPACK_fflas_fflas_level1_INL	799
13.109 fflas_level2.inl File Reference	799
13.109.1 Macro Definition Documentation	801
13.109.1.1 __FFLASFFPACK_fflas_fflas_level2_INL	801
13.110 fflas_level3.inl File Reference	801
13.110.1 Macro Definition Documentation	803
13.110.1.1 __FFLASFFPACK_fflas_fflas_level3_INL	803
13.110.1.2 __FFLAS__TRSM_READONLY	803
13.111 fflas_pfgemm.inl File Reference	804
13.111.1 Macro Definition Documentation	804
13.111.1.1 __FFLASFFPACK_fflas_pfgemm_INL	804
13.111.1.2 __FFLASFFPACK_SEQPARTHRESHOLD	804
13.111.1.3 __FFLASFFPACK_DIMKPENALTY	804
13.112 fflas_pftrsm.inl File Reference	804
13.112.1 Macro Definition Documentation	805
13.112.1.1 __FFLASFFPACK_fflas_pftrsm_INL	805
13.112.1.2 PTRSM_HYBRID_THRESHOLD	805
13.113 fflas_simd.h File Reference	805
13.113.1 Macro Definition Documentation	806
13.113.1.1 SIMD_INT	806
13.113.1.2 INLINE	806
13.113.1.3 CONST	806
13.113.1.4 PURE	806
13.113.1.5 NORML_MOD	806
13.113.1.6 FLOAT_MOD	806
13.113.2 Typedef Documentation	807
13.113.2.1 Simd	807
13.114 simd.doxy File Reference	807
13.115 simd128.inl File Reference	807
13.115.1 Macro Definition Documentation	807
13.115.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_INL	807
13.115.2 Typedef Documentation	807
13.115.2.1 Simd128	807

13.116 simd128_double.inl File Reference	807
13.116.1 Macro Definition Documentation	807
13.116.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL	807
13.117 simd128_float.inl File Reference	808
13.117.1 Macro Definition Documentation	808
13.117.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL	808
13.118 simd128_int16.inl File Reference	808
13.118.1 Macro Definition Documentation	808
13.118.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL	808
13.119 simd128_int32.inl File Reference	808
13.119.1 Macro Definition Documentation	808
13.119.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL	808
13.120 simd128_int64.inl File Reference	809
13.120.1 Macro Definition Documentation	809
13.120.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL	809
13.120.1.2 vect_t	809
13.121 simd256.inl File Reference	809
13.121.1 Macro Definition Documentation	809
13.121.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_INL	809
13.121.2 Typedef Documentation	809
13.121.2.1 Simd256	809
13.122 simd256_double.inl File Reference	810
13.122.1 Macro Definition Documentation	810
13.122.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL	810
13.123 simd256_float.inl File Reference	810
13.123.1 Macro Definition Documentation	810
13.123.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL	810
13.124 simd256_int16.inl File Reference	810
13.124.1 Macro Definition Documentation	810
13.124.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL	810
13.125 simd256_int32.inl File Reference	810
13.125.1 Macro Definition Documentation	811
13.125.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL	811
13.126 simd256_int64.inl File Reference	811
13.126.1 Macro Definition Documentation	811
13.126.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL	811
13.126.1.2 vect_t	811
13.127 simd512.inl File Reference	811
13.127.1 Macro Definition Documentation	812
13.127.1.1 __FFLASFFPACK_simd512_INL	812
13.127.2 Typedef Documentation	812
13.127.2.1 Simd512	812

13.128 simd512_double.inl File Reference	812
13.128.1 Macro Definition Documentation	812
13.128.1.1 __FFLASFFPACK_simd512_double_INL	812
13.129 simd512_float.inl File Reference	812
13.129.1 Macro Definition Documentation	812
13.129.1.1 __FFLASFFPACK_simd512_float_INL	812
13.130 simd512_int32.inl File Reference	812
13.130.1 Macro Definition Documentation	813
13.130.1.1 __FFLASFFPACK_simd512_int32_INL	813
13.131 simd512_int64.inl File Reference	813
13.131.1 Macro Definition Documentation	813
13.131.1.1 _simd512_int64_INL	813
13.131.1.2 vect_t	813
13.132 simd_modular.inl File Reference	813
13.133 fflas_sparse.h File Reference	813
13.133.1 Macro Definition Documentation	817
13.133.1.1 index_t	817
13.133.1.2 ROUND_DOWN	817
13.133.1.3 __FFLASFFPACK_CACHE_LINE_SIZE	817
13.133.1.4 assume_aligned	817
13.133.1.5 DENSE_THRESHOLD	817
13.134 fflas_sparse.inl File Reference	818
13.134.1 Macro Definition Documentation	819
13.134.1.1 __FFLASFFPACK_fflas_fflas_sparse_INL	819
13.135 coo.h File Reference	820
13.136 coo_spm.inl File Reference	820
13.136.1 Macro Definition Documentation	821
13.136.1.1 __FFLASFFPACK_fflas_sparse_coo_spm_INL	821
13.137 coo_spmv.inl File Reference	821
13.137.1 Macro Definition Documentation	822
13.137.1.1 __FFLASFFPACK_fflas_sparse_coo_spmv_INL	822
13.138 coo_utils.inl File Reference	822
13.138.1 Macro Definition Documentation	822
13.138.1.1 __FFLASFFPACK_fflas_sparse_coo_utils_INL	822
13.139 csr.h File Reference	822
13.140 csr_pspm.inl File Reference	823
13.140.1 Macro Definition Documentation	824
13.140.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspm_INL	824
13.141 csr_pspmv.inl File Reference	824
13.141.1 Macro Definition Documentation	824
13.141.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL	824
13.142 csr_spm.inl File Reference	824

13.142.1 Macro Definition Documentation	825
13.142.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmv_INL	825
13.143 csr_spmv.inl File Reference	826
13.143.1 Macro Definition Documentation	826
13.143.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmv_INL	826
13.144 csr_utils.inl File Reference	826
13.145 csr_hyb.h File Reference	827
13.146 csr_hyb_pspmm.inl File Reference	827
13.146.1 Macro Definition Documentation	828
13.146.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL	828
13.147 csr_hyb_pspmv.inl File Reference	828
13.147.1 Macro Definition Documentation	828
13.147.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL	828
13.148 csr_hyb_spmv.inl File Reference	829
13.148.1 Macro Definition Documentation	829
13.148.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL	829
13.149 csr_hyb_spmv.inl File Reference	829
13.149.1 Macro Definition Documentation	829
13.149.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL	829
13.150 csr_hyb_utils.inl File Reference	830
13.150.1 Macro Definition Documentation	830
13.150.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL	830
13.151 ell.h File Reference	830
13.152 ell_pspmm.inl File Reference	831
13.152.1 Macro Definition Documentation	831
13.152.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL	831
13.153 ell_pspmv.inl File Reference	831
13.153.1 Macro Definition Documentation	832
13.153.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL	832
13.154 ell_spmv.inl File Reference	832
13.154.1 Macro Definition Documentation	833
13.154.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmv_INL	833
13.155 ell_spmv.inl File Reference	833
13.155.1 Macro Definition Documentation	834
13.155.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmv_INL	834
13.156 ell_utils.inl File Reference	834
13.156.1 Macro Definition Documentation	834
13.156.1.1 __FFLASFFPACK_fflas_sparse_ELL_utils_INL	834
13.157 ell_simd.h File Reference	834
13.158 ell_simd_pspmv.inl File Reference	835
13.158.1 Macro Definition Documentation	836
13.158.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL	836

13.159 ell_simd_spmv.inl File Reference	836
13.159.1 Macro Definition Documentation	836
13.159.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL	836
13.160 ell_simd_utils.inl File Reference	837
13.160.1 Macro Definition Documentation	837
13.160.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL	837
13.161 hyb_zo.h File Reference	837
13.162 hyb_zo_pspmm.inl File Reference	837
13.162.1 Macro Definition Documentation	838
13.162.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL	838
13.163 hyb_zo_pspmv.inl File Reference	838
13.163.1 Macro Definition Documentation	838
13.163.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL	838
13.164 hyb_zo_spm্ম.inl File Reference	838
13.164.1 Macro Definition Documentation	839
13.164.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spm্ম_INL	839
13.165 hyb_zo_spmv.inl File Reference	839
13.165.1 Macro Definition Documentation	839
13.165.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL	839
13.166 hyb_zo_utils.inl File Reference	839
13.166.1 Macro Definition Documentation	840
13.166.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL	840
13.167 read_sparse.h File Reference	840
13.167.1 Macro Definition Documentation	841
13.167.1.1 DNS_BIN_VER	841
13.167.1.2 mask_t	841
13.168 sell.h File Reference	841
13.169 sell_pspmv.inl File Reference	841
13.169.1 Macro Definition Documentation	842
13.169.1.1 __FFLASFFPACK_fflas_sparse_sell_pspmv_INL	842
13.170 sell_spmv.inl File Reference	842
13.170.1 Macro Definition Documentation	843
13.170.1.1 __FFLASFFPACK_fflas_sparse_sell_spmv_INL	843
13.171 sell_utils.inl File Reference	843
13.171.1 Macro Definition Documentation	843
13.171.1.1 __FFLASFFPACK_fflas_sparse_sell_utils_INL	843
13.172 sparse_matrix_traits.h File Reference	844
13.173 utils.h File Reference	845
13.174 ffpack.dox File Reference	845
13.175 ffpack.h File Reference	845
13.175.1 Detailed Description	853
13.175.2 Macro Definition Documentation	853

13.175.2.1	__FFLASFFPACK_FTRSTR_THRESHOLD	853
13.175.2.2	__FFLASFFPACK_FTRSSYR2K_THRESHOLD	854
13.176	ffpack.inl File Reference	854
13.176.1	Macro Definition Documentation	855
13.176.1.1	__FFLASFFPACK_ffpack_INL	855
13.177	ffpack_charpoly.inl File Reference	855
13.177.1	Macro Definition Documentation	856
13.177.1.1	__FFLASFFPACK_charpoly_INL	856
13.178	ffpack_charpoly_danilevski.inl File Reference	856
13.178.1	Macro Definition Documentation	856
13.178.1.1	__FFLASFFPACK_ffpack_charpoly_danilveski_INL	856
13.179	ffpack_charpoly_kgfast.inl File Reference	856
13.179.1	Macro Definition Documentation	856
13.179.1.1	__FFLASFFPACK_ffpack_charpoly_kgfast_INL	856
13.180	ffpack_charpoly_kgfastgeneralized.inl File Reference	857
13.180.1	Macro Definition Documentation	857
13.180.1.1	__FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL	857
13.181	ffpack_charpoly_kglu.inl File Reference	857
13.181.1	Macro Definition Documentation	858
13.181.1.1	__FFLASFFPACK_ffpack_charpoly_kglu_INL	858
13.182	ffpack_charpoly_mp.inl File Reference	858
13.182.1	Macro Definition Documentation	858
13.182.1.1	__FFPACK_charpoly_mp_INL	858
13.183	ffpack_det_mp.inl File Reference	858
13.183.1	Macro Definition Documentation	859
13.183.1.1	__FFPACK_det_mp_INL	859
13.184	ffpack_echelonforms.inl File Reference	859
13.184.1	Macro Definition Documentation	860
13.184.1.1	__FFLASFFPACK_ffpack_echelon_forms_INL	860
13.184.1.2	__FFLASFFPACK_GAUSSJORDAN_BASECASE	860
13.185	ffpack_fgesv.inl File Reference	860
13.185.1	Macro Definition Documentation	861
13.185.1.1	__FFLASFFPACK_ffpack_fgesv_INL	861
13.186	ffpack_fgetrs.inl File Reference	861
13.186.1	Macro Definition Documentation	861
13.186.1.1	__FFLASFFPACK_ffpack_fgetrs_INL	861
13.187	ffpack_frobenius.inl File Reference	861
13.188	ffpack_fsytrf.inl File Reference	862
13.188.1	Macro Definition Documentation	863
13.188.1.1	__FFLASFFPACK_ffpack_fsytrf_INL	863
13.189	ffpack_ftrssyr2k.inl File Reference	863
13.189.1	Macro Definition Documentation	863

13.189.1.1 __FFLASFFPACK_ffpack_ftrssyr2k_INL	863
13.190 ffpack_ftrstr.inl File Reference	864
13.190.1 Macro Definition Documentation	864
13.190.1.1 __FFLASFFPACK_ffpack_ftrstr_INL	864
13.191 ffpack_ftrtr.inl File Reference	864
13.191.1 Macro Definition Documentation	865
13.191.1.1 ENABLE_ALL_CHECKINGS	865
13.191.1.2 __FFLASFFPACK_ffpack_ftrtr_INL	865
13.192 ffpack_invert.inl File Reference	865
13.192.1 Macro Definition Documentation	865
13.192.1.1 __FFLASFFPACK_ffpack_invert_INL	865
13.193 ffpack_krylovelim.inl File Reference	865
13.193.1 Macro Definition Documentation	865
13.193.1.1 __FFLASFFPACK_ffpack_krylovelim_INL	865
13.194 ffpack_ludivine.inl File Reference	865
13.194.1 Macro Definition Documentation	866
13.194.1.1 __FFLASFFPACK_ffpack_ludivine_INL	866
13.195 ffpack_ludivine_mp.inl File Reference	866
13.195.1 Macro Definition Documentation	867
13.195.1.1 __FFPACK_ludivine_mp_INL	867
13.196 ffpack_minpoly.inl File Reference	867
13.196.1 Macro Definition Documentation	867
13.196.1.1 __FFLASFFPACK_ffpack_minpoly_INL	867
13.197 ffpack_permutation.inl File Reference	868
13.197.1 Macro Definition Documentation	870
13.197.1.1 __FFLASFFPACK_ffpack_permutation_INL	870
13.197.1.2 FFLASFFPACK_PERM_BKSIZE	870
13.198 ffpack_pluq.inl File Reference	870
13.198.1 Macro Definition Documentation	870
13.198.1.1 __FFLASFFPACK_ffpack_pluq_INL	870
13.198.1.2 CROUT	871
13.199 ffpack_pluq_mp.inl File Reference	871
13.199.1 Macro Definition Documentation	871
13.199.1.1 __FFPACK_pluq_mp_INL	871
13.200 ffpack_ppluq.inl File Reference	871
13.200.1 Macro Definition Documentation	872
13.200.1.1 __FFLASFFPACK_ffpack_ppluq_INL	872
13.200.1.2 FFLAS__TRSM_READONLY	872
13.200.1.3 PBASECASE_K	872
13.201 ffpack_rankprofiles.inl File Reference	872
13.201.1 Macro Definition Documentation	873
13.201.1.1 __FFLASFFPACK_ffpack_rank_profiles_INL	873

13.202 field-traits.h File Reference	873
13.202.1 Detailed Description	875
13.203 field.doxy File Reference	875
13.204 rns-double-elt.h File Reference	875
13.204.1 Detailed Description	876
13.205 rns-double-recint.inl File Reference	876
13.205.1 Macro Definition Documentation	876
13.205.1.1 __FFLASFFPACK_field_rns_double_recint_INL	876
13.206 rns-double.h File Reference	876
13.206.1 Detailed Description	877
13.206.2 Macro Definition Documentation	877
13.206.2.1 ROUND_DOWN	877
13.207 rns-double.inl File Reference	877
13.207.1 Macro Definition Documentation	877
13.207.1.1 __FFLASFFPACK_field_rns_double_INL	877
13.208 rns-integer-mod.h File Reference	877
13.208.1 Detailed Description	878
13.209 rns-integer.h File Reference	878
13.209.1 Detailed Description	879
13.210 rns.h File Reference	879
13.211 rns.inl File Reference	879
13.211.1 Macro Definition Documentation	879
13.211.1.1 __FFLASFFPACK_field_rns_INL	879
13.212 interfaces.doxy File Reference	879
13.213 fflas_c.h File Reference	879
13.213.1 Macro Definition Documentation	881
13.213.1.1 FFLAS_COMPILED	881
13.213.2 Enumeration Type Documentation	881
13.213.2.1 FFLAS_C_ORDER	881
13.213.2.2 FFLAS_C_TRANSPOSE	882
13.213.2.3 FFLAS_C_UPLO	882
13.213.2.4 FFLAS_C_DIAG	882
13.213.2.5 FFLAS_C_SIDE	882
13.213.2.6 FFLAS_C_BASE	882
13.213.3 Function Documentation	883
13.213.3.1 freducein_1_modular_double()	883
13.213.3.2 freduce_1_modular_double()	883
13.213.3.3 fnegin_1_modular_double()	883
13.213.3.4 fneg_1_modular_double()	883
13.213.3.5 fzero_1_modular_double()	883
13.213.3.6 fiszero_1_modular_double()	883
13.213.3.7 fequal_1_modular_double()	884

13.213.3.8 fassign_1_modular_double()	884
13.213.3.9 fscaln_1_modular_double()	884
13.213.3.10 fscal_1_modular_double()	884
13.213.3.11 faxpy_1_modular_double()	884
13.213.3.12 fdot_1_modular_double()	884
13.213.3.13 fswap_1_modular_double()	885
13.213.3.14 fadd_1_modular_double()	885
13.213.3.15 fsub_1_modular_double()	885
13.213.3.16 faddin_1_modular_double()	885
13.213.3.17 fsubin_1_modular_double()	885
13.213.3.18 fassign_2_modular_double()	886
13.213.3.19 fzero_2_modular_double()	886
13.213.3.20 fequal_2_modular_double()	886
13.213.3.21 fiszero_2_modular_double()	886
13.213.3.22 fidentity_2_modular_double()	886
13.213.3.23 freducein_2_modular_double()	887
13.213.3.24 freduce_2_modular_double()	887
13.213.3.25 fnegin_2_modular_double()	887
13.213.3.26 fneg_2_modular_double()	887
13.213.3.27 fscaln_2_modular_double()	887
13.213.3.28 fscal_2_modular_double()	888
13.213.3.29 faxpy_2_modular_double()	888
13.213.3.30 fmove_2_modular_double()	888
13.213.3.31 fadd_2_modular_double()	888
13.213.3.32 fsub_2_modular_double()	888
13.213.3.33 fsubin_2_modular_double()	889
13.213.3.34 faddin_2_modular_double()	889
13.213.3.35 fgemv_2_modular_double()	889
13.213.3.36 fger_2_modular_double()	889
13.213.3.37 ftrsv_2_modular_double()	890
13.213.3.38 ftrsm_3_modular_double()	890
13.213.3.39 ftrmm_3_modular_double()	890
13.213.3.40 fgemm_3_modular_double()	890
13.213.3.41 fsquare_3_modular_double()	891
13.214 fflas_L1_inst.C File Reference	891
13.214.1 Macro Definition Documentation	891
13.214.1.1 __FFLAS_L1_INST_C	891
13.214.1.2 INST_OR_DECL	891
13.214.1.3 FFLAS_FIELD [1/2]	892
13.214.1.4 FFLAS_ELT [1/6]	892
13.214.1.5 FFLAS_ELT [2/6]	892
13.214.1.6 FFLAS_ELT [3/6]	892

13.214.1.7 FFLAS_FIELD [2/2]	892
13.214.1.8 FFLAS_ELT [4/6]	892
13.214.1.9 FFLAS_ELT [5/6]	892
13.214.1.10 FFLAS_ELT [6/6]	892
13.215 fflas_L1_inst.h File Reference	892
13.215.1 Macro Definition Documentation	892
13.215.1.1 INST_OR_DECL	892
13.215.1.2 FFLAS_FIELD [1/2]	893
13.215.1.3 FFLAS_ELT [1/6]	893
13.215.1.4 FFLAS_ELT [2/6]	893
13.215.1.5 FFLAS_ELT [3/6]	893
13.215.1.6 FFLAS_FIELD [2/2]	893
13.215.1.7 FFLAS_ELT [4/6]	893
13.215.1.8 FFLAS_ELT [5/6]	893
13.215.1.9 FFLAS_ELT [6/6]	893
13.216 fflas_L1_inst_implem.inl File Reference	893
13.217 fflas_L2_inst.C File Reference	894
13.217.1 Macro Definition Documentation	895
13.217.1.1 __FFLAS_L2_INST_C	895
13.217.1.2 INST_OR_DECL	895
13.217.1.3 FFLAS_FIELD [1/2]	895
13.217.1.4 FFLAS_ELT [1/6]	895
13.217.1.5 FFLAS_ELT [2/6]	895
13.217.1.6 FFLAS_ELT [3/6]	895
13.217.1.7 FFLAS_FIELD [2/2]	895
13.217.1.8 FFLAS_ELT [4/6]	895
13.217.1.9 FFLAS_ELT [5/6]	895
13.217.1.10 FFLAS_ELT [6/6]	895
13.218 fflas_L2_inst.h File Reference	895
13.218.1 Macro Definition Documentation	896
13.218.1.1 INST_OR_DECL	896
13.218.1.2 FFLAS_FIELD [1/2]	896
13.218.1.3 FFLAS_ELT [1/6]	896
13.218.1.4 FFLAS_ELT [2/6]	896
13.218.1.5 FFLAS_ELT [3/6]	896
13.218.1.6 FFLAS_FIELD [2/2]	896
13.218.1.7 FFLAS_ELT [4/6]	896
13.218.1.8 FFLAS_ELT [5/6]	896
13.218.1.9 FFLAS_ELT [6/6]	896
13.219 fflas_L2_inst_implem.inl File Reference	896
13.220 fflas_L3_inst.C File Reference	898
13.220.1 Macro Definition Documentation	898

13.220.1.1	__FFLAS_L3_INST_C	898
13.220.1.2	INST_OR_DECL	899
13.220.1.3	FFLAS_FIELD [1/2]	899
13.220.1.4	FFLAS_ELT [1/6]	899
13.220.1.5	FFLAS_ELT [2/6]	899
13.220.1.6	FFLAS_ELT [3/6]	899
13.220.1.7	FFLAS_FIELD [2/2]	899
13.220.1.8	FFLAS_ELT [4/6]	899
13.220.1.9	FFLAS_ELT [5/6]	899
13.220.1.10	FFLAS_ELT [6/6]	899
13.221	fflas_L3_inst.h File Reference	899
13.221.1	Macro Definition Documentation	900
13.221.1.1	INST_OR_DECL	900
13.221.1.2	FFLAS_FIELD [1/2]	900
13.221.1.3	FFLAS_ELT [1/6]	900
13.221.1.4	FFLAS_ELT [2/6]	900
13.221.1.5	FFLAS_ELT [3/6]	900
13.221.1.6	FFLAS_FIELD [2/2]	900
13.221.1.7	FFLAS_ELT [4/6]	900
13.221.1.8	FFLAS_ELT [5/6]	900
13.221.1.9	FFLAS_ELT [6/6]	900
13.222	fflas_L3_inst_implem.inl File Reference	900
13.222.1	Macro Definition Documentation	901
13.222.1.1	__FFLAS__TRSM_READONLY	901
13.223	fflas_lvl1.C File Reference	901
13.223.1	Detailed Description	902
13.223.2	Function Documentation	902
13.223.2.1	freducein_1_modular_double()	902
13.223.2.2	freduce_1_modular_double()	902
13.223.2.3	fnegin_1_modular_double()	902
13.223.2.4	fneg_1_modular_double()	903
13.223.2.5	fzero_1_modular_double()	903
13.223.2.6	fiszero_1_modular_double()	903
13.223.2.7	fequal_1_modular_double()	903
13.223.2.8	fassign_1_modular_double()	903
13.223.2.9	fscaln_1_modular_double()	903
13.223.2.10	fscal_1_modular_double()	904
13.223.2.11	faxpy_1_modular_double()	904
13.223.2.12	fdot_1_modular_double()	904
13.223.2.13	fswap_1_modular_double()	904
13.223.2.14	fadd_1_modular_double()	904
13.223.2.15	fsub_1_modular_double()	905

13.223.2.16 faddin_1_modular_double()	905
13.223.2.17 fsubin_1_modular_double()	905
13.224 fflas_lvl2.C File Reference	905
13.224.1 Detailed Description	906
13.224.2 Function Documentation	906
13.224.2.1 fassign_2_modular_double()	906
13.224.2.2 fzero_2_modular_double()	907
13.224.2.3 fequal_2_modular_double()	907
13.224.2.4 fiszero_2_modular_double()	907
13.224.2.5 fidentity_2_modular_double()	907
13.224.2.6 freducein_2_modular_double()	907
13.224.2.7 freduce_2_modular_double()	907
13.224.2.8 fnegin_2_modular_double()	908
13.224.2.9 fneg_2_modular_double()	908
13.224.2.10 fscaln_2_modular_double()	908
13.224.2.11 fscal_2_modular_double()	908
13.224.2.12 faxpy_2_modular_double()	908
13.224.2.13 fmove_2_modular_double()	909
13.224.2.14 fadd_2_modular_double()	909
13.224.2.15 fsub_2_modular_double()	909
13.224.2.16 fsubin_2_modular_double()	909
13.224.2.17 faddin_2_modular_double()	910
13.224.2.18 fgemv_2_modular_double()	910
13.224.2.19 fger_2_modular_double()	910
13.224.2.20 ftrsv_2_modular_double()	910
13.225 fflas_lvl3.C File Reference	911
13.225.1 Detailed Description	911
13.225.2 Function Documentation	911
13.225.2.1 ftrsm_3_modular_double()	911
13.225.2.2 ftrmm_3_modular_double()	912
13.225.2.3 fgemm_3_modular_double()	912
13.225.2.4 fsquare_3_modular_double()	912
13.226 fflas_sparse.C File Reference	912
13.226.1 Detailed Description	912
13.227 ffpack.C File Reference	913
13.227.1 Detailed Description	916
13.227.2 Function Documentation	916
13.227.2.1 LAPACKPerm2MathPerm()	916
13.227.2.2 MathPerm2LAPACKPerm()	916
13.227.2.3 MatrixApplyS_modular_double()	916
13.227.2.4 PermApplyS_double()	917
13.227.2.5 MatrixApplyT_modular_double()	917

13.227.2.6 PermApplyT_double()	917
13.227.2.7 composePermutationsLLM()	917
13.227.2.8 composePermutationsLLL()	917
13.227.2.9 composePermutationsMLM()	918
13.227.2.10 cyclic_shift_mathPerm()	918
13.227.2.11 cyclic_shift_row_modular_double()	918
13.227.2.12 cyclic_shift_col_modular_double()	918
13.227.2.13 applyP_modular_double()	918
13.227.2.14 fgetrsin_modular_double()	918
13.227.2.15 fgetrsv_modular_double()	919
13.227.2.16 fgesvin_modular_double()	919
13.227.2.17 fgesv_modular_double()	919
13.227.2.18 ftrtri_modular_double()	920
13.227.2.19 trinv_left_modular_double()	920
13.227.2.20 ftrtrm_modular_double()	920
13.227.2.21 PLUQ_modular_double()	920
13.227.2.22 LUdivine_modular_double()	920
13.227.2.23 ColumnEchelonForm_modular_double()	921
13.227.2.24 RowEchelonForm_modular_double()	921
13.227.2.25 ReducedColumnEchelonForm_modular_double()	921
13.227.2.26 ReducedRowEchelonForm_modular_double()	921
13.227.2.27 ColumnEchelonForm_modular_float()	922
13.227.2.28 RowEchelonForm_modular_float()	922
13.227.2.29 ReducedColumnEchelonForm_modular_float()	922
13.227.2.30 ReducedRowEchelonForm_modular_float()	922
13.227.2.31 ColumnEchelonForm_modular_int32_t()	922
13.227.2.32 RowEchelonForm_modular_int32_t()	923
13.227.2.33 ReducedColumnEchelonForm_modular_int32_t()	923
13.227.2.34 ReducedRowEchelonForm_modular_int32_t()	923
13.227.2.35 pColumnEchelonForm_modular_double()	923
13.227.2.36 pRowEchelonForm_modular_double()	924
13.227.2.37 pReducedColumnEchelonForm_modular_double()	924
13.227.2.38 pReducedRowEchelonForm_modular_double()	924
13.227.2.39 pColumnEchelonForm_modular_float()	924
13.227.2.40 pRowEchelonForm_modular_float()	925
13.227.2.41 pReducedColumnEchelonForm_modular_float()	925
13.227.2.42 pReducedRowEchelonForm_modular_float()	925
13.227.2.43 pColumnEchelonForm_modular_int32_t()	925
13.227.2.44 pRowEchelonForm_modular_int32_t()	926
13.227.2.45 pReducedColumnEchelonForm_modular_int32_t()	926
13.227.2.46 pReducedRowEchelonForm_modular_int32_t()	926
13.227.2.47 Invertin_modular_double()	926

13.227.2.48	Invert_modular_double()	926
13.227.2.49	Invert2_modular_double()	927
13.227.2.50	KrylovElim_modular_double()	927
13.227.2.51	SpecRankProfile_modular_double()	927
13.227.2.52	Rank_modular_double()	927
13.227.2.53	IsSingular_modular_double()	927
13.227.2.54	Det_modular_double()	928
13.227.2.55	Solve_modular_double()	928
13.227.2.56	solveLB_modular_double()	928
13.227.2.57	solveLB2_modular_double()	928
13.227.2.58	RandomNullSpaceVector_modular_double()	929
13.227.2.59	NullSpaceBasis_modular_double()	929
13.227.2.60	RowRankProfile_modular_double()	929
13.227.2.61	ColumnRankProfile_modular_double()	929
13.227.2.62	RankProfileFromLU()	929
13.227.2.63	LeadingSubmatrixRankProfiles()	930
13.227.2.64	RowRankProfileSubmatrixIndices_modular_double()	930
13.227.2.65	ColRankProfileSubmatrixIndices_modular_double()	930
13.227.2.66	RowRankProfileSubmatrix_modular_double()	930
13.227.2.67	ColRankProfileSubmatrix_modular_double()	930
13.227.2.68	getTriangular_modular_double()	931
13.227.2.69	getTriangularin_modular_double()	931
13.227.2.70	getEchelonForm_modular_double()	931
13.227.2.71	getEchelonFormin_modular_double()	931
13.227.2.72	getEchelonTransform_modular_double()	932
13.227.2.73	getReducedEchelonForm_modular_double()	932
13.227.2.74	getReducedEchelonFormin_modular_double()	932
13.227.2.75	getReducedEchelonTransform_modular_double()	932
13.227.2.76	PLUQtoEchelonPermutation()	933
13.228	ffpack_c.h File Reference	933
13.228.1	Macro Definition Documentation	936
13.228.1.1	FFPACK_COMPILED	936
13.228.2	Enumeration Type Documentation	936
13.228.2.1	FFLAS_C_ORDER	936
13.228.2.2	FFLAS_C_TRANSPOSE	936
13.228.2.3	FFLAS_C_UPLO	936
13.228.2.4	FFLAS_C_DIAG	937
13.228.2.5	FFLAS_C_SIDE	937
13.228.2.6	FFPACK_C_LU_TAG	937
13.228.2.7	FFPACK_C_CHARPOLY_TAG	937
13.228.2.8	FFPACK_C_MINPOLY_TAG	937
13.228.3	Function Documentation	938

13.228.3.1 LAPACKPerm2MathPerm()	938
13.228.3.2 MathPerm2LAPACKPerm()	938
13.228.3.3 MatrixApplyS_modular_double()	938
13.228.3.4 PermApplyS_double()	938
13.228.3.5 MatrixApplyT_modular_double()	938
13.228.3.6 PermApplyT_double()	939
13.228.3.7 composePermutationsLLM()	939
13.228.3.8 composePermutationsLLL()	939
13.228.3.9 composePermutationsMLM()	939
13.228.3.10 cyclic_shift_mathPerm()	939
13.228.3.11 cyclic_shift_row_modular_double()	939
13.228.3.12 cyclic_shift_col_modular_double()	939
13.228.3.13 applyP_modular_double()	940
13.228.3.14 fgetrsin_modular_double()	940
13.228.3.15 fgetrs_modular_double()	940
13.228.3.16 fgesvin_modular_double()	941
13.228.3.17 fgesv_modular_double()	941
13.228.3.18 ftrtri_modular_double()	941
13.228.3.19 trinv_left_modular_double()	941
13.228.3.20 ftrtrm_modular_double()	941
13.228.3.21 PLUQ_modular_double()	942
13.228.3.22 LUdivine_modular_double()	942
13.228.3.23 LUdivine_small_modular_double()	942
13.228.3.24 LUdivine_gauss_modular_double()	942
13.228.3.25 ColumnEchelonForm_modular_double()	943
13.228.3.26 RowEchelonForm_modular_double()	943
13.228.3.27 ColumnEchelonForm_modular_float()	943
13.228.3.28 RowEchelonForm_modular_float()	943
13.228.3.29 ColumnEchelonForm_modular_int32_t()	944
13.228.3.30 RowEchelonForm_modular_int32_t()	944
13.228.3.31 ReducedColumnEchelonForm_modular_double()	944
13.228.3.32 ReducedRowEchelonForm_modular_double()	944
13.228.3.33 ReducedColumnEchelonForm_modular_float()	944
13.228.3.34 ReducedRowEchelonForm_modular_float()	945
13.228.3.35 ReducedColumnEchelonForm_modular_int32_t()	945
13.228.3.36 ReducedRowEchelonForm_modular_int32_t()	945
13.228.3.37 ReducedRowEchelonForm2_modular_double()	945
13.228.3.38 REF_modular_double()	946
13.228.3.39 Invertin_modular_double()	946
13.228.3.40 Invert_modular_double()	946
13.228.3.41 Invert2_modular_double()	946
13.228.3.42 KrylovElim_modular_double()	946

13.228.3.43 SpecRankProfile_modular_double()	947
13.228.3.44 Rank_modular_double()	947
13.228.3.45 IsSingular_modular_double()	947
13.228.3.46 Det_modular_double()	947
13.228.3.47 Solve_modular_double()	947
13.228.3.48 solveLB_modular_double()	948
13.228.3.49 solveLB2_modular_double()	948
13.228.3.50 RandomNullSpaceVector_modular_double()	948
13.228.3.51 NullSpaceBasis_modular_double()	948
13.228.3.52 RowRankProfile_modular_double()	949
13.228.3.53 ColumnRankProfile_modular_double()	949
13.228.3.54 RankProfileFromLU()	949
13.228.3.55 LeadingSubmatrixRankProfiles()	949
13.228.3.56 RowRankProfileSubmatrixIndices_modular_double()	949
13.228.3.57 ColRankProfileSubmatrixIndices_modular_double()	950
13.228.3.58 RowRankProfileSubmatrix_modular_double()	950
13.228.3.59 ColRankProfileSubmatrix_modular_double()	950
13.228.3.60 getTriangular_modular_double()	950
13.228.3.61 getTriangularin_modular_double()	951
13.228.3.62 getEchelonForm_modular_double()	951
13.228.3.63 getEchelonFormin_modular_double()	951
13.228.3.64 getEchelonTransform_modular_double()	951
13.228.3.65 getReducedEchelonForm_modular_double()	952
13.228.3.66 getReducedEchelonFormin_modular_double()	952
13.228.3.67 getReducedEchelonTransform_modular_double()	952
13.228.3.68 PLUQtoEchelonPermutation()	952
13.229 fpack_inst.C File Reference	953
13.229.1 Macro Definition Documentation	953
13.229.1.1 __FFPACK_INST_C	953
13.229.1.2 FFLAS_COMPILED	953
13.229.1.3 INST_OR_DECL	953
13.229.1.4 FFLAS_FIELD [1/2]	953
13.229.1.5 FFLAS_ELT [1/6]	953
13.229.1.6 FFLAS_ELT [2/6]	953
13.229.1.7 FFLAS_ELT [3/6]	953
13.229.1.8 FFLAS_FIELD [2/2]	953
13.229.1.9 FFLAS_ELT [4/6]	954
13.229.1.10 FFLAS_ELT [5/6]	954
13.229.1.11 FFLAS_ELT [6/6]	954
13.230 fpack_inst.h File Reference	954
13.230.1 Macro Definition Documentation	954
13.230.1.1 FFLAS_COMPILED	954

13.230.1.2 INST_OR_DECL	954
13.230.1.3 FFLAS_FIELD [1/2]	954
13.230.1.4 FFLAS_ELT [1/6]	954
13.230.1.5 FFLAS_ELT [2/6]	954
13.230.1.6 FFLAS_ELT [3/6]	954
13.230.1.7 FFLAS_FIELD [2/2]	955
13.230.1.8 FFLAS_ELT [4/6]	955
13.230.1.9 FFLAS_ELT [5/6]	955
13.230.1.10 FFLAS_ELT [6/6]	955
13.231 fpack_inst_implement.inl File Reference	955
13.232 blockcuts.inl File Reference	958
13.232.1 Macro Definition Documentation	959
13.232.1.1 __FFLASFFPACK_fflas_blockcuts_INL	959
13.232.1.2 __FFLASFFPACK_MINBLOCKCUTS	959
13.233 fflas_plevel1.h File Reference	960
13.234 kaapi_routines.inl File Reference	960
13.234.1 Macro Definition Documentation	960
13.234.1.1 __FFLASFFPACK_KAAPI_ROUTINES_INL	960
13.235 parallel.h File Reference	960
13.235.1 Macro Definition Documentation	961
13.235.1.1 __FFLASFFPACK_SEQUENTIAL	961
13.235.1.2 index_t	961
13.235.1.3 TASK	961
13.235.1.4 WAIT	961
13.235.1.5 CHECK_DEPENDENCIES	961
13.235.1.6 BARRIER	961
13.235.1.7 PAR_BLOCK	962
13.235.1.8 SYNCH_GROUP	962
13.235.1.9 NUM_THREADS	962
13.235.1.10 MAX_THREADS	962
13.235.1.11 READ	962
13.235.1.12 WRITE	962
13.235.1.13 READWRITE	962
13.235.1.14 CONSTREFERENCE	962
13.235.1.15 VALUE	962
13.235.1.16 BEGIN_PARALLEL_MAIN	962
13.235.1.17 END_PARALLEL_MAIN	962
13.235.1.18 FORBLOCK1D	962
13.235.1.19 FOR1D	963
13.235.1.20 PARFORBLOCK1D	963
13.235.1.21 PARFOR1D	963
13.235.1.22 FORBLOCK2D	963

13.235.1.23 FOR2D	963
13.235.1.24 PARFORBLOCK2D	964
13.235.1.25 PARFOR2D	964
13.235.1.26 COMMA	964
13.235.1.27 MODE	964
13.235.1.28 RETURNPARAM	964
13.235.1.29 NUMARGS	964
13.235.1.30 PP_NARG_	964
13.235.1.31 PP_ARG_N	964
13.235.1.32 PP_RSEQ_N	966
13.235.1.33 NOSPLIT	966
13.235.1.34 splitting_0	966
13.235.1.35 splitting_1	966
13.235.1.36 splitting_2	966
13.235.1.37 splitting_3	966
13.235.1.38 splitt	966
13.235.1.39 SPLITTER	966
13.236 pfgemm_variants.inl File Reference	966
13.237 pfgemv.inl File Reference	967
13.238 align-allocator.h File Reference	968
13.239 args-parser.h File Reference	968
13.239.1 Macro Definition Documentation	968
13.239.1.1 TYPE_BOOL	968
13.239.1.2 END_OF_ARGUMENTS	969
13.239.1.3 type_integer	969
13.239.2 Enumeration Type Documentation	969
13.239.2.1 ArgumentType	969
13.239.3 Function Documentation	969
13.239.3.1 printHelpMessage()	969
13.239.3.2 findArgument()	969
13.239.3.3 getListArgs()	969
13.240 bit_manipulation.h File Reference	970
13.240.1 Macro Definition Documentation	970
13.240.1.1 __has_builtin	970
13.240.2 Function Documentation	970
13.240.2.1 clz() [1/2]	970
13.240.2.2 clz() [2/2]	970
13.240.2.3 ctz() [1/2]	970
13.240.2.4 ctz() [2/2]	970
13.241 cast.h File Reference	970
13.242 debug.h File Reference	971
13.242.1 Detailed Description	971

13.242.2 Macro Definition Documentation	971
13.242.2.1 FFLASFFPACK_check	971
13.242.2.2 FFLASFFPACK_abort	972
13.243 fflas_intrinsic.h File Reference	972
13.244 fflas_io.h File Reference	972
13.245 fflas_memory.h File Reference	973
13.246 fflas_randommatrix.h File Reference	973
13.247 flimits.h File Reference	975
13.247.1 Function Documentation	976
13.247.1.1 in_range() [1/3]	976
13.247.1.2 in_range() [2/3]	976
13.247.1.3 in_range() [3/3]	976
13.248 Matio.h File Reference	976
13.248.1 Function Documentation	977
13.248.1.1 read_field()	977
13.248.1.2 write_field()	977
13.249 test-utils.h File Reference	977
13.250 timer.h File Reference	978
13.251 cblas.C File Reference	978
13.251.1 Macro Definition Documentation	978
13.251.1.1 __FFLASFFPACK_CONFIGURATION	978
13.251.1.2 __FFLASFFPACK_HAVE_CBLAS	978
13.251.2 Function Documentation	978
13.251.2.1 main()	978
13.252 clapack.C File Reference	978
13.252.1 Macro Definition Documentation	979
13.252.1.1 __FFLASFFPACK_CONFIGURATION	979
13.252.1.2 __FFLASFFPACK_HAVE_LAPACK	979
13.252.1.3 __FFLASFFPACK_HAVE_CLAPACK	979
13.252.2 Function Documentation	979
13.252.2.1 main()	979
13.253 cuda.C File Reference	979
13.253.1 Function Documentation	979
13.253.1.1 main()	979
13.254 fblas.C File Reference	979
13.254.1 Macro Definition Documentation	980
13.254.1.1 __FFLASFFPACK_CONFIGURATION	980
13.254.2 Function Documentation	980
13.254.2.1 dgemm_()	980
13.254.2.2 main()	980
13.255 gmp.C File Reference	980
13.255.1 Function Documentation	980

13.255.1.1 main()	980
13.256 instrset.h File Reference	980
13.256.1 Macro Definition Documentation	981
13.256.1.1 INSTRSET_H	981
13.256.1.2 INSTRSET	981
13.256.1.3 const_int	981
13.256.1.4 const_uint	981
13.256.2 Typedef Documentation	981
13.256.2.1 int8_t	981
13.256.2.2 uint8_t	981
13.256.2.3 int16_t	981
13.256.2.4 uint16_t	982
13.256.2.5 int32_t	982
13.256.2.6 uint32_t	982
13.256.2.7 int64_t	982
13.256.2.8 uint64_t	982
13.256.2.9 intptr_t	982
13.256.3 Function Documentation	982
13.256.3.1 instrset_detect()	982
13.256.3.2 hasFMA3()	982
13.256.3.3 hasFMA4()	982
13.256.3.4 hasXOP()	982
13.256.3.5 hasAVX512ER()	982
13.257 instrset_detect.cpp File Reference	982
13.257.1 Function Documentation	983
13.257.1.1 instrset_detect()	983
13.257.1.2 hasFMA3()	983
13.257.1.3 hasFMA4()	983
13.257.1.4 hasXOP()	983
13.257.1.5 hasF16C()	983
13.257.1.6 hasAVX512ER()	983
13.258 lapack.C File Reference	983
13.258.1 Macro Definition Documentation	983
13.258.1.1 __FFLASFFPACK_CONFIGURATION	983
13.258.1.2 __FFLASFFPACK_HAVE_LAPACK	984
13.258.2 Function Documentation	984
13.258.2.1 main()	984
13.259 regression-check.C File Reference	984
13.259.1 Function Documentation	984
13.259.1.1 check1()	984
13.259.1.2 check2()	984
13.259.1.3 check3()	984

13.259.1.4 check4()	984
13.259.1.5 checkZeroDimCharpoly()	984
13.259.1.6 checkZeroDimMinPoly()	984
13.259.1.7 gf2ModularBalanced()	984
13.259.1.8 main()	985
13.260 test-charpoly-check.C File Reference	985
13.260.1 Macro Definition Documentation	985
13.260.1.1 ENABLE_CHECKER_charpoly	985
13.260.1.2 TIME_CHECKER_CHARPOLY	985
13.260.2 Function Documentation	985
13.260.2.1 printPolynomial()	985
13.260.2.2 main()	985
13.261 test-charpoly.C File Reference	985
13.261.1 Function Documentation	986
13.261.1.1 launch_test()	986
13.261.1.2 run_with_field()	986
13.261.1.3 main()	986
13.262 test-compressQ.C File Reference	986
13.262.1 Typedef Documentation	987
13.262.1.1 Field	987
13.262.2 Function Documentation	987
13.262.2.1 printvect()	987
13.262.2.2 main()	987
13.263 test-det-check.C File Reference	987
13.263.1 Macro Definition Documentation	988
13.263.1.1 ENABLE_CHECKER_Det	988
13.263.1.2 TIME_CHECKER_Det	988
13.263.2 Function Documentation	988
13.263.2.1 main()	988
13.264 test-det.C File Reference	988
13.264.1 Function Documentation	988
13.264.1.1 test_det()	988
13.264.1.2 main()	988
13.265 test-echelon.C File Reference	989
13.265.1 Macro Definition Documentation	989
13.265.1.1 __FFLASFFPACK_SEQUENTIAL	989
13.265.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE	989
13.265.1.3 __FFLASFFPACK_PLUQ_THRESHOLD	989
13.265.2 Function Documentation	989
13.265.2.1 test_colechelon()	989
13.265.2.2 test_rowechelon()	990
13.265.2.3 test_redcolechelon()	990

13.265.2.4 test_redrowechelon()	990
13.265.2.5 run_with_field()	991
13.265.2.6 main()	991
13.266 test-fadd.C File Reference	991
13.266.1 Function Documentation	991
13.266.1.1 test_fadd()	991
13.266.1.2 test_faddin()	991
13.266.1.3 test_fsub()	992
13.266.1.4 test_fsubin()	992
13.266.1.5 main()	992
13.267 test-fdot.C File Reference	992
13.267.1 Macro Definition Documentation	993
13.267.1.1 ENABLE_ALL_CHECKINGS	993
13.267.2 Function Documentation	993
13.267.2.1 check_fdot()	993
13.267.2.2 run_with_field()	993
13.267.2.3 run_with_Integer()	993
13.267.2.4 main()	993
13.268 test-fgemm-check.C File Reference	993
13.268.1 Macro Definition Documentation	994
13.268.1.1 ENABLE_ALL_CHECKINGS	994
13.268.2 Function Documentation	994
13.268.2.1 launch_MM_dispatch()	994
13.268.2.2 run_with_field()	994
13.268.2.3 main()	994
13.269 test-fgemm.C File Reference	995
13.269.1 Macro Definition Documentation	995
13.269.1.1 ENABLE_CHECKER_fgemm	995
13.269.2 Function Documentation	995
13.269.2.1 check_MM()	995
13.269.2.2 launch_MM()	996
13.269.2.3 launch_MM_dispatch()	996
13.269.2.4 run_with_field()	996
13.269.2.5 main()	997
13.270 test-fgemv.C File Reference	997
13.270.1 Function Documentation	997
13.270.1.1 check_MV()	997
13.270.1.2 launch_MV()	998
13.270.1.3 launch_MV_dispatch()	998
13.270.1.4 run_with_field()	998
13.270.1.5 main()	998
13.271 test-fger.C File Reference	998

13.271.1 Macro Definition Documentation	999
13.271.1.1 TIME	999
13.271.2 Function Documentation	999
13.271.2.1 check_fger()	999
13.271.2.2 launch_fger()	999
13.271.2.3 launch_fger_dispatch()	1000
13.271.2.4 run_with_field()	1000
13.271.2.5 main()	1000
13.272 test-fgesv.C File Reference	1000
13.272.1 Function Documentation	1001
13.272.1.1 test_square_fgesv()	1001
13.272.1.2 test_rect_fgesv()	1001
13.272.1.3 run_with_field()	1001
13.272.1.4 main()	1001
13.273 test-finit.C File Reference	1002
13.273.1 Function Documentation	1002
13.273.1.1 test_freduce()	1002
13.273.1.2 run_with_field()	1002
13.273.1.3 main()	1002
13.274 test-fscal.C File Reference	1003
13.274.1 Function Documentation	1003
13.274.1.1 test_fscal() [1/2]	1003
13.274.1.2 test_fscal() [2/2]	1003
13.274.1.3 test_fscalin() [1/2]	1003
13.274.1.4 test_fscalin() [2/2]	1004
13.274.1.5 main()	1004
13.275 test-fsyr2k.C File Reference	1004
13.275.1 Macro Definition Documentation	1004
13.275.1.1 ENABLE_ALL_CHECKINGS	1004
13.275.2 Function Documentation	1004
13.275.2.1 check_fsyr2k()	1004
13.275.2.2 run_with_field()	1005
13.275.2.3 main()	1005
13.276 test-fsyrk.C File Reference	1005
13.276.1 Macro Definition Documentation	1006
13.276.1.1 ENABLE_ALL_CHECKINGS	1006
13.276.2 Function Documentation	1006
13.276.2.1 check_fsyrk()	1006
13.276.2.2 check_fsyrk_diag()	1006
13.276.2.3 check_fsyrk_bkdiag()	1006
13.276.2.4 run_with_field()	1006
13.276.2.5 main()	1007

13.277 test-fsytrf.C File Reference	1007
13.277.1 Function Documentation	1007
13.277.1.1 operator<<()	1007
13.277.1.2 test_RPM_fsytrf()	1007
13.277.1.3 test_generic_fsytrf()	1008
13.277.1.4 run_with_field()	1008
13.277.1.5 main()	1008
13.278 test-ftmm.C File Reference	1008
13.278.1 Macro Definition Documentation	1009
13.278.1.1 __FFLASFFPACK_SEQUENTIAL	1009
13.278.2 Function Documentation	1009
13.278.2.1 check_ftmm()	1009
13.278.2.2 run_with_field()	1009
13.278.2.3 main()	1009
13.279 test-ftmv.C File Reference	1009
13.279.1 Macro Definition Documentation	1010
13.279.1.1 __FFLASFFPACK_SEQUENTIAL	1010
13.279.1.2 ENABLE_ALL_CHECKINGS	1010
13.279.2 Function Documentation	1010
13.279.2.1 check_ftmv()	1010
13.279.2.2 run_with_field()	1010
13.279.2.3 main()	1010
13.280 test-ftsm-check.C File Reference	1010
13.280.1 Macro Definition Documentation	1011
13.280.1.1 ENABLE_ALL_CHECKINGS	1011
13.280.2 Function Documentation	1011
13.280.2.1 main()	1011
13.281 test-ftsm.C File Reference	1011
13.281.1 Macro Definition Documentation	1012
13.281.1.1 __FFLASFFPACK_SEQUENTIAL	1012
13.281.1.2 ENABLE_ALL_CHECKINGS	1012
13.281.2 Function Documentation	1012
13.281.2.1 check_ftsm()	1012
13.281.2.2 run_with_field()	1012
13.281.2.3 main()	1012
13.282 test-ftssyr2k.C File Reference	1012
13.282.1 Macro Definition Documentation	1013
13.282.1.1 ENABLE_ALL_CHECKINGS	1013
13.282.2 Function Documentation	1013
13.282.2.1 check_ftssyr2k()	1013
13.282.2.2 run_with_field()	1013
13.282.2.3 main()	1013

13.283 test-ftsstr.C File Reference	1013
13.283.1 Macro Definition Documentation	1014
13.283.1.1 ENABLE_ALL_CHECKINGS	1014
13.283.2 Function Documentation	1014
13.283.2.1 check_ftsstr()	1014
13.283.2.2 run_with_field()	1014
13.283.2.3 main()	1014
13.284 test-ftsrv.C File Reference	1014
13.284.1 Macro Definition Documentation	1015
13.284.1.1 __FFLASFFPACK_SEQUENTIAL	1015
13.284.1.2 ENABLE_ALL_CHECKINGS	1015
13.284.2 Function Documentation	1015
13.284.2.1 check_ftsrv()	1015
13.284.2.2 run_with_field()	1015
13.284.2.3 main()	1015
13.285 test-ftsrti.C File Reference	1016
13.285.1 Macro Definition Documentation	1016
13.285.1.1 __FFLASFFPACK_SEQUENTIAL	1016
13.285.1.2 ENABLE_ALL_CHECKINGS	1016
13.285.2 Function Documentation	1016
13.285.2.1 check_ftsrti()	1016
13.285.2.2 run_with_field()	1016
13.285.2.3 main()	1017
13.286 test-interfaces-c.c File Reference	1017
13.286.1 Function Documentation	1017
13.286.1.1 main()	1017
13.287 test-invert-check.C File Reference	1017
13.287.1 Macro Definition Documentation	1017
13.287.1.1 ENABLE_ALL_CHECKINGS	1017
13.287.2 Function Documentation	1017
13.287.2.1 main()	1017
13.288 test-io.C File Reference	1018
13.288.1 Function Documentation	1018
13.288.1.1 run_with_field()	1018
13.288.1.2 main()	1018
13.289 test-lu.C File Reference	1018
13.289.1 Macro Definition Documentation	1019
13.289.1.1 BASECASE_K	1019
13.289.1.2 __FFLASFFPACK_SEQUENTIAL	1019
13.289.1.3 __LUDIVINE_CUTOFF	1019
13.289.2 Function Documentation	1019
13.289.2.1 test_LUdivine()	1019

13.289.2.2	verifPLUQ()	1020
13.289.2.3	test_pluq()	1021
13.289.2.4	launch_test()	1021
13.289.2.5	run_with_field()	1021
13.289.2.6	main()	1022
13.289.3	Variable Documentation	1022
13.289.3.1	tperm	1022
13.289.3.2	tgemm	1022
13.289.3.3	tBC	1022
13.289.3.4	ttrsm	1022
13.289.3.5	trest	1022
13.289.3.6	timtot	1022
13.289.3.7	mvcnt	1022
13.290	test-maxdelayeddim.C File Reference	1022
13.290.1	Macro Definition Documentation	1023
13.290.1.1	MAX_WITH_SIZE_T	1023
13.290.2	Function Documentation	1023
13.290.2.1	test()	1023
13.290.2.2	main()	1023
13.291	test-minpoly.C File Reference	1023
13.291.1	Function Documentation	1023
13.291.1.1	check_minpoly()	1023
13.291.1.2	run_with_field()	1024
13.291.1.3	main()	1024
13.292	test-multifile1.C File Reference	1024
13.293	test-multifile2.C File Reference	1024
13.293.1	Function Documentation	1024
13.293.1.1	main()	1024
13.294	test-nullspace.C File Reference	1024
13.294.1	Function Documentation	1025
13.294.1.1	checkingMessage()	1025
13.294.1.2	readOrRandomMatrixWithRankAndRandomRPM()	1025
13.294.1.3	test_nullspace()	1025
13.294.1.4	run_with_field()	1025
13.294.1.5	main()	1025
13.295	test-permutations.C File Reference	1026
13.295.1	Function Documentation	1026
13.295.1.1	checkMonotonicApplyP()	1026
13.295.1.2	main()	1026
13.295.2	Variable Documentation	1026
13.295.2.1	tperm	1026
13.295.2.2	tgemm	1026

13.295.2.3 tBC	1026
13.295.2.4 ttrsm	1026
13.295.2.5 trest	1026
13.295.2.6 timtot	1027
13.296 test-plug-check.C File Reference	1027
13.296.1 Macro Definition Documentation	1027
13.296.1.1 ENABLE_ALL_CHECKINGS	1027
13.296.2 Function Documentation	1027
13.296.2.1 main()	1027
13.297 test-rankprofiles.C File Reference	1027
13.297.1 Macro Definition Documentation	1028
13.297.1.1 __FFLASFFPACK_SEQUENTIAL	1028
13.297.2 Function Documentation	1028
13.297.2.1 run_with_field()	1028
13.297.2.2 main()	1028
13.298 test-rpm.C File Reference	1028
13.298.1 Function Documentation	1028
13.298.1.1 checkRPM()	1028
13.298.1.2 checkSymmetricRPM()	1028
13.298.1.3 main()	1029
13.299 test-simd.C File Reference	1029
13.299.1 Macro Definition Documentation	1030
13.299.1.1 REGISTER_TYPE_NAME	1030
13.299.1.2 TEST_ONE_OP	1030
13.299.2 Typedef Documentation	1030
13.299.2.1 integer	1030
13.299.3 Function Documentation	1030
13.299.3.1 TypeName()	1030
13.299.3.2 REGISTER_TYPE_NAME() [1/8]	1030
13.299.3.3 REGISTER_TYPE_NAME() [2/8]	1030
13.299.3.4 REGISTER_TYPE_NAME() [3/8]	1030
13.299.3.5 REGISTER_TYPE_NAME() [4/8]	1031
13.299.3.6 REGISTER_TYPE_NAME() [5/8]	1031
13.299.3.7 REGISTER_TYPE_NAME() [6/8]	1031
13.299.3.8 REGISTER_TYPE_NAME() [7/8]	1031
13.299.3.9 REGISTER_TYPE_NAME() [8/8]	1031
13.299.3.10 generate_random_vector() [1/2]	1031
13.299.3.11 generate_random_vector() [2/2]	1031
13.299.3.12 check_eq() [1/2]	1031
13.299.3.13 check_eq() [2/2]	1031
13.299.3.14 eval_func_on_array() [1/2]	1031
13.299.3.15 eval_func_on_array() [2/2]	1032

13.299.3.16 test_op()	1032
13.299.3.17 test_impl() [1/2]	1032
13.299.3.18 test_impl() [2/2]	1032
13.299.3.19 test() [1/2]	1032
13.299.3.20 test() [2/2]	1032
13.299.3.21 main()	1032
13.300 test-solve.C File Reference	1032
13.300.1 Function Documentation	1033
13.300.1.1 check_solve()	1033
13.300.1.2 run_with_field()	1033
13.300.1.3 main()	1033
13.301 101-fgemv.C File Reference	1033
13.301.1 Function Documentation	1033
13.301.1.1 main()	1033
13.302 2x2-fgemv.C File Reference	1033
13.302.1 Function Documentation	1034
13.302.1.1 main()	1034
13.303 2x2-ftrsv.C File Reference	1034
13.303.1 Function Documentation	1034
13.303.1.1 main()	1034
13.304 2x2-pluq.C File Reference	1034
13.304.1 Function Documentation	1035
13.304.1.1 main()	1035
13.305 fflas-101_1.C File Reference	1035
13.305.1 Function Documentation	1035
13.305.1.1 main()	1035
13.306 fflas-101_3.C File Reference	1035
13.306.1 Function Documentation	1035
13.306.1.1 main()	1035
13.307 fflas_101.C File Reference	1035
13.307.1 Function Documentation	1036
13.307.1.1 main()	1036
13.308 fflas_101_lvl1.C File Reference	1036
13.308.1 Function Documentation	1036
13.308.1.1 main()	1036
13.309 fpack-fgesv.C File Reference	1036
13.309.1 Function Documentation	1036
13.309.1.1 main()	1036
13.310 fpack-solve.C File Reference	1037
13.310.1 Function Documentation	1037
13.310.1.1 main()	1037

Chapter 1

FFLAS-FFPACK Documentation.

1.1 Introduction

FFLAS-FFPACK is a LGPL-2.1+ source code library for basic linear algebra operations over a finite field. It is inspired by BLAS interface (Basic Linear Algebra Subprograms) and the LAPACK library for numerical linear algebra, and shares part of their design. Yet it differs in many aspects due to the specifics of computing over a finite field:

- it is generic with respect to the finite field, so as to accomodate a large variety of field sizes and implementations;
- it is a pure source code library, to be included and compiled in the user's software. Its build system is only used for tests and benchmarks.

1.2 Goals

1.3 Design

1.4 Using FFLAS-FFPACK.

- [Copying and Licence](#).
- [Tutorial](#). This is a brief introduction to FFLAS-FFPACK capabilities.
- [Configuring and Installing FFLAS-FFPACK](#). Explains how to configure/install from sources or from the latest svn version.
- [Architecture of the library](#).. Describes how FFLAS-FFPACK is organized
- [Documentation for Users](#). If everything around is blue, then you are reading the lighter, user-oriented, documentation.
- [Documentation for Developers](#). If everything around is green, then you can get to everything (not necessarily yet) documented.

1.5 Contributing to fflas-ffpack, getting assistance.

Version

2.3.0

1.6 Copying and Licence

The FFLAS-FFPACK library is licensed under the terms of the GNU LGPL v2.1 or later.

See <https://www.gnu.org/licenses/lgpl-2.1.html>

1.7 Tutorial

no doc.

1.8 Configuring and Installing FFLAS-FFPACK

FFLAS-FFPACK is a header-only package.

Howver configuration process can be tweaked a lot. Configure looks for BLAS routines and [Givaro](#) library which are both mandatory dependencies. See the output of `./configure -help` for information about the LAPACK/↔ BLAS discovering strategies.

1.9 Architecture of the library.

no doc.

Chapter 2

Bug List

Global **DOUBLE_TO_FLOAT_CROSSOVER**

to be benchmarked.

Global **FFLAS::details::pack_lhs** (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)

this is fassign

this is fassign

Global **FFLAS::details::pack_rhs** (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)

this is fassign

this is fassign

Global **FFLAS::fconvert** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)

use cblas_(d)scal when possible

Global **FFLAS::fconvert** (const Field &F, const size_t n, OtherElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)

use cblas_(d)scal when possible

Global **FFLAS::finit** (const Field &F, const size_t n, const OtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global **FFLAS::finit** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

use cblas_(d)scal when possible

Global **FFLAS::fneg** (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global **FFLAS::fneg** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

use cblas_(d)scal when possible

Global **FFLAS::fnegin** (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

Global **FFLAS::fnegin** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

use cblas_(d)scal when possible

Global **FFLAS::freduce** (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)

use cblas_(d)scal when possible

- Global **FFLAS::reduce** (const Field &F, const size_t n, typename **Field::Element_ptr** X, const size_t incX)
use cblas_(d)scal when possible
- Global **FFLAS::reduce** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)
use cblas_(d)scal when possible
- Global **FFLAS::reduce** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)
use cblas_(d)scal when possible
- Global **FFLAS::fscal** (const Field &F, const size_t n, const typename **Field::Element** alpha, typename **Field::ConstElement_ptr** X, const size_t incX, typename **Field::Element_ptr** Y, const size_t incY)
use cblas_(d)scal when possible
- Global **FFLAS::fscal** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)
use cblas_(d)scal when possible
- Global **FFLAS::fscaln** (const Field &F, const size_t n, const typename **Field::Element** alpha, typename **Field::Element_ptr** X, const size_t incX)
use cblas_(d)scal when possible
- Global **FFLAS::fscaln** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)
use cblas_(d)scal when possible
- Global **FFLAS::fsquare** (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename **Field::Element** alpha, typename **Field::ConstElement_ptr** A, const size_t lda, const typename **Field::Element** beta, typename **Field::Element_ptr** C, const size_t ldc)
why double ?
- Global **FFLAS::fswap** (const Field &F, const size_t N, typename **Field::Element_ptr** X, const size_t incX, typename **Field::Element_ptr** Y, const size_t incY)
use cblas_dswap when double
- Global **FFLAS::fswap** (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)
use cblas_dswap when double
- Global **FFLAS::ftrsm** (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename **Field::Element** alpha, typename **Field::ConstElement_ptr** A, const size_t lda, typename **Field::Element_ptr** B, const size_t ldb)
 α must be non zero.
- Global **FFLAS::ftrsm** (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb)
 α must be non zero.
- Global **FFPACK::buildMatrix** (const Field &F, typename **Field::ConstElement_ptr** E, typename **Field::ConstElement_ptr** C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda, const size_t mu)
is this :
- Global **FFPACK::invert2** (const Field &F, const size_t M, typename **Field::Element_ptr** A, const size_t lda, typename **Field::Element_ptr** X, const size_t ldx, int &>nullity)
not tested.
- Global **launch_fger_dispatch** (const Field &F, const size_t nn, const typename **Field::Element** alpha, const size_t iters, RandIter &G)
test for transpo
test for incx equal

Global **launch_MM_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename **Field::Element** alpha, const typename **Field::Element** beta, const size_t iters, RandIter &G)

Global **launch_MM_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename **Field::Element** alpha, const typename **Field::Element** beta, const size_t iters, const int nbw, const bool par, RandIter &G)

test for IdX equal

test for transpo

Global **launch_MM_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename **Field::Element** alpha, const typename **Field::Element** beta, const size_t iters, RandIter &G)

test for transpo

test for IdX equal

Global **printvect** (std::ostream &o, vector< T > &vect)

does not belong here

Chapter 3

Bibliography

- Global **FFLAS::Protected::TRSMBound** (const Givaro::ModularBalanced< Element > &F) .
Dumas Giorgi Pernet 06, arXiv:cs/0601133
- Global **FFPACK::LeadingSubmatrixRankProfiles** (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP) .
Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13.
- Global **FFPACK::LUdivine** (const Field &F, const **FFLAS::FFLAS_DIAG** Diag, const **FFLAS::FFLAS_TRANSPOSE** trans, const size_t M, const size_t N, typename **Field::Element_ptr** A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE_THRESHOLD) .
Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013
• Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002
- Global **FFPACK::PLUQ** (const Field &F, const **FFLAS::FFLAS_DIAG** Diag, const size_t M, const size_t N, typename **Field::Element_ptr** A, const size_t lda, size_t *P, size_t *Q) .
Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13, 2013
- Global **FFPACK::Protected::GaussJordan** (const Field &F, const size_t M, const size_t N, typename **Field::Element_ptr** A, const size_t lda, const size_t colbeg, const size_t rowbeg, const size_t colsize, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag) .
Algorithm 2.8 of A. Storjohann Thesis 2000,
• Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013
- Class **ftsmLeftUpperNoTransNonUnit**< Element > .
Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

Chapter 4

Todo List

File `debug.h`

we should put vector printing elsewhere.

Global `FFLAS::fadd` (const Field &F, const size_t N, typename `Field::ConstElement_ptr` A, const size_t incA, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` B, const size_t incB, typename `Field::Element_ptr` C, const size_t incC)

optimise here

Global `FFLAS::fassign` (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)

variant for triangular matrix

Global `FFLAS::fassign` (const Field &F, const size_t N, typename `Field::ConstElement_ptr` Y, const size_t incY, typename `Field::Element_ptr` X, const size_t incX)

variant for triangular matrix

Global `FFLAS::fconvert` (const Field &F, const size_t m, const size_t n, OtherElement_ptr A, const size_t lda, typename `Field::ConstElement_ptr` B, const size_t ldb)

check if n == lda

Global `FFLAS::fneg` (const Field &F, const size_t m, const size_t n, typename `Field::ConstElement_ptr` B, const size_t ldb, typename `Field::Element_ptr` A, const size_t lda)

check if n == lda

Global `FFLAS::fnegin` (const Field &F, const size_t m, const size_t n, typename `Field::Element_ptr` A, const size_t lda)

check if n == lda

Global `FFLAS::fscal` (const Field &F, const size_t n, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` X, const size_t incX, typename `Field::Element_ptr` Y, const size_t incY)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::fscal` (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::fscaln` (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::fscaln` (const Field &F, const size_t n, const typename `Field::Element` alpha, typename `Field::Element_ptr` X, const size_t incX)

check if comparison with +/-1,0 is necessary.

Global **FFLAS::Protected::igemm** (const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, const int64_t beta, int64_t *C, size_t ldc)

use primitive (no [Field\(\)](#)) and specialise for int64.

Global **FFLAS::Protected::MatF2MatFI_Triangular** (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename [Field::ConstElement_ptr](#) const E, const size_t lde, const size_t m, const size_t n)

do finit(...,FFLAS_TRANS,FFLAS_DIAG)

do fconvert(...,FFLAS_TRANS,FFLAS_DIAG)

Global **FFPACK::getTriangular** (const Field &F, const **FFLAS::FFLAS_UPLO** Uplo, const **FFLAS::FFLAS_DIAG** diag, const size_t M, const size_t N, const size_t R, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) T, const size_t ldt, const bool OnlyNonZeroVectors=false)

just one triangular fzero+fassign ?

Global **FFPACK::getTriangular** (const Field &F, const **FFLAS::FFLAS_UPLO** Uplo, const **FFLAS::FFLAS_DIAG** diag, const size_t M, const size_t N, const size_t R, typename [Field::Element_ptr](#) A, const size_t lda)

just one triangular fzero+fassign ?

Global **FFPACK::invert2** (const Field &F, const size_t M, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) X, const size_t ldx, int &>nullity)

this init is not all necessary (done after ftrtri)

Global **FFPACK::LUdivine** (const Field &F, const **FFLAS::FFLAS_DIAG** Diag, const **FFLAS::FFLAS_TRANSPOSE** trans, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Q, const **FFPACK::FFPACK_LU_TAG** LuTag, const size_t cutoff)

std::swap ?

Global **FFPACK::Protected::RandomKrylovPrecond** (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)

don't assing K2 c*noc x N but only mas (c,noc) x N and store each one after the other

swap to save space ??

Module [field](#)

biblio

Global **launch_fger_dispatch** (const Field &F, const size_t nn, const typename [Field::Element](#) alpha, const size_t iters, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **launch_MM_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size_t iters, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **launch_MM_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size_t iters, const int nbw, const bool par, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Module [MMalgos](#)

biblio

Module [simd](#)

biblio

Global **test_colechelon** (Field &F, size_t m, size_t n, size_t r, size_t iters, **FFPACK::FFPACK_LU_TAG** LuTag, RandIter &G, bool par)

check lda

Global **test_det** (Field &F, size_t n, int iter, RandIter &G)

test with stride

Global **test_redcochelon** (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)

check Ida

Global **test_redrowechelon** (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)

check Ida

Global **test_rowechelon** (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)

check Ida

Chapter 5

Module Index

5.1 Modules

Here is a list of all modules:

CHECKER	41
FFLAS-FFPACK	41
FFLAS	42
Interfaces	42
Matrix Multiplication Algorithms	42
SIMD wrapper	42
FFPACK	43
FFLAS-FFPACK fields	43
RNS	43

Chapter 6

Namespace Index

6.1 Namespace List

Here is a list of all namespaces with brief descriptions:

FFLAS	45
FFLAS::BLAS3	181
FFLAS::csr_hyb_details	188
FFLAS::CuttingStrategy	188
FFLAS::details	188
FFLAS::details_spmv	196
FFLAS::ElementCategories	196
FFLAS::FieldCategories	
Traits and categories will need to be placed in a proper file later	196
FFLAS::MMHelperAlgo	197
FFLAS::ModeCategories	
Specifies the mode of action for an algorithm w.r.t	197
FFLAS::ParSeqHelper	
ParSeqHelper for both fgemm and ftrsm	198
FFLAS::Protected	198
FFLAS::sell_details	211
FFLAS::sparse_details	211
FFLAS::sparse_details_impl	225
FFLAS::StrategyParameter	265
FFLAS::StructureHelper	
StructureHelper for ftrsm	265
FFLAS::vectorised	266
FFLAS::vectorised::unswitch	270
FFPACK	
Finite Field PACK Set of elimination based routines for dense linear algebra	272
FFPACK::Protected	369
Givaro	376
MKL_CONFIG	376
RecInt	376

Chapter 7

Hierarchical Index

7.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

AlgoChooser< ModeT, ParSeq >	377
AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >	377
ArbitraryPreIntTag	377
AreEqual< X, Y >	378
AreEqual< X, X >	378
Argument	378
associatedDelayedField< Field >	379
associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >	379
associatedDelayedField< const Givaro::Modular< T, X > >	380
associatedDelayedField< const Givaro::ModularBalanced< T > >	380
associatedDelayedField< const Givaro::ZRing< T > >	381
Auto	381
Bini	381
Block	381
callLUdivine_small< Element >	381
callLUdivine_small< double >	382
callLUdivine_small< float >	382
CharpolyFailed	383
Checker_Empty< Field >	383
CheckerImplem_charpoly< Field, Polynomial >	384
CheckerImplem_Det< Field >	384
CheckerImplem_fgemm< Field >	385
CheckerImplem_ftdsm< Field >	386
CheckerImplem_invert< Field >	388
CheckerImplem_PLUQ< Field >	388
Classic	389
Column	390
CompactElement< Element >	390
CompactElement< double >	390
CompactElement< float >	390
CompactElement< int16_t >	390
CompactElement< int32_t >	391
CompactElement< int64_t >	391
compatible_data_type< Field >	391
compatible_data_type< Givaro::ZRing< double > >	391

compatible_data_type< Givaro::ZRing< float > >	392
Compose< H1, H2 >	392
Const_int_t< n >	393
Const_uint_t< n >	393
Simd128_impl< true, true, false, 2 >::Converter	393
Simd128_impl< true, true, false, 4 >::Converter	394
Simd128_impl< true, true, false, 8 >::Converter	394
Simd128_impl< true, true, true, 2 >::Converter	394
Simd128_impl< true, true, true, 4 >::Converter	395
Simd128_impl< true, true, true, 8 >::Converter	395
Simd256_impl< true, true, false, 2 >::Converter	395
Simd256_impl< true, true, false, 4 >::Converter	396
Simd256_impl< true, true, false, 8 >::Converter	396
Simd256_impl< true, true, true, 2 >::Converter	396
Simd256_impl< true, true, true, 4 >::Converter	397
Simd256_impl< true, true, true, 8 >::Converter	397
Simd512_impl< true, true, false, 8 >::Converter	397
Simd512_impl< true, true, true, 8 >::Converter	398
ConvertTo< T >	398
Coo< ValT, IdxT >	398
Coo< Field >	400
Coo< ValT, IdxT >	401
CooMat< Field >	403
CooMat< FFPACK::RNSInteger >	403
CsrMat< Field >	404
CsrMat< FFPACK::RNSInteger >	404
DefaultBoundedTag	405
DefaultTag	405
DelayedTag	405
ElementTraits< Element >	405
ElementTraits< double >	406
ElementTraits< FFPACK::rns_double_elt >	406
ElementTraits< float >	406
ElementTraits< Givaro::Integer >	406
ElementTraits< int16_t >	407
ElementTraits< int32_t >	407
ElementTraits< int64_t >	407
ElementTraits< int8_t >	408
ElementTraits< Reclnt::rint< K > >	408
ElementTraits< Reclnt::rmint< K, MG > >	408
ElementTraits< Reclnt::ruint< K > >	409
ElementTraits< uint16_t >	409
ElementTraits< uint32_t >	409
ElementTraits< uint64_t >	409
ElementTraits< uint8_t >	410
EllMat< Field >	410
EllMat< FFPACK::RNSInteger >	410
Failure	411
FailureCharpolyCheck	412
FailureDetCheck	413
FailureFgemmCheck	413
FailureInvertCheck	413
FailurePLUQCheck	413
FailureTrsmCheck	413
false_type	
isSparseMatrix< Field, M >	447
isSparseMatrixMKLFormat< F, M >	451
isSparseMatrixSimdFormat< F, M >	451

isZOSparseMatrix< F, M >	451
support_fast_mod< T >	704
support_simd< T >	705
support_simd_add< T >	705
support_simd_mod< T >	705
FieldSimd< _Field >	413
FieldTraits< Field >	419
FieldTraits< FFPACK::RNSInteger< T > >	420
FieldTraits< FFPACK::RNSIntegerMod< T > >	420
FieldTraits< Givaro::Modular< Element > >	421
FieldTraits< Givaro::ModularBalanced< Element > >	421
FieldTraits< Givaro::ZRing< double > >	422
FieldTraits< Givaro::ZRing< float > >	422
FieldTraits< Givaro::ZRing< Givaro::Integer > >	423
FieldTraits< Givaro::ZRing< int16_t > >	423
FieldTraits< Givaro::ZRing< int32_t > >	424
FieldTraits< Givaro::ZRing< int64_t > >	424
FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > >	425
FieldTraits< Givaro::ZRing< uint16_t > >	425
FieldTraits< Givaro::ZRing< uint32_t > >	426
FieldTraits< Givaro::ZRing< uint64_t > >	426
Fixed	427
FixedPrecIntTag	427
ForStrategy1D< blocksize_t, Cut, Param >	427
ForStrategy2D< blocksize_t, Cut, Param >	429
ftmmLeftLowerNoTransNonUnit< Element >	433
ftmmLeftLowerNoTransUnit< Element >	433
ftmmLeftLowerTransNonUnit< Element >	433
ftmmLeftLowerTransUnit< Element >	433
ftmmLeftUpperNoTransNonUnit< Element >	434
ftmmLeftUpperNoTransUnit< Element >	434
ftmmLeftUpperTransNonUnit< Element >	434
ftmmLeftUpperTransUnit< Element >	434
ftmmRightLowerNoTransNonUnit< Element >	434
ftmmRightLowerNoTransUnit< Element >	434
ftmmRightLowerTransNonUnit< Element >	434
ftmmRightLowerTransUnit< Element >	434
ftmmRightUpperNoTransNonUnit< Element >	435
ftmmRightUpperNoTransUnit< Element >	435
ftmmRightUpperTransNonUnit< Element >	435
ftmmRightUpperTransUnit< Element >	435
ftsmLeftLowerNoTransNonUnit< Element >	435
ftsmLeftLowerNoTransUnit< Element >	435
ftsmLeftLowerTransNonUnit< Element >	435
ftsmLeftLowerTransUnit< Element >	435
ftsmLeftUpperNoTransNonUnit< Element >	436
ftsmLeftUpperNoTransUnit< Element >	436
ftsmLeftUpperTransNonUnit< Element >	436
ftsmLeftUpperTransUnit< Element >	436
ftsmRightLowerNoTransNonUnit< Element >	436
ftsmRightLowerNoTransUnit< Element >	436
ftsmRightLowerTransNonUnit< Element >	437
ftsmRightLowerTransUnit< Element >	437
ftsmRightUpperNoTransNonUnit< Element >	437
ftsmRightUpperNoTransUnit< Element >	437
ftsmRightUpperTransNonUnit< Element >	437
ftsmRightUpperTransUnit< Element >	437
GenericTag	437

GenericTag	438
Grain	438
has_minus_eq_impl< C >	438
has_minus_impl< C >	438
has_mul_eq_impl< C >	438
has_mul_impl< C >	439
has_operation< T >	439
has_plus_eq_impl< C >	439
has_plus_impl< C >	440
HelperFlag	440
HelperMod< Field, ElementTraits >	441
HelperMod< Field, ElementCategories::MachineIntTag >	441
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >	442
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >	442
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >	443
Hybrid	444
Info	444
Info	445
is_simd< T >	446
Iterative	453
LazyTag	453
limits< T >	453
limits< char >	453
limits< double >	454
limits< float >	455
limits< Givaro::Integer >	455
limits< int >	456
limits< long >	456
limits< long long >	457
limits< Reclnt::rint< K > >	458
limits< Reclnt::ruint< K > >	458
limits< short int >	459
limits< signed char >	460
limits< unsigned char >	460
limits< unsigned int >	461
limits< unsigned long >	461
limits< unsigned long long >	462
limits< unsigned short int >	463
MachineFloatTag	463
MachineIntTag	463
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >	464
MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	469
MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	471
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >	473
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >	474
MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	476
ModeTraits< Field >	478
ModeTraits< Givaro::Modular< Element, Compute > >	478
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >	478
ModeTraits< Givaro::Modular< int16_t, Compute > >	479
ModeTraits< Givaro::Modular< int32_t, Compute > >	479
ModeTraits< Givaro::Modular< int8_t, Compute > >	479
ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >	480
ModeTraits< Givaro::Modular< uint16_t, Compute > >	480
ModeTraits< Givaro::Modular< uint32_t, Compute > >	480
ModeTraits< Givaro::Modular< uint8_t, Compute > >	481
ModeTraits< Givaro::ModularBalanced< Element > >	481

ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >	481
ModeTraits< Givaro::ModularBalanced< int16_t > >	482
ModeTraits< Givaro::ModularBalanced< int32_t > >	482
ModeTraits< Givaro::ModularBalanced< int8_t > >	482
ModeTraits< Givaro::Montgomery< T > >	483
ModeTraits< Givaro::ZRing< double > >	483
ModeTraits< Givaro::ZRing< float > >	483
ModeTraits< Givaro::ZRing< Givaro::Integer > >	484
ModularBalanced< T >	484
ModularTag	484
Montgomery< T >	484
need_field_characteristic< Field >	484
need_field_characteristic< Givaro::Modular< Field > >	485
need_field_characteristic< Givaro::ModularBalanced< Field > >	485
NoSimd< T >	485
Parallel< C, P >	486
readMyMachineType< Field, T >	490
readMyMachineType< Field, mpz_t >	490
Recursive	491
Recursive	491
rint< K >	491
rns_double	491
rns_double_elt	496
rns_double_elt_cstptr	497
rns_double_elt_ptr	500
rns_double_extended	503
RNSElementTag	506
RNSInteger< RNS >	507
RNSInteger< FFPACK::rns_double >	507
RNSIntegerMod< RNS >	510
RNSIntegerMod< FFPACK::rns_double >	510
rnsRandIter< RNS >	517
RNSInteger< RNS >::RandIter	487
RNSIntegerMod< RNS >::RandIter	489
Row	518
ruint< K >	518
ScalFunctions< Element, Enable >	518
ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type >	518
ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type >	522
Sequential	527
Simd128_impl< ArithType, Int, Signed, Size >	528
Simd128fp_base	579
Simd128_impl< true, false, true, 4 >	528
Simd128_impl< true, false, true, 8 >	529
Simd128i_base	580
Simd128_impl< true, true, true, 2 >	556
Simd128_impl< true, true, false, 2 >	529
Simd128_impl< true, true, true, 4 >	564
Simd128_impl< true, true, false, 4 >	538
Simd128_impl< true, true, true, 8 >	571
Simd128_impl< true, true, false, 8 >	547
Simd256_impl< ArithType, Int, Signed, Size >	581
Simd256fp_base	651
Simd256_impl< true, false, true, 4 >	581
Simd256_impl< true, false, true, 8 >	581
Simd256i_base	652

Simd256_impl< true, true, true, 2 >	621
Simd256_impl< true, true, false, 2 >	588
Simd256_impl< true, true, true, 4 >	629
Simd256_impl< true, true, false, 4 >	597
Simd256_impl< true, true, true, 8 >	644
Simd256_impl< true, true, false, 8 >	612
Simd512_impl< ArithType, Int, Signed, Size >	652
Simd512fp_base	677
Simd512_impl< true, false, true, 4 >	652
Simd512_impl< true, false, true, 8 >	653
Simd512i_base	677
Simd256_impl< true, true, true, 4 >	629
Simd512_impl< true, true, true, 8 >	668
Simd512_impl< true, true, false, 8 >	658
SimdChooser< T, bool, bool >	678
SimdChooser< T, false, b >	679
SimdChooser< T, true, false >	679
SimdChooser< T, true, true >	679
simdToType< T >	679
Single	680
Sparse< Field, SparseMatrix_t, IdxT, PtrT >	680
Sparse< _Field, SparseMatrix_t::COO >	680
Sparse< _Field, SparseMatrix_t::COO_ZO >	681
Sparse< _Field, SparseMatrix_t::CSR >	683
Sparse< _Field, SparseMatrix_t::CSR_ZO >	686
Sparse< _Field, SparseMatrix_t::CSR_HYB >	685
Sparse< _Field, SparseMatrix_t::ELL >	688
Sparse< _Field, SparseMatrix_t::ELL_ZO >	693
Sparse< _Field, SparseMatrix_t::ELL_simd >	690
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >	691
Sparse< _Field, SparseMatrix_t::HYB_ZO >	694
Sparse< _Field, SparseMatrix_t::SELL >	696
Sparse< _Field, SparseMatrix_t::SELL_ZO >	698
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int16_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int32_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int64_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int16_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int32_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int64_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int16_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int32_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int64_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int16_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int32_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int64_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int16_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int32_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int64_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int16_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int32_t >	680
Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int64_t >	680
SpMat< Field, flag >	700
Static_error_check< bool >	701
Static_error_check< false >	701
StatsMatrix	701
tfn_minus	706

tfn_minus_eq	706
tfn_mul	706
tfn_mul_eq	707
tfn_plus	707
tfn_plus_eq	708
Threads	708
ThreeD	708
ThreeDAdaptive	708
ThreeDInPlace	708
TRSMHelper< ReclterTrait, ParSeqTrait >	708
true_type	
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >	447
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >	447
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >	448
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >	448
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >	448
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >	449
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >	450
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >	449
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >	449
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >	450
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >	450
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >	450
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >	452
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >	452
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >	452
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >	452
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >	453
support_fast_mod< double >	704
support_fast_mod< float >	704
support_fast_mod< int64_t >	705
TwoD	710
TwoDAdaptive	710
UnparametricTag	710
Winograd	710
WinogradPar	710

Chapter 8

Data Structure Index

8.1 Data Structures

Here are the data structures with brief descriptions:

AlgoChooser< ModeT, ParSeq >	377
AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >	377
ArbitraryPrecIntTag	
Arbitrary precision integers: GMP	377
AreEqual< X, Y >	378
AreEqual< X, X >	378
Argument	378
associatedDelayedField< Field >	379
associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >	379
associatedDelayedField< const Givaro::Modular< T, X > >	380
associatedDelayedField< const Givaro::ModularBalanced< T > >	380
associatedDelayedField< const Givaro::ZRing< T > >	381
Auto	381
Bini	381
Block	381
callLUdivine_small< Element >	381
callLUdivine_small< double >	382
callLUdivine_small< float >	382
CharpolyFailed	383
Checker_Empty< Field >	383
CheckerImplem_charpoly< Field, Polynomial >	384
CheckerImplem_Det< Field >	384
CheckerImplem_fgemm< Field >	385
CheckerImplem_frsm< Field >	386
CheckerImplem_invert< Field >	388
CheckerImplem_PLUQ< Field >	388
Classic	389
Column	390
CompactElement< Element >	390
CompactElement< double >	390
CompactElement< float >	390
CompactElement< int16_t >	390
CompactElement< int32_t >	391
CompactElement< int64_t >	391
compatible_data_type< Field >	391

compatible_data_type< Givaro::ZRing< double > >	391
compatible_data_type< Givaro::ZRing< float > >	392
Compose< H1, H2 >	392
Const_int_t< n >	393
Const_uint_t< n >	393
Simd128_impl< true, true, false, 2 >::Converter	393
Simd128_impl< true, true, false, 4 >::Converter	394
Simd128_impl< true, true, false, 8 >::Converter	394
Simd128_impl< true, true, true, 2 >::Converter	394
Simd128_impl< true, true, true, 4 >::Converter	395
Simd128_impl< true, true, true, 8 >::Converter	395
Simd256_impl< true, true, false, 2 >::Converter	395
Simd256_impl< true, true, false, 4 >::Converter	396
Simd256_impl< true, true, false, 8 >::Converter	396
Simd256_impl< true, true, true, 2 >::Converter	396
Simd256_impl< true, true, true, 4 >::Converter	397
Simd256_impl< true, true, true, 8 >::Converter	397
Simd512_impl< true, true, false, 8 >::Converter	397
Simd512_impl< true, true, true, 8 >::Converter	398
ConvertTo< T >	
Force conversion to appropriate element type of ElementCategory T	398
Coo< ValT, IdxT >	398
Coo< Field >	400
Coo< ValT, IdxT >	401
CooMat< Field >	403
CsrMat< Field >	404
DefaultBoundedTag	
Use standard field operations, but keeps track of bounds on input and output	405
DefaultTag	
No specific mode of action: use standard field operations	405
DelayedTag	
Performs field operations with delayed mod reductions. Ensures result is reduced	405
ElementTraits< Element >	
ElementTraits	405
ElementTraits< double >	406
ElementTraits< FFPACK::rns_double_elt >	406
ElementTraits< float >	406
ElementTraits< Givaro::Integer >	406
ElementTraits< int16_t >	407
ElementTraits< int32_t >	407
ElementTraits< int64_t >	407
ElementTraits< int8_t >	408
ElementTraits< Reclnt::rint< K > >	408
ElementTraits< Reclnt::rmint< K, MG > >	408
ElementTraits< Reclnt::ruint< K > >	409
ElementTraits< uint16_t >	409
ElementTraits< uint32_t >	409
ElementTraits< uint64_t >	409
ElementTraits< uint8_t >	410
EllMat< Field >	410
Failure	
A precondition failed	411
FailureCharpolyCheck	412
FailureDetCheck	413
FailureFgemmCheck	413
FailureInvertCheck	413
FailurePLUQCheck	413
FailureTrsmCheck	413

FieldSimd< _Field >	413
FieldTraits< Field >	
FieldTrait	419
FieldTraits< FFPACK::RNSInteger< T > >	420
FieldTraits< FFPACK::RNSIntegerMod< T > >	420
FieldTraits< Givaro::Modular< Element > >	421
FieldTraits< Givaro::ModularBalanced< Element > >	421
FieldTraits< Givaro::ZRing< double > >	422
FieldTraits< Givaro::ZRing< float > >	422
FieldTraits< Givaro::ZRing< Givaro::Integer > >	423
FieldTraits< Givaro::ZRing< int16_t > >	423
FieldTraits< Givaro::ZRing< int32_t > >	424
FieldTraits< Givaro::ZRing< int64_t > >	424
FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > >	425
FieldTraits< Givaro::ZRing< uint16_t > >	425
FieldTraits< Givaro::ZRing< uint32_t > >	426
FieldTraits< Givaro::ZRing< uint64_t > >	426
Fixed	427
FixedPrecIntTag	
Fixed precision integers above machine precision: Givaro::reclnt	427
ForStrategy1D< blocksize_t, Cut, Param >	427
ForStrategy2D< blocksize_t, Cut, Param >	429
ftmmLeftLowerNoTransNonUnit< Element >	433
ftmmLeftLowerNoTransUnit< Element >	433
ftmmLeftLowerTransNonUnit< Element >	433
ftmmLeftLowerTransUnit< Element >	433
ftmmLeftUpperNoTransNonUnit< Element >	434
ftmmLeftUpperNoTransUnit< Element >	434
ftmmLeftUpperTransNonUnit< Element >	434
ftmmLeftUpperTransUnit< Element >	434
ftmmRightLowerNoTransNonUnit< Element >	434
ftmmRightLowerNoTransUnit< Element >	434
ftmmRightLowerTransNonUnit< Element >	434
ftmmRightLowerTransUnit< Element >	434
ftmmRightUpperNoTransNonUnit< Element >	435
ftmmRightUpperNoTransUnit< Element >	435
ftmmRightUpperTransNonUnit< Element >	435
ftmmRightUpperTransUnit< Element >	435
ftsmLeftLowerNoTransNonUnit< Element >	435
ftsmLeftLowerNoTransUnit< Element >	435
ftsmLeftLowerTransNonUnit< Element >	435
ftsmLeftLowerTransUnit< Element >	435
ftsmLeftUpperNoTransNonUnit< Element >	435
Computes the maximal size for delaying the modular reduction in a triangular system resolution	436
ftsmLeftUpperNoTransUnit< Element >	436
ftsmLeftUpperTransNonUnit< Element >	436
ftsmLeftUpperTransUnit< Element >	436
ftsmRightLowerNoTransNonUnit< Element >	436
ftsmRightLowerNoTransUnit< Element >	436
ftsmRightLowerTransNonUnit< Element >	437
ftsmRightLowerTransUnit< Element >	437
ftsmRightUpperNoTransNonUnit< Element >	437
ftsmRightUpperNoTransUnit< Element >	437
ftsmRightUpperTransNonUnit< Element >	437
ftsmRightUpperTransUnit< Element >	437
GenericTag	
Default is generic	437

GenericTag	
Generic ring	438
Grain	438
has_minus_eq_impl< C >	438
has_minus_impl< C >	438
has_mul_eq_impl< C >	438
has_mul_impl< C >	439
has_operation< T >	439
has_plus_eq_impl< C >	439
has_plus_impl< C >	440
HelperFlag	440
HelperMod< Field, ElementTraits >	441
HelperMod< Field, ElementCategories::MachineIntTag >	441
HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >	442
HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >	442
HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >	443
Hybrid	444
Info	444
Info	445
is_simd< T >	446
isSparseMatrix< Field, M >	447
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >	447
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >	447
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >	448
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >	448
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >	448
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >	449
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >	449
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >	449
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >	450
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >	450
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >	450
isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >	450
isSparseMatrixMKLFormat< F, M >	451
isSparseMatrixSimdFormat< F, M >	451
isZOSparseMatrix< F, M >	451
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >	452
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >	452
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >	452
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >	452
isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >	453
Iterative	453
LazyTag	
Performs field operations with delayed mod only when necessary. Result may not be reduced	453
limits< T >	453
limits< char >	453
limits< double >	454
limits< float >	455
limits< Givaro::Integer >	455
limits< int >	456
limits< long >	456
limits< long long >	457
limits< Reclnt::rint< K > >	458
limits< Reclnt::ruint< K > >	458
limits< short int >	459
limits< signed char >	460
limits< unsigned char >	460
limits< unsigned int >	461

limits< unsigned long >	461
limits< unsigned long long >	462
limits< unsigned short int >	463
MachineFloatTag	
Float or double	463
MachineIntTag	
Short, int, long, long long, and unsigned variants	463
MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >	464
MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	469
MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	471
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >	473
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >	474
MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >	
FGEMM Helper for Default and ConvertTo modes of operation	476
ModeTraits< Field >	
ModeTraits	478
ModeTraits< Givaro::Modular< Element, Compute > >	478
ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >	478
ModeTraits< Givaro::Modular< int16_t, Compute > >	479
ModeTraits< Givaro::Modular< int32_t, Compute > >	479
ModeTraits< Givaro::Modular< int8_t, Compute > >	479
ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >	480
ModeTraits< Givaro::Modular< uint16_t, Compute > >	480
ModeTraits< Givaro::Modular< uint32_t, Compute > >	480
ModeTraits< Givaro::Modular< uint8_t, Compute > >	481
ModeTraits< Givaro::ModularBalanced< Element > >	481
ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >	481
ModeTraits< Givaro::ModularBalanced< int16_t > >	482
ModeTraits< Givaro::ModularBalanced< int32_t > >	482
ModeTraits< Givaro::ModularBalanced< int8_t > >	482
ModeTraits< Givaro::Montgomery< T > >	483
ModeTraits< Givaro::ZRing< double > >	483
ModeTraits< Givaro::ZRing< float > >	483
ModeTraits< Givaro::ZRing< Givaro::Integer > >	484
ModularBalanced< T >	484
ModularTag	
This is a modular field like e.g. Modular<T> or ModularBalanced<T>	484
Montgomery< T >	484
need_field_characteristic< Field >	484
need_field_characteristic< Givaro::Modular< Field > >	485
need_field_characteristic< Givaro::ModularBalanced< Field > >	485
NoSimd< T >	485
Parallel< C, P >	486
RNSInteger< RNS >::RandIter	487
RNSIntegerMod< RNS >::RandIter	489
readMyMachineType< Field, T >	490
readMyMachineType< Field, mpz_t >	490
Recursive	491
Recursive	491
rint< K >	491
rns_double	491
rns_double_elt	496
rns_double_elt_cstptr	497
rns_double_elt_ptr	500
rns_double_extended	503
RNSElementTag	
Representation in a Residue Number System	506

RNSInteger< RNS >	507
RNSIntegerMod< RNS >	510
rnsRandIter< RNS >	517
Row	518
ruInt< K >	518
ScalFunctions< Element, Enable >	518
ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type >	518
ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type >	522
Sequential	527
Simd128_impl< ArithType, Int, Signed, Size >	528
Simd128_impl< true, false, true, 4 >	528
Simd128_impl< true, false, true, 8 >	529
Simd128_impl< true, true, false, 2 >	529
Simd128_impl< true, true, false, 4 >	538
Simd128_impl< true, true, false, 8 >	547
Simd128_impl< true, true, true, 2 >	556
Simd128_impl< true, true, true, 4 >	564
Simd128_impl< true, true, true, 8 >	571
Simd128fp_base	579
Simd128i_base	580
Simd256_impl< ArithType, Int, Signed, Size >	581
Simd256_impl< true, false, true, 4 >	581
Simd256_impl< true, false, true, 8 >	581
Simd256_impl< true, true, false, 2 >	588
Simd256_impl< true, true, false, 4 >	597
Simd256_impl< true, true, false, 8 >	612
Simd256_impl< true, true, true, 2 >	621
Simd256_impl< true, true, true, 4 >	629
Simd256_impl< true, true, true, 8 >	644
Simd256fp_base	651
Simd256i_base	652
Simd512_impl< ArithType, Int, Signed, Size >	652
Simd512_impl< true, false, true, 4 >	652
Simd512_impl< true, false, true, 8 >	653
Simd512_impl< true, true, false, 8 >	658
Simd512_impl< true, true, true, 8 >	668
Simd512fp_base	677
Simd512i_base	677
SimdChooser< T, bool, bool >	678
SimdChooser< T, false, b >	679
SimdChooser< T, true, false >	679
SimdChooser< T, true, true >	679
simdToType< T >	679
Single	680
Sparse< Field, SparseMatrix_t, IdxT, PtrT >	680
Sparse< _Field, SparseMatrix_t::COO >	680
Sparse< _Field, SparseMatrix_t::COO_ZO >	681
Sparse< _Field, SparseMatrix_t::CSR >	683
Sparse< _Field, SparseMatrix_t::CSR_HYB >	685
Sparse< _Field, SparseMatrix_t::CSR_ZO >	686
Sparse< _Field, SparseMatrix_t::ELL >	688
Sparse< _Field, SparseMatrix_t::ELL_simd >	690
Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >	691
Sparse< _Field, SparseMatrix_t::ELL_ZO >	693
Sparse< _Field, SparseMatrix_t::HYB_ZO >	694
Sparse< _Field, SparseMatrix_t::SELL >	696
Sparse< _Field, SparseMatrix_t::SELL_ZO >	698
SpMat< Field, flag >	700

Static_error_check< bool >	701
Static_error_check< false >	701
StatsMatrix	701
support_fast_mod< T >	704
support_fast_mod< double >	704
support_fast_mod< float >	704
support_fast_mod< int64_t >	705
support_simd< T >	705
support_simd_add< T >	705
support_simd_mod< T >	705
tfn_minus	706
tfn_minus_eq	706
tfn_mul	706
tfn_mul_eq	707
tfn_plus	707
tfn_plus_eq	708
Threads	708
ThreeD	708
ThreeDAdaptive	708
ThreeDInPlace	708
TRSMHelper< ReclterTrait, ParSeqTrait > TRSM Helper	708
TwoD	710
TwoDAdaptive	710
UnparametricTag If the field uses a representation with infix operators	710
Winograd	710
WinogradPar	710

Chapter 9

File Index

9.1 File List

Here is a list of all files with brief descriptions:

arithprog.C	711
fsyrk.C	712
fsytrf.C	713
ftrtri.C	713
winograd.C	714
benchmark-charpoly-mp.C	715
benchmark-charpoly.C	716
benchmark-checkers.C	717
benchmark-dgemm.C	718
benchmark-dgetrf.C	718
benchmark-dgetri.C	719
benchmark-dsytrf.C	720
benchmark-dtrsm.C	721
benchmark-dtrtri.C	721
benchmark-fadd-lvl2.C	722
benchmark-fdot.C	722
benchmark-fgemm-mp.C	723
benchmark-fgemm-rns.C	724
benchmark-fgemm.C	726
benchmark-fgemv-mp.C	726
benchmark-fgemv.C	728
benchmark-fgesv.C	731
benchmark-fsyrk.C	731
benchmark-fsytrf.C	732
benchmark-ftrsm-mp.C	733
benchmark-ftrsm.C	733
benchmark-ftrsv.C	734
benchmark-ftrtri.C	734
benchmark-inverse.C	735
benchmark-lqup-mp.C	736
benchmark-lqup.C	736
benchmark-pluq.C	737
benchmark-wino.C	738
mainpage.doxy	739
autotune/charpoly.C	739

examples/charpoly.C	740
det.C	740
matmul.C	741
autotune/pluq.C	741
examples/pluq.C	742
rank.C	742
solve.C	743
checker_charpoly.inl	743
checker_det.inl	743
checker_empty.h	744
checker_fgemm.inl	744
checker_ftrsm.inl	744
checker_invert.inl	745
checker_pluq.inl	745
checkers.doxy	746
checkers_fflas.h	746
checkers_fflas.inl	746
checkers_ffpack.h	747
checkers_ffpack.inl	747
config-blas.h	748
config.h	754
fflas-ffpack/config.h	757
fflas-ffpack-config.h	
Defaults for optimised values	761
fflas-ffpack-default-thresholds.h	761
fflas-ffpack-thresholds.h	762
fflas-ffpack.doxy	762
fflas-ffpack.h	
Includes FFLAS and FFPACK	762
fflas.doxy	762
fflas.h	
Finite Field Linear Algebra Subroutines	762
fflas_bounds.inl	764
fflas_enum.h	765
fflas_fadd.h	765
fflas_fadd.inl	767
fflas_fassign.h	768
fflas_fassign.inl	768
fflas_faxpy.inl	769
fflas_fdot.inl	769
fflas_fgemm.inl	770
fgemm_classical.inl	772
fgemm_classical_mp.inl	
Matrix multiplication with multiprecision input (either over \mathbb{Z} or over $\mathbb{Z}/p\mathbb{Z}$)	772
fgemm_winograd.inl	774
matmul.doxy	775
schedule_bini.inl	
Bini implementation	775
schedule_winograd.inl	776
schedule_winograd_acc.inl	776
schedule_winograd_acc_ip.inl	777
schedule_winograd_ip.inl	778
fflas_fgmv.inl	779
fflas_fgmv_mp.inl	780
fflas_fger.inl	781
fflas_fger_mp.inl	782
fflas_freduce.h	783
fflas_freduce.inl	784

fflas_freduce_mp.inl	785
fflas_freivalds.inl	786
fflas_fscal.h	786
fflas_fscal.inl	786
fflas_fscal_mp.inl	788
fflas_fsyr2k.inl	788
fflas_fsyrk.inl	789
fflas_ftrmm.inl	790
fflas_ftrsm.inl	790
fflas_ftrsm_mp.inl	
Triangular system with matrix right hand side over multiprecision domain (either over Z or over	
Z/pZ)	791
fflas_ftrsv.inl	792
fflas_helpers.inl	792
igemm.doxy	793
igemm.h	793
igemm.inl	794
igemm_kernels.h	794
igemm_kernels.inl	795
igemm_tools.h	796
igemm_tools.inl	796
fflas_level1.inl	796
fflas_level2.inl	799
fflas_level3.inl	801
fflas_pfgemm.inl	804
fflas_pftrsm.inl	804
fflas_simd.h	805
simd.doxy	807
simd128.inl	807
simd128_double.inl	807
simd128_float.inl	808
simd128_int16.inl	808
simd128_int32.inl	808
simd128_int64.inl	809
simd256.inl	809
simd256_double.inl	810
simd256_float.inl	810
simd256_int16.inl	810
simd256_int32.inl	810
simd256_int64.inl	811
simd512.inl	811
simd512_double.inl	812
simd512_float.inl	812
simd512_int32.inl	812
simd512_int64.inl	813
simd_modular.inl	813
fflas_sparse.h	813
fflas_sparse.inl	818
coo.h	820
coo_spmv.inl	820
coo_spmv.inl	821
coo_utils.inl	822
csr.h	822
csr_pspmm.inl	823
csr_pspmv.inl	824
csr_spmv.inl	824
csr_spmv.inl	826
csr_utils.inl	826

csr_hyb.h	827
csr_hyb_pspmm.inl	827
csr_hyb_pspmv.inl	828
csr_hyb_spmmm.inl	829
csr_hyb_spmv.inl	829
csr_hyb_utils.inl	830
ell.h	830
ell_pspmm.inl	831
ell_pspmv.inl	831
ell_spmmm.inl	832
ell_spmv.inl	833
ell_utils.inl	834
ell_simd.h	834
ell_simd_pspmv.inl	835
ell_simd_spmv.inl	836
ell_simd_utils.inl	837
hyb_zo.h	837
hyb_zo_pspmm.inl	837
hyb_zo_pspmv.inl	838
hyb_zo_spmmm.inl	838
hyb_zo_spmv.inl	839
hyb_zo_utils.inl	839
read_sparse.h	840
sell.h	841
sell_pspmv.inl	841
sell_spmv.inl	842
sell_utils.inl	843
sparse_matrix_traits.h	844
utils.h	845
ffpack.doxy	845
ffpack.h	
Set of elimination based routines for dense linear algebra	845
ffpack.inl	854
ffpack_charpoly.inl	855
ffpack_charpoly_danilevski.inl	856
ffpack_charpoly_kgfast.inl	856
ffpack_charpoly_kgfastgeneralized.inl	857
ffpack_charpoly_kglu.inl	857
ffpack_charpoly_mp.inl	858
ffpack_det_mp.inl	858
ffpack_echelonforms.inl	859
ffpack_fgesv.inl	860
ffpack_fgetrs.inl	861
ffpack_frobenius.inl	861
ffpack_fsytrf.inl	862
ffpack_ftrssyr2k.inl	863
ffpack_ftrstr.inl	864
ffpack_fttr.inl	864
ffpack_invert.inl	865
ffpack_krylovelim.inl	865
ffpack_ludivine.inl	865
ffpack_ludivine_mp.inl	866
ffpack_minpoly.inl	867
ffpack_permutation.inl	868
ffpack_pluq.inl	870
ffpack_pluq_mp.inl	871
ffpack_ppluq.inl	871
ffpack_rankprofiles.inl	872

field-traits.h	
Field Traits	873
field.doxy	875
rns-double-elt.h	
Rns elt structure with double support	875
rns-double-recint.inl	876
rns-double.h	
Rns structure with double support	876
rns-double.inl	877
rns-integer-mod.h	
Representation of $\mathbb{Z}/p\mathbb{Z}$ using RNS representation (note: fixed precision)	877
rns-integer.h	
Representation of \mathbb{Z} using RNS representation (note: fixed precision)	878
rns.h	879
rns.inl	879
interfaces.doxy	879
fflas_c.h	879
fflas_L1_inst.C	891
fflas_L1_inst.h	892
fflas_L1_inst_implem.inl	893
fflas_L2_inst.C	894
fflas_L2_inst.h	895
fflas_L2_inst_implem.inl	896
fflas_L3_inst.C	898
fflas_L3_inst.h	899
fflas_L3_inst_implem.inl	900
fflas_lv1.C	
C functions calls for level 1 FFLAS in fflas-c.h	901
fflas_lv2.C	
C functions calls for level 2 FFLAS in fflas-c.h	905
fflas_lv3.C	
C functions calls for level 3 FFLAS in fflas-c.h	911
fflas_sparse.C	
C functions calls for level 1.5 and 2.5 FFLAS in fflas-c.h	912
ffpack.C	
C functions calls for FFPACK in ffpack-c.h	913
ffpack_c.h	933
ffpack_inst.C	953
ffpack_inst.h	954
ffpack_inst_implem.inl	955
blockcuts.inl	958
fflas_plevel1.h	960
kaapi_routines.inl	960
parallel.h	960
pfgemm_variants.inl	966
pfgemv.inl	967
align-allocator.h	968
args-parser.h	968
bit_manipulation.h	970
cast.h	970
debug.h	
Various utilities for debugging	971
fflas_intrinsic.h	972
fflas_io.h	972
fflas_memory.h	973
fflas_randommatrix.h	973
flimits.h	975
Matio.h	976

test-utils.h	977
timer.h	978
cblas.C	978
clapack.C	978
cuda.C	979
fblas.C	979
gmp.C	980
instrset.h	980
instrset_detect.cpp	982
lapack.C	983
regression-check.C	984
test-charpoly-check.C	985
test-charpoly.C	985
test-compressQ.C	986
test-det-check.C	987
test-det.C	988
test-echelon.C	989
test-fadd.C	991
test-fdot.C	992
test-fgemm-check.C	993
test-fgemm.C	995
test-fgemv.C	997
test-fger.C	998
test-fgesv.C	1000
test-finit.C	1002
test-fscal.C	1003
test-fsyr2k.C	1004
test-fsyrk.C	1005
test-fsytrf.C	1007
test-ftrmm.C	1008
test-ftrmv.C	1009
test-ftrsm-check.C	1010
test-ftrsm.C	1011
test-ftrssyr2k.C	1012
test-ftrstr.C	1013
test-ftrsv.C	1014
test-ftrtri.C	1016
test-interfaces-c.c	1017
test-invert-check.C	1017
test-io.C	1018
test-lu.C	1018
test-maxdelayeddim.C	1022
test-minpoly.C	1023
test-multifile1.C	1024
test-multifile2.C	1024
test-nullspace.C	1024
test-permutations.C	1026
test-pluq-check.C	1027
test-rankprofiles.C	1027
test-rpm.C	1028
test-simd.C	1029
test-solve.C	1032
101-fgemm.C	1033
2x2-fgemm.C	1033
2x2-ftrsv.C	1034
2x2-pluq.C	1034
fflas-101_1.C	1035
fflas-101_3.C	1035

fflas_101.C	1035
fflas_101_lvl1.C	1036
ffpack-fgesv.C	1036
ffpack-solve.C	1037

Chapter 10

Module Documentation

10.1 CHECKER

Class CHECKER provides functions to verify computations in [FFLAS](#) and [FFPACK](#).

Class CHECKER provides functions to verify computations in [FFLAS](#) and [FFPACK](#).

10.2 FFLAS-FFPACK

the [FFLAS FFPACK](#) library

Modules

- [FFLAS](#)
The C-style wrapper of BLAS for finite field linear algebra.
- [Interfaces](#)
Interfaces for FFLAS-FFPACK.

10.2.1 Detailed Description

the [FFLAS FFPACK](#) library

C++ header library for fast exact dense linear algebra

See also

[FFLAS](#)
[FFPACK](#)

10.2.2 FFLAS

The C-style wrapper of BLAS for finite field linear algebra.

The C-style wrapper of BLAS for finite field linear algebra.

[FFLAS](#), Finite Field Linear Algebra Subroutines, provide basic linear algebra subroutines based on the BLAS interface. Therefore, the specifications are in C style; only the field given as a template parameter requires C++.

As much as possible, these routines use `ATLAS/BLAS` computations and achieve therefore high efficiency.

10.2.3 Interfaces

Interfaces for FFLAS-FFPACK.

Interfaces for FFLAS-FFPACK.

C interface in folder

See also

`libs`

10.3 Matrix Multiplication Algorithms

Matrix Multiplication (level 3) algorithms.

Files

- file [schedule_bini.inl](#)
Bini implementation.

10.3.1 Detailed Description

Matrix Multiplication (level 3) algorithms.

[Todo](#) biblio

10.4 SIMD wrapper

wraps SIMD functions Supportst SSE4.1, AVX, AVX2.

wraps SIMD functions Supportst SSE4.1, AVX, AVX2.

[Todo](#) biblio

10.5 FFPACK

Class [FFPACK](#) provides functions using fflas much as Lapack uses BLAS.

Namespaces

- namespace [FFPACK](#)
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

10.5.1 Detailed Description

Class [FFPACK](#) provides functions using fflas much as Lapack uses BLAS.

10.6 FFLAS-FFPACK fields

fields in the FFLAS-FFPACK library

Files

- file [rns-double-elt.h](#)
rns elt structure with double support
- file [rns-double.h](#)
rns structure with double support
- file [rns-integer-mod.h](#)
representation of $\mathbb{Z}/p\mathbb{Z}$ using RNS representation (note: fixed precision)
- file [rns-integer.h](#)
representation of \mathbb{Z} using RNS representation (note: fixed precision)
- file [rns.h](#)

10.6.1 Detailed Description

fields in the FFLAS-FFPACK library

Unparametric/Random elements

[Todo](#) biblio

10.7 RNS

just include them all

just include them all

Chapter 11

Namespace Documentation

11.1 FFLAS Namespace Reference

Namespaces

- namespace [BLAS3](#)
- namespace [csr_hyb_details](#)
- namespace [CuttingStrategy](#)
- namespace [details](#)
- namespace [details_spmv](#)
- namespace [ElementCategories](#)
- namespace [FieldCategories](#)

Traits and categories will need to be placed in a proper file later.

- namespace [MMHelperAlgo](#)
- namespace [ModeCategories](#)

Specifies the mode of action for an algorithm w.r.t.

- namespace [ParSeqHelper](#)

ParSeqHelper for both fgemv and ftrsm.

- namespace [Protected](#)
- namespace [sell_details](#)
- namespace [sparse_details](#)
- namespace [sparse_details_impl](#)
- namespace [StrategyParameter](#)
- namespace [StructureHelper](#)

StructureHelper for ftrsm.

- namespace [vectorised](#)

Data Structures

- struct [AlgoChooser](#)
- struct [AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >](#)
- struct [associatedDelayedField](#)
- struct [associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >](#)
- struct [associatedDelayedField< const Givaro::Modular< T, X > >](#)
- struct [associatedDelayedField< const Givaro::ModularBalanced< T > >](#)
- struct [associatedDelayedField< const Givaro::ZRing< T > >](#)
- struct [Checker_Empty](#)

- class [CheckerImplem_fgemm](#)
- class [CheckerImplem_ftsm](#)
- struct [CooMat](#)
- struct [CsrMat](#)
- struct [ElementTraits](#)
 - ElementTraits.*
 - struct [ElementTraits< double >](#)
 - struct [ElementTraits< FFPACK::rns_double_elt >](#)
 - struct [ElementTraits< float >](#)
 - struct [ElementTraits< Givaro::Integer >](#)
 - struct [ElementTraits< int16_t >](#)
 - struct [ElementTraits< int32_t >](#)
 - struct [ElementTraits< int64_t >](#)
 - struct [ElementTraits< int8_t >](#)
 - struct [ElementTraits< Reclnt::rint< K > >](#)
 - struct [ElementTraits< Reclnt::rmint< K, MG > >](#)
 - struct [ElementTraits< Reclnt::ruint< K > >](#)
 - struct [ElementTraits< uint16_t >](#)
 - struct [ElementTraits< uint32_t >](#)
 - struct [ElementTraits< uint64_t >](#)
 - struct [ElementTraits< uint8_t >](#)
 - struct [EIIIMat](#)
 - struct [FieldTraits](#)
 - FieldTrait.*
 - struct [FieldTraits< FFPACK::RNSInteger< T > >](#)
 - struct [FieldTraits< FFPACK::RNSIntegerMod< T > >](#)
 - struct [FieldTraits< Givaro::Modular< Element > >](#)
 - struct [FieldTraits< Givaro::ModularBalanced< Element > >](#)
 - struct [FieldTraits< Givaro::ZRing< double > >](#)
 - struct [FieldTraits< Givaro::ZRing< float > >](#)
 - struct [FieldTraits< Givaro::ZRing< Givaro::Integer > >](#)
 - struct [FieldTraits< Givaro::ZRing< int16_t > >](#)
 - struct [FieldTraits< Givaro::ZRing< int32_t > >](#)
 - struct [FieldTraits< Givaro::ZRing< int64_t > >](#)
 - struct [FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > >](#)
 - struct [FieldTraits< Givaro::ZRing< uint16_t > >](#)
 - struct [FieldTraits< Givaro::ZRing< uint32_t > >](#)
 - struct [FieldTraits< Givaro::ZRing< uint64_t > >](#)
 - struct [ForStrategy1D](#)
 - struct [ForStrategy2D](#)
 - struct [has_minus_eq_impl](#)
 - struct [has_minus_impl](#)
 - struct [has_mul_eq_impl](#)
 - struct [has_mul_impl](#)
 - struct [has_operation](#)
 - struct [has_plus_eq_impl](#)
 - struct [has_plus_impl](#)
 - struct [HelperFlag](#)
 - struct [isSparseMatrix](#)
 - struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >](#)
 - struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >](#)
 - struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >](#)
 - struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >](#)
 - struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >](#)

- struct `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >`
- struct `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >`
- struct `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >`
- struct `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >`
- struct `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >`
- struct `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >`
- struct `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >`
- struct `isSparseMatrixMKLFormat`
- struct `isSparseMatrixSimdFormat`
- struct `isZOSparseMatrix`
- struct `isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >`
- struct `isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >`
- struct `isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >`
- struct `isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >`
- struct `isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >`
- struct `MMHelper`
- struct `MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`
- struct `MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`
- struct `MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >`
- struct `MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >`
- struct `MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`

FGEMM Helper for Default and ConvertTo modes of operation.

- struct `ModeTraits`
 - ModeTraits.*
- struct `ModeTraits< Givaro::Modular< Element, Compute > >`
- struct `ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >`
- struct `ModeTraits< Givaro::Modular< int16_t, Compute > >`
- struct `ModeTraits< Givaro::Modular< int32_t, Compute > >`
- struct `ModeTraits< Givaro::Modular< int8_t, Compute > >`
- struct `ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >`
- struct `ModeTraits< Givaro::Modular< uint16_t, Compute > >`
- struct `ModeTraits< Givaro::Modular< uint32_t, Compute > >`
- struct `ModeTraits< Givaro::Modular< uint8_t, Compute > >`
- struct `ModeTraits< Givaro::ModularBalanced< Element > >`
- struct `ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >`
- struct `ModeTraits< Givaro::ModularBalanced< int16_t > >`
- struct `ModeTraits< Givaro::ModularBalanced< int32_t > >`
- struct `ModeTraits< Givaro::ModularBalanced< int8_t > >`
- struct `ModeTraits< Givaro::Montgomery< T > >`
- struct `ModeTraits< Givaro::ZRing< double > >`
- struct `ModeTraits< Givaro::ZRing< float > >`
- struct `ModeTraits< Givaro::ZRing< Givaro::Integer > >`
- struct `readMyMachineType`
- struct `readMyMachineType< Field, mpz_t >`
- struct `Sparse`
- struct `Sparse< _Field, SparseMatrix_t::COO >`
- struct `Sparse< _Field, SparseMatrix_t::COO_ZO >`
- struct `Sparse< _Field, SparseMatrix_t::CSR >`
- struct `Sparse< _Field, SparseMatrix_t::CSR_HYB >`
- struct `Sparse< _Field, SparseMatrix_t::CSR_ZO >`
- struct `Sparse< _Field, SparseMatrix_t::ELL >`
- struct `Sparse< _Field, SparseMatrix_t::ELL_simd >`
- struct `Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >`
- struct `Sparse< _Field, SparseMatrix_t::ELL_ZO >`

- struct [Sparse<_Field, SparseMatrix_t::HYB_ZO >](#)
- struct [Sparse<_Field, SparseMatrix_t::SELL >](#)
- struct [Sparse<_Field, SparseMatrix_t::SELL_ZO >](#)
- struct [SpMat](#)
- struct [StatsMatrix](#)
- struct [support_fast_mod](#)
- struct [support_fast_mod< double >](#)
- struct [support_fast_mod< float >](#)
- struct [support_fast_mod< int64_t >](#)
- struct [support_simd](#)
- struct [support_simd_add](#)
- struct [support_simd_mod](#)
- struct [tfn_minus](#)
- struct [tfn_minus_eq](#)
- struct [tfn_mul](#)
- struct [tfn_mul_eq](#)
- struct [tfn_plus](#)
- struct [tfn_plus_eq](#)
- struct [TRSMHelper](#)

TRSM Helper.

Typedefs

- template<class [Field](#) >
using [Checker_fgemm](#) = [FFLAS::Checker_Empty< Field >](#)
- template<class [Field](#) >
using [Checker_ftrsm](#) = [FFLAS::Checker_Empty< Field >](#)
- template<class [Field](#) >
using [ForceCheck_fgemm](#) = [CheckerImplem_fgemm< Field >](#)
- template<class [Field](#) >
using [ForceCheck_ftrsm](#) = [CheckerImplem_ftrsm< Field >](#)
- using [ZOSparseMatrix](#) = std::true_type
- using [NotZOSparseMatrix](#) = std::false_type
- using [SimdSparseMatrix](#) = std::true_type
- using [NoSimdSparseMatrix](#) = std::false_type
- using [MKLSparseMatrixFormat](#) = std::true_type
- using [NotMKLSparseMatrixFormat](#) = std::false_type
- template<class T >
using [has_plus](#) = typename std::conditional< std::is_arithmetic< T >::value, std::true_type, [has_plus_impl< T >](#) >::type
- template<class T >
using [has_minus](#) = typename std::conditional< std::is_arithmetic< T >::value, std::true_type, [has_minus_impl< T >](#) >::type
- template<class T >
using [has_equal](#) = typename std::conditional< std::is_arithmetic< T >::value, std::true_type, std::is_copyable< T > & assignable< T > >::type
- template<class T >
using [has_plus_eq](#) = typename std::conditional< std::is_arithmetic< T >::value, std::true_type, [has_plus_eq_impl< T >](#) >::type
- template<class T >
using [has_minus_eq](#) = typename std::conditional< std::is_arithmetic< T >::value, std::true_type, [has_minus_eq_impl< T >](#) >::type
- template<class T >
using [has_mul](#) = typename std::conditional< std::is_arithmetic< T >::value, std::true_type, [has_mul_impl< T >](#) >::type

- `template<class T >`
`using has_mul_eq = typename std::conditional< std::is_arithmetic< T >::value, std::true_type,`
`has_mul_eq_impl< T > >::type`
- `typedef Givaro::Timer Timer`
- `typedef Givaro::BaseTimer BaseTimer`
- `typedef Givaro::UserTimer UserTimer`
- `typedef Givaro::SysTimer SysTimer`

Enumerations

- `enum FFLAS_ORDER { FflasRowMajor = 101 , FflasColMajor = 102 }`
Storage by row or col ?
- `enum FFLAS_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 }`
Is matrix transposed ?
- `enum FFLAS_UPLO { FflasUpper = 121 , FflasLower = 122 }`
Is triangular matrix's shape upper ?
- `enum FFLAS_DIAG { FflasNonUnit = 131 , FflasUnit = 132 }`
Is the triangular matrix implicitly unit diagonal ?
- `enum FFLAS_SIDE { FflasLeft = 141 , FflasRight = 142 }`
On what side ?
- `enum FFLAS_BASE { FflasDouble = 151 , FflasFloat = 152 , FflasGeneric = 153 }`
FFLAS_BASE determines the type of the element representation for Matrix Mult kernel.
- `enum number_kind { zero = 0 , one = 1 , mone = -1 , other = 2 }`
- `enum class SparseMatrix_t {`
`CSR , CSR_ZO , CSC , CSC_ZO ,`
`COO , COO_ZO , ELL , ELL_ZO ,`
`SELL , SELL_ZO , ELL_simd , ELL_simd_ZO ,`
`CSR_HYB , HYB_ZO }`
- `enum FFLAS_FORMAT {`
`FflasAuto = 0 , FflasDense = 1 , FflasSMS = 2 , FflasBinary = 3 ,`
`FflasMath = 4 , FflasMaple = 5 , FflasSageMath = 6 }`

Functions

- `Givaro::Integer InfNorm (const size_t M, const size_t N, const Givaro::Integer *A, const size_t lda)`
- `template<class T >`
`const T & min3 (const T &m, const T &n, const T &k)`
- `template<class T >`
`const T & max3 (const T &m, const T &n, const T &k)`
- `template<class T >`
`const T & min4 (const T &m, const T &n, const T &k, const T &l)`
- `template<class T >`
`const T & max4 (const T &m, const T &n, const T &k, const T &l)`
- `template<class Field >`
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename`
`Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`

- `template<class Field >`
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
fadd : matrix addition.
- `template<class Field >`
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
fsub : matrix subtraction.
- `template<class Field >`
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
faddin
- `template<class Field >`
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
fsubin $C = C - B$
- `template<class Field >`
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
fadd : matrix addition with scaling.
- `template<class Field >`
`void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`
fassign : $x \leftarrow y$.
- `template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`

- `template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<class Field >`
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`

$$fassign : A \leftarrow B.$$
- `template<class Field >`
`void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t idx, typename Field::Element_ptr Y, const size_t ldy)`

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<class Field >`
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field >`
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DelayedTag &MT)`
- `template<> Givaro::DoubleDomain::Element fdot (const Givaro::DoubleDomain &, const size_t N, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<> Givaro::FloatDomain::Element fdot (const Givaro::FloatDomain &, const size_t N, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field, class T >`
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::ConvertTo< T > &MT)`
- `template<class Field >`
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultBoundedTag &dbt)`
- `template<class Field >`
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, const ParSeqHelper::Sequential seq)`
- `template<class Field >`
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`

$$fdot: dot\ product\ x^T y.$$
- `template<class Field >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)`
- `template<typename Field >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)`

- `template<typename Field , class Cut , class Param >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, type-`
`name Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t`
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const`
`ParSeqHelper::Parallel< Cut, Param > par)`
- `template<typename Field >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const`
`typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`

fgemm: Field GEneral Matrix Multiply.
- `template<typename Field , class ModeT , class ParSeq >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`
`MMHelperAlgo::Auto, ModeT, ParSeq > &H)`
- `template<class Field >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`
`MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)`
- `template<class Field >`
`Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename`
`Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element`
`beta, typename Field::Element_ptr C, const size_t ldc)`

fsquare: Squares a matrix.
- `template<> double * fsquare (const Givaro::ModularBalanced< double > &F, const FFLAS_TRANSPOSE`
`ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const`
`size_t ldc)`
- `template<> float * fsquare (const Givaro::ModularBalanced< float > &F, const FFLAS_TRANSPOSE ta,`
`const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)`
- `template<> double * fsquare (const Givaro::Modular< double > &F, const FFLAS_TRANSPOSE ta, const`
`size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t`
`ldc)`
- `template<> float * fsquare (const Givaro::Modular< float > &F, const FFLAS_TRANSPOSE ta, const size_t`
`n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)`
- `template<typename RNS , typename ParSeqTrait >`
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const`
`FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k,`
`const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS`
`>::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr`
`Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename`
`FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS`
`>, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential,`
`ParSeqTrait > > &H)`
- `template<typename RNS >`
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const`
`FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k,`
`const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS`
`>::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr`
`Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename`
`FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS`
`>, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)`

- `template<typename RNS , typename ParSeqTrait >`
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)`
- `template<typename RNS , typename Cut , typename Param >`
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)`
- `template<class ParSeq >`
`Givaro::Integer * fgemm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<typename RNS , class ModeT >`
`RNS::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)`
- `template<typename RNS >`
`RNS::Element_ptr fgemm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)`
- `Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)`
- `template<class ParSeq >`
`Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<size_t K1, size_t K2, class ParSeq >`
`Reclnt::ruint< K1 > * fgemm (const Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Reclnt::ruint< K1 > alpha, const Reclnt::ruint< K1 > *A, const size_t lda, const Reclnt::ruint< K1 > *B, const size_t ldb, Reclnt::ruint< K1 > beta, Reclnt::ruint< K1 > *C, const size_t ldc, MMHelper< Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<class Field , class ModeT >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`

const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), ModeT > &H)

- template<class [Field](#) , class ModeT , class Cut , class Param >
[Field::Element_ptr](#) [fgemm](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::WinogradPar](#), ModeT, [ParSeqHelper::Parallel](#)< Cut, Param > > &H)
 - template<class [Field](#) >
[Field::Element_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) X, const size_t incX, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) Y, const size_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo](#)< [ElementCategories::MachineFloatTag](#) > > &H)
 - template<class [Field](#) >
[Field::Element_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) X, const size_t incX, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) Y, const size_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DelayedTag](#) > &H)
 - template<class [Field](#) >
[Field::Element_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) X, const size_t incX, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) Y, const size_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
 - template<class [Field](#) >
[Field::Element_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) X, const size_t incX, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) Y, const size_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::LazyTag](#) > &H)
 - template<class [Field](#) >
[Field::Element_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) TransA, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) X, const size_t incX, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) Y, const size_t incY)
- finite prime Field GEneral Matrix Vector multiplication.*
- Givaro::ZRing< [int64_t](#) >::Element_ptr [fgemv](#) (const Givaro::ZRing< [int64_t](#) > &F, const [FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, const [int64_t](#) alpha, const [int64_t](#) *A, const size_t lda, const [int64_t](#) *X, const size_t incX, const [int64_t](#) beta, [int64_t](#) *Y, const size_t incY, [MMHelper](#)< Givaro::ZRing< [int64_t](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
 - Givaro::DoubleDomain::Element_ptr [fgemv](#) (const Givaro::DoubleDomain &F, const [FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr A, const size_t lda, const Givaro::DoubleDomain::ConstElement_ptr X, const size_t incX, const Givaro::DoubleDomain::Element beta, Givaro::DoubleDomain::Element_ptr Y, const size_t incY, [MMHelper](#)< Givaro::DoubleDomain, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
 - template<class [Field](#) >
[Field::Element_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, const typename [Field::Element](#) alpha, const typename [Field::ConstElement_ptr](#) A, const size_t lda, const typename [Field::ConstElement_ptr](#) X, const size_t incX, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) Y, const size_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultBoundedTag](#) > &H)
 - Givaro::FloatDomain::Element_ptr [fgemv](#) (const Givaro::FloatDomain &F, const [FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr A, const size_t lda, const Givaro::FloatDomain::ConstElement_ptr X, const size_t incX, const Givaro::FloatDomain::Element beta, Givaro::FloatDomain::Element_ptr Y, const size_t incY, [MMHelper](#)< Givaro::FloatDomain, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)

- `template<class Field, class Cut, class Param >`
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n,`
`const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const`
`typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename`
`Field::Element_ptr Y, const size_t incY, ParSeqHelper::Parallel< Cut, Param > &parH)`
- `template<class Field >`
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n,`
`const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const`
`typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename`
`Field::Element_ptr Y, const size_t incY, ParSeqHelper::Sequential &seqH)`
- `FFPACK::rns_double::Element_ptr fgemv (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const`
`FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const FFPACK::rns_double::Element alpha,`
`FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::ConstElement_ptr`
`X, const size_t incX, const FFPACK::rns_double::Element beta, FFPACK::rns_double::Element_ptr Y,`
`const size_t incY, MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic,`
`ModeCategories::DefaultTag > &H)`
- `FFPACK::rns_double::Element_ptr fgemv (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F,`
`const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const FFPACK::rns_double::Element alpha,`
`FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::ConstElement_ptr X,`
`const size_t incX, const FFPACK::rns_double::Element beta, FFPACK::rns_double::Element_ptr Y, const`
`size_t incY, MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic,`
`ModeCategories::DefaultTag > &H)`
- `Givaro::Integer * fgemv (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE`
`ta, const size_t m, const size_t n, const Givaro::Integer alpha, Givaro::Integer *A, const size_t`
`lda, Givaro::Integer *X, const size_t ldx, Givaro::Integer beta, Givaro::Integer *Y, const size_t`
`ldy, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<`
`ElementCategories::RNSElementTag > > &H)`
- `Givaro::Integer * fgemv (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE`
`ta, const size_t m, const size_t n, const Givaro::Integer alpha, Givaro::Integer *A, const size_t`
`lda, Givaro::Integer *X, const size_t ldx, Givaro::Integer beta, Givaro::Integer *Y, const size_t`
`ldy, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<`
`ElementCategories::RNSElementTag > > &H)`
- `template<size_t K1, size_t K2, class ParSeq >`
`Reclnt::ruint< K1 > * fgemv (const Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >`
`&F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const Reclnt::ruint< K1 > al-`
`pha, const Reclnt::ruint< K1 > *A, const size_t lda, const Reclnt::ruint< K1 > *X, const size_t incx,`
`Reclnt::ruint< K1 > beta, Reclnt::ruint< K1 > *Y, const size_t incy, MMHelper< Givaro::Modular<`
`Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<`
`ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<class Field >`
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t`
`incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic,`
`ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)`

fger: rank one update of a general matrix
- `template<class Field >`
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, type-`
`name Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t`
`incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic,`
`ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)`
- `template<class Field, class AnyTag >`
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, type-`
`name Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, AnyTag > &H)`
- `void fger (const Givaro::DoubleDomain &F, const size_t M, const size_t N, const Givaro::DoubleDomain::`
`Element alpha, const Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, const Givaro::Double`
`Domain::ConstElement_ptr y, const size_t incy, Givaro::DoubleDomain::Element_ptr A, const size_t lda,`
`MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`

- `template<class Field >`
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, const`
`typename Field::ConstElement_ptr x, const size_t incx, const typename Field::ConstElement_ptr y, const`
`size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic,`
`ModeCategories::DefaultBoundedTag > &H)`
- `void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::FloatDomain::Element`
`alpha, const Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, const Givaro::FloatDomain::↵`
`ConstElement_ptr y, const size_t incy, Givaro::FloatDomain::Element_ptr A, const size_t lda, MMHelper<`
`Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >`
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, type-`
`name Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t ↵`
`incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic,`
`ModeCategories::LazyTag > &H)`
- `template<class Field >`
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, type-`
`name Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t ↵`
`incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic,`
`ModeCategories::DelayedTag > &H)`
- `void fger (const Givaro::Modular< Givaro::Integer > &F, const size_t M, const size_t N, const typename`
`Givaro::Integer alpha, typename Givaro::Integer *x, const size_t incx, typename Givaro::Integer *y, const`
`size_t incy, typename Givaro::Integer *A, const size_t lda, MMHelper< Givaro::Modular< Givaro::Integer >,`
`MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)`
- `template<typename RNS >`
`void fger (const FFPACK::RNSInteger< RNS > &F, const size_t M, const size_t N, const typename`
`FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::Element_ptr x,`
`const size_t incx, typename FFPACK::RNSInteger< RNS >::Element_ptr y, const size_t incy, typename`
`FFPACK::RNSInteger< RNS >::Element_ptr A, const size_t lda, MMHelper< FFPACK::RNSInteger< RNS`
`>, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<typename RNS >`
`void fger (const FFPACK::RNSIntegerMod< RNS > &F, const size_t M, const size_t N, const type-`
`name FFPACK::RNSIntegerMod< RNS >::Element alpha, typename FFPACK::RNSIntegerMod< RNS`
`>::Element_ptr x, const size_t incx, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y, const`
`size_t incy, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A, const size_t lda, MMHelper<`
`FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > &H)`
- `template<class Field >`
`void freduce (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename`
`Field::Element_ptr X, const size_t incX)`

$$freduce\ x \leftarrow y \bmod F.$$
- `template<class Field >`
`void freduce (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`

$$freduce\ x \leftarrow x \bmod F.$$
- `template<class Field >`
`void freduce_constoverride (const Field &F, const size_t m, typename Field::ConstElement_ptr A, const`
`size_t incX)`
- `template<class Field, class ConstOtherElement_ptr >`
`void finit (const Field &F, const size_t n, ConstOtherElement_ptr Y, const size_t incY, typename`
`Field::Element_ptr X, const size_t incX)`
- `template<class Field >`
`void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`

$$finit\ \text{Initializes } X \text{ in } F\$.$$
- `template<class Field >`
`void freduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`

$$freduce\ A \leftarrow A \bmod F.$$
- `template<class Field >`
`void pfreduce (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t`
`lda, const size_t numths)`

- template<class [Field](#) >
void [freduce](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::ConstElement_ptr](#) B, const size_t ldb, typename [Field::Element_ptr](#) A, const size_t lda)
$$\text{freduce } A \leftarrow B \bmod F.$$
- template<class [Field](#) >
void [freduce_constoverride](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::ConstElement_ptr](#) A, const size_t lda)
- template<class [Field](#), class OtherElement_ptr >
void [finit](#) (const [Field](#) &F, const size_t m, const size_t n, const OtherElement_ptr B, const size_t ldb, typename [Field::Element_ptr](#) A, const size_t lda)
$$\text{finit } A \leftarrow B \bmod F.$$
- template<class [Field](#) >
void [finit](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, const size_t lda)
- template<> void [freduce](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns_double](#) > &F, const size_t n, [FFPACK::RNSIntegerMod](#)< [FFPACK::rns_double](#) >::Element_ptr A, size_t inc)
- template<> void [freduce](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns_double](#) > &F, const size_t m, const size_t n, [FFPACK::rns_double::Element_ptr](#) A, size_t lda)
- template<class [Field](#) >
bool [freivalds](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, typename [Field::ConstElement_ptr](#) C, const size_t ldc)
$$\text{freivalds: Freivalds } \mathbf{G} \mathbf{E} \mathbf{n} \mathbf{e} \mathbf{r} \mathbf{a} \mathbf{l} \mathbf{M} \mathbf{a} \mathbf{t} \mathbf{r} \mathbf{i} \mathbf{x} \mathbf{M} \mathbf{u} \mathbf{l} \mathbf{t} \mathbf{i} \mathbf{p} \mathbf{y} \mathbf{R} \mathbf{a} \mathbf{n} \mathbf{d} \mathbf{o} \mathbf{m} \mathbf{C} \mathbf{h} \mathbf{e} \mathbf{c} \mathbf{k}.$$
- template<class [Field](#) >
void [fscal](#) (const [Field](#) &F, const size_t n, const typename [Field::Element](#) alpha, typename [Field::Element_ptr](#) X, const size_t incX)
$$\text{fscal } x \leftarrow \alpha \cdot x.$$
- template<class [Field](#) >
void [fscal](#) (const [Field](#) &F, const size_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) X, const size_t incX, typename [Field::Element_ptr](#) Y, const size_t incY)
$$\text{fscal } y \leftarrow \alpha \cdot x.$$
- template<> void [fscal](#) (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void [fscal](#) (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)
- template<> void [fscal](#) (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void [fscal](#) (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::Element_ptr y, const size_t incy)
- template<class [Field](#) >
void [fscal](#) (const [Field](#) &F, const size_t m, const size_t n, const typename [Field::Element](#) alpha, typename [Field::Element_ptr](#) A, const size_t lda)
$$\text{fscal } A \leftarrow a \cdot A.$$
- template<class [Field](#) >
void [fscal](#) (const [Field](#) &F, const size_t m, const size_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb)
$$\text{fscal } B \leftarrow a \cdot A.$$
- template<> void [fscal](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns_double](#) > &F, const size_t n, const [FFPACK::rns_double::Element](#) alpha, [FFPACK::rns_double::Element_ptr](#) A, const size_t inc)
- template<> void [fscal](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns_double](#) > &F, const size_t n, const [FFPACK::rns_double::Element](#) alpha, [FFPACK::rns_double::ConstElement_ptr](#) A, const size_t Ainc, [FFPACK::rns_double::Element_ptr](#) B, const size_t Binc)
- template<> void [fscal](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns_double](#) > &F, const size_t m, const size_t n, const [FFPACK::rns_double::Element](#) alpha, [FFPACK::rns_double::Element_ptr](#) A, const size_t lda)

- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)`
- `template<> void fscaln (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha, typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, const size_t inc)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscaln (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)`
- `template<class Field >`
`Field::Element_ptr fsyr2k (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`
fsyr2k: Symmetric Rank 2K update
- `template<class Field >`
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`
fsyrk: Symmetric Rank K update
- `template<class Field >`
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)`
fsyrk: Symmetric Rank K update with diagonal scaling
- `template<class Field >`
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq, const size_t threshold)`
- `template<class Field , class Cut , class Param >`
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold)`
- `template<class Field >`
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const std::vector< bool > &twoBlock, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)`
fsyrk: Symmetric Rank K update with diagonal scaling
- `template<class Field >`
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`
ftrmm: TRIangular Matrix Multiply.
- `template<class Field >`
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE`

TransA, const [FFLAS_DIAG](#) Diag, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc)

ftmrm: TRIangular Matrix Multiply with 3 operands Computes $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$ or $C \leftarrow \alpha \text{Bop}(A) + \text{beta}C$.

- template<class [Field](#) >
void [ftrsm](#) (const [Field](#) &F, const [FFLAS_SIDE](#) Side, const [FFLAS_UPLO](#) Uplo, const [FFLAS_TRANSPOSE](#) TransA, const [FFLAS_DIAG](#) Diag, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb)
- template<class [Field](#) >
void [ftrsm](#) (const [Field](#) &F, const [FFLAS_SIDE](#) Side, const [FFLAS_UPLO](#) Uplo, const [FFLAS_TRANSPOSE](#) TransA, const [FFLAS_DIAG](#) Diag, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb, const [ParSeqHelper::Sequential](#) &PSH)
- template<class [Field](#) , class Cut , class Param >
void [ftrsm](#) (const [Field](#) &F, const [FFLAS_SIDE](#) Side, const [FFLAS_UPLO](#) Uplo, const [FFLAS_TRANSPOSE](#) TransA, const [FFLAS_DIAG](#) Diag, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb, const [ParSeqHelper::Parallel](#)< Cut, Param > &PSH)
- template<class [Field](#) , class ParSeqTrait = [ParSeqHelper::Sequential](#)>
void [ftrsm](#) (const [Field](#) &F, const [FFLAS_SIDE](#) Side, const [FFLAS_UPLO](#) Uplo, const [FFLAS_TRANSPOSE](#) TransA, const [FFLAS_DIAG](#) Diag, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb, [TRSMHelper](#)< [StructureHelper::Recursive](#), ParSeqTrait > &H)
- void [ftrsm](#) (const [Givaro::Modular](#)< [Givaro::Integer](#) > &F, const [FFLAS_SIDE](#) Side, const [FFLAS_UPLO](#) Uplo, const [FFLAS_TRANSPOSE](#) TransA, const [FFLAS_DIAG](#) Diag, const size_t M, const size_t N, const [Givaro::Integer](#) alpha, const [Givaro::Integer](#) *A, const size_t lda, [Givaro::Integer](#) *B, const size_t ldb)
- void [cbblas_imptrsm](#) (const enum [FFLAS_ORDER](#) Order, const enum [FFLAS_SIDE](#) Side, const enum [FFLAS_UPLO](#) Uplo, const enum [FFLAS_TRANSPOSE](#) TransA, const enum [FFLAS_DIAG](#) Diag, const int M, const int N, const [FFPACK::rns_double_elt](#) alpha, [FFPACK::rns_double_elt_cstptr](#) A, const int lda, [FFPACK::rns_double_elt_ptr](#) B, const int ldb)
- template<class [Field](#) >
void [ftrsv](#) (const [Field](#) &F, const [FFLAS_UPLO](#) Uplo, const [FFLAS_TRANSPOSE](#) TransA, const [FFLAS_DIAG](#) Diag, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) X, int incX)

ftrsv: TRIangular System solve with Vector Computes $X \leftarrow \text{op}(A^{-1})X$
- void [igemm_](#) (const enum [FFLAS_ORDER](#) Order, const enum [FFLAS_TRANSPOSE](#) TransA, const enum [FFLAS_TRANSPOSE](#) TransB, const size_t M, const size_t N, const size_t K, const [int64_t](#) alpha, const [int64_t](#) *A, const size_t lda, const [int64_t](#) *B, const size_t ldb, const [int64_t](#) beta, [int64_t](#) *C, const size_t ldc)
- template<class [Field](#) , class OtherElement_ptr >
void [finit](#) (const [Field](#) &F, const size_t n, const OtherElement_ptr Y, const size_t incY, typename [Field::Element_ptr](#) X, const size_t incX)

finit $x \leftarrow y \bmod F$.
- template<class [Field](#) , class OtherElement_ptr >
void [fconvert](#) (const [Field](#) &F, const size_t n, OtherElement_ptr X, const size_t incX, typename [Field::ConstElement_ptr](#) Y, const size_t incY)

fconvert $x \leftarrow y \bmod F$.
- template<class [Field](#) >
void [fnegin](#) (const [Field](#) &F, const size_t n, typename [Field::Element_ptr](#) X, const size_t incX)

fnegin $x \leftarrow -x$.
- template<class [Field](#) >
void [fneg](#) (const [Field](#) &F, const size_t n, typename [Field::ConstElement_ptr](#) Y, const size_t incY, typename [Field::Element_ptr](#) X, const size_t incX)

fneg $x \leftarrow -y$.
- template<class [Field](#) >
void [fzero](#) (const [Field](#) &F, const size_t n, typename [Field::Element_ptr](#) X, const size_t incX)

fzero : $A \leftarrow 0$.

- template<class [Field](#) , class RandIter >
void [frand](#) (const [Field](#) &F, RandIter &G, const size_t n, typename [Field::Element_ptr](#) X, const size_t incX)
frand : $A \leftarrow random.$
- template<class [Field](#) >
bool [fiszero](#) (const [Field](#) &F, const size_t n, typename [Field::ConstElement_ptr](#) X, const size_t incX)
fiszero : test $X = 0.$
- template<class [Field](#) >
bool [fequal](#) (const [Field](#) &F, const size_t n, typename [Field::ConstElement_ptr](#) X, const size_t incX, typename [Field::ConstElement_ptr](#) Y, const size_t incY)
fequal : test $X = Y.$
- template<class [Field](#) >
void [faxpby](#) (const [Field](#) &F, const size_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) X, const size_t incX, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) Y, const size_t incY)
faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template<typename [Field](#) , class Cut , class Param >
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size_t N, typename [Field::ConstElement_ptr](#) X, const size_t incX, typename [Field::ConstElement_ptr](#) Y, const size_t incY, const [ParSeqHelper::Parallel](#)< Cut, Param > par)
- template<class [Field](#) >
void [fswap](#) (const [Field](#) &F, const size_t N, typename [Field::Element_ptr](#) X, const size_t incX, typename [Field::Element_ptr](#) Y, const size_t incY)
fswap : $X \leftrightarrow Y.$
- template<class [Field](#) >
void [fzero](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, const size_t lda)
fzero : $A \leftarrow 0.$
- template<class [Field](#) , class RandIter >
void [frand](#) (const [Field](#) &F, RandIter &G, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, const size_t lda)
frand : $A \leftarrow random.$
- template<class [Field](#) >
bool [fequal](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb)
fequal : test $A = B.$
- template<class [Field](#) >
bool [fiszero](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::ConstElement_ptr](#) A, const size_t lda)
fiszero : test $A = 0.$
- template<class [Field](#) >
void [fidentity](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, const size_t lda, const typename [Field::Element](#) &d)
creates a diagonal matrix
- template<class [Field](#) >
void [fidentity](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, const size_t lda)
creates a diagonal matrix
- template<class [Field](#) , class OtherElement_ptr >
void [finit](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, const size_t lda)
finit Initializes A in $F^S.$
- template<class [Field](#) , class OtherElement_ptr >
void [fconvert](#) (const [Field](#) &F, const size_t m, const size_t n, OtherElement_ptr A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb)
fconvert $A \leftarrow B \bmod F.$
- template<class [Field](#) >
void [fnegin](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, const size_t lda)
fnegin $A \leftarrow -A.$

- template<class [Field](#) >
void [fneg](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::ConstElement_ptr](#) B, const size_t ldb, typename [Field::Element_ptr](#) A, const size_t lda)
$$\text{fneg } A \leftarrow -B.$$
- template<class [Field](#) >
void [faxpby](#) (const [Field](#) &F, const size_t m, const size_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) X, const size_t ldx, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) Y, const size_t ldy)
$$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- template<class [Field](#) >
void [fmove](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb)
$$\text{fmove} : A \leftarrow B \text{ and } B \leftarrow 0.$$
- template<class [Field](#) >
size_t [bitsize](#) (const [Field](#) &F, size_t M, size_t N, const typename [Field::ConstElement_ptr](#) A, size_t lda)
bitsize: Computes the largest bitsize of the matrix' coefficients.
- template<> size_t [bitsize](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > > (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, size_t M, size_t N, const [Givaro::Integer](#) *A, size_t lda)
- template<class [Field](#) >
void [ftrmv](#) (const [Field](#) &F, const [FFLAS_UPLO](#) Uplo, const [FFLAS_TRANSPOSE](#) TransA, const [FFLAS_DIAG](#) Diag, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) X, int incX)
ftrsm: TRIangular Matrix Vector prodcut Computes $X \leftarrow \text{op}(A)X$
- template<class [Field](#) >
void [ftrsm](#) (const [Field](#) &F, const [FFLAS_SIDE](#) Side, const [FFLAS_UPLO](#) Uplo, const [FFLAS_TRANSPOSE](#) TransA, const [FFLAS_DIAG](#) Diag, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb)
ftrsm: TRIangular System solve with Matrix.
- template<typename [Field](#) >
[Field::Element_ptr](#) [pfgemm](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, size_t numthreads=0)
- template<class [Field](#) >
[Field::Element](#) * [pfgemm_1D_rec](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, const typename [Field::Element_ptr](#) A, const size_t lda, const typename [Field::Element_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element](#) *C, const size_t ldc, size_t seuil)
- template<class [Field](#) >
[Field::Element](#) * [pfgemm_2D_rec](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, const typename [Field::Element_ptr](#) A, const size_t lda, const typename [Field::Element_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element](#) *C, const size_t ldc, size_t seuil)
- template<class [Field](#) >
[Field::Element](#) * [pfgemm_3D_rec](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, const typename [Field::Element_ptr](#) A, const size_t lda, const typename [Field::Element_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, size_t seuil, size_t *x)
- template<class [Field](#) >
[Field::Element_ptr](#) [pfgemm_3D_rec2](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, const typename [Field::Element_ptr](#) A, const size_t lda, const typename [Field::Element_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, size_t seuil, size_t *x)
- template<class [Field](#) , class [ModeTrait](#) , class [Strat](#) , class [Param](#) >
std::enable_if<!std::is_same< [ModeTrait](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) > >::value, typename [Field::Element_ptr](#) >::type [fgemm](#) (const [Field](#) &F, const [FFLAS::FFLAS_TRANSPOSE](#)

- ta, const FFLAS::FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat, Param > > &H)
- template<class Field, class Cut, class Param >
Field::Element_ptr ftrsm (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_TRANSPOSE TA, const FFLAS::FFLAS_DIAG Diag, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &H)
 - template<class Field, class Cut, class Param >
Field::Element_ptr ftrsm (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_UPLO UpLo, const FFLAS::FFLAS_TRANSPOSE TA, const FFLAS::FFLAS_DIAG Diag, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > &H)
 - template<class Field, class SM >
void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)
 - template<class Field, class SM >
void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)
 - template<class Field, class IndexT >
void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
 - template<class Field, class IndexT >
void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
 - template<class Field >
void sparse_delete (const Sparse< Field, SparseMatrix_t::COO > &A)
 - template<class Field >
void sparse_delete (const Sparse< Field, SparseMatrix_t::COO_ZO > &A)
 - template<class Field, class IndexT >
void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
 - template<class Field, class IndexT >
void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
 - template<class Field >
void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)
 - template<class Field >
void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)
 - template<class Field >
std::ostream & sparse_print (std::ostream &os, const Sparse< Field, SparseMatrix_t::CSR > &A)
 - template<class IndexT >
void sparse_init (const Givaro::Modular< Givaro::Integer > &F, Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
 - template<class IndexT >
void sparse_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)
 - template<class IndexT, size_t RECINT_SIZE>
void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)

- `template<class IndexT, size_t RECINT_SIZE>`
`void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)`
- `template<class Field, class IndexT >`
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)`
- `template<class Field >`
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)`
- `template<class Field, class IndexT >`
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)`
- `template<class Field >`
`void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`
`void sparse_delete (const Sparse< Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field, class IndexT >`
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::HYB_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<typename _Field >`
`std::ostream & operator<< (std::ostream &os, const Sparse< _Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field, bool sorted = true, bool read_integer = false>`
`void readSmsFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)`
- `template<class Field >`
`void readSprFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)`
- `template<class T >`
`std::enable_if< std::is_integral< T >::value, int > getDataType ()`
- `template<class T >`
`std::enable_if< std::is_floating_point< T >::value, int > getDataType ()`
- `template<class T >`
`std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()`
- `template<class T >`
`int getDataType ()`
- `template<class Field >`
`void readMachineType (const Field &F, typename Field::Element &modulo, typename Field::Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)`

- `template<class Field >`
`void readDnsFormat (const std::string &path, const Field &F, index_t &rowdim, index_t &coldim, typename Field::Element_ptr &val)`
- `template<class Field >`
`void writeDnsFormat (const std::string &path, const Field &F, const index_t &rowdim, const index_t &coldim, typename Field::Element_ptr A, index_t ldA)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag)`
- `template<class Field >`
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field >`
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL_ZO > &A)`
- `template<class Field >`
`void sparse_print (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field, class IndexT >`
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz, uint64_t sigma=0)`
- `template<class Field, class IndexT >`
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class It >`
`double computeDeviation (It begin, It end)`
- `template<class Field >`
`StatsMatrix getStat (const Field &F, const index_t *row, const index_t *col, typename Field::ConstElement_ptr val, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<> void fflas_delete (FFPACK::rns_double_elt_ptr A)`
- `template<> void fflas_delete (FFPACK::rns_double_elt_cstptr A)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`
`void finit_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`
`void finit_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`
`void fconvert_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`
- `template<typename RNS >`
`void fconvert_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`
`void finit_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::Element_ptr A)`
- `template<typename RNS >`
`void fconvert_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A)`
- `template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)`

- $\text{freduce } x \leftarrow x \bmod F.$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, const `FFLAS_ELT` *Y, const `size_t` incY, `FFLAS_ELT` *X, const `size_t` incX)
- $\text{freduce } x \leftarrow y \bmod F.$
- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, const `FFLAS_ELT` *Y, const `size_t` incY, `FFLAS_ELT` *X, const `size_t` incX)
- $\text{finit } x \leftarrow y \bmod F.$
- template `INST_OR_DECL` void `fconvert` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, `FFLAS_ELT` *X, const `size_t` incX, const `FFLAS_ELT` *Y, const `size_t` incY)
- $\text{fconvert } x \leftarrow y \bmod F.$
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, `FFLAS_ELT` *X, const `size_t` incX)
- $\text{fnegin } x \leftarrow -x.$
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, const `FFLAS_ELT` *Y, const `size_t` incY, `FFLAS_ELT` *X, const `size_t` incX)
- $\text{fneg } x \leftarrow -y.$
- template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, `FFLAS_ELT` *X, const `size_t` incX)
- $\text{fzero} : A \leftarrow 0.$
- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, const `FFLAS_ELT` *X, const `size_t` incX)
- $\text{fiszero} : \text{test } X = 0.$
- template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, const `FFLAS_ELT` *X, const `size_t` incX, const `FFLAS_ELT` *Y, const `size_t` incY)
- $\text{fequal} : \text{test } X = Y.$
- template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` N, const `FFLAS_ELT` *Y, const `size_t` incY, `FFLAS_ELT` *X, const `size_t` incX)
- $\text{fassign} : x \leftarrow y.$
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, const `FFLAS_ELT` alpha, `FFLAS_ELT` *X, const `size_t` incX)
- $\text{fscaln } x \leftarrow \alpha \cdot x.$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *X, const `size_t` incX, `FFLAS_ELT` *Y, const `size_t` incY)
- $\text{fscal } y \leftarrow \alpha \cdot x.$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *X, const `size_t` incX, `FFLAS_ELT` *Y, const `size_t` incY)
- $\text{faxpy} : y \leftarrow \alpha \cdot x + y.$
- template `INST_OR_DECL` `FFLAS_ELT` `fdot` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` N, const `FFLAS_ELT` *X, const `size_t` incX, const `FFLAS_ELT` *Y, const `size_t` incY)
- $\text{faxpy} : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template `INST_OR_DECL` void `fswap` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` N, `FFLAS_ELT` *X, const `size_t` incX, `FFLAS_ELT` *Y, const `size_t` incY)
- $\text{fswap} : X \leftrightarrow Y.$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` N, const `FFLAS_ELT` *A, const `size_t` inca, const `FFLAS_ELT` *B, const `size_t` incb, `FFLAS_ELT` *C, const `size_t` incc)
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` N, const `FFLAS_ELT` *A, const `size_t` inca, const `FFLAS_ELT` *B, const `size_t` incb, `FFLAS_ELT` *C, const `size_t` incc)
- template `INST_OR_DECL` void `faddn` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` N, const `FFLAS_ELT` *B, const `size_t` incb, `FFLAS_ELT` *C, const `size_t` incc)
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` N, const `FFLAS_ELT` *A, const `size_t` inca, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *B, const `size_t` incb, `FFLAS_ELT` *C, const `size_t` incc)

- template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *A, const size_t lda)
 $fassign : A \leftarrow B.$
- template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, `FFLAS_ELT` *A, const size_t lda)
 $fzero : A \leftarrow 0.$
- template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` *B, const size_t ldb)
 $fequal : \text{test } A = B.$
- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` *A, const size_t lda)
 $fiszero : \text{test } A = 0.$
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` &d)
creates a diagonal matrix
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, `FFLAS_ELT` *A, const size_t lda)
creates a diagonal matrix
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, `FFLAS_ELT` *A, const size_t lda)
 $freduce A \leftarrow A \bmod F.$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *A, const size_t lda)
 $freduce A \leftarrow B \bmod F.$
- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *A, const size_t lda)
 $finit A \leftarrow B \bmod F.$
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, `FFLAS_ELT` *A, const size_t lda)
 $fnegin A \leftarrow -A.$
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *A, const size_t lda)
 $fneg A \leftarrow -B.$
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` *A, const size_t lda)
 $fscaln A \leftarrow a \cdot A.$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, `FFLAS_ELT` *B, const size_t ldb)
 $fscal B \leftarrow a \cdot A.$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *X, const size_t ldX, `FFLAS_ELT` *Y, const size_t ldY)
 $faxpy : y \leftarrow \alpha \cdot x + y.$
- template `INST_OR_DECL` void `fmove` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t m, const size_t n, `FFLAS_ELT` *A, const size_t lda, `FFLAS_ELT` *B, const size_t ldb)
 $fmove : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t M, const size_t N, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *C, const size_t ldc)
fadd : matrix addition.
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t M, const size_t N, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *C, const size_t ldc)
fsub : matrix subtraction.

- template `INST_OR_DECL` void `fsubin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t M, const size_t N, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *C, const size_t ldc)

$$fsubin\ C = C - B$$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t M, const size_t N, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *C, const size_t ldc)

$$fadd : \text{matrix addition with scaling.}$$
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t M, const size_t N, const `FFLAS_ELT` *B, const size_t ldb, `FFLAS_ELT` *C, const size_t ldc)

$$faddin$$
- template `INST_OR_DECL` `FFLAS_ELT` * `fgemv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` TransA, const size_t M, const size_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` *X, const size_t incX, const `FFLAS_ELT` beta, `FFLAS_ELT` *Y, const size_t incY)

$$finite\ prime\ FFLAS_FIELD<FFLAS_ELT>\ GEneral\ Matrix\ Vector\ multiplication.$$
- template `INST_OR_DECL` void `fger` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t M, const size_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *x, const size_t incx, const `FFLAS_ELT` *y, const size_t incy, `FFLAS_ELT` *A, const size_t lda)

$$fger: \text{rank one update of a general matrix}$$
- template `INST_OR_DECL` void `ftsv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size_t N, const `FFLAS_ELT` *A, const size_t lda, `FFLAS_ELT` *X, int incX)

$$ftsv: \text{TRIangular System solve with Vector Computes } X \leftarrow op(A^{-1})X$$
- template `INST_OR_DECL` void `ftdsm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_SIDE` Side, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size_t M, const size_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, `FFLAS_ELT` *B, const size_t ldb)

$$ftdsm: \text{TRIangular System solve with Matrix.}$$
- template `INST_OR_DECL` void `ftmmm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_SIDE` Side, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size_t M, const size_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, `FFLAS_ELT` *B, const size_t ldb)

$$ftmmm: \text{TRIangular Matrix Multiply.}$$
- template `INST_OR_DECL` `FFLAS_ELT` * `fgemm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size_t m, const size_t n, const size_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` *B, const size_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` *C, const size_t ldc)

$$fgemm: \text{Field GEneral Matrix Multiply.}$$
- template `INST_OR_DECL` `FFLAS_ELT` * `fgemm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size_t m, const size_t n, const size_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` *B, const size_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` *C, const size_t ldc, const `ParSeqHelper::Sequential` seq)
- template `INST_OR_DECL` `FFLAS_ELT` * `fgemm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size_t m, const size_t n, const size_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` *B, const size_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` *C, const size_t ldc, const `ParSeqHelper::Parallel`< `CuttingStrategy::Recursive`, `StrategyParameter::TwoDAdaptive` > par)
- template `INST_OR_DECL` `FFLAS_ELT` * `fgemm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size_t m, const size_t n, const size_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` *B, const size_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` *C, const size_t ldc, const `ParSeqHelper::Parallel`< `CuttingStrategy::Block`, `StrategyParameter::Threads` > par)
- template `INST_OR_DECL` `FFLAS_ELT` * `fsquare` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` ta, const size_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` *A, const size_t lda, const `FFLAS_ELT` beta, `FFLAS_ELT` *C, const size_t ldc)

fsquare: Squares a matrix.

- `template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>`
`void BlockCuts (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>`
`void BlockCuts (size_t &rowBlockSize, size_t &colBlockSize, size_t &lastRBS, size_t &lastCBS, size_t &changeRBS, size_t &changeCBS, size_t &numRowBlock, size_t &numColBlock, size_t m, size_t n, const size_t numthreads)`
- `template<class Field >`
`void pfzero (const Field &F, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)`
- `template<class Field, class RandIter >`
`void pfrand (const Field &F, RandIter &G, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)`
- `template<class Field, class Cut, class Param >`
`Field::Element &fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element &d, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class Field, class AlgoT, class FieldTrait >`
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDAdaptive > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > > &H)`

- `template<class Field, class AlgoT, class FieldTrait >`
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`
`Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t`
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoD > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`
`Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`
`Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t`
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`
`Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t`
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace >`
`> &H)`
- `template<class Field, class AlgoT, class FieldTrait >`
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t`
`n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t`
`lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta,`
`typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel<`
`CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)`
- `template<class Field, class AlgoT, class FieldTrait, class Cut >`
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t`
`n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t`
`lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta,`
`typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel<`
`CuttingStrategy::Row, Cut > > &H)`
- `void parseArguments (int argc, char **argv, Argument *args, bool printDefaults=true)`
- `std::ostream & writeCommandString (std::ostream &os, Argument *args, const char *programName=nullptr)`
writes the values of all arguments, preceded by the programName
- `template<class Field >`
`std::ostream & WriteMatrix (std::ostream &c, const Field &F, size_t m, size_t n, typename Field::ConstElement_ptr`
`A, size_t lda, FFLAS_FORMAT format, bool column_major)`
WriteMatrix: write a matrix to an output stream.
- `void preamble (std::ifstream &if, FFLAS_FORMAT &format)`
- `template<class Field >`
`Field::Element_ptr ReadMatrix (std::ifstream &if, Field &F, size_t &m, size_t &n, typename Field::Element_ptr`
`&A, FFLAS_FORMAT format=FflasAuto)`
ReadMatrix: read a matrix from an input stream.
- `template<class Field >`
`Field::Element_ptr ReadMatrix (const std::string &matrix_file, Field &F, size_t &m, size_t &n, typename`
`Field::Element_ptr &A, FFLAS_FORMAT format=FflasAuto)`
ReadMatrix: read a matrix from a file.
- `template<class Field >`
`void WriteMatrix (std::string &matrix_file, const Field &F, int m, int n, typename Field::ConstElement_ptr A,`
`size_t lda, FFLAS_FORMAT format=FflasDense, bool column_major=false)`
WriteMatrix: write a matrix to a file.
- `std::ostream & WritePermutation (std::ostream &c, const size_t *P, size_t N)`
WritePermutation: write a permutation matrix to an output stream.
- `template<class Element >`
`bool alignable ()`
- `template<> bool alignable< Givaro::Integer * > ()`

- `template<class Field >`
`Field::Element_ptr fflas_new` (const Field &F, const size_t m, const Alignment align=Alignment::DEFAULT)
- `template<class Field >`
`Field::Element_ptr fflas_new` (const Field &F, const size_t m, const size_t n, const Alignment align=Alignment::DEFAULT)
- `template<class Element >`
`Element * fflas_new` (const size_t m, const Alignment align=Alignment::DEFAULT)
- `template<class Element_ptr >`
`void fflas_delete` (Element_ptr A)
- `template<class Ptr, class ... Args>`
`void fflas_delete` (Ptr p, Args ... args)
- `void prefetch` (const int64_t *)
- `void getTLBSize` (int &tlb)
- `void queryCacheSizes` (int &l1, int &l2, int &l3)
- `int queryL1CacheSize` ()
- `int queryTopLevelCacheSize` ()
- `uint64_t getSeed` ()

11.1.1 Typedef Documentation

11.1.1.1 Checker_fgemm

```
template<class Field >
using Checker_fgemm = FFLAS::Checker_Empty<Field>
```

11.1.1.2 Checker_ftrsm

```
template<class Field >
using Checker_ftrsm = FFLAS::Checker_Empty<Field>
```

11.1.1.3 ForceCheck_fgemm

```
template<class Field >
using ForceCheck_fgemm = CheckerImplem_fgemm<Field>
```

11.1.1.4 ForceCheck_ftrsm

```
template<class Field >
using ForceCheck_ftrsm = CheckerImplem_ftrsm<Field>
```

11.1.1.5 ZOSparseMatrix

```
using ZOSparseMatrix = std::true_type
```

11.1.1.6 NotZOSparseMatrix

```
using NotZOSparseMatrix = std::false_type
```

11.1.1.7 SimdSparseMatrix

```
using SimdSparseMatrix = std::true_type
```

11.1.1.8 NoSimdSparseMatrix

```
using NoSimdSparseMatrix = std::false_type
```

11.1.1.9 MKLSparseMatrixFormat

```
using MKLSparseMatrixFormat = std::true_type
```

11.1.1.10 NotMKLSparseMatrixFormat

```
using NotMKLSparseMatrixFormat = std::false_type
```

11.1.1.11 has_plus

```
template<class T >
using has_plus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_plus_impl<T> >::type
```

11.1.1.12 has_minus

```
template<class T >
using has_minus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_minus_impl<T> >::type
```

11.1.1.13 has_equal

```
template<class T >
using has_equal = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
std::is_copy_assignable<T> >::type
```

11.1.1.14 has_plus_eq

```
template<class T >
using has_plus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_plus_eq_impl<T> >::type
```

11.1.1.15 has_minus_eq

```
template<class T >
using has_minus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_minus_eq_impl<T> >::type
```

11.1.1.16 has_mul

```
template<class T >
using has_mul = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_impl<T>
>::type
```

11.1.1.17 has_mul_eq

```
template<class T >
using has_mul_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,
has_mul_eq_impl<T> >::type
```

11.1.1.18 Timer

```
typedef Givaro::Timer Timer
```

11.1.1.19 BaseTimer

```
typedef Givaro::BaseTimer BaseTimer
```

11.1.1.20 UserTimer

```
typedef Givaro::UserTimer UserTimer
```

11.1.1.21 SysTimer

```
typedef Givaro::SysTimer SysTimer
```

11.1.2 Enumeration Type Documentation**11.1.2.1 FFLAS_ORDER**

```
enum FFLAS_ORDER
```

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major

11.1.2.2 FFLAS_TRANSPOSE

enum [FFLAS_TRANSPOSE](#)

Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.

11.1.2.3 FFLAS_UPLO

enum [FFLAS_UPLO](#)

Is triangular matrix's shape upper ?

Enumerator

FflasUpper	Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$)
FflasLower	Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$)

11.1.2.4 FFLAS_DIAG

enum [FFLAS_DIAG](#)

Is the triangular matrix implicitly unit diagonal ?

Enumerator

FflasNonUnit	Triangular matrix has an explicit arbitrary diagonal.
FflasUnit	Triangular matrix has an implicit unit diagonal ($T_{i,i} = 1$)

11.1.2.5 FFLAS_SIDE

enum [FFLAS_SIDE](#)

On what side ?

Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the righth.

11.1.2.6 FFLAS_BASE

enum `FFLAS_BASE`

FFLAS_BASE determines the type of the element representation for Matrix Mult kernel.

(deprecated, should not be used)

Enumerator

FflasDouble	to use the double precision BLAS
FflasFloat	to use the single precison BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

11.1.2.7 number_kind

enum `number_kind`

Enumerator

zero	
one	
mone	
other	

11.1.2.8 SparseMatrix_t

enum class `SparseMatrix_t` [strong]

Enumerator

CSR	
CSR_ZO	
CSC	
CSC_ZO	
COO	
COO_ZO	
ELL	
ELL_ZO	
SELL	
SELL_ZO	
ELL_simd	
ELL_simd_ZO	
CSR_HYB	
HYB_ZO	

11.1.2.9 FFLAS_FORMAT

enum [FFLAS_FORMAT](#)

Enumerator

FflasAuto	
FflasDense	
FflasSMS	
FflasBinary	
FflasMath	
FflasMaple	
FflasSageMath	

11.1.3 Function Documentation

11.1.3.1 InfNorm()

```
Givaro::Integer InfNorm (
    const size_t M,
    const size_t N,
    const Givaro::Integer * A,
    const size_t lda ) [inline]
```

11.1.3.2 min3()

```
template<class T >
const T & min3 (
    const T & m,
    const T & n,
    const T & k )
```

11.1.3.3 max3()

```
template<class T >
const T & max3 (
    const T & m,
    const T & n,
    const T & k )
```

11.1.3.4 min4()

```
template<class T >
const T & min4 (
    const T & m,
    const T & n,
    const T & k,
    const T & l )
```

11.1.3.5 max4()

```
template<class T >
const T & max4 (
    const T & m,
    const T & n,
    const T & k,
    const T & l )
```

11.1.3.6 fadd() [1/8]

```
template<class Field >
void fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

11.1.3.7 faddin() [1/4]

```
template<class Field >
void faddin (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

11.1.3.8 fsub() [1/4]

```
template<class Field >
void fsub (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

11.1.3.9 fsubin() [1/3]

```
template<class Field >
void fsubin (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

11.1.3.10 fadd() [2/8]

```
template<class Field >
void fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

Todo optimise here

11.1.3.11 pfadd()

```
template<class Field >
void pfadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t numths )
```

11.1.3.12 pfsub()

```
template<class Field >
void pfsub (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t numths )
```

11.1.3.13 pfaddin()

```
template<class Field >
void pfaddin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numths )
```

11.1.3.14 pfsubin()

```
template<class Field >
void pfsubin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numths )
```

11.1.3.15 fadd() [3/8]

```
template<class Field >
void fadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )
```

fadd : matrix addition.

Computes $C = A + B$.

Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

11.1.3.16 fsub() [2/4]

```

template<class Field >
void fsub (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fsub : matrix subtraction.

Computes $C = A - B$.

Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

11.1.3.17 faddin() [2/4]

```

template<class Field >
void faddin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

faddin

11.1.3.18 fsubin() [2/3]

```

template<class Field >
void fsubin (
    const Field & F,
    const size_t M,
    const size_t N,

```

```

typename Field::ConstElement_ptr B,
const size_t ldb,
typename Field::Element_ptr C,
const size_t ldc )

```

fsubin $C = C - B$

11.1.3.19 fadd() [4/8]

```

template<class Field >
void fadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fadd : matrix addition with scaling.

Computes $C = A + \text{alpha } B$.

Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size MxN
<i>lda</i>	leading dimension of A
<i>alpha</i>	some scalar
<i>B</i>	dense matrix of size MxN
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size MxN
<i>ldc</i>	leading dimension of C

11.1.3.20 fassign() [1/10]

```

template<class Field >
void fassign (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX ) [inline]

```

fassign : $x \leftarrow y$.

X is preallocated

Todo variant for triangular matrix

Parameters

	F	field
	N	size of the vectors
out	X	vector in F
	$incX$	stride of X
in	Y	vector in F
	$incY$	stride of Y

11.1.3.21 fassign() [2/10]

```
template<>
void fassign (
    const Givaro::Modular< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

11.1.3.22 fassign() [3/10]

```
template<>
void fassign (
    const Givaro::ModularBalanced< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

11.1.3.23 fassign() [4/10]

```
template<>
void fassign (
    const Givaro::ZRing< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

11.1.3.24 fassign() [5/10]

```
template<>
void fassign (
    const Givaro::Modular< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

11.1.3.25 fassign() [6/10]

```
template<>
void fassign (
    const Givaro::ModularBalanced< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

11.1.3.26 fassign() [7/10]

```
template<>
void fassign (
    const Givaro::ZRing< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

11.1.3.27 fassign() [8/10]

```
template<class Field >
void fassign (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

$\text{fassign} : A \leftarrow B.$

Parameters

F	field
m	number of rows to copy
n	number of cols to copy
A	matrix in F
lda	stride of A
B	vector in F
ldb	stride of B

11.1.3.28 faxpy() [1/6]

```
template<class Field >
void faxpy (
```

```

const Field & F,
const size_t N,
const typename Field::Element alpha,
typename Field::ConstElement_ptr X,
const size_t incX,
typename Field::Element_ptr Y,
const size_t incY ) [inline]

```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

Parameters

	F	field
	N	size of the vectors
	α	scalar
in	X	vector in F
	incX	stride of X
in, out	Y	vector in F
	incY	stride of Y

11.1.3.29 faxpy() [2/6]

```

template<>
void faxpy (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]

```

11.1.3.30 faxpy() [3/6]

```

template<>
void faxpy (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]

```

11.1.3.31 faxpy() [4/6]

```

template<class Field >
void faxpy (
    const Field & F,
    const size_t m,
    const size_t n,

```

```

const typename Field::Element alpha,
typename Field::ConstElement_ptr X,
const size_t ldx,
typename Field::Element_ptr Y,
const size_t ldy ) [inline]

```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

Parameters

	F	field
	m	row dimension
	n	column dimension
	α	scalar
in	X	vector in F
	ldx	leading dimension of X
in, out	Y	vector in F
	ldy	leading dimension of Y

11.1.3.32 fdot() [1/11]

```

template<class Field >
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]

```

11.1.3.33 fdot() [2/11]

```

template<class Field >
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DelayedTag & MT ) [inline]

```

11.1.3.34 fdot() [3/11]

```

template<>
Givaro::DoubleDomain::Element fdot (
    const Givaro::DoubleDomain & ,
    const size_t N,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]

```

11.1.3.35 fdot() [4/11]

```
template<>
Givaro::FloatDomain::Element fdot (
    const Givaro::FloatDomain & ,
    const size_t N,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

11.1.3.36 fdot() [5/11]

```
template<class Field , class T >
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::ConvertTo< T > & MT ) [inline]
```

11.1.3.37 fdot() [6/11]

```
template<class Field >
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultBoundedTag & dbt ) [inline]
```

11.1.3.38 fdot() [7/11]

```
template<class Field >
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    const ParSeqHelper::Sequential seq ) [inline]
```

11.1.3.39 fdot() [8/11]

```
template<class Field >
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY ) [inline]
```

fdot: dot product $x^T y$.

Parameters

F	field
N	size of the vectors
X	vector in F
$incX$	stride of X
Y	vector in F
$incY$	stride of Y

11.1.3.40 fgemm() [1/23]

```
template<class Field >
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFl
>, ParSeqHelper::Sequential > & H ) [inline]
```

11.1.3.41 fgemm() [2/23]

```
template<typename Field >
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
```

```

    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Sequential seq ) [inline]

```

11.1.3.42 fgemm() [3/23]

```

template<typename Field , class Cut , class Param >
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

11.1.3.43 fgemm() [4/23]

```

template<typename Field >
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]

```

fgemm: Field GEneral Matrix Multiply.

Computes $C = \alpha \text{op}(A) \times \text{op}(B) + \beta C$ Automatically set Winograd recursion level

Parameters

<i>F</i>	field.
<i>ta</i>	if <code>ta==FflasTrans</code> then $\text{op}(A) = A^t$, else $\text{op}(A) = A$,
<i>tb</i>	same for matrix B
<i>m</i>	see A
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	$\text{op}(A)$ is $m \times k$
<i>B</i>	$\text{op}(B)$ is $k \times n$
<i>C</i>	<i>C</i> is $m \times n$
<i>lda</i>	leading dimension of A
<i>ldb</i>	leading dimension of B
<i>ldc</i>	leading dimension of C
<i>w</i>	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of w .

Warning

α must be invertible

11.1.3.44 fgemm() [5/23]

```
template<typename Field , class ModeT , class ParSeq >
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > & H ) [inline]
```

11.1.3.45 fgemm() [6/23]

```
template<class Field >
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
```



```

    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential
> & H ) [inline]

```

11.1.3.46 fsquare() [1/6]

```

template<class Field >
Field::Element_ptr fsquare (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]

```

fsquare: Squares a matrix.

compute $C \leftarrow \alpha \text{op}(A) \text{op}(A) + \beta C$ over a Field F Avoid the conversion of B

Parameters

<i>ta</i>	if $ta == \text{FflasTrans}$, $\text{op}(A) = A^T$.
<i>F</i>	field
<i>n</i>	size of A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	dense matrix of size $n \times n$
<i>lda</i>	leading dimension of A
<i>C</i>	dense matrix of size $n \times n$
<i>ldc</i>	leading dimension of C

Bug why double ?

11.1.3.47 fsquare() [2/6]

```

template<>
double * fsquare (
    const Givaro::ModularBalanced< double > & F,

```

```

    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double beta,
    double * C,
    const size_t ldc ) [inline]

```

11.1.3.48 fsquare() [3/6]

```

template<>
float * fsquare (
    const Givaro::ModularBalanced< float > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const float alpha,
    const float * A,
    const size_t lda,
    const float beta,
    float * C,
    const size_t ldc ) [inline]

```

11.1.3.49 fsquare() [4/6]

```

template<>
double * fsquare (
    const Givaro::Modular< double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double beta,
    double * C,
    const size_t ldc ) [inline]

```

11.1.3.50 fsquare() [5/6]

```

template<>
float * fsquare (
    const Givaro::Modular< float > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const float alpha,
    const float * A,
    const size_t lda,
    const float beta,
    float * C,
    const size_t ldc ) [inline]

```

11.1.3.51 fgemm() [7/23]

```

template<typename RNS , typename ParSeqTrait >
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > & H ) [inline]

```

11.1.3.52 fgemm() [8/23]

```

template<typename RNS >
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Sequential > & H ) [inline]

```

11.1.3.53 fgemm() [9/23]

```

template<typename RNS , typename ParSeqTrait >
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,

```

```

    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads
    >, ParSeqTrait > > & H ) [inline]

```

11.1.3.54 fgemm() [10/23]

```

template<typename RNS , typename Cut , typename Param >
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Parallel< Cut, Param > > & H ) [inline]

```

11.1.3.55 fgemm() [11/23]

```

template<class ParSeq >
Givaro::Integer * fgemm (
    const Givaro::ZRing< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
    ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]

```

11.1.3.56 fgemm() [12/23]

```

template<typename RNS , class ModeT >
RNS::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename RNS::Element alpha,
    typename RNS::ConstElement_ptr Ad,
    const size_t lda,
    typename RNS::ConstElement_ptr Bd,
    const size_t ldb,
    const typename RNS::Element beta,
    typename RNS::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential
> & H ) [inline]

```

11.1.3.57 fgemm() [13/23]

```

template<typename RNS >
RNS::Element_ptr fgemm (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename RNS::Element alpha,
    typename RNS::ConstElement_ptr Ad,
    const size_t lda,
    typename RNS::ConstElement_ptr Bd,
    const size_t ldb,
    const typename RNS::Element beta,
    typename RNS::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > & H ) [inline]

```

11.1.3.58 fgemm() [14/23]

```

Givaro::Integer * fgemm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,

```

```

        const size_t ldb,
        const Givaro::Integer beta,
        Givaro::Integer * C,
        const size_t ldc,
        MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]

```

11.1.3.59 fgemm() [15/23]

```

template<class ParSeq >
Givaro::Integer * fgemm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    const Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]

```

11.1.3.60 fgemm() [16/23]

```

template<size_t K1, size_t K2, class ParSeq >
RecInt::ruint< K1 > * fgemm (
    const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const RecInt::ruint< K1 > alpha,
    const RecInt::ruint< K1 > * A,
    const size_t lda,
    const RecInt::ruint< K1 > * B,
    const size_t ldb,
    RecInt::ruint< K1 > beta,
    RecInt::ruint< K1 > * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]

```

11.1.3.61 fgemm() [17/23]

```

template<class Field , class ModeT >
Field::Element_ptr fgemm (

```

```

const Field & F,
const FFLAS_TRANSPOSE ta,
const FFLAS_TRANSPOSE tb,
const size_t m,
const size_t n,
const size_t k,
const typename Field::Element alpha,
typename Field::ConstElement_ptr A,
const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
MMHelper< Field, MMHelperAlgo::Winograd, ModeT > & H ) [inline]

```

11.1.3.62 fgemm() [18/23]

```

template<class Field , class ModeT , class Cut , class Param >
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut,
Param > > & H ) [inline]

```

11.1.3.63 fgemv() [1/19]

```

template<class Field >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H ) [inline]

```

11.1.3.64 fgemv() [2/19]

```

template<class Field >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]

```

11.1.3.65 fgemv() [3/19]

```

template<class Field >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]

```

11.1.3.66 fgemv() [4/19]

```

template<class Field >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]

```


11.1.3.67 fgemv() [5/19]

```
template<class Field >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE TransA,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

finite prime Field GEneral Matrix Vector multiplication.

Computes $Y \leftarrow \alpha \text{op}(A)X + \beta Y$.

Parameters

	F	field
	$TransA$	if $TransA == FflasTrans$ then $\text{op}(A) = A^t$.
	M	rows
	N	cols
	$alpha$	scalar
	A	dense matrix of size $M \times N$
	lda	leading dimension of A
	X	dense vector of size N
	$incX$	stride of X
	$beta$	scalar
out	Y	dense vector of size M
	$incY$	stride of Y

11.1.3.68 fgemv() [6/19]

```
Givaro::ZRing< int64_t >::Element_ptr fgemv (
    const Givaro::ZRing< int64_t > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const int64_t alpha,
    const int64_t * A,
    const size_t lda,
    const int64_t * X,
    const size_t incX,
    const int64_t beta,
    int64_t * Y,
    const size_t incY,
    MMHelper< Givaro::ZRing< int64_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

11.1.3.69 fgemv() [7/19]

```
Givaro::DoubleDomain::Element_ptr fgemv (
    const Givaro::DoubleDomain & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const Givaro::DoubleDomain::Element alpha,
    const Givaro::DoubleDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::DoubleDomain::ConstElement_ptr X,
    const size_t incX,
    const Givaro::DoubleDomain::Element beta,
    Givaro::DoubleDomain::Element_ptr Y,
    const size_t incY,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

11.1.3.70 fgemv() [8/19]

```
template<class Field >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]
```

11.1.3.71 fgemv() [9/19]

```
Givaro::FloatDomain::Element_ptr fgemv (
    const Givaro::FloatDomain & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const Givaro::FloatDomain::Element alpha,
    const Givaro::FloatDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::FloatDomain::ConstElement_ptr X,
    const size_t incX,
    const Givaro::FloatDomain::Element beta,
    Givaro::FloatDomain::Element_ptr Y,
    const size_t incY,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]
```

11.1.3.72 fgemv() [10/19]

```

template<class Field , class Cut , class Param >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    ParSeqHelper::Parallel< Cut, Param > & parH )

```

11.1.3.73 fgemv() [11/19]

```

template<class Field >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    ParSeqHelper::Sequential & seqH )

```

11.1.3.74 fgemv() [12/19]

```

FFPACK::rns_double::Element_ptr fgemv (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::ConstElement_ptr X,
    const size_t incX,
    const FFPACK::rns_double::Element beta,
    FFPACK::rns_double::Element_ptr Y,
    const size_t incY,
    MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::Default > & H ) [inline]

```

11.1.3.75 fgemv() [13/19]

```
FFPACK::rns_double::Element_ptr fgemv (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::ConstElement_ptr X,
    const size_t incX,
    const FFPACK::rns_double::Element beta,
    FFPACK::rns_double::Element_ptr Y,
    const size_t incY,
    MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic,
    ModeCategories::DefaultTag > & H ) [inline]
```

11.1.3.76 fgemv() [14/19]

```
Givaro::Integer * fgemv (
    const Givaro::ZRing< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const Givaro::Integer alpha,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * X,
    const size_t ldx,
    Givaro::Integer beta,
    Givaro::Integer * Y,
    const size_t ldy,
    MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
    ElementCategories::RNSElementTag > > & H ) [inline]
```

11.1.3.77 fgemv() [15/19]

```
Givaro::Integer * fgemv (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const Givaro::Integer alpha,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * X,
    const size_t ldx,
    Givaro::Integer beta,
    Givaro::Integer * Y,
    const size_t ldy,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
    ElementCategories::RNSElementTag > > & H ) [inline]
```

11.1.3.78 fgemv() [16/19]

```

template<size_t K1, size_t K2, class ParSeq >
RecInt::ruint< K1 > * fgemv (
    const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const RecInt::ruint< K1 > alpha,
    const RecInt::ruint< K1 > * A,
    const size_t lda,
    const RecInt::ruint< K1 > * X,
    const size_t incx,
    RecInt::ruint< K1 > beta,
    RecInt::ruint< K1 > * Y,
    const size_t incy,
    MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
    ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]

```

11.1.3.79 fger() [1/12]

```

template<class Field >
void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]

```

fger: rank one update of a general matrix

Computes $A \leftarrow \alpha x y^T + A$

Parameters

	F	field
	M	rows
	N	cols
	α	scalar
in, out	A	dense matrix of size MxN and leading dimension lda
	lda	leading dimension of A
	x	dense vector of size M
	$incx$	stride of X
	y	dense vector of size N
	$incy$	stride of Y

11.1.3.80 fger() [2/12]

```

template<class Field >
void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H ) [inline]

```

11.1.3.81 fger() [3/12]

```

template<class Field , class AnyTag >
void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, AnyTag > & H ) [inline]

```

11.1.3.82 fger() [4/12]

```

void fger (
    const Givaro::DoubleDomain & F,
    const size_t M,
    const size_t N,
    const Givaro::DoubleDomain::Element alpha,
    const Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    const Givaro::DoubleDomain::ConstElement_ptr y,
    const size_t incy,
    Givaro::DoubleDomain::Element_ptr A,
    const size_t lda,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

11.1.3.83 fger() [5/12]

```

template<class Field >
void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr x,
    const size_t incx,
    const typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]

```

11.1.3.84 fger() [6/12]

```

void fger (
    const Givaro::FloatDomain & F,
    const size_t M,
    const size_t N,
    const Givaro::FloatDomain::Element alpha,
    const Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    const Givaro::FloatDomain::ConstElement_ptr y,
    const size_t incy,
    Givaro::FloatDomain::Element_ptr A,
    const size_t lda,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

11.1.3.85 fger() [7/12]

```

template<class Field >
void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]

```

11.1.3.86 fger() [8/12]

```

template<class Field >
void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]

```

11.1.3.87 fger() [9/12]

```

void fger (
    const Givaro::Modular< Givaro::Integer > & F,
    const size_t M,
    const size_t N,
    const typename Givaro::Integer alpha,
    typename Givaro::Integer * x,
    const size_t incx,
    typename Givaro::Integer * y,
    const size_t incy,
    typename Givaro::Integer * A,
    const size_t lda,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]

```

11.1.3.88 fger() [10/12]

```

template<typename RNS >
void fger (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t M,
    const size_t N,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::Element_ptr x,
    const size_t incx,
    typename FFPACK::RNSInteger< RNS >::Element_ptr y,
    const size_t incy,
    typename FFPACK::RNSInteger< RNS >::Element_ptr A,
    const size_t lda,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```


11.1.3.89 fger() [11/12]

```

template<typename RNS >
void fger (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t M,
    const size_t N,
    const typename FFPACK::RNSIntegerMod< RNS >::Element alpha,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr x,
    const size_t incx,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y,
    const size_t incy,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A,
    const size_t lda,
    MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > & H ) [inline]

```

11.1.3.90 freduce() [1/10]

```

template<class Field >
void freduce (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )

```

$\text{freduce } x \leftarrow y \bmod F.$

Parameters

F	field
n	size of the vectors
Y	vector of Element
$incY$	stride of Y
X	vector in F
$incX$	stride of X

Bug use cblas_(d)scal when possible

11.1.3.91 freduce() [2/10]

```

template<class Field >
void freduce (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )

```

$\text{freduce } x \leftarrow x \bmod F.$

Parameters

F	field
n	size of the vectors
X	vector in F
$incX$	stride of X

Bug use `cblas_(d)scal` when possible

11.1.3.92 `freduce_constoverride()` [1/2]

```
template<class Field >
void freduce_constoverride (
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr A,
    const size_t incX )
```

11.1.3.93 `finit()` [1/8]

```
template<class Field , class ConstOtherElement_ptr >
void finit (
    const Field & F,
    const size_t n,
    ConstOtherElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )
```

11.1.3.94 `finit()` [2/8]

```
template<class Field >
void finit (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

`finit` Initializes X in F .

Parameters

F	field
n	size of the vectors
X	vector in F
$incX$	stride of X

11.1.3.95 freduce() [3/10]

```
template<class Field >
void freduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

$\text{freduce } A \leftarrow A \bmod F.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.96 pfreduce()

```
template<class Field >
void pfreduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t numths )
```

11.1.3.97 freduce() [4/10]

```
template<class Field >
void freduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

$\text{freduce } A \leftarrow B \bmod F.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A
B	matrix in Element
ldb	stride of B

11.1.3.98 freduce_constoverride() [2/2]

```
template<class Field >
void freduce_constoverride (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

11.1.3.99 finit() [3/8]

```
template<class Field , class OtherElement_ptr >
void finit (
    const Field & F,
    const size_t m,
    const size_t n,
    const OtherElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

$\text{finit } A \leftarrow B \bmod F.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A
B	matrix in OtherElement
ldb	stride of B

11.1.3.100 finit() [4/8]

```
template<class Field >
void finit (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

11.1.3.101 freduce() [5/10]

```
template<>
void freduce (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
    size_t inc ) [inline]
```

11.1.3.102 freduce() [6/10]

```
template<>
void freduce (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    FFPACK::rns_double::Element_ptr A,
    size_t lda ) [inline]
```

11.1.3.103 freivalds()

```
template<class Field >
bool freivalds (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::ConstElement_ptr C,
    const size_t ldc ) [inline]
```

freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.

Randomly Checks $C = \alpha \text{op}(A) \times \text{op}(B)$

Parameters

F	field.
ta	if $ta == \text{FflasTrans}$ then $\text{op}(A) = A^t$, else $\text{op}(A) = A$,
tb	same for matrix B
m	see A
n	see B
k	see A
α	scalar
A	$\text{op}(A)$ is $m \times k$
B	$\text{op}(B)$ is $k \times n$
C	C is $m \times n$
lda	leading dimension of A
ldb	leading dimension of B
ldc	leading dimension of C

11.1.3.104 fscaln() [1/10]

```
template<class Field >
void fscaln (
```

```

const Field & F,
const size_t n,
const typename Field::Element alpha,
typename Field::Element_ptr X,
const size_t incX ) [inline]

```

fscal $x \leftarrow \alpha \cdot x$.

Parameters

F	field
n	size of the vectors
α	scalar
X	vector in F
$incX$	stride of X

Bug use `cblas_(d)scal` when possible

Todo check if comparison with $\pm 1, 0$ is necessary.

11.1.3.105 fscal() [1/10]

```

template<class Field >
void fscal (
    const Field & F,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]

```

fscal $y \leftarrow \alpha \cdot x$.

Parameters

	F	field
	n	size of the vectors
	α	scalar
in	X	vector in F
	$incX$	stride of X
out	Y	vector in F
	$incY$	stride of Y

Bug use `cblas_(d)scal` when possible

Todo check if comparison with $\pm 1, 0$ is necessary.

11.1.3.106 fscal() [2/10]

```
template<>
void fscal (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]
```

11.1.3.107 fscal() [3/10]

```
template<>
void fscal (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]
```

11.1.3.108 fscaln() [2/10]

```
template<>
void fscaln (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]
```

11.1.3.109 fscaln() [3/10]

```
template<>
void fscaln (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]
```

11.1.3.110 fscaln() [4/10]

```
template<class Field >
void fscaln (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

$\text{fscaln } A \leftarrow a \cdot A.$

Parameters

F	field
m	number of rows
n	number of cols
α	homotecie scalar
A	matrix in F
lda	stride of A

11.1.3.111 fscal() [4/10]

```
template<class Field >
void fscal (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]
```

$\text{fscal } B \leftarrow a \cdot A.$

Parameters

	F	field
	m	number of rows
	n	number of cols
	α	homotecie scalar
in	A	matrix in F
	lda	stride of A
out	B	matrix in F
	ldb	stride of B

11.1.3.112 fscaln() [5/10]

```
template<>
void fscaln (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t inc ) [inline]
```

11.1.3.113 fscal() [5/10]

```
template<>
void fscal (
```



```

const FFPACK::RNSInteger< FFPACK::rns_double > & F,
const size_t n,
const FFPACK::rns_double::Element alpha,
FFPACK::rns_double::ConstElement_ptr A,
const size_t Ainc,
FFPACK::rns_double::Element_ptr B,
const size_t Binc ) [inline]

```

11.1.3.114 fscaln() [6/10]

```

template<>
void fscaln (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t lda ) [inline]

```

11.1.3.115 fscal() [6/10]

```

template<>
void fscal (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::Element_ptr B,
    const size_t ldb ) [inline]

```

11.1.3.116 fscaln() [7/10]

```

template<>
void fscaln (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha,
    typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
    const size_t inc ) [inline]

```

11.1.3.117 fscal() [7/10]

```

template<>
void fscal (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t Ainc,
    FFPACK::rns_double::Element_ptr B,
    const size_t Binc ) [inline]

```

11.1.3.118 fscaln() [8/10]

```
template<>
void fscaln (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t lda ) [inline]
```

11.1.3.119 fscal() [8/10]

```
template<>
void fscal (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::Element_ptr B,
    const size_t ldb ) [inline]
```

11.1.3.120 fsyr2k()

```
template<class Field >
Field::Element_ptr fsyr2k (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsyr2k: Symmetric Rank 2K update

Computes the Lower or Upper triangular part of $C = \alpha(A \times B^T + B \times A^T) + \beta C$ or $C = \alpha(A^T \times B + B^T \times A) + \beta C$

Parameters

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if <code>ta==FflasNoTrans</code> then compute $C = \alpha(A \times B^T + B \times A^T) + \beta C$, else $C = \alpha(A^T \times B + B^T \times A) + \beta C$
<i>n</i>	order of matrix C
<i>k</i>	see A

Parameters

<i>alpha</i>	scalar
<i>A</i>	<i>A</i> is $n \times k$ (FflasNoTrans) or <i>A</i> is $k \times n$ (FflasTrans)
<i>lda</i>	leading dimension of <i>A</i>
<i>beta</i>	scalar
<i>C</i>	<i>C</i> is $n \times n$
<i>ldc</i>	leading dimension of <i>C</i>

Warning

α *must* be invertible

11.1.3.121 fsyrk() [1/5]

```
template<class Field >
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsyrk: Symmetric Rank K update

Computes the Lower or Upper triangular part of $C = \alpha A \times A^T + \beta C$ or $C = \alpha A^T \times A + \beta C$

Parameters

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix <i>C</i>
<i>trans</i>	if <i>ta</i> ==FflasNoTrans then compute $C = \alpha A \times A^T + \beta C$, else $C = \alpha A^T \times A + \beta C$
<i>n</i>	order of matrix <i>C</i>
<i>k</i>	see <i>A</i>
<i>alpha</i>	scalar
<i>A</i>	<i>A</i> is $n \times k$ or <i>A</i> is $k \times n$
<i>lda</i>	leading dimension of <i>A</i>
<i>beta</i>	scalar
<i>C</i>	<i>C</i> is $n \times n$
<i>ldc</i>	leading dimension of <i>C</i>

Warning

α *must* be invertible

11.1.3.122 fsyrk() [2/5]

```

template<class Field >
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD ) [inline]

```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of $C = \alpha A \times D \times A^T + \beta C$ or $C = \alpha A^T \times D \times A + \beta C$ where D is a diagonal matrix. Matrix A is updated into $D \times A$ (if trans = FflasTrans) or $A \times D$ (if trans = FflasNoTrans).

Parameters

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if ta==FflasNoTrans then compute $C = \alpha A \times A^T + \beta C$, else $C = \alpha A^T \times A + \beta C$
<i>n</i>	order of matrix C
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	A is $n \times k$ or A is $k \times n$
<i>lda</i>	leading dimension of A
<i>D</i>	D is $k \times k$ diagonal matrix, stored as a vector of k coefficients
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	C is $n \times n$
<i>ldc</i>	leading dimension of C

Warning

α must be invertible

11.1.3.123 fsyrk() [3/5]

```

template<class Field >
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,

```

```

const size_t N,
const size_t K,
const typename Field::Element alpha,
typename Field::Element_ptr A,
const size_t lda,
typename Field::ConstElement_ptr D,
const size_t incD,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const ParSeqHelper::Sequential seq,
const size_t threshold ) [inline]

```

11.1.3.124 fsyrk() [4/5]

```

template<class Field , class Cut , class Param >
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Parallel< Cut, Param > par,
    const size_t threshold ) [inline]

```

11.1.3.125 fsyrk() [5/5]

```

template<class Field >
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const std::vector< bool > & twoBlock,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD ) [inline]

```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of $C = \alpha A \times \text{Delta} D \times A^T + \beta C$ or $C = \alpha A^T \times \text{Delta} D \times A + \beta C$ where D is a diagonal matrix and Delta is a block diagonal with either 1 on the diagonal or 2x2 swap blocks Matrix A is updated into $D \times A$ (if trans = FflasTrans) or $A \times D$ (if trans = FflasNoTrans).

Parameters

<i>F</i>	field.
<i>UpLo</i>	whether to compute the upper or the lower triangular part of the symmetric matrix C
<i>trans</i>	if <code>ta==FflasNoTrans</code> then compute $C = \alpha A \Delta D \times A^T + \beta C$, else $C = \alpha A^T \Delta D \times A + \beta C$
<i>n</i>	see B
<i>k</i>	see A
<i>alpha</i>	scalar
<i>A</i>	A is $n \times k$ or A is $k \times n$
<i>lda</i>	leading dimension of A
<i>D</i>	D is $k \times k$ diagonal matrix, stored as a vector of k coefficients
<i>twoBlocks</i>	a vector boolean indicating the beginning of each 2x2 blocs in Delta
<i>lda</i>	leading dimension of A
<i>beta</i>	scalar
<i>C</i>	C is $n \times n$
<i>ldc</i>	leading dimension of C

Warning

α must be invertible

11.1.3.126 ftrmm() [1/3]

```
template<class Field >
void ftrmm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]
```

ftrmm: **TR**iangular **M**atrix **M**ultiply.

Computes $B \leftarrow \alpha \text{op}(A)B$ or $B \leftarrow \alpha B \text{op}(A)$.

Parameters

<i>F</i>	field
<i>Side</i>	if <code>Side==FflasLeft</code> then $B \leftarrow \alpha \text{op}(A)B$ is computed.
<i>Uplo</i>	if <code>Uplo==FflasUpper</code> then A is upper triangular
<i>TransA</i>	if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$.
<i>Diag</i>	if <code>Diag==FflasUnit</code> then A is implicitly unit.
<i>M</i>	rows of B

Parameters

N	cols of B
α	scalar
A	triangular matrix. If $Side == FflasLeft$ then A is $N \times N$, otherwise A is $M \times M$
lda	leading dim of A
B	matrix of size $M \times N$
ldb	leading dim of B

11.1.3.127 ftrmm() [2/3]

```
template<class Field >
void ftrmm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

ftrmm: **TR**angular **M**atrix **M**ultiply with 3 operands Computes $C \leftarrow \alpha op(A)B + beta C$ or $C \leftarrow \alpha B op(A) + beta C$.

Parameters

F	field
$Side$	if $Side == FflasLeft$ then $B \leftarrow \alpha op(A)B$ is computed.
$Uplo$	if $Uplo == FflasUpper$ then A is upper triangular
$TransA$	if $TransA == FflasTrans$ then $op(A) = A^t$.
$Diag$	if $Diag == FflasUnit$ then A is implicitly unit.
M	rows of B
N	cols of B
α	scalar
A	triangular matrix. If $Side == FflasLeft$ then A is $N \times N$, otherwise A is $M \times M$
lda	leading dim of A
B	matrix of size $M \times N$
ldb	leading dim of B
β	scalar
C	matrix of size $M \times N$
ldc	leading dim of C

11.1.3.128 ftrsm() [1/9]

```

template<class Field >
void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]

```

11.1.3.129 ftrsm() [2/9]

```

template<class Field >
void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const ParSeqHelper::Sequential & PSH ) [inline]

```

11.1.3.130 ftrsm() [3/9]

```

template<class Field , class Cut , class Param >
void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const ParSeqHelper::Parallel< Cut, Param > & PSH ) [inline]

```


11.1.3.131 ftrsm() [4/9]

```
template<class Field , class ParSeqTrait = ParSeqHelper::Sequential>
void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Recursive, ParSeqTrait > & H ) [inline]
```

11.1.3.132 ftrsm() [5/9]

```
void ftrsm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * B,
    const size_t ldb ) [inline]
```

11.1.3.133 cblas_impstrsm()

```
void cblas_impstrsm (
    const enum FFLAS_ORDER Order,
    const enum FFLAS_SIDE Side,
    const enum FFLAS_UPLO Uplo,
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_DIAG Diag,
    const int M,
    const int N,
    const FFPACK::rns_double_elt alpha,
    FFPACK::rns_double_elt_cstptr A,
    const int lda,
    FFPACK::rns_double_elt_ptr B,
    const int ldb ) [inline]
```

11.1.3.134 ftrsv() [1/2]

```
template<class Field >
void ftrsv (
    const Field & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    int incX ) [inline]
```

ftrsv: TRIangular System solve with Vector Computes $X \leftarrow \text{op}(A^{-1})X$

Parameters

<i>F</i>	field
<i>X</i>	vector of size N on a field F
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size N
<i>lda</i>	leading dimension of A
<i>N</i>	number of rows or columns of A according to TransA
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$.
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular

11.1.3.135 igemm_()

```
void igemm_ (
    const enum FFLAS_ORDER Order,
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_TRANSPOSE TransB,
    const size_t M,
    const size_t N,
    const size_t K,
    const int64_t alpha,
    const int64_t * A,
    const size_t lda,
    const int64_t * B,
    const size_t ldb,
    const int64_t beta,
    int64_t * C,
    const size_t ldc ) [inline]
```

11.1.3.136 finit() [5/8]

```
template<class Field , class OtherElement_ptr >
void finit (
    const Field & F,
```

```

    const size_t n,
    const OtherElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )

```

$\text{finit } x \leftarrow y \bmod F.$

Parameters

F	field
n	size of the vectors
Y	vector of OtherElement
$incY$	stride of Y
X	vector in F
$incX$	stride of X

Bug use `cblas_(d)scal` when possible

11.1.3.137 fconvert() [1/3]

```

template<class Field , class OtherElement_ptr >
void fconvert (
    const Field & F,
    const size_t n,
    OtherElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY )

```

$\text{fconvert } x \leftarrow y \bmod F.$

Parameters

F	field
n	size of the vectors
Y	vector of F
$incY$	stride of Y
X	vector in OtherElement
$incX$	stride of X

Bug use `cblas_(d)scal` when possible

11.1.3.138 fnegin() [1/4]

```

template<class Field >
void fnegin (

```

```

const Field & F,
const size_t n,
typename Field::Element_ptr X,
const size_t incX )

```

$\text{fnegin } x \leftarrow -x.$

Parameters

F	field
n	size of the vectors
X	vector in F
$incX$	stride of X

Bug use `cblas_(d)scal` when possible

11.1.3.139 fneg() [1/4]

```

template<class Field >
void fneg (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )

```

$\text{fneg } x \leftarrow -y.$

Parameters

F	field
n	size of the vectors
X	vector in F
$incX$	stride of X
Y	vector in F
$incY$	stride of Y

Bug use `cblas_(d)scal` when possible

11.1.3.140 fzero() [1/4]

```

template<class Field >
void fzero (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )

```

$\text{fzero} : A \leftarrow 0.$

Parameters

F	field
n	number of elements to zero
X	vector in F
$incX$	stride of X

11.1.3.141 frand() [1/2]

```
template<class Field , class RandIter >
void frand (
    const Field & F,
    RandIter & G,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

$frand : A \leftarrow random.$

Parameters

F	field
G	randomiterator
n	number of elements to randomize
X	vector in F
$incX$	stride of X

11.1.3.142 fiszero() [1/4]

```
template<class Field >
bool fiszero (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr X,
    const size_t incX )
```

$fiszero$: test $X = 0$.

Parameters

F	field
n	vector dimension
X	vector in F
$incX$	increment of X

11.1.3.143 fequal() [1/4]

```
template<class Field >
bool fequal (
```

```

const Field & F,
const size_t n,
typename Field::ConstElement_ptr X,
const size_t incX,
typename Field::ConstElement_ptr Y,
const size_t incY )

```

fequal : test $X = Y$.

Parameters

F	field
n	vector dimension
X	vector in F
$incX$	increment of X
Y	vector in F
$incY$	increment of Y

11.1.3.144 faxpby() [1/2]

```

template<class Field >
void faxpby (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY )

```

faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.

Parameters

	F	field
	N	size of the vectors
	$alpha$	scalar
in	X	vector in F
	$incX$	stride of X
	$beta$	scalar
in, out	Y	vector in F
	$incY$	stride of Y

Note

this is a catlas function

11.1.3.145 fdot() [9/11]

```

template<typename Field , class Cut , class Param >
Field::Element fdot (

```

```

const Field & F,
const size_t N,
typename Field::ConstElement_ptr X,
const size_t incX,
typename Field::ConstElement_ptr Y,
const size_t incY,
const ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

11.1.3.146 fswap() [1/2]

```

template<class Field >
void fswap (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY )

```

fswap: $X \leftrightarrow Y$.

Bug use cblas_dswap when double

Parameters

F	field
N	size of the vectors
X	vector in F
$incX$	stride of X
Y	vector in F
$incY$	stride of Y

11.1.3.147 fzero() [2/4]

```

template<class Field >
void fzero (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

fzero : $A \leftarrow 0$.

Parameters

F	field
m	number of rows to zero
n	number of cols to zero
A	matrix in F
lda	stride of A

Warning

may be buggy if Element is larger than int

11.1.3.148 frand() [2/2]

```
template<class Field , class RandIter >
void frand (
    const Field & F,
    RandIter & G,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

frand : $A \leftarrow \text{random}$.

Parameters

F	field
G	randomiterator
m	number of rows to randomize
n	number of cols to randomize
A	matrix in F
lda	stride of A

11.1.3.149 fequal() [2/4]

```
template<class Field >
bool fequal (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb )
```

fequal : test $A = B$.

Parameters

F	field
m	row dimension
n	column dimension
A	m x n matrix in F
lda	leading dimension of A
B	m x n matrix in F
ldb	leading dimension of B

11.1.3.150 fiszero() [2/4]

```
template<class Field >
bool fiszero (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

fiszero : test $A = 0$.

Parameters

F	field
m	row dimension
n	column dimension
A	m x n matrix in F
lda	leading dimension of A

11.1.3.151 fidentity() [1/4]

```
template<class Field >
void fidentity (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element & d )
```

creates a diagonal matrix

11.1.3.152 fidentity() [2/4]

```
template<class Field >
void fidentity (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

creates a diagonal matrix

11.1.3.153 finit() [6/8]

```
template<class Field , class OtherElement_ptr >
void finit (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

finit Initializes A in F^S .

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.154 fconvert() [2/3]

```
template<class Field , class OtherElement_ptr >
void fconvert (
    const Field & F,
    const size_t m,
    const size_t n,
    OtherElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb )
```

$\text{fconvert } A \leftarrow B \bmod F.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in OtherElement
lda	stride of A
B	matrix in F
ldb	stride of B

Todo check if $n == lda$

11.1.3.155 fnegin() [2/4]

```
template<class Field >
void fnegin (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

$\text{fnegin } A \leftarrow -A.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

Todo check if $n == lda$

11.1.3.156 fneg() [2/4]

```
template<class Field >
void fneg (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )
```

$\text{fneg } A \leftarrow -B.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

Todo check if $n == lda$

11.1.3.157 faxpby() [2/2]

```
template<class Field >
void faxpby (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t ldx,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t ldy )
```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

Parameters

	F	field
	m	row dimension
	n	column dimension
	α	scalar
in	X	vector in F
	ldx	leading dimension of X
	β	scalar
	ldy	leading dimension of Y

Note

this is a catlas function

11.1.3.158 fmove() [1/2]

```
template<class Field >
void fmove (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb )
```

$\text{fmove} : A \leftarrow B \text{ and } B \leftarrow 0.$

Parameters

F	field
m	number of rows to copy
n	number of cols to copy
A	matrix in F
lda	stride of A
B	matrix in F
ldb	stride of B

11.1.3.159 bitsize()

```
template<class Field >
size_t bitsize (
    const Field & F,
    size_t M,
    size_t N,
    const typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

bitsize: Computes the largest bitsize of the matrix' coefficients.

If the matrix is over a modular prime field, it returns the bitsize of the largest element (in a bsolute value)

Parameters

F	field
M	rows
N	cols
$incX$	stride of X
A	a matrix of leading dimension lda and size $M \times N$
lda	leading dimension of A

11.1.3.160 `bitsize< Givaro::ZRing< Givaro::Integer > >()`

```
template<>
size_t bitsize< Givaro::ZRing< Givaro::Integer > > (
    const Givaro::ZRing< Givaro::Integer > & F,
    size_t M,
    size_t N,
    const Givaro::Integer * A,
    size_t lda ) [inline]
```

11.1.3.161 `ftrmv()`

```
template<class Field >
void ftrmv (
    const Field & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    int incX )
```

ftrsm: TRIangular Matrix Vector prodcut Computes $X \leftarrow \text{op}(A)X$

Parameters

<i>F</i>	field
<i>X</i>	vector of size N on a field F
<i>incX</i>	stride of X
<i>A</i>	a matrix of leading dimension lda and size N
<i>lda</i>	leading dimension of A
<i>N</i>	number of rows and columns of A
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^T$.
<i>Diag</i>	if Diag==FflasUnit then A is unit diagonal.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular

11.1.3.162 `ftrsm()` [6/9]

```
template<class Field >
void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
```

```

const size_t lda,
typename Field::Element_ptr B,
const size_t ldb )

```

ftfrm: **TR**angular **S**ystem solve with **M**atrix.

Computes $B \leftarrow \alpha \text{op}(A^{-1})B$ or $B \leftarrow \alpha B \text{op}(A^{-1})$.

Parameters

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A^{-1})B$ is computed.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$.
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular invertible matrix. If Side==FflasLeft then A is $N \times N$, otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B

Bug α must be non zero.

11.1.3.163 pfgemm() [1/7]

```

template<typename Field >
Field::Element_ptr pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numthreads = 0 )

```

11.1.3.164 pfgemm_1D_rec()

```

template<class Field >
Field::Element * pfgemm_1D_rec (

```

```

const Field & F,
const FFLAS_TRANSPOSE ta,
const FFLAS_TRANSPOSE tb,
const size_t m,
const size_t n,
const size_t k,
const typename Field::Element alpha,
const typename Field::Element_ptr A,
const size_t lda,
const typename Field::Element_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element * C,
const size_t ldc,
size_t seuil )

```

11.1.3.165 pfgemm_2D_rec()

```

template<class Field >
Field::Element * pfgemm_2D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    size_t seuil )

```

11.1.3.166 pfgemm_3D_rec()

```

template<class Field >
Field::Element * pfgemm_3D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t seuil,
    size_t * x )

```

11.1.3.167 pfgemm_3D_rec2()

```

template<class Field >
Field::Element_ptr pfgemm_3D_rec2 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t seuil,
    size_t * x )

```

11.1.3.168 fgemm() [19/23]

```

template<class Field , class ModeTrait , class Strat , class Param >
std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag
> >::value, typename Field::Element_ptr >::type fgemm (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE ta,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,
Param > > & H ) [inline]

```

11.1.3.169 ftrsm() [7/9]

```

template<class Field , class Cut , class Param >
Field::Element_ptr ftrsm (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_TRANSPOSE TA,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t m,
    const size_t n,

```



```

        const typename Field::Element alpha,
        typename Field::Element_ptr A,
        const size_t lda,
        typename Field::Element_ptr B,
        const size_t ldb,
        TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &
H ) [inline]

```

11.1.3.170 ftrsm() [8/9]

```

template<class Field , class Cut , class Param >
Field::Element_ptr ftrsm (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_TRANSPOSE TA,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > & H
) [inline]

```

11.1.3.171 fspmv() [1/2]

```

template<class Field , class SM >
void fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    const typename Field::Element & beta,
    typename Field::Element_ptr y ) [inline]

```

11.1.3.172 fspmm()

```

template<class Field , class SM >
void fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    const typename Field::Element & beta,
    typename Field::Element_ptr y,
    int ldy ) [inline]

```

11.1.3.173 sparse_init() [1/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::COO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.174 sparse_init() [2/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.175 sparse_delete() [1/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::COO > & A ) [inline]
```

11.1.3.176 sparse_delete() [2/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A ) [inline]
```

11.1.3.177 sparse_init() [3/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.178 sparse_init() [4/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.179 sparse_delete() [3/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR > & A ) [inline]
```

11.1.3.180 sparse_delete() [4/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A ) [inline]
```

11.1.3.181 sparse_print() [1/3]

```
template<class Field >
std::ostream & sparse_print (
    std::ostream & os,
    const Sparse< Field, SparseMatrix_t::CSR > & A ) [inline]
```

11.1.3.182 sparse_init() [5/16]

```
template<class IndexT >
void sparse_init (
    const Givaro::Modular< Givaro::Integer > & F,
    Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > & A,
    const IndexT * row,
    const IndexT * col,
    Givaro::Integer * dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.183 sparse_init() [6/16]

```
template<class IndexT >
void sparse_init (
    const Givaro::ZRing< Givaro::Integer > & F,
    Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    Givaro::Integer * dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.184 sparse_init() [7/16]

```
template<class IndexT , size_t RECINT_SIZE>
void sparse_init (
    const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > & F,
    Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO >
    & A,
    const IndexT * row,
    const IndexT * col,
    typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.185 sparse_init() [8/16]

```
template<class IndexT , size_t RECINT_SIZE>
void sparse_init (
    const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > & F,
    Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &
    A,
    const IndexT * row,
    const IndexT * col,
    typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.186 sparse_delete() [5/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A ) [inline]
```

11.1.3.187 sparse_init() [9/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.188 sparse_init() [10/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.189 sparse_init() [11/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.190 sparse_delete() [6/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL > & A ) [inline]
```

11.1.3.191 sparse_delete() [7/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A ) [inline]
```

11.1.3.192 sparse_init() [12/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.193 sparse_init() [13/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.194 sparse_delete() [8/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A ) [inline]
```

11.1.3.195 sparse_delete() [9/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A ) [inline]
```

11.1.3.196 sparse_print() [2/3]

```
template<class Field >
void sparse_print (
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A ) [inline]
```

11.1.3.197 sparse_delete() [10/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A ) [inline]
```

11.1.3.198 sparse_init() [14/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.199 operator<<()

```
template<typename _Field >
std::ostream & operator<< (
    std::ostream & os,
    const Sparse< _Field, SparseMatrix_t::HYB_ZO > & A )
```

11.1.3.200 readSmsFormat()

```
template<class Field , bool sorted = true, bool read_integer = false>
void readSmsFormat (
    const std::string & path,
    const Field & f,
    index_t *& row,
    index_t *& col,
    typename Field::Element_ptr & val,
    index_t & rowdim,
    index_t & coldim,
    uint64_t & nnz )
```

11.1.3.201 readSprFormat()

```
template<class Field >
void readSprFormat (
    const std::string & path,
    const Field & f,
    index_t *& row,
    index_t *& col,
    typename Field::Element_ptr & val,
    index_t & rowdim,
    index_t & coldim,
    uint64_t & nnz )
```

11.1.3.202 getDataType() [1/4]

```
template<class T >
std::enable_if< std::is_integral< T >::value, int > getDataType ( )
```

11.1.3.203 `getDataType()` [2/4]

```
template<class T >
std::enable_if< std::is_floating_point< T >::value, int > getDataType ( )
```

11.1.3.204 `getDataType()` [3/4]

```
template<class T >
std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ( )
```

11.1.3.205 `getDataType()` [4/4]

```
template<class T >
int getDataType ( )
```

11.1.3.206 `readMachineType()`

```
template<class Field >
void readMachineType (
    const Field & F,
    typename Field::Element & modulo,
    typename Field::Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

11.1.3.207 `readDnsFormat()`

```
template<class Field >
void readDnsFormat (
    const std::string & path,
    const Field & F,
    index_t & rowdim,
    index_t & coldim,
    typename Field::Element_ptr & val )
```

11.1.3.208 `writeDnsFormat()`

```
template<class Field >
void writeDnsFormat (
    const std::string & path,
    const Field & F,
    const index_t & rowdim,
    const index_t & coldim,
    typename Field::Element_ptr A,
    index_t ldA )
```


11.1.3.209 fspmv() [2/2]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ) [inline]
```

11.1.3.210 sparse_delete() [11/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::SELL > & A ) [inline]
```

11.1.3.211 sparse_delete() [12/12]

```
template<class Field >
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A ) [inline]
```

11.1.3.212 sparse_print() [3/3]

```
template<class Field >
void sparse_print (
    const Sparse< Field, SparseMatrix_t::SELL > & A ) [inline]
```

11.1.3.213 sparse_init() [15/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::SELL > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz,
    uint64_t sigma = 0 ) [inline]
```

11.1.3.214 sparse_init() [16/16]

```
template<class Field , class IndexT >
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

11.1.3.215 computeDeviation()

```
template<class It >
double computeDeviation (
    It begin,
    It end )
```

11.1.3.216 getStat()

```
template<class Field >
StatsMatrix getStat (
    const Field & F,
    const index_t * row,
    const index_t * col,
    typename Field::ConstElement_ptr val,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz )
```

11.1.3.217 fflas_delete() [1/4]

```
template<>
void fflas_delete (
    FFPACK::rns_double_elt_ptr A ) [inline]
```

11.1.3.218 fflas_delete() [2/4]

```
template<>
void fflas_delete (
    FFPACK::rns_double_elt_cstptr A ) [inline]
```

11.1.3.219 fflas_new() [1/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const Alignment align ) [inline]
```

11.1.3.220 fflas_new() [2/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const Alignment align ) [inline]
```

11.1.3.221 finit_rns() [1/2]

```
template<typename RNS >
void finit_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename RNS::Element_ptr A )
```

11.1.3.222 finit_trans_rns()

```
template<typename RNS >
void finit_trans_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename RNS::Element_ptr A )
```

11.1.3.223 fconvert_rns() [1/2]

```
template<typename RNS >
void fconvert_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename RNS::ConstElement_ptr A )
```

11.1.3.224 fconvert_trans_rns()

```
template<typename RNS >
void fconvert_trans_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename RNS::ConstElement_ptr A )
```

11.1.3.225 fflas_new() [3/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const Alignment align ) [inline]
```

11.1.3.226 fflas_new() [4/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const Alignment align ) [inline]
```

11.1.3.227 finit_rns() [2/2]

```
template<typename RNS >
void finit_rns (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename FFPACK::RNSInteger< RNS >::Element_ptr A )
```

11.1.3.228 fconvert_rns() [2/2]

```
template<typename RNS >
void fconvert_rns (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A )
```

11.1.3.229 freduce() [7/10]

```
template INST_OR_DECL void freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

$\text{freduce } x \leftarrow x \bmod F.$

Parameters

F	field
n	size of the vectors
X	vector in F
incX	stride of X

Bug use cblas_(d)scal when possible

11.1.3.230 freduce() [8/10]

```
template INST_OR_DECL void freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

freduce $x \leftarrow y \bmod F$.

Parameters

F	field
n	size of the vectors
Y	vector of Element
$incY$	stride of Y
X	vector in F
$incX$	stride of X

Bug use cblas_(d)scal when possible

11.1.3.231 finit() [7/8]

```
template INST_OR_DECL void finit (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

finit $x \leftarrow y \bmod F$.

Parameters

F	field
n	size of the vectors
Y	vector of OtherElement
$incY$	stride of Y
X	vector in F
$incX$	stride of X

Bug use cblas_(d)scal when possible

11.1.3.232 fconvert() [3/3]

```
template INST_OR_DECL void fconvert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```

```

const size_t n,
FFLAS_ELT * X,
const size_t incX,
const FFLAS_ELT * Y,
const size_t incY )

```

$\text{fconvert } x \leftarrow y \bmod F.$

Parameters

F	field
n	size of the vectors
Y	vector of F
$incY$	stride of Y
X	vector in <code>OtherElement</code>
$incX$	stride of X

Bug use `cblas_(d)scal` when possible

11.1.3.233 `fnegin()` [3/4]

```

template INST_OR_DECL void fnegin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )

```

$\text{fnegin } x \leftarrow -x.$

Parameters

F	field
n	size of the vectors
X	vector in F
$incX$	stride of X

Bug use `cblas_(d)scal` when possible

11.1.3.234 `fneg()` [3/4]

```

template INST_OR_DECL void fneg (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )

```

$\text{fneg } x \leftarrow -y.$

Parameters

F	field
n	size of the vectors
X	vector in F
$incX$	stride of X
Y	vector in F
$incY$	stride of Y

Bug use `cblas_(d)scal` when possible

11.1.3.235 `fzero()` [3/4]

```
template INST_OR_DECL void fzero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

`fzero` : $A \leftarrow 0$.

Parameters

F	field
n	number of elements to zero
X	vector in F
$incX$	stride of X

11.1.3.236 `fiszero()` [3/4]

```
template INST_OR_DECL bool fiszero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * X,
    const size_t incX )
```

`fiszero` : test $X = 0$.

Parameters

F	field
n	vector dimension
X	vector in F
$incX$	increment of X

11.1.3.237 fequal() [3/4]

```
template INST_OR_DECL bool fequal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

fequal : test $X = Y$.

Parameters

F	field
n	vector dimension
X	vector in F
$incX$	increment of X
Y	vector in F
$incY$	increment of Y

11.1.3.238 fassign() [9/10]

```
template INST_OR_DECL void fassign (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

fassign : $x \leftarrow y$.

X is preallocated

Todo variant for triagular matrix

Parameters

	F	field
	N	size of the vectors
out	X	vector in F
	$incX$	stride of X
in	Y	vector in F
	$incY$	stride of Y

11.1.3.239 fscaln() [9/10]

```
template INST_OR_DECL void fscaln (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```



```

    const size_t n,
    const FFLAS_ELT alpha,
    FFLAS_ELT * X,
    const size_t incX )

```

fscal $\mathbf{in} \ x \leftarrow \alpha \cdot x$.

Parameters

F	field
n	size of the vectors
α	scalar
X	vector in F
$incX$	stride of X

Bug use cblas_(d)scal when possible

Todo check if comparison with +/-1,0 is necessary.

11.1.3.240 fscal() [9/10]

```

template INST_OR_DECL void fscal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )

```

fscal $\mathbf{out} \ y \leftarrow \alpha \cdot x$.

Parameters

	F	field
	n	size of the vectors
	α	scalar
in	X	vector in F
	$incX$	stride of X
out	Y	vector in F
	$incY$	stride of Y

Bug use cblas_(d)scal when possible

Todo check if comparison with +/-1,0 is necessary.

11.1.3.241 faxpy() [5/6]

```
template INST_OR_DECL void faxpy (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

Parameters

	F	field
	N	size of the vectors
	α	scalar
in	X	vector in F
	incX	stride of X
in, out	Y	vector in F
	incY	stride of Y

11.1.3.242 fdot() [10/11]

```
template INST_OR_DECL FFLAS_ELT fdot (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

Parameters

	F	field
	N	size of the vectors
	α	scalar
in	X	vector in F
	incX	stride of X
	β	scalar
in, out	Y	vector in F
	incY	stride of Y

Note

this is a catlas function

fdot: dot product $x^T y$.

Parameters

F	field
N	size of the vectors
X	vector in F
$incX$	stride of X
Y	vector in F
$incY$	stride of Y

11.1.3.243 fswap() [2/2]

```
template INST_OR_DECL void fswap (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

fswap: $X \leftrightarrow Y$.

Bug use cblas_dswap when double

Parameters

F	field
N	size of the vectors
X	vector in F
$incX$	stride of X
Y	vector in F
$incY$	stride of Y

11.1.3.244 fadd() [5/8]

```
template INST_OR_DECL void fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

11.1.3.245 fsub() [3/4]

```
template INST_OR_DECL void fsub (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```

```

const size_t N,
const FFLAS_ELT * A,
const size_t inca,
const FFLAS_ELT * B,
const size_t incb,
FFLAS_ELT * C,
const size_t incc )

```

11.1.3.246 faddin() [3/4]

```

template INST_OR_DECL void faddin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )

```

11.1.3.247 fadd() [6/8]

```

template INST_OR_DECL void fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )

```

11.1.3.248 fassign() [10/10]

```

template INST_OR_DECL void fassign (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )

```

fassign : $A \leftarrow B$.

Parameters

F	field
m	number of rows to copy
n	number of cols to copy
A	matrix in F
lda	stride of A
B	vector in F
ldb	stride of B

11.1.3.249 fzero() [4/4]

```
template INST_OR_DECL void fzero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

fzero : $A \leftarrow 0$.

Parameters

F	field
m	number of rows to zero
n	number of cols to zero
A	matrix in F
lda	stride of A

Warning

may be buggy if Element is larger than int

11.1.3.250 fequal() [4/4]

```
template INST_OR_DECL bool fequal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb )
```

fequal : test $A = B$.

Parameters

F	field
m	row dimension
n	column dimension
A	m x n matrix in F
lda	leading dimension of A
B	m x n matrix in F
ldb	leading dimension of B

11.1.3.251 fiszero() [4/4]

```
template INST_OR_DECL bool fiszero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```

```

    const size_t m,
    const size_t n,
    const FFLAS_ELT * A,
    const size_t lda )

```

fiszero : test $A = 0$.

Parameters

F	field
m	row dimension
n	column dimension
A	$m \times n$ matrix in F
lda	leading dimension of A

11.1.3.252 fidentity() [3/4]

```

template INST_OR_DECL void fidentity (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT & d )

```

creates a diagonal matrix

11.1.3.253 fidentity() [4/4]

```

template INST_OR_DECL void fidentity (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )

```

creates a diagonal matrix

11.1.3.254 freduce() [9/10]

```

template INST_OR_DECL void freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )

```

freduce $A \leftarrow A \bmod F$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.255 **freduce()** [10/10]

```
template INST_OR_DECL void freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{freduce } A \leftarrow B \bmod F.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A
B	matrix in Element
ldb	stride of B

11.1.3.256 **finit()** [8/8]

```
template INST_OR_DECL void finit (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{finit } A \leftarrow B \bmod F.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A
B	matrix in F
ldb	stride of B

11.1.3.257 fnegin() [4/4]

```
template INST_OR_DECL void fnegin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fnegin } A \leftarrow -A.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.258 fneg() [4/4]

```
template INST_OR_DECL void fneg (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fneg } A \leftarrow -B.$

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A

11.1.3.259 fscaln() [10/10]

```
template INST_OR_DECL void fscaln (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fscaln } A \leftarrow a \cdot A.$

Parameters

F	field
m	number of rows
n	number of cols
α	homotecie scalar
A	matrix in F
lda	stride of A

11.1.3.260 fscal() [10/10]

```
template INST_OR_DECL void fscal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

$\text{fscal } B \leftarrow a \cdot A.$

Parameters

	F	field
	m	number of rows
	n	number of cols
	α	homotecie scalar
in	A	matrix in F
	lda	stride of A
out	B	matrix in F
	ldb	stride of B

11.1.3.261 faxpy() [6/6]

```
template INST_OR_DECL void faxpy (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t ldx,
    FFLAS_ELT * Y,
    const size_t ldy )
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

Parameters

	F	field
--	-----	-------

Parameters

	m	row dimension
	n	column dimension
	α	scalar
in	X	vector in \mathbb{F}
	ldx	leading dimension of X
in, out	Y	vector in \mathbb{F}
	ldy	leading dimension of Y

11.1.3.262 fmove() [2/2]

```
template INST_OR_DECL void fmove (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.

Parameters

	F	field
	m	row dimension
	n	column dimension
	α	scalar
in	X	vector in \mathbb{F}
	ldx	leading dimension of X
	β	scalar
in, out	Y	vector in \mathbb{F}
	ldy	leading dimension of Y

Note

this is a catlas function

fmove : $A \leftarrow B$ and $B \leftarrow 0$.

Parameters

F	field
m	number of rows to copy
n	number of cols to copy
A	matrix in \mathbb{F}
lda	stride of A
B	vector in \mathbb{F}
ldb	stride of B

11.1.3.263 fadd() [7/8]

```
template INST_OR_DECL void fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

fadd : matrix addition.

Computes $C = A + B$.

Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size $M \times N$
<i>lda</i>	leading dimension of A
<i>B</i>	dense matrix of size $M \times N$
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size $M \times N$
<i>ldc</i>	leading dimension of C

11.1.3.264 fsub() [4/4]

```
template INST_OR_DECL void fsub (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

fsub : matrix subtraction.

Computes $C = A - B$.

Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size $M \times N$
<i>lda</i>	leading dimension of A

Parameters

<i>B</i>	dense matrix of size $M \times N$
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size $M \times N$
<i>ldc</i>	leading dimension of C

11.1.3.265 fsubin() [3/3]

```
template INST_OR_DECL void fsubin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

fsubin $C = C - B$

11.1.3.266 fadd() [8/8]

```
template INST_OR_DECL void fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

fadd : matrix addition with scaling.

Computes $C = A + \text{alpha } B$.

Parameters

<i>F</i>	field
<i>M</i>	rows
<i>N</i>	cols
<i>A</i>	dense matrix of size $M \times N$
<i>lda</i>	leading dimension of A
<i>alpha</i>	some scalar
<i>B</i>	dense matrix of size $M \times N$
<i>ldb</i>	leading dimension of B
<i>C</i>	dense matrix of size $M \times N$
<i>ldc</i>	leading dimension of C

11.1.3.267 faddin() [4/4]

```
template INST_OR_DECL void faddin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

faddin

11.1.3.268 fgemv() [17/19]

```
template INST_OR_DECL FFLAS_ELT * fgemv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE TransA,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT beta,
    FFLAS_ELT * Y,
    const size_t incY )
```

finite prime FFLAS_FIELD<FFLAS_ELT> GEneral Matrix Vector multiplication.

Computes $Y \leftarrow \alpha \text{op}(A)X + \beta Y$.

Parameters

	F	field
	$TransA$	if $TransA == FflasTrans$ then $\text{op}(A) = A^t$.
	M	rows
	N	cols
	$alpha$	scalar
	A	dense matrix of size $M \times N$
	lda	leading dimension of A
	X	dense vector of size N
	$incX$	stride of X
	$beta$	scalar
out	Y	dense vector of size M
	$incY$	stride of Y

11.1.3.269 fger() [12/12]

```
template INST_OR_DECL void fger (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```

```

const size_t M,
const size_t N,
const FFLAS_ELT alpha,
const FFLAS_ELT * x,
const size_t incx,
const FFLAS_ELT * y,
const size_t incy,
FFLAS_ELT * A,
const size_t lda )

```

fger: rank one update of a general matrix

Computes $A \leftarrow \alpha x y^T + A$

Parameters

	F	field
	M	rows
	N	cols
	α	scalar
in, out	A	dense matrix of size MxN and leading dimension lda
	lda	leading dimension of A
	x	dense vector of size M
	$incx$	stride of X
	y	dense vector of size N
	$incy$	stride of Y

11.1.3.270 ftrsv() [2/2]

```

template INST_OR_DECL void ftrsv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    int incX )

```

ftrsv: TRIangular System solve with Vector Computes $X \leftarrow \text{op}(A^{-1})X$

Parameters

F	field
X	vector of size N on a field F
$incX$	stride of X
A	a matrix of leading dimension lda and size N
lda	leading dimension of A
N	number of rows or columns of A according to TransA
$TransA$	if TransA==FflasTrans then $\text{op}(A) = A^t$.
$Diag$	if Diag==FflasUnit then A is unit.
$Uplo$	if Uplo==FflasUpper then A is upper triangular

11.1.3.271 ftrsm() [9/9]

```
template INST_OR_DECL void ftrsm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

ftrsm: **T**Riangular **S**ystem solve with **M**atrix.

Computes $B \leftarrow \alpha \text{op}(A^{-1})B$ or $B \leftarrow \alpha B \text{op}(A^{-1})$.

Parameters

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A^{-1})B$ is computed.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$.
<i>Diag</i>	if Diag==FflasUnit then A is unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular invertible matrix. If Side==FflasLeft then A is $N \times N$, otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B

Bug α must be non zero.

11.1.3.272 ftrmm() [3/3]

```
template INST_OR_DECL void ftrmm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
```

```

FFLAS_ELT * B,
const size_t ldb )

```

ftmrm: **TR**angular **M**atrix **M**ultiply.

Computes $B \leftarrow \alpha \text{op}(A)B$ or $B \leftarrow \alpha B \text{op}(A)$.

Parameters

<i>F</i>	field
<i>Side</i>	if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A)B$ is computed.
<i>Uplo</i>	if Uplo==FflasUpper then A is upper triangular
<i>TransA</i>	if TransA==FflasTrans then $\text{op}(A) = A^t$.
<i>Diag</i>	if Diag==FflasUnit then A is implicitly unit.
<i>M</i>	rows of B
<i>N</i>	cols of B
<i>alpha</i>	scalar
<i>A</i>	triangular matrix. If Side==FflasLeft then A is $N \times N$, otherwise A is $M \times M$
<i>lda</i>	leading dim of A
<i>B</i>	matrix of size MxN
<i>ldb</i>	leading dim of B

11.1.3.273 fgemm() [20/23]

```

template INST_OR_DECL FFLAS_ELT * fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc )

```

fgemm: **F**ield **G**eneral **M**atrix **M**ultiply.

Computes $C = \alpha \text{op}(A) \times \text{op}(B) + \beta C$ Automatically set Winograd recursion level

Parameters

<i>F</i>	field.
<i>ta</i>	if ta==FflasTrans then $\text{op}(A) = A^t$, else $\text{op}(A) = A$,
<i>tb</i>	same for matrix B
<i>m</i>	see A
<i>n</i>	see B
<i>k</i>	see A

Parameters

<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	$\text{op}(A)$ is $m \times k$
<i>B</i>	$\text{op}(B)$ is $k \times n$
<i>C</i>	<i>C</i> is $m \times n$
<i>lda</i>	leading dimension of A
<i>ldb</i>	leading dimension of B
<i>ldc</i>	leading dimension of C
<i>w</i>	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of <i>w</i> .

Warning

α must be invertible

11.1.3.274 fgemm() [21/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc,
    const ParSeqHelper::Sequential seq )
```

11.1.3.275 fgemm() [22/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc,
    const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive
> par )
```

11.1.3.276 fgemm() [23/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc,
    const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads
> par )
```

11.1.3.277 fsquare() [6/6]

```
template INST_OR_DECL FFLAS_ELT * fsquare (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc )
```

fsquare: Squares a matrix.

compute $C \leftarrow \alpha \text{op}(A) \text{op}(A) + \beta C$ over a FFLAS_FIELD <FFLAS_ELT> F Avoid the conversion of B

Parameters

<i>ta</i>	if ta==FflasTrans, $\text{op}(A) = A^T$.
<i>F</i>	field
<i>n</i>	size of A
<i>alpha</i>	scalar
<i>beta</i>	scalar
<i>A</i>	dense matrix of size n×n
<i>lda</i>	leading dimension of A
<i>C</i>	dense matrix of size n×n
<i>ldc</i>	leading dimension of C

11.1.3.278 BlockCuts() [1/2]

```
template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>
```

```
void BlockCuts (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.279 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()

```
template<>
void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.280 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()

```
template<>
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.281 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()

```
template<>
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

11.1.3.282 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()

```
template<>
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

11.1.3.283 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()

```
template<>
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.284 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()

```
template<>
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

11.1.3.285 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()

```
template<>
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.286 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()

```
template<>
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.287 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()

```
template<>
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.288 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()

```
template<>
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.289 BlockCuts() [2/2]

```
template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>
void BlockCuts (
    size_t & rowBlockSize,
    size_t & colBlockSize,
    size_t & lastRBS,
    size_t & lastCBS,
    size_t & changeRBS,
    size_t & changeCBS,
    size_t & numRowsBlock,
    size_t & numColBlock,
    size_t m,
    size_t n,
    const size_t numthreads ) [inline]
```

11.1.3.290 pfzero()

```
template<class Field >
void pfzero (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr C,
    size_t BS = 0 )
```

11.1.3.291 pfrand()

```
template<class Field , class RandIter >
void pfrand (
    const Field & F,
    RandIter & G,
    size_t m,
    size_t n,
    typename Field::Element_ptr C,
    size_t BS = 0 )
```

11.1.3.292 fdot() [11/11]

```
template<class Field , class Cut , class Param >
Field::Element & fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element & d,
    const ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

11.1.3.293 pfgemm() [2/7]

```
template<class Field , class AlgoT , class FieldTrait >
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block,
StrategyParameter::Threads > > & H )
```

11.1.3.294 pfgemm() [3/7]

```
template<class Field , class AlgoT , class FieldTrait >
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDAdaptive > > & H )
```

11.1.3.295 pfgemm() [4/7]

```
template<class Field , class AlgoT , class FieldTrait >
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoDAdaptive > > & H )
```

11.1.3.296 pfgemm() [5/7]

```
template<class Field , class AlgoT , class FieldTrait >
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoD > > & H )
```

11.1.3.297 pfgemm() [6/7]

```
template<class Field , class AlgoT , class FieldTrait >
Field::Element_ptr pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
```

```

    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeD > > & H )

```

11.1.3.298 pfgemm() [7/7]

```

template<class Field , class AlgoT , class FieldTrait >
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDInPlace > > & H )

```

11.1.3.299 fgemv() [18/19]

```

template<class Field , class AlgoT , class FieldTrait >
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::Threads > > & H )

```

11.1.3.300 fgemv() [19/19]

```

template<class Field , class AlgoT , class FieldTrait , class Cut >
Field::Element_ptr fgemv (

```



```

    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row,
Cut > > & H )

```

11.1.3.301 parseArguments()

```

void parseArguments (
    int argc,
    char ** argv,
    Argument * args,
    bool printDefaults = true )

```

11.1.3.302 writeCommandString()

```

std::ostream & writeCommandString (
    std::ostream & os,
    Argument * args,
    const char * programName = nullptr )

```

writes the values of all arguments, preceded by the programName

11.1.3.303 WriteMatrix() [1/2]

```

template<class Field >
std::ostream & WriteMatrix (
    std::ostream & c,
    const Field & F,
    size_t m,
    size_t n,
    typename Field::ConstElement_ptr A,
    size_t lda,
    FFLAS_FORMAT format,
    bool column_major ) [inline]

```

WriteMatrix: write a matrix to an output stream.

Parameters

<i>c</i>	output stream
<i>F</i>	base field
<i>m</i>	row dimension
<i>n</i>	column dimension
<i>A</i>	matrix
<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
<i>column_major</i>	whether the matrix is stored in column or row major (row by default)

11.1.3.304 preamble()

```
void preamble (
    std::ifstream & ifs,
    FFLAS_FORMAT & format ) [inline]
```

11.1.3.305 ReadMatrix() [1/2]

```
template<class Field >
Field::Element_ptr ReadMatrix (
    std::ifstream & ifs,
    Field & F,
    size_t & m,
    size_t & n,
    typename Field::Element_ptr & A,
    FFLAS_FORMAT format = FflasAuto )
```

ReadMatrix: read a matrix from an input stream.

Parameters

	<i>ifs</i>	input stream
	<i>F</i>	base field
out	<i>m</i>	row dimension
out	<i>n</i>	column dimension
out	<i>A</i>	output matrix
	<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

11.1.3.306 ReadMatrix() [2/2]

```
template<class Field >
Field::Element_ptr ReadMatrix (
    const std::string & matrix_file,
    Field & F,
    size_t & m,
    size_t & n,
    typename Field::Element_ptr & A,
    FFLAS_FORMAT format = FflasAuto ) [inline]
```

ReadMatrix: read a matrix from a file.

Parameters

	<i>matrix_file</i>	filename
	<i>F</i>	base field
out	<i>m</i>	row dimension
out	<i>n</i>	column dimension
out	<i>A</i>	output matrix
	<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

11.1.3.307 WriteMatrix() [2/2]

```
template<class Field >
void WriteMatrix (
    std::string & matrix_file,
    const Field & F,
    int m,
    int n,
    typename Field::ConstElement_ptr A,
    size_t lda,
    FFLAS_FORMAT format = FflasDense,
    bool column_major = false )
```

WriteMatrix: write a matrix to a file.

Parameters

<i>matrix_file</i>	file name
<i>F</i>	base field
<i>m</i>	row dimension
<i>n</i>	column dimension
<i>A</i>	matrix
<i>format</i>	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
<i>column_major</i>	whether the matrix is stored in column or row major (row by default)

11.1.3.308 WritePermutation()

```
std::ostream & WritePermutation (
    std::ostream & c,
    const size_t * P,
    size_t N ) [inline]
```

WritePermutation: write a permutation matrix to an output stream.

Parameters

<i>c</i>	output stream
<i>P</i>	permutation
<i>N</i>	size of the permutation

11.1.3.309 alignable()

```
template<class Element >
bool alignable ( ) [inline]
```

11.1.3.310 alignable< Givaro::Integer * >()

```
template<>
bool alignable< Givaro::Integer * > ( ) [inline]
```

11.1.3.311 fflas_new() [5/7]

```
template<class Field >
Field::Element_ptr fflas_new (
    const Field & F,
    const size_t m,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

11.1.3.312 fflas_new() [6/7]

```
template<class Field >
Field::Element_ptr fflas_new (
    const Field & F,
    const size_t m,
    const size_t n,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

11.1.3.313 fflas_new() [7/7]

```
template<class Element >
Element * fflas_new (
    const size_t m,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

11.1.3.314 fflas_delete() [3/4]

```
template<class Element_ptr >
void fflas_delete (
    Element_ptr A ) [inline]
```

11.1.3.315 fflas_delete() [4/4]

```
template<class Ptr , class ... Args>
void fflas_delete (
    Ptr p,
    Args ... args ) [inline]
```

11.1.3.316 prefetch()

```
void prefetch (
    const int64_t * ) [inline]
```

11.1.3.317 getTLBSize()

```
void getTLBSize (
    int & tlb ) [inline]
```

11.1.3.318 queryCacheSizes()

```
void queryCacheSizes (
    int & l1,
    int & l2,
    int & l3 ) [inline]
```

Queries and returns the cache sizes in Bytes of the L1, L2, and L3 data caches respectively

11.1.3.319 queryL1CacheSize()

```
int queryL1CacheSize ( ) [inline]
```

Returns

the size in Bytes of the L1 data cache

11.1.3.320 queryTopLevelCacheSize()

```
int queryTopLevelCacheSize ( ) [inline]
```

Returns

the size in Bytes of the L2 or L3 cache if this later is present

11.1.3.321 getSeed()

```
uint64_t getSeed ( )
```

11.2 FFLAS::BLAS3 Namespace Reference**Functions**

- `template<class Field >`
`void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr,`
`const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A,`
`const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta,`
`typename Field::Element_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE`
`base, const size_t rec_level)`
- `template<class Field , class FieldTrait , class Strat , class Param >`
`Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`
`MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)`

beta, typename [Field::Element_ptr](#) C, const size_t ldc, const [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldTrait](#) > &WH)

- `template<class Field , class FieldTrait >`
`void Winograd_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const type-
name Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

11.2.1 Function Documentation

11.2.1.1 Bini()

```
template<class Field >
void Bini (
    const Field & F,
    const FFLAS\_TRANSPOSE ta,
    const FFLAS\_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element\_ptr A,
    const size_t lda,
    const typename Field::Element\_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element\_ptr C,
    const size_t ldc,
    const size_t kmax,
    const size_t w,
    const FFLAS\_BASE base,
    const size_t rec_level ) [inline]
```

11.2.1.2 WinoPar()

```
template<class Field , class FieldTrait , class Strat , class Param >
Field::Element\_ptr WinoPar (
    const Field & F,
    const FFLAS\_TRANSPOSE ta,
    const FFLAS\_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement\_ptr A,
    const size_t lda,
    typename Field::ConstElement\_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element\_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel<
    Strat, Param > > & WH ) [inline]
```

11.2.1.3 Winograd()

```
template<class Field , class FieldTrait >
void Winograd (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

11.2.1.4 WinogradAcc_3_23()

```
template<class Field , class FieldTrait >
void WinogradAcc_3_23 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

11.2.1.5 WinogradAcc_3_21()

```
template<class Field , class FieldTrait >
void WinogradAcc_3_21 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
```



```

    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.2.1.6 WinogradAcc_2_24()

```

template<class Field , class FieldTrait >
void WinogradAcc_2_24 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.2.1.7 WinogradAcc_2_27()

```

template<class Field , class FieldTrait >
void WinogradAcc_2_27 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.2.1.8 WinogradAcc_LR()

```

template<class Field , class FieldTrait >
void WinogradAcc_LR (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,

```

```

const size_t mr,
const size_t nr,
const size_t kr,
const typename Field::Element alpha,
typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.2.1.9 WinogradAcc_R_S()

```

template<class Field , class FieldTrait >
void WinogradAcc_R_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.2.1.10 WinogradAcc_L_S()

```

template<class Field , class FieldTrait >
void WinogradAcc_L_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.2.1.11 Winograd_LR_S()

```

template<class Field , class FieldTrait >
void Winograd_LR_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.2.1.12 Winograd_L_S()

```

template<class Field , class FieldTrait >
void Winograd_L_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.2.1.13 Winograd_R_S()

```

template<class Field , class FieldTrait >
void Winograd_R_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,

```

```

const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

11.3 FFLAS::csr_hyb_details Namespace Reference

Data Structures

- struct [Coo](#)
- struct [Info](#)

11.4 FFLAS::CuttingStrategy Namespace Reference

Data Structures

- struct [Block](#)
- struct [Column](#)
- struct [Recursive](#)
- struct [Row](#)
- struct [Single](#)

Typedefs

- typedef [Row](#) [RNSModulus](#)

11.4.1 Typedef Documentation

11.4.1.1 RNSModulus

```
typedef Row RNSModulus
```

11.5 FFLAS::details Namespace Reference

Functions

- template<class [Field](#) , bool ADD>
std::enable_if< [FFLAS::support_simd_add](#)< typenameField::Element >::value, void >::type [fadd](#)
(const [Field](#) &F, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t inca, typename [Field::ConstElement_ptr](#) B, const size_t incb, typename [Field::Element_ptr](#) C, const size_t incc, [FieldCategories::ModularTag](#))
- template<class [Field](#) , bool ADD>
std::enable_if<![FFLAS::support_simd_add](#)< typenameField::Element >::value, void >::type [fadd](#)
(const [Field](#) &F, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t inca, typename [Field::ConstElement_ptr](#) B, const size_t incb, typename [Field::Element_ptr](#) C, const size_t incc, [FieldCategories::ModularTag](#))

- `template<class Field, bool ADD>`
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, type-`
`name Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc,`
`FieldCategories::GenericTag)`
- `template<class Field, bool ADD>`
`std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd`
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc,`
`FieldCategories::UnparametricTag)`
- `template<class Field, bool ADD>`
`std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd`
`(const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename`
`Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc,`
`FieldCategories::UnparametricTag)`
- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce (const`
`Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)`
- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce`
`(const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, typename`
`Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)`
- `template<class Field, class FC >`
`void freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FC)`
- `template<class Field, class FC >`
`void freduce (const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, type-`
`name Field::Element_ptr A, const size_t incX, FC)`
- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal (const`
`Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX,`
`FieldCategories::ModularTag)`
- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal (const`
`Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t`
`incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`
- `template<class Field, class FC >`
`void fscal (const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X,`
`const size_t incX, FC)`
- `template<class Field, class FC >`
`void fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr`
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`
- `template<enum number_kind K>`
`void igebb44 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *bIA, const int64_t`
`*bIB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`
`void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *bIA, const int64_t`
`*bIB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`
`void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *bIA, const int64_t`
`*bIB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`
`void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *bIA, const int64_t`
`*bIB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`
`void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *bIA, const int64_t`
`*bIB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`
`void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *bIA, const int64_t`
`*bIB, int64_t *C, size_t ldc)`

- template<enum [number_kind](#) K>
void [igebp](#) (size_t rows, size_t cols, size_t depth, const [int64_t](#) alpha, const [int64_t](#) *blockA, size_t lda, const [int64_t](#) *blockB, size_t ldb, [int64_t](#) *C, size_t ldc)
- template<size_t k, bool transpose>
void [pack_lhs](#) ([int64_t](#) *XX, const [int64_t](#) *X, size_t ldx, size_t rows, size_t cols)
- template<size_t k, bool transpose>
void [pack_rhs](#) ([int64_t](#) *XX, const [int64_t](#) *X, size_t ldx, size_t rows, size_t cols)
- void [gebp](#) (size_t rows, size_t cols, size_t depth, [int64_t](#) *C, size_t ldc, const [int64_t](#) *blockA, size_t lda, const [int64_t](#) *BlockB, size_t ldb, [int64_t](#) *BlockW)
- void [BlockingFactor](#) (size_t &m, size_t &n, size_t &k)

11.5.1 Function Documentation

11.5.1.1 [fadd\(\)](#) [1/5]

```
template<class Field , bool ADD>
std::enable_if< FFLAS::support\_simd\_add< typenameField::Element >::value, void >::type fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::ModularTag )
```

11.5.1.2 [fadd\(\)](#) [2/5]

```
template<class Field , bool ADD>
std::enable_if<!FFLAS::support\_simd\_add< typenameField::Element >::value, void >::type fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::ModularTag )
```

11.5.1.3 [fadd\(\)](#) [3/5]

```
template<class Field , bool ADD>
void fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::GenericTag )
```

11.5.1.4 fadd() [4/5]

```
template<class Field , bool ADD>
std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::UnparametricTag ) [inline]
```

11.5.1.5 fadd() [5/5]

```
template<class Field , bool ADD>
std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::UnparametricTag ) [inline]
```

11.5.1.6 freduce() [1/4]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce
(
    const Field & F,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

11.5.1.7 freduce() [2/4]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce
(
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr B,
    const size_t incY,
    typename Field::Element_ptr A,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

11.5.1.8 freduce() [3/4]

```
template<class Field , class FC >
void freduce (
    const Field & F,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t incX,
    FC ) [inline]
```

11.5.1.9 freduce() [4/4]

```
template<class Field , class FC >
void freduce (
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr B,
    const size_t incY,
    typename Field::Element_ptr A,
    const size_t incX,
    FC ) [inline]
```

11.5.1.10 fscaln() [1/2]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscaln
(
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::Element_ptr X,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

11.5.1.11 fscl() [1/2]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscl
(
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FieldCategories::ModularTag ) [inline]
```


11.5.1.12 fscaln() [2/2]

```
template<class Field , class FC >
void fscaln (
    const Field & F,
    const size_t n,
    const typename Field::Element a,
    typename Field::Element_ptr X,
    const size_t incX,
    FC ) [inline]
```

11.5.1.13 fscal() [2/2]

```
template<class Field , class FC >
void fscal (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FC ) [inline]
```

11.5.1.14 igebb44()

```
template<enum number_kind K>
void igebb44 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * b1A,
    const int64_t * b1B,
    int64_t * C,
    size_t ldc ) [inline]
```

11.5.1.15 igebb24()

```
template<enum number_kind K>
void igebb24 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * b1A,
    const int64_t * b1B,
    int64_t * C,
    size_t ldc ) [inline]
```

11.5.1.16 igebb14()

```
template<enum number_kind K>
void igebb14 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

11.5.1.17 igebb41()

```
template<enum number_kind K>
void igebb41 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

bug ,B_0 dans VEC_MADD_32 ?

bug ,B_0 dans VEC_MADD_32 ?

11.5.1.18 igebb21()

```
template<enum number_kind K>
void igebb21 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

11.5.1.19 igebb11()

```
template<enum number_kind K>
void igebb11 (
    size_t i,
    size_t j,
    size_t depth,
```

```

    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]

```

11.5.1.20 igebp()

```

template<enum number_kind K>
void igebp (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * blockA,
    size_t lda,
    const int64_t * blockB,
    size_t ldb,
    int64_t * C,
    size_t ldc )

```

11.5.1.21 pack_lhs()

```

template<size_t k, bool transpose>
void pack_lhs (
    int64_t * XX,
    const int64_t * X,
    size_t ldx,
    size_t rows,
    size_t cols )

```

Bug this is fassign

Bug this is fassign

Bug this is fassign

Bug this is fassign

11.5.1.22 pack_rhs()

```

template<size_t k, bool transpose>
void pack_rhs (
    int64_t * XX,
    const int64_t * X,
    size_t ldx,
    size_t rows,
    size_t cols )

```

Bug this is fassign

Bug this is fassign

Bug this is fassign

Bug this is fassign

11.5.1.23 `gebp()`

```
void gebp (
    size_t rows,
    size_t cols,
    size_t depth,
    int64_t * C,
    size_t ldc,
    const int64_t * blockA,
    size_t lda,
    const int64_t * BlockB,
    size_t ldb,
    int64_t * BlockW )
```

11.5.1.24 `BlockingFactor()`

```
void BlockingFactor (
    size_t & m,
    size_t & n,
    size_t & k ) [inline]
```

11.6 `FFLAS::details_spmv` Namespace Reference

Data Structures

- struct [Coo](#)

11.7 `FFLAS::ElementCategories` Namespace Reference

Data Structures

- struct [ArbitraryPrecIntTag](#)
Arbitrary precision integers: GMP.
- struct [FixedPrecIntTag](#)
Fixed precision integers above machine precision: Givaro::reclnt.
- struct [GenericTag](#)
default is generic
- struct [MachineFloatTag](#)
float or double
- struct [MachineIntTag](#)
short, int, long, long long, and unsigned variants
- struct [RNSElementTag](#)
Representation in a Residue Number System.

11.8 `FFLAS::FieldCategories` Namespace Reference

Traits and categories will need to be placed in a proper file later.

Data Structures

- struct [GenericTag](#)
generic ring.
- struct [ModularTag](#)
This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`
- struct [UnparametricTag](#)
If the field uses a representation with infix operators.

11.8.1 Detailed Description

Traits and categories will need to be placed in a proper file later.

11.9 FFLAS::MMHelperAlgo Namespace Reference**Data Structures**

- struct [Auto](#)
- struct [Bini](#)
- struct [Classic](#)
- struct [Winograd](#)
- struct [WinogradPar](#)

11.10 FFLAS::ModeCategories Namespace Reference

Specifies the mode of action for an algorithm w.r.t.

Data Structures

- struct [ConvertTo](#)
Force conversion to appropriate element type of `ElementCategory T`.
- struct [DefaultBoundedTag](#)
Use standard field operations, but keeps track of bounds on input and output.
- struct [DefaultTag](#)
No specific mode of action: use standard field operations.
- struct [DelayedTag](#)
Performs field operations with delayed mod reductions. Ensures result is reduced.
- struct [LazyTag](#)
Performs field operations with delayed mod only when necessary. Result may not be reduced.

11.10.1 Detailed Description

Specifies the mode of action for an algorithm w.r.t.

its field

11.11 FFLAS::ParSeqHelper Namespace Reference

[ParSeqHelper](#) for both fgemm and ftrsm.

Data Structures

- struct [Compose](#)
- struct [Parallel](#)
- struct [Sequential](#)

11.11.1 Detailed Description

[ParSeqHelper](#) for both fgemm and ftrsm.

[ParSeqHelper](#) for both fgemm and ftrsm

11.12 FFLAS::Protected Namespace Reference

Data Structures

- class [AreEqual](#)
- class [AreEqual< X, X >](#)
- class [ftrmmLeftLowerNoTransNonUnit](#)
- class [ftrmmLeftLowerNoTransUnit](#)
- class [ftrmmLeftLowerTransNonUnit](#)
- class [ftrmmLeftLowerTransUnit](#)
- class [ftrmmLeftUpperNoTransNonUnit](#)
- class [ftrmmLeftUpperNoTransUnit](#)
- class [ftrmmLeftUpperTransNonUnit](#)
- class [ftrmmLeftUpperTransUnit](#)
- class [ftrmmRightLowerNoTransNonUnit](#)
- class [ftrmmRightLowerNoTransUnit](#)
- class [ftrmmRightLowerTransNonUnit](#)
- class [ftrmmRightLowerTransUnit](#)
- class [ftrmmRightUpperNoTransNonUnit](#)
- class [ftrmmRightUpperNoTransUnit](#)
- class [ftrmmRightUpperTransNonUnit](#)
- class [ftrmmRightUpperTransUnit](#)
- class [ftrsmLeftLowerNoTransNonUnit](#)
- class [ftrsmLeftLowerNoTransUnit](#)
- class [ftrsmLeftLowerTransNonUnit](#)
- class [ftrsmLeftLowerTransUnit](#)
- class [ftrsmLeftUpperNoTransNonUnit](#)

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

- class [ftrsmLeftUpperNoTransUnit](#)
- class [ftrsmLeftUpperTransNonUnit](#)
- class [ftrsmLeftUpperTransUnit](#)
- class [ftrsmRightLowerNoTransNonUnit](#)
- class [ftrsmRightLowerNoTransUnit](#)
- class [ftrsmRightLowerTransNonUnit](#)
- class [ftrsmRightLowerTransUnit](#)
- class [ftrsmRightUpperNoTransNonUnit](#)
- class [ftrsmRightUpperNoTransUnit](#)
- class [ftrsmRightUpperTransNonUnit](#)
- class [ftrsmRightUpperTransUnit](#)

Functions

- template<class [Field](#) >
double [computeFactorClassic](#) (const [Field](#) &F)
- template<> double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< double > &F)
- template<> double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< float > &F)
- template<class [Field](#) >
size_t [DotProdBoundClassic](#) (const [Field](#) &F, const typename [Field::Element](#) &beta)
- template<class [Field](#) >
size_t [TRSMBound](#) (const [Field](#) &)
TRSMBound.
- template<class [Element](#) >
size_t [TRSMBound](#) (const [Givaro::Modular](#)< [Element](#) > &F)
Specialization for positive modular representation over float.
- template<class [Element](#) >
size_t [TRSMBound](#) (const [Givaro::ModularBalanced](#)< [Element](#) > &F)
Specialization for balanced modular representation over double.
- template<class [NewField](#) , class [Field](#) , class [FieldMode](#) >
[Field::Element_ptr](#) [fgemm_convert](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) > &H)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ParSeqTrait](#) >
bool [NeedPreAddReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeCategories::LazyTag](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ModeT](#) , class [ParSeqTrait](#) >
bool [NeedPreAddReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeT](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ParSeqTrait](#) >
bool [NeedPreSubReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeCategories::LazyTag](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ModeT](#) , class [ParSeqTrait](#) >
bool [NeedPreSubReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeT](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ParSeqTrait](#) >
bool [NeedDoublePreAddReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [Element](#) beta, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeCategories::LazyTag](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ModeT](#) , class [ParSeqTrait](#) >
bool [NeedDoublePreAddReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [Element](#) beta, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeT](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [AlgoT](#) , class [ParSeqTrait](#) >
void [ScaAndReduce](#) (const [Field](#) &F, const size_t N, const typename [Field::Element](#) alpha, typename [Field::Element_ptr](#) X, const size_t incX, const [MMHelper](#)< [Field](#), [AlgoT](#), [ModeCategories::LazyTag](#), [ParSeqTrait](#) > &H)
- template<class [Field](#) , class [AlgoT](#) , class [ParSeqTrait](#) >
void [ScaAndReduce](#) (const [Field](#) &F, const size_t M, const size_t N, const typename [Field::Element](#) alpha, typename [Field::Element_ptr](#) A, const size_t lda, const [MMHelper](#)< [Field](#), [AlgoT](#), [ModeCategories::LazyTag](#), [ParSeqTrait](#) > &H)
- template<class [Field](#) >
[Field::Element_ptr](#) [fsquareCommon](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const size_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc)

- `template<class Field >`
`int WinogradThreshold (const Field &F)`
Computes the number of recursive levels to perform.
- `template<> int WinogradThreshold (const Givaro::Modular< float > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< double > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< float > &F)`
- `template<class Field >`
`int WinogradSteps (const Field &F, const size_t &m)`
Computes the number of recursive levels to perform.
- `template<class Field , class FieldMode >`
`void DynamicPeeling (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`
`void DynamicPeeling2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`
`void WinogradCalc (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`
- `template<typename FloatElement , class Field >`
`Field::Element_ptr fgemv_convert (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)`
- `template<class FloatElement , class Field >`
`void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`
- `template<class DFE >`
`size_t min_types (const DFE &k)`
- `template<> size_t min_types (const Reclnt::rint< 6 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 7 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 8 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 9 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 10 > &k)`
- `template<> size_t min_types (const Givaro::Integer &k)`
- `template<class T >`
`bool unfit (T x)`
- `template<> bool unfit (int64_t x)`
- `template<size_t K>`
`bool unfit (Reclnt::rint< K > x)`
- `template<> bool unfit (Reclnt::rint< 6 > x)`
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>`
`void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`

- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>`
`void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`
- `void igemm (const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, const int64_t beta, int64_t *C, size_t ldc)`
- `template<class Field >`
`void MatF2MatD_Triangular (const Field &F, Givaro::DoubleDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`
`void MatF2MatFI_Triangular (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`

11.12.1 Function Documentation

11.12.1.1 computeFactorClassic() [1/3]

```
template<class Field >
double computeFactorClassic (
    const Field & F ) [inline]
```

11.12.1.2 computeFactorClassic() [2/3]

```
template<>
double computeFactorClassic (
    const Givaro::ModularBalanced< double > & F ) [inline]
```

11.12.1.3 computeFactorClassic() [3/3]

```
template<>
double computeFactorClassic (
    const Givaro::ModularBalanced< float > & F ) [inline]
```

11.12.1.4 DotProdBoundClassic()

```
template<class Field >
size_t DotProdBoundClassic (
    const Field & F,
    const typename Field::Element & beta ) [inline]
```

11.12.1.5 TRSMBound() [1/3]

```
template<class Field >
size_t TRSMBound (
    const Field & ) [inline]
```

TRSMBound.

computes the maximal size for delaying the modular reduction in a triangular system resolution

This is the default version over an arbitrary field. It is currently never used (the recursive algorithm is run until $n=1$ in this case)

Parameters

F	Finite Field/Ring of the computation
-----	--------------------------------------

11.12.1.6 TRSMBound() [2/3]

```
template<class Element >
size_t TRSMBound (
    const Givaro::Modular< Element > & F ) [inline]
```

Specialization for positive modular representation over float.

Computes n_{\max} s.t. $(p-1)/2 * (p^{\{n_{\max}-1\}} + (p-2)^{\{n_{\max}-1\}}) < 2^{24}$ @pbi See [Dumas Giorgi Pernet 06, arXiv:cs/0601133]

11.12.1.7 TRSMBound() [3/3]

```
template<class Element >
size_t TRSMBound (
    const Givaro::ModularBalanced< Element > & F ) [inline]
```

Specialization for balanced modular representation over double.

Computes n_{\max} s.t. $(p-1)/2 * ((p+1)/2)^{\{n_{\max}-1\}} < 2^{53}$

Bibliography • Dumas Giorgi Pernet 06, arXiv:cs/0601133

11.12.1.8 fgemmm_convert()

```
template<class NewField , class Field , class FieldMode >
Field::Element_ptr fgemmm_convert (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H ) [inline]
```

11.12.1.9 NeedPreAddReduction() [1/2]

```
template<class Field , class Element , class AlgoT , class ParSeqTrait >
bool NeedPreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

11.12.1.10 NeedPreAddReduction() [2/2]

```
template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >
bool NeedPreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

11.12.1.11 NeedPreSubReduction() [1/2]

```
template<class Field , class Element , class AlgoT , class ParSeqTrait >
bool NeedPreSubReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

11.12.1.12 NeedPreSubReduction() [2/2]

```
template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >
bool NeedPreSubReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

11.12.1.13 NeedDoublePreAddReduction() [1/2]

```
template<class Field , class Element , class AlgoT , class ParSeqTrait >
bool NeedDoublePreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    Element beta,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

11.12.1.14 NeedDoublePreAddReduction() [2/2]

```
template<class Field , class Element , class AlgoT , class ModeT , class ParSeqTrait >
bool NeedDoublePreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    Element beta,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

11.12.1.15 ScalAndReduce() [1/2]

```
template<class Field , class AlgoT , class ParSeqTrait >
void ScalAndReduce (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr X,
    const size_t incX,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]
```

11.12.1.16 ScalAndReduce() [2/2]

```
template<class Field , class AlgoT , class ParSeqTrait >
void ScalAndReduce (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]
```

11.12.1.17 fsquareCommon()

```
template<class Field >
Field::Element_ptr fsquareCommon (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

11.12.1.18 WinogradThreshold() [1/4]

```
template<class Field >
int WinogradThreshold (
    const Field & F ) [inline]
```

Computes the number of recursive levels to perform.

Parameters

<i>m</i>	the common dimension in the product AxB
----------	---

11.12.1.19 WinogradThreshold() [2/4]

```
template<>
int WinogradThreshold (
    const Givaro::Modular< float > & F ) [inline]
```

11.12.1.20 WinogradThreshold() [3/4]

```
template<>
int WinogradThreshold (
    const Givaro::ModularBalanced< double > & F ) [inline]
```

11.12.1.21 WinogradThreshold() [4/4]

```
template<>
int WinogradThreshold (
    const Givaro::ModularBalanced< float > & F ) [inline]
```

11.12.1.22 WinogradSteps()

```
template<class Field >
int WinogradSteps (
    const Field & F,
    const size_t & m ) [inline]
```

Computes the number of recursive levels to perform.

Parameters

m	the common dimension in the product $A \times B$
-----	--

11.12.1.23 DynamicPeeling()

```

template<class Field , class FieldMode >
void DynamicPeeling (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmin,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmax ) [inline]

```

11.12.1.24 DynamicPeeling2()

```

template<class Field , class FieldMode >
void DynamicPeeling2 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,

```

```

        const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmin,
        const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmax ) [inline]

```

11.12.1.25 WinogradCalc()

```

template<class Field , class FieldMode >
void WinogradCalc (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H ) [inline]

```

11.12.1.26 fgemv_convert()

```

template<typename FloatElement , class Field >
Field::Element_ptr fgemv_convert (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]

```

11.12.1.27 fger_convert()

```

template<class FloatElement , class Field >
void fger_convert (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]

```

11.12.1.28 min_types() [1/7]

```
template<class DFE >
size_t min_types (
    const DFE & k ) [inline]
```

11.12.1.29 min_types() [2/7]

```
template<>
size_t min_types (
    const RecInt::rint< 6 > & k ) [inline]
```

11.12.1.30 min_types() [3/7]

```
template<>
size_t min_types (
    const RecInt::rint< 7 > & k ) [inline]
```

11.12.1.31 min_types() [4/7]

```
template<>
size_t min_types (
    const RecInt::rint< 8 > & k ) [inline]
```

11.12.1.32 min_types() [5/7]

```
template<>
size_t min_types (
    const RecInt::rint< 9 > & k ) [inline]
```

11.12.1.33 min_types() [6/7]

```
template<>
size_t min_types (
    const RecInt::rint< 10 > & k ) [inline]
```

11.12.1.34 min_types() [7/7]

```
template<>
size_t min_types (
    const Givaro::Integer & k ) [inline]
```

11.12.1.35 unfit() [1/4]

```
template<class T >
bool unfit (
    T x ) [inline]
```


11.12.1.36 unfit() [2/4]

```
template<>
bool unfit (
    int64_t x ) [inline]
```

11.12.1.37 unfit() [3/4]

```
template<size_t K>
bool unfit (
    RecInt::rint< K > x ) [inline]
```

11.12.1.38 unfit() [4/4]

```
template<>
bool unfit (
    RecInt::rint< 6 > x ) [inline]
```

11.12.1.39 igemm_colmajor() [1/2]

```
template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>
void igemm_colmajor (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

11.12.1.40 igemm_colmajor() [2/2]

```
template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>
void igemm_colmajor (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

11.12.1.41 igemm()

```
void igemm (
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_TRANSPOSE TransB,
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    const int64_t beta,
    int64_t * C,
    size_t ldc ) [inline]
```

Todo use primitive (no `Field()`) and specialise for int64.

Todo use primitive (no `Field()`) and specialise for int64.

11.12.1.42 MatF2MatD_Triangular()

```
template<class Field >
void MatF2MatD_Triangular (
    const Field & F,
    Givaro::DoubleDomain::Element_ptr S,
    const size_t lds,
    typename Field::ConstElement_ptr const E,
    const size_t lde,
    const size_t m,
    const size_t n )
```

11.12.1.43 MatF2MatFI_Triangular()

```
template<class Field >
void MatF2MatFI_Triangular (
    const Field & F,
    Givaro::FloatDomain::Element_ptr S,
    const size_t lds,
    typename Field::ConstElement_ptr const E,
    const size_t lde,
    const size_t m,
    const size_t n )
```

Todo do `finit(...,FFLAS_TRANS,FFLAS_DIAG)`
do `fconvert(...,FFLAS_TRANS,FFLAS_DIAG)`

11.13 FFLAS::sell_details Namespace Reference

Data Structures

- struct [Coo](#)
- struct [Info](#)

11.14 FFLAS::sparse_details Namespace Reference

Functions

- template<class [Field](#) >
void [init_y](#) (const [Field](#) &F, const size_t m, const typename [Field::Element](#) b, typename [Field::Element_ptr](#) y)
- template<class [Field](#) >
void [init_y](#) (const [Field](#) &F, const size_t m, const size_t n, const typename [Field::Element](#) b, typename [Field::Element_ptr](#) y, const int ldy)
- template<class [Field](#) , class SM , class FC , class MZO >
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineFloat](#) >::value)|std::is_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineIntTag](#) >::value)>::type [fspmv_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, FC fc, MZO mzo)
- template<class [Field](#) , class SM , class FC , class MZO >
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineFloat](#) >::value)|std::is_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineIntTag](#) >::value >::type [fspmv_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, FC fc, MZO mzo)
- template<class [Field](#) , class SM >
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::GenericTag](#), [NotZOSparseMatrix](#))
- template<class [Field](#) , class SM >
std::enable_if<[isSparseMatrixSimdFormat](#)< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
- template<class [Field](#) , class SM >
std::enable_if< [isSparseMatrixSimdFormat](#)< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
- template<class [Field](#) , class SM >
std::enable_if<[isSparseMatrixSimdFormat](#)< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::ModularTag](#), [NotZOSparseMatrix](#))
- template<class [Field](#) , class SM >
std::enable_if< [isSparseMatrixSimdFormat](#)< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::ModularTag](#), [NotZOSparseMatrix](#))
- template<class [Field](#) , class SM >
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::GenericTag](#), [ZOSparseMatrix](#))
- template<class [Field](#) , class SM >
std::enable_if<[isSparseMatrixSimdFormat](#)< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::UnparametricTag](#), [ZOSparseMatrix](#))

- `template<class Field , class SM >`
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM`
`&A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag,`
`ZOSparseMatrix)`
- `template<class Field , class SM >`
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr`
`y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat`
`>::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag`
`>::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr`
`x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat`
`>::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag`
`>::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr`
`x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,`
`typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`
`FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`
`std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`
`FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`
`FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`
`std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`
`FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,`
`typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`
`FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`
`std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`
`FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,`
`typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM , class FCat , class MZO >`
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat`
`>::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag`
`>::value)>::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename`
`Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat`

- >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineIntTag](#) >::value >::type [pfspmm_dispatch](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FCat](#), [MZO](#))
- template<class [Field](#) , class [SM](#) >
void [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::GenericTag](#), [NotZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
std::enable_if< [support_simd](#)< typenameField::Element >::value >::type [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
std::enable_if< ![support_simd](#)< typenameField::Element >::value >::type [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
std::enable_if< [support_simd](#)< typenameField::Element >::value >::type [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::ModularTag](#), [NotZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
std::enable_if< ![support_simd](#)< typenameField::Element >::value >::type [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::ModularTag](#), [NotZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
void [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::GenericTag](#), [ZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
std::enable_if< [support_simd](#)< typenameField::Element >::value >::type [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#), [ZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
std::enable_if< ![support_simd](#)< typenameField::Element >::value >::type [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#), [ZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
void [pfspmm](#) (const [Field](#) &F, const [SM](#) &A, [size_t](#) blockSize, typename [Field::ConstElement_ptr](#) x, int ldx, typename [Field::Element_ptr](#) y, int ldy, [FieldCategories::ModularTag](#), [ZOSparseMatrix](#))
 - template<class [Field](#) , class [SM](#) >
void [pfspmv](#) (const [Field](#) &F, const [SM](#) &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::GenericTag](#), std::false_type)
 - template<class [Field](#) , class [SM](#) >
void [pfspmv](#) (const [Field](#) &F, const [SM](#) &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::UnparametricTag](#), std::false_type)
 - template<class [Field](#) , class [SM](#) >
void [pfspmv](#) (const [Field](#) &F, const [SM](#) &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::ModularTag](#), std::false_type)
 - template<class [Field](#) , class [SM](#) >
void [pfspmv](#) (const [Field](#) &F, const [SM](#) &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::GenericTag](#), std::true_type)
 - template<class [Field](#) , class [SM](#) >
void [pfspmv](#) (const [Field](#) &F, const [SM](#) &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::UnparametricTag](#), std::true_type)
 - template<class [Field](#) , class [SM](#) >
void [pfspmv](#) (const [Field](#) &F, const [SM](#) &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::ModularTag](#), std::true_type)
 - template<class [Field](#) , class [SM](#) >
std::enable_if< [isSparseMatrixSimdFormat](#)< [Field](#), [SM](#) >::value &&[support_simd](#)< typenameField::Element >::value >::type [fspmv](#) (const [Field](#) &F, const [SM](#) &A, typename [Field::ConstElement_ptr](#) x, typename [Field::Element_ptr](#) y, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))

- `template<class Field , class SM >`
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element`
`>::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename`
`Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element`
`>::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename`
`Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`

11.14.1 Function Documentation

11.14.1.1 init_y() [1/2]

```
template<class Field >
void init_y (
    const Field & F,
    const size_t m,
    const typename Field::Element b,
    typename Field::Element_ptr y ) [inline]
```

11.14.1.2 init_y() [2/2]

```
template<class Field >
void init_y (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element b,
    typename Field::Element_ptr y,
    const int ldy ) [inline]
```

11.14.1.3 fspmv_dispatch() [1/2]

```
template<class Field , class SM , class FC , class MZO >
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value)>::type fspmv_dispatch (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FC fc,
    MZO mzo ) [inline]
```

11.14.1.4 fspmv_dispatch() [2/2]

```
template<class Field , class SM , class FC , class MZO >
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value >::type fspmv_dispatch (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FC fc,
    MZO mzo ) [inline]
```

11.14.1.5 fspmv() [1/12]

```
template<class Field , class SM >
void fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.6 fspmv() [2/12]

```
template<class Field , class SM >
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.7 fspmv() [3/12]

```
template<class Field , class SM >
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.8 fspmv() [4/12]

```
template<class Field , class SM >
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.9 fspmv() [5/12]

```
template<class Field , class SM >
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.10 fspmv() [6/12]

```
template<class Field , class SM >
void fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.11 fspmv() [7/12]

```
template<class Field , class SM >
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.12 fspmv() [8/12]

```
template<class Field , class SM >
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.13 fspmv() [9/12]

```
template<class Field , class SM >
void fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::true_type ) [inline]
```


11.14.1.14 fspmm_dispatch() [1/2]

```
template<class Field , class SM , class FCat , class MZO >
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value)>::type fspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

11.14.1.15 fspmm_dispatch() [2/2]

```
template<class Field , class SM , class FCat , class MZO >
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value >::type fspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

11.14.1.16 fspmm() [1/9]

```
template<class Field , class SM >
void fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.17 fspmm() [2/9]

```
template<class Field , class SM >
std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
```

```

    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

11.14.1.18 fspmm() [3/9]

```

template<class Field , class SM >
std::enable_if<!support\_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

11.14.1.19 fspmm() [4/9]

```

template<class Field , class SM >
std::enable_if< support\_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]

```

11.14.1.20 fspmm() [5/9]

```

template<class Field , class SM >
std::enable_if<!support\_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]

```

11.14.1.21 fspmm() [6/9]

```
template<class Field , class SM >
void fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.22 fspmm() [7/9]

```
template<class Field , class SM >
std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.23 fspmm() [8/9]

```
template<class Field , class SM >
std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.24 fspmm() [9/9]

```
template<class Field , class SM >
void fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.25 pfspmm_dispatch() [1/2]

```
template<class Field , class SM , class FCat , class MZO >
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value)>::type pfspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

11.14.1.26 pfspmm_dispatch() [2/2]

```
template<class Field , class SM , class FCat , class MZO >
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value >::type pfspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]
```

11.14.1.27 pfspmm() [1/9]

```
template<class Field , class SM >
void pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.28 pfspmm() [2/9]

```
template<class Field , class SM >
std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
```

```

    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

11.14.1.29 pfspmm() [3/9]

```

template<class Field , class SM >
std::enable_if<!support\_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

11.14.1.30 pfspmm() [4/9]

```

template<class Field , class SM >
std::enable_if< support\_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]

```

11.14.1.31 pfspmm() [5/9]

```

template<class Field , class SM >
std::enable_if<!support\_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]

```

11.14.1.32 pfspmm() [6/9]

```
template<class Field , class SM >
void pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.33 pfspmm() [7/9]

```
template<class Field , class SM >
std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.34 pfspmm() [8/9]

```
template<class Field , class SM >
std::enable_if< !support_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.35 pfspmm() [9/9]

```
template<class Field , class SM >
void pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    ZOSparseMatrix ) [inline]
```

11.14.1.36 pfspmv() [1/6]

```
template<class Field , class SM >
void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    std::false_type ) [inline]
```

11.14.1.37 pfspmv() [2/6]

```
template<class Field , class SM >
void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    std::false_type ) [inline]
```

11.14.1.38 pfspmv() [3/6]

```
template<class Field , class SM >
void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::false_type ) [inline]
```

11.14.1.39 pfspmv() [4/6]

```
template<class Field , class SM >
void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    std::true_type ) [inline]
```

11.14.1.40 pfspmv() [5/6]

```
template<class Field , class SM >
void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    std::true_type ) [inline]
```

11.14.1.41 pfspmv() [6/6]

```
template<class Field , class SM >
void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::true_type ) [inline]
```

11.14.1.42 fspmv() [10/12]

```
template<class Field , class SM >
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↵
::Element >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.43 fspmv() [11/12]

```
template<class Field , class SM >
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↵
::Element >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

11.14.1.44 fspmv() [12/12]

```
template<class Field , class SM >
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↵
::Element >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```


11.15 FFLAS::sparse_details_impl Namespace Reference

Functions

- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`

- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

- `template<class Field >`
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t blockSize, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`

- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`

- `template<class Field >`
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`
- `template<class Field, class Func >`
`void pfsppmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, Func &&func)`
- `template<class Field, class Func >`
`void pfsppmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)`
- `template<class Field >`
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`
`void pfsppmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfsppmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfsppmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfsppmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`

- `template<class Field >`
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`

- `template<class Field >`
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`

- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`

- `template<class Field >`
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

11.15.1 Function Documentation

11.15.1.1 fspmm() [1/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.2 fspmm() [2/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.3 fspmm() [3/15]

```

template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

11.15.1.4 fspmm_simd_aligned() [1/2]

```

template<class Field >
void fspmm_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

11.15.1.5 fspmm_simd_unaligned() [1/2]

```

template<class Field >
void fspmm_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

11.15.1.6 fspmm_one() [1/4]

```

template<class Field >
void fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

11.15.1.7 fspmm_mone() [1/4]

```
template<class Field >
void fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.8 fspmm_one_simd_aligned() [1/3]

```
template<class Field >
void fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.9 fspmm_one_simd_unaligned() [1/3]

```
template<class Field >
void fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.10 fspmm_mone_simd_aligned() [1/3]

```
template<class Field >
void fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.11 fspmm_mone_simd_unaligned() [1/3]

```
template<class Field >
void fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.12 fspmv() [1/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.13 fspmv() [2/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.14 fspmv() [3/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

11.15.1.15 fspmv_one() [1/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.16 fspmv_mone() [1/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.17 fspmv_one() [2/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.18 fspmv_mone() [2/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.19 pfspmm() [1/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.20 pfspmm() [2/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.21 pfspmm() [3/18]

```

template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

11.15.1.22 pfspmm_one() [1/2]

```

template<class Field >
void pfspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

11.15.1.23 pfspmm_mone() [1/2]

```

template<class Field >
void pfspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

11.15.1.24 pfspmm_one() [2/2]

```

template<class Field >
void pfspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

11.15.1.25 pfspmm_mone() [2/2]

```
template<class Field >
void pfspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.26 pfspmv() [1/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.27 pfspmv_task()

```
template<class Field >
void pfspmv_task (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const index_t iStart,
    const index_t iStop,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.28 pfspmv() [2/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.29 pfspmv() [3/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

11.15.1.30 pfspmv_one() [1/8]

```
template<class Field >
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.31 pfspmv_mone() [1/8]

```
template<class Field >
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.32 pfspmv_one() [2/8]

```
template<class Field >
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.33 pfspmv_mone() [2/8]

```
template<class Field >
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.34 fspmm() [4/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```


11.15.1.35 fspmm() [5/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    index_t blockSize,
    typename Field::ConstElement_ptr x_,
    index_t ldx,
    typename Field::Element_ptr y_,
    index_t ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.36 fspmm_simd_aligned() [2/2]

```
template<class Field >
void fspmm_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.37 fspmm_simd_unaligned() [2/2]

```
template<class Field >
void fspmm_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.38 fspmm() [6/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

11.15.139 fspmm_one() [2/4]

```

template<class Field >
void fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

11.15.140 fspmm_mone() [2/4]

```

template<class Field >
void fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

11.15.141 fspmm_one_simd_aligned() [2/3]

```

template<class Field >
void fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

11.15.142 fspmm_one_simd_unaligned() [2/3]

```

template<class Field >
void fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

11.15.1.43 fspmm_mone_simd_aligned() [2/3]

```
template<class Field >
void fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.44 fspmm_mone_simd_unaligned() [2/3]

```
template<class Field >
void fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.45 fspmv() [4/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.46 fspmv() [5/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.47 fspmv() [6/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

11.15.148 fspmv_one() [3/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.149 fspmv_mone() [3/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.150 fspmv_one() [4/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.151 fspmv_mone() [4/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.152 pfspmm() [4/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

11.15.153 pfspmm() [5/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.154 pfspmm() [6/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.155 pfspmm() [7/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.156 pfspmm() [8/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    const int64_t kmax ) [inline]
```

11.15.1.57 pfspmm() [9/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    const int64_t kmax ) [inline]
```

11.15.1.58 pfspmv() [4/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.59 pfspmv() [5/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.60 pfspmv() [6/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

11.15.1.61 fspmm() [7/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.62 fspmm() [8/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.63 fspmm() [9/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

11.15.1.64 fspmv() [7/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.65 fspmv() [8/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.66 fspmv() [9/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

11.15.1.67 pfspmm() [10/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.68 pfspmm() [11/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.69 pfspmm() [12/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.70 pfspmm() [13/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```


11.15.1.71 pfspmm() [14/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    const int64_t kmax ) [inline]
```

11.15.1.72 pfspmm() [15/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    const int64_t kmax ) [inline]
```

11.15.1.73 pfspmm_zo() [1/2]

```
template<class Field , class Func >
void pfspmm_zo (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    Func && func ) [inline]
```

11.15.1.74 pfspmm_zo() [2/2]

```
template<class Field , class Func >
void pfspmm_zo (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    Func && func ) [inline]
```

11.15.1.75 pfspmv() [7/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.76 pfspmv() [8/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.77 pfspmv() [9/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

11.15.1.78 pfspmv_one() [3/8]

```
template<class Field >
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.79 pfspmv_mone() [3/8]

```
template<class Field >
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.80 pfspmv_one() [4/8]

```
template<class Field >
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.81 pfspmv_mone() [4/8]

```
template<class Field >
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.82 fspmm() [10/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.83 fspmm() [11/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.84 fspmm() [12/15]

```

template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

11.15.1.85 fspmm_mone() [3/4]

```

template<class Field >
void fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

11.15.1.86 fspmm_one() [3/4]

```

template<class Field >
void fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

11.15.1.87 fspmm_mone() [4/4]

```

template<class Field >
void fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

11.15.1.88 fspmm_one() [4/4]

```
template<class Field >
void fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.89 fspmm_one_simd_aligned() [3/3]

```
template<class Field >
void fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.90 fspmm_one_simd_unaligned() [3/3]

```
template<class Field >
void fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.91 fspmm_mone_simd_aligned() [3/3]

```
template<class Field >
void fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.92 fspmm_mone_simd_unaligned() [3/3]

```
template<class Field >
void fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldz,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.93 fspmv() [10/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.94 fspmv() [11/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.95 fspmv() [12/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

11.15.1.96 fspmv_one() [5/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.97 fspmv_mone() [5/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.98 fspmv_one() [6/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.99 fspmv_mone() [6/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.100 pfspmv() [10/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.101 pfspmv() [11/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.102 pfspmv() [12/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

11.15.1.103 pfspmv_one() [5/8]

```
template<class Field >
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.104 pfspmv_mone() [5/8]

```
template<class Field >
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.105 pfspmv_one() [6/8]

```
template<class Field >
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.106 pfspmv_mone() [6/8]

```
template<class Field >
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```


11.15.1.107 fspmv() [13/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.108 fspmv_simd() [1/4]

```
template<class Field >
void fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.109 fspmv() [14/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.110 fspmv_simd() [2/4]

```
template<class Field >
void fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

11.15.1.111 fspmv() [15/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

11.15.1.112 fspmv_one() [7/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.113 fspmv_mone() [7/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.114 fspmv_one() [8/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.115 fspmv_mone() [8/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.116 fspmv_one_simd() [1/2]

```
template<class Field >
void fspmv_one_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.117 fspmv_mone_simd() [1/2]

```
template<class Field >
void fspmv_mone_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.118 pfspmm() [16/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.119 pfspmm() [17/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.120 pfspmm() [18/18]

```
template<class Field >
void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    uint64_t kmax ) [inline]
```

11.15.1.121 pfspmv() [13/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.122 pfspmv() [14/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.123 pfspmv() [15/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    uint64_t kmax ) [inline]
```

11.15.1.124 fspmm() [13/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.125 fspmm() [14/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.126 fspmm() [15/15]

```
template<class Field >
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    uint64_t kmax ) [inline]
```

11.15.1.127 fspmv() [16/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.128 fspmv() [17/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.129 fspmv() [18/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    uint64_t kmax ) [inline]
```

11.15.1.130 pfspmv() [16/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.131 pfspmv() [17/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.132 pfspmv() [18/18]

```
template<class Field >
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

11.15.1.133 pfspmv_one() [7/8]

```
template<class Field >
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.134 pfspmv_mone() [7/8]

```
template<class Field >
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.135 pfspmv_one() [8/8]

```
template<class Field >
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.136 pfspmv_mone() [8/8]

```
template<class Field >
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.137 fspmv() [19/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.138 fspmv_simd() [3/4]

```
template<class Field >
void fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.139 fspmv() [20/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.140 fspmv_simd() [4/4]

```
template<class Field >
void fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

11.15.1.141 fspmv() [21/21]

```
template<class Field >
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

11.15.1.142 fspmv_one() [9/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.143 fspmv_mone() [9/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

11.15.1.144 fspmv_one_simd() [2/2]

```
template<class Field >
void fspmv_one_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.145 fspmv_mone_simd() [2/2]

```
template<class Field >
void fspmv_mone_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```


11.15.1.146 fspmv_one() [10/10]

```
template<class Field >
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.15.1.147 fspmv_mone() [10/10]

```
template<class Field >
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

11.16 FFLAS::StrategyParameter Namespace Reference**Data Structures**

- struct [Fixed](#)
- struct [Grain](#)
- struct [Threads](#)
- struct [ThreeD](#)
- struct [ThreeDAdaptive](#)
- struct [ThreeDInPlace](#)
- struct [TwoD](#)
- struct [TwoDAdaptive](#)

11.17 FFLAS::StructureHelper Namespace Reference

[StructureHelper](#) for ftrsm.

Data Structures

- struct [Hybrid](#)
- struct [Iterative](#)
- struct [Recursive](#)

11.17.1 Detailed Description

[StructureHelper](#) for ftrsm.

11.18 FFLAS::vectorised Namespace Reference

Namespaces

- namespace [unswitch](#)

Data Structures

- struct [HelperMod](#)
- struct [HelperMod](#)< Field, ElementCategories::MachineIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::FixedPrecIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::MachineFloatTag >

Functions

- template<class SimdT, class Element, bool positive>
std::enable_if< [is_simd](#)< SimdT >::value, void >::type [VEC_ADD](#) (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2 >
std::enable_if< [FFLAS::support_simd_add](#)< Element >::value, void >::type [addp](#) (Element *T, const Element *TA, const Element *TB, size_t n, Element p, T1 min_, T2 max_)
- template<class SimdT, class Element, bool positive>
std::enable_if< [is_simd](#)< SimdT >::value, void >::type [VEC_SUB](#) (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2 >
std::enable_if< [FFLAS::support_simd_add](#)< Element >::value, void >::type [subp](#) (Element *T, const Element *TA, const Element *TB, const size_t n, const Element p, const T1 min_, const T2 max_)
- template<class Element >
std::enable_if< [FFLAS::support_simd_add](#)< Element >::value, void >::type [add](#) (Element *T, const Element *TA, const Element *TB, size_t n)
- template<class Element >
std::enable_if< [FFLAS::support_simd_add](#)< Element >::value, void >::type [sub](#) (Element *T, const Element *TA, const Element *TB, size_t n)
- template<class T >
std::enable_if<!std::is_integral< T >::value, T >::type [reduce](#) (T A, T B)
- template<class T >
std::enable_if< std::is_integral< T >::value, T >::type [reduce](#) (T A, T B)
- template<> Givaro::Integer [reduce](#) (Givaro::Integer A, Givaro::Integer B)
- float [reduce](#) (float A, float B, float invB, float min, float max)
- double [reduce](#) (double A, double B, double invB, double min, double max)
- [int64_t reduce](#) ([int64_t](#) A, [int64_t](#) p, double invp, double min, double max, [int64_t](#) pow50rem)
- template<class Field >
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< Field, ElementCategories::MachineIntTag > &H)
- template<class Field >
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< Field, ElementCategories::MachineFloatTag > &H)
- template<class Field >
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< Field, ElementCategories::ArbitraryPrecIntTag > &H)
- template<class Field >
std::enable_if< [FFLAS::support_fast_mod](#)< typename [Field::Element](#) >::value, void >::type [modp](#) (const [Field](#) &F, typename [Field::ConstElement_ptr](#) U, const size_t &n, typename [Field::Element_ptr](#) T)

- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp`
`(const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename`
`Field::Element_ptr T)`
- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp`
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr U, const size_t n)`
- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp`
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr U, const size_t n, const size_t &incX)`

11.18.1 Function Documentation

11.18.1.1 VEC_ADD()

```
template<class SimdT , class Element , bool positive>
std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (
    SimdT & C,
    SimdT & A,
    SimdT & B,
    SimdT & Q,
    SimdT & T,
    SimdT & P,
    SimdT & NEGP,
    SimdT & MIN,
    SimdT & MAX ) [inline]
```

11.18.1.2 addp()

```
template<bool positive, class Element , class T1 , class T2 >
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n,
    Element p,
    T1 min_,
    T2 max_ ) [inline]
```

11.18.1.3 VEC_SUB()

```
template<class SimdT , class Element , bool positive>
std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (
    SimdT & C,
    SimdT & A,
    SimdT & B,
    SimdT & Q,
    SimdT & T,
    SimdT & P,
    SimdT & NEGP,
    SimdT & MIN,
    SimdT & MAX ) [inline]
```

11.18.1.4 subp()

```
template<bool positive, class Element , class T1 , class T2 >
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (
    Element * T,
    const Element * TA,
    const Element * TB,
    const size_t n,
    const Element p,
    const T1 min_,
    const T2 max_ ) [inline]
```

11.18.1.5 add()

```
template<class Element >
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n ) [inline]
```

11.18.1.6 sub()

```
template<class Element >
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n ) [inline]
```

11.18.1.7 reduce() [1/9]

```
template<class T >
std::enable_if<!std::is_integral< T >::value, T >::type reduce (
    T A,
    T B ) [inline]
```

11.18.1.8 reduce() [2/9]

```
template<class T >
std::enable_if< std::is_integral< T >::value, T >::type reduce (
    T A,
    T B ) [inline]
```

11.18.1.9 reduce() [3/9]

```
template<>
Givaro::Integer reduce (
    Givaro::Integer A,
    Givaro::Integer B ) [inline]
```

11.18.1.10 reduce() [4/9]

```
float reduce (
    float A,
    float B,
    float invB,
    float min,
    float max ) [inline]
```

11.18.1.11 reduce() [5/9]

```
double reduce (
    double A,
    double B,
    double invB,
    double min,
    double max ) [inline]
```

11.18.1.12 reduce() [6/9]

```
int64_t reduce (
    int64_t A,
    int64_t p,
    double invp,
    double min,
    double max,
    int64_t pow50rem ) [inline]
```

11.18.1.13 reduce() [7/9]

```
template<class Field >
Field::Element reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::MachineIntTag > & H ) [inline]
```

11.18.1.14 reduce() [8/9]

```
template<class Field >
Field::Element reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::MachineFloatTag > & H ) [inline]
```

11.18.1.15 reduce() [9/9]

```
template<class Field >
Field::Element reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > & H ) [inline]
```

11.18.1.16 modp() [1/2]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    typename Field::Element_ptr T ) [inline]
```

11.18.1.17 modp() [2/2]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    const size_t & incX,
    typename Field::Element_ptr T ) [inline]
```

11.18.1.18 scalp() [1/2]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n ) [inline]
```

11.18.1.19 scalp() [2/2]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX ) [inline]
```

11.19 FFLAS::vectorised::unswitch Namespace Reference**Functions**

- template<class Field >
std::enable_if< !FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, typename Field::Element_ptr T, HelperMod< Field > &H)

- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp`
`(const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename`
`Field::Element_ptr T, HelperMod< Field > &H)`
- `template<class Field >`
`std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<`
`typenameField::Element >::value, void >::type scalp` `(const Field &F, typename Field::Element_ptr T, const`
`typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, HelperMod< Field >`
`&H)`
- `template<class Field >`
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp`
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr U, const size_t n, const size_t &incX, HelperMod< Field > &H)`

11.19.1 Function Documentation

11.19.1.1 modp() [1/2]

```
template<class Field >
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<
typenameField::Element >::value, void >::type modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    typename Field::Element_ptr T,
    HelperMod< Field > & H ) [inline]
```

11.19.1.2 modp() [2/2]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    const size_t & incX,
    typename Field::Element_ptr T,
    HelperMod< Field > & H ) [inline]
```

11.19.1.3 scalp() [1/2]

```
template<class Field >
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<
typenameField::Element >::value, void >::type scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    HelperMod< Field > & H ) [inline]
```

11.19.1.4 `scalp()` [2/2]

```
template<class Field >
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX,
    HelperMod< Field > & H ) [inline]
```

11.20 FFPACK Namespace Reference

Finite Field **PACK** Set of elimination based routines for dense linear algebra.

Namespaces

- namespace [Protected](#)

Data Structures

- class [callLUdivine_small](#)
- class [callLUdivine_small< double >](#)
- class [callLUdivine_small< float >](#)
- class [CharpolyFailed](#)
- class [CheckerImplem_charpoly](#)
- class [CheckerImplem_Det](#)
- class [CheckerImplem_invert](#)
- class [CheckerImplem_PLUQ](#)
- class [Failure](#)
 - A precondition failed.*
- struct [rns_double](#)
- struct [rns_double_elt](#)
- struct [rns_double_elt_cstptr](#)
- struct [rns_double_elt_ptr](#)
- struct [rns_double_extended](#)
- class [RNSInteger](#)
- class [RNSIntegerMod](#)
- class [rnsRandIter](#)

Typedefs

- template<class [Field](#) >
using [Checker_PLUQ](#) = [FFLAS::Checker_Empty](#)< [Field](#) >
- template<class [Field](#) >
using [Checker_Det](#) = [FFLAS::Checker_Empty](#)< [Field](#) >
- template<class [Field](#) >
using [Checker_invert](#) = [FFLAS::Checker_Empty](#)< [Field](#) >
- template<class [Field](#) , class Polynomial >
using [Checker_charpoly](#) = [FFLAS::Checker_Empty](#)< [Field](#) >
- template<class [Field](#) >
using [ForceCheck_PLUQ](#) = [CheckerImplem_PLUQ](#)< [Field](#) >
- template<class [Field](#) >
using [ForceCheck_Det](#) = [CheckerImplem_Det](#)< [Field](#) >
- template<class [Field](#) >
using [ForceCheck_invert](#) = [CheckerImplem_invert](#)< [Field](#) >
- template<class [Field](#) , class Polynomial >
using [ForceCheck_charpoly](#) = [CheckerImplem_charpoly](#)< [Field](#) , Polynomial >

Functions

- void [LAPACKPerm2MathPerm](#) (size_t *MathP, const size_t *LapackP, const size_t N)
Conversion of a permutation from LAPACK format to Math format.
- void [MathPerm2LAPACKPerm](#) (size_t *LapackP, const size_t *MathP, const size_t N)
Conversion of a permutation from Maths format to LAPACK format.
- template<class [Field](#) >
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const [FFLAS::FFLAS_TRANSPOSE](#) Trans, const size_t M, const size_t ibeg, const size_t iend, typename [Field::Element_ptr](#) A, const size_t lda, const size_t *P)
Computes $P1 \times \text{Diag}(I_R, P2)$ where $P1$ is a LAPACK and $P2$ a LAPACK permutation and store the result in $P1$ as a LAPACK permutation.
- template<class [Field](#) >
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const [FFLAS::FFLAS_TRANSPOSE](#) Trans, const size_t m, const size_t ibeg, const size_t iend, typename [Field::Element_ptr](#) A, const size_t lda, const size_t *P, const [FFLAS::ParSeqHelper::Sequential](#) seq)
- template<class [Field](#) , class Cut , class Param >
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const [FFLAS::FFLAS_TRANSPOSE](#) Trans, const size_t m, const size_t ibeg, const size_t iend, typename [Field::Element_ptr](#) A, const size_t lda, const size_t *P, const [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > par)
- template<class [Field](#) >
void [MonotonicApplyP](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const [FFLAS::FFLAS_TRANSPOSE](#) Trans, const size_t M, const size_t ibeg, const size_t iend, typename [Field::Element_ptr](#) A, const size_t lda, const size_t *P, const size_t R)
Apply a R-monotonically increasing permutation P, to the matrix A.
- template<class [Field](#) >
void [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, const size_t R, typename [Field::Element_ptr](#) A, const size_t lda, const size_t *P, const size_t *Q, typename [Field::Element_ptr](#) B, const size_t ldb, int *info)
Solve the system $AX = B$ or $XA = B$.
- template<class [Field](#) >
[Field::Element_ptr](#) [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, typename [Field::Element_ptr](#) A, const size_t lda, const size_t *P, const size_t *Q, typename [Field::Element_ptr](#) X, const size_t ldx, typename [Field::ConstElement_ptr](#) B, const size_t ldb, int *info)
Solve the system $AX = B$ or $XA = B$.

- template<class [Field](#) >
size_t [fgesv](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb, int *info)
Square system solver.
- template<class [Field](#) >
size_t [fgesv](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, const size_t NRHS, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) X, const size_t ldx, typename [Field::ConstElement_ptr](#) B, const size_t ldb, int *info)
Rectangular system solver.
- template<class [Field](#) >
void [ftrtri](#) (const [Field](#) &F, const [FFLAS::FFLAS_UPLO](#) Uplo, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, const size_t threshold=__FFLASFFPACK_FTRTRI_THRESHOLD)
Compute the inverse of a triangular matrix.
- template<class [Field](#) >
void [trinv_left](#) (const [Field](#) &F, const size_t N, typename [Field::ConstElement_ptr](#) L, const size_t ldl, typename [Field::Element_ptr](#) X, const size_t ldx)
- template<class [Field](#) >
void [ftrtrm](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) side, const [FFLAS::FFLAS_DIAG](#) diag, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda)
Compute the product of two triangular matrices of opposite shape.
- template<class [Field](#) >
void [ftrstr](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) side, const [FFLAS::FFLAS_UPLO](#) Uplo, const [FFLAS::FFLAS_DIAG](#) diagA, const [FFLAS::FFLAS_DIAG](#) diagB, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSTR_THRESHOLD)
Solve a triangular system with a triangular right hand side of the same shape.
- template<class [Field](#) >
void [ftrssyr2k](#) (const [Field](#) &F, const [FFLAS::FFLAS_UPLO](#) Uplo, const [FFLAS::FFLAS_DIAG](#) diagA, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) B, const size_t ldx, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)
Solve a triangular system in a symmetric sum: find B upper/lower triangular such that $A^T B + B^T A = C$ where C is symmetric.
- template<class [Field](#) >
bool [fsytrf](#) (const [Field](#) &F, const [FFLAS::FFLAS_UPLO](#) UpLo, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)
Triangular factorization of symmetric matrices.
- template<class [Field](#) >
bool [fsytrf](#) (const [Field](#) &F, const [FFLAS::FFLAS_UPLO](#) UpLo, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, const [FFLAS::ParSeqHelper::Sequential](#) seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)
- template<class [Field](#) , class Cut , class Param >
bool [fsytrf](#) (const [Field](#) &F, const [FFLAS::FFLAS_UPLO](#) UpLo, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, const [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)
- template<class [Field](#) >
bool [fsytrf_nonunit](#) (const [Field](#) &F, const [FFLAS::FFLAS_UPLO](#) UpLo, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)
Triangular factorization of symmetric matrices.
- template<class [Field](#) >
size_t [PLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Q)
Compute a PLUQ factorization of the given matrix.
- template<class [Field](#) >
size_t [pPLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Q)

- `template<class Field >`
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, type-`
`name Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential`
`&PSHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field , class Cut , class Param >`
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename`
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut,`
`Param > &PSHelper)`
- `template<class Field >`
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans,`
`const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt,`
`const FFPACK_LU_TAG LuTag=FfpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE↵`
`_THRESHOLD)`

Compute the CUP or PLE factorization of the given matrix.
- `template<class Field >`
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`
`A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG Lu↵`
`Tag=FfpackSlabRecursive)`

Compute the Column Echelon form of the input matrix in-place.
- `template<class Field >`
`size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`
`const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG`
`LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`
`const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper`
`&psH)`
- `template<class Field >`
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`
`A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG Lu↵`
`Tag=FfpackSlabRecursive)`

Compute the Row Echelon form of the input matrix in-place.
- `template<class Field >`
`size_t pRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`
`const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_↵`
`LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A,`
`const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper`
`&psH)`
- `template<class Field >`
`size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_↵`
`LU_TAG LuTag=FfpackSlabRecursive)`

Compute the Reduced Column Echelon form of the input matrix in-place.
- `template<class Field >`
`size_t pReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0,`
`const FFPACK_LU_TAG LuTag=FfpackTileRecursive)`
- `template<class Field , class PSHelper >`
`size_t ReducedColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename`
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU↵`
`_TAG LuTag, const PSHelper &psH)`
- `template<class Field >`
`size_t ReducedRowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`
`A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG Lu↵`
`Tag=FfpackSlabRecursive)`

Compute the Reduced Row Echelon form of the input matrix in-place.

- template<class [Field](#) >
size_t [pReducedRowEchelonForm](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >
size_t [ReducedRowEchelonForm](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)
- template<class [Field](#) >
[Field::Element_ptr](#) [Invert](#) (const [Field](#) &F, const size_t M, typename [Field::Element_ptr](#) A, const size_t lda, int &>nullity)
Invert the given matrix in place or computes its nullity if it is singular.
- template<class [Field](#) >
[Field::Element_ptr](#) [Invert](#) (const [Field](#) &F, const size_t M, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) X, const size_t ldx, int &>nullity)
Invert the given matrix or computes its nullity if it is singular.
- template<class [Field](#) >
[Field::Element_ptr](#) [Invert2](#) (const [Field](#) &F, const size_t M, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) X, const size_t ldx, int &>nullity)
Invert the given matrix or computes its nullity if it is singular.
- template<class [PolRing](#) >
std::list< typename [PolRing::Element](#) > & [CharPoly](#) (const [PolRing](#) &R, std::list< typename [PolRing::Element](#) > &charp, const size_t N, typename [PolRing::Domain_t::Element_ptr](#) A, const size_t lda, typename [PolRing::Domain_t::Randlter](#) &G, const FFPACK_CHARPOLY_TAG CharpTag=[FfpackAuto](#), const size_t degree=[__FFLASFFPACK_ARITHPROG_THRESHOLD](#))
Compute the characteristic polynomial of the matrix A.
- template<class [PolRing](#) >
[PolRing::Element](#) & [CharPoly](#) (const [PolRing](#) &R, typename [PolRing::Element](#) &charp, const size_t N, typename [PolRing::Domain_t::Element_ptr](#) A, const size_t lda, typename [PolRing::Domain_t::Randlter](#) &G, const FFPACK_CHARPOLY_TAG CharpTag=[FfpackAuto](#), const size_t degree=[__FFLASFFPACK_ARITHPROG_THRESHOLD](#))
Compute the characteristic polynomial of the matrix A.
- template<class [PolRing](#) >
[PolRing::Element](#) & [CharPoly](#) (const [PolRing](#) &R, typename [PolRing::Element](#) &charp, const size_t N, typename [PolRing::Domain_t::Element_ptr](#) A, const size_t lda, const FFPACK_CHARPOLY_TAG CharpTag=[FfpackAuto](#), const size_t degree=[__FFLASFFPACK_ARITHPROG_THRESHOLD](#))
Compute the characteristic polynomial of the matrix A.
- template<class [Field](#) , class [Polynomial](#) >
[Polynomial](#) & [MinPoly](#) (const [Field](#) &F, [Polynomial](#) &minP, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda)
Compute the minimal polynomial of the matrix A.
- template<class [Field](#) , class [Polynomial](#) , class [Randlter](#) >
[Polynomial](#) & [MinPoly](#) (const [Field](#) &F, [Polynomial](#) &minP, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda, [Randlter](#) &G)
Compute the minimal polynomial of the matrix A.
- template<class [Field](#) , class [Polynomial](#) >
[Polynomial](#) & [MatVecMinPoly](#) (const [Field](#) &F, [Polynomial](#) &minP, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) v, const size_t incv)
Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis $(v, Av, \dots, A^N v)$.
- template<class [Field](#) >
size_t [Rank](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda)
Computes the rank of the given matrix using a PLUQ factorization.
- template<class [Field](#) >
size_t [pRank](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t numthreads=0)

- template<class [Field](#) , class PSHelper >
 size_t [Rank](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, const PSHelper &psH)
Returns true if the given matrix is singular.
- template<class [Field](#) >
 bool [IsSingular](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda)
Returns true if the given matrix is singular.
- template<class [Field](#) >
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P=NULL, size_t *Q=NULL)
Returns the determinant of the given square matrix.
- template<class [Field](#) >
[Field::Element](#) & [pDet](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t numthreads=0, size_t *P=NULL, size_t *Q=NULL)
- template<class [Field](#) , class PSHelper >
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, const PSHelper &psH, size_t *P=NULL, size_t *Q=NULL)
- template<class [Field](#) >
[Field::Element_ptr](#) [Solve](#) (const [Field](#) &F, const size_t M, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) x, const int incx, typename [Field::ConstElement_ptr](#) b, const int incb)
Solves a linear system $AX = b$ using PLUQ factorization.
- template<class [Field](#) , class PSHelper >
[Field::Element_ptr](#) [Solve](#) (const [Field](#) &F, const size_t M, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) x, const int incx, typename [Field::ConstElement_ptr](#) b, const int incb, PSHelper &psH)
- template<class [Field](#) >
[Field::Element_ptr](#) [pSolve](#) (const [Field](#) &F, const size_t M, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) x, const int incx, typename [Field::ConstElement_ptr](#) b, const int incb, size_t numthreads=0)
- template<class [Field](#) >
 *void [RandomNullSpaceVector](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) X, const size_t incX)
Solve $LX = B$ or $XL = B$ in place.
- template<class [Field](#) >
 size_t [NullSpaceBasis](#) (const [Field](#) &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) &NS, size_t &ldn, size_t &NSdim)
Computes a basis of the Left/Right nullspace of the matrix A.
- template<class [Field](#) >
 size_t [RowRankProfile](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=[FfpackSlabRecursive](#))
Computes the row rank profile of A.
- template<class [Field](#) >
 size_t [pRowRankProfile](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >
 size_t [RowRankProfile](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)
- template<class [Field](#) >
 size_t [ColumnRankProfile](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=[FfpackSlabRecursive](#))
Computes the column rank profile of A.
- template<class [Field](#) >
 size_t [pColumnRankProfile](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=[FfpackTileRecursive](#))

- template<class [Field](#) , class PSHelper >
 size_t [ColumnRankProfile](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)
- void [RankProfileFromLU](#) (const size_t *P, const size_t N, const size_t R, size_t *rkprofile, const FFPACK_LU_TAG LuTag)
Recovers the column/row rank profile from the permutation of an LU decomposition.
- size_t [LeadingSubmatrixRankProfiles](#) (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)
Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.
- template<class [Field](#) >
 size_t [RowRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)
RowRankProfileSubmatrixIndices.
- template<class [Field](#) >
 size_t [ColRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *&rowindices, size_t *&colindices, size_t &R)
Computes the indices of the submatrix $r \times r$ X of A whose columns correspond to the column rank profile of A.
- template<class [Field](#) >
 size_t [RowRankProfileSubmatrix](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) &X, size_t &R)
Computes the $r \times r$ submatrix X of A, by picking the row rank profile rows of A.
- template<class [Field](#) >
 size_t [ColRankProfileSubmatrix](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) &X, size_t &R)
Compute the $r \times r$ submatrix X of A, by picking the row rank profile rows of A.
- template<class [Field](#) >
 void [getTriangular](#) (const [Field](#) &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) T, const size_t ldt, const bool OnlyNonZeroVectors=false)
Extracts a triangular matrix from a compact storage $A=L\backslash U$ of rank R.
- template<class [Field](#) >
 void [getTriangular](#) (const [Field](#) &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename [Field::Element_ptr](#) A, const size_t lda)
Cleans up a compact storage $A=L\backslash U$ to reveal a triangular matrix of rank R.
- template<class [Field](#) >
 void [getEchelonForm](#) (const [Field](#) &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=[FpackSlabRecursive](#))
Extracts a matrix in echelon form from a compact storage $A=L\backslash U$ of rank R obtained by RowEchelonForm or ColumnEchelonForm.
- template<class [Field](#) >
 void [getEchelonForm](#) (const [Field](#) &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename [Field::Element_ptr](#) A, const size_t lda, const FFPACK_LU_TAG LuTag=[FpackSlabRecursive](#))
Cleans up a compact storage $A=L\backslash U$ obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank R.
- template<class [Field](#) >
 void [getEchelonTransform](#) (const [Field](#) &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::Element_ptr](#) T, const size_t ldt, const FFPACK_LU_TAG LuTag=[FpackSlabRecursive](#))
Extracts a transformation matrix to echelon form from a compact storage $A=L\backslash U$ of rank R obtained by RowEchelonForm or ColumnEchelonForm.

- `template<class Field >`
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FFpackSlabRecursive)`
Extracts a matrix in echelon form from a compact storage $A=L\backslash U$ of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.
- `template<class Field >`
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FFpackSlabRecursive)`
Cleans up a compact storage $A=L\backslash U$ of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.
- `template<class Field >`
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FFpackSlabRecursive)`
Extracts a transformation matrix to echelon form from a compact storage $A=L\backslash U$ of rank R obtained by RowEchelonForm or ColumnEchelonForm.
- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`
Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.
- `template<class Field >`
`Field::Element_ptr LQUptInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X, const size_t ldx)`
LQUptInverseOfFullRankMinor.
- `template<class Field >`
`void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t incX)`
Solve $LX = B$ or $XL = B$ in place.
- `template<class Field >`
`void solveLB (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field >`
`void solveLB2 (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field, class Polynomial >`
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`
`Field::Element_ptr buildMatrix (const Field &F, typename Field::ConstElement_ptr E, typename Field::ConstElement_ptr C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda, const size_t mu)`
- `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)`
- `template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R, Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp, const size_t N, Givaro::Integer *A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)`
- `template<class PSHelper >`
`FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (const FFPACK::RNSInteger<`

- `FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr &det, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, const PSHelper &psH)`
- `template<class PSHelper >`
`Givaro::Integer & Det (const Givaro::ZRing< Givaro::Integer > &F, Givaro::Integer &det, const size_t N, Givaro::Integer *A, const size_t lda, const PSHelper &psH, size_t *P, size_t *Q)`
 - `template<class Field >`
`bool fsytrf_BC_Crout (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)`
 - `template<class Field >`
`size_t fsytrf_BC_RL (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)`
 - `template<class Field >`
`size_t fsytrf_UP_RPM_BC_RL (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
 - `template<class Field >`
`size_t fsytrf_LOW_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
 - `template<class Field >`
`size_t fsytrf_UP_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
 - `template<class Field >`
`size_t fsytrf_UP_RPM (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P, size_t BCThreshold)`
 - `template<class Field >`
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Sequential seq, size_t threshold)`
 - `template<class Field, class Cut, class Param >`
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Parallel< Cut, Param > par, size_t threshold)`
 - `template<class Field >`
`size_t fsytrf_RPM (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t threshold)`
 - `template<class Field >`
`void getTridiagonal (const Field &F, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, size_t *P, typename Field::Element_ptr T, const size_t ldt)`
 - `template<class Field >`
`size_t LUdivine_gauss (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`
 - `template<class Field >`
`size_t LUdivine_small (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`
 - `template<class Field >`
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)`
 - `template<> size_t LUdivine (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)`
 - `template<class Field >`
`void MonotonicCompress (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`

- `template<class Field >`
`void MonotonicCompressMorePivots (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, type-`
`name Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const`
`size_t rowstomove, const size_t lenP)`
- `template<class Field >`
`void MonotonicCompressCycles (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename`
`Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t lenP)`
- `template<class Field >`
`void MonotonicExpand (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename`
`Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t`
`maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`
- `template<class Field >`
`void applyP_block (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE`
`Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda,`
`const size_t *P)`
- `template<class Field >`
`void doApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr`
`tmp, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const`
`size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t`
`width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const`
`FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field , class Cut , class Param >`
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t`
`width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const`
`FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`
`void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t`
`R2, const size_t R3, const size_t R4)`
- `template<class Field >`
`void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr`
`tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const`
`size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t`
`width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const`
`FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field , class Cut , class Param >`
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t`
`width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const`
`FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`
`void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t`
`R2, const size_t R3, const size_t R4)`
- `void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)`
Computes $P1 \times Diag(I_R, P2)$ where $P1$ is a LAPACK and $P2$ a LAPACK permutation and store the result in $P1$ as a LAPACK permutation.
- `void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const`
`size_t N)`
Computes $P1 \times Diag(I_R, P2)$ where $P1$ is a LAPACK and $P2$ a LAPACK permutation and store the result in $MathP$ as a MathPermutation format.
- `void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)`

Computes MathP1 x Diag (I_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.

- void [cyclic_shift_mathPerm](#) (size_t *P, const size_t s)
- template<class [Field](#) >
void [cyclic_shift_row_col](#) (const [Field](#) &F, typename [Field::Element_ptr](#) A, size_t m, size_t n, size_t lda)
- template<class [Field](#) >
void [cyclic_shift_row](#) (const [Field](#) &F, typename [Field::Element_ptr](#) A, size_t m, size_t n, size_t lda)
- template<typename T >
void [cyclic_shift_row](#) (const [RNSIntegerMod](#)< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)
- template<class [Field](#) >
void [cyclic_shift_col](#) (const [Field](#) &F, typename [Field::Element_ptr](#) A, size_t m, size_t n, size_t lda)
- template<typename T >
void [cyclic_shift_col](#) (const [RNSIntegerMod](#)< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)
- template<class [Field](#) >
size_t [PLUQ_basecaseV3](#) (const [Field](#) &Fi, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, typename [Field::Element](#) *A, const size_t lda, size_t *P, size_t *Q)
- template<class [Field](#) >
size_t [PLUQ_basecaseV2](#) (const [Field](#) &Fi, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, typename [Field::Element](#) *A, const size_t lda, size_t *P, size_t *Q)
- template<class [Field](#) >
size_t [PLUQ_basecaseCrout](#) (const [Field](#) &Fi, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Q)
- template<class [Field](#) >
size_t [_PLUQ](#) (const [Field](#) &Fi, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold)
- template<class Cut, class Param >
size_t [PLUQ](#) (const Givaro::Modular< Givaro::Integer > &F, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold, [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > &PSHelper)
- template<class [Field](#) >
void [threads_fgemm](#) (const size_t m, const size_t n, const size_t r, int nbthreads, size_t *W1, size_t *W2, size_t *W3, size_t gamma)
- template<class [Field](#) >
void [threads_ftrsm](#) (const size_t m, const size_t n, int nbthreads, size_t *t1, size_t *t2)
- template<class [Field](#) >
size_t [PLUQ](#) (const [Field](#) &Fi, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Q, const [FFLAS::ParSeqHelper::Parallel](#)< [FFLAS::CuttingStrategy::Recursive](#), [FFLAS::StrategyParameter::Threads](#) > &PSHelper)
- template<> [rns_double_elt_ptr_fflas_const_cast](#) ([rns_double_elt_cstptr](#) x)
- template<> [rns_double_elt_cstptr_fflas_const_cast](#) ([rns_double_elt_ptr](#) x)
- template<typename Base_t >
void [cyclic_shift_row_col](#) (Base_t *A, size_t m, size_t n, size_t lda)
- template [INST_OR_DECL](#) void [cyclic_shift_row](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, [FFLAS_ELT](#) *A, size_t m, size_t n, size_t lda)
- template [INST_OR_DECL](#) void [cyclic_shift_col](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, [FFLAS_ELT](#) *A, size_t m, size_t n, size_t lda)
- template [INST_OR_DECL](#) void [applyP](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_SIDE](#) Side, const [FFLAS::FFLAS_TRANSPOSE](#) Trans, const size_t M, const size_t ibeg, const size_t iend, [FFLAS_ELT](#) *A, const size_t lda, const size_t *P)
- template [INST_OR_DECL](#) void [fgetrs](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, const size_t R, [FFLAS_ELT](#) *A, const size_t lda, const size_t *P, const size_t *Q, [FFLAS_ELT](#) *B, const size_t ldb, int *info)
- template [INST_OR_DECL](#) [FFLAS_ELT](#) * [fgetrs](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, [FFLAS_ELT](#) *A, const size_t lda, const size_t *P, const size_t *Q, [FFLAS_ELT](#) *X, const size_t ldx, const [FFLAS_ELT](#) *B, const size_t ldb, int *info)

- template [INST_OR_DECL](#) size_t [fgesv](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, [FFLAS_ELT](#) *B, const size_t ldb, int *info)
- template [INST_OR_DECL](#) size_t [fgesv](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_SIDE](#) Side, const size_t M, const size_t N, const size_t NRHS, [FFLAS_ELT](#) *A, const size_t lda, [FFLAS_ELT](#) *X, const size_t ldx, const [FFLAS_ELT](#) *B, const size_t ldb, int *info)
- template [INST_OR_DECL](#) void [ftrtri](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_UPLO](#) Uplo, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, const size_t threshold)
- template [INST_OR_DECL](#) void [trinv_left](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const size_t N, const [FFLAS_ELT](#) *L, const size_t ldl, [FFLAS_ELT](#) *X, const size_t ldx)
- template [INST_OR_DECL](#) void [ftrtrm](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_SIDE](#) side, const [FFLAS::FFLAS_DIAG](#) diag, const size_t N, [FFLAS_ELT](#) *A, const size_t lda)
- template [INST_OR_DECL](#) size_t [PLUQ](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, size_t *P, size_t *Q)
- template [INST_OR_DECL](#) size_t [LUdivine](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_DIAG](#) Diag, const [FFLAS::FFLAS_TRANSPOSE](#) trans, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, size_t *P, size_t *Qt, const [FFPACK_LU_TAG](#) LuTag, const size_t cutoff)
- template [INST_OR_DECL](#) size_t [LUdivine_small](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_DIAG](#) Diag, const [FFLAS::FFLAS_TRANSPOSE](#) trans, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, size_t *P, size_t *Q, const [FFPACK_LU_TAG](#) LuTag)
- template [INST_OR_DECL](#) size_t [LUdivine_gauss](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const [FFLAS::FFLAS_DIAG](#) Diag, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, size_t *P, size_t *Q, const [FFPACK_LU_TAG](#) LuTag)
- template [INST_OR_DECL](#) size_t [RowEchelonForm](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const [FFPACK_LU_TAG](#) LuTag)
- template [INST_OR_DECL](#) size_t [ReducedRowEchelonForm](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const [FFPACK_LU_TAG](#) LuTag)
- template [INST_OR_DECL](#) size_t [ColumnEchelonForm](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const [FFPACK_LU_TAG](#) LuTag)
- template [INST_OR_DECL](#) size_t [ReducedColumnEchelonForm](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const size_t M, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const [FFPACK_LU_TAG](#) LuTag)
- template [INST_OR_DECL](#) [FFLAS_ELT](#) * [Invert](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const size_t M, [FFLAS_ELT](#) *A, const size_t lda, int &nullity)
- template [INST_OR_DECL](#) [FFLAS_ELT](#) * [Invert](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const size_t M, const [FFLAS_ELT](#) *A, const size_t lda, [FFLAS_ELT](#) *X, const size_t ldx, int &nullity)
- template [INST_OR_DECL](#) [FFLAS_ELT](#) * [Invert2](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, const size_t M, [FFLAS_ELT](#) *A, const size_t lda, [FFLAS_ELT](#) *X, const size_t ldx, int &nullity)
- template [INST_OR_DECL](#) std::list< [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > >::Element > & [CharPoly](#) (const [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > > &R, std::list< [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > >::Element > &charp, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, [FFLAS_FIELD](#)< [FFLAS_ELT](#) >::RandIter &G, const [FFPACK_CHARPOLY_TAG](#) CharpTag, const size_t degree)
- template [INST_OR_DECL](#) [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > >::Element & [CharPoly](#) (const [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > > &R, [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > >::Element &charp, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, [FFLAS_FIELD](#)< [FFLAS_ELT](#) >::RandIter &G, const [FFPACK_CHARPOLY_TAG](#) CharpTag, const size_t degree)
- template [INST_OR_DECL](#) [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > >::Element & [CharPoly](#) (const [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > > &R, [Givaro::Poly1Dom](#)< [FFLAS_FIELD](#)< [FFLAS_ELT](#) > >::Element &charp, const size_t N, [FFLAS_ELT](#) *A, const size_t lda, const [FFPACK_CHARPOLY_TAG](#) CharpTag, const size_t degree)
- template [INST_OR_DECL](#) std::vector< [FFLAS_ELT](#) > & [MinPoly](#) (const [FFLAS_FIELD](#)< [FFLAS_ELT](#) > &F, std::vector< [FFLAS_ELT](#) > &minP, const size_t N, const [FFLAS_ELT](#) *A, const size_t lda, [FFLAS_FIELD](#)< [FFLAS_ELT](#) >::RandIter &G)

- template [INST_OR_DECL](#) [std::vector< FFLAS_ELT > & MinPoly](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), [std::vector< FFLAS_ELT > &minP](#), const [size_t N](#), const [FFLAS_ELT *A](#), const [size_t lda](#))
- template [INST_OR_DECL](#) [std::vector< FFLAS_ELT > & MatVecMinPoly](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), [std::vector< FFLAS_ELT > &minP](#), const [size_t N](#), const [FFLAS_ELT *A](#), const [size_t lda](#), const [FFLAS_ELT *V](#), const [size_t incv](#))
- template [INST_OR_DECL](#) [size_t KrylovElim](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [size_t *P](#), [size_t *Q](#), const [size_t deg](#), [size_t *iterates](#), [size_t *inviterates](#), const [size_t maxit](#), [size_t virt](#))
- template [INST_OR_DECL](#) [size_t SpecRankProfile](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), const [size_t deg](#), [size_t *rankProfile](#))
- template [INST_OR_DECL](#) [size_t Rank](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#))
- template [INST_OR_DECL](#) [bool IsSingular](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#))
- template [INST_OR_DECL](#) [FFLAS_ELT & Det](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), [FFLAS_ELT &det](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [size_t *P](#), [size_t *Q](#))
- template [INST_OR_DECL](#) [FFLAS_ELT & Det](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), [FFLAS_ELT &det](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), const [FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &parH](#), [size_t *P](#), [size_t *Q](#))
- template [INST_OR_DECL](#) [FFLAS_ELT * Solve](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), [FFLAS_ELT *A](#), const [size_t lda](#), [FFLAS_ELT *x](#), const [int incx](#), const [FFLAS_ELT *b](#), const [int incb](#))
- template [INST_OR_DECL](#) [void solveLB](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [FFLAS::FFLAS_SIDE Side](#), const [size_t M](#), const [size_t N](#), const [size_t R](#), [FFLAS_ELT *L](#), const [size_t ldl](#), const [size_t *Q](#), [FFLAS_ELT *B](#), const [size_t ldb](#))
- template [INST_OR_DECL](#) [void solveLB2](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [FFLAS::FFLAS_SIDE Side](#), const [size_t M](#), const [size_t N](#), const [size_t R](#), [FFLAS_ELT *L](#), const [size_t ldl](#), const [size_t *Q](#), [FFLAS_ELT *B](#), const [size_t ldb](#))
- template [INST_OR_DECL](#) [void RandomNullSpaceVector](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [FFLAS::FFLAS_SIDE Side](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [FFLAS_ELT *X](#), const [size_t incX](#))
- template [INST_OR_DECL](#) [size_t NullSpaceBasis](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [FFLAS::FFLAS_SIDE Side](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [FFLAS_ELT *&NS](#), [size_t &ldn](#), [size_t &NSdim](#))
- template [INST_OR_DECL](#) [size_t RowRankProfile](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [size_t *rkprofile](#), const [FFPACK_LU_TAG LuTag](#))
- template [INST_OR_DECL](#) [size_t ColumnRankProfile](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [size_t *rkprofile](#), const [FFPACK_LU_TAG LuTag](#))
- template [INST_OR_DECL](#) [size_t RowRankProfileSubmatrixIndices](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [size_t *&rowindices](#), [size_t *&colindices](#), [size_t &R](#))
- template [INST_OR_DECL](#) [size_t ColRankProfileSubmatrixIndices](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [size_t *&rowindices](#), [size_t *&colindices](#), [size_t &R](#))
- template [INST_OR_DECL](#) [size_t RowRankProfileSubmatrix](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [FFLAS_ELT *&X](#), [size_t &R](#))
- template [INST_OR_DECL](#) [size_t ColRankProfileSubmatrix](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [size_t M](#), const [size_t N](#), [FFLAS_ELT *A](#), const [size_t lda](#), [FFLAS_ELT *&X](#), [size_t &R](#))
- template [INST_OR_DECL](#) [void getTriangular< FFLAS_FIELD< FFLAS_ELT > >](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [FFLAS::FFLAS_UPLO Uplo](#), const [FFLAS::FFLAS_DIAG diag](#), const [size_t M](#), const [size_t N](#), const [size_t R](#), const [FFLAS_ELT *A](#), const [size_t lda](#), [FFLAS_ELT *T](#), const [size_t ldt](#), const [bool OnlyNonZeroVectors](#))
- template [INST_OR_DECL](#) [void getTriangular< FFLAS_FIELD< FFLAS_ELT > >](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [FFLAS::FFLAS_UPLO Uplo](#), const [FFLAS::FFLAS_DIAG diag](#), const [size_t M](#), const [size_t N](#), const [size_t R](#), [FFLAS_ELT *A](#), const [size_t lda](#))
- template [INST_OR_DECL](#) [void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >](#) (const [FFLAS_FIELD< FFLAS_ELT > &F](#), const [FFLAS::FFLAS_UPLO Uplo](#), const [FFLAS::FFLAS_DIAG diag](#), const [size_t M](#), const [size_t N](#), const [size_t R](#), const [size_t *P](#), const [FFLAS_ELT *A](#), const [size_t lda](#), [FFLAS_ELT *T](#), const [size_t ldt](#), const [bool OnlyNonZeroVectors](#), const [FFPACK_LU_TAG LuTag](#))

- template `INST_OR_DECL` void `getEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size_t M, const size_t N, const size_t R, const size_t *P, `FFLAS_ELT` *A, const size_t Ida, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonTransform`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const `FFLAS_ELT` *A, const size_t Ida, `FFLAS_ELT` *T, const size_t Idt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const `FFLAS_ELT` *A, const size_t Ida, `FFLAS_ELT` *T, const size_t Idt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, `FFLAS_ELT` *A, const size_t Ida, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonTransform`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, const `FFLAS_ELT` *A, const size_t Ida, `FFLAS_ELT` *T, const size_t Idt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `FFLAS_ELT` * `LQUPtoInverseOfFullRankMinor` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size_t rank, `FFLAS_ELT` *A_factors, const size_t Ida, const size_t *QtPointer, `FFLAS_ELT` *X, const size_t Idx)
- template<class T, class CT = const T>
T `fflas_const_cast` (CT x)
- `Failure` & `failure` ()
- template<class T >
bool `isOdd` (const T &a)
- bool `isOdd` (const float &a)
- bool `isOdd` (const double &a)
- template<class `Field`, class `RandIter` >
`Field::Element_ptr` `NonZeroRandomMatrix` (const `Field` &F, size_t m, size_t n, typename `Field::Element_ptr` A, size_t Ida, `RandIter` &G)
Random non-zero Matrix.
- template<class `Field`, class `RandIter` >
`Field::Element_ptr` `NonZeroRandomMatrix` (const `Field` &F, size_t m, size_t n, typename `Field::Element_ptr` A, size_t Ida)
Random non-zero Matrix.
- template<class `Field`, class `RandIter` >
`Field::Element_ptr` `RandomMatrix` (const `Field` &F, size_t m, size_t n, typename `Field::Element_ptr` A, size_t Ida, `RandIter` &G)
Random Matrix.
- template<class `Field` >
`Field::Element_ptr` `RandomMatrix` (const `Field` &F, size_t m, size_t n, typename `Field::Element_ptr` A, size_t Ida)
Random Matrix.
- template<class `Field`, class `RandIter` >
`Field::Element_ptr` `RandomTriangularMatrix` (const `Field` &F, size_t m, size_t n, const `FFLAS::FFLAS_UPLO` UpLo, const `FFLAS::FFLAS_DIAG` Diag, bool nonsingular, typename `Field::Element_ptr` A, size_t Ida, `RandIter` &G)
Random Triangular Matrix.
- template<class `Field` >
`Field::Element_ptr` `RandomTriangularMatrix` (const `Field` &F, size_t m, size_t n, const `FFLAS::FFLAS_UPLO` UpLo, const `FFLAS::FFLAS_DIAG` Diag, bool nonsingular, typename `Field::Element_ptr` A, size_t Ida)
Random Triangular Matrix.
- size_t `RandInt` (size_t a, size_t b)

- `template<class Field , class RandIter >`
`Field::Element_ptr RandomSymmetricMatrix` (const `Field` &F, `size_t` n, bool nonsingular, typename `Field::Element_ptr` A, `size_t` lda, `RandIter` &G)
Random Symmetric Matrix.
- `template<class Field , class RandIter >`
`Field::Element_ptr RandomMatrixWithRank` (const `Field` &F, `size_t` m, `size_t` n, `size_t` r, typename `Field::Element_ptr` A, `size_t` lda, `RandIter` &G)
Random Matrix with prescribed rank.
- `template<class Field >`
`Field::Element_ptr RandomMatrixWithRank` (const `Field` &F, `size_t` m, `size_t` n, `size_t` r, typename `Field::Element_ptr` A, `size_t` lda)
Random Matrix with prescribed rank.
- `size_t * RandomIndexSubset` (`size_t` N, `size_t` R, `size_t` *P)
Pick uniformly at random a sequence of R distinct elements from the set $\{0, \dots, N - 1\}$ using Knuth's shuffle.
- `size_t * RandomPermutation` (`size_t` N, `size_t` *P)
Pick uniformly at random a permutation of size N stored in LAPACK format using Knuth's shuffle.
- `void RandomRankProfileMatrix` (`size_t` M, `size_t` N, `size_t` R, `size_t` *rows, `size_t` *cols)
Pick uniformly at random an R-subpermutation of dimension $M \times N$: a matrix with only R non-zeros equal to one, in a random rook placement.
- `void swapval` (`size_t` k, `size_t` N, `size_t` *P, `size_t` val)
- `void RandomSymmetricRankProfileMatrix` (`size_t` N, `size_t` R, `size_t` *rows, `size_t` *cols)
Pick uniformly at random a symmetric R-subpermutation of dimension $N \times N$: a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.
- `template<class Field , class RandIter >`
`Field::Element_ptr RandomMatrixWithRankandRPM` (const `Field` &F, `size_t` M, `size_t` N, `size_t` R, typename `Field::Element_ptr` A, `size_t` lda, const `size_t` *RRP, const `size_t` *CRP, `RandIter` &G)
Random Matrix with prescribed rank and rank profile matrix Creates an $m \times n$ matrix with random entries and rank r.
- `template<class Field >`
`Field::Element_ptr RandomMatrixWithRankandRPM` (const `Field` &F, `size_t` M, `size_t` N, `size_t` R, typename `Field::Element_ptr` A, `size_t` lda, const `size_t` *RRP, const `size_t` *CRP)
Random Matrix with prescribed rank and rank profile matrix Creates an $m \times n$ matrix with random entries and rank r.
- `template<class Field , class RandIter >`
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, `size_t` N, `size_t` R, typename `Field::Element_ptr` A, `size_t` lda, const `size_t` *RRP, const `size_t` *CRP, `RandIter` &G)
Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an $n \times n$ symmetric matrix with random entries and rank r.
- `template<class Field >`
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, `size_t` M, `size_t` N, `size_t` R, typename `Field::Element_ptr` A, `size_t` lda, const `size_t` *RRP, const `size_t` *CRP)
Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an $n \times n$ symmetric matrix with random entries and rank r.
- `template<class Field , class RandIter >`
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, `size_t` M, `size_t` N, `size_t` R, typename `Field::Element_ptr` A, `size_t` lda, `RandIter` &G)
Random Matrix with prescribed rank, with random rank profile matrix Creates an $m \times n$ matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- `template<class Field >`
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, `size_t` M, `size_t` N, `size_t` R, typename `Field::Element_ptr` A, `size_t` lda)
Random Matrix with prescribed rank, with random rank profile matrix Creates an $m \times n$ matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- `template<class Field , class RandIter >`
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, `size_t` N, `size_t` R, typename `Field::Element_ptr` A, `size_t` lda, `RandIter` &G)

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an $n \times n$ matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

- `template<class Field >`
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (const Field &F, size_t N, size_t R, typename Field::Element_ptr A, size_t lda)`
Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an $n \times n$ matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.
- `template<class Field >`
`Field::Element_ptr RandomMatrixWithDet (const Field &F, size_t n, const typename Field::Element d, typename Field::Element_ptr A, size_t lda)`
Random Matrix with prescribed det.
- `template<class Field , class RandIter >`
`Field::Element_ptr RandomMatrixWithDet (const Field &F, size_t n, const typename Field::Element d, typename Field::Element_ptr A, size_t lda, RandIter &G)`
Random Matrix with prescribed det.
- `template<typename Field >`
`Givaro::Integer maxFieldElt ()`
- `template<> Givaro::Integer maxFieldElt< Givaro::ZRing< Givaro::Integer > > ()`
- `template<typename Field >`
`Field * chooseField (Givaro::Integer q, uint64_t b, uint64_t seed)`
- `template<> Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (Givaro::Integer q, uint64_t b, uint64_t seed)`
- `template<> Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (Givaro::Integer q, uint64_t b, uint64_t seed)`
- `template<> Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (Givaro::Integer q, uint64_t b, uint64_t seed)`
- `template<> Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (Givaro::Integer q, uint64_t b, uint64_t seed)`

11.20.1 Detailed Description

Finite Field PACK Set of elimination based routines for dense linear algebra.

This namespace enlarges the set of BLAS routines of the class **FFLAS**, with higher level routines based on elimination.

11.20.2 Typedef Documentation

11.20.2.1 Checker_PLUQ

```
template<class Field >
using Checker_PLUQ = FFLAS::Checker_Empty<Field>
```

11.20.2.2 Checker_Det

```
template<class Field >
using Checker_Det = FFLAS::Checker_Empty<Field>
```

11.20.2.3 Checker_invert

```
template<class Field >
using Checker_invert = FFLAS::Checker_Empty<Field>
```

11.20.2.4 Checker_charpoly

```
template<class Field , class Polynomial >
using Checker_charpoly = FFLAS::Checker_Empty<Field>
```

11.20.2.5 ForceCheck_PLUQ

```
template<class Field >
using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
```

11.20.2.6 ForceCheck_Det

```
template<class Field >
using ForceCheck_Det = CheckerImplem_Det<Field>
```

11.20.2.7 ForceCheck_invert

```
template<class Field >
using ForceCheck_invert = CheckerImplem_invert<Field>
```

11.20.2.8 ForceCheck_charpoly

```
template<class Field , class Polynomial >
using ForceCheck_charpoly = CheckerImplem_charpoly<Field,Polynomial>
```

11.20.3 Function Documentation

11.20.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N ) [inline]
```

Conversion of a permutation from LAPACK format to Math format.

11.20.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N ) [inline]
```

Conversion of a permutation from Maths format to LAPACK format.

11.20.3.3 applyP() [1/4]

```
template<class Field >
void applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P ) [inline]
```

Computes $P1 \times \text{Diag}(I_R, P2)$ where $P1$ is a LAPACK and $P2$ a LAPACK permutation and store the result in $P1$ as a LAPACK permutation.

Parameters

in, out	$P1$	a LAPACK permutation of size N
	$P2$	a LAPACK permutation of size N-R

Applies a permutation P to the matrix A . Apply a permutation P , stored in the LAPACK format (a sequence of transpositions) between indices $ibeg$ and $iend$ of P to $(iend-ibeg)$ vectors of size M stored in A (as column for NoTrans and rows for Trans). $Side == \text{FFLAS::FflasLeft}$ for row permutation $Side == \text{FFLAS::FflasRight}$ for a column permutation $Trans == \text{FFLAS::FflasTrans}$ for the inverse permutation of P

Parameters

F	base field
$Side$	decides if rows (FflasLeft) or columns (FflasRight) are permuted
$Trans$	decides if the matrix is seen as columns (FflasTrans) or rows (FflasNoTrans)
M	size of the elements to permute
$ibeg$	first index to consider in P
$iend$	last index to consider in P
A	input matrix
lda	leading dimension of A
P	permutation in LAPACK format
psh	(optional): a sequential or parallel helper, to choose between sequential or parallel execution

Warning

not sure the submatrix is still a permutation and the one we expect in all cases... examples for iend=2, ibeg=1 and P=[2,2,2]

11.20.3.4 applyP() [2/4]

```
template<class Field >
void applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t m,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const FFLAS::ParSeqHelper::Sequential seq ) [inline]
```

11.20.3.5 applyP() [3/4]

```
template<class Field , class Cut , class Param >
void applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t m,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

11.20.3.6 MonotonicApplyP()

```
template<class Field >
void MonotonicApplyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t R ) [inline]
```

Apply a R-monotonically increasing permutation P, to the matrix A.

MonotonicApplyP Apply a permutation defined by the first R entries of the vector P (the pivots).

The permutation represented by P is defined as follows:

- the first R values of P is a LAPACK representation (a sequence of transpositions)
- the remaining $iend-ibeg-R$ values of the permutation are in a monotonically increasing progression $Side==FflasLeft$ for row permutation $Side==FflasRight$ for a column permutation $Trans==FflasTrans$ for the inverse permutation of P

Parameters

<i>F</i>	base field
<i>Side</i>	selects if it is a row (FflasLeft) or column (FflasRight) permutation
<i>Trans</i>	inverse permutation (FflasTrans/NoTrans)
<i>M</i>	
<i>ibeg</i>	
<i>iend</i>	
<i>A</i>	input matrix
<i>lda</i>	leading dimension of <i>A</i>
<i>P</i>	LAPACK permutation
<i>R</i>	first values of P

The non pivot elements, are located in monotonically increasing order.

11.20.3.7 fgetrs() [1/4]

```
template<class Field >
void fgetrs (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb,
    int * info )
```

Solve the system $AX = B$ or $XA = B$.

Solving using the PLUQ decomposition of A already computed inplace with PLUQ (FFLAS::FflasNonUnit). Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

Parameters

<i>F</i>	base field
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
<i>M</i>	row dimension of B
<i>N</i>	col dimension of B
<i>R</i>	rank of A
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	row permutation of the PLUQ decomposition of A

Parameters

<i>Q</i>	column permutation of the PLUQ decomposition of A
<i>B</i>	Right/Left hand side matrix. Initially stores B, finally stores the solution X.
<i>ldb</i>	leading dimension of B
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

11.20.3.8 fgetrs() [2/4]

```

template<class Field >
Field::Element_ptr fgetrs (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    int * info )

```

Solve the system $A X = B$ or $X A = B$.

Solving using the PLUQ decomposition of A already computed inplace with PLUQ(FFLAS::FflasNonUnit). Version for A rectangular. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

Parameters

<i>F</i>	base field
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
<i>M</i>	row dimension of A
<i>N</i>	col dimension of A
<i>NRHS</i>	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
<i>R</i>	rank of A
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	row permutation of the PLUQ decomposition of A
<i>Q</i>	column permutation of the PLUQ decomposition of A
<i>X</i>	solution matrix
<i>ldx</i>	leading dimension of X
<i>B</i>	Right/Left hand side matrix.
<i>ldb</i>	leading dimension of B
<i>info</i>	Succes of the computation: 0 if successfull, >0 if system is inconsistent

11.20.3.9 fgesv() [1/4]

```
template<class Field >
size_t fgesv (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    int * info )
```

Square system solver.

Parameters

<i>F</i>	The computation domain
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
<i>M</i>	row dimension of B
<i>N</i>	col dimension of B
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>B</i>	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
<i>ldb</i>	leading dimension of B
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

Returns

the rank of the system

Solve the system $A X = B$ or $X A = B$. Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

11.20.3.10 fgesv() [2/4]

```
template<class Field >
size_t fgesv (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    int * info )
```

Rectangular system solver.

Parameters

<i>F</i>	The computation domain
<i>Side</i>	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
<i>M</i>	row dimension of A
<i>N</i>	col dimension of A
<i>NRHS</i>	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>B</i>	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
<i>ldb</i>	leading dimension of B
<i>X</i>	
<i>ldx</i>	
<i>info</i>	Success of the computation: 0 if successfull, >0 if system is inconsistent

Returns

the rank of the system

Solve the system $A X = B$ or $X A = B$. Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

11.20.3.11 ftrtri() [1/2]

```
template<class Field >
void ftrtri (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t threshold = __FFLASFFPACK_FTRTRI_THRESHOLD )
```

Compute the inverse of a triangular matrix.

Parameters

<i>F</i>	base field
<i>Uplo</i>	whether the matrix is upper or lower triangular
<i>Diag</i>	whether the matrix is unit diagonal (FflasUnit/NoUnit)
<i>N</i>	input matrix order
<i>A</i>	the input matrix
<i>lda</i>	leading dimension of A

11.20.3.12 trinv_left() [1/2]

```
template<class Field >
```

```

void trinv_left (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr L,
    const size_t ldl,
    typename Field::Element_ptr X,
    const size_t ldx )

```

11.20.3.13 ftrtrm() [1/2]

```

template<class Field >
void ftrtrm (
    const Field & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )

```

Compute the product of two triangular matrices of opposite shape.

Product UL or LU of the upper, resp lower triangular matrices U and L stored one above the other in the square matrix A.

Parameters

<i>F</i>	base field
<i>Side</i>	set to FflasLeft to compute the product UL, FflasRight to compute LU
<i>diag</i>	whether the matrix U is unit diagonal (FflasUnit/NoUnit)
<i>N</i>	input matrix order
<i>A</i>	the input matrix
<i>lda</i>	leading dimension of A

11.20.3.14 ftrstr()

```

template<class Field >
void ftrstr (
    const Field & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diagA,
    const FFLAS::FFLAS_DIAG diagB,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const size_t threshold = __FFLASFFPACK_FTRSTR_THRESHOLD ) [inline]

```

Solve a triangular system with a triangular right hand side of the same shape.

Parameters

<i>F</i>	base field
<i>Side</i>	set to FflsLeft to compute $U1^{-1} * U2$ or $L1^{-1} * L2$, FflsRight to compute $U1 * U2^{-1}$ or $L1 * L2^{-1}$
<i>Uplo</i>	whether the matrix A is upper or lower triangular
<i>diag1</i>	whether the matrix U1 or L2 is unit diagonal (FflsUnit/NoUnit)
<i>diag2</i>	whether the matrix U2 or L2 is unit diagonal (FflsUnit/NoUnit)
<i>N</i>	order of the input matrices
<i>A</i>	the input matrix to be inverted (U1 or L1)
<i>lda</i>	leading dimension of A
<i>B</i>	the input right hand side (U2 or L2)
<i>ldb</i>	leading dimension of B

11.20.3.15 ftrssyr2k()

```
template<class Field >
void ftrssyr2k (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diagA,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const size_t threshold = __FFLASFFPACK_FTRSSYR2K_THRESHOLD ) [inline]
```

Solve a triangular system in a symmetric sum: find B upper/lower triangular such that $A^T B + B^T A = C$ where C is symmetric.

C is overwritten by B.

Parameters

	<i>F</i>	base field
	<i>Side</i>	set to FflsLeft to compute $U1^{-1} * U2$ or $L1^{-1} * L2$, FflsRight to compute $U1 * U2^{-1}$ or $L1 * L2^{-1}$
	<i>Uplo</i>	whether the matrix A is upper or lower triangular
	<i>diagA</i>	whether the matrix A is unit diagonal (FflsUnit/NoUnit)
	<i>N</i>	order of the input matrices
	<i>A</i>	the input matrix
	<i>lda</i>	leading dimension of A
<i>in, out</i>	<i>B</i>	the input right hand side where the output is written
	<i>ldb</i>	leading dimension of B

11.20.3.16 fsytrf() [1/3]

```
template<class Field >
bool fsytrf (
```



```

const Field & F,
const FFLAS::FFLAS_UPLO UpLo,
const size_t N,
typename Field::Element_ptr A,
const size_t lda,
const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )

```

Triangular factorization of symmetric matrices.

Parameters

	F	The computation domain
	$UpLo$	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor
	N	order of the matrix A
in, out	A	input matrix
	lda	leading dimension of A

Returns

false if the A does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix A: $A = L \times D \times L^T$ if UpLo = FflasLower or $A = U^T \times D \times U$ otherwise. D is a diagonal matrix. The matrices L and U are unit diagonal lower (resp. upper) triangular and overwrite the input matrix A. The matrix D is stored on the diagonal of A, as the diagonal of L or U is known to be all ones. If A does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

11.20.3.17 fsytrf() [2/3]

```

template<class Field >
bool fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Sequential seq,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD ) [inline]

```

11.20.3.18 fsytrf() [3/3]

```

template<class Field , class Cut , class Param >
bool fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD ) [inline]

```

11.20.3.19 fsytrf_nonunit() [1/3]

```
template<class Field >
bool fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr D,
    const size_t incD,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

Triangular factorization of symmetric matrices.

Parameters

	F	The computation domain
	$UpLo$	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor
	N	order of the matrix A
in, out	A	input matrix
in, out	D	
	lda	leading dimension of A

Returns

false if the A does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix A: $A = L \times D_{inv} \times L^T$ if UpLo = FflasLower or $A = U^T \times D \times U$ otherwise. D is a diagonal matrix. The matrices L and U are lower (resp. upper) triangular and overwrite the input matrix A. The matrix D need to be stored separately, as the diagonal of L or U are not unit. If A does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

11.20.3.20 PLUQ() [1/6]

```
template<class Field >
size_t PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

Compute a PLUQ factorization of the given matrix.

Return its rank. The permutations P and Q are represented using LAPACK's convention.

Parameters

F	base field
-----	------------

Parameters

<i>Diag</i>	whether U should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
<i>M</i>	matrix row dimension
<i>N</i>	matrix column dimension
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	the row permutation
<i>Q</i>	the column permutation

Returns

the rank of A

Bibliography

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13, 2013

11.20.3.21 pPLUQ()

```
template<class Field >
size_t pPLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

11.20.3.22 PLUQ() [2/6]

```
template<class Field >
size_t PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Sequential & PSHelper,
    size_t BCThreshold = __FFLASFFPACK_PLUQ_THRESHOLD ) [inline]
```

11.20.3.23 PLUQ() [3/6]

```
template<class Field , class Cut , class Param >
size_t PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper )
```

11.20.3.24 LUdivine() [1/4]

```
template<class Field >
size_t LUdivine (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive,
    const size_t cutoff = __FFLASFFPACK_LUDIVINE_THRESHOLD )
```

Compute the CUP or PLE factorization of the given matrix.

Using a block algorithm and return its rank. The permutations P and Q are represented using LAPACK's convention.

Parameters

<i>F</i>	base field
<i>Diag</i>	whether the transformation matrix (U of the CUP, L of the PLE) should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
<i>trans</i>	whether to compute the CUP decomposition (FflasNoTrans) or the PLE decomposition (FflasTrans)
<i>M</i>	matrix row dimension
<i>N</i>	matrix column dimension
<i>A</i>	input matrix
<i>lda</i>	leading dimension of A
<i>P</i>	the factor of CUP or PLE
<i>Q</i>	a permutation indicating the pivot position in the echelon form C or E in its first r positions
<i>LuTag</i>	flag for setting the earling termination if the matrix is singular
<i>cutoff</i>	threshold to basecase

Returns

the rank of A

Bibliography

- Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013
- Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002

11.20.3.25 ColumnEchelonForm() [1/3]

```
template<class Field >
size_t ColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Compute the Column Echelon form of the input matrix in-place.

If $LuTag == FfpackTileRecursive$, then after the computation $A = [M \setminus V]$ such that $AU = C$ is a column echelon decomposition of A , with $U = P^T [V]$ and $C = M + Q [I_r] [0 \text{ } I_{n-r}] [0]$ If $LuTag == FfpackTileRecursive$ then $A = [N \setminus V]$ such that the same holds with $M = Q N$

$Qt = Q^T$ If $transform=false$, the matrix V is not computed. See also test-colechelon for an example of use

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	A	input matrix
	lda	leading dimension of A
	P	the column permutation
	Qt	the row position of the pivots in the echelon form
	$transform$	decides whether V is computed
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.20.3.26 pColumnEchelonForm()

```
template<class Field >
size_t pColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]
```

11.20.3.27 ColumnEchelonForm() [2/3]

```
template<class Field , class PSHelper >
size_t ColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]
```

11.20.3.28 RowEchelonForm() [1/3]

```
template<class Field >
size_t RowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Compute the Row Echelon form of the input matrix in-place.

If $\text{LuTag} == \text{FfpackTileRecursive}$, then after the computation $A = [L \setminus M]$ such that $X A = R$ is a row echelon decomposition of A , with $X = [L \ 0] P$ and $R = M + [I_r \ 0] Q^T [In-r]$ If $\text{LuTag} == \text{FfpackTileRecursive}$ then $A = [L \setminus N]$ such that the same holds with $M = N Q^T Qt = Q^T$ If $\text{transform} = \text{false}$, the matrix L is not computed. See also `test-rowechelon` for an example of use

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	A	the input matrix
	lda	leading dimension of A
	P	the row permutation
	Qt	the column position of the pivots in the echelon form
	$transform$	decides whether L is computed
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.20.3.29 pRowEchelonForm()

```
template<class Field >
size_t pRowEchelonForm (
```

```

const Field & F,
const size_t M,
const size_t N,
typename Field::Element_ptr A,
const size_t lda,
size_t * P,
size_t * Qt,
const bool transform = false,
size_t numthreads = 0,
const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

11.20.3.30 RowEchelonForm() [2/3]

```

template<class Field , class PSHelper >
size_t RowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]

```

11.20.3.31 ReducedColumnEchelonForm() [1/3]

```

template<class Field >
size_t ReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Reduced Column Echelon form of the input matrix in-place.

After the computation $A = [V]$ such that $AX = R$ is a reduced col echelon $[M \ 0]$ decomposition of A , where $X = P^T [V]$ and $R = Q [I_r] [0 \ I_{n-r}] [M \ 0] Q^T = Q^T$ If transform=false, the matrix X is not computed and the matrix $A = R$

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	A	input matrix
	lda	leading dimension of A
	P	the column permutation
	Qt	the row position of the pivots in the echelon form
	$transform$	decides whether X is computed
Generated on Wed Jul 19 2023 00:00:00 for FFPACK by Doxygen		
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.20.3.32 pReducedColumnEchelonForm()

```
template<class Field >
size_t pReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]
```

11.20.3.33 ReducedColumnEchelonForm() [2/3]

```
template<class Field , class PSHelper >
size_t ReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]
```

11.20.3.34 ReducedRowEchelonForm() [1/3]

```
template<class Field >
size_t ReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Compute the Reduced Row Echelon form of the input matrix in-place.

After the computation $A = [V1 \ M]$ such that $X \ A = R$ is a reduced row echelon $[V2 \ 0]$ decomposition of A , where $X = [V1 \ 0] \ P$ and $R = [I_r \ M] \ Q^T [V2 \ In-r] [0] \ Qt = Q^T$ If transform=false, the matrix X is not computed and the matrix $A = R$

Parameters

	F	base field
	M	number of rows

Parameters

	N	number of columns
in	A	input matrix
	lda	leading dimension of A
	P	the row permutation
	Qt	the column position of the pivots in the echelon form
	$transform$	decides whether X is computed
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.20.3.35 pReducedRowEchelonForm()

```
template<class Field >
size_t pReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]
```

11.20.3.36 ReducedRowEchelonForm() [2/3]

```
template<class Field , class PSHelper >
size_t ReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]
```

11.20.3.37 Invert() [1/4]

```
template<class Field >
Field::Element_ptr Invert (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    int & nullity )
```

Invert the given matrix in place or computes its nullity if it is singular.

An inplace $2n^3$ algorithm is used.

Parameters

	F	The computation domain
	M	order of the matrix
in, out	A	input matrix ($M \times M$)
	lda	leading dimension of A
	$nullity$	dimension of the kernel of A

Returns

pointer to A and $A \leftarrow A^{-1}$

11.20.3.38 Invert() [2/4]

```
template<class Field >
Field::Element_ptr Invert (
    const Field & F,
    const size_t M,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    int & nullity )
```

Invert the given matrix or computes its nullity if it is singular.

Precondition

X is preallocated and should be large enough to store the $m \times m$ matrix A.

Parameters

	F	The computation domain
	M	order of the matrix
in	A	input matrix ($M \times M$)
	lda	leading dimension of A
out	X	this is the inverse of A if A is invertible (non NULL and <code>nullity = 0</code>). It is untouched otherwise.
	ldx	leading dimension of X
	$nullity$	dimension of the kernel of A

Returns

pointer to $X = A^{-1}$

11.20.3.39 Invert2() [1/2]

```
template<class Field >
Field::Element_ptr Invert2 (
```

```

const Field & F,
const size_t M,
typename Field::Element_ptr A,
const size_t lda,
typename Field::Element_ptr X,
const size_t ldx,
int & nullity )

```

Invert the given matrix or computes its nullity if it is singular.

An $2n^3f$ algorithm is used. This routine can be % faster than `FFPACK::Invert` but is not totally inplace.

Precondition

X is preallocated and should be large enough to store the $m \times m$ matrix A.

Warning

A is overwritten here !

Bug not tested.

Parameters

	<i>F</i>	the computation domain
	<i>M</i>	order of the matrix
in, out	<i>A</i>	input matrix ($M \times M$). On output, A is modified and represents a "psycological" factorisation LU.
	<i>lda</i>	leading dimension of A
out	<i>X</i>	this is the inverse of A if A is invertible (non NULL and <code>nullity = 0</code>). It is untouched otherwise.
	<i>ldx</i>	leading dimension of X
	<i>nullity</i>	dimension of the kernel of A

Returns

pointer to $X = A^{-1}$

Todo this init is not all necessary (done after `fttrtri`)

Todo this init is not all necessary (done after `fttrtri`)

11.20.3.40 CharPoly() [1/8]

```

template<class PolRing >
std::list< typename PolRing::Element > & CharPoly (
    const PolRing & R,
    std::list< typename PolRing::Element > & charp,
    const size_t N,

```

```

typename PolRing::Domain_t::Element_ptr A,
const size_t lda,
typename PolRing::Domain_t::RandIter & G,
const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

Compute the characteristic polynomial of the matrix A.

Parameters

	<i>R</i>	the polynomial ring of charp (contains the base field)
out	<i>charp</i>	the characteristic polynomial of as a list of factors
	<i>N</i>	order of the matrix A
in	<i>A</i>	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	<i>lda</i>	leading dimension of A
	<i>CharpTag</i>	the algorithmic variant
	<i>G</i>	a random iterator (required for the randomized variants LUKrylov and ArithProg)

11.20.3.41 CharPoly() [2/8]

```

template<class PolRing >
PolRing::Element & CharPoly (
    const PolRing & R,
    typename PolRing::Element & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    typename PolRing::Domain_t::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

Compute the characteristic polynomial of the matrix A.

Parameters

	<i>R</i>	the polynomial ring of charp (contains the base field)
out	<i>charp</i>	the characteristic polynomial of as a single polynomial
	<i>N</i>	order of the matrix A
in	<i>A</i>	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	<i>lda</i>	leading dimension of A
	<i>CharpTag</i>	the algorithmic variant
	<i>G</i>	a random iterator (required for the randomized variants LUKrylov and ArithProg)

11.20.3.42 CharPoly() [3/8]

```

template<class PolRing >
PolRing::Element & CharPoly (
    const PolRing & R,
    typename PolRing::Element & charp,

```

```

const size_t N,
typename PolRing::Domain_t::Element_ptr A,
const size_t lda,
const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

Compute the characteristic polynomial of the matrix A.

Parameters

	<i>R</i>	the polynomial ring of charp (contains the base field)
out	<i>charp</i>	the characteristic polynomial of A as a single polynomial
	<i>N</i>	order of the matrix A
in	<i>A</i>	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	<i>lda</i>	leading dimension of A
	<i>CharpTag</i>	the algorithmic variant

11.20.3.43 MinPoly() [1/4]

```

template<class Field , class Polynomial >
Polynomial & MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]

```

Compute the minimal polynomial of the matrix A.

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector: (v, Av, ..., A^kv)

Parameters

	<i>F</i>	the base field
out	<i>minP</i>	the minimal polynomial of A
	<i>N</i>	order of the matrix A
in	<i>A</i>	the input matrix ($N \times N$)
	<i>lda</i>	leading dimension of A

11.20.3.44 MinPoly() [2/4]

```

template<class Field , class Polynomial , class RandIter >
Polynomial & MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    RandIter & G ) [inline]

```

Compute the minimal polynomial of the matrix A .

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector: (v, Av, \dots, A^kv)

Parameters

	F	the base field
out	$minP$	the minimal polynomial of A
	N	order of the matrix A
in	A	the input matrix ($N \times N$)
	lda	leading dimension of A
	G	a random iterator

11.20.3.45 MatVecMinPoly() [1/2]

```
template<class Field , class Polynomial >
Polynomial & MatVecMinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr v,
    const size_t incv ) [inline]
```

Compute the minimal polynomial of the matrix A and a vector v , namely the first linear dependency relation in the Krylov basis $(v, Av, \dots, A^N v)$.

Parameters

	F	the base field
out	$minP$	the minimal polynomial of A and v
	N	order of the matrix A
in	A	the input matrix ($N \times N$)
	lda	leading dimension of A
	K	an $N \times (N + 1)$ matrix containing the vector v on its first row
	ldk	leading dimension of K
	P	[out] (optional) the permutation used in the elimination of the Krylov matrix K

11.20.3.46 Rank() [1/3]

```
template<class Field >
size_t Rank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

Computes the rank of the given matrix using a PLUQ factorization.

The input matrix is modified.

Parameters

	F	base field
	M	row dimension of the matrix
	N	column dimension of the matrix
in	A	input matrix
	lda	leading dimension of A
	psH	(optional) a ParSeqHelper to choose between sequential and parallel execution

11.20.3.47 pRank()

```
template<class Field >
size_t pRank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t numthreads = 0 )
```

11.20.3.48 Rank() [2/3]

```
template<class Field , class PSHelper >
size_t Rank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const PSHelper & psH )
```

11.20.3.49 IsSingular() [1/2]

```
template<class Field >
bool IsSingular (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

Returns true if the given matrix is singular.

The method is a block elimination with early termination

using LQUP factorization with early termination. If $M \neq N$, then the matrix is virtually padded with zeros to make it square and its determinant is zero.

Warning

The input matrix is modified.

Parameters

	F	base field
	M	row dimension of the matrix
	N	column dimension of the matrix.
in, out	A	input matrix
	lda	leading dimension of A

11.20.3.50 Det() [1/6]

```
template<class Field >
Field::Element & Det (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P = NULL,
    size_t * Q = NULL ) [inline]
```

Returns the determinant of the given square matrix.

The method is a block elimination using PLUQ factorization. The input matrix A is overwritten.

Warning

The input matrix is modified.

Parameters

	F	base field
out	det	the determinant of A
	N	the order of the square matrix A.
in, out	A	input matrix
	lda	leading dimension of A
	psH	(optional) a ParSeqHelper to choose between sequential and parallel execution
	P,Q	(optional) row and column permutations to be used by the PLUQ factorization. randomized checkers (see cherckes/checker_det.inl) need them for certification

11.20.3.51 pDet()

```
template<class Field >
Field::Element & pDet (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t numthreads = 0,
```



```
size_t * P = NULL,
size_t * Q = NULL ) [inline]
```

11.20.3.52 Det() [2/6]

```
template<class Field , class PSHelper >
Field::Element & Det (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const PSHelper & psH,
    size_t * P = NULL,
    size_t * Q = NULL )
```

11.20.3.53 Solve() [1/3]

```
template<class Field >
Field::Element_ptr Solve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb ) [inline]
```

Solves a linear system $AX = b$ using PLUQ factorization.

@oaram F base field @oaram M matrix order

Parameters

in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
out	<i>x</i>	output solution vector
	<i>incx</i>	increment of x
	<i>b</i>	input right hand side of the system
	<i>incb</i>	increment of b

11.20.3.54 Solve() [2/3]

```
template<class Field , class PSHelper >
Field::Element_ptr Solve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
```

```

typename Field::Element_ptr x,
const int incx,
typename Field::ConstElement_ptr b,
const int incb,
PSHelper & psH )

```

11.20.3.55 pSolve()

```

template<class Field >
Field::Element_ptr pSolve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb,
    size_t numthreads = 0 ) [inline]

```

11.20.3.56 RandomNullSpaceVector() [1/3]

```

template<class Field >
*void RandomNullSpaceVector (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t incX )

```

Solve $LX = B$ or $XL = B$ in place.

L is $M \times M$ if `Side == FFLAS::FflasLeft` and $N \times N$ if `Side == FFLAS::FflasRight`, B is $M \times N$. Only the R non trivial column of L are stored in the $M \times R$ matrix L Requirement : so that L could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix A .

Parameters

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in, out	<i>A</i>	input matrix of dimension $M \times N$, A is modified to its LU version
	<i>lda</i>	leading dimension of A
out	<i>X</i>	output vector
	<i>incX</i>	increment of X

11.20.3.57 NullSpaceBasis() [1/2]

```
template<class Field >
size_t NullSpaceBasis (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & NS,
    size_t & ldn,
    size_t & NSdim )
```

Computes a basis of the Left/Right nullspace of the matrix A.

return the dimension of the nullspace.

Parameters

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in, out	<i>A</i>	input matrix of dimension M x N, A is modified
	<i>lda</i>	leading dimension of A
out	<i>NS</i>	output matrix of dimension N x NSdim (allocated here)
out	<i>ldn</i>	leading dimension of NS
out	<i>NSdim</i>	the dimension of the Nullspace (N-rank(A))

11.20.3.58 RowRankProfile() [1/3]

```
template<class Field >
size_t RowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Computes the row rank profile of A.

Parameters

	<i>F</i>	base field
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in	<i>A</i>	input matrix of dimension M x N
	<i>lda</i>	leading dimension of A
out	<i>rkprofile</i>	return the rank profile as an array of row indexes, of dimension r=rank(A)
	<i>LuTag</i>	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A modified rkprofile is allocated during the computation.

Returns

R

11.20.3.59 pRowRankProfile()

```
template<class Field >
size_t pRowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]
```

11.20.3.60 RowRankProfile() [2/3]

```
template<class Field , class PSHelper >
size_t RowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    const FFPACK_LU_TAG LuTag,
    PSHelper & psH ) [inline]
```

11.20.3.61 ColumnRankProfile() [1/3]

```
template<class Field >
size_t ColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Computes the column rank profile of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	A	input matrix of dimension
	lda	leading dimension of A
out	$rkprofile$	return the rank profile as an array of row indexes, of dimension $r=\text{rank}(A)$
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

Returns

R

11.20.3.62 pColumnRankProfile()

```
template<class Field >
size_t pColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * rkprofile,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]
```

11.20.3.63 ColumnRankProfile() [2/3]

```
template<class Field , class PSHelper >
size_t ColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * rkprofile,
    const FFPACK_LU_TAG LuTag,
    PSHelper & psH ) [inline]
```

11.20.3.64 RankProfileFromLU()

```
void RankProfileFromLU (
    const size_t * P,
    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const FFPACK_LU_TAG LuTag ) [inline]
```

Recovers the column/row rank profile from the permutation of an LU decomposition.

Works with both the CUP/PLE decompositions (obtained by LUdivine) or the PLUQ decomposition. Assumes that the output vector containing the rank profile is already allocated.

Parameters

	P	the permutation carrying the rank profile information
	N	the row/col dimension for a row/column rank profile
	R	the rank of the matrix
out	$rkprofile$	return the rank profile as an array of indices
	$LuTag$	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.20.3.65 LeadingSubmatrixRankProfiles()

```
size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP ) [inline]
```

Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.

Only works with the PLUQ decomposition Assumes that the output vectors containing the rank profiles are already allocated.

Parameters

<i>P</i>	the permutation carrying the rank profile information
<i>M</i>	the row dimension of the initial matrix
<i>N</i>	the column dimension of the initial matrix
<i>R</i>	the rank of the initial matrix
<i>LSm</i>	the row dimension of the leading submatrix considered
<i>LSn</i>	the column dimension of the leading submatrix considered
<i>P</i>	the row permutation of the PLUQ decomposition
<i>Q</i>	the column permutation of the PLUQ decomposition
<i>RRP</i>	return the row rank profile of the leading submatrix

Returns

the rank of the LSm x LSn leading submatrix

A is modified

Bibliography • Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13.

11.20.3.66 RowRankProfileSubmatrixIndices() [1/2]

```
template<class Field >
size_t RowRankProfileSubmatrixIndices (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

RowRankProfileSubmatrixIndices.

Computes the indices of the submatrix $r \times r$ X of A whose rows correspond to the row rank profile of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	A	input matrix of dimension
	<i>rowindices</i>	array of the row indices of X in A
	<i>colindices</i>	array of the col indices of X in A
	<i>lda</i>	leading dimension of A
out	R	list of indices

rowindices and colindices are allocated during the computation. A is modified

Returns

R

11.20.3.67 ColRankProfileSubmatrixIndices() [1/2]

```
template<class Field >
size_t ColRankProfileSubmatrixIndices (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

Computes the indices of the submatrix $r \times r$ X of A whose columns correspond to the column rank profile of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	A	input matrix of dimension
	<i>rowindices</i>	array of the row indices of X in A
	<i>colindices</i>	array of the col indices of X in A
	<i>lda</i>	leading dimension of A
out	R	list of indices

rowindices and colindices are allocated during the computation.

Warning

A is modified

Returns

R

11.20.3.68 RowRankProfileSubmatrix() [1/2]

```
template<class Field >
size_t RowRankProfileSubmatrix (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & X,
    size_t & R )
```

Computes the $r \times r$ submatrix X of A, by picking the row rank profile rows of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	A	input matrix of dimension M x N
	lda	leading dimension of A
out	X	the output matrix
out	R	list of indices

A is not modified X is allocated during the computation.

Returns

R

11.20.3.69 ColRankProfileSubmatrix() [1/2]

```
template<class Field >
size_t ColRankProfileSubmatrix (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & X,
    size_t & R )
```

Compute the $r \times r$ submatrix X of A, by picking the row rank profile rows of A.

Parameters

	F	base field
	M	number of rows

Parameters

	N	number of columns
in	A	input matrix of dimension $M \times N$
	lda	leading dimension of A
out	X	the output matrix
out	R	list of indices

A is not modified X is allocated during the computation.

Returns

R

11.20.3.70 getTriangular() [1/2]

```
template<class Field >
void getTriangular (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false ) [inline]
```

Extracts a triangular matrix from a compact storage $A=L\backslash U$ of rank R .

if OnlyNonZeroVectors is false, then T and A have the same dimensions Otherwise, T is $R \times N$ if UpLo = FflasUpper, else T is $M \times R$

Parameters

	F	base field
	$UpLo$	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	$diag$	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	M	row dimension of T
	N	column dimension of T
	R	rank of the triangular matrix (how many rows/columns need to be copied)
in	A	input matrix
	lda	leading dimension of A
out	T	output matrix
	ldt	leading dimension of T
	$OnlyNonZeroVectors$	decides whether the last zero rows/columns should be ignored

Todo just one triangular fzero+fassign ?

Todo just one triangular fzero+fassign ?

11.20.3.71 getTriangular() [2/2]

```
template<class Field >
void getTriangular (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

Cleans up a compact storage $A=LU$ to reveal a triangular matrix of rank R .

Parameters

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is revealed
	<i>diag</i>	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
	<i>R</i>	rank of the triangular matrix
<i>in, out</i>	<i>A</i>	input/output matrix
	<i>lda</i>	leading dimension of A

Todo just one triangular fzero+fassign ?

Todo just one triangular fzero+fassign ?

11.20.3.72 getEchelonForm() [1/2]

```
template<class Field >
void getEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
```

```

const size_t ldt,
const bool OnlyNonZeroVectors = false,
const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Extracts a matrix in echelon form from a compact storage $A=L\backslash U$ of rank R obtained by RowEchelonForm or ColumnEchelonForm.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P . row and column dimension of T are greater or equal to that of A

Parameters

	F	base field
	$UpLo$	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	$diag$	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of T
	N	column dimension of T
	R	rank of the triangular matrix (how many rows/columns need to be copied)
	P	positions of the R pivots
in	A	input matrix
	lda	leading dimension of A
out	T	output matrix
	ldt	leading dimension of T
	$OnlyNonZeroVectors$	decides whether the last zero rows/columns should be ignored
	$LuTag$	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.20.3.73 getEchelonForm() [2/2]

```

template<class Field >
void getEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Cleans up a compact storage $A=L\backslash U$ obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank R .

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P .

Parameters

	F	base field
	$UpLo$	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned

Parameters

	<i>diag</i>	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
	<i>R</i>	rank of the triangular matrix (how many rows/columns need to be copied)
	<i>P</i>	positions of the R pivots
<i>in, out</i>	<i>A</i>	input/output matrix
	<i>lda</i>	leading dimension of A
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.20.3.74 getEchelonTransform()

```
template<class Field >
void getEchelonTransform (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Extracts a transformation matrix to echelon form from a compact storage $A=LU$ of rank R obtained by Row↔EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower: T is $N \times N$ (already allocated) such that $A T = C$ is a transformation of A in Column echelon form Else T is $M \times M$ (already allocated) such that $T A = E$ is a transformation of A in Row Echelon form

Parameters

	<i>F</i>	base field
	<i>UpLo</i>	Lower (FflasLower) means Transformation to Column Echelon Form, Upper (FflasUpper), to Row Echelon Form
	<i>diag</i>	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
	<i>R</i>	rank of the triangular matrix
	<i>P</i>	permutation matrix
<i>in</i>	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
<i>out</i>	<i>T</i>	output matrix
	<i>ldt</i>	leading dimension of T
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.20.3.75 getReducedEchelonForm() [1/2]

```
template<class Field >
void getReducedEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Extracts a matrix in echelon form from a compact storage $A=L\backslash U$ of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P . row and column dimension of T are greater or equal to that of A

Parameters

	<i>F</i>	base field
	<i>UpLo</i>	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	<i>diag</i>	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	<i>M</i>	row dimension of T
	<i>N</i>	column dimension of T
	<i>R</i>	rank of the triangular matrix (how many rows/columns need to be copied)
	<i>P</i>	positions of the R pivots
in	<i>A</i>	input matrix
	<i>lda</i>	leading dimension of A
	<i>ldt</i>	leading dimension of T
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)
	<i>OnlyNonZeroVectors</i>	decides whether the last zero rows/columns should be ignored

11.20.3.76 getReducedEchelonForm() [2/2]

```
template<class Field >
void getReducedEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Cleans up a compact storage $A=L\backslash U$ of rank R obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.

Either L or U is in Echelon form (depending on `Uplo`) The echelon structure is defined by the first R values of the array P .

Parameters

	F	base field
	$UpLo$	selects if the upper (<code>FflasUpper</code>) or lower (<code>FflasLower</code>) triangular matrix is returned
	$diag$	selects if the echelon matrix has unit pivots (<code>FflasUnit/NoUnit</code>)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix (how many rows/columns need to be copied)
	P	positions of the R pivots
in, out	A	input/output matrix
	lda	leading dimension of A
	$LuTag$	which factorized form (<code>CUP/PLE</code> if <code>FfpackSlabRecursive</code> , <code>PLUQ</code> if <code>FfpackTileRecursive</code>)

11.20.3.77 `getReducedEchelonTransform()`

```
template<class Field >
void getReducedEchelonTransform (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Extracts a transformation matrix to echelon form from a compact storage $A=L\backslash U$ of rank R obtained by `RowEchelonForm` or `ColumnEchelonForm`.

If `Uplo == FflasLower`: T is $N \times N$ (already allocated) such that $A T = C$ is a transformation of A in Column echelon form Else T is $M \times M$ (already allocated) such that $T A = E$ is a transformation of A in Row Echelon form

Parameters

	F	base field
	$UpLo$	selects Col (<code>FflasLower</code>) or Row (<code>FflasUpper</code>) Echelon Form
	$diag$	selects if the echelon matrix has unit pivots (<code>FflasUnit/NoUnit</code>)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix
	P	permutation matrix
in	A	input matrix

Parameters

	<i>lda</i>	leading dimension of A
out	<i>T</i>	output matrix
	<i>ldt</i>	leading dimension of T
	<i>LuTag</i>	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.20.3.78 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm ) [inline]
```

Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.

11.20.3.79 LQUPtoInverseOfFullRankMinor() [1/2]

```
template<class Field >
Field::Element_ptr LQUPtoInverseOfFullRankMinor (
    const Field & F,
    const size_t rank,
    typename Field::Element_ptr A_factors,
    const size_t lda,
    const size_t * QtPointer,
    typename Field::Element_ptr X,
    const size_t ldx )
```

LQUPtoInverseOfFullRankMinor.

Suppose A has been factorized as L.Q.U.P, with rank r. Then Qt.A.Pt has an invertible leading principal r x r submatrix This procedure efficiently computes the inverse of this minor and puts it into X.

Note

It changes the lower entries of A_factors in the process (NB: unless A was nonsingular and square)

Parameters

<i>F</i>	base field
<i>rank</i>	rank of the matrix.
<i>A_factors</i>	matrix containing the L and U entries of the factorization
<i>lda</i>	leading dimension of A
<i>QtPointer</i>	theLQUP->getQ()->getPointer() (note: getQ returns Qt!)
<i>X</i>	desired location for output
<i>ldx</i>	leading dimension of X

11.20.3.80 RandomNullSpaceVector() [2/3]

```
template<class Field >
void RandomNullSpaceVector (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t incX )
```

Solve $LX = B$ or $XL = B$ in place.

L is $M \times M$ if Side == [FFLAS::FflasLeft](#) and $N \times N$ if Side == [FFLAS::FflasRight](#), B is $M \times N$. Only the R non trivial column of L are stored in the $M \times R$ matrix L Requirement : so that L could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix A.

Parameters

	<i>F</i>	The computation domain
	<i>Side</i>	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	<i>M</i>	number of rows
	<i>N</i>	number of columns
in, out	<i>A</i>	input matrix of dimension $M \times N$, A is modified to its LU version
	<i>lda</i>	leading dimension of A
out	<i>X</i>	output vector
	<i>incX</i>	increment of X

11.20.3.81 solveLB() [1/2]

```
template<class Field >
void solveLB (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr L,
    const size_t ldl,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb )
```

11.20.3.82 solveLB2() [1/2]

```
template<class Field >
void solveLB2 (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
```



```

    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr L,
    const size_t ldl,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb )

```

11.20.3.83 Danilevski()

```

template<class Field , class Polynomial >
std::list< Polynomial > & Danilevski (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )

```

11.20.3.84 buildMatrix()

```

template<class Field >
Field::Element_ptr buildMatrix (
    const Field & F,
    typename Field::ConstElement_ptr E,
    typename Field::ConstElement_ptr C,
    const size_t lda,
    const size_t * B,
    const size_t * T,
    const size_t me,
    const size_t mc,
    const size_t lambda,
    const size_t mu )

```

Bug is this :

11.20.3.85 CharPoly() [4/8]

```

FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp,
    const size_t N,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
    const size_t lda,
    Givaro::ZRing< Givaro::Integer >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    size_t degree ) [inline]

```

11.20.3.86 CharPoly() [5/8]

```
template<>
Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (
    const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > & R,
    Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & charp,
    const size_t N,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::ZRing< Givaro::Integer >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    size_t degree ) [inline]
```

11.20.3.87 Det() [3/6]

```
template<class PSHelper >
FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & det,
    const size_t N,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
    const size_t lda,
    const PSHelper & psH ) [inline]
```

11.20.3.88 Det() [4/6]

```
template<class PSHelper >
Givaro::Integer & Det (
    const Givaro::ZRing< Givaro::Integer > & F,
    Givaro::Integer & det,
    const size_t N,
    Givaro::Integer * A,
    const size_t lda,
    const PSHelper & psH,
    size_t * P,
    size_t * Q ) [inline]
```

11.20.3.89 fsytrf_BC_Crout()

```
template<class Field >
bool fsytrf_BC_Crout (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv ) [inline]
```

11.20.3.90 fsytrf_BC_RL()

```

template<class Field >
size_t fsytrf_BC_RL (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv ) [inline]

```

11.20.3.91 fsytrf_UP_RPM_BC_RL()

```

template<class Field >
size_t fsytrf_UP_RPM_BC_RL (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]

```

11.20.3.92 fsytrf_LOW_RPM_BC_Crout()

```

template<class Field >
size_t fsytrf_LOW_RPM_BC_Crout (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]

```

11.20.3.93 fsytrf_UP_RPM_BC_Crout()

```

template<class Field >
size_t fsytrf_UP_RPM_BC_Crout (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]

```

11.20.3.94 fsytrf_UP_RPM()

```
template<class Field >
size_t fsytrf_UP_RPM (
    const Field & Fi,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P,
    size_t BCThreshold ) [inline]
```

MathP <-[[I] x P1 |] [L_(N1+R2)] [P2^T] |] x [P3^T] [-----|----] [Q2^T]

Changing [U1 V1 | E1 E21 E22] into [U1 E11 E12 V1 E* E*] [0 | L2 \ U2 V21 V22] [U4 V41 0 V42 V43] [0 | M2 0 0] [U3 0 0 V3] [----|-----] [0 0 0] [0 | H1 H21 H22] [0 | U3 V3] [0 | 0] where U4 is the 2R2 x 2R2 matrix formed by interleaving U2, L2^T and H1

11.20.3.95 fsytrf_nonunit() [2/3]

```
template<class Field >
bool fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    FFLAS::ParSeqHelper::Sequential seq,
    size_t threshold ) [inline]
```

11.20.3.96 fsytrf_nonunit() [3/3]

```
template<class Field , class Cut , class Param >
bool fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
    size_t threshold ) [inline]
```

11.20.3.97 fsytrf_RPM()

```
template<class Field >
size_t fsytrf_RPM (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t threshold ) [inline]
```

11.20.3.98 getTridiagonal()

```
template<class Field >
void getTridiagonal (
    const Field & F,
    const size_t N,
    const size_t R,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    size_t * P,
    typename Field::Element_ptr T,
    const size_t ldt ) [inline]
```

11.20.3.99 LUdivine_gauss() [1/2]

```
template<class Field >
size_t LUdivine_gauss (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

11.20.3.100 LUdivine_small() [1/2]

```
template<class Field >
size_t LUdivine_small (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

11.20.3.101 LUdivine() [2/4]

```
template<class Field >
size_t LUdivine (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
```

```

    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag,
    const size_t cutoff ) [inline]

```

Todo std::swap ?

11.20.3.102 LUdivine() [3/4]

```

template<>
size_t LUdivine (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Givaro::Integer * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag,
    const size_t cutoff ) [inline]

```

11.20.3.103 MonotonicCompress()

```

template<class Field >
void MonotonicCompress (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t maxpiv,
    const size_t rowstomove,
    const std::vector< bool > & ispiv ) [inline]

```

11.20.3.104 MonotonicCompressMorePivots()

```

template<class Field >
void MonotonicCompressMorePivots (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t rowstomove,
    const size_t lenP ) [inline]

```

11.20.3.105 MonotonicCompressCycles()

```
template<class Field >
void MonotonicCompressCycles (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t lenP ) [inline]
```

11.20.3.106 MonotonicExpand()

```
template<class Field >
void MonotonicExpand (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t maxpiv,
    const size_t rowstomove,
    const std::vector< bool > & ispiv )
```

11.20.3.107 applyP_block()

```
template<class Field >
void applyP_block (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P ) [inline]
```

11.20.3.108 doApplyS()

```
template<class Field >
void doApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

11.20.3.109 MatrixApplyS() [1/3]

```
template<class Field >
void MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

11.20.3.110 MatrixApplyS() [2/3]

```
template<class Field >
void MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Sequential seq ) [inline]
```

11.20.3.111 MatrixApplyS() [3/3]

```
template<class Field , class Cut , class Param >
void MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

11.20.3.112 PermApplyS()

```
template<class T >
void PermApplyS (
    T * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```


11.20.3.113 doApplyT()

```
template<class Field >
void doApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

11.20.3.114 MatrixApplyT() [1/3]

```
template<class Field >
void MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

11.20.3.115 MatrixApplyT() [2/3]

```
template<class Field >
void MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Sequential seq ) [inline]
```

11.20.3.116 MatrixApplyT() [3/3]

```
template<class Field , class Cut , class Param >
void MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
```

```

    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

11.20.3.117 PermApplyT()

```

template<class T >
void PermApplyT (
    T * A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

11.20.3.118 composePermutationsLLL()

```

void composePermutationsLLL (
    size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]

```

Computes $P1 \times \text{Diag}(I_R, P2)$ where $P1$ is a LAPACK and $P2$ a LAPACK permutation and store the result in $P1$ as a LAPACK permutation.

Parameters

in, out	$P1$	a LAPACK permutation of size N
	$P2$	a LAPACK permutation of size N-R

11.20.3.119 composePermutationsLLM()

```

void composePermutationsLLM (
    size_t * MathP,
    const size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]

```

Computes $P1 \times \text{Diag}(I_R, P2)$ where $P1$ is a LAPACK and $P2$ a LAPACK permutation and store the result in MathP as a MathPermutation format.

Parameters

out		
-----	--	--

a MathPermutation of size N

Parameters

<i>P1</i>	a LAPACK permutation of size N
<i>P2</i>	a LAPACK permutation of size N-R

11.20.3.120 composePermutationsMLM()

```
void composePermutationsMLM (
    size_t * MathP1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]
```

Computes MathP1 x Diag (I_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.

Parameters

in, out	<i>MathP1</i>	a MathPermutation of size N
	<i>P2</i>	a LAPACK permutation of size N-R

11.20.3.121 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
    size_t * P,
    const size_t s ) [inline]
```

11.20.3.122 cyclic_shift_row_col() [1/2]

```
template<class Field >
void cyclic_shift_row_col (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

11.20.3.123 cyclic_shift_row() [1/3]

```
template<class Field >
void cyclic_shift_row (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

11.20.3.124 cyclic_shift_row() [2/3]

```
template<typename T >
void cyclic_shift_row (
    const RNSIntegerMod< T > & F,
    typename T::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

11.20.3.125 cyclic_shift_col() [1/3]

```
template<class Field >
void cyclic_shift_col (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

11.20.3.126 cyclic_shift_col() [2/3]

```
template<typename T >
void cyclic_shift_col (
    const RNSIntegerMod< T > & F,
    typename T::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

11.20.3.127 PLUQ_basecaseV3()

```
template<class Field >
size_t PLUQ_basecaseV3 (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element * A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

11.20.3.128 PLUQ_basecaseV2()

```
template<class Field >
size_t PLUQ_basecaseV2 (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element * A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

11.20.3.129 PLUQ_basecaseCrout()

```
template<class Field >
size_t PLUQ_basecaseCrout (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

11.20.3.130 _PLUQ()

```
template<class Field >
size_t _PLUQ (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    size_t BCThreshold ) [inline]
```

11.20.3.131 PLUQ() [4/6]

```
template<class Cut , class Param >
size_t PLUQ (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Givaro::Integer * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    size_t BCThreshold,
    FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper ) [inline]
```

11.20.3.132 threads_fgemm()

```
template<class Field >
void threads_fgemm (
    const size_t m,
    const size_t n,
    const size_t r,
    int nbthreads,
    size_t * W1,
    size_t * W2,
    size_t * W3,
    size_t gamma )
```

11.20.3.133 threads_ftrsm()

```
template<class Field >
void threads_ftrsm (
    const size_t m,
    const size_t n,
    int nbthreads,
    size_t * t1,
    size_t * t2 )
```

11.20.3.134 PLUQ() [5/6]

```
template<class Field >
size_t PLUQ (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & PSHelper ) [inline]
```

11.20.3.135 fflas_const_cast() [1/3]

```
template<>
rns_double_elt_ptr fflas_const_cast (
    rns_double_elt_cstptr x ) [inline]
```

11.20.3.136 fflas_const_cast() [2/3]

```
template<>
rns_double_elt_cstptr fflas_const_cast (
    rns_double_elt_ptr x ) [inline]
```

11.20.3.137 cyclic_shift_row_col() [2/2]

```
template<typename Base_t >
void cyclic_shift_row_col (
    Base_t * A,
    size_t m,
    size_t n,
    size_t lda )
```

11.20.3.138 cyclic_shift_row() [3/3]

```
template INST_OR_DECL void cyclic_shift_row (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT * A,
    size_t m,
    size_t n,
    size_t lda )
```

11.20.3.139 cyclic_shift_col() [3/3]

```
template INST_OR_DECL void cyclic_shift_col (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT * A,
    size_t m,
    size_t n,
    size_t lda )
```

11.20.3.140 applyP() [4/4]

```
template INST_OR_DECL void applyP (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P )
```

11.20.3.141 fgetrs() [3/4]

```
template INST_OR_DECL void fgetrs (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

11.20.3.142 fgetrs() [4/4]

```
template INST_OR_DECL FFLAS_ELT * fgetrs (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    FFLAS_ELT * X,
    const size_t ldx,
    const FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

11.20.3.143 fgesv() [3/4]

```
template INST_OR_DECL size_t fgesv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

11.20.3.144 fgesv() [4/4]

```
template INST_OR_DECL size_t fgesv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    const FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

11.20.3.145 ftrtri() [2/2]

```
template INST_OR_DECL void ftrtri (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```



```

const FFLAS::FFLAS_UPLO Uplo,
const FFLAS::FFLAS_DIAG Diag,
const size_t N,
FFLAS_ELT * A,
const size_t lda,
const size_t threshold )

```

11.20.3.146 trinv_left() [2/2]

```

template INST_OR_DECL void trinv_left (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * L,
    const size_t ldl,
    FFLAS_ELT * X,
    const size_t idx )

```

11.20.3.147 ftrtrm() [2/2]

```

template INST_OR_DECL void ftrtrm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )

```

11.20.3.148 PLUQ() [6/6]

```

template INST_OR_DECL size_t PLUQ (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q )

```

11.20.3.149 LUdivine() [4/4]

```

template INST_OR_DECL size_t LUdivine (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const FFPACK_LU_TAG LuTag,
    const size_t cutoff )

```

11.20.3.150 LUdivine_small() [2/2]

```
template INST_OR_DECL size_t LUdivine_small (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.151 LUdivine_gauss() [2/2]

```
template INST_OR_DECL size_t LUdivine_gauss (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.152 RowEchelonForm() [3/3]

```
template INST_OR_DECL size_t RowEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.153 ReducedRowEchelonForm() [3/3]

```
template INST_OR_DECL size_t ReducedRowEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.154 ColumnEchelonForm() [3/3]

```
template INST_OR_DECL size_t ColumnEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.155 ReducedColumnEchelonForm() [3/3]

```
template INST_OR_DECL size_t ReducedColumnEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.156 Invert() [3/4]

```
template INST_OR_DECL FFLAS_ELT * Invert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    int & nullity )
```

11.20.3.157 Invert() [4/4]

```
template INST_OR_DECL FFLAS_ELT * Invert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    int & nullity )
```

11.20.3.158 Invert2() [2/2]

```
template INST_OR_DECL FFLAS_ELT * Invert2 (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    int & nullity )
```

11.20.3.159 CharPoly() [6/8]

```
template INST_OR_DECL std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > &
CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

11.20.3.160 CharPoly() [7/8]

```
template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

11.20.3.161 CharPoly() [8/8]

```
template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

11.20.3.162 MinPoly() [3/4]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G )
```

11.20.3.163 MinPoly() [4/4]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda )
```

11.20.3.164 MatVecMinPoly() [2/2]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MatVecMinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * V,
    const size_t incv )
```

11.20.3.165 KrylovElim()

```
template INST_OR_DECL size_t KrylovElim (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const size_t deg,
    size_t * iterates,
    size_t * inviterates,
    const size_t maxit,
    size_t virt )
```

11.20.3.166 SpecRankProfile()

```
template INST_OR_DECL size_t SpecRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t deg,
    size_t * rankProfile )
```

11.20.3.167 Rank() [3/3]

```
template INST_OR_DECL size_t Rank (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

11.20.3.168 IsSingular() [2/2]

```
template INST_OR_DECL bool IsSingular (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

11.20.3.169 Det() [5/6]

```
template INST_OR_DECL FFLAS_ELT & Det (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT & det,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

11.20.3.170 Det() [6/6]

```
template INST_OR_DECL FFLAS_ELT & Det (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT & det,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & parH,
    size_t * P,
    size_t * Q )
```

11.20.3.171 Solve() [3/3]

```
template INST_OR_DECL FFLAS_ELT * Solve (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * x,
    const int incx,
    const FFLAS_ELT * b,
    const int incb )
```

11.20.3.172 solveLB() [2/2]

```
template INST_OR_DECL void solveLB (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * L,
    const size_t ldl,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb )
```

11.20.3.173 solveLB2() [2/2]

```
template INST_OR_DECL void solveLB2 (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * L,
    const size_t ldl,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb )
```

11.20.3.174 RandomNullSpaceVector() [3/3]

```
template INST_OR_DECL void RandomNullSpaceVector (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t incX )
```

11.20.3.175 NullSpaceBasis() [2/2]

```
template INST_OR_DECL size_t NullSpaceBasis (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& NS,
    size_t & ldn,
    size_t & NSdim )
```

11.20.3.176 RowRankProfile() [3/3]

```
template INST_OR_DECL size_t RowRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.177 ColumnRankProfile() [3/3]

```
template INST_OR_DECL size_t ColumnRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * rkprofile,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.178 RowRankProfileSubmatrixIndices() [2/2]

```
template INST_OR_DECL size_t RowRankProfileSubmatrixIndices (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * rowindices,
    size_t * colindices,
    size_t & R )
```

11.20.3.179 ColRankProfileSubmatrixIndices() [2/2]

```
template INST_OR_DECL size_t ColRankProfileSubmatrixIndices (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * rowindices,
    size_t * colindices,
    size_t & R )
```

11.20.3.180 RowRankProfileSubmatrix() [2/2]

```
template INST_OR_DECL size_t RowRankProfileSubmatrix (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    size_t & R )
```

11.20.3.181 ColRankProfileSubmatrix() [2/2]

```
template INST_OR_DECL size_t ColRankProfileSubmatrix (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    size_t & R )
```


11.20.3.182 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [1/2]

```
template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors )
```

11.20.3.183 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [2/2]

```
template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda )
```

11.20.3.184 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2]

```
template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.185 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2]

```
template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.186 getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()

```
template INST_OR_DECL void getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.187 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2]

```
template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.188 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2]

```
template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag )
```

11.20.3.189 getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()

```
template INST_OR_DECL void getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
```

```

    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag )

```

11.20.3.190 LQUPtoInverseOfFullRankMinor() [2/2]

```

template INST_OR_DECL FFLAS_ELT * LQUPtoInverseOfFullRankMinor (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t rank,
    FFLAS_ELT * A_factors,
    const size_t lda,
    const size_t * QtPointer,
    FFLAS_ELT * X,
    const size_t ldx )

```

11.20.3.191 fflas_const_cast() [3/3]

```

template<class T , class CT = const T>
T fflas_const_cast (
    CT x )

```

11.20.3.192 failure()

```

Failure & failure ( ) [inline]

```

11.20.3.193 isOdd() [1/3]

```

template<class T >
bool isOdd (
    const T & a ) [inline]

```

11.20.3.194 isOdd() [2/3]

```

bool isOdd (
    const float & a ) [inline]

```

11.20.3.195 isOdd() [3/3]

```

bool isOdd (
    const double & a ) [inline]

```

11.20.3.196 NonZeroRandomMatrix() [1/2]

```
template<class Field , class RandIter >
Field::Element_ptr NonZeroRandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random non-zero Matrix.

Creates a $m \times n$ matrix with random entries, and at least one of them is non zero.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	A	the matrix (preallocated to at least $m \times lda$ field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

11.20.3.197 NonZeroRandomMatrix() [2/2]

```
template<class Field , class RandIter >
Field::Element_ptr NonZeroRandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random non-zero Matrix.

Creates a $m \times n$ matrix with random entries, and at least one of them is non zero.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	A	the matrix (preallocated to at least $m \times lda$ field elements)
	lda	leading dimension of A

Returns

A .

11.20.3.198 RandomMatrix() [1/2]

```
template<class Field , class RandIter >
Field::Element_ptr RandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix.

Creates a $m \times n$ matrix with random entries.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	A	the matrix (preallocated to at least $m \times lda$ field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A .

11.20.3.199 RandomMatrix() [2/2]

```
template<class Field >
Field::Element_ptr RandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix.

Creates a $m \times n$ matrix with random entries.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
out	A	the matrix (preallocated to at least $m \times lda$ field elements)
	lda	leading dimension of A

Returns

A.

11.20.3.200 RandomTriangularMatrix() [1/2]

```
template<class Field , class RandIter >
Field::Element_ptr RandomTriangularMatrix (
    const Field & F,
    size_t m,
    size_t n,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG Diag,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Triangular Matrix.

Creates a $m \times n$ triangular matrix with random entries. The UpLo parameter defines whether it is upper or lower triangular.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
	UpLo	whether A is upper or lower triangular
out	A	the matrix (preallocated to at least $m \times lda$ field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

11.20.3.201 RandomTriangularMatrix() [2/2]

```
template<class Field >
Field::Element_ptr RandomTriangularMatrix (
    const Field & F,
    size_t m,
    size_t n,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG Diag,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Triangular Matrix.

Creates a $m \times n$ triangular matrix with random entries. The UpLo parameter defines whether it is upper or lower triangular.

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
	$UpLo$	whether A is upper or lower triangular
out	A	the matrix (preallocated to at least $m \times lda$ field elements)
	lda	leading dimension of A

Returns

A .

11.20.3.202 RandInt()

```
size_t RandInt (
    size_t a,
    size_t b ) [inline]
```

11.20.3.203 RandomSymmetricMatrix()

```
template<class Field , class RandIter >
Field::Element_ptr RandomSymmetricMatrix (
    const Field & F,
    size_t n,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Symmetric Matrix.

Creates a $m \times n$ triangular matrix with random entries. The $UpLo$ parameter defines whether it is upper or lower triangular.

Parameters

	F	field
	n	order of A
out	A	the matrix (preallocated to at least $n \times lda$ field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A .

11.20.3.204 RandomMatrixWithRank() [1/2]

```
template<class Field , class RandIter >
Field::Element_ptr RandomMatrixWithRank (
    const Field & F,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank.

Creates an $m \times n$ matrix with random entries and rank r .

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least $m \times lda$ field elements)
lda	leading dimension of A
G	a random iterator

Returns

A .

11.20.3.205 RandomMatrixWithRank() [2/2]

```
template<class Field >
Field::Element_ptr RandomMatrixWithRank (
    const Field & F,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed rank.

Creates an $m \times n$ matrix with random entries and rank r .

Parameters

	F	field
	m	number of rows in A
	n	number of cols in A
	r	rank of the matrix to build
out	A	the matrix (preallocated to at least $m \times lda$ field elements)
	lda	leading dimension of A

Returns

A.

11.20.3.206 RandomIndexSubset()

```
size_t * RandomIndexSubset (
    size_t N,
    size_t R,
    size_t * P ) [inline]
```

Pick uniformly at random a sequence of R distinct elements from the set $\{0, \dots, N - 1\}$ using Knuth's shuffle.

Parameters

	N	the cardinality of the sampling set
	R	the number of elements to sample
out	P	the output sequence (pre-allocated to at least R indices)

11.20.3.207 RandomPermutation()

```
size_t * RandomPermutation (
    size_t N,
    size_t * P ) [inline]
```

Pick uniformly at random a permutation of size N stored in LAPACK format using Knuth's shuffle.

Parameters

	N	the length of the permutation
out	P	the output permutation (pre-allocated to at least N indices)

11.20.3.208 RandomRankProfileMatrix()

```
void RandomRankProfileMatrix (
    size_t M,
    size_t N,
    size_t R,
    size_t * rows,
    size_t * cols ) [inline]
```

Pick uniformly at random an R -subpermutation of dimension $M \times N$: a matrix with only R non-zeros equal to one, in a random rook placement.

Parameters

	M	row dimension
	N	column dimension
out	$rows$	the row position of each non zero element (pre-allocated)
out	$cols$	the column position of each non zero element (pre-allocated)

11.20.3.209 swapval()

```
void swapval (
    size_t k,
    size_t N,
    size_t * P,
    size_t val ) [inline]
```

11.20.3.210 RandomSymmetricRankProfileMatrix()

```
void RandomSymmetricRankProfileMatrix (
    size_t N,
    size_t R,
    size_t * rows,
    size_t * cols ) [inline]
```

Pick uniformly at random a symmetric R-subpermutation of dimension $N \times N$: a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.

Parameters

	<i>N</i>	matrix order
out	<i>rows</i>	the row position of each non zero element (pre-allocated)
out	<i>cols</i>	the column position of each non zero element (pre-allocated)

11.20.3.211 RandomMatrixWithRankandRPM() [1/2]

```
template<class Field , class RandIter >
Field::Element_ptr RandomMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an $m \times n$ matrix with random entries and rank r .

Parameters

<i>F</i>	field
<i>m</i>	number of rows in A
<i>n</i>	number of cols in A
<i>r</i>	rank of the matrix to build
<i>A</i>	the matrix (preallocated to at least $m \times lda$ field elements)
<i>lda</i>	leading dimension of A
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots
<i>G</i>	a random iterator

Returns

A.

11.20.3.212 RandomMatrixWithRankandRPM() [2/2]

```
template<class Field >
Field::Element_ptr RandomMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP ) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an $m \times n$ matrix with random entries and rank r .

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least $m \times lda$ field elements)
lda	leading dimension of A
RRP	the R dimensional array with row positions of the rank profile matrix' pivots
CRP	the R dimensional array with column positions of the rank profile matrix' pivots

Returns

A.

11.20.3.213 RandomSymmetricMatrixWithRankandRPM() [1/2]

```
template<class Field , class RandIter >
Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP,
    RandIter & G ) [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an $n \times n$ symmetric matrix with random entries and rank r .

Parameters

<i>F</i>	field
<i>n</i>	order of A
<i>r</i>	rank of A
<i>A</i>	the matrix (preallocated to at least $n \times lda$ field elements)
<i>lda</i>	leading dimension of A
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots
<i>G</i>	a random iterator

Returns

A .

11.20.3.214 RandomSymmetricMatrixWithRankandRPM() [2/2]

```
template<class Field >
Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP ) [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an $n \times n$ symmetric matrix with random entries and rank r .

Parameters

<i>F</i>	field
<i>n</i>	order of A
<i>r</i>	rank of A
<i>A</i>	the matrix (preallocated to at least $n \times lda$ field elements)
<i>lda</i>	leading dimension of A
<i>RRP</i>	the R dimensional array with row positions of the rank profile matrix' pivots
<i>CRP</i>	the R dimensional array with column positions of the rank profile matrix' pivots

Returns

A .

11.20.3.215 RandomMatrixWithRankandRandomRPM() [1/2]

```
template<class Field , class RandIter >
Field::Element_ptr RandomMatrixWithRankandRandomRPM (
```

```

    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]

```

Random Matrix with prescribed rank, with random rank profile matrix Creates an $m \times n$ matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least $m \times lda$ field elements)
lda	leading dimension of A
G	a random iterator

Returns

A .

11.20.3.216 RandomMatrixWithRankandRandomRPM() [2/2]

```

template<class Field >
Field::Element_ptr RandomMatrixWithRankandRandomRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda ) [inline]

```

Random Matrix with prescribed rank, with random rank profile matrix Creates an $m \times n$ matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

Parameters

F	field
m	number of rows in A
n	number of cols in A
r	rank of the matrix to build
A	the matrix (preallocated to at least $m \times lda$ field elements)
lda	leading dimension of A

Returns

A .

11.20.3.217 RandomSymmetricMatrixWithRankandRandomRPM() [1/2]

```
template<class Field , class RandIter >
Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an $n \times n$ matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

Parameters

F	field
n	order of A
r	rank of A
A	the matrix (preallocated to at least $n \times lda$ field elements)
lda	leading dimension of A
G	a random iterator

Returns

A .

11.20.3.218 RandomSymmetricMatrixWithRankandRandomRPM() [2/2]

```
template<class Field >
Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an $n \times n$ matrix with random entries, rank r and with a rank profile matrix chosen uniformly at random.

Parameters

F	field
n	order of A
r	rank of A
A	the matrix (preallocated to at least $n \times lda$ field elements)
lda	leading dimension of A

Returns

A .

11.20.3.219 RandomMatrixWithDet() [1/2]

```
template<class Field >
Field::Element_ptr RandomMatrixWithDet (
    const Field & F,
    size_t n,
    const typename Field::Element d,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed det.

Creates a $m \times n$ matrix with random entries and rank r .

Parameters

F	field
d	the prescribed value for the determinant of A
n	number of cols in A
A	the matrix to be generated (preallocated to at least $n \times lda$ field elements)
lda	leading dimension of A

Returns

A.

11.20.3.220 RandomMatrixWithDet() [2/2]

```
template<class Field , class RandIter >
Field::Element_ptr RandomMatrixWithDet (
    const Field & F,
    size_t n,
    const typename Field::Element d,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed det.

Creates a $m \times n$ matrix with random entries and rank r .

Parameters

F	field
d	the prescribed value for the determinant of A
n	number of cols in A
A	the matrix to be generated (preallocated to at least $n \times lda$ field elements)
lda	leading dimension of A

Returns

A.

11.20.3.221 maxFieldElt()

```
template<typename Field >
Givaro::Integer maxFieldElt ( )
```

11.20.3.222 maxFieldElt< Givaro::ZRing< Givaro::Integer > >()

```
template<>
Givaro::Integer maxFieldElt< Givaro::ZRing< Givaro::Integer > > ( )
```

11.20.3.223 chooseField()

```
template<typename Field >
Field * chooseField (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

11.20.3.224 chooseField< Givaro::ZRing< int32_t > >()

```
template<>
Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

11.20.3.225 chooseField< Givaro::ZRing< int64_t > >()

```
template<>
Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

11.20.3.226 chooseField< Givaro::ZRing< float > >()

```
template<>
Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```


11.20.3.227 chooseField< Givaro::ZRing< double > >()

```
template<>
Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

11.21 FFPACK::Protected Namespace Reference

Functions

- template<class Field >
size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK_MINPOLY_TAG MinTag=FFpackDense, const size_t kg_mc=0, const size_t kg_mb=0, const size_t kg_j=0)
- template<class Field >
size_t GaussJordan (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t colbeg, const size_t rowbeg, const size_t colsize, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)
Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.
- template<class Field , class Polynomial >
std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)
- template<class Field , class Polynomial >
int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)
- template<class Field , class Polynomial >
std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)
- template<class Field >
void fgemv_kgf (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)
- template<class Field , class Polynomial , class RandIter >
std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr U, const size_t ldu, RandIter &G)
- template<class Field , class Polynomial >
std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)
- template<class PolRing >
void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)
- template<class PolRing >
std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const size_t degree)
- template<class Field , class Polynomial >
std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)

- `template<class Field , class Polynomial >`
`Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr v, const size_t incv, typename Field::Element_ptr K, const size_t ldk, size_t *P)`
- `template<class Field , class Polynomial >`
`Polynomial & Hybrid_KGF_LUK_MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, size_t *P, const FFPACK_MINPOLY_TAG MinTag=FFPACK::FfpackDense, const size_t kg_mc=0, const size_t kg_↵ mb=0, const size_t kg_j=0)`
- `template<class Field >`
`size_t updatedD (const Field &F, size_t *d, size_t k, std::vector< std::vector< typename Field::Element > > &minpt)`
- `template<class Field >`
`size_t newD (const Field &F, size_t *d, bool &KeepOn, const size_t l, const size_t N, typename Field::Element_ptr X, const size_t *Q, std::vector< std::vector< typename Field::Element > > &minpt)`
- `template<class Field >`
`void CompressRows (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`
`void CompressRowsQK (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`
`void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t ↵ _t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`
`void DeCompressRowsQK (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t ↵ _t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`
`void CompressRowsQA (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`
`void DeCompressRowsQA (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t ↵ _t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`
`size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t ↵ _t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ↵ ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK::FFPACK ↵ _MINPOLY_TAG MinTag, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)`

11.21.1 Function Documentation

11.21.1.1 LUdivine_construct() [1/2]

```
template<class Field >
size_t LUdivine_construct (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::Element_ptr u,
```

```

const size_t incu,
size_t * P,
bool computeX,
const FFPACK_MINPOLY_TAG MinTag = FfpackDense,
const size_t kg_mc = 0,
const size_t kg_mb = 0,
const size_t kg_j = 0 )

```

11.21.1.2 GaussJordan()

```

template<class Field >
size_t GaussJordan (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t colbeg,
    const size_t rowbeg,
    const size_t colsize,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.

- Bibliography**
- Algorithm 2.8 of A. Storjohann Thesis 2000,
 - Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013

Parameters

	<i>M</i>	row dimension of A
	<i>N</i>	column dimension of A
<i>in, out</i>	<i>A</i>	an m x n matrix
	<i>lda</i>	leading dimension of A
	<i>P</i>	row permutation
	<i>Q</i>	column permutation
	<i>LuTag</i>	set the base case to a Tile (FfpackGaussJordanTile) or Slab (FfpackGaussJordanSlab) recursive RedEchelon

where the transformation matrix is stored at the pivot column position

11.21.1.3 KellerGehrig()

```

template<class Field , class Polynomial >
std::list< Polynomial > & KellerGehrig (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda )

```

11.21.1.4 KGFast()

```
template<class Field , class Polynomial >
int KGFast (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * kg_mc,
    size_t * kg_mb,
    size_t * kg_j )
```

11.21.1.5 KGFast_generalized()

```
template<class Field , class Polynomial >
std::list< Polynomial > & KGFast_generalized (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

11.21.1.6 fgemv_kgf()

```
template<class Field >
void fgemv_kgf (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    const size_t kg_mc,
    const size_t kg_mb,
    const size_t kg_j )
```

11.21.1.7 LUKrylov()

```
template<class Field , class Polynomial , class RandIter >
std::list< Polynomial > & LUKrylov (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr U,
    const size_t ldu,
    RandIter & G )
```

11.21.1.8 Danilevski()

```
template<class Field , class Polynomial >
std::list< Polynomial > & Danilevski (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
```

```

typename Field::Element_ptr A,
const size_t lda )

```

11.21.1.9 RandomKrylovPrecond()

```

template<class PolRing >
void RandomKrylovPrecond (
    const PolRing & PR,
    std::list< typename PolRing::Element > & completedFactors,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    size_t & Nb,
    typename PolRing::Domain_t::Element_ptr & B,
    size_t & ldb,
    typename PolRing::Domain_t::RandIter & g,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

Todo swap to save space ??

Todo

Todo don't assing K2 c*noc x N but only mas (c,noc) x N and store each one after the other

Todo swap to save space ??

Todo

Todo don't assing K2 c*noc x N but only mas (c,noc) x N and store each one after the other

11.21.1.10 ArithProg()

```

template<class PolRing >
std::list< typename PolRing::Element > & ArithProg (
    const PolRing & PR,
    std::list< typename PolRing::Element > & frobeniusForm,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    const size_t degree ) [inline]

```

11.21.1.11 LUKrylov_KGFast()

```

template<class Field , class Polynomial >
std::list< Polynomial > & LUKrylov_KGFast (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx )

```

11.21.1.12 MatVecMinPoly()

```

template<class Field , class Polynomial >
Polynomial & MatVecMinPoly (
    const Field & F,

```

```

    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr v,
    const size_t incv,
    typename Field::Element_ptr K,
    const size_t ldk,
    size_t * P ) [inline]

```

11.21.1.13 Hybrid_KGF_LUK_MinPoly()

```

template<class Field , class Polynomial >
Polynomial & Hybrid_KGF_LUK_MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    size_t * P,
    const FFPACK_MINPOLY_TAG MinTag = FFPACK::FfpackDense,
    const size_t kg_mc = 0,
    const size_t kg_mb = 0,
    const size_t kg_j = 0 )

```

11.21.1.14 updateD()

```

template<class Field >
size_t updateD (
    const Field & F,
    size_t * d,
    size_t k,
    std::vector< std::vector< typename Field::Element > > & minpt )

```

11.21.1.15 newD()

```

template<class Field >
size_t newD (
    const Field & F,
    size_t * d,
    bool & KeepOn,
    const size_t l,
    const size_t N,
    typename Field::Element_ptr X,
    const size_t * Q,
    std::vector< std::vector< typename Field::Element > > & minpt )

```

11.21.1.16 CompressRows()

```

template<class Field >
void CompressRows (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,

```

```
const size_t * d,
const size_t nb_blocs ) [inline]
```

11.21.1.17 CompressRowsQK()

```
template<class Field >
void CompressRowsQK (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t deg,
    const size_t nb_blocs ) [inline]
```

11.21.1.18 DeCompressRows()

```
template<class Field >
void DeCompressRows (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]
```

11.21.1.19 DeCompressRowsQK()

```
template<class Field >
void DeCompressRowsQK (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t deg,
    const size_t nb_blocs ) [inline]
```

11.21.1.20 CompressRowsQA()

```
template<class Field >
void CompressRowsQA (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]
```

11.21.1.21 DeCompressRowsQA()

```
template<class Field >
void DeCompressRowsQA (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]
```

11.21.1.22 LUdivine_construct() [2/2]

```
template<class Field >
size_t LUdivine_construct (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::Element_ptr u,
    const size_t incu,
    size_t * P,
    bool computeX,
    const FFPACK::FFPACK_MINPOLY_TAG MinTag,
    const size_t kg_mc,
    const size_t kg_mb,
    const size_t kg_j )
```

11.22 Givaro Namespace Reference

Data Structures

- class [ModularBalanced](#)
- class [Montgomery](#)

11.23 MKL_CONFIG Namespace Reference

11.24 Reclnt Namespace Reference

Data Structures

- class [rint](#)
- class [ruint](#)

Chapter 12

Data Structure Documentation

12.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference

Public Types

- typedef [MMHelperAlgo::Winograd value](#)

12.1.1 Member Typedef Documentation

12.1.1.1 value

```
template<class ModeT , class ParSeq >
typedef MMHelperAlgo::Winograd value
```

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference

Public Types

- typedef [MMHelperAlgo::Classic value](#)

12.2.1 Member Typedef Documentation

12.2.1.1 value

```
template<class ParSeq >
typedef MMHelperAlgo::Classic value
```

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.3 ArbitraryPrecIntTag Struct Reference

Arbitrary precision integers: GMP.
`#include <field-traits.h>`

12.3.1 Detailed Description

Arbitrary precision integers: GMP.
The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.4 AreEqual< X, Y > Class Template Reference

```
#include <fflas_enum.h>
```

Static Public Attributes

- static const bool [value](#) = false

12.4.1 Field Documentation

12.4.1.1 value

```
template<class X , class Y >
const bool value = false [static]
```

The documentation for this class was generated from the following file:

- [fflas_enum.h](#)

12.5 AreEqual< X, X > Class Template Reference

```
#include <fflas_enum.h>
```

Static Public Attributes

- static const bool [value](#) = true

12.5.1 Field Documentation

12.5.1.1 value

```
template<class X >
const bool value = true [static]
```

The documentation for this class was generated from the following file:

- [fflas_enum.h](#)

12.6 Argument Struct Reference

```
#include <args-parser.h>
```

Data Fields

- char [c](#)
- const char * [example](#)
- const char * [helpString](#)
- [ArgumentType](#) type
- void * [data](#)

12.6.1 Field Documentation

12.6.1.1 c

```
char c
```

12.6.1.2 example

```
const char* example
```

12.6.1.3 helpString

```
const char* helpString
```

12.6.1.4 type

```
ArgumentType type
```

12.6.1.5 data

```
void* data
```

The documentation for this struct was generated from the following file:

- [args-parser.h](#)

12.7 associatedDelayedField< Field > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [Field](#) [field](#)
- typedef [Field](#) & [type](#)

12.7.1 Member Typedef Documentation

12.7.1.1 field

```
template<class Field >
typedef Field field
```

12.7.1.2 type

```
template<class Field >
typedef Field& type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.8 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FFPACK::RNSInteger](#)< [RNS](#) > [field](#)
- typedef [FFPACK::RNSInteger](#)< [RNS](#) > [type](#)

12.8.1 Member Typedef Documentation

12.8.1.1 field

```
template<typename RNS >
typedef FFPACK::RNSInteger<RNS> field
```

12.8.1.2 type

```
template<typename RNS >
typedef FFPACK::RNSInteger<RNS> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.9 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

12.9.1 Member Typedef Documentation

12.9.1.1 field

```
template<typename T , typename X >
typedef Givaro::ZRing<T> field
```

12.9.1.2 type

```
template<typename T , typename X >
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.10 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

12.10.1 Member Typedef Documentation

12.10.1.1 field

```
template<typename T >
typedef Givaro::ZRing<T> field
```

12.10.1.2 type

```
template<typename T >
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.11 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

12.11.1 Member Typedef Documentation

12.11.1.1 field

```
template<typename T >
typedef Givaro::ZRing<T> field
```

12.11.1.2 type

```
template<typename T >
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.12 Auto Struct Reference

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.13 Bini Struct Reference

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.14 Block Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.15 callLUdivine_small< Element > Class Template Reference

Public Member Functions

- template<class [Field](#) >
size_t [operator\(\)](#) (const [Field](#) &F, const [FFLAS::FFLAS_DIAG](#) Diag, const [FFLAS::FFLAS_TRANSPOSE](#) trans, const size_t M, const size_t N, typename [Field::Element_ptr](#) A, const size_t lda, size_t *P, size_t *Q, const [FFPACK::FFPACK_LU_TAG](#) LuTag)

12.15.1 Member Function Documentation

12.15.1.1 operator>()()

```
template<class Element >
template<class Field >
```

```

size_t operator() (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

The documentation for this class was generated from the following file:

- [ffpack_ludivine.inl](#)

12.16 callLUdivine_small< double > Class Reference

Public Member Functions

- `template<class Field >`
`size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`

12.16.1 Member Function Documentation

12.16.1.1 operator>()

```

template<class Field >
size_t operator() (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

The documentation for this class was generated from the following file:

- [ffpack_ludivine.inl](#)

12.17 callLUdivine_small< float > Class Reference

Public Member Functions

- `template<class Field >`
`size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`

12.17.1 Member Function Documentation

12.17.1.1 operator>()

```

template<class Field >
size_t operator() (

```

```

const Field & F,
const FFLAS::FFLAS_DIAG Diag,
const FFLAS::FFLAS_TRANSPOSE trans,
const size_t M,
const size_t N,
typename Field::Element_ptr A,
const size_t lda,
size_t * P,
size_t * Q,
const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

The documentation for this class was generated from the following file:

- [ffpack_ludivine.inl](#)

12.18 CharpolyFailed Class Reference

```
#include <ffpack.h>
```

The documentation for this class was generated from the following file:

- [ffpack.h](#)

12.19 Checker_Empty< Field > Struct Template Reference

```
#include <checker_empty.h>
```

Public Member Functions

- `template<typename... Params>`
[Checker_Empty](#) (Params... parameters)
- `template<typename... Params>`
 bool [check](#) (Params... parameters)

12.19.1 Constructor & Destructor Documentation

12.19.1.1 Checker_Empty()

```

template<class Field >
template<typename... Params>
Checker_Empty (
    Params... parameters ) [inline]

```

12.19.2 Member Function Documentation

12.19.2.1 check()

```

template<class Field >
template<typename... Params>
bool check (
    Params... parameters ) [inline]

```

The documentation for this struct was generated from the following file:

- [checker_empty.h](#)

12.20 CheckerImplem_charpoly< Field, Polynomial > Class Template Reference

Public Member Functions

- [CheckerImplem_charpoly](#) (const [Field](#) &F_, const [size_t](#) n_, typename [Field::ConstElement_ptr](#) A, [size_t](#) lda_)
- [CheckerImplem_charpoly](#) (typename [Field::RandIter](#) &G, const [size_t](#) n_, typename [Field::ConstElement_ptr](#) A, [size_t](#) lda_)
- [~CheckerImplem_charpoly](#) ()
- bool [check](#) (Polynomial &g)

12.20.1 Constructor & Destructor Documentation

12.20.1.1 CheckerImplem_charpoly() [1/2]

```
template<class Field , class Polynomial >
CheckerImplem\_charpoly (
    const Field & F_,
    const size\_t n_,
    typename Field::ConstElement\_ptr A,
    size\_t lda_ ) [inline]
```

12.20.1.2 CheckerImplem_charpoly() [2/2]

```
template<class Field , class Polynomial >
CheckerImplem\_charpoly (
    typename Field::RandIter & G,
    const size\_t n_,
    typename Field::ConstElement\_ptr A,
    size\_t lda_ ) [inline]
```

12.20.1.3 ~CheckerImplem_charpoly()

```
template<class Field , class Polynomial >
~CheckerImplem\_charpoly ( ) [inline]
```

12.20.2 Member Function Documentation

12.20.2.1 check()

```
template<class Field , class Polynomial >
bool check (
    Polynomial & g ) [inline]
```

The documentation for this class was generated from the following file:

- [checker_charpoly.inl](#)

12.21 CheckerImplem_Det< Field > Class Template Reference

Public Member Functions

- [CheckerImplem_Det](#) (const [Field](#) &F_, [size_t](#) n_, typename [Field::ConstElement_ptr](#) A, [size_t](#) lda)
- [CheckerImplem_Det](#) (typename [Field::RandIter](#) &G, [size_t](#) n_, typename [Field::ConstElement_ptr](#) A, [size_t](#) lda)
- [~CheckerImplem_Det](#) ()
- bool [check](#) (const typename [Field::Element](#) &det, typename [Field::ConstElement_ptr](#) LU, [size_t](#) lda, [size_t](#) *P, [size_t](#) *Q) const
check if the Det factorization is correct.

12.21.1 Constructor & Destructor Documentation

12.21.1.1 CheckerImplem_Det() [1/2]

```
template<class Field >
CheckerImplem_Det (
    const Field & F_,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

12.21.1.2 CheckerImplem_Det() [2/2]

```
template<class Field >
CheckerImplem_Det (
    typename Field::RandIter & G,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

12.21.1.3 ~CheckerImplem_Det()

```
template<class Field >
~CheckerImplem_Det ( ) [inline]
```

12.21.2 Member Function Documentation

12.21.2.1 check()

```
template<class Field >
bool check (
    const typename Field::Element & det,
    typename Field::ConstElement_ptr LU,
    size_t lda,
    size_t * P,
    size_t * Q ) const [inline]
```

check if the Det factorization is correct.

Needs matrix in LU form

Parameters

<i>LU,storage</i>	for L and U
<i>det</i>	
<i>P</i>	
<i>Q</i>	

The documentation for this class was generated from the following file:

- [checker_det.inl](#)

12.22 CheckerImplem_fgemm< Field > Class Template Reference

Public Member Functions

- [CheckerImplem_fgemm](#) (const [Field](#) &F_, const size_t m_, const size_t n_, const size_t k_, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc_)
- [CheckerImplem_fgemm](#) (typename [Field::RandIter](#) &G, const size_t m_, const size_t n_, const size_t k_, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc_)
- [~CheckerImplem_fgemm](#) ()

- bool [check](#) (const [FFLAS::FFLAS_TRANSPOSE](#) ta, const [FFLAS::FFLAS_TRANSPOSE](#) tb, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, typename [Field::ConstElement_ptr](#) C)

12.22.1 Constructor & Destructor Documentation

12.22.1.1 CheckerImplem_fgemm() [1/2]

```
template<class Field >
CheckerImplem_fgemm (
    const Field & F_,
    const size_t m_,
    const size_t n_,
    const size_t k_,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc_ ) [inline]
```

12.22.1.2 CheckerImplem_fgemm() [2/2]

```
template<class Field >
CheckerImplem_fgemm (
    typename Field::RandIter & G,
    const size_t m_,
    const size_t n_,
    const size_t k_,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc_ ) [inline]
```

12.22.1.3 ~CheckerImplem_fgemm()

```
template<class Field >
~CheckerImplem_fgemm ( ) [inline]
```

12.22.2 Member Function Documentation

12.22.2.1 check()

```
template<class Field >
bool check (
    const FFLAS::FFLAS_TRANSPOSE ta,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::ConstElement_ptr C ) [inline]
```

The documentation for this class was generated from the following file:

- [checker_fgemm.inl](#)

12.23 CheckerImplem_ftrsm< Field > Class Template Reference

Public Member Functions

- [CheckerImplem_ftrsm](#) (const [Field](#) &F_, const size_t m, const size_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement_ptr](#) B, const size_t ldb)

- [CheckerImplem_ftsrsm](#) (typename [Field::RandIter](#) &G, const size_t m, const size_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement_ptr](#) B, const size_t ldb)
- [~CheckerImplem_ftsrsm](#) ()
- bool [check](#) (const [FFLAS::FFLAS_SIDE](#) side, const [FFLAS::FFLAS_UPLO](#) uplo, const [FFLAS::FFLAS_TRANSPOSE](#) trans, const [FFLAS::FFLAS_DIAG](#) diag, const size_t m, const size_t n, typename [Field::Element_ptr](#) A, size_t lda, const typename [Field::ConstElement_ptr](#) X, size_t ldx)

12.23.1 Constructor & Destructor Documentation

12.23.1.1 CheckerImplem_ftsrsm() [1/2]

```
template<class Field >
CheckerImplem_ftsrsm (
    const Field & F_,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

12.23.1.2 CheckerImplem_ftsrsm() [2/2]

```
template<class Field >
CheckerImplem_ftsrsm (
    typename Field::RandIter & G,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

12.23.1.3 ~CheckerImplem_ftsrsm()

```
template<class Field >
~CheckerImplem_ftsrsm ( ) [inline]
```

12.23.2 Member Function Documentation

12.23.2.1 check()

```
template<class Field >
bool check (
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_UPLO uplo,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const FFLAS::FFLAS_DIAG diag,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    const typename Field::ConstElement_ptr X,
    size_t ldx ) [inline]
```

The documentation for this class was generated from the following file:

- [checker_ftsrsm.inl](#)

12.24 CheckerImplem_invert< Field > Class Template Reference

Public Member Functions

- [CheckerImplem_invert](#) (const [Field](#) &F_, const size_t m_, typename [Field::ConstElement_ptr](#) A, const size_t lda_)
- [CheckerImplem_invert](#) (typename [Field::RandIter](#) &G, const size_t m_, typename [Field::ConstElement_ptr](#) A, const size_t lda_)
- [~CheckerImplem_invert](#) ()
- bool [check](#) (typename [Field::ConstElement_ptr](#) A, int nullity)

12.24.1 Constructor & Destructor Documentation

12.24.1.1 CheckerImplem_invert() [1/2]

```
template<class Field >
CheckerImplem_invert (
    const Field & F_,
    const size_t m_,
    typename Field::ConstElement_ptr A,
    const size_t lda_ ) [inline]
```

12.24.1.2 CheckerImplem_invert() [2/2]

```
template<class Field >
CheckerImplem_invert (
    typename Field::RandIter & G,
    const size_t m_,
    typename Field::ConstElement_ptr A,
    const size_t lda_ ) [inline]
```

12.24.1.3 ~CheckerImplem_invert()

```
template<class Field >
~CheckerImplem_invert ( ) [inline]
```

12.24.2 Member Function Documentation

12.24.2.1 check()

```
template<class Field >
bool check (
    typename Field::ConstElement_ptr A,
    int nullity ) [inline]
```

The documentation for this class was generated from the following file:

- [checker_invert.inl](#)

12.25 CheckerImplem_PLUQ< Field > Class Template Reference

Public Member Functions

- [CheckerImplem_PLUQ](#) (const [Field](#) &F_, size_t m_, size_t n_, typename [Field::ConstElement_ptr](#) A, size_t lda)
- [CheckerImplem_PLUQ](#) (typename [Field::RandIter](#) &G, size_t m_, size_t n_, typename [Field::ConstElement_ptr](#) A, size_t lda)
- [~CheckerImplem_PLUQ](#) ()
- bool [check](#) (typename [Field::ConstElement_ptr](#) A, size_t lda, const [FFLAS::FFLAS_DIAG](#) Diag, size_t r, size_t *P, size_t *Q) const
check if the PLUQ factorization is correct.

12.25.1 Constructor & Destructor Documentation

12.25.1.1 CheckerImplem_PLUQ() [1/2]

```
template<class Field >
CheckerImplem_PLUQ (
    const Field & F_,
    size_t m_,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

12.25.1.2 CheckerImplem_PLUQ() [2/2]

```
template<class Field >
CheckerImplem_PLUQ (
    typename Field::RandIter & G,
    size_t m_,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

12.25.1.3 ~CheckerImplem_PLUQ()

```
template<class Field >
~CheckerImplem_PLUQ ( ) [inline]
```

12.25.2 Member Function Documentation

12.25.2.1 check()

```
template<class Field >
bool check (
    typename Field::ConstElement_ptr A,
    size_t lda,
    const FFLAS::FFLAS_DIAG Diag,
    size_t r,
    size_t * P,
    size_t * Q ) const [inline]
```

check if the PLUQ factorization is correct.

Returns true if $w - P(L(U(Q.v))) == 0$

Parameters

<i>A</i>	
<i>r</i>	
<i>P</i>	
<i>Q</i>	

The documentation for this class was generated from the following file:

- [checker_pluq.inl](#)

12.26 Classic Struct Reference

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.27 Column Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.28 CompactElement< Element > Struct Template Reference

Public Types

- typedef Element [type](#)

12.28.1 Member Typedef Documentation

12.28.1.1 type

```
template<class Element >
typedef Element type
```

The documentation for this struct was generated from the following file:

- [test-io.C](#)

12.29 CompactElement< double > Struct Reference

Public Types

- typedef [int32_t](#) [type](#)

12.29.1 Member Typedef Documentation

12.29.1.1 type

```
typedef int32\_t type
```

The documentation for this struct was generated from the following file:

- [test-io.C](#)

12.30 CompactElement< float > Struct Reference

Public Types

- typedef [int16_t](#) [type](#)

12.30.1 Member Typedef Documentation

12.30.1.1 type

```
typedef int16\_t type
```

The documentation for this struct was generated from the following file:

- [test-io.C](#)

12.31 CompactElement< int16_t > Struct Reference

Public Types

- typedef [int8_t](#) [type](#)

12.31.1 Member Typedef Documentation

12.31.1.1 type

typedef [int8_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

12.32 CompactElement< int32_t > Struct Reference

Public Types

- typedef [int16_t](#) type

12.32.1 Member Typedef Documentation

12.32.1.1 type

typedef [int16_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

12.33 CompactElement< int64_t > Struct Reference

Public Types

- typedef [int32_t](#) type

12.33.1 Member Typedef Documentation

12.33.1.1 type

typedef [int32_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

12.34 compatible_data_type< Field > Struct Template Reference

Static Public Attributes

- static constexpr bool [value](#) = true

12.34.1 Field Documentation

12.34.1.1 value

template<typename [Field](#) >

constexpr bool value = true [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

12.35 compatible_data_type< Givaro::ZRing< double > > Struct Reference

Static Public Attributes

- static constexpr bool [value](#) = false

12.35.1 Field Documentation

12.35.1.1 value

```
constexpr bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

12.36 compatible_data_type< Givaro::ZRing< float > > Struct Reference

Static Public Attributes

- static constexpr bool [value](#) = false

12.36.1 Field Documentation

12.36.1.1 value

```
constexpr bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

12.37 Compose< H1, H2 > Struct Template Reference

Public Member Functions

- [Compose](#) ()
- [Compose](#) (const [Compose](#) &[other](#))
- [Compose](#) (const [Sequential](#) &[S](#))
- [Compose](#) (size_t [th1](#), size_t [th2](#))
- [Compose](#) (const [H1](#) &[o1](#), const [H2](#) &[o2](#))
- [H1 first_component](#) () const
- [H2 second_component](#) () const

Friends

- std::ostream & [operator<<](#) (std::ostream &[o](#), const [Compose](#) &[c](#))

12.37.1 Constructor & Destructor Documentation

12.37.1.1 Compose() [1/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose ( ) [inline]
```

12.37.1.2 Compose() [2/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
    const Compose< H1, H2 > & other ) [inline]
```

12.37.1.3 Compose() [3/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
    const Sequential & S ) [inline]
```


12.37.1.4 Compose() [4/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
    size_t th1,
    size_t th2 ) [inline]
```

12.37.1.5 Compose() [5/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
    const H1 & o1,
    const H2 & o2 ) [inline]
```

12.37.2 Member Function Documentation**12.37.2.1 first_component()**

```
template<typename H1 = Sequential, typename H2 = Sequential>
H1 first_component ( ) const [inline]
```

12.37.2.2 second_component()

```
template<typename H1 = Sequential, typename H2 = Sequential>
H2 second_component ( ) const [inline]
```

12.37.3 Friends And Related Symbol Documentation**12.37.3.1 operator<<**

```
template<typename H1 = Sequential, typename H2 = Sequential>
std::ostream & operator<< (
    std::ostream & o,
    const Compose< H1, H2 > & c ) [friend]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.38 Const_int_t< n > Class Template Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

12.39 Const_uint_t< n > Class Template Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

12.40 Simd128_impl< true, true, false, 2 >::Converter Union Reference**Data Fields**

- [vect_t v](#)
- [scalar_t t](#) [vect_size]

12.40.1 Field Documentation

12.40.1.1 `v`

`vect_t v`

12.40.1.2 `t`

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128_int16.inl](#)

12.41 `Simd128_impl< true, true, false, 4 >::Converter Union Reference`

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.41.1 Field Documentation

12.41.1.1 `v`

`vect_t v`

12.41.1.2 `t`

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128_int32.inl](#)

12.42 `Simd128_impl< true, true, false, 8 >::Converter Union Reference`

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.42.1 Field Documentation

12.42.1.1 `v`

`vect_t v`

12.42.1.2 `t`

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128_int64.inl](#)

12.43 `Simd128_impl< true, true, true, 2 >::Converter Union Reference`

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.43.1 Field Documentation

12.43.1.1 v

[vect_t](#) v

12.43.1.2 t

[scalar_t](#) t[vect_size]

The documentation for this union was generated from the following file:

- [simd128_int16.inl](#)

12.44 Simd128_impl< true, true, true, 4 >::Converter Union Reference

Data Fields

- [vect_t](#) v
- [scalar_t](#) t[vect_size]

12.44.1 Field Documentation

12.44.1.1 v

[vect_t](#) v

12.44.1.2 t

[scalar_t](#) t[vect_size]

The documentation for this union was generated from the following file:

- [simd128_int32.inl](#)

12.45 Simd128_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- [vect_t](#) v
- [scalar_t](#) t[vect_size]

12.45.1 Field Documentation

12.45.1.1 v

[vect_t](#) v

12.45.1.2 t

[scalar_t](#) t[vect_size]

The documentation for this union was generated from the following file:

- [simd128_int64.inl](#)

12.46 Simd256_impl< true, true, false, 2 >::Converter Union Reference

Data Fields

- [vect_t](#) v
- [scalar_t](#) t[vect_size]

12.46.1 Field Documentation

12.46.1.1 `v`

`vect_t v`

12.46.1.2 `t`

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd256_int16.inl](#)

12.47 `Simd256_impl< true, true, false, 4 >::Converter Union Reference`

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.47.1 Field Documentation

12.47.1.1 `v`

`vect_t v`

12.47.1.2 `t`

`scalar_t t`

The documentation for this union was generated from the following files:

- [simd256_int32.inl](#)
- [simd512_int32.inl](#)

12.48 `Simd256_impl< true, true, false, 8 >::Converter Union Reference`

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.48.1 Field Documentation

12.48.1.1 `v`

`vect_t v`

12.48.1.2 `t`

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd256_int64.inl](#)

12.49 `Simd256_impl< true, true, true, 2 >::Converter Union Reference`

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.49.1 Field Documentation

12.49.1.1 v

`vect_t v`

12.49.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd256_int16.inl](#)

12.50 Simd256_impl< true, true, true, 4 >::Converter Union Reference

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.50.1 Field Documentation

12.50.1.1 v

`vect_t v`

12.50.1.2 t

`scalar_t t`

The documentation for this union was generated from the following files:

- [simd256_int32.inl](#)
- [simd512_int32.inl](#)

12.51 Simd256_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.51.1 Field Documentation

12.51.1.1 v

`vect_t v`

12.51.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd256_int64.inl](#)

12.52 Simd512_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- [vect_t v](#)
- [scalar_t t\[vect_size\]](#)

12.52.1 Field Documentation

12.52.1.1 `v`

`vect_t v`

12.52.1.2 `t`

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd512_int64.inl](#)

12.53 `Simd512_impl< true, true, true, 8 >::Converter` Union Reference

Data Fields

- `vect_t v`
- `scalar_t t[vect_size]`

12.53.1 Field Documentation

12.53.1.1 `v`

`vect_t v`

12.53.1.2 `t`

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd512_int64.inl](#)

12.54 `ConvertTo< T > Struct` Template Reference

Force conversion to appropriate element type of `ElementCategory T`.

```
#include <field-traits.h>
```

12.54.1 Detailed Description

```
template<class T>
```

```
struct FFLAS::ModeCategories::ConvertTo< T >
```

Force conversion to appropriate element type of `ElementCategory T`.

e.g.

- `ConvertTo<ElementCategories::MachineFloatTag>` tries conversion of `Modular<int>` to `Modular<double>`
- `ConvertTo<ElementCategories::FixedPreIntTag>` tries conversion of `Modular<Integer>` to `Modular<RecInt<K>>`
- `ConvertTo<ElementCategories::ArbitraryPreIntTag>` tries conversion of `Modular<Integer>` to `RNSInteger`

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.55 `Coo< ValT, IdxT > Struct` Template Reference

Public Types

- using `Self` = `Coo< ValT, IdxT >`

Public Member Functions

- `Coo` (`ValT v, IdxT r, IdxT c`)
- `Coo` ()=default
- `Coo` (`const Self &`)=default
- `Coo` (`Self &&`)=default
- `Self & operator=` (`const Self &`)=default
- `Self & operator=` (`Self &&`)=default

Data Fields

- `ValT val` = 0
- `IdxT row` = 0
- `IdxT col` = 0

12.55.1 Member Typedef Documentation**12.55.1.1 Self**

```
template<class ValT , class IdxT >
using Self = Coo<ValT, IdxT>
```

12.55.2 Constructor & Destructor Documentation**12.55.2.1 Coo() [1/4]**

```
template<class ValT , class IdxT >
Coo (
    ValT v,
    IdxT r,
    IdxT c ) [inline]
```

12.55.2.2 Coo() [2/4]

```
template<class ValT , class IdxT >
Coo ( ) [default]
```

12.55.2.3 Coo() [3/4]

```
template<class ValT , class IdxT >
Coo (
    const Self & ) [default]
```

12.55.2.4 Coo() [4/4]

```
template<class ValT , class IdxT >
Coo (
    Self && ) [default]
```

12.55.3 Member Function Documentation**12.55.3.1 operator=() [1/2]**

```
template<class ValT , class IdxT >
Self & operator= (
    const Self & ) [default]
```

12.55.3.2 operator=() [2/2]

```
template<class ValT , class IdxT >
Self & operator= (
    Self && ) [default]
```

12.55.4 Field Documentation**12.55.4.1 val**

```
template<class ValT , class IdxT >
ValT val = 0
```

12.55.4.2 row

```
template<class ValT , class IdxT >
IdxT row = 0
```

12.55.4.3 col

```
template<class ValT , class IdxT >
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- [csr_hyb_utils.inl](#)

12.56 Coo< Field > Struct Template Reference

```
#include <read_sparse.h>
```

Public Member Functions

- [Coo](#) ()=default
- [Coo](#) (typename [Field::Element](#) v, [index_t](#) r, [index_t](#) c)
- [Coo](#) (const [Self](#) &)=default
- [Coo](#) ([Self](#) &&)=default
- [Self](#) & [operator=](#) (const [Self](#) &)=default
- [Self](#) & [operator=](#) ([Self](#) &&)=default

Data Fields

- [Field::Element](#) val = 0
- [index_t](#) col = 0
- [index_t](#) row = 0
- bool [deleted](#) = false

12.56.1 Constructor & Destructor Documentation**12.56.1.1 Coo()** [1/4]

```
template<class Field >
Coo ( ) [default]
```

12.56.1.2 Coo() [2/4]

```
template<class Field >
Coo (
    typename Field::Element v,
    index\_t r,
    index\_t c ) [inline]
```


12.56.1.3 `Coo()` [3/4]

```
template<class Field >
Coo (
    const Self & ) [default]
```

12.56.1.4 `Coo()` [4/4]

```
template<class Field >
Coo (
    Self && ) [default]
```

12.56.2 Member Function Documentation**12.56.2.1 `operator=()` [1/2]**

```
template<class Field >
Self & operator= (
    const Self & ) [default]
```

12.56.2.2 `operator=()` [2/2]

```
template<class Field >
Self & operator= (
    Self && ) [default]
```

12.56.3 Field Documentation**12.56.3.1 `val`**

```
template<class Field >
Field::Element val = 0
```

12.56.3.2 `col`

```
template<class Field >
index_t col = 0
```

12.56.3.3 `row`

```
template<class Field >
index_t row = 0
```

12.56.3.4 `deleted`

```
template<class Field >
bool deleted = false
```

The documentation for this struct was generated from the following file:

- [read_sparse.h](#)

12.57 `Coo< ValT, IdxT >` Struct Template Reference**Public Types**

- using `Self` = `Coo< ValT, IdxT >`

Public Member Functions

- `Coo` (`ValT v`, `IdxT r`, `IdxT c`)
- `Coo` ()=default
- `Coo` (const `Self` &)=default
- `Coo` (`Self` &&)=default
- `Self` & `operator=` (const `Self` &)=default
- `Self` & `operator=` (`Self` &&)=default

Data Fields

- `ValT val` = 0
- `IdxT row` = 0
- `IdxT col` = 0

12.57.1 Member Typedef Documentation

12.57.1.1 Self

```
template<class ValT , class IdxT >
using Self = Coo<ValT, IdxT>
```

12.57.2 Constructor & Destructor Documentation

12.57.2.1 Coo() [1/4]

```
template<class ValT , class IdxT >
Coo (
    ValT v,
    IdxT r,
    IdxT c ) [inline]
```

12.57.2.2 Coo() [2/4]

```
template<class ValT , class IdxT >
Coo ( ) [default]
```

12.57.2.3 Coo() [3/4]

```
template<class ValT , class IdxT >
Coo (
    const Self & ) [default]
```

12.57.2.4 Coo() [4/4]

```
template<class ValT , class IdxT >
Coo (
    Self && ) [default]
```

12.57.3 Member Function Documentation

12.57.3.1 operator=() [1/2]

```
template<class ValT , class IdxT >
Self & operator= (
    const Self & ) [default]
```

12.57.3.2 operator=() [2/2]

```
template<class ValT , class IdxT >
Self & operator= (
    Self && ) [default]
```

12.57.4 Field Documentation**12.57.4.1 val**

```
template<class ValT , class IdxT >
ValT val = 0
```

12.57.4.2 row

```
template<class ValT , class IdxT >
IdxT row = 0
```

12.57.4.3 col

```
template<class ValT , class IdxT >
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- [sell_utils.inl](#)

12.58 CooMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

Data Fields

- [FFLAS::Sparse< Field, SparseMatrix_t::COO, int16_t > * _coo16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::COO, int32_t > * _coo32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::COO, int64_t > * _coo64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::COO_ZO, int16_t > * _coo16_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::COO_ZO, int32_t > * _coo32_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::COO_ZO, int64_t > * _coo64_zo](#) = nullptr

12.58.1 Field Documentation**12.58.1.1 _coo16**

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::COO, int16_t>* _coo16 = nullptr
```

12.58.1.2 _coo32

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::COO, int32_t>* _coo32 = nullptr
```

12.58.1.3 _coo64

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::COO, int64_t>* _coo64 = nullptr
```

12.58.1.4 _coo16_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int16_t>* _coo16_zo = nullptr
```

12.58.1.5 _coo32_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int32_t>* _coo32_zo = nullptr
```

12.58.1.6 _coo64_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int64_t>* _coo64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas_sparse.h](#)

12.59 CsrMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

Data Fields

- [FFLAS::Sparse< Field, SparseMatrix_t::CSR, int16_t > * _csr16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::CSR, int32_t > * _csr32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::CSR, int64_t > * _csr64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::CSR_ZO, int16_t > * _csr16_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::CSR_ZO, int32_t > * _csr32_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::CSR_ZO, int64_t > * _csr64_zo](#) = nullptr

12.59.1 Field Documentation**12.59.1.1 _csr16**

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int16_t>* _csr16 = nullptr
```

12.59.1.2 _csr32

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int32_t>* _csr32 = nullptr
```

12.59.1.3 _csr64

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int64_t>* _csr64 = nullptr
```

12.59.1.4 _csr16_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int16_t>* _csr16_zo = nullptr
```

12.59.1.5 _csr32_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int32_t>* _csr32_zo = nullptr
```

12.59.1.6 _csr64_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int64_t>* _csr64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas_sparse.h](#)

12.60 DefaultBoundedTag Struct Reference

Use standard field operations, but keeps track of bounds on input and output.

```
#include <field-traits.h>
```

12.60.1 Detailed Description

Use standard field operations, but keeps track of bounds on input and output.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.61 DefaultTag Struct Reference

No specific mode of action: use standard field operations.

```
#include <field-traits.h>
```

12.61.1 Detailed Description

No specific mode of action: use standard field operations.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.62 DelayedTag Struct Reference

Performs field operations with delayed mod reductions. Ensures result is reduced.

```
#include <field-traits.h>
```

12.62.1 Detailed Description

Performs field operations with delayed mod reductions. Ensures result is reduced.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.63 ElementTraits< Element > Struct Template Reference

[ElementTraits](#).

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::GenericTag](#) value

12.63.1 Detailed Description

```
template<class Element>
```

```
struct FFLAS::ElementTraits< Element >
```

[ElementTraits](#).

12.63.2 Member Typedef Documentation

12.63.2.1 value

```
template<class Element >
```

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.64 ElementTraits< double > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineFloatTag](#) value

12.64.1 Member Typedef Documentation

12.64.1.1 value

```
typedef ElementCategories::MachineFloatTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.65 ElementTraits< FFPACK::rns_double_elt > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::RNSElementTag](#) value

12.65.1 Member Typedef Documentation

12.65.1.1 value

```
typedef ElementCategories::RNSElementTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.66 ElementTraits< float > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineFloatTag](#) value

12.66.1 Member Typedef Documentation

12.66.1.1 value

```
typedef ElementCategories::MachineFloatTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.67 ElementTraits< Givaro::Integer > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::ArbitraryPrecIntTag](#) value

12.67.1 Member Typedef Documentation

12.67.1.1 value

typedef [ElementCategories::ArbitraryPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.68 ElementTraits< int16_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineIntTag](#) value

12.68.1 Member Typedef Documentation

12.68.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.69 ElementTraits< int32_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineIntTag](#) value

12.69.1 Member Typedef Documentation

12.69.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.70 ElementTraits< int64_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineIntTag](#) value

12.70.1 Member Typedef Documentation

12.70.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.71 ElementTraits< int8_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineIntTag](#) value

12.71.1 Member Typedef Documentation

12.71.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.72 ElementTraits< RecInt::rint< K > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

12.72.1 Member Typedef Documentation

12.72.1.1 value

```
template<size_t K>
```

```
typedef ElementCategories::FixedPrecIntTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.73 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

12.73.1 Member Typedef Documentation

12.73.1.1 value

```
template<size_t K, int MG>
```

```
typedef ElementCategories::FixedPrecIntTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.74 ElementTraits< RecInt::ruint< K > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

12.74.1 Member Typedef Documentation

12.74.1.1 value

```
template<size_t K>
```

```
typedef ElementCategories::FixedPrecIntTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.75 ElementTraits< uint16_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineIntTag](#) value

12.75.1 Member Typedef Documentation

12.75.1.1 value

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.76 ElementTraits< uint32_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineIntTag](#) value

12.76.1 Member Typedef Documentation

12.76.1.1 value

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.77 ElementTraits< uint64_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineIntTag](#) value

12.77.1 Member Typedef Documentation**12.77.1.1 value**

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.78 ElementTraits< uint8_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ElementCategories::MachineIntTag](#) value

12.78.1 Member Typedef Documentation**12.78.1.1 value**

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.79 EIMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

Data Fields

- [FFLAS::Sparse< Field, SparseMatrix_t::ELL, int16_t > * _ell16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::ELL, int32_t > * _ell32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::ELL, int64_t > * _ell64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::ELL_ZO, int16_t > * _ell16_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::ELL_ZO, int32_t > * _ell32_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix_t::ELL_ZO, int64_t > * _ell64_zo](#) = nullptr

12.79.1 Field Documentation**12.79.1.1 _ell16**

```
template<class Field >
```

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int16_t>* _ell16 = nullptr
```

12.79.1.2 _ell32

```
template<class Field >
```

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int32_t>* _ell32 = nullptr
```

12.79.1.3 _ell64

```
template<class Field >
```

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int64_t>* _ell64 = nullptr
```

12.79.1.4 _ell16_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int16_t>* _ell16_zo = nullptr
```

12.79.1.5 _ell32_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int32_t>* _ell32_zo = nullptr
```

12.79.1.6 _ell64_zo

```
template<class Field >
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int64_t>* _ell64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas_sparse.h](#)

12.80 Failure Class Reference

A precondition failed.

```
#include <debug.h>
```

Public Member Functions

- [Failure](#) ()
- void [operator\(\)](#) (const char *function, int line, const char *check)
- void [operator\(\)](#) (const char *function, const char *file, int line, const char *check)
- void [setErrorStream](#) (std::ostream &stream)
- std::ostream & [print](#) (std::ostream &o) const

Protected Attributes

- std::ostream * [_errorStream](#)

12.80.1 Detailed Description

A precondition failed.

The `throw` mechanism is usually used here as in

```
if (!check)
failure()(__func__, __LINE__, "this check just failed");
```

The parameters of the constructor help debugging.

12.80.2 Constructor & Destructor Documentation**12.80.2.1 Failure()**

```
Failure ( ) [inline]
```

12.80.3 Member Function Documentation**12.80.3.1 operator>() [1/2]**

```
void operator() (
    const char * function,
    int line,
    const char * check ) [inline]
```

A precondition failed.

Parameters

<i>function</i>	usually <code>__func__</code> , the function that threw the error
<i>line</i>	usually <code>__LINE__</code> , the line where it happened
<i>check</i>	a string telling what failed.

12.80.3.2 operator>() [2/2]

```
void operator() (
    const char * function,
    const char * file,
    int line,
    const char * check ) [inline]
```

A precondition failed. The parameter help debugging. This is not much different from the previous except we can digg faster in the file where the exception was triggered.

Parameters

<i>function</i>	usually <code>__func__</code> , the function that threw the error
<i>file</i>	usually <code>__FILE__</code> , the file where this function is
<i>line</i>	usually <code>__LINE__</code> , the line where it happened
<i>check</i>	a string telling what failed.

12.80.3.3 setErrorMessage()

```
void setErrorMessage (
    std::ostream & stream )
```

12.80.3.4 print()

```
std::ostream & print (
    std::ostream & o ) const [inline]
```

overload the virtual print of LinboxError.

Parameters

<i>o</i>	output stream
----------	---------------

12.80.4 Field Documentation**12.80.4.1 _errorMessage**

```
std::ostream* _errorMessage [protected]
```

The documentation for this class was generated from the following file:

- [debug.h](#)

12.81 FailureCharpolyCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers_ffpack.h](#)

12.82 FailureDetCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers_ffpack.h](#)

12.83 FailureFgemmCheck Class Reference

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- [checkers_fflas.h](#)

12.84 FailureInvertCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers_ffpack.h](#)

12.85 FailurePLUQCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers_ffpack.h](#)

12.86 FailureTrsmCheck Class Reference

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- [checkers_fflas.h](#)

12.87 FieldSimd< _Field > Class Template Reference

Public Types

- using [Field](#) = [_Field](#)
- using [Element](#) = typename [Field::Element](#)
- using [simd](#) = [Simd](#)< typename [_Field::Element](#) >
- using [vect_t](#) = typename [simd::vect_t](#)
- using [scalar_t](#) = typename [simd::scalar_t](#)

Public Member Functions

- [FieldSimd](#) (const [Field](#) &f)
- [FieldSimd](#) (const [Self](#) &)=default
- [FieldSimd](#) ([Self](#) &&)=default
- [Self](#) & operator= (const [Self](#) &)=default
- [Self](#) & operator= ([Self](#) &&)=default
- [INLINE vect_t](#) init ([vect_t](#) &x, const [vect_t](#) a) const
- [INLINE vect_t](#) init (const [vect_t](#) a) const
- [INLINE vect_t](#) add ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- [INLINE vect_t](#) add (const [vect_t](#) a, const [vect_t](#) b)

- `INLINE vect_t addin (vect_t &a, const vect_t b) const`
- `INLINE vect_t add_r (vect_t &c, const vect_t a, const vect_t b) const`
- `INLINE vect_t add_r (const vect_t a, const vect_t b) const`
- `INLINE vect_t addin_r (vect_t &a, const vect_t b) const`
- `INLINE vect_t sub (vect_t &c, const vect_t a, const vect_t b)`
- `INLINE vect_t sub (const vect_t a, const vect_t b)`
- `INLINE vect_t subin (vect_t &a, const vect_t b) const`
- `INLINE vect_t sub_r (vect_t &c, const vect_t a, const vect_t b) const`
- `INLINE vect_t sub_r (const vect_t a, const vect_t b) const`
- `INLINE vect_t subin_r (vect_t &a, const vect_t b) const`
- `INLINE vect_t zero (vect_t &x) const`
- `INLINE vect_t zero () const`
- `INLINE vect_t mod (vect_t &c) const`
- `INLINE vect_t mul (vect_t &c, const vect_t a, const vect_t b) const`
- `INLINE vect_t mul (const vect_t a, const vect_t b) const`
- `INLINE vect_t mulin (vect_t &a, const vect_t b) const`
- `INLINE vect_t mul_r (vect_t &c, const vect_t a, const vect_t b) const`
- `INLINE vect_t mul_r (const vect_t a, const vect_t b) const`
- `INLINE vect_t axpy (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const`
- `INLINE vect_t axpy (const vect_t c, const vect_t a, const vect_t b) const`
- `INLINE vect_t axpyin (vect_t &c, const vect_t a, const vect_t b) const`
- `INLINE vect_t axpy_r (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const`
- `INLINE vect_t axpy_r (const vect_t c, const vect_t a, const vect_t b) const`
- `INLINE vect_t axpyin_r (vect_t &c, const vect_t a, const vect_t b) const`
- `INLINE vect_t maxpy (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const`
- `INLINE vect_t maxpy (const vect_t c, const vect_t a, const vect_t b) const`
- `INLINE vect_t maxpyin (vect_t &c, const vect_t a, const vect_t b) const`

Static Public Attributes

- static const constexpr size_t `vect_size` = `simd::vect_size`
- static const constexpr size_t `alignment` = `simd::alignment`

12.87.1 Member Typedef Documentation

12.87.1.1 Field

```
template<class _Field >
using Field = _Field
```

12.87.1.2 Element

```
template<class _Field >
using Element = typename Field::Element
```

12.87.1.3 simd

```
template<class _Field >
using simd = Simd<typename _Field::Element>
```

12.87.1.4 vect_t

```
template<class _Field >
using vect_t = typename simd::vect_t
```

12.87.1.5 scalar_t

```
template<class _Field >
using scalar_t = typename simd::scalar_t
```

12.87.2 Constructor & Destructor Documentation

12.87.2.1 FieldSimd() [1/3]

```
template<class _Field >
FieldSimd (
    const Field & f ) [inline]
```

12.87.2.2 FieldSimd() [2/3]

```
template<class _Field >
FieldSimd (
    const Self & ) [default]
```

12.87.2.3 FieldSimd() [3/3]

```
template<class _Field >
FieldSimd (
    Self && ) [default]
```

12.87.3 Member Function Documentation

12.87.3.1 operator=() [1/2]

```
template<class _Field >
Self & operator= (
    const Self & ) [default]
```

12.87.3.2 operator=() [2/2]

```
template<class _Field >
Self & operator= (
    Self && ) [default]
```

12.87.3.3 init() [1/2]

```
template<class _Field >
INLINE vect_t init (
    vect_t & x,
    const vect_t a ) const [inline]
```

12.87.3.4 init() [2/2]

```
template<class _Field >
INLINE vect_t init (
    const vect_t a ) const [inline]
```

12.87.3.5 add() [1/2]

```
template<class _Field >
INLINE vect_t add (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline]
```

12.87.3.6 add() [2/2]

```
template<class _Field >
INLINE vect_t add (
    const vect_t a,
    const vect_t b ) [inline]
```

12.87.3.7 addin()

```
template<class _Field >
INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) const [inline]
```

12.87.3.8 add_r() [1/2]

```
template<class _Field >
INLINE vect_t add_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.9 add_r() [2/2]

```
template<class _Field >
INLINE vect_t add_r (
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.10 addin_r()

```
template<class _Field >
INLINE vect_t addin_r (
    vect_t & a,
    const vect_t b ) const [inline]
```

12.87.3.11 sub() [1/2]

```
template<class _Field >
INLINE vect_t sub (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline]
```

12.87.3.12 sub() [2/2]

```
template<class _Field >
INLINE vect_t sub (
    const vect_t a,
    const vect_t b ) [inline]
```

12.87.3.13 subin()

```
template<class _Field >
INLINE vect_t subin (
    vect_t & a,
    const vect_t b ) const [inline]
```


12.87.3.14 sub_r() [1/2]

```
template<class _Field >
INLINE vect_t sub_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.15 sub_r() [2/2]

```
template<class _Field >
INLINE vect_t sub_r (
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.16 subin_r()

```
template<class _Field >
INLINE vect_t subin_r (
    vect_t & a,
    const vect_t b ) const [inline]
```

12.87.3.17 zero() [1/2]

```
template<class _Field >
INLINE vect_t zero (
    vect_t & x ) const [inline]
```

12.87.3.18 zero() [2/2]

```
template<class _Field >
INLINE vect_t zero ( ) const [inline]
```

12.87.3.19 mod()

```
template<class _Field >
INLINE vect_t mod (
    vect_t & c ) const [inline]
```

12.87.3.20 mul() [1/2]

```
template<class _Field >
INLINE vect_t mul (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.21 mul() [2/2]

```
template<class _Field >
INLINE vect_t mul (
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.22 mulin()

```
template<class _Field >
INLINE vect_t mulin (
    vect_t & a,
    const vect_t b ) const [inline]
```

12.87.3.23 mul_r() [1/2]

```
template<class _Field >
INLINE vect_t mul_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.24 mul_r() [2/2]

```
template<class _Field >
INLINE vect_t mul_r (
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.25 axpy() [1/2]

```
template<class _Field >
INLINE vect_t axpy (
    vect_t & r,
    const vect_t a,
    const vect_t b,
    const vect_t c ) const [inline]
```

12.87.3.26 axpy() [2/2]

```
template<class _Field >
INLINE vect_t axpy (
    const vect_t c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.27 axpyin()

```
template<class _Field >
INLINE vect_t axpyin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.28 axpy_r() [1/2]

```
template<class _Field >
INLINE vect_t axpy_r (
    vect_t & r,
    const vect_t a,
    const vect_t b,
    const vect_t c ) const [inline]
```

12.87.3.29 axpy_r() [2/2]

```
template<class _Field >
INLINE vect_t axpy_r (
    const vect_t c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.30 axpyin_r()

```
template<class _Field >
INLINE vect_t axpyin_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.31 maxpy() [1/2]

```
template<class _Field >
INLINE vect_t maxpy (
    vect_t & r,
    const vect_t a,
    const vect_t b,
    const vect_t c ) const [inline]
```

12.87.3.32 maxpy() [2/2]

```
template<class _Field >
INLINE vect_t maxpy (
    const vect_t c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.3.33 maxpyin()

```
template<class _Field >
INLINE vect_t maxpyin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

12.87.4 Field Documentation**12.87.4.1 vect_size**

```
template<class _Field >
const constexpr size_t vect_size = simd::vect_size [static], [constexpr]
```

12.87.4.2 alignment

```
template<class _Field >
const constexpr size_t alignment = simd::alignment [static], [constexpr]
```

The documentation for this class was generated from the following file:

- [simd_modular.inl](#)

12.88 FieldTraits< Field > Struct Template Reference

FieldTrait.

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::GenericTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.88.1 Detailed Description

```
template<class Field>
struct FFLAS::FieldTraits< Field >
```

FieldTrait.

12.88.2 Member Typedef Documentation

12.88.2.1 category

```
template<class Field >
typedef FieldCategories::GenericTag category
```

12.88.3 Field Documentation

12.88.3.1 balanced

```
template<class Field >
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.89 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.89.1 Member Typedef Documentation

12.89.1.1 category

```
template<typename T >
typedef FieldCategories::UnparametricTag category
```

12.89.2 Field Documentation

12.89.2.1 balanced

```
template<typename T >
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.90 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::ModularTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.90.1 Member Typedef Documentation**12.90.1.1 category**

```
template<typename T >
typedef FieldCategories::ModularTag category
```

12.90.2 Field Documentation**12.90.2.1 balanced**

```
template<typename T >
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.91 FieldTraits< Givaro::Modular< Element > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::ModularTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.91.1 Member Typedef Documentation**12.91.1.1 category**

```
template<class Element >
typedef FieldCategories::ModularTag category
```

12.91.2 Field Documentation**12.91.2.1 balanced**

```
template<class Element >
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.92 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::ModularTag](#) category

Static Public Attributes

- static const bool [balanced](#) = true

12.92.1 Member Typedef Documentation**12.92.1.1 category**

```
template<class Element >
typedef FieldCategories::ModularTag category
```

12.92.2 Field Documentation**12.92.2.1 balanced**

```
template<class Element >
const bool balanced = true [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.93 FieldTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.93.1 Member Typedef Documentation**12.93.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

12.93.2 Field Documentation**12.93.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.94 FieldTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.94.1 Member Typedef Documentation**12.94.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

12.94.2 Field Documentation**12.94.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.95 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.95.1 Member Typedef Documentation**12.95.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

12.95.2 Field Documentation**12.95.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.96 FieldTraits< Givaro::ZRing< int16_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.96.1 Member Typedef Documentation

12.96.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

12.96.2 Field Documentation

12.96.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.97 FieldTraits< Givaro::ZRing< int32_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.97.1 Member Typedef Documentation

12.97.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

12.97.2 Field Documentation

12.97.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.98 FieldTraits< Givaro::ZRing< int64_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.98.1 Member Typedef Documentation

12.98.1.1 category

```
typedef FieldCategories::UnparametricTag category
```


12.98.2 Field Documentation

12.98.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.99 FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.99.1 Member Typedef Documentation

12.99.1.1 category

```
template<size_t K>
typedef FieldCategories::UnparametricTag category
```

12.99.2 Field Documentation

12.99.2.1 balanced

```
template<size_t K>
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.100 FieldTraits< Givaro::ZRing< uint16_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) category

Static Public Attributes

- static const bool [balanced](#) = false

12.100.1 Member Typedef Documentation

12.100.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

12.100.2 Field Documentation

12.100.2.1 `balanced`

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.101 `FieldTraits< Givaro::ZRing< uint32_t > >` Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) `category`

Static Public Attributes

- static const bool [balanced](#) = false

12.101.1 Member Typedef Documentation

12.101.1.1 `category`

```
typedef FieldCategories::UnparametricTag category
```

12.101.2 Field Documentation

12.101.2.1 `balanced`

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.102 `FieldTraits< Givaro::ZRing< uint64_t > >` Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [FieldCategories::UnparametricTag](#) `category`

Static Public Attributes

- static const bool [balanced](#) = false

12.102.1 Member Typedef Documentation

12.102.1.1 `category`

```
typedef FieldCategories::UnparametricTag category
```

12.102.2 Field Documentation

12.102.2.1 `balanced`

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.103 Fixed Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.104 FixedPrecIntTag Struct Reference

Fixed precision integers above machine precision: Givaro::reclnt.

```
#include <field-traits.h>
```

12.104.1 Detailed Description

Fixed precision integers above machine precision: Givaro::reclnt.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.105 ForStrategy1D< blocksize_t, Cut, Param > Struct Template Reference

Public Member Functions

- [ForStrategy1D](#) (const blocksize_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- [ForStrategy1D](#) (const blocksize_t b, const blocksize_t e, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- void [build](#) (const blocksize_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- blocksize_t [initialize](#) ()
- bool [isTerminated](#) () const
- blocksize_t [begin](#) () const
- blocksize_t [end](#) () const
- blocksize_t [numblocks](#) () const
- blocksize_t [blockindex](#) () const
- blocksize_t [operator++](#) ()

Protected Attributes

- blocksize_t [ibeg](#)
- blocksize_t [iend](#)
- blocksize_t [current](#)
- blocksize_t [firstBlockSize](#)
- blocksize_t [lastBlockSize](#)
- blocksize_t [changeBS](#)
- blocksize_t [numBlock](#)

12.105.1 Constructor & Destructor Documentation

12.105.1.1 ForStrategy1D() [1/2]

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
```

```
ForStrategy1D (
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

12.105.1.2 ForStrategy1D() [2/2]

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
ForStrategy1D (
    const blocksize_t b,
    const blocksize_t e,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

12.105.2 Member Function Documentation**12.105.2.1 build()**

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
void build (
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

12.105.2.2 initialize()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t initialize ( ) [inline]
```

12.105.2.3 isTerminated()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
bool isTerminated ( ) const [inline]
```

12.105.2.4 begin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t begin ( ) const [inline]
```

12.105.2.5 end()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t end ( ) const [inline]
```

12.105.2.6 numblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numblocks ( ) const [inline]
```

12.105.2.7 blockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t blockindex ( ) const [inline]
```

12.105.2.8 operator++()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t operator++ ( ) [inline]
```

12.105.3 Field Documentation

12.105.3.1 ibeg

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t ibeg [protected]
```

12.105.3.2 iend

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t iend [protected]
```

12.105.3.3 current

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t current [protected]
```

12.105.3.4 firstBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t firstBlockSize [protected]
```

12.105.3.5 lastBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastBlockSize [protected]
```

12.105.3.6 changeBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeBS [protected]
```

12.105.3.7 numBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numBlock [protected]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.106 ForStrategy2D< blocksize_t, Cut, Param > Struct Template Reference

Public Member Functions

- [ForStrategy2D](#) (const blocksize_t m, const blocksize_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- blocksize_t [initialize](#) ()
- bool [isTerminated](#) () const
- blocksize_t [ibegin](#) () const
- blocksize_t [jbegin](#) () const
- blocksize_t [iend](#) () const
- blocksize_t [jend](#) () const

- `blocksize_t operator++ ()`
- `blocksize_t rownumblocks () const`
- `blocksize_t colnumblocks () const`
- `blocksize_t blockindex () const`
- `blocksize_t rowblockindex () const`
- `blocksize_t colblockindex () const`

Protected Attributes

- `blocksize_t _ibeg`
- `blocksize_t _iend`
- `blocksize_t _jbeg`
- `blocksize_t _jend`
- `blocksize_t rowBlockSize`
- `blocksize_t colBlockSize`
- `blocksize_t current`
- `blocksize_t lastRBS`
- `blocksize_t lastCBS`
- `blocksize_t changeRBS`
- `blocksize_t changeCBS`
- `blocksize_t numRowsBlock`
- `blocksize_t numColsBlock`
- `blocksize_t BLOCKS`

Friends

- `std::ostream & operator<< (std::ostream &out, const ForStrategy2D &FS2D)`

12.106.1 Constructor & Destructor Documentation

12.106.1.1 ForStrategy2D()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
ForStrategy2D (
    const blocksize_t m,
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

12.106.2 Member Function Documentation

12.106.2.1 initialize()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t initialize ( ) [inline]
```

12.106.2.2 isTerminated()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
bool isTerminated ( ) const [inline]
```

12.106.2.3 ibegin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t ibegin ( ) const [inline]
```

12.106.2.4 jbegin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t jbegin ( ) const [inline]
```

12.106.2.5 iend()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t iend ( ) const [inline]
```

12.106.2.6 jend()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t jend ( ) const [inline]
```

12.106.2.7 operator++()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t operator++ ( ) [inline]
```

12.106.2.8 rownumblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rownumblocks ( ) const [inline]
```

12.106.2.9 colnumblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colnumblocks ( ) const [inline]
```

12.106.2.10 blockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t blockindex ( ) const [inline]
```

12.106.2.11 rowblockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rowblockindex ( ) const [inline]
```

12.106.2.12 colblockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colblockindex ( ) const [inline]
```

12.106.3 Friends And Related Symbol Documentation

12.106.3.1 operator<<

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
```

```
std::ostream & operator<< (
    std::ostream & out,
    const ForStrategy2D< blocksize_t, Cut, Param > & FS2D ) [friend]
```

12.106.4 Field Documentation

12.106.4.1 `_ibeg`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _ibeg [protected]
```

12.106.4.2 `_iend`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _iend [protected]
```

12.106.4.3 `_jbeg`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _jbeg [protected]
```

12.106.4.4 `_jend`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _jend [protected]
```

12.106.4.5 `rowBlockSize`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rowBlockSize [protected]
```

12.106.4.6 `colBlockSize`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colBlockSize [protected]
```

12.106.4.7 `current`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t current [protected]
```

12.106.4.8 `lastRBS`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastRBS [protected]
```

12.106.4.9 `lastCBS`

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastCBS [protected]
```


12.106.4.10 changeRBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeRBS [protected]
```

12.106.4.11 changeCBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeCBS [protected]
```

12.106.4.12 numRowsBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numRowsBlock [protected]
```

12.106.4.13 numColBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numColBlock [protected]
```

12.106.4.14 BLOCKS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t BLOCKS [protected]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.107 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.108 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.109 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.110 ftrmmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.111 **ftrmmLeftUpperNoTransNonUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.112 **ftrmmLeftUpperNoTransUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.113 **ftrmmLeftUpperTransNonUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.114 **ftrmmLeftUpperTransUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.115 **ftrmmRightLowerNoTransNonUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.116 **ftrmmRightLowerNoTransUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.117 **ftrmmRightLowerTransNonUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.118 **ftrmmRightLowerTransUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.119 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.120 ftrmmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.121 ftrmmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.122 ftrmmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.123 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.124 ftrsmLeftLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.125 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.126 ftrsmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.127 `ftrsmLeftUpperNoTransNonUnit< Element >` Class Template Reference

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

12.127.1 Detailed Description

`template<class Element>`

`class FFLAS::Protected::ftrsmLeftUpperNoTransNonUnit< Element >`

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

Compute the maximal dimension k , such that a unit diagonal triangular system of dimension k can be solved over Z without overflow of the underlying floating point representation.

Bibliography • Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

Parameters

F	Finite Field/Ring of the computation
-----	--------------------------------------

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.128 `ftrsmLeftUpperNoTransUnit< Element >` Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.129 `ftrsmLeftUpperTransNonUnit< Element >` Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.130 `ftrsmLeftUpperTransUnit< Element >` Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.131 `ftrsmRightLowerNoTransNonUnit< Element >` Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.132 `ftrsmRightLowerNoTransUnit< Element >` Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.133 ftrsmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.134 ftrsmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.135 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.136 ftrsmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.137 ftrsmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.138 ftrsmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas_level3.inl](#)

12.139 GenericTag Struct Reference

default is generic

```
#include <field-traits.h>
```

12.139.1 Detailed Description

default is generic

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.140 GenericTag Struct Reference

generic ring.

```
#include <field-traits.h>
```

12.140.1 Detailed Description

generic ring.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.141 Grain Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.142 has_minus_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool [value](#) = type::value

12.142.1 Field Documentation

12.142.1.1 value

```
template<typename C >
```

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.143 has_minus_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool [value](#) = type::value

12.143.1 Field Documentation

12.143.1.1 value

```
template<typename C >
```

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.144 has_mul_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool [value](#) = type::value

12.144.1 Field Documentation**12.144.1.1 value**

```
template<typename C >
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.145 has_mul_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool [value](#) = type::value

12.145.1 Field Documentation**12.145.1.1 value**

```
template<typename C >
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.146 has_operation< T > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool [value](#)

12.146.1 Field Documentation**12.146.1.1 value**

```
template<class T >
constexpr bool value [static], [constexpr]
```

Initial value:

```
= (has_plus<T>::value && has_minus<T>::value && has_equal<T>::value &&
    has_plus_eq<T>::value && has_minus_eq<T>::value && has_mul<T>::value
    && has_mul_eq<T>::value)
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.147 has_plus_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool [value](#) = type::value

12.147.1 Field Documentation

12.147.1.1 value

```
template<typename C >
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.148 has_plus_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool [value](#) = type::value

12.148.1 Field Documentation

12.148.1.1 value

```
template<typename C >
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.149 HelperFlag Struct Reference

```
#include <fflas_sparse.h>
```

Static Public Attributes

- static constexpr [uint64_t none](#) = 0_ui64
- static constexpr [uint64_t coo](#) = 1_ui64
- static constexpr [uint64_t csr](#) = 1_ui64 << 1
- static constexpr [uint64_t ell](#) = 1_ui64 << 2
- static constexpr [uint64_t aut](#) = 1_ui64 << 32
- static constexpr [uint64_t pm1](#) = 1_ui64 << 33

12.149.1 Field Documentation

12.149.1.1 none

```
constexpr uint64\_t none = 0_ui64 [static], [constexpr]
```

12.149.1.2 coo

```
constexpr uint64\_t coo = 1_ui64 [static], [constexpr]
```

12.149.1.3 csr

```
constexpr uint64\_t csr = 1_ui64 << 1 [static], [constexpr]
```

12.149.1.4 ell

```
constexpr uint64\_t ell = 1_ui64 << 2 [static], [constexpr]
```


12.149.1.5 aut

```
constexpr uint64_t aut = 1_ui64 << 32 [static], [constexpr]
```

12.149.1.6 pm1

```
constexpr uint64_t pm1 = 1_ui64 << 33 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [fflas_sparse.h](#)

12.150 HelperMod< Field, ElementTraits > Struct Template Reference

The documentation for this struct was generated from the following file:

- [fflas_freduce.inl](#)

12.151 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference**Public Member Functions**

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)

Data Fields

- [Field::Element](#) p
- double [invp](#)
- double [min](#)
- double [max](#)
- [int64_t](#) pow50rem

12.151.1 Constructor & Destructor Documentation**12.151.1.1 HelperMod() [1/2]**

```
template<class Field >
HelperMod ( ) [inline]
```

12.151.1.2 HelperMod() [2/2]

```
template<class Field >
HelperMod (
    const Field & F ) [inline]
```

12.151.2 Field Documentation**12.151.2.1 p**

```
template<class Field >
Field::Element p
```

12.151.2.2 invp

```
template<class Field >
double invp
```

12.151.2.3 min

```
template<class Field >
double min
```

12.151.2.4 max

```
template<class Field >
double max
```

12.151.2.5 pow50rem

```
template<class Field >
int64_t pow50rem
```

The documentation for this struct was generated from the following file:

- [fflas_freduce.inl](#)

12.152 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference

Public Member Functions

- [HelperMod \(\)](#)
- [HelperMod \(const Field &F\)](#)

Data Fields

- [Field::Element p](#)

12.152.1 Constructor & Destructor Documentation**12.152.1.1 HelperMod() [1/2]**

```
template<class Field >
HelperMod ( ) [inline]
```

12.152.1.2 HelperMod() [2/2]

```
template<class Field >
HelperMod (
    const Field & F ) [inline]
```

12.152.2 Field Documentation**12.152.2.1 p**

```
template<class Field >
Field::Element p
```

The documentation for this struct was generated from the following file:

- [fflas_freduce.inl](#)

12.153 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference

Public Member Functions

- [HelperMod \(\)](#)

- [HelperMod](#) (const [Field](#) &F)

Data Fields

- [Field::Element](#) p

12.153.1 Constructor & Destructor Documentation

12.153.1.1 HelperMod() [1/2]

```
template<class Field >
HelperMod ( ) [inline]
```

12.153.1.2 HelperMod() [2/2]

```
template<class Field >
HelperMod (
    const Field & F ) [inline]
```

12.153.2 Field Documentation

12.153.2.1 p

```
template<class Field >
Field::Element p
```

The documentation for this struct was generated from the following file:

- [fflas_freduce.inl](#)

12.154 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference

Public Member Functions

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)

Data Fields

- [Field::Element](#) p
- [Field::Element](#) invp
- [Field::Element](#) min
- [Field::Element](#) max

12.154.1 Constructor & Destructor Documentation

12.154.1.1 HelperMod() [1/2]

```
template<class Field >
HelperMod ( ) [inline]
```

12.154.1.2 HelperMod() [2/2]

```
template<class Field >
HelperMod (
    const Field & F ) [inline]
```

12.154.2 Field Documentation

12.154.2.1 p

```
template<class Field >
Field::Element p
```

12.154.2.2 invp

```
template<class Field >
Field::Element invp
```

12.154.2.3 min

```
template<class Field >
Field::Element min
```

12.154.2.4 max

```
template<class Field >
Field::Element max
```

The documentation for this struct was generated from the following file:

- [fflas_freduce.inl](#)

12.155 Hybrid Struct Reference

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.156 Info Struct Reference

Public Member Functions

- [Info](#) ([uint64_t](#) it, [uint64_t](#) s, [uint64_t](#) p)
- [Info](#) ()=default
- [Info](#) (const [Info](#) &)=default
- [Info](#) ([Info](#) &&)=default
- [Info](#) & [operator=](#) (const [Info](#) &)=default
- [Info](#) & [operator=](#) ([Info](#) &&)=default

Data Fields

- [uint64_t](#) size = 0
- [uint64_t](#) perm = 0
- [uint64_t](#) begin = 0

12.156.1 Constructor & Destructor Documentation

12.156.1.1 Info() [1/4]

```
Info (
    uint64_t it,
    uint64_t s,
    uint64_t p ) [inline]
```

12.156.1.2 Info() [2/4]

```
Info ( ) [default]
```

12.156.1.3 Info() [3/4]

```
Info (
    const Info & ) [default]
```

12.156.1.4 Info() [4/4]

```
Info (
    Info && ) [default]
```

12.156.2 Member Function Documentation**12.156.2.1 operator=()** [1/2]

```
Info & operator= (
    const Info & ) [default]
```

12.156.2.2 operator=() [2/2]

```
Info & operator= (
    Info && ) [default]
```

12.156.3 Field Documentation**12.156.3.1 size**

```
uint64_t size = 0
```

12.156.3.2 perm

```
uint64_t perm = 0
```

12.156.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- [csr_hyb_utils.inl](#)

12.157 Info Struct Reference**Public Member Functions**

- [Info](#) (uint64_t it, uint64_t s, uint64_t p)
- [Info](#) ()=default
- [Info](#) (const [Info](#) &)=default
- [Info](#) ([Info](#) &&)=default
- [Info](#) & [operator=](#) (const [Info](#) &)=default
- [Info](#) & [operator=](#) ([Info](#) &&)=default

Data Fields

- [uint64_t](#) [size](#) = 0
- [uint64_t](#) [perm](#) = 0
- [uint64_t](#) [begin](#) = 0

12.157.1 Constructor & Destructor Documentation

12.157.1.1 Info() [1/4]

```
Info (
    uint64_t it,
    uint64_t s,
    uint64_t p ) [inline]
```

12.157.1.2 Info() [2/4]

```
Info ( ) [default]
```

12.157.1.3 Info() [3/4]

```
Info (
    const Info & ) [default]
```

12.157.1.4 Info() [4/4]

```
Info (
    Info && ) [default]
```

12.157.2 Member Function Documentation

12.157.2.1 operator=() [1/2]

```
Info & operator= (
    const Info & ) [default]
```

12.157.2.2 operator=() [2/2]

```
Info & operator= (
    Info && ) [default]
```

12.157.3 Field Documentation

12.157.3.1 size

```
uint64_t size = 0
```

12.157.3.2 perm

```
uint64_t perm = 0
```

12.157.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- [sell_utils.inl](#)

12.158 is_simd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using [type](#) = std::integral_constant< bool, false >

Static Public Attributes

- static const constexpr bool [value](#) = false

12.158.1 Member Typedef Documentation**12.158.1.1 type**

```
template<class T >
using type = std::integral_constant<bool, false>
```

12.158.2 Field Documentation**12.158.2.1 value**

```
template<class T >
const constexpr bool value = false [static], [constexpr]
```

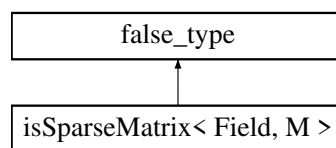
The documentation for this struct was generated from the following file:

- [fflas_simd.h](#)

12.159 isSparseMatrix< Field, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, M >:



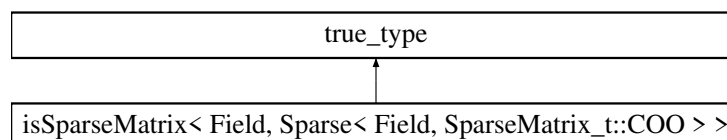
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.160 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >:



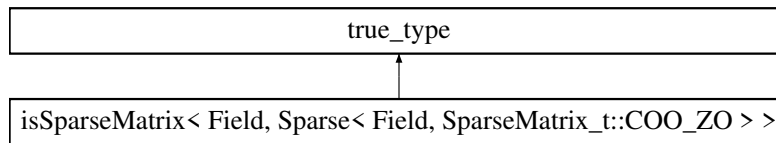
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.161 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >`:



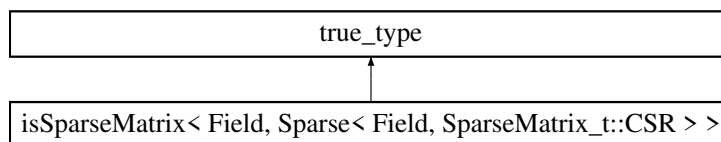
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.162 `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >`:



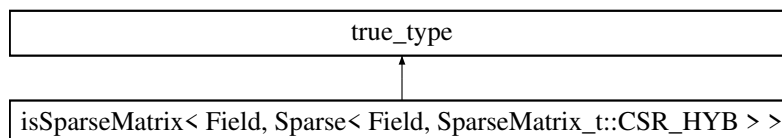
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.163 `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >`:



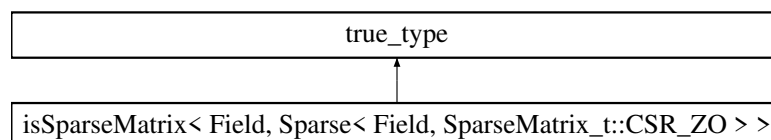
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.164 `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >`:



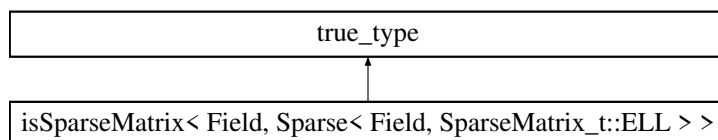
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.165 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >:



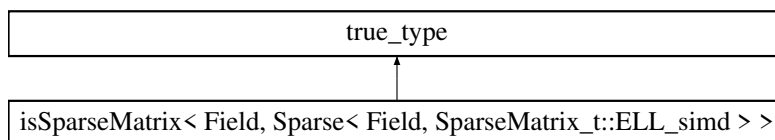
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.166 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >:



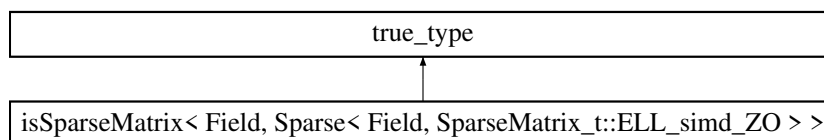
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.167 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >:



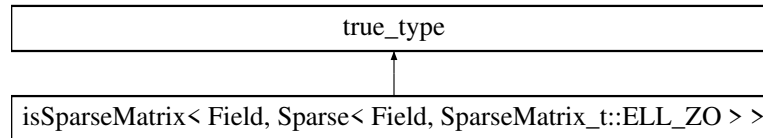
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.168 **isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >:



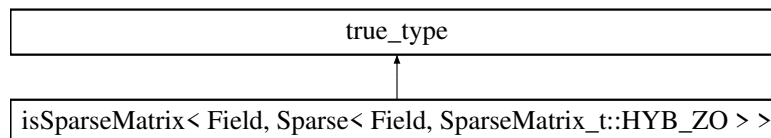
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.169 **isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >:



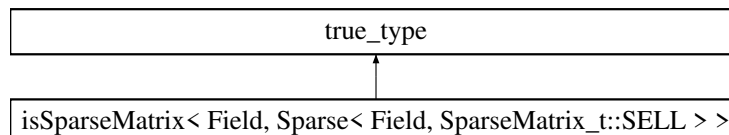
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.170 **isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >:



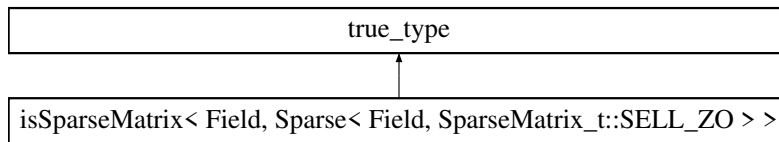
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.171 **isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >:



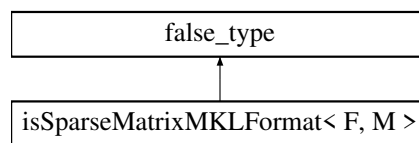
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.172 isSparseMatrixMKLFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixMKLFormat< F, M >:



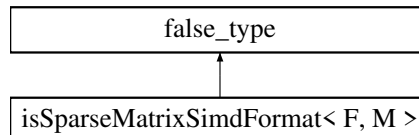
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.173 isSparseMatrixSimdFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixSimdFormat< F, M >:



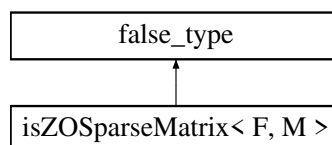
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.174 isZOSparseMatrix< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< F, M >:



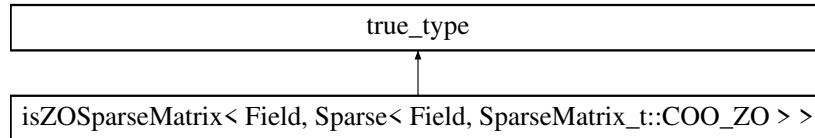
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.175 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >:



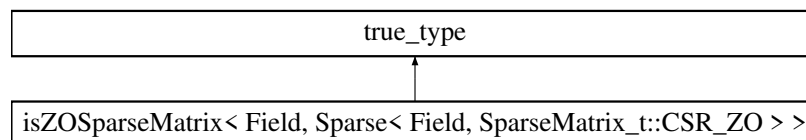
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.176 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >:



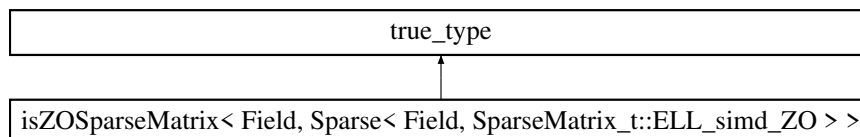
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.177 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >:



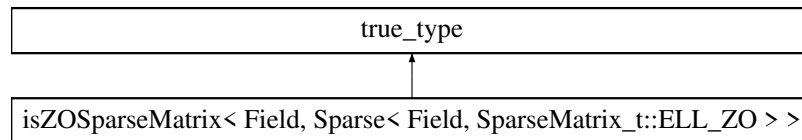
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.178 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >:



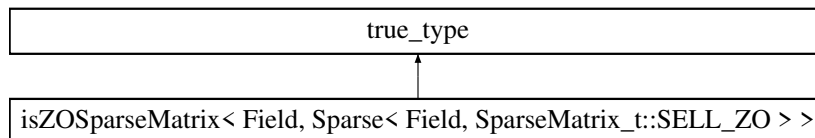
The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.179 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >:



The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.180 Iterative Struct Reference

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.181 LazyTag Struct Reference

Performs field operations with delayed mod only when necessary. Result may not be reduced.

```
#include <field-traits.h>
```

12.181.1 Detailed Description

Performs field operations with delayed mod only when necessary. Result may not be reduced.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.182 limits< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.183 limits< char > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef char [T](#)

Static Public Member Functions

- static constexpr char [max](#) () noexcept
- static constexpr char [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.183.1 Member Typedef Documentation**12.183.1.1 T**

typedef char [T](#)

12.183.2 Member Function Documentation**12.183.2.1 max()**

static constexpr char max () [inline], [static], [constexpr], [noexcept]

12.183.2.2 min()

static constexpr char min () [inline], [static], [constexpr], [noexcept]

12.183.2.3 digits()

static constexpr [int32_t](#) digits () [inline], [static], [constexpr], [noexcept]

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.184 limits< double > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef double [T](#)

Static Public Member Functions

- static constexpr [int64_t](#) [max](#) () noexcept
- static constexpr [int64_t](#) [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.184.1 Member Typedef Documentation**12.184.1.1 T**

typedef double [T](#)

12.184.2 Member Function Documentation**12.184.2.1 max()**

static constexpr [int64_t](#) max () [inline], [static], [constexpr], [noexcept]

12.184.2.2 min()

```
static constexpr int64_t min ( ) [inline], [static], [constexpr], [noexcept]
```

12.184.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.185 limits< float > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef float [T](#)

Static Public Member Functions

- static constexpr [int32_t max](#) () noexcept
- static constexpr [int32_t min](#) () noexcept
- static constexpr [int32_t digits](#) () noexcept

12.185.1 Member Typedef Documentation**12.185.1.1 T**

```
typedef float T
```

12.185.2 Member Function Documentation**12.185.2.1 max()**

```
static constexpr int32_t max ( ) [inline], [static], [constexpr], [noexcept]
```

12.185.2.2 min()

```
static constexpr int32_t min ( ) [inline], [static], [constexpr], [noexcept]
```

12.185.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.186 limits< Givaro::Integer > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef Givaro::Integer [T](#)

Static Public Member Functions

- static constexpr int [max](#) () noexcept
- static constexpr int [min](#) () noexcept

12.186.1 Member Typedef Documentation

12.186.1.1 T

```
typedef Givaro::Integer T
```

12.186.2 Member Function Documentation

12.186.2.1 max()

```
static constexpr int max ( ) [inline], [static], [constexpr], [noexcept]
```

12.186.2.2 min()

```
static constexpr int min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.187 limits< int > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef int [T](#)

Static Public Member Functions

- static constexpr int [max](#) () noexcept
- static constexpr int [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.187.1 Member Typedef Documentation

12.187.1.1 T

```
typedef int T
```

12.187.2 Member Function Documentation

12.187.2.1 max()

```
static constexpr int max ( ) [inline], [static], [constexpr], [noexcept]
```

12.187.2.2 min()

```
static constexpr int min ( ) [inline], [static], [constexpr], [noexcept]
```

12.187.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.188 limits< long > Struct Reference

```
#include <flimits.h>
```


Public Types

- typedef long [T](#)

Static Public Member Functions

- static constexpr long [max](#) () noexcept
- static constexpr long [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.188.1 Member Typedef Documentation

12.188.1.1 T

typedef long [T](#)

12.188.2 Member Function Documentation

12.188.2.1 max()

static constexpr long max () [inline], [static], [constexpr], [noexcept]

12.188.2.2 min()

static constexpr long min () [inline], [static], [constexpr], [noexcept]

12.188.2.3 digits()

static constexpr [int32_t](#) digits () [inline], [static], [constexpr], [noexcept]

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.189 limits< long long > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef long long [T](#)

Static Public Member Functions

- static constexpr long long [max](#) () noexcept
- static constexpr long long [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.189.1 Member Typedef Documentation

12.189.1.1 T

typedef long long [T](#)

12.189.2 Member Function Documentation

12.189.2.1 max()

static constexpr long long max () [inline], [static], [constexpr], [noexcept]

12.189.2.2 min()

```
static constexpr long long min ( ) [inline], [static], [constexpr], [noexcept]
```

12.189.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.190 limits< RecInt::rint< K > > Struct Template Reference

```
#include <flimits.h>
```

Public Types

- typedef [RecInt::ruint< K > T](#)

Static Public Member Functions

- static constexpr [RecInt::rint< K > max](#) () noexcept
- static constexpr [RecInt::rint< K > min](#) () noexcept

12.190.1 Member Typedef Documentation**12.190.1.1 T**

```
template<size_t K>
typedef RecInt::ruint<K> T
```

12.190.2 Member Function Documentation**12.190.2.1 max()**

```
template<size_t K>
static constexpr RecInt::rint< K > max ( ) [inline], [static], [constexpr], [noexcept]
```

12.190.2.2 min()

```
template<size_t K>
static constexpr RecInt::rint< K > min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.191 limits< RecInt::ruint< K > > Struct Template Reference

```
#include <flimits.h>
```

Public Types

- typedef [RecInt::ruint< K > T](#)

Static Public Member Functions

- static constexpr [RecInt::ruint< K > max](#) () noexcept
- static constexpr [RecInt::ruint< K > min](#) () noexcept

12.191.1 Member Typedef Documentation

12.191.1.1 T

```
template<size_t K>
typedef RecInt::ruint<K> T
```

12.191.2 Member Function Documentation

12.191.2.1 max()

```
template<size_t K>
static constexpr RecInt::ruint< K > max ( ) [inline], [static], [constexpr], [noexcept]
```

12.191.2.2 min()

```
template<size_t K>
static constexpr RecInt::ruint< K > min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.192 limits< short int > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef short int [T](#)

Static Public Member Functions

- static constexpr short int [max](#) () noexcept
- static constexpr short int [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.192.1 Member Typedef Documentation

12.192.1.1 T

```
typedef short int T
```

12.192.2 Member Function Documentation

12.192.2.1 max()

```
static constexpr short int max ( ) [inline], [static], [constexpr], [noexcept]
```

12.192.2.2 min()

```
static constexpr short int min ( ) [inline], [static], [constexpr], [noexcept]
```

12.192.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.193 limits< signed char > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef signed char [T](#)

Static Public Member Functions

- static constexpr signed char [max](#) () noexcept
- static constexpr signed char [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.193.1 Member Typedef Documentation

12.193.1.1 T

```
typedef signed char T
```

12.193.2 Member Function Documentation

12.193.2.1 max()

```
static constexpr signed char max ( ) [inline], [static], [constexpr], [noexcept]
```

12.193.2.2 min()

```
static constexpr signed char min ( ) [inline], [static], [constexpr], [noexcept]
```

12.193.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.194 limits< unsigned char > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef unsigned char [T](#)

Static Public Member Functions

- static constexpr unsigned char [max](#) () noexcept
- static constexpr unsigned char [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.194.1 Member Typedef Documentation

12.194.1.1 T

```
typedef unsigned char T
```

12.194.2 Member Function Documentation

12.194.2.1 max()

```
static constexpr unsigned char max ( ) [inline], [static], [constexpr], [noexcept]
```

12.194.2.2 min()

```
static constexpr unsigned char min ( ) [inline], [static], [constexpr], [noexcept]
```

12.194.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.195 limits< unsigned int > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef unsigned int [T](#)

Static Public Member Functions

- static constexpr unsigned int [max](#) () noexcept
- static constexpr unsigned int [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.195.1 Member Typedef Documentation

12.195.1.1 T

```
typedef unsigned int T
```

12.195.2 Member Function Documentation

12.195.2.1 max()

```
static constexpr unsigned int max ( ) [inline], [static], [constexpr], [noexcept]
```

12.195.2.2 min()

```
static constexpr unsigned int min ( ) [inline], [static], [constexpr], [noexcept]
```

12.195.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.196 limits< unsigned long > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef unsigned long [T](#)

Static Public Member Functions

- static constexpr unsigned long [max](#) () noexcept
- static constexpr unsigned long [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.196.1 Member Typedef Documentation

12.196.1.1 T

```
typedef unsigned long T
```

12.196.2 Member Function Documentation

12.196.2.1 max()

```
static constexpr unsigned long max ( ) [inline], [static], [constexpr], [noexcept]
```

12.196.2.2 min()

```
static constexpr unsigned long min ( ) [inline], [static], [constexpr], [noexcept]
```

12.196.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.197 limits< unsigned long long > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef unsigned long long [T](#)

Static Public Member Functions

- static constexpr unsigned long long [max](#) () noexcept
- static constexpr unsigned long long [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.197.1 Member Typedef Documentation

12.197.1.1 T

```
typedef unsigned long long T
```

12.197.2 Member Function Documentation

12.197.2.1 max()

```
static constexpr unsigned long long max ( ) [inline], [static], [constexpr], [noexcept]
```

12.197.2.2 min()

```
static constexpr unsigned long long min ( ) [inline], [static], [constexpr], [noexcept]
```

12.197.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.198 limits< unsigned short int > Struct Reference

```
#include <flimits.h>
```

Public Types

- typedef unsigned short int [T](#)

Static Public Member Functions

- static constexpr unsigned short int [max](#) () noexcept
- static constexpr unsigned short int [min](#) () noexcept
- static constexpr [int32_t](#) [digits](#) () noexcept

12.198.1 Member Typedef Documentation

12.198.1.1 T

```
typedef unsigned short int T
```

12.198.2 Member Function Documentation

12.198.2.1 max()

```
static constexpr unsigned short int max ( ) [inline], [static], [constexpr], [noexcept]
```

12.198.2.2 min()

```
static constexpr unsigned short int min ( ) [inline], [static], [constexpr], [noexcept]
```

12.198.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

12.199 MachineFloatTag Struct Reference

float or double

```
#include <field-traits.h>
```

12.199.1 Detailed Description

float or double

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.200 MachineIntTag Struct Reference

short, int, long, long long, and unsigned variants

```
#include <field-traits.h>
```

12.200.1 Detailed Description

short, int, long, long long, and unsigned variants

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.201 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference

Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeTrait](#), [ParSeqTrait](#) > [Self_t](#)
- typedef [associatedDelayedField](#)< constField >::type [DelayedField_t](#)
- typedef [associatedDelayedField](#)< constField >::field [DelayedField](#)
- typedef [DelayedField::Element](#) [DFElt](#)

Public Member Functions

- void [initC](#) ()
- void [initA](#) ()
- void [initB](#) ()
- void [initOut](#) ()
- size_t [MaxDelayedDim](#) ([DFElt](#) beta)
- bool [Aunfit](#) ()
- bool [Bunfit](#) ()
- void [setOutBounds](#) (const size_t k, const [DFElt](#) alpha, const [DFElt](#) beta)
- bool [checkA](#) (const [Field](#) &F, const [FFLAS::FFLAS_TRANSPOSE](#) ta, const size_t M, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda)
- bool [checkB](#) (const [Field](#) &F, const [FFLAS::FFLAS_TRANSPOSE](#) tb, const size_t M, const size_t N, typename [Field::ConstElement_ptr](#) B, const size_t ldb)
- bool [checkOut](#) (const [Field](#) &F, const size_t M, const size_t N, typename [Field::ConstElement_ptr](#) A, const size_t lda)
- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size_t m, size_t k, size_t n, [ParSeqTrait](#) _PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) _PS=[ParSeqTrait](#)())
- template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)
- [MMHelper](#) (const [Field](#) &F, int w, [DFElt](#) _Amin, [DFElt](#) _Amax, [DFElt](#) _Bmin, [DFElt](#) _Bmax, [DFElt](#) _Cmin, [DFElt](#) _Cmax, [ParSeqTrait](#) _PS=[ParSeqTrait](#)())

Data Fields

- int [recLevel](#)
- [DFElt](#) [FieldMin](#)
- [DFElt](#) [FieldMax](#)
- [DFElt](#) [Amin](#)
- [DFElt](#) [Amax](#)
- [DFElt](#) [Bmin](#)
- [DFElt](#) [Bmax](#)
- [DFElt](#) [Cmin](#)
- [DFElt](#) [Cmax](#)
- [DFElt](#) [Outmin](#)
- [DFElt](#) [Outmax](#)
- [DFElt](#) [MaxStorableValue](#)
- const [DelayedField_t](#) [delayedField](#)
- [ParSeqTrait](#) [parseq](#)

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self_t](#) &M)

12.201.1 Member Typedef Documentation**12.201.1.1 Self_t**

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
typedef MMHelper<Field,AlgoTrait,ModeTrait,ParSeqTrait> Self\_t
```

12.201.1.2 DelayedField_t

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
typedef associatedDelayedField<constField>::type DelayedField\_t
```

12.201.1.3 DelayedField

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
typedef associatedDelayedField<constField>::field DelayedField
```

12.201.1.4 DFElt

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
typedef DelayedField::Element DFElt
```

12.201.2 Constructor & Destructor Documentation**12.201.2.1 MMHelper() [1/5]**

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
MMHelper ( ) [inline]
```

12.201.2.2 MMHelper() [2/5]

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

12.201.2.3 MMHelper() [3/5]

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

12.201.2.4 MMHelper() [4/5]

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

12.201.2.5 MMHelper() [5/5]

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
MMHelper (
    const Field & F,
    int w,
    DFelt _Amin,
    DFelt _Amax,
    DFelt _Bmin,
    DFelt _Bmax,
    DFelt _Cmin,
    DFelt _Cmax,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

12.201.3 Member Function Documentation**12.201.3.1 initC()**

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
void initC ( ) [inline]
```

12.201.3.2 initA()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
void initA ( ) [inline]
```

12.201.3.3 initB()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
void initB ( ) [inline]
```

12.201.3.4 initOut()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
void initOut ( ) [inline]
```

12.201.3.5 MaxDelayedDim()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
size_t MaxDelayedDim (
    DFelt beta ) [inline]
```

12.201.3.6 Aunfit()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
bool Aunfit ( ) [inline]
```

12.201.3.7 Bunfit()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
bool Bunfit ( ) [inline]
```

12.201.3.8 setOutBounds()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
void setOutBounds (
    const size_t k,
    const DFelt alpha,
    const DFelt beta ) [inline]
```

12.201.3.9 checkA()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
bool checkA (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

12.201.3.10 checkB()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
bool checkB (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

12.201.3.11 checkOut()

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
bool checkOut (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

12.201.4 Friends And Related Symbol Documentation**12.201.4.1 operator<<**

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

12.201.5 Field Documentation**12.201.5.1 recLevel**

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
int recLevel
```

12.201.5.2 FieldMin

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
DFElt FieldMin
```

12.201.5.3 FieldMax

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >
DFElt FieldMax
```

12.201.5.4 Amin

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt Amin
```

12.201.5.5 Amax

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt Amax
```

12.201.5.6 Bmin

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt Bmin
```

12.201.5.7 Bmax

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt Bmax
```

12.201.5.8 Cmin

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt Cmin
```

12.201.5.9 Cmax

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt Cmax
```

12.201.5.10 Outmin

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt Outmin
```

12.201.5.11 Outmax

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt Outmax
```

12.201.5.12 MaxStorableValue

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
DFelt MaxStorableValue
```

12.201.5.13 delayedField

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
const DelayedField_t delayedField
```

12.201.5.14 parseq

```
template<class Field , typename AlgoTrait , typename ModeTrait , typename ParSeqTrait >  
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.202 MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

Public Types

- typedef [MMHelper](#)< [FFPACK::RNSInteger](#)< E >, [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self_t](#)

Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (Givaro::Integer Amax, Givaro::Integer Bmax)
- [MMHelper](#) (const [FFPACK::RNSInteger](#)< E > &F, size_t m, size_t n, size_t k, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- [MMHelper](#) (const [FFPACK::RNSInteger](#)< E > &F, int wino, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- template<class F2, class A2, class M2, class PS2 >
[MMHelper](#) ([MMHelper](#)< F2, A2, M2, PS2 > H2)
- void [setNorm](#) (Givaro::Integer p)

Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self_t](#) &M)

12.202.1 Member Typedef Documentation

12.202.1.1 Self_t

```
template<typename E, typename AlgoTrait, typename ParSeqTrait >
typedef MMHelper<FFPACK::RNSInteger<E>, AlgoTrait,ModeCategories::DefaultTag, ParSeqTrait>
Self\_t
```

12.202.2 Constructor & Destructor Documentation

12.202.2.1 MMHelper() [1/5]

```
template<typename E, typename AlgoTrait, typename ParSeqTrait >
MMHelper ( ) [inline]
```

12.202.2.2 MMHelper() [2/5]

```
template<typename E, typename AlgoTrait, typename ParSeqTrait >
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

12.202.2.3 MMHelper() [3/5]

```
template<typename E, typename AlgoTrait, typename ParSeqTrait >
MMHelper (
    const FFPACK::RNSInteger< E > & F,
    size_t m,
    size_t n,
```

```
size_t k,
ParSeqTrait PS = ParSeqTrait() ) [inline]
```

12.202.2.4 MMHelper() [4/5]

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    const FFPACK::RNSInteger< E > & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

12.202.2.5 MMHelper() [5/5]

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
template<class F2 , class A2 , class M2 , class PS2 >
MMHelper (
    MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

12.202.3 Member Function Documentation

12.202.3.1 setNorm()

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
void setNorm (
    Givaro::Integer p ) [inline]
```

12.202.4 Friends And Related Symbol Documentation

12.202.4.1 operator<<

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

12.202.5 Field Documentation

12.202.5.1 normA

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
Givaro::Integer normA
```

12.202.5.2 normB

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
Givaro::Integer normB
```

12.202.5.3 recLevel

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
int recLevel
```

12.202.5.4 parseq

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fgemm_classical_mp.ini](#)

12.203 MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

Public Types

- typedef [MMHelper](#)< [FFPACK::RNSIntegerMod](#)< E >, [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self_t](#)

Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (Givaro::Integer Amax, Givaro::Integer Bmax)
- [MMHelper](#) (const [FFPACK::RNSIntegerMod](#)< E > &F, size_t m, size_t n, size_t k, [ParSeqTrait](#) PS=[ParSeqTrait](#)()
- [MMHelper](#) (const [FFPACK::RNSIntegerMod](#)< E > &F, int wino, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- template<class F2, typename AlgoT2, typename FT2, typename PS2 >
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)
- void [setNorm](#) (Givaro::Integer p)

Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self_t](#) &M)

12.203.1 Member Typedef Documentation

12.203.1.1 Self_t

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
typedef MMHelper<FFPACK::RNSIntegerMod<E>, AlgoTrait,ModeCategories::DefaultTag, ParSeqTrait>
Self\_t
```

12.203.2 Constructor & Destructor Documentation

12.203.2.1 MMHelper() [1/5]

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
MMHelper ( ) [inline]
```

12.203.2.2 MMHelper() [2/5]

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

12.203.2.3 MMHelper() [3/5]

```
template<typename E , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    const FFPACK::RNSIntegerMod< E > & F,
    size_t m,
```

```

    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]

```

12.203.2.4 MMHelper() [4/5]

```

template<typename E , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    const FFPACK::RNSIntegerMod< E > & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]

```

12.203.2.5 MMHelper() [5/5]

```

template<typename E , typename AlgoTrait , typename ParSeqTrait >
template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]

```

12.203.3 Member Function Documentation

12.203.3.1 setNorm()

```

template<typename E , typename AlgoTrait , typename ParSeqTrait >
void setNorm (
    Givaro::Integer p ) [inline]

```

12.203.4 Friends And Related Symbol Documentation

12.203.4.1 operator<<

```

template<typename E , typename AlgoTrait , typename ParSeqTrait >
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]

```

12.203.5 Field Documentation

12.203.5.1 normA

```

template<typename E , typename AlgoTrait , typename ParSeqTrait >
Givaro::Integer normA

```

12.203.5.2 normB

```

template<typename E , typename AlgoTrait , typename ParSeqTrait >
Givaro::Integer normB

```

12.203.5.3 recLevel

```

template<typename E , typename AlgoTrait , typename ParSeqTrait >
int recLevel

```

12.203.5.4 parseq

```

template<typename E , typename AlgoTrait , typename ParSeqTrait >
ParSeqTrait parseq

```

The documentation for this struct was generated from the following file:

- [fgemm_classical_mp.ini](#)

12.204 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > Struct Template Reference

Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< Dest >, [ParSeqTrait](#) > [Self_t](#)

Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size_t m, size_t k, size_t n, [ParSeqTrait](#) _PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) _PS=[ParSeqTrait](#)())
- template<class F2, typename AlgoT2, typename FT2, typename PS2 >
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)

Data Fields

- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

Friends

- std::ostream & [operator](#)<< (std::ostream &out, const [Self_t](#) &M)

12.204.1 Member Typedef Documentation

12.204.1.1 Self_t

```
template<class Field , typename AlgoTrait , typename Dest , typename ParSeqTrait >
typedef MMHelper<Field,AlgoTrait, ModeCategories::ConvertTo<Dest>,ParSeqTrait> Self\_t
```

12.204.2 Constructor & Destructor Documentation

12.204.2.1 MMHelper() [1/4]

```
template<class Field , typename AlgoTrait , typename Dest , typename ParSeqTrait >
MMHelper ( ) [inline]
```

12.204.2.2 MMHelper() [2/4]

```
template<class Field , typename AlgoTrait , typename Dest , typename ParSeqTrait >
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

12.204.2.3 MMHelper() [3/4]

```
template<class Field , typename AlgoTrait , typename Dest , typename ParSeqTrait >
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

12.204.2.4 MMHelper() [4/4]

```
template<class Field , typename AlgoTrait , typename Dest , typename ParSeqTrait >
template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

12.204.3 Friends And Related Symbol Documentation

12.204.3.1 operator<<

```
template<class Field , typename AlgoTrait , typename Dest , typename ParSeqTrait >
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

12.204.4 Field Documentation

12.204.4.1 recLevel

```
template<class Field , typename AlgoTrait , typename Dest , typename ParSeqTrait >
int recLevel
```

12.204.4.2 parseq

```
template<class Field , typename AlgoTrait , typename Dest , typename ParSeqTrait >
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.205 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Struct Template Reference

Public Types

- typedef [MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Self_t](#)

Public Member Functions

- [MMHelper](#) ()
- template<class F2 , class A2 , class M2 , class PS2 > [MMHelper](#) ([MMHelper](#)< F2, A2, M2, PS2 > H2)
- [MMHelper](#) (Givaro::Integer Amax, Givaro::Integer Bmax)
- [MMHelper](#) (const [Field](#) &F, size_t m, size_t n, size_t k, ParSeqTrait PS=ParSeqTrait())
- [MMHelper](#) (const [Field](#) &F, int wino, ParSeqTrait PS=ParSeqTrait())
- void [setNorm](#) (Givaro::Integer p)

Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- ParSeqTrait [parseq](#)

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self_t](#) &M)

12.205.1 Member Typedef Documentation

12.205.1.1 Self_t

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
typedef MMHelper<Field, AlgoTrait,ModeCategories::ConvertTo<ElementCategories::RNSElementTag>,
ParSeqTrait> Self\_t
```

12.205.2 Constructor & Destructor Documentation

12.205.2.1 MMHelper() [1/5]

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
MMHelper ( ) [inline]
```

12.205.2.2 MMHelper() [2/5]

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
template<class F2 , class A2 , class M2 , class PS2 >
MMHelper (
    MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

12.205.2.3 MMHelper() [3/5]

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

12.205.2.4 MMHelper() [4/5]

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    const Field & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

12.205.2.5 MMHelper() [5/5]

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    const Field & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

12.205.3 Member Function Documentation

12.205.3.1 setNorm()

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
void setNorm (
    Givaro::Integer p ) [inline]
```

12.205.4 Friends And Related Symbol Documentation

12.205.4.1 operator<<

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

12.205.5 Field Documentation

12.205.5.1 normA

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
Givaro::Integer normA
```

12.205.5.2 normB

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
Givaro::Integer normB
```

12.205.5.3 recLevel

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
int recLevel
```

12.205.5.4 parseq

```
template<typename Field , typename AlgoTrait , typename ParSeqTrait >
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fgemm_classical_mp.ini](#)

12.206 MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

FGEMM Helper for Default and ConvertTo modes of operation.

Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self_t](#)

Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size_t m, size_t k, size_t n, [ParSeqTrait](#) _PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) _PS=[ParSeqTrait](#)())
- template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 > [MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)

Data Fields

- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

Friends

- std::ostream & [operator](#)<< (std::ostream &out, const [Self_t](#) &M)

12.206.1 Detailed Description

```
template<class Field, typename AlgoTrait, typename ParSeqTrait>
struct FFLAS::MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >
```

FGEMM Helper for Default and ConvertTo modes of operation.

12.206.2 Member Typedef Documentation

12.206.2.1 Self_t

```
template<class Field , typename AlgoTrait , typename ParSeqTrait >
typedef MMHelper<Field,AlgoTrait, ModeCategories::DefaultTag,ParSeqTrait> Self_t
```

12.206.3 Constructor & Destructor Documentation

12.206.3.1 MMHelper() [1/4]

```
template<class Field , typename AlgoTrait , typename ParSeqTrait >
MMHelper ( ) [inline]
```

12.206.3.2 MMHelper() [2/4]

```
template<class Field , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

12.206.3.3 MMHelper() [3/4]

```
template<class Field , typename AlgoTrait , typename ParSeqTrait >
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

12.206.3.4 MMHelper() [4/4]

```
template<class Field , typename AlgoTrait , typename ParSeqTrait >
template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

12.206.4 Friends And Related Symbol Documentation

12.206.4.1 operator<<

```
template<class Field , typename AlgoTrait , typename ParSeqTrait >
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

12.206.5 Field Documentation

12.206.5.1 recLevel

```
template<class Field , typename AlgoTrait , typename ParSeqTrait >
int recLevel
```

12.206.5.2 parseq

```
template<class Field , typename AlgoTrait , typename ParSeqTrait >
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.207 ModeTraits< Field > Struct Template Reference

[ModeTraits.](#)

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::DefaultTag](#) value

12.207.1 Detailed Description

```
template<class Field>
struct FFLAS::ModeTraits< Field >
```

[ModeTraits.](#)

12.207.2 Member Typedef Documentation

12.207.2.1 value

```
template<class Field >
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.208 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::DelayedTag](#) value

12.208.1 Member Typedef Documentation

12.208.1.1 value

```
template<typename Element , typename Compute >
typedef ModeCategories::DelayedTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.209 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value](#)

12.209.1 Member Typedef Documentation**12.209.1.1 value**

```
template<typename Compute >
```

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.210 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.210.1 Member Typedef Documentation**12.210.1.1 value**

```
template<typename Compute >
```

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.211 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.211.1 Member Typedef Documentation**12.211.1.1 value**

```
template<typename Compute >
```

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.212 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.212.1 Member Typedef Documentation**12.212.1.1 value**

```
template<typename Compute >
```

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.213 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value](#)

12.213.1 Member Typedef Documentation**12.213.1.1 value**

```
template<typename Compute , size_t K>
```

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.214 ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.214.1 Member Typedef Documentation**12.214.1.1 value**

```
template<typename Compute >
```

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.215 ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```


Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.215.1 Member Typedef Documentation**12.215.1.1 value**

```
template<typename Compute >
```

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.216 ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.216.1 Member Typedef Documentation**12.216.1.1 value**

```
template<typename Compute >
```

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.217 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::DelayedTag value](#)

12.217.1 Member Typedef Documentation**12.217.1.1 value**

```
template<typename Element >
```

```
typedef ModeCategories::DelayedTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.218 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value](#)

12.218.1 Member Typedef Documentation**12.218.1.1 value**

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.219 ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.219.1 Member Typedef Documentation**12.219.1.1 value**

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.220 ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.220.1 Member Typedef Documentation**12.220.1.1 value**

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.221 ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

12.221.1 Member Typedef Documentation

12.221.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.222 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

12.222.1 Member Typedef Documentation

12.222.1.1 value

```
template<class T >
```

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.223 ModeTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

12.223.1 Member Typedef Documentation

12.223.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.224 ModeTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

12.224.1 Member Typedef Documentation

12.224.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.225 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value](#)

12.225.1 Member Typedef Documentation

12.225.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.226 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

12.227 ModularTag Struct Reference

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`

```
#include <field-traits.h>
```

12.227.1 Detailed Description

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.228 Montgomery< T > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

12.229 need_field_characteristic< Field > Struct Template Reference

Static Public Attributes

- static constexpr bool [value](#) = false

12.229.1 Field Documentation

12.229.1.1 value

```
template<typename Field >
constexpr bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

12.230 need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference

Static Public Attributes

- static constexpr bool [value](#) = true

12.230.1 Field Documentation

12.230.1.1 value

```
template<typename Field >
constexpr bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

12.231 need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference

Static Public Attributes

- static constexpr bool [value](#) = true

12.231.1 Field Documentation

12.231.1.1 value

```
template<typename Field >
constexpr bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

12.232 NoSimd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using [vect_t](#) = T *
- using [scalar_t](#) = T

Static Public Member Functions

- static const std::string [type_string](#) ()
- template<class TT >
static constexpr bool [valid](#) (TT p)
- template<class TT >
static constexpr bool [compliant](#) (TT n)

Static Public Attributes

- static const constexpr size_t [vect_size](#) = 1

12.232.1 Member Typedef Documentation

12.232.1.1 vect_t

```
template<typename T >
using vect_t = T*
```

12.232.1.2 scalar_t

```
template<typename T >
using scalar_t = T
```

12.232.2 Member Function Documentation

12.232.2.1 type_string()

```
template<typename T >
static const std::string type_string ( ) [inline], [static]
```

12.232.2.2 valid()

```
template<typename T >
template<class TT >
static constexpr bool valid (
    TT p ) [inline], [static], [constexpr]
```

12.232.2.3 compliant()

```
template<typename T >
template<class TT >
static constexpr bool compliant (
    TT n ) [inline], [static], [constexpr]
```

12.232.3 Field Documentation

12.232.3.1 vect_size

```
template<typename T >
const constexpr size_t vect_size = 1 [static], [constexpr]
The documentation for this struct was generated from the following file:
```

- [fflas_simd.h](#)

12.233 Parallel< C, P > Struct Template Reference

Public Types

- typedef C [Cut](#)
- typedef P [Param](#)

Public Member Functions

- [Parallel](#) (size_t n=[NUM_THREADS](#))
- size_t [numthreads](#) () const
- size_t & [set_numthreads](#) (size_t n)

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Parallel](#) &p)

12.233.1 Member Typedef Documentation

12.233.1.1 Cut

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
typedef C Cut
```

12.233.1.2 Param

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
typedef P Param
```

12.233.2 Constructor & Destructor Documentation

12.233.2.1 Parallel()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
Parallel (
    size_t n = NUM\_THREADS ) [inline]
```

12.233.3 Member Function Documentation

12.233.3.1 numthreads()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
size_t numthreads ( ) const [inline]
```

12.233.3.2 set_numthreads()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
size_t & set_numthreads (
    size_t n ) [inline]
```

12.233.4 Friends And Related Symbol Documentation

12.233.4.1 operator<<

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
std::ostream & operator<< (
    std::ostream & out,
    const Parallel< C, P > & p ) [friend]
```

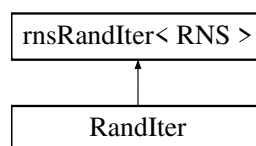
The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.234 RNSInteger< RNS >::RandIter Class Reference

```
#include <rns-integer.h>
```

Inheritance diagram for RNSInteger< RNS >::RandIter:



Public Member Functions

- [RandIter](#) (const [RNSInteger](#)< [RNS](#) > &F, size_t size=0, uint64_t seed=0)
- [RNS::Element](#) & [random](#) (typename [RNS::Element](#) &elt) const
RNS ring Element random assignement.
- [RNS::Element](#) [random](#) () const
- [RNS::Element](#) & [operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element](#) [operator\(\)](#) () const
- const [RNS](#) & [ring](#) () const

12.234.1 Constructor & Destructor Documentation

12.234.1.1 RandIter()

```
template<typename RNS >
RandIter (
    const RNSInteger< RNS > & F,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]
```

12.234.2 Member Function Documentation

12.234.2.1 random() [1/2]

```
template<typename RNS >
RNS::Element & random (
    typename RNS::Element & elt ) const [inline], [inherited]
```

RNS ring Element random assignement.

Element is supposed to be initialized

Returns

random ring Element

12.234.2.2 random() [2/2]

```
template<typename RNS >
RNS::Element random ( ) const [inline], [inherited]
```

12.234.2.3 operator>() [1/2]

```
template<typename RNS >
RNS::Element & operator\(\) (
    typename RNS::Element & elt ) const [inline], [inherited]
```

12.234.2.4 operator>() [2/2]

```
template<typename RNS >
RNS::Element operator\(\) ( ) const [inline], [inherited]
```

12.234.2.5 ring()

```
template<typename RNS >
const RNS & ring ( ) const [inline], [inherited]
```

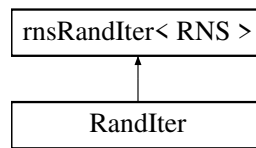
The documentation for this class was generated from the following file:

- [rns-integer.h](#)

12.235 RNSIntegerMod< RNS >::RandIter Class Reference

```
#include <rns-integer-mod.h>
```

Inheritance diagram for RNSIntegerMod< RNS >::RandIter:



Public Member Functions

- [RandIter](#) (const [RNSIntegerMod< RNS >](#) &F, [size_t](#) size=0, [uint64_t](#) seed=0)
- [RNS::Element](#) & [random](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element](#) [random](#) () const
- [RNS::Element](#) & [operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element](#) [operator\(\)](#) () const
- const [RNS](#) & [ring](#) () const

12.235.1 Constructor & Destructor Documentation

12.235.1.1 RandIter()

```
template<typename RNS >
RandIter (
    const RNSIntegerMod< RNS > & F,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]
```

12.235.2 Member Function Documentation

12.235.2.1 random() [1/2]

```
template<typename RNS >
RNS::Element & random (
    typename RNS::Element & elt ) const [inline]
```

12.235.2.2 random() [2/2]

```
template<typename RNS >
RNS::Element random ( ) const [inline], [inherited]
```

12.235.2.3 operator>() [1/2]

```
template<typename RNS >
RNS::Element & operator() (
    typename RNS::Element & elt ) const [inline], [inherited]
```

12.235.2.4 operator>() [2/2]

```
template<typename RNS >
RNS::Element operator() ( ) const [inline], [inherited]
```

12.235.2.5 ring()

```
template<typename RNS >
const RNS & ring ( ) const [inline], [inherited]
```

The documentation for this class was generated from the following file:

- [rns-integer-mod.h](#)

12.236 readMyMachineType< Field, T > Struct Template Reference

```
#include <read_sparse.h>
```

Public Types

- typedef [Field::Element](#) [Element](#)
- typedef [Field::Element_ptr](#) [Element_ptr](#)

Public Member Functions

- void [operator\(\)](#) (const [Field](#) &F, [Element](#) &modulo, [Element_ptr](#) val, std::ifstream &file, const [uint64_t](#) dims, const [mask_t](#) data_type, const [mask_t](#) field_desc)

12.236.1 Member Typedef Documentation

12.236.1.1 Element

```
template<class Field , class T >
typedef Field::Element Element
```

12.236.1.2 Element_ptr

```
template<class Field , class T >
typedef Field::Element\_ptr Element\_ptr
```

12.236.2 Member Function Documentation

12.236.2.1 operator>()

```
template<class Field , typename T >
void operator() (
    const Field & F,
    Element & modulo,
    Element\_ptr val,
    std::ifstream & file,
    const uint64\_t dims,
    const mask\_t data_type,
    const mask\_t field_desc )
```

The documentation for this struct was generated from the following file:

- [read_sparse.h](#)

12.237 readMyMachineType< Field, mpz_t > Struct Template Reference

```
#include <read_sparse.h>
```

Public Types

- typedef [Field::Element](#) [Element](#)
- typedef [Field::Element_ptr](#) [Element_ptr](#)

Public Member Functions

- void [operator\(\)](#) (const [Field](#) &F, [Element](#) &modulo, [Element_ptr](#) val, std::ifstream &file, const [uint64_t](#) dims, const [mask_t](#) data_type, const [mask_t](#) field_desc)

12.237.1 Member Typedef Documentation

12.237.1.1 Element

```
template<class Field >
typedef Field::Element Element
```

12.237.1.2 Element_ptr

```
template<class Field >
typedef Field::Element_ptr Element_ptr
```

12.237.2 Member Function Documentation

12.237.2.1 operator>()()

```
template<class Field >
void operator() (
    const Field & F,
    typename Field::Element & modulo,
    typename Field::Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

The documentation for this struct was generated from the following file:

- [read_sparse.h](#)

12.238 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.239 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.240 rint< K > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

12.241 rns_double Struct Reference

```
#include <rns-double.h>
```

Public Types

- typedef Givaro::Integer [integer](#)
- typedef Givaro::Modular< double > [ModField](#)
- typedef double [BasisElement](#)
- typedef [rns_double_elt](#) [Element](#)
- typedef [rns_double_elt_ptr](#) [Element_ptr](#)
- typedef [rns_double_elt_cstptr](#) [ConstElement_ptr](#)

Public Member Functions

- [rns_double](#) (const [integer](#) &bound, size_t pbits, bool rnsmod=false, long seed=time(NULL))
- [rns_double](#) (size_t pbits, size_t size, long seed=time(NULL))
- template<typename Vect >
 [rns_double](#) (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- [rns_double](#) (const [RNSIntegerMod](#)< [rns_double](#) > &basis, bool rnsmod=false, long seed=time(NULL))
- void [precompute_cst](#) (size_t K=0)
- template<typename T >
 void [init](#) (size_t m, size_t n, double *Arns, size_t rda, const T *A, size_t lda, const [integer](#) &maxA, bool RNS_MAJOR=false) const
- void [init](#) (size_t m, size_t n, double *Arns, size_t rda, const [integer](#) *A, size_t lda, size_t k, bool RNS_MAJOR=false) const
- void [init_transpose](#) (size_t m, size_t n, double *Arns, size_t rda, const [integer](#) *A, size_t lda, size_t k, bool RNS_MAJOR=false) const
- void [convert](#) (size_t m, size_t n, [integer](#) gamma, [integer](#) *A, size_t lda, const double *Arns, size_t rda, bool RNS_MAJOR=false) const
- void [convert_transpose](#) (size_t m, size_t n, [integer](#) gamma, [integer](#) *A, size_t lda, const double *Arns, size_t rda, bool RNS_MAJOR=false) const
- void [reduce](#) (size_t n, double *Arns, size_t rda, bool RNS_MAJOR=false) const
- template<size_t K>
 void [init](#) (size_t m, size_t n, double *Arns, size_t rda, const [Reclnt::ruint](#)< K > *A, size_t lda, size_t k, bool RNS_MAJOR=false) const
- template<size_t K>
 void [convert](#) (size_t m, size_t n, [integer](#) gamma, [Reclnt::ruint](#)< K > *A, size_t lda, const double *Arns, size_t rda, [integer](#) p=0, bool RNS_MAJOR=false) const

Data Fields

- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > [_basis](#)
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > [_basisMax](#)
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > [_negbasis](#)
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > [_invbasis](#)
- std::vector< [ModField](#) > [_field_rns](#)
- [integer](#) [_M](#)
- std::vector< [integer](#) > [_Mi](#)
- std::vector< double > [_MMi](#)
- std::vector< double > [_crt_in](#)
- std::vector< double > [_crt_out](#)
- size_t [_size](#)
- size_t [_pbits](#)
- size_t [_ldm](#)
- [integer](#) [_mi_sum](#)

12.241.1 Member Typedef Documentation

12.241.1.1 integer

```
typedef Givaro::Integer integer
```

12.241.1.2 ModField

```
typedef Givaro::Modular<double> ModField
```

12.241.1.3 BasisElement

```
typedef double BasisElement
```

12.241.1.4 Element

```
typedef rns_double_elt Element
```

12.241.1.5 Element_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

12.241.1.6 ConstElement_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

12.241.2 Constructor & Destructor Documentation

12.241.2.1 rns_double() [1/4]

```
rns_double (
    const integer & bound,
    size_t pbits,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

12.241.2.2 rns_double() [2/4]

```
rns_double (
    size_t pbits,
    size_t size,
    long seed = time(NULL) ) [inline]
```

12.241.2.3 rns_double() [3/4]

```
template<typename Vect >
rns_double (
    const Vect & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

12.241.2.4 rns_double() [4/4]

```
rns_double (
    const RNSIntegerMod< rns_double > & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

12.241.3 Member Function Documentation

12.241.3.1 precompute_cst()

```
void precompute_cst (
    size_t K = 0 ) [inline]
```

12.241.3.2 init() [1/3]

```
template<typename T >
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const T * A,
```

```

size_t lda,
const integer & maxA,
bool RNS_MAJOR = false ) const [inline]

```

12.241.3.3 init() [2/3]

```

void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) const [inline]

```

12.241.3.4 init_transpose()

```

void init_transpose (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) const [inline]

```

12.241.3.5 convert() [1/2]

```

void convert (
    size_t m,
    size_t n,
    integer gamma,
    integer * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]

```

12.241.3.6 convert_transpose()

```

void convert_transpose (
    size_t m,
    size_t n,
    integer gamma,
    integer * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]

```

12.241.3.7 reduce()

```

void reduce (
    size_t n,
    double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]

```

12.241.3.8 init() [3/3]

```
template<size_t K>
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const RecInt::ruint< K > * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) const [inline]
```

12.241.3.9 convert() [2/2]

```
template<size_t K>
void convert (
    size_t m,
    size_t n,
    integer gamma,
    RecInt::ruint< K > * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    integer p = 0,
    bool RNS_MAJOR = false ) const [inline]
```

12.241.4 Field Documentation**12.241.4.1 _basis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

12.241.4.2 _basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

12.241.4.3 _negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

12.241.4.4 _invbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

12.241.4.5 _field_rns

```
std::vector<ModField> _field_rns
```

12.241.4.6 _M

```
integer _M
```

12.241.4.7 _Mi

```
std::vector<integer> _Mi
```

12.241.4.8 _MMi

```
std::vector<double> _MMi
```

12.241.4.9 _crt_in

```
std::vector<double> _crt_in
```

12.241.4.10 _crt_out

```
std::vector<double> _crt_out
```

12.241.4.11 _size

```
size_t _size
```

12.241.4.12 _pbits

```
size_t _pbits
```

12.241.4.13 _ldm

```
size_t _ldm
```

12.241.4.14 _mi_sum

```
integer _mi_sum
```

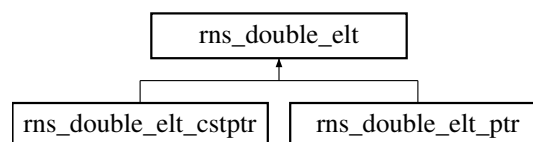
The documentation for this struct was generated from the following files:

- [rns-double.h](#)
- [rns-double-recint.inl](#)
- [rns-double.inl](#)

12.242 rns_double_elt Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for rns_double_elt:

**Public Member Functions**

- [rns_double_elt](#) ()
- [~rns_double_elt](#) ()
- [rns_double_elt](#) (double *p, size_t r, size_t a=false)
- [rns_double_elt_ptr](#) operator& ()
- [rns_double_elt_cstptr](#) operator& () const
- [rns_double_elt](#) (const [rns_double_elt](#) &x)

Data Fields

- double * [_ptr](#)
- size_t [_stride](#)
- bool [_alloc](#)

12.242.1 Constructor & Destructor Documentation

12.242.1.1 rns_double_elt() [1/3]

```
rns_double_elt ( ) [inline]
```

12.242.1.2 ~rns_double_elt()

```
~rns_double_elt ( ) [inline]
```

12.242.1.3 rns_double_elt() [2/3]

```
rns_double_elt (
    double * p,
    size_t r,
    size_t a = false ) [inline]
```

12.242.1.4 rns_double_elt() [3/3]

```
rns_double_elt (
    const rns_double_elt & x ) [inline]
```

12.242.2 Member Function Documentation

12.242.2.1 operator&() [1/2]

```
rns_double_elt_ptr operator& ( ) [inline]
```

12.242.2.2 operator&() [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline]
```

12.242.3 Field Documentation

12.242.3.1 _ptr

```
double* _ptr
```

12.242.3.2 _stride

```
size_t _stride
```

12.242.3.3 _alloc

```
bool _alloc
```

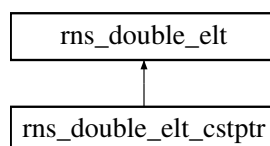
The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

12.243 rns_double_elt_cstptr Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for rns_double_elt_cstptr:



Public Member Functions

- [rns_double_elt_cstptr](#) ()
- [rns_double_elt_cstptr](#) (double *p, size_t r)
- [rns_double_elt_cstptr](#) (const [rns_double_elt_ptr](#) &x)
- [rns_double_elt_cstptr](#) (const [rns_double_elt_cstptr](#) &x)
- [rns_double_elt_cstptr](#) ([rns_double_elt_cstptr](#) &&)=default
- [rns_double_elt_cstptr](#) * [operator&](#) ()
- [rns_double_elt](#) & [operator*](#) () const
- [rns_double_elt](#) [operator\[\]](#) (size_t i) const
- [rns_double_elt](#) & [operator\[\]](#) (size_t i)
- [rns_double_elt_cstptr](#) [operator++](#) ()
- [rns_double_elt_cstptr](#) [operator--](#) ()
- [rns_double_elt_cstptr](#) [operator+](#) (size_t inc) const
- [rns_double_elt_cstptr](#) [operator-](#) (size_t inc) const
- [rns_double_elt_cstptr](#) & [operator+=](#) (size_t inc)
- [rns_double_elt_cstptr](#) & [operator-=](#) (size_t inc)
- [rns_double_elt_cstptr](#) & [operator=](#) (const [rns_double_elt_cstptr](#) &x)
- bool [operator<](#) (const [rns_double_elt_cstptr](#) &x)
- bool [operator!=](#) (const [rns_double_elt_cstptr](#) &x)
- [rns_double_elt_cstptr](#) [operator&](#) () const

Data Fields

- [rns_double_elt](#) other
- double * [_ptr](#)
- size_t [_stride](#)
- bool [_alloc](#)

12.243.1 Constructor & Destructor Documentation

12.243.1.1 [rns_double_elt_cstptr](#)() [1/5]

```
rns\_double\_elt\_cstptr ( ) [inline]
```

12.243.1.2 [rns_double_elt_cstptr](#)() [2/5]

```
rns\_double\_elt\_cstptr (
    double * p,
    size_t r ) [inline]
```

12.243.1.3 [rns_double_elt_cstptr](#)() [3/5]

```
rns\_double\_elt\_cstptr (
    const rns\_double\_elt\_ptr & x ) [inline]
```

12.243.1.4 [rns_double_elt_cstptr](#)() [4/5]

```
rns\_double\_elt\_cstptr (
    const rns\_double\_elt\_cstptr & x ) [inline]
```

12.243.1.5 [rns_double_elt_cstptr](#)() [5/5]

```
rns\_double\_elt\_cstptr (
    rns\_double\_elt\_cstptr && ) [default]
```

12.243.2 Member Function Documentation

12.243.2.1 operator&() [1/2]

```
rns_double_elt_cstptr * operator& ( ) [inline]
```

12.243.2.2 operator*()

```
rns_double_elt & operator* ( ) const [inline]
```

12.243.2.3 operator[]() [1/2]

```
rns_double_elt operator[] (
    size_t i ) const [inline]
```

12.243.2.4 operator[]() [2/2]

```
rns_double_elt & operator[] (
    size_t i ) [inline]
```

12.243.2.5 operator++()

```
rns_double_elt_cstptr operator++ ( ) [inline]
```

12.243.2.6 operator--()

```
rns_double_elt_cstptr operator-- ( ) [inline]
```

12.243.2.7 operator+()

```
rns_double_elt_cstptr operator+ (
    size_t inc ) const [inline]
```

12.243.2.8 operator-()

```
rns_double_elt_cstptr operator- (
    size_t inc ) const [inline]
```

12.243.2.9 operator+=()

```
rns_double_elt_cstptr & operator+= (
    size_t inc ) [inline]
```

12.243.2.10 operator-=()

```
rns_double_elt_cstptr & operator-= (
    size_t inc ) [inline]
```

12.243.2.11 operator=()

```
rns_double_elt_cstptr & operator= (
    const rns_double_elt_cstptr & x ) [inline]
```

12.243.2.12 operator<()

```
bool operator< (
    const rns_double_elt_cstptr & x ) [inline]
```

12.243.2.13 operator"!=()

```
bool operator!= (
    const rns\_double\_elt\_cstptr & x ) [inline]
```

12.243.2.14 operator&() [2/2]

```
rns\_double\_elt\_cstptr operator& ( ) const [inline], [inherited]
```

12.243.3 Field Documentation**12.243.3.1 other**

```
rns\_double\_elt other
```

12.243.3.2 _ptr

```
double* _ptr [inherited]
```

12.243.3.3 _stride

```
size_t _stride [inherited]
```

12.243.3.4 _alloc

```
bool _alloc [inherited]
```

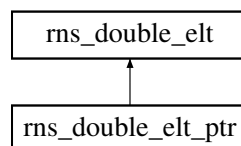
The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

12.244 rns_double_elt_ptr Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for `rns_double_elt_ptr`:

**Public Member Functions**

- [rns_double_elt_ptr](#) ()
- [rns_double_elt_ptr](#) (double *p, size_t r)
- [rns_double_elt_ptr](#) (const [rns_double_elt_ptr](#) &x)
- [rns_double_elt_ptr](#) (const [rns_double_elt_cstptr](#) &x)
- [rns_double_elt_ptr](#) ([rns_double_elt_ptr](#) &&)=default
- [rns_double_elt_ptr](#) * [operator&](#) ()
- [rns_double_elt](#) & [operator*](#) ()
- [rns_double_elt](#) [operator\[\]](#) (size_t i) const
- [rns_double_elt](#) & [operator\[\]](#) (size_t i)
- [rns_double_elt_ptr](#) [operator++](#) ()
- [rns_double_elt_ptr](#) [operator--](#) ()
- [rns_double_elt_ptr](#) [operator+](#) (size_t inc)
- [rns_double_elt_ptr](#) [operator-](#) (size_t inc)
- [rns_double_elt_ptr](#) & [operator+=](#) (size_t inc)
- [rns_double_elt_ptr](#) & [operator-=](#) (size_t inc)

- `rns_double_elt_ptr & operator=` (const `rns_double_elt_ptr` &x)
- `bool operator<` (const `rns_double_elt_ptr` &x)
- `bool operator!=` (const `rns_double_elt_ptr` &x)
- `rns_double_elt_cstptr operator& ()` const

Data Fields

- `rns_double_elt` other
- `double * _ptr`
- `size_t _stride`
- `bool _alloc`

12.244.1 Constructor & Destructor Documentation

12.244.1.1 `rns_double_elt_ptr()` [1/5]

```
rns_double_elt_ptr ( ) [inline]
```

12.244.1.2 `rns_double_elt_ptr()` [2/5]

```
rns_double_elt_ptr (
    double * p,
    size_t r ) [inline]
```

12.244.1.3 `rns_double_elt_ptr()` [3/5]

```
rns_double_elt_ptr (
    const rns_double_elt_ptr & x ) [inline]
```

12.244.1.4 `rns_double_elt_ptr()` [4/5]

```
rns_double_elt_ptr (
    const rns_double_elt_cstptr & x ) [inline]
```

12.244.1.5 `rns_double_elt_ptr()` [5/5]

```
rns_double_elt_ptr (
    rns_double_elt_ptr && ) [default]
```

12.244.2 Member Function Documentation

12.244.2.1 `operator&()` [1/2]

```
rns_double_elt_ptr * operator& ( ) [inline]
```

12.244.2.2 `operator*()`

```
rns_double_elt & operator* ( ) [inline]
```

12.244.2.3 `operator[]()` [1/2]

```
rns_double_elt operator[] (
    size_t i ) const [inline]
```

12.244.2.4 `operator[]()` [2/2]

```
rns_double_elt & operator[] (
    size_t i ) [inline]
```

12.244.2.5 operator++()

```
rns_double_elt_ptr operator++ ( ) [inline]
```

12.244.2.6 operator--()

```
rns_double_elt_ptr operator-- ( ) [inline]
```

12.244.2.7 operator+()

```
rns_double_elt_ptr operator+ (
    size_t inc ) [inline]
```

12.244.2.8 operator-()

```
rns_double_elt_ptr operator- (
    size_t inc ) [inline]
```

12.244.2.9 operator+=()

```
rns_double_elt_ptr & operator+= (
    size_t inc ) [inline]
```

12.244.2.10 operator-=()

```
rns_double_elt_ptr & operator-= (
    size_t inc ) [inline]
```

12.244.2.11 operator=()

```
rns_double_elt_ptr & operator= (
    const rns_double_elt_ptr & x ) [inline]
```

12.244.2.12 operator<()

```
bool operator< (
    const rns_double_elt_ptr & x ) [inline]
```

12.244.2.13 operator"!=()

```
bool operator!= (
    const rns_double_elt_ptr & x ) [inline]
```

12.244.2.14 operator&() [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```

12.244.3 Field Documentation**12.244.3.1 other**

```
rns_double_elt other
```

12.244.3.2 _ptr

```
double* _ptr [inherited]
```

12.244.3.3 _stride

```
size_t _stride [inherited]
```

12.244.3.4 _alloc

```
bool _alloc [inherited]
```

The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

12.245 rns_double_extended Struct Reference

```
#include <rns-double.h>
```

Public Types

- typedef Givaro::Integer [integer](#)
- typedef Givaro::ModularExtended< double > [ModField](#)
- typedef double [BasisElement](#)
- typedef [rns_double_elt](#) [Element](#)
- typedef [rns_double_elt_ptr](#) [Element_ptr](#)
- typedef [rns_double_elt_cstptr](#) [ConstElement_ptr](#)

Public Member Functions

- [rns_double_extended](#) (const [integer](#) &bound, size_t pbits, bool rnsmod=false, long seed=time(NULL))
- [rns_double_extended](#) (size_t pbits, size_t size, long seed=time(NULL))
- template<typename Vect >
 [rns_double_extended](#) (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- void [precompute_cst](#) ()
- void [init](#) (size_t m, size_t n, double *Arns, size_t rda, const [integer](#) *A, size_t lda, const [integer](#) &maxA, bool RNS_MAJOR=false) const
- void [init](#) (size_t m, size_t n, double *Arns, size_t rda, const [integer](#) *A, size_t lda, size_t k, bool RNS_MAJOR=false)
- void [convert](#) (size_t m, size_t n, [integer](#) gamma, [integer](#) *A, size_t lda, const double *Arns, size_t rda, bool RNS_MAJOR=false)
- void [init](#) (size_t m, double *Arns, const [integer](#) *A, size_t lda) const
- void [convert](#) (size_t m, [integer](#) *A, const double *Arns) const
- void [reduce](#) (size_t n, double *Arns, size_t rda, bool RNS_MAJOR=false) const

Data Fields

- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > [_basis](#)
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > [_basisMax](#)
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > [_negbasis](#)
- std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > [_invbasis](#)
- std::vector< [ModField](#) > [_field_rns](#)
- [integer](#) [_M](#)
- std::vector< [integer](#) > [_Mi](#)
- std::vector< double > [_MMi](#)
- std::vector< double > [_crt_in](#)
- std::vector< double > [_crt_out](#)
- size_t [_size](#)
- size_t [_pbits](#)
- size_t [_ldm](#)

12.245.1 Member Typedef Documentation

12.245.1.1 integer

```
typedef Givaro::Integer integer
```

12.245.1.2 ModField

```
typedef Givaro::ModularExtended<double> ModField
```

12.245.1.3 BasisElement

```
typedef double BasisElement
```

12.245.1.4 Element

```
typedef rns_double_elt Element
```

12.245.1.5 Element_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

12.245.1.6 ConstElement_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

12.245.2 Constructor & Destructor Documentation**12.245.2.1 rns_double_extended() [1/3]**

```
rns_double_extended (
    const integer & bound,
    size_t pbits,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

12.245.2.2 rns_double_extended() [2/3]

```
rns_double_extended (
    size_t pbits,
    size_t size,
    long seed = time(NULL) ) [inline]
```

12.245.2.3 rns_double_extended() [3/3]

```
template<typename Vect >
rns_double_extended (
    const Vect & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

12.245.3 Member Function Documentation**12.245.3.1 precompute_cst()**

```
void precompute_cst ( ) [inline]
```

12.245.3.2 init() [1/3]

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
```



```
const integer & maxA,
bool RNS_MAJOR = false ) const [inline]
```

12.245.3.3 init() [2/3]

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) [inline]
```

12.245.3.4 convert() [1/2]

```
void convert (
    size_t m,
    size_t n,
    integer gamma,
    integer * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) [inline]
```

12.245.3.5 init() [3/3]

```
void init (
    size_t m,
    double * Arns,
    const integer * A,
    size_t lda ) const [inline]
```

12.245.3.6 convert() [2/2]

```
void convert (
    size_t m,
    integer * A,
    const double * Arns ) const [inline]
```

12.245.3.7 reduce()

```
void reduce (
    size_t n,
    double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

12.245.4 Field Documentation

12.245.4.1 _basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

12.245.4.2 _basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

12.245.4.3 _negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

12.245.4.4 _invbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

12.245.4.5 _field_rns

```
std::vector<ModField> _field_rns
```

12.245.4.6 _M

```
integer _M
```

12.245.4.7 _Mi

```
std::vector<integer> _Mi
```

12.245.4.8 _MMi

```
std::vector<double> _MMi
```

12.245.4.9 _crt_in

```
std::vector<double> _crt_in
```

12.245.4.10 _crt_out

```
std::vector<double> _crt_out
```

12.245.4.11 _size

```
size_t _size
```

12.245.4.12 _pbits

```
size_t _pbits
```

12.245.4.13 _ldm

```
size_t _ldm
```

The documentation for this struct was generated from the following files:

- [rns-double.h](#)
- [rns-double.inl](#)

12.246 RNSElementTag Struct Reference

Representation in a Residue Number System.

```
#include <field-traits.h>
```

12.246.1 Detailed Description

Representation in a Residue Number System.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.247 RNSInteger< RNS > Class Template Reference

```
#include <rns-integer.h>
```

Data Structures

- class [RandIter](#)

Public Types

- typedef [RNS::Element](#) [Element](#)
- typedef [RNS::Element_ptr](#) [Element_ptr](#)
- typedef [RNS::ConstElement_ptr](#) [ConstElement_ptr](#)

Public Member Functions

- [RNSInteger](#) (const [RNS](#) &myrns)
- template<typename T >
 [RNSInteger](#) (const T &F)
- const [RNS](#) & [rns](#) () const
- size_t [size](#) () const
- bool [isOne](#) (const [Element](#) &x) const
- bool [isMOne](#) (const [Element](#) &x) const
- bool [isZero](#) (const [Element](#) &x) const
- integer characteristic ([integer](#) &p) const
- integer cardinality ([integer](#) &p) const
- [Element](#) & [init](#) ([Element](#) &x) const
- [Element](#) & [init](#) ([Element](#) &x, const Givaro::Integer &y) const
- [Element](#) & [reduce](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) & [reduce](#) ([Element](#) &x) const
- Givaro::Integer [convert](#) (Givaro::Integer &x, const [Element](#) &y) const
- [Element](#) & [assign](#) ([Element](#) &x, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os) const

Data Fields

- [Element](#) [one](#)
- [Element](#) [mOne](#)
- [Element](#) [zero](#)

Protected Types

- typedef [RNS::BasisElement](#) [BasisElement](#)
- typedef Givaro::Integer [integer](#)

Protected Attributes

- const [RNS](#) * [_rns](#)

12.247.1 Member Typedef Documentation

12.247.1.1 BasisElement

```
template<typename RNS >
typedef RNS::BasisElement BasisElement [protected]
```

12.247.1.2 integer

```
template<typename RNS >
typedef Givaro::Integer integer [protected]
```

12.247.1.3 Element

```
template<typename RNS >
typedef RNS::Element Element
```

12.247.1.4 Element_ptr

```
template<typename RNS >
typedef RNS::Element_ptr Element_ptr
```

12.247.1.5 ConstElement_ptr

```
template<typename RNS >
typedef RNS::ConstElement_ptr ConstElement_ptr
```

12.247.2 Constructor & Destructor Documentation**12.247.2.1 RNSInteger() [1/2]**

```
template<typename RNS >
RNSInteger (
    const RNS & myrns ) [inline]
```

12.247.2.2 RNSInteger() [2/2]

```
template<typename RNS >
template<typename T >
RNSInteger (
    const T & F ) [inline]
```

12.247.3 Member Function Documentation**12.247.3.1 rns()**

```
template<typename RNS >
const RNS & rns ( ) const [inline]
```

12.247.3.2 size()

```
template<typename RNS >
size_t size ( ) const [inline]
```

12.247.3.3 isOne()

```
template<typename RNS >
bool isOne (
    const Element & x ) const [inline]
```

12.247.3.4 isMOne()

```
template<typename RNS >
bool isMOne (
    const Element & x ) const [inline]
```

12.247.3.5 isZero()

```
template<typename RNS >
bool isZero (
    const Element & x ) const [inline]
```

12.247.3.6 characteristic()

```
template<typename RNS >
integer characteristic (
    integer & p ) const [inline]
```

12.247.3.7 cardinality()

```
template<typename RNS >
integer cardinality (
    integer & p ) const [inline]
```

12.247.3.8 init() [1/2]

```
template<typename RNS >
Element & init (
    Element & x ) const [inline]
```

12.247.3.9 init() [2/2]

```
template<typename RNS >
Element & init (
    Element & x,
    const Givaro::Integer & y ) const [inline]
```

12.247.3.10 reduce() [1/2]

```
template<typename RNS >
Element & reduce (
    Element & x,
    const Element & y ) const [inline]
```

12.247.3.11 reduce() [2/2]

```
template<typename RNS >
Element & reduce (
    Element & x ) const [inline]
```

12.247.3.12 convert()

```
template<typename RNS >
Givaro::Integer convert (
    Givaro::Integer & x,
    const Element & y ) const [inline]
```

12.247.3.13 assign()

```
template<typename RNS >
Element & assign (
    Element & x,
    const Element & y ) const [inline]
```

12.247.3.14 write() [1/2]

```
template<typename RNS >
std::ostream & write (
    std::ostream & os,
    const Element & y ) const [inline]
```

12.247.3.15 write() [2/2]

```
template<typename RNS >
std::ostream & write (
    std::ostream & os ) const [inline]
```

12.247.4 Field Documentation**12.247.4.1 _rns**

```
template<typename RNS >
const RNS* _rns [protected]
```

12.247.4.2 one

```
template<typename RNS >
Element one
```

12.247.4.3 mOne

```
template<typename RNS >
Element mOne
```

12.247.4.4 zero

```
template<typename RNS >
Element zero
```

The documentation for this class was generated from the following files:

- [field-traits.h](#)
- [rns-integer.h](#)

12.248 RNSIntegerMod< RNS > Class Template Reference

```
#include <rns-integer-mod.h>
```

Data Structures

- class [RandIter](#)

Public Types

- typedef [RNS::Element](#) [Element](#)
- typedef [RNS::Element_ptr](#) [Element_ptr](#)
- typedef [RNS::ConstElement_ptr](#) [ConstElement_ptr](#)

Public Member Functions

- [RNSIntegerMod](#) (const [integer](#) &p, const [RNS](#) &myrns)
- const [rns_double](#) & [rns](#) () const
- const [RNSInteger](#)< [RNS](#) > & [delayed](#) () const
- [size_t](#) [size](#) () const

- bool [isOne](#) (const [Element](#) &x) const
- bool [isMOne](#) (const [Element](#) &x) const
- bool [isZero](#) (const [Element](#) &x) const
- [integer](#) & [characteristic](#) ([integer](#) &p) const
- [integer](#) [characteristic](#) () const
- [integer](#) & [cardinality](#) ([integer](#) &p) const
- [integer](#) [cardinality](#) () const
- [integer](#) [minElement](#) () const
- [integer](#) [maxElement](#) () const
- [Element](#) & [init](#) ([Element](#) &x) const
- [Element](#) & [init](#) ([Element](#) &x, const Givaro::Integer &y) const
- [Element](#) & [reduce](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) & [reduce](#) ([Element](#) &x) const
- [Element](#) & [init](#) ([Element](#) &x, const [Element](#) &y) const
- Givaro::Integer [convert](#) (Givaro::Integer &x, const [Element](#) &y) const
- [Element](#) & [assign](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) & [add](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) & [sub](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) & [neg](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) & [mul](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) & [axpyin](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) & [inv](#) ([Element](#) &x, const [Element](#) &y) const
- bool [areEqual](#) (const [Element](#) &x, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os) const
- void [reduce_modp](#) (size_t n, [Element_ptr](#) B) const
- std::ostream & [write_matrix](#) (std::ostream &c, const double *E, int n, int m, int lda) const
- std::ostream & [write_matrix_long](#) (std::ostream &c, const double *E, int n, int m, int lda) const
- void [reduce_modp](#) (size_t m, size_t n, [Element_ptr](#) B, size_t lda) const
- void [reduce_modp_rnsmajor](#) (size_t n, [Element_ptr](#) B) const

Data Fields

- [Element one](#)
- [Element mOne](#)
- [Element zero](#)

Protected Types

- typedef [RNS::BasisElement](#) [BasisElement](#)
- typedef Givaro::Modular< [BasisElement](#) > [ModField](#)
- typedef Givaro::Integer [integer](#)

Protected Attributes

- [integer _p](#)
- std::vector< [BasisElement](#), AlignedAllocator< [BasisElement](#), Alignment::CACHE_LINE > > [_Mi_modp_rns](#)
- std::vector< [BasisElement](#), AlignedAllocator< [BasisElement](#), Alignment::CACHE_LINE > > [_iM_modp_rns](#)
- const [RNS](#) * [_rns](#)
- Givaro::Modular< Givaro::Integer > [_F](#)
- [RNSInteger](#)< [RNS](#) > [_RNSdelayed](#)

12.248.1 Member Typedef Documentation

12.248.1.1 Element

```
template<typename RNS >
typedef RNS::Element Element
```

12.248.1.2 Element_ptr

```
template<typename RNS >
typedef RNS::Element_ptr Element_ptr
```

12.248.1.3 ConstElement_ptr

```
template<typename RNS >
typedef RNS::ConstElement_ptr ConstElement_ptr
```

12.248.1.4 BasisElement

```
template<typename RNS >
typedef RNS::BasisElement BasisElement [protected]
```

12.248.1.5 ModField

```
template<typename RNS >
typedef Givaro::Modular<BasisElement> ModField [protected]
```

12.248.1.6 integer

```
template<typename RNS >
typedef Givaro::Integer integer [protected]
```

12.248.2 Constructor & Destructor Documentation**12.248.2.1 RNSIntegerMod()**

```
template<typename RNS >
RNSIntegerMod (
    const integer & p,
    const RNS & myrns ) [inline]
```

12.248.3 Member Function Documentation**12.248.3.1 rns()**

```
template<typename RNS >
const rns_double & rns ( ) const [inline]
```

12.248.3.2 delayed()

```
template<typename RNS >
const RNSInteger< RNS > & delayed ( ) const [inline]
```

12.248.3.3 size()

```
template<typename RNS >
size_t size ( ) const [inline]
```

12.248.3.4 isOne()

```
template<typename RNS >
bool isOne (
    const Element & x ) const [inline]
```


12.248.3.5 isMOne()

```
template<typename RNS >
bool isMOne (
    const Element & x ) const [inline]
```

12.248.3.6 isZero()

```
template<typename RNS >
bool isZero (
    const Element & x ) const [inline]
```

12.248.3.7 characteristic() [1/2]

```
template<typename RNS >
integer & characteristic (
    integer & p ) const [inline]
```

12.248.3.8 characteristic() [2/2]

```
template<typename RNS >
integer characteristic ( ) const [inline]
```

12.248.3.9 cardinality() [1/2]

```
template<typename RNS >
integer & cardinality (
    integer & p ) const [inline]
```

12.248.3.10 cardinality() [2/2]

```
template<typename RNS >
integer cardinality ( ) const [inline]
```

12.248.3.11 minElement()

```
template<typename RNS >
integer minElement ( ) const [inline]
```

12.248.3.12 maxElement()

```
template<typename RNS >
integer maxElement ( ) const [inline]
```

12.248.3.13 init() [1/3]

```
template<typename RNS >
Element & init (
    Element & x ) const [inline]
```

12.248.3.14 init() [2/3]

```
template<typename RNS >
Element & init (
    Element & x,
    const Givaro::Integer & y ) const [inline]
```

12.248.3.15 reduce() [1/2]

```
template<typename RNS >
Element & reduce (
    Element & x,
    const Element & y ) const [inline]
```

12.248.3.16 reduce() [2/2]

```
template<typename RNS >
Element & reduce (
    Element & x ) const [inline]
```

12.248.3.17 init() [3/3]

```
template<typename RNS >
Element & init (
    Element & x,
    const Element & y ) const [inline]
```

12.248.3.18 convert()

```
template<typename RNS >
Givaro::Integer convert (
    Givaro::Integer & x,
    const Element & y ) const [inline]
```

12.248.3.19 assign()

```
template<typename RNS >
Element & assign (
    Element & x,
    const Element & y ) const [inline]
```

12.248.3.20 add()

```
template<typename RNS >
Element & add (
    Element & x,
    const Element & y,
    const Element & z ) const [inline]
```

12.248.3.21 sub()

```
template<typename RNS >
Element & sub (
    Element & x,
    const Element & y,
    const Element & z ) const [inline]
```

12.248.3.22 neg()

```
template<typename RNS >
Element & neg (
    Element & x,
    const Element & y ) const [inline]
```

12.248.3.23 mul()

```
template<typename RNS >
Element & mul (
    Element & x,
    const Element & y,
    const Element & z ) const [inline]
```

12.248.3.24 axpyin()

```
template<typename RNS >
Element & axpyin (
    Element & x,
    const Element & y,
    const Element & z ) const [inline]
```

12.248.3.25 inv()

```
template<typename RNS >
Element & inv (
    Element & x,
    const Element & y ) const [inline]
```

12.248.3.26 areEqual()

```
template<typename RNS >
bool areEqual (
    const Element & x,
    const Element & y ) const [inline]
```

12.248.3.27 write() [1/2]

```
template<typename RNS >
std::ostream & write (
    std::ostream & os,
    const Element & y ) const [inline]
```

12.248.3.28 write() [2/2]

```
template<typename RNS >
std::ostream & write (
    std::ostream & os ) const [inline]
```

12.248.3.29 reduce_modp() [1/2]

```
template<typename RNS >
void reduce_modp (
    size_t n,
    Element_ptr B ) const [inline]
```

12.248.3.30 write_matrix()

```
template<typename RNS >
std::ostream & write_matrix (
    std::ostream & c,
    const double * E,
    int n,
    int m,
    int lda ) const [inline]
```

12.248.3.31 write_matrix_long()

```
template<typename RNS >
std::ostream & write_matrix_long (
    std::ostream & c,
    const double * E,
    int n,
    int m,
    int lda ) const [inline]
```

12.248.3.32 reduce_modp() [2/2]

```
template<typename RNS >
void reduce_modp (
    size_t m,
    size_t n,
    Element_ptr B,
    size_t lda ) const [inline]
```

12.248.3.33 reduce_modp_rnsmajor()

```
template<typename RNS >
void reduce_modp_rnsmajor (
    size_t n,
    Element_ptr B ) const [inline]
```

12.248.4 Field Documentation**12.248.4.1 _p**

```
template<typename RNS >
integer _p [protected]
```

12.248.4.2 _Mi_modp_rns

```
template<typename RNS >
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _Mi_modp_↔
rns [protected]
```

12.248.4.3 _iM_modp_rns

```
template<typename RNS >
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _iM_modp_↔
rns [protected]
```

12.248.4.4 _rns

```
template<typename RNS >
const RNS* _rns [protected]
```

12.248.4.5 _F

```
template<typename RNS >
Givaro::Modular<Givaro::Integer> _F [protected]
```

12.248.4.6 _RNSdelayed

```
template<typename RNS >
RNSInteger<RNS> _RNSdelayed [protected]
```

12.248.4.7 one

```
template<typename RNS >
Element one
```

12.248.4.8 mOne

```
template<typename RNS >
Element mOne
```

12.248.4.9 zero

```
template<typename RNS >
Element zero
```

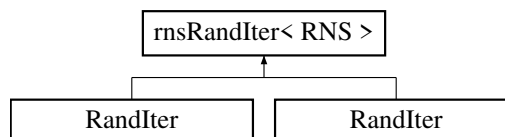
The documentation for this class was generated from the following files:

- [field-traits.h](#)
- [rns-integer-mod.h](#)

12.249 rnsRandIter< RNS > Class Template Reference

```
#include <rns-double.h>
```

Inheritance diagram for rnsRandIter< RNS >:

**Public Member Functions**

- [rnsRandIter](#) (const [RNS](#) &R, size_t size=0, uint64_t seed=0)
- [RNS::Element & random](#) (typename [RNS::Element](#) &elt) const
RNS ring Element random assignement.
- [RNS::Element & operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element operator\(\)](#) () const
- [RNS::Element random](#) () const
- const [RNS](#) & [ring](#) () const

12.249.1 Constructor & Destructor Documentation**12.249.1.1 rnsRandIter()**

```
template<typename RNS >
rnsRandIter (
    const RNS & R,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]
```

12.249.2 Member Function Documentation**12.249.2.1 random() [1/2]**

```
template<typename RNS >
RNS::Element & random (
    typename RNS::Element & elt ) const [inline]
```

RNS ring Element random assignement.

Element is supposed to be initialized

Returns

random ring Element

12.249.2.2 operator>() [1/2]

```
template<typename RNS >
RNS::Element & operator() (
    typename RNS::Element & elt ) const [inline]
```

12.249.2.3 operator>() [2/2]

```
template<typename RNS >
RNS::Element operator() ( ) const [inline]
```

12.249.2.4 random() [2/2]

```
template<typename RNS >
RNS::Element random ( ) const [inline]
```

12.249.2.5 ring()

```
template<typename RNS >
const RNS & ring ( ) const [inline]
```

The documentation for this class was generated from the following file:

- [rns-double.h](#)

12.250 Row Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.251 ruint< K > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

12.252 ScalFunctions< Element, Enable > Struct Template Reference

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

12.253 ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type > Struct Template Reference**Static Public Member Functions**

- static Element [zero](#) ()
- static Element [vand](#) (Element x1, Element x2)
- static Element [vor](#) (Element x1, Element x2)
- static Element [vxor](#) (Element x1, Element x2)
- static Element [vandnot](#) (Element x1, Element x2)

- static Element [ceil](#) (Element x)
- static Element [floor](#) (Element x)
- static Element [round](#) (Element x)
- static Element [add](#) (Element x1, Element x2)
- static Element [addin](#) (Element &x1, Element x2)
- static Element [sub](#) (Element x1, Element x2)
- static Element [subin](#) (Element &x1, Element x2)
- static Element [mul](#) (Element x1, Element x2)
- static Element [mulin](#) (Element &x1, Element x2)
- static Element [div](#) (Element x1, Element x2)
- static Element [fmadd](#) (Element x1, Element x2, Element x3)
- static Element [fmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsub](#) (Element x1, Element x2, Element x3)
- static Element [fmsubin](#) (Element &x1, Element x2, Element x3)
- static Element [fnmadd](#) (Element x1, Element x2, Element x3)
- static Element [fnmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [lesser](#) (Element x1, Element x2)
- static Element [lesser_eq](#) (Element x1, Element x2)
- static Element [greater](#) (Element x1, Element x2)
- static Element [greater_eq](#) (Element x1, Element x2)
- static Element [eq](#) (Element x1, Element x2)

12.253.1 Member Function Documentation

12.253.1.1 zero()

```
template<class Element >  
static Element zero ( ) [inline], [static]
```

12.253.1.2 vand()

```
template<class Element >  
static Element vand (  
    Element x1,  
    Element x2 ) [inline], [static]
```

12.253.1.3 vor()

```
template<class Element >  
static Element vor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

12.253.1.4 vxor()

```
template<class Element >  
static Element vxor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

12.253.1.5 vandnot()

```
template<class Element >  
static Element vandnot (  
    Element x1,  
    Element x2 ) [inline], [static]
```

12.253.1.6 ceil()

```
template<class Element >
static Element ceil (
    Element x ) [inline], [static]
```

12.253.1.7 floor()

```
template<class Element >
static Element floor (
    Element x ) [inline], [static]
```

12.253.1.8 round()

```
template<class Element >
static Element round (
    Element x ) [inline], [static]
```

12.253.1.9 add()

```
template<class Element >
static Element add (
    Element x1,
    Element x2 ) [inline], [static]
```

12.253.1.10 addin()

```
template<class Element >
static Element addin (
    Element & x1,
    Element x2 ) [inline], [static]
```

12.253.1.11 sub()

```
template<class Element >
static Element sub (
    Element x1,
    Element x2 ) [inline], [static]
```

12.253.1.12 subin()

```
template<class Element >
static Element subin (
    Element & x1,
    Element x2 ) [inline], [static]
```

12.253.1.13 mul()

```
template<class Element >
static Element mul (
    Element x1,
    Element x2 ) [inline], [static]
```

12.253.1.14 mulin()

```
template<class Element >
static Element mulin (
    Element & x1,
    Element x2 ) [inline], [static]
```


12.253.1.15 div()

```
template<class Element >
static Element div (
    Element x1,
    Element x2 ) [inline], [static]
```

12.253.1.16 fmadd()

```
template<class Element >
static Element fmadd (
    Element x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.253.1.17 fmaddin()

```
template<class Element >
static Element fmaddin (
    Element & x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.253.1.18 fmsub()

```
template<class Element >
static Element fmsub (
    Element x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.253.1.19 fmsubin()

```
template<class Element >
static Element fmsubin (
    Element & x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.253.1.20 fnmadd()

```
template<class Element >
static Element fnmadd (
    Element x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.253.1.21 fnmaddin()

```
template<class Element >
static Element fnmaddin (
    Element & x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.253.1.22 lesser()

```
template<class Element >
static Element lesser (
```

```

        Element x1,
        Element x2 ) [inline], [static]

```

12.253.1.23 lesser_eq()

```

template<class Element >
static Element lesser_eq (
        Element x1,
        Element x2 ) [inline], [static]

```

12.253.1.24 greater()

```

template<class Element >
static Element greater (
        Element x1,
        Element x2 ) [inline], [static]

```

12.253.1.25 greater_eq()

```

template<class Element >
static Element greater_eq (
        Element x1,
        Element x2 ) [inline], [static]

```

12.253.1.26 eq()

```

template<class Element >
static Element eq (
        Element x1,
        Element x2 ) [inline], [static]

```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

12.254 ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type > Struct Template Reference

Static Public Member Functions

- static Element [zero](#) ()
- static Element [round](#) (Element x)
- static Element [vand](#) (Element x1, Element x2)
- static Element [vor](#) (Element x1, Element x2)
- static Element [vxor](#) (Element x1, Element x2)
- static Element [vandnot](#) (Element x1, Element x2)
- static Element [add](#) (Element x1, Element x2)
- static Element [addin](#) (Element &x1, Element x2)
- static Element [sub](#) (Element x1, Element x2)
- static Element [subin](#) (Element &x1, Element x2)
- static Element [mul](#) (Element x1, Element x2)
- static Element [mullo](#) (Element x1, Element x2)
- static Element [mulhi](#) (Element x1, Element x2)
- static Element [mulx](#) (Element x1, Element x2)
- static Element [fmadd](#) (Element x1, Element x2, Element x3)
- static Element [fmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fmaddx](#) (Element x1, Element x2, Element x3)
- static Element [fmaddxin](#) (Element &x1, Element x2, Element x3)

- static Element [fmsub](#) (Element x1, Element x2, Element x3)
- static Element [fmsubin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsubx](#) (Element x1, Element x2, Element x3)
- static Element [fmsubxin](#) (Element &x1, Element x2, Element x3)
- static Element [fmadd](#) (Element x1, Element x2, Element x3)
- static Element [fmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fmaddx](#) (Element x1, Element x2, Element x3)
- static Element [fmaddxin](#) (Element &x1, Element x2, Element x3)
- template<int s, bool EnableTrue = true>
static enable_if<!is_signed< Element >::value &&EnableTrue, Element >::type [sra](#) (Element x1)
- template<int s, bool EnableTrue = true>
static enable_if< is_signed< Element >::value &&EnableTrue, Element >::type [sra](#) (Element x1)
- template<int s>
static Element [srl](#) (Element x1)
- template<int s>
static Element [sll](#) (Element x1)
- static Element [lesser](#) (Element x1, Element x2)
- static Element [lesser_eq](#) (Element x1, Element x2)
- static Element [greater](#) (Element x1, Element x2)
- static Element [greater_eq](#) (Element x1, Element x2)
- static Element [eq](#) (Element x1, Element x2)

12.254.1 Member Function Documentation

12.254.1.1 zero()

```
template<class Element >  
static Element zero ( ) [inline], [static]
```

12.254.1.2 round()

```
template<class Element >  
static Element round (  
    Element x ) [inline], [static]
```

12.254.1.3 vand()

```
template<class Element >  
static Element vand (  
    Element x1,  
    Element x2 ) [inline], [static]
```

12.254.1.4 vor()

```
template<class Element >  
static Element vor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

12.254.1.5 vxor()

```
template<class Element >  
static Element vxor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

12.254.1.6 vandnot()

```
template<class Element >
static Element vandnot (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.7 add()

```
template<class Element >
static Element add (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.8 addin()

```
template<class Element >
static Element addin (
    Element & x1,
    Element x2 ) [inline], [static]
```

12.254.1.9 sub()

```
template<class Element >
static Element sub (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.10 subin()

```
template<class Element >
static Element subin (
    Element & x1,
    Element x2 ) [inline], [static]
```

12.254.1.11 mul()

```
template<class Element >
static Element mul (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.12 mullo()

```
template<class Element >
static Element mullo (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.13 mulhi()

```
template<class Element >
static Element mulhi (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.14 mulx()

```
template<class Element >
static Element mulx (
```

```
Element x1,  
Element x2 ) [inline], [static]
```

12.254.1.15 fmadd()

```
template<class Element >  
static Element fmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

12.254.1.16 fmaddin()

```
template<class Element >  
static Element fmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

12.254.1.17 fmaddx()

```
template<class Element >  
static Element fmaddx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

12.254.1.18 fmaddxin()

```
template<class Element >  
static Element fmaddxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

12.254.1.19 fmsub()

```
template<class Element >  
static Element fmsub (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

12.254.1.20 fmsubin()

```
template<class Element >  
static Element fmsubin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

12.254.1.21 fmsubx()

```
template<class Element >  
static Element fmsubx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

12.254.1.22 fmsubxin()

```
template<class Element >
static Element fmsubxin (
    Element & x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.254.1.23 fnmadd()

```
template<class Element >
static Element fnmadd (
    Element x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.254.1.24 fnmaddin()

```
template<class Element >
static Element fnmaddin (
    Element & x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.254.1.25 fnmaddx()

```
template<class Element >
static Element fnmaddx (
    Element x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.254.1.26 fnmaddxin()

```
template<class Element >
static Element fnmaddxin (
    Element & x1,
    Element x2,
    Element x3 ) [inline], [static]
```

12.254.1.27 sra() [1/2]

```
template<class Element >
template<int s, bool EnableTrue = true>
static enable_if<!is_signed< Element >::value &&EnableTrue, Element >::type sra (
    Element x1 ) [inline], [static]
```

12.254.1.28 sra() [2/2]

```
template<class Element >
template<int s, bool EnableTrue = true>
static enable_if< is_signed< Element >::value &&EnableTrue, Element >::type sra (
    Element x1 ) [inline], [static]
```

12.254.1.29 srl()

```
template<class Element >
template<int s>
static Element srl (
    Element x1 ) [inline], [static]
```

12.254.1.30 sll()

```
template<class Element >
template<int s>
static Element sll (
    Element x1 ) [inline], [static]
```

12.254.1.31 lesser()

```
template<class Element >
static Element lesser (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.32 lesser_eq()

```
template<class Element >
static Element lesser_eq (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.33 greater()

```
template<class Element >
static Element greater (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.34 greater_eq()

```
template<class Element >
static Element greater_eq (
    Element x1,
    Element x2 ) [inline], [static]
```

12.254.1.35 eq()

```
template<class Element >
static Element eq (
    Element x1,
    Element x2 ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

12.255 Sequential Struct Reference**Public Member Functions**

- [Sequential](#) ()
- [Sequential](#) (size_t nth)
- template<class Cut , class Param >
 [Sequential](#) ([Parallel](#)< Cut, Param > &)
- size_t [numthreads](#) () const

Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Sequential](#) &)

12.255.1 Constructor & Destructor Documentation

12.255.1.1 Sequential() [1/3]

```
Sequential ( ) [inline]
```

12.255.1.2 Sequential() [2/3]

```
Sequential (
    size_t nth ) [inline]
```

12.255.1.3 Sequential() [3/3]

```
template<class Cut , class Param >
Sequential (
    Parallel< Cut, Param > & ) [inline]
```

12.255.2 Member Function Documentation

12.255.2.1 numthreads()

```
size_t numthreads ( ) const [inline]
```

12.255.3 Friends And Related Symbol Documentation

12.255.3.1 operator<<

```
std::ostream & operator<< (
    std::ostream & out,
    const Sequential & ) [friend]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

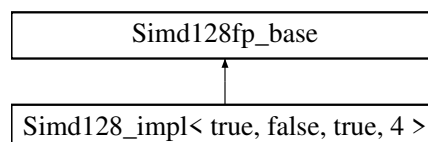
12.256 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

12.257 Simd128_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd128_impl< true, false, true, 4 >:



Static Public Member Functions

- static const std::string [type_string](#) ()

12.257.1 Member Function Documentation

12.257.1.1 type_string()

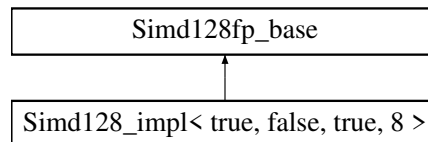
`static const std::string type_string () [inline], [static], [inherited]`

The documentation for this struct was generated from the following file:

- [simd128_float.inl](#)

12.258 Simd128_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd128_impl< true, false, true, 8 >:



Static Public Member Functions

- static const std::string [type_string](#) ()

12.258.1 Member Function Documentation

12.258.1.1 type_string()

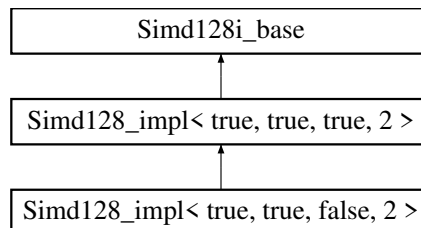
`static const std::string type_string () [inline], [static], [inherited]`

The documentation for this struct was generated from the following file:

- [simd128_double.inl](#)

12.259 Simd128_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, false, 2 >:



Data Structures

- union [Converter](#)

Public Types

- using [scalar_t](#) = [uint16_t](#)
- using [vect_t](#) = [__m128i](#)

Static Public Member Functions

- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3, const [scalar_t](#) x4, const [scalar_t](#) x5, const [scalar_t](#) x6, const [scalar_t](#) x7)

- `template<class T >`
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`
`static INLINE CONST vect_t sra (const vect_t a)`
- `static INLINE CONST vect_t greater (vect_t a, vect_t b)`
- `static INLINE CONST vect_t lesser (vect_t a, vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `template<class T >`
`static constexpr bool valid (T *p)`
- `template<class T >`
`static constexpr bool compliant (T n)`
- `static INLINE CONST vect_t set1 (const scalar_t x)`
- `static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)`
- `template<class T >`
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`
`static INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<uint32_t s>`
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `template<uint8_t s>`
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`

- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE vect_t mod (vect_t &C, const vect_t &P, const __m64 &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- static const std::string `type_string ()`
- static `INLINE CONST vect_t zero ()`
- `template<uint8_t s>`
static `INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`
static `INLINE CONST vect_t srl128 (const vect_t a)`
- static `INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

Static Public Attributes

- static const constexpr size_t `vect_size` = 8
- static const constexpr size_t `alignment` = 16

12.259.1 Member Typedef Documentation

12.259.1.1 scalar_t

using `scalar_t` = `uint16_t`

12.259.1.2 vect_t

using `vect_t` = `__m128i` [inherited]

12.259.2 Member Function Documentation

12.259.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.259.2.2 set() [1/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

12.259.2.3 gather() [1/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.259.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p )  [inline], [static]
```

12.259.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p )  [inline], [static]
```

12.259.2.6 store() [1/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v )  [inline], [static]
```

12.259.2.7 storeu() [1/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v )  [inline], [static]
```

12.259.2.8 stream() [1/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v )  [inline], [static]
```

12.259.2.9 sra()

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a )  [inline], [static]
```

12.259.2.10 greater()

```
static INLINE CONST vect_t greater (
    vect_t a,
    vect_t b )  [inline], [static]
```

12.259.2.11 lesser()

```
static INLINE CONST vect_t lesser (
    vect_t a,
    vect_t b )  [inline], [static]
```

12.259.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b )  [inline], [static]
```

12.259.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b )  [inline], [static]
```

12.259.2.14 mulhi()

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.259.2.15 mulx()

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.259.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.259.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.259.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.259.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.259.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.259.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.259.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

12.259.2.23 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

12.259.2.24 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

12.259.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

12.259.2.26 set() [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static], [inherited]
```

12.259.2.27 gather() [2/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

12.259.2.28 load() [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.259.2.29 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.259.2.30 store() [2/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.259.2.31 storeu() [2/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.259.2.32 stream() [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static], [inherited]
```

12.259.2.33 sll()

```
template<int s>  
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

12.259.2.34 srl()

```
template<int s>  
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

12.259.2.35 shuffle()

```
template<uint32_t s>  
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

12.259.2.36 unpacklo()

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.37 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.38 blend()

```
template<uint8_t s>  
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.39 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.40 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.41 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.42 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.43 mullo()

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.44 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.45 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.46 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.47 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.48 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.49 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.50 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```


12.259.2.51 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.52 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

12.259.2.53 mod()

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const __m64 & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

12.259.2.54 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.259.2.55 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.259.2.56 sll128()

```
template<uint8_t s>  
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```

12.259.2.57 srl128()

```
template<uint8_t s>  
static INLINE CONST vect_t srl128 (  
    const vect_t a ) [inline], [static], [inherited]
```

12.259.2.58 vand()

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.59 vor()

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.60 vxor()

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.259.2.61 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.259.3 Field Documentation

12.259.3.1 vect_size

```
const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

12.259.3.2 alignment

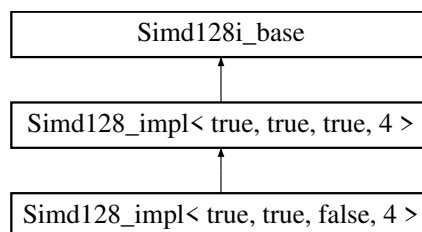
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128_int16.inl](#)

12.260 Simd128_impl< true, true, false, 4 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, false, 4 >:



Data Structures

- union [Converter](#)

Public Types

- using [scalar_t](#) = [uint32_t](#)
- using [vect_t](#) = [__m128i](#)

Static Public Member Functions

- static **INLINE** **CONST** [vect_t](#) set1 (const [scalar_t](#) x)
- static **INLINE** **CONST** [vect_t](#) set (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3)
- template<class T >
 - static **INLINE** **PURE** [vect_t](#) gather (const [scalar_t](#) *const p, const T *const idx)
- static **INLINE** **PURE** [vect_t](#) load (const [scalar_t](#) *const p)
- static **INLINE** **PURE** [vect_t](#) loadu (const [scalar_t](#) *const p)
- static **INLINE** void store ([scalar_t](#) *p, [vect_t](#) v)
- static **INLINE** void storeu ([scalar_t](#) *p, [vect_t](#) v)
- static **INLINE** void stream ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
 - static **INLINE** **CONST** [vect_t](#) sra (const [vect_t](#) a)
- static **INLINE** **CONST** [vect_t](#) greater ([vect_t](#) a, [vect_t](#) b)
- static **INLINE** **CONST** [vect_t](#) lesser ([vect_t](#) a, [vect_t](#) b)
- static **INLINE** **CONST** [vect_t](#) greater_eq (const [vect_t](#) a, const [vect_t](#) b)
- static **INLINE** **CONST** [vect_t](#) lesser_eq (const [vect_t](#) a, const [vect_t](#) b)

- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >
static constexpr bool `valid` (T *p)
- template<class T >
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3)
- template<class T >
static `INLINE PURE vect_t gather` (const `scalar_t` *const p, const T *const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` *const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` *const p)
- static `INLINE void store` (`scalar_t` *p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` *p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` *p, const `vect_t` v)
- template<int s>
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8_t s>
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- template<uint8_t s>
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- template<uint8_t s>
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- template<uint8_t s>
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

Static Public Attributes

- static const constexpr size_t `vect_size` = 4
- static const constexpr size_t `alignment` = 16

12.260.1 Member Typedef Documentation

12.260.1.1 `scalar_t`

```
using scalar_t = uint32_t
```

12.260.1.2 `vect_t`

```
using vect_t = __m128i [inherited]
```

12.260.2 Member Function Documentation

12.260.2.1 `set1()` [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.260.2.2 `set()` [1/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

12.260.2.3 `gather()` [1/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.260.2.4 `load()` [1/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.260.2.5 `loadu()` [1/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.260.2.6 `store()` [1/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.260.2.7 `storeu()` [1/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.260.2.8 stream() [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.260.2.9 sra()

```
template<int s>  
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

12.260.2.10 greater()

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

12.260.2.11 lesser()

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

12.260.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.260.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.260.2.14 mulhi()

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.260.2.15 mulx()

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.260.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.260.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.260.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.260.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.260.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.260.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.260.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.260.2.23 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

12.260.2.24 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

12.260.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

12.260.2.26 set() [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static], [inherited]
```

12.260.2.27 gather() [2/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

12.260.2.28 load() [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.260.2.29 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.260.2.30 store() [2/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.260.2.31 storeu() [2/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.260.2.32 stream() [2/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static], [inherited]
```

12.260.2.33 sll()

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

12.260.2.34 srl()

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

12.260.2.35 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

12.260.2.36 unpacklo()

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.37 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.38 blend()

```
template<uint8_t s>  
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.39 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.40 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.41 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.42 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.43 mullo()

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.44 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.45 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```


12.260.2.46 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.47 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.48 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.49 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.50 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.51 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.52 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

12.260.2.53 mod()

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVp,  
    const vect_t & NEGp,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

12.260.2.54 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.260.2.55 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.260.2.56 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

12.260.2.57 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

12.260.2.58 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.59 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.60 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.260.2.61 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.260.3 Field Documentation**12.260.3.1 vect_size**

```
const constexpr size_t vect_size = 4 [static], [constexpr], [inherited]
```

12.260.3.2 alignment

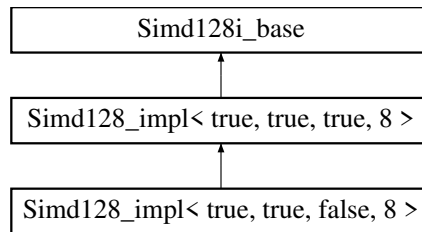
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128_int32.inl](#)

12.261 Simd128_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, false, 8 >:



Data Structures

- union [Converter](#)

Public Types

- using [scalar_t](#) = [uint64_t](#)
- using [vect_t](#) = [__m128i](#)

Static Public Member Functions

- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE void store](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE void storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE void stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
static [INLINE CONST vect_t sra](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t greater](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t lesser](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t greater_eq](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t lesser_eq](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mullo](#) (const [vect_t](#) x0, const [vect_t](#) x1)
- static [INLINE CONST vect_t mulx](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmaddx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmaddxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fnmaddx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fnmaddxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmsubx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmsubxin](#) ([vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST scalar_t hadd_to_scal](#) (const [vect_t](#) a)
- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)

- `template<int idx>`
static `INLINE CONST scalar_t get (vect_t v)`
- static `INLINE PURE vect_t load (const scalar_t *const p)`
- static `INLINE PURE vect_t loadu (const scalar_t *const p)`
- static `INLINE void store (scalar_t *p, vect_t v)`
- static `INLINE void storeu (scalar_t *p, vect_t v)`
- static `INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`
static `INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`
static `INLINE CONST vect_t srl (const vect_t a)`
- `template<uint8_t s>`
static `INLINE CONST vect_t shuffle (const vect_t a)`
- static `INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `template<uint8_t s>`
static `INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- static `INLINE vect_t addin (vect_t &a, const vect_t b)`
- static `INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- static `INLINE vect_t subin (vect_t &a, const vect_t b)`
- static `INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE CONST vect_t mask_high ()`
- static `INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- static `INLINE vect_t mod (vect_t &C, const __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)`
- static `const std::string type_string ()`
- static `INLINE CONST vect_t zero ()`
- `template<uint8_t s>`
static `INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`
static `INLINE CONST vect_t srl128 (const vect_t a)`
- static `INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

Static Public Attributes

- static `const constexpr size_t vect_size = 2`
- static `const constexpr size_t alignment = 16`

Static Protected Member Functions

- static `INLINE CONST vect_t signbits (const vect_t x)`

12.261.1 Member Typedef Documentation

12.261.1.1 scalar_t

```
using scalar_t = uint64_t
```

12.261.1.2 vect_t

```
using vect_t = __m128i [inherited]
```

12.261.2 Member Function Documentation

12.261.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

12.261.2.2 set() [1/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1 ) [inline], [static]
```

12.261.2.3 gather() [1/2]

```
template<class T >  
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

12.261.2.4 load() [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

12.261.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

12.261.2.6 store() [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

12.261.2.7 storeu() [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

12.261.2.8 stream() [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.261.2.9 sra()

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.261.2.10 greater()

```
static INLINE CONST vect_t greater (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.261.2.11 lesser()

```
static INLINE CONST vect_t lesser (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.261.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.14 mullo()

```
static INLINE CONST vect_t mullo (
    const vect_t x0,
    const vect_t x1 ) [inline], [static]
```

12.261.2.15 mulx()

```
static INLINE CONST vect_t mulx (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.21 fmsubxin() [1/2]

```
static INLINE CONST vect_t fmsubxin (
    vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.261.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.261.2.23 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

12.261.2.24 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

12.261.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

12.261.2.26 set() [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1 ) [inline], [static], [inherited]
```

12.261.2.27 gather() [2/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

12.261.2.28 get()

```
template<int idx>
static INLINE CONST scalar_t get (
    vect_t v ) [inline], [static], [inherited]
```

12.261.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.261.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.261.2.31 store() [2/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.261.2.32 storeu() [2/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.261.2.33 stream() [2/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static], [inherited]
```

12.261.2.34 sll()

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

12.261.2.35 srl()

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

12.261.2.36 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```


12.261.2.37 unpacklo()

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.38 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.39 blend()

```
template<uint8_t s>  
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.40 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.41 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.42 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.43 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.44 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.45 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.46 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.47 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.48 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.49 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.50 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.51 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.52 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.261.2.53 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

12.261.2.54 mask_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

12.261.2.55 mulhi_fast()

```

INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]

```

12.261.2.56 mod()

```

INLINE vect_t mod (
    vect_t & C,
    const __m128d & P,
    const __m128d & INV_P,
    const __m128d & NEG_P,
    const vect_t & POW50REM,
    const __m128d & MIN,
    const __m128d & MAX,
    __m128d & Q,
    __m128d & T ) [static], [inherited]

```

12.261.2.57 signbits()

```

static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected], [inherited]

```

12.261.2.58 type_string()

```

static const std::string type_string ( ) [inline], [static], [inherited]

```

12.261.2.59 zero()

```

static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]

```

12.261.2.60 sll128()

```

template<uint8_t s>
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]

```

12.261.2.61 srl128()

```

template<uint8_t s>
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]

```

12.261.2.62 vand()

```

static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

12.261.2.63 vor()

```

static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

12.261.2.64 vxor()

```

static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

12.261.2.65 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.261.3 Field Documentation

12.261.3.1 vect_size

```
const constexpr size_t vect_size = 2 [static], [constexpr], [inherited]
```

12.261.3.2 alignment

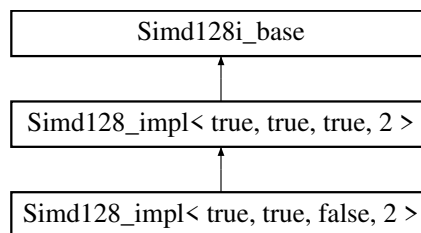
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128_int64.inl](#)

12.262 Simd128_impl< true, true, true, 2 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, true, 2 >:



Data Structures

- union [Converter](#)

Public Types

- using [vect_t](#) = __m128i
- using [scalar_t](#) = int16_t

Static Public Member Functions

- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)
- static **INLINE** **CONST** [vect_t](#) [set1](#) (const [scalar_t](#) x)
- static **INLINE** **CONST** [vect_t](#) [set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3, const [scalar_t](#) x4, const [scalar_t](#) x5, const [scalar_t](#) x6, const [scalar_t](#) x7)
- template<class T >
static **INLINE** **PURE** [vect_t](#) [gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static **INLINE** **PURE** [vect_t](#) [load](#) (const [scalar_t](#) *const p)
- static **INLINE** **PURE** [vect_t](#) [loadu](#) (const [scalar_t](#) *const p)
- static **INLINE** void [store](#) ([scalar_t](#) *p, [vect_t](#) v)
- static **INLINE** void [storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static **INLINE** void [stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
static **INLINE** **CONST** [vect_t](#) [sll](#) (const [vect_t](#) a)

- `template<int s>`
static `INLINE CONST vect_t srl` (const `vect_t` a)
- `template<int s>`
static `INLINE CONST vect_t sra` (const `vect_t` a)
- `template<uint32_t s>`
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- `template<uint8_t s>`
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `__m64` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- `template<uint8_t s>`
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- `template<uint8_t s>`
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

Static Public Attributes

- static const constexpr size_t `vect_size` = 8
- static const constexpr size_t `alignment` = 16

12.262.1 Member Typedef Documentation

12.262.1.1 vect_t

```
using vect_t = __m128i
```

12.262.1.2 scalar_t

```
using scalar_t = int16_t
```

12.262.2 Member Function Documentation

12.262.2.1 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

12.262.2.2 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.262.2.3 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.262.2.4 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

12.262.2.5 gather()

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.262.2.6 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.262.2.7 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.262.2.8 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.262.2.9 storeu()

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.262.2.10 stream()

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static]
```

12.262.2.11 sll()

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static]
```

12.262.2.12 srl()

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static]
```

12.262.2.13 sra()

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.262.2.14 shuffle()

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static]
```

12.262.2.15 unpacklo()

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.16 unpackhi()

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.17 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.18 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.19 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.262.2.20 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.21 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.262.2.22 mullo()

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.23 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.24 mulhi()

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.25 mulx()

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.26 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.27 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```


12.262.2.28 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.29 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.30 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.31 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.32 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.33 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.34 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.35 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.262.2.36 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,
```

```
const vect_t a,
const vect_t b ) [inline], [static]
```

12.262.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.38 eq()

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.39 greater()

```
static INLINE CONST vect_t greater (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.40 lesser()

```
static INLINE CONST vect_t lesser (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.41 greater_eq()

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.42 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.262.2.43 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.262.2.44 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

12.262.2.45 mod()

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const __m64 & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
```

```

    vect_t & Q,
    vect_t & T ) [inline], [static]

```

12.262.2.46 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.262.2.47 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.262.2.48 sll128()

```

template<uint8_t s>
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]

```

12.262.2.49 srl128()

```

template<uint8_t s>
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]

```

12.262.2.50 vand()

```

static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

12.262.2.51 vor()

```

static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

12.262.2.52 vxor()

```

static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

12.262.2.53 vandnot()

```

static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

12.262.3 Field Documentation

12.262.3.1 vect_size

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```

12.262.3.2 alignment

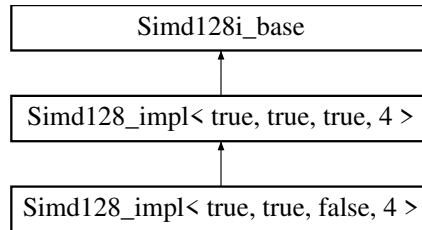
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128_int16.inl](#)

12.263 Simd128_impl< true, true, true, 4 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, true, 4 >:



Data Structures

- union [Converter](#)

Public Types

- using [vect_t](#) = __m128i
- using [scalar_t](#) = int32_t

Static Public Member Functions

- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE](#) void [store](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
static [INLINE CONST vect_t sll](#) (const [vect_t](#) a)
- template<int s>
static [INLINE CONST vect_t srl](#) (const [vect_t](#) a)
- template<int s>
static [INLINE CONST vect_t sra](#) (const [vect_t](#) a)
- template<uint8_t s>
static [INLINE CONST vect_t shuffle](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t unpacklo](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpackhi](#) (const [vect_t](#) a, const [vect_t](#) b)
- template<uint8_t s>
static [INLINE CONST vect_t blend](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t add](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE](#) [vect_t addin](#) ([vect_t](#) &a, const [vect_t](#) b)
- static [INLINE CONST vect_t sub](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE](#) [vect_t subin](#) ([vect_t](#) &a, const [vect_t](#) b)
- static [INLINE CONST vect_t mullo](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mul](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mulhi](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mulx](#) (const [vect_t](#) a, const [vect_t](#) b)

- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- template<uint8_t s>
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- template<uint8_t s>
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

Static Public Attributes

- static const constexpr size_t `vect_size` = 4
- static const constexpr size_t `alignment` = 16

12.263.1 Member Typedef Documentation

12.263.1.1 vect_t

```
using vect_t = __m128i
```

12.263.1.2 scalar_t

```
using scalar_t = int32_t
```

12.263.2 Member Function Documentation

12.263.2.1 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

12.263.2.2 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.263.2.3 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.263.2.4 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

12.263.2.5 gather()

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.263.2.6 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.263.2.7 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.263.2.8 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.263.2.9 storeu()

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.263.2.10 stream()

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static]
```

12.263.2.11 sll()

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static]
```

12.263.2.12 srl()

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static]
```

12.263.2.13 sra()

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.263.2.14 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static]
```

12.263.2.15 unpacklo()

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.16 unpackhi()

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.17 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.18 add()

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.19 addin()

```
static INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.263.2.20 sub()

```
static INLINE CONST vect_t sub (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.21 subin()

```
static INLINE vect_t subin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.263.2.22 mullo()

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.263.2.23 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.263.2.24 mulhi()

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.263.2.25 mulx()

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.263.2.26 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.263.2.27 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.263.2.28 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.263.2.29 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.263.2.30 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```


12.263.2.31 fnmaddin()

```
static INLINE vect_t fnmaddin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.32 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.33 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.34 fmsub()

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.35 fmsubin()

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.36 fmsubx()

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.38 eq()

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.39 greater()

```
static INLINE CONST vect_t greater (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.40 lesser()

```
static INLINE CONST vect_t lesser (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.41 greater_eq()

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.42 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.263.2.43 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.263.2.44 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

12.263.2.45 mod()

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

12.263.2.46 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.263.2.47 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.263.2.48 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

12.263.2.49 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

12.263.2.50 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.263.2.51 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.263.2.52 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.263.2.53 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.263.3 Field Documentation**12.263.3.1 vect_size**

```
const constexpr size_t vect_size = 4 [static], [constexpr]
```

12.263.3.2 alignment

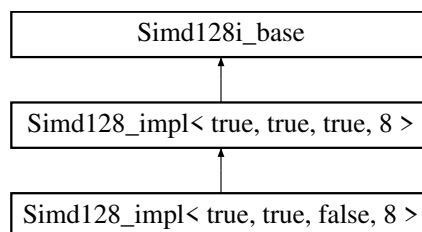
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128_int32.inl](#)

12.264 Simd128_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd128_impl< true, true, true, 8 >:

**Data Structures**

- union [Converter](#)

Public Types

- using [vect_t](#) = __m128i
- using [scalar_t](#) = int64_t

Static Public Member Functions

- `template<class T >`
`static constexpr bool valid (T *p)`
- `template<class T >`
`static constexpr bool compliant (T n)`
- `static INLINE CONST vect_t set1 (const scalar_t x)`
- `static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1)`
- `template<class T >`
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `template<int idx>`
`static INLINE CONST scalar_t get (vect_t v)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`
`static INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<int s>`
`static INLINE CONST vect_t sra (const vect_t a)`
- `template<uint8_t s>`
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `template<uint8_t s>`
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t x0, const vect_t x1)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE CONST vect_t mask_high ()`

- static `INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- static `INLINE vect_t mod (vect_t &C, const __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)`
- static const std::string `type_string ()`
- static `INLINE CONST vect_t zero ()`
- `template<uint8_t s>`
static `INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`
static `INLINE CONST vect_t srl128 (const vect_t a)`
- static `INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

Static Public Attributes

- static const constexpr size_t `vect_size` = 2
- static const constexpr size_t `alignment` = 16

Static Protected Member Functions

- static `INLINE CONST vect_t signbits (const vect_t x)`

12.264.1 Member Typedef Documentation

12.264.1.1 vect_t

```
using vect_t = __m128i
```

12.264.1.2 scalar_t

```
using scalar_t = int64_t
```

12.264.2 Member Function Documentation

12.264.2.1 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

12.264.2.2 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.264.2.3 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.264.2.4 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1 ) [inline], [static]
```

12.264.2.5 gather()

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.264.2.6 get()

```
template<int idx>
static INLINE CONST scalar_t get (
    vect_t v ) [inline], [static]
```

12.264.2.7 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.264.2.8 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.264.2.9 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.264.2.10 storeu()

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.264.2.11 stream()

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static]
```

12.264.2.12 sll()

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static]
```

12.264.2.13 srl()

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static]
```

12.264.2.14 sra()

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.264.2.15 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static]
```

12.264.2.16 unpacklo()

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.264.2.17 unpackhi()

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.264.2.18 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.264.2.19 add()

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.264.2.20 addin()

```
static INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.264.2.21 sub()

```
static INLINE CONST vect_t sub (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.264.2.22 subin()

```
static INLINE vect_t subin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.264.2.23 mullo()

```
static INLINE CONST vect_t mullo (
    const vect_t x0,
    const vect_t x1 ) [inline], [static]
```

12.264.2.24 mul()

```
static INLINE CONST vect_t mul (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.264.2.25 mulx()

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.26 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.27 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.28 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.29 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.30 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.31 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.32 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.33 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,
```



```
const vect_t a,  
const vect_t b ) [inline], [static]
```

12.264.2.34 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.35 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.36 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.38 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.39 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.40 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.41 greater_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.42 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.264.2.43 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.264.2.44 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

12.264.2.45 mask_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

12.264.2.46 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]
```

12.264.2.47 mod()

```
INLINE vect_t mod (
    vect_t & C,
    const __m128d & P,
    const __m128d & INV_P,
    const __m128d & NEG_P,
    const vect_t & POW50REM,
    const __m128d & MIN,
    const __m128d & MAX,
    __m128d & Q,
    __m128d & T ) [static]
```

12.264.2.48 signbits()

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]
```

12.264.2.49 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.264.2.50 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.264.2.51 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

12.264.2.52 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

12.264.2.53 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.264.2.54 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.264.2.55 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.264.2.56 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.264.3 Field Documentation**12.264.3.1 vect_size**

```
const constexpr size_t vect_size = 2 [static], [constexpr]
```

12.264.3.2 alignment

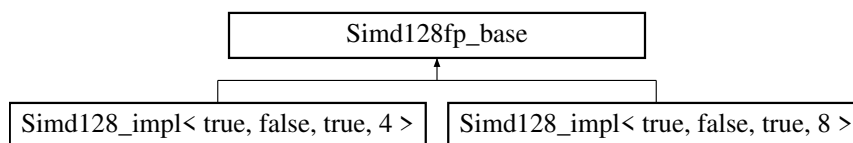
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128_int64.inl](#)

12.265 Simd128fp_base Struct Reference

Inheritance diagram for Simd128fp_base:

**Static Public Member Functions**

- static const std::string [type_string](#) ()

12.265.1 Member Function Documentation**12.265.1.1 type_string()**

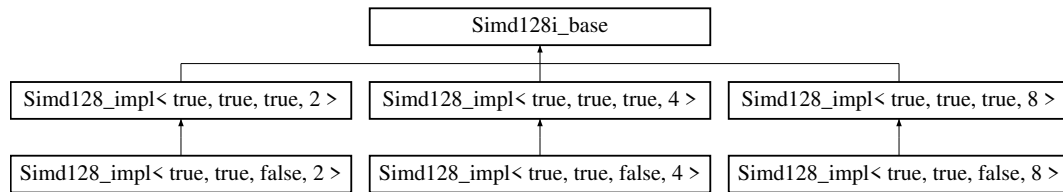
```
static const std::string type_string ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

12.266 Simd128i_base Struct Reference

Inheritance diagram for Simd128i_base:



Public Types

- using `vect_t` = `__m128i`

Static Public Member Functions

- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- template<uint8_t s>
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- template<uint8_t s>
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

12.266.1 Member Typedef Documentation

12.266.1.1 `vect_t`

using `vect_t` = `__m128i`

12.266.2 Member Function Documentation

12.266.2.1 `type_string()`

```
static const std::string type_string ( ) [inline], [static]
```

12.266.2.2 `zero()`

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

12.266.2.3 `sll128()`

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static]
```

12.266.2.4 `srl128()`

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static]
```

12.266.2.5 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.266.2.6 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.266.2.7 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.266.2.8 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

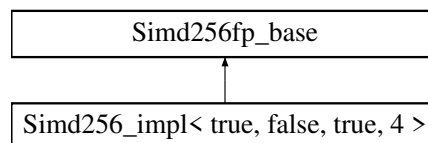
12.267 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd256.inl](#)

12.268 Simd256_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd256_impl< true, false, true, 4 >:

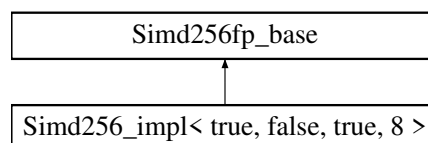


The documentation for this struct was generated from the following file:

- [simd256_float.inl](#)

12.269 Simd256_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd256_impl< true, false, true, 8 >:



Public Types

- using `vect_t` = `__m256d`
- using `scalar_t` = `double`

Static Public Member Functions

- `template<class T >`
static constexpr bool `valid` (T *p)
- `template<class T >`
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4)
- `template<class T >`
static `INLINE PURE vect_t gather` (const `scalar_t` *const p, const T *const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` *const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` *const p)
- static `INLINE void store` (const `scalar_t` *p, const `vect_t` v)
- static `INLINE void storeu` (const `scalar_t` *p, const `vect_t` v)
- static `INLINE void stream` (const `scalar_t` *p, const `vect_t` v)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- `template<uint8_t s>`
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t blendv` (const `vect_t` a, const `vect_t` b, const `vect_t` mask)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t div` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t floor` (const `vect_t` a)
- static `INLINE CONST vect_t ceil` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t hadd` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)

Static Public Attributes

- static const constexpr size_t `vect_size` = 4
- static const constexpr size_t `alignment` = 32

12.269.1 Member Typedef Documentation

12.269.1.1 `vect_t`

```
using vect_t = __m256d
```

12.269.1.2 `scalar_t`

```
using scalar_t = double
```

12.269.2 Member Function Documentation

12.269.2.1 `valid()`

```
template<class T >  
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

12.269.2.2 `compliant()`

```
template<class T >  
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

12.269.2.3 `zero()`

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

12.269.2.4 `set1()`

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

12.269.2.5 `set()`

```
static INLINE CONST vect_t set (  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4 ) [inline], [static]
```

12.269.2.6 `gather()`

```
template<class T >  
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

12.269.2.7 `load()`

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

12.269.2.8 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

12.269.2.9 store()

```
static INLINE void store (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.269.2.10 storeu()

```
static INLINE void storeu (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.269.2.11 stream()

```
static INLINE void stream (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.269.2.12 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.13 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.14 blend()

```
template<uint8_t s>  
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.15 blendv()

```
static INLINE CONST vect_t blendv (  
    const vect_t a,  
    const vect_t b,  
    const vect_t mask ) [inline], [static]
```

12.269.2.16 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.17 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```


12.269.2.18 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.19 subin()

```
static INLINE CONST vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.269.2.20 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.21 mulin()

```
static INLINE CONST vect_t mulin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.269.2.22 div()

```
static INLINE CONST vect_t div (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.23 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.24 fmaddin()

```
static INLINE CONST vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.25 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.26 fnmaddin()

```
static INLINE CONST vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.27 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.28 fmsubin()

```
static INLINE CONST vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.29 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.30 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.31 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.32 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.33 greater_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.34 vand()

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.35 vor()

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.36 vxor()

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.269.2.37 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.269.2.38 floor()

```
static INLINE CONST vect_t floor (
    const vect_t a ) [inline], [static]
```

12.269.2.39 ceil()

```
static INLINE CONST vect_t ceil (
    const vect_t a ) [inline], [static]
```

12.269.2.40 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

12.269.2.41 hadd()

```
static INLINE CONST vect_t hadd (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.269.2.42 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.269.2.43 mod()

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

12.269.3 Field Documentation**12.269.3.1 vect_size**

```
const constexpr size_t vect_size = 4 [static], [constexpr]
```

12.269.3.2 alignment

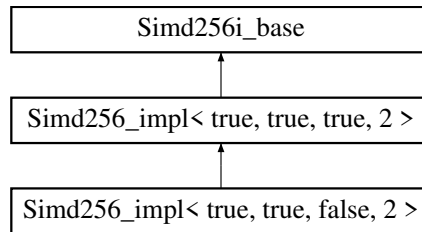
```
const constexpr size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd256_double.inl](#)

12.270 Simd256_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, false, 2 >:



Data Structures

- union [Converter](#)

Public Types

- using [scalar_t](#) = [uint16_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar_t](#) >
- using [vect_t](#) = [__m256i](#)
- using [half_t](#) = [__m128i](#)

Static Public Member Functions

- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3, const [scalar_t](#) x4, const [scalar_t](#) x5, const [scalar_t](#) x6, const [scalar_t](#) x7, const [scalar_t](#) x8, const [scalar_t](#) x9, const [scalar_t](#) x10, const [scalar_t](#) x11, const [scalar_t](#) x12, const [scalar_t](#) x13, const [scalar_t](#) x14, const [scalar_t](#) x15)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE void store](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE void storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE void stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
static [INLINE CONST vect_t sra](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t greater](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t lesser](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t greater_eq](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t lesser_eq](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mulhi](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mulx](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t fmaddx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmaddxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fnmaddx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fnmaddxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmsubx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmsubxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST scalar_t hadd_to_scal](#) (const [vect_t](#) a)
- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)

- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8, const `scalar_t` x9, const `scalar_t` x10, const `scalar_t` x11, const `scalar_t` x12, const `scalar_t` x13, const `scalar_t` x14, const `scalar_t` x15)
- template<class T >
static `INLINE PURE vect_t gather` (const `scalar_t` *const p, const T *const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` *const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` *const p)
- static `INLINE void store` (`scalar_t` *p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` *p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` *p, const `vect_t` v)
- template<int s>
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint64_t s>
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &s1, `vect_t` &s2, const `vect_t` a, const `vect_t` b)
- template<uint8_t s>
static `INLINE CONST vect_t blend_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

Static Public Attributes

- static const constexpr size_t `vect_size` = 16
- static const constexpr size_t `alignment` = 32

12.270.1 Member Typedef Documentation

12.270.1.1 scalar_t

```
using scalar_t = uint16_t
```

12.270.1.2 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.270.1.3 vect_t

```
using vect_t = __m256i [inherited]
```

12.270.1.4 half_t

```
using half_t = __m128i [inherited]
```

12.270.2 Member Function Documentation

12.270.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.270.2.2 set() [1/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static]
```

12.270.2.3 gather() [1/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.270.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.270.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.270.2.6 store() [1/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.270.2.7 storeu() [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

12.270.2.8 stream() [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.270.2.9 sra()

```
template<int s>  
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

12.270.2.10 greater()

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

12.270.2.11 lesser()

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

12.270.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.270.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.270.2.14 mulhi()

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.270.2.15 mulx()

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

12.270.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.270.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.270.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.270.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.270.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.270.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.270.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.270.2.23 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

12.270.2.24 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

12.270.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```


12.270.2.26 set() [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static], [inherited]
```

12.270.2.27 gather() [2/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

12.270.2.28 load() [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.270.2.29 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.270.2.30 store() [2/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.270.2.31 storeu() [2/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.270.2.32 stream() [2/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static], [inherited]
```

12.270.2.33 sll()

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

12.270.2.34 srl()

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

12.270.2.35 shuffle()

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

12.270.2.36 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.37 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.38 unpacklo()

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.39 unpackhi()

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.40 unpacklohi()

```
static INLINE void unpacklohi (
    vect_t & s1,
    vect_t & s2,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.41 blend_twice()

```
template<uint8_t s>
static INLINE CONST vect_t blend_twice (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.42 add()

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.43 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.44 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.45 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.46 mullo()

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.47 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.48 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.49 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.50 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.51 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.52 fmsub()

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.53 fmsubin()

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.54 eq()

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.270.2.55 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

12.270.2.56 mod()

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]
```

12.270.2.57 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.270.2.58 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.270.3 Field Documentation**12.270.3.1 vect_size**

```
const constexpr size_t vect_size = 16 [static], [constexpr], [inherited]
```

12.270.3.2 alignment

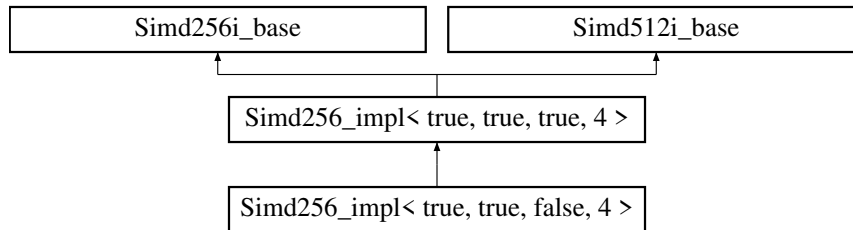
```
const constexpr size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd256_int16.inl](#)

12.271 Simd256_impl< true, true, false, 4 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, false, 4 >:



Data Structures

- union [Converter](#)

Public Types

- using `scalar_t` = `uint32_t`
- using `simdHalf` = `Simd128< scalar_t >`
- using `scalar_t` = `uint32_t`
- using `simdHalf` = `Simd128< scalar_t >`
- using `vect_t` = `__m256i`
- using `vect_t` = `__m512i`
- using `half_t` = `m128i`
- using `half_t` = `m256i`

Static Public Member Functions

- static `INLINE CONST vect_t set1` (`const scalar_t x`)
- static `INLINE CONST vect_t set` (`const scalar_t x0`, `const scalar_t x1`, `const scalar_t x2`, `const scalar_t x3`, `const scalar_t x4`, `const scalar_t x5`, `const scalar_t x6`, `const scalar_t x7`)
- `template<class T >`
static `INLINE PURE vect_t gather` (`const scalar_t *const p`, `const T *const idx`)
- static `INLINE PURE vect_t load` (`const scalar_t *const p`)
- static `INLINE PURE vect_t loadu` (`const scalar_t *const p`)
- static `INLINE void store` (`scalar_t *p`, `vect_t v`)
- static `INLINE void storeu` (`scalar_t *p`, `vect_t v`)
- static `INLINE void stream` (`scalar_t *p`, `const vect_t v`)
- `template<int s>`
static `INLINE CONST vect_t sra` (`const vect_t a`)
- static `INLINE CONST vect_t greater` (`vect_t a`, `vect_t b`)
- static `INLINE CONST vect_t lesser` (`vect_t a`, `vect_t b`)
- static `INLINE CONST vect_t greater_eq` (`const vect_t a`, `const vect_t b`)
- static `INLINE CONST vect_t lesser_eq` (`const vect_t a`, `const vect_t b`)
- static `INLINE CONST vect_t mulhi` (`const vect_t a`, `const vect_t b`)
- static `INLINE CONST vect_t mulx` (`vect_t a`, `vect_t b`)
- static `INLINE CONST vect_t fmaddx` (`const vect_t c`, `const vect_t a`, `const vect_t b`)
- static `INLINE vect_t fmaddxin` (`vect_t &c`, `const vect_t a`, `const vect_t b`)
- static `INLINE CONST vect_t fnmaddx` (`const vect_t c`, `const vect_t a`, `const vect_t b`)
- static `INLINE vect_t fnmaddxin` (`vect_t &c`, `const vect_t a`, `const vect_t b`)
- static `INLINE CONST vect_t fmsubx` (`const vect_t c`, `const vect_t a`, `const vect_t b`)
- static `INLINE vect_t fmsubxin` (`vect_t &c`, `const vect_t a`, `const vect_t b`)
- static `INLINE CONST scalar_t hadd_to_scal` (`const vect_t a`)
- static `INLINE CONST vect_t set1` (`const scalar_t x`)

- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- template<class T >
 - static `INLINE PURE vect_t gather` (const `scalar_t` *const p, const T *const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` *const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` *const p)
- static `INLINE void store` (`scalar_t` *p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` *p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` *p, const `vect_t` v)
- template<int s>
 - static `INLINE CONST vect_t sra` (const `vect_t` a)
 - static `INLINE CONST vect_t greater` (`vect_t` a, `vect_t` b)
 - static `INLINE CONST vect_t lesser` (`vect_t` a, `vect_t` b)
 - static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
 - static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
 - static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
 - static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
 - static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
 - static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
 - static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
 - static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
 - static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
 - static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
 - static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >
 - static constexpr bool `valid` (T *p)
- template<class T >
 - static constexpr bool `valid` (T *p)
- template<class T >
 - static constexpr bool `compliant` (T n)
- template<class T >
 - static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8, const `scalar_t` x9, const `scalar_t` x10, const `scalar_t` x11, const `scalar_t` x12, const `scalar_t` x13, const `scalar_t` x14, const `scalar_t` x15)
- template<class T >
 - static `INLINE PURE vect_t gather` (const `scalar_t` *const p, const T *const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` *const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` *const p)
- static `INLINE void store` (`scalar_t` *p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` *p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` *p, const `vect_t` v)
- template<int s>
 - static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
 - static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
 - static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>
 - static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8_t s>
 - static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)

- `template<uint8_t s>`
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- `template<uint32_t s>`
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- `template<uint64_t s>`
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &s1, `vect_t` &s2, const `vect_t` a, const `vect_t` b)
- `template<uint8_t s>`
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

Static Public Attributes

- static const constexpr size_t `vect_size` = 8
- static const constexpr size_t `alignment` = 32

12.271.1 Member Typedef Documentation

12.271.1.1 `scalar_t` [1/2]

```
using scalar_t = uint32_t
```

12.271.1.2 `simdHalf` [1/2]

```
using simdHalf = Simd128<scalar_t>
```

12.271.1.3 `scalar_t` [2/2]

```
using scalar_t = uint32_t
```

12.271.1.4 `simdHalf` [2/2]

```
using simdHalf = Simd128<scalar_t>
```

12.271.1.5 `vect_t` [1/2]

```
using vect_t = __m256i [inherited]
```

12.271.1.6 `vect_t` [2/2]

```
using vect_t = __m512i [inherited]
```

12.271.1.7 `half_t` [1/2]

```
using half_t = __m128i [inherited]
```

12.271.1.8 `half_t` [2/2]

```
using half_t = __m256i [inherited]
```

12.271.2 Member Function Documentation

12.271.2.1 `set1()` [1/3]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.271.2.2 `set()` [1/4]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```


12.271.2.3 gather() [1/3]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.271.2.4 load() [1/3]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.271.2.5 loadu() [1/3]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.271.2.6 store() [1/3]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.271.2.7 storeu() [1/3]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.271.2.8 stream() [1/3]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static]
```

12.271.2.9 sra() [1/2]

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.271.2.10 greater() [1/2]

```
static INLINE CONST vect_t greater (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.271.2.11 lesser() [1/2]

```
static INLINE CONST vect_t lesser (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.271.2.12 greater_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.13 lesser_eq() [1/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.271.2.14 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.271.2.15 mulx() [1/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

12.271.2.16 fmaddx() [1/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.271.2.17 fmaddxin() [1/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.271.2.18 fnmaddx() [1/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.271.2.19 fnmaddxin() [1/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.271.2.20 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.271.2.21 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.271.2.22 hadd_to_scal() [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.271.2.23 set1() [2/3]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.271.2.24 set() [2/4]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

12.271.2.25 gather() [2/3]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.271.2.26 load() [2/3]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.271.2.27 loadu() [2/3]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.271.2.28 store() [2/3]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.271.2.29 storeu() [2/3]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.271.2.30 stream() [2/3]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static]
```

12.271.2.31 sra() [2/2]

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.271.2.32 greater() [2/2]

```
static INLINE CONST vect_t greater (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.271.2.33 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.271.2.34 greater_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.35 lesser_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.36 mulhi() [2/2]

```
static INLINE CONST vect_t mulhi (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.37 mulx() [2/2]

```
static INLINE CONST vect_t mulx (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.271.2.38 fmaddx() [2/2]

```
static INLINE CONST vect_t fmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.39 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.40 fmmaddx() [2/2]

```
static INLINE CONST vect_t fmmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.41 fmmaddxin() [2/2]

```
static INLINE vect_t fmmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.42 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.43 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.271.2.44 hadd_to_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.271.2.45 valid() [1/2]

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

12.271.2.46 valid() [2/2]

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

12.271.2.47 compliant() [1/2]

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

12.271.2.48 compliant() [2/2]

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

12.271.2.49 set1() [3/3]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

12.271.2.50 set() [3/4]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static], [inherited]
```

12.271.2.51 set() [4/4]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static], [inherited]
```

12.271.2.52 gather() [3/3]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

12.271.2.53 load() [3/3]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.271.2.54 loadu() [3/3]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.271.2.55 store() [3/3]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.271.2.56 storeu() [3/3]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.271.2.57 stream() [3/3]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static], [inherited]
```

12.271.2.58 sll() [1/2]

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.59 sll() [2/2]

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.60 srl() [1/2]

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.61 srl() [2/2]

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.62 shuffle_twice() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.63 shuffle_twice() [2/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.64 shuffle() [1/2]

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.65 shuffle() [2/2]

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.66 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.67 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.68 unpacklo()

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.69 unpackhi()

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.70 unpacklohi()

```
static INLINE void unpacklohi (
    vect_t & s1,
    vect_t & s2,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.71 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.72 add() [1/2]

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.73 add() [2/2]

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.74 addin() [1/2]

```
static INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.75 addin() [2/2]

```
static INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) [inline], [static], [inherited]
```


12.271.2.76 sub() [1/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.77 sub() [2/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.78 subin() [1/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.79 subin() [2/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.80 mullo() [1/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.81 mullo() [2/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.82 mul() [1/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.83 mul() [2/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.84 fmadd() [1/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.85 fmadd() [2/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.86 fmaddin() [1/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.87 fmaddin() [2/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.88 fnmadd() [1/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.89 fnmadd() [2/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.90 fnmaddin() [1/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.91 fnmaddin() [2/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.92 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.93 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.94 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,
```

```
const vect_t a,
const vect_t b ) [inline], [static], [inherited]
```

12.271.2.95 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.96 eq() [1/2]

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.97 eq() [2/2]

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.98 round() [1/2]

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.99 round() [2/2]

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

12.271.2.100 mod() [1/2]

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]
```

12.271.2.101 mod() [2/2]

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]
```

12.271.2.102 type_string() [1/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.271.2.103 type_string() [2/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.271.2.104 zero() [1/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.271.2.105 zero() [2/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.271.2.106 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.107 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.108 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.2.109 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.271.3 Field Documentation**12.271.3.1 vect_size**

```
static const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

12.271.3.2 alignment

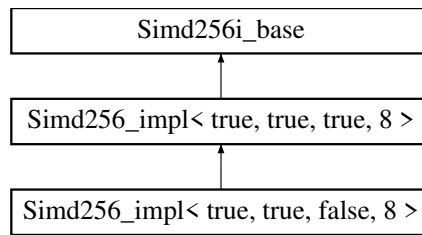
```
static const constexpr size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following files:

- [simd256_int32.inl](#)
- [simd512_int32.inl](#)

12.272 Simd256_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, false, 8 >:



Data Structures

- union [Converter](#)

Public Types

- using [scalar_t](#) = [uint64_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar_t](#) >
- using [vect_t](#) = [__m256i](#)
- using [half_t](#) = [__m128i](#)

Static Public Member Functions

- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE void store](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE void storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE void stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
static [INLINE CONST vect_t sra](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t greater](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t lesser](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t greater_eq](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t lesser_eq](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mullo](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t mulx](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmaddx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmaddxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fnmaddx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fnmaddxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmsubx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmsubxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST scalar_t hadd_to_scal](#) (const [vect_t](#) a)
- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- template<int idx>
static [INLINE CONST scalar_t get](#) ([vect_t](#) v)

- static `INLINE PURE vect_t load` (const `scalar_t` *const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` *const p)
- static `INLINE void store` (`scalar_t` *p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` *p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` *p, const `vect_t` v)
- template<int s>
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8_t s>
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &l, `vect_t` &h, const `vect_t` a, const `vect_t` b)
- template<uint8_t s>
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const __m256d &P, const __m256d &INVP, const __m256d &NEGP, const `vect_t` &POW50REM, const __m256d &MIN, const __m256d &MAX, __m256d &Q, __m256d &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

Static Public Attributes

- static const constexpr size_t `vect_size` = 4
- static const constexpr size_t `alignment` = 32

Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

12.272.1 Member Typedef Documentation

12.272.1.1 scalar_t

using `scalar_t` = `uint64_t`

12.272.1.2 simdHalf

using `simdHalf` = `Simd128<scalar_t>`

12.272.1.3 vect_t

```
using vect_t = __m256i [inherited]
```

12.272.1.4 half_t

```
using half_t = __m128i [inherited]
```

12.272.2 Member Function Documentation

12.272.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.272.2.2 set() [1/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

12.272.2.3 gather() [1/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.272.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.272.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.272.2.6 store() [1/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.272.2.7 storeu() [1/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.272.2.8 stream() [1/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static]
```

12.272.2.9 sra()

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.272.2.10 greater()

```
static INLINE CONST vect_t greater (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.272.2.11 lesser()

```
static INLINE CONST vect_t lesser (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.272.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.272.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.272.2.14 mullo()

```
static INLINE CONST vect_t mullo (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.272.2.15 mulx()

```
static INLINE CONST vect_t mulx (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.272.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.272.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```


12.272.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.272.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.272.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.272.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.272.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.272.2.23 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

12.272.2.24 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

12.272.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

12.272.2.26 set() [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static], [inherited]
```

12.272.2.27 gather() [2/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

12.272.2.28 get()

```
template<int idx>
static INLINE CONST scalar_t get (
    vect_t v ) [inline], [static], [inherited]
```

12.272.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.272.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.272.2.31 store() [2/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.272.2.32 storeu() [2/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.272.2.33 stream() [2/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static], [inherited]
```

12.272.2.34 sll()

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static], [inherited]
```

12.272.2.35 srl()

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static], [inherited]
```

12.272.2.36 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static], [inherited]
```

12.272.2.37 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.38 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.39 unpacklo()

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.40 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.41 unpacklohi()

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.42 blend()

```
template<uint8_t s>  
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.43 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.44 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.45 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.46 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.47 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.48 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.49 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.50 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.51 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.52 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.53 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.54 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.272.2.55 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

12.272.2.56 mask_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

12.272.2.57 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

12.272.2.58 mod()

```
INLINE vect_t mod (
    vect_t & C,
    const __m256d & P,
    const __m256d & INV_P,
    const __m256d & NEG_P,
    const vect_t & POW50REM,
    const __m256d & MIN,
    const __m256d & MAX,
    __m256d & Q,
    __m256d & T ) [static], [inherited]
```

12.272.2.59 signbits()

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected], [inherited]
```

12.272.2.60 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.272.2.61 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.272.3 Field Documentation**12.272.3.1 vect_size**

```
const constexpr size_t vect_size = 4 [static], [constexpr], [inherited]
```

12.272.3.2 alignment

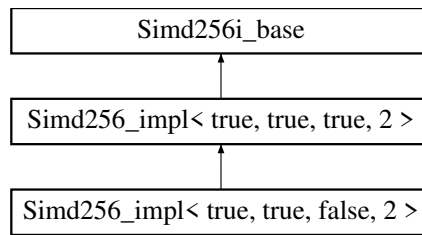
```
const constexpr size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd256_int64.inl](#)

12.273 Simd256_impl< true, true, true, 2 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, true, 2 >:



Data Structures

- union [Converter](#)

Public Types

- using [vect_t](#) = __m256i
- using [half_t](#) = __m128i
- using [scalar_t](#) = [int16_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar_t](#) >

Static Public Member Functions

- `template<class T >`
static constexpr bool [valid](#) (T *p)
- `template<class T >`
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3, const [scalar_t](#) x4, const [scalar_t](#) x5, const [scalar_t](#) x6, const [scalar_t](#) x7, const [scalar_t](#) x8, const [scalar_t](#) x9, const [scalar_t](#) x10, const [scalar_t](#) x11, const [scalar_t](#) x12, const [scalar_t](#) x13, const [scalar_t](#) x14, const [scalar_t](#) x15)
- `template<class T >`
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE](#) void [store](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- `template<int s>`
static [INLINE CONST vect_t sll](#) (const [vect_t](#) a)
- `template<int s>`
static [INLINE CONST vect_t srl](#) (const [vect_t](#) a)
- `template<int s>`
static [INLINE CONST vect_t sra](#) (const [vect_t](#) a)
- `template<uint64_t s>`
static [INLINE CONST vect_t shuffle](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t unpacklo_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpackhi_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpacklo](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpackhi](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE](#) void [unpacklohi](#) ([vect_t](#) &s1, [vect_t](#) &s2, const [vect_t](#) a, const [vect_t](#) b)
- `template<uint8_t s>`
static [INLINE CONST vect_t blend_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t add](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE](#) [vect_t](#) [addin](#) ([vect_t](#) &a, const [vect_t](#) b)
- static [INLINE CONST vect_t sub](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE](#) [vect_t](#) [subin](#) ([vect_t](#) &a, const [vect_t](#) b)

- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

Static Public Attributes

- static const constexpr size_t `vect_size` = 16
- static const constexpr size_t `alignment` = 32

12.273.1 Member Typedef Documentation

12.273.1.1 vect_t

```
using vect_t = __m256i
```

12.273.1.2 half_t

```
using half_t = __m128i
```

12.273.1.3 scalar_t

```
using scalar_t = int16_t
```

12.273.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.273.2 Member Function Documentation

12.273.2.1 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

12.273.2.2 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.273.2.3 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.273.2.4 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static]
```

12.273.2.5 gather()

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.273.2.6 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.273.2.7 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.273.2.8 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.273.2.9 storeu()

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```


12.273.2.10 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.273.2.11 sll()

```
template<int s>  
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

12.273.2.12 srl()

```
template<int s>  
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

12.273.2.13 sra()

```
template<int s>  
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

12.273.2.14 shuffle()

```
template<uint64_t s>  
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

12.273.2.15 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.16 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.17 unpacklo()

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.18 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.19 unpacklohi()

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.20 blend_twice()

```
template<uint8_t s>
static INLINE CONST vect_t blend_twice (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.273.2.21 add()

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.273.2.22 addin()

```
static INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.273.2.23 sub()

```
static INLINE CONST vect_t sub (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.273.2.24 subin()

```
static INLINE vect_t subin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.273.2.25 mullo()

```
static INLINE CONST vect_t mullo (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.273.2.26 mul()

```
static INLINE CONST vect_t mul (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.273.2.27 mulhi()

```
static INLINE CONST vect_t mulhi (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.273.2.28 mulx()

```
static INLINE CONST vect_t mulx (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.273.2.29 fmadd()

```
static INLINE CONST vect_t fmadd (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.273.2.30 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.31 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.32 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.33 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.34 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.35 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.36 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.37 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.38 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

12.273.2.39 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.40 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.41 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.42 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.43 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.44 greater_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.45 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.273.2.46 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

12.273.2.47 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

12.273.2.48 mod()

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

12.273.2.49 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.273.2.50 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.273.3 Field Documentation**12.273.3.1 vect_size**

```
const constexpr size_t vect_size = 16 [static], [constexpr]
```

12.273.3.2 alignment

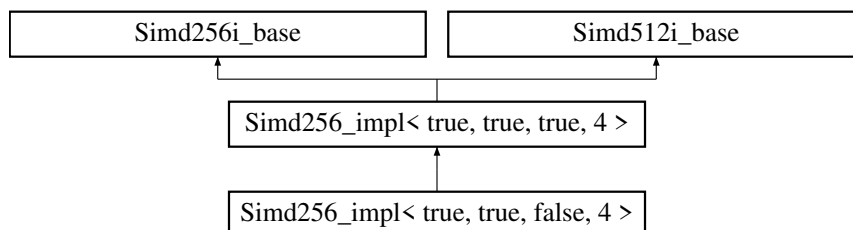
```
const constexpr size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd256_int16.inl](#)

12.274 Simd256_impl< true, true, true, 4 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, true, 4 >:

**Data Structures**

- union [Converter](#)

Public Types

- using `vect_t` = `__m256i`
- using `half_t` = `__m128i`
- using `scalar_t` = `int32_t`
- using `simdHalf` = `Simd128< scalar_t >`
- using `vect_t` = `__m512i`
- using `half_t` = `__m256i`
- using `scalar_t` = `int32_t`
- using `simdHalf` = `Simd256< scalar_t >`

Static Public Member Functions

- `template<class T >`
`static constexpr bool valid (T *p)`
- `template<class T >`
`static constexpr bool compliant (T n)`
- `static INLINE CONST vect_t set1 (const scalar_t x)`
- `static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7)`
- `template<class T >`
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`
`static INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<int s>`
`static INLINE CONST vect_t sra (const vect_t a)`
- `template<uint8_t s>`
`static INLINE CONST vect_t shuffle_twice (const vect_t a)`
- `template<uint32_t s>`
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo_twice (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi_twice (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `static INLINE void unpacklohi (vect_t &s1, vect_t &s2, const vect_t a, const vect_t b)`
- `template<uint8_t s>`
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (vect_t a, vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`

- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- template<class T >
static constexpr bool `valid` (T *p)
- template<class T >
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8, const `scalar_t` x9, const `scalar_t` x10, const `scalar_t` x11, const `scalar_t` x12, const `scalar_t` x13, const `scalar_t` x14, const `scalar_t` x15)
- template<class T >
static `INLINE PURE vect_t gather` (const `scalar_t` *const p, const T *const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` *const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` *const p)
- static `INLINE void store` (`scalar_t` *p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` *p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` *p, const `vect_t` v)
- template<int s>
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>
static `INLINE CONST vect_t sra` (const `vect_t` a)
- template<uint8_t s>
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint64_t s>
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)

- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

Static Public Attributes

- static const constexpr size_t `vect_size` = 8
- static const constexpr size_t `alignment` = 32

12.274.1 Member Typedef Documentation

12.274.1.1 `vect_t` [1/2]

```
using vect_t = __m256i
```

12.274.1.2 `half_t` [1/2]

```
using half_t = __m128i
```

12.274.1.3 `scalar_t` [1/2]

```
using scalar_t = int32_t
```

12.274.1.4 `simdHalf` [1/2]

```
using simdHalf = Simd128<scalar_t>
```

12.274.1.5 `vect_t` [2/2]

```
using vect_t = __m512i
```

12.274.1.6 `half_t` [2/2]

```
using half_t = __m256i
```

12.274.1.7 `scalar_t` [2/2]

```
using scalar_t = int32_t
```

12.274.1.8 `simdHalf` [2/2]

```
using simdHalf = Simd256<scalar_t>
```

12.274.2 Member Function Documentation

12.274.2.1 `valid()` [1/2]

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```


12.274.2.2 compliant() [1/2]

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.274.2.3 set1() [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.274.2.4 set() [1/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

12.274.2.5 gather() [1/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.274.2.6 load() [1/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.274.2.7 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.274.2.8 store() [1/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.274.2.9 storeu() [1/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.274.2.10 stream() [1/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static]
```

12.274.2.11 sll() [1/2]

```
template<int s>
static INLINE CONST vect_t sll (
    const vect_t a ) [inline], [static]
```

12.274.2.12 srl() [1/2]

```
template<int s>
static INLINE CONST vect_t srl (
    const vect_t a ) [inline], [static]
```

12.274.2.13 sra() [1/2]

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.274.2.14 shuffle_twice() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
    const vect_t a ) [inline], [static]
```

12.274.2.15 shuffle() [1/2]

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
    const vect_t a ) [inline], [static]
```

12.274.2.16 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.17 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.20 unpacklohi()

```
static INLINE void unpacklohi (
    vect_t & s1,
    vect_t & s2,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.21 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.22 add() [1/2]

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.23 addin() [1/2]

```
static INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.274.2.24 sub() [1/2]

```
static INLINE CONST vect_t sub (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.25 subin() [1/2]

```
static INLINE vect_t subin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.274.2.26 mullo() [1/2]

```
static INLINE CONST vect_t mullo (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.27 mul() [1/2]

```
static INLINE CONST vect_t mul (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.28 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.29 mulx() [1/2]

```
static INLINE CONST vect_t mulx (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.274.2.30 fmadd() [1/2]

```
static INLINE CONST vect_t fmadd (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.31 fmaddin() [1/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.32 fmaddx() [1/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.33 fmaddxin() [1/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.34 fnmadd() [1/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.35 fnmaddin() [1/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.36 fnmaddx() [1/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.37 fnmaddxin() [1/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.38 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.39 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

12.274.2.40 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.41 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.42 eq() [1/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.43 greater() [1/2]

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.44 lesser() [1/2]

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.45 greater_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.46 lesser_eq() [1/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.47 hadd_to_scal() [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

12.274.2.48 round() [1/2]

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

12.274.2.49 mod() [1/2]

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

12.274.2.50 valid() [2/2]

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

12.274.2.51 compliant() [2/2]

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.274.2.52 set1() [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.274.2.53 set() [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static]
```

12.274.2.54 gather() [2/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.274.2.55 load() [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.274.2.56 loadu() [2/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

12.274.2.57 store() [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

12.274.2.58 storeu() [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

12.274.2.59 stream() [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.274.2.60 sll() [2/2]

```
template<int s>  
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

12.274.2.61 srl() [2/2]

```
template<int s>  
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

12.274.2.62 sra() [2/2]

```
template<int s>  
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

12.274.2.63 shuffle_twice() [2/2]

```
template<uint8_t s>  
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static]
```

12.274.2.64 shuffle() [2/2]

```
template<uint64_t s>  
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

12.274.2.65 add() [2/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.66 addin() [2/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.274.2.67 sub() [2/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.68 subin() [2/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.274.2.69 mullo() [2/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.70 mul() [2/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.71 mulhi() [2/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.72 mulx() [2/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

12.274.2.73 fmadd() [2/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.74 fmaddin() [2/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```


12.274.2.75 fmaddx() [2/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.76 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.77 fnmadd() [2/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.78 fnmaddin() [2/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.79 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.80 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.81 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.82 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.274.2.83 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,
```

```
const vect_t a,
const vect_t b ) [inline], [static]
```

12.274.2.84 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.85 eq() [2/2]

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.86 greater() [2/2]

```
static INLINE CONST vect_t greater (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.87 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.88 greater_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.89 lesser_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.274.2.90 hadd_to_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.274.2.91 round() [2/2]

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

12.274.2.92 mod() [2/2]

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
```

```

    vect_t & Q,
    vect_t & T ) [inline], [static]

```

12.274.2.93 type_string() [1/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.274.2.94 zero() [1/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.274.2.95 type_string() [2/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.274.2.96 zero() [2/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.274.2.97 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.274.2.98 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.274.2.99 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.274.2.100 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.274.3 Field Documentation**12.274.3.1 vect_size**

```
static const constexpr size_t vect_size = 8 [static], [constexpr]
```

12.274.3.2 alignment

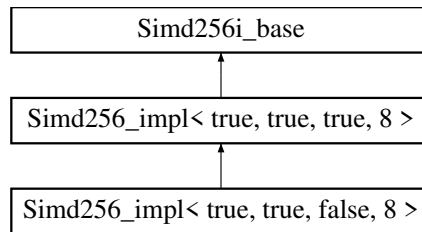
```
static const constexpr size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following files:

- [simd256_int32.inl](#)
- [simd512_int32.inl](#)

12.275 Simd256_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd256_impl< true, true, true, 8 >:



Data Structures

- union [Converter](#)

Public Types

- using [vect_t](#) = __m256i
- using [half_t](#) = __m128i
- using [scalar_t](#) = int64_t
- using [simdHalf](#) = Simd128< [scalar_t](#) >

Static Public Member Functions

- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- template<int idx>
static [INLINE CONST scalar_t get](#) ([vect_t](#) v)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE](#) void [store](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
static [INLINE CONST vect_t sll](#) (const [vect_t](#) a)
- template<int s>
static [INLINE CONST vect_t srl](#) (const [vect_t](#) a)
- template<int s>
static [INLINE CONST vect_t sra](#) (const [vect_t](#) a)
- template<uint8_t s>
static [INLINE CONST vect_t shuffle](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t unpacklo_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpackhi_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpacklo](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpackhi](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE](#) void [unpacklohi](#) ([vect_t](#) &l, [vect_t](#) &h, const [vect_t](#) a, const [vect_t](#) b)
- template<uint8_t s>
static [INLINE CONST vect_t blend](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t add](#) (const [vect_t](#) a, const [vect_t](#) b)

- static `INLINE vect_t addin (vect_t &a, const vect_t b)`
- static `INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- static `INLINE vect_t subin (vect_t &a, const vect_t b)`
- static `INLINE CONST vect_t mullo (vect_t a, vect_t b)`
- static `INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE CONST vect_t mask_high ()`
- static `INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- static `INLINE vect_t mod (vect_t &C, const __m256d &P, const __m256d &INVP, const __m256d &NEGP, const vect_t &POW50REM, const __m256d &MIN, const __m256d &MAX, __m256d &Q, __m256d &T)`
- static const std::string `type_string ()`
- static `INLINE CONST vect_t zero ()`

Static Public Attributes

- static const constexpr size_t `vect_size` = 4
- static const constexpr size_t `alignment` = 32

Static Protected Member Functions

- static `INLINE CONST vect_t signbits (const vect_t x)`

12.275.1 Member Typedef Documentation

12.275.1.1 vect_t

```
using vect_t = __m256i
```

12.275.1.2 half_t

```
using half_t = __m128i
```

12.275.1.3 scalar_t

```
using scalar_t = int64_t
```

12.275.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.275.2 Member Function Documentation

12.275.2.1 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

12.275.2.2 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.275.2.3 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.275.2.4 set()

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

12.275.2.5 gather()

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.275.2.6 get()

```
template<int idx>
static INLINE CONST scalar_t get (
    vect_t v ) [inline], [static]
```

12.275.2.7 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.275.2.8 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.275.2.9 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.275.2.10 storeu()

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.275.2.11 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.275.2.12 sll()

```
template<int s>  
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

12.275.2.13 srl()

```
template<int s>  
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

12.275.2.14 sra()

```
template<int s>  
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

12.275.2.15 shuffle()

```
template<uint8_t s>  
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

12.275.2.16 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.17 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.20 unpacklohi()

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.21 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.275.2.22 add()

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.275.2.23 addin()

```
static INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.275.2.24 sub()

```
static INLINE CONST vect_t sub (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.275.2.25 subin()

```
static INLINE vect_t subin (
    vect_t & a,
    const vect_t b ) [inline], [static]
```

12.275.2.26 mullo()

```
static INLINE CONST vect_t mullo (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.275.2.27 mul()

```
static INLINE CONST vect_t mul (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.275.2.28 mulx()

```
static INLINE CONST vect_t mulx (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.275.2.29 fmadd()

```
static INLINE CONST vect_t fmadd (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```


12.275.2.30 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.31 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.32 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.33 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.34 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.35 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.36 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.37 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.38 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

12.275.2.39 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.40 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.41 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.42 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.43 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.44 greater_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.45 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.275.2.46 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

12.275.2.47 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

12.275.2.48 mask_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

12.275.2.49 mulhi_fast()

```

INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]

```

12.275.2.50 mod()

```

INLINE vect_t mod (
    vect_t & C,
    const __m256d & P,
    const __m256d & INV_P,
    const __m256d & NEG_P,
    const vect_t & POW50REM,
    const __m256d & MIN,
    const __m256d & MAX,
    __m256d & Q,
    __m256d & T ) [static]

```

12.275.2.51 signbits()

```

static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]

```

12.275.2.52 type_string()

```

static const std::string type_string ( ) [inline], [static], [inherited]

```

12.275.2.53 zero()

```

static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]

```

12.275.3 Field Documentation**12.275.3.1 vect_size**

```

const constexpr size_t vect_size = 4 [static], [constexpr]

```

12.275.3.2 alignment

```

const constexpr size_t alignment = 32 [static], [constexpr]

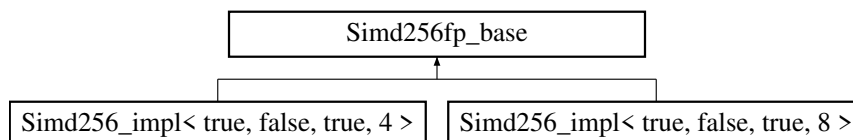
```

The documentation for this struct was generated from the following file:

- [simd256_int64.inl](#)

12.276 Simd256fp_base Struct Reference

Inheritance diagram for Simd256fp_base:

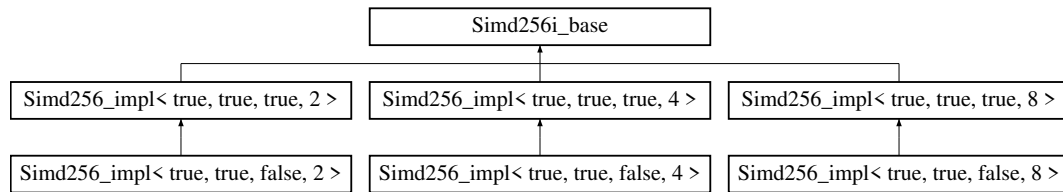


The documentation for this struct was generated from the following file:

- [simd256.inl](#)

12.277 Simd256i_base Struct Reference

Inheritance diagram for Simd256i_base:



Public Types

- using [vect_t](#) = __m256i

Static Public Member Functions

- static const std::string [type_string](#) ()
- static [INLINE CONST vect_t zero](#) ()

12.277.1 Member Typedef Documentation

12.277.1.1 vect_t

using [vect_t](#) = __m256i

12.277.2 Member Function Documentation

12.277.2.1 type_string()

```
static const std::string type_string ( ) [inline], [static]
```

12.277.2.2 zero()

```
static INLINE CONST vect\_t zero ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd256.inl](#)

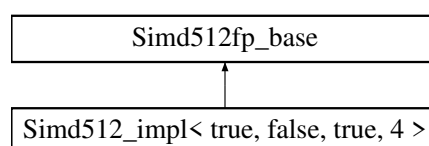
12.278 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

12.279 Simd512_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd512_impl< true, false, true, 4 >:



Static Public Member Functions

- static const std::string [type_string](#) ()

12.279.1 Member Function Documentation**12.279.1.1 type_string()**

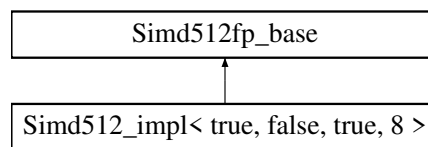
```
static const std::string type_string ( ) [inline], [static], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd512_float.inl](#)

12.280 Simd512_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd512_impl< true, false, true, 8 >:

**Public Types**

- using [vect_t](#) = __m512d
- using [scalar_t](#) = double

Static Public Member Functions

- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect_t zero](#) ()
- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3, const [scalar_t](#) x4, const [scalar_t](#) x5, const [scalar_t](#) x6, const [scalar_t](#) x7, const [scalar_t](#) x8)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE void store](#) (const [scalar_t](#) *p, const [vect_t](#) v)
- static [INLINE void storeu](#) (const [scalar_t](#) *p, const [vect_t](#) v)
- static [INLINE void stream](#) (const [scalar_t](#) *p, const [vect_t](#) v)
- template<uint8_t s>
static [INLINE CONST vect_t shuffle](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t unpacklo_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpackhi_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- template<uint8_t s>
static [INLINE CONST vect_t blend](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t blendv](#) (const [vect_t](#) a, const [vect_t](#) b, const [vect_t](#) mask)
- static [INLINE CONST vect_t add](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t addin](#) ([vect_t](#) &a, const [vect_t](#) b)
- static [INLINE CONST vect_t sub](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t subin](#) ([vect_t](#) &a, const [vect_t](#) b)
- static [INLINE CONST vect_t mul](#) (const [vect_t](#) a, const [vect_t](#) b)

- static `INLINE CONST vect_t mulin (vect_t &a, const vect_t b)`
- static `INLINE CONST vect_t div (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t floor (const vect_t a)`
- static `INLINE CONST vect_t ceil (const vect_t a)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE CONST vect_t hadd (const vect_t a, const vect_t b)`
- static `INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- static const std::string `type_string ()`

Static Public Attributes

- static const constexpr size_t `vect_size` = 8
- static const constexpr size_t `alignment` = 64

12.280.1 Member Typedef Documentation

12.280.1.1 vect_t

using `vect_t` = __m512d

12.280.1.2 scalar_t

using `scalar_t` = double

12.280.2 Member Function Documentation

12.280.2.1 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

12.280.2.2 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.280.2.3 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

12.280.2.4 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.280.2.5 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7,  
    const scalar_t x8 ) [inline], [static]
```

12.280.2.6 gather()

```
template<class T >  
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

12.280.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

12.280.2.8 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

12.280.2.9 store()

```
static INLINE void store (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.280.2.10 storeu()

```
static INLINE void storeu (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.280.2.11 stream()

```
static INLINE void stream (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.280.2.12 shuffle()

```
template<uint8_t s>  
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

12.280.2.13 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.280.2.14 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.280.2.15 blend()

```
template<uint8_t s>  
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.280.2.16 blendv()

```
static INLINE CONST vect_t blendv (  
    const vect_t a,  
    const vect_t b,  
    const vect_t mask ) [inline], [static]
```

12.280.2.17 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.280.2.18 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.280.2.19 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.280.2.20 subin()

```
static INLINE CONST vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.280.2.21 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.280.2.22 mulin()

```
static INLINE CONST vect_t mulin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.280.2.23 div()

```
static INLINE CONST vect_t div (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```


12.280.2.24 fmadd()

```
static INLINE CONST vect_t fmadd (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.25 fmaddin()

```
static INLINE CONST vect_t fmaddin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.26 fnmadd()

```
static INLINE CONST vect_t fnmadd (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.27 fnmaddin()

```
static INLINE CONST vect_t fnmaddin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.28 fmsub()

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.29 fmsubin()

```
static INLINE CONST vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.30 eq()

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.31 lesser()

```
static INLINE CONST vect_t lesser (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.32 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.33 greater()

```
static INLINE CONST vect_t greater (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.34 greater_eq()

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.35 floor()

```
static INLINE CONST vect_t floor (
    const vect_t a ) [inline], [static]
```

12.280.2.36 ceil()

```
static INLINE CONST vect_t ceil (
    const vect_t a ) [inline], [static]
```

12.280.2.37 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

12.280.2.38 hadd()

```
static INLINE CONST vect_t hadd (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.280.2.39 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

12.280.2.40 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.280.3 Field Documentation**12.280.3.1 vect_size**

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```

12.280.3.2 alignment

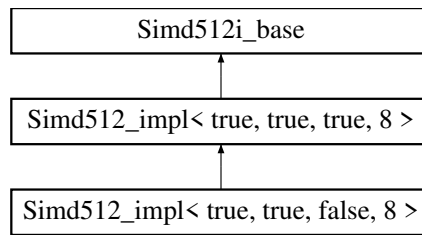
```
const constexpr size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd512_double.inl](#)

12.281 Simd512_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd512_impl< true, true, false, 8 >:



Data Structures

- union [Converter](#)

Public Types

- using [scalar_t](#) = [uint64_t](#)
- using [simdHalf](#) = [Simd256](#)< [scalar_t](#) >
- using [vect_t](#) = [__m512i](#)
- using [half_t](#) = [__m256i](#)

Static Public Member Functions

- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3, const [scalar_t](#) x4, const [scalar_t](#) x5, const [scalar_t](#) x6, const [scalar_t](#) x7)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE](#) void [store](#) ([scalar_t](#) *p, [vect_t](#) v)
- template<[uint8_t](#) k>
static [INLINE](#) void [maskstore](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
static [INLINE CONST vect_t sra](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t greater](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t lesser](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t greater_eq](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t lesser_eq](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mullo](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t mulx](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmaddx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmaddxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fnmaddx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fnmaddxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmsubx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmsubxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST scalar_t hadd_to_scal](#) (const [vect_t](#) a)
- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3, const [scalar_t](#) x4, const [scalar_t](#) x5, const [scalar_t](#) x6, const [scalar_t](#) x7)

- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3)
- template<class T >
static `INLINE PURE vect_t gather` (const `scalar_t` *const p, const T *const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` *const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` *const p)
- static `INLINE void store` (`scalar_t` *p, `vect_t` v)
- template<uint8_t k>
static `INLINE void maskstore` (`scalar_t` *p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` *p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` *p, const `vect_t` v)
- template<int s>
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8_t s>
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &l, `vect_t` &h, const `vect_t` a, const `vect_t` b)
- template<uint8_t s>
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const __m512d &P, const __m512d &INVP, const __m512d &NEGP, const `vect_t` &POW50REM, const __m512d &MIN, const __m512d &MAX, __m512d &Q, __m512d &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

Static Public Attributes

- static const constexpr size_t `vect_size` = 8
- static const constexpr size_t `alignment` = 64

Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

12.281.1 Member Typedef Documentation

12.281.1.1 scalar_t

```
using scalar_t = uint64_t
```

12.281.1.2 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

12.281.1.3 vect_t

```
using vect_t = __m512i [inherited]
```

12.281.1.4 half_t

```
using half_t = __m256i [inherited]
```

12.281.2 Member Function Documentation

12.281.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.281.2.2 set() [1/3]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

12.281.2.3 gather() [1/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.281.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.281.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.281.2.6 store() [1/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.281.2.7 maskstore() [1/2]

```
template<uint8_t k>
static INLINE void maskstore (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.281.2.8 storeu() [1/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.281.2.9 stream() [1/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static]
```

12.281.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
    const vect_t a ) [inline], [static]
```

12.281.2.11 greater()

```
static INLINE CONST vect_t greater (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.281.2.12 lesser()

```
static INLINE CONST vect_t lesser (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.281.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.281.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.281.2.15 mullo()

```
static INLINE CONST vect_t mullo (
    vect_t a,
    vect_t b ) [inline], [static]
```

12.281.2.16 mulx()

```
static INLINE CONST vect_t mulx (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

12.281.2.17 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.281.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.281.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.281.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.281.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.281.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.281.2.23 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

12.281.2.24 valid()

```
template<class T >  
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

12.281.2.25 compliant()

```
template<class T >  
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

12.281.2.26 set1() [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

12.281.2.27 set() [2/3]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static], [inherited]
```

12.281.2.28 set() [3/3]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static], [inherited]
```

12.281.2.29 gather() [2/2]

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

12.281.2.30 load() [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.281.2.31 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

12.281.2.32 store() [2/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.281.2.33 maskstore() [2/2]

```
template<uint8_t k>
static INLINE void maskstore (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

12.281.2.34 storeu() [2/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```


12.281.2.35 stream() [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static], [inherited]
```

12.281.2.36 sll()

```
template<int s>  
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

12.281.2.37 srl()

```
template<int s>  
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

12.281.2.38 shuffle()

```
template<uint8_t s>  
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

12.281.2.39 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.40 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.41 unpacklo()

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.42 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.43 unpacklohi()

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.44 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.45 add()

```
static INLINE CONST vect_t add (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.46 addin()

```
static INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.47 sub()

```
static INLINE CONST vect_t sub (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.48 subin()

```
static INLINE vect_t subin (
    vect_t & a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.49 mul()

```
static INLINE CONST vect_t mul (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.50 fmadd()

```
static INLINE CONST vect_t fmadd (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.51 fmaddin()

```
static INLINE vect_t fmaddin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.52 fnmadd()

```
static INLINE CONST vect_t fnmadd (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.53 fnmaddin()

```
static INLINE vect_t fnmaddin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.54 fmsub()

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.55 fmsubin()

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.56 eq()

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.57 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

12.281.2.58 mask_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

12.281.2.59 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

12.281.2.60 mod()

```
INLINE vect_t mod (
    vect_t & C,
    const __m512d & P,
    const __m512d & INVP,
    const __m512d & NEGP,
    const vect_t & POW50REM,
    const __m512d & MIN,
    const __m512d & MAX,
    __m512d & Q,
    __m512d & T ) [static], [inherited]
```

12.281.2.61 signbits()

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected], [inherited]
```

12.281.2.62 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.281.2.63 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.281.2.64 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.65 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.66 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.2.67 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.281.3 Field Documentation**12.281.3.1 vect_size**

```
const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

12.281.3.2 alignment

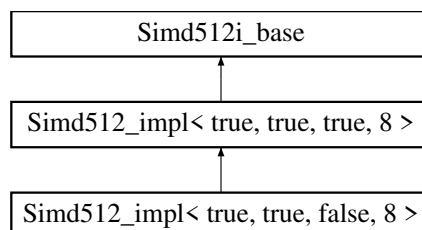
```
const constexpr size_t alignment = 64 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd512_int64.inl](#)

12.282 Simd512_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd512_impl< true, true, true, 8 >:



Data Structures

- union [Converter](#)

Public Types

- using [vect_t](#) = __m512i
- using [half_t](#) = __m256i
- using [scalar_t](#) = int64_t
- using [simdHalf](#) = Simd256< [scalar_t](#) >

Static Public Member Functions

- template<class T >
static constexpr bool [valid](#) (T *p)
- template<class T >
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect_t set1](#) (const [scalar_t](#) x)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3, const [scalar_t](#) x4, const [scalar_t](#) x5, const [scalar_t](#) x6, const [scalar_t](#) x7)
- static [INLINE CONST vect_t set](#) (const [scalar_t](#) x0, const [scalar_t](#) x1, const [scalar_t](#) x2, const [scalar_t](#) x3)
- template<class T >
static [INLINE PURE vect_t gather](#) (const [scalar_t](#) *const p, const T *const idx)
- static [INLINE PURE vect_t load](#) (const [scalar_t](#) *const p)
- static [INLINE PURE vect_t loadu](#) (const [scalar_t](#) *const p)
- static [INLINE void store](#) ([scalar_t](#) *p, [vect_t](#) v)
- template<uint8_t k>
static [INLINE void maskstore](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE void storeu](#) ([scalar_t](#) *p, [vect_t](#) v)
- static [INLINE void stream](#) ([scalar_t](#) *p, const [vect_t](#) v)
- template<int s>
static [INLINE CONST vect_t sll](#) (const [vect_t](#) a)
- template<int s>
static [INLINE CONST vect_t srl](#) (const [vect_t](#) a)
- template<int s>
static [INLINE CONST vect_t sra](#) (const [vect_t](#) a)
- template<uint8_t s>
static [INLINE CONST vect_t shuffle](#) (const [vect_t](#) a)
- static [INLINE CONST vect_t unpacklo_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpackhi_twice](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpacklo](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t unpackhi](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE void unpacklohi](#) ([vect_t](#) &l, [vect_t](#) &h, const [vect_t](#) a, const [vect_t](#) b)
- template<uint8_t s>
static [INLINE CONST vect_t blend](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t add](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t addin](#) ([vect_t](#) &a, const [vect_t](#) b)
- static [INLINE CONST vect_t sub](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t subin](#) ([vect_t](#) &a, const [vect_t](#) b)
- static [INLINE CONST vect_t mullo](#) ([vect_t](#) a, [vect_t](#) b)
- static [INLINE CONST vect_t mul](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t mulx](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmadd](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmaddin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t fmadx](#) (const [vect_t](#) c, const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE vect_t fmadxin](#) ([vect_t](#) &c, const [vect_t](#) a, const [vect_t](#) b)

- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const `__m512d` &P, const `__m512d` &INVP, const `__m512d` &NEGP, const `vect_t` &POW50REM, const `__m512d` &MIN, const `__m512d` &MAX, `__m512d` &Q, `__m512d` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

Static Public Attributes

- static const constexpr size_t `vect_size` = 8
- static const constexpr size_t `alignment` = 64

Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

12.282.1 Member Typedef Documentation

12.282.1.1 vect_t

```
using vect_t = __m512i
```

12.282.1.2 half_t

```
using half_t = __m256i
```

12.282.1.3 scalar_t

```
using scalar_t = int64_t
```

12.282.1.4 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

12.282.2 Member Function Documentation

12.282.2.1 valid()

```
template<class T >
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

12.282.2.2 compliant()

```
template<class T >
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

12.282.2.3 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

12.282.2.4 set() [1/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

12.282.2.5 set() [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static]
```

12.282.2.6 gather()

```
template<class T >
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

12.282.2.7 load()

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

12.282.2.8 loadu()

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

12.282.2.9 store()

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.282.2.10 maskstore()

```
template<uint8_t k>
static INLINE void maskstore (
    scalar_t * p,
    vect_t v ) [inline], [static]
```

12.282.2.11 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

12.282.2.12 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

12.282.2.13 sll()

```
template<int s>  
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

12.282.2.14 srl()

```
template<int s>  
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

12.282.2.15 sra()

```
template<int s>  
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

12.282.2.16 shuffle()

```
template<uint8_t s>  
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

12.282.2.17 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.18 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.19 unpacklo()

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.20 unpackhi()

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```


12.282.2.21 unpacklohi()

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.22 blend()

```
template<uint8_t s>  
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.23 add()

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.24 addin()

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.282.2.25 sub()

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.26 subin()

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

12.282.2.27 mullo()

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

12.282.2.28 mul()

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.29 mulx()

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.30 fmadd()

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.31 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.32 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.33 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.34 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.35 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.36 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.37 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.38 fmsub()

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

12.282.2.39 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.40 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.41 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.42 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.43 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.44 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.45 greater_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.46 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

12.282.2.47 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

12.282.2.48 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

12.282.2.49 mask_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

12.282.2.50 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]
```

12.282.2.51 mod()

```
INLINE vect_t mod (
    vect_t & C,
    const __m512d & P,
    const __m512d & INV_P,
    const __m512d & NEGP,
    const vect_t & POW50REM,
    const __m512d & MIN,
    const __m512d & MAX,
    __m512d & Q,
    __m512d & T ) [static]
```

12.282.2.52 signbits()

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]
```

12.282.2.53 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

12.282.2.54 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

12.282.2.55 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.282.2.56 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.282.2.57 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.282.2.58 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

12.282.3 Field Documentation**12.282.3.1 vect_size**

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```

12.282.3.2 alignment

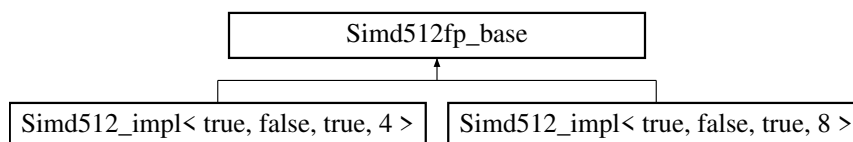
```
const constexpr size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd512_int64.inl](#)

12.283 Simd512fp_base Struct Reference

Inheritance diagram for Simd512fp_base:

**Static Public Member Functions**

- static const std::string [type_string](#) ()

12.283.1 Member Function Documentation**12.283.1.1 type_string()**

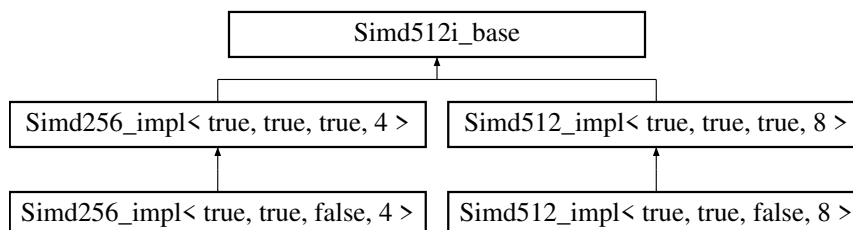
```
static const std::string type_string ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

12.284 Simd512i_base Struct Reference

Inheritance diagram for Simd512i_base:

**Public Types**

- using [vect_t](#) = __m512i

Static Public Member Functions

- static const std::string [type_string](#) ()
- static [INLINE CONST vect_t zero](#) ()
- static [INLINE CONST vect_t vor](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t vxor](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t vand](#) (const [vect_t](#) a, const [vect_t](#) b)
- static [INLINE CONST vect_t vandnot](#) (const [vect_t](#) a, const [vect_t](#) b)

12.284.1 Member Typedef Documentation

12.284.1.1 [vect_t](#)

using [vect_t](#) = __m512i

12.284.2 Member Function Documentation

12.284.2.1 [type_string\(\)](#)

```
static const std::string type_string ( ) [inline], [static]
```

12.284.2.2 [zero\(\)](#)

```
static INLINE CONST vect\_t zero ( ) [inline], [static]
```

12.284.2.3 [vor\(\)](#)

```
static INLINE CONST vect\_t vor (
    const vect\_t a,
    const vect\_t b ) [inline], [static]
```

12.284.2.4 [vxor\(\)](#)

```
static INLINE CONST vect\_t vxor (
    const vect\_t a,
    const vect\_t b ) [inline], [static]
```

12.284.2.5 [vand\(\)](#)

```
static INLINE CONST vect\_t vand (
    const vect\_t a,
    const vect\_t b ) [inline], [static]
```

12.284.2.6 [vandnot\(\)](#)

```
static INLINE CONST vect\_t vandnot (
    const vect\_t a,
    const vect\_t b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

12.285 [SimdChooser< T, bool, bool >](#) Struct Template Reference

```
#include <fflas_simd.h>
```

The documentation for this struct was generated from the following file:

- [fflas_simd.h](#)

12.286 SimdChooser< T, false, b > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using [value](#) = [NoSimd](#)< T >

12.286.1 Member Typedef Documentation

12.286.1.1 value

```
template<class T , bool b>
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas_simd.h](#)

12.287 SimdChooser< T, true, false > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using [value](#) = [NoSimd](#)< T >

12.287.1 Member Typedef Documentation

12.287.1.1 value

```
template<class T >
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas_simd.h](#)

12.288 SimdChooser< T, true, true > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using [value](#) = [NoSimd](#)< T >

12.288.1 Member Typedef Documentation

12.288.1.1 value

```
template<class T >
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas_simd.h](#)

12.289 simdToType< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- [fflas_simd.h](#)

12.290 Single Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.291 `Sparse< Field, SparseMatrix_t, IdxT, PtrT >` Struct Template Reference

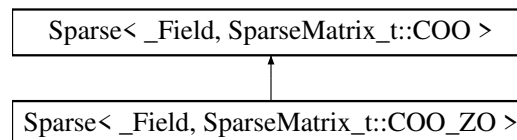
The documentation for this struct was generated from the following file:

- [fflas_sparse.h](#)

12.292 `Sparse< _Field, SparseMatrix_t::COO >` Struct Template Reference

```
#include <coo.h>
```

Inheritance diagram for `Sparse< _Field, SparseMatrix_t::COO >`:



Public Types

- using [Field](#) = `_Field`

Data Fields

- `index_t * col` = nullptr
- `index_t * row` = nullptr
- `_Field::Element_ptr dat`
- `bool delayed` = false
- `uint64_t kmax` = 0
- `index_t m` = 0
- `index_t n` = 0
- `uint64_t nnz` = 0
- `uint64_t nElements` = 0
- `uint64_t maxrow` = 0

12.292.1 Member Typedef Documentation

12.292.1.1 Field

```
template<class _Field >
using Field = _Field
```

12.292.2 Field Documentation

12.292.2.1 col

```
template<class _Field >
index\_t* col = nullptr
```


12.292.2.2 row

```
template<class _Field >
index_t* row = nullptr
```

12.292.2.3 dat

```
template<class _Field >
_Field::Element_ptr dat
```

12.292.2.4 delayed

```
template<class _Field >
bool delayed = false
```

12.292.2.5 kmax

```
template<class _Field >
uint64_t kmax = 0
```

12.292.2.6 m

```
template<class _Field >
index_t m = 0
```

12.292.2.7 n

```
template<class _Field >
index_t n = 0
```

12.292.2.8 nnz

```
template<class _Field >
uint64_t nnz = 0
```

12.292.2.9 nElements

```
template<class _Field >
uint64_t nElements = 0
```

12.292.2.10 maxrow

```
template<class _Field >
uint64_t maxrow = 0
```

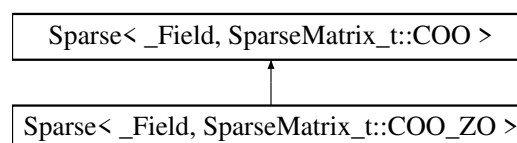
The documentation for this struct was generated from the following file:

- [coo.h](#)

12.293 Sparse< _Field, SparseMatrix_t::COO_ZO > Struct Template Reference

```
#include <coo.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::COO_ZO >:



Public Types

- using `Field` = `_Field`

Data Fields

- `_Field::Element` `cst` = 1
- `index_t` * `col` = `nullptr`
- `index_t` * `row` = `nullptr`
- `_Field::Element_ptr` `dat`
- `bool` `delayed` = `false`
- `uint64_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0

12.293.1 Member Typedef Documentation

12.293.1.1 Field

```
template<class _Field >
using Field = _Field
```

12.293.2 Field Documentation

12.293.2.1 cst

```
template<class _Field >
_Field::Element cst = 1
```

12.293.2.2 col

```
template<class _Field >
index_t* col = nullptr [inherited]
```

12.293.2.3 row

```
template<class _Field >
index_t* row = nullptr [inherited]
```

12.293.2.4 dat

```
template<class _Field >
_Field::Element_ptr dat [inherited]
```

12.293.2.5 delayed

```
template<class _Field >
bool delayed = false [inherited]
```

12.293.2.6 kmax

```
template<class _Field >
uint64_t kmax = 0 [inherited]
```

12.293.2.7 m

```
template<class _Field >
index_t m = 0 [inherited]
```

12.293.2.8 n

```
template<class _Field >
index_t n = 0 [inherited]
```

12.293.2.9 nnz

```
template<class _Field >
uint64_t nnz = 0 [inherited]
```

12.293.2.10 nElements

```
template<class _Field >
uint64_t nElements = 0 [inherited]
```

12.293.2.11 maxrow

```
template<class _Field >
uint64_t maxrow = 0 [inherited]
```

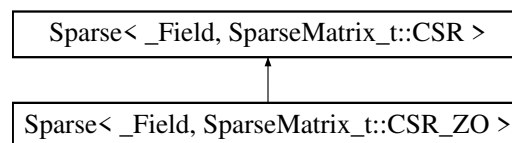
The documentation for this struct was generated from the following file:

- [coo.h](#)

12.294 Sparse< _Field, SparseMatrix_t::CSR > Struct Template Reference

```
#include <csr.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::CSR >:

**Public Types**

- using [Field](#) = [_Field](#)

Data Fields

- bool [delayed](#) = false
- [uint64_t](#) [kmax](#) = 0
- [index_t](#) [m](#) = 0
- [index_t](#) [n](#) = 0
- [uint64_t](#) [nnz](#) = 0
- [uint64_t](#) [nElements](#) = 0
- [uint64_t](#) [maxrow](#) = 0
- [index_t](#) * [col](#) = nullptr
- [index_t](#) * [st](#) = nullptr
- [index_t](#) * [stend](#) = nullptr
- [_Field::Element_ptr](#) [dat](#)

12.294.1 Member Typedef Documentation**12.294.1.1 Field**

```
template<class _Field >
using Field = \_Field
```

12.294.2 Field Documentation

12.294.2.1 delayed

```
template<class _Field >
bool delayed = false
```

12.294.2.2 kmax

```
template<class _Field >
uint64_t kmax = 0
```

12.294.2.3 m

```
template<class _Field >
index_t m = 0
```

12.294.2.4 n

```
template<class _Field >
index_t n = 0
```

12.294.2.5 nnz

```
template<class _Field >
uint64_t nnz = 0
```

12.294.2.6 nElements

```
template<class _Field >
uint64_t nElements = 0
```

12.294.2.7 maxrow

```
template<class _Field >
uint64_t maxrow = 0
```

12.294.2.8 col

```
template<class _Field >
index_t* col = nullptr
```

12.294.2.9 st

```
template<class _Field >
index_t* st = nullptr
```

12.294.2.10 stend

```
template<class _Field >
index_t* stend = nullptr
```

12.294.2.11 dat

```
template<class _Field >
_Field::Element_ptr dat
```

The documentation for this struct was generated from the following file:

- [csr.h](#)

12.295 Sparse< _Field, SparseMatrix_t::CSR_HYB > Struct Template Reference

```
#include <csr_hyb.h>
```

Public Types

- using [Field](#) = [_Field](#)

Data Fields

- bool [delayed](#) = false
- [index_t](#) * [col](#) = nullptr
- [index_t](#) * [st](#) = nullptr
- [_Field::Element_ptr](#) [dat](#)
- [uint64_t](#) [kmax](#) = 0
- [index_t](#) [m](#) = 0
- [index_t](#) [n](#) = 0
- [uint64_t](#) [nnz](#) = 0
- [uint64_t](#) [nElements](#) = 0
- [uint64_t](#) [maxrow](#) = 0
- [uint64_t](#) [nOnes](#) = 0
- [uint64_t](#) [nMOnes](#) = 0
- [uint64_t](#) [nOthers](#) = 0

12.295.1 Member Typedef Documentation

12.295.1.1 Field

```
template<class _Field >
using Field = \_Field
```

12.295.2 Field Documentation

12.295.2.1 delayed

```
template<class _Field >
bool delayed = false
```

12.295.2.2 col

```
template<class _Field >
index\_t* col = nullptr
```

12.295.2.3 st

```
template<class _Field >
index\_t* st = nullptr
```

12.295.2.4 dat

```
template<class _Field >
\_Field::Element\_ptr dat
```

12.295.2.5 kmax

```
template<class _Field >
uint64\_t kmax = 0
```

12.295.2.6 m

```
template<class _Field >
index_t m = 0
```

12.295.2.7 n

```
template<class _Field >
index_t n = 0
```

12.295.2.8 nnz

```
template<class _Field >
uint64_t nnz = 0
```

12.295.2.9 nElements

```
template<class _Field >
uint64_t nElements = 0
```

12.295.2.10 maxrow

```
template<class _Field >
uint64_t maxrow = 0
```

12.295.2.11 nOnes

```
template<class _Field >
uint64_t nOnes = 0
```

12.295.2.12 nMOnes

```
template<class _Field >
uint64_t nMOnes = 0
```

12.295.2.13 nOthers

```
template<class _Field >
uint64_t nOthers = 0
```

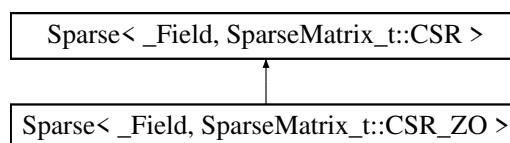
The documentation for this struct was generated from the following file:

- [csr_hyb.h](#)

12.296 Sparse< _Field, SparseMatrix_t::CSR_ZO > Struct Template Reference

```
#include <csr.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::CSR_ZO >:

**Public Types**

- using [Field](#) = _Field

Data Fields

- `int64_t` `cst` = 1
- `bool` `delayed` = false
- `uint64_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0
- `index_t` * `col` = nullptr
- `index_t` * `st` = nullptr
- `index_t` * `stend` = nullptr
- `_Field::Element_ptr` `dat`

12.296.1 Member Typedef Documentation

12.296.1.1 Field

```
template<class _Field >
using Field = _Field
```

12.296.2 Field Documentation

12.296.2.1 cst

```
template<class _Field >
int64_t cst = 1
```

12.296.2.2 delayed

```
template<class _Field >
bool delayed = false
```

12.296.2.3 kmax

```
template<class _Field >
uint64_t kmax = 0 [inherited]
```

12.296.2.4 m

```
template<class _Field >
index_t m = 0 [inherited]
```

12.296.2.5 n

```
template<class _Field >
index_t n = 0 [inherited]
```

12.296.2.6 nnz

```
template<class _Field >
uint64_t nnz = 0 [inherited]
```

12.296.2.7 nElements

```
template<class _Field >
uint64_t nElements = 0 [inherited]
```

12.296.2.8 maxrow

```
template<class _Field >
uint64_t maxrow = 0 [inherited]
```

12.296.2.9 col

```
template<class _Field >
index_t* col = nullptr [inherited]
```

12.296.2.10 st

```
template<class _Field >
index_t* st = nullptr [inherited]
```

12.296.2.11 stend

```
template<class _Field >
index_t* stend = nullptr [inherited]
```

12.296.2.12 dat

```
template<class _Field >
_Field::Element_ptr dat [inherited]
```

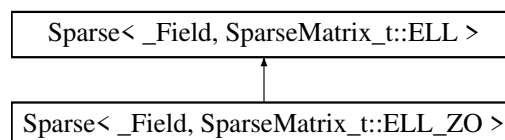
The documentation for this struct was generated from the following file:

- [csr.h](#)

12.297 Sparse< _Field, SparseMatrix_t::ELL > Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::ELL >:

**Public Types**

- using [Field](#) = _Field

Data Fields

- bool [delayed](#) = false
- [uint64_t](#) [kmax](#) = 0
- [index_t](#) [m](#) = 0
- [index_t](#) [n](#) = 0
- [index_t](#) [ld](#) = 0
- [uint64_t](#) [nnz](#) = 0
- [uint64_t](#) [nElements](#) = 0
- [uint64_t](#) [maxrow](#) = 0
- [index_t](#) * [col](#) = nullptr
- _Field::Element_ptr [dat](#)

12.297.1 Member Typedef Documentation

12.297.1.1 Field

```
template<class _Field >
using Field = _Field
```

12.297.2 Field Documentation

12.297.2.1 delayed

```
template<class _Field >
bool delayed = false
```

12.297.2.2 kmax

```
template<class _Field >
uint64_t kmax = 0
```

12.297.2.3 m

```
template<class _Field >
index_t m = 0
```

12.297.2.4 n

```
template<class _Field >
index_t n = 0
```

12.297.2.5 ld

```
template<class _Field >
index_t ld = 0
```

12.297.2.6 nnz

```
template<class _Field >
uint64_t nnz = 0
```

12.297.2.7 nElements

```
template<class _Field >
uint64_t nElements = 0
```

12.297.2.8 maxrow

```
template<class _Field >
uint64_t maxrow = 0
```

12.297.2.9 col

```
template<class _Field >
index_t* col = nullptr
```

12.297.2.10 dat

```
template<class _Field >
_Field::Element_ptr dat
```

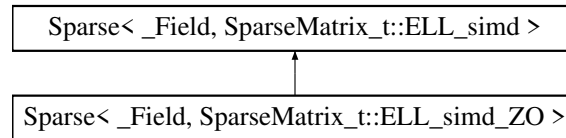
The documentation for this struct was generated from the following file:

- [ell.h](#)

12.298 Sparse< _Field, SparseMatrix_t::ELL_simd > Struct Template Reference

```
#include <ell_simd.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::ELL_simd >:



Data Fields

- bool `delayed` = false
- int `chunk` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `index_t` `ld` = 0
- `uint64_t` `kmax` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0
- `uint64_t` `nChunks` = 0
- `index_t` * `col` = nullptr
- `_Field::Element_ptr` `dat`

12.298.1 Field Documentation

12.298.1.1 `delayed`

```
template<class _Field >
bool delayed = false
```

12.298.1.2 `chunk`

```
template<class _Field >
int chunk = 0
```

12.298.1.3 `m`

```
template<class _Field >
index_t m = 0
```

12.298.1.4 `n`

```
template<class _Field >
index_t n = 0
```

12.298.1.5 `ld`

```
template<class _Field >
index_t ld = 0
```

12.298.1.6 `kmax`

```
template<class _Field >
uint64_t kmax = 0
```

12.298.1.7 nnz

```
template<class _Field >
uint64_t nnz = 0
```

12.298.1.8 nElements

```
template<class _Field >
uint64_t nElements = 0
```

12.298.1.9 maxrow

```
template<class _Field >
uint64_t maxrow = 0
```

12.298.1.10 nChunks

```
template<class _Field >
uint64_t nChunks = 0
```

12.298.1.11 col

```
template<class _Field >
index_t* col = nullptr
```

12.298.1.12 dat

```
template<class _Field >
_Field::Element_ptr dat
```

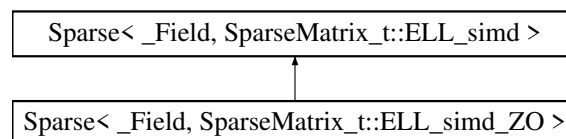
The documentation for this struct was generated from the following file:

- [ell_simd.h](#)

12.299 Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > Struct Template Reference

```
#include <ell_simd.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >:

**Data Fields**

- _Field::Element **cst** = 1
- bool **delayed** = false
- int **chunk** = 0
- **index_t** **m** = 0
- **index_t** **n** = 0
- **index_t** **ld** = 0
- **uint64_t** **kmax** = 0
- **uint64_t** **nnz** = 0
- **uint64_t** **nElements** = 0
- **uint64_t** **maxrow** = 0

- `uint64_t nChunks = 0`
- `index_t * col = nullptr`
- `_Field::Element_ptr dat`

12.299.1 Field Documentation

12.299.1.1 cst

```
template<class _Field >
_Field::Element cst = 1
```

12.299.1.2 delayed

```
template<class _Field >
bool delayed = false [inherited]
```

12.299.1.3 chunk

```
template<class _Field >
int chunk = 0 [inherited]
```

12.299.1.4 m

```
template<class _Field >
index_t m = 0 [inherited]
```

12.299.1.5 n

```
template<class _Field >
index_t n = 0 [inherited]
```

12.299.1.6 ld

```
template<class _Field >
index_t ld = 0 [inherited]
```

12.299.1.7 kmax

```
template<class _Field >
uint64_t kmax = 0 [inherited]
```

12.299.1.8 nnz

```
template<class _Field >
uint64_t nnz = 0 [inherited]
```

12.299.1.9 nElements

```
template<class _Field >
uint64_t nElements = 0 [inherited]
```

12.299.1.10 maxrow

```
template<class _Field >
uint64_t maxrow = 0 [inherited]
```

12.299.1.11 nChunks

```
template<class _Field >
uint64_t nChunks = 0 [inherited]
```

12.299.1.12 col

```
template<class _Field >
index_t* col = nullptr [inherited]
```

12.299.1.13 dat

```
template<class _Field >
_Field::Element_ptr dat [inherited]
```

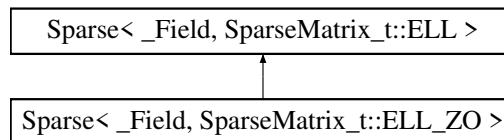
The documentation for this struct was generated from the following file:

- [ell_simd.h](#)

12.300 Sparse< _Field, SparseMatrix_t::ELL_ZO > Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::ELL_ZO >:

**Public Types**

- using [Field](#) = _Field

Data Fields

- _Field::Element [cst](#) = 1
- bool [delayed](#) = false
- [uint64_t](#) [kmax](#) = 0
- [index_t](#) [m](#) = 0
- [index_t](#) [n](#) = 0
- [index_t](#) [ld](#) = 0
- [uint64_t](#) [nnz](#) = 0
- [uint64_t](#) [nElements](#) = 0
- [uint64_t](#) [maxrow](#) = 0
- [index_t](#)* [col](#) = nullptr
- _Field::Element_ptr [dat](#)

12.300.1 Member Typedef Documentation**12.300.1.1 Field**

```
template<class _Field >
using Field = _Field
```

12.300.2 Field Documentation**12.300.2.1 cst**

```
template<class _Field >
_Field::Element cst = 1
```

12.300.2.2 delayed

```
template<class _Field >
bool delayed = false [inherited]
```

12.300.2.3 kmax

```
template<class _Field >
uint64_t kmax = 0 [inherited]
```

12.300.2.4 m

```
template<class _Field >
index_t m = 0 [inherited]
```

12.300.2.5 n

```
template<class _Field >
index_t n = 0 [inherited]
```

12.300.2.6 ld

```
template<class _Field >
index_t ld = 0 [inherited]
```

12.300.2.7 nnz

```
template<class _Field >
uint64_t nnz = 0 [inherited]
```

12.300.2.8 nElements

```
template<class _Field >
uint64_t nElements = 0 [inherited]
```

12.300.2.9 maxrow

```
template<class _Field >
uint64_t maxrow = 0 [inherited]
```

12.300.2.10 col

```
template<class _Field >
index_t* col = nullptr [inherited]
```

12.300.2.11 dat

```
template<class _Field >
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- [ell.h](#)

12.301 Sparse<_Field, SparseMatrix_t::HYB_ZO > Struct Template Reference

```
#include <hyb_zo.h>
```

Public Types

- using [Field](#) = [_Field](#)
- typedef [Sparse](#)<_Field, [SparseMatrix_t::HYB_ZO](#)> [Self_t](#)

Data Fields

- bool [delayed](#) = false
- [uint64_t](#) [kmax](#) = 0
- [index_t](#) [m](#) = 0
- [index_t](#) [n](#) = 0
- [uint64_t](#) [nnz](#) = 0
- [uint64_t](#) [maxrow](#) = 0
- [uint64_t](#) [nElements](#) = 0
- [Sparse](#)<_Field, [SparseMatrix_t::CSR](#)> * [dat](#) = nullptr
- [Sparse](#)<_Field, [SparseMatrix_t::CSR_ZO](#)> * [one](#) = nullptr
- [Sparse](#)<_Field, [SparseMatrix_t::CSR_ZO](#)> * [mone](#) = nullptr

12.301.1 Member Typedef Documentation

12.301.1.1 Field

```
template<class _Field >
using Field = \_Field
```

12.301.1.2 Self_t

```
template<class _Field >
typedef Sparse<_Field, SparseMatrix\_t::HYB\_ZO> Self\_t
```

12.301.2 Field Documentation

12.301.2.1 delayed

```
template<class _Field >
bool delayed = false
```

12.301.2.2 kmax

```
template<class _Field >
uint64\_t kmax = 0
```

12.301.2.3 m

```
template<class _Field >
index\_t m = 0
```

12.301.2.4 n

```
template<class _Field >
index\_t n = 0
```

12.301.2.5 nnz

```
template<class _Field >
uint64\_t nnz = 0
```

12.301.2.6 maxrow

```
template<class _Field >
uint64\_t maxrow = 0
```

12.301.2.7 nElements

```
template<class _Field >
uint64_t nElements = 0
```

12.301.2.8 dat

```
template<class _Field >
Sparse<_Field, SparseMatrix_t::CSR>* dat = nullptr
```

12.301.2.9 one

```
template<class _Field >
Sparse<_Field, SparseMatrix_t::CSR_ZO>* one = nullptr
```

12.301.2.10 mone

```
template<class _Field >
Sparse<_Field, SparseMatrix_t::CSR_ZO>* mone = nullptr
```

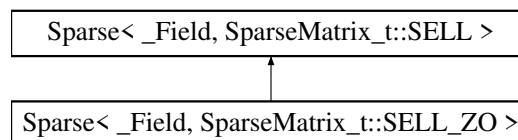
The documentation for this struct was generated from the following file:

- [hyb_zo.h](#)

12.302 Sparse< _Field, SparseMatrix_t::SELL > Struct Template Reference

```
#include <sell.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::SELL >:



Public Types

- using [Field](#) = [_Field](#)

Data Fields

- bool [delayed](#) = false
- int [chunk](#) = 0
- [index_t](#) [kmax](#) = 0
- [index_t](#) [m](#) = 0
- [index_t](#) [n](#) = 0
- [index_t](#) [maxrow](#) = 0
- [index_t](#) [sigma](#) = 0
- [index_t](#) [nChunks](#) = 0
- [uint64_t](#) [nnz](#) = 0
- [uint64_t](#) [nElements](#) = 0
- [index_t](#) * [perm](#) = nullptr
- [uint64_t](#) * [st](#) = nullptr
- [index_t](#) * [chunkSize](#) = nullptr
- [index_t](#) * [col](#) = nullptr
- [_Field::Element_ptr](#) [dat](#)

12.302.1 Member Typedef Documentation

12.302.1.1 Field

```
template<class _Field >
using Field = _Field
```

12.302.2 Field Documentation

12.302.2.1 delayed

```
template<class _Field >
bool delayed = false
```

12.302.2.2 chunk

```
template<class _Field >
int chunk = 0
```

12.302.2.3 kmax

```
template<class _Field >
index_t kmax = 0
```

12.302.2.4 m

```
template<class _Field >
index_t m = 0
```

12.302.2.5 n

```
template<class _Field >
index_t n = 0
```

12.302.2.6 maxrow

```
template<class _Field >
index_t maxrow = 0
```

12.302.2.7 sigma

```
template<class _Field >
index_t sigma = 0
```

12.302.2.8 nChunks

```
template<class _Field >
index_t nChunks = 0
```

12.302.2.9 nnz

```
template<class _Field >
uint64_t nnz = 0
```

12.302.2.10 nElements

```
template<class _Field >
uint64_t nElements = 0
```

12.302.2.11 perm

```
template<class _Field >
index_t* perm = nullptr
```

12.302.2.12 st

```
template<class _Field >
uint64_t* st = nullptr
```

12.302.2.13 chunkSize

```
template<class _Field >
index_t* chunkSize = nullptr
```

12.302.2.14 col

```
template<class _Field >
index_t* col = nullptr
```

12.302.2.15 dat

```
template<class _Field >
_Field::Element_ptr dat
```

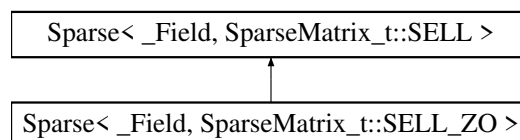
The documentation for this struct was generated from the following file:

- [sell.h](#)

12.303 Sparse< _Field, SparseMatrix_t::SELL_ZO > Struct Template Reference

```
#include <sell.h>
```

Inheritance diagram for Sparse< _Field, SparseMatrix_t::SELL_ZO >:

**Public Types**

- using [Field](#) = _Field

Data Fields

- _Field::Element [cst](#) = 1
- bool [delayed](#) = false
- int [chunk](#) = 0
- [index_t](#) [kmax](#) = 0
- [index_t](#) [m](#) = 0
- [index_t](#) [n](#) = 0
- [index_t](#) [maxrow](#) = 0
- [index_t](#) [sigma](#) = 0
- [index_t](#) [nChunks](#) = 0
- [uint64_t](#) [nnz](#) = 0
- [uint64_t](#) [nElements](#) = 0

- `index_t * perm` = nullptr
- `uint64_t * st` = nullptr
- `index_t * chunkSize` = nullptr
- `index_t * col` = nullptr
- `_Field::Element_ptr dat`

12.303.1 Member Typedef Documentation

12.303.1.1 Field

```
template<class _Field >
using Field = _Field
```

12.303.2 Field Documentation

12.303.2.1 cst

```
template<class _Field >
_Field::Element cst = 1
```

12.303.2.2 delayed

```
template<class _Field >
bool delayed = false [inherited]
```

12.303.2.3 chunk

```
template<class _Field >
int chunk = 0 [inherited]
```

12.303.2.4 kmax

```
template<class _Field >
index_t kmax = 0 [inherited]
```

12.303.2.5 m

```
template<class _Field >
index_t m = 0 [inherited]
```

12.303.2.6 n

```
template<class _Field >
index_t n = 0 [inherited]
```

12.303.2.7 maxrow

```
template<class _Field >
index_t maxrow = 0 [inherited]
```

12.303.2.8 sigma

```
template<class _Field >
index_t sigma = 0 [inherited]
```

12.303.2.9 nChunks

```
template<class _Field >
index_t nChunks = 0 [inherited]
```

12.303.2.10 nnz

```
template<class _Field >
uint64_t nnz = 0 [inherited]
```

12.303.2.11 nElements

```
template<class _Field >
uint64_t nElements = 0 [inherited]
```

12.303.2.12 perm

```
template<class _Field >
index_t* perm = nullptr [inherited]
```

12.303.2.13 st

```
template<class _Field >
uint64_t* st = nullptr [inherited]
```

12.303.2.14 chunkSize

```
template<class _Field >
index_t* chunkSize = nullptr [inherited]
```

12.303.2.15 col

```
template<class _Field >
index_t* col = nullptr [inherited]
```

12.303.2.16 dat

```
template<class _Field >
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- [sell.h](#)

12.304 SpMat< Field, flag > Struct Template Reference

```
#include <fflas_sparse.h>
```

Data Fields

- [FFLAS::CooMat< Field >](#) * _coo = nullptr
- [FFLAS::CsrMat< Field >](#) * _csr = nullptr
- [FFLAS::EllMat< Field >](#) * _ell = nullptr

12.304.1 Field Documentation**12.304.1.1 _coo**

```
template<class Field , int flag = HelperFlag::none>
FFLAS::CooMat<Field>* _coo = nullptr
```

12.304.1.2 _csr

```
template<class Field , int flag = HelperFlag::none>
FFLAS::CsrMat<Field>* _csr = nullptr
```

12.304.1.3 _ell

```
template<class Field , int flag = HelperFlag::none>
FFLAS::EllMat<Field>* _ell = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas_sparse.h](#)

12.305 Static_error_check< bool > Class Template Reference

```
#include <instrset.h>
```

Public Member Functions

- [Static_error_check\(\)](#)

12.305.1 Constructor & Destructor Documentation**12.305.1.1 Static_error_check()**

```
template<bool >
Static_error_check ( ) [inline]
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

12.306 Static_error_check< false > Class Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

12.307 StatsMatrix Struct Reference

```
#include <utils.h>
```

Data Fields

- [uint64_t rowdim](#) = 0
- [uint64_t coldim](#) = 0
- [uint64_t nOnes](#) = 0
- [uint64_t nMOnes](#) = 0
- [uint64_t nOthers](#) = 0
- [uint64_t nnz](#) = 0
- [uint64_t maxRow](#) = 0
- [uint64_t minRow](#) = 0
- [uint64_t averageRow](#) = 0
- [uint64_t deviationRow](#) = 0
- [uint64_t maxCol](#) = 0
- [uint64_t minCol](#) = 0
- [uint64_t averageCol](#) = 0
- [uint64_t deviationCol](#) = 0
- [uint64_t minColDifference](#) = 0
- [uint64_t maxColDifference](#) = 0
- [uint64_t averageColDifference](#) = 0
- [uint64_t deviationColDifference](#) = 0

- `uint64_t minRowDifference = 0`
- `uint64_t maxRowDifference = 0`
- `uint64_t averageRowDifference = 0`
- `uint64_t deviationRowDifference = 0`
- `uint64_t nDenseRows = 0`
- `uint64_t nDenseCols = 0`
- `uint64_t nEmptyRows = 0`
- `uint64_t nEmptyCols = 0`
- `uint64_t nEmptyColsEnd = 0`
- `std::vector< uint64_t > denseRows`
- `std::vector< uint64_t > denseCols`

12.307.1 Field Documentation

12.307.1.1 rowdim

`uint64_t rowdim = 0`

12.307.1.2 coldim

`uint64_t coldim = 0`

12.307.1.3 nOnes

`uint64_t nOnes = 0`

12.307.1.4 nMOnes

`uint64_t nMOnes = 0`

12.307.1.5 nOthers

`uint64_t nOthers = 0`

12.307.1.6 nnz

`uint64_t nnz = 0`

12.307.1.7 maxRow

`uint64_t maxRow = 0`

12.307.1.8 minRow

`uint64_t minRow = 0`

12.307.1.9 averageRow

`uint64_t averageRow = 0`

12.307.1.10 deviationRow

`uint64_t deviationRow = 0`

12.307.1.11 maxCol

`uint64_t maxCol = 0`

12.307.1.12 minCol

```
uint64_t minCol = 0
```

12.307.1.13 averageCol

```
uint64_t averageCol = 0
```

12.307.1.14 deviationCol

```
uint64_t deviationCol = 0
```

12.307.1.15 minColDifference

```
uint64_t minColDifference = 0
```

12.307.1.16 maxColDifference

```
uint64_t maxColDifference = 0
```

12.307.1.17 averageColDifference

```
uint64_t averageColDifference = 0
```

12.307.1.18 deviationColDifference

```
uint64_t deviationColDifference = 0
```

12.307.1.19 minRowDifference

```
uint64_t minRowDifference = 0
```

12.307.1.20 maxRowDifference

```
uint64_t maxRowDifference = 0
```

12.307.1.21 averageRowDifference

```
uint64_t averageRowDifference = 0
```

12.307.1.22 deviationRowDifference

```
uint64_t deviationRowDifference = 0
```

12.307.1.23 nDenseRows

```
uint64_t nDenseRows = 0
```

12.307.1.24 nDenseCols

```
uint64_t nDenseCols = 0
```

12.307.1.25 nEmptyRows

```
uint64_t nEmptyRows = 0
```

12.307.1.26 nEmptyCols

```
uint64_t nEmptyCols = 0
```

12.307.1.27 nEmptyColsEnd

```
uint64_t nEmptyColsEnd = 0
```

12.307.1.28 denseRows

```
std::vector<uint64_t> denseRows
```

12.307.1.29 denseCols

```
std::vector<uint64_t> denseCols
```

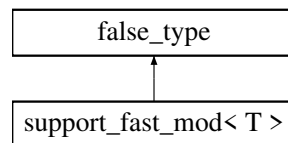
The documentation for this struct was generated from the following file:

- [utils.h](#)

12.308 support_fast_mod< T > Struct Template Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< T >:



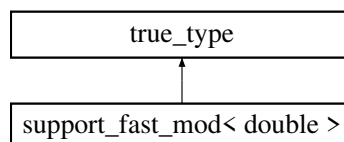
The documentation for this struct was generated from the following file:

- [fflas_freduce.h](#)

12.309 support_fast_mod< double > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< double >:



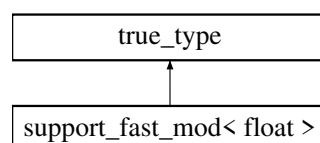
The documentation for this struct was generated from the following file:

- [fflas_freduce.h](#)

12.310 support_fast_mod< float > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< float >:



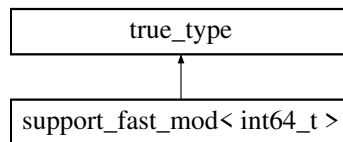
The documentation for this struct was generated from the following file:

- [fflas_freduce.h](#)

12.311 support_fast_mod< int64_t > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< int64_t >:



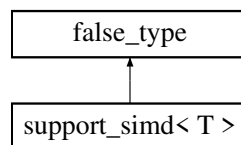
The documentation for this struct was generated from the following file:

- [fflas_freduce.h](#)

12.312 support_simd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

Inheritance diagram for support_simd< T >:



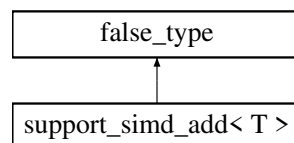
The documentation for this struct was generated from the following file:

- [fflas_simd.h](#)

12.313 support_simd_add< T > Struct Template Reference

```
#include <fflas_fadd.h>
```

Inheritance diagram for support_simd_add< T >:



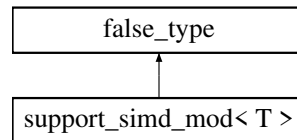
The documentation for this struct was generated from the following file:

- [fflas_fadd.h](#)

12.314 support_simd_mod< T > Struct Template Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_simd_mod< T >:



The documentation for this struct was generated from the following file:

- [fflas_freduce.h](#)

12.315 tfn_minus Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- `template<typename... Args>`
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

12.315.1 Member Function Documentation

12.315.1.1 operator>()()

```
template<typename... Args>
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.316 tfn_minus_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- `template<typename... Args>`
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

12.316.1 Member Function Documentation

12.316.1.1 operator>()()

```
template<typename... Args>
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.317 tfn_mul Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- `template<typename... Args>`
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

12.317.1 Member Function Documentation**12.317.1.1 [operator\(\)](#)()**

```
template<typename... Args>
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.318 tfn_mul_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- `template<typename... Args>`
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

12.318.1 Member Function Documentation**12.318.1.1 [operator\(\)](#)()**

```
template<typename... Args>
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.319 tfn_plus Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- `template<typename... Args>`
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

12.319.1 Member Function Documentation**12.319.1.1 [operator\(\)](#)()**

```
template<typename... Args>
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.320 tfn_plus_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

Public Member Functions

- `template<typename... Args>`
`auto operator() (Args &&... args) const -> decltype(operator+(std::forward<Args>(args)...))`

12.320.1 Member Function Documentation

12.320.1.1 operator>()()

```
template<typename... Args>
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse_matrix_traits.h](#)

12.321 Threads Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.322 ThreeD Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.323 ThreeDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.324 ThreeDInPlace Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.325 TRSMHelper< ReclterTrait, ParSeqTrait > Struct Template Reference

TRSM Helper.

Public Member Functions

- `template<class Cut , class Param >`
`TRSMHelper (ParSeqHelper::Parallel< Cut, Param > _PS)`
- `TRSMHelper (ParSeqHelper::Sequential _PS)`
- `template<typename RIT , typename PST >`
`TRSMHelper (TRSMHelper< RIT, PST > &_TH)`

- `template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value> FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (Dom &D, size_t m, size_t k, size_t n, ParSeqTrait p) const`
- `template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value> FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (Dom &D, size_t m, size_t k, size_t n) const`

Data Fields

- ParSeqTrait [parseq](#)

12.325.1 Detailed Description

`template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>`

`struct FFLAS::TRSMHelper< RecIterTrait, ParSeqTrait >`

TRSM Helper.

12.325.2 Constructor & Destructor Documentation

12.325.2.1 TRSMHelper() [1/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
template<class Cut , class Param >
TRSMHelper (
    ParSeqHelper::Parallel< Cut, Param > _PS ) [inline]
```

12.325.2.2 TRSMHelper() [2/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
TRSMHelper (
    ParSeqHelper::Sequential _PS ) [inline]
```

12.325.2.3 TRSMHelper() [3/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
template<typename RIT , typename PST >
TRSMHelper (
    TRSMHelper< RIT, PST > & _TH ) [inline]
```

12.325.3 Member Function Documentation

12.325.3.1 pMMH() [1/2]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value>
FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (
    Dom & D,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait p ) const [inline]
```

12.325.3.2 pMMH() [2/2]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq↵
Helper::Sequential>
template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS↵
::ModeTraits<Dom>::value>
FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (
    Dom & D,
    size_t m,
    size_t k,
    size_t n ) const [inline]
```

12.325.4 Field Documentation

12.325.4.1 parseq

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq↵
Helper::Sequential>
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.326 TwoD Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.327 TwoDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

12.328 UnparametricTag Struct Reference

If the field uses a representation with infix operators.

```
#include <field-traits.h>
```

12.328.1 Detailed Description

If the field uses a representation with infix operators.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

12.329 Winograd Struct Reference

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

12.330 WinogradPar Struct Reference

The documentation for this struct was generated from the following file:

- [fflas_helpers.inl](#)

Chapter 13

File Documentation

13.1 arithprog.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

Macros

- #define CUBE(x) ((x)*(x)*(x))
- #define GFOPS(m, n, r, t) (2.7*CUBE(double(n)/1000.0))/t

Typedefs

- typedef Givaro::Timer TTimer

Functions

- int main (int argc, char **argv)

13.1.1 Macro Definition Documentation

13.1.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

13.1.1.2 GFOPS

```
#define GFOPS(  
    m,  
    n,  
    r,  
    t )  (2.7*CUBE(double(n)/1000.0))/t
```

13.1.2 Typedef Documentation

13.1.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.1.3 Function Documentation

13.1.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.2 fsyrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <ctime>
```

Macros

- #define CUBE(x) ((x)*(x)*(x))
- #define GFOPS(n, t) (CUBE(double(n)/1000.0)/(3.0*t))

Typedefs

- typedef Givaro::Timer TTimer

Functions

- int main ()

13.2.1 Macro Definition Documentation

13.2.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

13.2.1.2 GFOPS

```
#define GFOPS(
    n,
    t ) (CUBE(double(n)/1000.0)/(3.0*t))
```

13.2.2 Typedef Documentation

13.2.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.2.3 Function Documentation

13.2.3.1 main()

```
int main (
    void )
```


13.3 fsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <ctime>
```

Macros

- `#define CUBE(x) ((x)*(x)*(x))`
- `#define GFOPS(n, t) (CUBE(double(n)/1000.0)/(3.0*t))`

Typedefs

- `typedef Givaro::Timer TTimer`

Functions

- `int main ()`

13.3.1 Macro Definition Documentation

13.3.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

13.3.1.2 GFOPS

```
#define GFOPS(
    n,
    t ) (CUBE(double(n)/1000.0)/(3.0*t))
```

13.3.2 Typedef Documentation

13.3.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.3.3 Function Documentation

13.3.3.1 main()

```
int main (
    void )
```

13.4 ftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

Macros

- `#define CUBE(x) ((x)*(x)*(x))`
- `#define GFOPS(n, t) (CUBE(double(n)/1000.0)/(3.0*t))`

Typedefs

- `typedef Givaro::Timer TTimer`

Functions

- `int main ()`

13.4.1 Macro Definition Documentation

13.4.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

13.4.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t )  (CUBE(double(n)/1000.0)/(3.0*t))
```

13.4.2 Typedef Documentation

13.4.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.4.3 Function Documentation

13.4.3.1 main()

```
int main (  
    void )
```

13.5 winograd.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include "fflas-ffpack/utils/fflas_randommatrix.h"  
#include <iostream>  
#include <fstream>  
#include <givaro/modular.h>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/fflas/fflas.h"  
#include <ctime>
```

Macros

- `#define DOUBLE_TO_FLOAT_CROSSOVER 0`
- `#define GFOPS(n, t) (2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)`

Typedefs

- `typedef Givaro::Timer TTimer`

Functions

- template<class [Field](#) >
bool [balanced](#) (const [Field](#) &)
- template<class T >
bool [balanced](#) (const [Givaro::ModularBalanced](#)< T > &)
- int [main](#) ()

13.5.1 Macro Definition Documentation

13.5.1.1 DOUBLE_TO_FLOAT_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 0
```

13.5.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t ) (2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)
```

13.5.2 Typedef Documentation

13.5.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.5.3 Function Documentation

13.5.3.1 balanced() [1/2]

```
template<class Field >  
bool balanced (  
    const Field & )
```

13.5.3.2 balanced() [2/2]

```
template<class T >  
bool balanced (  
    const Givaro::ModularBalanced< T > & )
```

13.5.3.3 main()

```
int main (  
    void )
```

13.6 benchmark-charpoly-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/Matio.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [__FFLASFFPACK_FORCE_SEQ](#)

Functions

- int [main](#) (int argc, char **argv)

13.6.1 Macro Definition Documentation

13.6.1.1 __FFLASFFPACK_FORCE_SEQ

```
#define __FFLASFFPACK_FORCE_SEQ
```

13.6.2 Function Documentation

13.6.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.7 benchmark-charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givpoly1.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET](#) 1

Functions

- template<class [Field](#) >
void [run_with_field](#) (int q, size_t bits, size_t n, size_t d, size_t iter, std::string file, int variant)
- int [main](#) (int argc, char **argv)

13.7.1 Macro Definition Documentation

13.7.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.7.2 Function Documentation

13.7.2.1 run_with_field()

```
template<class Field >
void run_with_field (
    int q,
    size_t bits,
    size_t n,
    size_t d,
    size_t iter,
    std::string file,
    int variant )
```

13.7.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

13.8 benchmark-checkers.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include <fstream>
```

Macros

- `#define ENABLE_ALL_CHECKINGS 1`
- `#define _NR_TESTS 5`
- `#define _MAX_SIZE_MATRICES 1000`
- `#define CUBE(x) ((x)*(x)*(x))`

Functions

- int [main](#) (int argc, char **argv)

13.8.1 Macro Definition Documentation

13.8.1.1 [ENABLE_ALL_CHECKINGS](#)

```
#define ENABLE\_ALL\_CHECKINGS 1
```

13.8.1.2 [_NR_TESTS](#)

```
#define \_NR\_TESTS 5
```

13.8.1.3 [_MAX_SIZE_MATRICES](#)

```
#define \_MAX\_SIZE\_MATRICES 1000
```

13.8.1.4 [CUBE](#)

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

13.8.2 Function Documentation

13.8.2.1 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

13.9 benchmark-dgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [CBLAS_GEMM](#) cblas_dgemm

Typedefs

- typedef [FFLAS::Timer](#) TTimer
- typedef double [Floats](#)

Functions

- int [main](#) (int argc, char **argv)

13.9.1 Macro Definition Documentation

13.9.1.1 CBLAS_GEMM

```
#define CBLAS_GEMM cblas_dgemm
```

13.9.2 Typedef Documentation

13.9.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

13.9.2.2 Floats

```
typedef double Floats
```

13.9.3 Function Documentation

13.9.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.10 benchmark-dgetrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- `#define __FFLASFFPACK_HAVE_DGETRF 1`

Typedefs

- `typedef FFLAS::Timer TTimer`

Functions

- `int main (int argc, char **argv)`

13.10.1 Macro Definition Documentation**13.10.1.1 __FFLASFFPACK_HAVE_DGETRF**

```
#define __FFLASFFPACK_HAVE_DGETRF 1
```

13.10.2 Typedef Documentation**13.10.2.1 TTimer**

```
typedef FFLAS::Timer TTimer
```

13.10.3 Function Documentation**13.10.3.1 main()**

```
int main (
    int argc,
    char ** argv )
```

13.11 benchmark-dgetri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Typedefs

- `typedef FFLAS::Timer TTimer`

Functions

- `int main (int argc, char **argv)`

13.11.1 Typedef Documentation**13.11.1.1 TTimer**

```
typedef FFLAS::Timer TTimer
```

13.11.2 Function Documentation

13.11.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.12 benchmark-dsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [EFFGFF](#)(n, t, i) ((double(n)/1000.*double(n)/1000.*double(n)/1000.0) / double(t) * double(i) / 3.)

Typedefs

- typedef [FFLAS::Timer](#) [TTimer](#)

Functions

- int [main](#) (int argc, char **argv)

13.12.1 Macro Definition Documentation

13.12.1.1 EFFGFF

```
#define EFFGFF(
    n,
    t,
    i ) ( (double(n)/1000.*double(n)/1000.*double(n)/1000.0) / double(t) * double(i)
/ 3.)
```

13.12.2 Typedef Documentation

13.12.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

13.12.3 Function Documentation

13.12.3.1 main()

```
int main (
    int argc,
    char ** argv )
```


13.13 benchmark-dtrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Typedefs

- typedef [FFLAS::Timer](#) [TTimer](#)

Functions

- int [main](#) (int argc, char **argv)

13.13.1 Typedef Documentation

13.13.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

13.13.2 Function Documentation

13.13.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.14 benchmark-dtrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [__FFLASFFPACK_HAVE_DTRTRI](#) 1

Typedefs

- typedef [FFLAS::Timer](#) [TTimer](#)

Functions

- int [main](#) (int argc, char **argv)

13.14.1 Macro Definition Documentation

13.14.1.1 __FFLASFFPACK_HAVE_DTRTRI

```
#define __FFLASFFPACK_HAVE_DTRTRI 1
```

13.14.2 Typedef Documentation

13.14.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

13.14.3 Function Documentation

13.14.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.15 benchmark-fadd-lvl2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET](#) 1

Functions

- int [main](#) (int argc, char **argv)

13.15.1 Macro Definition Documentation

13.15.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.15.2 Function Documentation

13.15.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.16 benchmark-fdot.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givrational.h>
```

```
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

Functions

- `template<class Field >`
`Field::Element run_with_field (int q, size_t iter, size_t N, const size_t BS, const size_t p, const size_t threads)`
- `int main (int argc, char **argv)`

13.16.1 Macro Definition Documentation

13.16.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.16.2 Function Documentation

13.16.2.1 run_with_field()

```
template<class Field >
Field::Element run_with_field (
    int q,
    size_t iter,
    size_t N,
    const size_t BS,
    const size_t p,
    const size_t threads )
```

13.16.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

13.17 benchmark-fgemm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`
- `#define MG_DEFAULT MG_ACTIVE`
- `#define STD_RECINT_SIZE 8`

Functions

- `template<typename Ints >`
`int tmain ()`
- `int main (int argc, char **argv)`

13.17.1 Macro Definition Documentation**13.17.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.17.1.2 MG_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

13.17.1.3 STD_RECINT_SIZE

```
#define STD_RECINT_SIZE 8
```

13.17.2 Function Documentation**13.17.2.1 tmain()**

```
template<typename Ints >
int tmain ( )
```

13.17.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

13.18 benchmark-fgemm-rns.C File Reference

```
#include "fflas-ffpack/fflas/fflas.h"
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

Typedefs

- `typedef FFPACK::rns_double RNS`
- `typedef FFPACK::RNSInteger< RNS > Field`
- `typedef Field::Element_ptr Element_ptr`
- `typedef Field::ConstElement_ptr ConstElement_ptr`
- `typedef StrategyParameter::Threads THREADS`
- `typedef StrategyParameter::Grain GRAIN`

- typedef [StrategyParameter::TwoD](#) TWOD
- typedef [StrategyParameter::TwoDAdaptive](#) TWODA
- typedef [StrategyParameter::ThreeD](#) THREED
- typedef [StrategyParameter::ThreeDAdaptive](#) THREEDA
- typedef [StrategyParameter::ThreeDInPlace](#) THREEDIP
- typedef [ParSeqHelper::Sequential](#) PSeq

Functions

- int [main](#) (int argc, char *argv[])

13.18.1 Macro Definition Documentation

13.18.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.18.2 Typedef Documentation

13.18.2.1 RNS

```
typedef FFPACK::rns\_double RNS
```

13.18.2.2 Field

```
typedef FFPACK::RNSInteger<RNS> Field
```

13.18.2.3 Element_ptr

```
typedef Field::Element\_ptr Element_ptr
```

13.18.2.4 ConstElement_ptr

```
typedef Field::ConstElement\_ptr ConstElement_ptr
```

13.18.2.5 THREADS

```
typedef StrategyParameter::Threads THREADS
```

13.18.2.6 GRAIN

```
typedef StrategyParameter::Grain GRAIN
```

13.18.2.7 TWOD

```
typedef StrategyParameter::TwoD TWOD
```

13.18.2.8 TWODA

```
typedef StrategyParameter::TwoDAdaptive TWODA
```

13.18.2.9 THREED

```
typedef StrategyParameter::ThreeD THREED
```

13.18.2.10 THREEDA

```
typedef StrategyParameter::ThreeDAdaptive THREEDA
```

13.18.2.11 THREEDIP

```
typedef StrategyParameter::ThreeDInPlace THREEDIP
```

13.18.2.12 PSeq

```
typedef ParSeqHelper::Sequential PSeq
```

13.18.3 Function Documentation

13.18.3.1 main()

```
int main (
    int argc,
    char * argv[] )
```

13.19 benchmark-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [CLASSIC_HYBRID](#)

Functions

- int [main](#) (int argc, char **argv)

13.19.1 Macro Definition Documentation

13.19.1.1 CLASSIC_HYBRID

```
#define CLASSIC_HYBRID
```

13.19.2 Function Documentation

13.19.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.20 benchmark-fgemv-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
```

```
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

Macros

- #define `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET` 1
- #define `MG_DEFAULT` `MG_ACTIVE`
- #define `STD_RECINT_SIZE` 8

Functions

- template<typename T >
std::ostream & `write_matrix` (std::ostream &out, Givaro::Integer p, size_t m, size_t n, T *C, size_t ldc)
- template<typename Ints >
int `tmain` ()
- int `main` (int argc, char **argv)

13.20.1 Macro Definition Documentation

13.20.1.1 `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET`

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.20.1.2 `MG_DEFAULT`

```
#define MG_DEFAULT MG_ACTIVE
```

13.20.1.3 `STD_RECINT_SIZE`

```
#define STD_RECINT_SIZE 8
```

13.20.2 Function Documentation

13.20.2.1 `write_matrix()`

```
template<typename T >
std::ostream & write_matrix (
    std::ostream & out,
    Givaro::Integer p,
    size_t m,
    size_t n,
    T * C,
    size_t ldc )
```

13.20.2.2 `tmain()`

```
template<typename Ints >
int tmain ( )
```

13.20.2.3 `main()`

```
int main (
    int argc,
    char ** argv )
```

13.21 benchmark-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
```

Data Structures

- struct [need_field_characteristic](#)< Field >
- struct [need_field_characteristic](#)< Givaro::Modular< Field > >
- struct [need_field_characteristic](#)< Givaro::ModularBalanced< Field > >
- struct [compatible_data_type](#)< Field >
- struct [compatible_data_type](#)< Givaro::ZRing< float > >
- struct [compatible_data_type](#)< Givaro::ZRing< double > >

Macros

- #define [__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET](#) 1

Functions

- template<class [Field](#) , class Randlter , class Matrix , class Vector >
void [fill_value](#) ([Field](#) &F, Randlter &Rand, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK)
- template<class [Field](#) , class Matrix , class Vector >
void [genData](#) ([Field](#) &F, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK, int bitsize, [uint64_t](#) seed)
- template<class [Field](#) , class Matrix , class Vector >
bool [check_result](#) ([Field](#) &F, size_t m, size_t lda, Matrix &A, Vector &X, size_t incX, Vector &Y, size_t incY)
- template<class [Field](#) , class Matrix , class Vector >
bool [benchmark_with_timer](#) ([Field](#) &F, int p, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, size_t iters, int t, double &time, size_t GrainSize)
- template<class [Field](#) , class arg >
void [benchmark_disp](#) ([Field](#) &F, bool pass, double &time, size_t iters, int p, size_t m, size_t k, arg &as)
- template<class [Field](#) , class arg >
void [benchmark_in_Field](#) ([Field](#) &F, int p, size_t m, size_t k, int NBK, int bitsize, [uint64_t](#) seed, size_t iters, int t, arg &as, size_t GrainSize)
- template<class [Field](#) , class arg >
void [benchmark_with_field](#) (int p, size_t m, size_t k, int NBK, int bitsize, [uint64_t](#) seed, size_t iters, int t, arg &as, size_t GrainSize)
- template<class [Field](#) , class arg >
void [benchmark_with_field](#) (const Givaro::Integer &q, int p, size_t m, size_t k, int NBK, int bitsize, [uint64_t](#) seed, size_t iters, int t, arg &as, size_t GrainSize)
- int [main](#) (int argc, char **argv)

13.21.1 Macro Definition Documentation

13.21.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```


13.21.2 Function Documentation

13.21.2.1 fill_value()

```
template<class Field , class RandIter , class Matrix , class Vector >
void fill_value (
    Field & F,
    RandIter & Rand,
    Matrix & A,
    Vector & X,
    Vector & Y,
    size_t m,
    size_t k,
    size_t incX,
    size_t incY,
    size_t lda,
    int NBK )
```

13.21.2.2 genData()

```
template<class Field , class Matrix , class Vector >
void genData (
    Field & F,
    Matrix & A,
    Vector & X,
    Vector & Y,
    size_t m,
    size_t k,
    size_t incX,
    size_t incY,
    size_t lda,
    int NBK,
    int bitsize,
    uint64_t seed )
```

13.21.2.3 check_result()

```
template<class Field , class Matrix , class Vector >
bool check_result (
    Field & F,
    size_t m,
    size_t lda,
    Matrix & A,
    Vector & X,
    size_t incX,
    Vector & Y,
    size_t incY )
```

13.21.2.4 benchmark_with_timer()

```
template<class Field , class Matrix , class Vector >
bool benchmark_with_timer (
    Field & F,
    int p,
    Matrix & A,
    Vector & X,
    Vector & Y,
    size_t m,
    size_t k,
    size_t incX,
```

```

    size_t incY,
    size_t lda,
    size_t iters,
    int t,
    double & time,
    size_t GrainSize )

```

13.21.2.5 benchmark_disp()

```

template<class Field , class arg >
void benchmark_disp (
    Field & F,
    bool pass,
    double & time,
    size_t iters,
    int p,
    size_t m,
    size_t k,
    arg & as )

```

13.21.2.6 benchmark_in_Field()

```

template<class Field , class arg >
void benchmark_in_Field (
    Field & F,
    int p,
    size_t m,
    size_t k,
    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )

```

13.21.2.7 benchmark_with_field() [1/2]

```

template<class Field , class arg >
void benchmark_with_field (
    int p,
    size_t m,
    size_t k,
    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )

```

13.21.2.8 benchmark_with_field() [2/2]

```

template<class Field , class arg >
void benchmark_with_field (
    const Givaro::Integer & q,
    int p,
    size_t m,
    size_t k,

```

```

    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )

```

13.21.2.9 main()

```

int main (
    int argc,
    char ** argv )

```

13.22 benchmark-fgesv.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"

```

Macros

- #define `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET` 1

Functions

- int `main` (int argc, char **argv)

13.22.1 Macro Definition Documentation

13.22.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.22.2 Function Documentation

13.22.2.1 main()

```

int main (
    int argc,
    char ** argv )

```

13.23 benchmark-fsyrk.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"

```

Macros

- #define [__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET](#) 1
- #define [CUBE](#)(x) ((x)*(x)*(x))

Functions

- int [main](#) (int argc, char **argv)

13.23.1 Macro Definition Documentation**13.23.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.23.1.2 CUBE

```
#define CUBE(
    x )  ((x)*(x)*(x))
```

13.23.2 Function Documentation**13.23.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

13.24 benchmark-fsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [__FFPACK_FSYTRF_BC_CROUT](#)
- #define [__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET](#) 1
- #define [CUBE](#)(x) ((x)*(x)*(x))

Functions

- int [main](#) (int argc, char **argv)

13.24.1 Macro Definition Documentation**13.24.1.1 __FFPACK_FSYTRF_BC_CROUT**

```
#define __FFPACK_FSYTRF_BC_CROUT
```

13.24.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.24.1.3 CUBE

```
#define CUBE(
    x )  ((x)*(x)*(x))
```

13.24.2 Function Documentation

13.24.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.25 benchmark-fftrsm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
```

Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

Functions

- `int main (int argc, char **argv)`

13.25.1 Macro Definition Documentation

13.25.1.1 `__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET`

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.25.2 Function Documentation

13.25.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.26 benchmark-fftrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

Functions

- `int main (int argc, char **argv)`

13.26.1 Macro Definition Documentation**13.26.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.26.2 Function Documentation**13.26.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

13.27 benchmark-ftsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

Functions

- `int main (int argc, char **argv)`

13.27.1 Macro Definition Documentation**13.27.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.27.2 Function Documentation**13.27.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

13.28 benchmark-ftytri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
```

```
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET](#) 1
- #define [CUBE](#)(x) ((x)*(x)*(x))

Functions

- int [main](#) (int argc, char **argv)

13.28.1 Macro Definition Documentation

13.28.1.1 [__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET](#)

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.28.1.2 [CUBE](#)

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

13.28.2 Function Documentation

13.28.2.1 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

13.29 benchmark-inverse.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [CUBE](#)(x) ((x)*(x)*(x))

Functions

- int [main](#) (int argc, char **argv)

13.29.1 Macro Definition Documentation

13.29.1.1 [CUBE](#)

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

13.29.2 Function Documentation

13.29.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.30 benchmark-lqup-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
```

Functions

- int [main](#) (int argc, char **argv)

13.30.1 Function Documentation

13.30.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.31 benchmark-lqup.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define [CUBE](#)(x) ((x)*(x)*(x))

Functions

- int [main](#) (int argc, char **argv)

13.31.1 Macro Definition Documentation

13.31.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```


13.31.2 Function Documentation

13.31.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.32 benchmark-pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include <givaro/givranditer.h>
#include <iostream>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`
- `#define CUBE(x) ((x)*(x)*(x))`

Typedefs

- `typedef Givaro::ModularBalanced< double > Field`

Functions

- `void verification_PLUQ (const Field &F, typename Field::Element *B, typename Field::Element *A, size_t *P, size_t *Q, size_t m, size_t n, size_t R)`
- `void Rec_Initialize (Field &F, Field::Element *C, size_t m, size_t n, size_t ldc)`
- `int main (int argc, char **argv)`

13.32.1 Macro Definition Documentation

13.32.1.1 [__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET](#)

```
#define \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET 1
```

13.32.1.2 [CUBE](#)

```
#define CUBE(
    x )  ((x)*(x)*(x))
```

13.32.2 Typedef Documentation

13.32.2.1 [Field](#)

```
typedef Givaro::ModularBalanced<double> Field
```

13.32.3 Function Documentation

13.32.3.1 verification_PLUQ()

```
void verification_PLUQ (
    const Field & F,
    typename Field::Element * B,
    typename Field::Element * A,
    size_t * P,
    size_t * Q,
    size_t m,
    size_t n,
    size_t R )
```

13.32.3.2 Rec_Initialize()

```
void Rec_Initialize (
    Field & F,
    Field::Element * C,
    size_t m,
    size_t n,
    size_t ldc )
```

13.32.3.3 main()

```
int main (
    int argc,
    char ** argv )
```

13.33 benchmark-wino.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

Macros

- #define CUBE(x) ((x)*(x)*(x))

Functions

- template<class Field >
void launch_wino (const Field &F, const size_t &n, const size_t &NB, const size_t &wino, const bool &asmax, const size_t &seed, const bool compare)
- int main (int argc, char **argv)

13.33.1 Macro Definition Documentation

13.33.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

13.33.2 Function Documentation

13.33.2.1 launch_wino()

```
template<class Field >
void launch_wino (
    const Field & F,
    const size_t & n,
    const size_t & NB,
    const size_t & wino,
    const bool & asmax,
    const size_t & seed,
    const bool compare )
```

13.33.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

13.34 mainpage.dox File Reference

13.35 charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

Macros

- #define CUBE(x) ((x)*(x)*(x))
- #define GFOPS(m, n, r, t) (2.7*CUBE(double(n)/1000.0))/t

Typedefs

- typedef Givaro::Timer TTimer

Functions

- int main ()

13.35.1 Macro Definition Documentation

13.35.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

13.35.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
```

```

r,
t ) (2.7*CUBE(double(n)/1000.0))/t

```

13.35.2 Typedef Documentation

13.35.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.35.3 Function Documentation

13.35.3.1 main()

```

int main (
    void )

```

13.36 charpoly.C File Reference

```

#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"

```

Functions

- int [main](#) (int argc, char **argv)

This example computes the characteristic polynomial of a matrix over a defined finite field.

13.36.1 Function Documentation

13.36.1.1 main()

```

int main (
    int argc,
    char ** argv )

```

This example computes the characteristic polynomial of a matrix over a defined finite field. Outputs the characteristic polynomial.

13.37 det.C File Reference

```

#include <givaro/modular.h>
#include <iostream>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"

```

Functions

- int [main](#) (int argc, char **argv)

This example computes the determinant of a matrix over a defined finite field.

13.37.1 Function Documentation

13.37.1.1 main()

```

int main (
    int argc,
    char ** argv )

```

This example computes the determinant of a matrix over a defined finite field.
Outputs the determinant.

13.38 matmul.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int [main](#) (int argc, char **argv)

This example computes the matrix multiplication over a defined finite field.

13.38.1 Function Documentation

13.38.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the matrix multiplication over a defined finite field.
Outputs the product of the matrix given as input.

13.39 pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

Macros

- #define [CUBE](#)(x) ((x)*(x)*(x))
- #define [GFOPS](#)(m, n, r, t) (2.0/3.0*[CUBE](#)(double(n)/1000.0) +2*m/1000.0*n/1000.0*double(r)/1000.0 - double(r)/1000.0*double(r)/1000.0*(m+n)/1000)/t

Typedefs

- typedef Givaro::Timer [TTimer](#)

Functions

- int [main](#) ()

13.39.1 Macro Definition Documentation

13.39.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

13.39.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
    r,
    t ) (2.0/3.0*CUBE(double(n)/1000.0) +2*m/1000.0*n/1000.0*double(r)/1000.0 - double(r)/1000.0↵
0*double(r)/1000.0*(m+n)/1000)/t
```

13.39.2 Typedef Documentation

13.39.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

13.39.3 Function Documentation

13.39.3.1 main()

```
int main (
    void )
```

13.40 pluq.C File Reference

```
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Functions

- int [main](#) (int argc, char **argv)

13.40.1 Function Documentation

13.40.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

13.41 rank.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int [main](#) (int argc, char **argv)

This example computes the rank of a matrix over a defined finite field.

13.41.1 Function Documentation

13.41.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the rank of a matrix over a defined finite field.
Outputs the rank.

13.42 solve.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int [main](#) (int argc, char **argv)

This example solve the quare system defined by the input over a defined finite field.

13.42.1 Function Documentation

13.42.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example solve the quare system defined by the input over a defined finite field.

13.43 checker_charpoly.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class [CheckerImplem_charpoly](#)< [Field](#), [Polynomial](#) >

Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define [__FFLASFFPACK_checker_charpoly_INL](#)

13.43.1 Macro Definition Documentation

13.43.1.1 __FFLASFFPACK_checker_charpoly_INL

```
#define __FFLASFFPACK_checker_charpoly_INL
```

13.44 checker_det.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class [CheckerImplem_Det< Field >](#)

Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- `#define` [__FFLASFFPACK_checker_det_INL](#)

13.44.1 Macro Definition Documentation**13.44.1.1 __FFLASFFPACK_checker_det_INL**

```
#define __FFLASFFPACK_checker_det_INL
```

13.45 checker_empty.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
```

Data Structures

- struct [Checker_Empty< Field >](#)

Namespaces

- namespace [FFLAS](#)

13.46 checker_fgemm.inl File Reference**Data Structures**

- class [CheckerImplem_fgemm< Field >](#)

Namespaces

- namespace [FFLAS](#)

Macros

- `#define` [__FFLASFFPACK_checker_fgemm_INL](#)

13.46.1 Macro Definition Documentation**13.46.1.1 __FFLASFFPACK_checker_fgemm_INL**

```
#define __FFLASFFPACK_checker_fgemm_INL
```

13.47 checker_ftrsm.inl File Reference**Data Structures**

- class [CheckerImplem_ftrsm< Field >](#)

Namespaces

- namespace [FFLAS](#)

Macros

- `#define __FFLASFFPACK_checker_ftsm_INL`

13.47.1 Macro Definition Documentation**13.47.1.1 __FFLASFFPACK_checker_ftsm_INL**

```
#define __FFLASFFPACK_checker_ftsm_INL
```

13.48 checker_invert.inl File Reference**Data Structures**

- class [CheckerImplem_invert< Field >](#)

Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- `#define __FFLASFFPACK_checker_invert_INL`

13.48.1 Macro Definition Documentation**13.48.1.1 __FFLASFFPACK_checker_invert_INL**

```
#define __FFLASFFPACK_checker_invert_INL
```

13.49 checker_pluq.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

Data Structures

- class [CheckerImplem_PLUQ< Field >](#)

Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- `#define __FFLASFFPACK_checker_pluq_INL`

13.49.1 Macro Definition Documentation**13.49.1.1 __FFLASFFPACK_checker_pluq_INL**

```
#define __FFLASFFPACK_checker_pluq_INL
```

13.50 checkers.doxy File Reference

13.51 checkers_fflas.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
```

Data Structures

- class [FailureFgemmCheck](#)
- class [FailureTrsmCheck](#)

Namespaces

- namespace [FFLAS](#)

Typedefs

- template<class [Field](#) >
using [Checker_fgemm](#) = [FFLAS::Checker_Empty](#)< [Field](#) >
- template<class [Field](#) >
using [Checker_ftrsm](#) = [FFLAS::Checker_Empty](#)< [Field](#) >

13.52 checkers_fflas.inl File Reference

```
#include "checker_fgemm.inl"
#include "checker_ftrsm.inl"
```

Namespaces

- namespace [FFLAS](#)

Macros

- #define [FFLASFFPACK_checkers_fflas_inl_H](#)

Typedefs

- template<class [Field](#) >
using [ForceCheck_fgemm](#) = [CheckerImplem_fgemm](#)< [Field](#) >
- template<class [Field](#) >
using [ForceCheck_ftrsm](#) = [CheckerImplem_ftrsm](#)< [Field](#) >

13.52.1 Macro Definition Documentation

13.52.1.1 FFLASFFPACK_checkers_fflas_inl_H

```
#define FFLASFFPACK_checkers_fflas_inl_H
```

13.53 checkers_ffpack.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class [FailurePLUQCheck](#)
- class [FailureDetCheck](#)
- class [FailureInvertCheck](#)
- class [FailureCharpolyCheck](#)

Namespaces

- namespace [FFPACK](#)
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

Typedefs

- `template<class Field >`
 using [Checker_PLUQ](#) = [FFLAS::Checker_Empty](#)< [Field](#) >
- `template<class Field >`
 using [Checker_Det](#) = [FFLAS::Checker_Empty](#)< [Field](#) >
- `template<class Field >`
 using [Checker_invert](#) = [FFLAS::Checker_Empty](#)< [Field](#) >
- `template<class Field , class Polynomial >`
 using [Checker_charpoly](#) = [FFLAS::Checker_Empty](#)< [Field](#) >

13.54 checkers_ffpack.inl File Reference

```
#include "checker_pluq.inl"
#include "checker_det.inl"
#include "checker_invert.inl"
#include "checker_charpoly.inl"
```

Namespaces

- namespace [FFPACK](#)
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- `#define FFLASFFPACK_checkers_ffpack_inl_H`

Typedefs

- `template<class Field >`
 using [ForceCheck_PLUQ](#) = [CheckerImplem_PLUQ](#)< [Field](#) >
- `template<class Field >`
 using [ForceCheck_Det](#) = [CheckerImplem_Det](#)< [Field](#) >
- `template<class Field >`
 using [ForceCheck_invert](#) = [CheckerImplem_invert](#)< [Field](#) >
- `template<class Field , class Polynomial >`
 using [ForceCheck_charpoly](#) = [CheckerImplem_charpoly](#)< [Field](#) , [Polynomial](#) >

13.54.1 Macro Definition Documentation

13.54.1.1 FFLASFFPACK_checkers_ffpack_inl_H

```
#define FFLASFFPACK_checkers_ffpack_inl_H
```

13.55 config-blas.h File Reference

Macros

- #define [CBLAS_INT](#) int
- #define [CBLAS_ENUM_DEFINED_H](#)
- #define [CBLAS_EXTERNALS](#)
- #define [blas_enum](#) enum

Enumerations

- enum [CBLAS_ORDER](#) { [CblasRowMajor](#) =101 , [CblasColMajor](#) =102 }
- enum [CBLAS_TRANSPOSE](#) { [CblasNoTrans](#) =111 , [CblasTrans](#) =112 , [CblasConjTrans](#) =113 , [AtlasConj](#) =114 }
- enum [CBLAS_UPLO](#) { [CblasUpper](#) =121 , [CblasLower](#) =122 }
- enum [CBLAS_DIAG](#) { [CblasNonUnit](#) =131 , [CblasUnit](#) =132 }
- enum [CBLAS_SIDE](#) { [CblasLeft](#) =141 , [CblasRight](#) =142 }

Functions

- void [daxpy_](#) (const int *, const double *, const double *, const int *, double *, const int *)
- void [saxpy_](#) (const int *, const float *, const float *, const int *, float *, const int *)
- double [ddot_](#) (const int *, const double *, const int *, const double *, const int *)
- float [sdot_](#) (const int *, const float *, const int *, const float *, const int *)
- double [dasum_](#) (const int *, const double *, const int *)
- int [idamax_](#) (const int *, const double *, const int *)
- double [dnrm2_](#) (const int *, const double *, const int *)
- void [dgemv_](#) (const char *, const int *, const int *, const double *, const double *, const int *, const double *, const int *, const double *, double *, const int *)
- void [sgemv_](#) (const char *, const int *, const int *, const float *, const float *, const int *, const float *, const int *, const float *, float *, const int *)
- void [dger_](#) (const int *, const int *, const double *, const double *, const int *, const double *, const int *, double *, const int *)
- void [sger_](#) (const int *, const int *, const float *, const float *, const int *, const float *, const int *, float *, const int *)
- void [dcopy_](#) (const int *, const double *, const int *, double *, const int *)
- void [scopy_](#) (const int *, const float *, const int *, float *, const int *)
- void [dscal_](#) (const int *, const double *, double *, const int *)
- void [sscal_](#) (const int *, const float *, float *, const int *)
- void [dtrsm_](#) (const char *, const char *, const char *, const char *, const int *, const int *, const double *, const double *, const int *, double *, const int *)
- void [strsm_](#) (const char *, const char *, const char *, const char *, const int *, const int *, const float *, const float *, const int *, float *, const int *)
- void [dtrmm_](#) (const char *, const char *, const char *, const char *, const int *, const int *, const double *, const double *, const int *, double *, const int *)
- void [strmm_](#) (const char *, const char *, const char *, const char *, const int *, const int *, const float *, const float *, const int *, float *, const int *)
- void [sgemm_](#) (const char *, const char *, const int *, const int *, const int *, const float *, const float *, const int *, const float *, const int *, const float *, float *, const int *)
- void [dgemm_](#) (const char *, const char *, const int *, const int *, const int *, const double *, const double *, const int *, const double *, const int *, const double *, double *, const int *)

13.55.1 Macro Definition Documentation

13.55.1.1 CBLAS_INT

```
#define CBLAS_INT int
```

13.55.1.2 CBLAS_ENUM_DEFINED_H

```
#define CBLAS_ENUM_DEFINED_H
```

13.55.1.3 CBLAS_EXTERNALS

```
#define CBLAS_EXTERNALS
```

13.55.1.4 blas_enum

```
#define blas_enum enum
```

13.55.2 Enumeration Type Documentation

13.55.2.1 CBLAS_ORDER

```
enum CBLAS_ORDER
```

Enumerator

CblasRowMajor	
CblasColMajor	

13.55.2.2 CBLAS_TRANSPOSE

```
enum CBLAS_TRANSPOSE
```

Enumerator

CblasNoTrans	
CblasTrans	
CblasConjTrans	
AtlasConj	

13.55.2.3 CBLAS_UPLO

```
enum CBLAS_UPLO
```

Enumerator

CblasUpper	
CblasLower	

13.55.2.4 CBLAS_DIAG

```
enum CBLAS_DIAG
```

Enumerator

CblasNonUnit	
--------------	--

Enumerator

CblasUnit	
-----------	--

13.55.2.5 CBLAS_SIDE

enum [CBLAS_SIDE](#)

Enumerator

CblasLeft	
CblasRight	

13.55.3 Function Documentation

13.55.3.1 daxpy_()

```
void daxpy_ (
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

13.55.3.2 saxpy_()

```
void saxpy_ (
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

13.55.3.3 ddot_()

```
double ddot_ (
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    const int * )
```

13.55.3.4 sdot_()

```
float sdot_ (
    const int * ,
    const float * ,
    const int * ,
    const float * ,
    const int * )
```

13.55.3.5 dasum_()

```
double dasum_ (
    const int * ,
```

```
const double * ,  
const int * )
```

13.55.3.6 idamax_()

```
int idamax_ (  
    const int * ,  
    const double * ,  
    const int * )
```

13.55.3.7 dnrn2_()

```
double dnrn2_ (  
    const int * ,  
    const double * ,  
    const int * )
```

13.55.3.8 dgemv_()

```
void dgemv_ (  
    const char * ,  
    const int * ,  
    const int * ,  
    const double * ,  
    const double * ,  
    const int * ,  
    const double * ,  
    const int * ,  
    const double * ,  
    double * ,  
    const int * )
```

13.55.3.9 sgemv_()

```
void sgemv_ (  
    const char * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    const float * ,  
    const int * ,  
    const float * ,  
    float * ,  
    const int * )
```

13.55.3.10 dger_()

```
void dger_ (  
    const int * ,  
    const int * ,  
    const double * ,  
    const double * ,  
    const int * ,  
    const double * ,  
    const int * ,  
    double * ,  
    const int * )
```

13.55.3.11 sger_()

```
void sger_ (
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

13.55.3.12 dcopy_()

```
void dcopy_ (
    const int * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

13.55.3.13 scopy_()

```
void scopy_ (
    const int * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

13.55.3.14 dscal_()

```
void dscal_ (
    const int * ,
    const double * ,
    double * ,
    const int * )
```

13.55.3.15 sscal_()

```
void sscal_ (
    const int * ,
    const float * ,
    float * ,
    const int * )
```

13.55.3.16 dtrsm_()

```
void dtrsm_ (
    const char * ,
    const char * ,
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
```



```
double * ,  
const int * )
```

13.55.3.17 strsm_()

```
void strsm_ (  
    const char * ,  
    const char * ,  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    float * ,  
    const int * )
```

13.55.3.18 dtrmm_()

```
void dtrmm_ (  
    const char * ,  
    const char * ,  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const double * ,  
    const double * ,  
    const int * ,  
    double * ,  
    const int * )
```

13.55.3.19 strmm_()

```
void strmm_ (  
    const char * ,  
    const char * ,  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    float * ,  
    const int * )
```

13.55.3.20 sgemm_()

```
void sgemm_ (  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,
```

```

const float * ,
const int * ,
const float * ,
float * ,
const int * )

```

13.55.3.21 dgemm_()

```

void dgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    double * ,
    const int * )

```

13.56 config.h File Reference

Macros

- #define [HAVE_BLAS](#) 1
- #define [HAVE_CBLAS](#) 1
- #define [HAVE_CXX11](#) 1
- #define [HAVE_DLFCN_H](#) 1
- #define [HAVE_FLOAT_H](#) 1
- #define [HAVE_INTTYPES_H](#) 1
- #define [HAVE_LAPACK](#) 1
- #define [HAVE_LIMITS_H](#) 1
- #define [HAVE_LITTLE_ENDIAN](#) 1
- #define [HAVE_PTHREAD_H](#) 1
- #define [HAVE_STDDEF_H](#) 1
- #define [HAVE_STDINT_H](#) 1
- #define [HAVE_STDIO_H](#) 1
- #define [HAVE_STDLIB_H](#) 1
- #define [HAVE_STRINGS_H](#) 1
- #define [HAVE_STRING_H](#) 1
- #define [HAVE_SYS_STAT_H](#) 1
- #define [HAVE_SYS_TIME_H](#) 1
- #define [HAVE_SYS_TYPES_H](#) 1
- #define [HAVE_UNISTD_H](#) 1
- #define [LT_OBJDIR](#) ".libs/"
- #define [OPENBLAS_NUM_THREADS](#) 1
- #define [PACKAGE](#) "fflas-ffpack"
- #define [PACKAGE_BUGREPORT](#) "ffpack-devel@googlegroups.com"
- #define [PACKAGE_NAME](#) "FFLAS-FFPACK"
- #define [PACKAGE_STRING](#) "FFLAS-FFPACK 2.4.3"
- #define [PACKAGE_TARNAME](#) "fflas-ffpack"
- #define [PACKAGE_URL](#) "https://github.com/linbox-team/fflas-ffpack"
- #define [PACKAGE_VERSION](#) "2.4.3"

- #define [SIZEOF_CHAR](#) 1
- #define [SIZEOF_INT](#) 4
- #define [SIZEOF_LONG](#) 4
- #define [SIZEOF_LONG_LONG](#) 8
- #define [SIZEOF_SHORT](#) 2
- #define [SIZEOF___INT64](#) 0
- #define [STDC_HEADERS](#) 1
- #define [USE_OPENMP](#) 1
- #define [VERSION](#) "2.4.3"

13.56.1 Macro Definition Documentation

13.56.1.1 HAVE_BLAS

```
#define HAVE_BLAS 1
```

13.56.1.2 HAVE_CBLAS

```
#define HAVE_CBLAS 1
```

13.56.1.3 HAVE_CXX11

```
#define HAVE_CXX11 1
```

13.56.1.4 HAVE_DLFCN_H

```
#define HAVE_DLFCN_H 1
```

13.56.1.5 HAVE_FLOAT_H

```
#define HAVE_FLOAT_H 1
```

13.56.1.6 HAVE_INTPYPES_H

```
#define HAVE_INTPYPES_H 1
```

13.56.1.7 HAVE_LAPACK

```
#define HAVE_LAPACK 1
```

13.56.1.8 HAVE_LIMITS_H

```
#define HAVE_LIMITS_H 1
```

13.56.1.9 HAVE_LITTLE_ENDIAN

```
#define HAVE_LITTLE_ENDIAN 1
```

13.56.1.10 HAVE_PTHREAD_H

```
#define HAVE_PTHREAD_H 1
```

13.56.1.11 HAVE_STDDEF_H

```
#define HAVE_STDDEF_H 1
```

13.56.1.12 HAVE_STDINT_H

```
#define HAVE_STDINT_H 1
```

13.56.1.13 HAVE_STDIO_H

```
#define HAVE_STDIO_H 1
```

13.56.1.14 HAVE_STDLIB_H

```
#define HAVE_STDLIB_H 1
```

13.56.1.15 HAVE_STRINGS_H

```
#define HAVE_STRINGS_H 1
```

13.56.1.16 HAVE_STRING_H

```
#define HAVE_STRING_H 1
```

13.56.1.17 HAVE_SYS_STAT_H

```
#define HAVE_SYS_STAT_H 1
```

13.56.1.18 HAVE_SYS_TIME_H

```
#define HAVE_SYS_TIME_H 1
```

13.56.1.19 HAVE_SYS_TYPES_H

```
#define HAVE_SYS_TYPES_H 1
```

13.56.1.20 HAVE_UNISTD_H

```
#define HAVE_UNISTD_H 1
```

13.56.1.21 LT_OBJDIR

```
#define LT_OBJDIR ".libs/"
```

13.56.1.22 OPENBLAS_NUM_THREADS

```
#define OPENBLAS_NUM_THREADS 1
```

13.56.1.23 PACKAGE

```
#define PACKAGE "fflas-ffpack"
```

13.56.1.24 PACKAGE_BUGREPORT

```
#define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

13.56.1.25 PACKAGE_NAME

```
#define PACKAGE_NAME "FFLAS-FFPACK"
```

13.56.1.26 PACKAGE_STRING

```
#define PACKAGE_STRING "FFLAS-FFPACK 2.4.3"
```

13.56.1.27 PACKAGE_TARNAME

```
#define PACKAGE_TARNAME "fflas-ffpack"
```

13.56.1.28 PACKAGE_URL

```
#define PACKAGE_URL "https://github.com/linbox-team/fblas-ffpack"
```

13.56.1.29 PACKAGE_VERSION

```
#define PACKAGE_VERSION "2.4.3"
```

13.56.1.30 SIZEOF_CHAR

```
#define SIZEOF_CHAR 1
```

13.56.1.31 SIZEOF_INT

```
#define SIZEOF_INT 4
```

13.56.1.32 SIZEOF_LONG

```
#define SIZEOF_LONG 4
```

13.56.1.33 SIZEOF_LONG_LONG

```
#define SIZEOF_LONG_LONG 8
```

13.56.1.34 SIZEOF_SHORT

```
#define SIZEOF_SHORT 2
```

13.56.1.35 SIZEOF___INT64

```
#define SIZEOF___INT64 0
```

13.56.1.36 STDC_HEADERS

```
#define STDC_HEADERS 1
```

13.56.1.37 USE_OPENMP

```
#define USE_OPENMP 1
```

13.56.1.38 VERSION

```
#define VERSION "2.4.3"
```

13.57 config.h File Reference

Macros

- [#define __FBLASFFPACK_HAVE_BLAS 1](#)
- [#define __FBLASFFPACK_HAVE_CBLAS 1](#)
- [#define __FBLASFFPACK_HAVE_CXX11 1](#)
- [#define __FBLASFFPACK_HAVE_DLFCN_H 1](#)
- [#define __FBLASFFPACK_HAVE_FLOAT_H 1](#)
- [#define __FBLASFFPACK_HAVE_INTTYPES_H 1](#)
- [#define __FBLASFFPACK_HAVE_LAPACK 1](#)
- [#define __FBLASFFPACK_HAVE_LIMITS_H 1](#)
- [#define __FBLASFFPACK_HAVE_LITTLE_ENDIAN 1](#)
- [#define __FBLASFFPACK_HAVE_PTHREAD_H 1](#)
- [#define __FBLASFFPACK_HAVE_STDDEF_H 1](#)

- `#define __FFLASFFPACK_HAVE_STDINT_H 1`
- `#define __FFLASFFPACK_HAVE_STDIO_H 1`
- `#define __FFLASFFPACK_HAVE_STDLIB_H 1`
- `#define __FFLASFFPACK_HAVE_STRINGS_H 1`
- `#define __FFLASFFPACK_HAVE_STRING_H 1`
- `#define __FFLASFFPACK_HAVE_SYS_STAT_H 1`
- `#define __FFLASFFPACK_HAVE_SYS_TIME_H 1`
- `#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1`
- `#define __FFLASFFPACK_HAVE_UNISTD_H 1`
- `#define __FFLASFFPACK_LT_OBJDIR ".libs/"`
- `#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1`
- `#define __FFLASFFPACK_PACKAGE "fflas-ffpack"`
- `#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"`
- `#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"`
- `#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.4.3"`
- `#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"`
- `#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"`
- `#define __FFLASFFPACK_PACKAGE_VERSION "2.4.3"`
- `#define __FFLASFFPACK_SIZEOF_CHAR 1`
- `#define __FFLASFFPACK_SIZEOF_INT 4`
- `#define __FFLASFFPACK_SIZEOF_LONG 4`
- `#define __FFLASFFPACK_SIZEOF_LONG_LONG 8`
- `#define __FFLASFFPACK_SIZEOF_SHORT 2`
- `#define __FFLASFFPACK_SIZEOF__INT64 0`
- `#define __FFLASFFPACK_STDC_HEADERS 1`
- `#define __FFLASFFPACK_USE_OPENMP 1`
- `#define __FFLASFFPACK_VERSION "2.4.3"`

13.57.1 Macro Definition Documentation

13.57.1.1 `__FFLASFFPACK_HAVE_BLAS`

```
#define __FFLASFFPACK_HAVE_BLAS 1
```

13.57.1.2 `__FFLASFFPACK_HAVE_CBLAS`

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

13.57.1.3 `__FFLASFFPACK_HAVE_CXX11`

```
#define __FFLASFFPACK_HAVE_CXX11 1
```

13.57.1.4 `__FFLASFFPACK_HAVE_DLFCN_H`

```
#define __FFLASFFPACK_HAVE_DLFCN_H 1
```

13.57.1.5 `__FFLASFFPACK_HAVE_FLOAT_H`

```
#define __FFLASFFPACK_HAVE_FLOAT_H 1
```

13.57.1.6 `__FFLASFFPACK_HAVE_INTTYPES_H`

```
#define __FFLASFFPACK_HAVE_INTTYPES_H 1
```

13.57.1.7 `__FFLASFFPACK_HAVE_LAPACK`

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

13.57.1.8 __FFLASFFPACK_HAVE_LIMITS_H

```
#define __FFLASFFPACK_HAVE_LIMITS_H 1
```

13.57.1.9 __FFLASFFPACK_HAVE_LITTLE_ENDIAN

```
#define __FFLASFFPACK_HAVE_LITTLE_ENDIAN 1
```

13.57.1.10 __FFLASFFPACK_HAVE_PTHREAD_H

```
#define __FFLASFFPACK_HAVE_PTHREAD_H 1
```

13.57.1.11 __FFLASFFPACK_HAVE_STDDEF_H

```
#define __FFLASFFPACK_HAVE_STDDEF_H 1
```

13.57.1.12 __FFLASFFPACK_HAVE_STDINT_H

```
#define __FFLASFFPACK_HAVE_STDINT_H 1
```

13.57.1.13 __FFLASFFPACK_HAVE_STDIO_H

```
#define __FFLASFFPACK_HAVE_STDIO_H 1
```

13.57.1.14 __FFLASFFPACK_HAVE_STDLIB_H

```
#define __FFLASFFPACK_HAVE_STDLIB_H 1
```

13.57.1.15 __FFLASFFPACK_HAVE_STRINGS_H

```
#define __FFLASFFPACK_HAVE_STRINGS_H 1
```

13.57.1.16 __FFLASFFPACK_HAVE_STRING_H

```
#define __FFLASFFPACK_HAVE_STRING_H 1
```

13.57.1.17 __FFLASFFPACK_HAVE_SYS_STAT_H

```
#define __FFLASFFPACK_HAVE_SYS_STAT_H 1
```

13.57.1.18 __FFLASFFPACK_HAVE_SYS_TIME_H

```
#define __FFLASFFPACK_HAVE_SYS_TIME_H 1
```

13.57.1.19 __FFLASFFPACK_HAVE_SYS_TYPES_H

```
#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1
```

13.57.1.20 __FFLASFFPACK_HAVE_UNISTD_H

```
#define __FFLASFFPACK_HAVE_UNISTD_H 1
```

13.57.1.21 __FFLASFFPACK_LT_OBJDIR

```
#define __FFLASFFPACK_LT_OBJDIR ".libs/"
```

13.57.1.22 __FFLASFFPACK_OPENBLAS_NUM_THREADS

```
#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1
```

13.57.1.23 __FFLASFFPACK_PACKAGE

```
#define __FFLASFFPACK_PACKAGE "fflas-ffpack"
```

13.57.1.24 __FFLASFFPACK_PACKAGE_BUGREPORT

```
#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

13.57.1.25 __FFLASFFPACK_PACKAGE_NAME

```
#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"
```

13.57.1.26 __FFLASFFPACK_PACKAGE_STRING

```
#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.4.3"
```

13.57.1.27 __FFLASFFPACK_PACKAGE_TARNAME

```
#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
```

13.57.1.28 __FFLASFFPACK_PACKAGE_URL

```
#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

13.57.1.29 __FFLASFFPACK_PACKAGE_VERSION

```
#define __FFLASFFPACK_PACKAGE_VERSION "2.4.3"
```

13.57.1.30 __FFLASFFPACK_SIZEOF_CHAR

```
#define __FFLASFFPACK_SIZEOF_CHAR 1
```

13.57.1.31 __FFLASFFPACK_SIZEOF_INT

```
#define __FFLASFFPACK_SIZEOF_INT 4
```

13.57.1.32 __FFLASFFPACK_SIZEOF_LONG

```
#define __FFLASFFPACK_SIZEOF_LONG 4
```

13.57.1.33 __FFLASFFPACK_SIZEOF_LONG_LONG

```
#define __FFLASFFPACK_SIZEOF_LONG_LONG 8
```

13.57.1.34 __FFLASFFPACK_SIZEOF_SHORT

```
#define __FFLASFFPACK_SIZEOF_SHORT 2
```

13.57.1.35 __FFLASFFPACK_SIZEOF__INT64

```
#define __FFLASFFPACK_SIZEOF__INT64 0
```

13.57.1.36 __FFLASFFPACK_STDC_HEADERS

```
#define __FFLASFFPACK_STDC_HEADERS 1
```

13.57.1.37 __FFLASFFPACK_USE_OPENMP

```
#define __FFLASFFPACK_USE_OPENMP 1
```


13.57.1.38 __FFLASFFPACK_VERSION

```
#define __FFLASFFPACK_VERSION "2.4.3"
```

13.58 fflas-ffpack-config.h File Reference

Defaults for optimised values.

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/fflas-ffpack-thresholds.h"
#include "fflas-ffpack/fflas-ffpack-default-thresholds.h"
#include "givaro/givconfig.h"
```

Macros

- #define [GCC_VERSION](#) (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)

13.58.1 Detailed Description

Defaults for optimised values.

While `fflas-ffpack-optimize.h` is created by `configure` script, (either left blank or filled by optimiser), this file produces the defaults for the optimised values. If `fflas-ffpack-optimize.h` is not empty, then its values preceeds the defaults here.

13.58.2 Macro Definition Documentation**13.58.2.1 GCC_VERSION**

```
#define GCC_VERSION (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)
```

13.59 fflas-ffpack-default-thresholds.h File Reference**Macros**

- #define [__FFLASFFPACK_WINOTHRESHOLD](#) 1000
- #define [__FFLASFFPACK_WINOTHRESHOLD_FLT](#) 2000
- #define [__FFLASFFPACK_WINOTHRESHOLD_BAL](#) 1000
- #define [__FFLASFFPACK_WINOTHRESHOLD_BAL_FLT](#) 2000
- #define [__FFLASFFPACK_PLUQ_THRESHOLD](#) 256
- #define [__FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD](#) 1000
- #define [__FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD](#) 16
- #define [__FFLASFFPACK_ARITHPROG_THRESHOLD](#) 30
- #define [__FFLASFFPACK_FTRTRI_THRESHOLD](#) 32
- #define [__FFLASFFPACK_FSYTRF_THRESHOLD](#) 64
- #define [__FFLASFFPACK_FSYRK_THRESHOLD](#) 3000

13.59.1 Macro Definition Documentation**13.59.1.1 __FFLASFFPACK_WINOTHRESHOLD**

```
#define __FFLASFFPACK_WINOTHRESHOLD 1000
```

13.59.1.2 __FFLASFFPACK_WINOTHRESHOLD_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_FLT 2000
```

13.59.1.3 __FFLASFFPACK_WINOTHRESHOLD_BAL

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL 1000
```

13.59.1.4 __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT 2000
```

13.59.1.5 __FFLASFFPACK_PLUQ_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 256
```

13.59.1.6 __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD

```
#define __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD 1000
```

13.59.1.7 __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD

```
#define __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD 16
```

13.59.1.8 __FFLASFFPACK_ARITHPROG_THRESHOLD

```
#define __FFLASFFPACK_ARITHPROG_THRESHOLD 30
```

13.59.1.9 __FFLASFFPACK_FTRTRI_THRESHOLD

```
#define __FFLASFFPACK_FTRTRI_THRESHOLD 32
```

13.59.1.10 __FFLASFFPACK_FSYTRF_THRESHOLD

```
#define __FFLASFFPACK_FSYTRF_THRESHOLD 64
```

13.59.1.11 __FFLASFFPACK_FSYRK_THRESHOLD

```
#define __FFLASFFPACK_FSYRK_THRESHOLD 3000
```

13.60 fflas-ffpack-thresholds.h File Reference**13.61 fflas-ffpack.doxy File Reference****13.62 fflas-ffpack.h File Reference**

Includes [FFLAS](#) and [FFPACK](#).

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas/fflas.h"
```

13.62.1 Detailed Description

Includes [FFLAS](#) and [FFPACK](#).

13.63 fflas.doxy File Reference**13.64 fflas.h File Reference**

Finite Field Linear Algebra Subroutines

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include <cmath>
#include <cstring>
```

```

#include <float.h>
#include <algorithm>
#include "fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas_level1.inl"
#include "fflas_level2.inl"
#include "fflas_level3.inl"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas_freduce.h"
#include "fflas_fadd.h"
#include "fflas_fscal.h"
#include "fflas_fassign.h"
#include "fflas_fgemm.inl"
#include "fflas_pfgemm.inl"
#include "fflas_fgemv.inl"
#include "fflas-ffpack/paladin/pfgemv.inl"
#include "fflas_freivalds.inl"
#include "fflas_fger.inl"
#include "fflas_fsyrk.inl"
#include "fflas_fsyr2k.inl"
#include "fflas_ftrsm.inl"
#include "fflas_pftrsm.inl"
#include "fflas_ftrmm.inl"
#include "fflas_ftrsv.inl"
#include "fflas_faxpy.inl"
#include "fflas_fdot.inl"
#include "fflas-ffpack/field/rns.h"
#include "fflas_fscal_mp.inl"
#include "fflas_freduce_mp.inl"
#include "fflas-ffpack/fflas/fflas_fger_mp.inl"
#include "fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas_ftrsm_mp.inl"
#include "fflas_fgemv_mp.inl"
#include "fflas-ffpack/field/rns.inl"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas_sparse.h"
#include "fflas-ffpack/checkers/checkers_fflas.inl"

```

Macros

- #define [WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD](#)
- #define [DOUBLE_TO_FLOAT_CROSSOVER](#) 800

Thresholds determining which floating point representation to use, depending on the cardinality of the finite field.

13.64.1 Detailed Description

Finite Field Linear Algebra Subroutines

Author

Clément Pernet.

13.64.2 Macro Definition Documentation

13.64.2.1 WINOTHRESHOLD

```
#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD
```

13.64.2.2 DOUBLE_TO_FLOAT_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 800
```

Thresholds determining which floating point representation to use, depending on the cardinality of the finite field. This is only used when the element representation is not a floating point type.

Bug to be benchmarked.

13.65 fflas_bounds.inl File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/flimits.h"
#include <givaro/udl.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

Macros

- `#define` [__FFLASFFPACK_fflas_bounds_INL](#)
- `#define` [FFLAS_INT_TYPE](#) `uint64_t`

Functions

- `template<class Field >`
`double computeFactorClassic (const Field &F)`
- `template<> double computeFactorClassic (const Givaro::ModularBalanced< double > &F)`
- `template<> double computeFactorClassic (const Givaro::ModularBalanced< float > &F)`
- `template<class Field >`
`size_t DotProdBoundClassic (const Field &F, const typename Field::Element &beta)`
- `Givaro::Integer InfNorm (const size_t M, const size_t N, const Givaro::Integer *A, const size_t lda)`
- `template<class Field >`
`size_t TRSMBound (const Field &)`
TRSMBound.
- `template<class Element >`
`size_t TRSMBound (const Givaro::Modular< Element > &F)`
Specialization for positive modular representation over float.
- `template<class Element >`
`size_t TRSMBound (const Givaro::ModularBalanced< Element > &F)`
Specialization for balanced modular representation over double.

13.65.1 Macro Definition Documentation

13.65.1.1 __FFLASFFPACK_fflas_bounds_INL

```
#define __FFLASFFPACK_fflas_bounds_INL
```

13.65.1.2 FFLAS_INT_TYPE

```
#define FFLAS_INT_TYPE uint64\_t
```

13.66 fflas_enum.h File Reference

```
#include <algorithm>
```

Data Structures

- class [AreEqual< X, Y >](#)
- class [AreEqual< X, X >](#)

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

Enumerations

- enum [FFLAS_ORDER](#) { [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 }
Storage by row or col ?
- enum [FFLAS_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }
Is matrix transposed ?
- enum [FFLAS_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 }
Is triangular matrix's shape upper ?
- enum [FFLAS_DIAG](#) { [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 }
Is the triangular matrix implicitly unit diagonal ?
- enum [FFLAS_SIDE](#) { [FflasLeft](#) = 141 , [FflasRight](#) = 142 }
On what side ?
- enum [FFLAS_BASE](#) { [FflasDouble](#) = 151 , [FflasFloat](#) = 152 , [FflasGeneric](#) = 153 }
FFLAS_BASE determines the type of the element representation for Matrix Mult kernel.

Functions

- template<class T >
const T & [min3](#) (const T &m, const T &n, const T &k)
- template<class T >
const T & [max3](#) (const T &m, const T &n, const T &k)
- template<class T >
const T & [min4](#) (const T &m, const T &n, const T &k, const T &l)
- template<class T >
const T & [max4](#) (const T &m, const T &n, const T &k, const T &l)

13.67 fflas_fadd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas_fadd.inl"
```

Data Structures

- struct [support_simd_add< T >](#)

Namespaces

- namespace [FFLAS](#)

Functions

- `template<class Field >`
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
fadd : matrix addition.
- `template<class Field >`
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
fsub : matrix subtraction.
- `template<class Field >`
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
faddin
- `template<class Field >`
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
fsubin $C = C - B$
- `template<class Field >`
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`
fadd : matrix addition with scaling.

13.68 fflas_fadd.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::vectorised](#)
- namespace [FFLAS::details](#)

Macros

- `#define` [__FFLASFFPACK_fadd_INL](#)

Functions

- `template<class SimdT , class Element , bool positive>`
`std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)`
- `template<bool positive, class Element , class T1 , class T2 >`
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (Element *T, const Element *TA, const Element *TB, size_t n, Element p, T1 min_, T2 max_)`
- `template<class SimdT , class Element , bool positive>`
`std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)`
- `template<bool positive, class Element , class T1 , class T2 >`
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (Element *T, const Element *TA, const Element *TB, const size_t n, const Element p, const T1 min_, const T2 max_)`
- `template<class Element >`
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (Element *T, const Element *TA, const Element *TB, size_t n)`
- `template<class Element >`
`std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (Element *T, const Element *TA, const Element *TB, size_t n)`
- `template<class Field , bool ADD>`
`std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::ModularTag)`
- `template<class Field , bool ADD>`
`std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::ModularTag)`
- `template<class Field , bool ADD>`
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::GenericTag)`
- `template<class Field , bool ADD>`
`std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::UnparametricTag)`
- `template<class Field , bool ADD>`
`std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, FieldCategories::UnparametricTag)`

13.68.1 Macro Definition Documentation

13.68.1.1 __FFLASFFPACK_fadd_INL

```
#define __FFLASFFPACK_fadd_INL
```

13.69 fflas_fassign.h File Reference

```
#include "fflas_fassign.inl"
```

13.70 fflas_fassign.inl File Reference

```
#include <string.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/debug.h"
```

Namespaces

- namespace [FFLAS](#)

Macros

- #define [__FFLASFFPACK_fassign_INL](#)

Functions

- template<class [Field](#) >
void [fassign](#) (const [Field](#) &F, const size_t N, typename [Field::ConstElement_ptr](#) Y, const size_t incY, typename [Field::Element_ptr](#) X, const size_t incX)
 $fassign : x \leftarrow y.$
- template<> void [fassign](#) (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void [fassign](#) (const [Givaro::ModularBalanced](#)< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void [fassign](#) (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)
- template<> void [fassign](#) (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<> void [fassign](#) (const [Givaro::ModularBalanced](#)< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<> void [fassign](#) (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)
- template<class [Field](#) >
void [fassign](#) (const [Field](#) &F, const size_t m, const size_t n, typename [Field::ConstElement_ptr](#) B, const size_t ldb, typename [Field::Element_ptr](#) A, const size_t lda)
 $fassign : A \leftarrow B.$

13.70.1 Macro Definition Documentation

13.70.1.1 __FFLASFFPACK_fassign_INL

```
#define __FFLASFFPACK_fassign_INL
```


13.71 fflas_faxpy.inl File Reference

Namespaces

- namespace [FFLAS](#)

Macros

- `#define __FFLASFFPACK_faxpy_INL`

Functions

- `template<class Field >`
`void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)`

$$faxpy : y \leftarrow \alpha \cdot x + y.$$

13.71.1 Macro Definition Documentation

13.71.1.1 __FFLASFFPACK_faxpy_INL

```
#define __FFLASFFPACK_faxpy_INL
```

13.72 fflas_fdot.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

Namespaces

- namespace [FFLAS](#)

Macros

- `#define __FFLASFFPACK_fdot_INL`

Functions

- `template<class Field >`
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field >`
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DelayedTag &MT)`
- `template<> Givaro::DoubleDomain::Element fdot (const Givaro::DoubleDomain &, const size_t N, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`

- `template<> Givaro::FloatDomain::Element fdot (const Givaro::FloatDomain &, const size_t N, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field, class T>
Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::ConvertTo< T > &MT)`
- `template<class Field>
Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultBoundedTag &dbt)`
- `template<class Field>
Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, const ParSeqHelper::Sequential seq)`
- `template<class Field>
Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`

fdot: dot product $x^T y$.

13.72.1 Macro Definition Documentation

13.72.1.1 __FFLASFFPACK_fdot_INL

```
#define __FFLASFFPACK_fdot_INL
```

13.73 fflas_fgemv.inl File Reference

```
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utis/debug.h"
#include "fflas_fgemv/fgemv_classical.inl"
#include "fflas_fgemv/fgemv_winograd.inl"
```

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

Macros

- `#define __FFLASFFPACK_fgemv_INL`

Functions

- `template<class NewField, class Field, class FieldMode>
Field::Element_ptr fgemv_convert (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`
- `template<class Field, class Element, class AlgoT, class ParSeqTrait>
bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
bool NeedPreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`

- `template<class Field, class Element, class AlgoT, class ParSeqTrait >`
`bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max,`
`Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait >`
`&WH)`
- `template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait >`
`bool NeedPreSubReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max,`
`Element &Op2min, Element &Op2max, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class Field, class Element, class AlgoT, class ParSeqTrait >`
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Ele-`
`ment &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT,`
`ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait >`
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element`
`&Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeT, Par-`
`SeqTrait > &WH)`
- `template<class Field, class AlgoT, class ParSeqTrait >`
`void ScalAndReduce (const Field &F, const size_t N, const typename Field::Element alpha, typename`
`Field::Element_ptr X, const size_t incX, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, Par-`
`SeqTrait > &H)`
- `template<class Field, class AlgoT, class ParSeqTrait >`
`void ScalAndReduce (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha,`
`typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag,`
`ParSeqTrait > &H)`
- `template<class Field >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper<`
`Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > ,`
`ParSeqHelper::Sequential > &H)`
- `template<typename Field >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, type-`
`name Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size-`
`_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const`
`ParSeqHelper::Sequential seq)`
- `template<typename Field, class Cut, class Param >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, type-`
`name Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size-`
`_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const`
`ParSeqHelper::Parallel< Cut, Param > par)`
- `template<typename Field >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const`
`typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`

fgemm: Field GENeral Matrix Multiply.
- `template<typename Field, class ModeT, class ParSeq >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`
`MMHelperAlgo::Auto, ModeT, ParSeq > &H)`
- `template<class Field >`
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename`

`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)`

- `template<class Field >`
`Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`
fsquare: Squares a matrix.
- `template<class Field >`
`Field::Element_ptr fsquareCommon (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`
- `template<> double * fsquare (const Givaro::ModularBalanced< double > &F, const FFLAS_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)`
- `template<> float * fsquare (const Givaro::ModularBalanced< float > &F, const FFLAS_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)`
- `template<> double * fsquare (const Givaro::Modular< double > &F, const FFLAS_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)`
- `template<> float * fsquare (const Givaro::Modular< float > &F, const FFLAS_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)`

13.73.1 Macro Definition Documentation

13.73.1.1 __FFLASFFPACK_fgemv_INL

```
#define __FFLASFFPACK_fgemv_INL
```

13.74 fgemv_classical.inl File Reference

13.75 fgemv_classical_mp.inl File Reference

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

Data Structures

- struct `MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >`
- struct `MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`
- struct `MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >`

Namespaces

- namespace `FFLAS`

Macros

- `#define __FFPACK_fgemv_classical_INL`

Functions

- `template<typename RNS , typename ParSeqTrait >`
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)`
- `template<typename RNS >`
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)`
- `template<typename RNS , typename ParSeqTrait >`
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)`
- `template<typename RNS , typename Cut , typename Param >`
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)`
- `template<class ParSeq >`
`Givaro::Integer * fgemm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<typename RNS , class ModeT >`
`RNS::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)`
- `template<typename RNS >`
`RNS::Element_ptr fgemm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)`
- `Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer *`

```

::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >,
MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
• template<class ParSeq >
Givaro::Integer * fgemmm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta,
const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer al-
pha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::
Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >,
MMHelperAlgo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
• template<size_t K1, size_t K2, class ParSeq >
Reclnt::ruint< K1 > * fgemmm (const Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >
&F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n,
const size_t k, const Reclnt::ruint< K1 > alpha, const Reclnt::ruint< K1 > *A, const size_t lda, const
Reclnt::ruint< K1 > *B, const size_t ldb, Reclnt::ruint< K1 > beta, Reclnt::ruint< K1 > *C, const size_t
ldc, MMHelper< Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)

```

13.75.1 Detailed Description

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

13.75.2 Macro Definition Documentation

13.75.2.1 __FFPACK_fgemmm_classical_INL

```
#define __FFPACK_fgemmm_classical_INL
```

13.76 fgemmm_winograd.inl File Reference

```

#include <stdint.h>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fgemmm_classical.inl"
#include "schedule_winograd.inl"
#include "schedule_winograd_acc.inl"
#include "schedule_winograd_acc_ip.inl"
#include "schedule_winograd_ip.inl"
#include "fflas-ffpack/fflas-ffpack-config.h"

```

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

Macros

- #define [__FFLASFFPACK_fflas_fflas_fgemmm_winograd_INL](#)
- #define [NEWWINO](#)

Functions

- template<class [Field](#) >
int [WinogradThreshold](#) (const [Field](#) &F)
Computes the number of recursive levels to perform.
- template<> int [WinogradThreshold](#) (const Givaro::Modular< float > &F)
- template<> int [WinogradThreshold](#) (const Givaro::ModularBalanced< double > &F)
- template<> int [WinogradThreshold](#) (const Givaro::ModularBalanced< float > &F)

- template<class [Field](#) >
int [WinogradSteps](#) (const [Field](#) &F, const size_t &m)
Computes the number of recursive levels to perform.
- template<class [Field](#) , class [FieldMode](#) >
void [DynamicPeeling](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) > &H, const typename [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) >::DelayedField::Element Cmin, const typename [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) >::DelayedField::Element Cmax)
- template<class [Field](#) , class [FieldMode](#) >
void [DynamicPeeling2](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) > &H, const typename [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) >::DelayedField::Element Cmin, const typename [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) >::DelayedField::Element Cmax)
- template<class [Field](#) , class [FieldMode](#) >
void [WinogradCalc](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t mr, const size_t nr, const size_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) > &H)
- template<class [Field](#) , class [ModeT](#) >
[Field::Element_ptr](#) [fgemm](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [ModeT](#) > &H)
- template<class [Field](#) , class [ModeT](#) , class [Cut](#) , class [Param](#) >
[Field::Element_ptr](#) [fgemm](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [FFLAS_TRANSPOSE](#) tb, const size_t m, const size_t n, const size_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement_ptr](#) A, const size_t lda, typename [Field::ConstElement_ptr](#) B, const size_t ldb, const typename [Field::Element](#) beta, typename [Field::Element_ptr](#) C, const size_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::WinogradPar](#), [ModeT](#), [ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > > &H)

13.76.1 Macro Definition Documentation

13.76.1.1 `__FFLASFFPACK_fflas_fflas_fgemm_winograd_INL`

```
#define __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
```

13.76.1.2 `NEWWINO`

```
#define NEWWINO
```

13.77 matmul.doxy File Reference

13.78 `schedule_bini.inl` File Reference

Bini implementation.

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::BLAS3](#)

Macros

- `#define __FFLASFFPACK_fgemm_bini_INL`

Functions

- `template<class Field >`
`void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr,`
`const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A,`
`const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta,`
`typename Field::Element_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE`
`base, const size_t rec_level)`

13.78.1 Detailed Description

Bini implementation.

13.78.2 Macro Definition Documentation

13.78.2.1 __FFLASFFPACK_fgemm_bini_INL

```
#define __FFLASFFPACK_fgemm_bini_INL
```

13.79 schedule_winograd.inl File Reference

Namespaces

- namespace `FFLAS`
- namespace `FFLAS::BLAS3`

Macros

- `#define __FFLASFFPACK_fgemm_winograd_INL`

Functions

- `template<class Field , class FieldTrait , class Strat , class Param >`
`Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`
`tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename`
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`
`MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)`
- `template<class Field , class FieldTrait >`
`void Winograd (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t`
`mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr`
`A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element`
`beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, Field↵`
`Trait > &WH)`

13.79.1 Macro Definition Documentation

13.79.1.1 __FFLASFFPACK_fgemm_winograd_INL

```
#define __FFLASFFPACK_fgemm_winograd_INL
```

13.80 schedule_winograd_acc.inl File Reference

Namespaces

- namespace `FFLAS`

- namespace [FFLAS::BLAS3](#)

Macros

- `#define __FFLASFFPACK_fgemm_winograd_acc_INL`

Functions

- `template<class Field , class FieldTrait >`
`void WinogradAcc_3_23 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`
`void WinogradAcc_3_21 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`
`void WinogradAcc_2_24 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`
`void WinogradAcc_2_27 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

13.80.1 Macro Definition Documentation

13.80.1.1 __FFLASFFPACK_fgemm_winograd_acc_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_INL
```

13.81 schedule_winograd_acc_ip.inl File Reference

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::BLAS3](#)

Macros

- `#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL`

Functions

- `template<class Field , class FieldTrait >`
`void WinogradAcc_LR (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

- `template<class Field , class FieldTrait >`
`void WinogradAcc_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`
`void WinogradAcc_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

13.81.1 Macro Definition Documentation

13.81.1.1 __FFLASFFPACK_fgemm_winograd_acc_ip_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL
```

13.82 schedule_winograd_ip.inl File Reference

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::BLAS3](#)

Macros

- `#define __FFLASFFPACK_fgemm_winograd_ip_INL`

Functions

- `template<class Field , class FieldTrait >`
`void Winograd_LR_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`
`void Winograd_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`
`void Winograd_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

13.82.1 Macro Definition Documentation

13.82.1.1 __FFLASFFPACK_fgemm_winograd_ip_INL

```
#define __FFLASFFPACK_fgemm_winograd_ip_INL
```

13.83 fflas_fgemv.inl File Reference

```
#include <givaro/zring.h>
```

Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

Macros

- #define [__FFLASFFPACK_fgemv_INL](#)

Functions

- [template<typename FloatElement , class Field >](#)
[Field::Element_ptr fgemv_convert](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [size_t](#) M, const [size_t](#) N, const [typename](#) [Field::Element](#) alpha, [typename](#) [Field::ConstElement_ptr](#) A, const [size_t](#) lda, [typename](#) [Field::ConstElement_ptr](#) X, const [size_t](#) incX, const [typename](#) [Field::Element](#) beta, [typename](#) [Field::Element_ptr](#) Y, const [size_t](#) incY)
- [template<class Field >](#)
[Field::Element_ptr fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [size_t](#) M, const [size_t](#) N, const [typename](#) [Field::Element](#) alpha, [typename](#) [Field::ConstElement_ptr](#) A, const [size_t](#) lda, [typename](#) [Field::ConstElement_ptr](#) X, const [size_t](#) incX, const [typename](#) [Field::Element](#) beta, [typename](#) [Field::Element_ptr](#) Y, const [size_t](#) incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo](#)< [ElementCategories::MachineFloatTag](#) > > &H)
- [template<class Field >](#)
[Field::Element_ptr fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [size_t](#) M, const [size_t](#) N, const [typename](#) [Field::Element](#) alpha, [typename](#) [Field::ConstElement_ptr](#) A, const [size_t](#) lda, [typename](#) [Field::ConstElement_ptr](#) X, const [size_t](#) incX, const [typename](#) [Field::Element](#) beta, [typename](#) [Field::Element_ptr](#) Y, const [size_t](#) incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DelayedTag](#) > &H)
- [template<class Field >](#)
[Field::Element_ptr fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [size_t](#) M, const [size_t](#) N, const [typename](#) [Field::Element](#) alpha, [typename](#) [Field::ConstElement_ptr](#) A, const [size_t](#) lda, [typename](#) [Field::ConstElement_ptr](#) X, const [size_t](#) incX, const [typename](#) [Field::Element](#) beta, [typename](#) [Field::Element_ptr](#) Y, const [size_t](#) incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [template<class Field >](#)
[Field::Element_ptr fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [size_t](#) M, const [size_t](#) N, const [typename](#) [Field::Element](#) alpha, [typename](#) [Field::ConstElement_ptr](#) A, const [size_t](#) lda, [typename](#) [Field::ConstElement_ptr](#) X, const [size_t](#) incX, const [typename](#) [Field::Element](#) beta, [typename](#) [Field::Element_ptr](#) Y, const [size_t](#) incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::LazyTag](#) > &H)
- [template<class Field >](#)
[Field::Element_ptr fgemv](#) (const [Field](#) &F, const [FFLAS_TRANSPOSE](#) TransA, const [size_t](#) M, const [size_t](#) N, const [typename](#) [Field::Element](#) alpha, [typename](#) [Field::ConstElement_ptr](#) A, const [size_t](#) lda, [typename](#) [Field::ConstElement_ptr](#) X, const [size_t](#) incX, const [typename](#) [Field::Element](#) beta, [typename](#) [Field::Element_ptr](#) Y, const [size_t](#) incY)
finite prime Field GEneral Matrix Vector multiplication.
- [Givaro::ZRing< int64_t >::Element_ptr fgemv](#) (const [Givaro::ZRing< int64_t >](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [size_t](#) M, const [size_t](#) N, const [int64_t](#) alpha, const [int64_t](#) *A, const [size_t](#) lda, const [int64_t](#) *X, const [size_t](#) incX, const [int64_t](#) beta, [int64_t](#) *Y, const [size_t](#) incY, [MMHelper](#)< [Givaro::ZRing< int64_t >](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [Givaro::DoubleDomain::Element_ptr fgemv](#) (const [Givaro::DoubleDomain](#) &F, const [FFLAS_TRANSPOSE](#) ta, const [size_t](#) M, const [size_t](#) N, const [Givaro::DoubleDomain::Element](#) alpha, const [Givaro::DoubleDomain::ConstElement_ptr](#) A, const [size_t](#) lda, const [Givaro::DoubleDomain::ConstElement_ptr](#) X, const [size_t](#) incX, const [Givaro::DoubleDomain::Element](#) beta, [Givaro::DoubleDomain::Element_ptr](#) Y, const [size_t](#) incY)

- ```
t incX, const Givaro::DoubleDomain::Element beta, Givaro::DoubleDomain::Element_ptr Y, const size_t incY,
MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
```
- `template<class Field >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)`
  - `Givaro::FloatDomain::Element_ptr fgemv (const Givaro::FloatDomain &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr A, const size_t lda, const Givaro::FloatDomain::ConstElement_ptr X, const size_t incX, const Givaro::FloatDomain::Element beta, Givaro::FloatDomain::Element_ptr Y, const size_t incY, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
  - `template<class Field, class Cut, class Param >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, ParSeqHelper::Parallel< Cut, Param > &parH)`
  - `template<class Field >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY, ParSeqHelper::Sequential &seqH)`

### 13.83.1 Macro Definition Documentation

#### 13.83.1.1 \_\_FFLASFFPACK\_fgemv\_INL

```
#define __FFLASFFPACK_fgemv_INL
```

## 13.84 fflas\_fgemv\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- namespace `FFLAS`

### Macros

- `#define __FFLASFFPACK_fgemv_mp_INL`

### Functions

- `FFPACK::rns_double::Element_ptr fgemv (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::ConstElement_ptr X, const size_t incX, const FFPACK::rns_double::Element beta, FFPACK::rns_double::Element_ptr Y, const size_t incY, MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `FFPACK::rns_double::Element_ptr fgemv (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::ConstElement_ptr X, const size_t incX, const FFPACK::rns_double::Element beta, FFPACK::rns_double::Element_ptr Y, const size_t incY, MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`

- `Givaro::Integer * fgemv (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const Givaro::Integer alpha, Givaro::Integer *A, const size_t lda, Givaro::Integer *X, const size_t ldx, Givaro::Integer beta, Givaro::Integer *Y, const size_t ldy, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)`
- `Givaro::Integer * fgemv (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const Givaro::Integer alpha, Givaro::Integer *A, const size_t lda, Givaro::Integer *X, const size_t ldx, Givaro::Integer beta, Givaro::Integer *Y, const size_t ldy, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)`
- `template<size_t K1, size_t K2, class ParSeq >  
Reclnt::ruint< K1 > * fgemv (const Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > > &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t n, const Reclnt::ruint< K1 > alpha, const Reclnt::ruint< K1 > *A, const size_t lda, const Reclnt::ruint< K1 > *X, const size_t incx, Reclnt::ruint< K1 > beta, Reclnt::ruint< K1 > *Y, const size_t incy, MMHelper< Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`

## 13.84.1 Macro Definition Documentation

### 13.84.1.1 \_\_FFLASFFPACK\_fgemv\_mp\_INL

```
#define __FFLASFFPACK_fgemv_mp_INL
```

## 13.85 fflas\_fger.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::Protected`

### Macros

- `#define __FFLASFFPACK_fger_INL`

### Functions

- `template<class Field >  
void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`  
*fger: rank one update of a general matrix*
- `template<class FloatElement, class Field >  
void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >  
void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)`
- `template<class Field, class AnyTag >  
void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, AnyTag > &H)`
- `void fger (const Givaro::DoubleDomain &F, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, const Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, Givaro::DoubleDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`

- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, const`  
`typename Field::ConstElement_ptr x, const size_t incx, const typename Field::ConstElement_ptr y, const`  
`size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic,`  
`ModeCategories::DefaultBoundedTag > &H)`
- `void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::FloatDomain::Element`  
`alpha, const Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, const Givaro::FloatDomain::↔`  
`ConstElement_ptr y, const size_t incy, Givaro::FloatDomain::Element_ptr A, const size_t lda, MMHelper<`  
`Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, type-`  
`name Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t ↔`  
`incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic,`  
`ModeCategories::LazyTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, type-`  
`name Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t ↔`  
`incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic,`  
`ModeCategories::DelayedTag > &H)`

## 13.85.1 Macro Definition Documentation

### 13.85.1.1 \_\_FFLASFFPACK\_fger\_INL

```
#define __FFLASFFPACK_fger_INL
```

## 13.86 fflas\_fger\_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- `#define` [\\_\\_FFPACK\\_fger\\_mp\\_INL](#)

### Functions

- `void fger (const Givaro::Modular< Givaro::Integer > &F, const size_t M, const size_t N, const typename`  
`Givaro::Integer alpha, typename Givaro::Integer *x, const size_t incx, typename Givaro::Integer *y, const`  
`size_t incy, typename Givaro::Integer *A, const size_t lda, MMHelper< Givaro::Modular< Givaro::Integer >,`  
`MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)`
- `template<typename RNS >`  
`void fger (const FFPACK::RNSInteger< RNS > &F, const size_t M, const size_t N, const typename`  
`FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::Element_ptr x,`  
`const size_t incx, typename FFPACK::RNSInteger< RNS >::Element_ptr y, const size_t incy, typename`  
`FFPACK::RNSInteger< RNS >::Element_ptr A, const size_t lda, MMHelper< FFPACK::RNSInteger< RNS`  
`>, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`

- template<typename [RNS](#) >  
void [fger](#) (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const size\_t M, const size\_t N, const type-  
name [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSIntegerMod](#)< [RNS](#)  
>::Element\_ptr x, const size\_t incx, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr y, const  
size\_t incy, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr A, const size\_t lda, [MMHelper](#)<  
[FFPACK::RNSIntegerMod](#)< [RNS](#) >, [MMHelperAlgo::Classic](#) > &H)

## 13.86.1 Macro Definition Documentation

### 13.86.1.1 \_\_FFPACK\_fger\_mp\_INL

```
#define __FFPACK_fger_mp_INL
```

## 13.87 fflas\_freduce.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/utils/cast.h"
#include "fflas-ffpack/fflas/fflas_freduce.inl"
```

### Data Structures

- struct [support\\_simd\\_mod](#)< T >
- struct [support\\_fast\\_mod](#)< T >
- struct [support\\_fast\\_mod](#)< float >
- struct [support\\_fast\\_mod](#)< double >
- struct [support\\_fast\\_mod](#)< int64\_t >

### Namespaces

- namespace [FFLAS](#)

### Functions

- template<class [Field](#) >  
void [freduce](#) (const [Field](#) &F, const size\_t n, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, typename  
[Field::Element\\_ptr](#) X, const size\_t incX)  
$$\text{freduce } x \leftarrow y \bmod F.$$
- template<class [Field](#) >  
void [freduce](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
$$\text{freduce } x \leftarrow x \bmod F.$$
- template<class [Field](#) >  
void [freduce\\_constoverride](#) (const [Field](#) &F, const size\_t m, typename [Field::ConstElement\\_ptr](#) A, const  
size\_t incX)
- template<class [Field](#), class [ConstOtherElement\\_ptr](#) >  
void [finit](#) (const [Field](#) &F, const size\_t n, [ConstOtherElement\\_ptr](#) Y, const size\_t incY, typename  
[Field::Element\\_ptr](#) X, const size\_t incX)
- template<class [Field](#) >  
void [finit](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
$$\text{finit initializes } X \text{ in } F.$$
- template<class [Field](#) >  
void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
$$\text{freduce } A \leftarrow A \bmod F.$$
- template<class [Field](#) >  
void [pfreduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t  
lda, const size\_t numths)



- template<class [Field](#) >  
void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $freduce\ A \leftarrow BmodF.$
- template<class [Field](#) >  
void [freduce\\_constoverride](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)
- template<class [Field](#) , class OtherElement\_ptr >  
void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, const OtherElement\_ptr B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $finit\ A \leftarrow BmodF.$
- template<class [Field](#) >  
void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)

## 13.88 fflas\_freduce.inl File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

### Data Structures

- struct [HelperMod](#)< [Field](#), [ElementCategories::MachineIntTag](#) >
- struct [HelperMod](#)< [Field](#), [FFLAS::ElementCategories::MachineFloatTag](#) >
- struct [HelperMod](#)< [Field](#), [FFLAS::ElementCategories::ArbitraryPrecIntTag](#) >
- struct [HelperMod](#)< [Field](#), [FFLAS::ElementCategories::FixedPrecIntTag](#) >

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::vectorised](#)
- namespace [FFLAS::vectorised::unswitch](#)
- namespace [FFLAS::details](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_freduce\\_INL](#)
- #define [FFLASFFPACK\\_COPY\\_REDUCE](#) 32 /\* TODO TO BENCHMARK LATER \*/

### Functions

- template<class T >  
std::enable\_if<!std::is\_integral< T >::value, T >::type [reduce](#) (T A, T B)
- template<class T >  
std::enable\_if< std::is\_integral< T >::value, T >::type [reduce](#) (T A, T B)
- template<> [Givaro::Integer](#) [reduce](#) ([Givaro::Integer](#) A, [Givaro::Integer](#) B)
- float [reduce](#) (float A, float B, float invB, float min, float max)
- double [reduce](#) (double A, double B, double invB, double min, double max)
- [int64\\_t](#) [reduce](#) ([int64\\_t](#) A, [int64\\_t](#) p, double invp, double min, double max, [int64\\_t](#) pow50rem)
- template<class [Field](#) >  
[Field::Element](#) [reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< [Field](#), [ElementCategories::MachineIntTag](#) > &H)
- template<class [Field](#) >  
[Field::Element](#) [reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< [Field](#), [ElementCategories::MachineFloatTag](#) > &H)



- `template<class Field >`  
`Field::Element reduce` (typename `Field::Element` A, `HelperMod`< `Field`, `ElementCategories::ArbitraryPrecIntTag` > &H)
- `template<class Field >`  
`std::enable_if< !FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp` (const `Field` &F, typename `Field::ConstElement_ptr` U, const size\_t &n, typename `Field::Element_ptr` T, `HelperMod`< `Field` > &H)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp` (const `Field` &F, typename `Field::ConstElement_ptr` U, const size\_t &n, const size\_t &incX, typename `Field::Element_ptr` T, `HelperMod`< `Field` > &H)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp` (const `Field` &F, typename `Field::ConstElement_ptr` U, const size\_t &n, typename `Field::Element_ptr` T)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp` (const `Field` &F, typename `Field::ConstElement_ptr` U, const size\_t &n, const size\_t &incX, typename `Field::Element_ptr` T)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce` (const `Field` &F, const size\_t m, typename `Field::Element_ptr` A, const size\_t incX, `FieldCategories::ModularTag`)
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce` (const `Field` &F, const size\_t m, typename `Field::ConstElement_ptr` B, const size\_t incY, typename `Field::Element_ptr` A, const size\_t incX, `FieldCategories::ModularTag`)
- `template<class Field , class FC >`  
`void freduce` (const `Field` &F, const size\_t m, typename `Field::Element_ptr` A, const size\_t incX, FC)
- `template<class Field , class FC >`  
`void freduce` (const `Field` &F, const size\_t m, typename `Field::ConstElement_ptr` B, const size\_t incY, typename `Field::Element_ptr` A, const size\_t incX, FC)

### 13.88.1 Macro Definition Documentation

### 13.88.1.1 FFLASFFPACK fflas freduce INL

```
#define FFLASFFPACK fflas_freduce_INL
```

### 13.88.1.2 FFLASFFPACK COPY REDUCE

```
#define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCHMARK LATER */
```

### 13.89 fflas\_freduce mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

## Namespaces

- namespace **FFLAS**

## Macros

- `#define FFLASFFPACK fflas freduce mp INL`

## Functions

- template<> void [freduce](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element\_ptr A, size\_t inc)
- template<> void [freduce](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, [FFPACK::rns\\_double::Element\\_ptr](#) A, size\_t lda)

## 13.89.1 Macro Definition Documentation

### 13.89.1.1 \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL

```
#define __FFLASFFPACK_fflas_freduce_mp_INL
```

## 13.90 fflas\_freivalds.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_freivalds\\_INL](#)

## Functions

- template<class [Field](#) >  
bool [freivalds](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::ConstElement\\_ptr](#) C, const size\_t ldc)

*freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*

## 13.90.1 Macro Definition Documentation

### 13.90.1.1 \_\_FFLASFFPACK\_freivalds\_INL

```
#define __FFLASFFPACK_freivalds_INL
```

## 13.91 fflas\_fscal.h File Reference

```
#include "fflas_fscal.inl"
```

## 13.92 fflas\_fscal.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::vectorised](#)
- namespace [FFLAS::vectorised::unswitch](#)
- namespace [FFLAS::details](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fscal\\_INL](#)

## Functions

- `template<class Field >`  
`std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<`  
`typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element_ptr T, const`  
`typename Field::Element alpha, typename Field::ConstElement_ptr U, const size_t n, HelperMod< Field >`  
`&H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX,`  
`FieldCategories::ModularTag)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t`  
`incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`
- `template<class Field, class FC >`  
`void fscal (const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X,`  
`const size_t incX, FC)`
- `template<class Field, class FC >`  
`void fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`
- `template<class Field >`  
`void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr`  
`X, const size_t incX)`  

$$fscal\, x \leftarrow \alpha \cdot x.$$
- `template<class Field >`  
`void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$fscal\, y \leftarrow \alpha \cdot x.$$
- `template<> void fscal (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a,`  
`Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y,`  
`const size_t incy)`
- `template<> void fscal (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element`  
`a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y,`  
`const size_t incy)`
- `template<> void fscal (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a,`  
`Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void fscal (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element`  
`a, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void fscal (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename`  
`Field::Element_ptr A, const size_t lda)`  

$$fscal\, A \leftarrow a \cdot A.$$
- `template<class Field >`  
`void fscal (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`

$$fscal\ B \leftarrow a \cdot A.$$

## 13.92.1 Macro Definition Documentation

### 13.92.1.1 \_\_FFLASFFPACK\_fscal\_INL

```
#define __FFLASFFPACK_fscal_INL
```

## 13.93 fflas\_fscal\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas_fscal.h"
#include "fflas_fgemm.inl"
#include "fflas-ffpack/fflas/fflas_freduce_mp.inl"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fscal\\_mp\\_INL](#)

### Functions

- template<> void [fscal](#)(const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t inc)
- template<> void [fscal](#)(const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t Ainc, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t Binc)
- template<> void [fscal](#)(const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t lda)
- template<> void [fscal](#)(const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t ldb)
- template<> void [fscal](#)(const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const typename [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element alpha, typename [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element\_ptr A, const size\_t inc)
- template<> void [fscal](#)(const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t Ainc, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t Binc)
- template<> void [fscal](#)(const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t lda)
- template<> void [fscal](#)(const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t ldb)

## 13.93.1 Macro Definition Documentation

### 13.93.1.1 \_\_FFLASFFPACK\_fscal\_mp\_INL

```
#define __FFLASFFPACK_fscal_mp_INL
```

## 13.94 fflas\_fsyr2k.inl File Reference

### Namespaces

- namespace [FFLAS](#)

**Macros**

- `#define __FFLASFFPACK_fflas_fsyk2k_INL`

**Functions**

- `template<class Field >`  
`Field::Element_ptr fsyr2k` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc)  
*fsyr2k: Symmetric Rank 2K update*

**13.94.1 Macro Definition Documentation****13.94.1.1 \_\_FFLASFFPACK\_fflas\_fsyk2k\_INL**

```
#define __FFLASFFPACK_fflas_fsyk2k_INL
```

**13.95 fflas\_fsyk.inl File Reference****Namespaces**

- namespace `FFLAS`

**Macros**

- `#define __FFLASFFPACK_fflas_fsyk_INL`

**Functions**

- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc)  
*fsyrk: Symmetric Rank K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const size\_t threshold=`__FFLASFFPACK_FSYRK_THRESHOLD`)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const size\_t N, const size\_t K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const `ParSeqHelper::Sequential` seq, const size\_t threshold)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const size\_t N, const size\_t K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const `ParSeqHelper::Parallel`< Cut, Param > par, const size\_t threshold)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const `FFLAS_UPLO` UpLo, const `FFLAS_TRANSPOSE` trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const std::vector< bool > &two← Block, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const size\_t threshold=`__FFLASFFPACK_FSYRK_THRESHOLD`)

*fsyrk: Symmetric Rank K update with diagonal scaling*

## 13.95.1 Macro Definition Documentation

### 13.95.1.1 \_\_FFLASFFPACK\_fflas\_fsyk\_INL

```
#define __FFLASFFPACK_fflas_fsyk_INL
```

## 13.96 fflas\_ftrmm.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ftrmm\\_INL](#)

### Functions

- template<class [Field](#) >  
void [ftrmm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
*ftrmm: **TR**angular **M**atrix **M**ultiply.*
- template<class [Field](#) >  
void [ftrmm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*ftrmm: **TR**angular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .*

## 13.96.1 Macro Definition Documentation

### 13.96.1.1 \_\_FFLASFFPACK\_ftrmm\_INL

```
#define __FFLASFFPACK_ftrmm_INL
```

## 13.97 fflas\_ftrsm.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ftrsm\\_INL](#)

### Functions

- template<class [Field](#) >  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)
- template<class [Field](#) >  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const [ParSeqHelper::Sequential](#) &PSH)

- template<class [Field](#) , class Cut , class Param >  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const [ParSeqHelper::Parallel](#)< Cut, Param > &PSH)
- template<class [Field](#) , class ParSeqTrait = [ParSeqHelper::Sequential](#)>  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, [TRSMHelper](#)< [StructureHelper::Recursive](#), ParSeqTrait > &H)

## 13.97.1 Macro Definition Documentation

### 13.97.1.1 \_\_FFLASFFPACK\_ftrsm\_INL

```
#define __FFLASFFPACK_ftrsm_INL
```

## 13.98 fflas\_ftrsm\_mp.inl File Reference

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

```
#include <cmath>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFPACK\\_ftrsm\\_mp\\_INL](#)

### Functions

- void [ftrsm](#) (const Givaro::Modular< Givaro::Integer > &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const Givaro::Integer alpha, const Givaro::Integer \*A, const size\_t lda, Givaro::Integer \*B, const size\_t ldb)
- void [cblas\\_impftrsm](#) (const enum [FFLAS\\_ORDER](#) Order, const enum [FFLAS\\_SIDE](#) Side, const enum [FFLAS\\_UPLO](#) Uplo, const enum [FFLAS\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_DIAG](#) Diag, const int M, const int N, const [FFPACK::rns\\_double\\_elt](#) alpha, [FFPACK::rns\\_double\\_elt\\_cstptr](#) A, const int lda, [FFPACK::rns\\_double\\_elt\\_ptr](#) B, const int ldb)

## 13.98.1 Detailed Description

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

## 13.98.2 Macro Definition Documentation

### 13.98.2.1 \_\_FFPACK\_ftrsm\_mp\_INL

```
#define __FFPACK_ftrsm_mp_INL
```

## 13.99 fflas\_ftrsv.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_ftrsv\\_INL](#)

### Functions

- `template<class Field >`  
`void ftrsv (const Field &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size_t N, typename Field::ConstElement\_ptr A, const size_t lda, typename Field::Element\_ptr X, int incX)`  
*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*

### 13.99.1 Macro Definition Documentation

#### 13.99.1.1 [\\_\\_FFLASFFPACK\\_ftrsv\\_INL](#)

```
#define __FFLASFFPACK_ftrsv_INL
```

## 13.100 fflas\_helpers.inl File Reference

```
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/flimits.h"
#include <algorithm>
```

### Data Structures

- struct [Auto](#)
- struct [Classic](#)
- struct [Winograd](#)
- struct [WinogradPar](#)
- struct [Bini](#)
- struct [AlgoChooser](#)< [ModeT](#), [ParSeq](#) >
- struct [AlgoChooser](#)< [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeq](#) >
- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) >  
*FGEMM Helper for Default and ConvertTo modes of operation.*
- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< [Dest](#) >, [ParSeqTrait](#) >
- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeTrait](#), [ParSeqTrait](#) >
- struct [Recursive](#)
- struct [Iterative](#)
- struct [Hybrid](#)
- struct [TRSMHelper](#)< [ReclterTrait](#), [ParSeqTrait](#) >  
*TRSM Helper.*

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)
- namespace [FFLAS::MMHelperAlgo](#)
- namespace [FFLAS::StructureHelper](#)  
*StructureHelper for ftrsm.*



## Macros

- `#define __FFLASFFPACK_fflas_fflas_mmhelper_INL`

## Functions

- `template<class Field >`  
`int WinogradSteps (const Field &F, const size_t &m)`  
*Computes the number of recursive levels to perform.*
- `template<class DFE >`  
`size_t min_types (const DFE &k)`
- `template<> size_t min_types (const Reclnt::rint< 6 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 7 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 8 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 9 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 10 > &k)`
- `template<> size_t min_types (const Givaro::Integer &k)`
- `template<class T >`  
`bool unfit (T x)`
- `template<> bool unfit (int64_t x)`
- `template<size_t K>`  
`bool unfit (Reclnt::rint< K > x)`
- `template<> bool unfit (Reclnt::rint< 6 > x)`

### 13.100.1 Macro Definition Documentation

#### 13.100.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL

```
#define __FFLASFFPACK_fflas_fflas_mmhelper_INL
```

## 13.101 igemm.doxy File Reference

## 13.102 igemm.h File Reference

```
#include "igemm_kernels.h"
#include "igemm_tools.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm.inl"
```

## Namespaces

- namespace `FFLAS`
- namespace `FFLAS::Protected`

## Enumerations

- enum `number_kind` { `zero` =0 , `one` =1 , `mone` =-1 , `other` =2 }

## Functions

- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>`  
`void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>`  
`void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`

- void `igemm` (const enum `FFLAS_TRANSPOSE` TransA, const enum `FFLAS_TRANSPOSE` TransB, size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, const `int64_t` beta, `int64_t` \*C, size\_t ldc)
- void `igemm_` (const enum `FFLAS_ORDER` Order, const enum `FFLAS_TRANSPOSE` TransA, const enum `FFLAS_TRANSPOSE` TransB, const size\_t M, const size\_t N, const size\_t K, const `int64_t` alpha, const `int64_t` \*A, const size\_t lda, const `int64_t` \*B, const size\_t ldb, const `int64_t` beta, `int64_t` \*C, const size\_t ldc)

### 13.103 igemm.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
```

#### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::Protected`

#### Macros

- #define `__FFLASFFPACK_fflas_igemm_igemm_INL`

#### Functions

- template<enum `FFLAS_TRANSPOSE` tA, enum `FFLAS_TRANSPOSE` tB>  
void `igemm_colmajor` (size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, `int64_t` \*C, size\_t ldc)
- template<enum `FFLAS_TRANSPOSE` tA, enum `FFLAS_TRANSPOSE` tB, enum `number_kind` alpha\_kind>  
void `igemm_colmajor` (size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, `int64_t` \*C, size\_t ldc)
- void `igemm` (const enum `FFLAS_TRANSPOSE` TransA, const enum `FFLAS_TRANSPOSE` TransB, size\_t rows, size\_t cols, size\_t depth, const `int64_t` alpha, const `int64_t` \*A, size\_t lda, const `int64_t` \*B, size\_t ldb, const `int64_t` beta, `int64_t` \*C, size\_t ldc)
- void `igemm_` (const enum `FFLAS_ORDER` Order, const enum `FFLAS_TRANSPOSE` TransA, const enum `FFLAS_TRANSPOSE` TransB, const size\_t M, const size\_t N, const size\_t K, const `int64_t` alpha, const `int64_t` \*A, const size\_t lda, const `int64_t` \*B, const size\_t ldb, const `int64_t` beta, `int64_t` \*C, const size\_t ldc)

#### 13.103.1 Macro Definition Documentation

##### 13.103.1.1 `__FFLASFFPACK_fflas_igemm_igemm_INL`

```
#define __FFLASFFPACK_fflas_igemm_igemm_INL
```

### 13.104 igemm\_kernels.h File Reference

```
#include "igemm_kernels.inl"
```

#### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::details`

#### Functions

- template<enum `number_kind` K>  
void `igebb44` (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const `int64_t` alpha, const `int64_t` \*bIA, const `int64_t` \*bIB, `int64_t` \*C, size\_t ldc)

- template<enum [number\\_kind](#) K>  
void [igebb24](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb14](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb41](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb21](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb11](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebp](#) (size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blockA, size\_t lda, const [int64\\_t](#) \*blockB, size\_t ldb, [int64\\_t](#) \*C, size\_t ldc)

## 13.105 igemm\_kernels.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm_tools.h"
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_igemm\\_igemm\\_kernels\\_INL](#)

### Functions

- template<enum [number\\_kind](#) K>  
void [igebb44](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb24](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb14](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb41](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb21](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebb11](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum [number\\_kind](#) K>  
void [igebp](#) (size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blockA, size\_t lda, const [int64\\_t](#) \*blockB, size\_t ldb, [int64\\_t](#) \*C, size\_t ldc)

### 13.105.1 Macro Definition Documentation

#### 13.105.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_kernels_INL
```

### 13.106 igemm\_tools.h File Reference

```
#include "igemm_tools.inl"
```

#### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details](#)

#### Functions

- template<size\_t k, bool transpose>  
void [pack\\_lhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- template<size\_t k, bool transpose>  
void [pack\\_rhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- void [gebp](#) (size\_t rows, size\_t cols, size\_t depth, int64\_t \*C, size\_t ldc, const int64\_t \*blockA, size\_t lda, const int64\_t \*BlockB, size\_t ldb, int64\_t \*BlockW)
- void [BlockingFactor](#) (size\_t &m, size\_t &n, size\_t &k)

### 13.107 igemm\_tools.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

#### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details](#)

#### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_igemm\\_igemm\\_tools\\_INL](#)

#### Functions

- template<size\_t k, bool transpose>  
void [pack\\_rhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- template<size\_t k, bool transpose>  
void [pack\\_lhs](#) (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)
- void [BlockingFactor](#) (size\_t &m, size\_t &n, size\_t &k)

### 13.107.1 Macro Definition Documentation

#### 13.107.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL
```

### 13.108 fflas\_level1.inl File Reference

#### Namespaces

- namespace [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_fflas_fflas_level1_INL`

## Functions

- `template<class Field >`  
`void freduce (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$\text{freduce } x \leftarrow x \bmod F.$$
- `template<class Field >`  
`void freduce (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$\text{freduce } x \leftarrow y \bmod F.$$
- `template<class Field, class OtherElement_ptr >`  
`void finit (const Field &F, const size_t n, const OtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$\text{finit } x \leftarrow y \bmod F.$$
- `template<class Field >`  
`void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$\text{finit Initializes } X \text{ in } F.$$
- `template<class Field, class OtherElement_ptr >`  
`void fconvert (const Field &F, const size_t n, OtherElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  

$$\text{fconvert } x \leftarrow y \bmod F.$$
- `template<class Field >`  
`void fnegin (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$\text{fnegin } x \leftarrow -x.$$
- `template<class Field >`  
`void fneg (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$\text{fneg } x \leftarrow -y.$$
- `template<class Field >`  
`void fzero (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$\text{fzero} : A \leftarrow 0.$$
- `template<class Field, class Randlter >`  
`void frand (const Field &F, Randlter &G, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$\text{frand} : A \leftarrow \text{random}.$$
- `template<class Field >`  
`bool fiszero (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX)`  

$$\text{fiszero} : \text{test } X = 0.$$
- `template<class Field >`  
`bool fequal (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  

$$\text{fequal} : \text{test } X = Y.$$
- `template<class Field >`  
`void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$\text{fassign} : x \leftarrow y.$$
- `template<class Field >`  
`void fscaln (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)`  

$$\text{fscaln } x \leftarrow \alpha \cdot x.$$
- `template<class Field >`  
`void fscal (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`

- $$fscal\ y \leftarrow \alpha \cdot x.$$
  - template<class [Field](#) >  
void [faxpy](#) (const [Field](#) &F, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- template<class [Field](#) >  
void [faxpby](#) (const [Field](#) &F, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  

$$faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY)  

$$fdot: \text{dot product } x^T y.$$
- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, const [ParSeqHelper::Sequential](#) seq)
- template<typename [Field](#) , class Cut , class Param >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, const [ParSeqHelper::Parallel](#)< Cut, Param > par)
- template<class [Field](#) >  
void [fswap](#) (const [Field](#) &F, const size\_t N, typename [Field::Element\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  

$$fswap: X \leftrightarrow Y.$$
- template<class [Field](#) >  
void [pfadd](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const size\_t numths)
- template<class [Field](#) >  
void [pfsub](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const size\_t numths)
- template<class [Field](#) >  
void [pfaddin](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc, size\_t numths)
- template<class [Field](#) >  
void [pfsubin](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc, size\_t numths)
- template<class [Field](#) >  
void [fadd](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc)
- template<class [Field](#) >  
void [fsub](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc)
- template<class [Field](#) >  
void [faddin](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc)
- template<class [Field](#) >  
void [fsubin](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc)
- template<class [Field](#) >  
void [fadd](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc)

## 13.108.1 Macro Definition Documentation

### 13.108.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL

```
#define __FFLASFFPACK_fflas_fflas_level1_INL
```

## 13.109 fflas\_level2.inl File Reference

```
#include "givaro/zring.h"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_level2\\_INL](#)

### Functions

- template<class [Field](#) >  
void [fassign](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fassign :  $A \leftarrow B$ .*
- template<class [Field](#) >  
void [fzero](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fzero :  $A \leftarrow 0$ .*
- template<class [Field](#) , class RandIter >  
void [frand](#) (const [Field](#) &F, RandIter &G, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*frand :  $A \leftarrow random$ .*
- template<class [Field](#) >  
bool [fequal](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)  
*fequal : test  $A = B$ .*
- template<class [Field](#) >  
bool [fiszero](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)  
*fiszero : test  $A = 0$ .*
- template<class [Field](#) >  
void [fidentity](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) &d)  
*creates a diagonal matrix*
- template<class [Field](#) >  
void [fidentity](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*creates a diagonal matrix*
- template<class [Field](#) >  
void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*freduce  $A \leftarrow A \bmod F$ .*
- template<class [Field](#) >  
void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*freduce  $A \leftarrow B \bmod F$ .*
- template<class [Field](#) , class OtherElement\_ptr >  
void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, const OtherElement\_ptr B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)

- $finit\ A \leftarrow B \bmod F.$ 
  - template<class [Field](#) , class OtherElement\_ptr >  
void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*finit initializes A in  $\mathbb{F}_S$ .*
  - template<class [Field](#) , class OtherElement\_ptr >  
void [fconvert](#) (const [Field](#) &F, const size\_t m, const size\_t n, OtherElement\_ptr A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)  
*fconvert  $A \leftarrow B \bmod F$ .*
  - template<class [Field](#) >  
void [fnegin](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fnegin  $A \leftarrow -A$ .*
  - template<class [Field](#) >  
void [fneg](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fneg  $A \leftarrow -B$ .*
  - template<class [Field](#) >  
void [fscaln](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fscaln  $A \leftarrow a \cdot A$ .*
  - template<class [Field](#) >  
void [fscal](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
*fscal  $B \leftarrow a \cdot A$ .*
  - template<class [Field](#) >  
void [faxpy](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t ldx, typename [Field::Element\\_ptr](#) Y, const size\_t ldy)  
*faxpy :  $y \leftarrow \alpha \cdot x + y$ .*
  - template<class [Field](#) >  
void [faxpby](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t ldx, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t ldy)  
*faxpby :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .*
  - template<class [Field](#) >  
void [fmove](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
*fmove :  $A \leftarrow B$  and  $B \leftarrow 0$ .*
  - template<class [Field](#) >  
void [fadd](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fadd : matrix addition.*
  - template<class [Field](#) >  
void [fsub](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fsub : matrix subtraction.*
  - template<class [Field](#) >  
void [fsubin](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fsubin  $C = C - B$*
  - template<class [Field](#) >  
void [fadd](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fadd : matrix addition with scaling.*



- template<class [Field](#) >  
void [faddin](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*faddin*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) TransA, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  
*finite prime Field GEneral Matrix Vector multiplication.*
- template<class [Field](#) >  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fger: rank one update of a general matrix*
- template<class [Field](#) >  
void [ftrsv](#) (const [Field](#) &F, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)  
*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*
- template<class [Field](#) >  
size\_t [bitsize](#) (const [Field](#) &F, size\_t M, size\_t N, const typename [Field::ConstElement\\_ptr](#) A, size\_t lda)  
*bitsize: Computes the largest bitsize of the matrix' coefficients.*
- template<> size\_t [bitsize](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > > (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, size\_t M, size\_t N, const [Givaro::Integer](#) \*A, size\_t lda)
- template<class [Field](#) >  
void [ftrmv](#) (const [Field](#) &F, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)  
*ftrsm: TRIangular Matrix Vector prodcut Computes  $X \leftarrow \text{op}(A)X$*

## 13.109.1 Macro Definition Documentation

### 13.109.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL

```
#define __FFLASFFPACK_fflas_fflas_level2_INL
```

## 13.110 fflas\_level3.inl File Reference

```
#include "fflas_bounds.inl"
#include "fflas_helpers.inl"
#include "fflas-ffpack/paladin/parallel.h"
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_level3\\_INL](#)
- #define [\\_\\_FFLAS\\_\\_TRSM\\_READONLY](#)

## Functions

- `template<class Field >`  
`void MatF2MatD_Triangular (const Field &F, Givaro::DoubleDomain::Element_ptr S, const size_t lds, type-`  
`name Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`  
`void MatF2MatFI_Triangular (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename`  
`Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE`  
`TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*ftrsm: **TR**angular **S**ystem solve with **M**atrix.*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE`  
`TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*ftrmm: **TR**angular **M**atrix **M**ultiply.*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE`  
`TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*ftrmm: **TR**angular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha \text{Bop}(A) + \text{beta}C$ .*
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const`  
`size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const`  
`size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fsyrk: Symmetric Rank K update*
- `template<class Field >`  
`Field::Element_ptr fsyr2k (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans,`  
`const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A,`  
`const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta,`  
`typename Field::Element_ptr C, const size_t ldc)`  
*fsyr2k: Symmetric Rank 2K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const`  
`size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t`  
`lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename`  
`Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)`  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const`  
`size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t`  
`lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename`  
`Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq, const size_t threshold)`
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const`  
`size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t`  
`lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename`  
`Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size_t thresh-`  
`old)`
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans,`  
`const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const`  
`size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const std::vector< bool > &two←`  
`Block, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t`  
`threshold=__FFLASFFPACK_FSYRK_THRESHOLD)`

*fsyrk: Symmetric Rank K update with diagonal scaling*

- template<typename `Field` >  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc)  
*fgemm: Field GENERAL Matrix Multiply.*
- template<typename `Field` >  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, const `ParSeqHelper::Sequential` seq)
- template<typename `Field` , class `Cut` , class `Param` >  
`Field::Element_ptr fgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, const `ParSeqHelper::Parallel`< `Cut`, `Param` > par)
- template<typename `Field` >  
`Field::Element_ptr pfgemm` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `size_t` numthreads=0)
- template<class `Field` >  
`Field::Element * pfgemm_1D_rec` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, const typename `Field::Element_ptr` A, const `size_t` lda, const typename `Field::Element_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element` \*C, const `size_t` ldc, `size_t` seuil)
- template<class `Field` >  
`Field::Element * pfgemm_2D_rec` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, const typename `Field::Element_ptr` A, const `size_t` lda, const typename `Field::Element_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element` \*C, const `size_t` ldc, `size_t` seuil)
- template<class `Field` >  
`Field::Element * pfgemm_3D_rec` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, const typename `Field::Element_ptr` A, const `size_t` lda, const typename `Field::Element_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `size_t` seuil, `size_t` \*x)
- template<class `Field` >  
`Field::Element_ptr pfgemm_3D_rec2` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const typename `Field::Element` alpha, const typename `Field::Element_ptr` A, const `size_t` lda, const typename `Field::Element_ptr` B, const `size_t` ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc, `size_t` seuil, `size_t` \*x)
- template<class `Field` >  
`Field::Element_ptr fsquare` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const `size_t` ldc)

*fsquare: Squares a matrix.*

## 13.110.1 Macro Definition Documentation

### 13.110.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL

```
#define __FFLASFFPACK_fflas_fflas_level3_INL
```

### 13.110.1.2 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

## 13.111 fflas\_pfgemm.inl File Reference

```
#include "fflas-ffpack/paladin/blockcuts.inl"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/paladin/pfgemm_variants.inl"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_pfgemm\\_INL](#)
- #define [\\_\\_FFLASFFPACK\\_SEQPARTHRESHOLD](#) 220
- #define [\\_\\_FFLASFFPACK\\_DIMKPENALTY](#) 1

### Functions

- template<class [Field](#) , class ModeTrait , class Strat , class Param >  
std::enable\_if<!std::is\_same< ModeTrait, [ModeCategories::ConvertTo< ElementCategories::RNSElementTag](#)  
>>::value, typename Field::Element\_ptr >::type [fgemm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_TRANSPOSE](#)  
ta, const [FFLAS::FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const  
typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename  
[Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#)  
C, const size\_t ldc, [MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,](#)  
Param >> &H)

### 13.111.1 Macro Definition Documentation

#### 13.111.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_pfgemm\\_INL](#)

```
#define __FFLASFFPACK_fflas_pfgemm_INL
```

#### 13.111.1.2 [\\_\\_FFLASFFPACK\\_SEQPARTHRESHOLD](#)

```
#define __FFLASFFPACK_SEQPARTHRESHOLD 220
```

#### 13.111.1.3 [\\_\\_FFLASFFPACK\\_DIMKPENALTY](#)

```
#define __FFLASFFPACK_DIMKPENALTY 1
```

## 13.112 fflas\_pfttrsm.inl File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_pfttrsm\\_INL](#)
- #define [PTRSM\\_HYBRID\\_THRESHOLD](#) 256

## Functions

- template<class [Field](#) , class [Cut](#) , class [Param](#) >  
[Field::Element\\_ptr](#) [ftrsm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_TRANSPOSE](#) TA, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, [TRSMHelper](#)< [StructureHelper::Iterative](#), [ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > > &H)
- template<class [Field](#) , class [Cut](#) , class [Param](#) >  
[Field::Element\\_ptr](#) [ftrsm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_TRANSPOSE](#) TA, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, [TRSMHelper](#)< [StructureHelper::Hybrid](#), [ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > > &H)

## 13.112.1 Macro Definition Documentation

### 13.112.1.1 \_\_FFLASFFPACK\_fflas\_pftrsm\_INL

```
#define __FFLASFFPACK_fflas_pftrsm_INL
```

### 13.112.1.2 PTRSM\_HYBRID\_THRESHOLD

```
#define PTRSM_HYBRID_THRESHOLD 256
```

## 13.113 fflas\_simd.h File Reference

```
#include "fflas-ffpack/utils/fflas_intrinsic.h"
#include <iostream>
#include <type_traits>
#include <limits>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include <fflas-ffpack/fflas/fflas_simd/simd_modular.inl>
```

## Data Structures

- struct [support\\_simd](#)< [T](#) >
- struct [is\\_simd](#)< [T](#) >
- struct [NoSimd](#)< [T](#) >
- struct [SimdChooser](#)< [T](#), bool, bool >
- struct [SimdChooser](#)< [T](#), false, b >
- struct [SimdChooser](#)< [T](#), true, false >
- struct [SimdChooser](#)< [T](#), true, true >

## Namespaces

- namespace [FFLAS](#)

## Macros

- #define [SIMD\\_INT](#) 1
- #define [INLINE](#) inline
- #define [CONST](#)
- #define [PURE](#)
- #define [NORML\\_MOD](#)(C, P, NEGP, MIN, MAX, Q, T)
- #define [FLOAT\\_MOD](#)(C, P, INV, Q)

## Typedefs

- `template<class T >`  
using `Simd` = `typename SimdChooser< T >::value`

## 13.113.1 Macro Definition Documentation

### 13.113.1.1 SIMD\_INT

```
#define SIMD_INT 1
```

### 13.113.1.2 INLINE

```
#define INLINE inline
```

### 13.113.1.3 CONST

```
#define CONST
```

### 13.113.1.4 PURE

```
#define PURE
```

### 13.113.1.5 NORML\_MOD

```
#define NORML_MOD(
 C,
 P,
 NEGP,
 MIN,
 MAX,
 Q,
 T)
```

#### Value:

```
{
 \
 Q = greater(C, MAX);
 \
 T = lesser(C, MIN);
 \
 Q = vand(Q, NEGP);
 \
 T = vand(T, P);
 \
 Q = vor(Q, T);
 \
 C = add(C, Q);
 \
}
```

### 13.113.1.6 FLOAT\_MOD

```
#define FLOAT_MOD(
 C,
 P,
 INVP,
 Q)
```

#### Value:

```
{
 \
 Q = mul(C, INVP);
 \
 Q = floor(Q);
 \
 C = fnmadd(C, Q, P);
 \
}
```

### 13.113.2 Typedef Documentation

#### 13.113.2.1 Simd

```
template<class T >
using Simd = typename SimdChooser<T>::value
```

## 13.114 simd.doxy File Reference

### 13.115 simd128.inl File Reference

```
#include "simd128_float.inl"
#include "simd128_double.inl"
```

#### Data Structures

- struct [Simd128fp\\_base](#)
- struct [Simd128i\\_base](#)

#### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_INL](#)

#### Typedefs

- template<class T >  
using [Simd128](#) = [Simd128\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

### 13.115.1 Macro Definition Documentation

#### 13.115.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL
```

### 13.115.2 Typedef Documentation

#### 13.115.2.1 Simd128

```
template<class T >
using Simd128 = Simd128_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↵
::is_signed<T>::value, sizeof(T)>
```

## 13.116 simd128\_double.inl File Reference

#### Data Structures

- struct [Simd128\\_impl](#)< true, false, true, 8 >

#### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_double\\_INL](#)

### 13.116.1 Macro Definition Documentation

#### 13.116.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL
```

## 13.117 simd128\_float.inl File Reference

### Data Structures

- struct [Simd128\\_impl< true, false, true, 4 >](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_float\\_INL](#)

### 13.117.1 Macro Definition Documentation

#### 13.117.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL
```

## 13.118 simd128\_int16.inl File Reference

### Data Structures

- struct [Simd128\\_impl< true, true, true, 2 >](#)
- union [Simd128\\_impl< true, true, true, 2 >::Converter](#)
- struct [Simd128\\_impl< true, true, false, 2 >](#)
- union [Simd128\\_impl< true, true, false, 2 >::Converter](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int16\\_INL](#)

### 13.118.1 Macro Definition Documentation

#### 13.118.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL
```

## 13.119 simd128\_int32.inl File Reference

### Data Structures

- struct [Simd128\\_impl< true, true, true, 4 >](#)
- union [Simd128\\_impl< true, true, true, 4 >::Converter](#)
- struct [Simd128\\_impl< true, true, false, 4 >](#)
- union [Simd128\\_impl< true, true, false, 4 >::Converter](#)

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int32\\_INL](#)

### 13.119.1 Macro Definition Documentation

#### 13.119.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL
```



## 13.120 simd128\_int64.inl File Reference

### Data Structures

- struct [Simd128\\_impl](#)< true, true, true, 8 >
- union [Simd128\\_impl](#)< true, true, true, 8 >::Converter
- struct [Simd128\\_impl](#)< true, true, false, 8 >
- union [Simd128\\_impl](#)< true, true, false, 8 >::Converter

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int64\\_INL](#)
- #define [vect\\_t](#) [Simd128\\_impl](#)<true,true,true,8>::vect\_t

### 13.120.1 Macro Definition Documentation

#### 13.120.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
```

#### 13.120.1.2 vect\_t

```
#define vect_t Simd128_impl<true,true,true,8>::vect_t
```

## 13.121 simd256.inl File Reference

```
#include "simd256_float.inl"
#include "simd256_double.inl"
```

### Data Structures

- struct [Simd256fp\\_base](#)
- struct [Simd256i\\_base](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_INL](#)

### Typedefs

- template<class T >  
using [Simd256](#) = [Simd256\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

### 13.121.1 Macro Definition Documentation

#### 13.121.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL
```

### 13.121.2 Typedef Documentation

#### 13.121.2.1 Simd256

```
template<class T >
using Simd256 = Simd256_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↵
::is_signed<T>::value, sizeof(T)>
```

## 13.122 simd256\_double.inl File Reference

### Data Structures

- struct [Simd256\\_impl](#)< true, false, true, 8 >

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_double\\_INL](#)

### 13.122.1 Macro Definition Documentation

#### 13.122.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL
```

## 13.123 simd256\_float.inl File Reference

### Data Structures

- struct [Simd256\\_impl](#)< true, false, true, 4 >

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_float\\_INL](#)

### 13.123.1 Macro Definition Documentation

#### 13.123.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL
```

## 13.124 simd256\_int16.inl File Reference

### Data Structures

- struct [Simd256\\_impl](#)< true, true, true, 2 >
- union [Simd256\\_impl](#)< true, true, true, 2 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 2 >
- union [Simd256\\_impl](#)< true, true, false, 2 >::Converter

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int16\\_INL](#)

### 13.124.1 Macro Definition Documentation

#### 13.124.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL
```

## 13.125 simd256\_int32.inl File Reference

### Data Structures

- struct [Simd256\\_impl](#)< true, true, true, 4 >
- union [Simd256\\_impl](#)< true, true, true, 4 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 4 >
- union [Simd256\\_impl](#)< true, true, false, 4 >::Converter

**Macros**

- `#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL`

**13.125.1 Macro Definition Documentation****13.125.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL**

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL
```

**13.126 simd256\_int64.inl File Reference****Data Structures**

- struct [Simd256\\_impl< true, true, true, 8 >](#)
- union [Simd256\\_impl< true, true, true, 8 >::Converter](#)
- struct [Simd256\\_impl< true, true, false, 8 >](#)
- union [Simd256\\_impl< true, true, false, 8 >::Converter](#)

**Macros**

- `#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL`
- `#define vect_t Simd256_impl<true, true, true, 8>::vect_t`

**13.126.1 Macro Definition Documentation****13.126.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL**

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
```

**13.126.1.2 vect\_t**

```
#define vect_t Simd256_impl<true, true, true, 8>::vect_t
```

**13.127 simd512.inl File Reference**

```
#include "simd512_float.inl"
#include "simd512_double.inl"
#include "simd512_int64.inl"
```

**Data Structures**

- struct [Simd512fp\\_base](#)
- struct [Simd512i\\_base](#)

**Macros**

- `#define __FFLASFFPACK_simd512_INL`

**Typedefs**

- `template<class T >`  
`using Simd512 = Simd512_impl< std::is_arithmetic< T >::value, std::is_integral< T >::value, std::is_↵`  
`signed< T >::value, sizeof(T)>`

### 13.127.1 Macro Definition Documentation

#### 13.127.1.1 \_\_FFLASFFPACK\_simd512\_INL

```
#define __FFLASFFPACK_simd512_INL
```

### 13.127.2 Typedef Documentation

#### 13.127.2.1 Simd512

```
template<class T >
using Simd512 = Simd512_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

## 13.128 simd512\_double.inl File Reference

### Data Structures

- struct [Simd512\\_impl< true, false, true, 8 >](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_double\\_INL](#)

### 13.128.1 Macro Definition Documentation

#### 13.128.1.1 \_\_FFLASFFPACK\_simd512\_double\_INL

```
#define __FFLASFFPACK_simd512_double_INL
```

## 13.129 simd512\_float.inl File Reference

### Data Structures

- struct [Simd512\\_impl< true, false, true, 4 >](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_float\\_INL](#)

### 13.129.1 Macro Definition Documentation

#### 13.129.1.1 \_\_FFLASFFPACK\_simd512\_float\_INL

```
#define __FFLASFFPACK_simd512_float_INL
```

## 13.130 simd512\_int32.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd/simd512_int64.inl"
```

### Data Structures

- struct [Simd256\\_impl< true, true, true, 4 >](#)
- union [Simd256\\_impl< true, true, true, 4 >::Converter](#)
- struct [Simd256\\_impl< true, true, false, 4 >](#)
- union [Simd256\\_impl< true, true, false, 4 >::Converter](#)

**Macros**

- `#define __FFLASFFPACK_simd512_int32_INL`

**13.130.1 Macro Definition Documentation****13.130.1.1 \_\_FFLASFFPACK\_simd512\_int32\_INL**

```
#define __FFLASFFPACK_simd512_int32_INL
```

**13.131 simd512\_int64.inl File Reference****Data Structures**

- struct [Simd512\\_impl< true, true, true, 8 >](#)
- union [Simd512\\_impl< true, true, true, 8 >::Converter](#)
- struct [Simd512\\_impl< true, true, false, 8 >](#)
- union [Simd512\\_impl< true, true, false, 8 >::Converter](#)

**Macros**

- `#define _simd512_int64_INL`
- `#define vect_t Simd512_impl<true, true, true, 8>::vect_t`

**13.131.1 Macro Definition Documentation****13.131.1.1 \_simd512\_int64\_INL**

```
#define _simd512_int64_INL
```

**13.131.1.2 vect\_t**

```
#define vect_t Simd512_impl<true, true, true, 8>::vect_t
```

**13.132 simd\_modular.inl File Reference****Data Structures**

- class [FieldSimd< \\_Field >](#)

**13.133 fflas\_sparse.h File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/paladin/parallel.h"
#include <recint/recint.h>
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <type_traits>
#include <vector>
#include <iostream>
#include "fflas-ffpack/fflas/fflas_sparse/sparse_matrix_traits.h"
#include "fflas-ffpack/fflas/fflas_sparse/utils.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr.h"
```

```
#include "fflas-ffpack/fflas/fflas_sparse/coo.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell.h"
#include "fflas-ffpack/fflas/fflas_sparse/sell.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd.h"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo.h"
#include "fflas-ffpack/fflas/fflas_sparse.inl"
#include "fflas-ffpack/fflas/fflas_sparse/read_sparse.h"
```

## Data Structures

- struct [HelperFlag](#)
- struct [CsrMat< Field >](#)
- struct [CooMat< Field >](#)
- struct [EllMat< Field >](#)
- struct [SpMat< Field, flag >](#)

## Namespaces

- namespace [MKL\\_CONFIG](#)
- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details](#)

## Macros

- [#define index\\_t uint32\\_t](#)
- [#define ROUND\\_DOWN\(x, s\) \(\(x\) & ~\(\(s\)-1\)\)](#)
- [#define \\_\\_FFLASFFPACK\\_CACHE\\_LINE\\_SIZE 64](#)
- [#define assume\\_aligned\(pout, pin, v\) decltype\(pin\) pout = pin;](#)
- [#define DENSE\\_THRESHOLD 0.5](#)

## Enumerations

- enum class [SparseMatrix\\_t](#) {  
[CSR](#) , [CSR\\_ZO](#) , [CSC](#) , [CSC\\_ZO](#) ,  
[COO](#) , [COO\\_ZO](#) , [ELL](#) , [ELL\\_ZO](#) ,  
[SELL](#) , [SELL\\_ZO](#) , [ELL\\_simd](#) , [ELL\\_simd\\_ZO](#) ,  
[CSR\\_HYB](#) , [HYB\\_ZO](#) }

## Functions

- template<class [Field](#) >  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y)
- template<class [Field](#) >  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y, const int ldy)
- template<class [Field](#) , class SM , class FC , class MZO >  
std::enable\_if<!(std::is\_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineFloat](#)  
>::value||std::is\_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineIntTag](#)  
>::value)>::type [fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, type-  
name [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- template<class [Field](#) , class SM , class FC , class MZO >  
std::enable\_if< std::is\_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineFloat](#)  
>::value||std::is\_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineIntTag](#)  
>::value >::type [fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, type-  
name [Field::Element\\_ptr](#) y, FC fc, MZO mzo)

- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`

- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type pfspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type pfspmm\_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`



- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::false_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::true_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::true_type)`
- `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)`

## 13.133.1 Macro Definition Documentation

### 13.133.1.1 index\_t

```
#define index_t uint32_t
```

### 13.133.1.2 ROUND\_DOWN

```
#define ROUND_DOWN(
 x,
 s) ((x) & ~((s)-1))
```

### 13.133.1.3 \_\_FFLASFFPACK\_CACHE\_LINE\_SIZE

```
#define __FFLASFFPACK_CACHE_LINE_SIZE 64
```

### 13.133.1.4 assume\_aligned

```
#define assume_aligned(
 pout,
 pin,
 v) decltype(pin) pout = pin;
```

### 13.133.1.5 DENSE\_THRESHOLD

```
#define DENSE_THRESHOLD 0.5
```

## 13.134 fflas\_sparse.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_sparse\\_INL](#)

### Functions

- `template<class Field >`  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y)
- `template<class Field >`  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y, const int ldy)
- `template<class Field , class SM , class FC , class MZO >`  
std::enable\_if<!std::is\_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineFloat](#) >::value||std::is\_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineIntTag](#) >::value>::type [fspmvp\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- `template<class Field , class SM , class FC , class MZO >`  
std::enable\_if< std::is\_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineFloat](#) >::value||std::is\_same< typenameElementTraits< typenameField::Element >::value, [ElementCategories::MachineIntTag](#) >::value >::type [fspmvp\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- `template<class Field , class SM >`  
void [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::GenericTag](#), [NotZOSparseMatrix](#))
- `template<class Field , class SM >`  
std::enable\_if<[isSparseMatrixSimdFormat](#)< [Field](#), SM >::value >::type [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
- `template<class Field , class SM >`  
std::enable\_if< [isSparseMatrixSimdFormat](#)< [Field](#), SM >::value &&[support\\_simd](#)< typenameField::Element >::value >::type [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
- `template<class Field , class SM >`  
std::enable\_if<[isSparseMatrixSimdFormat](#)< [Field](#), SM >::value >::type [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::ModularTag](#), [NotZOSparseMatrix](#))
- `template<class Field , class SM >`  
std::enable\_if< [isSparseMatrixSimdFormat](#)< [Field](#), SM >::value &&[support\\_simd](#)< typenameField::Element >::value >::type [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::ModularTag](#), [NotZOSparseMatrix](#))
- `template<class Field , class SM >`  
void [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::GenericTag](#), [ZOSparseMatrix](#))
- `template<class Field , class SM >`  
std::enable\_if<[isSparseMatrixSimdFormat](#)< [Field](#), SM >::value >::type [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::UnparametricTag](#), [ZOSparseMatrix](#))
- `template<class Field , class SM >`  
std::enable\_if< [isSparseMatrixSimdFormat](#)< [Field](#), SM >::value &&[support\\_simd](#)< typenameField::Element >::value >::type [fspmvp](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::UnparametricTag](#), [ZOSparseMatrix](#))

- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)`

## 13.134.1 Macro Definition Documentation

### 13.134.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL

```
#define __FFLASFFPACK_fflas_fflas_sparse_INL
```

## 13.135 coo.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmmm.inl"
```

### Data Structures

- struct [Sparse< \\_Field, SparseMatrix\\_t::COO >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::COO\\_ZO >](#)

### Namespaces

- namespace [FFLAS](#)

### Functions

- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse< Field, SparseMatrix\\_t::COO >](#) &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse< Field, SparseMatrix\\_t::COO\\_ZO >](#) &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const [Sparse< Field, SparseMatrix\\_t::COO >](#) &A)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const [Sparse< Field, SparseMatrix\\_t::COO\\_ZO >](#) &A)

## 13.136 coo\_spmmm.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_coo\\_spmmm\\_INL](#)

### Functions

- template<class [Field](#) >  
void [fspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::COO >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [fspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::COO >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
void [fspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::COO >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, const [int64\\_t](#) kmax)
- template<class [Field](#) >  
void [fspmm\\_simd\\_aligned](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::COO >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, const [int64\\_t](#) kmax)
- template<class [Field](#) >  
void [fspmm\\_simd\\_unaligned](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::COO >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, const [int64\\_t](#) kmax)

- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

### 13.136.1 Macro Definition Documentation

#### 13.136.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

## 13.137 coo\_spmv.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::sparse_details_impl`

### Macros

- `#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`

- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 13.137.1 Macro Definition Documentation

#### 13.137.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

## 13.138 coo\_utils.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_coo_utils_INL`

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::COO > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::COO_ZO > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::COO_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

### 13.138.1 Macro Definition Documentation

#### 13.138.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_utils_INL
```

## 13.139 csr.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
```

### Data Structures

- struct [Sparse< \\_Field, SparseMatrix\\_t::CSR >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_ZO >](#)

## Namespaces

- namespace [FFLAS](#)

## Functions

- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A)

## 13.140 csr\_pspmm.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_pspmm\\_INL](#)

### Functions

- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, const [int64\\_t](#) kmax)
- template<class [Field](#) >  
void [pfspmm\\_one](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmm\\_mone](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmm\\_one](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
void [pfspmm\\_mone](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, [FieldCategories::UnparametricTag](#))

### 13.140.1 Macro Definition Documentation

#### 13.140.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL
```

## 13.141 csr\_pspmv.inl File Reference

```
#include <thread>
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_pspmv\\_INL](#)

### Functions

- [template<class Field >](#)  
void [pfspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- [template<class Field >](#)  
void [pfspmv\\_task](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [index\\_t](#) iStart, const [index\\_t](#) iStop, [FieldCategories::UnparametricTag](#))
- [template<class Field >](#)  
void [pfspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
- [template<class Field >](#)  
void [pfspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [int64\\_t](#) kmax)
- [template<class Field >](#)  
void [pfspmv\\_one](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- [template<class Field >](#)  
void [pfspmv\\_mone](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- [template<class Field >](#)  
void [pfspmv\\_one](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
- [template<class Field >](#)  
void [pfspmv\\_mone](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))

### 13.141.1 Macro Definition Documentation

#### 13.141.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL
```

## 13.142 csr\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)



- namespace [FFLAS::sparse\\_details\\_impl](#)

## Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL`

## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t blockSize, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

## 13.142.1 Macro Definition Documentation

### 13.142.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

## 13.143 csr\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_spmv\\_INL](#)

### Functions

- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [int64\\_t](#) kmax)
- `template<class Field >`  
void [fspmv\\_one](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- `template<class Field >`  
void [fspmv\\_mone](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- `template<class Field >`  
void [fspmv\\_one](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
- `template<class Field >`  
void [fspmv\\_mone](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))

### 13.143.1 Macro Definition Documentation

#### 13.143.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

## 13.144 csr\_utils.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Functions

- `template<class Field >`  
void [sparse\\_delete](#) (const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A)
- `template<class Field >`  
void [sparse\\_delete](#) (const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > &A)
- `template<class Field >`  
std::ostream & [sparse\\_print](#) (std::ostream &os, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::CSR](#) > &A)
- `template<class IndexT >`  
void [sparse\\_init](#) (const Givaro::Modular< Givaro::Integer > &F, [Sparse](#)< Givaro::Modular< Givaro::Integer >, [SparseMatrix\\_t::CSR](#) > &A, const IndexT \*row, const IndexT \*col, Givaro::Integer \*dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

- `template<class IndexT >`  
`void sparse_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 13.145 csr\_hyb.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmmm.inl"
```

### Data Structures

- struct `Sparse< _Field, SparseMatrix_t::CSR_HYB >`

### Namespaces

- namespace `FFLAS`

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 13.146 csr\_hyb\_pspmm.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::sparse_details_impl`

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL`

## Functions

- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`

## 13.146.1 Macro Definition Documentation

### 13.146.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL
```

## 13.147 csr\_hyb\_pspmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_pspmv\\_INL](#)

## Functions

- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`

## 13.147.1 Macro Definition Documentation

### 13.147.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL
```

## 13.148 csr\_hyb\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`

### 13.148.1 Macro Definition Documentation

#### 13.148.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

## 13.149 csr\_hyb\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`

### 13.149.1 Macro Definition Documentation

#### 13.149.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

## 13.150 csr\_hyb\_utils.inl File Reference

### Data Structures

- struct [Info](#)
- struct [Coo< ValT, IdxT >](#)

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::csr\\_hyb\\_details](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_utils\\_INL](#)

### Functions

- `template<class Field >`  
void [sparse\\_delete](#) (const [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A)
- `template<class Field , class IndexT >`  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

### 13.150.1 Macro Definition Documentation

#### 13.150.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL
```

## 13.151 ell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmv.inl"
```

### Data Structures

- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_ZO >](#)

### Namespaces

- namespace [FFLAS](#)

### Functions

- `template<class Field , class IndexT >`  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse< Field, SparseMatrix\\_t::ELL >](#) &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- `template<class Field , class IndexT >`  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse< Field, SparseMatrix\\_t::ELL\\_ZO >](#) &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- `template<class Field >`  
void [sparse\\_delete](#) (const [Sparse< Field, SparseMatrix\\_t::ELL >](#) &A)
- `template<class Field >`  
void [sparse\\_delete](#) (const [Sparse< Field, SparseMatrix\\_t::ELL\\_ZO >](#) &A)

## 13.152 ell\_pspmm.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_pspmm\\_INL](#)

### Functions

- `template<class Field >`  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::GenericTag](#))
- `template<class Field >`  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [FieldCategories::GenericTag](#))
- `template<class Field >`  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::UnparametricTag](#))
- `template<class Field >`  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#))
- `template<class Field >`  
void [pfspmm](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, const [int64\\_t](#) kmax)
- `template<class Field , class Func >`  
void [pfspmm\\_zo](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, Func &&func)
- `template<class Field , class Func >`  
void [pfspmm\\_zo](#) (const [Field](#) &F, const [Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#) > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, Func &&func)

### 13.152.1 Macro Definition Documentation

#### 13.152.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL
```

## 13.153 ell\_pspmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_pspmv\\_INL](#)

## Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 13.153.1 Macro Definition Documentation

### 13.153.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL
```

## 13.154 ell\_spmv.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::sparse_details_impl`

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`



- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

### 13.154.1 Macro Definition Documentation

#### 13.154.1.1 \_\_FflasFfpack\_fflas\_sparse\_ELL\_spmv\_INL

```
#define __FflasFfpack_fflas_sparse_ELL_spmv_INL
```

## 13.155 ell\_spmv.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::sparse_details_impl`

### Macros

- `#define __FflasFfpack_fflas_sparse_ELL_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`

- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 13.155.1 Macro Definition Documentation

#### 13.155.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL
```

## 13.156 ell\_utils.inl File Reference

```
#include <vector>
```

### Namespaces

- namespace `FFLAS`

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_utils_INL`

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

### 13.156.1 Macro Definition Documentation

#### 13.156.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_utils_INL
```

## 13.157 ell\_simd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_spmv.inl"
```

## Data Structures

- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd >](#)
- struct [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#)

## Namespaces

- namespace [FFLAS](#)

## Functions

- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse< Field, SparseMatrix\\_t::ELL\\_simd >](#) &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#) &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd >](#) &A)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#) &A)

# 13.158 ell\_simd\_pspmv.inl File Reference

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_simd\\_pspmv\\_INL](#)

## Functions

- template<class [Field](#) >  
void [pfspmv](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd >](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmv](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd >](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
void [pfspmv](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd >](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [uint64\\_t](#) kmax)
- template<class [Field](#) >  
void [pfspmv\\_one](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmv\\_mone](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmv\\_one](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
void [pfspmv\\_mone](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))

## 13.158.1 Macro Definition Documentation

### 13.158.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL
```

## 13.159 ell\_simd\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_simd\\_spmv\\_INL](#)

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 13.159.1 Macro Definition Documentation

### 13.159.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL
```

## 13.160 ell\_simd\_utils.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL`

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field , class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

### 13.160.1 Macro Definition Documentation

#### 13.160.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL
```

## 13.161 hyb\_zo.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmv.inl"
```

### Data Structures

- struct `Sparse< _Field, SparseMatrix_t::HYB_ZO >`

### Namespaces

- namespace [FFLAS](#)

## 13.162 hyb\_zo\_pspmm.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL`

## Functions

- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfsppmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)`

### 13.162.1 Macro Definition Documentation

#### 13.162.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL
```

## 13.163 hyb\_zo\_pspmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL`

### Functions

- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfsppmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)`

### 13.163.1 Macro Definition Documentation

#### 13.163.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL
```

## 13.164 hyb\_zo\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL`

## Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)`

### 13.164.1 Macro Definition Documentation

#### 13.164.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

## 13.165 hyb\_zo\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL`

## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr`  
`x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr`  
`x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename Field::ConstElement_ptr`  
`x, typename Field::Element_ptr y, uint64_t kmax)`

### 13.165.1 Macro Definition Documentation

#### 13.165.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

## 13.166 hyb\_zo\_utils.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL`

## Functions

- template<class [Field](#) >  
void [sparse\\_delete](#) (const [Sparse](#)< [Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) > &A)
- template<class [Field](#) , class [IndexT](#) >  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse](#)< [Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) > &A, const [IndexT](#) \*row, const [IndexT](#) \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<typename [\\_Field](#) >  
std::ostream & [operator<<](#) (std::ostream &os, const [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) > &A)

## 13.166.1 Macro Definition Documentation

### 13.166.1.1 \_\_FflasFfpack\_fflas\_sparse\_HYB\_ZO\_utils\_INL

```
#define __FflasFfpack_fflas_sparse_HYB_ZO_utils_INL
```

## 13.167 read\_sparse.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <fstream>
#include <string>
#include <cstdlib>
#include <iterator>
```

## Data Structures

- struct [Coo](#)< [Field](#) >
- struct [readMyMachineType](#)< [Field](#), [T](#) >
- struct [readMyMachineType](#)< [Field](#), [mpz\\_t](#) >

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details\\_spmv](#)

## Macros

- #define [DNS\\_BIN\\_VER](#) 0
- #define [mask\\_t](#) [uint64\\_t](#)

## Functions

- template<class [Field](#) , bool sorted = true, bool read\_integer = false>  
void [readSmsFormat](#) (const std::string &path, const [Field](#) &f, [index\\_t](#) \*&row, [index\\_t](#) \*&col, typename [Field::Element\\_ptr](#) &val, [index\\_t](#) &rowdim, [index\\_t](#) &coldim, [uint64\\_t](#) &nnz)
- template<class [Field](#) >  
void [readSprFormat](#) (const std::string &path, const [Field](#) &f, [index\\_t](#) \*&row, [index\\_t](#) \*&col, typename [Field::Element\\_ptr](#) &val, [index\\_t](#) &rowdim, [index\\_t](#) &coldim, [uint64\\_t](#) &nnz)
- template<class [T](#) >  
std::enable\_if< std::is\_integral< [T](#) >::value, int > [getDataType](#) ()
- template<class [T](#) >  
std::enable\_if< std::is\_floating\_point< [T](#) >::value, int > [getDataType](#) ()
- template<class [T](#) >  
std::enable\_if< std::is\_same< [T](#), [mpz\\_t](#) >::value, int > [getDataType](#) ()
- template<class [T](#) >  
int [getDataType](#) ()



- template<class [Field](#) >  
void [readMachineType](#) (const [Field](#) &F, typename [Field::Element](#) &modulo, typename [Field::Element\\_ptr](#) val, std::ifstream &file, const [uint64\\_t](#) dims, const [mask\\_t](#) data\_type, const [mask\\_t](#) field\_desc)
- template<class [Field](#) >  
void [readDnsFormat](#) (const std::string &path, const [Field](#) &F, [index\\_t](#) &rowdim, [index\\_t](#) &colldim, typename [Field::Element\\_ptr](#) &val)
- template<class [Field](#) >  
void [writeDnsFormat](#) (const std::string &path, const [Field](#) &F, const [index\\_t](#) &rowdim, const [index\\_t](#) &colldim, typename [Field::Element\\_ptr](#) A, [index\\_t](#) ldA)

## 13.167.1 Macro Definition Documentation

### 13.167.1.1 DNS\_BIN\_VER

```
#define DNS_BIN_VER 0
```

### 13.167.1.2 mask\_t

```
#define mask_t uint64_t
```

## 13.168 sell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_spmv.inl"
```

### Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::SELL>](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::SELL\\_ZO>](#)

### Namespaces

- namespace [FFLAS](#)

## 13.169 sell\_pspmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_sell\\_pspmv\\_INL](#)

### Functions

- template<class [Field](#) >  
void [pfspmv](#) (const [Field](#) &F, const [Sparse<Field, SparseMatrix\\_t::SELL>](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmv](#) (const [Field](#) &F, const [Sparse<Field, SparseMatrix\\_t::SELL>](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
void [pfspmv](#) (const [Field](#) &F, const [Sparse<Field, SparseMatrix\\_t::SELL>](#) &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [int64\\_t](#) kmax)

- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 13.169.1 Macro Definition Documentation

### 13.169.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_pspmv_INL
```

## 13.170 sell\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 13.170.1 Macro Definition Documentation

#### 13.170.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL
```

## 13.171 sell\_utils.inl File Reference

### Data Structures

- struct [Info](#)
- struct [Coo< ValT, IdxT >](#)

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sell\\_details](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_sell_utils_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz, uint64_t sigma=0)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

### 13.171.1 Macro Definition Documentation

#### 13.171.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_utils_INL
```

## 13.172 `sparse_matrix_traits.h` File Reference

```
#include <type_traits>
```

### Data Structures

- struct [isSparseMatrix](#)< Field, M >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::COO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >
- struct [isZOSparseMatrix](#)< F, M >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >
- struct [isSparseMatrixSimdFormat](#)< F, M >
- struct [isSparseMatrixMKLFormat](#)< F, M >
- struct [tfn\\_plus](#)
- struct [tfn\\_mul](#)
- struct [tfn\\_mul\\_eq](#)
- struct [tfn\\_minus](#)
- struct [tfn\\_plus\\_eq](#)
- struct [tfn\\_minus\\_eq](#)
- struct [has\\_plus\\_impl](#)< C >
- struct [has\\_mul\\_impl](#)< C >
- struct [has\\_mul\\_eq\\_impl](#)< C >
- struct [has\\_plus\\_eq\\_impl](#)< C >
- struct [has\\_minus\\_eq\\_impl](#)< C >
- struct [has\\_minus\\_impl](#)< C >
- struct [has\\_operation](#)< T >

### Namespaces

- namespace [FFLAS](#)

### Typedefs

- using [ZOSparseMatrix](#) = std::true\_type
- using [NotZOSparseMatrix](#) = std::false\_type
- using [SimdSparseMatrix](#) = std::true\_type
- using [NoSimdSparseMatrix](#) = std::false\_type
- using [MKLSparseMatrixFormat](#) = std::true\_type
- using [NotMKLSparseMatrixFormat](#) = std::false\_type
- template<class T >
  - using [has\\_plus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_plus\\_impl](#)< T > >::type

- `template<class T >`  
using `has_minus` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_minus_impl< T > >::type`
- `template<class T >`  
using `has_equal` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, std::is_copyable< T > >::type`
- `template<class T >`  
using `has_plus_eq` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_plus_eq_impl< T > >::type`
- `template<class T >`  
using `has_minus_eq` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_minus_eq_impl< T > >::type`
- `template<class T >`  
using `has_mul` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_mul_impl< T > >::type`
- `template<class T >`  
using `has_mul_eq` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_mul_eq_impl< T > >::type`

## 13.173 utils.h File Reference

```
#include <algorithm>
#include <numeric>
#include <vector>
```

### Data Structures

- struct [StatsMatrix](#)

### Namespaces

- namespace [FFLAS](#)

### Functions

- `template<class It >`  
double [computeDeviation](#) (It begin, It end)
- `template<class Field >`  
[StatsMatrix](#) [getStat](#) (const [Field](#) &F, const [index\\_t](#) \*row, const [index\\_t](#) \*col, `typename Field::ConstElement_ptr` val, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

## 13.174 ffpack.doxy File Reference

## 13.175 ffpack.h File Reference

Set of elimination based routines for dense linear algebra.

```
#include "givaro/givpoly1.h"
#include <fflas-ffpack/fflas-ffpack-config.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include <list>
#include <vector>
#include <iostream>
#include <algorithm>
#include "fflas-ffpack/checkers/checkers_ffpack.h"
```

```
#include "ffpack_fgesv.inl"
#include "ffpack_fgetrs.inl"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
#include "ffpack_pluq.inl"
#include "ffpack_pluq_mp.inl"
#include "ffpack_ppluq.inl"
#include "ffpack_ludivine.inl"
#include "ffpack_ludivine_mp.inl"
#include "ffpack_echelonforms.inl"
#include "ffpack_fsytrf.inl"
#include "ffpack_invert.inl"
#include "ffpack_ftrtr.inl"
#include "ffpack_ftrstr.inl"
#include "ffpack_ftrssyr2k.inl"
#include "ffpack_charpoly_kglu.inl"
#include "ffpack_charpoly_kgfast.inl"
#include "ffpack_charpoly_kgfastgeneralized.inl"
#include "ffpack_charpoly_danilevski.inl"
#include "ffpack_charpoly.inl"
#include "ffpack_frobenius.inl"
#include "ffpack_minpoly.inl"
#include "ffpack_krylovelim.inl"
#include "ffpack_permutation.inl"
#include "ffpack_rankprofiles.inl"
#include "ffpack_det_mp.inl"
#include "ffpack.inl"
```

## Data Structures

- class [CharpolyFailed](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#) 64
- `#define` [\\_\\_FFLASFFPACK\\_FTRSSYR2K\\_THRESHOLD](#) 64

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)  
*Conversion of a permutation from LAPACK format to Math format.*
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)  
*Conversion of a permutation from Maths format to LAPACK format.*
- template<class [Field](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- template<class [Field](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans,

const size\_t m, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const FFLAS::ParSeqHelper::Sequential seq)

- template<class Field, class Cut, class Param >  
void applyP (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)
- template<class Field >  
void MonotonicApplyP (const Field &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t R)

*Apply a R-monotonically increasing permutation P, to the matrix A.*

- template<class Field >  
void fgetrs (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr B, const size\_t ldb, int \*info)

*Solve the system  $AX = B$  or  $XA = B$ .*

- template<class Field >  
Field::Element\_ptr fgetrs (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, typename Field::Element\_ptr A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename Field::Element\_ptr X, const size\_t ldx, typename Field::ConstElement\_ptr B, const size\_t ldb, int \*info)

*Solve the system  $A X = B$  or  $X A = B$ .*

- template<class Field >  
size\_t fgesv (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, int \*info)

*Square system solver.*

- template<class Field >  
size\_t fgesv (const Field &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, typename Field::ConstElement\_ptr B, const size\_t ldb, int \*info)

*Rectangular system solver.*

- template<class Field >  
void ftrtri (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG Diag, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const size\_t threshold=\_\_FFLASFFPACK\_FTRTRI\_THRESHOLD)

*Compute the inverse of a triangular matrix.*

- template<class Field >  
void trinv\_left (const Field &F, const size\_t N, typename Field::ConstElement\_ptr L, const size\_t ldl, typename Field::Element\_ptr X, const size\_t ldx)

- template<class Field >  
void ftrtrm (const Field &F, const FFLAS::FFLAS\_SIDE side, const FFLAS::FFLAS\_DIAG diag, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)

*Compute the product of two triangular matrices of opposite shape.*

- template<class Field >  
void ftrstr (const Field &F, const FFLAS::FFLAS\_SIDE side, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diagA, const FFLAS::FFLAS\_DIAG diagB, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, const size\_t threshold=\_\_FFLASFFPACK\_FTRSTR\_THRESHOLD)

*Solve a triangular system with a triangular right hand side of the same shape.*

- template<class Field >  
void ftrssyr2k (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diagA, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, const size\_t threshold=\_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD)

*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*

- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`  
*Compute a PLUQ factorization of the given matrix.*
- `template<class Field >`  
`size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut, Param > &PHelper)`
- `template<class Field >`  
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE_THRESHOLD)`  
*Compute the CUP or PLE factorization of the given matrix.*
- `template<class Field >`  
`size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t idx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK_MINPOLY_TAG MinTag=FpackDense, const size_t kg_mc=0, const size_t kg_mb=0, const size_t kg_j=0)`
- `template<class Field >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Compute the Column Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)`
- `template<class Field >`  
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr`



A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG LuTag=FFpackSlabRecursive)

*Compute the Row Echelon form of the input matrix in-place.*

- template<class Field >  
size\_t pRowEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=FFpackTileRecursive)
- template<class Field , class PSHelper >  
size\_t RowEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class Field >  
size\_t ReducedColumnEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG LuTag=FFpackSlabRecursive)

*Compute the Reduced Column Echelon form of the input matrix in-place.*

- template<class Field >  
size\_t pReducedColumnEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=FFpackTileRecursive)
- template<class Field , class PSHelper >  
size\_t ReducedColumnEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class Field >  
size\_t ReducedRowEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG LuTag=FFpackSlabRecursive)

*Compute the Reduced Row Echelon form of the input matrix in-place.*

- template<class Field >  
size\_t pReducedRowEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=FFpackTileRecursive)
- template<class Field , class PSHelper >  
size\_t ReducedRowEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class Field >  
size\_t GaussJordan (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag)

*Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.*

- template<class Field >  
Field::Element\_ptr Invert (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, int &>nullity)

*Invert the given matrix in place or computes its nullity if it is singular.*

- template<class Field >  
Field::Element\_ptr Invert (const Field &F, const size\_t M, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, int &>nullity)

*Invert the given matrix or computes its nullity if it is singular.*

- template<class Field >  
Field::Element\_ptr Invert2 (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, int &>nullity)

*Invert the given matrix or computes its nullity if it is singular.*

- `template<class PolRing >`  
`std::list< typename PolRing::Element > & CharPoly (const PolRing &R, std::list< typename PolRing::Element > &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class PolRing >`  
`PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class PolRing >`  
`PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)`
- `template<class Field , class Polynomial >`  
`int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`  
`void fgemv_kgf (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)`
- `template<class Field , class Polynomial , class RandIter >`  
`std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr U, const size_t ldu, RandIter &G)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)`
- `template<class PolRing >`  
`void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`
- `template<class PolRing >`  
`std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, const size_t degree)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field , class Polynomial >`  
`Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda)`  
*Compute the minimal polynomial of the matrix A.*
- `template<class Field , class Polynomial , class RandIter >`  
`Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, RandIter &G)`  
*Compute the minimal polynomial of the matrix A.*

- template<class [Field](#) , class Polynomial >  
Polynomial & [MatVecMinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) v, const size\_t incv)  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- template<class [Field](#) , class Polynomial >  
Polynomial & [MatVecMinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) v, const size\_t incv, typename [Field::Element\\_ptr](#) K, const size\_t ldk, size\_t \*P)  
size\_t [Rank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Computes the rank of the given matrix using a PLUQ factorization.*
- template<class [Field](#) >  
size\_t [pRank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t numthreads=0)
- template<class [Field](#) , class PSHelper >  
size\_t [Rank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const PSHelper &psH)
- template<class [Field](#) >  
bool [IsSingular](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Returns true if the given matrix is singular.*
- template<class [Field](#) >  
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P=NULL, size\_t \*Q=NULL)  
*Returns the determinant of the given square matrix.*
- template<class [Field](#) >  
[Field::Element](#) & [pDet](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t numthreads=0, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class [Field](#) , class PSHelper >  
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const PSHelper &psH, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [Solve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb)  
*Solves a linear system  $AX = b$  using PLUQ factorization.*
- template<class [Field](#) , class PSHelper >  
[Field::Element\\_ptr](#) [Solve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb, PSHelper &psH)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [pSolve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb, size\_t numthreads=0)
- template<class [Field](#) >  
\*void [RandomNullSpaceVector](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
*Solve  $LX = B$  or  $XL = B$  in place.*
- template<class [Field](#) >  
size\_t [NullSpaceBasis](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &NS, size\_t &ldn, size\_t &← t &NSdim)

*Computes a basis of the Left/Right nullspace of the matrix A.*

- template<class [Field](#) >  
size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))

*Computes the row rank profile of A.*

- template<class [Field](#) >  
size\_t [pRowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >  
size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- template<class [Field](#) >  
size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))

*Computes the column rank profile of A.*

- template<class [Field](#) >  
size\_t [pColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >  
size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- void [RankProfileFromLU](#) (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag)

*Recovers the column/row rank profile from the permutation of an LU decomposition.*

- size\_t [LeadingSubmatrixRankProfiles](#) (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)

*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*

- template<class [Field](#) >  
size\_t [RowRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rowindices, size\_t \*colindices, size\_t &R)

*RowRankProfileSubmatrixIndices.*

- template<class [Field](#) >  
size\_t [ColRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rowindices, size\_t \*colindices, size\_t &R)

*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*

- template<class [Field](#) >  
size\_t [RowRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)

*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*

- template<class [Field](#) >  
size\_t [ColRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)

*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*

- template<class [Field](#) >  
void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false)

*Extracts a triangular matrix from a compact storage  $A=L|U$  of rank R.*

- template<class [Field](#) >  
void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda)

*Cleans up a compact storage  $A=L|U$  to reveal a triangular matrix of rank R.*

- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .*
- `template<class Field >`  
`void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- `template<class Field >`  
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- `template<class Field >`  
`Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X, const size_t ldx)`  
*LQUPtoInverseOfFullRankMinor.*

## 13.175.1 Detailed Description

Set of elimination based routines for dense linear algebra.

Matrices are supposed over finite prime field of characteristic less than  $2^{26}$ .

## 13.175.2 Macro Definition Documentation

### 13.175.2.1 \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD

```
#define __FFLASFFPACK_FTRSTR_THRESHOLD 64
```

### 13.175.2.2 \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD

```
#define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64
```

## 13.176 ffpack.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_INL](#)

### Functions

- template<class [Field](#) >  
size\_t [Rank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Computes the rank of the given matrix using a PLUQ factorization.*
- template<class [Field](#) >  
size\_t [pRank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t numthreads=0)
- template<class [Field](#) , class PSHelper >  
size\_t [Rank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const PSHelper &psH)
- template<class [Field](#) >  
bool [IsSingular](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Returns true if the given matrix is singular.*
- template<class [Field](#) >  
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P=NULL, size\_t \*Q=NULL)  
*Returns the determinant of the given square matrix.*
- template<class [Field](#) >  
[Field::Element](#) & [pDet](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t numthreads=0, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class [Field](#) , class PSHelper >  
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const PSHelper &psH, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [Solve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb)  
*Solves a linear system  $AX = b$  using PLUQ factorization.*
- template<class [Field](#) , class PSHelper >  
[Field::Element\\_ptr](#) [Solve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb, PSHelper &psH)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [pSolve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb, size\_t numthreads=0)
- template<class [Field](#) >  
void [RandomNullSpaceVector](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
*Solve  $LX = B$  or  $XL = B$  in place.*



- template<class [Field](#) >  
size\_t [NullSpaceBasis](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &NS, size\_t &ldn, size\_t &NSdim)  
*Computes a basis of the Left/Right nullspace of the matrix A.*
- template<class [Field](#) >  
void [solveLB](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) L, const size\_t ldl, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb)
- template<class [Field](#) >  
void [solveB2](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) L, const size\_t ldl, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb)

## 13.176.1 Macro Definition Documentation

### 13.176.1.1 \_\_FFLASFFPACK\_ffpack\_INL

```
#define __FFLASFFPACK_ffpack_INL
```

## 13.177 ffpack\_charpoly.inl File Reference

```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "ffpack_charpoly_mp.inl"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_charpoly\\_INL](#)

### Functions

- template<class [PolRing](#) >  
std::list< typename [PolRing::Element](#) > & [CharPoly](#) (const [PolRing](#) &R, std::list< typename [PolRing::Element](#) > &charp, const size\_t N, typename [PolRing::Domain\\_t::Element\\_ptr](#) A, const size\_t lda, typename [PolRing::Domain\\_t::RandIter](#) &G, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag=FfpackAuto, const size\_t degree=[\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#))  
*Compute the characteristic polynomial of the matrix A.*
- template<class [PolRing](#) >  
[PolRing::Element](#) & [CharPoly](#) (const [PolRing](#) &R, typename [PolRing::Element](#) &charp, const size\_t N, typename [PolRing::Domain\\_t::Element\\_ptr](#) A, const size\_t lda, typename [PolRing::Domain\\_t::RandIter](#) &G, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag=FfpackAuto, const size\_t degree=[\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#))  
*Compute the characteristic polynomial of the matrix A.*
- template<class [Field](#) , class [Polynomial](#) , class [RandIter](#) >  
std::list< [Polynomial](#) > & [LUKrylov](#) (const [Field](#) &F, std::list< [Polynomial](#) > &charp, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) U, const size\_t ldu, [RandIter](#) &G)
- template<class [Field](#) , class [Polynomial](#) >  
std::list< [Polynomial](#) > & [LUKrylov\\_KGFast](#) (const [Field](#) &F, std::list< [Polynomial](#) > &charp, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx)

### 13.177.1 Macro Definition Documentation

#### 13.177.1.1 \_\_FFLASFFPACK\_charpoly\_INL

```
#define __FFLASFFPACK_charpoly_INL
```

## 13.178 ffpack\_charpoly\_danilevski.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_danilveski\\_INL](#)

### Functions

- template<class [Field](#) , class Polynomial >  
std::list< Polynomial > & [Danilevski](#) (const [Field](#) &F, std::list< Polynomial > &charp, const size\_t N, type-  
name [Field::Element\\_ptr](#) A, const size\_t lda)

### 13.178.1 Macro Definition Documentation

#### 13.178.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_danilveski_INL
```

## 13.179 ffpack\_charpoly\_kgfast.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kgfast\\_INL](#)

### Functions

- template<class [Field](#) , class Polynomial >  
int [KGFast](#) (const [Field](#) &F, std::list< Polynomial > &charp, const size\_t N, typename [Field::Element\\_ptr](#) A,  
const size\_t lda, size\_t \*kg\_mc, size\_t \*kg\_mb, size\_t \*kg\_j)
- template<class [Field](#) >  
void [fgemv\\_kgf](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, type-  
name [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY, const  
size\_t kg\_mc, const size\_t kg\_mb, const size\_t kg\_j)

### 13.179.1 Macro Definition Documentation

#### 13.179.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfast_INL
```



## 13.180 ffpack\_charpoly\_kgfastgeneralized.inl File Reference

```
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kgfastgeneralized\\_INL](#)

### Functions

- template<class [Field](#) >  
[Field::Element\\_ptr buildMatrix](#) (const [Field](#) &F, typename [Field::ConstElement\\_ptr](#) E, typename [Field::ConstElement\\_ptr](#) C, const size\_t lda, const size\_t \*B, const size\_t \*T, const size\_t me, const size\_t mc, const size\_t lambda, const size\_t mu)
- template<class [Field](#) , class Polynomial >  
std::list< Polynomial > & [KGFast\\_generalized](#) (const [Field](#) &F, std::list< Polynomial > &charp, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)

### 13.180.1 Macro Definition Documentation

#### 13.180.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL
```

## 13.181 ffpack\_charpoly\_kglu.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kglu\\_INL](#)

### Functions

- template<class [Field](#) >  
size\_t [updateD](#) (const [Field](#) &F, size\_t \*d, size\_t k, std::vector< std::vector< typename [Field::Element](#) > > &minpt)
- template<class [Field](#) >  
size\_t [newD](#) (const [Field](#) &F, size\_t \*d, bool &KeepOn, const size\_t l, const size\_t N, typename [Field::Element\\_ptr](#) X, const size\_t \*Q, std::vector< std::vector< typename [Field::Element](#) > > &minpt)
- template<class [Field](#) , class Polynomial >  
std::list< Polynomial > & [KellerGehrig](#) (const [Field](#) &F, std::list< Polynomial > &charp, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)

### 13.181.1 Macro Definition Documentation

#### 13.181.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kglu_INL
```

### 13.182 ffpack\_charpoly\_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "givaro/givpoly1.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

#### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

#### Macros

- #define [\\_\\_FFPACK\\_charpoly\\_mp\\_INL](#)

#### Functions

- [FFPACK::RNSInteger< FFPACK::rns\\_double >::Element\\_ptr CharPoly](#) (const [FFPACK::RNSInteger< FFPACK::rns\\_double > &F](#), typename [FFPACK::RNSInteger< FFPACK::rns\\_double >::Element\\_ptr](#) charp, const size\_t N, typename [FFPACK::RNSInteger< FFPACK::rns\\_double >::Element\\_ptr](#) A, const size\_t Ida, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag, size\_t degree)
- [template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly](#) (const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R, Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp, const size\_t N, Givaro::Integer \*A, const size\_t Ida, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag, size\_t degree)

### 13.182.1 Macro Definition Documentation

#### 13.182.1.1 \_\_FFPACK\_charpoly\_mp\_INL

```
#define __FFPACK_charpoly_mp_INL
```

### 13.183 ffpack\_det\_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

#### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

#### Macros

- #define [\\_\\_FFPACK\\_det\\_mp\\_INL](#)

## Functions

- `template<class PSHelper >`  
`FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (const FFPACK::RNSInteger<`  
`FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr &det,`  
`const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t Ida,`  
`const PSHelper &psH)`
- `template<class PSHelper >`  
`Givaro::Integer & Det (const Givaro::ZRing< Givaro::Integer > &F, Givaro::Integer &det, const size_t N,`  
`Givaro::Integer *A, const size_t Ida, const PSHelper &psH, size_t *P, size_t *Q)`

## 13.183.1 Macro Definition Documentation

### 13.183.1.1 \_\_FFPACK\_det\_mp\_INL

```
#define __FFPACK_det_mp_INL
```

## 13.184 ffpack\_echelonforms.inl File Reference

### Namespaces

- namespace `FFPACK`  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_echelon_forms_INL`
- `#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256`

### Functions

- `template<class Field >`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const`  
`size_t M, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t Ida, typename`  
`Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false)`  
*Extracts a triangular matrix from a compact storage A=L\U of rank R.*
- `template<class Field >`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const`  
`size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t Ida)`  
*Cleans up a compact storage A=L\U to reveal a triangular matrix of rank R.*
- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag,`  
`const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const`  
`size_t Ida, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const`  
`FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage A=L\U of rank R obtained by RowEchelonForm or Column↔EchelonForm.*
- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag,`  
`const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t`  
`Ida, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Cleans up a compact storage A=L\U obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank R.*

- `template<class Field >`  
`void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t lda, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- `template<class Field >`  
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr T, const size_t ldt, const FFPACK_LU_TAG LuTag=FfpackSlabRecursive)`  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*

### 13.184.1 Macro Definition Documentation

#### 13.184.1.1 \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL

```
#define __FFLASFFPACK_ffpack_echelon_forms_INL
```

#### 13.184.1.2 \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256
```

## 13.185 ffpack\_fgesv.inl File Reference

### Namespaces

- namespace **FFPACK**  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_fgesv_INL`

### Functions

- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, int *info)`  
*Square system solver.*
- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`

*Rectangular system solver.*

## 13.185.1 Macro Definition Documentation

### 13.185.1.1 \_\_FFLASFFPACK\_ffpack\_fgesv\_INL

```
#define __FFLASFFPACK_ffpack_fgesv_INL
```

## 13.186 ffpack\_fgetrs.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_fgetrs\\_INL](#)

### Functions

- template<class [Field](#) >  
void [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) X, const size\_t ldX, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*

## 13.186.1 Macro Definition Documentation

### 13.186.1.1 \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL

```
#define __FFLASFFPACK_ffpack_fgetrs_INL
```

## 13.187 ffpack\_frobenius.inl File Reference

```
#include <givaro/givranditer.h>
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

### Functions

- template<class [Field](#) >  
void [CompressRows](#) ([Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t nb\_blocs)
- template<class [Field](#) >  
void [CompressRowsQK](#) ([Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t deg, const size\_t nb\_blocs)

- `template<class Field >`  
`void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRowsQK (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQA (Field &F, const size_t M, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRowsQA (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class PolRing >`  
`void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t Ida, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`
- `template<class PolRing >`  
`std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t Ida, const size_t degree)`

## 13.188 ffpack\_fsytrf.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- namespace **FFPACK**  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_fsytrf_INL`

### Functions

- `template<class Field >`  
`bool fsytrf_BC_Crout (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr Dinv, const size_t incDinv)`
- `template<class Field >`  
`size_t fsytrf_BC_RL (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr Dinv, const size_t incDinv)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_RL (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_LOW_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM (const Field &Fi, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P, size_t BCThreshold)`

- template<class [Field](#) >  
bool [fsytrf\\_nonunit](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) DinV, const size\_t incDinv, [FFLAS::ParSeqHelper::Sequential](#) seq, size\_t threshold)
- template<class [Field](#) , class Cut , class Param >  
bool [fsytrf\\_nonunit](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) DinV, const size\_t incDinv, [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > par, size\_t threshold)
- template<class [Field](#) >  
bool [fsytrf](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#))  
*Triangular factorization of symmetric matrices.*
- template<class [Field](#) >  
bool [fsytrf](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const [FFLAS::ParSeqHelper::Sequential](#) seq, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#))
- template<class [Field](#) , class Cut , class Param >  
bool [fsytrf](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > par, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#))
- template<class [Field](#) >  
size\_t [fsytrf\\_RPM](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t threshold)
- template<class [Field](#) >  
void [getTridiagonal](#) (const [Field](#) &F, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, size\_t \*P, typename [Field::Element\\_ptr](#) T, const size\_t ldt)

## 13.188.1 Macro Definition Documentation

### 13.188.1.1 [\\_\\_FFLASFFPACK\\_ffpack\\_fsytrf\\_INL](#)

```
#define __FFLASFFPACK_ffpack_fsytrf_INL
```

## 13.189 ffpack\_ftrssyr2k.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrssyr2k\\_INL](#)

### Functions

- template<class [Field](#) >  
void [ftrssyr2k](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diagA, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRSSYR2K\\_THRESHOLD](#))  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*

## 13.189.1 Macro Definition Documentation

### 13.189.1.1 [\\_\\_FFLASFFPACK\\_ffpack\\_ftrssyr2k\\_INL](#)

```
#define __FFLASFFPACK_ffpack_ftrssyr2k_INL
```

## 13.190 ffpack\_ftrstr.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrstr\\_INL](#)

### Functions

- template<class [Field](#) >  
void [ftrstr](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diagA, const [FFLAS::FFLAS\\_DIAG](#) diagB, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#))  
*Solve a triangular system with a triangular right hand side of the same shape.*

## 13.190.1 Macro Definition Documentation

### 13.190.1.1 \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL

```
#define __FFLASFFPACK_ffpack_ftrstr_INL
```

## 13.191 ffpack\_ftrtr.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1
- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrtr\\_INL](#)

### Functions

- template<class [Field](#) >  
void [ftrtri](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#))  
*Compute the inverse of a triangular matrix.*
- template<class [Field](#) >  
void [ftrtrm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Compute the product of two triangular matrices of opposite shape.*
- template<class [Field](#) >  
void [trinv\\_left](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) L, const size\_t ldl, typename [Field::Element\\_ptr](#) X, const size\_t idx)



### 13.191.1 Macro Definition Documentation

#### 13.191.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

#### 13.191.1.2 \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL

```
#define __FFLASFFPACK_ffpack_ftrtr_INL
```

## 13.192 ffpack\_invert.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_invert\\_INL](#)

### Functions

- template<class [Field](#) >  
[Field::Element\\_ptr Invert](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, int &>nullity)  
*Invert the given matrix in place or computes its nullity if it is singular.*
- template<class [Field](#) >  
[Field::Element\\_ptr Invert](#) (const [Field](#) &F, const size\_t M, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &>nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class [Field](#) >  
[Field::Element\\_ptr Invert2](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &>nullity)  
*Invert the given matrix or computes its nullity if it is singular.*

### 13.192.1 Macro Definition Documentation

#### 13.192.1.1 \_\_FFLASFFPACK\_ffpack\_invert\_INL

```
#define __FFLASFFPACK_ffpack_invert_INL
```

## 13.193 ffpack\_krylovelim.inl File Reference

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_krylovelim\\_INL](#)

### 13.193.1 Macro Definition Documentation

#### 13.193.1.1 \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL

```
#define __FFLASFFPACK_ffpack_krylovelim_INL
```

## 13.194 ffpack\_ludivine.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

## Data Structures

- class [callLUdivine\\_small< Element >](#)
- class [callLUdivine\\_small< double >](#)
- class [callLUdivine\\_small< float >](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_ffpack\\_ludivine\\_INL](#)

## Functions

- `template<class Field >`  
size\_t [LUdivine\\_gauss](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
- `template<class Field >`  
size\_t [LUdivine\\_small](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
- `template<class Field >`  
size\_t [LUdivine](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag, const size\_t cutoff)
- `template<class Field >`  
size\_t [LUdivine\\_construct](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, typename [Field::Element\\_ptr](#) u, const size\_t incu, size\_t \*P, bool computeX, const [FFPACK::FFPACK\\_MINPOLY\\_TAG](#) MinTag, const size\_t kg\_mc, const size\_t kg\_mb, const size\_t kg\_j)

## 13.194.1 Macro Definition Documentation

### 13.194.1.1 [\\_\\_FFLASFFPACK\\_ffpack\\_ludivine\\_INL](#)

```
#define __FFLASFFPACK_ffpack_ludivine_INL
```

## 13.195 [ffpack\\_ludivine\\_mp.inl](#) File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack_ludivine.inl"
```

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

**Macros**

- `#define __FFPACK_ludivine_mp_INL`

**Functions**

- `template<> size_t LUdivine` (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS\_DIAG Diag, const FFLAS::FFLAS\_TRANSPOSE trans, const size\_t M, const size\_t N, typename Givaro::Integer \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag, const size\_t cutoff)

**13.195.1 Macro Definition Documentation****13.195.1.1 \_\_FFPACK\_ludivine\_mp\_INL**

```
#define __FFPACK_ludivine_mp_INL
```

**13.196 ffpack\_minpoly.inl File Reference****Namespaces**

- namespace `FFPACK`  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace `FFPACK::Protected`

**Macros**

- `#define __FFLASFFPACK_ffpack_minpoly_INL`

**Functions**

- `template<class Field , class Polynomial >`  
Polynomial & `MinPoly` (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda)  
*Compute the minimal polynomial of the matrix A.*
- `template<class Field , class Polynomial , class RandIter >`  
Polynomial & `MinPoly` (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, RandIter &G)  
*Compute the minimal polynomial of the matrix A.*
- `template<class Field , class Polynomial >`  
Polynomial & `MatVecMinPoly` (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr v, const size\_t incv)  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- `template<class Field , class Polynomial >`  
Polynomial & `MatVecMinPoly` (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr v, const size\_t incv, typename Field::Element\_ptr K, const size\_t ldk, size\_t \*P)
- `template<class Field , class Polynomial >`  
Polynomial & `Hybrid_KGF_LUK_MinPoly` (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr X, const size\_t ldx, size\_t \*P, const FFPACK\_MINPOLY\_TAG MinTag=FFPACK::FfpackDense, const size\_t kg\_mc=0, const size\_t kg\_↔ mb=0, const size\_t kg\_j=0)

**13.196.1 Macro Definition Documentation****13.196.1.1 \_\_FFLASFFPACK\_ffpack\_minpoly\_INL**

```
#define __FFLASFFPACK_ffpack_minpoly_INL
```

## 13.197 ffpack\_permutation.inl File Reference

```
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_permutation\\_INL](#)
- #define [FFLASFFPACK\\_PERM\\_BKSIZE](#) 32

### Functions

- template<class [Field](#) >  
void [MonotonicApplyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t R)  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- template<class [Field](#) >  
void [MonotonicCompress](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t maxpiv, const size\_t rowstomove, const std::vector< bool > &ispiv)
- template<class [Field](#) >  
void [MonotonicCompressMorePivots](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t rowstomove, const size\_t lenP)
- template<class [Field](#) >  
void [MonotonicCompressCycles](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t lenP)
- template<class [Field](#) >  
void [MonotonicExpand](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t maxpiv, const size\_t rowstomove, const std::vector< bool > &ispiv)
- template<class [Field](#) >  
void [applyP\\_block](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P)
- template<class [Field](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const [FFLAS::ParSeqHelper::Sequential](#) seq)
- template<class [Field](#) >  
void [doApplyS](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- template<class [Field](#) >  
void [MatrixApplyS](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- template<class [Field](#) >  
void [MatrixApplyS](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, const [FFLAS::ParSeqHelper::Sequential](#) seq)

- `template<class Field , class Cut , class Param >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field , class Cut , class Param >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)`  
*Conversion of a permutation from LAPACK format to Math format.*
- `void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)`  
*Conversion of a permutation from Maths format to LAPACK format.*
- `void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag} (I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- `void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag} (I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $MathP$  as a MathPermutation format.*
- `void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $MathP1 \times \text{Diag} (I_R, P2)$  where  $MathP1$  is a MathPermutation and  $P2$  a LAPACK permutation and store the result in  $MathP1$  as a MathPermutation format.*
- `void cyclic_shift_mathPerm (size_t *P, const size_t s)`
- `template<class Field >`  
`void cyclic_shift_row_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`void cyclic_shift_row (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<typename T >`  
`void cyclic_shift_row (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`void cyclic_shift_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<typename T >`  
`void cyclic_shift_col (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)`

Computes  $P1 \times Diag(L_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.

- `template<class Field, class Cut, class Param >`  
`void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans,`  
`const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const`  
`size_t *P, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`

## 13.197.1 Macro Definition Documentation

### 13.197.1.1 \_\_FFLASFFPACK\_ffpack\_permutation\_INL

```
#define __FFLASFFPACK_ffpack_permutation_INL
```

### 13.197.1.2 FFLASFFPACK\_PERM\_BKSIZE

```
#define FFLASFFPACK_PERM_BKSIZE 32
```

## 13.198 ffpack\_pluq.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

### Macros

- `#define __FFLASFFPACK_ffpack_pluq_INL`
- `#define CROUT`

### Functions

- `template<class Field >`  
`size_t PLUQ_basecaseV3 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t`  
`N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ_basecaseV2 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t`  
`N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ_basecaseCROUT (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t`  
`*N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t _PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, type-`  
`name Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential`  
`&PSHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`

*Compute a PLUQ factorization of the given matrix.*

## 13.198.1 Macro Definition Documentation

### 13.198.1.1 \_\_FFLASFFPACK\_ffpack\_pluq\_INL

```
#define __FFLASFFPACK_ffpack_pluq_INL
```

### 13.198.1.2 CROUT

```
#define CROUT
```

## 13.199 ffpack\_pluq\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "givaro/givinteger.h"
#include "givaro/modular-integer.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFPACK\\_pluq\\_mp\\_INL](#)

### Functions

- template<class Cut, class Param >  
size\_t [PLUQ](#) (const Givaro::Modular< Givaro::Integer > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename Givaro::Integer \*A, const size\_t lda, size\_t \*P, size\_t \*Q, size\_t BCThreshold, [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > &PSHelper)

## 13.199.1 Macro Definition Documentation

### 13.199.1.1 \_\_FFPACK\_pluq\_mp\_INL

```
#define __FFPACK_pluq_mp_INL
```

## 13.200 ffpack\_ppluq.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ppluq\\_INL](#)
- #define [\\_\\_FFLAS\\_TRSM\\_READONLY](#)
- #define [PBASECASE\\_K](#) 256

### Functions

- template<class Field >  
void [threads\\_fgemm](#) (const size\_t m, const size\_t n, const size\_t r, int nbthreads, size\_t \*W1, size\_t \*W2, size\_t \*W3, size\_t gamma)
- template<class Field >  
void [threads\\_frsm](#) (const size\_t m, const size\_t n, int nbthreads, size\_t \*t1, size\_t \*t2)
- template<class Field >  
size\_t [PLUQ](#) (const Field &Fi, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, type-name [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFLAS::ParSeqHelper::Parallel](#)< [FFLAS::CuttingStrategy::Recursive](#), [FFLAS::StrategyParameter::Threads](#) > &PSHelper)

- template<class [Field](#) >  
size\_t [pPLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q)

### 13.200.1 Macro Definition Documentation

#### 13.200.1.1 `__FFLASFFPACK_ffpack_ppluq_INL`

```
#define __FFLASFFPACK_ffpack_ppluq_INL
```

#### 13.200.1.2 `__FFLAS__TRSM_READONLY`

```
#define __FFLAS__TRSM_READONLY
```

#### 13.200.1.3 `PBASECASE_K`

```
#define PBASECASE_K 256
```

## 13.201 `ffpack_rankprofiles.inl` File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define `__FFLASFFPACK_ffpack_rank_profiles_INL`

### Functions

- template<class [Field](#) >  
size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const [FFPACK\\_LU\\_TAG](#) LuTag=[FpackSlabRecursive](#))  
*Computes the row rank profile of A.*
- template<class [Field](#) >  
size\_t [pRowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const [FFPACK\\_LU\\_TAG](#) LuTag=[FpackTileRecursive](#))
- template<class [Field](#) , class [PSHelper](#) >  
size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const [FFPACK\\_LU\\_TAG](#) LuTag, [PSHelper](#) &psH)
- template<class [Field](#) >  
size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const [FFPACK\\_LU\\_TAG](#) LuTag=[FpackSlabRecursive](#))  
*Computes the column rank profile of A.*
- template<class [Field](#) >  
size\_t [pColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const [FFPACK\\_LU\\_TAG](#) LuTag=[FpackTileRecursive](#))
- template<class [Field](#) , class [PSHelper](#) >  
size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const [FFPACK\\_LU\\_TAG](#) LuTag, [PSHelper](#) &psH)
- void [RankProfileFromLU](#) (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const [FFPACK\\_LU\\_TAG](#) LuTag)  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- size\_t [LeadingSubmatrixRankProfiles](#) (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*



- template<class [Field](#) >  
size\_t [RowRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*RowRankProfileSubmatrixIndices.*
- template<class [Field](#) >  
size\_t [ColRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*Computes the indices of the submatrix  $r \times r$   $X$  of  $A$  whose columns correspond to the column rank profile of  $A$ .*
- template<class [Field](#) >  
size\_t [RowRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Computes the  $r \times r$  submatrix  $X$  of  $A$ , by picking the row rank profile rows of  $A$ .*
- template<class [Field](#) >  
size\_t [ColRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Compute the  $r \times r$  submatrix  $X$  of  $A$ , by picking the row rank profile rows of  $A$ .*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [LQUPtoInverseOfFullRankMinor](#) (const [Field](#) &F, const size\_t rank, typename [Field::Element\\_ptr](#) A\_factors, const size\_t lda, const size\_t \*QtPointer, typename [Field::Element\\_ptr](#) X, const size\_t ldx)  
*LQUPtoInverseOfFullRankMinor.*

### 13.201.1 Macro Definition Documentation

#### 13.201.1.1 \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL

```
#define __FFLASFFPACK_ffpack_rank_profiles_INL
```

## 13.202 field-traits.h File Reference

Field Traits.

```
#include <type_traits>
#include "fflas-ffpack/field/rns-double-elt.h"
#include "recint/rmint.h"
#include "givaro/modular-general.h"
#include "givaro/zring.h"
```

### Data Structures

- struct [GenericTag](#)  
*generic ring.*
- struct [ModularTag](#)  
*This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`*
- struct [UnparametricTag](#)  
*If the field uses a representation with infix operators.*
- struct [DefaultTag](#)  
*No specific mode of action: use standard field operations.*
- struct [DefaultBoundedTag](#)  
*Use standard field operations, but keeps track of bounds on input and output.*
- struct [ConvertTo< T >](#)  
*Force conversion to appropriate element type of `ElementCategory T`.*
- struct [DelayedTag](#)  
*Performs field operations with delayed mod reductions. Ensures result is reduced.*

- struct [LazyTag](#)  
*Performs field operations with delayed mod only when necessary. Result may not be reduced.*
- struct [GenericTag](#)  
*default is generic*
- struct [MachineFloatTag](#)  
*float or double*
- struct [MachineIntTag](#)  
*short, int, long, long long, and unsigned variants*
- struct [FixedPrecIntTag](#)  
*Fixed precision integers above machine precision: Givaro::reclnt.*
- struct [ArbitraryPrecIntTag](#)  
*Arbitrary precision integers: GMP.*
- struct [RNSElementTag](#)  
*Representation in a Residue Number System.*
- struct [ElementTraits< Element >](#)  
*ElementTraits.*
- struct [ElementTraits< float >](#)
- struct [ElementTraits< double >](#)
- struct [ElementTraits< int8\\_t >](#)
- struct [ElementTraits< int16\\_t >](#)
- struct [ElementTraits< int32\\_t >](#)
- struct [ElementTraits< int64\\_t >](#)
- struct [ElementTraits< uint8\\_t >](#)
- struct [ElementTraits< uint16\\_t >](#)
- struct [ElementTraits< uint32\\_t >](#)
- struct [ElementTraits< uint64\\_t >](#)
- struct [ElementTraits< Givaro::Integer >](#)
- struct [ElementTraits< Reclnt::rint< K > >](#)
- struct [ElementTraits< Reclnt::ruint< K > >](#)
- struct [ElementTraits< Reclnt::rmint< K, MG > >](#)
- struct [ElementTraits< FFPACK::rns\\_double\\_elt >](#)
- struct [ModeTraits< Field >](#)  
*ModeTraits.*
- struct [ModeTraits< Givaro::Modular< Element, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< int8\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< int16\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< int32\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< uint8\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< uint16\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< uint32\\_t, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >](#)
- struct [ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< Element > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< int8\\_t > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< int16\\_t > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< int32\\_t > >](#)
- struct [ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >](#)
- struct [ModeTraits< Givaro::ZRing< Givaro::Integer > >](#)
- struct [ModeTraits< Givaro::ZRing< float > >](#)
- struct [ModeTraits< Givaro::ZRing< double > >](#)
- struct [ModeTraits< Givaro::Montgomery< T > >](#)
- struct [FieldTraits< Field >](#)  
*FieldTrait.*

- struct [FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > >](#)
- struct [FieldTraits< Givaro::Modular< Element > >](#)
- struct [FieldTraits< Givaro::ModularBalanced< Element > >](#)
- struct [FieldTraits< Givaro::ZRing< double > >](#)
- struct [FieldTraits< Givaro::ZRing< float > >](#)
- struct [FieldTraits< Givaro::ZRing< int16\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< uint16\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< int32\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< uint32\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< int64\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< uint64\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< Givaro::Integer > >](#)
- struct [FieldTraits< FFPACK::RNSInteger< T > >](#)
- struct [FieldTraits< FFPACK::RNSIntegerMod< T > >](#)
- struct [associatedDelayedField< Field >](#)
- struct [associatedDelayedField< const Givaro::Modular< T, X > >](#)
- struct [associatedDelayedField< const Givaro::ModularBalanced< T > >](#)
- struct [associatedDelayedField< const Givaro::ZRing< T > >](#)
- struct [associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >](#)

### Namespaces

- namespace [Reclnt](#)
- namespace [Givaro](#)
- namespace [FFPACK](#)
  - Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFLAS](#)
- namespace [FFLAS::FieldCategories](#)
  - Traits and categories will need to be placed in a proper file later.*
- namespace [FFLAS::ModeCategories](#)
  - Specifies the mode of action for an algorithm w.r.t.*
- namespace [FFLAS::ElementCategories](#)

### 13.202.1 Detailed Description

Field Traits.

## 13.203 field.dox File Reference

### 13.204 rns-double-elt.h File Reference

rns elt structure with double support

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/cast.h"
```

### Data Structures

- struct [rns\\_double\\_elt](#)
- struct [rns\\_double\\_elt\\_ptr](#)
- struct [rns\\_double\\_elt\\_cstptr](#)

### Namespaces

- namespace [FFPACK](#)
  - Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Functions

- template<> [rns\\_double\\_elt\\_ptr fflas\\_const\\_cast \(rns\\_double\\_elt\\_cstptr x\)](#)
- template<> [rns\\_double\\_elt\\_cstptr fflas\\_const\\_cast \(rns\\_double\\_elt\\_ptr x\)](#)

### 13.204.1 Detailed Description

rns elt structure with double support

## 13.205 rns-double-recint.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_field\\_rns\\_double\\_recint\\_INL](#)

### 13.205.1 Macro Definition Documentation

#### 13.205.1.1 \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL

```
#define __FFLASFFPACK_field_rns_double_recint_INL
```

## 13.206 rns-double.h File Reference

rns structure with double support

```
#include <iterator>
#include <vector>
#include <givaro/modular-floating.h>
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include "givaro/modular-extended.h"
#include <recint/ruint.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/field/rns-double-elt.h"
#include "rns-double.inl"
#include "rns-double-recint.inl"
```

## Data Structures

- struct [rns\\_double](#)
- struct [rns\\_double\\_extended](#)
- class [rnsRandIter< RNS >](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFLAS](#)

**Macros**

- `#define ROUND_DOWN(x, s) ((x) & ~((s)-1))`

**Functions**

- `template<> void fflas_delete (FFPACK::rns_double_elt_ptr A)`
- `template<> void fflas_delete (FFPACK::rns_double_elt_cstptr A)`

**13.206.1 Detailed Description**

rns structure with double support

**13.206.2 Macro Definition Documentation****13.206.2.1 ROUND\_DOWN**

```
#define ROUND_DOWN(
 x,
 s) ((x) & ~((s)-1))
```

**13.207 rns-double.inl File Reference**

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

**Namespaces**

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

**Macros**

- `#define __FFLASFFPACK_field_rns_double_INL`

**13.207.1 Macro Definition Documentation****13.207.1.1 \_\_FFLASFFPACK\_field\_rns\_double\_INL**

```
#define __FFLASFFPACK_field_rns_double_INL
```

**13.208 rns-integer-mod.h File Reference**

representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)

```
#include <vector>
#include <cmath>
#include <recint/recint.h>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include <givaro/udl.h>
#include "givaro/modular-extended.h"
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas/fflas_level1.inl"
#include "fflas-ffpack/fflas/fflas_level2.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/fflas/fflas_fscal_mp.inl"
```

## Data Structures

- class [RNSIntegerMod< RNS >](#)
- class [RNSIntegerMod< RNS >::RandIter](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFLAS](#)

## Functions

- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`  
`void finit_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`  
`void finit_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`  
`void fconvert_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`
- `template<typename RNS >`  
`void fconvert_trans_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`

### 13.208.1 Detailed Description

representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)

## 13.209 rns-integer.h File Reference

representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)

```
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-double.h"
```

## Data Structures

- class [RNSInteger< RNS >](#)
- class [RNSInteger< RNS >::RandIter](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFLAS](#)

## Functions

- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)`

- template<typename [RNS](#) >  
void [finit\\_rns](#) (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const size\_t m, const size\_t n, size\_t k, const Givaro::Integer \*B, const size\_t ldb, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr A)
- template<typename [RNS](#) >  
void [fconvert\\_rns](#) (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const size\_t m, const size\_t n, Givaro::Integer alpha, Givaro::Integer \*B, const size\_t ldb, typename [FFPACK::RNSInteger](#)< [RNS](#) >::ConstElement\_ptr A)

### 13.209.1 Detailed Description

representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)

## 13.210 rns.h File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## 13.211 rns.inl File Reference

```
#include "rns-double.h"
#include "rns-integer.h"
#include "rns-integer-mod.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_field\\_rns\\_INL](#)

### 13.211.1 Macro Definition Documentation

#### 13.211.1.1 [\\_\\_FFLASFFPACK\\_field\\_rns\\_INL](#)

```
#define __FFLASFFPACK_field_rns_INL
```

## 13.212 interfaces.doxy File Reference

## 13.213 fflas\_c.h File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>
```

### Macros

- #define [FFLAS\\_COMPILED](#)

### Enumerations

- enum [FFLAS\\_C\\_ORDER](#) { [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 }  
*Storage by row or col ?*
- enum [FFLAS\\_C\\_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }  
*Is matrix transposed ?*
- enum [FFLAS\\_C\\_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 }  
*Is triangular matrix's shape upper ?*

- enum `FFLAS_C_DIAG` { `FflasNonUnit` = 131 , `FflasUnit` = 132 }  
*Is the triangular matrix implicitly unit diagonal ?*
- enum `FFLAS_C_SIDE` { `FflasLeft` = 141 , `FflasRight` = 142 }  
*On what side ?*
- enum `FFLAS_C_BASE` { `FflasDouble` = 151 , `FflasFloat` = 152 , `FflasGeneric` = 153 }  
*FFLAS\_C\_BASE determines the type of the element representation for Matrix Mult kernel.*

## Functions

- void `freducein_1_modular_double` (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void `freduce_1_modular_double` (const double F, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void `fnegin_1_modular_double` (const double F, const size\_t n, double \*X, const size\_t incX, bool positive)
- void `fneg_1_modular_double` (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void `fzero_1_modular_double` (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- bool `fiszero_1_modular_double` (const double p, const size\_t n, const double \*X, const size\_t incX, bool positive)
- bool `fequal_1_modular_double` (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void `fassign_1_modular_double` (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void `fscalin_1_modular_double` (const double p, const size\_t n, const double alpha, double \*X, const size\_t incX, bool positive)
- void `fscal_1_modular_double` (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void `faxy_1_modular_double` (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- double `fdot_1_modular_double` (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void `fswap_1_modular_double` (const double p, const size\_t n, double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void `fadd_1_modular_double` (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void `fsub_1_modular_double` (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void `faddin_1_modular_double` (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void `fsubin_1_modular_double` (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void `fassign_2_modular_double` (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*A, const size\_t incA, bool positive)
- void `fzero_2_modular_double` (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- bool `fequal_2_modular_double` (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, bool positive)
- bool `fiszero_2_modular_double` (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, bool positive)
- void `fidentity_2_modular_double` (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, const double d, bool positive)
- void `freducein_2_modular_double` (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- void `freduce_2_modular_double` (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, double \*A, const size\_t incA, bool positive)
- void `fnegin_2_modular_double` (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)



- void [fneg\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldB, double \*A, const size\_t ldA, bool positive)
- void [fscaln\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t ldA, bool positive)
- void [fscal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- void [faxpy\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*X, const size\_t ldX, double \*Y, const size\_t ldY, bool positive)
- void [fmove\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- void [fadd\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, double \*C, const size\_t ldC, bool positive)
- void [fsub\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, double \*C, const size\_t ldC, bool positive)
- void [fsubin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldB, double \*C, const size\_t ldC, bool positive)
- void [faddin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldB, double \*C, const size\_t ldC, bool positive)
- double \* [fgemv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double \*X, const size\_t incX, const double betA, double \*Y, const size\_t incY, bool positive)
- void [fger\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*x, const size\_t incX, const double \*y, const size\_t incY, double \*A, const size\_t ldA, bool positive)
- void [ftrsv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t n, const double \*A, const size\_t ldA, double \*X, int incX, bool positive)
- void [ftrsm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- void [ftrmm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- double \* [fgemm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_TRANSPOSE](#) tB, const size\_t m, const size\_t n, const size\_t k, const double alpha, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, const double betA, double \*C, const size\_t ldC, bool positive)
- double \* [fsquare\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double betA, double \*C, const size\_t ldC, bool positive)

## 13.213.1 Macro Definition Documentation

### 13.213.1.1 FFLAS\_COMPILED

```
#define FFLAS_COMPILED
```

## 13.213.2 Enumeration Type Documentation

### 13.213.2.1 FFLAS\_C\_ORDER

```
enum FFLAS_C_ORDER
```

Storage by row or col ?

Enumerator

|                               |           |
|-------------------------------|-----------|
| <a href="#">FflasRowMajor</a> | row major |
| <a href="#">FflasColMajor</a> | col major |

**13.213.2.2 FFLAS\_C\_TRANSPOSE**enum [FFLAS\\_C\\_TRANSPOSE](#)

Is matrix transposed ?

**Enumerator**

|              |                           |
|--------------|---------------------------|
| FflasNoTrans | Matrix is not transposed. |
| FflasTrans   | Matrix is transposed.     |

**13.213.2.3 FFLAS\_C\_UPLO**enum [FFLAS\\_C\\_UPLO](#)

Is triangular matrix's shape upper ?

**Enumerator**

|            |                                                                        |
|------------|------------------------------------------------------------------------|
| FflasUpper | Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ ) |
| FflasLower | Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ ) |

**13.213.2.4 FFLAS\_C\_DIAG**enum [FFLAS\\_C\\_DIAG](#)

Is the triangular matrix implicitly unit diagonal ?

**Enumerator**

|              |                                                                   |
|--------------|-------------------------------------------------------------------|
| FflasNonUnit | Triangular matrix has an explicit arbitrary diagonal.             |
| FflasUnit    | Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ ) |

**13.213.2.5 FFLAS\_C\_SIDE**enum [FFLAS\\_C\\_SIDE](#)

On what side ?

**Enumerator**

|            |                                 |
|------------|---------------------------------|
| FflasLeft  | Operator applied on the left.   |
| FflasRight | Operator applied on the righth. |

**13.213.2.6 FFLAS\_C\_BASE**enum [FFLAS\\_C\\_BASE](#)

FFLAS\_C\_BASE determines the type of the element representation for Matrix Mult kernel.  
(deprecated, should not be used)

**Enumerator**

|              |                                                                            |
|--------------|----------------------------------------------------------------------------|
| FflasDouble  | to use the double precision BLAS                                           |
| FflasFloat   | to use the single precision BLAS                                           |
| FflasGeneric | for any other domain, that can not be converted to floating point integers |

### 13.213.3 Function Documentation

#### 13.213.3.1 `freducein_1_modular_double()`

```
void frducein_1_modular_double (
 const double p,
 const size_t n,
 double * X,
 const size_t incX,
 bool positive)
```

#### 13.213.3.2 `freduce_1_modular_double()`

```
void frduce_1_modular_double (
 const double F,
 const size_t n,
 const double * Y,
 const size_t incY,
 double * X,
 const size_t incX,
 bool positive)
```

#### 13.213.3.3 `fnegin_1_modular_double()`

```
void fnegin_1_modular_double (
 const double F,
 const size_t n,
 double * X,
 const size_t incX,
 bool positive)
```

#### 13.213.3.4 `fneg_1_modular_double()`

```
void fneg_1_modular_double (
 const double p,
 const size_t n,
 const double * Y,
 const size_t incY,
 double * X,
 const size_t incX,
 bool positive)
```

#### 13.213.3.5 `fzero_1_modular_double()`

```
void fzero_1_modular_double (
 const double p,
 const size_t n,
 double * X,
 const size_t incX,
 bool positive)
```

#### 13.213.3.6 `fiszero_1_modular_double()`

```
bool fiszero_1_modular_double (
 const double p,
 const size_t n,
 const double * X,
 const size_t incX,
 bool positive)
```

**13.213.3.7 fequal\_1\_modular\_double()**

```
bool fequal_1_modular_double (
 const double p,
 const size_t n,
 const double * X,
 const size_t incX,
 const double * Y,
 const size_t incY,
 bool positive)
```

**13.213.3.8 fassign\_1\_modular\_double()**

```
void fassign_1_modular_double (
 const double p,
 const size_t n,
 const double * Y,
 const size_t incY,
 double * X,
 const size_t incX,
 bool positive)
```

**13.213.3.9 fscal\_1\_modular\_double()**

```
void fscal_1_modular_double (
 const double p,
 const size_t n,
 const double alpha,
 double * X,
 const size_t incX,
 bool positive)
```

**13.213.3.10 fscale\_1\_modular\_double()**

```
void fscale_1_modular_double (
 const double p,
 const size_t n,
 const double alpha,
 const double * X,
 const size_t incX,
 double * Y,
 const size_t incY,
 bool positive)
```

**13.213.3.11 faxpy\_1\_modular\_double()**

```
void faxpy_1_modular_double (
 const double p,
 const size_t n,
 const double alpha,
 const double * X,
 const size_t incX,
 double * Y,
 const size_t incY,
 bool positive)
```

**13.213.3.12 fdot\_1\_modular\_double()**

```
double fdot_1_modular_double (
 const double p,
```

```
 const size_t n,
 const double * X,
 const size_t incX,
 const double * Y,
 const size_t incY,
 bool positive)
```

#### 13.213.3.13 fswap\_1\_modular\_double()

```
void fswap_1_modular_double (
 const double p,
 const size_t n,
 double * X,
 const size_t incX,
 double * Y,
 const size_t incY,
 bool positive)
```

#### 13.213.3.14 fadd\_1\_modular\_double()

```
void fadd_1_modular_double (
 const double p,
 const size_t n,
 const double * A,
 const size_t incA,
 const double * B,
 const size_t incB,
 double * C,
 const size_t incC,
 bool positive)
```

#### 13.213.3.15 fsub\_1\_modular\_double()

```
void fsub_1_modular_double (
 const double p,
 const size_t n,
 const double * A,
 const size_t incA,
 const double * B,
 const size_t incB,
 double * C,
 const size_t incC,
 bool positive)
```

#### 13.213.3.16 faddin\_1\_modular\_double()

```
void faddin_1_modular_double (
 const double p,
 const size_t n,
 const double * B,
 const size_t incB,
 double * C,
 const size_t incC,
 bool positive)
```

#### 13.213.3.17 fsubin\_1\_modular\_double()

```
void fsubin_1_modular_double (
 const double p,
```

```

 const size_t n,
 const double * B,
 const size_t incB,
 double * C,
 const size_t incC,
 bool positive)

```

#### 13.213.3.18 fassign\_2\_modular\_double()

```

void fassign_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * B,
 const size_t ldB,
 double * A,
 const size_t ldA,
 bool positive)

```

#### 13.213.3.19 fzero\_2\_modular\_double()

```

void fzero_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t ldA,
 bool positive)

```

#### 13.213.3.20 fequal\_2\_modular\_double()

```

bool fequal_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t ldA,
 const double * B,
 const size_t ldB,
 bool positive)

```

#### 13.213.3.21 fiszero\_2\_modular\_double()

```

bool fiszero_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t ldA,
 bool positive)

```

#### 13.213.3.22 fidentity\_2\_modular\_double()

```

void fidentity_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t ldA,

```

```
 const double d,
 bool positive)
```

#### 13.213.3.23 freducein\_2\_modular\_double()

```
void freducein_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t ldA,
 bool positive)
```

#### 13.213.3.24 freduce\_2\_modular\_double()

```
void freduce_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * B,
 const size_t ldB,
 double * A,
 const size_t ldA,
 bool positive)
```

#### 13.213.3.25 fnegin\_2\_modular\_double()

```
void fnegin_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t ldA,
 bool positive)
```

#### 13.213.3.26 fneg\_2\_modular\_double()

```
void fneg_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * B,
 const size_t ldB,
 double * A,
 const size_t ldA,
 bool positive)
```

#### 13.213.3.27 fscaln\_2\_modular\_double()

```
void fscaln_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double alpha,
 double * A,
 const size_t ldA,
 bool positive)
```

**13.213.3.28 fscal\_2\_modular\_double()**

```
void fscal_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * A,
 const size_t ldA,
 double * B,
 const size_t ldB,
 bool positive)
```

**13.213.3.29 faxpy\_2\_modular\_double()**

```
void faxpy_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * X,
 const size_t ldX,
 double * Y,
 const size_t ldY,
 bool positive)
```

**13.213.3.30 fmove\_2\_modular\_double()**

```
void fmove_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t ldA,
 double * B,
 const size_t ldB,
 bool positive)
```

**13.213.3.31 fadd\_2\_modular\_double()**

```
void fadd_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t ldA,
 const double * B,
 const size_t ldB,
 double * C,
 const size_t ldC,
 bool positive)
```

**13.213.3.32 fsub\_2\_modular\_double()**

```
void fsub_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
```



```
 const size_t ldA,
 const double * B,
 const size_t ldB,
 double * C,
 const size_t ldC,
 bool positive)
```

#### 13.213.3.33 fsubin\_2\_modular\_double()

```
void fsubin_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * B,
 const size_t ldB,
 double * C,
 const size_t ldC,
 bool positive)
```

#### 13.213.3.34 faddin\_2\_modular\_double()

```
void faddin_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * B,
 const size_t ldB,
 double * C,
 const size_t ldC,
 bool positive)
```

#### 13.213.3.35 fgemv\_2\_modular\_double()

```
double * fgemv_2_modular_double (
 const double p,
 const enum FFLAS_C_TRANSPOSE TransA,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * A,
 const size_t ldA,
 const double * X,
 const size_t incX,
 const double betaA,
 double * Y,
 const size_t incY,
 bool positive)
```

#### 13.213.3.36 fger\_2\_modular\_double()

```
void fger_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * x,
 const size_t incX,
 const double * y,
 const size_t incY,
```

```
double * A,
const size_t ldA,
bool positive)
```

### 13.213.3.37 ftrsv\_2\_modular\_double()

```
void ftrsv_2_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_TRANSPOSE TransA,
 const enum FFLAS_C_DIAG Diag,
 const size_t n,
 const double * A,
 const size_t ldA,
 double * X,
 int incX,
 bool positive)
```

### 13.213.3.38 ftrsm\_3\_modular\_double()

```
void ftrsm_3_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_TRANSPOSE TransA,
 const enum FFLAS_C_DIAG Diag,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * A,
 const size_t ldA,
 double * B,
 const size_t ldB,
 bool positive)
```

### 13.213.3.39 ftrmm\_3\_modular\_double()

```
void ftrmm_3_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_TRANSPOSE TransA,
 const enum FFLAS_C_DIAG Diag,
 const size_t m,
 const size_t n,
 const double alpha,
 double * A,
 const size_t ldA,
 double * B,
 const size_t ldB,
 bool positive)
```

### 13.213.3.40 fgemm\_3\_modular\_double()

```
double * fgemm_3_modular_double (
 const double p,
 const enum FFLAS_C_TRANSPOSE tA,
 const enum FFLAS_C_TRANSPOSE tB,
 const size_t m,
```

```

 const size_t n,
 const size_t k,
 const double alpha,
 const double * A,
 const size_t ldA,
 const double * B,
 const size_t ldB,
 const double betA,
 double * C,
 const size_t ldC,
 bool positive)

```

#### 13.213.3.41 fsquare\_3\_modular\_double()

```

double * fsquare_3_modular_double (
 const double p,
 const enum FFLAS_C_TRANSPOSE tA,
 const size_t n,
 const double alpha,
 const double * A,
 const size_t ldA,
 const double betA,
 double * C,
 const size_t ldC,
 bool positive)

```

## 13.214 fflas\_L1\_inst.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"

```

### Macros

- `#define __FFLAS_L1_INST_C`
- `#define INST_OR_DECL`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

### 13.214.1 Macro Definition Documentation

#### 13.214.1.1 \_\_FFLAS\_L1\_INST\_C

```
#define __FFLAS_L1_INST_C
```

#### 13.214.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL
```

**13.214.1.3 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**13.214.1.4 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**13.214.1.5 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**13.214.1.6 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**13.214.1.7 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**13.214.1.8 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**13.214.1.9 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**13.214.1.10 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**13.215 fflas\_L1\_inst.h File Reference**

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"
```

**Macros**

- #define [INST\\_OR\\_DECL](#) <>
- #define [FFLAS\\_FIELD](#) Givaro::ModularBalanced
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) int64\_t
- #define [FFLAS\\_FIELD](#) Givaro::Modular
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) int64\_t

**13.215.1 Macro Definition Documentation****13.215.1.1 INST\_OR\_DECL**

```
#define INST_OR_DECL <>
```

**13.215.1.2 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**13.215.1.3 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**13.215.1.4 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**13.215.1.5 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**13.215.1.6 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**13.215.1.7 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**13.215.1.8 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**13.215.1.9 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**13.216 fflas\_L1\_inst\_implem.inl File Reference****Namespaces**

- namespace [FFLAS](#)

**Functions**

- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{freduce } x \leftarrow x \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{freduce } x \leftarrow y \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [finit](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{finit } x \leftarrow y \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [fconvert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX, const [FFLAS\\_ELT](#) \*Y, const size\_t incY)  

$$\text{fconvert } x \leftarrow y \bmod F.$$
- template [INST\\_OR\\_DECL](#) void [fnegin](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{fnegin } x \leftarrow -x.$$
- template [INST\\_OR\\_DECL](#) void [fneg](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$\text{fneg } x \leftarrow -y.$$

- template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, `FFLAS_ELT` \*X, const size\_t incX)  
 $fzero : A \leftarrow 0.$
- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*X, const size\_t incX)  
 $fiszero : test\ X = 0.$
- template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)  
 $fequal : test\ X = Y.$
- template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*Y, const size\_t incY, `FFLAS_ELT` \*X, const size\_t incX)  
 $fassign : x \leftarrow y.$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*X, const size\_t incX)  
 $fscal\ x \leftarrow \alpha \cdot x.$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  
 $fscal\ y \leftarrow \alpha \cdot x.$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  
 $faxpy : y \leftarrow \alpha \cdot x + y.$
- template `INST_OR_DECL` `FFLAS_ELT` `fdot` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)  
 $faxpy : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template `INST_OR_DECL` void `fswap` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)  
 $fswap : X \leftrightarrow Y.$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)

## 13.217 fflas\_L2\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

### Macros

- #define `__FFLAS_L2_INST_C`
- #define `INST_OR_DECL`
- #define `FFLAS_FIELD` Givaro::ModularBalanced

- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT int64_t`
- #define `FFLAS_FIELD` Givaro::Modular
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT int64_t`

## 13.217.1 Macro Definition Documentation

### 13.217.1.1 `__FFLAS_L2_INST_C`

```
#define __FFLAS_L2_INST_C
```

### 13.217.1.2 `INST_OR_DECL`

```
#define INST_OR_DECL
```

### 13.217.1.3 `FFLAS_FIELD` [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 13.217.1.4 `FFLAS_ELT` [1/6]

```
#define FFLAS_ELT double
```

### 13.217.1.5 `FFLAS_ELT` [2/6]

```
#define FFLAS_ELT float
```

### 13.217.1.6 `FFLAS_ELT` [3/6]

```
#define FFLAS_ELT int64_t
```

### 13.217.1.7 `FFLAS_FIELD` [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 13.217.1.8 `FFLAS_ELT` [4/6]

```
#define FFLAS_ELT double
```

### 13.217.1.9 `FFLAS_ELT` [5/6]

```
#define FFLAS_ELT float
```

### 13.217.1.10 `FFLAS_ELT` [6/6]

```
#define FFLAS_ELT int64_t
```

## 13.218 fflas\_L2\_inst.h File Reference

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

## Macros

- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 13.218.1 Macro Definition Documentation

### 13.218.1.1 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

### 13.218.1.2 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 13.218.1.3 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 13.218.1.4 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 13.218.1.5 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 13.218.1.6 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 13.218.1.7 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 13.218.1.8 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 13.218.1.9 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 13.219 fflas\_L2\_inst\_implem.inl File Reference

### Namespaces

- namespace `FFLAS`



## Functions

- template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $fassign : A \leftarrow B.$
- template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $fzero : A \leftarrow 0.$
- template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb)  
 $fequal : test A = B.$
- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*A, const size\_t lda)  
 $fiszero : test A = 0.$
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` &d)  
 $creates\ a\ diagonal\ matrix$
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $creates\ a\ diagonal\ matrix$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $freduce\ A \leftarrow A mod F.$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $freduce\ A \leftarrow B mod F.$
- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $finit\ A \leftarrow B mod F.$
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $fnegin\ A \leftarrow -A.$
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $fneg\ A \leftarrow -B.$
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*A, const size\_t lda)  
 $fscaln\ A \leftarrow a \cdot A.$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)  
 $fscal\ B \leftarrow a \cdot A.$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t ldx, `FFLAS_ELT` \*Y, const size\_t ldy)  
 $faxpy : y \leftarrow \alpha \cdot x + y.$
- template `INST_OR_DECL` void `fmove` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)  
 $faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
 $fadd : matrix\ addition.$
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)

- fsub* : matrix subtraction.
- template `INST_OR_DECL` void `fsubin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
- fsubin*  $C = C - B$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
- fadd* : matrix addition with scaling.
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
- faddin*
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` TransA, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` beta, `FFLAS_ELT` \*Y, const size\_t incY)
- finite prime* `FFLAS_FIELD`<`FFLAS_ELT`> *GEneral Matrix Vector multiplication.*
- template `INST_OR_DECL` void `fger` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*x, const size\_t incx, const `FFLAS_ELT` \*y, const size\_t incy, `FFLAS_ELT` \*A, const size\_t lda)
- fger*: rank one update of a general matrix
- template `INST_OR_DECL` void `ftsv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, int incX)
- ftsv*: *TRI*angular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

## 13.220 fflas\_L3\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L3_inst_implem.inl"
```

### Macros

- #define `__FFLAS_L3_INST_C`
- #define `INST_OR_DECL`
- #define `FFLAS_FIELD` Givaro::ModularBalanced
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t
- #define `FFLAS_FIELD` Givaro::Modular
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t

### 13.220.1 Macro Definition Documentation

#### 13.220.1.1 \_\_FFLAS\_L3\_INST\_C

```
#define __FFLAS_L3_INST_C
```

**13.220.1.2 INST\_OR\_DECL**

```
#define INST_OR_DECL
```

**13.220.1.3 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**13.220.1.4 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**13.220.1.5 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**13.220.1.6 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**13.220.1.7 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**13.220.1.8 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**13.220.1.9 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**13.220.1.10 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**13.221 fflas\_L3\_inst.h File Reference**

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L3_inst_implem.inl"
```

**Macros**

- #define [INST\\_OR\\_DECL](#) <>
- #define [FFLAS\\_FIELD](#) [Givaro::ModularBalanced](#)
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) [int64\\_t](#)
- #define [FFLAS\\_FIELD](#) [Givaro::Modular](#)
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) [int64\\_t](#)

### 13.221.1 Macro Definition Documentation

#### 13.221.1.1 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

#### 13.221.1.2 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

#### 13.221.1.3 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

#### 13.221.1.4 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

#### 13.221.1.5 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

#### 13.221.1.6 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

#### 13.221.1.7 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

#### 13.221.1.8 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

#### 13.221.1.9 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 13.222 fflas\_L3\_inst\_implem.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLAS\\_TRSM\\_READONLY](#)

### Functions

- template [INST\\_OR\\_DECL](#) void [ftrsm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
*ftrsm: **TRI**angular **S**ystem solve with **M**atrix.*
- template [INST\\_OR\\_DECL](#) void [ftrmm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
*ftrmm: **TRI**angular **M**atrix **M**ultiply.*

- template `INST_OR_DECL FFLAS_ELT * fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc)

*fgemm: Field **GE**neral **M**atrix **M**ultiply.*

- template `INST_OR_DECL FFLAS_ELT * fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Sequential` seq)
- template `INST_OR_DECL FFLAS_ELT * fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive >` par)
- template `INST_OR_DECL FFLAS_ELT * fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*B, const `size_t` ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc, const `ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads >` par)
- template `INST_OR_DECL FFLAS_ELT * fsquare` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const `size_t` ldc)

*fsquare: Squares a matrix.*

## 13.222.1 Macro Definition Documentation

### 13.222.1.1 \_\_FFLAS\_TRSM\_READONLY

```
#define __FFLAS_TRSM_READONLY
```

## 13.223 fflas\_lvl1.C File Reference

C functions calls for level 1 `FFLAS` in `fflas-c.h`.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

### Functions

- void `freducein_1_modular_double` (const double p, const `size_t` n, double \*X, const `size_t` incX, bool positive)
- void `freduce_1_modular_double` (const double p, const `size_t` n, const double \*Y, const `size_t` incY, double \*X, const `size_t` incX, bool positive)
- void `fnegin_1_modular_double` (const double p, const `size_t` n, double \*X, const `size_t` incX, bool positive)
- void `fneg_1_modular_double` (const double p, const `size_t` n, const double \*Y, const `size_t` incY, double \*X, const `size_t` incX, bool positive)
- void `fzero_1_modular_double` (const double p, const `size_t` n, double \*X, const `size_t` incX, bool positive)
- bool `fiszero_1_modular_double` (const double p, const `size_t` n, const double \*X, const `size_t` incX, bool positive)
- bool `fequal_1_modular_double` (const double p, const `size_t` n, const double \*X, const `size_t` incX, const double \*Y, const `size_t` incY, bool positive)
- void `fassign_1_modular_double` (const double p, const `size_t` n, const double \*Y, const `size_t` incY, double \*X, const `size_t` incX, bool positive)
- void `fscaln_1_modular_double` (const double p, const `size_t` n, const double alpha, double \*X, const `size_t` incX, bool positive)

- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [faxpy\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- double [fdot\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fswap\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [fadd\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsub\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [faddin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsubin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)

### 13.223.1 Detailed Description

C functions calls for level 1 [FFLAS](#) in `fflas-c.h`.

Author

Brice Boyer

See also

[fflas/fflas\\_level1.inl](#)

### 13.223.2 Function Documentation

#### 13.223.2.1 `freducein_1_modular_double()`

```
void freducein_1_modular_double (
 const double p,
 const size_t n,
 double * X,
 const size_t incX,
 bool positive)
```

#### 13.223.2.2 `freduce_1_modular_double()`

```
void freduce_1_modular_double (
 const double p,
 const size_t n,
 const double * Y,
 const size_t incY,
 double * X,
 const size_t incX,
 bool positive)
```

#### 13.223.2.3 `fnegin_1_modular_double()`

```
void fnegin_1_modular_double (
 const double p,
 const size_t n,
 double * X,
 const size_t incX,
 bool positive)
```

**13.223.2.4 fneg\_1\_modular\_double()**

```
void fneg_1_modular_double (
 const double p,
 const size_t n,
 const double * Y,
 const size_t incY,
 double * X,
 const size_t incX,
 bool positive)
```

**13.223.2.5 fzero\_1\_modular\_double()**

```
void fzero_1_modular_double (
 const double p,
 const size_t n,
 double * X,
 const size_t incX,
 bool positive)
```

**13.223.2.6 fiszero\_1\_modular\_double()**

```
bool fiszero_1_modular_double (
 const double p,
 const size_t n,
 const double * X,
 const size_t incX,
 bool positive)
```

**13.223.2.7 fequal\_1\_modular\_double()**

```
bool fequal_1_modular_double (
 const double p,
 const size_t n,
 const double * X,
 const size_t incX,
 const double * Y,
 const size_t incY,
 bool positive)
```

**13.223.2.8 fassign\_1\_modular\_double()**

```
void fassign_1_modular_double (
 const double p,
 const size_t n,
 const double * Y,
 const size_t incY,
 double * X,
 const size_t incX,
 bool positive)
```

**13.223.2.9 fscaln\_1\_modular\_double()**

```
void fscaln_1_modular_double (
 const double p,
 const size_t n,
 const double alpha,
 double * X,
 const size_t incX,
 bool positive)
```

**13.223.2.10 fscal\_1\_modular\_double()**

```
void fscal_1_modular_double (
 const double p,
 const size_t n,
 const double alpha,
 const double * X,
 const size_t incX,
 double * Y,
 const size_t incY,
 bool positive)
```

**13.223.2.11 faxpy\_1\_modular\_double()**

```
void faxpy_1_modular_double (
 const double p,
 const size_t n,
 const double alpha,
 const double * X,
 const size_t incX,
 double * Y,
 const size_t incY,
 bool positive)
```

**13.223.2.12 fdot\_1\_modular\_double()**

```
double fdot_1_modular_double (
 const double p,
 const size_t n,
 const double * X,
 const size_t incX,
 const double * Y,
 const size_t incY,
 bool positive)
```

**13.223.2.13 fswap\_1\_modular\_double()**

```
void fswap_1_modular_double (
 const double p,
 const size_t n,
 double * X,
 const size_t incX,
 double * Y,
 const size_t incY,
 bool positive)
```

**13.223.2.14 fadd\_1\_modular\_double()**

```
void fadd_1_modular_double (
 const double p,
 const size_t n,
 const double * A,
 const size_t incA,
 const double * B,
 const size_t incB,
 double * C,
 const size_t incC,
 bool positive)
```



**13.223.2.15 fsub\_1\_modular\_double()**

```
void fsub_1_modular_double (
 const double p,
 const size_t n,
 const double * A,
 const size_t incA,
 const double * B,
 const size_t incB,
 double * C,
 const size_t incC,
 bool positive)
```

**13.223.2.16 faddin\_1\_modular\_double()**

```
void faddin_1_modular_double (
 const double p,
 const size_t n,
 const double * B,
 const size_t incB,
 double * C,
 const size_t incC,
 bool positive)
```

**13.223.2.17 fsubin\_1\_modular\_double()**

```
void fsubin_1_modular_double (
 const double p,
 const size_t n,
 const double * B,
 const size_t incB,
 double * C,
 const size_t incC,
 bool positive)
```

**13.224 fflas\_lvl2.C File Reference**

C functions calls for level 2 [FFLAS](#) in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

**Functions**

- void [fassign\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fzero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- bool [fequal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, bool positive)
- bool [fiszero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, bool positive)
- void [fidentity\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, const double d, bool positive)
- void [freducein\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)

- void [freduce\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fnegin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [fneg\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fscalin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t lda, bool positive)
- void [fscal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [faxpy\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fmove\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fadd\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsub\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsubin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [faddin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- double \* [fgemv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double \*X, const size\_t incX, const double beta, double \*Y, const size\_t incY, bool positive)
- void [fger\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, double \*A, const size\_t lda, bool positive)
- void [ftrsv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t n, const double \*A, const size\_t lda, double \*X, int incX, bool positive)

### 13.224.1 Detailed Description

C functions calls for level 2 [FFLAS](#) in `fflas-c.h`.

Author

Brice Boyer

See also

[fflas/fflas\\_level2.inl](#)

### 13.224.2 Function Documentation

#### 13.224.2.1 `fassign_2_modular_double()`

```
void fassign_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t lda,
 double * B,
 const size_t ldb,
 bool positive)
```

**13.224.2.2 fzero\_2\_modular\_double()**

```
void fzero_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t lda,
 bool positive)
```

**13.224.2.3 fequal\_2\_modular\_double()**

```
bool fequal_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t lda,
 const double * B,
 const size_t ldb,
 bool positive)
```

**13.224.2.4 fiszero\_2\_modular\_double()**

```
bool fiszero_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t lda,
 bool positive)
```

**13.224.2.5 fidentity\_2\_modular\_double()**

```
void fidentity_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t lda,
 const double d,
 bool positive)
```

**13.224.2.6 freducein\_2\_modular\_double()**

```
void freducein_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t lda,
 bool positive)
```

**13.224.2.7 freduce\_2\_modular\_double()**

```
void freduce_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
```

```
 const double * A,
 const size_t lda,
 double * B,
 const size_t ldb,
 bool positive)
```

#### 13.224.2.8 fnegin\_2\_modular\_double()

```
void fnegin_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t lda,
 bool positive)
```

#### 13.224.2.9 fneg\_2\_modular\_double()

```
void fneg_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t lda,
 double * B,
 const size_t ldb,
 bool positive)
```

#### 13.224.2.10 fscaln\_2\_modular\_double()

```
void fscaln_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double alpha,
 double * A,
 const size_t lda,
 bool positive)
```

#### 13.224.2.11 fscal\_2\_modular\_double()

```
void fscal_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * A,
 const size_t lda,
 double * B,
 const size_t ldb,
 bool positive)
```

#### 13.224.2.12 faxpy\_2\_modular\_double()

```
void faxpy_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double alpha,
```

```
 const double * A,
 const size_t lda,
 double * B,
 const size_t ldb,
 bool positive)
```

#### 13.224.2.13 fmove\_2\_modular\_double()

```
void fmove_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 double * A,
 const size_t lda,
 double * B,
 const size_t ldb,
 bool positive)
```

#### 13.224.2.14 fadd\_2\_modular\_double()

```
void fadd_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t lda,
 const double * B,
 const size_t ldb,
 double * C,
 const size_t ldc,
 bool positive)
```

#### 13.224.2.15 fsub\_2\_modular\_double()

```
void fsub_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * A,
 const size_t lda,
 const double * B,
 const size_t ldb,
 double * C,
 const size_t ldc,
 bool positive)
```

#### 13.224.2.16 fsubin\_2\_modular\_double()

```
void fsubin_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * B,
 const size_t ldb,
 double * C,
 const size_t ldc,
 bool positive)
```

**13.224.2.17 faddin\_2\_modular\_double()**

```
void faddin_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double * B,
 const size_t ldb,
 double * C,
 const size_t ldc,
 bool positive)
```

**13.224.2.18 fgemv\_2\_modular\_double()**

```
double * fgemv_2_modular_double (
 const double p,
 const enum FFLAS_C_TRANSPOSE TransA,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * A,
 const size_t lda,
 const double * X,
 const size_t incX,
 const double beta,
 double * Y,
 const size_t incY,
 bool positive)
```

**13.224.2.19 fger\_2\_modular\_double()**

```
void fger_2_modular_double (
 const double p,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * X,
 const size_t incX,
 const double * Y,
 const size_t incY,
 double * A,
 const size_t lda,
 bool positive)
```

**13.224.2.20 ftrsv\_2\_modular\_double()**

```
void ftrsv_2_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_TRANSPOSE TransA,
 const enum FFLAS_C_DIAG Diag,
 const size_t n,
 const double * A,
 const size_t lda,
 double * X,
 int incX,
 bool positive)
```

## 13.225 fflas\_lvl3.C File Reference

C functions calls for level 3 [FFLAS](#) in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

### Functions

- void [ftrsm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- void [ftrmm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- double \* [fgemm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_TRANSPOSE](#) tB, const size\_t m, const size\_t n, const size\_t k, const double alpha, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, const double betA, double \*C, const size\_t ldC, bool positive)
- double \* [fsquare\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double betA, double \*C, const size\_t ldC, bool positive)

### 13.225.1 Detailed Description

C functions calls for level 3 [FFLAS](#) in fflas-c.h.

Author

Brice Boyer

See also

[fflas/fflas\\_level3.inl](#)

### 13.225.2 Function Documentation

#### 13.225.2.1 ftrsm\_3\_modular\_double()

```
void ftrsm_3_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_TRANSPOSE tA,
 const enum FFLAS_C_DIAG Diag,
 const size_t m,
 const size_t n,
 const double alpha,
 const double * A,
 const size_t ldA,
 double * B,
 const size_t ldB,
 bool positive)
```

### 13.225.2.2 ftrmm\_3\_modular\_double()

```
void ftrmm_3_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_TRANSPOSE tA,
 const enum FFLAS_C_DIAG Diag,
 const size_t m,
 const size_t n,
 const double alpha,
 double * A,
 const size_t ldA,
 double * B,
 const size_t ldB,
 bool positive)
```

### 13.225.2.3 fgemv\_3\_modular\_double()

```
double * fgemv_3_modular_double (
 const double p,
 const enum FFLAS_C_TRANSPOSE tA,
 const enum FFLAS_C_TRANSPOSE tB,
 const size_t m,
 const size_t n,
 const size_t k,
 const double alpha,
 const double * A,
 const size_t ldA,
 const double * B,
 const size_t ldB,
 const double betaA,
 double * C,
 const size_t ldC,
 bool positive)
```

### 13.225.2.4 fsquare\_3\_modular\_double()

```
double * fsquare_3_modular_double (
 const double p,
 const enum FFLAS_C_TRANSPOSE tA,
 const size_t n,
 const double alpha,
 const double * A,
 const size_t ldA,
 const double betaA,
 double * C,
 const size_t ldC,
 bool positive)
```

## 13.226 fflas\_sparse.C File Reference

C functions calls for level 1.5 and 2.5 [FFLAS](#) in fflas-c.h.

### 13.226.1 Detailed Description

C functions calls for level 1.5 and 2.5 [FFLAS](#) in fflas-c.h.



## Author

Brice Boyer

## See also

[fflas/fflas\\_sparse.h](#)

## 13.227 ffpack.C File Reference

C functions calls for [FFPACK](#) in `ffpack-c.h`.

```
#include "fflas-ffpack/interfaces/libs/ffpack_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

### Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)
- void [MatrixApplyS\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyS\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [MatrixApplyT\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyT\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- void [cyclic\\_shift\\_row\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [cyclic\\_shift\\_col\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [applyP\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const enum [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, double \*A, const size\_t lda, const size\_t \*P, bool positive)
- void [fgetrsin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*B, const size\_t ldb, int \*info, bool positive)
- double \* [fgetrsv\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*X, const size\_t idx, const double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesvin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesv\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, double \*A, const size\_t lda, double \*X, const size\_t idx, const double \*B, const size\_t ldb, int \*info, bool positive)
- void [ftrtri\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- void [trinv\\_left\\_modular\\_double](#) (const double p, const size\_t N, const double \*L, const size\_t ldl, double \*X, const size\_t idx, bool positive)
- void [ftrtrm\\_modular\\_double](#) (const double p, const [FFLAS::FFLAS\\_SIDE](#) side, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, double \*A, const size\_t lda, bool positive)



- `size_t pReducedColumnEchelonForm_modular_int32_t` (const `int32_t` p, const `size_t` M, const `size_t` N, `int32_t` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t pReducedRowEchelonForm_modular_int32_t` (const `int32_t` p, const `size_t` M, const `size_t` N, `int32_t` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `double * Invertin_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, int \*nullity, bool positive)
- `double * Invert_modular_double` (const double p, const `size_t` M, const double \*A, const `size_t` lda, double \*X, const `size_t` idx, int \*nullity, bool positive)
- `double * Invert2_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, double \*X, const `size_t` idx, int \*nullity, bool positive)
- `size_t KrylovElim_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `size_t` deg, `size_t` \*iterates, `size_t` \*inviterates, const `size_t` maxit, `size_t` virt, bool positive)
- `size_t SpecRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, const `size_t` deg, `size_t` \*rankProfile, bool positive)
- `size_t Rank_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `bool IsSingular_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `double Det_modular_double` (const double p, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `double * Solve_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, double \*x, const int incx, const double \*b, const int incb, bool positive)
- `void solveLB_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, double \*L, const `size_t` ldl, const `size_t` \*Q, double \*B, const `size_t` ldb, bool positive)
- `void solveLB2_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, double \*L, const `size_t` ldl, const `size_t` \*Q, double \*B, const `size_t` ldb, bool positive)
- `void RandomNullSpaceVector_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*X, const `size_t` incX, bool positive)
- `size_t NullSpaceBasis_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*NS, `size_t` \*ldn, `size_t` \*NSdim, bool positive)
- `size_t RowRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ColumnRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `void RankProfileFromLU` (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag)
- `size_t LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)
- `size_t RowRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t RowRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)
- `void getTriangular_modular_double` (const double p, const enum `FFLAS::FFLAS_UPLO` Uplo, const enum `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, const `size_t` R, const double \*A, const `size_t` lda, double \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, bool positive)
- `void getTriangularin_modular_double` (const double p, const enum `FFLAS::FFLAS_UPLO` Uplo, const enum `FFLAS::FFLAS_DIAG` Diag, const `size_t` M, const `size_t` N, const `size_t` R, double \*A, const `size_t` lda, bool positive)

- void [getEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [PLUQtoEchelonPermutation](#) (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)

### 13.227.1 Detailed Description

C functions calls for [FFPACK](#) in `ffpack-c.h`.

Author

Brice Boyer

See also

[ffpack/ffpack.h](#)

### 13.227.2 Function Documentation

#### 13.227.2.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
 size_t * MathP,
 const size_t * LapackP,
 const size_t N)
```

#### 13.227.2.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
 size_t * LapackP,
 const size_t * MathP,
 const size_t N)
```

#### 13.227.2.3 MatrixApplyS\_modular\_double()

```
void MatrixApplyS_modular_double (
 const double p,
 double * A,
 const size_t lda,
 const size_t width,
 const size_t M2,
 const size_t R1,
 const size_t R2,
```

```
 const size_t R3,
 const size_t R4,
 bool positive)
```

#### 13.227.2.4 PermApplyS\_double()

```
void PermApplyS_double (
 double * A,
 const size_t lda,
 const size_t width,
 const size_t M2,
 const size_t R1,
 const size_t R2,
 const size_t R3,
 const size_t R4)
```

#### 13.227.2.5 MatrixApplyT\_modular\_double()

```
void MatrixApplyT_modular_double (
 const double p,
 double * A,
 const size_t lda,
 const size_t width,
 const size_t N2,
 const size_t R1,
 const size_t R2,
 const size_t R3,
 const size_t R4,
 bool positive)
```

#### 13.227.2.6 PermApplyT\_double()

```
void PermApplyT_double (
 double * A,
 const size_t lda,
 const size_t width,
 const size_t N2,
 const size_t R1,
 const size_t R2,
 const size_t R3,
 const size_t R4)
```

#### 13.227.2.7 composePermutationsLLM()

```
void composePermutationsLLM (
 size_t * MathP,
 const size_t * P1,
 const size_t * P2,
 const size_t R,
 const size_t N)
```

#### 13.227.2.8 composePermutationsLLL()

```
void composePermutationsLLL (
 size_t * P1,
 const size_t * P2,
 const size_t R,
 const size_t N)
```

**13.227.2.9 composePermutationsMLM()**

```
void composePermutationsMLM (
 size_t * MathP1,
 const size_t * P2,
 const size_t R,
 const size_t N)
```

**13.227.2.10 cyclic\_shift\_mathPerm()**

```
void cyclic_shift_mathPerm (
 size_t * P,
 const size_t s)
```

**13.227.2.11 cyclic\_shift\_row\_modular\_double()**

```
void cyclic_shift_row_modular_double (
 const double p,
 double * A,
 size_t m,
 size_t n,
 size_t lda,
 bool positive)
```

**13.227.2.12 cyclic\_shift\_col\_modular\_double()**

```
void cyclic_shift_col_modular_double (
 const double p,
 double * A,
 size_t m,
 size_t n,
 size_t lda,
 bool positive)
```

**13.227.2.13 applyP\_modular\_double()**

```
void applyP_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const enum FFLAS::FFLAS_TRANSPOSE Trans,
 const size_t M,
 const size_t ibeg,
 const size_t iend,
 double * A,
 const size_t lda,
 const size_t * P,
 bool positive)
```

**13.227.2.14 fgetrsin\_modular\_double()**

```
void fgetrsin_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t R,
 double * A,
 const size_t lda,
 const size_t * P,
```

```
 const size_t * Q,
 double * B,
 const size_t ldb,
 int * info,
 bool positive)
```

#### 13.227.2.15 fgetrsv\_modular\_double()

```
double * fgetrsv_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t NRHS,
 const size_t R,
 double * A,
 const size_t lda,
 const size_t * P,
 const size_t * Q,
 double * X,
 const size_t ldx,
 const double * B,
 const size_t ldb,
 int * info,
 bool positive)
```

#### 13.227.2.16 fgesvin\_modular\_double()

```
size_t fgesvin_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 double * B,
 const size_t ldb,
 int * info,
 bool positive)
```

#### 13.227.2.17 fgesv\_modular\_double()

```
size_t fgesv_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t NRHS,
 double * A,
 const size_t lda,
 double * X,
 const size_t ldx,
 const double * B,
 const size_t ldb,
 int * info,
 bool positive)
```

**13.227.2.18 ftrtri\_modular\_double()**

```
void ftrtri_modular_double (
 const double p,
 const enum FFLAS::FFLAS_UPLO Uplo,
 const enum FFLAS::FFLAS_DIAG Diag,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)
```

**13.227.2.19 trinv\_left\_modular\_double()**

```
void trinv_left_modular_double (
 const double p,
 const size_t N,
 const double * L,
 const size_t ldl,
 double * X,
 const size_t ldx,
 bool positive)
```

**13.227.2.20 ftrtrm\_modular\_double()**

```
void ftrtrm_modular_double (
 const double p,
 const FFLAS::FFLAS_SIDE side,
 const enum FFLAS::FFLAS_DIAG Diag,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)
```

**13.227.2.21 PLUQ\_modular\_double()**

```
size_t PLUQ_modular_double (
 const double p,
 const enum FFLAS::FFLAS_DIAG Diag,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Q,
 bool positive)
```

**13.227.2.22 LUdivine\_modular\_double()**

```
size_t LUdivine_modular_double (
 const double p,
 const enum FFLAS::FFLAS_DIAG Diag,
 const enum FFLAS::FFLAS_TRANSPOSE Trans,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const enum FFPACK_C_LU_TAG LuTag,
```



```
 const size_t cutoff,
 bool positive)
```

#### 13.227.2.23 ColumnEchelonForm\_modular\_double()

```
size_t ColumnEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

#### 13.227.2.24 RowEchelonForm\_modular\_double()

```
size_t RowEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

#### 13.227.2.25 ReducedColumnEchelonForm\_modular\_double()

```
size_t ReducedColumnEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

#### 13.227.2.26 ReducedRowEchelonForm\_modular\_double()

```
size_t ReducedRowEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.27 ColumnEchelonForm\_modular\_float()**

```
size_t ColumnEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.28 RowEchelonForm\_modular\_float()**

```
size_t RowEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.29 ReducedColumnEchelonForm\_modular\_float()**

```
size_t ReducedColumnEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.30 ReducedRowEchelonForm\_modular\_float()**

```
size_t ReducedRowEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.31 ColumnEchelonForm\_modular\_int32\_t()**

```
size_t ColumnEchelonForm_modular_int32_t (
 const int32_t p,
```

```

 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.32 RowEchelonForm\_modular\_int32\_t()

```

size_t RowEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.33 ReducedColumnEchelonForm\_modular\_int32\_t()

```

size_t ReducedColumnEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.34 ReducedRowEchelonForm\_modular\_int32\_t()

```

size_t ReducedRowEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.35 pColumnEchelonForm\_modular\_double()

```

size_t pColumnEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,

```

```

 const size_t lda,
 size_t * P,
 size_t * Qt,
 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.36 pRowEchelonForm\_modular\_double()

```

size_t pRowEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.37 pReducedColumnEchelonForm\_modular\_double()

```

size_t pReducedColumnEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.38 pReducedRowEchelonForm\_modular\_double()

```

size_t pReducedRowEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.39 pColumnEchelonForm\_modular\_float()

```

size_t pColumnEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,

```

```

 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.40 pRowEchelonForm\_modular\_float()

```

size_t pRowEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.41 pReducedColumnEchelonForm\_modular\_float()

```

size_t pReducedColumnEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.42 pReducedRowEchelonForm\_modular\_float()

```

size_t pReducedRowEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.43 pColumnEchelonForm\_modular\_int32\_t()

```

size_t pColumnEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

**13.227.2.44 pRowEchelonForm\_modular\_int32\_t()**

```
size_t pRowEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.45 pReducedColumnEchelonForm\_modular\_int32\_t()**

```
size_t pReducedColumnEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.46 pReducedRowEchelonForm\_modular\_int32\_t()**

```
size_t pReducedRowEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.47 Invertin\_modular\_double()**

```
double * Invertin_modular_double (
 const double p,
 const size_t M,
 double * A,
 const size_t lda,
 int * nullity,
 bool positive)
```

**13.227.2.48 Invert\_modular\_double()**

```
double * Invert_modular_double (
 const double p,
 const size_t M,
 const double * A,
 const size_t lda,
```

```
double * X,
const size_t ldx,
int * nullity,
bool positive)
```

#### 13.227.2.49 Invert2\_modular\_double()

```
double * Invert2_modular_double (
 const double p,
 const size_t M,
 double * A,
 const size_t lda,
 double * X,
 const size_t ldx,
 int * nullity,
 bool positive)
```

#### 13.227.2.50 KrylovElim\_modular\_double()

```
size_t KrylovElim_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Q,
 const size_t deg,
 size_t * iterates,
 size_t * inviterates,
 const size_t maxit,
 size_t virt,
 bool positive)
```

#### 13.227.2.51 SpecRankProfile\_modular\_double()

```
size_t SpecRankProfile_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 const size_t deg,
 size_t * rankProfile,
 bool positive)
```

#### 13.227.2.52 Rank\_modular\_double()

```
size_t Rank_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)
```

#### 13.227.2.53 IsSingular\_modular\_double()

```
bool IsSingular_modular_double (

```

```

 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)

```

#### 13.227.2.54 Det\_modular\_double()

```

double Det_modular_double (
 const double p,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)

```

#### 13.227.2.55 Solve\_modular\_double()

```

double * Solve_modular_double (
 const double p,
 const size_t M,
 double * A,
 const size_t lda,
 double * x,
 const int incx,
 const double * b,
 const int incb,
 bool positive)

```

#### 13.227.2.56 solveLB\_modular\_double()

```

void solveLB_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t R,
 double * L,
 const size_t ldl,
 const size_t * Q,
 double * B,
 const size_t ldb,
 bool positive)

```

#### 13.227.2.57 solveLB2\_modular\_double()

```

void solveLB2_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t R,
 double * L,
 const size_t ldl,
 const size_t * Q,
 double * B,
 const size_t ldb,
 bool positive)

```



**13.227.2.58 RandomNullSpaceVector\_modular\_double()**

```
void RandomNullSpaceVector_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 double * X,
 const size_t incX,
 bool positive)
```

**13.227.2.59 NullSpaceBasis\_modular\_double()**

```
size_t NullSpaceBasis_modular_double (
 const double p,
 const enum FFLAS::FFLAS_SIDE Side,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 double ** NS,
 size_t * ldn,
 size_t * NSdim,
 bool positive)
```

**13.227.2.60 RowRankProfile\_modular\_double()**

```
size_t RowRankProfile_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t ** rkprofile,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.61 ColumnRankProfile\_modular\_double()**

```
size_t ColumnRankProfile_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t ** rkprofile,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.227.2.62 RankProfileFromLU()**

```
void RankProfileFromLU (
 const size_t * P,
 const size_t N,
 const size_t R,
 size_t * rkprofile,
 const enum FFPACK_C_LU_TAG LuTag)
```

**13.227.2.63 LeadingSubmatrixRankProfiles()**

```
size_t LeadingSubmatrixRankProfiles (
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t LSm,
 const size_t LSn,
 const size_t * P,
 const size_t * Q,
 size_t * RRP,
 size_t * CRP)
```

**13.227.2.64 RowRankProfileSubmatrixIndices\_modular\_double()**

```
size_t RowRankProfileSubmatrixIndices_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t ** rowindices,
 size_t ** colindices,
 size_t * R,
 bool positive)
```

**13.227.2.65 ColRankProfileSubmatrixIndices\_modular\_double()**

```
size_t ColRankProfileSubmatrixIndices_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t ** rowindices,
 size_t ** colindices,
 size_t * R,
 bool positive)
```

**13.227.2.66 RowRankProfileSubmatrix\_modular\_double()**

```
size_t RowRankProfileSubmatrix_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 double ** X,
 size_t * R,
 bool positive)
```

**13.227.2.67 ColRankProfileSubmatrix\_modular\_double()**

```
size_t ColRankProfileSubmatrix_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
```

```
double ** X,
size_t * R,
bool positive)
```

### 13.227.2.68 getTriangular\_modular\_double()

```
void getTriangular_modular_double (
 const double p,
 const enum FFLAS::FFLAS_UPLO Uplo,
 const enum FFLAS::FFLAS_DIAG Diag,
 const size_t M,
 const size_t N,
 const size_t R,
 const double * A,
 const size_t lda,
 double * T,
 const size_t ldt,
 const bool OnlyNonZeroVectors,
 bool positive)
```

### 13.227.2.69 getTriangularin\_modular\_double()

```
void getTriangularin_modular_double (
 const double p,
 const enum FFLAS::FFLAS_UPLO Uplo,
 const enum FFLAS::FFLAS_DIAG Diag,
 const size_t M,
 const size_t N,
 const size_t R,
 double * A,
 const size_t lda,
 bool positive)
```

### 13.227.2.70 getEchelonForm\_modular\_double()

```
void getEchelonForm_modular_double (
 const double p,
 const enum FFLAS::FFLAS_UPLO Uplo,
 const enum FFLAS::FFLAS_DIAG Diag,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 const double * A,
 const size_t lda,
 double * T,
 const size_t ldt,
 const bool OnlyNonZeroVectors,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.227.2.71 getEchelonFormin\_modular\_double()

```
void getEchelonFormin_modular_double (
 const double p,
 const enum FFLAS::FFLAS_UPLO Uplo,
 const enum FFLAS::FFLAS_DIAG Diag,
 const size_t M,
 const size_t N,
```

```

 const size_t R,
 const size_t * P,
 double * A,
 const size_t lda,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.72 getEchelonTransform\_modular\_double()

```

void getEchelonTransform_modular_double (
 const double p,
 const enum FFLAS::FFLAS_UPLO Uplo,
 const enum FFLAS::FFLAS_DIAG Diag,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 const size_t * Q,
 const double * A,
 const size_t lda,
 double * T,
 const size_t ldt,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.73 getReducedEchelonForm\_modular\_double()

```

void getReducedEchelonForm_modular_double (
 const double p,
 const enum FFLAS::FFLAS_UPLO Uplo,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 const double * A,
 const size_t lda,
 double * T,
 const size_t ldt,
 const bool OnlyNonZeroVectors,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.74 getReducedEchelonFormin\_modular\_double()

```

void getReducedEchelonFormin_modular_double (
 const double p,
 const enum FFLAS::FFLAS_UPLO Uplo,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 double * A,
 const size_t lda,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.227.2.75 getReducedEchelonTransform\_modular\_double()

```

void getReducedEchelonTransform_modular_double (

```

```

const double p,
const enum FFLAS::FFLAS_UPLO Uplo,
const size_t M,
const size_t N,
const size_t R,
const size_t * P,
const size_t * Q,
const double * A,
const size_t lda,
double * T,
const size_t ldt,
const enum FFPACK_C_LU_TAG LuTag,
bool positive)

```

### 13.227.2.76 PLUQtoEchelonPermutation()

```

void PLUQtoEchelonPermutation (
 const size_t N,
 const size_t R,
 const size_t * P,
 size_t * outPerm)

```

## 13.228 ffpack\_c.h File Reference

```

#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>

```

### Macros

- #define [FFPACK\\_COMPILED](#)

### Enumerations

- enum [FFLAS\\_C\\_ORDER](#) { [FflasRowMajor](#) =101 , [FflasColMajor](#) =102 }
- enum [FFLAS\\_C\\_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }
- enum [FFLAS\\_C\\_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 }
- enum [FFLAS\\_C\\_DIAG](#) { [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 }
- enum [FFLAS\\_C\\_SIDE](#) { [FflasLeft](#) = 141 , [FflasRight](#) = 142 }
- enum [FFPACK\\_C\\_LU\\_TAG](#) { [FfpackSlabRecursive](#) = 1 , [FfpackTileRecursive](#) = 2 , [FfpackSingular](#) = 3 }
- enum [FFPACK\\_C\\_CHARPOLY\\_TAG](#) {  
[FfpackLUK](#) =1 , [FfpackKG](#) =2 , [FfpackHybrid](#) =3 , [FfpackKGFast](#) =4 ,  
[FfpackDanilevski](#) =5 , [FfpackArithProg](#) =6 , [FfpackKGFastG](#) =7 }
- enum [FFPACK\\_C\\_MINPOLY\\_TAG](#) { [FfpackDense](#) =1 , [FfpackKGF](#) =2 }

### Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)
- void [MatrixApplyS\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyS\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [MatrixApplyT\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyT\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)

- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- void [cyclic\\_shift\\_row\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [cyclic\\_shift\\_col\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [applyP\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, double \*A, const size\_t lda, const size\_t \*P, bool positive)
- void [fgetrsin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*B, const size\_t ldb, int \*info, bool positive)
- double \* [fgetrs\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*X, const size\_t ldx, const double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesvin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesv\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, double \*A, const size\_t lda, double \*X, const size\_t ldx, const double \*B, const size\_t ldb, int \*info)
- void [ftrtri\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- void [trinv\\_left\\_modular\\_double](#) (const double p, const size\_t N, const double \*L, const size\_t ldl, double \*X, const size\_t ldx, bool positive)
- void [ftrtrm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- size\_t [PLUQ\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, bool positive)
- size\_t [LUdivine\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const enum [FFLAS\\_C\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, const size\_t cutoff, bool positive)
- size\_t [LUdivine\\_small\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const enum [FFLAS\\_C\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [LUdivine\\_gauss\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_float](#) (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_float](#) (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_int32\\_t](#) (const [int32\\_t](#) p, const size\_t M, const size\_t N, [int32\\_t](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_int32\\_t](#) (const [int32\\_t](#) p, const size\_t M, const size\_t N, [int32\\_t](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedColumnEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedRowEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)

- `size_t ReducedColumnEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedColumnEchelonForm_modular_int32_t` (const `int32_t` p, const `size_t` M, const `size_t` N, `int32_t` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_int32_t` (const `int32_t` p, const `size_t` M, const `size_t` N, `int32_t` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ReducedRowEchelonForm2_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, bool positive)
- `size_t REF_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, const `size_t` colbeg, const `size_t` rowbeg, const `size_t` colsize, `size_t` \*Qt, `size_t` \*P, bool positive)
- `double * Invertin_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, int \*nullity, bool positive)
- `double * Invert_modular_double` (const double p, const `size_t` M, const double \*A, const `size_t` lda, double \*X, const `size_t` idx, int \*nullity, bool positive)
- `double * Invert2_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, double \*X, const `size_t` idx, int \*nullity, bool positive)
- `size_t KrylovElim_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `size_t` deg, `size_t` \*iterates, `size_t` \*inviterates, const `size_t` maxit, `size_t` virt, bool positive)
- `size_t SpecRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, const `size_t` deg, `size_t` \*rankProfile, bool positive)
- `size_t Rank_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `bool IsSingular_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `double Det_modular_double` (const double p, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `double * Solve_modular_double` (const double p, const `size_t` M, double \*A, const `size_t` lda, double \*x, const int incx, const double \*b, const int incb, bool positive)
- `void solveLB_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, double \*L, const `size_t` ldl, const `size_t` \*Q, double \*B, const `size_t` ldb)
- `void solveLB2_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, double \*L, const `size_t` ldl, const `size_t` \*Q, double \*B, const `size_t` ldb, bool positive)
- `void RandomNullSpaceVector_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*X, const `size_t` incX, bool positive)
- `size_t NullSpaceBasis_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*NS, `size_t` \*ldn, `size_t` \*NSdim, bool positive)
- `size_t RowRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ColumnRankProfile_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `void RankProfileFromLU` (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag)
- `size_t LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)
- `size_t RowRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrixIndices_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*\*rowindices, `size_t` \*\*colindices, `size_t` \*R, bool positive)
- `size_t RowRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)
- `size_t ColRankProfileSubmatrix_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, double \*\*X, `size_t` \*R, bool positive)

- void [getTriangular\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, bool positive)
- void [getTriangularin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, bool positive)
- void [getEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonForm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonFormin\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [getReducedEchelonTransform\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- void [PLUQtoEchelonPermutation](#) (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)

## 13.228.1 Macro Definition Documentation

### 13.228.1.1 FFPACK\_COMPILED

```
#define FFPACK_COMPILED
```

## 13.228.2 Enumeration Type Documentation

### 13.228.2.1 FFLAS\_C\_ORDER

```
enum FFLAS_C_ORDER
```

Enumerator

|               |  |
|---------------|--|
| FflasRowMajor |  |
| FflasColMajor |  |

### 13.228.2.2 FFLAS\_C\_TRANSPOSE

```
enum FFLAS_C_TRANSPOSE
```

Enumerator

|              |  |
|--------------|--|
| FflasNoTrans |  |
| FflasTrans   |  |

### 13.228.2.3 FFLAS\_C\_UPLO

```
enum FFLAS_C_UPLO
```



## Enumerator

|            |  |
|------------|--|
| FflasUpper |  |
| FflasLower |  |

**13.228.2.4 FFLAS\_C\_DIAG**enum [FFLAS\\_C\\_DIAG](#)

## Enumerator

|              |  |
|--------------|--|
| FflasNonUnit |  |
| FflasUnit    |  |

**13.228.2.5 FFLAS\_C\_SIDE**enum [FFLAS\\_C\\_SIDE](#)

## Enumerator

|            |  |
|------------|--|
| FflasLeft  |  |
| FflasRight |  |

**13.228.2.6 FFPACK\_C\_LU\_TAG**enum [FFPACK\\_C\\_LU\\_TAG](#)

## Enumerator

|                     |  |
|---------------------|--|
| FfpackSlabRecursive |  |
| FfpackTileRecursive |  |
| FfpackSingular      |  |

**13.228.2.7 FFPACK\_C\_CHARPOLY\_TAG**enum [FFPACK\\_C\\_CHARPOLY\\_TAG](#)

## Enumerator

|                  |  |
|------------------|--|
| FfpackLUK        |  |
| FfpackKG         |  |
| FfpackHybrid     |  |
| FfpackKGFast     |  |
| FfpackDanilevski |  |
| FfpackArithProg  |  |
| FfpackKGFastG    |  |

**13.228.2.8 FFPACK\_C\_MINPOLY\_TAG**enum [FFPACK\\_C\\_MINPOLY\\_TAG](#)

## Enumerator

|             |  |
|-------------|--|
| FfpackDense |  |
| FfpackKGF   |  |

**13.228.3 Function Documentation****13.228.3.1 LAPACKPerm2MathPerm()**

```
void LAPACKPerm2MathPerm (
 size_t * MathP,
 const size_t * LapackP,
 const size_t N)
```

**13.228.3.2 MathPerm2LAPACKPerm()**

```
void MathPerm2LAPACKPerm (
 size_t * LapackP,
 const size_t * MathP,
 const size_t N)
```

**13.228.3.3 MatrixApplyS\_modular\_double()**

```
void MatrixApplyS_modular_double (
 const double p,
 double * A,
 const size_t lda,
 const size_t width,
 const size_t M2,
 const size_t R1,
 const size_t R2,
 const size_t R3,
 const size_t R4,
 bool positive)
```

**13.228.3.4 PermApplyS\_double()**

```
void PermApplyS_double (
 double * A,
 const size_t lda,
 const size_t width,
 const size_t M2,
 const size_t R1,
 const size_t R2,
 const size_t R3,
 const size_t R4)
```

**13.228.3.5 MatrixApplyT\_modular\_double()**

```
void MatrixApplyT_modular_double (
 const double p,
 double * A,
 const size_t lda,
 const size_t width,
 const size_t N2,
 const size_t R1,
 const size_t R2,
 const size_t R3,
```

```
 const size_t R4,
 bool positive)
```

### 13.228.3.6 PermApplyT\_double()

```
void PermApplyT_double (
 double * A,
 const size_t lda,
 const size_t width,
 const size_t N2,
 const size_t R1,
 const size_t R2,
 const size_t R3,
 const size_t R4)
```

### 13.228.3.7 composePermutationsLLM()

```
void composePermutationsLLM (
 size_t * MathP,
 const size_t * P1,
 const size_t * P2,
 const size_t R,
 const size_t N)
```

### 13.228.3.8 composePermutationsLLL()

```
void composePermutationsLLL (
 size_t * P1,
 const size_t * P2,
 const size_t R,
 const size_t N)
```

### 13.228.3.9 composePermutationsMLM()

```
void composePermutationsMLM (
 size_t * MathP1,
 const size_t * P2,
 const size_t R,
 const size_t N)
```

### 13.228.3.10 cyclic\_shift\_mathPerm()

```
void cyclic_shift_mathPerm (
 size_t * P,
 const size_t s)
```

### 13.228.3.11 cyclic\_shift\_row\_modular\_double()

```
void cyclic_shift_row_modular_double (
 const double p,
 double * A,
 size_t m,
 size_t n,
 size_t lda,
 bool positive)
```

### 13.228.3.12 cyclic\_shift\_col\_modular\_double()

```
void cyclic_shift_col_modular_double (
 const double p,
```

```
double * A,
size_t m,
size_t n,
size_t lda,
bool positive)
```

### 13.228.3.13 applyP\_modular\_double()

```
void applyP_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const enum FFLAS_C_TRANSPOSE Trans,
 const size_t M,
 const size_t ibeg,
 const size_t iend,
 double * A,
 const size_t lda,
 const size_t * P,
 bool positive)
```

### 13.228.3.14 fgetrsin\_modular\_double()

```
void fgetrsin_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t R,
 double * A,
 const size_t lda,
 const size_t * P,
 const size_t * Q,
 double * B,
 const size_t ldb,
 int * info,
 bool positive)
```

### 13.228.3.15 fgetrs\_modular\_double()

```
double * fgetrs_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t NRHS,
 const size_t R,
 double * A,
 const size_t lda,
 const size_t * P,
 const size_t * Q,
 double * X,
 const size_t ldx,
 const double * B,
 const size_t ldb,
 int * info,
 bool positive)
```

**13.228.3.16 fgesvin\_modular\_double()**

```
size_t fgesvin_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 double * B,
 const size_t ldb,
 int * info,
 bool positive)
```

**13.228.3.17 fgesv\_modular\_double()**

```
size_t fgesv_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t NRHS,
 double * A,
 const size_t lda,
 double * X,
 const size_t ldx,
 const double * B,
 const size_t ldb,
 int * info)
```

**13.228.3.18 ftrtri\_modular\_double()**

```
void ftrtri_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_DIAG Diag,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)
```

**13.228.3.19 trinv\_left\_modular\_double()**

```
void trinv_left_modular_double (
 const double p,
 const size_t N,
 const double * L,
 const size_t ldl,
 double * X,
 const size_t ldx,
 bool positive)
```

**13.228.3.20 ftrtrm\_modular\_double()**

```
void ftrtrm_modular_double (
 const double p,
 const enum FFLAS_C_DIAG diag,
 const size_t N,
 double * A,
```

```
 const size_t lda,
 bool positive)
```

### 13.228.3.21 PLUQ\_modular\_double()

```
size_t PLUQ_modular_double (
 const double p,
 const enum FFLAS_C_DIAG Diag,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Q,
 bool positive)
```

### 13.228.3.22 LUdivine\_modular\_double()

```
size_t LUdivine_modular_double (
 const double p,
 const enum FFLAS_C_DIAG Diag,
 const enum FFLAS_C_TRANSPOSE trans,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const enum FFPACK_C_LU_TAG LuTag,
 const size_t cutoff,
 bool positive)
```

### 13.228.3.23 LUdivine\_small\_modular\_double()

```
size_t LUdivine_small_modular_double (
 const double p,
 const enum FFLAS_C_DIAG Diag,
 const enum FFLAS_C_TRANSPOSE trans,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Q,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.24 LUdivine\_gauss\_modular\_double()

```
size_t LUdivine_gauss_modular_double (
 const double p,
 const enum FFLAS_C_DIAG Diag,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Q,
```

```
const enum FFPACK_C_LU_TAG LuTag,
bool positive)
```

### 13.228.3.25 ColumnEchelonForm\_modular\_double()

```
size_t ColumnEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.26 RowEchelonForm\_modular\_double()

```
size_t RowEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.27 ColumnEchelonForm\_modular\_float()

```
size_t ColumnEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.28 RowEchelonForm\_modular\_float()

```
size_t RowEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.228.3.29 ColumnEchelonForm\_modular\_int32\_t()**

```
size_t ColumnEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.228.3.30 RowEchelonForm\_modular\_int32\_t()**

```
size_t RowEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.228.3.31 ReducedColumnEchelonForm\_modular\_double()**

```
size_t ReducedColumnEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.228.3.32 ReducedRowEchelonForm\_modular\_double()**

```
size_t ReducedRowEchelonForm_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.228.3.33 ReducedColumnEchelonForm\_modular\_float()**

```
size_t ReducedColumnEchelonForm_modular_float (
 const float p,
```



```

 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.228.3.34 ReducedRowEchelonForm\_modular\_float()

```

size_t ReducedRowEchelonForm_modular_float (
 const float p,
 const size_t M,
 const size_t N,
 float * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.228.3.35 ReducedColumnEchelonForm\_modular\_int32\_t()

```

size_t ReducedColumnEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.228.3.36 ReducedRowEchelonForm\_modular\_int32\_t()

```

size_t ReducedRowEchelonForm_modular_int32_t (
 const int32_t p,
 const size_t M,
 const size_t N,
 int32_t * A,
 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)

```

#### 13.228.3.37 ReducedRowEchelonForm2\_modular\_double()

```

size_t ReducedRowEchelonForm2_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,

```

```

 const size_t lda,
 size_t * P,
 size_t * Qt,
 const bool transform,
 bool positive)

```

#### 13.228.3.38 REF\_modular\_double()

```

size_t REF_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 const size_t colbeg,
 const size_t rowbeg,
 const size_t colsize,
 size_t * Qt,
 size_t * P,
 bool positive)

```

#### 13.228.3.39 Invertin\_modular\_double()

```

double * Invertin_modular_double (
 const double p,
 const size_t M,
 double * A,
 const size_t lda,
 int * nullity,
 bool positive)

```

#### 13.228.3.40 Invert\_modular\_double()

```

double * Invert_modular_double (
 const double p,
 const size_t M,
 const double * A,
 const size_t lda,
 double * X,
 const size_t ldx,
 int * nullity,
 bool positive)

```

#### 13.228.3.41 Invert2\_modular\_double()

```

double * Invert2_modular_double (
 const double p,
 const size_t M,
 double * A,
 const size_t lda,
 double * X,
 const size_t ldx,
 int * nullity,
 bool positive)

```

#### 13.228.3.42 KrylovElim\_modular\_double()

```

size_t KrylovElim_modular_double (
 const double p,

```

```
const size_t M,
const size_t N,
double * A,
const size_t lda,
size_t * P,
size_t * Q,
const size_t deg,
size_t * iterates,
size_t * inviterates,
const size_t maxit,
size_t virt,
bool positive)
```

#### 13.228.3.43 SpecRankProfile\_modular\_double()

```
size_t SpecRankProfile_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 const size_t deg,
 size_t * rankProfile,
 bool positive)
```

#### 13.228.3.44 Rank\_modular\_double()

```
size_t Rank_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)
```

#### 13.228.3.45 IsSingular\_modular\_double()

```
bool IsSingular_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)
```

#### 13.228.3.46 Det\_modular\_double()

```
double Det_modular_double (
 const double p,
 const size_t N,
 double * A,
 const size_t lda,
 bool positive)
```

#### 13.228.3.47 Solve\_modular\_double()

```
double * Solve_modular_double (
 const double p,
 const size_t M,
```

```

double * A,
const size_t lda,
double * x,
const int incx,
const double * b,
const int incb,
bool positive)

```

#### 13.228.3.48 solveLB\_modular\_double()

```

void solveLB_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t R,
 double * L,
 const size_t ldl,
 const size_t * Q,
 double * B,
 const size_t ldb)

```

#### 13.228.3.49 solveLB2\_modular\_double()

```

void solveLB2_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const size_t M,
 const size_t N,
 const size_t R,
 double * L,
 const size_t ldl,
 const size_t * Q,
 double * B,
 const size_t ldb,
 bool positive)

```

#### 13.228.3.50 RandomNullSpaceVector\_modular\_double()

```

void RandomNullSpaceVector_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 double * X,
 const size_t incX,
 bool positive)

```

#### 13.228.3.51 NullSpaceBasis\_modular\_double()

```

size_t NullSpaceBasis_modular_double (
 const double p,
 const enum FFLAS_C_SIDE Side,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,

```

```
double ** NS,
size_t * ldn,
size_t * NSdim,
bool positive)
```

### 13.228.3.52 RowRankProfile\_modular\_double()

```
size_t RowRankProfile_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t ** rkprofile,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.53 ColumnRankProfile\_modular\_double()

```
size_t ColumnRankProfile_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t ** rkprofile,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.54 RankProfileFromLU()

```
void RankProfileFromLU (
 const size_t * P,
 const size_t N,
 const size_t R,
 size_t * rkprofile,
 const enum FFPACK_C_LU_TAG LuTag)
```

### 13.228.3.55 LeadingSubmatrixRankProfiles()

```
size_t LeadingSubmatrixRankProfiles (
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t LSm,
 const size_t LSn,
 const size_t * P,
 const size_t * Q,
 size_t * RRP,
 size_t * CRP)
```

### 13.228.3.56 RowRankProfileSubmatrixIndices\_modular\_double()

```
size_t RowRankProfileSubmatrixIndices_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
```

```

 size_t ** rowindices,
 size_t ** colindices,
 size_t * R,
 bool positive)

```

### 13.228.3.57 ColRankProfileSubmatrixIndices\_modular\_double()

```

size_t ColRankProfileSubmatrixIndices_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 size_t ** rowindices,
 size_t ** colindices,
 size_t * R,
 bool positive)

```

### 13.228.3.58 RowRankProfileSubmatrix\_modular\_double()

```

size_t RowRankProfileSubmatrix_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 double ** X,
 size_t * R,
 bool positive)

```

### 13.228.3.59 ColRankProfileSubmatrix\_modular\_double()

```

size_t ColRankProfileSubmatrix_modular_double (
 const double p,
 const size_t M,
 const size_t N,
 double * A,
 const size_t lda,
 double ** X,
 size_t * R,
 bool positive)

```

### 13.228.3.60 getTriangular\_modular\_double()

```

void getTriangular_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_DIAG diag,
 const size_t M,
 const size_t N,
 const size_t R,
 const double * A,
 const size_t lda,
 double * T,
 const size_t ldt,
 const bool OnlyNonZeroVectors,
 bool positive)

```

**13.228.3.61 getTriangularin\_modular\_double()**

```
void getTriangularin_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_DIAG diag,
 const size_t M,
 const size_t N,
 const size_t R,
 double * A,
 const size_t lda,
 bool positive)
```

**13.228.3.62 getEchelonForm\_modular\_double()**

```
void getEchelonForm_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_DIAG diag,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 const double * A,
 const size_t lda,
 double * T,
 const size_t ldt,
 const bool OnlyNonZeroVectors,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.228.3.63 getEchelonFormin\_modular\_double()**

```
void getEchelonFormin_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_DIAG diag,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 double * A,
 const size_t lda,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

**13.228.3.64 getEchelonTransform\_modular\_double()**

```
void getEchelonTransform_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const enum FFLAS_C_DIAG diag,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 const size_t * Q,
 const double * A,
 const size_t lda,
```

```
double * T,
const size_t ldt,
const enum FFPACK_C_LU_TAG LuTag,
bool positive)
```

### 13.228.3.65 getReducedEchelonForm\_modular\_double()

```
void getReducedEchelonForm_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 const double * A,
 const size_t lda,
 double * T,
 const size_t ldt,
 const bool OnlyNonZeroVectors,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.66 getReducedEchelonFormin\_modular\_double()

```
void getReducedEchelonFormin_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 double * A,
 const size_t lda,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.67 getReducedEchelonTransform\_modular\_double()

```
void getReducedEchelonTransform_modular_double (
 const double p,
 const enum FFLAS_C_UPLO Uplo,
 const size_t M,
 const size_t N,
 const size_t R,
 const size_t * P,
 const size_t * Q,
 const double * A,
 const size_t lda,
 double * T,
 const size_t ldt,
 const enum FFPACK_C_LU_TAG LuTag,
 bool positive)
```

### 13.228.3.68 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
 const size_t N,
 const size_t R,
```



```
const size_t * P,
size_t * outPerm)
```

## 13.229 ffpack\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "ffpack_inst_implem.inl"
```

### Macros

- [#define \\_\\_FFPACK\\_INST\\_C](#)
- [#define FFLAS\\_COMPILED](#)
- [#define INST\\_OR\\_DECL](#)
- [#define FFLAS\\_FIELD Givaro::ModularBalanced](#)
- [#define FFLAS\\_ELT double](#)
- [#define FFLAS\\_ELT float](#)
- [#define FFLAS\\_ELT int64\\_t](#)
- [#define FFLAS\\_FIELD Givaro::Modular](#)
- [#define FFLAS\\_ELT double](#)
- [#define FFLAS\\_ELT float](#)
- [#define FFLAS\\_ELT int64\\_t](#)

### 13.229.1 Macro Definition Documentation

#### 13.229.1.1 \_\_FFPACK\_INST\_C

```
#define __FFPACK_INST_C
```

#### 13.229.1.2 FFLAS\_COMPILED

```
#define FFLAS_COMPILED
```

#### 13.229.1.3 INST\_OR\_DECL

```
#define INST_OR_DECL
```

#### 13.229.1.4 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

#### 13.229.1.5 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

#### 13.229.1.6 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

#### 13.229.1.7 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

#### 13.229.1.8 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

**13.229.1.9 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**13.229.1.10 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**13.229.1.11 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**13.230 ffpack\_inst.h File Reference**

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "ffpack_inst_implem.inl"
```

**Macros**

- `#define FFLAS_COMPILED`
- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

**13.230.1 Macro Definition Documentation****13.230.1.1 FFLAS\_COMPILED**

```
#define FFLAS_COMPILED
```

**13.230.1.2 INST\_OR\_DECL**

```
#define INST_OR_DECL <>
```

**13.230.1.3 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**13.230.1.4 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**13.230.1.5 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**13.230.1.6 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**13.230.1.7 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**13.230.1.8 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**13.230.1.9 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**13.230.1.10 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**13.231 ffpack\_inst\_implem.inl File Reference****Namespaces**

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

**Functions**

- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $\text{MathP}$  as a  $\text{MathPermutation}$  format.*
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes  $\text{MathP1} \times \text{Diag}(I_R, P2)$  where  $\text{MathP1}$  is a  $\text{MathPermutation}$  and  $P2$  a LAPACK permutation and store the result in  $\text{MathP1}$  as a  $\text{MathPermutation}$  format.*
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- template<typename Base\_t >  
void [cyclic\\_shift\\_row\\_col](#) (Base\_t \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [cyclic\\_shift\\_row](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, [FFLAS\\_ELT](#) \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [cyclic\\_shift\\_col](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, [FFLAS\\_ELT](#) \*A, size\_t m, size\_t n, size\_t lda)
- template [INST\\_OR\\_DECL](#) void [applyP](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t \*P)
- template [INST\\_OR\\_DECL](#) void [fgetrs](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgetrs](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, [FFLAS\\_ELT](#) \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, [FFLAS\\_ELT](#) \*X, const size\_t idx, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) size\_t [fgesv](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)
- template [INST\\_OR\\_DECL](#) size\_t [fgesv](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*X, const size\_t idx, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, int \*info)

- template `INST_OR_DECL` void `fttri` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` Diag, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, const size\_t threshold)
- template `INST_OR_DECL` void `trinv_left` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*L, const size\_t ldl, `FFLAS_ELT` \*X, const size\_t ldx)
- template `INST_OR_DECL` void `fttrm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_SIDE` side, const `FFLAS::FFLAS_DIAG` diag, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda)
- template `INST_OR_DECL` size\_t `PLUQ` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_DIAG` Diag, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template `INST_OR_DECL` size\_t `LUdivine` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_DIAG` Diag, const `FFLAS::FFLAS_TRANSPOSE` trans, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const `FFPACK_LU_TAG` LuTag, const size\_t cutoff)
- template `INST_OR_DECL` size\_t `LUdivine_small` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_DIAG` Diag, const `FFLAS::FFLAS_TRANSPOSE` trans, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` size\_t `LUdivine_gauss` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_DIAG` Diag, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` size\_t `RowEchelonForm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` size\_t `ReducedRowEchelonForm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` size\_t `ColumnEchelonForm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` size\_t `ReducedColumnEchelonForm` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, `FFLAS_ELT` \*A, const size\_t lda, int &nullity)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, const size\_t ldx, int &nullity)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert2` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, const size\_t ldx, int &nullity)
- template `INST_OR_DECL` std::list< Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > >::Element > & `CharPoly` (const Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > > &R, std::list< Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > >::Element > &charp, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_FIELD`< `FFLAS_ELT` >::RandIter &G, const `FFPACK_CHARPOLY_TAG` CharpTag, const size\_t degree)
- template `INST_OR_DECL` Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > >::Element & `CharPoly` (const Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > > &R, Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > >::Element &charp, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_FIELD`< `FFLAS_ELT` >::RandIter &G, const `FFPACK_CHARPOLY_TAG` CharpTag, const size\_t degree)
- template `INST_OR_DECL` Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > >::Element & `CharPoly` (const Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > > &R, Givaro::Poly1Dom< `FFLAS_FIELD`< `FFLAS_ELT` > >::Element &charp, const size\_t N, `FFLAS_ELT` \*A, const size\_t lda, const `FFPACK_CHARPOLY_TAG` CharpTag, const size\_t degree)
- template `INST_OR_DECL` std::vector< `FFLAS_ELT` > & `MinPoly` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, std::vector< `FFLAS_ELT` > &minP, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_FIELD`< `FFLAS_ELT` >::RandIter &G)
- template `INST_OR_DECL` std::vector< `FFLAS_ELT` > & `MinPoly` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, std::vector< `FFLAS_ELT` > &minP, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda)
- template `INST_OR_DECL` std::vector< `FFLAS_ELT` > & `MatVecMinPoly` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, std::vector< `FFLAS_ELT` > &minP, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*V, const size\_t incv)

- template `INST_OR_DECL` `size_t KrylovElim` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `size_t` deg, `size_t` \*iterates, `size_t` \*inviterates, const `size_t` maxit, `size_t` virt)
- template `INST_OR_DECL` `size_t SpecRankProfile` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `size_t` deg, `size_t` \*rankProfile)
- template `INST_OR_DECL` `size_t Rank` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` `bool IsSingular` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` `FFLAS_ELT & Det` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, `FFLAS_ELT` &det, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q)
- template `INST_OR_DECL` `FFLAS_ELT & Det` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, `FFLAS_ELT` &det, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS::ParSeqHelper::Parallel`< `FFLAS::CuttingStrategy::Recursive`, `FFLAS::StrategyParameter::Threads` > &parH, `size_t` \*P, `size_t` \*Q)
- template `INST_OR_DECL` `FFLAS_ELT * Solve` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*x, const `int` incx, const `FFLAS_ELT` \*b, const `int` incb)
- template `INST_OR_DECL` `void solveLB` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*L, const `size_t` ldl, const `size_t` \*Q, `FFLAS_ELT` \*B, const `size_t` ldb)
- template `INST_OR_DECL` `void solveLB2` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*L, const `size_t` ldl, const `size_t` \*Q, `FFLAS_ELT` \*B, const `size_t` ldb)
- template `INST_OR_DECL` `void RandomNullSpaceVector` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` incX)
- template `INST_OR_DECL` `size_t NullSpaceBasis` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&NS, `size_t` &ldn, `size_t` &NSdim)
- template `INST_OR_DECL` `size_t RowRankProfile` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*rkprofile, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t ColumnRankProfile` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*rkprofile, const `FFPACK_LU_TAG` LuTag)
- `void RankProfileFromLU` (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*rkprofile, const `FFPACK_LU_TAG` LuTag)

*Recovers the column/row rank profile from the permutation of an LU decomposition.*

- `size_t LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)

*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*

- template `INST_OR_DECL` `size_t RowRankProfileSubmatrixIndices` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*rowindices, `size_t` \*colindices, `size_t` &R)
- template `INST_OR_DECL` `size_t ColRankProfileSubmatrixIndices` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*rowindices, `size_t` \*colindices, `size_t` &R)
- template `INST_OR_DECL` `size_t RowRankProfileSubmatrix` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&X, `size_t` &R)
- template `INST_OR_DECL` `size_t ColRankProfileSubmatrix` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&X, `size_t` &R)
- template `INST_OR_DECL` `void getTriangular`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `bool` OnlyNonZeroVectors)
- template `INST_OR_DECL` `void getTriangular`< `FFLAS_FIELD`< `FFLAS_ELT` > > (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*A, const `size_t` lda)

- template `INST_OR_DECL` void `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `FFPACK_LU_TAG` LuTag)
- void `PLUQtoEchelonPermutation` (const `size_t` N, const `size_t` R, const `size_t` \*P, `size_t` \*outPerm)  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- template `INST_OR_DECL` `FFLAS_ELT` \* `LQUptInverseOfFullRankMinor` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` rank, `FFLAS_ELT` \*A\_factors, const `size_t` lda, const `size_t` \*QtPointer, `FFLAS_ELT` \*X, const `size_t` ldx)

## 13.232 blockcuts.inl File Reference

```
#include <fflas-ffpack/fflas/fflas_enum.h>
#include <math.h>
#include <cassert>
```

### Data Structures

- struct [Single](#)
- struct [Row](#)
- struct [Column](#)
- struct [Block](#)
- struct [Recursive](#)
- struct [Fixed](#)
- struct [Threads](#)
- struct [Grain](#)
- struct [TwoD](#)
- struct [TwoDAdaptive](#)
- struct [ThreeD](#)
- struct [ThreeDInPlace](#)
- struct [ThreeDAdaptive](#)
- struct [Parallel< C, P >](#)
- struct [Sequential](#)
- struct [Compose< H1, H2 >](#)
- struct [ForStrategy1D< blocksize\\_t, Cut, Param >](#)
- struct [ForStrategy2D< blocksize\\_t, Cut, Param >](#)

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::CuttingStrategy](#)
- namespace [FFLAS::StrategyParameter](#)
- namespace [FFLAS::ParSeqHelper](#)

*[ParSeqHelper](#) for both [fgemm](#) and [ftrsm](#).*

## Macros

- `#define \_\_FFLASFFPACK\_fflas\_blockcuts\_INL`
- `#define \_\_FFLASFFPACK\_MINBLOCKCUTS ((size_t)256)`

## Typedefs

- typedef [Row](#) [RNSModulus](#)

## Functions

- `template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>  
void BlockCuts (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>  
void BlockCuts (size_t &rowBlockSize, size_t &colBlockSize, size_t &lastRBS, size_t &lastCBS, size_t &changeRBS, size_t &changeCBS, size_t &numRowBlock, size_t &numColBlock, size_t m, size_t n, const size_t numthreads)`

## 13.232.1 Macro Definition Documentation

### 13.232.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_blockcuts\\_INL](#)

```
#define __FFLASFFPACK_fflas_blockcuts_INL
```

### 13.232.1.2 [\\_\\_FFLASFFPACK\\_MINBLOCKCUTS](#)

```
#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)
```

## 13.233 fflas\_plevel1.h File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

### Namespaces

- namespace [FFLAS](#)

### Functions

- template<class [Field](#) >  
void [pfzero](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) C, size\_t BS=0)
- template<class [Field](#) , class [RandIter](#) >  
void [pfrand](#) (const [Field](#) &F, [RandIter](#) &G, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) C, size\_t BS=0)
- template<class [Field](#) , class [Cut](#) , class [Param](#) >  
[Field::Element](#) & [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element](#) &d, const [ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > par)
- template<typename [Field](#) , class [Cut](#) , class [Param](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, const [ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > par)

## 13.234 kaapi\_routines.inl File Reference

### Macros

- #define [\\_\\_FFLASFFPACK\\_KAAPI\\_ROUTINES\\_INL](#)

### 13.234.1 Macro Definition Documentation

#### 13.234.1.1 [\\_\\_FFLASFFPACK\\_KAAPI\\_ROUTINES\\_INL](#)

```
#define __FFLASFFPACK_KAAPI_ROUTINES_INL
```

## 13.235 parallel.h File Reference

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/paladin/blockcuts.inl"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [index\\_t](#) size\_t
- #define [TASK](#)(M, l) {l;}
- #define [WAIT](#)
- #define [CHECK\\_DEPENDENCIES](#)
- #define [BARRIER](#)
- #define [PAR\\_BLOCK](#)
- #define [SYNCH\\_GROUP](#)(Args...) {{Args};}
- #define [NUM\\_THREADS](#) 1
- #define [MAX\\_THREADS](#) 1
- #define [READ](#)(Args...)
- #define [WRITE](#)(Args...)
- #define [READWRITE](#)(Args...)
- #define [CONSTREFERENCE](#)(...)



- `#define VALUE(...)`
- `#define BEGIN_PARALLEL_MAIN(Args...) int main(Args) {`
- `#define END_PARALLEL_MAIN(void) return 0; }`
- `#define FORBLOCK1D(iter, m, Helper, Args...)`
- `#define FOR1D(i, m, Helper, Args...)`
- `#define PARFORBLOCK1D(iter, m, Helper, Args...)`
- `#define PARFOR1D(iter, m, Helper, Args...)`
- `#define FORBLOCK2D(iter, m, n, Helper, Args...)`
- `#define FOR2D(i, j, m, n, Helper, Args...)`
- `#define PARFORBLOCK2D(iter, m, n, Helper, Args...) FORBLOCK2D(iter, m, n, Helper, Args)`
- `#define PARFOR2D(i, j, m, n, Helper, Args...) FOR2D(i, j, m, n, Helper, Args)`
- `#define COMMA ,`
- `#define MODE(...) __VA_ARGS__`
- `#define RETURNPARAM(f, P1, Args...) P1=f(Args)`
- `#define NUMARGS(...) PP_NARG(__VA_ARGS__, PP_RSEQ_N())`
- `#define PP_NARG(...) PP_ARG_N(__VA_ARGS__)`
- `#define PP_ARG_N(_1, _2, _3, _4, _5, _6, _7, _8, _9, _10, _11, _12, _13, _14, _15, _16, _17, _18, _19, _20, _21, _22, _23, _24, _25, _26, _27, _28, _29, _30, _31, _32, _33, _34, _35, _36, _37, _38, _39, _40, _41, _42, _43, _44, _45, _46, _47, _48, _49, _50, _51, _52, _53, _54, _55, _56, _57, _58, _59, _60, _61, _62, _63, N, ...) N`
- `#define PP_RSEQ_N()`
- `#define NOSPLIT() FFLAS::ParSeqHelper::Sequential()`
- `#define splitting_0() FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Threads>()`
- `#define splitting_1(a) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Threads>(a)`
- `#define splitting_2(a, c) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, c>(a)`
- `#define splitting_3(a, b, c) FFLAS::ParSeqHelper::Parallel<b, c>(a)`
- `#define splitt(_1, _2, _3, NAME, ...) NAME`
- `#define SPLITTER(...) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0)(__VA_ARGS__)`

## 13.235.1 Macro Definition Documentation

### 13.235.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 13.235.1.2 index\_t

```
#define index_t size_t
```

### 13.235.1.3 TASK

```
#define TASK(
 M,
 I) { I; }
```

### 13.235.1.4 WAIT

```
#define WAIT
```

### 13.235.1.5 CHECK\_DEPENDENCIES

```
#define CHECK_DEPENDENCIES
```

### 13.235.1.6 BARRIER

```
#define BARRIER
```

**13.235.1.7 PAR\_BLOCK**

```
#define PAR_BLOCK
```

**13.235.1.8 SYNCH\_GROUP**

```
#define SYNCH_GROUP(
 Args...) {{Args}};
```

**13.235.1.9 NUM\_THREADS**

```
#define NUM_THREADS 1
```

**13.235.1.10 MAX\_THREADS**

```
#define MAX_THREADS 1
```

**13.235.1.11 READ**

```
#define READ(
 Args...)
```

**13.235.1.12 WRITE**

```
#define WRITE(
 Args...)
```

**13.235.1.13 READWRITE**

```
#define READWRITE(
 Args...)
```

**13.235.1.14 CONSTREFERENCE**

```
#define CONSTREFERENCE(
 ...)
```

**13.235.1.15 VALUE**

```
#define VALUE(
 ...)
```

**13.235.1.16 BEGIN\_PARALLEL\_MAIN**

```
#define BEGIN_PARALLEL_MAIN(
 Args...) int main(Args) {
```

**13.235.1.17 END\_PARALLEL\_MAIN**

```
#define END_PARALLEL_MAIN(
 void) return 0; }
```

**13.235.1.18 FORBLOCK1D**

```
#define FORBLOCK1D(
 iter,
 m,
 Helper,
 Args...)
```

**Value:**

```
{ FFLAS::ForStrategy1D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
 decltype(Helper)::Param> iter(m, Helper); \
 for(iter.initialize(); !iter.isTerminated(); ++iter) \
 { Args; } }
```

### 13.235.1.19 FOR1D

```
#define FOR1D(
 i,
 m,
 Helper,
 Args...)
```

#### Value:

```
FORBLOCK1D(_internal_iterator, m, Helper,
 for(auto i=_internal_iterator.begin(); i!=_internal_iterator.end(); ++i) \
 { Args; })
```

### 13.235.1.20 PARFORBLOCK1D

```
#define PARFORBLOCK1D(
 iter,
 m,
 Helper,
 Args...)
```

#### Value:

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter) \
{ Args; }
```

### 13.235.1.21 PARFOR1D

```
#define PARFOR1D(
 iter,
 m,
 Helper,
 Args...)
```

#### Value:

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter) \
{ Args; }
```

### 13.235.1.22 FORBLOCK2D

```
#define FORBLOCK2D(
 iter,
 m,
 n,
 Helper,
 Args...)
```

#### Value:

```
{ FFLAS::ForStrategy2D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
 decltype(Helper)::Param> iter(m,n,Helper); \
 for(iter.initialize(); !iter.isTerminated(); ++iter) \
 { Args; } }
```

### 13.235.1.23 FOR2D

```
#define FOR2D(
 i,
 j,
 m,
 n,
 Helper,
 Args...)
```

#### Value:

```
FORBLOCK2D(_internal_iterator, m, n, Helper,
 \
```

```

for(auto i=_internal_iterator.ibegin(); i!=_internal_iterator.iend(); ++i) \
for(auto j=_internal_iterator.jbegin(); j!=_internal_iterator.jend(); ++j) \
{ Args; }

```

#### 13.235.1.24 PARFORBLOCK2D

```

#define PARFORBLOCK2D(
 iter,
 m,
 n,
 Helper,
 Args...) FORBLOCK2D(iter, m, n, Helper, Args)

```

#### 13.235.1.25 PARFOR2D

```

#define PARFOR2D(
 i,
 j,
 m,
 n,
 Helper,
 Args...) FOR2D(i, j, m, n, Helper, Args)

```

#### 13.235.1.26 COMMA

```

#define COMMA ,

```

#### 13.235.1.27 MODE

```

#define MODE(
 ...) __VA_ARGS__

```

#### 13.235.1.28 RETURNPARAM

```

#define RETURNPARAM(
 f,
 Pl,
 Args...) Pl=f(Args)

```

#### 13.235.1.29 NUMARGS

```

#define NUMARGS(
 ...) PP_NARG__(__VA_ARGS__, PP_RSEQ_N())

```

#### 13.235.1.30 PP\_NARG\_\_

```

#define PP_NARG__(
 ...) PP_ARG_N(__VA_ARGS__)

```

#### 13.235.1.31 PP\_ARG\_N

```

#define PP_ARG_N(
 _1,
 _2,
 _3,
 _4,
 _5,
 _6,
 _7,
 _8,

```

```
_9,
_10,
_11,
_12,
_13,
_14,
_15,
_16,
_17,
_18,
_19,
_20,
_21,
_22,
_23,
_24,
_25,
_26,
_27,
_28,
_29,
_30,
_31,
_32,
_33,
_34,
_35,
_36,
_37,
_38,
_39,
_40,
_41,
_42,
_43,
_44,
_45,
_46,
_47,
_48,
_49,
_50,
_51,
_52,
_53,
_54,
_55,
_56,
_57,
_58,
_59,
_60,
_61,
_62,
_63,
N,
...) N
```

**13.235.1.32 PP\_RSEQ\_N**

```
#define PP_RSEQ_N()
```

**Value:**

```
63, 62, 61, 60, \
59, 58, 57, 56, 55, 54, 53, 52, 51, 50, \
49, 48, 47, 46, 45, 44, 43, 42, 41, 40, \
39, 38, 37, 36, 35, 34, 33, 32, 31, 30, \
29, 28, 27, 26, 25, 24, 23, 22, 21, 20, \
19, 18, 17, 16, 15, 14, 13, 12, 11, 10, \
9, 8, 7, 6, 5, 4, 3, 2, 1, 0
```

**13.235.1.33 NOSPLIT**

```
#define NOSPLIT() FFLAS::ParSeqHelper::Sequential()
```

**13.235.1.34 splitting\_0**

```
#define splitting_0() FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Threading>(a)
```

**13.235.1.35 splitting\_1**

```
#define splitting_1(
 a) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Threading>(a)
```

**13.235.1.36 splitting\_2**

```
#define splitting_2(
 a,
 c) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, c>(a)
```

**13.235.1.37 splitting\_3**

```
#define splitting_3(
 a,
 b,
 c) FFLAS::ParSeqHelper::Parallel<b, c>(a)
```

**13.235.1.38 splitt**

```
#define splitt(
 _1,
 _2,
 _3,
 NAME,
 ...) NAME
```

**13.235.1.39 SPLITTER**

```
#define SPLITTER(
 ...) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0) (↔
__VA_ARGS__)
```

**13.236 pfgemm\_variants.inl File Reference****Namespaces**

- namespace [FFLAS](#)

## Functions

- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > > &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDAdaptive`  
`> > &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive`  
`> > &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoD`  
`> > &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)`
- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace`  
`> > &H)`

## 13.237 pfgemv.inl File Reference

### Namespaces

- namespace `FFLAS`

### Functions

- `template<class Field , class AlgoT , class FieldTrait >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t`  
`n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda,`  
`const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, type-`  
`name Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel<`  
`CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)`
- `template<class Field , class AlgoT , class FieldTrait , class Cut >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t`  
`n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t lda,`

```
const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, type-
name Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel<
CuttingStrategy::Row, Cut > > &H)
```

## 13.238 align-allocator.h File Reference

```
#include "fflas-ffpack/config.h"
```

## 13.239 args-parser.h File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/givinteger.h>
#include <givaro/givprint.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <cstring>
#include <list>
#include <stdlib.h>
```

### Data Structures

- struct [Argument](#)

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [TYPE\\_BOOL](#) [TYPE\\_NONE](#)
- #define [END\\_OF\\_ARGUMENTS](#) { '\0', "\0", "\0", [TYPE\\_NONE](#), NULL }
- #define [type\\_integer](#) long int

### Enumerations

- enum [ArgumentType](#) {  
[TYPE\\_NONE](#) , [TYPE\\_INT](#) , [TYPE\\_UINT64](#) , [TYPE\\_LONGLONG](#) ,  
[TYPE\\_INTEGER](#) , [TYPE\\_DOUBLE](#) , [TYPE\\_INTLIST](#) , [TYPE\\_STR](#) }

### Functions

- void [parseArguments](#) (int argc, char \*\*argv, [Argument](#) \*args, bool printDefaults=true)
- void [printHelpMessage](#) (const char \*program, [Argument](#) \*args, bool printDefaults=false)
- [Argument](#) \* [findArgument](#) ([Argument](#) \*args, char c)
- int [getListArgs](#) (std::list< int > &outlist, std::string &instring)  
*transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}*
- std::ostream & [writeCommandString](#) (std::ostream &os, [Argument](#) \*args, const char \*programName=nullptr)  
*writes the values of all arguments, preceded by the programName*

## 13.239.1 Macro Definition Documentation

### 13.239.1.1 TYPE\_BOOL

```
#define TYPE_BOOL TYPE_NONE
```



### 13.239.1.2 END\_OF\_ARGUMENTS

```
#define END_OF_ARGUMENTS { '\0', "\0", "\0", TYPE_NONE, NULL }
```

### 13.239.1.3 type\_integer

```
#define type_integer long int
```

## 13.239.2 Enumeration Type Documentation

### 13.239.2.1 ArgumentType

```
enum ArgumentType
```

Enumerator

|               |  |
|---------------|--|
| TYPE_NONE     |  |
| TYPE_INT      |  |
| TYPE_UINT64   |  |
| TYPE_LONGLONG |  |
| TYPE_INTEGER  |  |
| TYPE_DOUBLE   |  |
| TYPE_INTLIST  |  |
| TYPE_STR      |  |

## 13.239.3 Function Documentation

### 13.239.3.1 printHelpMessage()

```
void printHelpMessage (
 const char * program,
 Argument * args,
 bool printDefaults = false)
```

### 13.239.3.2 findArgument()

```
Argument * findArgument (
 Argument * args,
 char c)
```

### 13.239.3.3 getListArgs()

```
int getListArgs (
 std::list< int > & outlist,
 std::string & instring)
```

transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}

Parameters

|                 |                      |
|-----------------|----------------------|
| <i>outlist</i>  | list once converted  |
| <i>instring</i> | list to be converted |

**Returns**

status message.

**13.240 bit\_manipulation.h File Reference**

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
```

**Macros**

- `#define __has_builtin(x) 0`

**Functions**

- [int32\\_t clz \(uint64\\_t val\)](#)
- [int32\\_t clz \(uint32\\_t val\)](#)
- [int32\\_t ctz \(uint32\\_t val\)](#)
- [int32\\_t ctz \(uint64\\_t val\)](#)

**13.240.1 Macro Definition Documentation****13.240.1.1 \_\_has\_builtin**

```
#define __has_builtin(
 x) 0
```

**13.240.2 Function Documentation****13.240.2.1 clz() [1/2]**

```
int32_t clz (
 uint64_t val) [inline]
```

**13.240.2.2 clz() [2/2]**

```
int32_t clz (
 uint32_t val) [inline]
```

**13.240.2.3 ctz() [1/2]**

```
int32_t ctz (
 uint32_t val) [inline]
```

**13.240.2.4 ctz() [2/2]**

```
int32_t ctz (
 uint64_t val) [inline]
```

**13.241 cast.h File Reference****Namespaces**

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Functions

- `template<class T, class CT = const T>`  
`T fflas_const_cast (CT x)`

## 13.242 debug.h File Reference

Various utilities for debugging.

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <iostream>
#include <sstream>
#include <cmath>
#include <stdexcept>
```

## Data Structures

- class `Failure`  
*A precondition failed.*

## Namespaces

- namespace `FFPACK`  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define FFLASFFPACK_check(check)`
- `#define FFLASFFPACK_abort(msg)`

## Functions

- `Failure` & `failure ()`
- `template<class T >`  
`bool isOdd (const T &a)`
- `bool isOdd (const float &a)`
- `bool isOdd (const double &a)`

### 13.242.1 Detailed Description

Various utilities for debugging.

**Todo** we should put vector printing elsewhere.

### 13.242.2 Macro Definition Documentation

#### 13.242.2.1 FFLASFFPACK\_check

```
#define FFLASFFPACK_check(
 check)
```

**Value:**

```
if (!(check)) {\
 FFPACK::failure() (__func__, __FILE__, __LINE__, #check); \
 throw std::runtime_error(#check); \
}
```

### 13.242.2.2 FFLASFFPACK\_abort

```
#define FFLASFFPACK_abort(
 msg)
```

#### Value:

```
{
 FFPACK::failure() (__func__, __FILE__, __LINE__, msg); \
 throw std::runtime_error(msg); \
}
```

## 13.243 fflas\_intrinsic.h File Reference

## 13.244 fflas\_io.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include <fstream>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas_memory.h"
```

### Namespaces

- namespace [FFLAS](#)

### Enumerations

- enum [FFLAS\\_FORMAT](#) {  
    FflasAuto = 0 , FflasDense = 1 , FflasSMS = 2 , FflasBinary = 3 ,  
    FflasMath = 4 , FflasMaple = 5 , FflasSageMath = 6 }

### Functions

- template<class [Field](#) >  
std::ostream & [WriteMatrix](#) (std::ostream &c, const [Field](#) &F, size\_t m, size\_t n, typename [Field::ConstElement\\_ptr](#) A, size\_t lda, [FFLAS\\_FORMAT](#) format, bool column\_major)  
*WriteMatrix: write a matrix to an output stream.*
- void [preamble](#) (std::ifstream &if, [FFLAS\\_FORMAT](#) &format)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [ReadMatrix](#) (std::ifstream &if, [Field](#) &F, size\_t &m, size\_t &n, typename [Field::Element\\_ptr](#) &A, [FFLAS\\_FORMAT](#) format=[FflasAuto](#))  
*ReadMatrix: read a matrix from an input stream.*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [ReadMatrix](#) (const std::string &matrix\_file, [Field](#) &F, size\_t &m, size\_t &n, typename [Field::Element\\_ptr](#) &A, [FFLAS\\_FORMAT](#) format=[FflasAuto](#))  
*ReadMatrix: read a matrix from a file.*
- template<class [Field](#) >  
void [WriteMatrix](#) (std::string &matrix\_file, const [Field](#) &F, int m, int n, typename [Field::ConstElement\\_ptr](#) A, size\_t lda, [FFLAS\\_FORMAT](#) format=[FflasDense](#), bool column\_major=false)  
*WriteMatrix: write a matrix to a file.*
- std::ostream & [WritePermutation](#) (std::ostream &c, const size\_t \*P, size\_t N)  
*WritePermutation: write a permutation matrix to an output stream.*

## 13.245 fflas\_memory.h File Reference

```
#include "fflas-ffpack/utils/align-allocator.h"
#include <givaro/givinteger.h>
```

### Namespaces

- namespace [FFLAS](#)

### Functions

- template<class Element >  
bool [alignable](#) ()
- template<> bool [alignable](#)< Givaro::Integer \* > ()
- template<class Field >  
[Field::Element\\_ptr](#) [fflas\\_new](#) (const [Field](#) &F, const size\_t m, const Alignment align=Alignment::DEFAULT)
- template<class Field >  
[Field::Element\\_ptr](#) [fflas\\_new](#) (const [Field](#) &F, const size\_t m, const size\_t n, const Alignment align=Alignment::DEFAULT)
- template<class Element >  
Element \* [fflas\\_new](#) (const size\_t m, const Alignment align=Alignment::DEFAULT)
- template<class Element\_ptr >  
void [fflas\\_delete](#) (Element\_ptr A)
- template<class Ptr , class ... Args>  
void [fflas\\_delete](#) (Ptr p, Args ... args)
- void [prefetch](#) (const int64\_t \*)
- void [getTLBSize](#) (int &tlb)
- void [queryCacheSizes](#) (int &l1, int &l2, int &l3)
- int [queryL1CacheSize](#) ()
- int [queryTopLevelCacheSize](#) ()

## 13.246 fflas\_randommatrix.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- template<class Field , class RandIter >  
[Field::Element\\_ptr](#) [NonZeroRandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, RandIter &G)  
*Random non-zero Matrix.*
- template<class Field , class RandIter >  
[Field::Element\\_ptr](#) [NonZeroRandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda)

*Random non-zero Matrix.*

- template<class [Field](#) , class [RandIter](#) >  
[Field::Element\\_ptr](#) [RandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, [RandIter](#) &G)

*Random Matrix.*

- template<class [Field](#) >  
[Field::Element\\_ptr](#) [RandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda)

*Random Matrix.*

- template<class [Field](#) , class [RandIter](#) >  
[Field::Element\\_ptr](#) [RandomTriangularMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_DIAG](#) Diag, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda, [RandIter](#) &G)

*Random Triangular Matrix.*

- template<class [Field](#) >  
[Field::Element\\_ptr](#) [RandomTriangularMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_DIAG](#) Diag, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda)

*Random Triangular Matrix.*

- size\_t [RandInt](#) (size\_t a, size\_t b)
- template<class [Field](#) , class [RandIter](#) >  
[Field::Element\\_ptr](#) [RandomSymmetricMatrix](#) (const [Field](#) &F, size\_t n, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda, [RandIter](#) &G)

*Random Symmetric Matrix.*

- template<class [Field](#) , class [RandIter](#) >  
[Field::Element\\_ptr](#) [RandomMatrixWithRank](#) (const [Field](#) &F, size\_t m, size\_t n, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda, [RandIter](#) &G)

*Random Matrix with prescribed rank.*

- template<class [Field](#) >  
[Field::Element\\_ptr](#) [RandomMatrixWithRank](#) (const [Field](#) &F, size\_t m, size\_t n, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda)

*Random Matrix with prescribed rank.*

- size\_t \* [RandomIndexSubset](#) (size\_t N, size\_t R, size\_t \*P)  
*Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.*
- size\_t \* [RandomPermutation](#) (size\_t N, size\_t \*P)  
*Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.*
- void [RandomRankProfileMatrix](#) (size\_t M, size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.*
- void [swapval](#) (size\_t k, size\_t N, size\_t \*P, size\_t val)
- void [RandomSymmetricRankProfileMatrix](#) (size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random a symmetric  $R$ -subpermutation of dimension  $N \times N$  : a symmetric matrix with only  $R$  non-zeros, all equal to one, in a random rook placement.*

- template<class [Field](#) , class [RandIter](#) >  
[Field::Element\\_ptr](#) [RandomMatrixWithRankandRPM](#) (const [Field](#) &F, size\_t M, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, [RandIter](#) &G)

*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*

- template<class [Field](#) >  
[Field::Element\\_ptr](#) [RandomMatrixWithRankandRPM](#) (const [Field](#) &F, size\_t M, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)

*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*

- template<class [Field](#) , class [RandIter](#) >  
[Field::Element\\_ptr](#) [RandomSymmetricMatrixWithRankandRPM](#) (const [Field](#) &F, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, [RandIter](#) &G)

*Random Symmetric Matrix with prescribed rank and rank profile matrix* Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .

- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix* Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, RandIter &G)  
*Random Matrix with prescribed rank, with random rank profile matrix* Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)  
*Random Matrix with prescribed rank, with random rank profile matrix* Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, RandIter &G)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix* Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.
- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix* Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, size\_t n, const typename `Field::Element` d, typename `Field::Element_ptr` A, size\_t lda)  
*Random Matrix with prescribed det.*
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, size\_t n, const typename `Field::Element` d, typename `Field::Element_ptr` A, size\_t lda, RandIter &G)  
*Random Matrix with prescribed det.*

## 13.247 flimits.h File Reference

```
#include <climits>
#include <limits>
#include <type_traits>
#include <givaro/givinteger.h>
```

### Data Structures

- `struct limits< unsigned char >`
- `struct limits< signed char >`
- `struct limits< char >`
- `struct limits< unsigned short int >`
- `struct limits< short int >`
- `struct limits< unsigned int >`
- `struct limits< int >`
- `struct limits< unsigned long >`
- `struct limits< long >`

- struct [limits< unsigned long long >](#)
- struct [limits< long long >](#)
- struct [limits< float >](#)
- struct [limits< double >](#)
- struct [limits< Givaro::Integer >](#)
- struct [limits< Reclnt::ruint< K > >](#)
- struct [limits< Reclnt::rint< K > >](#)

## Functions

- template<class T, class E >  
std::enable\_if< std::is\_signed< T >::value==std::is\_signed< E >::value, bool >::type [in\\_range](#) (E e)
- template<class T, class E >  
std::enable\_if<(std::is\_signed< T >::value)&&!(std::is\_signed< E >::value), bool >::type [in\\_range](#) (E e)
- template<class T, class E >  
std::enable\_if<!(std::is\_signed< T >::value)&&(std::is\_signed< E >::value), bool >::type [in\\_range](#) (E e)

## 13.247.1 Function Documentation

### 13.247.1.1 in\_range() [1/3]

```
template<class T, class E >
std::enable_if< std::is_signed< T >::value==std::is_signed< E >::value, bool >::type in_↵
range (
 E e)
```

### 13.247.1.2 in\_range() [2/3]

```
template<class T, class E >
std::enable_if<(std::is_signed< T >::value)&&!(std::is_signed< E >::value), bool >::type
in_range (
 E e)
```

### 13.247.1.3 in\_range() [3/3]

```
template<class T, class E >
std::enable_if<!(std::is_signed< T >::value)&&(std::is_signed< E >::value), bool >::type
in_range (
 E e)
```

## 13.248 Matio.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include "fflas_memory.h"
```

## Functions

- template<class [Field](#) >  
[Field::Element\\_ptr read\\_field](#) (const [Field](#) &F, const char \*mat\_file, size\_t \*tni, size\_t \*tnj)
- template<class [Field](#) >  
std::ostream & [write\\_field](#) (const [Field](#) &F, std::ostream &c, typename [Field::ConstElement\\_ptr](#) E, int n, int m, int id, bool mapFormat=false, bool column\_major=false)



## 13.248.1 Function Documentation

### 13.248.1.1 read\_field()

```
template<class Field >
Field::Element_ptr read_field (
 const Field & F,
 const char * mat_file,
 size_t * tni,
 size_t * tnj)
```

### 13.248.1.2 write\_field()

```
template<class Field >
std::ostream & write_field (
 const Field & F,
 std::ostream & c,
 typename Field::ConstElement_ptr E,
 int n,
 int m,
 int id,
 bool mapleFormat = false,
 bool column_major = false)
```

## 13.249 test-utils.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include <random>
#include <functional>
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- [uint64\\_t](#) [getSeed](#) ()
- template<typename [Field](#) >  
Givaro::Integer [maxFieldElt](#) ()
- template<> Givaro::Integer [maxFieldElt](#)< Givaro::ZRing< Givaro::Integer > > ()
- template<typename [Field](#) >  
[Field](#) \* [chooseField](#) (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- template<> Givaro::ZRing< [int32\\_t](#) > \* [chooseField](#)< Givaro::ZRing< [int32\\_t](#) > > (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- template<> Givaro::ZRing< [int64\\_t](#) > \* [chooseField](#)< Givaro::ZRing< [int64\\_t](#) > > (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- template<> Givaro::ZRing< float > \* [chooseField](#)< Givaro::ZRing< float > > (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- template<> Givaro::ZRing< double > \* [chooseField](#)< Givaro::ZRing< double > > (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)

## 13.250 timer.h File Reference

```
#include <time.h>
#include <givaro/givtimer.h>
```

### Namespaces

- namespace [FFLAS](#)

### Typedefs

- typedef Givaro::Timer [Timer](#)
- typedef Givaro::BaseTimer [BaseTimer](#)
- typedef Givaro::UserTimer [UserTimer](#)
- typedef Givaro::SysTimer [SysTimer](#)

## 13.251 cblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)
- #define [\\_\\_FFLASFFPACK\\_HAVE\\_CBLAS](#) 1

### Functions

- int [main](#) ()

### 13.251.1 Macro Definition Documentation

#### 13.251.1.1 [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)

```
#define __FFLASFFPACK_CONFIGURATION
```

#### 13.251.1.2 [\\_\\_FFLASFFPACK\\_HAVE\\_CBLAS](#)

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

### 13.251.2 Function Documentation

#### 13.251.2.1 [main\(\)](#)

```
int main (
 void)
```

## 13.252 clapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)
- #define [\\_\\_FFLASFFPACK\\_HAVE\\_LAPACK](#) 1
- #define [\\_\\_FFLASFFPACK\\_HAVE\\_CLAPACK](#) 1

**Functions**

- int [main](#) ()

**13.252.1 Macro Definition Documentation****13.252.1.1 \_\_FFLASFFPACK\_CONFIGURATION**

```
#define __FFLASFFPACK_CONFIGURATION
```

**13.252.1.2 \_\_FFLASFFPACK\_HAVE\_LAPACK**

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

**13.252.1.3 \_\_FFLASFFPACK\_HAVE\_CLAPACK**

```
#define __FFLASFFPACK_HAVE_CLAPACK 1
```

**13.252.2 Function Documentation****13.252.2.1 main()**

```
int main (
 void)
```

**13.253 cuda.C File Reference**

```
#include <stdio.h>
#include <cuda_runtime.h>
#include <cusparse.h>
```

**Functions**

- int [main](#) ()

**13.253.1 Function Documentation****13.253.1.1 main()**

```
int main (
 void)
```

**13.254 fblas.C File Reference**

```
#include "fflas-ffpack/config-blas.h"
```

**Macros**

- #define [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)

**Functions**

- void [dgemm\\_](#) (const char \*, const char \*, const int \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)
- int [main](#) ()

### 13.254.1 Macro Definition Documentation

#### 13.254.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

### 13.254.2 Function Documentation

#### 13.254.2.1 dgemm\_()

```
void dgemm_ (
 const char * ,
 const char * ,
 const int * ,
 const int * ,
 const int * ,
 const double * ,
 const double * ,
 const int * ,
 const double * ,
 const int * ,
 const double * ,
 double * ,
 const int *)
```

#### 13.254.2.2 main()

```
int main (
 void)
```

## 13.255 gmp.C File Reference

```
#include <gmpxx.h>
```

### Functions

- int [main](#) ()

### 13.255.1 Function Documentation

#### 13.255.1.1 main()

```
int main (
 void)
```

## 13.256 instrset.h File Reference

```
#include <stdlib.h>
```

### Data Structures

- class [Const\\_int\\_t< n >](#)
- class [Const\\_uint\\_t< n >](#)
- class [Static\\_error\\_check< bool >](#)
- class [Static\\_error\\_check< false >](#)

## Macros

- #define `INSTRSET_H` 125
- #define `INSTRSET` 0
- #define `const_int(n)` (`Const_int_t` <n>())
- #define `const_uint(n)` (`Const_uint_t` <n>())

## Typedefs

- typedef signed char `int8_t`
- typedef unsigned char `uint8_t`
- typedef signed short int `int16_t`
- typedef unsigned short int `uint16_t`
- typedef signed int `int32_t`
- typedef unsigned int `uint32_t`
- typedef long long `int64_t`
- typedef unsigned long long `uint64_t`
- typedef `int32_t` `intptr_t`

## Functions

- int `instrset_detect` (void)
- bool `hasFMA3` (void)
- bool `hasFMA4` (void)
- bool `hasXOP` (void)
- bool `hasAVX512ER` (void)

## 13.256.1 Macro Definition Documentation

### 13.256.1.1 INSTRSET\_H

```
#define INSTRSET_H 125
```

### 13.256.1.2 INSTRSET

```
#define INSTRSET 0
```

### 13.256.1.3 const\_int

```
#define const_int(
 n) (Const_int_t <n>())
```

### 13.256.1.4 const\_uint

```
#define const_uint(
 n) (Const_uint_t <n>())
```

## 13.256.2 Typedef Documentation

### 13.256.2.1 int8\_t

```
typedef signed char int8_t
```

### 13.256.2.2 uint8\_t

```
typedef unsigned char uint8_t
```

### 13.256.2.3 int16\_t

```
typedef signed short int int16_t
```

#### 13.256.2.4 uint16\_t

```
typedef unsigned short int uint16_t
```

#### 13.256.2.5 int32\_t

```
typedef signed int int32_t
```

#### 13.256.2.6 uint32\_t

```
typedef unsigned int uint32_t
```

#### 13.256.2.7 int64\_t

```
typedef long long int64_t
```

#### 13.256.2.8 uint64\_t

```
typedef unsigned long long uint64_t
```

#### 13.256.2.9 intptr\_t

```
typedef int32_t intptr_t
```

### 13.256.3 Function Documentation

#### 13.256.3.1 instrset\_detect()

```
int instrset_detect (
 void)
```

#### 13.256.3.2 hasFMA3()

```
bool hasFMA3 (
 void)
```

#### 13.256.3.3 hasFMA4()

```
bool hasFMA4 (
 void)
```

#### 13.256.3.4 hasXOP()

```
bool hasXOP (
 void)
```

#### 13.256.3.5 hasAVX512ER()

```
bool hasAVX512ER (
 void)
```

### 13.257 instrset\_detect.cpp File Reference

```
#include "instrset.h"
```

**Functions**

- int [instrset\\_detect](#) (void)
- bool [hasFMA3](#) (void)
- bool [hasFMA4](#) (void)
- bool [hasXOP](#) (void)
- bool [hasF16C](#) (void)
- bool [hasAVX512ER](#) (void)

**13.257.1 Function Documentation****13.257.1.1 instrset\_detect()**

```
int instrset_detect (
 void)
```

**13.257.1.2 hasFMA3()**

```
bool hasFMA3 (
 void)
```

**13.257.1.3 hasFMA4()**

```
bool hasFMA4 (
 void)
```

**13.257.1.4 hasXOP()**

```
bool hasXOP (
 void)
```

**13.257.1.5 hasF16C()**

```
bool hasF16C (
 void)
```

**13.257.1.6 hasAVX512ER()**

```
bool hasAVX512ER (
 void)
```

**13.258 lapack.C File Reference**

```
#include "fflas-ffpack/config-blas.h"
```

**Macros**

- `#define \_\_FFLASFFPACK\_CONFIGURATION`
- `#define \_\_FFLASFFPACK\_HAVE\_LAPACK 1`

**Functions**

- int [main](#) ()

**13.258.1 Macro Definition Documentation****13.258.1.1 \_\_FFLASFFPACK\_CONFIGURATION**

```
#define __FFLASFFPACK_CONFIGURATION
```

**13.258.1.2 \_\_FFLASFFPACK\_HAVE\_LAPACK**

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

**13.258.2 Function Documentation****13.258.2.1 main()**

```
int main (
 void)
```

**13.259 regression-check.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
```

**Functions**

- bool [check1](#) ()
- bool [check2](#) ()
- bool [check3](#) ()
- bool [check4](#) ()
- bool [checkZeroDimCharpoly](#) ()
- bool [checkZeroDimMinPoly](#) ()
- bool [gf2ModularBalanced](#) ()
- int [main](#) ()

**13.259.1 Function Documentation****13.259.1.1 check1()**

```
bool check1 ()
```

**13.259.1.2 check2()**

```
bool check2 ()
```

**13.259.1.3 check3()**

```
bool check3 ()
```

**13.259.1.4 check4()**

```
bool check4 ()
```

**13.259.1.5 checkZeroDimCharpoly()**

```
bool checkZeroDimCharpoly ()
```

**13.259.1.6 checkZeroDimMinPoly()**

```
bool checkZeroDimMinPoly ()
```

**13.259.1.7 gf2ModularBalanced()**

```
bool gf2ModularBalanced ()
```



**13.259.1.8 main()**

```
int main (
 void)
```

**13.260 test-charpoly-check.C File Reference**

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

**Macros**

- `#define` [ENABLE\\_CHECKER\\_charpoly](#) 1
- `#define` [TIME\\_CHECKER\\_CHARPOLY](#) 1

**Functions**

- `template<class` [Field](#) `, class Polynomial >`  
void [printPolynomial](#) (const [Field](#) &F, Polynomial &v)
- `int` [main](#) (int argc, char \*\*argv)

**13.260.1 Macro Definition Documentation****13.260.1.1 ENABLE\_CHECKER\_charpoly**

```
#define ENABLE_CHECKER_charpoly 1
```

**13.260.1.2 TIME\_CHECKER\_CHARPOLY**

```
#define TIME_CHECKER_CHARPOLY 1
```

**13.260.2 Function Documentation****13.260.2.1 printPolynomial()**

```
template<class Field , class Polynomial >
void printPolynomial (
 const Field & F,
 Polynomial & v)
```

**13.260.2.2 main()**

```
int main (
 int argc,
 char ** argv)
```

**13.261 test-charpoly.C File Reference**

```
#include <iostream>
#include <iomanip>
#include "givaro/modular.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
#include <chrono>
```

## Functions

- `template<class Field , class RandIter >`  
`bool launch\_test (const Field &F, size_t n, typename Field::Element *A, size_t lda, size_t nbit, RandIter &G, FFPACK::FFPACK_CHARPOLY_TAG CT)`
- `template<class Field >`  
`bool run\_with\_field (const Givaro::Integer p, uint64\_t bits, size_t n, std::string file, int variant, size_t iter, uint64\_t seed)`
- `int main (int argc, char **argv)`

## 13.261.1 Function Documentation

### 13.261.1.1 [launch\\_test\(\)](#)

```
template<class Field , class RandIter >
bool launch_test (
 const Field & F,
 size_t n,
 typename Field::Element * A,
 size_t lda,
 size_t nbit,
 RandIter & G,
 FFPACK::FFPACK_CHARPOLY_TAG CT)
```

### 13.261.1.2 [run\\_with\\_field\(\)](#)

```
template<class Field >
bool run_with_field (
 const Givaro::Integer p,
 uint64_t bits,
 size_t n,
 std::string file,
 int variant,
 size_t iter,
 uint64_t seed)
```

### 13.261.1.3 [main\(\)](#)

```
int main (
 int argc,
 char ** argv)
```

## 13.262 test-compressQ.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <list>
#include <vector>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
```

```
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Typedefs

- typedef Givaro::Modular< double > [Field](#)

## Functions

- template<class T >  
std::ostream & [printvect](#) (std::ostream &o, vector< T > &vect)
- int [main](#) (int argc, char \*\*argv)

## 13.262.1 Typedef Documentation

### 13.262.1.1 Field

```
typedef Givaro::Modular<double> Field
```

## 13.262.2 Function Documentation

### 13.262.2.1 printvect()

```
template<class T >
std::ostream & printvect (
 std::ostream & o,
 vector< T > & vect)
```

[Bug](#) does not belong here

### 13.262.2.2 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.263 test-det-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
```

## Macros

- #define [ENABLE\\_CHECKER\\_Det](#) 1
- #define [TIME\\_CHECKER\\_Det](#) 1

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.263.1 Macro Definition Documentation

#### 13.263.1.1 ENABLE\_CHECKER\_Det

```
#define ENABLE_CHECKER_Det 1
```

#### 13.263.1.2 TIME\_CHECKER\_Det

```
#define TIME_CHECKER_Det 1
```

### 13.263.2 Function Documentation

#### 13.263.2.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.264 test-det.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

### Functions

- template<class [Field](#) , class RandIter >  
bool [test\\_det](#) ([Field](#) &F, size\_t n, int iter, RandIter &G)
- int [main](#) (int argc, char \*\*argv)

### 13.264.1 Function Documentation

#### 13.264.1.1 test\_det()

```
template<class Field , class RandIter >
bool test_det (
 Field & F,
 size_t n,
 int iter,
 RandIter & G)
```

**Todo** test with stride

#### 13.264.1.2 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.265 test-echelon.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <iomanip>
#include <givaro/modular-balanced.h>
#include <givaro/udl.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include <random>
#include <chrono>
```

### Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25`
- `#define __FFLASFFPACK_PLUQ_THRESHOLD 25`

### Functions

- `template<class Field , class RandIter >`  
`bool test_colechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag,`  
`RandIter &G, bool par)`
- `template<class Field , class RandIter >`  
`bool test_rowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag,`  
`RandIter &G, bool par)`
- `template<class Field , class RandIter >`  
`bool test_redcolechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag,`  
`RandIter &G, bool par)`
- `template<class Field , class RandIter >`  
`bool test_redrowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag,`  
`RandIter &G, bool par)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

### 13.265.1 Macro Definition Documentation

#### 13.265.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

#### 13.265.1.2 \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25
```

#### 13.265.1.3 \_\_FFLASFFPACK\_PLUQ\_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 25
```

### 13.265.2 Function Documentation

#### 13.265.2.1 test\_colechelon()

```
template<class Field , class RandIter >
bool test_colechelon (
```

```

 Field & F,
 size_t m,
 size_t n,
 size_t r,
 size_t iters,
 FFPACK::FFPACK_LU_TAG LuTag,
 RandIter & G,
 bool par)

```

**Todo** check Ida

### 13.265.2.2 test\_rowechelon()

```

template<class Field , class RandIter >
bool test_rowechelon (
 Field & F,
 size_t m,
 size_t n,
 size_t r,
 size_t iters,
 FFPACK::FFPACK_LU_TAG LuTag,
 RandIter & G,
 bool par)

```

**Todo** check Ida

### 13.265.2.3 test\_redcolechelon()

```

template<class Field , class RandIter >
bool test_redcolechelon (
 Field & F,
 size_t m,
 size_t n,
 size_t r,
 size_t iters,
 FFPACK::FFPACK_LU_TAG LuTag,
 RandIter & G,
 bool par)

```

**Todo** check Ida

### 13.265.2.4 test\_redrowechelon()

```

template<class Field , class RandIter >
bool test_redrowechelon (
 Field & F,
 size_t m,
 size_t n,
 size_t r,
 size_t iters,
 FFPACK::FFPACK_LU_TAG LuTag,
 RandIter & G,
 bool par)

```

**Todo** check Ida

### 13.265.2.5 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 size_t m,
 size_t n,
 size_t r,
 size_t iters,
 uint64_t seed)
```

### 13.265.2.6 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.266 test-fadd.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
```

### Functions

- template<class Field >  
bool [test\\_fadd](#) (const Field &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- template<class Field >  
bool [test\\_faddin](#) (const Field &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- template<class Field >  
bool [test\\_fsub](#) (const Field &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- template<class Field >  
bool [test\\_fsubin](#) (const Field &F, size\_t m, size\_t k, size\_t n, bool timing, uint64\_t seed)
- int [main](#) (int ac, char \*\*av)

### 13.266.1 Function Documentation

#### 13.266.1.1 test\_fadd()

```
template<class Field >
bool test_fadd (
 const Field & F,
 size_t m,
 size_t k,
 size_t n,
 bool timing,
 uint64_t seed)
```

#### 13.266.1.2 test\_faddin()

```
template<class Field >
bool test_faddin (
```

```

 const Field & F,
 size_t m,
 size_t k,
 size_t n,
 bool timing,
 uint64_t seed)

```

#### 13.266.1.3 test\_fsub()

```

template<class Field >
bool test_fsub (
 const Field & F,
 size_t m,
 size_t k,
 size_t n,
 bool timing,
 uint64_t seed)

```

#### 13.266.1.4 test\_fsubin()

```

template<class Field >
bool test_fsubin (
 const Field & F,
 size_t m,
 size_t k,
 size_t n,
 bool timing,
 uint64_t seed)

```

#### 13.266.1.5 main()

```

int main (
 int ac,
 char ** av)

```

### 13.267 test-fdot.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include <givaro/zring.h>
#include <givaro/modular.h>
#include <random>
#include <chrono>

```

#### Macros

- #define `ENABLE_ALL_CHECKINGS` 1

#### Functions

- template<typename `Field` >



```
bool check_fdot (const Field &F, size_t n, typename Field::ConstElement_ptr a, size_t inca, typename
Field::ConstElement_ptr b, size_t incb)
```

- template<class Field >  
bool run\_with\_field (Givaro::Integer q, size\_t BS, size\_t n, size\_t iters, uint64\_t seed)
- bool run\_with\_Integer (size\_t BS, size\_t n, size\_t iters, uint64\_t seed)
- int main (int argc, char \*\*argv)

## 13.267.1 Macro Definition Documentation

### 13.267.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.267.2 Function Documentation

### 13.267.2.1 check\_fdot()

```
template<typename Field >
bool check_fdot (
 const Field & F,
 size_t n,
 typename Field::ConstElement_ptr a,
 size_t inca,
 typename Field::ConstElement_ptr b,
 size_t incb)
```

### 13.267.2.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t BS,
 size_t n,
 size_t iters,
 uint64_t seed)
```

### 13.267.2.3 run\_with\_Integer()

```
bool run_with_Integer (
 size_t BS,
 size_t n,
 size_t iters,
 uint64_t seed)
```

### 13.267.2.4 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.268 test-fgemm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

## Macros

- `#define ENABLE_ALL_CHECKINGS 1`

## Functions

- `template<class Field , class RandIter >`  
`bool launch_MM_dispatch (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, RandIter &G)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, int m, int n, int k, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 13.268.1 Macro Definition Documentation

### 13.268.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.268.2 Function Documentation

### 13.268.2.1 launch\_MM\_dispatch()

```
template<class Field , class RandIter >
bool launch_MM_dispatch (
 const Field & F,
 const int mm,
 const int nn,
 const int kk,
 const typename Field::Element alpha,
 const typename Field::Element beta,
 const size_t iters,
 RandIter & G)
```

**Bug** test for IdX equal

**Bug**

**Bug** test for transpo

**Bug**

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 13.268.2.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 int m,
 int n,
 int k,
 size_t iters,
 uint64_t seed)
```

### 13.268.2.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.269 test-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
```

### Macros

- #define `ENABLE_CHECKER_fgemm` 1

### Functions

- template<class `Field` >  
bool `check_MM` (const `Field` &F, const typename `Field::Element_ptr` Cd, enum `FFLAS_TRANSPOSE` &ta, enum `FFLAS_TRANSPOSE` &tb, const size\_t m, const size\_t n, const size\_t k, const typename `Field::Element` &alpha, const typename `Field::Element_ptr` A, size\_t lda, const typename `Field::Element_ptr` B, size\_t ldb, const typename `Field::Element` &beta, const typename `Field::Element_ptr` C, size\_t ldc)
- template<class `Field`, class `RandIter` >  
bool `launch_MM` (const `Field` &F, const size\_t m, const size\_t n, const size\_t k, const typename `Field::Element` alpha, const typename `Field::Element` beta, const size\_t ldc, const size\_t lda, enum `FFLAS_TRANSPOSE` ta, const size\_t ldb, enum `FFLAS_TRANSPOSE` tb, size\_t iters, int nbw, bool par, `RandIter` &G)
- template<class `Field`, class `RandIter` >  
bool `launch_MM_dispatch` (const `Field` &F, const int mm, const int nn, const int kk, const typename `Field::Element` alpha, const typename `Field::Element` beta, const size\_t iters, const int nbw, const bool par, `RandIter` &G)
- template<class `Field` >  
bool `run_with_field` (`Givaro::Integer` q, `uint64_t` b, int m, int n, int k, int nbw, size\_t iters, bool par, size\_t seed)
- int `main` (int argc, char \*\*argv)

### 13.269.1 Macro Definition Documentation

#### 13.269.1.1 `ENABLE_CHECKER_fgemm`

```
#define ENABLE_CHECKER_fgemm 1
```

### 13.269.2 Function Documentation

#### 13.269.2.1 `check_MM()`

```
template<class Field >
bool check_MM (
 const Field & F,
 const typename Field::Element_ptr Cd,
 enum FFLAS_TRANSPOSE & ta,
 enum FFLAS_TRANSPOSE & tb,
 const size_t m,
 const size_t n,
 const size_t k,
 const typename Field::Element & alpha,
 const typename Field::Element_ptr A,
 size_t lda,
```

```

 const typename Field::Element_ptr B,
 size_t ldb,
 const typename Field::Element & beta,
 const typename Field::Element_ptr C,
 size_t ldc)

```

### 13.269.2.2 launch\_MM()

```

template<class Field , class RandIter >
bool launch_MM (
 const Field & F,
 const size_t m,
 const size_t n,
 const size_t k,
 const typename Field::Element alpha,
 const typename Field::Element beta,
 const size_t ldc,
 const size_t lda,
 enum FFLAS_TRANSPOSE ta,
 const size_t ldb,
 enum FFLAS_TRANSPOSE tb,
 size_t iters,
 int nbw,
 bool par,
 RandIter & G)

```

### 13.269.2.3 launch\_MM\_dispatch()

```

template<class Field , class RandIter >
bool launch_MM_dispatch (
 const Field & F,
 const int mm,
 const int nn,
 const int kk,
 const typename Field::Element alpha,
 const typename Field::Element beta,
 const size_t iters,
 const int nbw,
 const bool par,
 RandIter & G)

```

**Bug** test for ldX equal

**Bug**

**Bug** test for transpo

**Bug**

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 13.269.2.4 run\_with\_field()

```

template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 int m,
 int n,

```

```

 int k,
 int nbw,
 size_t iters,
 bool par,
 size_t seed)

```

### 13.269.2.5 main()

```

int main (
 int argc,
 char ** argv)

```

## 13.270 test-fgemv.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"

```

### Functions

- `template<class Field >`  
`bool check_MV (const Field &F, const typename Field::Element_ptr Cd, enum FFLAS_TRANSPOSE &ta, const size_t m, const size_t k, const typename Field::Element &alpha, const typename Field::Element_ptr A, size_t lda, const typename Field::Element_ptr X, size_t incX, const typename Field::Element &beta, const typename Field::Element_ptr Y, size_t incY)`
- `template<class Field, class RandIter >`  
`bool launch_MV (const Field &F, const size_t m, const size_t k, const typename Field::Element alpha, const typename Field::Element beta, const size_t lda, enum FFLAS_TRANSPOSE ta, const size_t incX, const size_t incY, size_t iters, bool par, RandIter &G)`
- `template<class Field, class RandIter >`  
`bool launch_MV_dispatch (const Field &F, const int mm, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, const bool par, RandIter &G)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, int m, int k, size_t iters, bool par, uint64_t seed)`
- `int main (int argc, char **argv)`

## 13.270.1 Function Documentation

### 13.270.1.1 check\_MV()

```

template<class Field >
bool check_MV (
 const Field & F,
 const typename Field::Element_ptr Cd,
 enum FFLAS_TRANSPOSE & ta,
 const size_t m,
 const size_t k,
 const typename Field::Element & alpha,
 const typename Field::Element_ptr A,
 size_t lda,
 const typename Field::Element_ptr X,

```

```

size_t incX,
const typename Field::Element & beta,
const typename Field::Element_ptr Y,
size_t incY)

```

### 13.270.1.2 launch\_MV()

```

template<class Field , class RandIter >
bool launch_MV (
 const Field & F,
 const size_t m,
 const size_t k,
 const typename Field::Element alpha,
 const typename Field::Element beta,
 const size_t lda,
 enum FFLAS_TRANSPOSE ta,
 const size_t incX,
 const size_t incY,
 size_t iters,
 bool par,
 RandIter & G)

```

### 13.270.1.3 launch\_MV\_dispatch()

```

template<class Field , class RandIter >
bool launch_MV_dispatch (
 const Field & F,
 const int mm,
 const int kk,
 const typename Field::Element alpha,
 const typename Field::Element beta,
 const size_t iters,
 const bool par,
 RandIter & G)

```

### 13.270.1.4 run\_with\_field()

```

template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 int m,
 int k,
 size_t iters,
 bool par,
 uint64_t seed)

```

### 13.270.1.5 main()

```

int main (
 int argc,
 char ** argv)

```

## 13.271 test-fger.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>

```

```
#include <givaro/modular-integral.h>
#include <givaro/modular-balanced.h>
#include <givaro/givintprime.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

## Macros

- `#define TIME 1`

## Functions

- `template<class Field >`  
`bool check_fger (const Field &F, const typename Field::Element_ptr Cd, const size_t m, const size_t n, const`  
`typename Field::Element &alpha, const typename Field::Element_ptr x, const size_t incx, const typename`  
`Field::Element_ptr y, const size_t incy, const typename Field::Element_ptr C, const size_t ldc)`
- `template<class Field , class RandIter >`  
`bool launch_fger (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, const`  
`size_t ldc, const size_t inca, const size_t incb, size_t iters, RandIter &G)`
- `template<class Field , class RandIter >`  
`bool launch_fger_dispatch (const Field &F, const size_t nn, const typename Field::Element alpha, const`  
`size_t iters, RandIter &G)`
- `template<class Field >`  
`bool run_with_field (int64_t q, uint64_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 13.271.1 Macro Definition Documentation

### 13.271.1.1 TIME

```
#define TIME 1
```

## 13.271.2 Function Documentation

### 13.271.2.1 check\_fger()

```
template<class Field >
bool check_fger (
 const Field & F,
 const typename Field::Element_ptr Cd,
 const size_t m,
 const size_t n,
 const typename Field::Element & alpha,
 const typename Field::Element_ptr x,
 const size_t incx,
 const typename Field::Element_ptr y,
 const size_t incy,
 const typename Field::Element_ptr C,
 const size_t ldc)
```

### 13.271.2.2 launch\_fger()

```
template<class Field , class RandIter >
bool launch_fger (
 const Field & F,
```

```

 const size_t m,
 const size_t n,
 const typename Field::Element alpha,
 const size_t ldc,
 const size_t inca,
 const size_t incb,
 size_t iters,
 RandIter & G)

```

### 13.271.2.3 launch\_fger\_dispatch()

```

template<class Field , class RandIter >
bool launch_fger_dispatch (
 const Field & F,
 const size_t nn,
 const typename Field::Element alpha,
 const size_t iters,
 RandIter & G)

```

**Bug** test for incx equal

**Bug**

**Bug** test for transpo

**Bug**

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 13.271.2.4 run\_with\_field()

```

template<class Field >
bool run_with_field (
 int64_t q,
 uint64_t b,
 size_t n,
 size_t iters,
 uint64_t seed)

```

### 13.271.2.5 main()

```

int main (
 int argc,
 char ** argv)

```

## 13.272 test-fgesv.C File Reference

```

#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>

```



## Functions

- template<class [Field](#) , class RandIter >  
bool [test\\_square\\_fgesv](#) ([Field](#) &F, [FFLAS\\_SIDE](#) side, string fileA, string fileB, size\_t m, size\_t k, size\_t r, RandIter &G)
- template<class [Field](#) , class RandIter >  
bool [test\\_rect\\_fgesv](#) ([Field](#) &F, [FFLAS\\_SIDE](#) side, string fileA, string fileB, size\_t m, size\_t n, size\_t k, size\_t r, RandIter &G)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, size\_t m, size\_t n, size\_t k, size\_t r, size\_t iters, string fileA, string fileB, [uint64\\_t](#) &seed)
- int [main](#) (int argc, char \*\*argv)

## 13.272.1 Function Documentation

### 13.272.1.1 test\_square\_fgesv()

```
template<class Field , class RandIter >
bool test_square_fgesv (
 Field & F,
 FFLAS_SIDE side,
 string fileA,
 string fileB,
 size_t m,
 size_t k,
 size_t r,
 RandIter & G)
```

### 13.272.1.2 test\_rect\_fgesv()

```
template<class Field , class RandIter >
bool test_rect_fgesv (
 Field & F,
 FFLAS_SIDE side,
 string fileA,
 string fileB,
 size_t m,
 size_t n,
 size_t k,
 size_t r,
 RandIter & G)
```

### 13.272.1.3 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 size_t m,
 size_t n,
 size_t k,
 size_t r,
 size_t iters,
 string fileA,
 string fileB,
 uint64_t & seed)
```

### 13.272.1.4 main()

```
int main (
```

```
int argc,
char ** argv)
```

## 13.273 test-finit.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
#include <random>
#include <chrono>
```

### Functions

- template<class [Field](#) >  
bool [test\\_freduce](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t m, size\_t k, size\_t n, size\_t iters, bool timing, [uint64\\_t](#) seed)
- int [main](#) (int ac, char \*\*av)

## 13.273.1 Function Documentation

### 13.273.1.1 test\_freduce()

```
template<class Field >
bool test_freduce (
 const Field & F,
 size_t m,
 size_t k,
 size_t n,
 bool timing,
 uint64_t seed)
```

### 13.273.1.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t m,
 size_t k,
 size_t n,
 size_t iters,
 bool timing,
 uint64_t seed)
```

### 13.273.1.3 main()

```
int main (
 int ac,
 char ** av)
```

## 13.274 test-fscal.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
```

### Functions

- template<class [Field](#) , class RandIter >  
bool [test\\_fscal](#) (const [Field](#) &F, const typename [Field::Element](#) &alpha, size\_t m, size\_t k, size\_t n, bool timing, RandIter &G)
- template<class [Field](#) >  
bool [test\\_fscal](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class [Field](#) , class RandIter >  
bool [test\\_fscalin](#) (const [Field](#) &F, const typename [Field::Element](#) &alpha, size\_t m, size\_t k, size\_t n, bool timing, RandIter &G)
- template<class [Field](#) >  
bool [test\\_fscalin](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- int [main](#) (int ac, char \*\*av)

### 13.274.1 Function Documentation

#### 13.274.1.1 [test\\_fscal\(\)](#) [1/2]

```
template<class Field , class RandIter >
bool test_fscal (
 const Field & F,
 const typename Field::Element & alpha,
 size_t m,
 size_t k,
 size_t n,
 bool timing,
 RandIter & G)
```

#### 13.274.1.2 [test\\_fscal\(\)](#) [2/2]

```
template<class Field >
bool test_fscal (
 const Field & F,
 size_t m,
 size_t k,
 size_t n,
 bool timing,
 uint64_t seed)
```

#### 13.274.1.3 [test\\_fscalin\(\)](#) [1/2]

```
template<class Field , class RandIter >
bool test_fscalin (
 const Field & F,
 const typename Field::Element & alpha,
 size_t m,
 size_t k,
```

```

 size_t n,
 bool timing,
 RandIter & G)

```

#### 13.274.1.4 test\_fscalin() [2/2]

```

template<class Field >
bool test_fscalin (
 const Field & F,
 size_t m,
 size_t k,
 size_t n,
 bool timing,
 uint64_t seed)

```

#### 13.274.1.5 main()

```

int main (
 int ac,
 char ** av)

```

## 13.275 test-fsyr2k.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>

```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- template<typename [Field](#) , class RandIter >  
 bool [check\\_fsyr2k](#) (const [Field](#) &F, size\_t n, size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element](#) &beta, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, RandIter &Rand)
- template<class [Field](#) >  
 bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t k, int a, int c, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 13.275.1 Macro Definition Documentation

### 13.275.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.275.2 Function Documentation

### 13.275.2.1 check\_fsyr2k()

```

template<typename Field , class RandIter >
bool check_fsyr2k (

```

```

 const Field & F,
 size_t n,
 size_t k,
 const typename Field::Element & alpha,
 const typename Field::Element & beta,
 FFLAS::FFLAS_UPLO uplo,
 FFLAS::FFLAS_TRANSPOSE trans,
 RandIter & Rand)

```

### 13.275.2.2 run\_with\_field()

```

template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t n,
 size_t k,
 int a,
 int c,
 size_t iters,
 uint64_t seed)

```

### 13.275.2.3 main()

```

int main (
 int argc,
 char ** argv)

```

## 13.276 test-fsyrc.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>

```

### Macros

- #define `ENABLE_ALL_CHECKINGS` 1

### Functions

- template<typename `Field` , class `RandIter` >  
bool `check_fsyrc` (const `Field` &F, size\_t n, size\_t k, const typename `Field::Element` &alpha, const typename `Field::Element` &beta, `FFLAS::FFLAS_UPLO` uplo, `FFLAS::FFLAS_TRANSPOSE` trans, `RandIter` &Rand)
- template<typename `Field` , class `RandIter` >  
bool `check_fsyrc_diag` (const `Field` &F, size\_t n, size\_t k, const typename `Field::Element` &alpha, const typename `Field::Element` &beta, `FFLAS::FFLAS_UPLO` uplo, `FFLAS::FFLAS_TRANSPOSE` trans, `RandIter` &Rand)
- template<typename `Field` , class `RandIter` >  
bool `check_fsyrc_bkdiag` (const `Field` &F, size\_t n, size\_t k, const typename `Field::Element` &alpha, const

typename `Field::Element` &beta, `FFLAS::FFLAS_UPLO` uplo, `FFLAS::FFLAS_TRANSPOSE` trans, RandIter &Rand)

- template<class `Field` >  
bool `run_with_field` (Givaro::Integer q, size\_t b, size\_t n, size\_t k, int a, int c, size\_t iters, uint64\_t seed)
- int `main` (int argc, char \*\*argv)

## 13.276.1 Macro Definition Documentation

### 13.276.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.276.2 Function Documentation

### 13.276.2.1 check\_fsyrk()

```
template<typename Field , class RandIter >
bool check_fsyrk (
 const Field & F,
 size_t n,
 size_t k,
 const typename Field::Element & alpha,
 const typename Field::Element & beta,
 FFLAS::FFLAS_UPLO uplo,
 FFLAS::FFLAS_TRANSPOSE trans,
 RandIter & Rand)
```

### 13.276.2.2 check\_fsyrk\_diag()

```
template<typename Field , class RandIter >
bool check_fsyrk_diag (
 const Field & F,
 size_t n,
 size_t k,
 const typename Field::Element & alpha,
 const typename Field::Element & beta,
 FFLAS::FFLAS_UPLO uplo,
 FFLAS::FFLAS_TRANSPOSE trans,
 RandIter & Rand)
```

### 13.276.2.3 check\_fsyrk\_bkdiag()

```
template<typename Field , class RandIter >
bool check_fsyrk_bkdiag (
 const Field & F,
 size_t n,
 size_t k,
 const typename Field::Element & alpha,
 const typename Field::Element & beta,
 FFLAS::FFLAS_UPLO uplo,
 FFLAS::FFLAS_TRANSPOSE trans,
 RandIter & Rand)
```

### 13.276.2.4 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
```

```

 size_t n,
 size_t k,
 int a,
 int c,
 size_t iters,
 uint64_t seed)

```

### 13.276.2.5 main()

```

int main (
 int argc,
 char ** argv)

```

## 13.277 test-fsytrf.C File Reference

```

#include <iostream>
#include <iterator>
#include <vector>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <iomanip>
#include <random>
#include <chrono>
#include <givaro/modular.h>
#include "fflas-ffpack/utils/test-utils.h"

```

### Functions

- `template<typename T > std::ostream & operator<< (std::ostream &os, const std::vector< T > &x)`
- `template<class Field , class RandIter > bool test_RPM_fsytrf (Field &F, FFLAS_UPLO uplo, string file, size_t n, size_t r, RandIter &G, size_t threshold)`
- `template<class Field , class RandIter > bool test_generic_fsytrf (Field &F, FFLAS_UPLO uplo, string file, size_t n, RandIter &G, size_t threshold)`
- `template<class Field > bool run_with_field (Givaro::Integer q, uint64_t b, size_t n, size_t r, size_t iters, string file, size_t threshold, uint64_t &seed)`
- `int main (int argc, char **argv)`

## 13.277.1 Function Documentation

### 13.277.1.1 operator<<()

```

template<typename T >
std::ostream & operator<< (
 std::ostream & os,
 const std::vector< T > & x)

```

### 13.277.1.2 test\_RPM\_fsytrf()

```

template<class Field , class RandIter >
bool test_RPM_fsytrf (
 Field & F,
 FFLAS_UPLO uplo,
 string file,

```

```

 size_t n,
 size_t r,
 RandIter & G,
 size_t threshold)

```

### 13.277.1.3 test\_generic\_fsytrf()

```

template<class Field , class RandIter >
bool test_generic_fsytrf (
 Field & F,
 FFLAS_UPLO uplo,
 string file,
 size_t n,
 RandIter & G,
 size_t threshold)

```

### 13.277.1.4 run\_with\_field()

```

template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 size_t n,
 size_t r,
 size_t iters,
 string file,
 size_t threshold,
 uint64_t & seed)

```

### 13.277.1.5 main()

```

int main (
 int argc,
 char ** argv)

```

## 13.278 test-fftrmm.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>

```

### Macros

- #define `__FFLASFFPACK_SEQUENTIAL`

### Functions

- template<typename `Field` , class `RandIter` >  
 bool `check_fftrmm` (const `Field` &F, size\_t m, size\_t n, const typename `Field::Element` &alpha,



`FFLAS::FFLAS_SIDE` side, `FFLAS::FFLAS_UPLO` uplo, `FFLAS::FFLAS_TRANSPOSE` trans, `FFLAS::FFLAS_DIAG` diag, `RandIter` & `Rand`)

- `template<class Field >`  
`bool run_with_field` (Givaro::Integer q, size\_t b, size\_t m, size\_t n, `uint64_t` a, size\_t iters, `uint64_t` seed)
- `int main` (int argc, char \*\*argv)

## 13.278.1 Macro Definition Documentation

### 13.278.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

## 13.278.2 Function Documentation

### 13.278.2.1 check\_ffrmv()

```
template<typename Field , class RandIter >
bool check_ffrmv (
 const Field & F,
 size_t m,
 size_t n,
 const typename Field::Element & alpha,
 FFLAS::FFLAS_SIDE side,
 FFLAS::FFLAS_UPLO uplo,
 FFLAS::FFLAS_TRANSPOSE trans,
 FFLAS::FFLAS_DIAG diag,
 RandIter & Rand)
```

### 13.278.2.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t m,
 size_t n,
 uint64_t a,
 size_t iters,
 uint64_t seed)
```

### 13.278.2.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.279 test-ffrmv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <chrono>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define ENABLE_ALL_CHECKINGS 1`

## Functions

- `template<typename Field , class RandIter >`  
`bool check_ftrmv (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_TRANSPOSE trans, FFLAS_DIAG diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 13.279.1 Macro Definition Documentation

### 13.279.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 13.279.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.279.2 Function Documentation

### 13.279.2.1 check\_ftrmv()

```
template<typename Field , class RandIter >
bool check_ftrmv (
 const Field & F,
 size_t n,
 FFLAS_UPLO uplo,
 FFLAS_TRANSPOSE trans,
 FFLAS_DIAG diag,
 RandIter & Rand)
```

### 13.279.2.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t n,
 size_t iters,
 uint64_t seed)
```

### 13.279.2.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.280 test-ftsm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
```

```
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- int [main](#) (int argc, char \*\*argv)

## 13.280.1 Macro Definition Documentation

### 13.280.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.280.2 Function Documentation

### 13.280.2.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.281 test-fftrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

## Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- template<typename [Field](#) , class RandIter >  
bool [check\\_fftrsm](#) (const [Field](#) &F, size\_t m, size\_t n, const typename [Field::Element](#) &alpha, [FFLAS::FFLAS\\_SIDE](#) side, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, [FFLAS::FFLAS\\_DIAG](#) diag, RandIter &Rand)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t m, size\_t n, [uint64\\_t](#) a, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 13.281.1 Macro Definition Documentation

### 13.281.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 13.281.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.281.2 Function Documentation

### 13.281.2.1 check\_ftrsm()

```
template<typename Field , class RandIter >
bool check_ftrsm (
 const Field & F,
 size_t m,
 size_t n,
 const typename Field::Element & alpha,
 FFLAS::FFLAS_SIDE side,
 FFLAS::FFLAS_UPLO uplo,
 FFLAS::FFLAS_TRANSPOSE trans,
 FFLAS::FFLAS_DIAG diag,
 RandIter & Rand)
```

### 13.281.2.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t m,
 size_t n,
 uint64_t a,
 size_t iters,
 uint64_t seed)
```

### 13.281.2.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.282 test-ffrssyr2k.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

## Macros

- `#define ENABLE_ALL_CHECKINGS 1`

## Functions

- `template<typename Field, class RandIter >`  
`bool check_ffrssyr2k (const Field &F, size_t n, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_DIAG diagA,`  
`RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 13.282.1 Macro Definition Documentation

### 13.282.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.282.2 Function Documentation

### 13.282.2.1 check\_ffrssyr2k()

```
template<typename Field, class RandIter >
bool check_ffrssyr2k (
 const Field & F,
 size_t n,
 FFLAS::FFLAS_UPLO uplo,
 FFLAS::FFLAS_DIAG diagA,
 RandIter & Rand)
```

### 13.282.2.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t n,
 size_t iters,
 uint64_t seed)
```

### 13.282.2.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.283 test-ffrstr.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

```
#include <givaro/modular-balanced.h>
#include <random>
```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- template<typename [Field](#) , class RandIter >  
bool [check\\_ftrstr](#) (const [Field](#) &F, size\_t n, [FFLAS::FFLAS\\_SIDE](#) side, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_DIAG](#) diagA, [FFLAS::FFLAS\\_DIAG](#) diagB, RandIter &Rand)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 13.283.1 Macro Definition Documentation

### 13.283.1.1 [ENABLE\\_ALL\\_CHECKINGS](#)

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.283.2 Function Documentation

### 13.283.2.1 [check\\_ftrstr\(\)](#)

```
template<typename Field , class RandIter >
bool check_ftrstr (
 const Field & F,
 size_t n,
 FFLAS::FFLAS_SIDE side,
 FFLAS::FFLAS_UPLO uplo,
 FFLAS::FFLAS_DIAG diagA,
 FFLAS::FFLAS_DIAG diagB,
 RandIter & Rand)
```

### 13.283.2.2 [run\\_with\\_field\(\)](#)

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t n,
 size_t iters,
 uint64_t seed)
```

### 13.283.2.3 [main\(\)](#)

```
int main (
 int argc,
 char ** argv)
```

## 13.284 test-ftrsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
```

```
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define ENABLE_ALL_CHECKINGS 1`

## Functions

- `template<typename Field , class RandIter >`  
`bool check_ffrsv (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_TRANSPOSE trans, FFLAS_DIAG diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 13.284.1 Macro Definition Documentation

### 13.284.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 13.284.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.284.2 Function Documentation

### 13.284.2.1 check\_ffrsv()

```
template<typename Field , class RandIter >
bool check_ffrsv (
 const Field & F,
 size_t n,
 FFLAS_UPLO uplo,
 FFLAS_TRANSPOSE trans,
 FFLAS_DIAG diag,
 RandIter & Rand)
```

### 13.284.2.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t n,
 size_t iters,
 uint64_t seed)
```

### 13.284.2.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.285 test-fftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- template<typename [Field](#) , class RandIter >  
bool [check\\_fftrtri](#) (const [Field](#) &F, size\_t n, [FFLAS\\_UPLO](#) uplo, [FFLAS\\_DIAG](#) diag, RandIter &Rand)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 13.285.1 Macro Definition Documentation

### 13.285.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 13.285.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 13.285.2 Function Documentation

### 13.285.2.1 check\_fftrtri()

```
template<typename Field , class RandIter >
bool check_fftrtri (
 const Field & F,
 size_t n,
 FFLAS_UPLO uplo,
 FFLAS_DIAG diag,
 RandIter & Rand)
```

### 13.285.2.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t n,
 size_t iters,
 uint64_t seed)
```



### 13.285.2.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.286 test-interfaces-c.c File Reference

```
#include <fflas-ffpack/interfaces/libs/fflas_c.h>
#include <fflas-ffpack/interfaces/libs/ffpack_c.h>
#include <stdlib.h>
#include <stdio.h>
```

### Functions

- int [main](#) ()

### 13.286.1 Function Documentation

#### 13.286.1.1 main()

```
int main (
 void)
```

## 13.287 test-invert-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- int [main](#) (int argc, char \*\*argv)

### 13.287.1 Macro Definition Documentation

#### 13.287.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

### 13.287.2 Function Documentation

#### 13.287.2.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.288 test-io.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <random>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Data Structures

- struct [CompactElement< Element >](#)
- struct [CompactElement< double >](#)
- struct [CompactElement< float >](#)
- struct [CompactElement< int64\\_t >](#)
- struct [CompactElement< int32\\_t >](#)
- struct [CompactElement< int16\\_t >](#)

### Functions

- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, size\_t m, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 13.288.1 Function Documentation

### 13.288.1.1 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 size_t m,
 size_t n,
 size_t iters,
 uint64_t seed)
```

### 13.288.1.2 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.289 test-lu.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-balanced.h>
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <random>
```

## Macros

- #define `BASECASE_K` 37
- #define `__FFLASFFPACK_SEQUENTIAL`
- #define `__LUDIVINE_CUTOFF` 1

## Functions

- template<class `Field` , `FFLAS::FFLAS_DIAG` diag, `FFLAS_TRANSPOSE` trans>  
 bool `test_LUdivine` (const `Field` &F, typename `Field::ConstElement_ptr` A, size\_t lda, size\_t r, size\_t m, size\_t n)  
*Tests the LUdivine routine.*
- template<class `Field` , `FFLAS_DIAG` diag>  
 bool `verifPLUQ` (const `Field` &F, typename `Field::ConstElement_ptr` A, size\_t lda, typename `Field::Element_ptr` PLUQ, size\_t ldpluq, size\_t \*P, size\_t \*Q, size\_t m, size\_t n, size\_t R)  
*Verifies that  $B = PLUQ$  where A stores  $[L|U]$ .*
- template<class `Field` , `FFLAS_DIAG` diag, class `Randlter` >  
 bool `test_pluq` (const `Field` &F, typename `Field::ConstElement_ptr` A, size\_t r, size\_t m, size\_t n, size\_t lda, `Randlter` &G)  
*Tests the LUdivine routine.*
- template<class `Field` , `FFLAS_DIAG` diag, `FFLAS_TRANSPOSE` trans, class `Randlter` >  
 bool `launch_test` (const `Field` &F, size\_t r, size\_t m, size\_t n, `Randlter` &G)
- template<class `Field` >  
 bool `run_with_field` (`Givaro::Integer` q, `uint64_t` b, size\_t m, size\_t n, size\_t r, size\_t iters, `uint64_t` seed)
- int `main` (int argc, char \*\*argv)

## Variables

- `Givaro::Timer` `tperm`
- `Givaro::Timer` `tgemm`
- `Givaro::Timer` `tBC`
- `Givaro::Timer` `ttrsm`
- `Givaro::Timer` `trest`
- `Givaro::Timer` `timtot`
- size\_t `mvcnt` = 0

## 13.289.1 Macro Definition Documentation

### 13.289.1.1 BASECASE\_K

```
#define BASECASE_K 37
```

### 13.289.1.2 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 13.289.1.3 \_\_LUDIVINE\_CUTOFF

```
#define __LUDIVINE_CUTOFF 1
```

## 13.289.2 Function Documentation

### 13.289.2.1 test\_LUdivine()

```
template<class Field , FFLAS::FFLAS_DIAG diag, FFLAS_TRANSPOSE trans>
bool test_LUdivine (
 const Field & F,
 typename Field::ConstElement_ptr A,
```

```

 size_t lda,
 size_t r,
 size_t m,
 size_t n)

```

Tests the LUdivine routine.

#### Template Parameters

|              |                    |
|--------------|--------------------|
| <i>Field</i> | Field              |
| <i>Diag</i>  | Unit diagonal in U |
| <i>Trans</i> |                    |

#### Parameters

|            |                       |
|------------|-----------------------|
| <i>F</i>   | field                 |
| <i>A</i>   | Matrix (preallocated) |
| <i>r</i>   | rank of A             |
| <i>m</i>   | rows                  |
| <i>n</i>   | cols                  |
| <i>lda</i> | leading dim of A      |

#### Returns

0 iff correct, 1 otherwise

### 13.289.2.2 **verifPLUQ()**

```

template<class Field , FFLAS_DIAG diag>
bool verificPLUQ (
 const Field & F,
 typename Field::ConstElement_ptr A,
 size_t lda,
 typename Field::Element_ptr PLUQ,
 size_t ldpluq,
 size_t * P,
 size_t * Q,
 size_t m,
 size_t n,
 size_t R)

```

Verifies that  $B = PLUQ$  where A stores  $[L|U]$ .

#### Template Parameters

|              |                    |
|--------------|--------------------|
| <i>Field</i> | Field              |
| <i>Diag</i>  | Unit diagonal in U |

#### Parameters

|            |                       |
|------------|-----------------------|
| <i>F</i>   | field                 |
| <i>A</i>   | Matrix (preallocated) |
| <i>r</i>   | rank of A             |
| <i>m</i>   | rows                  |
| <i>n</i>   | cols                  |
| <i>lda</i> | leading dim of A      |

**Returns**

0 iff correct, 1 otherwise

**13.289.2.3 test\_pluq()**

```
template<class Field , FFLAS_DIAG diag, class RandIter >
bool test_pluq (
 const Field & F,
 typename Field::ConstElement_ptr A,
 size_t r,
 size_t m,
 size_t n,
 size_t lda,
 RandIter & G)
```

Tests the LUdivine routine.

**Template Parameters**

|              |                    |
|--------------|--------------------|
| <i>Field</i> | Field              |
| <i>Diag</i>  | Unit diagonal in U |
| <i>Trans</i> |                    |

**Parameters**

|            |                       |
|------------|-----------------------|
| <i>F</i>   | field                 |
| <i>A</i>   | Matrix (preallocated) |
| <i>r</i>   | rank of A             |
| <i>m</i>   | rows                  |
| <i>n</i>   | cols                  |
| <i>lda</i> | leading dim of A      |

**Returns**

0 iff correct, 1 otherwise

**13.289.2.4 launch\_test()**

```
template<class Field , FFLAS_DIAG diag, FFLAS_TRANSPOSE trans, class RandIter >
bool launch_test (
 const Field & F,
 size_t r,
 size_t m,
 size_t n,
 RandIter & G)
```

**13.289.2.5 run\_with\_field()**

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 size_t m,
 size_t n,
 size_t r,
 size_t iters,
 uint64_t seed)
```

**13.289.2.6 main()**

```
int main (
 int argc,
 char ** argv)
```

**13.289.3 Variable Documentation****13.289.3.1 tperm**

```
Givaro::Timer tperm
```

**13.289.3.2 tgemm**

```
Givaro::Timer tgemm
```

**13.289.3.3 tBC**

```
Givaro::Timer tBC
```

**13.289.3.4 ttrsm**

```
Givaro::Timer ttrsm
```

**13.289.3.5 trest**

```
Givaro::Timer trest
```

**13.289.3.6 timtot**

```
Givaro::Timer timtot
```

**13.289.3.7 mvcnt**

```
size_t mvcnt = 0
```

**13.290 test-maxdelayeddim.C File Reference**

```
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include <stdlib.h>
#include <stdio.h>
```

**Macros**

- `#define MAX_WITH_SIZE_T(x) ( (static_cast<uint64_t>(std::numeric_limits<size_t>::max()) < x)? std::numeric_limits<size_t>::max() : x )`

**Functions**

- `template<class Field >`  
`bool test (Givaro::Integer p, size_t kmax)`
- `int main ()`

## 13.290.1 Macro Definition Documentation

### 13.290.1.1 MAX\_WITH\_SIZE\_T

```
#define MAX_WITH_SIZE_T(
 x) ((static_cast<uint64_t>(std::numeric_limits<size_t>::max()) < x)? std::numeric_limits<size_t>::max() : x)
```

## 13.290.2 Function Documentation

### 13.290.2.1 test()

```
template<class Field >
bool test (
 Givaro::Integer p,
 size_t kmax)
```

### 13.290.2.2 main()

```
int main (
 void)
```

## 13.291 test-minpoly.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include <chrono>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utlis/args-parser.h"
#include "fflas-ffpack/utlis/fflas_randommatrix.h"
#include "fflas-ffpack/utlis/test-utlis.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/modular-integer.h>
#include <givaro/givpoly1factor.h>
#include <givaro/givpoly1.h>
```

## Functions

- template<typename [Field](#) , class RandIter >  
bool [check\\_minpoly](#) (const [Field](#) &F, size\_t n, RandIter &G)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 13.291.1 Function Documentation

### 13.291.1.1 check\_minpoly()

```
template<typename Field , class RandIter >
bool check_minpoly (
 const Field & F,
 size_t n,
 RandIter & G)
```

**13.291.1.2 run\_with\_field()**

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t n,
 size_t iters,
 uint64_t seed)
```

**13.291.1.3 main()**

```
int main (
 int argc,
 char ** argv)
```

**13.292 test-multifile1.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack.h"
```

**13.293 test-multifile2.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack.h"
```

**Functions**

- int [main](#) (void)

**13.293.1 Function Documentation****13.293.1.1 main()**

```
int main (
 void)
```

**13.294 test-nullspace.C File Reference**

```
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/timer.h"
```

**Functions**

- template<class [Field](#) >  
std::string [checkingMessage](#) (const [Field](#) &F)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [readOrRandomMatrixWithRankAndRandomRPM](#) (const [Field](#) &F, std::string file, size\_t [m](#), size\_t [n](#), size\_t [lda](#), size\_t [r](#), [uint64\\_t](#) seed)  
*If file is not empty, read it and set m, n, lda and r.*



- `template<class Field >`  
`bool test_nullspace (Field &F, FFLAS::FFLAS_SIDE side, size_t m, size_t n, size_t r, typename Field::Element_ptr A, size_t lda)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, std::string file, uint64_t &seed)`
- `int main (int argc, char **argv)`

## 13.294.1 Function Documentation

### 13.294.1.1 checkingMessage()

```
template<class Field >
std::string checkingMessage (
 const Field & F)
```

### 13.294.1.2 readOrRandomMatrixWithRankAndRandomRPM()

```
template<class Field >
Field::Element_ptr readOrRandomMatrixWithRankAndRandomRPM (
 const Field & F,
 std::string file,
 size_t & m,
 size_t & n,
 size_t & lda,
 size_t & r,
 uint64_t seed)
```

If file is not empty, read it and set m, n, lda and r.

Otherwise, generate a random matrix of size m x n with random lda.

### 13.294.1.3 test\_nullspace()

```
template<class Field >
bool test_nullspace (
 Field & F,
 FFLAS::FFLAS_SIDE side,
 size_t m,
 size_t n,
 size_t r,
 typename Field::Element_ptr A,
 size_t lda)
```

### 13.294.1.4 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 size_t m,
 size_t n,
 size_t r,
 size_t iters,
 std::string file,
 uint64_t & seed)
```

### 13.294.1.5 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.295 test-permutations.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Functions

- bool [checkMonotonicApplyP](#) (FFLAS\_SIDE Side, FFLAS\_TRANSPOSE trans, size\_t \*P, size\_t N, size\_t R)
- int [main](#) ()

### Variables

- Givaro::Timer [tperm](#)
- Givaro::Timer [tgemm](#)
- Givaro::Timer [tBC](#)
- Givaro::Timer [ttrsm](#)
- Givaro::Timer [trest](#)
- Givaro::Timer [timtot](#)

### 13.295.1 Function Documentation

#### 13.295.1.1 checkMonotonicApplyP()

```
bool checkMonotonicApplyP (
 FFLAS_SIDE Side,
 FFLAS_TRANSPOSE trans,
 size_t * P,
 size_t N,
 size_t R)
```

#### 13.295.1.2 main()

```
int main (
 void)
```

### 13.295.2 Variable Documentation

#### 13.295.2.1 tperm

```
Givaro::Timer tperm
```

#### 13.295.2.2 tgemm

```
Givaro::Timer tgemm
```

#### 13.295.2.3 tBC

```
Givaro::Timer tBC
```

#### 13.295.2.4 ttrsm

```
Givaro::Timer ttrsm
```

#### 13.295.2.5 trest

```
Givaro::Timer trest
```

**13.295.2.6 timtot**

```
Givaro::Timer timtot
```

**13.296 test-pluq-check.C File Reference**

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

**Macros**

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

**Functions**

- `int` [main](#) (`int` argc, `char **`argv)

**13.296.1 Macro Definition Documentation****13.296.1.1 ENABLE\_ALL\_CHECKINGS**

```
#define ENABLE_ALL_CHECKINGS 1
```

**13.296.2 Function Documentation****13.296.2.1 main()**

```
int main (
 int argc,
 char ** argv)
```

**13.297 test-rankprofiles.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <iostream>
#include <iomanip>
#include <random>
#include <chrono>
```

**Macros**

- `#define` [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)

**Functions**

- `template<class` [Field](#) `>`  
`bool` [run\\_with\\_field](#) (`Givaro::Integer` q, [uint64\\_t](#) b, `size_t` m, `size_t` n, `size_t` r, `size_t` iters, [uint64\\_t](#) seed, `bool` par)
- `int` [main](#) (`int` argc, `char **`argv)

## 13.297.1 Macro Definition Documentation

### 13.297.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

## 13.297.2 Function Documentation

### 13.297.2.1 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 uint64_t b,
 size_t m,
 size_t n,
 size_t r,
 size_t iters,
 uint64_t seed,
 bool par)
```

### 13.297.2.2 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.298 test-rpm.C File Reference

```
#include <iostream>
#include "givaro/modular.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Functions

- bool [checkRPM](#) (size\_t M, size\_t N, size\_t R)
- bool [checkSymmetricRPM](#) (size\_t N, size\_t R)
- int [main](#) (int argc, char \*\*argv)

## 13.298.1 Function Documentation

### 13.298.1.1 checkRPM()

```
bool checkRPM (
 size_t M,
 size_t N,
 size_t R)
```

### 13.298.1.2 checkSymmetricRPM()

```
bool checkSymmetricRPM (
 size_t N,
 size_t R)
```

### 13.298.1.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.299 test-simd.C File Reference

```
#include "givaro/givinteger.h"
#include "givaro/givprint.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <array>
#include <vector>
#include <random>
#include <string>
#include <functional>
#include <limits>
#include <type_traits>
#include <algorithm>
```

### Data Structures

- struct [ScalFunctions](#)< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >
- struct [ScalFunctions](#)< Element, typename enable\_if< is\_integral< Element >::value >::type >

### Macros

- #define [REGISTER\\_TYPE\\_NAME](#)(type) template<> const char \*[TypeName](#)<type>(){return #type;}
- #define [TEST\\_ONE\\_OP](#)(name) btest &= [test\\_op](#)<simd> (simd::name, Scal::name, #name);

### Typedefs

- typedef Givaro::Integer [integer](#)

### Functions

- template<typename... >  
const char \* [TypeName](#) ()
- [REGISTER\\_TYPE\\_NAME](#) (float)
- [REGISTER\\_TYPE\\_NAME](#) (double)
- [REGISTER\\_TYPE\\_NAME](#) (int16\_t)
- [REGISTER\\_TYPE\\_NAME](#) (int32\_t)
- [REGISTER\\_TYPE\\_NAME](#) (int64\_t)
- [REGISTER\\_TYPE\\_NAME](#) (uint16\_t)
- [REGISTER\\_TYPE\\_NAME](#) (uint32\_t)
- [REGISTER\\_TYPE\\_NAME](#) (uint64\_t)
- template<class Element , class Alloc >  
enable\_if< is\_integral< Element >::value >::type [generate\\_random\\_vector](#) (vector< Element, Alloc > &a)
- template<class Element , class Alloc >  
enable\_if< is\_floating\_point< Element >::value >::type [generate\\_random\\_vector](#) (vector< Element, Alloc > &a)
- template<class Element >  
enable\_if< is\_integral< Element >::value, bool >::type [check\\_eq](#) (Element x, Element y)

- `template<class Element >`  
`enable_if< is_floating_point< Element >::value, bool >::type` `check_eq` (Element x, Element y)
- `template<class Ret , class T >`  
`Ret` `eval_func_on_array` (function< Ret()> f, array< T, 0 > arr)
- `template<class Ret , class T , class... TArgs>`  
`Ret` `eval_func_on_array` (function< Ret(T, TArgs...)> f, array< typename remove\_reference< T >::type, sizeof...(TArgs)+1 > &arr)
- `template<class Simd , class RScal , class... AScal, class RSimd , class... ASimd>`  
`enable_if< sizeof...(AScal)==sizeof...(ASimd), bool >::type` `test_op` (RSimd(&FSimd)(ASimd...), RScal(&FScal)(AScal...), string frame)
- `template<class simd , class Element >`  
`enable_if< is_floating_point< Element >::value, bool >::type` `test_impl` ()
- `template<class simd , class Element >`  
`enable_if< is_integral< Element >::value, bool >::type` `test_impl` ()
- `template<class Element >`  
`enable_if< is_integral< Element >::value, bool >::type` `test` ()
- `template<class Element >`  
`enable_if< is_floating_point< Element >::value, bool >::type` `test` ()
- `int` `main` (int argc, char \*argv[])

## 13.299.1 Macro Definition Documentation

### 13.299.1.1 REGISTER\_TYPE\_NAME

```
#define REGISTER_TYPE_NAME(
 type) template<> const char *TypeName<type>() {return #type;}
```

### 13.299.1.2 TEST\_ONE\_OP

```
#define TEST_ONE_OP(
 name) btest &= test_op<simd> (simd::name, Scal::name, #name);
```

## 13.299.2 Typedef Documentation

### 13.299.2.1 integer

```
typedef Givaro::Integer integer
```

## 13.299.3 Function Documentation

### 13.299.3.1 TypeName()

```
template<typename... >
const char * TypeName ()
```

### 13.299.3.2 REGISTER\_TYPE\_NAME() [1/8]

```
REGISTER_TYPE_NAME (
 float)
```

### 13.299.3.3 REGISTER\_TYPE\_NAME() [2/8]

```
REGISTER_TYPE_NAME (
 double)
```

### 13.299.3.4 REGISTER\_TYPE\_NAME() [3/8]

```
REGISTER_TYPE_NAME (
 int16_t)
```

**13.299.3.5 REGISTER\_TYPE\_NAME() [4/8]**

```
REGISTER_TYPE_NAME (
 int32_t)
```

**13.299.3.6 REGISTER\_TYPE\_NAME() [5/8]**

```
REGISTER_TYPE_NAME (
 int64_t)
```

**13.299.3.7 REGISTER\_TYPE\_NAME() [6/8]**

```
REGISTER_TYPE_NAME (
 uint16_t)
```

**13.299.3.8 REGISTER\_TYPE\_NAME() [7/8]**

```
REGISTER_TYPE_NAME (
 uint32_t)
```

**13.299.3.9 REGISTER\_TYPE\_NAME() [8/8]**

```
REGISTER_TYPE_NAME (
 uint64_t)
```

**13.299.3.10 generate\_random\_vector() [1/2]**

```
template<class Element , class Alloc >
enable_if< is_integral< Element >::value >::type generate_random_vector (
 vector< Element, Alloc > & a)
```

**13.299.3.11 generate\_random\_vector() [2/2]**

```
template<class Element , class Alloc >
enable_if< is_floating_point< Element >::value >::type generate_random_vector (
 vector< Element, Alloc > & a)
```

**13.299.3.12 check\_eq() [1/2]**

```
template<class Element >
enable_if< is_integral< Element >::value, bool >::type check_eq (
 Element x,
 Element y)
```

**13.299.3.13 check\_eq() [2/2]**

```
template<class Element >
enable_if< is_floating_point< Element >::value, bool >::type check_eq (
 Element x,
 Element y)
```

**13.299.3.14 eval\_func\_on\_array() [1/2]**

```
template<class Ret , class T >
Ret eval_func_on_array (
 function< Ret ()> f,
 array< T, 0 > arr)
```

**13.299.3.15 eval\_func\_on\_array() [2/2]**

```
template<class Ret , class T , class... TArgs>
Ret eval_func_on_array (
 function< Ret(T, TArgs...)> f,
 array< typename remove_reference< T >::type, sizeof...(TArgs)+1 > & arr)
```

**13.299.3.16 test\_op()**

```
template<class Simd , class RScal , class... AScal, class RSimd , class... ASimd>
enable_if< sizeof...(AScal)==sizeof...(ASimd), bool >::type test_op (
 RSimd(&)(ASimd...) FSimd,
 RScal(&)(AScal...) FScal,
 string fname)
```

**13.299.3.17 test\_impl() [1/2]**

```
template<class simd , class Element >
enable_if< is_floating_point< Element >::value, bool >::type test_impl ()
```

**13.299.3.18 test\_impl() [2/2]**

```
template<class simd , class Element >
enable_if< is_integral< Element >::value, bool >::type test_impl ()
```

**13.299.3.19 test() [1/2]**

```
template<class Element >
enable_if< is_integral< Element >::value, bool >::type test ()
```

**13.299.3.20 test() [2/2]**

```
template<class Element >
enable_if< is_floating_point< Element >::value, bool >::type test ()
```

**13.299.3.21 main()**

```
int main (
 int argc,
 char * argv[])
```

**13.300 test-solve.C File Reference**

```
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

**Functions**

- template<typename Field , class RandIter >  
bool [check\\_solve](#) (const Field &F, size\_t m, RandIter &Rand, bool isParallel)



- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

### 13.300.1 Function Documentation

#### 13.300.1.1 check\_solve()

```
template<typename Field , class RandIter >
bool check_solve (
 const Field & F,
 size_t m,
 RandIter & Rand,
 bool isParallel)
```

#### 13.300.1.2 run\_with\_field()

```
template<class Field >
bool run_with_field (
 Givaro::Integer q,
 size_t b,
 size_t m,
 size_t iters,
 uint64_t seed)
```

#### 13.300.1.3 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.301 101-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

### Functions

- `int main (int argc, char **argv)`

### 13.301.1 Function Documentation

#### 13.301.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.302 2x2-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
```

```
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.302.1 Function Documentation

#### 13.302.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

### 13.303 2x2-ftsv.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
#include <array>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.303.1 Function Documentation

#### 13.303.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

### 13.304 2x2-pluq.C File Reference

```
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.304.1 Function Documentation

#### 13.304.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.305 fflas-101\_1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 13.305.1 Function Documentation

#### 13.305.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.306 fflas-101\_3.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 13.306.1 Function Documentation

#### 13.306.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.307 fflas\_101.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.307.1 Function Documentation

#### 13.307.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.308 fflas\_101\_lvl1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.308.1 Function Documentation

#### 13.308.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.309 ffpack-fgesv.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <fflas-ffpack/ffpack/ffpack.h>
#include <iostream>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 13.309.1 Function Documentation

#### 13.309.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

## 13.310 ffpack-solve.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <fflas-ffpack/ffpack/ffpack.h>
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 13.310.1 Function Documentation

#### 13.310.1.1 main()

```
int main (
 int argc,
 char ** argv)
```

PS: the function Solve will modify the matrix A so here we used a duplicate matrix A2 otherwise  $A \cdot x$  will not be equal to b for the later verification stage



# Index

- [\\_F](#)
  - [RNSIntegerMod< RNS >, 516](#)
- [\\_M](#)
  - [rns\\_double, 495](#)
  - [rns\\_double\\_extended, 506](#)
- [\\_MAX\\_SIZE\\_MATRICES](#)
  - [benchmark-checkers.C, 717](#)
- [\\_MMi](#)
  - [rns\\_double, 495](#)
  - [rns\\_double\\_extended, 506](#)
- [\\_Mi](#)
  - [rns\\_double, 495](#)
  - [rns\\_double\\_extended, 506](#)
- [\\_Mi\\_modp\\_rns](#)
  - [RNSIntegerMod< RNS >, 516](#)
- [\\_NR\\_TESTS](#)
  - [benchmark-checkers.C, 717](#)
- [\\_PLUQ](#)
  - [FFPACK, 341](#)
- [\\_RNSdelayed](#)
  - [RNSIntegerMod< RNS >, 516](#)
- [\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 762](#)
- [\\_\\_FFLASFFPACK\\_CACHE\\_LINE\\_SIZE](#)
  - [fflas\\_sparse.h, 817](#)
- [\\_\\_FFLASFFPACK\\_CHARPOLY\\_Danilevskii\\_LUKrylov\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 762](#)
- [\\_\\_FFLASFFPACK\\_CHARPOLY\\_LUKrylov\\_ArithProg\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 762](#)
- [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)
  - [cblas.C, 978](#)
  - [clapack.C, 979](#)
  - [fblas.C, 980](#)
  - [lapack.C, 983](#)
- [\\_\\_FFLASFFPACK\\_DIMKPENALTY](#)
  - [fflas\\_pfgemm.inl, 804](#)
- [\\_\\_FFLASFFPACK\\_FORCE\\_SEQ](#)
  - [benchmark-charpoly-mp.C, 716](#)
- [\\_\\_FFLASFFPACK\\_FSYRK\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 762](#)
- [\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 762](#)
- [\\_\\_FFLASFFPACK\\_FTRSSYR2K\\_THRESHOLD](#)
  - [ffpack.h, 853](#)
- [\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#)
  - [ffpack.h, 853](#)
- [\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 762](#)
- [\\_\\_FFLASFFPACK\\_GAUSSJORDAN\\_BASECASE](#)
  - [ffpack\\_echelonforms.inl, 860](#)
  - [test-echelon.C, 989](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_BLAS](#)
  - [fflas-ffpack/config.h, 758](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CBLAS](#)
  - [cblas.C, 978](#)
  - [fflas-ffpack/config.h, 758](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CLAPACK](#)
  - [clapack.C, 979](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CXX11](#)
  - [fflas-ffpack/config.h, 758](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DGETRF](#)
  - [benchmark-dgetrf.C, 719](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DLFCN\\_H](#)
  - [fflas-ffpack/config.h, 758](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DTRTRI](#)
  - [benchmark-dtrtri.C, 722](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_FLOAT\\_H](#)
  - [fflas-ffpack/config.h, 758](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_INTTYPES\\_H](#)
  - [fflas-ffpack/config.h, 758](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LAPACK](#)
  - [clapack.C, 979](#)
  - [fflas-ffpack/config.h, 758](#)
  - [lapack.C, 983](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LIMITS\\_H](#)
  - [fflas-ffpack/config.h, 758](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LITTLE\\_ENDIAN](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_PTHREAD\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDDEF\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDINT\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDIO\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDLIB\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STRINGS\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STRING\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_STAT\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_TIME\\_H](#)
  - [fflas-ffpack/config.h, 759](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_TYPES\\_H](#)
  - [fflas-ffpack/config.h, 759](#)

- \_\_FFLASFFPACK\_HAVE\_UNISTD\_H
  - fflas-ffpack/config.h, [759](#)
- \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL
  - kaapi\_routines.inl, [960](#)
- \_\_FFLASFFPACK\_LT\_OBJDIR
  - fflas-ffpack/config.h, [759](#)
- \_\_FFLASFFPACK\_MINBLOCKCUTS
  - blockcuts.inl, [959](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET
  - benchmark-charpoly.C, [716](#)
  - benchmark-fadd-lvl2.C, [722](#)
  - benchmark-fdot.C, [723](#)
  - benchmark-fgemm-mp.C, [724](#)
  - benchmark-fgemm-rns.C, [725](#)
  - benchmark-fgemv-mp.C, [727](#)
  - benchmark-fgemv.C, [728](#)
  - benchmark-fgesv.C, [731](#)
  - benchmark-fsyrc.C, [732](#)
  - benchmark-fsytrf.C, [732](#)
  - benchmark-ftrsm-mp.C, [733](#)
  - benchmark-ftrsm.C, [734](#)
  - benchmark-ftrsv.C, [734](#)
  - benchmark-ftrtri.C, [735](#)
  - benchmark-pluq.C, [737](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS
  - fflas-ffpack/config.h, [759](#)
- \_\_FFLASFFPACK\_PACKAGE
  - fflas-ffpack/config.h, [759](#)
- \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_PACKAGE\_NAME
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_PACKAGE\_STRING
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_PACKAGE\_TARNAME
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_PACKAGE\_URL
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_PACKAGE\_VERSION
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_PLUQ\_THRESHOLD
  - fflas-ffpack-default-thresholds.h, [762](#)
  - test-echelon.C, [989](#)
- \_\_FFLASFFPACK\_SEQPARTHRESHOLD
  - fflas\_ptgemm.inl, [804](#)
- \_\_FFLASFFPACK\_SEQUENTIAL
  - parallel.h, [961](#)
  - test-echelon.C, [989](#)
  - test-ftrmm.C, [1009](#)
  - test-ftrmv.C, [1010](#)
  - test-ftrsm.C, [1012](#)
  - test-ftrsv.C, [1015](#)
  - test-ftrtri.C, [1016](#)
  - test-lu.C, [1019](#)
  - test-rankprofiles.C, [1028](#)
- \_\_FFLASFFPACK\_SIZEOF\_CHAR
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_SIZEOF\_INT
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_SIZEOF\_SHORT
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_SIZEOF\_\_INT64
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_STDC\_HEADERS
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_USE\_OPENMP
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_VERSION
  - fflas-ffpack/config.h, [760](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD
  - fflas-ffpack-default-thresholds.h, [761](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL
  - fflas-ffpack-default-thresholds.h, [761](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT
  - fflas-ffpack-default-thresholds.h, [761](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT
  - fflas-ffpack-default-thresholds.h, [761](#)
- \_\_FFLASFFPACK\_charpoly\_INL
  - ffpack\_charpoly.inl, [856](#)
- \_\_FFLASFFPACK\_checker\_charpoly\_INL
  - checker\_charpoly.inl, [743](#)
- \_\_FFLASFFPACK\_checker\_det\_INL
  - checker\_det.inl, [744](#)
- \_\_FFLASFFPACK\_checker\_fgemm\_INL
  - checker\_fgemm.inl, [744](#)
- \_\_FFLASFFPACK\_checker\_ftrsm\_INL
  - checker\_ftrsm.inl, [745](#)
- \_\_FFLASFFPACK\_checker\_invert\_INL
  - checker\_invert.inl, [745](#)
- \_\_FFLASFFPACK\_checker\_pluq\_INL
  - checker\_pluq.inl, [745](#)
- \_\_FFLASFFPACK\_fadd\_INL
  - fflas\_fadd.inl, [768](#)
- \_\_FFLASFFPACK\_fassign\_INL
  - fflas\_fassign.inl, [768](#)
- \_\_FFLASFFPACK\_faxpy\_INL
  - fflas\_faxpy.inl, [769](#)
- \_\_FFLASFFPACK\_fdot\_INL
  - fflas\_fdot.inl, [770](#)
- \_\_FFLASFFPACK\_fflas\_blockcuts\_INL
  - blockcuts.inl, [959](#)
- \_\_FFLASFFPACK\_fflas\_bounds\_INL
  - fflas\_bounds.inl, [764](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL
  - fgemm\_winograd.inl, [775](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL
  - fflas\_level1.inl, [799](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL
  - fflas\_level2.inl, [801](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL
  - fflas\_level3.inl, [803](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL



- fflas\_helpers.inl, [793](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL
- fflas\_sparse.inl, [819](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL
- simd128.inl, [807](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL
- simd128\_double.inl, [807](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL
- simd128\_float.inl, [808](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL
- simd128\_int16.inl, [808](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL
- simd128\_int32.inl, [808](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL
- simd128\_int64.inl, [809](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL
- simd256.inl, [809](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL
- simd256\_double.inl, [810](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL
- simd256\_float.inl, [810](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL
- simd256\_int16.inl, [810](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL
- simd256\_int32.inl, [811](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL
- simd256\_int64.inl, [811](#)
- \_\_FFLASFFPACK\_fflas\_freduce\_INL
- fflas\_freduce.inl, [785](#)
- \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL
- fflas\_freduce\_mp.inl, [786](#)
- \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL
- fflas\_fsyr2k.inl, [789](#)
- \_\_FFLASFFPACK\_fflas\_fsyrrk\_INL
- fflas\_fsyrrk.inl, [790](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL
- igemm.inl, [794](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL
- igemm\_kernels.inl, [796](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL
- igemm\_tools.inl, [796](#)
- \_\_FFLASFFPACK\_fflas\_pfgemm\_INL
- fflas\_pfgemm.inl, [804](#)
- \_\_FFLASFFPACK\_fflas\_pftsrsm\_INL
- fflas\_pftsrsm.inl, [805](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL
- csr\_hyb\_pspmm.inl, [828](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL
- csr\_hyb\_pspmv.inl, [828](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spm INL
- csr\_hyb\_spm.inl, [829](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL
- csr\_hyb\_spmv.inl, [829](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL
- csr\_hyb\_utils.inl, [830](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL
- csr\_pspmm.inl, [824](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL
- csr\_pspmv.inl, [824](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spm INL
- csr\_spm.inl, [825](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL
- csr\_spmv.inl, [826](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL
- ell\_pspmm.inl, [831](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL
- ell\_pspmv.inl, [832](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL
- ell\_simd\_pspmv.inl, [836](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL
- ell\_simd\_spmv.inl, [836](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL
- ell\_simd\_utils.inl, [837](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spm INL
- ell\_spm.inl, [833](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL
- ell\_spmv.inl, [834](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL
- ell\_utils.inl, [834](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL
- hyb\_zo\_pspmm.inl, [838](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL
- hyb\_zo\_pspmv.inl, [838](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spm INL
- hyb\_zo\_spm.inl, [839](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL
- hyb\_zo\_spmv.inl, [839](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL
- hyb\_zo\_utils.inl, [840](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spm INL
- coo\_spm.inl, [821](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL
- coo\_spmv.inl, [822](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL
- coo\_utils.inl, [822](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL
- sell\_pspmv.inl, [842](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL
- sell\_spmv.inl, [843](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL
- sell\_utils.inl, [843](#)
- \_\_FFLASFFPACK\_ffpack\_INL
- ffpack.inl, [855](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL
- ffpack\_charpoly\_danilevski.inl, [856](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL
- ffpack\_charpoly\_kgfast.inl, [856](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL
- ffpack\_charpoly\_kgfastgeneralized.inl, [857](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL
- ffpack\_charpoly\_kglu.inl, [858](#)
- \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL
- ffpack\_echelonforms.inl, [860](#)
- \_\_FFLASFFPACK\_ffpack\_fgesv\_INL
- ffpack\_fgesv.inl, [861](#)
- \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL

- ffpack\_fgetrs.inl, [861](#)
- \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL
  - ffpack\_fsytrf.inl, [863](#)
- \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL
  - ffpack\_ftrssyr2k.inl, [863](#)
- \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL
  - ffpack\_ftrstr.inl, [864](#)
- \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL
  - ffpack\_ftrtr.inl, [865](#)
- \_\_FFLASFFPACK\_ffpack\_invert\_INL
  - ffpack\_invert.inl, [865](#)
- \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL
  - ffpack\_krylovelim.inl, [865](#)
- \_\_FFLASFFPACK\_ffpack\_ludivine\_INL
  - ffpack\_ludivine.inl, [866](#)
- \_\_FFLASFFPACK\_ffpack\_minpoly\_INL
  - ffpack\_minpoly.inl, [867](#)
- \_\_FFLASFFPACK\_ffpack\_permutation\_INL
  - ffpack\_permutation.inl, [870](#)
- \_\_FFLASFFPACK\_ffpack\_pluq\_INL
  - ffpack\_pluq.inl, [870](#)
- \_\_FFLASFFPACK\_ffpack\_ppluq\_INL
  - ffpack\_ppluq.inl, [872](#)
- \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL
  - ffpack\_rankprofiles.inl, [873](#)
- \_\_FFLASFFPACK\_ffgemm\_INL
  - fflas\_fgemm.inl, [772](#)
- \_\_FFLASFFPACK\_ffgemm\_bini\_INL
  - schedule\_bini.inl, [776](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_INL
  - schedule\_winograd.inl, [776](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_acc\_INL
  - schedule\_winograd\_acc.inl, [777](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_acc\_ip\_INL
  - schedule\_winograd\_acc\_ip.inl, [778](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_ip\_INL
  - schedule\_winograd\_ip.inl, [778](#)
- \_\_FFLASFFPACK\_ffgemv\_INL
  - fflas\_fgemv.inl, [780](#)
- \_\_FFLASFFPACK\_ffgemv\_mp\_INL
  - fflas\_fgemv\_mp.inl, [781](#)
- \_\_FFLASFFPACK\_fger\_INL
  - fflas\_fger.inl, [782](#)
- \_\_FFLASFFPACK\_field\_rns\_INL
  - rns.inl, [879](#)
- \_\_FFLASFFPACK\_field\_rns\_double\_INL
  - rns-double.inl, [877](#)
- \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL
  - rns-double-recint.inl, [876](#)
- \_\_FFLASFFPACK\_freivalds\_INL
  - fflas\_freivalds.inl, [786](#)
- \_\_FFLASFFPACK\_fscal\_INL
  - fflas\_fscal.inl, [788](#)
- \_\_FFLASFFPACK\_fscal\_mp\_INL
  - fflas\_fscal\_mp.inl, [788](#)
- \_\_FFLASFFPACK\_ftrmm\_INL
  - fflas\_ftrmm.inl, [790](#)
- \_\_FFLASFFPACK\_ftrsm\_INL
  - fflas\_ftrsm.inl, [791](#)
- \_\_FFLASFFPACK\_ftrsv\_INL
  - fflas\_ftrsv.inl, [792](#)
- \_\_FFLASFFPACK\_simd512\_INL
  - simd512.inl, [812](#)
- \_\_FFLASFFPACK\_simd512\_double\_INL
  - simd512\_double.inl, [812](#)
- \_\_FFLASFFPACK\_simd512\_float\_INL
  - simd512\_float.inl, [812](#)
- \_\_FFLASFFPACK\_simd512\_int32\_INL
  - simd512\_int32.inl, [813](#)
- \_\_FFLAS\_L1\_INST\_C
  - fflas\_L1\_inst.C, [891](#)
- \_\_FFLAS\_L2\_INST\_C
  - fflas\_L2\_inst.C, [895](#)
- \_\_FFLAS\_L3\_INST\_C
  - fflas\_L3\_inst.C, [898](#)
- \_\_FFLAS\_TRSM\_READONLY
  - fflas\_L3\_inst\_implem.inl, [901](#)
  - fflas\_level3.inl, [803](#)
  - ffpack\_ppluq.inl, [872](#)
- \_\_FFPACK\_FSYTRF\_BC\_CROUT
  - benchmark-fsytrf.C, [732](#)
- \_\_FFPACK\_INST\_C
  - ffpack\_inst.C, [953](#)
- \_\_FFPACK\_charpoly\_mp\_INL
  - ffpack\_charpoly\_mp.inl, [858](#)
- \_\_FFPACK\_det\_mp\_INL
  - ffpack\_det\_mp.inl, [859](#)
- \_\_FFPACK\_fgemm\_classical\_INL
  - fgemm\_classical\_mp.inl, [774](#)
- \_\_FFPACK\_fger\_mp\_INL
  - fflas\_fger\_mp.inl, [783](#)
- \_\_FFPACK\_ftrsm\_mp\_INL
  - fflas\_ftrsm\_mp.inl, [791](#)
- \_\_FFPACK\_ludivine\_mp\_INL
  - ffpack\_ludivine\_mp.inl, [867](#)
- \_\_FFPACK\_pluq\_mp\_INL
  - ffpack\_pluq\_mp.inl, [871](#)
- \_\_LUDIVINE\_CUTOFF
  - test-lu.C, [1019](#)
- \_\_has\_builtin
  - bit\_manipulation.h, [970](#)
- \_alloc
  - rns\_double\_elt, [497](#)
  - rns\_double\_elt\_cstptr, [500](#)
  - rns\_double\_elt\_ptr, [502](#)
- \_basis
  - rns\_double, [495](#)
  - rns\_double\_extended, [505](#)
- \_basisMax
  - rns\_double, [495](#)
  - rns\_double\_extended, [505](#)
- \_coo
  - SpMat< Field, flag >, [700](#)
- \_coo16
  - CooMat< Field >, [403](#)
- \_coo16\_zo

- CooMat< Field >, [403](#)
- \_coo32
  - CooMat< Field >, [403](#)
- \_coo32\_zo
  - CooMat< Field >, [403](#)
- \_coo64
  - CooMat< Field >, [403](#)
- \_coo64\_zo
  - CooMat< Field >, [404](#)
- \_crt\_in
  - rns\_double, [495](#)
  - rns\_double\_extended, [506](#)
- \_crt\_out
  - rns\_double, [496](#)
  - rns\_double\_extended, [506](#)
- \_csr
  - SpMat< Field, flag >, [700](#)
- \_csr16
  - CsrMat< Field >, [404](#)
- \_csr16\_zo
  - CsrMat< Field >, [404](#)
- \_csr32
  - CsrMat< Field >, [404](#)
- \_csr32\_zo
  - CsrMat< Field >, [404](#)
- \_csr64
  - CsrMat< Field >, [404](#)
- \_csr64\_zo
  - CsrMat< Field >, [404](#)
- \_ell
  - SpMat< Field, flag >, [700](#)
- \_ell16
  - EllMat< Field >, [410](#)
- \_ell16\_zo
  - EllMat< Field >, [410](#)
- \_ell32
  - EllMat< Field >, [410](#)
- \_ell32\_zo
  - EllMat< Field >, [411](#)
- \_ell64
  - EllMat< Field >, [410](#)
- \_ell64\_zo
  - EllMat< Field >, [411](#)
- \_errorStream
  - Failure, [412](#)
- \_field\_rns
  - rns\_double, [495](#)
  - rns\_double\_extended, [506](#)
- \_iM\_modp\_rns
  - RNSIntegerMod< RNS >, [516](#)
- \_ibeg
  - ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- \_iend
  - ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- \_invbasis
  - rns\_double, [495](#)
  - rns\_double\_extended, [506](#)
- \_jbeg
  - ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- \_jend
  - ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- \_ldm
  - rns\_double, [496](#)
  - rns\_double\_extended, [506](#)
- \_mi\_sum
  - rns\_double, [496](#)
- \_negbasis
  - rns\_double, [495](#)
  - rns\_double\_extended, [505](#)
- \_p
  - RNSIntegerMod< RNS >, [516](#)
- \_pbits
  - rns\_double, [496](#)
  - rns\_double\_extended, [506](#)
- \_ptr
  - rns\_double\_elt, [497](#)
  - rns\_double\_elt\_cstptr, [500](#)
  - rns\_double\_elt\_ptr, [502](#)
- \_rns
  - RNSInteger< RNS >, [510](#)
  - RNSIntegerMod< RNS >, [516](#)
- \_simd512\_int64\_INL
  - simd512\_int64.inl, [813](#)
- \_size
  - rns\_double, [496](#)
  - rns\_double\_extended, [506](#)
- \_stride
  - rns\_double\_elt, [497](#)
  - rns\_double\_elt\_cstptr, [500](#)
  - rns\_double\_elt\_ptr, [502](#)
- ~CheckerImplem\_Det
  - CheckerImplem\_Det< Field >, [385](#)
- ~CheckerImplem\_PLUQ
  - CheckerImplem\_PLUQ< Field >, [389](#)
- ~CheckerImplem\_charpoly
  - CheckerImplem\_charpoly< Field, Polynomial >, [384](#)
- ~CheckerImplem\_fgemm
  - CheckerImplem\_fgemm< Field >, [386](#)
- ~CheckerImplem\_ftrsm
  - CheckerImplem\_ftrsm< Field >, [387](#)
- ~CheckerImplem\_invert
  - CheckerImplem\_invert< Field >, [388](#)
- ~rns\_double\_elt
  - rns\_double\_elt, [497](#)
- 101-fgemm.C, [1033](#)
  - main, [1033](#)
- 2x2-fgemm.C, [1033](#)
  - main, [1034](#)
- 2x2-ftrsv.C, [1034](#)
  - main, [1034](#)
- 2x2-pluq.C, [1034](#)
  - main, [1035](#)
- add
  - FFLAS::vectorised, [268](#)
  - FieldSimd< \_Field >, [415](#)

- RNSIntegerMod< RNS >, 514
- ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 520
- ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 524
- Simd128\_impl< true, true, false, 2 >, 535
- Simd128\_impl< true, true, false, 4 >, 544
- Simd128\_impl< true, true, false, 8 >, 553
- Simd128\_impl< true, true, true, 2 >, 559
- Simd128\_impl< true, true, true, 4 >, 567
- Simd128\_impl< true, true, true, 8 >, 575
- Simd256\_impl< true, false, true, 8 >, 584
- Simd256\_impl< true, true, false, 2 >, 594
- Simd256\_impl< true, true, false, 4 >, 608
- Simd256\_impl< true, true, false, 8 >, 619
- Simd256\_impl< true, true, true, 2 >, 626
- Simd256\_impl< true, true, true, 4 >, 635, 639
- Simd256\_impl< true, true, true, 8 >, 648
- Simd512\_impl< true, false, true, 8 >, 656
- Simd512\_impl< true, true, false, 8 >, 666
- Simd512\_impl< true, true, true, 8 >, 673
- add\_r
  - FieldSimd< \_Field >, 416
- addin
  - FieldSimd< \_Field >, 416
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 520
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 524
  - Simd128\_impl< true, true, false, 2 >, 535
  - Simd128\_impl< true, true, false, 4 >, 544
  - Simd128\_impl< true, true, false, 8 >, 553
  - Simd128\_impl< true, true, true, 2 >, 560
  - Simd128\_impl< true, true, true, 4 >, 567
  - Simd128\_impl< true, true, true, 8 >, 575
  - Simd256\_impl< true, false, true, 8 >, 584
  - Simd256\_impl< true, true, false, 2 >, 594
  - Simd256\_impl< true, true, false, 4 >, 608
  - Simd256\_impl< true, true, false, 8 >, 619
  - Simd256\_impl< true, true, true, 2 >, 626
  - Simd256\_impl< true, true, true, 4 >, 635, 639
  - Simd256\_impl< true, true, true, 8 >, 648
  - Simd512\_impl< true, false, true, 8 >, 656
  - Simd512\_impl< true, true, false, 8 >, 666
  - Simd512\_impl< true, true, true, 8 >, 673
- addin\_r
  - FieldSimd< \_Field >, 416
- addp
  - FFLAS::vectorised, 267
- AlgoChooser< ModeCategories::ConvertTo< Element-Categories::RNSElementTag >, ParSeq >, 377
  - value, 377
- AlgoChooser< ModeT, ParSeq >, 377
  - value, 377
- align-allocator.h, 968
- alignable
  - FFLAS, 179
- alignable< Givaro::Integer \* >
  - FFLAS, 179
- alignment
  - FieldSimd< \_Field >, 419
  - Simd128\_impl< true, true, false, 2 >, 538
  - Simd128\_impl< true, true, false, 4 >, 546
  - Simd128\_impl< true, true, false, 8 >, 556
  - Simd128\_impl< true, true, true, 2 >, 563
  - Simd128\_impl< true, true, true, 4 >, 571
  - Simd128\_impl< true, true, true, 8 >, 579
  - Simd256\_impl< true, false, true, 8 >, 587
  - Simd256\_impl< true, true, false, 2 >, 596
  - Simd256\_impl< true, true, false, 4 >, 612
  - Simd256\_impl< true, true, false, 8 >, 621
  - Simd256\_impl< true, true, true, 2 >, 629
  - Simd256\_impl< true, true, true, 4 >, 643
  - Simd256\_impl< true, true, true, 8 >, 651
  - Simd512\_impl< true, false, true, 8 >, 658
  - Simd512\_impl< true, true, false, 8 >, 668
  - Simd512\_impl< true, true, true, 8 >, 677
- Amax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 468
- Amin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 467
- applyP
  - FFPACK, 289, 290, 343
- applyP\_block
  - FFPACK, 335
- applyP\_modular\_double
  - ffpack.C, 918
  - ffpack\_c.h, 940
- ArbitraryPreIntTag, 377
- Architecture of the library., 2
- areEqual
  - RNSIntegerMod< RNS >, 515
- AreEqual< X, X >, 378
  - value, 378
- AreEqual< X, Y >, 378
  - value, 378
- args-parser.h, 968
  - ArgumentType, 969
  - END\_OF\_ARGUMENTS, 968
  - findArgument, 969
  - getListArgs, 969
  - printHelpMessage, 969
  - TYPE\_BOOL, 968
  - TYPE\_DOUBLE, 969
  - TYPE\_INT, 969
  - TYPE\_INTEGER, 969
  - type\_integer, 969
  - TYPE\_INTLIST, 969
  - TYPE\_LONGLONG, 969
  - TYPE\_NONE, 969
  - TYPE\_STR, 969

- TYPE\_UINT64, 969
- Argument, 378
  - c, 378
  - data, 379
  - example, 378
  - helpString, 378
  - type, 379
- ArgumentType
  - args-parser.h, 969
- ArithProg
  - FFPACK::Protected, 373
- arithprog.C, 711
  - CUBE, 711
  - GFOPS, 711
  - main, 712
  - TTimer, 711
- assign
  - RNSInteger< RNS >, 509
  - RNSIntegerMod< RNS >, 514
- associatedDelayedField< const FFPACK::RNSIntegerMod< RNS >, 379
  - field, 379
  - type, 379
- associatedDelayedField< const Givaro::Modular< T, X >, 380
  - field, 380
  - type, 380
- associatedDelayedField< const Givaro::ModularBalanced< T >, 380
  - field, 380
  - type, 380
- associatedDelayedField< const Givaro::ZRing< T >, 381
  - field, 381
  - type, 381
- associatedDelayedField< Field >, 379
  - field, 379
  - type, 379
- assume\_aligned
  - fflas\_sparse.h, 817
- AtlasConj
  - config-blas.h, 749
- Aunfit
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 466
- aut
  - HelperFlag, 440
- Auto, 381
- autotune/charpoly.C
  - CUBE, 739
  - GFOPS, 739
  - main, 740
  - TTimer, 740
- autotune/pluq.C
  - CUBE, 741
  - GFOPS, 741
  - main, 742
  - TTimer, 742
- averageCol
  - StatsMatrix, 703
- averageColDifference
  - StatsMatrix, 703
- averageRow
  - StatsMatrix, 702
- averageRowDifference
  - StatsMatrix, 703
- axpy
  - FieldSimd< \_Field >, 418
- axpy\_r
  - FieldSimd< \_Field >, 418
- axpyin
  - FieldSimd< \_Field >, 418
  - RNSIntegerMod< RNS >, 515
- axpyin\_r
  - FieldSimd< \_Field >, 418
- balanced
  - FieldTraits< FFPACK::RNSInteger< T >, 420
  - FieldTraits< FFPACK::RNSIntegerMod< T >, 421
  - FieldTraits< Field >, 420
  - FieldTraits< Givaro::Modular< Element >, 421
  - FieldTraits< Givaro::ModularBalanced< Element >, 422
  - FieldTraits< Givaro::ZRing< double >, 422
  - FieldTraits< Givaro::ZRing< float >, 423
  - FieldTraits< Givaro::ZRing< Givaro::Integer >, 423
  - FieldTraits< Givaro::ZRing< int16\_t >, 424
  - FieldTraits< Givaro::ZRing< int32\_t >, 424
  - FieldTraits< Givaro::ZRing< int64\_t >, 425
  - FieldTraits< Givaro::ZRing< Reclnt::ruint< K >, 425
  - FieldTraits< Givaro::ZRing< uint16\_t >, 426
  - FieldTraits< Givaro::ZRing< uint32\_t >, 426
  - FieldTraits< Givaro::ZRing< uint64\_t >, 426
  - winograd.C, 715
- BARRIER
  - parallel.h, 961
- BASECASE\_K
  - test-lu.C, 1019
- BaseTimer
  - FFLAS, 72
- BasisElement
  - rns\_double, 492
  - rns\_double\_extended, 504
  - RNSInteger< RNS >, 507
  - RNSIntegerMod< RNS >, 512
- begin
  - ForStrategy1D< blocksize\_t, Cut, Param >, 428
  - Info, 445, 446
- BEGIN\_PARALLEL\_MAIN
  - parallel.h, 962
- benchmark-charpoly-mp.C, 715
  - \_\_FFLASFFPACK\_FORCE\_SEQ, 716
  - main, 716
- benchmark-charpoly.C, 716

- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 716
  - main, 716
  - run\_with\_field, 716
- benchmark-checkers.C, 717
  - \_MAX\_SIZE\_MATRICES, 717
  - \_NR\_TESTS, 717
  - CUBE, 717
  - ENABLE\_ALL\_CHECKINGS, 717
  - main, 717
- benchmark-dgemm.C, 718
  - CBLAS\_GEMM, 718
  - Floats, 718
  - main, 718
  - TTimer, 718
- benchmark-dgetrf.C, 718
  - \_\_FFLASFFPACK\_HAVE\_DGETRF, 719
  - main, 719
  - TTimer, 719
- benchmark-dgetri.C, 719
  - main, 720
  - TTimer, 719
- benchmark-dsytrf.C, 720
  - EFFGFF, 720
  - main, 720
  - TTimer, 720
- benchmark-dtrsm.C, 721
  - main, 721
  - TTimer, 721
- benchmark-dtrtri.C, 721
  - \_\_FFLASFFPACK\_HAVE\_DTRTRI, 722
  - main, 722
  - TTimer, 722
- benchmark-fadd-lvl2.C, 722
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 722
  - main, 722
- benchmark-fdot.C, 722
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 723
  - main, 723
  - run\_with\_field, 723
- benchmark-fgemm-mp.C, 723
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 724
  - main, 724
  - MG\_DEFAULT, 724
  - STD\_RECINT\_SIZE, 724
  - tmain, 724
- benchmark-fgemm-rns.C, 724
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 725
  - ConstElement\_ptr, 725
  - Element\_ptr, 725
  - Field, 725
  - GRAIN, 725
  - main, 726
  - PSeq, 726
- RNS, 725
  - THREADS, 725
  - THREED, 725
  - THREEDA, 725
  - THREEDIP, 725
  - TWOD, 725
  - TWODA, 725
- benchmark-fgemm.C, 726
  - CLASSIC\_HYBRID, 726
  - main, 726
- benchmark-fgemv-mp.C, 726
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 727
  - main, 727
  - MG\_DEFAULT, 727
  - STD\_RECINT\_SIZE, 727
  - tmain, 727
  - write\_matrix, 727
- benchmark-fgemv.C, 728
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 728
  - benchmark\_disp, 730
  - benchmark\_in\_Field, 730
  - benchmark\_with\_field, 730
  - benchmark\_with\_timer, 729
  - check\_result, 729
  - fill\_value, 729
  - genData, 729
  - main, 731
- benchmark-fgesv.C, 731
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 731
  - main, 731
- benchmark-fsyrk.C, 731
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 732
  - CUBE, 732
  - main, 732
- benchmark-fsytrf.C, 732
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 732
  - \_\_FFPACK\_FSYTRF\_BC\_CROUT, 732
  - CUBE, 732
  - main, 733
- benchmark-fttrsm-mp.C, 733
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 733
  - main, 733
- benchmark-fttrsm.C, 733
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 734
  - main, 734
- benchmark-fttrsv.C, 734
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 734
  - main, 734
- benchmark-fttrtri.C, 734

- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 735
- CUBE, 735
- main, 735
- benchmark-inverse.C, 735
  - CUBE, 735
  - main, 736
- benchmark-lqup-mp.C, 736
  - main, 736
- benchmark-lqup.C, 736
  - CUBE, 736
  - main, 737
- benchmark-pluq.C, 737
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 737
  - CUBE, 737
  - Field, 737
  - main, 738
  - Rec\_Initialize, 738
  - verification\_PLUQ, 738
- benchmark-wino.C, 738
  - CUBE, 738
  - launch\_wino, 739
  - main, 739
- benchmark\_disp
  - benchmark-fgemv.C, 730
- benchmark\_in\_Field
  - benchmark-fgemv.C, 730
- benchmark\_with\_field
  - benchmark-fgemv.C, 730
- benchmark\_with\_timer
  - benchmark-fgemv.C, 729
- Bibliography, 7
- Bini, 381
  - FFLAS::BLAS3, 183
- bit\_manipulation.h, 970
  - \_\_has\_builtin, 970
  - clz, 970
  - ctz, 970
- bitsize
  - FFLAS, 132
- bitsize< Givaro::ZRing< Givaro::Integer > >
  - FFLAS, 132
- blas\_enum
  - config-blas.h, 749
- blend
  - Simd128\_impl< true, true, false, 2 >, 535
  - Simd128\_impl< true, true, false, 4 >, 544
  - Simd128\_impl< true, true, false, 8 >, 553
  - Simd128\_impl< true, true, true, 2 >, 559
  - Simd128\_impl< true, true, true, 4 >, 567
  - Simd128\_impl< true, true, true, 8 >, 575
  - Simd256\_impl< true, false, true, 8 >, 584
  - Simd256\_impl< true, true, false, 4 >, 608
  - Simd256\_impl< true, true, false, 8 >, 619
  - Simd256\_impl< true, true, true, 4 >, 634
  - Simd256\_impl< true, true, true, 8 >, 647
  - Simd512\_impl< true, false, true, 8 >, 656
  - Simd512\_impl< true, true, false, 8 >, 665
  - Simd512\_impl< true, true, true, 8 >, 673
- blend\_twice
  - Simd256\_impl< true, true, false, 2 >, 594
  - Simd256\_impl< true, true, true, 2 >, 625
- blendv
  - Simd256\_impl< true, false, true, 8 >, 584
  - Simd512\_impl< true, false, true, 8 >, 656
- Block, 381
- BlockCuts
  - FFLAS, 170, 173
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >
  - FFLAS, 172
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >
  - FFLAS, 171
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >
  - FFLAS, 172
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >
  - FFLAS, 171
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >
  - FFLAS, 172
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >
  - FFLAS, 172
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >
  - FFLAS, 171
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >
  - FFLAS, 171
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >
  - FFLAS, 172
- BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >
  - FFLAS, 171
- blockcuts.inl, 958
  - \_\_FFLASFFPACK\_MINBLOCKCUTS, 959
  - \_\_FFLASFFPACK\_fflas\_blockcuts\_INL, 959
- blockindex
  - ForStrategy1D< blocksize\_t, Cut, Param >, 428
  - ForStrategy2D< blocksize\_t, Cut, Param >, 431
- BlockingFactor
  - FFLAS::details, 196
- BLOCKS
  - ForStrategy2D< blocksize\_t, Cut, Param >, 433
- Bmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 468
- Bmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 468
- Bug List, 3



- build
  - ForStrategy1D< blocksize\_t, Cut, Param >, 428
- buildMatrix
  - FFPACK, 329
- Bunfit
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 466
- c
  - Argument, 378
- callLUdivine\_small< double >, 382
  - operator(), 382
- callLUdivine\_small< Element >, 381
  - operator(), 381
- callLUdivine\_small< float >, 382
  - operator(), 382
- cardinality
  - RNSInteger< RNS >, 509
  - RNSIntegerMod< RNS >, 513
- cast.h, 970
- category
  - FieldTraits< FFPACK::RNSInteger< T > >, 420
  - FieldTraits< FFPACK::RNSIntegerMod< T > >, 421
  - FieldTraits< Field >, 420
  - FieldTraits< Givaro::Modular< Element > >, 421
  - FieldTraits< Givaro::ModularBalanced< Element > >, 422
  - FieldTraits< Givaro::ZRing< double > >, 422
  - FieldTraits< Givaro::ZRing< float > >, 423
  - FieldTraits< Givaro::ZRing< Givaro::Integer > >, 423
  - FieldTraits< Givaro::ZRing< int16\_t > >, 424
  - FieldTraits< Givaro::ZRing< int32\_t > >, 424
  - FieldTraits< Givaro::ZRing< int64\_t > >, 424
  - FieldTraits< Givaro::ZRing< Reclnt::ruint< K > >, 425
  - FieldTraits< Givaro::ZRing< uint16\_t > >, 425
  - FieldTraits< Givaro::ZRing< uint32\_t > >, 426
  - FieldTraits< Givaro::ZRing< uint64\_t > >, 426
- cblas.C, 978
  - \_\_FFLASFFPACK\_CONFIGURATION, 978
  - \_\_FFLASFFPACK\_HAVE\_CBLAS, 978
  - main, 978
- CBLAS\_DIAG
  - config-blas.h, 749
- CBLAS\_ENUM\_DEFINED\_H
  - config-blas.h, 749
- CBLAS\_EXTERNALS
  - config-blas.h, 749
- CBLAS\_GEMM
  - benchmark-dgemm.C, 718
- cblas\_imptrsm
  - FFLAS, 121
- CBLAS\_INT
  - config-blas.h, 749
- CBLAS\_ORDER
  - config-blas.h, 749
- CBLAS\_SIDE
  - config-blas.h, 750
- CBLAS\_TRANSPOSE
  - config-blas.h, 749
- CBLAS\_UPLO
  - config-blas.h, 749
- CblasColMajor
  - config-blas.h, 749
- CblasConjTrans
  - config-blas.h, 749
- CblasLeft
  - config-blas.h, 750
- CblasLower
  - config-blas.h, 749
- CblasNonUnit
  - config-blas.h, 749
- CblasNoTrans
  - config-blas.h, 749
- CblasRight
  - config-blas.h, 750
- CblasRowMajor
  - config-blas.h, 749
- CblasTrans
  - config-blas.h, 749
- CblasUnit
  - config-blas.h, 750
- CblasUpper
  - config-blas.h, 749
- ceil
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 519
  - Simd256\_impl< true, false, true, 8 >, 587
  - Simd512\_impl< true, false, true, 8 >, 658
- changeBS
  - ForStrategy1D< blocksize\_t, Cut, Param >, 429
- changeCBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, 433
- changeRBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, 432
- characteristic
  - RNSInteger< RNS >, 509
  - RNSIntegerMod< RNS >, 513
- CharPoly
  - FFPACK, 307, 308, 329, 347, 348
- charpoly.C, 739, 740
- CharpolyFailed, 383
- check
  - Checker\_Empty< Field >, 383
  - CheckerImplem\_charpoly< Field, Polynomial >, 384
  - CheckerImplem\_Det< Field >, 385
  - CheckerImplem\_fgemm< Field >, 386
  - CheckerImplem\_ftsm< Field >, 387
  - CheckerImplem\_invert< Field >, 388
  - CheckerImplem\_PLUQ< Field >, 389
- check1
  - regression-check.C, 984
- check2



- regression-check.C, [984](#)
- check3
  - regression-check.C, [984](#)
- check4
  - regression-check.C, [984](#)
- CHECK\_DEPENDENCIES
  - parallel.h, [961](#)
- check\_eq
  - test-simd.C, [1031](#)
- check\_fdot
  - test-fdot.C, [993](#)
- check\_fger
  - test-fger.C, [999](#)
- check\_fsyr2k
  - test-fsyr2k.C, [1004](#)
- check\_fsyrk
  - test-fsyrk.C, [1006](#)
- check\_fsyrk\_bkdiag
  - test-fsyrk.C, [1006](#)
- check\_fsyrk\_diag
  - test-fsyrk.C, [1006](#)
- check\_ftrmm
  - test-ftrmm.C, [1009](#)
- check\_ftrmv
  - test-ftrmv.C, [1010](#)
- check\_ftrsm
  - test-ftrsm.C, [1012](#)
- check\_ftrssyr2k
  - test-ftrssyr2k.C, [1013](#)
- check\_ftrstr
  - test-ftrstr.C, [1014](#)
- check\_ftrsv
  - test-ftrsv.C, [1015](#)
- check\_ftrtri
  - test-ftrtri.C, [1016](#)
- check\_minpoly
  - test-minpoly.C, [1023](#)
- check\_MM
  - test-fgemm.C, [995](#)
- check\_MV
  - test-fgemv.C, [997](#)
- check\_result
  - benchmark-fgemv.C, [729](#)
- check\_solve
  - test-solve.C, [1033](#)
- checkA
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [466](#)
- checkB
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [467](#)
- CHECKER, [41](#)
- Checker\_charpoly
  - FFPACK, [288](#)
- checker\_charpoly.inl, [743](#)
  - \_\_FFLASFFPACK\_checker\_charpoly\_INL, [743](#)
- Checker\_Det
  - FFPACK, [287](#)
- checker\_det.inl, [743](#)
  - \_\_FFLASFFPACK\_checker\_det\_INL, [744](#)
- Checker\_Empty
  - Checker\_Empty< Field >, [383](#)
- Checker\_Empty< Field >, [383](#)
  - check, [383](#)
  - Checker\_Empty, [383](#)
- checker\_empty.h, [744](#)
- Checker\_fgemm
  - FFLAS, [70](#)
- checker\_fgemm.inl, [744](#)
  - \_\_FFLASFFPACK\_checker\_fgemm\_INL, [744](#)
- Checker\_ftrsm
  - FFLAS, [70](#)
- checker\_ftrsm.inl, [744](#)
  - \_\_FFLASFFPACK\_checker\_ftrsm\_INL, [745](#)
- Checker\_invert
  - FFPACK, [287](#)
- checker\_invert.inl, [745](#)
  - \_\_FFLASFFPACK\_checker\_invert\_INL, [745](#)
- Checker\_PLUQ
  - FFPACK, [287](#)
- checker\_pluq.inl, [745](#)
  - \_\_FFLASFFPACK\_checker\_pluq\_INL, [745](#)
- CheckerImplem\_charpoly
  - CheckerImplem\_charpoly< Field, Polynomial >, [384](#)
- CheckerImplem\_charpoly< Field, Polynomial >, [384](#)
  - ~CheckerImplem\_charpoly, [384](#)
  - check, [384](#)
  - CheckerImplem\_charpoly, [384](#)
- CheckerImplem\_Det
  - CheckerImplem\_Det< Field >, [385](#)
- CheckerImplem\_Det< Field >, [384](#)
  - ~CheckerImplem\_Det, [385](#)
  - check, [385](#)
  - CheckerImplem\_Det, [385](#)
- CheckerImplem\_fgemm
  - CheckerImplem\_fgemm< Field >, [386](#)
- CheckerImplem\_fgemm< Field >, [385](#)
  - ~CheckerImplem\_fgemm, [386](#)
  - check, [386](#)
  - CheckerImplem\_fgemm, [386](#)
- CheckerImplem\_ftrsm
  - CheckerImplem\_ftrsm< Field >, [387](#)
- CheckerImplem\_ftrsm< Field >, [386](#)
  - ~CheckerImplem\_ftrsm, [387](#)
  - check, [387](#)
  - CheckerImplem\_ftrsm, [387](#)
- CheckerImplem\_invert
  - CheckerImplem\_invert< Field >, [388](#)
- CheckerImplem\_invert< Field >, [388](#)
  - ~CheckerImplem\_invert, [388](#)
  - check, [388](#)
  - CheckerImplem\_invert, [388](#)
- CheckerImplem\_PLUQ
  - CheckerImplem\_PLUQ< Field >, [389](#)
- CheckerImplem\_PLUQ< Field >, [388](#)

- ~CheckerImplem\_PLUQ, [389](#)
- check, [389](#)
- CheckerImplem\_PLUQ, [389](#)
- checkers.doxy, [746](#)
- checkers\_fflas.h, [746](#)
- checkers\_fflas.inl, [746](#)
  - FFLASFFPACK\_checkers\_fflas\_inl\_H, [746](#)
- checkers\_ffpack.h, [747](#)
- checkers\_ffpack.inl, [747](#)
  - FFLASFFPACK\_checkers\_ffpack\_inl\_H, [748](#)
- checkingMessage
  - test-nullspace.C, [1025](#)
- checkMonotonicApplyP
  - test-permutations.C, [1026](#)
- checkOut
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [467](#)
- checkRPM
  - test-rpm.C, [1028](#)
- checkSymmetricRPM
  - test-rpm.C, [1028](#)
- checkZeroDimCharpoly
  - regression-check.C, [984](#)
- checkZeroDimMinPoly
  - regression-check.C, [984](#)
- chooseField
  - FFPACK, [368](#)
- chooseField< Givaro::ZRing< double > >
  - FFPACK, [368](#)
- chooseField< Givaro::ZRing< float > >
  - FFPACK, [368](#)
- chooseField< Givaro::ZRing< int32\_t > >
  - FFPACK, [368](#)
- chooseField< Givaro::ZRing< int64\_t > >
  - FFPACK, [368](#)
- chunk
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [690](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- chunkSize
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [698](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [700](#)
- clapack.C, [978](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [979](#)
  - \_\_FFLASFFPACK\_HAVE\_CLAPACK, [979](#)
  - \_\_FFLASFFPACK\_HAVE\_LAPACK, [979](#)
  - main, [979](#)
- Classic, [389](#)
- CLASSIC\_HYBRID
  - benchmark-fgemm.C, [726](#)
- clz
  - bit\_manipulation.h, [970](#)
- Cmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [468](#)
- Cmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [468](#)
- col
  - Coo< Field >, [401](#)
  - Coo< ValT, IdxT >, [400](#), [403](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [680](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [685](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [688](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [691](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [698](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [700](#)
- colblockindex
  - ForStrategy2D< blocksize\_t, Cut, Param >, [431](#)
- colBlockSize
  - ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- coldim
  - StatsMatrix, [702](#)
- colnumblocks
  - ForStrategy2D< blocksize\_t, Cut, Param >, [431](#)
- ColRankProfileSubmatrix
  - FFPACK, [320](#), [352](#)
- ColRankProfileSubmatrix\_modular\_double
  - ffpack.C, [930](#)
  - ffpack\_c.h, [950](#)
- ColRankProfileSubmatrixIndices
  - FFPACK, [319](#), [352](#)
- ColRankProfileSubmatrixIndices\_modular\_double
  - ffpack.C, [930](#)
  - ffpack\_c.h, [950](#)
- Column, [390](#)
- ColumnEchelonForm
  - FFPACK, [301](#), [346](#)
- ColumnEchelonForm\_modular\_double
  - ffpack.C, [921](#)
  - ffpack\_c.h, [943](#)
- ColumnEchelonForm\_modular\_float
  - ffpack.C, [921](#)
  - ffpack\_c.h, [943](#)
- ColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [922](#)
  - ffpack\_c.h, [943](#)
- ColumnRankProfile
  - FFPACK, [316](#), [317](#), [351](#)
- ColumnRankProfile\_modular\_double
  - ffpack.C, [929](#)
  - ffpack\_c.h, [949](#)
- COMMA
  - parallel.h, [964](#)
- CompactElement< double >, [390](#)
  - type, [390](#)
- CompactElement< Element >, [390](#)

- type, [390](#)
- CompactElement< float >, [390](#)
  - type, [390](#)
- CompactElement< int16\_t >, [390](#)
  - type, [391](#)
- CompactElement< int32\_t >, [391](#)
  - type, [391](#)
- CompactElement< int64\_t >, [391](#)
  - type, [391](#)
- compatible\_data\_type< Field >, [391](#)
  - value, [391](#)
- compatible\_data\_type< Givaro::ZRing< double > >, [391](#)
  - value, [392](#)
- compatible\_data\_type< Givaro::ZRing< float > >, [392](#)
  - value, [392](#)
- compliant
  - NoSimd< T >, [486](#)
  - Simd128\_impl< true, true, false, 2 >, [534](#)
  - Simd128\_impl< true, true, false, 4 >, [542](#)
  - Simd128\_impl< true, true, false, 8 >, [551](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [565](#)
  - Simd128\_impl< true, true, true, 8 >, [573](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [592](#)
  - Simd256\_impl< true, true, false, 4 >, [605](#)
  - Simd256\_impl< true, true, false, 8 >, [617](#)
  - Simd256\_impl< true, true, true, 2 >, [623](#)
  - Simd256\_impl< true, true, true, 4 >, [632](#), [638](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [654](#)
  - Simd512\_impl< true, true, false, 8 >, [663](#)
  - Simd512\_impl< true, true, true, 8 >, [670](#)
- Compose
  - Compose< H1, H2 >, [392](#), [393](#)
- Compose< H1, H2 >, [392](#)
  - Compose, [392](#), [393](#)
  - first\_component, [393](#)
  - operator<<, [393](#)
  - second\_component, [393](#)
- composePermutationsLLL
  - FFPACK, [338](#)
  - ffpack.C, [917](#)
  - ffpack\_c.h, [939](#)
- composePermutationsLLM
  - FFPACK, [338](#)
  - ffpack.C, [917](#)
  - ffpack\_c.h, [939](#)
- composePermutationsMLM
  - FFPACK, [339](#)
  - ffpack.C, [917](#)
  - ffpack\_c.h, [939](#)
- CompressRows
  - FFPACK::Protected, [374](#)
- CompressRowsQA
  - FFPACK::Protected, [375](#)
- CompressRowsQK
  - FFPACK::Protected, [375](#)
- computeDeviation
  - FFLAS, [145](#)
- computeFactorClassic
  - FFLAS::Protected, [201](#)
- config-blas.h, [748](#)
  - AtlasConj, [749](#)
  - blas\_enum, [749](#)
  - CBLAS\_DIAG, [749](#)
  - CBLAS\_ENUM\_DEFINED\_H, [749](#)
  - CBLAS\_EXTERNALS, [749](#)
  - CBLAS\_INT, [749](#)
  - CBLAS\_ORDER, [749](#)
  - CBLAS\_SIDE, [750](#)
  - CBLAS\_TRANSPOSE, [749](#)
  - CBLAS\_UPLO, [749](#)
  - CblasColMajor, [749](#)
  - CblasConjTrans, [749](#)
  - CblasLeft, [750](#)
  - CblasLower, [749](#)
  - CblasNonUnit, [749](#)
  - CblasNoTrans, [749](#)
  - CblasRight, [750](#)
  - CblasRowMajor, [749](#)
  - CblasTrans, [749](#)
  - CblasUnit, [750](#)
  - CblasUpper, [749](#)
  - dasum\_, [750](#)
  - daxpy\_, [750](#)
  - dcopy\_, [752](#)
  - ddot\_, [750](#)
  - dgemm\_, [754](#)
  - dgemv\_, [751](#)
  - dger\_, [751](#)
  - dnrm2\_, [751](#)
  - dscal\_, [752](#)
  - dtrmm\_, [753](#)
  - dtrsm\_, [752](#)
  - idamax\_, [751](#)
  - saxpy\_, [750](#)
  - scopy\_, [752](#)
  - sdot\_, [750](#)
  - sgemm\_, [753](#)
  - sgemv\_, [751](#)
  - sger\_, [751](#)
  - sscal\_, [752](#)
  - strmm\_, [753](#)
  - strsm\_, [753](#)
- config.h, [754](#), [757](#)
  - HAVE\_BLAS, [755](#)
  - HAVE\_CBLAS, [755](#)
  - HAVE\_CXX11, [755](#)
  - HAVE\_DLFCN\_H, [755](#)
  - HAVE\_FLOAT\_H, [755](#)
  - HAVE\_INTPYPES\_H, [755](#)
  - HAVE\_LAPACK, [755](#)
  - HAVE\_LIMITS\_H, [755](#)
  - HAVE\_LITTLE\_ENDIAN, [755](#)

- HAVE\_PTHREAD\_H, 755
- HAVE\_STDDEF\_H, 755
- HAVE\_STDINT\_H, 755
- HAVE\_STDIO\_H, 755
- HAVE\_STDLIB\_H, 756
- HAVE\_STRING\_H, 756
- HAVE\_STRINGS\_H, 756
- HAVE\_SYS\_STAT\_H, 756
- HAVE\_SYS\_TIME\_H, 756
- HAVE\_SYS\_TYPES\_H, 756
- HAVE\_UNISTD\_H, 756
- LT\_OBJDIR, 756
- OPENBLAS\_NUM\_THREADS, 756
- PACKAGE, 756
- PACKAGE\_BUGREPORT, 756
- PACKAGE\_NAME, 756
- PACKAGE\_STRING, 756
- PACKAGE\_TARNAME, 756
- PACKAGE\_URL, 756
- PACKAGE\_VERSION, 757
- SIZEOF\_\_\_INT64, 757
- SIZEOF\_CHAR, 757
- SIZEOF\_INT, 757
- SIZEOF\_LONG, 757
- SIZEOF\_LONG\_LONG, 757
- SIZEOF\_SHORT, 757
- STDC\_HEADERS, 757
- USE\_OPENMP, 757
- VERSION, 757
- Configuring and Installing FFLAS-FFPACK, 2
- CONST
  - fflas\_simd.h, 806
- const\_int
  - instrset.h, 981
- Const\_int\_t< n >, 393
- const\_uint
  - instrset.h, 981
- Const\_uint\_t< n >, 393
- ConstElement\_ptr
  - benchmark-fgemm-rns.C, 725
  - rns\_double, 493
  - rns\_double\_extended, 504
  - RNSInteger< RNS >, 508
  - RNSIntegerMod< RNS >, 512
- CONSTREFERENCE
  - parallel.h, 962
- convert
  - rns\_double, 494, 495
  - rns\_double\_extended, 505
  - RNSInteger< RNS >, 509
  - RNSIntegerMod< RNS >, 514
- convert\_transpose
  - rns\_double, 494
- ConvertTo< T >, 398
- COO
  - FFLAS, 74
- Coo
  - Coo< Field >, 400, 401
  - Coo< ValT, IdxT >, 399, 402
- coo
  - HelperFlag, 440
- Coo< Field >, 400
  - col, 401
  - Coo, 400, 401
  - deleted, 401
  - operator=, 401
  - row, 401
  - val, 401
- Coo< ValT, IdxT >, 398, 401
  - col, 400, 403
  - Coo, 399, 402
  - operator=, 399, 402
  - row, 400, 403
  - Self, 399, 402
  - val, 400, 403
- coo.h, 820
- coo\_spm.inl, 820
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spm\_INL, 821
- coo\_spmv.inl, 821
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL, 822
- coo\_utils.inl, 822
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL, 822
- COO\_ZO
  - FFLAS, 74
- CooMat< Field >, 403
  - \_coo16, 403
  - \_coo16\_zo, 403
  - \_coo32, 403
  - \_coo32\_zo, 403
  - \_coo64, 403
  - \_coo64\_zo, 404
- Copying and Licence, 2
- CROUT
  - ffpack\_pluq.inl, 870
- CSC
  - FFLAS, 74
- CSC\_ZO
  - FFLAS, 74
- CSR
  - FFLAS, 74
- csr
  - HelperFlag, 440
- csr.h, 822
- CSR\_HYB
  - FFLAS, 74
- csr\_hyb.h, 827
- csr\_hyb\_pspmm.inl, 827
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL, 828
- csr\_hyb\_pspmv.inl, 828
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL, 828
- csr\_hyb\_spm.inl, 829

- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL, ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- [829](#)
- csr\_hyb\_spmv.inl, [829](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL, [829](#)
- csr\_hyb\_utils.inl, [830](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL, [830](#)
- csr\_pspmm.inl, [823](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL, [824](#)
- csr\_pspmv.inl, [824](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL, [824](#)
- csr\_spm.inl, [824](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spm\_INL, [825](#)
- csr\_spmv.inl, [826](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL, [826](#)
- csr\_utils.inl, [826](#)
- CSR\_ZO
- FFLAS, [74](#)
- CsrMat< Field >, [404](#)
- \_csr16, [404](#)
- \_csr16\_zo, [404](#)
- \_csr32, [404](#)
- \_csr32\_zo, [404](#)
- \_csr64, [404](#)
- \_csr64\_zo, [404](#)
- cst
- Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [693](#)
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- ctz
- bit\_manipulation.h, [970](#)
- CUBE
- arithprog.C, [711](#)
- autotune/charpoly.C, [739](#)
- autotune/pluq.C, [741](#)
- benchmark-checkers.C, [717](#)
- benchmark-fsyk.C, [732](#)
- benchmark-fsytrf.C, [732](#)
- benchmark-ftsri.C, [735](#)
- benchmark-inverse.C, [735](#)
- benchmark-lqp.C, [736](#)
- benchmark-pluq.C, [737](#)
- benchmark-wino.C, [738](#)
- fsyk.C, [712](#)
- fsytrf.C, [713](#)
- ftsri.C, [714](#)
- cuda.C, [979](#)
- main, [979](#)
- current
- ForStrategy1D< blocksize\_t, Cut, Param >, [429](#)
- ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- Cut
- Parallel< C, P >, [487](#)
- cyclic\_shift\_col
- FFPACK, [340](#), [343](#)
- cyclic\_shift\_col\_modular\_double
- ffpack.C, [918](#)
- ffpack\_c.h, [939](#)
- cyclic\_shift\_mathPerm
- FFPACK, [339](#)
- ffpack.C, [918](#)
- ffpack\_c.h, [939](#)
- cyclic\_shift\_row
- FFPACK, [339](#), [342](#)
- cyclic\_shift\_row\_col
- FFPACK, [339](#), [342](#)
- cyclic\_shift\_row\_modular\_double
- ffpack.C, [918](#)
- ffpack\_c.h, [939](#)
- Danilevski
- FFPACK, [329](#)
- FFPACK::Protected, [372](#)
- dasum\_
- config-blas.h, [750](#)
- dat
- Sparse< \_Field, SparseMatrix\_t::COO >, [681](#)
- Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
- Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [685](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [688](#)
- Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [691](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [693](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [696](#)
- Sparse< \_Field, SparseMatrix\_t::SELL >, [698](#)
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [700](#)
- data
- Argument, [379](#)
- daxpy\_
- config-blas.h, [750](#)
- dcopy\_
- config-blas.h, [752](#)
- ddot\_
- config-blas.h, [750](#)
- debug.h, [971](#)
- FFLASFFPACK\_abort, [971](#)
- FFLASFFPACK\_check, [971](#)
- DeCompressRows
- FFPACK::Protected, [375](#)
- DeCompressRowsQA
- FFPACK::Protected, [375](#)
- DeCompressRowsQK
- FFPACK::Protected, [375](#)
- DefaultBoundedTag, [405](#)
- DefaultTag, [405](#)

- delayed
  - RNSIntegerMod< RNS >, [512](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [681](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [685](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [690](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [693](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- DelayedField
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [465](#)
- delayedField
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [468](#)
- DelayedField\_t
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [465](#)
- DelayedTag, [405](#)
- deleted
  - Coo< Field >, [401](#)
- DENSE\_THRESHOLD
  - fflas\_sparse.h, [817](#)
- denseCols
  - StatsMatrix, [704](#)
- denseRows
  - StatsMatrix, [704](#)
- Det
  - FFPACK, [312](#), [313](#), [330](#), [349](#), [350](#)
- det.C, [740](#)
  - main, [740](#)
- Det\_modular\_double
  - ffpack.C, [928](#)
  - ffpack\_c.h, [947](#)
- deviationCol
  - StatsMatrix, [703](#)
- deviationColDifference
  - StatsMatrix, [703](#)
- deviationRow
  - StatsMatrix, [702](#)
- deviationRowDifference
  - StatsMatrix, [703](#)
- DFElt
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [465](#)
- dgemm\_
  - config-blas.h, [754](#)
  - fblas.C, [980](#)
- dgemv\_
  - config-blas.h, [751](#)
- dger\_
  - config-blas.h, [751](#)
- digits
  - limits< char >, [454](#)
  - limits< double >, [455](#)
  - limits< float >, [455](#)
  - limits< int >, [456](#)
  - limits< long >, [457](#)
  - limits< long long >, [458](#)
  - limits< short int >, [459](#)
  - limits< signed char >, [460](#)
  - limits< unsigned char >, [461](#)
  - limits< unsigned int >, [461](#)
  - limits< unsigned long >, [462](#)
  - limits< unsigned long long >, [462](#)
  - limits< unsigned short int >, [463](#)
- div
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [520](#)
  - Simd256\_impl< true, false, true, 8 >, [585](#)
  - Simd512\_impl< true, false, true, 8 >, [656](#)
- dnrm2\_
  - config-blas.h, [751](#)
- DNS\_BIN\_VER
  - read\_sparse.h, [841](#)
- doApplyS
  - FFPACK, [335](#)
- doApplyT
  - FFPACK, [336](#)
- DotProdBoundClassic
  - FFLAS::Protected, [201](#)
- DOUBLE\_TO\_FLOAT\_CROSSOVER
  - fflas.h, [763](#)
  - winograd.C, [715](#)
- dscal\_
  - config-blas.h, [752](#)
- dtrmm\_
  - config-blas.h, [753](#)
- dtrsm\_
  - config-blas.h, [752](#)
- DynamicPeeling
  - FFLAS::Protected, [206](#)
- DynamicPeeling2
  - FFLAS::Protected, [206](#)
- EFFGFF
  - benchmark-dsytrf.C, [720](#)
- Element
  - FieldSimd< \_Field >, [414](#)
  - readMyMachineType< Field, mpz\_t >, [491](#)
  - readMyMachineType< Field, T >, [490](#)
  - rns\_double, [492](#)
  - rns\_double\_extended, [504](#)
  - RNSInteger< RNS >, [508](#)
  - RNSIntegerMod< RNS >, [511](#)
- Element\_ptr
  - benchmark-fgemm-rns.C, [725](#)
  - readMyMachineType< Field, mpz\_t >, [491](#)
  - readMyMachineType< Field, T >, [490](#)

- rns\_double, [493](#)
- rns\_double\_extended, [504](#)
- RNSInteger< RNS >, [508](#)
- RNSIntegerMod< RNS >, [511](#)
- ElementTraits< double >, [406](#)
  - value, [406](#)
- ElementTraits< Element >, [405](#)
  - value, [405](#)
- ElementTraits< FFPACK::rns\_double\_elt >, [406](#)
  - value, [406](#)
- ElementTraits< float >, [406](#)
  - value, [406](#)
- ElementTraits< Givaro::Integer >, [406](#)
  - value, [407](#)
- ElementTraits< int16\_t >, [407](#)
  - value, [407](#)
- ElementTraits< int32\_t >, [407](#)
  - value, [407](#)
- ElementTraits< int64\_t >, [407](#)
  - value, [408](#)
- ElementTraits< int8\_t >, [408](#)
  - value, [408](#)
- ElementTraits< Reclnt::rint< K > >, [408](#)
  - value, [408](#)
- ElementTraits< Reclnt::rmint< K, MG > >, [408](#)
  - value, [408](#)
- ElementTraits< Reclnt::ruint< K > >, [409](#)
  - value, [409](#)
- ElementTraits< uint16\_t >, [409](#)
  - value, [409](#)
- ElementTraits< uint32\_t >, [409](#)
  - value, [409](#)
- ElementTraits< uint64\_t >, [409](#)
  - value, [410](#)
- ElementTraits< uint8\_t >, [410](#)
  - value, [410](#)
- ELL
  - FFLAS, [74](#)
- ell
  - HelperFlag, [440](#)
- ell.h, [830](#)
- ell\_pspmm.inl, [831](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL, [831](#)
- ell\_pspmv.inl, [831](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL, [832](#)
- ELL\_simd
  - FFLAS, [74](#)
- ell\_simd.h, [834](#)
- ell\_simd\_pspmv.inl, [835](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL, [836](#)
- ell\_simd\_spmv.inl, [836](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL, [836](#)
- ell\_simd\_utils.inl, [837](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL, [837](#)
- ELL\_simd\_ZO
  - FFLAS, [74](#)
- ell\_spm.inl, [832](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spm\_INL, [833](#)
- ell\_spmv.inl, [833](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL, [834](#)
- ell\_utils.inl, [834](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL, [834](#)
- ELL\_ZO
  - FFLAS, [74](#)
- ElIMat< Field >, [410](#)
  - \_ell16, [410](#)
  - \_ell16\_zo, [410](#)
  - \_ell32, [410](#)
  - \_ell32\_zo, [411](#)
  - \_ell64, [410](#)
  - \_ell64\_zo, [411](#)
- ENABLE\_ALL\_CHECKINGS
  - benchmark-checkers.C, [717](#)
  - ffpack\_ftrtr.inl, [865](#)
  - test-fdot.C, [993](#)
  - test-fgemm-check.C, [994](#)
  - test-fsyr2k.C, [1004](#)
  - test-fsyrk.C, [1006](#)
  - test-ftrmv.C, [1010](#)
  - test-ftrsm-check.C, [1011](#)
  - test-ftrsm.C, [1012](#)
  - test-ftrssyr2k.C, [1013](#)
  - test-ftrstr.C, [1014](#)
  - test-ftrsv.C, [1015](#)
  - test-ftrtri.C, [1016](#)
  - test-invert-check.C, [1017](#)
  - test-pluq-check.C, [1027](#)
- ENABLE\_CHECKER\_charpoly
  - test-charpoly-check.C, [985](#)
- ENABLE\_CHECKER\_Det
  - test-det-check.C, [988](#)
- ENABLE\_CHECKER\_fgemm
  - test-fgemm.C, [995](#)
- end
  - ForStrategy1D< blocksize\_t, Cut, Param >, [428](#)
- END\_OF\_ARGUMENTS
  - args-parser.h, [968](#)
- END\_PARALLEL\_MAIN
  - parallel.h, [962](#)
- eq
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [522](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [527](#)
- Simd128\_impl< true, true, false, 2 >, [536](#)
- Simd128\_impl< true, true, false, 4 >, [545](#)



- Simd128\_impl< true, true, false, 8 >, [554](#)
- Simd128\_impl< true, true, true, 2 >, [562](#)
- Simd128\_impl< true, true, true, 4 >, [569](#)
- Simd128\_impl< true, true, true, 8 >, [577](#)
- Simd256\_impl< true, false, true, 8 >, [586](#)
- Simd256\_impl< true, true, false, 2 >, [596](#)
- Simd256\_impl< true, true, false, 4 >, [611](#)
- Simd256\_impl< true, true, false, 8 >, [620](#)
- Simd256\_impl< true, true, true, 2 >, [628](#)
- Simd256\_impl< true, true, true, 4 >, [637](#), [642](#)
- Simd256\_impl< true, true, true, 8 >, [650](#)
- Simd512\_impl< true, false, true, 8 >, [657](#)
- Simd512\_impl< true, true, false, 8 >, [667](#)
- Simd512\_impl< true, true, true, 8 >, [675](#)
- eval\_func\_on\_array
  - test-simd.C, [1031](#)
- example
  - Argument, [378](#)
- examples/charpoly.C
  - main, [740](#)
- examples/pluq.C
  - main, [742](#)
- fadd
  - FFLAS, [76](#), [78](#), [80](#), [155](#), [156](#), [163](#), [164](#)
  - FFLAS::details, [190](#), [191](#)
- fadd\_1\_modular\_double
  - fflas\_c.h, [885](#)
  - fflas\_lvl1.C, [904](#)
- fadd\_2\_modular\_double
  - fflas\_c.h, [888](#)
  - fflas\_lvl2.C, [909](#)
- faddin
  - FFLAS, [76](#), [79](#), [156](#), [164](#)
- faddin\_1\_modular\_double
  - fflas\_c.h, [885](#)
  - fflas\_lvl1.C, [905](#)
- faddin\_2\_modular\_double
  - fflas\_c.h, [889](#)
  - fflas\_lvl2.C, [909](#)
- Failure, [411](#)
  - \_errorStream, [412](#)
  - Failure, [411](#)
  - operator(), [411](#), [412](#)
  - print, [412](#)
  - setErrorStream, [412](#)
- failure
  - FFPACK, [355](#)
- FailureCharpolyCheck, [412](#)
- FailureDetCheck, [413](#)
- FailureFgemmCheck, [413](#)
- FailureInvertCheck, [413](#)
- FailurePLUQCheck, [413](#)
- FailureTrsmCheck, [413](#)
- fassign
  - FFLAS, [80–82](#), [152](#), [156](#)
- fassign\_1\_modular\_double
  - fflas\_c.h, [884](#)
  - fflas\_lvl1.C, [903](#)
- fassign\_2\_modular\_double
  - fflas\_c.h, [886](#)
  - fflas\_lvl2.C, [906](#)
- faxpby
  - FFLAS, [126](#), [131](#)
- faxpy
  - FFLAS, [82](#), [83](#), [153](#), [161](#)
- faxpy\_1\_modular\_double
  - fflas\_c.h, [884](#)
  - fflas\_lvl1.C, [904](#)
- faxpy\_2\_modular\_double
  - fflas\_c.h, [888](#)
  - fflas\_lvl2.C, [908](#)
- fblas.C, [979](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [980](#)
  - dgemm\_, [980](#)
  - main, [980](#)
- fconvert
  - FFLAS, [123](#), [130](#), [149](#)
- fconvert\_rns
  - FFLAS, [147](#), [148](#)
- fconvert\_trans\_rns
  - FFLAS, [147](#)
- fdot
  - FFLAS, [84](#), [85](#), [126](#), [154](#), [173](#)
- fdot\_1\_modular\_double
  - fflas\_c.h, [884](#)
  - fflas\_lvl1.C, [904](#)
- fequal
  - FFLAS, [125](#), [128](#), [151](#), [157](#)
- fequal\_1\_modular\_double
  - fflas\_c.h, [883](#)
  - fflas\_lvl1.C, [903](#)
- fequal\_2\_modular\_double
  - fflas\_c.h, [886](#)
  - fflas\_lvl2.C, [907](#)
- FFLAS, [42](#), [45](#)
  - alignable, [179](#)
  - alignable< Givaro::Integer \* >, [179](#)
  - BaseTimer, [72](#)
  - bitsize, [132](#)
  - bitsize< Givaro::ZRing< Givaro::Integer > >, [132](#)
  - BlockCuts, [170](#), [173](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >, [172](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >, [171](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >, [172](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >, [171](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >, [172](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >, [172](#)
  - BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >, [171](#)



BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >, 171  
BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >, 172  
BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >, 171  
cblas\_imprsm, 121  
Checker\_fgemm, 70  
Checker\_ftrsm, 70  
computeDeviation, 145  
COO, 74  
COO\_ZO, 74  
CSC, 74  
CSC\_ZO, 74  
CSR, 74  
CSR\_HYB, 74  
CSR\_ZO, 74  
ELL, 74  
ELL\_simd, 74  
ELL\_simd\_ZO, 74  
ELL\_ZO, 74  
fadd, 76, 78, 80, 155, 156, 163, 164  
faddin, 76, 79, 156, 164  
fassign, 80–82, 152, 156  
faxpby, 126, 131  
faxpy, 82, 83, 153, 161  
fconvert, 123, 130, 149  
fconvert\_rns, 147, 148  
fconvert\_trans\_rns, 147  
fdot, 84, 85, 126, 154, 173  
fequal, 125, 128, 151, 157  
FFLAS\_BASE, 73  
fflas\_delete, 146, 180  
FFLAS\_DIAG, 73  
FFLAS\_FORMAT, 74  
fflas\_new, 146, 147, 179, 180  
FFLAS\_ORDER, 72  
FFLAS\_SIDE, 73  
FFLAS\_TRANSPOSE, 72  
FFLAS\_UPLO, 73  
FflasAuto, 75  
FflasBinary, 75  
FflasColMajor, 72  
FflasDense, 75  
FflasDouble, 74  
FflasFloat, 74  
FflasGeneric, 74  
FflasLeft, 73  
FflasLower, 73  
FflasMaple, 75  
FflasMath, 75  
FflasNonUnit, 73  
FflasNoTrans, 73  
FflasRight, 73  
FflasRowMajor, 72  
FflasSageMath, 75  
FflasSMS, 75  
FflasTrans, 73  
FflasUnit, 73  
FflasUpper, 73  
fgemm, 86–88, 90–95, 136, 168, 169  
fgemv, 95–100, 165, 176  
fger, 101–104, 165  
fidentity, 129, 158  
finit, 106, 108, 122, 129, 149, 159  
finit\_rns, 146, 148  
finit\_trans\_rns, 147  
fiszero, 125, 128, 151, 157  
fmove, 132, 162  
fneg, 124, 131, 150, 160  
fnegin, 123, 130, 150, 160  
ForceCheck\_fgemm, 70  
ForceCheck\_ftrsm, 70  
frand, 125, 128  
freduce, 105–108, 148, 149, 158, 159  
freduce\_constoverride, 106, 108  
freivalds, 109  
fscal, 110–114, 153, 161  
fscaln, 109, 111–113, 152, 160  
fspmm, 137  
fspmv, 137, 144  
fsquare, 89, 90, 170  
fsub, 76, 79, 155, 163  
fsubin, 76, 79, 164  
fswap, 127, 155  
fsyr2k, 114  
fsyrk, 115–117  
ftrmm, 118, 119, 167  
ftrmv, 133  
ftrsm, 119–121, 133, 136, 137, 167  
ftrsv, 121, 166  
fzero, 124, 127, 151, 157  
getDataTypes, 143, 144  
getSeed, 181  
getStat, 146  
getTLBSize, 180  
has\_equal, 71  
has\_minus, 71  
has\_minus\_eq, 71  
has\_mul, 71  
has\_mul\_eq, 72  
has\_plus, 71  
has\_plus\_eq, 71  
HYB\_ZO, 74  
igemm\_, 122  
InfNorm, 75  
max3, 75  
max4, 75  
min3, 75  
min4, 75  
MKLSparseMatrixFormat, 71  
none, 74  
NoSimdSparseMatrix, 71  
NotMKLSparseMatrixFormat, 71  
NotZOSparseMatrix, 70  
number\_kind, 74



- Winograd, 183
- Winograd\_L\_S, 187
- Winograd\_LR\_S, 186
- Winograd\_R\_S, 187
- WinogradAcc\_2\_24, 185
- WinogradAcc\_2\_27, 185
- WinogradAcc\_3\_21, 184
- WinogradAcc\_3\_23, 184
- WinogradAcc\_L\_S, 186
- WinogradAcc\_LR, 185
- WinogradAcc\_R\_S, 186
- WinoPar, 183
- FFLAS::csr\_hyb\_details, 188
- FFLAS::CuttingStrategy, 188
  - RNSModulus, 188
- FFLAS::details, 188
  - BlockingFactor, 196
  - fadd, 190, 191
  - freduce, 191, 192
  - fscal, 192, 193
  - fscaln, 192
  - gebp, 195
  - igebb11, 194
  - igebb14, 193
  - igebb21, 194
  - igebb24, 193
  - igebb41, 194
  - igebb44, 193
  - igebp, 195
  - pack\_lhs, 195
  - pack\_rhs, 195
- FFLAS::details\_spmv, 196
- FFLAS::ElementCategories, 196
- FFLAS::FieldCategories, 196
- FFLAS::MMHelperAlgo, 197
- FFLAS::ModeCategories, 197
- FFLAS::ParSeqHelper, 198
- FFLAS::Protected, 198
  - computeFactorClassic, 201
  - DotProdBoundClassic, 201
  - DynamicPeeling, 206
  - DynamicPeeling2, 206
  - fgemm\_convert, 202
  - fgemv\_convert, 207
  - fger\_convert, 207
  - fsquareCommon, 204
  - igemm, 209
  - igemm\_colmajor, 209
  - MatF2MatD\_Triangular, 210
  - MatF2MatFI\_Triangular, 210
  - min\_types, 207, 208
  - NeedDoublePreAddReduction, 203, 204
  - NeedPreAddReduction, 202, 203
  - NeedPreSubReduction, 203
  - ScalAndReduce, 204
  - TRSMBound, 201, 202
  - unfit, 208, 209
  - WinogradCalc, 207
  - WinogradSteps, 205
  - WinogradThreshold, 205
- FFLAS::sell\_details, 211
- FFLAS::sparse\_details, 211
  - fspmm, 217–219
  - fspmm\_dispatch, 216, 217
  - fspmv, 214–216, 224
  - fspmv\_dispatch, 214
  - init\_y, 214
  - pfspmm, 220–222
  - pfspmm\_dispatch, 219, 220
  - pfspmv, 222, 223
- FFLAS::sparse\_details\_impl, 225
  - fspmm, 233, 240, 241, 246, 247, 251, 260
  - fspmm\_mone, 234, 242, 252
  - fspmm\_mone\_simd\_aligned, 235, 242, 253
  - fspmm\_mone\_simd\_unaligned, 235, 243, 253
  - fspmm\_one, 234, 241, 252
  - fspmm\_one\_simd\_aligned, 235, 242, 253
  - fspmm\_one\_simd\_unaligned, 235, 242, 253
  - fspmm\_simd\_aligned, 234, 241
  - fspmm\_simd\_unaligned, 234, 241
  - fspmv, 236, 243, 247, 254, 256, 257, 261, 263
  - fspmv\_mone, 236, 237, 244, 254, 255, 258, 264, 265
  - fspmv\_mone\_simd, 258, 264
  - fspmv\_one, 236, 237, 243, 244, 254, 255, 257, 258, 264
  - fspmv\_one\_simd, 258, 264
  - fspmv\_simd, 257, 263
  - pfspmm, 237, 244, 245, 247–249, 259
  - pfspmm\_mone, 238
  - pfspmm\_one, 238
  - pfspmm\_zo, 249
  - pfspmv, 239, 246, 249, 250, 255, 259–262
  - pfspmv\_mone, 240, 250, 251, 256, 262
  - pfspmv\_one, 239, 240, 250, 256, 262
  - pfspmv\_task, 239
- FFLAS::StrategyParameter, 265
- FFLAS::StructureHelper, 265
- FFLAS::vectorised, 266
  - add, 268
  - addp, 267
  - modp, 269, 270
  - reduce, 268, 269
  - scalp, 270
  - sub, 268
  - subp, 267
  - VEC\_ADD, 267
  - VEC\_SUB, 267
- FFLAS::vectorised::unswitch, 270
  - modp, 271
  - scalp, 271
- fflas\_101.C, 1035
  - main, 1036
- fflas\_101\_lm1.C, 1036
  - main, 1036
- FFLAS\_BASE

- FFLAS, 73
- fflas\_bounds.inl, 764
  - \_\_FFLASFFPACK\_fflas\_bounds\_INL, 764
  - FFLAS\_INT\_TYPE, 764
- fflas\_c.h, 879
  - fadd\_1\_modular\_double, 885
  - fadd\_2\_modular\_double, 888
  - faddin\_1\_modular\_double, 885
  - faddin\_2\_modular\_double, 889
  - fassign\_1\_modular\_double, 884
  - fassign\_2\_modular\_double, 886
  - faxpy\_1\_modular\_double, 884
  - faxpy\_2\_modular\_double, 888
  - fdot\_1\_modular\_double, 884
  - fequal\_1\_modular\_double, 883
  - fequal\_2\_modular\_double, 886
  - FFLAS\_C\_BASE, 882
  - FFLAS\_C\_DIAG, 882
  - FFLAS\_C\_ORDER, 881
  - FFLAS\_C\_SIDE, 882
  - FFLAS\_C\_TRANSPOSE, 882
  - FFLAS\_C\_UPLO, 882
  - FFLAS\_COMPILED, 881
  - FflasColMajor, 881
  - FflasDouble, 882
  - FflasFloat, 882
  - FflasGeneric, 882
  - FflasLeft, 882
  - FflasLower, 882
  - FflasNonUnit, 882
  - FflasNoTrans, 882
  - FflasRight, 882
  - FflasRowMajor, 881
  - FflasTrans, 882
  - FflasUnit, 882
  - FflasUpper, 882
  - fgemm\_3\_modular\_double, 890
  - fgemv\_2\_modular\_double, 889
  - fger\_2\_modular\_double, 889
  - fidentity\_2\_modular\_double, 886
  - fiszero\_1\_modular\_double, 883
  - fiszero\_2\_modular\_double, 886
  - fmove\_2\_modular\_double, 888
  - fneg\_1\_modular\_double, 883
  - fneg\_2\_modular\_double, 887
  - fnegin\_1\_modular\_double, 883
  - fnegin\_2\_modular\_double, 887
  - freduce\_1\_modular\_double, 883
  - freduce\_2\_modular\_double, 887
  - freducein\_1\_modular\_double, 883
  - freducein\_2\_modular\_double, 887
  - fscal\_1\_modular\_double, 884
  - fscal\_2\_modular\_double, 887
  - fscalin\_1\_modular\_double, 884
  - fscalin\_2\_modular\_double, 887
  - fsquare\_3\_modular\_double, 891
  - fsub\_1\_modular\_double, 885
  - fsub\_2\_modular\_double, 888
  - fsubin\_1\_modular\_double, 885
  - fsubin\_2\_modular\_double, 889
  - fswap\_1\_modular\_double, 885
  - ftmm\_3\_modular\_double, 890
  - ftsm\_3\_modular\_double, 890
  - ftsv\_2\_modular\_double, 890
  - fzero\_1\_modular\_double, 883
  - fzero\_2\_modular\_double, 886
- FFLAS\_C\_BASE
  - fflas\_c.h, 882
- FFLAS\_C\_DIAG
  - fflas\_c.h, 882
  - ffpack\_c.h, 937
- FFLAS\_C\_ORDER
  - fflas\_c.h, 881
  - ffpack\_c.h, 936
- FFLAS\_C\_SIDE
  - fflas\_c.h, 882
  - ffpack\_c.h, 937
- FFLAS\_C\_TRANSPOSE
  - fflas\_c.h, 882
  - ffpack\_c.h, 936
- FFLAS\_C\_UPLO
  - fflas\_c.h, 882
  - ffpack\_c.h, 936
- FFLAS\_COMPILED
  - fflas\_c.h, 881
  - ffpack\_inst.C, 953
  - ffpack\_inst.h, 954
- fflas\_const\_cast
  - FFPACK, 342, 355
- fflas\_delete
  - FFLAS, 146, 180
- FFLAS\_DIAG
  - FFLAS, 73
- FFLAS\_ELT
  - fflas\_L1\_inst.C, 892
  - fflas\_L1\_inst.h, 893
  - fflas\_L2\_inst.C, 895
  - fflas\_L2\_inst.h, 896
  - fflas\_L3\_inst.C, 899
  - fflas\_L3\_inst.h, 900
  - ffpack\_inst.C, 953, 954
  - ffpack\_inst.h, 954, 955
- fflas\_enum.h, 765
- fflas\_fadd.h, 765
- fflas\_fadd.inl, 767
  - \_\_FFLASFFPACK\_fadd\_INL, 768
- fflas\_fassign.h, 768
- fflas\_fassign.inl, 768
  - \_\_FFLASFFPACK\_fassign\_INL, 768
- fflas\_faxpy.inl, 769
  - \_\_FFLASFFPACK\_faxpy\_INL, 769
- fflas\_fdot.inl, 769
  - \_\_FFLASFFPACK\_fdot\_INL, 770
- fflas\_fgemm.inl, 770
  - \_\_FFLASFFPACK\_fgemm\_INL, 772
- fflas\_fgemv.inl, 779

- \_\_FFLASFFPACK\_fgmv\_INL, 780
- fflas\_fgmv\_mp.inl, 780
- \_\_FFLASFFPACK\_fgmv\_mp\_INL, 781
- fflas\_fger.inl, 781
- \_\_FFLASFFPACK\_fger\_INL, 782
- fflas\_fger\_mp.inl, 782
- \_\_FFPACK\_fger\_mp\_INL, 783
- FFLAS\_FIELD
  - fflas\_L1\_inst.C, 891, 892
  - fflas\_L1\_inst.h, 892, 893
  - fflas\_L2\_inst.C, 895
  - fflas\_L2\_inst.h, 896
  - fflas\_L3\_inst.C, 899
  - fflas\_L3\_inst.h, 900
  - ffpack\_inst.C, 953
  - ffpack\_inst.h, 954
- FFLAS\_FORMAT
  - FFLAS, 74
- fflas\_freduce.h, 783
- fflas\_freduce.inl, 784
- \_\_FFLASFFPACK\_fflas\_freduce\_INL, 785
- FFLASFFPACK\_COPY\_REDUCE, 785
- fflas\_freduce\_mp.inl, 785
- \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL, 786
- fflas\_freivalds.inl, 786
- \_\_FFLASFFPACK\_freivalds\_INL, 786
- fflas\_fscal.h, 786
- fflas\_fscal.inl, 786
- \_\_FFLASFFPACK\_fscal\_INL, 788
- fflas\_fscal\_mp.inl, 788
- \_\_FFLASFFPACK\_fscal\_mp\_INL, 788
- fflas\_fsyr2k.inl, 788
- \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL, 789
- fflas\_fsyrk.inl, 789
- \_\_FFLASFFPACK\_fflas\_fsyrk\_INL, 790
- fflas\_ftrmm.inl, 790
- \_\_FFLASFFPACK\_ftrmm\_INL, 790
- fflas\_ftrsm.inl, 790
- \_\_FFLASFFPACK\_ftrsm\_INL, 791
- fflas\_ftrsm\_mp.inl, 791
- \_\_FFPACK\_ftrsm\_mp\_INL, 791
- fflas\_ftrsv.inl, 792
- \_\_FFLASFFPACK\_ftrsv\_INL, 792
- fflas\_helpers.inl, 792
- \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL, 793
- FFLAS\_INT\_TYPE
  - fflas\_bounds.inl, 764
- fflas\_intrinsic.h, 972
- fflas\_io.h, 972
- fflas\_L1\_inst.C, 891
- \_\_FFLAS\_L1\_INST\_C, 891
- FFLAS\_ELT, 892
- FFLAS\_FIELD, 891, 892
- INST\_OR\_DECL, 891
- fflas\_L1\_inst.h, 892
- FFLAS\_ELT, 893
- FFLAS\_FIELD, 892, 893
- INST\_OR\_DECL, 892
- fflas\_L1\_inst\_implem.inl, 893
- fflas\_L2\_inst.C, 894
- \_\_FFLAS\_L2\_INST\_C, 895
- FFLAS\_ELT, 895
- FFLAS\_FIELD, 895
- INST\_OR\_DECL, 895
- fflas\_L2\_inst.h, 895
- FFLAS\_ELT, 896
- FFLAS\_FIELD, 896
- INST\_OR\_DECL, 896
- fflas\_L2\_inst\_implem.inl, 896
- fflas\_L3\_inst.C, 898
- \_\_FFLAS\_L3\_INST\_C, 898
- FFLAS\_ELT, 899
- FFLAS\_FIELD, 899
- INST\_OR\_DECL, 898
- fflas\_L3\_inst.h, 899
- FFLAS\_ELT, 900
- FFLAS\_FIELD, 900
- INST\_OR\_DECL, 900
- fflas\_L3\_inst\_implem.inl, 900
- \_\_FFLAS\_\_TRSM\_READONLY, 901
- fflas\_level1.inl, 796
- \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL, 799
- fflas\_level2.inl, 799
- \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL, 801
- fflas\_level3.inl, 801
- \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL, 803
- \_\_FFLAS\_\_TRSM\_READONLY, 803
- fflas\_lvl1.C, 901
- fadd\_1\_modular\_double, 904
- faddin\_1\_modular\_double, 905
- fassign\_1\_modular\_double, 903
- faxpy\_1\_modular\_double, 904
- fdot\_1\_modular\_double, 904
- fequal\_1\_modular\_double, 903
- fiszero\_1\_modular\_double, 903
- fneg\_1\_modular\_double, 902
- fnegin\_1\_modular\_double, 902
- freduce\_1\_modular\_double, 902
- freducein\_1\_modular\_double, 902
- fscal\_1\_modular\_double, 903
- fscalin\_1\_modular\_double, 903
- fsub\_1\_modular\_double, 904
- fsubin\_1\_modular\_double, 905
- fswap\_1\_modular\_double, 904
- fzero\_1\_modular\_double, 903
- fflas\_lvl2.C, 905
- fadd\_2\_modular\_double, 909
- faddin\_2\_modular\_double, 909
- fassign\_2\_modular\_double, 906
- faxpy\_2\_modular\_double, 908
- fequal\_2\_modular\_double, 907
- fgemv\_2\_modular\_double, 910
- fger\_2\_modular\_double, 910
- fidentity\_2\_modular\_double, 907
- fiszero\_2\_modular\_double, 907
- fmove\_2\_modular\_double, 909

- fneg\_2\_modular\_double, 908
- fnegin\_2\_modular\_double, 908
- freduce\_2\_modular\_double, 907
- freducein\_2\_modular\_double, 907
- fscal\_2\_modular\_double, 908
- fscalin\_2\_modular\_double, 908
- fsub\_2\_modular\_double, 909
- fsubin\_2\_modular\_double, 909
- ftsv\_2\_modular\_double, 910
- fzero\_2\_modular\_double, 906
- fflas\_lvl3.C, 911
  - fgemm\_3\_modular\_double, 912
  - fsquare\_3\_modular\_double, 912
  - ftmm\_3\_modular\_double, 911
  - ftsm\_3\_modular\_double, 911
- fflas\_memory.h, 973
- fflas\_new
  - FFLAS, 146, 147, 179, 180
- FFLAS\_ORDER
  - FFLAS, 72
- fflas\_pfgemm.inl, 804
  - \_\_FFLASFFPACK\_DIMKPENALTY, 804
  - \_\_FFLASFFPACK\_SEQPARTHRESHOLD, 804
  - \_\_FFLASFFPACK\_fflas\_pfgemm\_INL, 804
- fflas\_pftrsm.inl, 804
  - \_\_FFLASFFPACK\_fflas\_pftrsm\_INL, 805
  - PTRSM\_HYBRID\_THRESHOLD, 805
- fflas\_plevel1.h, 960
- fflas\_randommatrix.h, 973
- FFLAS\_SIDE
  - FFLAS, 73
- fflas\_simd.h, 805
  - CONST, 806
  - FLOAT\_MOD, 806
  - INLINE, 806
  - NORML\_MOD, 806
  - PURE, 806
  - Simd, 807
  - SIMD\_INT, 806
- fflas\_sparse.C, 912
- fflas\_sparse.h, 813
  - \_\_FFLASFFPACK\_CACHE\_LINE\_SIZE, 817
  - assume\_aligned, 817
  - DENSE\_THRESHOLD, 817
  - index\_t, 817
  - ROUND\_DOWN, 817
- fflas\_sparse.inl, 818
  - \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL, 819
- FFLAS\_TRANSPOSE
  - FFLAS, 72
- FFLAS\_UPLO
  - FFLAS, 73
- FflasAuto
  - FFLAS, 75
- FflasBinary
  - FFLAS, 75
- FflasColMajor
  - FFLAS, 72
- fflas\_c.h, 881
- ffpack\_c.h, 936
- FflasDense
  - FFLAS, 75
- FflasDouble
  - FFLAS, 74
  - fflas\_c.h, 882
- FFLASFFPACK\_abort
  - debug.h, 971
- FFLASFFPACK\_check
  - debug.h, 971
- FFLASFFPACK\_checkers\_fflas\_inl\_H
  - checkers\_fflas.inl, 746
- FFLASFFPACK\_checkers\_ffpack\_inl\_H
  - checkers\_ffpack.inl, 748
- FFLASFFPACK\_COPY\_REDUCE
  - fflas\_freduce.inl, 785
- FFLASFFPACK\_PERM\_BKSIZE
  - ffpack\_permutation.inl, 870
- FflasFloat
  - FFLAS, 74
  - fflas\_c.h, 882
- FflasGeneric
  - FFLAS, 74
  - fflas\_c.h, 882
- FflasLeft
  - FFLAS, 73
  - fflas\_c.h, 882
  - ffpack\_c.h, 937
- FflasLower
  - FFLAS, 73
  - fflas\_c.h, 882
  - ffpack\_c.h, 937
- FflasMaple
  - FFLAS, 75
- FflasMath
  - FFLAS, 75
- FflasNonUnit
  - FFLAS, 73
  - fflas\_c.h, 882
  - ffpack\_c.h, 937
- FflasNoTrans
  - FFLAS, 73
  - fflas\_c.h, 882
  - ffpack\_c.h, 936
- FflasRight
  - FFLAS, 73
  - fflas\_c.h, 882
  - ffpack\_c.h, 937
- FflasRowMajor
  - FFLAS, 72
  - fflas\_c.h, 881
  - ffpack\_c.h, 936
- FflasSageMath
  - FFLAS, 75
- FflasSMS
  - FFLAS, 75
- FflasTrans

- FFLAS, [73](#)
- fflas\_c.h, [882](#)
- ffpack\_c.h, [936](#)
- FflasUnit
  - FFLAS, [73](#)
  - fflas\_c.h, [882](#)
  - ffpack\_c.h, [937](#)
- FflasUpper
  - FFLAS, [73](#)
  - fflas\_c.h, [882](#)
  - ffpack\_c.h, [937](#)
- FFPACK, [43](#), [272](#)
  - \_PLUQ, [341](#)
  - applyP, [289](#), [290](#), [343](#)
  - applyP\_block, [335](#)
  - buildMatrix, [329](#)
  - CharPoly, [307](#), [308](#), [329](#), [347](#), [348](#)
  - Checker\_charpoly, [288](#)
  - Checker\_Det, [287](#)
  - Checker\_invert, [287](#)
  - Checker\_PLUQ, [287](#)
  - chooseField, [368](#)
  - chooseField< Givaro::ZRing< double > >, [368](#)
  - chooseField< Givaro::ZRing< float > >, [368](#)
  - chooseField< Givaro::ZRing< int32\_t > >, [368](#)
  - chooseField< Givaro::ZRing< int64\_t > >, [368](#)
  - ColRankProfileSubmatrix, [320](#), [352](#)
  - ColRankProfileSubmatrixIndices, [319](#), [352](#)
  - ColumnEchelonForm, [301](#), [346](#)
  - ColumnRankProfile, [316](#), [317](#), [351](#)
  - composePermutationsLLL, [338](#)
  - composePermutationsLLM, [338](#)
  - composePermutationsMLM, [339](#)
  - cyclic\_shift\_col, [340](#), [343](#)
  - cyclic\_shift\_mathPerm, [339](#)
  - cyclic\_shift\_row, [339](#), [342](#)
  - cyclic\_shift\_row\_col, [339](#), [342](#)
  - Danilevski, [329](#)
  - Det, [312](#), [313](#), [330](#), [349](#), [350](#)
  - doApplyS, [335](#)
  - doApplyT, [336](#)
  - failure, [355](#)
  - fflas\_const\_cast, [342](#), [355](#)
  - fgesv, [292](#), [293](#), [344](#)
  - fgetrs, [291](#), [292](#), [343](#)
  - ForceCheck\_charpoly, [288](#)
  - ForceCheck\_Det, [288](#)
  - ForceCheck\_invert, [288](#)
  - ForceCheck\_PLUQ, [288](#)
  - fsytrf, [296](#), [297](#)
  - fsytrf\_BC\_Crout, [330](#)
  - fsytrf\_BC\_RL, [330](#)
  - fsytrf\_LOW\_RPM\_BC\_Crout, [331](#)
  - fsytrf\_nonunit, [297](#), [332](#)
  - fsytrf\_RPM, [332](#)
  - fsytrf\_UP\_RPM, [331](#)
  - fsytrf\_UP\_RPM\_BC\_Crout, [331](#)
  - fsytrf\_UP\_RPM\_BC\_RL, [331](#)
  - ftsrssyr2k, [296](#)
  - ftstr, [295](#)
  - fttrtri, [294](#), [344](#)
  - fttrrm, [295](#), [345](#)
  - getEchelonForm, [322](#), [323](#)
  - getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >, [353](#)
  - getEchelonTransform, [324](#)
  - getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >, [353](#)
  - getReducedEchelonForm, [324](#), [325](#)
  - getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >, [354](#)
  - getReducedEchelonTransform, [326](#)
  - getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >, [354](#)
  - getTriangular, [321](#), [322](#)
  - getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >, [352](#), [353](#)
  - getTridiagonal, [332](#)
  - Invert, [305](#), [306](#), [347](#)
  - Invert2, [306](#), [347](#)
  - isOdd, [355](#)
  - IsSingular, [311](#), [349](#)
  - KrylovElim, [349](#)
  - LAPACKPerm2MathPerm, [288](#)
  - LeadingSubmatrixRankProfiles, [318](#)
  - LQUPtoInverseOfFullRankMinor, [327](#), [355](#)
  - LUdivine, [300](#), [333](#), [334](#), [345](#)
  - LUdivine\_gauss, [333](#), [346](#)
  - LUdivine\_small, [333](#), [345](#)
  - MathPerm2LAPACKPerm, [288](#)
  - MatrixApplyS, [335](#), [336](#)
  - MatrixApplyT, [337](#)
  - MatVecMinPoly, [310](#), [348](#)
  - maxFieldElt, [368](#)
  - maxFieldElt< Givaro::ZRing< Givaro::Integer > >, [368](#)
  - MinPoly, [309](#), [348](#)
  - MonotonicApplyP, [290](#)
  - MonotonicCompress, [334](#)
  - MonotonicCompressCycles, [334](#)
  - MonotonicCompressMorePivots, [334](#)
  - MonotonicExpand, [335](#)
  - NonZeroRandomMatrix, [355](#), [356](#)
  - NullSpaceBasis, [314](#), [351](#)
  - pColumnEchelonForm, [301](#)
  - pColumnRankProfile, [317](#)
  - pDet, [312](#)
  - PermApplyS, [336](#)
  - PermApplyT, [338](#)
  - PLUQ, [298](#), [299](#), [341](#), [342](#), [345](#)
  - PLUQ\_basecaseCrout, [340](#)
  - PLUQ\_basecaseV2, [340](#)
  - PLUQ\_basecaseV3, [340](#)
  - PLUQtoEchelonPermutation, [327](#)
  - pPLUQ, [299](#)
  - pRank, [311](#)



- pReducedColumnEchelonForm, 304
- pReducedRowEchelonForm, 305
- pRowEchelonForm, 302
- pRowRankProfile, 316
- pSolve, 314
- RandInt, 359
- RandomIndexSubset, 361
- RandomMatrix, 357
- RandomMatrixWithDet, 366, 367
- RandomMatrixWithRank, 359, 360
- RandomMatrixWithRankandRandomRPM, 364, 365
- RandomMatrixWithRankandRPM, 362, 363
- RandomNullSpaceVector, 314, 327, 351
- RandomPermutation, 361
- RandomRankProfileMatrix, 361
- RandomSymmetricMatrix, 359
- RandomSymmetricMatrixWithRankandRandomRPM, 365, 366
- RandomSymmetricMatrixWithRankandRPM, 363, 364
- RandomSymmetricRankProfileMatrix, 362
- RandomTriangularMatrix, 358
- Rank, 310, 311, 349
- RankProfileFromLU, 317
- ReducedColumnEchelonForm, 303, 304, 347
- ReducedRowEchelonForm, 304, 305, 346
- RowEchelonForm, 302, 303, 346
- RowRankProfile, 315, 316, 351
- RowRankProfileSubmatrix, 320, 352
- RowRankProfileSubmatrixIndices, 318, 352
- Solve, 313, 350
- solveLB, 328, 350
- solveLB2, 328, 350
- SpecRankProfile, 349
- swapval, 362
- threads\_fgemm, 341
- threads\_ftsm, 341
- trinv\_left, 294, 345
- ffpack-fgesv.C, 1036
  - main, 1036
- ffpack-solve.C, 1037
  - main, 1037
- ffpack.C, 913
  - applyP\_modular\_double, 918
  - ColRankProfileSubmatrix\_modular\_double, 930
  - ColRankProfileSubmatrixIndices\_modular\_double, 930
  - ColumnEchelonForm\_modular\_double, 921
  - ColumnEchelonForm\_modular\_float, 921
  - ColumnEchelonForm\_modular\_int32\_t, 922
  - ColumnRankProfile\_modular\_double, 929
  - composePermutationsLLL, 917
  - composePermutationsLLM, 917
  - composePermutationsMLM, 917
  - cyclic\_shift\_col\_modular\_double, 918
  - cyclic\_shift\_mathPerm, 918
  - cyclic\_shift\_row\_modular\_double, 918
  - Det\_modular\_double, 928
  - fgesv\_modular\_double, 919
  - fgesvin\_modular\_double, 919
  - fgetrsin\_modular\_double, 918
  - fgetrsv\_modular\_double, 919
  - fttrtri\_modular\_double, 919
  - fttrrm\_modular\_double, 920
  - getEchelonForm\_modular\_double, 931
  - getEchelonFormin\_modular\_double, 931
  - getEchelonTransform\_modular\_double, 932
  - getReducedEchelonForm\_modular\_double, 932
  - getReducedEchelonFormin\_modular\_double, 932
  - getReducedEchelonTransform\_modular\_double, 932
  - getTriangular\_modular\_double, 931
  - getTriangularin\_modular\_double, 931
  - Invert2\_modular\_double, 927
  - Invert\_modular\_double, 926
  - Invertin\_modular\_double, 926
  - IsSingular\_modular\_double, 927
  - KrylovElim\_modular\_double, 927
  - LAPACKPerm2MathPerm, 916
  - LeadingSubmatrixRankProfiles, 929
  - LUdivine\_modular\_double, 920
  - MathPerm2LAPACKPerm, 916
  - MatrixApplyS\_modular\_double, 916
  - MatrixApplyT\_modular\_double, 917
  - NullSpaceBasis\_modular\_double, 929
  - pColumnEchelonForm\_modular\_double, 923
  - pColumnEchelonForm\_modular\_float, 924
  - pColumnEchelonForm\_modular\_int32\_t, 925
  - PermApplyS\_double, 917
  - PermApplyT\_double, 917
  - PLUQ\_modular\_double, 920
  - PLUQtoEchelonPermutation, 933
  - pReducedColumnEchelonForm\_modular\_double, 924
  - pReducedColumnEchelonForm\_modular\_float, 925
  - pReducedColumnEchelonForm\_modular\_int32\_t, 926
  - pReducedRowEchelonForm\_modular\_double, 924
  - pReducedRowEchelonForm\_modular\_float, 925
  - pReducedRowEchelonForm\_modular\_int32\_t, 926
  - pRowEchelonForm\_modular\_double, 924
  - pRowEchelonForm\_modular\_float, 925
  - pRowEchelonForm\_modular\_int32\_t, 925
  - RandomNullSpaceVector\_modular\_double, 928
  - Rank\_modular\_double, 927
  - RankProfileFromLU, 929
  - ReducedColumnEchelonForm\_modular\_double, 921
  - ReducedColumnEchelonForm\_modular\_float, 922
  - ReducedColumnEchelonForm\_modular\_int32\_t, 923
  - ReducedRowEchelonForm\_modular\_double, 921
  - ReducedRowEchelonForm\_modular\_float, 922
  - ReducedRowEchelonForm\_modular\_int32\_t, 923



- RowEchelonForm\_modular\_double, 921
- RowEchelonForm\_modular\_float, 922
- RowEchelonForm\_modular\_int32\_t, 923
- RowRankProfile\_modular\_double, 929
- RowRankProfileSubmatrix\_modular\_double, 930
- RowRankProfileSubmatrixIndices\_modular\_double, 930
- Solve\_modular\_double, 928
- solveLB2\_modular\_double, 928
- solveLB\_modular\_double, 928
- SpecRankProfile\_modular\_double, 927
- trinv\_left\_modular\_double, 920
- ffpack.dox, 845
- ffpack.h, 845
  - \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD, 853
  - \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD, 853
- ffpack.inl, 854
  - \_\_FFLASFFPACK\_ffpack\_INL, 855
- FFPACK::Protected, 369
  - ArithProg, 373
  - CompressRows, 374
  - CompressRowsQA, 375
  - CompressRowsQK, 375
  - Danilevski, 372
  - DeCompressRows, 375
  - DeCompressRowsQA, 375
  - DeCompressRowsQK, 375
  - fgemv\_kgf, 372
  - GaussJordan, 371
  - Hybrid\_KGF\_LUK\_MinPoly, 374
  - KellerGehrig, 371
  - KGFast, 371
  - KGFast\_generalized, 372
  - LUdivine\_construct, 370, 376
  - LUKrylov, 372
  - LUKrylov\_KGFast, 373
  - MatVecMinPoly, 373
  - newD, 374
  - RandomKrylovPrecond, 373
  - updatedD, 374
- ffpack\_c.h, 933
  - applyP\_modular\_double, 940
  - ColRankProfileSubmatrix\_modular\_double, 950
  - ColRankProfileSubmatrixIndices\_modular\_double, 950
  - ColumnEchelonForm\_modular\_double, 943
  - ColumnEchelonForm\_modular\_float, 943
  - ColumnEchelonForm\_modular\_int32\_t, 943
  - ColumnRankProfile\_modular\_double, 949
  - composePermutationsLLL, 939
  - composePermutationsLLM, 939
  - composePermutationsMLM, 939
  - cyclic\_shift\_col\_modular\_double, 939
  - cyclic\_shift\_mathPerm, 939
  - cyclic\_shift\_row\_modular\_double, 939
  - Det\_modular\_double, 947
  - FFLAS\_C\_DIAG, 937
  - FFLAS\_C\_ORDER, 936
  - FFLAS\_C\_SIDE, 937
  - FFLAS\_C\_TRANSPOSE, 936
  - FFLAS\_C\_UPLO, 936
  - FflasColMajor, 936
  - FflasLeft, 937
  - FflasLower, 937
  - FflasNonUnit, 937
  - FflasNoTrans, 936
  - FflasRight, 937
  - FflasRowMajor, 936
  - FflasTrans, 936
  - FflasUnit, 937
  - FflasUpper, 937
  - FFPACK\_C\_CHARPOLY\_TAG, 937
  - FFPACK\_C\_LU\_TAG, 937
  - FFPACK\_C\_MINPOLY\_TAG, 937
  - FFPACK\_COMPILED, 936
  - FfpackArithProg, 937
  - FfpackDanilevski, 937
  - FfpackDense, 938
  - FfpackHybrid, 937
  - FfpackKG, 937
  - FfpackKGF, 938
  - FfpackKGFast, 937
  - FfpackKGFastG, 937
  - FfpackLUK, 937
  - FfpackSingular, 937
  - FfpackSlabRecursive, 937
  - FfpackTileRecursive, 937
  - fgesv\_modular\_double, 941
  - fgesvin\_modular\_double, 940
  - fgetrsv\_modular\_double, 940
  - fgetrsvsin\_modular\_double, 940
  - fttrsv\_modular\_double, 941
  - fttrsvrm\_modular\_double, 941
  - getEchelonForm\_modular\_double, 951
  - getEchelonFormin\_modular\_double, 951
  - getEchelonTransform\_modular\_double, 951
  - getReducedEchelonForm\_modular\_double, 952
  - getReducedEchelonFormin\_modular\_double, 952
  - getReducedEchelonTransform\_modular\_double, 952
  - getTriangular\_modular\_double, 950
  - getTriangularin\_modular\_double, 950
  - Invert2\_modular\_double, 946
  - Invert\_modular\_double, 946
  - Invertin\_modular\_double, 946
  - IsSingular\_modular\_double, 947
  - KrylovElim\_modular\_double, 946
  - LAPACKPerm2MathPerm, 938
  - LeadingSubmatrixRankProfiles, 949
  - LUdivine\_gauss\_modular\_double, 942
  - LUdivine\_modular\_double, 942
  - LUdivine\_small\_modular\_double, 942
  - MathPerm2LAPACKPerm, 938
  - MatrixApplyS\_modular\_double, 938
  - MatrixApplyT\_modular\_double, 938

NullSpaceBasis\_modular\_double, 948  
 PermApplyS\_double, 938  
 PermApplyT\_double, 939  
 PLUQ\_modular\_double, 942  
 PLUQtoEchelonPermutation, 952  
 RandomNullSpaceVector\_modular\_double, 948  
 Rank\_modular\_double, 947  
 RankProfileFromLU, 949  
 ReducedColumnEchelonForm\_modular\_double, 944  
 ReducedColumnEchelonForm\_modular\_float, 944  
 ReducedColumnEchelonForm\_modular\_int32\_t, 945  
 ReducedRowEchelonForm2\_modular\_double, 945  
 ReducedRowEchelonForm\_modular\_double, 944  
 ReducedRowEchelonForm\_modular\_float, 945  
 ReducedRowEchelonForm\_modular\_int32\_t, 945  
 REF\_modular\_double, 946  
 RowEchelonForm\_modular\_double, 943  
 RowEchelonForm\_modular\_float, 943  
 RowEchelonForm\_modular\_int32\_t, 944  
 RowRankProfile\_modular\_double, 949  
 RowRankProfileSubmatrix\_modular\_double, 950  
 RowRankProfileSubmatrixIndices\_modular\_double, 949  
 Solve\_modular\_double, 947  
 solveLB2\_modular\_double, 948  
 solveLB\_modular\_double, 948  
 SpecRankProfile\_modular\_double, 947  
 trinv\_left\_modular\_double, 941  
 FFPACK\_C\_CHARPOLY\_TAG  
   ffpack\_c.h, 937  
 FFPACK\_C\_LU\_TAG  
   ffpack\_c.h, 937  
 FFPACK\_C\_MINPOLY\_TAG  
   ffpack\_c.h, 937  
 ffpack\_charpoly.inl, 855  
   \_\_FFLASFFPACK\_charpoly\_INL, 856  
 ffpack\_charpoly\_danilevski.inl, 856  
   \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL, 856  
 ffpack\_charpoly\_kgfast.inl, 856  
   \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL, 856  
 ffpack\_charpoly\_kgfastgeneralized.inl, 857  
   \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL, 857  
 ffpack\_charpoly\_kglu.inl, 857  
   \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL, 858  
 ffpack\_charpoly\_mp.inl, 858  
   \_\_FFPACK\_charpoly\_mp\_INL, 858  
 FFPACK\_COMPILED  
   ffpack\_c.h, 936  
 ffpack\_det\_mp.inl, 858  
   \_\_FFPACK\_det\_mp\_INL, 859  
 ffpack\_echelonforms.inl, 859  
   \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE, 860  
   \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL, 860  
 ffpack\_fgesv.inl, 860  
   \_\_FFLASFFPACK\_ffpack\_fgesv\_INL, 861  
 ffpack\_fgetrs.inl, 861  
   \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL, 861  
 ffpack\_frobenius.inl, 861  
 ffpack\_fsytrf.inl, 862  
   \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL, 863  
 ffpack\_ftrssyr2k.inl, 863  
   \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL, 863  
 ffpack\_ftrstr.inl, 864  
   \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL, 864  
 ffpack\_ftrtr.inl, 864  
   \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL, 865  
 ENABLE\_ALL\_CHECKINGS, 865  
 ffpack\_inst.C, 953  
   \_\_FFPACK\_INST\_C, 953  
 FFLAS\_COMPILED, 953  
 FFLAS\_ELT, 953, 954  
 FFLAS\_FIELD, 953  
 INST\_OR\_DECL, 953  
 ffpack\_inst.h, 954  
   FFLAS\_COMPILED, 954  
   FFLAS\_ELT, 954, 955  
   FFLAS\_FIELD, 954  
   INST\_OR\_DECL, 954  
 ffpack\_inst\_implement.inl, 955  
 ffpack\_invert.inl, 865  
   \_\_FFLASFFPACK\_ffpack\_invert\_INL, 865  
 ffpack\_krylovelim.inl, 865  
   \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL, 865  
 ffpack\_ludivine.inl, 865  
   \_\_FFLASFFPACK\_ffpack\_ludivine\_INL, 866  
 ffpack\_ludivine\_mp.inl, 866  
   \_\_FFPACK\_ludivine\_mp\_INL, 867  
 ffpack\_minpoly.inl, 867  
   \_\_FFLASFFPACK\_ffpack\_minpoly\_INL, 867  
 ffpack\_permutation.inl, 868  
   \_\_FFLASFFPACK\_ffpack\_permutation\_INL, 870  
   FFLASFFPACK\_PERM\_BKSIZE, 870  
 ffpack\_pluq.inl, 870  
   \_\_FFLASFFPACK\_ffpack\_pluq\_INL, 870  
 CROUT, 870  
 ffpack\_pluq\_mp.inl, 871  
   \_\_FFPACK\_pluq\_mp\_INL, 871  
 ffpack\_ppluq.inl, 871  
   \_\_FFLASFFPACK\_ffpack\_ppluq\_INL, 872  
   \_\_FFLAS\_\_TRSM\_READONLY, 872  
 PBASECASE\_K, 872  
 ffpack\_rankprofiles.inl, 872  
   \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL, 873  
 FfpackArithProg  
   ffpack\_c.h, 937  
 FfpackDanilevski  
   ffpack\_c.h, 937  
 FfpackDense  
   ffpack\_c.h, 938

- FpackHybrid
  - ffpack\_c.h, [937](#)
- FpackKG
  - ffpack\_c.h, [937](#)
- FpackKGF
  - ffpack\_c.h, [938](#)
- FpackKGFast
  - ffpack\_c.h, [937](#)
- FpackKGFastG
  - ffpack\_c.h, [937](#)
- FpackLUK
  - ffpack\_c.h, [937](#)
- FpackSingular
  - ffpack\_c.h, [937](#)
- FpackSlabRecursive
  - ffpack\_c.h, [937](#)
- FpackTileRecursive
  - ffpack\_c.h, [937](#)
- fgemm
  - FFLAS, [86–88](#), [90–95](#), [136](#), [168](#), [169](#)
- fgemm\_3\_modular\_double
  - fflas\_c.h, [890](#)
  - fflas\_lvl3.C, [912](#)
- fgemm\_classical.inl, [772](#)
- fgemm\_classical\_mp.inl, [772](#)
  - \_\_FFPACK\_fgemm\_classical\_INL, [774](#)
- fgemm\_convert
  - FFLAS::Protected, [202](#)
- fgemm\_winograd.inl, [774](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL, [775](#)
  - NEWWINO, [775](#)
- fgemv
  - FFLAS, [95–100](#), [165](#), [176](#)
- fgemv\_2\_modular\_double
  - fflas\_c.h, [889](#)
  - fflas\_lvl2.C, [910](#)
- fgemv\_convert
  - FFLAS::Protected, [207](#)
- fgemv\_kgf
  - FFPACK::Protected, [372](#)
- fger
  - FFLAS, [101–104](#), [165](#)
- fger\_2\_modular\_double
  - fflas\_c.h, [889](#)
  - fflas\_lvl2.C, [910](#)
- fger\_convert
  - FFLAS::Protected, [207](#)
- fgesv
  - FFPACK, [292](#), [293](#), [344](#)
- fgesv\_modular\_double
  - ffpack.C, [919](#)
  - ffpack\_c.h, [941](#)
- fgesvin\_modular\_double
  - ffpack.C, [919](#)
  - ffpack\_c.h, [940](#)
- fgetrs
  - FFPACK, [291](#), [292](#), [343](#)
- fgetrs\_modular\_double
  - ffpack\_c.h, [940](#)
- fgetrsin\_modular\_double
  - ffpack.C, [918](#)
  - ffpack\_c.h, [940](#)
- fgetrsv\_modular\_double
  - ffpack.C, [919](#)
- fidentity
  - FFLAS, [129](#), [158](#)
- fidentity\_2\_modular\_double
  - fflas\_c.h, [886](#)
  - fflas\_lvl2.C, [907](#)
- Field
  - benchmark-fgemm-rns.C, [725](#)
  - benchmark-pluq.C, [737](#)
  - FieldSimd< \_Field >, [414](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [680](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [683](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [685](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [693](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
  - test-compressQ.C, [987](#)
- field
  - associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, [379](#)
  - associatedDelayedField< const Givaro::Modular< T, X > >, [380](#)
  - associatedDelayedField< const Givaro::ModularBalanced< T > >, [380](#)
  - associatedDelayedField< const Givaro::ZRing< T > >, [381](#)
  - associatedDelayedField< Field >, [379](#)
- field-traits.h, [873](#)
- field.doxy, [875](#)
- FieldMax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [467](#)
- FieldMin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [467](#)
- FieldSimd
  - FieldSimd< \_Field >, [415](#)
- FieldSimd< \_Field >, [413](#)
  - add, [415](#)
  - add\_r, [416](#)
  - addin, [416](#)
  - addin\_r, [416](#)
  - alignment, [419](#)
  - axpy, [418](#)
  - axpy\_r, [418](#)
  - axpyin, [418](#)
  - axpyin\_r, [418](#)

- Element, [414](#)
- Field, [414](#)
- FieldSimd, [415](#)
- init, [415](#)
- maxpy, [419](#)
- maxpyin, [419](#)
- mod, [417](#)
- mul, [417](#)
- mul\_r, [417](#), [418](#)
- mulin, [417](#)
- operator=, [415](#)
- scalar\_t, [414](#)
- simd, [414](#)
- sub, [416](#)
- sub\_r, [416](#), [417](#)
- subin, [416](#)
- subin\_r, [417](#)
- vect\_size, [419](#)
- vect\_t, [414](#)
- zero, [417](#)
- FieldTraits< FFPACK::RNSInteger< T > >, [420](#)
  - balanced, [420](#)
  - category, [420](#)
- FieldTraits< FFPACK::RNSIntegerMod< T > >, [420](#)
  - balanced, [421](#)
  - category, [421](#)
- FieldTraits< Field >, [419](#)
  - balanced, [420](#)
  - category, [420](#)
- FieldTraits< Givaro::Modular< Element > >, [421](#)
  - balanced, [421](#)
  - category, [421](#)
- FieldTraits< Givaro::ModularBalanced< Element > >, [421](#)
  - balanced, [422](#)
  - category, [422](#)
- FieldTraits< Givaro::ZRing< double > >, [422](#)
  - balanced, [422](#)
  - category, [422](#)
- FieldTraits< Givaro::ZRing< float > >, [422](#)
  - balanced, [423](#)
  - category, [423](#)
- FieldTraits< Givaro::ZRing< Givaro::Integer > >, [423](#)
  - balanced, [423](#)
  - category, [423](#)
- FieldTraits< Givaro::ZRing< int16\_t > >, [423](#)
  - balanced, [424](#)
  - category, [424](#)
- FieldTraits< Givaro::ZRing< int32\_t > >, [424](#)
  - balanced, [424](#)
  - category, [424](#)
- FieldTraits< Givaro::ZRing< int64\_t > >, [424](#)
  - balanced, [425](#)
  - category, [424](#)
- FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > >, [425](#)
  - balanced, [425](#)
  - category, [425](#)
- FieldTraits< Givaro::ZRing< uint16\_t > >, [425](#)
  - balanced, [426](#)
  - category, [425](#)
- FieldTraits< Givaro::ZRing< uint32\_t > >, [426](#)
  - balanced, [426](#)
  - category, [426](#)
- FieldTraits< Givaro::ZRing< uint64\_t > >, [426](#)
  - balanced, [426](#)
  - category, [426](#)
- fill\_value
  - benchmark-fgemv.C, [729](#)
- findArgument
  - args-parser.h, [969](#)
- finit
  - FFLAS, [106](#), [108](#), [122](#), [129](#), [149](#), [159](#)
- finit\_rns
  - FFLAS, [146](#), [148](#)
- finit\_trans\_rns
  - FFLAS, [147](#)
- first\_component
  - Compose< H1, H2 >, [393](#)
- firstBlockSize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [429](#)
- fiszero
  - FFLAS, [125](#), [128](#), [151](#), [157](#)
- fiszero\_1\_modular\_double
  - fflas\_c.h, [883](#)
  - fflas\_lvl1.C, [903](#)
- fiszero\_2\_modular\_double
  - fflas\_c.h, [886](#)
  - fflas\_lvl2.C, [907](#)
- Fixed, [427](#)
- FixedPreclntTag, [427](#)
- flimits.h, [975](#)
  - in\_range, [976](#)
- FLOAT\_MOD
  - fflas\_simd.h, [806](#)
- Floats
  - benchmark-dgemm.C, [718](#)
- floor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [520](#)
  - Simd256\_impl< true, false, true, 8 >, [587](#)
  - Simd512\_impl< true, false, true, 8 >, [658](#)
- fmadd
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [521](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [525](#)
  - Simd128\_impl< true, true, false, 2 >, [536](#)
  - Simd128\_impl< true, true, false, 4 >, [544](#)
  - Simd128\_impl< true, true, false, 8 >, [553](#)
  - Simd128\_impl< true, true, true, 2 >, [560](#)
  - Simd128\_impl< true, true, true, 4 >, [568](#)
  - Simd128\_impl< true, true, true, 8 >, [576](#)
  - Simd256\_impl< true, false, true, 8 >, [585](#)

- Simd256\_impl< true, true, false, 2 >, [595](#)
- Simd256\_impl< true, true, false, 4 >, [609](#)
- Simd256\_impl< true, true, false, 8 >, [620](#)
- Simd256\_impl< true, true, true, 2 >, [626](#)
- Simd256\_impl< true, true, true, 4 >, [635](#), [640](#)
- Simd256\_impl< true, true, true, 8 >, [648](#)
- Simd512\_impl< true, false, true, 8 >, [656](#)
- Simd512\_impl< true, true, false, 8 >, [666](#)
- Simd512\_impl< true, true, true, 8 >, [673](#)
- fmaddin
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [521](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [525](#)
  - Simd128\_impl< true, true, false, 2 >, [536](#)
  - Simd128\_impl< true, true, false, 4 >, [544](#)
  - Simd128\_impl< true, true, false, 8 >, [553](#)
  - Simd128\_impl< true, true, true, 2 >, [560](#)
  - Simd128\_impl< true, true, true, 4 >, [568](#)
  - Simd128\_impl< true, true, true, 8 >, [576](#)
  - Simd256\_impl< true, false, true, 8 >, [585](#)
  - Simd256\_impl< true, true, false, 2 >, [595](#)
  - Simd256\_impl< true, true, false, 4 >, [609](#), [610](#)
  - Simd256\_impl< true, true, false, 8 >, [620](#)
  - Simd256\_impl< true, true, true, 2 >, [626](#)
  - Simd256\_impl< true, true, true, 4 >, [635](#), [640](#)
  - Simd256\_impl< true, true, true, 8 >, [648](#)
  - Simd512\_impl< true, false, true, 8 >, [657](#)
  - Simd512\_impl< true, true, false, 8 >, [666](#)
  - Simd512\_impl< true, true, true, 8 >, [674](#)
- fmaddx
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [525](#)
  - Simd128\_impl< true, true, false, 2 >, [533](#)
  - Simd128\_impl< true, true, false, 4 >, [541](#)
  - Simd128\_impl< true, true, false, 8 >, [550](#)
  - Simd128\_impl< true, true, true, 2 >, [560](#)
  - Simd128\_impl< true, true, true, 4 >, [568](#)
  - Simd128\_impl< true, true, true, 8 >, [576](#)
  - Simd256\_impl< true, true, false, 2 >, [591](#)
  - Simd256\_impl< true, true, false, 4 >, [602](#), [604](#)
  - Simd256\_impl< true, true, false, 8 >, [616](#)
  - Simd256\_impl< true, true, true, 2 >, [627](#)
  - Simd256\_impl< true, true, true, 4 >, [636](#), [640](#)
  - Simd256\_impl< true, true, true, 8 >, [649](#)
  - Simd512\_impl< true, true, false, 8 >, [662](#)
  - Simd512\_impl< true, true, true, 8 >, [674](#)
- fmaddxin
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [525](#)
  - Simd128\_impl< true, true, false, 2 >, [533](#)
  - Simd128\_impl< true, true, false, 4 >, [541](#)
  - Simd128\_impl< true, true, false, 8 >, [550](#)
  - Simd128\_impl< true, true, true, 2 >, [561](#)
  - Simd128\_impl< true, true, true, 4 >, [568](#)
  - Simd128\_impl< true, true, true, 8 >, [576](#)
  - Simd256\_impl< true, true, false, 2 >, [591](#)
- Simd256\_impl< true, true, false, 4 >, [602](#), [604](#)
- Simd256\_impl< true, true, false, 8 >, [616](#)
- Simd256\_impl< true, true, true, 2 >, [627](#)
- Simd256\_impl< true, true, true, 4 >, [636](#), [641](#)
- Simd256\_impl< true, true, true, 8 >, [649](#)
- Simd512\_impl< true, true, false, 8 >, [663](#)
- Simd512\_impl< true, true, true, 8 >, [674](#)
- fmove
  - FFLAS, [132](#), [162](#)
- fmove\_2\_modular\_double
  - fflas\_c.h, [888](#)
  - fflas\_lvl2.C, [909](#)
- fmsub
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [521](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [525](#)
  - Simd128\_impl< true, true, false, 2 >, [536](#)
  - Simd128\_impl< true, true, false, 4 >, [545](#)
  - Simd128\_impl< true, true, false, 8 >, [554](#)
  - Simd128\_impl< true, true, true, 2 >, [561](#)
  - Simd128\_impl< true, true, true, 4 >, [569](#)
  - Simd128\_impl< true, true, true, 8 >, [577](#)
  - Simd256\_impl< true, false, true, 8 >, [585](#)
  - Simd256\_impl< true, true, false, 2 >, [595](#)
  - Simd256\_impl< true, true, false, 4 >, [610](#)
  - Simd256\_impl< true, true, false, 8 >, [620](#)
  - Simd256\_impl< true, true, true, 2 >, [627](#)
  - Simd256\_impl< true, true, true, 4 >, [636](#), [641](#)
  - Simd256\_impl< true, true, true, 8 >, [649](#)
  - Simd512\_impl< true, false, true, 8 >, [657](#)
  - Simd512\_impl< true, true, false, 8 >, [667](#)
  - Simd512\_impl< true, true, true, 8 >, [674](#)
- fmsubin
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [521](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [525](#)
  - Simd128\_impl< true, true, false, 2 >, [536](#)
  - Simd128\_impl< true, true, false, 4 >, [545](#)
  - Simd128\_impl< true, true, false, 8 >, [554](#)
  - Simd128\_impl< true, true, true, 2 >, [561](#)
  - Simd128\_impl< true, true, true, 4 >, [569](#)
  - Simd128\_impl< true, true, true, 8 >, [577](#)
  - Simd256\_impl< true, false, true, 8 >, [586](#)
  - Simd256\_impl< true, true, false, 2 >, [596](#)
  - Simd256\_impl< true, true, false, 4 >, [610](#), [611](#)
  - Simd256\_impl< true, true, false, 8 >, [620](#)
  - Simd256\_impl< true, true, true, 2 >, [627](#)
  - Simd256\_impl< true, true, true, 4 >, [636](#), [641](#)
  - Simd256\_impl< true, true, true, 8 >, [649](#)
  - Simd512\_impl< true, false, true, 8 >, [657](#)
  - Simd512\_impl< true, true, false, 8 >, [667](#)
  - Simd512\_impl< true, true, true, 8 >, [675](#)
- fmsubx

- ScalFunctions< Element, typename enable\_if<  
  is\_integral< Element >::value >::type >, 525
- Simd128\_impl< true, true, false, 2 >, 533
- Simd128\_impl< true, true, false, 4 >, 542
- Simd128\_impl< true, true, false, 8 >, 551
- Simd128\_impl< true, true, true, 2 >, 561
- Simd128\_impl< true, true, true, 4 >, 569
- Simd128\_impl< true, true, true, 8 >, 577
- Simd256\_impl< true, true, false, 2 >, 592
- Simd256\_impl< true, true, false, 4 >, 602, 605
- Simd256\_impl< true, true, false, 8 >, 617
- Simd256\_impl< true, true, true, 2 >, 628
- Simd256\_impl< true, true, true, 4 >, 637, 641
- Simd256\_impl< true, true, true, 8 >, 650
- Simd512\_impl< true, true, false, 8 >, 663
- Simd512\_impl< true, true, true, 8 >, 675
- fmsubxin
  - ScalFunctions< Element, typename enable\_if<  
  is\_integral< Element >::value >::type >, 525
  - Simd128\_impl< true, true, false, 2 >, 533
  - Simd128\_impl< true, true, false, 4 >, 542
  - Simd128\_impl< true, true, false, 8 >, 551, 554
  - Simd128\_impl< true, true, true, 2 >, 562
  - Simd128\_impl< true, true, true, 4 >, 569
  - Simd128\_impl< true, true, true, 8 >, 577
  - Simd256\_impl< true, true, false, 2 >, 592
  - Simd256\_impl< true, true, false, 4 >, 602, 605
  - Simd256\_impl< true, true, false, 8 >, 617
  - Simd256\_impl< true, true, true, 2 >, 628
  - Simd256\_impl< true, true, true, 4 >, 637, 642
  - Simd256\_impl< true, true, true, 8 >, 650
  - Simd512\_impl< true, true, false, 8 >, 663
  - Simd512\_impl< true, true, true, 8 >, 675
- fneg
  - FFLAS, 124, 131, 150, 160
- fneg\_1\_modular\_double
  - fflas\_c.h, 883
  - fflas\_lvl1.C, 902
- fneg\_2\_modular\_double
  - fflas\_c.h, 887
  - fflas\_lvl2.C, 908
- fnegin
  - FFLAS, 123, 130, 150, 160
- fnegin\_1\_modular\_double
  - fflas\_c.h, 883
  - fflas\_lvl1.C, 902
- fnegin\_2\_modular\_double
  - fflas\_c.h, 887
  - fflas\_lvl2.C, 908
- fnmadd
  - ScalFunctions< Element, typename enable\_if<  
  is\_floating\_point< Element >::value >::type  
  >, 521
  - ScalFunctions< Element, typename enable\_if<  
  is\_integral< Element >::value >::type >, 526
  - Simd128\_impl< true, true, false, 2 >, 536
  - Simd128\_impl< true, true, false, 4 >, 545
  - Simd128\_impl< true, true, false, 8 >, 554
  - Simd128\_impl< true, true, true, 2 >, 561
  - Simd128\_impl< true, true, true, 4 >, 568
  - Simd128\_impl< true, true, true, 8 >, 576
  - Simd256\_impl< true, true, false, 2 >, 592
  - Simd256\_impl< true, true, false, 4 >, 602, 604
  - Simd256\_impl< true, true, false, 8 >, 616
  - Simd256\_impl< true, true, true, 2 >, 627
  - Simd256\_impl< true, true, true, 4 >, 636, 641
  - Simd256\_impl< true, true, true, 8 >, 649
  - Simd512\_impl< true, true, false, 8 >, 663
  - Simd512\_impl< true, true, true, 8 >, 674
- Simd128\_impl< true, true, true, 2 >, 561
- Simd128\_impl< true, true, true, 4 >, 568
- Simd128\_impl< true, true, true, 8 >, 576
- Simd256\_impl< true, false, true, 8 >, 585
- Simd256\_impl< true, true, false, 2 >, 595
- Simd256\_impl< true, true, false, 4 >, 610
- Simd256\_impl< true, true, false, 8 >, 620
- Simd256\_impl< true, true, true, 2 >, 627
- Simd256\_impl< true, true, true, 4 >, 636, 641
- Simd256\_impl< true, true, true, 8 >, 649
- Simd512\_impl< true, false, true, 8 >, 657
- Simd512\_impl< true, true, false, 8 >, 666
- Simd512\_impl< true, true, true, 8 >, 674
- fnmaddin
  - ScalFunctions< Element, typename enable\_if<  
  is\_floating\_point< Element >::value >::type  
  >, 521
  - ScalFunctions< Element, typename enable\_if<  
  is\_integral< Element >::value >::type >, 526
  - Simd128\_impl< true, true, false, 2 >, 536
  - Simd128\_impl< true, true, false, 4 >, 545
  - Simd128\_impl< true, true, false, 8 >, 554
  - Simd128\_impl< true, true, true, 2 >, 561
  - Simd128\_impl< true, true, true, 4 >, 568
  - Simd128\_impl< true, true, true, 8 >, 576
  - Simd256\_impl< true, false, true, 8 >, 585
  - Simd256\_impl< true, true, false, 2 >, 595
  - Simd256\_impl< true, true, false, 4 >, 610
  - Simd256\_impl< true, true, false, 8 >, 620
  - Simd256\_impl< true, true, true, 2 >, 627
  - Simd256\_impl< true, true, true, 4 >, 636, 641
  - Simd256\_impl< true, true, true, 8 >, 649
  - Simd512\_impl< true, false, true, 8 >, 657
  - Simd512\_impl< true, true, false, 8 >, 666
  - Simd512\_impl< true, true, true, 8 >, 674
- fnmaddx
  - ScalFunctions< Element, typename enable\_if<  
  is\_integral< Element >::value >::type >, 526
  - Simd128\_impl< true, true, false, 2 >, 533
  - Simd128\_impl< true, true, false, 4 >, 541
  - Simd128\_impl< true, true, false, 8 >, 550
  - Simd128\_impl< true, true, true, 2 >, 561
  - Simd128\_impl< true, true, true, 4 >, 569
  - Simd128\_impl< true, true, true, 8 >, 576
  - Simd256\_impl< true, true, false, 2 >, 592
  - Simd256\_impl< true, true, false, 4 >, 602, 604
  - Simd256\_impl< true, true, false, 8 >, 616
  - Simd256\_impl< true, true, true, 2 >, 627
  - Simd256\_impl< true, true, true, 4 >, 636, 641
  - Simd256\_impl< true, true, true, 8 >, 649
  - Simd512\_impl< true, true, false, 8 >, 663
  - Simd512\_impl< true, true, true, 8 >, 674
- fnmaddxin
  - ScalFunctions< Element, typename enable\_if<  
  is\_integral< Element >::value >::type >, 526
  - Simd128\_impl< true, true, false, 2 >, 533
  - Simd128\_impl< true, true, false, 4 >, 542
  - Simd128\_impl< true, true, false, 8 >, 551



- Simd128\_impl< true, true, true, 2 >, [561](#)
- Simd128\_impl< true, true, true, 4 >, [569](#)
- Simd128\_impl< true, true, true, 8 >, [576](#)
- Simd256\_impl< true, true, false, 2 >, [592](#)
- Simd256\_impl< true, true, false, 4 >, [602](#), [605](#)
- Simd256\_impl< true, true, false, 8 >, [617](#)
- Simd256\_impl< true, true, true, 2 >, [627](#)
- Simd256\_impl< true, true, true, 4 >, [636](#), [641](#)
- Simd256\_impl< true, true, true, 8 >, [649](#)
- Simd512\_impl< true, true, false, 8 >, [663](#)
- Simd512\_impl< true, true, true, 8 >, [674](#)
- FOR1D
  - parallel.h, [963](#)
- FOR2D
  - parallel.h, [963](#)
- FORBLOCK1D
  - parallel.h, [962](#)
- FORBLOCK2D
  - parallel.h, [963](#)
- ForceCheck\_charpoly
  - FFPACK, [288](#)
- ForceCheck\_Det
  - FFPACK, [288](#)
- ForceCheck\_fgemm
  - FFLAS, [70](#)
- ForceCheck\_ftsrsm
  - FFLAS, [70](#)
- ForceCheck\_invert
  - FFPACK, [288](#)
- ForceCheck\_PLUQ
  - FFPACK, [288](#)
- ForStrategy1D
  - ForStrategy1D< blocksize\_t, Cut, Param >, [427](#)
- ForStrategy1D< blocksize\_t, Cut, Param >, [427](#)
  - begin, [428](#)
  - blockindex, [428](#)
  - build, [428](#)
  - changeBS, [429](#)
  - current, [429](#)
  - end, [428](#)
  - firstBlockSize, [429](#)
  - ForStrategy1D, [427](#)
  - ibeg, [429](#)
  - iend, [429](#)
  - initialize, [428](#)
  - isTerminated, [428](#)
  - lastBlockSize, [429](#)
  - numBlock, [429](#)
  - numblocks, [428](#)
  - operator++, [428](#)
- ForStrategy2D
  - ForStrategy2D< blocksize\_t, Cut, Param >, [430](#)
- ForStrategy2D< blocksize\_t, Cut, Param >, [429](#)
  - \_ibeg, [432](#)
  - \_iend, [432](#)
  - \_jbeg, [432](#)
  - \_jend, [432](#)
  - blockindex, [431](#)
  - BLOCKS, [433](#)
  - changeCBS, [433](#)
  - changeRBS, [432](#)
  - colblockindex, [431](#)
  - colBlockSize, [432](#)
  - colnumblocks, [431](#)
  - current, [432](#)
  - ForStrategy2D, [430](#)
  - ibegin, [430](#)
  - iend, [431](#)
  - initialize, [430](#)
  - isTerminated, [430](#)
  - jbegin, [430](#)
  - jend, [431](#)
  - lastCBS, [432](#)
  - lastRBS, [432](#)
  - numColBlock, [433](#)
  - numRowBlock, [433](#)
  - operator<<, [431](#)
  - operator++, [431](#)
  - rowblockindex, [431](#)
  - rowBlockSize, [432](#)
  - rownumblocks, [431](#)
- frand
  - FFLAS, [125](#), [128](#)
- freduce
  - FFLAS, [105–108](#), [148](#), [149](#), [158](#), [159](#)
  - FFLAS::details, [191](#), [192](#)
- freduce\_1\_modular\_double
  - fflas\_c.h, [883](#)
  - fflas\_lvl1.C, [902](#)
- freduce\_2\_modular\_double
  - fflas\_c.h, [887](#)
  - fflas\_lvl2.C, [907](#)
- freduce\_constoverride
  - FFLAS, [106](#), [108](#)
- freducein\_1\_modular\_double
  - fflas\_c.h, [883](#)
  - fflas\_lvl1.C, [902](#)
- freducein\_2\_modular\_double
  - fflas\_c.h, [887](#)
  - fflas\_lvl2.C, [907](#)
- freivalds
  - FFLAS, [109](#)
- fscal
  - FFLAS, [110–114](#), [153](#), [161](#)
  - FFLAS::details, [192](#), [193](#)
- fscal\_1\_modular\_double
  - fflas\_c.h, [884](#)
  - fflas\_lvl1.C, [903](#)
- fscal\_2\_modular\_double
  - fflas\_c.h, [887](#)
  - fflas\_lvl2.C, [908](#)
- fscalin
  - FFLAS, [109](#), [111–113](#), [152](#), [160](#)
  - FFLAS::details, [192](#)
- fscalin\_1\_modular\_double
  - fflas\_c.h, [884](#)

- fflas\_lvl1.C, 903
- fscalin\_2\_modular\_double
  - fflas\_c.h, 887
  - fflas\_lvl2.C, 908
- fspmm
  - FFLAS, 137
  - FFLAS::sparse\_details, 217–219
  - FFLAS::sparse\_details\_impl, 233, 240, 241, 246, 247, 251, 260
- fspmm\_dispatch
  - FFLAS::sparse\_details, 216, 217
- fspmm\_mone
  - FFLAS::sparse\_details\_impl, 234, 242, 252
- fspmm\_mone\_simd\_aligned
  - FFLAS::sparse\_details\_impl, 235, 242, 253
- fspmm\_mone\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, 235, 243, 253
- fspmm\_one
  - FFLAS::sparse\_details\_impl, 234, 241, 252
- fspmm\_one\_simd\_aligned
  - FFLAS::sparse\_details\_impl, 235, 242, 253
- fspmm\_one\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, 235, 242, 253
- fspmm\_simd\_aligned
  - FFLAS::sparse\_details\_impl, 234, 241
- fspmm\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, 234, 241
- fspmv
  - FFLAS, 137, 144
  - FFLAS::sparse\_details, 214–216, 224
  - FFLAS::sparse\_details\_impl, 236, 243, 247, 254, 256, 257, 261, 263
- fspmv\_dispatch
  - FFLAS::sparse\_details, 214
- fspmv\_mone
  - FFLAS::sparse\_details\_impl, 236, 237, 244, 254, 255, 258, 264, 265
- fspmv\_mone\_simd
  - FFLAS::sparse\_details\_impl, 258, 264
- fspmv\_one
  - FFLAS::sparse\_details\_impl, 236, 237, 243, 244, 254, 255, 257, 258, 264
- fspmv\_one\_simd
  - FFLAS::sparse\_details\_impl, 258, 264
- fspmv\_simd
  - FFLAS::sparse\_details\_impl, 257, 263
- fsquare
  - FFLAS, 89, 90, 170
- fsquare\_3\_modular\_double
  - fflas\_c.h, 891
  - fflas\_lvl3.C, 912
- fsquareCommon
  - FFLAS::Protected, 204
- fsub
  - FFLAS, 76, 79, 155, 163
- fsub\_1\_modular\_double
  - fflas\_c.h, 885
  - fflas\_lvl1.C, 904
- fsub\_2\_modular\_double
  - fflas\_c.h, 888
  - fflas\_lvl2.C, 909
- fsubin
  - FFLAS, 76, 79, 164
- fsubin\_1\_modular\_double
  - fflas\_c.h, 885
  - fflas\_lvl1.C, 905
- fsubin\_2\_modular\_double
  - fflas\_c.h, 889
  - fflas\_lvl2.C, 909
- fswap
  - FFLAS, 127, 155
- fswap\_1\_modular\_double
  - fflas\_c.h, 885
  - fflas\_lvl1.C, 904
- fsyr2k
  - FFLAS, 114
- fsyrk
  - FFLAS, 115–117
- fsyrk.C, 712
  - CUBE, 712
  - GFOPS, 712
  - main, 712
  - TTimer, 712
- fsytrf
  - FFPACK, 296, 297
- fsytrf.C, 713
  - CUBE, 713
  - GFOPS, 713
  - main, 713
  - TTimer, 713
- fsytrf\_BC\_Crout
  - FFPACK, 330
- fsytrf\_BC\_RL
  - FFPACK, 330
- fsytrf\_LOW\_RPM\_BC\_Crout
  - FFPACK, 331
- fsytrf\_nonunit
  - FFPACK, 297, 332
- fsytrf\_RPM
  - FFPACK, 332
- fsytrf\_UP\_RPM
  - FFPACK, 331
- fsytrf\_UP\_RPM\_BC\_Crout
  - FFPACK, 331
- fsytrf\_UP\_RPM\_BC\_RL
  - FFPACK, 331
- ftrmm
  - FFLAS, 118, 119, 167
- ftrmm\_3\_modular\_double
  - fflas\_c.h, 890
  - fflas\_lvl3.C, 911
- ftrmmLeftLowerNoTransNonUnit< Element >, 433
- ftrmmLeftLowerNoTransUnit< Element >, 433
- ftrmmLeftLowerTransNonUnit< Element >, 433
- ftrmmLeftLowerTransUnit< Element >, 433
- ftrmmLeftUpperNoTransNonUnit< Element >, 434



- ftmmLeftUpperNoTransUnit< Element >, [434](#)
- ftmmLeftUpperTransNonUnit< Element >, [434](#)
- ftmmLeftUpperTransUnit< Element >, [434](#)
- ftmmRightLowerNoTransNonUnit< Element >, [434](#)
- ftmmRightLowerNoTransUnit< Element >, [434](#)
- ftmmRightLowerTransNonUnit< Element >, [434](#)
- ftmmRightLowerTransUnit< Element >, [434](#)
- ftmmRightUpperNoTransNonUnit< Element >, [435](#)
- ftmmRightUpperNoTransUnit< Element >, [435](#)
- ftmmRightUpperTransNonUnit< Element >, [435](#)
- ftmmRightUpperTransUnit< Element >, [435](#)
- ftmv
  - FFLAS, [133](#)
- ftsm
  - FFLAS, [119–121](#), [133](#), [136](#), [137](#), [167](#)
- ftsm\_3\_modular\_double
  - fflas\_c.h, [890](#)
  - fflas\_lvl3.C, [911](#)
- ftsmLeftLowerNoTransNonUnit< Element >, [435](#)
- ftsmLeftLowerNoTransUnit< Element >, [435](#)
- ftsmLeftLowerTransNonUnit< Element >, [435](#)
- ftsmLeftLowerTransUnit< Element >, [435](#)
- ftsmLeftUpperNoTransNonUnit< Element >, [436](#)
- ftsmLeftUpperNoTransUnit< Element >, [436](#)
- ftsmLeftUpperTransNonUnit< Element >, [436](#)
- ftsmLeftUpperTransUnit< Element >, [436](#)
- ftsmRightLowerNoTransNonUnit< Element >, [436](#)
- ftsmRightLowerNoTransUnit< Element >, [436](#)
- ftsmRightLowerTransNonUnit< Element >, [437](#)
- ftsmRightLowerTransUnit< Element >, [437](#)
- ftsmRightUpperNoTransNonUnit< Element >, [437](#)
- ftsmRightUpperNoTransUnit< Element >, [437](#)
- ftsmRightUpperTransNonUnit< Element >, [437](#)
- ftsmRightUpperTransUnit< Element >, [437](#)
- ftssyr2k
  - FFPACK, [296](#)
- ftstr
  - FFPACK, [295](#)
- ftsv
  - FFLAS, [121](#), [166](#)
- ftsv\_2\_modular\_double
  - fflas\_c.h, [890](#)
  - fflas\_lvl2.C, [910](#)
- fttri
  - FFPACK, [294](#), [344](#)
- fttri.C, [713](#)
  - CUBE, [714](#)
  - GFOPS, [714](#)
  - main, [714](#)
  - TTimer, [714](#)
- fttri\_modular\_double
  - ffpack.C, [919](#)
  - ffpack\_c.h, [941](#)
- fttrm
  - FFPACK, [295](#), [345](#)
- fttrm\_modular\_double
  - ffpack.C, [920](#)
  - ffpack\_c.h, [941](#)
- fzero
  - FFLAS, [124](#), [127](#), [151](#), [157](#)
- fzero\_1\_modular\_double
  - fflas\_c.h, [883](#)
  - fflas\_lvl1.C, [903](#)
- fzero\_2\_modular\_double
  - fflas\_c.h, [886](#)
  - fflas\_lvl2.C, [906](#)
- gather
  - Simd128\_impl< true, true, false, 2 >, [531](#), [534](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#), [542](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#), [551](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [573](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [590](#), [593](#)
  - Simd256\_impl< true, true, false, 4 >, [600](#), [603](#), [606](#)
  - Simd256\_impl< true, true, false, 8 >, [615](#), [617](#)
  - Simd256\_impl< true, true, true, 2 >, [624](#)
  - Simd256\_impl< true, true, true, 4 >, [633](#), [638](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [655](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#), [664](#)
  - Simd512\_impl< true, true, true, 8 >, [671](#)
- GaussJordan
  - FFPACK::Protected, [371](#)
- GCC\_VERSION
  - fflas-ffpack-config.h, [761](#)
- gebp
  - FFLAS::details, [195](#)
- genData
  - benchmark-fgemv.C, [729](#)
- generate\_random\_vector
  - test-simd.C, [1031](#)
- GenericTag, [437](#), [438](#)
- get
  - Simd128\_impl< true, true, false, 8 >, [552](#)
  - Simd128\_impl< true, true, true, 8 >, [574](#)
  - Simd256\_impl< true, true, false, 8 >, [618](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
- getDataType
  - FFLAS, [143](#), [144](#)
- getEchelonForm
  - FFPACK, [322](#), [323](#)
- getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [353](#)
- getEchelonForm\_modular\_double
  - ffpack.C, [931](#)
  - ffpack\_c.h, [951](#)
- getEchelonFormin\_modular\_double
  - ffpack.C, [931](#)
  - ffpack\_c.h, [951](#)
- getEchelonTransform
  - FFPACK, [324](#)
- getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >
  - >

- FFPACK, [353](#)
- getEchelonTransform\_modular\_double
  - ffpack.C, [932](#)
  - ffpack\_c.h, [951](#)
- getListArgs
  - args-parser.h, [969](#)
- getReducedEchelonForm
  - FFPACK, [324](#), [325](#)
- getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT  
> >
  - FFPACK, [354](#)
- getReducedEchelonForm\_modular\_double
  - ffpack.C, [932](#)
  - ffpack\_c.h, [952](#)
- getReducedEchelonFormin\_modular\_double
  - ffpack.C, [932](#)
  - ffpack\_c.h, [952](#)
- getReducedEchelonTransform
  - FFPACK, [326](#)
- getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [354](#)
- getReducedEchelonTransform\_modular\_double
  - ffpack.C, [932](#)
  - ffpack\_c.h, [952](#)
- getSeed
  - FFLAS, [181](#)
- getStat
  - FFLAS, [146](#)
- getTLBSize
  - FFLAS, [180](#)
- getTriangular
  - FFPACK, [321](#), [322](#)
- getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [352](#), [353](#)
- getTriangular\_modular\_double
  - ffpack.C, [931](#)
  - ffpack\_c.h, [950](#)
- getTriangularin\_modular\_double
  - ffpack.C, [931](#)
  - ffpack\_c.h, [950](#)
- getTridiagonal
  - FFPACK, [332](#)
- gf2ModularBalanced
  - regression-check.C, [984](#)
- GFOPS
  - arithprog.C, [711](#)
  - autotune/charpoly.C, [739](#)
  - autotune/pluq.C, [741](#)
  - fsyrk.C, [712](#)
  - fsytrf.C, [713](#)
  - fttri.C, [714](#)
  - winograd.C, [715](#)
- Givaro, [376](#)
- gmp.C, [980](#)
  - main, [980](#)
- GRAIN
  - benchmark-fgemm-rns.C, [725](#)
- Grain, [438](#)
- greater
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [522](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [527](#)
  - Simd128\_impl< true, true, false, 2 >, [532](#)
  - Simd128\_impl< true, true, false, 4 >, [541](#)
  - Simd128\_impl< true, true, false, 8 >, [550](#)
  - Simd128\_impl< true, true, true, 2 >, [562](#)
  - Simd128\_impl< true, true, true, 4 >, [569](#)
  - Simd128\_impl< true, true, true, 8 >, [577](#)
  - Simd256\_impl< true, false, true, 8 >, [586](#)
  - Simd256\_impl< true, true, false, 2 >, [591](#)
  - Simd256\_impl< true, true, false, 4 >, [601](#), [604](#)
  - Simd256\_impl< true, true, false, 8 >, [616](#)
  - Simd256\_impl< true, true, true, 2 >, [628](#)
  - Simd256\_impl< true, true, true, 4 >, [637](#), [642](#)
  - Simd256\_impl< true, true, true, 8 >, [650](#)
  - Simd512\_impl< true, false, true, 8 >, [657](#)
  - Simd512\_impl< true, true, false, 8 >, [662](#)
  - Simd512\_impl< true, true, true, 8 >, [675](#)
- greater\_eq
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [522](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [527](#)
  - Simd128\_impl< true, true, false, 2 >, [532](#)
  - Simd128\_impl< true, true, false, 4 >, [541](#)
  - Simd128\_impl< true, true, false, 8 >, [550](#)
  - Simd128\_impl< true, true, true, 2 >, [562](#)
  - Simd128\_impl< true, true, true, 4 >, [570](#)
  - Simd128\_impl< true, true, true, 8 >, [577](#)
  - Simd256\_impl< true, false, true, 8 >, [586](#)
  - Simd256\_impl< true, true, false, 2 >, [591](#)
  - Simd256\_impl< true, true, false, 4 >, [601](#), [604](#)
  - Simd256\_impl< true, true, false, 8 >, [616](#)
  - Simd256\_impl< true, true, true, 2 >, [628](#)
  - Simd256\_impl< true, true, true, 4 >, [637](#), [642](#)
  - Simd256\_impl< true, true, true, 8 >, [650](#)
  - Simd512\_impl< true, false, true, 8 >, [658](#)
  - Simd512\_impl< true, true, false, 8 >, [662](#)
  - Simd512\_impl< true, true, true, 8 >, [675](#)
- hadd
  - Simd256\_impl< true, false, true, 8 >, [587](#)
  - Simd512\_impl< true, false, true, 8 >, [658](#)
- hadd\_to\_scal
  - Simd128\_impl< true, true, false, 2 >, [533](#)
  - Simd128\_impl< true, true, false, 4 >, [542](#)
  - Simd128\_impl< true, true, false, 8 >, [551](#)
  - Simd128\_impl< true, true, true, 2 >, [562](#)
  - Simd128\_impl< true, true, true, 4 >, [570](#)
  - Simd128\_impl< true, true, true, 8 >, [577](#)
  - Simd256\_impl< true, false, true, 8 >, [587](#)
  - Simd256\_impl< true, true, false, 2 >, [592](#)
  - Simd256\_impl< true, true, false, 4 >, [602](#), [605](#)

- Simd256\_impl< true, true, false, 8 >, [617](#)
- Simd256\_impl< true, true, true, 2 >, [628](#)
- Simd256\_impl< true, true, true, 4 >, [637](#), [642](#)
- Simd256\_impl< true, true, true, 8 >, [650](#)
- Simd512\_impl< true, false, true, 8 >, [658](#)
- Simd512\_impl< true, true, false, 8 >, [663](#)
- Simd512\_impl< true, true, true, 8 >, [675](#)
- half\_t
  - Simd256\_impl< true, true, false, 2 >, [590](#)
  - Simd256\_impl< true, true, false, 4 >, [600](#)
  - Simd256\_impl< true, true, false, 8 >, [615](#)
  - Simd256\_impl< true, true, true, 2 >, [623](#)
  - Simd256\_impl< true, true, true, 4 >, [632](#)
  - Simd256\_impl< true, true, true, 8 >, [645](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#)
  - Simd512\_impl< true, true, true, 8 >, [670](#)
- has\_equal
  - FFLAS, [71](#)
- has\_minus
  - FFLAS, [71](#)
- has\_minus\_eq
  - FFLAS, [71](#)
- has\_minus\_eq\_impl< C >, [438](#)
  - value, [438](#)
- has\_minus\_impl< C >, [438](#)
  - value, [438](#)
- has\_mul
  - FFLAS, [71](#)
- has\_mul\_eq
  - FFLAS, [72](#)
- has\_mul\_eq\_impl< C >, [438](#)
  - value, [439](#)
- has\_mul\_impl< C >, [439](#)
  - value, [439](#)
- has\_operation< T >, [439](#)
  - value, [439](#)
- has\_plus
  - FFLAS, [71](#)
- has\_plus\_eq
  - FFLAS, [71](#)
- has\_plus\_eq\_impl< C >, [439](#)
  - value, [440](#)
- has\_plus\_impl< C >, [440](#)
  - value, [440](#)
- hasAVX512ER
  - instrset.h, [982](#)
  - instrset\_detect.cpp, [983](#)
- hasF16C
  - instrset\_detect.cpp, [983](#)
- hasFMA3
  - instrset.h, [982](#)
  - instrset\_detect.cpp, [983](#)
- hasFMA4
  - instrset.h, [982](#)
  - instrset\_detect.cpp, [983](#)
- hasXOP
  - instrset.h, [982](#)
  - instrset\_detect.cpp, [983](#)
- HAVE\_BLAS
  - config.h, [755](#)
- HAVE\_CBLAS
  - config.h, [755](#)
- HAVE\_CXX11
  - config.h, [755](#)
- HAVE\_DLFCN\_H
  - config.h, [755](#)
- HAVE\_FLOAT\_H
  - config.h, [755](#)
- HAVE\_INTTYPES\_H
  - config.h, [755](#)
- HAVE\_LAPACK
  - config.h, [755](#)
- HAVE\_LIMITS\_H
  - config.h, [755](#)
- HAVE\_LITTLE\_ENDIAN
  - config.h, [755](#)
- HAVE\_PTHREAD\_H
  - config.h, [755](#)
- HAVE\_STDDEF\_H
  - config.h, [755](#)
- HAVE\_STDINT\_H
  - config.h, [755](#)
- HAVE\_STDIO\_H
  - config.h, [755](#)
- HAVE\_STDLIB\_H
  - config.h, [756](#)
- HAVE\_STRING\_H
  - config.h, [756](#)
- HAVE\_STRINGS\_H
  - config.h, [756](#)
- HAVE\_SYS\_STAT\_H
  - config.h, [756](#)
- HAVE\_SYS\_TIME\_H
  - config.h, [756](#)
- HAVE\_SYS\_TYPES\_H
  - config.h, [756](#)
- HAVE\_UNISTD\_H
  - config.h, [756](#)
- HelperFlag, [440](#)
  - aut, [440](#)
  - coo, [440](#)
  - csr, [440](#)
  - ell, [440](#)
  - none, [440](#)
  - pm1, [441](#)
- HelperMod
  - HelperMod< Field, ElementCategories::MachineIntTag >, [441](#)
  - HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, [442](#)
  - HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, [443](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [443](#)
  - HelperMod< Field, ElementCategories::MachineIntTag >, [441](#)

- HelperMod, 441
- invp, 441
- max, 442
- min, 441
- p, 441
- pow50rem, 442
- HelperMod< Field, ElementTraits >, 441
- HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecisionFlag >, 442
  - HelperMod, 442
  - p, 442
- HelperMod< Field, FFLAS::ElementCategories::FixedPrecisionFlag >, 442
  - HelperMod, 443
  - p, 443
- HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, 443
  - HelperMod, 443
  - invp, 444
  - max, 444
  - min, 444
  - p, 444
- helpString
  - Argument, 378
- HYB\_ZO
  - FFLAS, 74
- hyb\_zo.h, 837
- hyb\_zo\_pspmm.inl, 837
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL, 838
- hyb\_zo\_pspmv.inl, 838
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL, 838
- hyb\_zo\_spm.inl, 838
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spm\_INL, 839
- hyb\_zo\_spmv.inl, 839
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL, 839
- hyb\_zo\_utils.inl, 839
  - \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL, 840
- Hybrid, 444
- Hybrid\_KGF\_LUK\_MinPoly
  - FFPACK::Protected, 374
- ibeg
  - ForStrategy1D< blocksize\_t, Cut, Param >, 429
- ibegin
  - ForStrategy2D< blocksize\_t, Cut, Param >, 430
- idamax\_
  - config-blas.h, 751
- iend
  - ForStrategy1D< blocksize\_t, Cut, Param >, 429
  - ForStrategy2D< blocksize\_t, Cut, Param >, 431
- igebb11
  - FFLAS::details, 194
- igebb14
  - FFLAS::details, 193
- igebb21
  - FFLAS::details, 194
- igebb24
  - FFLAS::details, 193
- igebb41
  - FFLAS::details, 194
- igebb44
  - FFLAS::details, 193
- igebp
  - FFLAS::details, 195
- igemm
  - FFLAS::Protected, 209
  - igemm.doxy, 793
  - igemm.h, 793
  - igemm.inl, 794
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL, 794
- igemm\_
  - FFLAS, 122
- igemm\_colmajor
  - FFLAS::Protected, 209
- igemm\_kernels.h, 794
- igemm\_kernels.inl, 795
  - \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL, 796
- igemm\_tools.h, 796
- igemm\_tools.inl, 796
  - \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL, 796
- inl\_range
  - flimits.h, 976
- index\_t
  - fflas\_sparse.h, 817
  - parallel.h, 961
- InfNorm
  - FFLAS, 75
- Info, 444, 445
  - begin, 445, 446
  - Info, 444–446
  - operator=, 445, 446
  - perm, 445, 446
  - size, 445, 446
- init
  - FieldSimd< \_Field >, 415
  - rns\_double, 493, 494
  - rns\_double\_extended, 504, 505
  - RNSInteger< RNS >, 509
  - RNSIntegerMod< RNS >, 513, 514
- init\_transpose
  - rns\_double, 494
- init\_y
  - FFLAS::sparse\_details, 214
- initA
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 466
- initB
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 466
- initC

- MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
Trait >, 466
- initialize
  - ForStrategy1D< blocksize\_t, Cut, Param >, 428
  - ForStrategy2D< blocksize\_t, Cut, Param >, 430
- initOut
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
Trait >, 466
- INLINE
  - fflas\_simd.h, 806
- INST\_OR\_DECL
  - fflas\_L1\_inst.C, 891
  - fflas\_L1\_inst.h, 892
  - fflas\_L2\_inst.C, 895
  - fflas\_L2\_inst.h, 896
  - fflas\_L3\_inst.C, 898
  - fflas\_L3\_inst.h, 900
  - ffpack\_inst.C, 953
  - ffpack\_inst.h, 954
- INSTRSET
  - instrset.h, 981
- instrset.h, 980
  - const\_int, 981
  - const\_uint, 981
  - hasAVX512ER, 982
  - hasFMA3, 982
  - hasFMA4, 982
  - hasXOP, 982
  - INSTRSET, 981
  - instrset\_detect, 982
  - INSTRSET\_H, 981
  - int16\_t, 981
  - int32\_t, 982
  - int64\_t, 982
  - int8\_t, 981
  - intptr\_t, 982
  - uint16\_t, 981
  - uint32\_t, 982
  - uint64\_t, 982
  - uint8\_t, 981
- instrset\_detect
  - instrset.h, 982
  - instrset\_detect.cpp, 983
- instrset\_detect.cpp, 982
  - hasAVX512ER, 983
  - hasF16C, 983
  - hasFMA3, 983
  - hasFMA4, 983
  - hasXOP, 983
  - instrset\_detect, 983
- INSTRSET\_H
  - instrset.h, 981
- int16\_t
  - instrset.h, 981
- int32\_t
  - instrset.h, 982
- int64\_t
  - instrset.h, 982
- int8\_t
  - instrset.h, 981
- integer
  - rns\_double, 492
  - rns\_double\_extended, 503
  - RNSInteger< RNS >, 507
  - RNSIntegerMod< RNS >, 512
  - test-simd.C, 1030
- Interfaces, 42
- interfaces.doxy, 879
- intptr\_t
  - instrset.h, 982
- inv
  - RNSIntegerMod< RNS >, 515
- Invert
  - FFPACK, 305, 306, 347
- Invert2
  - FFPACK, 306, 347
- Invert2\_modular\_double
  - ffpack.C, 927
  - ffpack\_c.h, 946
- Invert\_modular\_double
  - ffpack.C, 926
  - ffpack\_c.h, 946
- Invertin\_modular\_double
  - ffpack.C, 926
  - ffpack\_c.h, 946
- invp
  - HelperMod< Field, ElementCategories::MachineIntTag  
>, 441
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag  
>, 444
- is\_simd< T >, 446
  - type, 447
  - value, 447
- isMOne
  - RNSInteger< RNS >, 508
  - RNSIntegerMod< RNS >, 512
- isOdd
  - FFPACK, 355
- isOne
  - RNSInteger< RNS >, 508
  - RNSIntegerMod< RNS >, 512
- IsSingular
  - FFPACK, 311, 349
- IsSingular\_modular\_double
  - ffpack.C, 927
  - ffpack\_c.h, 947
- isSparseMatrix< Field, M >, 447
- isSparseMatrix< Field, Sparse< Field, SparseMa-  
trix\_t::COO > >, 447
- isSparseMatrix< Field, Sparse< Field, SparseMa-  
trix\_t::COO\_ZO > >, 447
- isSparseMatrix< Field, Sparse< Field, SparseMa-  
trix\_t::CSR > >, 448
- isSparseMatrix< Field, Sparse< Field, SparseMa-  
trix\_t::CSR\_HYB > >, 448

- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >, [448](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > >, [449](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >, [449](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >, [449](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >, [450](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >, [450](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > >, [450](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >, [450](#)
- isSparseMatrixMKLFormat< F, M >, [451](#)
- isSparseMatrixSimdFormat< F, M >, [451](#)
- isTerminated
  - ForStrategy1D< blocksize\_t, Cut, Param >, [428](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [430](#)
- isZero
  - RNSInteger< RNS >, [508](#)
  - RNSIntegerMod< RNS >, [513](#)
- isZOSparseMatrix< F, M >, [451](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >, [452](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >, [452](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >, [452](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >, [452](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >, [453](#)
- Iterative, [453](#)
- jbegin
  - ForStrategy2D< blocksize\_t, Cut, Param >, [430](#)
- jend
  - ForStrategy2D< blocksize\_t, Cut, Param >, [431](#)
- kaapi\_routines.inl, [960](#)
- \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL, [960](#)
- KellerGehrig
  - FFPACK::Protected, [371](#)
- KGFast
  - FFPACK::Protected, [371](#)
- KGFast\_generalized
  - FFPACK::Protected, [372](#)
- kmax
  - Sparse< \_Field, SparseMatrix\_t::COO >, [681](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [685](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [690](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- KrylovElim
  - FFPACK, [349](#)
- KrylovElim\_modular\_double
  - ffpack.C, [927](#)
  - ffpack\_c.h, [946](#)
- lapack.C, [983](#)
- \_\_FFLASFFPACK\_CONFIGURATION, [983](#)
- \_\_FFLASFFPACK\_HAVE\_LAPACK, [983](#)
- main, [984](#)
- LAPACKPerm2MathPerm
  - FFPACK, [288](#)
  - ffpack.C, [916](#)
  - ffpack\_c.h, [938](#)
- lastBlockSize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [429](#)
- lastCBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- lastRBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- launch\_fger
  - test-fger.C, [999](#)
- launch\_fger\_dispatch
  - test-fger.C, [1000](#)
- launch\_MM
  - test-fgemm.C, [996](#)
- launch\_MM\_dispatch
  - test-fgemm-check.C, [994](#)
  - test-fgemm.C, [996](#)
- launch\_MV
  - test-fgemv.C, [998](#)
- launch\_MV\_dispatch
  - test-fgemv.C, [998](#)
- launch\_test
  - test-charpoly.C, [986](#)
  - test-lu.C, [1021](#)
- launch\_wino
  - benchmark-wino.C, [739](#)
- LazyTag, [453](#)
- ld
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [690](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
- LeadingSubmatrixRankProfiles
  - FFPACK, [318](#)
  - ffpack.C, [929](#)
  - ffpack\_c.h, [949](#)
- lesser
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [521](#)



ScalFunctions< Element, typename enable\_if<  
   is\_integral< Element >::value >::type >, 527  
 Simd128\_impl< true, true, false, 2 >, 532  
 Simd128\_impl< true, true, false, 4 >, 541  
 Simd128\_impl< true, true, false, 8 >, 550  
 Simd128\_impl< true, true, true, 2 >, 562  
 Simd128\_impl< true, true, true, 4 >, 569  
 Simd128\_impl< true, true, true, 8 >, 577  
 Simd256\_impl< true, false, true, 8 >, 586  
 Simd256\_impl< true, true, false, 2 >, 591  
 Simd256\_impl< true, true, false, 4 >, 601, 604  
 Simd256\_impl< true, true, false, 8 >, 616  
 Simd256\_impl< true, true, true, 2 >, 628  
 Simd256\_impl< true, true, true, 4 >, 637, 642  
 Simd256\_impl< true, true, true, 8 >, 650  
 Simd512\_impl< true, false, true, 8 >, 657  
 Simd512\_impl< true, true, false, 8 >, 662  
 Simd512\_impl< true, true, true, 8 >, 675  
 lesser\_eq  
   ScalFunctions< Element, typename enable\_if<  
     is\_floating\_point< Element >::value >::type  
     >, 522  
   ScalFunctions< Element, typename enable\_if<  
     is\_integral< Element >::value >::type >, 527  
   Simd128\_impl< true, true, false, 2 >, 532  
   Simd128\_impl< true, true, false, 4 >, 541  
   Simd128\_impl< true, true, false, 8 >, 550  
   Simd128\_impl< true, true, true, 2 >, 562  
   Simd128\_impl< true, true, true, 4 >, 570  
   Simd128\_impl< true, true, true, 8 >, 577  
   Simd256\_impl< true, false, true, 8 >, 586  
   Simd256\_impl< true, true, false, 2 >, 591  
   Simd256\_impl< true, true, false, 4 >, 601, 604  
   Simd256\_impl< true, true, false, 8 >, 616  
   Simd256\_impl< true, true, true, 2 >, 628  
   Simd256\_impl< true, true, true, 4 >, 637, 642  
   Simd256\_impl< true, true, true, 8 >, 650  
   Simd512\_impl< true, false, true, 8 >, 657  
   Simd512\_impl< true, true, false, 8 >, 662  
   Simd512\_impl< true, true, true, 8 >, 675  
 limits< char >, 453  
   digits, 454  
   max, 454  
   min, 454  
   T, 454  
 limits< double >, 454  
   digits, 455  
   max, 454  
   min, 454  
   T, 454  
 limits< float >, 455  
   digits, 455  
   max, 455  
   min, 455  
   T, 455  
 limits< Givaro::Integer >, 455  
   max, 456  
   min, 456  
   T, 456  
 limits< int >, 456  
   digits, 456  
   max, 456  
   min, 456  
   T, 456  
 limits< long >, 456  
   digits, 457  
   max, 457  
   min, 457  
   T, 457  
 limits< long long >, 457  
   digits, 458  
   max, 457  
   min, 457  
   T, 457  
 limits< ReclInt::rint< K > >, 458  
   max, 458  
   min, 458  
   T, 458  
 limits< ReclInt::ruint< K > >, 458  
   max, 459  
   min, 459  
   T, 459  
 limits< short int >, 459  
   digits, 459  
   max, 459  
   min, 459  
   T, 459  
 limits< signed char >, 460  
   digits, 460  
   max, 460  
   min, 460  
   T, 460  
 limits< T >, 453  
 limits< unsigned char >, 460  
   digits, 461  
   max, 461  
   min, 461  
   T, 460  
 limits< unsigned int >, 461  
   digits, 461  
   max, 461  
   min, 461  
   T, 461  
 limits< unsigned long >, 461  
   digits, 462  
   max, 462  
   min, 462  
   T, 462  
 limits< unsigned long long >, 462  
   digits, 462  
   max, 462  
   min, 462  
   T, 462  
 limits< unsigned short int >, 463  
   digits, 463  
   max, 463

- min, [463](#)
- T, [463](#)
- load
  - Simd128\_impl< true, true, false, 2 >, [531](#), [534](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#), [543](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#), [552](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [574](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [590](#), [593](#)
  - Simd256\_impl< true, true, false, 4 >, [601](#), [603](#), [606](#)
  - Simd256\_impl< true, true, false, 8 >, [615](#), [618](#)
  - Simd256\_impl< true, true, true, 2 >, [624](#)
  - Simd256\_impl< true, true, true, 4 >, [633](#), [638](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [655](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#), [664](#)
  - Simd512\_impl< true, true, true, 8 >, [671](#)
- loadu
  - Simd128\_impl< true, true, false, 2 >, [532](#), [534](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#), [543](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#), [552](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [574](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [590](#), [593](#)
  - Simd256\_impl< true, true, false, 4 >, [601](#), [603](#), [606](#)
  - Simd256\_impl< true, true, false, 8 >, [615](#), [618](#)
  - Simd256\_impl< true, true, true, 2 >, [624](#)
  - Simd256\_impl< true, true, true, 4 >, [633](#), [638](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [655](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#), [664](#)
  - Simd512\_impl< true, true, true, 8 >, [671](#)
- LQUPtoInverseOfFullRankMinor
  - FFPACK, [327](#), [355](#)
- LT\_OBJDIR
  - config.h, [756](#)
- LUdivine
  - FFPACK, [300](#), [333](#), [334](#), [345](#)
- LUdivine\_construct
  - FFPACK::Protected, [370](#), [376](#)
- LUdivine\_gauss
  - FFPACK, [333](#), [346](#)
- LUdivine\_gauss\_modular\_double
  - ffpack\_c.h, [942](#)
- LUdivine\_modular\_double
  - ffpack.C, [920](#)
  - ffpack\_c.h, [942](#)
- LUdivine\_small
  - FFPACK, [333](#), [345](#)
- LUdivine\_small\_modular\_double
  - ffpack\_c.h, [942](#)
- LUKrylov
  - FFPACK::Protected, [372](#)
- LUKrylov\_KGFast
  - FFPACK::Protected, [373](#)
- m
  - Sparse< \_Field, SparseMatrix\_t::COO >, [681](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [685](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [690](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- MachineFloatTag, [463](#)
- MachineIntTag, [463](#)
- main
  - 101-fgemm.C, [1033](#)
  - 2x2-fgemm.C, [1034](#)
  - 2x2-ftsv.C, [1034](#)
  - 2x2-pluq.C, [1035](#)
  - arithprog.C, [712](#)
  - autotune/charpoly.C, [740](#)
  - autotune/pluq.C, [742](#)
  - benchmark-charpoly-mp.C, [716](#)
  - benchmark-charpoly.C, [716](#)
  - benchmark-checkers.C, [717](#)
  - benchmark-dgemm.C, [718](#)
  - benchmark-dgetrf.C, [719](#)
  - benchmark-dgetri.C, [720](#)
  - benchmark-dsytrf.C, [720](#)
  - benchmark-dtrsm.C, [721](#)
  - benchmark-dtrtri.C, [722](#)
  - benchmark-fadd-lvl2.C, [722](#)
  - benchmark-fdot.C, [723](#)
  - benchmark-fgemm-mp.C, [724](#)
  - benchmark-fgemm-rns.C, [726](#)
  - benchmark-fgemm.C, [726](#)
  - benchmark-fgemv-mp.C, [727](#)
  - benchmark-fgemv.C, [731](#)
  - benchmark-fgesv.C, [731](#)
  - benchmark-fsyrr.C, [732](#)
  - benchmark-fsytrf.C, [733](#)
  - benchmark-ftsm-mp.C, [733](#)
  - benchmark-ftsm.C, [734](#)
  - benchmark-ftsv.C, [734](#)
  - benchmark-fttri.C, [735](#)
  - benchmark-inverse.C, [736](#)
  - benchmark-lqup-mp.C, [736](#)
  - benchmark-lqup.C, [737](#)
  - benchmark-pluq.C, [738](#)
  - benchmark-wino.C, [739](#)
  - cblas.C, [978](#)
  - clapack.C, [979](#)



- cuda.C, [979](#)
- det.C, [740](#)
- examples/charpoly.C, [740](#)
- examples/pluq.C, [742](#)
- fblas.C, [980](#)
- fflas-101\_1.C, [1035](#)
- fflas-101\_3.C, [1035](#)
- fflas\_101.C, [1036](#)
- fflas\_101\_lvl1.C, [1036](#)
- ffpack-fgesv.C, [1036](#)
- ffpack-solve.C, [1037](#)
- fsyrk.C, [712](#)
- fsytrf.C, [713](#)
- fttrtri.C, [714](#)
- gmp.C, [980](#)
- lapack.C, [984](#)
- matmul.C, [741](#)
- rank.C, [743](#)
- regression-check.C, [984](#)
- solve.C, [743](#)
- test-charpoly-check.C, [985](#)
- test-charpoly.C, [986](#)
- test-compressQ.C, [987](#)
- test-det-check.C, [988](#)
- test-det.C, [988](#)
- test-echelon.C, [991](#)
- test-fadd.C, [992](#)
- test-fdot.C, [993](#)
- test-fgemm-check.C, [994](#)
- test-fgemm.C, [997](#)
- test-fgemv.C, [998](#)
- test-fger.C, [1000](#)
- test-fgesv.C, [1001](#)
- test-finit.C, [1002](#)
- test-fscal.C, [1004](#)
- test-fsyr2k.C, [1005](#)
- test-fsyrk.C, [1007](#)
- test-fsytrf.C, [1008](#)
- test-ftmm.C, [1009](#)
- test-ftmv.C, [1010](#)
- test-ftsm-check.C, [1011](#)
- test-ftsm.C, [1012](#)
- test-ftssyr2k.C, [1013](#)
- test-ftstr.C, [1014](#)
- test-ftsv.C, [1015](#)
- test-fttrtri.C, [1016](#)
- test-interfaces-c.c, [1017](#)
- test-invert-check.C, [1017](#)
- test-io.C, [1018](#)
- test-lu.C, [1021](#)
- test-maxdelayeddim.C, [1023](#)
- test-minpoly.C, [1024](#)
- test-multifile2.C, [1024](#)
- test-nullspace.C, [1025](#)
- test-permutations.C, [1026](#)
- test-pluq-check.C, [1027](#)
- test-rankprofiles.C, [1028](#)
- test-rpm.C, [1028](#)
- test-simd.C, [1032](#)
- test-solve.C, [1033](#)
- winograd.C, [715](#)
- mainpage.doxy, [739](#)
- mask\_high
  - Simd128\_impl< true, true, false, 8 >, [554](#)
  - Simd128\_impl< true, true, true, 8 >, [578](#)
  - Simd256\_impl< true, true, false, 8 >, [621](#)
  - Simd256\_impl< true, true, true, 8 >, [650](#)
  - Simd512\_impl< true, true, false, 8 >, [667](#)
  - Simd512\_impl< true, true, true, 8 >, [676](#)
- mask\_t
  - read\_sparse.h, [841](#)
- maskstore
  - Simd512\_impl< true, true, false, 8 >, [661](#), [664](#)
  - Simd512\_impl< true, true, true, 8 >, [671](#)
- MatF2MatD\_Triangular
  - FFLAS::Protected, [210](#)
- MatF2MatFI\_Triangular
  - FFLAS::Protected, [210](#)
- MathPerm2LAPACKPerm
  - FFPACK, [288](#)
  - ffpack.C, [916](#)
  - ffpack\_c.h, [938](#)
- Matio.h, [976](#)
  - read\_field, [977](#)
  - write\_field, [977](#)
- matmul.C, [741](#)
  - main, [741](#)
- matmul.doxy, [775](#)
- Matrix Multiplication Algorithms, [42](#)
- MatrixApplyS
  - FFPACK, [335](#), [336](#)
- MatrixApplyS\_modular\_double
  - ffpack.C, [916](#)
  - ffpack\_c.h, [938](#)
- MatrixApplyT
  - FFPACK, [337](#)
- MatrixApplyT\_modular\_double
  - ffpack.C, [917](#)
  - ffpack\_c.h, [938](#)
- MatVecMinPoly
  - FFPACK, [310](#), [348](#)
  - FFPACK::Protected, [373](#)
- max
  - HelperMod< Field, ElementCategories::MachineIntTag >, [442](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [444](#)
  - limits< char >, [454](#)
  - limits< double >, [454](#)
  - limits< float >, [455](#)
  - limits< Givaro::Integer >, [456](#)
  - limits< int >, [456](#)
  - limits< long >, [457](#)
  - limits< long long >, [457](#)
  - limits< RecInt::rint< K > >, [458](#)
  - limits< RecInt::ruint< K > >, [459](#)

- limits< short int >, [459](#)
- limits< signed char >, [460](#)
- limits< unsigned char >, [461](#)
- limits< unsigned int >, [461](#)
- limits< unsigned long >, [462](#)
- limits< unsigned long long >, [462](#)
- limits< unsigned short int >, [463](#)
- max3
  - FFLAS, [75](#)
- max4
  - FFLAS, [75](#)
- MAX\_THREADS
  - parallel.h, [962](#)
- MAX\_WITH\_SIZE\_T
  - test-maxdelayeddim.C, [1023](#)
- maxCol
  - StatsMatrix, [702](#)
- maxColDifference
  - StatsMatrix, [703](#)
- MaxDelayedDim
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [466](#)
- maxElement
  - RNSIntegerMod< RNS >, [513](#)
- maxFieldElt
  - FFPACK, [368](#)
- maxFieldElt< Givaro::ZRing< Givaro::Integer > >
  - FFPACK, [368](#)
- maxpy
  - FieldSimd< \_Field >, [419](#)
- maxpyin
  - FieldSimd< \_Field >, [419](#)
- maxRow
  - StatsMatrix, [702](#)
- maxrow
  - Sparse< \_Field, SparseMatrix\_t::COO >, [681](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [683](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [686](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [691](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- maxRowDifference
  - StatsMatrix, [703](#)
- MaxStorableValue
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [468](#)
- MG\_DEFAULT
  - benchmark-fgemm-mp.C, [724](#)
  - benchmark-fgemv-mp.C, [727](#)
- min
  - HelperMod< Field, ElementCategories::MachineIntTag >, [441](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [444](#)
  - limits< char >, [454](#)
  - limits< double >, [454](#)
  - limits< float >, [455](#)
  - limits< Givaro::Integer >, [456](#)
  - limits< int >, [456](#)
  - limits< long >, [457](#)
  - limits< long long >, [457](#)
  - limits< RecInt::rint< K > >, [458](#)
  - limits< RecInt::ruint< K > >, [459](#)
  - limits< short int >, [459](#)
  - limits< signed char >, [460](#)
  - limits< unsigned char >, [461](#)
  - limits< unsigned int >, [461](#)
  - limits< unsigned long >, [462](#)
  - limits< unsigned long long >, [462](#)
  - limits< unsigned short int >, [463](#)
- min3
  - FFLAS, [75](#)
- min4
  - FFLAS, [75](#)
- min\_types
  - FFLAS::Protected, [207](#), [208](#)
- minCol
  - StatsMatrix, [702](#)
- minColDifference
  - StatsMatrix, [703](#)
- minElement
  - RNSIntegerMod< RNS >, [513](#)
- MinPoly
  - FFPACK, [309](#), [348](#)
- minRow
  - StatsMatrix, [702](#)
- minRowDifference
  - StatsMatrix, [703](#)
- MKL\_CONFIG, [376](#)
- MKLSparseMatrixFormat
  - FFLAS, [71](#)
- MMHelper
  - MMHelper< FFPACK::RNSInteger< E >, Algo-Trait, ModeCategories::DefaultTag, ParSeq-Trait >, [469](#), [470](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, Algo-Trait, ModeCategories::DefaultTag, ParSeq-Trait >, [471](#), [472](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [473](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [475](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [477](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [465](#)
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait,

- ModeCategories::DefaultTag, ParSeqTrait >, 469
- MMHelper, 469, 470
- normA, 470
- normB, 470
- operator<<, 470
- parseq, 470
- recLevel, 470
- Self\_t, 469
- setNorm, 470
- MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 471
  - MMHelper, 471, 472
  - normA, 472
  - normB, 472
  - operator<<, 472
  - parseq, 472
  - recLevel, 472
  - Self\_t, 471
  - setNorm, 472
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<mod Dest >, ParSeqTrait >, 473
  - MMHelper, 473
  - operator<<, 474
  - parseq, 474
  - recLevel, 474
  - Self\_t, 473
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 474
  - MMHelper, 475
  - normA, 476
  - normB, 476
  - operator<<, 476
  - parseq, 476
  - recLevel, 476
  - Self\_t, 475
  - setNorm, 475
- MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 476
  - MMHelper, 477
  - operator<<, 477
  - parseq, 477
  - recLevel, 477
  - Self\_t, 477
- MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 464
  - Amax, 468
  - Amin, 467
  - Aunfit, 466
  - Bmax, 468
  - Bmin, 468
  - Bunfit, 466
  - checkA, 466
  - checkB, 467
  - checkOut, 467
  - Cmax, 468
  - Cmin, 468
  - DelayedField, 465
  - delayedField, 468
  - DelayedField\_t, 465
  - DFElt, 465
  - FieldMax, 467
  - FieldMin, 467
  - initA, 466
  - initB, 466
  - initC, 466
  - initOut, 466
  - MaxDelayedDim, 466
  - MaxStorableValue, 468
  - MMHelper, 465
  - operator<<, 467
  - Outmax, 468
  - Outmin, 468
  - parseq, 468
  - recLevel, 467
  - Self\_t, 465
  - setOutBounds, 466
  - FieldSimd< \_Field >, 417
  - Simd128\_impl< true, true, false, 2 >, 537
  - Simd128\_impl< true, true, false, 4 >, 545
  - Simd128\_impl< true, true, false, 8 >, 555
  - Simd128\_impl< true, true, true, 2 >, 562
  - Simd128\_impl< true, true, true, 4 >, 570
  - Simd128\_impl< true, true, true, 8 >, 578
  - Simd256\_impl< true, false, true, 8 >, 587
  - Simd256\_impl< true, true, false, 2 >, 596
  - Simd256\_impl< true, true, false, 4 >, 611
  - Simd256\_impl< true, true, false, 8 >, 621
  - Simd256\_impl< true, true, true, 2 >, 628
  - Simd256\_impl< true, true, true, 4 >, 637, 642
  - Simd256\_impl< true, true, true, 8 >, 651
  - Simd512\_impl< true, true, false, 8 >, 667
  - Simd512\_impl< true, true, true, 8 >, 676
- MODE
  - parallel.h, 964
- ModeTraits< Field >, 478
  - value, 478
- ModeTraits< Givaro::Modular< Element, Compute > >, 478
  - value, 478
- ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >, 478
  - value, 479
- ModeTraits< Givaro::Modular< int16\_t, Compute > >, 479
  - value, 479
- ModeTraits< Givaro::Modular< int32\_t, Compute > >, 479
  - value, 479
- ModeTraits< Givaro::Modular< int8\_t, Compute > >, 479
  - value, 480

- ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >, [480](#)
  - value, [480](#)
- ModeTraits< Givaro::Modular< uint16\_t, Compute > >, [480](#)
  - value, [480](#)
- ModeTraits< Givaro::Modular< uint32\_t, Compute > >, [480](#)
  - value, [481](#)
- ModeTraits< Givaro::Modular< uint8\_t, Compute > >, [481](#)
  - value, [481](#)
- ModeTraits< Givaro::ModularBalanced< Element > >, [481](#)
  - value, [481](#)
- ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, [481](#)
  - value, [482](#)
- ModeTraits< Givaro::ModularBalanced< int16\_t > >, [482](#)
  - value, [482](#)
- ModeTraits< Givaro::ModularBalanced< int32\_t > >, [482](#)
  - value, [482](#)
- ModeTraits< Givaro::ModularBalanced< int8\_t > >, [482](#)
  - value, [483](#)
- ModeTraits< Givaro::Montgomery< T > >, [483](#)
  - value, [483](#)
- ModeTraits< Givaro::ZRing< double > >, [483](#)
  - value, [483](#)
- ModeTraits< Givaro::ZRing< float > >, [483](#)
  - value, [483](#)
- ModeTraits< Givaro::ZRing< Givaro::Integer > >, [484](#)
  - value, [484](#)
- ModField
  - rns\_double, [492](#)
  - rns\_double\_extended, [503](#)
  - RNSIntegerMod< RNS >, [512](#)
- modp
  - FFLAS::vectorised, [269](#), [270](#)
  - FFLAS::vectorised::unswitch, [271](#)
- ModularBalanced< T >, [484](#)
- ModularTag, [484](#)
- mOne
  - RNSInteger< RNS >, [510](#)
  - RNSIntegerMod< RNS >, [517](#)
- mone
  - FFLAS, [74](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [696](#)
- MonotonicApplyP
  - FFPACK, [290](#)
- MonotonicCompress
  - FFPACK, [334](#)
- MonotonicCompressCycles
  - FFPACK, [334](#)
- MonotonicCompressMorePivots
  - FFPACK, [334](#)
- MonotonicExpand
  - FFPACK, [335](#)
- Montgomery< T >, [484](#)
- mul
  - FieldSimd< \_Field >, [417](#)
  - RNSIntegerMod< RNS >, [514](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [520](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [524](#)
  - Simd128\_impl< true, true, false, 2 >, [536](#)
  - Simd128\_impl< true, true, false, 4 >, [544](#)
  - Simd128\_impl< true, true, false, 8 >, [553](#)
  - Simd128\_impl< true, true, true, 2 >, [560](#)
  - Simd128\_impl< true, true, true, 4 >, [568](#)
  - Simd128\_impl< true, true, true, 8 >, [575](#)
  - Simd256\_impl< true, false, true, 8 >, [585](#)
  - Simd256\_impl< true, true, false, 2 >, [595](#)
  - Simd256\_impl< true, true, false, 4 >, [609](#)
  - Simd256\_impl< true, true, false, 8 >, [620](#)
  - Simd256\_impl< true, true, true, 2 >, [626](#)
  - Simd256\_impl< true, true, true, 4 >, [635](#), [640](#)
  - Simd256\_impl< true, true, true, 8 >, [648](#)
  - Simd512\_impl< true, false, true, 8 >, [656](#)
  - Simd512\_impl< true, true, false, 8 >, [666](#)
  - Simd512\_impl< true, true, true, 8 >, [673](#)
- mul\_r
  - FieldSimd< \_Field >, [417](#), [418](#)
- mulhi
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [524](#)
  - Simd128\_impl< true, true, false, 2 >, [532](#)
  - Simd128\_impl< true, true, false, 4 >, [541](#)
  - Simd128\_impl< true, true, true, 2 >, [560](#)
  - Simd128\_impl< true, true, true, 4 >, [568](#)
  - Simd256\_impl< true, true, false, 2 >, [591](#)
  - Simd256\_impl< true, true, false, 4 >, [602](#), [604](#)
  - Simd256\_impl< true, true, true, 2 >, [626](#)
  - Simd256\_impl< true, true, true, 4 >, [635](#), [640](#)
- mulhi\_fast
  - Simd128\_impl< true, true, false, 8 >, [554](#)
  - Simd128\_impl< true, true, true, 8 >, [578](#)
  - Simd256\_impl< true, true, false, 8 >, [621](#)
  - Simd256\_impl< true, true, true, 8 >, [650](#)
  - Simd512\_impl< true, true, false, 8 >, [667](#)
  - Simd512\_impl< true, true, true, 8 >, [676](#)
- mulin
  - FieldSimd< \_Field >, [417](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [520](#)
  - Simd256\_impl< true, false, true, 8 >, [585](#)
  - Simd512\_impl< true, false, true, 8 >, [656](#)
- mullo
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [524](#)
  - Simd128\_impl< true, true, false, 2 >, [536](#)

- Simd128\_impl< true, true, false, 4 >, [544](#)
- Simd128\_impl< true, true, false, 8 >, [550](#)
- Simd128\_impl< true, true, true, 2 >, [560](#)
- Simd128\_impl< true, true, true, 4 >, [567](#)
- Simd128\_impl< true, true, true, 8 >, [575](#)
- Simd256\_impl< true, true, false, 2 >, [595](#)
- Simd256\_impl< true, true, false, 4 >, [609](#)
- Simd256\_impl< true, true, false, 8 >, [616](#)
- Simd256\_impl< true, true, true, 2 >, [626](#)
- Simd256\_impl< true, true, true, 4 >, [635](#), [640](#)
- Simd256\_impl< true, true, true, 8 >, [648](#)
- Simd512\_impl< true, true, false, 8 >, [662](#)
- Simd512\_impl< true, true, true, 8 >, [673](#)
- mulx
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [524](#)
  - Simd128\_impl< true, true, false, 2 >, [533](#)
  - Simd128\_impl< true, true, false, 4 >, [541](#)
  - Simd128\_impl< true, true, false, 8 >, [550](#)
  - Simd128\_impl< true, true, true, 2 >, [560](#)
  - Simd128\_impl< true, true, true, 4 >, [568](#)
  - Simd128\_impl< true, true, true, 8 >, [575](#)
  - Simd256\_impl< true, true, false, 2 >, [591](#)
  - Simd256\_impl< true, true, false, 4 >, [602](#), [604](#)
  - Simd256\_impl< true, true, false, 8 >, [616](#)
  - Simd256\_impl< true, true, true, 2 >, [626](#)
  - Simd256\_impl< true, true, true, 4 >, [635](#), [640](#)
  - Simd256\_impl< true, true, true, 8 >, [648](#)
  - Simd512\_impl< true, true, false, 8 >, [662](#)
  - Simd512\_impl< true, true, true, 8 >, [673](#)
- mvcnt
  - test-lu.C, [1022](#)
- n
  - Sparse< \_Field, SparseMatrix\_t::COO >, [681](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [686](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [690](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- nChunks
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [691](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- nDenseCols
  - StatsMatrix, [703](#)
- nDenseRows
  - StatsMatrix, [703](#)
- need\_field\_characteristic< Field >, [484](#)
- value, [484](#)
- need\_field\_characteristic< Givaro::Modular< Field > >, [485](#)
- value, [485](#)
- need\_field\_characteristic< Givaro::ModularBalanced< Field > >, [485](#)
- value, [485](#)
- NeedDoublePreAddReduction
  - FFLAS::Protected, [203](#), [204](#)
- NeedPreAddReduction
  - FFLAS::Protected, [202](#), [203](#)
- NeedPreSubReduction
  - FFLAS::Protected, [203](#)
- neg
  - RNSIntegerMod< RNS >, [514](#)
- nElements
  - Sparse< \_Field, SparseMatrix\_t::COO >, [681](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [683](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [686](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [691](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [700](#)
- nEmptyCols
  - StatsMatrix, [703](#)
- nEmptyColsEnd
  - StatsMatrix, [703](#)
- nEmptyRows
  - StatsMatrix, [703](#)
- newD
  - FFPACK::Protected, [374](#)
- NEWWINO
  - fgemm\_winograd.inl, [775](#)
- nMOnes
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [686](#)
  - StatsMatrix, [702](#)
- nnz
  - Sparse< \_Field, SparseMatrix\_t::COO >, [681](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [683](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [686](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [687](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [689](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [690](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [692](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [694](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)

- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 699
- StatsMatrix, 702
- none
  - HelperFlag, 440
- nOnes
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 686
  - StatsMatrix, 702
- NonZeroRandomMatrix
  - FFPACK, 355, 356
- normA
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 470
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 472
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 476
- normB
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 470
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 472
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 476
- NORML\_MOD
  - fflas\_simd.h, 806
- NoSimd< T >, 485
  - compliant, 486
  - scalar\_t, 486
  - type\_string, 486
  - valid, 486
  - vect\_size, 486
  - vect\_t, 486
- NoSimdSparseMatrix
  - FFLAS, 71
- NOSPLIT
  - parallel.h, 966
- nOthers
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 686
  - StatsMatrix, 702
- NotMKLSparseMatrixFormat
  - FFLAS, 71
- NotZOSparseMatrix
  - FFLAS, 70
- NullSpaceBasis
  - FFPACK, 314, 351
- NullSpaceBasis\_modular\_double
  - ffpack.C, 929
  - ffpack\_c.h, 948
- NUM\_THREADS
  - parallel.h, 962
- NUMARGS
  - parallel.h, 964
- number\_kind
  - FFLAS, 74
- numBlock
  - ForStrategy1D< blocksize\_t, Cut, Param >, 429
- numblocks
  - ForStrategy1D< blocksize\_t, Cut, Param >, 428
- numColBlock
  - ForStrategy2D< blocksize\_t, Cut, Param >, 433
- numRowBlock
  - ForStrategy2D< blocksize\_t, Cut, Param >, 433
- numthreads
  - Parallel< C, P >, 487
  - Sequential, 528
- one
  - FFLAS, 74
  - RNSInteger< RNS >, 510
  - RNSIntegerMod< RNS >, 516
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 696
- OPENBLAS\_NUM\_THREADS
  - config.h, 756
- operator!=
  - rns\_double\_elt\_cstptr, 499
  - rns\_double\_elt\_ptr, 502
- operator<
  - rns\_double\_elt\_cstptr, 499
  - rns\_double\_elt\_ptr, 502
- operator<<
  - Compose< H1, H2 >, 393
  - FFLAS, 143
  - ForStrategy2D< blocksize\_t, Cut, Param >, 431
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 470
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 472
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, 474
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 476
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 477
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 467
  - Parallel< C, P >, 487
  - Sequential, 528
  - test-fsytrf.C, 1007
- operator()
  - callLUdivine\_small< double >, 382
  - callLUdivine\_small< Element >, 381
  - callLUdivine\_small< float >, 382
  - Failure, 411, 412
  - readMyMachineType< Field, mpz\_t >, 491
  - readMyMachineType< Field, T >, 490
  - RNSInteger< RNS >::RandIter, 488



- RNSIntegerMod< RNS >::RandIter, [489](#)
- rnsRandIter< RNS >, [518](#)
- tfn\_minus, [706](#)
- tfn\_minus\_eq, [706](#)
- tfn\_mul, [707](#)
- tfn\_mul\_eq, [707](#)
- tfn\_plus, [707](#)
- tfn\_plus\_eq, [708](#)
- operator+
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [502](#)
- operator++
  - ForStrategy1D< blocksize\_t, Cut, Param >, [428](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [431](#)
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [501](#)
- operator+=
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [502](#)
- operator-
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [502](#)
- operator--
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [502](#)
- operator-=
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [502](#)
- operator=
  - Coo< Field >, [401](#)
  - Coo< ValT, IdxT >, [399](#), [402](#)
  - FieldSimd< \_Field >, [415](#)
  - Info, [445](#), [446](#)
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [502](#)
- operator&
  - rns\_double\_elt, [497](#)
  - rns\_double\_elt\_cstptr, [499](#), [500](#)
  - rns\_double\_elt\_ptr, [501](#), [502](#)
- operator[]
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [501](#)
- operator\*
  - rns\_double\_elt\_cstptr, [499](#)
  - rns\_double\_elt\_ptr, [501](#)
- other
  - FFLAS, [74](#)
  - rns\_double\_elt\_cstptr, [500](#)
  - rns\_double\_elt\_ptr, [502](#)
- Outmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [468](#)
- Outmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [468](#)
- p
  - HelperMod< Field, ElementCategories::MachineIntTag >, [441](#)
- HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, [442](#)
- HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, [443](#)
- HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [444](#)
- pack\_lhs
  - FFLAS::details, [195](#)
- pack\_rhs
  - FFLAS::details, [195](#)
- PACKAGE
  - config.h, [756](#)
- PACKAGE\_BUGREPORT
  - config.h, [756](#)
- PACKAGE\_NAME
  - config.h, [756](#)
- PACKAGE\_STRING
  - config.h, [756](#)
- PACKAGE\_TARNAME
  - config.h, [756](#)
- PACKAGE\_URL
  - config.h, [756](#)
- PACKAGE\_VERSION
  - config.h, [757](#)
- PAR\_BLOCK
  - parallel.h, [961](#)
- Parallel
  - Parallel< C, P >, [487](#)
- Parallel< C, P >, [486](#)
  - Cut, [487](#)
  - numthreads, [487](#)
  - operator<<, [487](#)
  - Parallel, [487](#)
  - Param, [487](#)
  - set\_numthreads, [487](#)
- parallel.h, [960](#)
  - \_\_FFLASFFPACK\_SEQUENTIAL, [961](#)
  - BARRIER, [961](#)
  - BEGIN\_PARALLEL\_MAIN, [962](#)
  - CHECK\_DEPENDENCIES, [961](#)
  - COMMA, [964](#)
  - CONSTREFERENCE, [962](#)
  - END\_PARALLEL\_MAIN, [962](#)
  - FOR1D, [963](#)
  - FOR2D, [963](#)
  - FORBLOCK1D, [962](#)
  - FORBLOCK2D, [963](#)
  - index\_t, [961](#)
  - MAX\_THREADS, [962](#)
  - MODE, [964](#)
  - NOSPLIT, [966](#)
  - NUM\_THREADS, [962](#)
  - NUMARGS, [964](#)
  - PAR\_BLOCK, [961](#)
  - PARFOR1D, [963](#)
  - PARFOR2D, [964](#)
  - PARFORBLOCK1D, [963](#)
  - PARFORBLOCK2D, [964](#)

- PP\_ARG\_N, [964](#)
- PP\_NARG\_, [964](#)
- PP\_RSEQ\_N, [965](#)
- READ, [962](#)
- READWRITE, [962](#)
- RETURNPARAM, [964](#)
- splitt, [966](#)
- SPLITTER, [966](#)
- splitting\_0, [966](#)
- splitting\_1, [966](#)
- splitting\_2, [966](#)
- splitting\_3, [966](#)
- SYNCH\_GROUP, [962](#)
- TASK, [961](#)
- VALUE, [962](#)
- WAIT, [961](#)
- WRITE, [962](#)
- Param
  - Parallel< C, P >, [487](#)
- PARFOR1D
  - parallel.h, [963](#)
- PARFOR2D
  - parallel.h, [964](#)
- PARFORBLOCK1D
  - parallel.h, [963](#)
- PARFORBLOCK2D
  - parallel.h, [964](#)
- parseArguments
  - FFLAS, [177](#)
- parseq
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [470](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [472](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [474](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [476](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [477](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [468](#)
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [710](#)
- PBASECASE\_K
  - ffpack\_ppluq.inl, [872](#)
- pColumnEchelonForm
  - FFPACK, [301](#)
- pColumnEchelonForm\_modular\_double
  - ffpack.C, [923](#)
- pColumnEchelonForm\_modular\_float
  - ffpack.C, [924](#)
- pColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [925](#)
- pColumnRankProfile
  - FFPACK, [317](#)
- pDet
  - FFPACK, [312](#)
- perm
  - Info, [445](#), [446](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [700](#)
- PermApplyS
  - FFPACK, [336](#)
- PermApplyS\_double
  - ffpack.C, [917](#)
  - ffpack\_c.h, [938](#)
- PermApplyT
  - FFPACK, [338](#)
- PermApplyT\_double
  - ffpack.C, [917](#)
  - ffpack\_c.h, [939](#)
- pfadd
  - FFLAS, [77](#)
- pfaddin
  - FFLAS, [77](#)
- pfgemm
  - FFLAS, [134](#), [174–176](#)
- pfgemm\_1D\_rec
  - FFLAS, [134](#)
- pfgemm\_2D\_rec
  - FFLAS, [135](#)
- pfgemm\_3D\_rec
  - FFLAS, [135](#)
- pfgemm\_3D\_rec2
  - FFLAS, [135](#)
- pfgemm\_variants.inl, [966](#)
- pfgemv.inl, [967](#)
- pfrend
  - FFLAS, [173](#)
- pfreduce
  - FFLAS, [107](#)
- pfspmm
  - FFLAS::sparse\_details, [220–222](#)
  - FFLAS::sparse\_details\_impl, [237](#), [244](#), [245](#), [247–249](#), [259](#)
- pfspmm\_dispatch
  - FFLAS::sparse\_details, [219](#), [220](#)
- pfspmm\_mone
  - FFLAS::sparse\_details\_impl, [238](#)
- pfspmm\_one
  - FFLAS::sparse\_details\_impl, [238](#)
- pfspmm\_zo
  - FFLAS::sparse\_details\_impl, [249](#)
- pfspmv
  - FFLAS::sparse\_details, [222](#), [223](#)
  - FFLAS::sparse\_details\_impl, [239](#), [246](#), [249](#), [250](#), [255](#), [259–262](#)
- pfspmv\_mone
  - FFLAS::sparse\_details\_impl, [240](#), [250](#), [251](#), [256](#), [262](#)
- pfspmv\_one
  - FFLAS::sparse\_details\_impl, [239](#), [240](#), [250](#), [256](#), [262](#)



- pfspmv\_task
  - FFLAS::sparse\_details\_impl, [239](#)
- pfsb
  - FFLAS, [77](#)
- pfsubin
  - FFLAS, [78](#)
- pfzero
  - FFLAS, [173](#)
- PLUQ
  - FFPACK, [298](#), [299](#), [341](#), [342](#), [345](#)
- pluq.C, [741](#), [742](#)
- PLUQ\_basecaseCrout
  - FFPACK, [340](#)
- PLUQ\_basecaseV2
  - FFPACK, [340](#)
- PLUQ\_basecaseV3
  - FFPACK, [340](#)
- PLUQ\_modular\_double
  - ffpack.C, [920](#)
  - ffpack\_c.h, [942](#)
- PLUQtoEchelonPermutation
  - FFPACK, [327](#)
  - ffpack.C, [933](#)
  - ffpack\_c.h, [952](#)
- pm1
  - HelperFlag, [441](#)
- pMMH
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [709](#)
- pow50rem
  - HelperMod< Field, ElementCategories::MachineIntTag >, [442](#)
- PP\_ARG\_N
  - parallel.h, [964](#)
- PP\_NARG\_
  - parallel.h, [964](#)
- PP\_RSEQ\_N
  - parallel.h, [965](#)
- pPLUQ
  - FFPACK, [299](#)
- pRank
  - FFPACK, [311](#)
- preamble
  - FFLAS, [178](#)
- precompute\_cst
  - rns\_double, [493](#)
  - rns\_double\_extended, [504](#)
- pReducedColumnEchelonForm
  - FFPACK, [304](#)
- pReducedColumnEchelonForm\_modular\_double
  - ffpack.C, [924](#)
- pReducedColumnEchelonForm\_modular\_float
  - ffpack.C, [925](#)
- pReducedColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [926](#)
- pReducedRowEchelonForm
  - FFPACK, [305](#)
- pReducedRowEchelonForm\_modular\_double
  - ffpack.C, [924](#)
- pReducedRowEchelonForm\_modular\_float
  - ffpack.C, [925](#)
- pReducedRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [926](#)
- prefetch
  - FFLAS, [180](#)
- print
  - Failure, [412](#)
- printHelpMessage
  - args-parser.h, [969](#)
- printPolynomial
  - test-charpoly-check.C, [985](#)
- printvect
  - test-compressQ.C, [987](#)
- pRowEchelonForm
  - FFPACK, [302](#)
- pRowEchelonForm\_modular\_double
  - ffpack.C, [924](#)
- pRowEchelonForm\_modular\_float
  - ffpack.C, [925](#)
- pRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [925](#)
- pRowRankProfile
  - FFPACK, [316](#)
- PSeq
  - benchmark-fgemm-rns.C, [726](#)
- pSolve
  - FFPACK, [314](#)
- PTRSM\_HYBRID\_THRESHOLD
  - fflas\_ptrsm.inl, [805](#)
- PURE
  - fflas\_simd.h, [806](#)
- queryCacheSizes
  - FFLAS, [180](#)
- queryL1CacheSize
  - FFLAS, [181](#)
- queryTopLevelCacheSize
  - FFLAS, [181](#)
- RandInt
  - FFPACK, [359](#)
- RandIter
  - RNSInteger< RNS >::RandIter, [488](#)
  - RNSIntegerMod< RNS >::RandIter, [489](#)
- random
  - RNSInteger< RNS >::RandIter, [488](#)
  - RNSIntegerMod< RNS >::RandIter, [489](#)
  - rnsRandIter< RNS >, [517](#), [518](#)
- RandomIndexSubset
  - FFPACK, [361](#)
- RandomKrylovPrecond
  - FFPACK::Protected, [373](#)
- RandomMatrix
  - FFPACK, [357](#)
- RandomMatrixWithDet
  - FFPACK, [366](#), [367](#)
- RandomMatrixWithRank
  - FFPACK, [359](#), [360](#)

- RandomMatrixWithRankandRandomRPM
  - FFPACK, [364](#), [365](#)
- RandomMatrixWithRankandRPM
  - FFPACK, [362](#), [363](#)
- RandomNullSpaceVector
  - FFPACK, [314](#), [327](#), [351](#)
- RandomNullSpaceVector\_modular\_double
  - ffpack.C, [928](#)
  - ffpack\_c.h, [948](#)
- RandomPermutation
  - FFPACK, [361](#)
- RandomRankProfileMatrix
  - FFPACK, [361](#)
- RandomSymmetricMatrix
  - FFPACK, [359](#)
- RandomSymmetricMatrixWithRankandRandomRPM
  - FFPACK, [365](#), [366](#)
- RandomSymmetricMatrixWithRankandRPM
  - FFPACK, [363](#), [364](#)
- RandomSymmetricRankProfileMatrix
  - FFPACK, [362](#)
- RandomTriangularMatrix
  - FFPACK, [358](#)
- Rank
  - FFPACK, [310](#), [311](#), [349](#)
- rank.C, [742](#)
  - main, [743](#)
- Rank\_modular\_double
  - ffpack.C, [927](#)
  - ffpack\_c.h, [947](#)
- RankProfileFromLU
  - FFPACK, [317](#)
  - ffpack.C, [929](#)
  - ffpack\_c.h, [949](#)
- READ
  - parallel.h, [962](#)
- read\_field
  - Matio.h, [977](#)
- read\_sparse.h, [840](#)
  - DNS\_BIN\_VER, [841](#)
  - mask\_t, [841](#)
- readDnsFormat
  - FFLAS, [144](#)
- readMachineType
  - FFLAS, [144](#)
- ReadMatrix
  - FFLAS, [178](#)
- readMyMachineType< Field, mpz\_t >, [490](#)
  - Element, [491](#)
  - Element\_ptr, [491](#)
  - operator(), [491](#)
- readMyMachineType< Field, T >, [490](#)
  - Element, [490](#)
  - Element\_ptr, [490](#)
  - operator(), [490](#)
- readOrRandomMatrixWithRankAndRandomRPM
  - test-nullspace.C, [1025](#)
- readSmsFormat
  - FFLAS, [143](#)
- readSprFormat
  - FFLAS, [143](#)
- READWRITE
  - parallel.h, [962](#)
- Rec\_Initialize
  - benchmark-pluq.C, [738](#)
- RecInt, [376](#)
- recLevel
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [470](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [472](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [474](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [476](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [477](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [467](#)
- Recursive, [491](#)
- reduce
  - FFLAS::vectorised, [268](#), [269](#)
  - rns\_double, [494](#)
  - rns\_double\_extended, [505](#)
  - RNSInteger< RNS >, [509](#)
  - RNSIntegerMod< RNS >, [513](#), [514](#)
- reduce\_modp
  - RNSIntegerMod< RNS >, [515](#), [516](#)
- reduce\_modp\_rnsmajor
  - RNSIntegerMod< RNS >, [516](#)
- ReducedColumnEchelonForm
  - FFPACK, [303](#), [304](#), [347](#)
- ReducedColumnEchelonForm\_modular\_double
  - ffpack.C, [921](#)
  - ffpack\_c.h, [944](#)
- ReducedColumnEchelonForm\_modular\_float
  - ffpack.C, [922](#)
  - ffpack\_c.h, [944](#)
- ReducedColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [923](#)
  - ffpack\_c.h, [945](#)
- ReducedRowEchelonForm
  - FFPACK, [304](#), [305](#), [346](#)
- ReducedRowEchelonForm2\_modular\_double
  - ffpack\_c.h, [945](#)
- ReducedRowEchelonForm\_modular\_double
  - ffpack.C, [921](#)
  - ffpack\_c.h, [944](#)
- ReducedRowEchelonForm\_modular\_float
  - ffpack.C, [922](#)
  - ffpack\_c.h, [945](#)
- ReducedRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [923](#)

- ffpack\_c.h, 945
- REF\_modular\_double
  - ffpack\_c.h, 946
- REGISTER\_TYPE\_NAME
  - test-simd.C, 1030, 1031
- regression-check.C, 984
  - check1, 984
  - check2, 984
  - check3, 984
  - check4, 984
  - checkZeroDimCharpoly, 984
  - checkZeroDimMinPoly, 984
  - gf2ModularBalanced, 984
  - main, 984
- RETURNPARAM
  - parallel.h, 964
- ring
  - RNSInteger< RNS >::RandIter, 488
  - RNSIntegerMod< RNS >::RandIter, 489
  - rnsRandIter< RNS >, 518
- rint< K >, 491
- RNS, 43
  - benchmark-fgemm-rns.C, 725
- rns
  - RNSInteger< RNS >, 508
  - RNSIntegerMod< RNS >, 512
- rns-double-elt.h, 875
- rns-double-recint.inl, 876
  - \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL, 876
- rns-double.h, 876
  - ROUND\_DOWN, 877
- rns-double.inl, 877
  - \_\_FFLASFFPACK\_field\_rns\_double\_INL, 877
- rns-integer-mod.h, 877
- rns-integer.h, 878
- rns.h, 879
- rns.inl, 879
  - \_\_FFLASFFPACK\_field\_rns\_INL, 879
- rns\_double, 491
  - \_M, 495
  - \_MMi, 495
  - \_Mi, 495
  - \_basis, 495
  - \_basisMax, 495
  - \_crt\_in, 495
  - \_crt\_out, 496
  - \_field\_rns, 495
  - \_invbasis, 495
  - \_ldm, 496
  - \_mi\_sum, 496
  - \_negbasis, 495
  - \_pbits, 496
  - \_size, 496
  - BasisElement, 492
  - ConstElement\_ptr, 493
  - convert, 494, 495
  - convert\_transpose, 494
  - Element, 492
  - Element\_ptr, 493
  - init, 493, 494
  - init\_transpose, 494
  - integer, 492
  - ModField, 492
  - precompute\_cst, 493
  - reduce, 494
  - rns\_double, 493
- rns\_double\_elt, 496
  - \_alloc, 497
  - \_ptr, 497
  - \_stride, 497
  - ~rns\_double\_elt, 497
  - operator&, 497
  - rns\_double\_elt, 497
- rns\_double\_elt\_cstptr, 497
  - \_alloc, 500
  - \_ptr, 500
  - \_stride, 500
  - operator!=, 499
  - operator<, 499
  - operator+, 499
  - operator++, 499
  - operator+=", 499
  - operator-, 499
  - operator--, 499
  - operator=, 499
  - operator=, 499
  - operator&, 499, 500
  - operator[], 499
  - operator\*, 499
  - other, 500
  - rns\_double\_elt\_cstptr, 498
- rns\_double\_elt\_ptr, 500
  - \_alloc, 502
  - \_ptr, 502
  - \_stride, 502
  - operator!=, 502
  - operator<, 502
  - operator+, 502
  - operator++, 501
  - operator+=", 502
  - operator-, 502
  - operator--, 502
  - operator=, 502
  - operator=, 502
  - operator&, 501, 502
  - operator[], 501
  - operator\*, 501
  - other, 502
  - rns\_double\_elt\_ptr, 501
- rns\_double\_extended, 503
  - \_M, 506
  - \_MMi, 506
  - \_Mi, 506
  - \_basis, 505
  - \_basisMax, 505

- `_crt_in`, 506
- `_crt_out`, 506
- `_field_rns`, 506
- `_invbasis`, 506
- `_ldm`, 506
- `_negbasis`, 505
- `_pbits`, 506
- `_size`, 506
- `BasisElement`, 504
- `ConstElement_ptr`, 504
- `convert`, 505
- `Element`, 504
- `Element_ptr`, 504
- `init`, 504, 505
- `integer`, 503
- `ModField`, 503
- `precompute_cst`, 504
- `reduce`, 505
- `rns_double_extended`, 504
- `RNSElementTag`, 506
- `RNSInteger`
  - `RNSInteger< RNS >`, 508
- `RNSInteger< RNS >`, 507
  - `_rns`, 510
  - `assign`, 509
  - `BasisElement`, 507
  - `cardinality`, 509
  - `characteristic`, 509
  - `ConstElement_ptr`, 508
  - `convert`, 509
  - `Element`, 508
  - `Element_ptr`, 508
  - `init`, 509
  - `integer`, 507
  - `isMOne`, 508
  - `isOne`, 508
  - `isZero`, 508
  - `mOne`, 510
  - `one`, 510
  - `reduce`, 509
  - `rns`, 508
  - `RNSInteger`, 508
  - `size`, 508
  - `write`, 509, 510
  - `zero`, 510
- `RNSInteger< RNS >::RandIter`, 487
  - `operator()`, 488
  - `RandIter`, 488
  - `random`, 488
  - `ring`, 488
- `RNSIntegerMod`
  - `RNSIntegerMod< RNS >`, 512
- `RNSIntegerMod< RNS >`, 510
  - `_F`, 516
  - `_Mi_modp_rns`, 516
  - `_RNSdelayed`, 516
  - `_iM_modp_rns`, 516
  - `_p`, 516
  - `_rns`, 516
  - `add`, 514
  - `areEqual`, 515
  - `assign`, 514
  - `axpyin`, 515
  - `BasisElement`, 512
  - `cardinality`, 513
  - `characteristic`, 513
  - `ConstElement_ptr`, 512
  - `convert`, 514
  - `delayed`, 512
  - `Element`, 511
  - `Element_ptr`, 511
  - `init`, 513, 514
  - `integer`, 512
  - `inv`, 515
  - `isMOne`, 512
  - `isOne`, 512
  - `isZero`, 513
  - `maxElement`, 513
  - `minElement`, 513
  - `ModField`, 512
  - `mOne`, 517
  - `mul`, 514
  - `neg`, 514
  - `one`, 516
  - `reduce`, 513, 514
  - `reduce_modp`, 515, 516
  - `reduce_modp_rnsmajor`, 516
  - `rns`, 512
  - `RNSIntegerMod`, 512
  - `size`, 512
  - `sub`, 514
  - `write`, 515
  - `write_matrix`, 515
  - `write_matrix_long`, 515
  - `zero`, 517
- `RNSIntegerMod< RNS >::RandIter`, 489
  - `operator()`, 489
  - `RandIter`, 489
  - `random`, 489
  - `ring`, 489
- `RNSModulus`
  - `FFLAS::CuttingStrategy`, 188
- `rnsRandIter`
  - `rnsRandIter< RNS >`, 517
- `rnsRandIter< RNS >`, 517
  - `operator()`, 518
  - `random`, 517, 518
  - `ring`, 518
  - `rnsRandIter`, 517
- `round`
  - `ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type >`, 520
  - `ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type >`, 523
  - `Simd128_impl< true, true, false, 2 >`, 537

- Simd128\_impl< true, true, false, 4 >, [545](#)
- Simd128\_impl< true, true, false, 8 >, [554](#)
- Simd128\_impl< true, true, true, 2 >, [562](#)
- Simd128\_impl< true, true, true, 4 >, [570](#)
- Simd128\_impl< true, true, true, 8 >, [578](#)
- Simd256\_impl< true, false, true, 8 >, [587](#)
- Simd256\_impl< true, true, false, 2 >, [596](#)
- Simd256\_impl< true, true, false, 4 >, [611](#)
- Simd256\_impl< true, true, false, 8 >, [620](#)
- Simd256\_impl< true, true, true, 2 >, [628](#)
- Simd256\_impl< true, true, true, 4 >, [637](#), [642](#)
- Simd256\_impl< true, true, true, 8 >, [650](#)
- Simd512\_impl< true, false, true, 8 >, [658](#)
- Simd512\_impl< true, true, false, 8 >, [667](#)
- Simd512\_impl< true, true, true, 8 >, [675](#)
- ROUND\_DOWN
  - fflas\_sparse.h, [817](#)
  - rns-double.h, [877](#)
- Row, [518](#)
- row
  - Coo< Field >, [401](#)
  - Coo< ValT, IdxT >, [400](#), [403](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [680](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [682](#)
- rowblockindex
  - ForStrategy2D< blocksize\_t, Cut, Param >, [431](#)
- rowBlockSize
  - ForStrategy2D< blocksize\_t, Cut, Param >, [432](#)
- rowdim
  - StatsMatrix, [702](#)
- RowEchelonForm
  - FFPACK, [302](#), [303](#), [346](#)
- RowEchelonForm\_modular\_double
  - ffpack.C, [921](#)
  - ffpack\_c.h, [943](#)
- RowEchelonForm\_modular\_float
  - ffpack.C, [922](#)
  - ffpack\_c.h, [943](#)
- RowEchelonForm\_modular\_int32\_t
  - ffpack.C, [923](#)
  - ffpack\_c.h, [944](#)
- rownumblocks
  - ForStrategy2D< blocksize\_t, Cut, Param >, [431](#)
- RowRankProfile
  - FFPACK, [315](#), [316](#), [351](#)
- RowRankProfile\_modular\_double
  - ffpack.C, [929](#)
  - ffpack\_c.h, [949](#)
- RowRankProfileSubmatrix
  - FFPACK, [320](#), [352](#)
- RowRankProfileSubmatrix\_modular\_double
  - ffpack.C, [930](#)
  - ffpack\_c.h, [950](#)
- RowRankProfileSubmatrixIndices
  - FFPACK, [318](#), [352](#)
- RowRankProfileSubmatrixIndices\_modular\_double
  - ffpack.C, [930](#)
  - ffpack\_c.h, [949](#)
- ruint< K >, [518](#)
- run\_with\_field
  - benchmark-charpoly.C, [716](#)
  - benchmark-fdot.C, [723](#)
  - test-charpoly.C, [986](#)
  - test-echelon.C, [990](#)
  - test-fdot.C, [993](#)
  - test-fgemm-check.C, [994](#)
  - test-fgemm.C, [996](#)
  - test-fgemv.C, [998](#)
  - test-fger.C, [1000](#)
  - test-fgesv.C, [1001](#)
  - test-finit.C, [1002](#)
  - test-fsyr2k.C, [1005](#)
  - test-fsyrk.C, [1006](#)
  - test-fsytrf.C, [1008](#)
  - test-ftrmm.C, [1009](#)
  - test-ftrmv.C, [1010](#)
  - test-ftrsm.C, [1012](#)
  - test-ftrssyr2k.C, [1013](#)
  - test-ftrstr.C, [1014](#)
  - test-ftrsv.C, [1015](#)
  - test-ftrtri.C, [1016](#)
  - test-io.C, [1018](#)
  - test-lu.C, [1021](#)
  - test-minpoly.C, [1023](#)
  - test-nullspace.C, [1025](#)
  - test-rankprofiles.C, [1028](#)
  - test-solve.C, [1033](#)
- run\_with\_Integer
  - test-fdot.C, [993](#)
- saxpy\_
  - config-blas.h, [750](#)
- ScalAndReduce
  - FFLAS::Protected, [204](#)
- scalar\_t
  - FieldSimd< \_Field >, [414](#)
  - NoSimd< T >, [486](#)
  - Simd128\_impl< true, true, false, 2 >, [531](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [565](#)
  - Simd128\_impl< true, true, true, 8 >, [573](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [589](#)
  - Simd256\_impl< true, true, false, 4 >, [600](#)
  - Simd256\_impl< true, true, false, 8 >, [614](#)
  - Simd256\_impl< true, true, true, 2 >, [623](#)
  - Simd256\_impl< true, true, true, 4 >, [632](#)
  - Simd256\_impl< true, true, true, 8 >, [645](#)
  - Simd512\_impl< true, false, true, 8 >, [654](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#)
  - Simd512\_impl< true, true, true, 8 >, [670](#)
- ScalFunctions< Element, Enable >, [518](#)
- ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [518](#)

- add, [520](#)
- addin, [520](#)
- ceil, [519](#)
- div, [520](#)
- eq, [522](#)
- floor, [520](#)
- fmadd, [521](#)
- fmaddin, [521](#)
- fmsub, [521](#)
- fmsubin, [521](#)
- fnmadd, [521](#)
- fnmaddin, [521](#)
- greater, [522](#)
- greater\_eq, [522](#)
- lesser, [521](#)
- lesser\_eq, [522](#)
- mul, [520](#)
- mulin, [520](#)
- round, [520](#)
- sub, [520](#)
- subin, [520](#)
- vand, [519](#)
- vandnot, [519](#)
- vor, [519](#)
- vxor, [519](#)
- zero, [519](#)
- ScalFunctions< Element, typename enable\_if<  
is\_integral< Element >::value >::type >,  
[522](#)  
add, [524](#)  
addin, [524](#)  
eq, [527](#)  
fmadd, [525](#)  
fmaddin, [525](#)  
fmaddx, [525](#)  
fmaddxin, [525](#)  
fmsub, [525](#)  
fmsubin, [525](#)  
fmsubx, [525](#)  
fmsubxin, [525](#)  
fnmadd, [526](#)  
fnmaddin, [526](#)  
fnmaddx, [526](#)  
fnmaddxin, [526](#)  
greater, [527](#)  
greater\_eq, [527](#)  
lesser, [527](#)  
lesser\_eq, [527](#)  
mul, [524](#)  
mulhi, [524](#)  
mullo, [524](#)  
mulx, [524](#)  
round, [523](#)  
sll, [526](#)  
sra, [526](#)  
srl, [526](#)  
sub, [524](#)  
subin, [524](#)  
vand, [523](#)  
vandnot, [523](#)  
vor, [523](#)  
vxor, [523](#)  
zero, [523](#)  
scalp  
FFLAS::vectorised, [270](#)  
FFLAS::vectorised::unswitch, [271](#)  
schedule\_bini.inl, [775](#)  
\_\_FFLASFFPACK\_fgemm\_bini\_INL, [776](#)  
schedule\_winograd.inl, [776](#)  
\_\_FFLASFFPACK\_fgemm\_winograd\_INL, [776](#)  
schedule\_winograd\_acc.inl, [776](#)  
\_\_FFLASFFPACK\_fgemm\_winograd\_acc\_INL,  
[777](#)  
schedule\_winograd\_acc\_ip.inl, [777](#)  
\_\_FFLASFFPACK\_fgemm\_winograd\_acc\_ip\_INL,  
[778](#)  
schedule\_winograd\_ip.inl, [778](#)  
\_\_FFLASFFPACK\_fgemm\_winograd\_ip\_INL, [778](#)  
scopy\_  
config-blas.h, [752](#)  
sdot\_  
config-blas.h, [750](#)  
second\_component  
Compose< H1, H2 >, [393](#)  
Self  
Coo< ValT, IdxT >, [399](#), [402](#)  
Self\_t  
MMHelper< FFPACK::RNSInteger< E >, Algo-  
Trait, ModeCategories::DefaultTag, ParSeq-  
Trait >, [469](#)  
MMHelper< FFPACK::RNSIntegerMod< E >, Al-  
goTrait, ModeCategories::DefaultTag, ParSeq-  
Trait >, [471](#)  
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
Dest >, ParSeqTrait >, [473](#)  
MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
ElementCategories::RNSElementTag >,  
ParSeqTrait >, [475](#)  
MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag,  
ParSeqTrait >, [477](#)  
MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
Trait >, [465](#)  
Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [695](#)  
SELL  
FFLAS, [74](#)  
sell.h, [841](#)  
sell\_pspmv.inl, [841](#)  
\_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL,  
[842](#)  
sell\_spmv.inl, [842](#)  
\_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL,  
[843](#)  
sell\_utils.inl, [843](#)  
\_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL,  
[843](#)  
SELL\_ZO

- FFLAS, [74](#)
- Sequential, [527](#)
  - numthreads, [528](#)
  - operator<<, [528](#)
  - Sequential, [528](#)
- set
  - Simd128\_impl< true, true, false, 2 >, [531](#), [534](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#), [542](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#), [551](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [573](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [590](#), [592](#)
  - Simd256\_impl< true, true, false, 4 >, [600](#), [603](#), [605](#), [606](#)
  - Simd256\_impl< true, true, false, 8 >, [615](#), [617](#)
  - Simd256\_impl< true, true, true, 2 >, [624](#)
  - Simd256\_impl< true, true, true, 4 >, [633](#), [638](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [654](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#), [664](#)
  - Simd512\_impl< true, true, true, 8 >, [671](#)
- set1
  - Simd128\_impl< true, true, false, 2 >, [531](#), [534](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#), [542](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#), [551](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [573](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [590](#), [592](#)
  - Simd256\_impl< true, true, false, 4 >, [600](#), [603](#), [605](#)
  - Simd256\_impl< true, true, false, 8 >, [615](#), [617](#)
  - Simd256\_impl< true, true, true, 2 >, [624](#)
  - Simd256\_impl< true, true, true, 4 >, [633](#), [638](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [654](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#), [663](#)
  - Simd512\_impl< true, true, true, 8 >, [671](#)
- set\_numthreads
  - Parallel< C, P >, [487](#)
- setErrorStream
  - Failure, [412](#)
- setNorm
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [470](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [472](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [475](#)
- setOutBounds
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [466](#)
- sgemm\_
  - config-blas.h, [753](#)
- sgemv\_
  - config-blas.h, [751](#)
- sger\_
  - config-blas.h, [751](#)
- shuffle
  - Simd128\_impl< true, true, false, 2 >, [535](#)
  - Simd128\_impl< true, true, false, 4 >, [543](#)
  - Simd128\_impl< true, true, false, 8 >, [552](#)
  - Simd128\_impl< true, true, true, 2 >, [559](#)
  - Simd128\_impl< true, true, true, 4 >, [567](#)
  - Simd128\_impl< true, true, true, 8 >, [574](#)
  - Simd256\_impl< true, true, false, 2 >, [594](#)
  - Simd256\_impl< true, true, false, 4 >, [607](#)
  - Simd256\_impl< true, true, false, 8 >, [618](#)
  - Simd256\_impl< true, true, true, 2 >, [625](#)
  - Simd256\_impl< true, true, true, 4 >, [634](#), [639](#)
  - Simd256\_impl< true, true, true, 8 >, [647](#)
  - Simd512\_impl< true, false, true, 8 >, [655](#)
  - Simd512\_impl< true, true, false, 8 >, [665](#)
  - Simd512\_impl< true, true, true, 8 >, [672](#)
- shuffle\_twice
  - Simd256\_impl< true, true, false, 4 >, [607](#)
  - Simd256\_impl< true, true, true, 4 >, [634](#), [639](#)
- sigma
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [697](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [699](#)
- signbits
  - Simd128\_impl< true, true, false, 8 >, [555](#)
  - Simd128\_impl< true, true, true, 8 >, [578](#)
  - Simd256\_impl< true, true, false, 8 >, [621](#)
  - Simd256\_impl< true, true, true, 8 >, [651](#)
  - Simd512\_impl< true, true, false, 8 >, [667](#)
  - Simd512\_impl< true, true, true, 8 >, [676](#)
- Simd
  - fflas\_simd.h, [807](#)
- simd
  - FieldSimd< \_Field >, [414](#)
- SIMD wrapper, [42](#)
- simd.doxy, [807](#)
- Simd128
  - simd128.inl, [807](#)
- simd128.inl, [807](#)
  - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL, [807](#)
  - Simd128, [807](#)
- simd128\_double.inl, [807](#)
  - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL, [807](#)
- simd128\_float.inl, [808](#)
  - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL, [808](#)
- Simd128\_impl< ArithType, Int, Signed, Size >, [528](#)
- Simd128\_impl< true, false, true, 4 >, [528](#)
  - type\_string, [529](#)
- Simd128\_impl< true, false, true, 8 >, [529](#)
  - type\_string, [529](#)



Simd128\_impl< true, true, false, 2 >, 529  
   add, 535  
   addin, 535  
   alignment, 538  
   blend, 535  
   compliant, 534  
   eq, 536  
   fmadd, 536  
   fmaddin, 536  
   fmaddx, 533  
   fmaddxin, 533  
   fmsub, 536  
   fmsubin, 536  
   fmsubx, 533  
   fmsubxin, 533  
   fnmadd, 536  
   fnmaddin, 536  
   fnmaddx, 533  
   fnmaddxin, 533  
   gather, 531, 534  
   greater, 532  
   greater\_eq, 532  
   hadd\_to\_scal, 533  
   lesser, 532  
   lesser\_eq, 532  
   load, 531, 534  
   loadu, 532, 534  
   mod, 537  
   mul, 536  
   mulhi, 532  
   mullo, 536  
   mulx, 533  
   round, 537  
   scalar\_t, 531  
   set, 531, 534  
   set1, 531, 534  
   shuffle, 535  
   sll, 535  
   sll128, 537  
   sra, 532  
   srl, 535  
   srl128, 537  
   store, 532, 534  
   storeu, 532, 534  
   stream, 532, 534  
   sub, 535  
   subin, 535  
   type\_string, 537  
   unpackhi, 535  
   unpacklo, 535  
   valid, 533  
   vand, 537  
   vandnot, 537  
   vect\_size, 538  
   vect\_t, 531  
   vor, 537  
   vxor, 537  
   zero, 537

Simd128\_impl< true, true, false, 2 >::Converter, 393  
   t, 394  
   v, 394  
 Simd128\_impl< true, true, false, 4 >, 538  
   add, 544  
   addin, 544  
   alignment, 546  
   blend, 544  
   compliant, 542  
   eq, 545  
   fmadd, 544  
   fmaddin, 544  
   fmaddx, 541  
   fmaddxin, 541  
   fmsub, 545  
   fmsubin, 545  
   fmsubx, 542  
   fmsubxin, 542  
   fnmadd, 545  
   fnmaddin, 545  
   fnmaddx, 541  
   fnmaddxin, 542  
   gather, 540, 542  
   greater, 541  
   greater\_eq, 541  
   hadd\_to\_scal, 542  
   lesser, 541  
   lesser\_eq, 541  
   load, 540, 543  
   loadu, 540, 543  
   mod, 545  
   mul, 544  
   mulhi, 541  
   mullo, 544  
   mulx, 541  
   round, 545  
   scalar\_t, 540  
   set, 540, 542  
   set1, 540, 542  
   shuffle, 543  
   sll, 543  
   sll128, 546  
   sra, 541  
   srl, 543  
   srl128, 546  
   store, 540, 543  
   storeu, 540, 543  
   stream, 540, 543  
   sub, 544  
   subin, 544  
   type\_string, 545  
   unpackhi, 543  
   unpacklo, 543  
   valid, 542  
   vand, 546  
   vandnot, 546  
   vect\_size, 546  
   vect\_t, 540



- vor, [546](#)
- vxor, [546](#)
- zero, [546](#)
- Simd128\_impl< true, true, false, 4 >::Converter, [394](#)
  - t, [394](#)
  - v, [394](#)
- Simd128\_impl< true, true, false, 8 >, [547](#)
  - add, [553](#)
  - addin, [553](#)
  - alignment, [556](#)
  - blend, [553](#)
  - compliant, [551](#)
  - eq, [554](#)
  - fmadd, [553](#)
  - fmaddin, [553](#)
  - fmaddx, [550](#)
  - fmaddxin, [550](#)
  - fmsub, [554](#)
  - fmsubin, [554](#)
  - fmsubx, [551](#)
  - fmsubxin, [551](#), [554](#)
  - fnmadd, [554](#)
  - fnmaddin, [554](#)
  - fnmaddx, [550](#)
  - fnmaddxin, [551](#)
  - gather, [549](#), [551](#)
  - get, [552](#)
  - greater, [550](#)
  - greater\_eq, [550](#)
  - hadd\_to\_scal, [551](#)
  - lesser, [550](#)
  - lesser\_eq, [550](#)
  - load, [549](#), [552](#)
  - loadu, [549](#), [552](#)
  - mask\_high, [554](#)
  - mod, [555](#)
  - mul, [553](#)
  - mulhi\_fast, [554](#)
  - mullo, [550](#)
  - mulx, [550](#)
  - round, [554](#)
  - scalar\_t, [549](#)
  - set, [549](#), [551](#)
  - set1, [549](#), [551](#)
  - shuffle, [552](#)
  - signbits, [555](#)
  - sll, [552](#)
  - sll128, [555](#)
  - sra, [549](#)
  - srl, [552](#)
  - srl128, [555](#)
  - store, [549](#), [552](#)
  - storeu, [549](#), [552](#)
  - stream, [549](#), [552](#)
  - sub, [553](#)
  - subin, [553](#)
  - type\_string, [555](#)
  - unpackhi, [553](#)
  - unpacklo, [552](#)
  - valid, [551](#)
  - vand, [555](#)
  - vandnot, [555](#)
  - vect\_size, [556](#)
  - vect\_t, [549](#)
  - vor, [555](#)
  - vxor, [555](#)
  - zero, [555](#)
  - Simd128\_impl< true, true, false, 8 >::Converter, [394](#)
    - t, [394](#)
    - v, [394](#)
  - Simd128\_impl< true, true, true, 2 >, [556](#)
    - add, [559](#)
    - addin, [560](#)
    - alignment, [563](#)
    - blend, [559](#)
    - compliant, [558](#)
    - eq, [562](#)
    - fmadd, [560](#)
    - fmaddin, [560](#)
    - fmaddx, [560](#)
    - fmaddxin, [561](#)
    - fmsub, [561](#)
    - fmsubin, [561](#)
    - fmsubx, [561](#)
    - fmsubxin, [562](#)
    - fnmadd, [561](#)
    - fnmaddin, [561](#)
    - fnmaddx, [561](#)
    - fnmaddxin, [561](#)
    - gather, [558](#)
    - greater, [562](#)
    - greater\_eq, [562](#)
    - hadd\_to\_scal, [562](#)
    - lesser, [562](#)
    - lesser\_eq, [562](#)
    - load, [558](#)
    - loadu, [558](#)
    - mod, [562](#)
    - mul, [560](#)
    - mulhi, [560](#)
    - mullo, [560](#)
    - mulx, [560](#)
    - round, [562](#)
    - scalar\_t, [558](#)
    - set, [558](#)
    - set1, [558](#)
    - shuffle, [559](#)
    - sll, [559](#)
    - sll128, [563](#)
    - sra, [559](#)
    - srl, [559](#)
    - srl128, [563](#)
    - store, [558](#)
    - storeu, [559](#)
    - stream, [559](#)
    - sub, [560](#)

- subin, [560](#)
- type\_string, [563](#)
- unpackhi, [559](#)
- unpacklo, [559](#)
- valid, [558](#)
- vand, [563](#)
- vandnot, [563](#)
- vect\_size, [563](#)
- vect\_t, [558](#)
- vor, [563](#)
- vxor, [563](#)
- zero, [563](#)
- Simd128\_impl< true, true, true, 2 >::Converter, [394](#)
- t, [395](#)
- v, [395](#)
- Simd128\_impl< true, true, true, 4 >, [564](#)
- add, [567](#)
- addin, [567](#)
- alignment, [571](#)
- blend, [567](#)
- compliant, [565](#)
- eq, [569](#)
- fmadd, [568](#)
- fmaddin, [568](#)
- fmaddx, [568](#)
- fmaddxin, [568](#)
- fmsub, [569](#)
- fmsubin, [569](#)
- fmsubx, [569](#)
- fmsubxin, [569](#)
- fnmadd, [568](#)
- fnmaddin, [568](#)
- fnmaddx, [569](#)
- fnmaddxin, [569](#)
- gather, [566](#)
- greater, [569](#)
- greater\_eq, [570](#)
- hadd\_to\_scal, [570](#)
- lesser, [569](#)
- lesser\_eq, [570](#)
- load, [566](#)
- loadu, [566](#)
- mod, [570](#)
- mul, [568](#)
- mulhi, [568](#)
- mullo, [567](#)
- mulx, [568](#)
- round, [570](#)
- scalar\_t, [565](#)
- set, [566](#)
- set1, [566](#)
- shuffle, [567](#)
- sll, [566](#)
- sll128, [570](#)
- sra, [567](#)
- srl, [566](#)
- srl128, [570](#)
- store, [566](#)
- storeu, [566](#)
- stream, [566](#)
- sub, [567](#)
- subin, [567](#)
- type\_string, [570](#)
- unpackhi, [567](#)
- unpacklo, [567](#)
- valid, [565](#)
- vand, [570](#)
- vandnot, [571](#)
- vect\_size, [571](#)
- vect\_t, [565](#)
- vor, [571](#)
- vxor, [571](#)
- zero, [570](#)
- Simd128\_impl< true, true, true, 4 >::Converter, [395](#)
- t, [395](#)
- v, [395](#)
- Simd128\_impl< true, true, true, 8 >, [571](#)
- add, [575](#)
- addin, [575](#)
- alignment, [579](#)
- blend, [575](#)
- compliant, [573](#)
- eq, [577](#)
- fmadd, [576](#)
- fmaddin, [576](#)
- fmaddx, [576](#)
- fmaddxin, [576](#)
- fmsub, [577](#)
- fmsubin, [577](#)
- fmsubx, [577](#)
- fmsubxin, [577](#)
- fnmadd, [576](#)
- fnmaddin, [576](#)
- fnmaddx, [576](#)
- fnmaddxin, [576](#)
- gather, [573](#)
- get, [574](#)
- greater, [577](#)
- greater\_eq, [577](#)
- hadd\_to\_scal, [577](#)
- lesser, [577](#)
- lesser\_eq, [577](#)
- load, [574](#)
- loadu, [574](#)
- mask\_high, [578](#)
- mod, [578](#)
- mul, [575](#)
- mulhi\_fast, [578](#)
- mullo, [575](#)
- mulx, [575](#)
- round, [578](#)
- scalar\_t, [573](#)
- set, [573](#)
- set1, [573](#)
- shuffle, [574](#)
- signbits, [578](#)

- sll, [574](#)
- sll128, [578](#)
- sra, [574](#)
- srl, [574](#)
- srl128, [578](#)
- store, [574](#)
- storeu, [574](#)
- stream, [574](#)
- sub, [575](#)
- subin, [575](#)
- type\_string, [578](#)
- unpackhi, [575](#)
- unpacklo, [575](#)
- valid, [573](#)
- vand, [578](#)
- vandnot, [579](#)
- vect\_size, [579](#)
- vect\_t, [573](#)
- vor, [579](#)
- vxor, [579](#)
- zero, [578](#)
- Simd128\_impl< true, true, true, 8 >::Converter, [395](#)
- t, [395](#)
- v, [395](#)
- simd128\_int16.inl, [808](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL, [808](#)
- simd128\_int32.inl, [808](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL, [808](#)
- simd128\_int64.inl, [809](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL, [809](#)
- vect\_t, [809](#)
- Simd128fp\_base, [579](#)
- type\_string, [579](#)
- Simd128i\_base, [580](#)
- sll128, [580](#)
- srl128, [580](#)
- type\_string, [580](#)
- vand, [580](#)
- vandnot, [581](#)
- vect\_t, [580](#)
- vor, [581](#)
- vxor, [581](#)
- zero, [580](#)
- Simd256
- simd256.inl, [809](#)
- simd256.inl, [809](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL, [809](#)
- Simd256, [809](#)
- simd256\_double.inl, [810](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL, [810](#)
- simd256\_float.inl, [810](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL, [810](#)
- Simd256\_impl< ArithType, Int, Signed, Size >, [581](#)
- Simd256\_impl< true, false, true, 4 >, [581](#)
- Simd256\_impl< true, false, true, 8 >, [581](#)
- add, [584](#)
- addin, [584](#)
- alignment, [587](#)
- blend, [584](#)
- blendv, [584](#)
- ceil, [587](#)
- compliant, [583](#)
- div, [585](#)
- eq, [586](#)
- floor, [587](#)
- fmadd, [585](#)
- fmaddin, [585](#)
- fmsub, [585](#)
- fmsubin, [586](#)
- fnmadd, [585](#)
- fnmaddin, [585](#)
- gather, [583](#)
- greater, [586](#)
- greater\_eq, [586](#)
- hadd, [587](#)
- hadd\_to\_scal, [587](#)
- lesser, [586](#)
- lesser\_eq, [586](#)
- load, [583](#)
- loadu, [583](#)
- mod, [587](#)
- mul, [585](#)
- mulin, [585](#)
- round, [587](#)
- scalar\_t, [583](#)
- set, [583](#)
- set1, [583](#)
- store, [584](#)
- storeu, [584](#)
- stream, [584](#)
- sub, [584](#)
- subin, [585](#)
- unpackhi\_twice, [584](#)
- unpacklo\_twice, [584](#)
- valid, [583](#)
- vand, [586](#)
- vandnot, [586](#)
- vect\_size, [587](#)
- vect\_t, [583](#)
- vor, [586](#)
- vxor, [586](#)
- zero, [583](#)
- Simd256\_impl< true, true, false, 2 >, [588](#)
- add, [594](#)
- addin, [594](#)
- alignment, [596](#)
- blend\_twice, [594](#)
- compliant, [592](#)
- eq, [596](#)
- fmadd, [595](#)

- fmaddin, 595
- fmaddx, 591
- fmaddxin, 591
- fmsub, 595
- fmsubin, 596
- fmsubx, 592
- fmsubxin, 592
- fnmadd, 595
- fnmaddin, 595
- fnmaddx, 592
- fnmaddxin, 592
- gather, 590, 593
- greater, 591
- greater\_eq, 591
- hadd\_to\_scal, 592
- half\_t, 590
- lesser, 591
- lesser\_eq, 591
- load, 590, 593
- loadu, 590, 593
- mod, 596
- mul, 595
- mulhi, 591
- mullo, 595
- mulx, 591
- round, 596
- scalar\_t, 589
- set, 590, 592
- set1, 590, 592
- shuffle, 594
- simdHalf, 589
- sll, 593
- sra, 591
- srl, 593
- store, 590, 593
- storeu, 590, 593
- stream, 591, 593
- sub, 595
- subin, 595
- type\_string, 596
- unpackhi, 594
- unpackhi\_twice, 594
- unpacklo, 594
- unpacklo\_twice, 594
- unpacklohi, 594
- valid, 592
- vect\_size, 596
- vect\_t, 590
- zero, 596
- Simd256\_impl< true, true, false, 2 >::Converter, 395
  - t, 396
  - v, 396
- Simd256\_impl< true, true, false, 4 >, 597
  - add, 608
  - addin, 608
  - alignment, 612
  - blend, 608
  - compliant, 605
  - eq, 611
  - fmadd, 609
  - fmaddin, 609, 610
  - fmaddx, 602, 604
  - fmaddxin, 602, 604
  - fmsub, 610
  - fmsubin, 610, 611
  - fmsubx, 602, 605
  - fmsubxin, 602, 605
  - fnmadd, 610
  - fnmaddin, 610
  - fnmaddx, 602, 604
  - fnmaddxin, 602, 605
  - gather, 600, 603, 606
  - greater, 601, 604
  - greater\_eq, 601, 604
  - hadd\_to\_scal, 602, 605
  - half\_t, 600
  - lesser, 601, 604
  - lesser\_eq, 601, 604
  - load, 601, 603, 606
  - loadu, 601, 603, 606
  - mod, 611
  - mul, 609
  - mulhi, 602, 604
  - mullo, 609
  - mulx, 602, 604
  - round, 611
  - scalar\_t, 600
  - set, 600, 603, 605, 606
  - set1, 600, 603, 605
  - shuffle, 607
  - shuffle\_twice, 607
  - simdHalf, 600
  - sll, 607
  - sra, 601, 603
  - srl, 607
  - store, 601, 603, 606
  - storeu, 601, 603, 606
  - stream, 601, 603, 606
  - sub, 608, 609
  - subin, 609
  - type\_string, 611
  - unpackhi, 608
  - unpackhi\_twice, 607
  - unpacklo, 608
  - unpacklo\_twice, 607
  - unpacklohi, 608
  - valid, 605
  - vand, 612
  - vandnot, 612
  - vect\_size, 612
  - vect\_t, 600
  - vor, 612
  - vxor, 612
  - zero, 612
- Simd256\_impl< true, true, false, 4 >::Converter, 396
  - t, 396

- v, [396](#)
- Simd256\_impl< true, true, false, 8 >, [612](#)
  - add, [619](#)
  - addin, [619](#)
  - alignment, [621](#)
  - blend, [619](#)
  - compliant, [617](#)
  - eq, [620](#)
  - fmadd, [620](#)
  - fmaddin, [620](#)
  - fmaddx, [616](#)
  - fmaddxin, [616](#)
  - fmsub, [620](#)
  - fmsubin, [620](#)
  - fmsubx, [617](#)
  - fmsubxin, [617](#)
  - fnmadd, [620](#)
  - fnmaddin, [620](#)
  - fnmaddx, [616](#)
  - fnmaddxin, [617](#)
  - gather, [615](#), [617](#)
  - get, [618](#)
  - greater, [616](#)
  - greater\_eq, [616](#)
  - hadd\_to\_scal, [617](#)
  - half\_t, [615](#)
  - lesser, [616](#)
  - lesser\_eq, [616](#)
  - load, [615](#), [618](#)
  - loadu, [615](#), [618](#)
  - mask\_high, [621](#)
  - mod, [621](#)
  - mul, [620](#)
  - mulhi\_fast, [621](#)
  - mullo, [616](#)
  - mulx, [616](#)
  - round, [620](#)
  - scalar\_t, [614](#)
  - set, [615](#), [617](#)
  - set1, [615](#), [617](#)
  - shuffle, [618](#)
  - signbits, [621](#)
  - simdHalf, [614](#)
  - sll, [618](#)
  - sra, [615](#)
  - srl, [618](#)
  - store, [615](#), [618](#)
  - storeu, [615](#), [618](#)
  - stream, [615](#), [618](#)
  - sub, [619](#)
  - subin, [619](#)
  - type\_string, [621](#)
  - unpackhi, [619](#)
  - unpackhi\_twice, [619](#)
  - unpacklo, [619](#)
  - unpacklo\_twice, [618](#)
  - unpacklohi, [619](#)
  - valid, [617](#)
  - vect\_size, [621](#)
  - vect\_t, [614](#)
  - zero, [621](#)
- Simd256\_impl< true, true, false, 8 >::Converter, [396](#)
  - t, [396](#)
  - v, [396](#)
- Simd256\_impl< true, true, true, 2 >, [621](#)
  - add, [626](#)
  - addin, [626](#)
  - alignment, [629](#)
  - blend\_twice, [625](#)
  - compliant, [623](#)
  - eq, [628](#)
  - fmadd, [626](#)
  - fmaddin, [626](#)
  - fmaddx, [627](#)
  - fmaddxin, [627](#)
  - fmsub, [627](#)
  - fmsubin, [627](#)
  - fmsubx, [628](#)
  - fmsubxin, [628](#)
  - fnmadd, [627](#)
  - fnmaddin, [627](#)
  - fnmaddx, [627](#)
  - fnmaddxin, [627](#)
  - gather, [624](#)
  - greater, [628](#)
  - greater\_eq, [628](#)
  - hadd\_to\_scal, [628](#)
  - half\_t, [623](#)
  - lesser, [628](#)
  - lesser\_eq, [628](#)
  - load, [624](#)
  - loadu, [624](#)
  - mod, [628](#)
  - mul, [626](#)
  - mulhi, [626](#)
  - mullo, [626](#)
  - mulx, [626](#)
  - round, [628](#)
  - scalar\_t, [623](#)
  - set, [624](#)
  - set1, [624](#)
  - shuffle, [625](#)
  - simdHalf, [623](#)
  - sll, [625](#)
  - sra, [625](#)
  - srl, [625](#)
  - store, [624](#)
  - storeu, [624](#)
  - stream, [624](#)
  - sub, [626](#)
  - subin, [626](#)
  - type\_string, [629](#)
  - unpackhi, [625](#)
  - unpackhi\_twice, [625](#)
  - unpacklo, [625](#)
  - unpacklo\_twice, [625](#)

- unpacklohi, 625
- valid, 623
- vect\_size, 629
- vect\_t, 623
- zero, 629
- Simd256\_impl< true, true, true, 2 >::Converter, 396
  - t, 397
  - v, 397
- Simd256\_impl< true, true, true, 4 >, 629
  - add, 635, 639
  - addin, 635, 639
  - alignment, 643
  - blend, 634
  - compliant, 632, 638
  - eq, 637, 642
  - fmadd, 635, 640
  - fmaddin, 635, 640
  - fmaddx, 636, 640
  - fmaddxin, 636, 641
  - fmsub, 636, 641
  - fmsubin, 636, 641
  - fmsubx, 637, 641
  - fmsubxin, 637, 642
  - fnmadd, 636, 641
  - fnmaddin, 636, 641
  - fnmaddx, 636, 641
  - fnmaddxin, 636, 641
  - gather, 633, 638
  - greater, 637, 642
  - greater\_eq, 637, 642
  - hadd\_to\_scal, 637, 642
  - half\_t, 632
  - lesser, 637, 642
  - lesser\_eq, 637, 642
  - load, 633, 638
  - loadu, 633, 638
  - mod, 637, 642
  - mul, 635, 640
  - mulhi, 635, 640
  - mullo, 635, 640
  - mulx, 635, 640
  - round, 637, 642
  - scalar\_t, 632
  - set, 633, 638
  - set1, 633, 638
  - shuffle, 634, 639
  - shuffle\_twice, 634, 639
  - simdHalf, 632
  - sll, 633, 639
  - sra, 634, 639
  - srl, 634, 639
  - store, 633, 639
  - storeu, 633, 639
  - stream, 633, 639
  - sub, 635, 640
  - subin, 635, 640
  - type\_string, 643
  - unpackhi, 634
  - unpackhi\_twice, 634
  - unpacklo, 634
  - unpacklo\_twice, 634
  - unpacklohi, 634
  - valid, 632, 638
  - vand, 643
  - vandnot, 643
  - vect\_size, 643
  - vect\_t, 632
  - vor, 643
  - vxor, 643
  - zero, 643
- Simd256\_impl< true, true, true, 4 >::Converter, 397
  - t, 397
  - v, 397
- Simd256\_impl< true, true, true, 8 >, 644
  - add, 648
  - addin, 648
  - alignment, 651
  - blend, 647
  - compliant, 646
  - eq, 650
  - fmadd, 648
  - fmaddin, 648
  - fmaddx, 649
  - fmaddxin, 649
  - fmsub, 649
  - fmsubin, 649
  - fmsubx, 650
  - fmsubxin, 650
  - fnmadd, 649
  - fnmaddin, 649
  - fnmaddx, 649
  - fnmaddxin, 649
  - gather, 646
  - get, 646
  - greater, 650
  - greater\_eq, 650
  - hadd\_to\_scal, 650
  - half\_t, 645
  - lesser, 650
  - lesser\_eq, 650
  - load, 646
  - loadu, 646
  - mask\_high, 650
  - mod, 651
  - mul, 648
  - mulhi\_fast, 650
  - mullo, 648
  - mulx, 648
  - round, 650
  - scalar\_t, 645
  - set, 646
  - set1, 646
  - shuffle, 647
  - signbits, 651
  - simdHalf, 645
  - sll, 647

- sra, [647](#)
- srl, [647](#)
- store, [646](#)
- storeu, [646](#)
- stream, [646](#)
- sub, [648](#)
- subin, [648](#)
- type\_string, [651](#)
- unpackhi, [647](#)
- unpackhi\_twice, [647](#)
- unpacklo, [647](#)
- unpacklo\_twice, [647](#)
- unpacklohi, [647](#)
- valid, [646](#)
- vect\_size, [651](#)
- vect\_t, [645](#)
- zero, [651](#)
- Simd256\_impl< true, true, true, 8 >::Converter, [397](#)
- t, [397](#)
- v, [397](#)
- simd256\_int16.inl, [810](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL, [810](#)
- simd256\_int32.inl, [810](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL, [811](#)
- simd256\_int64.inl, [811](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL, [811](#)
- vect\_t, [811](#)
- Simd256fp\_base, [651](#)
- Simd256i\_base, [652](#)
- type\_string, [652](#)
- vect\_t, [652](#)
- zero, [652](#)
- Simd512
- simd512.inl, [812](#)
- simd512.inl, [811](#)
- \_\_FFLASFFPACK\_simd512\_INL, [812](#)
- Simd512, [812](#)
- simd512\_double.inl, [812](#)
- \_\_FFLASFFPACK\_simd512\_double\_INL, [812](#)
- simd512\_float.inl, [812](#)
- \_\_FFLASFFPACK\_simd512\_float\_INL, [812](#)
- Simd512\_impl< ArithType, Int, Signed, Size >, [652](#)
- Simd512\_impl< true, false, true, 4 >, [652](#)
- type\_string, [653](#)
- Simd512\_impl< true, false, true, 8 >, [653](#)
- add, [656](#)
- addin, [656](#)
- alignment, [658](#)
- blend, [656](#)
- blendv, [656](#)
- ceil, [658](#)
- compliant, [654](#)
- div, [656](#)
- eq, [657](#)
- floor, [658](#)
- fmadd, [656](#)
- fmaddin, [657](#)
- fmsub, [657](#)
- fmsubin, [657](#)
- fnmadd, [657](#)
- fnmaddin, [657](#)
- gather, [655](#)
- greater, [657](#)
- greater\_eq, [658](#)
- hadd, [658](#)
- hadd\_to\_scal, [658](#)
- lesser, [657](#)
- lesser\_eq, [657](#)
- load, [655](#)
- loadu, [655](#)
- mul, [656](#)
- mulin, [656](#)
- round, [658](#)
- scalar\_t, [654](#)
- set, [654](#)
- set1, [654](#)
- shuffle, [655](#)
- store, [655](#)
- storeu, [655](#)
- stream, [655](#)
- sub, [656](#)
- subin, [656](#)
- type\_string, [658](#)
- unpackhi\_twice, [655](#)
- unpacklo\_twice, [655](#)
- valid, [654](#)
- vect\_size, [658](#)
- vect\_t, [654](#)
- zero, [654](#)
- Simd512\_impl< true, true, false, 8 >, [658](#)
- add, [666](#)
- addin, [666](#)
- alignment, [668](#)
- blend, [665](#)
- compliant, [663](#)
- eq, [667](#)
- fmadd, [666](#)
- fmaddin, [666](#)
- fmaddx, [662](#)
- fmaddxin, [663](#)
- fmsub, [667](#)
- fmsubin, [667](#)
- fmsubx, [663](#)
- fmsubxin, [663](#)
- fnmadd, [666](#)
- fnmaddin, [666](#)
- fnmaddx, [663](#)
- fnmaddxin, [663](#)
- gather, [661](#), [664](#)
- greater, [662](#)
- greater\_eq, [662](#)
- hadd\_to\_scal, [663](#)
- half\_t, [661](#)

- lesser, [662](#)
- lesser\_eq, [662](#)
- load, [661](#), [664](#)
- loadu, [661](#), [664](#)
- mask\_high, [667](#)
- maskstore, [661](#), [664](#)
- mod, [667](#)
- mul, [666](#)
- mulhi\_fast, [667](#)
- mullo, [662](#)
- mulx, [662](#)
- round, [667](#)
- scalar\_t, [661](#)
- set, [661](#), [664](#)
- set1, [661](#), [663](#)
- shuffle, [665](#)
- signbits, [667](#)
- simdHalf, [661](#)
- sll, [665](#)
- sra, [662](#)
- srl, [665](#)
- store, [661](#), [664](#)
- storeu, [662](#), [664](#)
- stream, [662](#), [664](#)
- sub, [666](#)
- subin, [666](#)
- type\_string, [667](#)
- unpackhi, [665](#)
- unpackhi\_twice, [665](#)
- unpacklo, [665](#)
- unpacklo\_twice, [665](#)
- unpacklohi, [665](#)
- valid, [663](#)
- vand, [668](#)
- vandnot, [668](#)
- vect\_size, [668](#)
- vect\_t, [661](#)
- vor, [668](#)
- vxor, [668](#)
- zero, [668](#)
- Simd512\_impl< true, true, false, 8 >::Converter, [397](#)
- t, [398](#)
- v, [398](#)
- Simd512\_impl< true, true, true, 8 >, [668](#)
- add, [673](#)
- addin, [673](#)
- alignment, [677](#)
- blend, [673](#)
- compliant, [670](#)
- eq, [675](#)
- fmadd, [673](#)
- fmaddin, [674](#)
- fmaddx, [674](#)
- fmaddxin, [674](#)
- fmsub, [674](#)
- fmsubin, [675](#)
- fmsubx, [675](#)
- fmsubxin, [675](#)
- fnmadd, [674](#)
- fnmaddin, [674](#)
- fnmaddx, [674](#)
- fnmaddxin, [674](#)
- gather, [671](#)
- greater, [675](#)
- greater\_eq, [675](#)
- hadd\_to\_scal, [675](#)
- half\_t, [670](#)
- lesser, [675](#)
- lesser\_eq, [675](#)
- load, [671](#)
- loadu, [671](#)
- mask\_high, [676](#)
- maskstore, [671](#)
- mod, [676](#)
- mul, [673](#)
- mulhi\_fast, [676](#)
- mullo, [673](#)
- mulx, [673](#)
- round, [675](#)
- scalar\_t, [670](#)
- set, [671](#)
- set1, [671](#)
- shuffle, [672](#)
- signbits, [676](#)
- simdHalf, [670](#)
- sll, [672](#)
- sra, [672](#)
- srl, [672](#)
- store, [671](#)
- storeu, [671](#)
- stream, [672](#)
- sub, [673](#)
- subin, [673](#)
- type\_string, [676](#)
- unpackhi, [672](#)
- unpackhi\_twice, [672](#)
- unpacklo, [672](#)
- unpacklo\_twice, [672](#)
- unpacklohi, [672](#)
- valid, [670](#)
- vand, [676](#)
- vandnot, [676](#)
- vect\_size, [677](#)
- vect\_t, [670](#)
- vor, [676](#)
- vxor, [676](#)
- zero, [676](#)
- Simd512\_impl< true, true, true, 8 >::Converter, [398](#)
- t, [398](#)
- v, [398](#)
- simd512\_int32.inl, [812](#)
- \_\_FFLASFFPACK\_simd512\_int32\_INL, [813](#)
- simd512\_int64.inl, [813](#)
- \_\_simd512\_int64\_INL, [813](#)
- vect\_t, [813](#)
- Simd512fp\_base, [677](#)



- type\_string, [677](#)
- Simd512i\_base, [677](#)
  - type\_string, [678](#)
  - vand, [678](#)
  - vandnot, [678](#)
  - vect\_t, [678](#)
  - vor, [678](#)
  - vxor, [678](#)
  - zero, [678](#)
- SIMD\_INT
  - fflas\_simd.h, [806](#)
- simd\_modular.inl, [813](#)
- SimdChooser< T, bool, bool >, [678](#)
- SimdChooser< T, false, b >, [679](#)
  - value, [679](#)
- SimdChooser< T, true, false >, [679](#)
  - value, [679](#)
- SimdChooser< T, true, true >, [679](#)
  - value, [679](#)
- simdHalf
  - Simd256\_impl< true, true, false, 2 >, [589](#)
  - Simd256\_impl< true, true, false, 4 >, [600](#)
  - Simd256\_impl< true, true, false, 8 >, [614](#)
  - Simd256\_impl< true, true, true, 2 >, [623](#)
  - Simd256\_impl< true, true, true, 4 >, [632](#)
  - Simd256\_impl< true, true, true, 8 >, [645](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#)
  - Simd512\_impl< true, true, true, 8 >, [670](#)
- SimdSparseMatrix
  - FFLAS, [70](#)
- simdToType< T >, [679](#)
- Single, [680](#)
- size
  - Info, [445](#), [446](#)
  - RNSInteger< RNS >, [508](#)
  - RNSIntegerMod< RNS >, [512](#)
- SIZEOF\_\_INT64
  - config.h, [757](#)
- SIZEOF\_CHAR
  - config.h, [757](#)
- SIZEOF\_INT
  - config.h, [757](#)
- SIZEOF\_LONG
  - config.h, [757](#)
- SIZEOF\_LONG\_LONG
  - config.h, [757](#)
- SIZEOF\_SHORT
  - config.h, [757](#)
- sll
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [526](#)
  - Simd128\_impl< true, true, false, 2 >, [535](#)
  - Simd128\_impl< true, true, false, 4 >, [543](#)
  - Simd128\_impl< true, true, false, 8 >, [552](#)
  - Simd128\_impl< true, true, true, 2 >, [559](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [574](#)
  - Simd256\_impl< true, true, false, 2 >, [593](#)
  - Simd256\_impl< true, true, false, 4 >, [607](#)
  - Simd256\_impl< true, true, false, 8 >, [618](#)
  - Simd256\_impl< true, true, true, 2 >, [625](#)
  - Simd256\_impl< true, true, true, 4 >, [633](#), [639](#)
  - Simd256\_impl< true, true, true, 8 >, [647](#)
  - Simd512\_impl< true, true, false, 8 >, [665](#)
  - Simd512\_impl< true, true, true, 8 >, [672](#)
- sll128
  - Simd128\_impl< true, true, false, 2 >, [537](#)
  - Simd128\_impl< true, true, false, 4 >, [546](#)
  - Simd128\_impl< true, true, false, 8 >, [555](#)
  - Simd128\_impl< true, true, true, 2 >, [563](#)
  - Simd128\_impl< true, true, true, 4 >, [570](#)
  - Simd128\_impl< true, true, true, 8 >, [578](#)
  - Simd128i\_base, [580](#)
- Solve
  - FFPACK, [313](#), [350](#)
- solve.C, [743](#)
  - main, [743](#)
- Solve\_modular\_double
  - ffpack.C, [928](#)
  - ffpack\_c.h, [947](#)
- solveLB
  - FFPACK, [328](#), [350](#)
- solveLB2
  - FFPACK, [328](#), [350](#)
- solveLB2\_modular\_double
  - ffpack.C, [928](#)
  - ffpack\_c.h, [948](#)
- solveLB\_modular\_double
  - ffpack.C, [928](#)
  - ffpack\_c.h, [948](#)
- Sparse< \_Field, SparseMatrix\_t::COO >, [680](#)
  - col, [680](#)
  - dat, [681](#)
  - delayed, [681](#)
  - Field, [680](#)
  - kmax, [681](#)
  - m, [681](#)
  - maxrow, [681](#)
  - n, [681](#)
  - nElements, [681](#)
  - nnz, [681](#)
  - row, [680](#)
- Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [681](#)
  - col, [682](#)
  - cst, [682](#)
  - dat, [682](#)
  - delayed, [682](#)
  - Field, [682](#)
  - kmax, [682](#)
  - m, [682](#)
  - maxrow, [683](#)
  - n, [682](#)
  - nElements, [683](#)
  - nnz, [683](#)
  - row, [682](#)
- Sparse< \_Field, SparseMatrix\_t::CSR >, [683](#)

- col, [684](#)
- dat, [684](#)
- delayed, [684](#)
- Field, [683](#)
- kmax, [684](#)
- m, [684](#)
- maxrow, [684](#)
- n, [684](#)
- nElements, [684](#)
- nnz, [684](#)
- st, [684](#)
- stend, [684](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [685](#)
  - col, [685](#)
  - dat, [685](#)
  - delayed, [685](#)
  - Field, [685](#)
  - kmax, [685](#)
  - m, [685](#)
  - maxrow, [686](#)
  - n, [686](#)
  - nElements, [686](#)
  - nMOnes, [686](#)
  - nnz, [686](#)
  - nOnes, [686](#)
  - nOthers, [686](#)
  - st, [685](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [686](#)
  - col, [688](#)
  - cst, [687](#)
  - dat, [688](#)
  - delayed, [687](#)
  - Field, [687](#)
  - kmax, [687](#)
  - m, [687](#)
  - maxrow, [687](#)
  - n, [687](#)
  - nElements, [687](#)
  - nnz, [687](#)
  - st, [688](#)
  - stend, [688](#)
- Sparse< \_Field, SparseMatrix\_t::ELL >, [688](#)
  - col, [689](#)
  - dat, [689](#)
  - delayed, [689](#)
  - Field, [689](#)
  - kmax, [689](#)
  - ld, [689](#)
  - m, [689](#)
  - maxrow, [689](#)
  - n, [689](#)
  - nElements, [689](#)
  - nnz, [689](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [690](#)
  - chunk, [690](#)
  - col, [691](#)
  - dat, [691](#)
  - delayed, [690](#)
  - kmax, [690](#)
  - ld, [690](#)
  - m, [690](#)
  - maxrow, [691](#)
  - n, [690](#)
  - nChunks, [691](#)
  - nElements, [691](#)
  - nnz, [690](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [691](#)
  - chunk, [692](#)
  - col, [692](#)
  - cst, [692](#)
  - dat, [693](#)
  - delayed, [692](#)
  - kmax, [692](#)
  - ld, [692](#)
  - m, [692](#)
  - maxrow, [692](#)
  - n, [692](#)
  - nChunks, [692](#)
  - nElements, [692](#)
  - nnz, [692](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [693](#)
  - col, [694](#)
  - cst, [693](#)
  - dat, [694](#)
  - delayed, [693](#)
  - Field, [693](#)
  - kmax, [694](#)
  - ld, [694](#)
  - m, [694](#)
  - maxrow, [694](#)
  - n, [694](#)
  - nElements, [694](#)
  - nnz, [694](#)
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [694](#)
  - dat, [696](#)
  - delayed, [695](#)
  - Field, [695](#)
  - kmax, [695](#)
  - m, [695](#)
  - maxrow, [695](#)
  - mone, [696](#)
  - n, [695](#)
  - nElements, [695](#)
  - nnz, [695](#)
  - one, [696](#)
  - Self\_t, [695](#)
- Sparse< \_Field, SparseMatrix\_t::SELL >, [696](#)
  - chunk, [697](#)
  - chunkSize, [698](#)
  - col, [698](#)
  - dat, [698](#)
  - delayed, [697](#)
  - Field, [697](#)
  - kmax, [697](#)
  - m, [697](#)
  - maxrow, [697](#)

- n, [697](#)
  - nChunks, [697](#)
  - nElements, [697](#)
  - nnz, [697](#)
  - perm, [697](#)
  - sigma, [697](#)
  - st, [698](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [698](#)
    - chunk, [699](#)
    - chunkSize, [700](#)
    - col, [700](#)
    - cst, [699](#)
    - dat, [700](#)
    - delayed, [699](#)
    - Field, [699](#)
    - kmax, [699](#)
    - m, [699](#)
    - maxrow, [699](#)
    - n, [699](#)
    - nChunks, [699](#)
    - nElements, [700](#)
    - nnz, [699](#)
    - perm, [700](#)
    - sigma, [699](#)
    - st, [700](#)
  - Sparse< Field, SparseMatrix\_t, IdxT, PtrT >, [680](#)
  - sparse\_delete
    - FFLAS, [138–142](#), [145](#)
  - sparse\_init
    - FFLAS, [137–142](#), [145](#)
  - sparse\_matrix\_traits.h, [844](#)
  - sparse\_print
    - FFLAS, [139](#), [142](#), [145](#)
  - SparseMatrix\_t
    - FFLAS, [74](#)
  - SpecRankProfile
    - FFPACK, [349](#)
  - SpecRankProfile\_modular\_double
    - ffpack.C, [927](#)
    - ffpack\_c.h, [947](#)
  - splitt
    - parallel.h, [966](#)
  - SPLITTER
    - parallel.h, [966](#)
  - splitting\_0
    - parallel.h, [966](#)
  - splitting\_1
    - parallel.h, [966](#)
  - splitting\_2
    - parallel.h, [966](#)
  - splitting\_3
    - parallel.h, [966](#)
  - SpMat< Field, flag >, [700](#)
    - \_coo, [700](#)
    - \_csr, [700](#)
    - \_ell, [700](#)
  - sra
    - ScalFunctions< Element, typename enable\_if<
      - is\_integral< Element >::value >::type >, [526](#)
    - Simd128\_impl< true, true, false, 2 >, [532](#)
    - Simd128\_impl< true, true, false, 4 >, [541](#)
    - Simd128\_impl< true, true, false, 8 >, [549](#)
    - Simd128\_impl< true, true, true, 2 >, [559](#)
    - Simd128\_impl< true, true, true, 4 >, [567](#)
    - Simd128\_impl< true, true, true, 8 >, [574](#)
    - Simd256\_impl< true, true, false, 2 >, [591](#)
    - Simd256\_impl< true, true, false, 4 >, [601](#), [603](#)
    - Simd256\_impl< true, true, false, 8 >, [615](#)
    - Simd256\_impl< true, true, true, 2 >, [625](#)
    - Simd256\_impl< true, true, true, 4 >, [634](#), [639](#)
    - Simd256\_impl< true, true, true, 8 >, [647](#)
    - Simd512\_impl< true, true, false, 8 >, [662](#)
    - Simd512\_impl< true, true, true, 8 >, [672](#)
- srl
  - ScalFunctions< Element, typename enable\_if<
    - is\_integral< Element >::value >::type >, [526](#)
  - Simd128\_impl< true, true, false, 2 >, [535](#)
  - Simd128\_impl< true, true, false, 4 >, [543](#)
  - Simd128\_impl< true, true, false, 8 >, [552](#)
  - Simd128\_impl< true, true, true, 2 >, [559](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [574](#)
  - Simd256\_impl< true, true, false, 2 >, [593](#)
  - Simd256\_impl< true, true, false, 4 >, [607](#)
  - Simd256\_impl< true, true, false, 8 >, [618](#)
  - Simd256\_impl< true, true, true, 2 >, [625](#)
  - Simd256\_impl< true, true, true, 4 >, [634](#), [639](#)
  - Simd256\_impl< true, true, true, 8 >, [647](#)
  - Simd512\_impl< true, true, false, 8 >, [665](#)
  - Simd512\_impl< true, true, true, 8 >, [672](#)
- srl128
  - Simd128\_impl< true, true, false, 2 >, [537](#)
  - Simd128\_impl< true, true, false, 4 >, [546](#)
  - Simd128\_impl< true, true, false, 8 >, [555](#)
  - Simd128\_impl< true, true, true, 2 >, [563](#)
  - Simd128\_impl< true, true, true, 4 >, [570](#)
  - Simd128\_impl< true, true, true, 8 >, [578](#)
  - Simd128i\_base, [580](#)
- sscal\_
  - config-blas.h, [752](#)
- st
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [685](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [688](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [698](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [700](#)
- Static\_error\_check
  - Static\_error\_check< bool >, [701](#)
- Static\_error\_check< bool >, [701](#)
  - Static\_error\_check, [701](#)
- Static\_error\_check< false >, [701](#)
- StatsMatrix, [701](#)
  - averageCol, [703](#)
  - averageColDifference, [703](#)

- averageRow, [702](#)
- averageRowDifference, [703](#)
- coldim, [702](#)
- denseCols, [704](#)
- denseRows, [704](#)
- deviationCol, [703](#)
- deviationColDifference, [703](#)
- deviationRow, [702](#)
- deviationRowDifference, [703](#)
- maxCol, [702](#)
- maxColDifference, [703](#)
- maxRow, [702](#)
- maxRowDifference, [703](#)
- minCol, [702](#)
- minColDifference, [703](#)
- minRow, [702](#)
- minRowDifference, [703](#)
- nDenseCols, [703](#)
- nDenseRows, [703](#)
- nEmptyCols, [703](#)
- nEmptyColsEnd, [703](#)
- nEmptyRows, [703](#)
- nMOnes, [702](#)
- nnz, [702](#)
- nOnes, [702](#)
- nOthers, [702](#)
- rowdim, [702](#)
- STD\_RECINT\_SIZE
  - benchmark-fgemm-mp.C, [724](#)
  - benchmark-fgemv-mp.C, [727](#)
- STDC\_HEADERS
  - config.h, [757](#)
- stend
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [684](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [688](#)
- store
  - Simd128\_impl< true, true, false, 2 >, [532](#), [534](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#), [543](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#), [552](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [574](#)
  - Simd256\_impl< true, false, true, 8 >, [584](#)
  - Simd256\_impl< true, true, false, 2 >, [590](#), [593](#)
  - Simd256\_impl< true, true, false, 4 >, [601](#), [603](#), [606](#)
  - Simd256\_impl< true, true, false, 8 >, [615](#), [618](#)
  - Simd256\_impl< true, true, true, 2 >, [624](#)
  - Simd256\_impl< true, true, true, 4 >, [633](#), [639](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [655](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#), [664](#)
  - Simd512\_impl< true, true, true, 8 >, [671](#)
- storeu
  - Simd128\_impl< true, true, false, 2 >, [532](#), [534](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#), [543](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#), [552](#)
  - Simd128\_impl< true, true, true, 2 >, [559](#)
- Simd128\_impl< true, true, true, 4 >, [566](#)
- Simd128\_impl< true, true, true, 8 >, [574](#)
- Simd256\_impl< true, true, false, 8 >, [615](#), [618](#)
- Simd256\_impl< true, true, true, 2 >, [624](#)
- Simd256\_impl< true, true, true, 4 >, [633](#), [639](#)
- Simd256\_impl< true, true, true, 8 >, [646](#)
- Simd512\_impl< true, false, true, 8 >, [655](#)
- Simd512\_impl< true, true, false, 8 >, [662](#), [664](#)
- Simd512\_impl< true, true, true, 8 >, [671](#)
- stream
  - Simd128\_impl< true, true, false, 2 >, [532](#), [534](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#), [543](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#), [552](#)
  - Simd128\_impl< true, true, true, 2 >, [559](#)
  - Simd128\_impl< true, true, true, 4 >, [566](#)
  - Simd128\_impl< true, true, true, 8 >, [574](#)
  - Simd256\_impl< true, false, true, 8 >, [584](#)
  - Simd256\_impl< true, true, false, 2 >, [591](#), [593](#)
  - Simd256\_impl< true, true, false, 4 >, [601](#), [603](#), [606](#)
  - Simd256\_impl< true, true, false, 8 >, [615](#), [618](#)
  - Simd256\_impl< true, true, true, 2 >, [624](#)
  - Simd256\_impl< true, true, true, 4 >, [633](#), [639](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [655](#)
  - Simd512\_impl< true, true, false, 8 >, [662](#), [664](#)
  - Simd512\_impl< true, true, true, 8 >, [672](#)
- strmm\_
  - config-blas.h, [753](#)
- strsm\_
  - config-blas.h, [753](#)
- sub
  - FFLAS::vectorised, [268](#)
  - FieldSimd< \_Field >, [416](#)
  - RNSIntegerMod< RNS >, [514](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [520](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [524](#)
  - Simd128\_impl< true, true, false, 2 >, [535](#)
  - Simd128\_impl< true, true, false, 4 >, [544](#)
  - Simd128\_impl< true, true, false, 8 >, [553](#)
  - Simd128\_impl< true, true, true, 2 >, [560](#)
  - Simd128\_impl< true, true, true, 4 >, [567](#)
  - Simd128\_impl< true, true, true, 8 >, [575](#)
  - Simd256\_impl< true, false, true, 8 >, [584](#)
  - Simd256\_impl< true, true, false, 2 >, [595](#)
  - Simd256\_impl< true, true, false, 4 >, [608](#), [609](#)
  - Simd256\_impl< true, true, false, 8 >, [619](#)
  - Simd256\_impl< true, true, true, 2 >, [626](#)
  - Simd256\_impl< true, true, true, 4 >, [635](#), [640](#)
  - Simd256\_impl< true, true, true, 8 >, [648](#)
  - Simd512\_impl< true, false, true, 8 >, [656](#)

- Simd512\_impl< true, true, false, 8 >, [666](#)
  - Simd512\_impl< true, true, true, 8 >, [673](#)
- sub\_r
  - FieldSimd< \_Field >, [416](#), [417](#)
- subin
  - FieldSimd< \_Field >, [416](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [520](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [524](#)
  - Simd128\_impl< true, true, false, 2 >, [535](#)
  - Simd128\_impl< true, true, false, 4 >, [544](#)
  - Simd128\_impl< true, true, false, 8 >, [553](#)
  - Simd128\_impl< true, true, true, 2 >, [560](#)
  - Simd128\_impl< true, true, true, 4 >, [567](#)
  - Simd128\_impl< true, true, true, 8 >, [575](#)
  - Simd256\_impl< true, false, true, 8 >, [585](#)
  - Simd256\_impl< true, true, false, 2 >, [595](#)
  - Simd256\_impl< true, true, false, 4 >, [609](#)
  - Simd256\_impl< true, true, false, 8 >, [619](#)
  - Simd256\_impl< true, true, true, 2 >, [626](#)
  - Simd256\_impl< true, true, true, 4 >, [635](#), [640](#)
  - Simd256\_impl< true, true, true, 8 >, [648](#)
  - Simd512\_impl< true, false, true, 8 >, [656](#)
  - Simd512\_impl< true, true, false, 8 >, [666](#)
  - Simd512\_impl< true, true, true, 8 >, [673](#)
- subin\_r
  - FieldSimd< \_Field >, [417](#)
- subp
  - FFLAS::vectorised, [267](#)
- support\_fast\_mod< double >, [704](#)
- support\_fast\_mod< float >, [704](#)
- support\_fast\_mod< int64\_t >, [705](#)
- support\_fast\_mod< T >, [704](#)
- support\_simd< T >, [705](#)
- support\_simd\_add< T >, [705](#)
- support\_simd\_mod< T >, [705](#)
- swapval
  - FFPACK, [362](#)
- SYNCH\_GROUP
  - parallel.h, [962](#)
- SysTimer
  - FFLAS, [72](#)
- T
  - limits< char >, [454](#)
  - limits< double >, [454](#)
  - limits< float >, [455](#)
  - limits< Givaro::Integer >, [456](#)
  - limits< int >, [456](#)
  - limits< long >, [457](#)
  - limits< long long >, [457](#)
  - limits< Reclnt::rint< K > >, [458](#)
  - limits< Reclnt::ruint< K > >, [459](#)
  - limits< short int >, [459](#)
  - limits< signed char >, [460](#)
  - limits< unsigned char >, [460](#)
  - limits< unsigned int >, [461](#)
  - limits< unsigned long >, [462](#)
  - limits< unsigned long long >, [462](#)
  - limits< unsigned short int >, [463](#)
- t
  - Simd128\_impl< true, true, false, 2 >::Converter, [394](#)
  - Simd128\_impl< true, true, false, 4 >::Converter, [394](#)
  - Simd128\_impl< true, true, false, 8 >::Converter, [394](#)
  - Simd128\_impl< true, true, true, 2 >::Converter, [395](#)
  - Simd128\_impl< true, true, true, 4 >::Converter, [395](#)
  - Simd128\_impl< true, true, true, 8 >::Converter, [395](#)
  - Simd256\_impl< true, true, false, 2 >::Converter, [396](#)
  - Simd256\_impl< true, true, false, 4 >::Converter, [396](#)
  - Simd256\_impl< true, true, false, 8 >::Converter, [396](#)
  - Simd256\_impl< true, true, true, 2 >::Converter, [397](#)
  - Simd256\_impl< true, true, true, 4 >::Converter, [397](#)
  - Simd256\_impl< true, true, true, 8 >::Converter, [397](#)
  - Simd512\_impl< true, true, false, 8 >::Converter, [398](#)
  - Simd512\_impl< true, true, true, 8 >::Converter, [398](#)
- TASK
  - parallel.h, [961](#)
- tBC
  - test-lu.C, [1022](#)
  - test-permutations.C, [1026](#)
- test
  - test-maxdelayeddim.C, [1023](#)
  - test-simd.C, [1032](#)
- test-charpoly-check.C, [985](#)
  - ENABLE\_CHECKER\_charpoly, [985](#)
  - main, [985](#)
  - printPolynomial, [985](#)
  - TIME\_CHECKER\_CHARPOLY, [985](#)
- test-charpoly.C, [985](#)
  - launch\_test, [986](#)
  - main, [986](#)
  - run\_with\_field, [986](#)
- test-compressQ.C, [986](#)
  - Field, [987](#)
  - main, [987](#)
  - printvect, [987](#)
- test-det-check.C, [987](#)
  - ENABLE\_CHECKER\_Det, [988](#)
  - main, [988](#)
  - TIME\_CHECKER\_Det, [988](#)
- test-det.C, [988](#)

- main, 988
- test\_det, 988
- test-echelon.C, 989
  - \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE, 989
  - \_\_FFLASFFPACK\_PLUQ\_THRESHOLD, 989
  - \_\_FFLASFFPACK\_SEQUENTIAL, 989
  - main, 991
  - run\_with\_field, 990
  - test\_colechelon, 989
  - test\_redcoechelon, 990
  - test\_redrowechelon, 990
  - test\_rowechelon, 990
- test-fadd.C, 991
  - main, 992
  - test\_fadd, 991
  - test\_faddin, 991
  - test\_fsub, 992
  - test\_fsubin, 992
- test-fdot.C, 992
  - check\_fdot, 993
  - ENABLE\_ALL\_CHECKINGS, 993
  - main, 993
  - run\_with\_field, 993
  - run\_with\_Integer, 993
- test-fgemm-check.C, 993
  - ENABLE\_ALL\_CHECKINGS, 994
  - launch\_MM\_dispatch, 994
  - main, 994
  - run\_with\_field, 994
- test-fgemm.C, 995
  - check\_MM, 995
  - ENABLE\_CHECKER\_fgemm, 995
  - launch\_MM, 996
  - launch\_MM\_dispatch, 996
  - main, 997
  - run\_with\_field, 996
- test-fgemv.C, 997
  - check\_MV, 997
  - launch\_MV, 998
  - launch\_MV\_dispatch, 998
  - main, 998
  - run\_with\_field, 998
- test-fger.C, 998
  - check\_fger, 999
  - launch\_fger, 999
  - launch\_fger\_dispatch, 1000
  - main, 1000
  - run\_with\_field, 1000
  - TIME, 999
- test-fgesv.C, 1000
  - main, 1001
  - run\_with\_field, 1001
  - test\_rect\_fgesv, 1001
  - test\_square\_fgesv, 1001
- test-finit.C, 1002
  - main, 1002
  - run\_with\_field, 1002
- test\_freduce, 1002
- test-fscal.C, 1003
  - main, 1004
  - test\_fscal, 1003
  - test\_fscaln, 1003, 1004
- test-fsyr2k.C, 1004
  - check\_fsyr2k, 1004
  - ENABLE\_ALL\_CHECKINGS, 1004
  - main, 1005
  - run\_with\_field, 1005
- test-fsyrk.C, 1005
  - check\_fsyrk, 1006
  - check\_fsyrk\_bkdiag, 1006
  - check\_fsyrk\_diag, 1006
  - ENABLE\_ALL\_CHECKINGS, 1006
  - main, 1007
  - run\_with\_field, 1006
- test-fsytrf.C, 1007
  - main, 1008
  - operator<<, 1007
  - run\_with\_field, 1008
  - test\_generic\_fsytrf, 1008
  - test\_RPM\_fsytrf, 1007
- test-ftmm.C, 1008
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1009
  - check\_ftmm, 1009
  - main, 1009
  - run\_with\_field, 1009
- test-ftmv.C, 1009
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1010
  - check\_ftmv, 1010
  - ENABLE\_ALL\_CHECKINGS, 1010
  - main, 1010
  - run\_with\_field, 1010
- test-ftsm-check.C, 1010
  - ENABLE\_ALL\_CHECKINGS, 1011
  - main, 1011
- test-ftsm.C, 1011
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1012
  - check\_ftsm, 1012
  - ENABLE\_ALL\_CHECKINGS, 1012
  - main, 1012
  - run\_with\_field, 1012
- test-ftssyr2k.C, 1012
  - check\_ftssyr2k, 1013
  - ENABLE\_ALL\_CHECKINGS, 1013
  - main, 1013
  - run\_with\_field, 1013
- test-ftstr.C, 1013
  - check\_ftstr, 1014
  - ENABLE\_ALL\_CHECKINGS, 1014
  - main, 1014
  - run\_with\_field, 1014
- test-ftsv.C, 1014
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1015
  - check\_ftsv, 1015
  - ENABLE\_ALL\_CHECKINGS, 1015
  - main, 1015

- run\_with\_field, 1015
- test-ftsrti.C, 1016
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1016
  - check\_ftsrti, 1016
  - ENABLE\_ALL\_CHECKINGS, 1016
  - main, 1016
  - run\_with\_field, 1016
- test-interfaces-c.c, 1017
  - main, 1017
- test-invert-check.C, 1017
  - ENABLE\_ALL\_CHECKINGS, 1017
  - main, 1017
- test-io.C, 1018
  - main, 1018
  - run\_with\_field, 1018
- test-lu.C, 1018
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1019
  - \_\_LUDIVINE\_CUTOFF, 1019
  - BASECASE\_K, 1019
  - launch\_test, 1021
  - main, 1021
  - mvcnt, 1022
  - run\_with\_field, 1021
  - tBC, 1022
  - test\_LUdivine, 1019
  - test\_pluq, 1021
  - tgemm, 1022
  - timtot, 1022
  - tperm, 1022
  - trest, 1022
  - ttrsm, 1022
  - verifPLUQ, 1020
- test-maxdelayeddim.C, 1022
  - main, 1023
  - MAX\_WITH\_SIZE\_T, 1023
  - test, 1023
- test-minpoly.C, 1023
  - check\_minpoly, 1023
  - main, 1024
  - run\_with\_field, 1023
- test-multifile1.C, 1024
- test-multifile2.C, 1024
  - main, 1024
- test-nullspace.C, 1024
  - checkingMessage, 1025
  - main, 1025
  - readOrRandomMatrixWithRankAndRandomRPM, 1025
  - run\_with\_field, 1025
  - test\_nullspace, 1025
- test-permutations.C, 1026
  - checkMonotonicApplyP, 1026
  - main, 1026
  - tBC, 1026
  - tgemm, 1026
  - timtot, 1026
  - tperm, 1026
  - trest, 1026
- ttrsm, 1026
- test-pluq-check.C, 1027
  - ENABLE\_ALL\_CHECKINGS, 1027
  - main, 1027
- test-rankprofiles.C, 1027
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1028
  - main, 1028
  - run\_with\_field, 1028
- test-rpm.C, 1028
  - checkRPM, 1028
  - checkSymmetricRPM, 1028
  - main, 1028
- test-simd.C, 1029
  - check\_eq, 1031
  - eval\_func\_on\_array, 1031
  - generate\_random\_vector, 1031
  - integer, 1030
  - main, 1032
  - REGISTER\_TYPE\_NAME, 1030, 1031
  - test, 1032
  - test\_impl, 1032
  - TEST\_ONE\_OP, 1030
  - test\_op, 1032
  - TypeName, 1030
- test-solve.C, 1032
  - check\_solve, 1033
  - main, 1033
  - run\_with\_field, 1033
- test-utils.h, 977
- test\_colechelon
  - test-echelon.C, 989
- test\_det
  - test-det.C, 988
- test\_fadd
  - test-fadd.C, 991
- test\_faddin
  - test-fadd.C, 991
- test\_freduce
  - test-finit.C, 1002
- test\_fscal
  - test-fscal.C, 1003
- test\_fscaln
  - test-fscal.C, 1003, 1004
- test\_fsub
  - test-fadd.C, 992
- test\_fsubin
  - test-fadd.C, 992
- test\_generic\_fsytrf
  - test-fsytrf.C, 1008
- test\_impl
  - test-simd.C, 1032
- test\_LUdivine
  - test-lu.C, 1019
- test\_nullspace
  - test-nullspace.C, 1025
- TEST\_ONE\_OP
  - test-simd.C, 1030
- test\_op



- test-simd.C, [1032](#)
- test\_pluq
  - test-lu.C, [1021](#)
- test\_rect\_fgesv
  - test-fgesv.C, [1001](#)
- test\_redcolechelon
  - test-echelon.C, [990](#)
- test\_redrowechelon
  - test-echelon.C, [990](#)
- test\_rowechelon
  - test-echelon.C, [990](#)
- test\_RPM\_fsytrf
  - test-fsytrf.C, [1007](#)
- test\_square\_fgesv
  - test-fgesv.C, [1001](#)
- tfn\_minus, [706](#)
  - operator(), [706](#)
- tfn\_minus\_eq, [706](#)
  - operator(), [706](#)
- tfn\_mul, [706](#)
  - operator(), [707](#)
- tfn\_mul\_eq, [707](#)
  - operator(), [707](#)
- tfn\_plus, [707](#)
  - operator(), [707](#)
- tfn\_plus\_eq, [708](#)
  - operator(), [708](#)
- tgemm
  - test-lu.C, [1022](#)
  - test-permutations.C, [1026](#)
- THREADS
  - benchmark-fgemm-rns.C, [725](#)
- Threads, [708](#)
- threads\_fgemm
  - FFPACK, [341](#)
- threads\_ftsm
  - FFPACK, [341](#)
- THREED
  - benchmark-fgemm-rns.C, [725](#)
- ThreeD, [708](#)
- THREEDA
  - benchmark-fgemm-rns.C, [725](#)
- ThreeDAdaptive, [708](#)
- ThreeDInPlace, [708](#)
- THREEDIIP
  - benchmark-fgemm-rns.C, [725](#)
- TIME
  - test-fger.C, [999](#)
- TIME\_CHECKER\_CHARPOLY
  - test-charpoly-check.C, [985](#)
- TIME\_CHECKER\_Det
  - test-det-check.C, [988](#)
- Timer
  - FFLAS, [72](#)
- timer.h, [978](#)
- timtot
  - test-lu.C, [1022](#)
  - test-permutations.C, [1026](#)
- tmain
  - benchmark-fgemm-mp.C, [724](#)
  - benchmark-fgemv-mp.C, [727](#)
- Todo List, [9](#)
- tperm
  - test-lu.C, [1022](#)
  - test-permutations.C, [1026](#)
- trest
  - test-lu.C, [1022](#)
  - test-permutations.C, [1026](#)
- trinv\_left
  - FFPACK, [294](#), [345](#)
- trinv\_left\_modular\_double
  - ffpack.C, [920](#)
  - ffpack\_c.h, [941](#)
- TRSMBound
  - FFLAS::Protected, [201](#), [202](#)
- TRSMHelper
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [709](#)
- TRSMHelper< ReclterTrait, ParSeqTrait >, [708](#)
  - parseq, [710](#)
  - pMMH, [709](#)
  - TRSMHelper, [709](#)
- TTimer
  - arithprog.C, [711](#)
  - autotune/charpoly.C, [740](#)
  - autotune/pluq.C, [742](#)
  - benchmark-dgemm.C, [718](#)
  - benchmark-dgetrf.C, [719](#)
  - benchmark-dgetri.C, [719](#)
  - benchmark-dsytrf.C, [720](#)
  - benchmark-dtrsm.C, [721](#)
  - benchmark-dtrtri.C, [722](#)
  - fsyrk.C, [712](#)
  - fsytrf.C, [713](#)
  - fttrtri.C, [714](#)
  - winograd.C, [715](#)
- ttrsm
  - test-lu.C, [1022](#)
  - test-permutations.C, [1026](#)
- Tutorial, [2](#)
- TWOD
  - benchmark-fgemm-rns.C, [725](#)
- TwoD, [710](#)
- TWODA
  - benchmark-fgemm-rns.C, [725](#)
- TwoDAdaptive, [710](#)
- type
  - Argument, [379](#)
  - associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, [379](#)
  - associatedDelayedField< const Givaro::Modular< T, X > >, [380](#)
  - associatedDelayedField< const Givaro::ModularBalanced< T > >, [380](#)
  - associatedDelayedField< const Givaro::ZRing< T > >, [381](#)
  - associatedDelayedField< Field >, [379](#)



- CompactElement< double >, 390
- CompactElement< Element >, 390
- CompactElement< float >, 390
- CompactElement< int16\_t >, 391
- CompactElement< int32\_t >, 391
- CompactElement< int64\_t >, 391
- is\_simd< T >, 447
- TYPE\_BOOL
  - args-parser.h, 968
- TYPE\_DOUBLE
  - args-parser.h, 969
- TYPE\_INT
  - args-parser.h, 969
- TYPE\_INTEGER
  - args-parser.h, 969
- type\_integer
  - args-parser.h, 969
- TYPE\_INTLIST
  - args-parser.h, 969
- TYPE\_LONGLONG
  - args-parser.h, 969
- TYPE\_NONE
  - args-parser.h, 969
- TYPE\_STR
  - args-parser.h, 969
- type\_string
  - NoSimd< T >, 486
  - Simd128\_impl< true, false, true, 4 >, 529
  - Simd128\_impl< true, false, true, 8 >, 529
  - Simd128\_impl< true, true, false, 2 >, 537
  - Simd128\_impl< true, true, false, 4 >, 545
  - Simd128\_impl< true, true, false, 8 >, 555
  - Simd128\_impl< true, true, true, 2 >, 563
  - Simd128\_impl< true, true, true, 4 >, 570
  - Simd128\_impl< true, true, true, 8 >, 578
  - Simd128fp\_base, 579
  - Simd128i\_base, 580
  - Simd256\_impl< true, true, false, 2 >, 596
  - Simd256\_impl< true, true, false, 4 >, 611
  - Simd256\_impl< true, true, false, 8 >, 621
  - Simd256\_impl< true, true, true, 2 >, 629
  - Simd256\_impl< true, true, true, 4 >, 643
  - Simd256\_impl< true, true, true, 8 >, 651
  - Simd256i\_base, 652
  - Simd512\_impl< true, false, true, 4 >, 653
  - Simd512\_impl< true, false, true, 8 >, 658
  - Simd512\_impl< true, true, false, 8 >, 667
  - Simd512\_impl< true, true, true, 8 >, 676
  - Simd512fp\_base, 677
  - Simd512i\_base, 678
- TYPE\_UINT64
  - args-parser.h, 969
- TypeName
  - test-simd.C, 1030
- uint16\_t
  - instrset.h, 981
- uint32\_t
  - instrset.h, 982
- uint64\_t
  - instrset.h, 982
- uint8\_t
  - instrset.h, 981
- unfit
  - FFLAS::Protected, 208, 209
- unpackhi
  - Simd128\_impl< true, true, false, 2 >, 535
  - Simd128\_impl< true, true, false, 4 >, 543
  - Simd128\_impl< true, true, false, 8 >, 553
  - Simd128\_impl< true, true, true, 2 >, 559
  - Simd128\_impl< true, true, true, 4 >, 567
  - Simd128\_impl< true, true, true, 8 >, 575
  - Simd256\_impl< true, true, false, 2 >, 594
  - Simd256\_impl< true, true, false, 4 >, 608
  - Simd256\_impl< true, true, false, 8 >, 619
  - Simd256\_impl< true, true, true, 2 >, 625
  - Simd256\_impl< true, true, true, 4 >, 634
  - Simd256\_impl< true, true, true, 8 >, 647
  - Simd512\_impl< true, true, false, 8 >, 665
  - Simd512\_impl< true, true, true, 8 >, 672
- unpackhi\_twice
  - Simd256\_impl< true, false, true, 8 >, 584
  - Simd256\_impl< true, true, false, 2 >, 594
  - Simd256\_impl< true, true, false, 4 >, 607
  - Simd256\_impl< true, true, false, 8 >, 619
  - Simd256\_impl< true, true, true, 2 >, 625
  - Simd256\_impl< true, true, true, 4 >, 634
  - Simd256\_impl< true, true, true, 8 >, 647
  - Simd512\_impl< true, false, true, 8 >, 655
  - Simd512\_impl< true, true, false, 8 >, 665
  - Simd512\_impl< true, true, true, 8 >, 672
- unpacklo
  - Simd128\_impl< true, true, false, 2 >, 535
  - Simd128\_impl< true, true, false, 4 >, 543
  - Simd128\_impl< true, true, false, 8 >, 552
  - Simd128\_impl< true, true, true, 2 >, 559
  - Simd128\_impl< true, true, true, 4 >, 567
  - Simd128\_impl< true, true, true, 8 >, 575
  - Simd256\_impl< true, true, false, 2 >, 594
  - Simd256\_impl< true, true, false, 4 >, 608
  - Simd256\_impl< true, true, false, 8 >, 619
  - Simd256\_impl< true, true, true, 2 >, 625
  - Simd256\_impl< true, true, true, 4 >, 634
  - Simd256\_impl< true, true, true, 8 >, 647
  - Simd512\_impl< true, true, false, 8 >, 665
  - Simd512\_impl< true, true, true, 8 >, 672
- unpacklo\_twice
  - Simd256\_impl< true, false, true, 8 >, 584
  - Simd256\_impl< true, true, false, 2 >, 594
  - Simd256\_impl< true, true, false, 4 >, 607
  - Simd256\_impl< true, true, false, 8 >, 618
  - Simd256\_impl< true, true, true, 2 >, 625
  - Simd256\_impl< true, true, true, 4 >, 634
  - Simd256\_impl< true, true, true, 8 >, 647
  - Simd512\_impl< true, false, true, 8 >, 655
  - Simd512\_impl< true, true, false, 8 >, 665
  - Simd512\_impl< true, true, true, 8 >, 672

- unpacklohi
  - Simd256\_impl< true, true, false, 2 >, [594](#)
  - Simd256\_impl< true, true, false, 4 >, [608](#)
  - Simd256\_impl< true, true, false, 8 >, [619](#)
  - Simd256\_impl< true, true, true, 2 >, [625](#)
  - Simd256\_impl< true, true, true, 4 >, [634](#)
  - Simd256\_impl< true, true, true, 8 >, [647](#)
  - Simd512\_impl< true, true, false, 8 >, [665](#)
  - Simd512\_impl< true, true, true, 8 >, [672](#)
- UnparametricTag, [710](#)
- updateD
  - FFPACK::Protected, [374](#)
- USE\_OPENMP
  - config.h, [757](#)
- UserTimer
  - FFLAS, [72](#)
- utils.h, [845](#)
- v
  - Simd128\_impl< true, true, false, 2 >::Converter, [394](#)
  - Simd128\_impl< true, true, false, 4 >::Converter, [394](#)
  - Simd128\_impl< true, true, false, 8 >::Converter, [394](#)
  - Simd128\_impl< true, true, true, 2 >::Converter, [395](#)
  - Simd128\_impl< true, true, true, 4 >::Converter, [395](#)
  - Simd128\_impl< true, true, true, 8 >::Converter, [395](#)
  - Simd256\_impl< true, true, false, 2 >::Converter, [396](#)
  - Simd256\_impl< true, true, false, 4 >::Converter, [396](#)
  - Simd256\_impl< true, true, false, 8 >::Converter, [396](#)
  - Simd256\_impl< true, true, true, 2 >::Converter, [397](#)
  - Simd256\_impl< true, true, true, 4 >::Converter, [397](#)
  - Simd256\_impl< true, true, true, 8 >::Converter, [397](#)
  - Simd512\_impl< true, true, false, 8 >::Converter, [398](#)
  - Simd512\_impl< true, true, true, 8 >::Converter, [398](#)
- val
  - Coo< Field >, [401](#)
  - Coo< ValT, IdxT >, [400](#), [403](#)
- valid
  - NoSimd< T >, [486](#)
  - Simd128\_impl< true, true, false, 2 >, [533](#)
  - Simd128\_impl< true, true, false, 4 >, [542](#)
  - Simd128\_impl< true, true, false, 8 >, [551](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [565](#)
  - Simd128\_impl< true, true, true, 8 >, [573](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [592](#)
  - Simd256\_impl< true, true, false, 4 >, [605](#)
  - Simd256\_impl< true, true, false, 8 >, [617](#)
  - Simd256\_impl< true, true, true, 2 >, [623](#)
  - Simd256\_impl< true, true, true, 4 >, [632](#), [638](#)
  - Simd256\_impl< true, true, true, 8 >, [646](#)
  - Simd512\_impl< true, false, true, 8 >, [654](#)
  - Simd512\_impl< true, true, false, 8 >, [663](#)
  - Simd512\_impl< true, true, true, 8 >, [670](#)
- VALUE
  - parallel.h, [962](#)
- value
  - AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >, [377](#)
  - AlgoChooser< ModeT, ParSeq >, [377](#)
  - AreEqual< X, X >, [378](#)
  - AreEqual< X, Y >, [378](#)
  - compatible\_data\_type< Field >, [391](#)
  - compatible\_data\_type< Givaro::ZRing< double > >, [392](#)
  - compatible\_data\_type< Givaro::ZRing< float > >, [392](#)
  - ElementTraits< double >, [406](#)
  - ElementTraits< Element >, [405](#)
  - ElementTraits< FFPACK::rns\_double\_elt >, [406](#)
  - ElementTraits< float >, [406](#)
  - ElementTraits< Givaro::Integer >, [407](#)
  - ElementTraits< int16\_t >, [407](#)
  - ElementTraits< int32\_t >, [407](#)
  - ElementTraits< int64\_t >, [408](#)
  - ElementTraits< int8\_t >, [408](#)
  - ElementTraits< Reclnt::rint< K > >, [408](#)
  - ElementTraits< Reclnt::rmint< K, MG > >, [408](#)
  - ElementTraits< Reclnt::ruint< K > >, [409](#)
  - ElementTraits< uint16\_t >, [409](#)
  - ElementTraits< uint32\_t >, [409](#)
  - ElementTraits< uint64\_t >, [410](#)
  - ElementTraits< uint8\_t >, [410](#)
  - has\_minus\_eq\_impl< C >, [438](#)
  - has\_minus\_impl< C >, [438](#)
  - has\_mul\_eq\_impl< C >, [439](#)
  - has\_mul\_impl< C >, [439](#)
  - has\_operation< T >, [439](#)
  - has\_plus\_eq\_impl< C >, [440](#)
  - has\_plus\_impl< C >, [440](#)
  - is\_simd< T >, [447](#)
  - ModeTraits< Field >, [478](#)
  - ModeTraits< Givaro::Modular< Element, Compute > >, [478](#)
  - ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >, [479](#)
  - ModeTraits< Givaro::Modular< int16\_t, Compute > >, [479](#)
  - ModeTraits< Givaro::Modular< int32\_t, Compute > >, [479](#)
  - ModeTraits< Givaro::Modular< int8\_t, Compute > >, [480](#)

- ModeTraits< Givaro::Modular< RecInt::ruint< K  
>, Compute > >, [480](#)
- ModeTraits< Givaro::Modular< uint16\_t, Compute  
> >, [480](#)
- ModeTraits< Givaro::Modular< uint32\_t, Compute  
> >, [481](#)
- ModeTraits< Givaro::Modular< uint8\_t, Compute  
> >, [481](#)
- ModeTraits< Givaro::ModularBalanced< Element  
> >, [481](#)
- ModeTraits< Givaro::ModularBalanced< Gi-  
varo::Integer > >, [482](#)
- ModeTraits< Givaro::ModularBalanced< int16\_t >  
>, [482](#)
- ModeTraits< Givaro::ModularBalanced< int32\_t >  
>, [482](#)
- ModeTraits< Givaro::ModularBalanced< int8\_t >  
>, [483](#)
- ModeTraits< Givaro::Montgomery< T > >, [483](#)
- ModeTraits< Givaro::ZRing< double > >, [483](#)
- ModeTraits< Givaro::ZRing< float > >, [483](#)
- ModeTraits< Givaro::ZRing< Givaro::Integer > >,  
[484](#)
- need\_field\_characteristic< Field >, [484](#)
- need\_field\_characteristic< Givaro::Modular< Field  
> >, [485](#)
- need\_field\_characteristic< Givaro::ModularBalanced<  
Field > >, [485](#)
- SimdChooser< T, false, b >, [679](#)
- SimdChooser< T, true, false >, [679](#)
- SimdChooser< T, true, true >, [679](#)
- vand
  - ScalFunctions< Element, typename enable\_if<  
is\_floating\_point< Element >::value >::type  
>, [519](#)
  - ScalFunctions< Element, typename enable\_if<  
is\_integral< Element >::value >::type >, [523](#)
  - Simd128\_impl< true, true, false, 2 >, [537](#)
  - Simd128\_impl< true, true, false, 4 >, [546](#)
  - Simd128\_impl< true, true, false, 8 >, [555](#)
  - Simd128\_impl< true, true, true, 2 >, [563](#)
  - Simd128\_impl< true, true, true, 4 >, [570](#)
  - Simd128\_impl< true, true, true, 8 >, [578](#)
  - Simd128i\_base, [580](#)
  - Simd256\_impl< true, false, true, 8 >, [586](#)
  - Simd256\_impl< true, true, false, 4 >, [612](#)
  - Simd256\_impl< true, true, true, 4 >, [643](#)
  - Simd512\_impl< true, true, false, 8 >, [668](#)
  - Simd512\_impl< true, true, true, 8 >, [676](#)
  - Simd512i\_base, [678](#)
- vandnot
  - ScalFunctions< Element, typename enable\_if<  
is\_floating\_point< Element >::value >::type  
>, [519](#)
  - ScalFunctions< Element, typename enable\_if<  
is\_integral< Element >::value >::type >, [523](#)
  - Simd128\_impl< true, true, false, 2 >, [537](#)
  - Simd128\_impl< true, true, false, 4 >, [546](#)
- Simd128\_impl< true, true, false, 8 >, [555](#)
- Simd128\_impl< true, true, true, 2 >, [563](#)
- Simd128\_impl< true, true, true, 4 >, [570](#)
- Simd128\_impl< true, true, true, 8 >, [578](#)
- Simd128i\_base, [580](#)
- Simd256\_impl< true, false, true, 8 >, [586](#)
- Simd256\_impl< true, true, false, 4 >, [612](#)
- Simd256\_impl< true, true, true, 4 >, [643](#)
- Simd512\_impl< true, true, false, 8 >, [668](#)
- Simd512\_impl< true, true, true, 8 >, [676](#)
- Simd512i\_base, [678](#)
- Simd128\_impl< true, true, false, 8 >, [555](#)
- Simd128\_impl< true, true, true, 2 >, [563](#)
- Simd128\_impl< true, true, true, 4 >, [571](#)
- Simd128\_impl< true, true, true, 8 >, [579](#)
- Simd128i\_base, [581](#)
- Simd256\_impl< true, false, true, 8 >, [586](#)
- Simd256\_impl< true, true, false, 4 >, [612](#)
- Simd256\_impl< true, true, true, 4 >, [643](#)
- Simd512\_impl< true, true, false, 8 >, [668](#)
- Simd512\_impl< true, true, true, 8 >, [676](#)
- Simd512i\_base, [678](#)
- VEC\_ADD
  - FFLAS::vectorised, [267](#)
- VEC\_SUB
  - FFLAS::vectorised, [267](#)
- vect\_size
  - FieldSimd< \_Field >, [419](#)
  - NoSimd< T >, [486](#)
  - Simd128\_impl< true, true, false, 2 >, [538](#)
  - Simd128\_impl< true, true, false, 4 >, [546](#)
  - Simd128\_impl< true, true, false, 8 >, [556](#)
  - Simd128\_impl< true, true, true, 2 >, [563](#)
  - Simd128\_impl< true, true, true, 4 >, [571](#)
  - Simd128\_impl< true, true, true, 8 >, [579](#)
  - Simd256\_impl< true, false, true, 8 >, [587](#)
  - Simd256\_impl< true, true, false, 2 >, [596](#)
  - Simd256\_impl< true, true, false, 4 >, [612](#)
  - Simd256\_impl< true, true, false, 8 >, [621](#)
  - Simd256\_impl< true, true, true, 2 >, [629](#)
  - Simd256\_impl< true, true, true, 4 >, [643](#)
  - Simd256\_impl< true, true, true, 8 >, [651](#)
  - Simd512\_impl< true, false, true, 8 >, [658](#)
  - Simd512\_impl< true, true, false, 8 >, [668](#)
  - Simd512\_impl< true, true, true, 8 >, [677](#)
- vect\_t
  - FieldSimd< \_Field >, [414](#)
  - NoSimd< T >, [486](#)
  - Simd128\_impl< true, true, false, 2 >, [531](#)
  - Simd128\_impl< true, true, false, 4 >, [540](#)
  - Simd128\_impl< true, true, false, 8 >, [549](#)
  - Simd128\_impl< true, true, true, 2 >, [558](#)
  - Simd128\_impl< true, true, true, 4 >, [565](#)
  - Simd128\_impl< true, true, true, 8 >, [573](#)
  - simd128\_int64.inl, [809](#)
  - Simd128i\_base, [580](#)
  - Simd256\_impl< true, false, true, 8 >, [583](#)
  - Simd256\_impl< true, true, false, 2 >, [590](#)
  - Simd256\_impl< true, true, false, 4 >, [600](#)
  - Simd256\_impl< true, true, false, 8 >, [614](#)
  - Simd256\_impl< true, true, true, 2 >, [623](#)
  - Simd256\_impl< true, true, true, 4 >, [632](#)
  - Simd256\_impl< true, true, true, 8 >, [645](#)
  - simd256\_int64.inl, [811](#)
  - Simd256i\_base, [652](#)
  - Simd512\_impl< true, false, true, 8 >, [654](#)
  - Simd512\_impl< true, true, false, 8 >, [661](#)
  - Simd512\_impl< true, true, true, 8 >, [670](#)
  - simd512\_int64.inl, [813](#)

- Simd512i\_base, 678
- verification\_PLUQ
  - benchmark-pluq.C, 738
- verifPLUQ
  - test-lu.C, 1020
- VERSION
  - config.h, 757
- vor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 519
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 523
  - Simd128\_impl< true, true, false, 2 >, 537
  - Simd128\_impl< true, true, false, 4 >, 546
  - Simd128\_impl< true, true, false, 8 >, 555
  - Simd128\_impl< true, true, true, 2 >, 563
  - Simd128\_impl< true, true, true, 4 >, 571
  - Simd128\_impl< true, true, true, 8 >, 579
  - Simd128i\_base, 581
  - Simd256\_impl< true, false, true, 8 >, 586
  - Simd256\_impl< true, true, false, 4 >, 612
  - Simd256\_impl< true, true, true, 4 >, 643
  - Simd512\_impl< true, true, false, 8 >, 668
  - Simd512\_impl< true, true, true, 8 >, 676
  - Simd512i\_base, 678
- vxor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 519
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 523
  - Simd128\_impl< true, true, false, 2 >, 537
  - Simd128\_impl< true, true, false, 4 >, 546
  - Simd128\_impl< true, true, false, 8 >, 555
  - Simd128\_impl< true, true, true, 2 >, 563
  - Simd128\_impl< true, true, true, 4 >, 571
  - Simd128\_impl< true, true, true, 8 >, 579
  - Simd128i\_base, 581
  - Simd256\_impl< true, false, true, 8 >, 586
  - Simd256\_impl< true, true, false, 4 >, 612
  - Simd256\_impl< true, true, true, 4 >, 643
  - Simd512\_impl< true, true, false, 8 >, 668
  - Simd512\_impl< true, true, true, 8 >, 676
  - Simd512i\_base, 678
- WAIT
  - parallel.h, 961
- Winograd, 710
  - FFLAS::BLAS3, 183
- winograd.C, 714
  - balanced, 715
  - DOUBLE\_TO\_FLOAT\_CROSSOVER, 715
  - GFOPS, 715
  - main, 715
  - TTimer, 715
- Winograd\_L\_S
  - FFLAS::BLAS3, 187
- Winograd\_LR\_S
  - FFLAS::BLAS3, 186
- Winograd\_R\_S
  - FFLAS::BLAS3, 187
- WinogradAcc\_2\_24
  - FFLAS::BLAS3, 185
- WinogradAcc\_2\_27
  - FFLAS::BLAS3, 185
- WinogradAcc\_3\_21
  - FFLAS::BLAS3, 184
- WinogradAcc\_3\_23
  - FFLAS::BLAS3, 184
- WinogradAcc\_L\_S
  - FFLAS::BLAS3, 186
- WinogradAcc\_LR
  - FFLAS::BLAS3, 185
- WinogradAcc\_R\_S
  - FFLAS::BLAS3, 186
- WinogradCalc
  - FFLAS::Protected, 207
- WinogradPar, 710
- WinogradSteps
  - FFLAS::Protected, 205
- WinogradThreshold
  - FFLAS::Protected, 205
- WinoPar
  - FFLAS::BLAS3, 183
- WINOTHRESHOLD
  - fflas.h, 763
- WRITE
  - parallel.h, 962
- write
  - RNSInteger< RNS >, 509, 510
  - RNSIntegerMod< RNS >, 515
- write\_field
  - Matio.h, 977
- write\_matrix
  - benchmark-fgemv-mp.C, 727
  - RNSIntegerMod< RNS >, 515
- write\_matrix\_long
  - RNSIntegerMod< RNS >, 515
- writeCommandString
  - FFLAS, 177
- writeDnsFormat
  - FFLAS, 144
- WriteMatrix
  - FFLAS, 177, 178
- WritePermutation
  - FFLAS, 179
- zero
  - FFLAS, 74
  - FieldSimd< \_Field >, 417
  - RNSInteger< RNS >, 510
  - RNSIntegerMod< RNS >, 517
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 519
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 523

[Simd128\\_impl< true, true, false, 2 >, 537](#)  
[Simd128\\_impl< true, true, false, 4 >, 546](#)  
[Simd128\\_impl< true, true, false, 8 >, 555](#)  
[Simd128\\_impl< true, true, true, 2 >, 563](#)  
[Simd128\\_impl< true, true, true, 4 >, 570](#)  
[Simd128\\_impl< true, true, true, 8 >, 578](#)  
[Simd128i\\_base, 580](#)  
[Simd256\\_impl< true, false, true, 8 >, 583](#)  
[Simd256\\_impl< true, true, false, 2 >, 596](#)  
[Simd256\\_impl< true, true, false, 4 >, 612](#)  
[Simd256\\_impl< true, true, false, 8 >, 621](#)  
[Simd256\\_impl< true, true, true, 2 >, 629](#)  
[Simd256\\_impl< true, true, true, 4 >, 643](#)  
[Simd256\\_impl< true, true, true, 8 >, 651](#)  
[Simd256i\\_base, 652](#)  
[Simd512\\_impl< true, false, true, 8 >, 654](#)  
[Simd512\\_impl< true, true, false, 8 >, 668](#)  
[Simd512\\_impl< true, true, true, 8 >, 676](#)  
[Simd512i\\_base, 678](#)  
[ZOSparseMatrix](#)  
[FFLAS, 70](#)