FflasFfpack

Generated by Doxygen 1.9.6

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 specifities 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

Chapter 2

Configuring and Installing FFLAS-FFPACK

FFLAS-FFPACK is a header-only package.

Hower 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.

Chapter 3

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`

Chapter 4

Tutorial

no doc.

Chapter 5

Architecture of the library.

no doc.

Chapter 6

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

use cblas_(d)scal when possible

Global FFLAS::freduce (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 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::fscalin (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::fscalin (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 ldX 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 ldX equal

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

does not belong here

Chapter 7

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 ftrsmLeftUpperNoTransNonUnit< Element >

Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

Chapter 8

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 triagular 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 triagular 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::fscalin (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::fscalin (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::MatF2MatFl_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_redcolechelon (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_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 lda

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 lda

Chapter 9

Module Index

9.1 Modules

Here is a list of all modules:

Chapter 10

Namespace Index

10.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 11

Hierarchical Index

11.1 Class Hierarchy

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

Chapter 12

Data Structure Index

12.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 13

File Index

13.1 File List

Here is a list of all files with brief descriptions:

Chapter 14

Module Documentation

14.1 CHECKER

Class CHECKER provides functions to verify computations in FFLAS and FFPACK.

Class CHECKER provides functions to verify computations in FFLAS and FFPACK.

14.2 FFLAS-FFPACK

the FFLAS FFPACK library

Modules

- FFLAS
  The C-style wrapper of BLAS for finite field linear algebra.
- Interfaces
  Intefaces for FFLAS-FFPACK.

14.2.1 Detailed Description

the FFLAS FFPACK library

C++ header library for fast exact dense linear algebra

See also

FFLAS
FFPACK

14.3 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.

14.4 Matrix Multiplication Algorithms

Matrix Multiplication (level 3) algorithms.

Files

- file schedule_bini.inl

  Bini implementation.

14.4.1 Detailed Description

Matrix Multiplication (level 3) algorithms.

Todo biblio

14.5 SIMD wrapper

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

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

Todo biblio

14.6 FFPACK

Class FFPACK provides functions using fflas much as Lapack uses BLAS.

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

14.6.1 Detailed Description

Class FFPACK provides functions using fflas much as Lapack uses BLAS.

14.7 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

14.7.1 Detailed Description

fields in the FFLAS-FFPACK library

Unparametric/Random elements

Todo biblio

14.8 RNS

just include them all

just include them all

14.9 Interfaces

Intefaces for FFLAS-FFPACK.

Intefaces for FFLAS-FFPACK.

C interface in folder

See also

libs

Chapter 15

Namespace Documentation

15.1 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 fgemm 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_ftrsm
- 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< RecInt::rint< K > >
- struct ElementTraits< RecInt::rmint< K, MG > >
- struct ElementTraits< RecInt::ruint< K > >
- struct ElementTraits< uint16_t >
- struct ElementTraits< uint32_t >
- struct ElementTraits< uint64_t >
- struct ElementTraits< uint8_t >
- struct EllMat
- 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< RecInt::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< RecInt::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_copy↩_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 ldx, 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, 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 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 GE**neral **M**atrix **M**ultiply.*
- 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 >
  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)

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

  freduce $x \leftarrow xmodF$.
- 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 Initializes X *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 $A \leftarrow AmodF$.
- 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)
- 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)

  *freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*
- template<class Field >
  void fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- 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 fscalin (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↩ Element a, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void fscalin (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 fscalin (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)

  fscalin $A \leftarrow a \cdot A$.
- 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<> void fscalin (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 fscalin (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 fscalin (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 fscalin (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: **TR**iangular **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**iangular **M**atrix **M**ultiply with 3 operands Computes* $C \leftarrow \alpha \mathrm{op}(A)B + betaC$ *or* $C \leftarrow \alpha B\mathrm{op}(A) + betaC$.
- 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 cblas_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 $X \leftarrow \mathrm{op}(A^{-1})X$
- 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 $x \leftarrow ymodF$.
- 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 $x \leftarrow ymodF$.
- 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* $F\$$.

- 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 $A \leftarrow BmodF$.
- 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 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 >
  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 $X \leftarrow \mathrm{op}(A)X$
- 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: **TR**iangular **S**ystem solve with **M**atrix.*
- 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, typenameField::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)

  freduce $x \leftarrow x mod F$.
- 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)

  freduce $x \leftarrow y mod F$.
- 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)

  finit $x \leftarrow y mod F$.
- 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)

  fconvert $x \leftarrow y mod F$.
- template INST_OR_DECL void fnegin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)

  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)

  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 fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- 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)

  faxpby : $y \leftarrow \alpha \cdot x + \beta \cdot y$.
- 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)
- 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 $A \leftarrow AmodF$.
- 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 $A \leftarrow BmodF$.
- 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 $A \leftarrow BmodF$.
- 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 fscalin (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)

  fscalin $A \leftarrow a \cdot A$.
- 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 ftrsv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, int incX)

  ftrsv: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$
- 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: **TR**iangular **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: **TR**iangular **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: **F**ield **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.
- 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 &ifs, FFLAS_FORMAT &format)
- 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.

- 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 ()

15.1.1 Typedef Documentation

15.1.1.1 Checker_fgemm

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

15.1.1.2 Checker_ftrsm

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

15.1.1.3 ForceCheck_fgemm

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

15.1.1.4 ForceCheck_ftrsm

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

15.1.1.5 ZOSparseMatrix

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

15.1.1.6 NotZOSparseMatrix

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

15.1.1.7 SimdSparseMatrix

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

15.1.1.8 NoSimdSparseMatrix

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

15.1.1.9 MKLSparseMatrixFormat

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

15.1.1.10 NotMKLSparseMatrixFormat

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

15.1.1.11 has_plus

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

15.1.1.12 has_minus

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

15.1.1.13 has_equal

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

15.1.1.14 has_plus_eq

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

15.1.1.15 has_minus_eq

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

15.1.1.16 has_mul

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

15.1.1.17 has_mul_eq

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

15.1.1.18 Timer

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

15.1.1.19 BaseTimer

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

15.1.1.20 UserTimer

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

15.1.1.21 SysTimer

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

15.1.2 Enumeration Type Documentation

15.1.2.1 FFLAS_ORDER

```
enum FFLAS_ORDER
```

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major

15.1.2.2 FFLAS_TRANSPOSE

```
enum FFLAS_TRANSPOSE
```

Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.

15.1.2.3 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 $T_{i,j} = 0$)

15.1.2.4 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 ($T_{i,i} = 1$)

15.1.2.5 FFLAS_SIDE

enum FFLAS_SIDE

On what side ?

Enumerator

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

15.1.2.6 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

15.1.2.7 number_kind

```
enum number_kind
```

Enumerator

zero	
one	
mone	
other	

15.1.2.8 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	

15.1.2.9 FFLAS_FORMAT

```
enum FFLAS_FORMAT
```

Enumerator

FflasAuto	
FflasDense	
FflasSMS	
FflasBinary	
FflasMath	
FflasMaple	
FflasSageMath	

15.1.3 Function Documentation

15.1.3.1 InfNorm()

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

15.1.3.2 min3()

```
const T & min3 (
            const T & m,
            const T & n,
            const T & k )
```

15.1.3.3 max3()

```
const T & max3 (
            const T & m,
            const T & n,
            const T & k )
```

15.1.3.4 min4()

```
const T & min4 (
            const T & m,
            const T & n,
            const T & k,
            const T & l )
```

15.1.3.5 max4()

```
const T & max4 (
            const T & m,
            const T & n,
            const T & k,
            const T & l )
```

15.1.3.6 fadd() [1/8]

```
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 )
```

15.1.3.7 faddin() [1/4]

```
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 )
```

15.1.3.8 fsub() [1/4]

```
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 )
```

15.1.3.9 fsubin() [1/3]

```
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 )
```

15.1.3.10 fadd() [2/8]

```
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

15.1.3.11 pfadd()

```
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 )
```

15.1.3.12 pfsub()

```
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 )
```

15.1.3.13 pfaddin()

```
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 )
```

15.1.3.14 pfsubin()

```
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 )
```

15.1.3.15 fadd() [3/8]

```
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

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

15.1.3.16 fsub() [2/4]

```
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

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

15.1.3.17 faddin() [2/4]

```
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

15.1.3.18 fsubin() [2/3]

```
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

15.1.3.19 fadd() [4/8]

```
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 + alpha B.

Parameters

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

15.1.3.20 fassign() [1/10]

```
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 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

15.1.3.21 fassign() [2/10]

```
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]
```

15.1.3.22 fassign() [3/10]

```
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]
```

15.1.3.23 fassign() [4/10]

```
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]
```

15.1.3.24 fassign() [5/10]

```
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]
```

15.1.3.25 fassign() [6/10]

```
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]
```

15.1.3.26 fassign() [7/10]

```
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]
```

15.1.3.27 fassign() [8/10]

```
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 : $A \leftarrow B$.

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

15.1.3.28 faxpy() [1/6]

```
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]
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

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

15.1.3.29 faxpy() [2/6]

```
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]
```

15.1.3.30 faxpy() [3/6]

```
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]
```

15.1.3.31 faxpy() [4/6]

```
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]
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

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

15.1.3.32 fdot() [1/11]

```
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]
```

15.1.3.33 fdot() [2/11]

```
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]
```

15.1.3.34 fdot() [3/11]

```
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]
```

15.1.3.35 fdot() [4/11]

```
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]
```

15.1.3.36 fdot() [5/11]

```
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]
```

15.1.3.37 fdot() [6/11]

```
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]
```

15.1.3.38 fdot() [7/11]

```
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]
```

15.1.3.39 fdot() [8/11]

```
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

15.1.3.40 fgemm() [1/23]

```
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]
```

15.1.3.41 fgemm() [2/23]

```
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]
```

15.1.3.42 fgemm() [3/23]

```
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]
```

15.1.3.43 fgemm() [4/23]

```
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: **F**ield **GE**neral **M**atrix **M**ultiply.

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

Parameters

F	field.
ta	if `ta==FflasTrans` then $\mathrm{op}(A) = A^t$, else $\mathrm{op}(A) = A$,
tb	same for matrix `B`
m	see `A`
n	see `B`
k	see `A`
alpha	scalar
beta	scalar
A	$\mathrm{op}(A)$ is $m \times k$
B	$\mathrm{op}(B)$ is $k \times n$

Parameters

C	C is $m \times n$
lda	leading dimension of A
ldb	leading dimension of B
ldc	leading dimension of C
w	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of w.

Warning

α *must* be invertible

15.1.3.44 fgemm() [5/23]

```
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]
```

15.1.3.45 fgemm() [6/23]

```
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]
```

15.1.3.46 fsquare() [1/6]

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

Parameters

ta	if ta==FflasTrans, $\mathrm{op}(A) = A^T$.
F	field
n	size of A
alpha	scalar
beta	scalar
A	dense matrix of size nxn
lda	leading dimension of A
C	dense matrix of size nxn
ldc	leading dimension of C

Bug why double ?

15.1.3.47 fsquare() [2/6]

```
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]
```

15.1.3.48 fsquare() [3/6]

```
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]
```

15.1.3.49 fsquare() [4/6]

```
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]
```

15.1.3.50 fsquare() [5/6]

```
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]
```

15.1.3.51 fgemm() [7/23]

```
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]
```

15.1.3.52 fgemm() [8/23]

```
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]
```

15.1.3.53 fgemm() [9/23]

```
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]
```

15.1.3.54 fgemm() [10/23]

```
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]
```

15.1.3.55 fgemm() [11/23]

```
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]
```

15.1.3.56 fgemm() [12/23]

```
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]
```

15.1.3.57 fgemm() [13/23]

```
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]
```

15.1.3.58 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]
```

15.1.3.59 fgemm() [15/23]

```
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::Auto, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag >, ParSeq > & H )  [inline]
```

15.1.3.60 fgemm() [16/23]

```
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]
```

15.1.3.61 fgemm() [17/23]

```
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]
```

15.1.3.62 fgemm() [18/23]

```
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]
```

15.1.3.63 fgemv() [1/19]

```
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]
```

15.1.3.64 fgemv() [2/19]

```
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]
```

15.1.3.65 fgemv() [3/19]

```
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]
```

15.1.3.66 fgemv() [4/19]

```
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]
```

15.1.3.67 fgemv() [5/19]

```
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 $Y \leftarrow \alpha \mathrm{op}(A)X + \beta Y$.

Parameters

	F	field
	TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
	M	rows
	N	cols
	alpha	scalar
	A	dense matrix of size `MxN`
	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`

15.1.3.68 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]
```

15.1.3.69 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]
```

15.1.3.70 fgemv() [8/19]

```
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]
```

15.1.3.71 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]
```

15.1.3.72 fgemv() [10/19]

```
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 )
```

15.1.3.73 fgemv() [11/19]

```
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 )
```

15.1.3.74 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::Defau
> & H ) [inline]
```

15.1.3.75 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]
```

15.1.3.76 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]
```

15.1.3.77 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]
```

15.1.3.78 fgemv() [16/19]

```
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]
```

15.1.3.79 fger() [1/12]

```
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
	alpha	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

15.1.3.80 fger() [2/12]

```
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]
```

15.1.3.81 fger() [3/12]

```
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]
```

15.1.3.82 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]
```

15.1.3.83 fger() [5/12]

```
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]
```

15.1.3.84 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]
```

15.1.3.85 fger() [7/12]

```
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]
```

15.1.3.86 fger() [8/12]

```
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]
```

15.1.3.87 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]
```

15.1.3.88 fger() [10/12]

```
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]
```

15.1.3.89 fger() [11/12]

```
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]
```

15.1.3.90 freduce() [1/10]

```
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 $x \leftarrow y mod F$.

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

15.1.3.91 freduce() [2/10]

```
void freduce (
            const Field & F,
            const size_t n,
            typename Field::Element_ptr X,
            const size_t incX )
```

freduce $x \leftarrow x mod F$.

Parameters

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

Bug use cblas_(d)scal when possible

15.1.3.92 freduce_constoverride() [1/2]

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

15.1.3.93 finit() [1/8]

```
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 )
```

15.1.3.94 finit() [2/8]

```
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

15.1.3.95 freduce() [3/10]

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

freduce $A \leftarrow AmodF$.

Parameters

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

15.1.3.96 pfreduce()

```
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 )
```

15.1.3.97 freduce() [4/10]

```
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 $A \leftarrow BmodF$.

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

15.1.3.98 freduce_constoverride() [2/2]

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

15.1.3.99 finit() [3/8]

```
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 $A \leftarrow BmodF$.

Parameters

F	field
m	number of rows

Parameters

n	number of cols
A	matrix in `F`
lda	stride of `A`
B	matrix in `OtherElement`
ldb	stride of `B`

15.1.3.100 finit() [4/8]

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

15.1.3.101 freduce() [5/10]

```
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]
```

15.1.3.102 freduce() [6/10]

```
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]
```

15.1.3.103 freivalds()

```
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: **F**reivalds **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==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`
alpha	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`

15.1.3.104 fscalin() [1/10]

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

fscalin $x \leftarrow \alpha \cdot x$.

Parameters

F	field

Parameters

n	size of the vectors
alpha	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.

15.1.3.105 fscal() [1/10]

```
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
	alpha	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.

15.1.3.106 fscal() [2/10]

```
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]
```

15.1.3.107 fscal() [3/10]

```
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]
```

15.1.3.108 fscalin() [2/10]

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

15.1.3.109 fscalin() [3/10]

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

15.1.3.110 fscalin() [4/10]

```
void fscalin (
            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]
```

fscalin $A \leftarrow a \cdot A$.

Parameters

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

15.1.3.111 fscal() [4/10]

```
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]
```

fscal $B \leftarrow a \cdot A$.

Parameters

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

15.1.3.112 fscalin() [5/10]

```
void fscalin (
            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]
```

15.1.3.113 fscal() [5/10]

```
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]
```

15.1.3.114 fscalin() [6/10]

```
void fscalin (
            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]
```

15.1.3.115 fscal() [6/10]

```
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]
```

15.1.3.116 fscalin() [7/10]

```
void fscalin (
            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]
```

15.1.3.117 fscal() [7/10]

```
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]
```

15.1.3.118 fscalin() [8/10]

```
void fscalin (
        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]
```

15.1.3.119 fscal() [8/10]

```
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]
```

15.1.3.120 fsyr2k()

```
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

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

Warning

α *must* be invertible

15.1.3.121 fsyrk() [1/5]

```
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

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

Warning

α *must* be invertible

15.1.3.122 fsyrk() [2/5]

```
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

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

Warning

α *must* be invertible

15.1.3.123 fsyrk() [3/5]

```
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]
```

15.1.3.124 fsyrk() [4/5]

```
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]
```

15.1.3.125 fsyrk() [5/5]

```
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 $C = \alpha A \times DeltaD \times A^T + \beta C$ or $C = \alpha A^T \times DeltaD \times A + \beta C$ 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

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

Warning

α *must* be invertible

15.1.3.126 ftrmm() [1/3]

```
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 $B \leftarrow \alpha \mathrm{op}(A) B$ or $B \leftarrow \alpha B \mathrm{op}(A)$.

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A) B$ is computed.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is implicitly unit.
M	rows of `B`
N	cols of `B`
alpha	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 `MxN`
ldb	leading dim of `B`

15.1.3.127 ftrmm() [2/3]

```
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**iangular **M**atrix **M**ultiply with 3 operands Computes $C \leftarrow \alpha \mathrm{op}(A) B + beta C$ or $C \leftarrow \alpha B \mathrm{op}(A) + beta C$.

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A) B$ is computed.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is implicitly unit.
M	rows of `B`
N	cols of `B`
alpha	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 `MxN`

Parameters

ldb	leading dim of B
beta	scalar
C	matrix of size MxN
ldc	leading dim of C

15.1.3.128 ftrsm() [1/9]

```
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]
```

15.1.3.129 ftrsm() [2/9]

```
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]
```

15.1.3.130 ftrsm() [3/9]

```
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]
```

15.1.3.131 ftrsm() [4/9]

```
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]
```

15.1.3.132 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]
```

15.1.3.133 cblas_imptrsm()

```
void cblas_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 ) [inline]
```

15.1.3.134 ftrsv() [1/2]

```
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 $X \leftarrow \operatorname{op}(A^{-1})X$

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 $\operatorname{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular

15.1.3.135 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]
```

15.1.3.136 finit() [5/8]

```
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 $x \leftarrow ymodF$.

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

15.1.3.137 fconvert() [1/3]

```
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 $x \leftarrow ymodF$.

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

15.1.3.138 fnegin() [1/4]

```
void fnegin (
            const Field & F,
            const size_t n,
            typename Field::Element_ptr X,
            const size_t incX )
```

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

15.1.3.139 fneg() [1/4]

```
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 $x \leftarrow -y$.

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

15.1.3.140 fzero() [1/4]

```
void fzero (
            const Field & F,
            const size_t n,
            typename Field::Element_ptr 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

15.1.3.141 frand() [1/2]

```
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

15.1.3.142 fiszero() [1/4]

```
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

15.1.3.143 fequal() [1/4]

```
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

15.1.3.144 faxpby() [1/2]

```
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

15.1.3.145 fdot() [9/11]

```
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]
```

15.1.3.146 fswap() [1/2]

```
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

15.1.3.147 fzero() [2/4]

```
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

15.1.3.148 frand() [2/2]

```
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 : $A \leftarrow random$.

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

15.1.3.149 fequal() [2/4]

```
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

15.1.3.150 fiszero() [2/4]

```
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

Parameters

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

15.1.3.151 fidentity() [1/4]

```
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

15.1.3.152 fidentity() [2/4]

```
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

15.1.3.153 finit() [6/8]

```
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$`.

Parameters

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

15.1.3.154 fconvert() [2/3]

```
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 $A \leftarrow BmodF$.

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

15.1.3.155 fnegin() [2/4]

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

fnegin $A \leftarrow -A$.

Parameters

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

Todo check if n == lda

15.1.3.156 fneg() [2/4]

```
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 $A \leftarrow -B$.

Parameters

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

Todo check if n == lda

15.1.3.157 faxpby() [2/2]

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

Parameters

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

Note

this is a catlas function

15.1.3.158 fmove() [1/2]

```
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 $B \leftarrow 0$.

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`

15.1.3.159 bitsize()

```
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 `MxN`
lda	leading dimension of `A`

15.1.3.160 bitsize< Givaro::ZRing< Givaro::Integer > >()

```
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]
```

15.1.3.161 ftrmv()

```
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 $X \leftarrow \mathrm{op}(A)X$

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 and columns of `A`
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^T$.
Diag	if `Diag==FflasUnit` then `A` is unit diagonal.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular

15.1.3.162 ftrsm() [6/9]

```
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**iangular **S**ystem solve with **M**atrix.

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

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A^{-1})B$ is computed.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular invertible 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 `MxN`
ldb	leading dim of `B`

Bug α must be non zero.

15.1.3.163 pfgemm() [1/7]

```
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 )
```

15.1.3.164 pfgemm_1D_rec()

```
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 )
```

15.1.3.165 pfgemm_2D_rec()

```
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 )
```

15.1.3.166 pfgemm_3D_rec()

```
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 )
```

15.1.3.167 pfgemm_3D_rec2()

```
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 )
```

15.1.3.168 fgemm() [19/23]

```
std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag
> >::value, typenameField::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]
```

15.1.3.169 ftrsm() [7/9]

```
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]
```

15.1.3.170 ftrsm() [8/9]

```
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]
```

15.1.3.171 fspmv() [1/2]

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

15.1.3.172 fspmm()

```
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]
```

15.1.3.173 sparse_init() [1/16]

```
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]
```

15.1.3.174 sparse_init() [2/16]

```
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]
```

15.1.3.175 sparse_delete() [1/12]

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

15.1.3.176 sparse_delete() [2/12]

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

15.1.3.177 sparse_init() [3/16]

```
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]
```

15.1.3.178 sparse_init() [4/16]

```
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]
```

15.1.3.179 sparse_delete() [3/12]

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

15.1.3.180 sparse_delete() [4/12]

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

15.1.3.181 sparse_print() [1/3]

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

15.1.3.182 sparse_init() [5/16]

```
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]
```

15.1.3.183 sparse_init() [6/16]

```
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]
```

15.1.3.184 sparse_init() [7/16]

```
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]
```

15.1.3.185 sparse_init() [8/16]

```
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]
```

15.1.3.186 sparse_delete() [5/12]

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

15.1.3.187 sparse_init() [9/16]

```
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]
```

15.1.3.188 sparse_init() [10/16]

```
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]
```

15.1.3.189 sparse_init() [11/16]

```
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]
```

15.1.3.190 sparse_delete() [6/12]

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

15.1.3.191 sparse_delete() [7/12]

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

15.1.3.192 sparse_init() [12/16]

```
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]
```

15.1.3.193 sparse_init() [13/16]

```
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]
```

15.1.3.194 sparse_delete() [8/12]

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

15.1.3.195 sparse_delete() [9/12]

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

15.1.3.196 sparse_print() [2/3]

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

15.1.3.197 sparse_delete() [10/12]

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

15.1.3.198 sparse_init() [14/16]

```
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]
```

15.1.3.199 operator<<()

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

15.1.3.200 readSmsFormat()

```
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 )
```

15.1.3.201 readSprFormat()

```
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 )
```

15.1.3.202 getDataType() [1/4]

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

15.1.3.203 getDataType() [2/4]

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

15.1.3.204 getDataType() [3/4]

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

15.1.3.205 getDataType() [4/4]

```
int getDataType ( )
```

15.1.3.206 readMachineType()

```
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 )
```

15.1.3.207 readDnsFormat()

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

15.1.3.208 writeDnsFormat()

```
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 )
```

15.1.3.209 fspmv() [2/2]

```
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]
```

15.1.3.210 sparse_delete() [11/12]

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

15.1.3.211 sparse_delete() [12/12]

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

15.1.3.212 sparse_print() [3/3]

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

15.1.3.213 sparse_init() [15/16]

```
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]
```

15.1.3.214 sparse_init() [16/16]

```
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]
```

15.1.3.215 computeDeviation()

```
double computeDeviation (
            It begin,
            It end )
```

15.1.3.216 getStat()

```
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 )
```

15.1.3.217 fflas_delete() [1/4]

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

15.1.3.218 fflas_delete() [2/4]

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

15.1.3.219 fflas_new() [1/7]

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

15.1.3.220 fflas_new() [2/7]

```
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]
```

15.1.3.221 finit_rns() [1/2]

```
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 )
```

15.1.3.222 finit_trans_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 )
```

15.1.3.223 fconvert_rns() [1/2]

```
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 )
```

15.1.3.224 fconvert_trans_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 )
```

15.1.3.225 fflas_new() [3/7]

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

15.1.3.226 fflas_new() [4/7]

```
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]
```

15.1.3.227 finit_rns() [2/2]

```
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 )
```

15.1.3.228 fconvert_rns() [2/2]

```
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 )
```

15.1.3.229 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 )
```

freduce $x \leftarrow xmodF$.

Parameters

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

Bug use cblas_(d)scal when possible

15.1.3.230 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 $x \leftarrow y mod F$.

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

15.1.3.231 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 $x \leftarrow y mod F$.

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

15.1.3.232 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 )
```

fconvert $x \leftarrow ymodF$.

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

15.1.3.233 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 )
```

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

15.1.3.234 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 )
```

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

15.1.3.235 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

15.1.3.236 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

15.1.3.237 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

15.1.3.238 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

15.1.3.239 fscalin() [9/10]

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

fscalin $x \leftarrow \alpha \cdot x$.

Parameters

F	field
n	size of the vectors
alpha	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.

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

Parameters

	F	field
	n	size of the vectors
	alpha	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.

15.1.3.241 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 )
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

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

15.1.3.242 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 )
```

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

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

15.1.3.243 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

15.1.3.244 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 )
```

15.1.3.245 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 )
```

15.1.3.246 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 )
```

15.1.3.247 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 )
```

15.1.3.248 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

15.1.3.249 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

15.1.3.250 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

15.1.3.251 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 x n matrix in F
lda	leading dimension of A

15.1.3.252 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

15.1.3.253 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

15.1.3.254 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 $A \leftarrow A mod F$.

Parameters

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

15.1.3.255 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 )
```

freduce $A \leftarrow BmodF$.

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`

15.1.3.256 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 )
```

finit $A \leftarrow BmodF$.

Parameters

F	field

Parameters

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

15.1.3.257 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 )
```

fnegin $A \leftarrow -A$.

Parameters

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

15.1.3.258 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 )
```

fneg $A \leftarrow -B$.

Parameters

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

15.1.3.259 fscalin() [10/10]

```
template INST_OR_DECL void fscalin (
            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 )
```

fscalin $A \leftarrow a \cdot A$.

Parameters

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

15.1.3.260 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 )
```

fscal $B \leftarrow a \cdot A$.

Parameters

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

15.1.3.261 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 )
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

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

15.1.3.262 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
	alpha	scalar
in	*X*	vector in F
	ldx	leading dimension of X
	beta	scalar
in,out	*Y*	vector in 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 `F`
lda	stride of `A`
B	vector in `F`
ldb	stride of `B`

15.1.3.263 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

F	field
M	rows
N	cols
A	dense matrix of size `MxN`
lda	leading dimension of `A`
B	dense matrix of size `MxN`
ldb	leading dimension of `B`
C	dense matrix of size `MxN`
ldc	leading dimension of `C`

15.1.3.264 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

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

15.1.3.265 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

15.1.3.266 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 + alpha B.

Parameters

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

15.1.3.267 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

15.1.3.268 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 $Y \leftarrow \alpha \mathrm{op}(A) X + \beta Y$.

Parameters

	F	field
	TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
	M	rows
	N	cols
	alpha	scalar
	A	dense matrix of size `MxN`
	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`

15.1.3.269 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
	alpha	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`

15.1.3.270 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 $X \leftarrow \mathrm{op}(A^{-1})X$

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 $\mathrm{op}(A) = A^t$.
Diag	if Diag==FflasUnit then A is unit.
Uplo	if Uplo==FflasUpper then A is upper triangular

15.1.3.271 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: **TR**iangular **S**ystem solve with **M**atrix.

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

Parameters

F	field
Side	if Side==FflasLeft then $B \leftarrow \alpha \mathrm{op}(A^{-1})B$ is computed.

Parameters

Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular invertible 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 `MxN`
ldb	leading dim of `B`

Bug α must be non zero.

15.1.3.272 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 )
```

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

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

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A) B$ is computed.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is implicitly unit.
M	rows of `B`
N	cols of `B`
alpha	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 `MxN`
ldb	leading dim of `B`

15.1.3.273 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 **GE**neral **M**atrix **M**ultiply.

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

Parameters

F	field.
ta	if `ta==FflasTrans` then $\mathrm{op}(A) = A^t$, else $\mathrm{op}(A) = A$,
tb	same for matrix `B`
m	see `A`
n	see `B`
k	see `A`
alpha	scalar
beta	scalar
A	$\mathrm{op}(A)$ is $m \times k$
B	$\mathrm{op}(B)$ 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`
w	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of `w`.

Warning

α *must* be invertible

15.1.3.274 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 )
```

15.1.3.275 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 )
```

15.1.3.276 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 )
```

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

Parameters

ta	if `ta==FflasTrans`, $\mathrm{op}(A) = A^T$.
F	field
n	size of `A`
alpha	scalar
beta	scalar
A	dense matrix of size `nxn`
lda	leading dimension of `A`
C	dense matrix of size `nxn`
ldc	leading dimension of `C`

15.1.3.278 BlockCuts() [1/2]

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

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

```
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]
```

15.1.3.280 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()

```
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]
```

15.1.3.281 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()

```
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]
```

15.1.3.282 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()

```
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]
```

15.1.3.283 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()

```
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]
```

15.1.3.284 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()

```
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]
```

15.1.3.285 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()

```
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]
```

15.1.3.286 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()

```
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]
```

15.1.3.287 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()

```
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]
```

15.1.3.288 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()

```
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]
```

15.1.3.289 BlockCuts() [2/2]

```
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 )  [inline]
```

15.1.3.290 pfzero()

```
void pfzero (
            const Field & F,
            size_t m,
            size_t n,
            typename Field::Element_ptr C,
            size_t BS = 0 )
```

15.1.3.291 pfrand()

```
void pfrand (
            const Field & F,
            RandIter & G,
            size_t m,
            size_t n,
            typename Field::Element_ptr C,
            size_t BS = 0 )
```

15.1.3.292 fdot() [11/11]

```
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]
```

15.1.3.293 pfgemm() [2/7]

```
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 )
```

15.1.3.294 pfgemm() [3/7]

```
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 )
```

15.1.3.295 pfgemm() [4/7]

```
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 )
```

15.1.3.296 pfgemm() [5/7]

```
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 )
```

15.1.3.297 pfgemm() [6/7]

```
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 )
```

15.1.3.298 pfgemm() [7/7]

```
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 )
```

15.1.3.299 fgemv() [18/19]

```
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 )
```

15.1.3.300 fgemv() [19/19]

```
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 )
```

15.1.3.301 parseArguments()

```
void parseArguments (
            int argc,
            char ** argv,
            Argument * args,
            bool printDefaults = true )
```

15.1.3.302 writeCommandString()

```
std::ostream & writeCommandString (
            std::ostream & os,
            Argument * args,
            const char * programName = nullptr )
```

writes the values of all arguments, preceded by the programName

15.1.3.303 WriteMatrix() [1/2]

```
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

c	output stream
F	base field
m	row dimension
n	column dimension
A	matrix
format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
column_major	whether the matrix is stored in column or row major (row by default)

15.1.3.304 preamble()

```
void preamble (
            std::ifstream & ifs,
            FFLAS_FORMAT & format )  [inline]
```

15.1.3.305 ReadMatrix() [1/2]

```
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

	ifs	input stream
	F	base field
out	*m*	row dimension
out	*n*	column dimension
out	*A*	output matrix
	format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

15.1.3.306 ReadMatrix() [2/2]

```
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

	matrix_file	filename
	F	base field
out	*m*	row dimension
out	*n*	column dimension
out	*A*	output matrix
	format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

15.1.3.307 WriteMatrix() [2/2]

```
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

matrix_file	file name
F	base field
m	row dimension
n	column dimension
A	matrix
format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
column_major	whether the matrix is stored in column or row major (row by default)

15.1.3.308 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

c	output stream
P	permutation
N	size of the permutation

15.1.3.309 alignable()

```
bool alignable ( )  [inline]
```

15.1.3.310 alignable< Givaro::Integer * >()

```
bool alignable< Givaro::Integer * > ( )  [inline]
```

15.1.3.311 fflas_new() [5/7]

```
Field::Element_ptr fflas_new (
            const Field & F,
            const size_t m,
            const Alignment align = Alignment::DEFAULT )  [inline]
```

15.1.3.312 fflas_new() [6/7]

```
Field::Element_ptr fflas_new (
            const Field & F,
            const size_t m,
            const size_t n,
            const Alignment align = Alignment::DEFAULT )  [inline]
```

15.1.3.313 fflas_new() [7/7]

```
Element * fflas_new (
            const size_t m,
            const Alignment align = Alignment::DEFAULT )  [inline]
```

15.1.3.314 fflas_delete() [3/4]

```
void fflas_delete (
            Element_ptr A )  [inline]
```

15.1.3.315 fflas_delete() [4/4]

```
void fflas_delete (
            Ptr p,
            Args ...  args )  [inline]
```

15.1.3.316 prefetch()

```
void prefetch (
            const int64_t * )  [inline]
```

15.1.3.317 getTLBSize()

```
void getTLBSize (
            int & tlb )  [inline]
```

15.1.3.318 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

15.1.3.319 queryL1CacheSize()

```
int queryL1CacheSize ( )  [inline]
```

Returns

the size in Bytes of the L1 data cache

15.1.3.320 queryTopLevelCacheSize()

```
int queryTopLevelCacheSize ( )  [inline]
```

Returns

the size in Bytes of the L2 or L3 cache if this later is present

15.1.3.321 getSeed()

```
uint64_t getSeed ( )
```

15.2 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)
- 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)
- 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)
- 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)
- 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)

15.2.1 Function Documentation

15.2.1.1 Bini()

```
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]
```

15.2.1.2 WinoPar()

```
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]
```

15.2.1.3 Winograd()

```
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]
```

15.2.1.4 WinogradAcc_3_23()

```
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]
```

15.2.1.5 WinogradAcc_3_21()

```
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]
```

15.2.1.6 WinogradAcc_2_24()

```
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]
```

15.2.1.7 WinogradAcc_2_27()

```
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]
```

15.2.1.8 WinogradAcc_LR()

```
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]
```

15.2.1.9 WinogradAcc_R_S()

```
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]
```

15.2.1.10 WinogradAcc_L_S()

```
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]
```

15.2.1.11 Winograd_LR_S()

```
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]
```

15.2.1.12 Winograd_L_S()

```
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]
```

15.2.1.13 Winograd_R_S()

```
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]
```

15.3 FFLAS::csr_hyb_details Namespace Reference

Data Structures

- struct Coo
- struct Info

15.4 FFLAS::CuttingStrategy Namespace Reference

Data Structures

- struct Block
- struct Column
- struct Recursive
- struct Row
- struct Single

Typedefs

- typedef Row RNSModulus

15.4.1 Typedef Documentation

15.4.1.1 RNSModulus

```
typedef Row RNSModulus
```

15.5 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, 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)
- 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)

- template<class Field >
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscalin (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 fscalin (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 ∗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)
- 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)

15.5.1 Function Documentation

15.5.1.1 fadd() [1/5]

```
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 )
```

15.5.1.2 fadd() [2/5]

```
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 )
```

15.5.1.3 fadd() [3/5]

```
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 )
```

15.5.1.4 fadd() [4/5]

```
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]
```

15.5.1.5 fadd() [5/5]

```
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]
```

15.5.1.6 freduce() [1/4]

```
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]
```

15.5.1.7 freduce() [2/4]

```
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]
```

15.5.1.8 freduce() [3/4]

```
void freduce (
            const Field & F,
            const size_t m,
            typename Field::Element_ptr A,
            const size_t incX,
            FC ) [inline]
```

15.5.1.9 freduce() [4/4]

```
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]
```

15.5.1.10 fscalin() [1/2]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscalin
(
            const Field & F,
            const size_t N,
            const typename Field::Element a,
            typename Field::Element_ptr X,
            const size_t incX,
            FieldCategories::ModularTag ) [inline]
```

15.5.1.11 fscal() [1/2]

```
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 ) [inline]
```

15.5.1.12 fscalin() [2/2]

```
void fscalin (
            const Field & F,
            const size_t n,
            const typename Field::Element a,
            typename Field::Element_ptr X,
            const size_t incX,
            FC ) [inline]
```

15.5.1.13 fscal() [2/2]

```
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]
```

15.5.1.14 igebb44()

```
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 ) [inline]
```

15.5.1.15 igebb24()

```
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 ) [inline]
```

15.5.1.16 igebb14()

```
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]
```

15.5.1.17 igebb41()

```
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 ?

15.5.1.18 igebb21()

```
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]
```

15.5.1.19 igebb11()

```
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]
```

15.5.1.20 igebp()

```
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 )
```

15.5.1.21 pack_lhs()

```
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

15.5.1.22 pack_rhs()

```
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

15.5.1.23 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 )
```

15.5.1.24 BlockingFactor()

```
void BlockingFactor (
            size_t & m,
            size_t & n,
            size_t & k )  [inline]
```

15.6 FFLAS::details_spmv Namespace Reference

Data Structures

- struct Coo

15.7 FFLAS::ElementCategories Namespace Reference

Data Structures

- struct ArbitraryPrecIntTag

    Arbitrary precision integers: GMP.
- struct FixedPrecIntTag

    Fixed precision integers above machine precision: Givaro::recInt.
- 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.

15.8 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.

15.8.1 Detailed Description

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

15.9 FFLAS::MMHelperAlgo Namespace Reference

Data Structures

- struct Auto
- struct Bini
- struct Classic
- struct Winograd
- struct WinogradPar

15.10 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.

15.10.1 Detailed Description

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

its field

15.11 FFLAS::ParSeqHelper Namespace Reference

ParSeqHelper for both fgemm and ftrsm.

Data Structures

- struct Compose
- struct Parallel
- struct Sequential

15.11.1 Detailed Description

ParSeqHelper for both fgemm and ftrsm.

ParSeqHelper for both fgemm and ftrsm

15.12 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 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)
- 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 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 RecInt::rint< 6 > &k)
- template<> size_t min_types (const RecInt::rint< 7 > &k)
- template<> size_t min_types (const RecInt::rint< 8 > &k)
- template<> size_t min_types (const RecInt::rint< 9 > &k)
- template<> size_t min_types (const RecInt::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 (RecInt::rint< K > x)
- template<> bool unfit (RecInt::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 MatF2MatFl_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)

15.12.1 Function Documentation

15.12.1.1 computeFactorClassic() [1/3]

```
double computeFactorClassic (
            const Field & F )   [inline]
```

15.12.1.2 computeFactorClassic() [2/3]

```
double computeFactorClassic (
            const Givaro::ModularBalanced< double > & F )   [inline]
```

15.12.1.3 computeFactorClassic() [3/3]

```
double computeFactorClassic (
            const Givaro::ModularBalanced< float > & F ) [inline]
```

15.12.1.4 DotProdBoundClassic()

```
size_t DotProdBoundClassic (
            const Field & F,
            const typename Field::Element & beta ) [inline]
```

15.12.1.5 TRSMBound() [1/3]

```
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

15.12.1.6 TRSMBound() [2/3]

```
size_t TRSMBound (
            const Givaro::Modular< Element > & F ) [inline]
```

Specialization for positive modular representation over float.

Computes nmax s.t. (p-1)/2∗(p^{nmax-1} + (p-2)^{nmax-1}) < 2^24 @pbi See [Dumas Giorgi Pernet 06, arXiv:cs/0601133]

15.12.1.7 TRSMBound() [3/3]

```
size_t TRSMBound (
            const Givaro::ModularBalanced< Element > & F ) [inline]
```

Specialization for balanced modular representation over double.

Computes nmax s.t. (p-1)/2∗(((p+1)/2)^{nmax-1}) < 2^53

Bibliography
- Dumas Giorgi Pernet 06, arXiv:cs/0601133

15.12.1.8 fgemm_convert()

```
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 )  [inline]
```

15.12.1.9 NeedPreAddReduction() [1/2]

```
bool NeedPreAddReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH )  [inline]
```

15.12.1.10 NeedPreAddReduction() [2/2]

```
bool NeedPreAddReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH )  [inline]
```

15.12.1.11 NeedPreSubReduction() [1/2]

```
bool NeedPreSubReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH )  [inline]
```

15.12.1.12 NeedPreSubReduction() [2/2]

```
bool NeedPreSubReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH )  [inline]
```

15.12.1.13 NeedDoublePreAddReduction() [1/2]

```
bool NeedDoublePreAddReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            Element beta,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH )  [inline]
```

15.12.1.14 NeedDoublePreAddReduction() [2/2]

```
bool NeedDoublePreAddReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            Element beta,
            MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH )  [inline]
```

15.12.1.15 ScalAndReduce() [1/2]

```
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]
```

15.12.1.16 ScalAndReduce() [2/2]

```
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]
```

15.12.1.17 fsquareCommon()

```
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]
```

15.12.1.18 WinogradThreshold() [1/4]

```
int WinogradThreshold (
            const Field & F )  [inline]
```

Computes the number of recursive levels to perform.

Parameters

m	the common dimension in the product AxB

15.12.1.19 WinogradThreshold() [2/4]

```
int WinogradThreshold (
            const Givaro::Modular< float > & F )  [inline]
```

15.12.1.20 WinogradThreshold() [3/4]

```
int WinogradThreshold (
            const Givaro::ModularBalanced< double > & F )  [inline]
```

15.12.1.21 WinogradThreshold() [4/4]

```
int WinogradThreshold (
            const Givaro::ModularBalanced< float > & F )  [inline]
```

15.12.1.22 WinogradSteps()

```
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 AxB

15.12.1.23 DynamicPeeling()

```
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]
```

15.12.1.24 DynamicPeeling2()

```
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]
```

15.12.1.25 WinogradCalc()

```
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]
```

15.12.1.26 fgemv_convert()

```
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]
```

15.12.1.27 fger_convert()

```
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]
```

15.12.1.28 min_types() [1/7]

```
size_t min_types (
            const DFE & k )  [inline]
```

15.12.1.29 min_types() [2/7]

```
size_t min_types (
            const RecInt::rint< 6 > & k )  [inline]
```

15.12.1.30 min_types() [3/7]

```
size_t min_types (
            const RecInt::rint< 7 > & k )  [inline]
```

15.12.1.31 min_types() [4/7]

```
size_t min_types (
            const RecInt::rint< 8 > & k )  [inline]
```

15.12.1.32 min_types() [5/7]

```
size_t min_types (
            const RecInt::rint< 9 > & k )  [inline]
```

15.12.1.33 min_types() [6/7]

```
size_t min_types (
            const RecInt::rint< 10 > & k )  [inline]
```

15.12.1.34 min_types() [7/7]

```
size_t min_types (
            const Givaro::Integer & k )  [inline]
```

15.12.1.35 unfit() [1/4]

```
bool unfit (
            T x )  [inline]
```

15.12.1.36 unfit() [2/4]

```
bool unfit (
            int64_t x )  [inline]
```

15.12.1.37 unfit() [3/4]

```
bool unfit (
            RecInt::rint< K > x )  [inline]
```

15.12.1.38 unfit() [4/4]

```
bool unfit (
            RecInt::rint< 6 > x )  [inline]
```

15.12.1.39 igemm_colmajor() [1/2]

```
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 )
```

15.12.1.40 igemm_colmajor() [2/2]

```
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 )
```

15.12.1.41 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.

15.12.1.42 MatF2MatD_Triangular()

```
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 )
```

15.12.1.43 MatF2MatFl_Triangular()

```
void MatF2MatFl_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)

15.13 FFLAS::sell_details Namespace Reference

Data Structures

- struct Coo
- struct Info

15.14 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::MachineFloa >::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::MachineFloatT >::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::MachineFloa >::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::MachineFloa
  >::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)

15.14.1 Function Documentation

15.14.1.1 init_y() [1/2]

```
void init_y (
            const Field & F,
            const size_t m,
            const typename Field::Element b,
            typename Field::Element_ptr y )  [inline]
```

15.14.1.2 init_y() [2/2]

```
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]
```

15.14.1.3 fspmv_dispatch() [1/2]

```
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]
```

15.14.1.4 fspmv_dispatch() [2/2]

```
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]
```

15.14.1.5 fspmv() [1/12]

```
void fspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::GenericTag ,
            NotZOSparseMatrix  )  [inline]
```

15.14.1.6 fspmv() [2/12]

```
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]
```

15.14.1.7 fspmv() [3/12]

```
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]
```

15.14.1.8 fspmv() [4/12]

```
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]
```

15.14.1.9 fspmv() [5/12]

```
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]
```

15.14.1.10 fspmv() [6/12]

```
void fspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::GenericTag ,
            ZOSparseMatrix  )  [inline]
```

15.14.1.11 fspmv() [7/12]

```
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]
```

15.14.1.12 fspmv() [8/12]

```
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]
```

15.14.1.13 fspmv() [9/12]

```
void fspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::ModularTag ,
            std::true_type ) [inline]
```

15.14.1.14 fspmm_dispatch() [1/2]

```
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::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]
```

15.14.1.15 fspmm_dispatch() [2/2]

```
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::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]
```

15.14.1.16 fspmm() [1/9]

```
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]
```

15.14.1.17 fspmm() [2/9]

```
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]
```

15.14.1.18 fspmm() [3/9]

```
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]
```

15.14.1.19 fspmm() [4/9]

```
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]
```

15.14.1.20 fspmm() [5/9]

```
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]
```

15.14.1.21 fspmm() [6/9]

```
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]
```

15.14.1.22 fspmm() [7/9]

```
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]
```

15.14.1.23 fspmm() [8/9]

```
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]
```

15.14.1.24 fspmm() [9/9]

```
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]
```

15.14.1.25 pfspmm_dispatch() [1/2]

```
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]
```

15.14.1.26 pfspmm_dispatch() [2/2]

```
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]
```

15.14.1.27 pfspmm() [1/9]

```
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]
```

15.14.1.28 pfspmm() [2/9]

```
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]
```

15.14.1.29 pfspmm() [3/9]

```
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]
```

15.14.1.30 pfspmm() [4/9]

```
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]
```

15.14.1.31 pfspmm() [5/9]

```
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]
```

15.14.1.32 pfspmm() [6/9]

```
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]
```

15.14.1.33 pfspmm() [7/9]

```
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]
```

15.14.1.34 pfspmm() [8/9]

```
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]
```

15.14.1.35 pfspmm() [9/9]

```
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]
```

15.14.1.36 pfspmv() [1/6]

```
void pfspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::GenericTag ,
            std::false_type  )  [inline]
```

15.14.1.37 pfspmv() [2/6]

```
void pfspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::UnparametricTag ,
            std::false_type  )  [inline]
```

15.14.1.38 pfspmv() [3/6]

```
void pfspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::ModularTag ,
            std::false_type  )  [inline]
```

15.14.1.39 pfspmv() [4/6]

```
void pfspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::GenericTag ,
            std::true_type ) [inline]
```

15.14.1.40 pfspmv() [5/6]

```
void pfspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::UnparametricTag ,
            std::true_type ) [inline]
```

15.14.1.41 pfspmv() [6/6]

```
void pfspmv (
            const Field & F,
            const SM & A,
            typename Field::ConstElement_ptr x,
            typename Field::Element_ptr y,
            FieldCategories::ModularTag ,
            std::true_type ) [inline]
```

15.14.1.42 fspmv() [10/12]

```
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]
```

15.14.1.43 fspmv() [11/12]

```
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]
```

15.14.1.44 fspmv() [12/12]

```
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]
```

15.15 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 block↩Size, 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 block↵
  Size, 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 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 >
  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)
- 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)
- 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)
- 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)

15.15.1 Function Documentation

15.15.1.1 fspmm() [1/15]

```
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]
```

15.15.1.2 fspmm() [2/15]

```
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]
```

15.15.1.3 fspmm() [3/15]

```
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]
```

15.15.1.4 fspmm_simd_aligned() [1/2]

```
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]
```

15.15.1.5 fspmm_simd_unaligned() [1/2]

```
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]
```

15.15.1.6 fspmm_one() [1/4]

```
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]
```

15.15.1.7 fspmm_mone() [1/4]

```
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]
```

15.15.1.8 fspmm_one_simd_aligned() [1/3]

```
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]
```

15.15.1.9 fspmm_one_simd_unaligned() [1/3]

```
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]
```

15.15.1.10 fspmm_mone_simd_aligned() [1/3]

```
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]
```

15.15.1.11 fspmm_mone_simd_unaligned() [1/3]

```
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]
```

15.15.1.12 fspmv() [1/21]

```
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]
```

15.15.1.13 fspmv() [2/21]

```
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]
```

15.15.1.14 fspmv() [3/21]

```
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]
```

15.15.1.15 fspmv_one() [1/10]

```
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]
```

15.15.1.16 fspmv_mone() [1/10]

```
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]
```

15.15.1.17 fspmv_one() [2/10]

```
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]
```

15.15.1.18 fspmv_mone() [2/10]

```
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]
```

15.15.1.19 pfspmm() [1/18]

```
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]
```

15.15.1.20 pfspmm() [2/18]

```
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]
```

15.15.1.21 pfspmm() [3/18]

```
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]
```

15.15.1.22 pfspmm_one() [1/2]

```
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]
```

15.15.1.23 pfspmm_mone() [1/2]

```
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]
```

15.15.1.24 pfspmm_one() [2/2]

```
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]
```

15.15.1.25 pfspmm_mone() [2/2]

```
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]
```

15.15.1.26 pfspmv() [1/18]

```
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]
```

15.15.1.27 pfspmv_task()

```
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]
```

15.15.1.28 pfspmv() [2/18]

```
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]
```

15.15.1.29 pfspmv() [3/18]

```
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]
```

15.15.1.30 pfspmv_one() [1/8]

```
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]
```

15.15.1.31 pfspmv_mone() [1/8]

```
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]
```

15.15.1.32 pfspmv_one() [2/8]

```
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]
```

15.15.1.33 pfspmv_mone() [2/8]

```
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]
```

15.15.1.34 fspmm() [4/15]

```
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]
```

15.15.1.35 fspmm() [5/15]

```
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]
```

15.15.1.36 fspmm_simd_aligned() [2/2]

```
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]
```

15.15.1.37 fspmm_simd_unaligned() [2/2]

```
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]
```

15.15.1.38 fspmm() [6/15]

```
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]
```

15.15.1.39 fspmm_one() [2/4]

```
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]
```

15.15.1.40 fspmm_mone() [2/4]

```
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]
```

15.15.1.41 fspmm_one_simd_aligned() [2/3]

```
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]
```

15.15.1.42 fspmm_one_simd_unaligned() [2/3]

```
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]
```

15.15.1.43 fspmm_mone_simd_aligned() [2/3]

```
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]
```

15.15.1.44 fspmm_mone_simd_unaligned() [2/3]

```
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]
```

15.15.1.45 fspmv() [4/21]

```
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]
```

15.15.1.46 fspmv() [5/21]

```
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]
```

15.15.1.47 fspmv() [6/21]

```
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]
```

15.15.1.48 fspmv_one() [3/10]

```
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]
```

15.15.1.49 fspmv_mone() [3/10]

```
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]
```

15.15.1.50 fspmv_one() [4/10]

```
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]
```

15.15.1.51 fspmv_mone() [4/10]

```
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]
```

15.15.1.52 pfspmm() [4/18]

```
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]
```

15.15.1.53 pfspmm() [5/18]

```
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]
```

15.15.1.54 pfspmm() [6/18]

```
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]
```

15.15.1.55 pfspmm() [7/18]

```
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]
```

15.15.1.56 pfspmm() [8/18]

```
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]
```

15.15.1.57 pfspmm() [9/18]

```
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]
```

15.15.1.58 pfspmv() [4/18]

```
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]
```

15.15.1.59 pfspmv() [5/18]

```
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]
```

15.15.1.60 pfspmv() [6/18]

```
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]
```

15.15.1.61 fspmm() [7/15]

```
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]
```

15.15.1.62 fspmm() [8/15]

```
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]
```

15.15.1.63 fspmm() [9/15]

```
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]
```

15.15.1.64 fspmv() [7/21]

```
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]
```

15.15.1.65 fspmv() [8/21]

```
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]
```

15.15.1.66 fspmv() [9/21]

```
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]
```

15.15.1.67 pfspmm() [10/18]

```
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]
```

15.15.1.68 pfspmm() [11/18]

```
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]
```

15.15.1.69 pfspmm() [12/18]

```
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]
```

15.15.1.70 pfspmm() [13/18]

```
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]
```

15.15.1.71 pfspmm() [14/18]

```
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]
```

15.15.1.72 pfspmm() [15/18]

```
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]
```

15.15.1.73 pfspmm_zo() [1/2]

```
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]
```

15.15.1.74 pfspmm_zo() [2/2]

```
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]
```

15.15.1.75 pfspmv() [7/18]

```
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]
```

15.15.1.76 pfspmv() [8/18]

```
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]
```

15.15.1.77 pfspmv() [9/18]

```
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]
```

15.15.1.78 pfspmv_one() [3/8]

```
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]
```

15.15.1.79 pfspmv_mone() [3/8]

```
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]
```

15.15.1.80 pfspmv_one() [4/8]

```
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]
```

15.15.1.81 pfspmv_mone() [4/8]

```
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]
```

15.15.1.82 fspmm() [10/15]

```
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]
```

15.15.1.83 fspmm() [11/15]

```
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]
```

15.15.1.84 fspmm() [12/15]

```
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]
```

15.15.1.85 fspmm_mone() [3/4]

```
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]
```

15.15.1.86 fspmm_one() [3/4]

```
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]
```

15.15.1.87 fspmm_mone() [4/4]

```
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]
```

15.15.1.88 fspmm_one() [4/4]

```
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]
```

15.15.1.89 fspmm_one_simd_aligned() [3/3]

```
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]
```

15.15.1.90 fspmm_one_simd_unaligned() [3/3]

```
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]
```

15.15.1.91 fspmm_mone_simd_aligned() [3/3]

```
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]
```

15.15.1.92 fspmm_mone_simd_unaligned() [3/3]

```
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 ) [inline]
```

15.15.1.93 fspmv() [10/21]

```
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]
```

15.15.1.94 fspmv() [11/21]

```
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]
```

15.15.1.95 fspmv() [12/21]

```
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]
```

15.15.1.96 fspmv_one() [5/10]

```
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]
```

15.15.1.97 fspmv_mone() [5/10]

```
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]
```

15.15.1.98 fspmv_one() [6/10]

```
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]
```

15.15.1.99 fspmv_mone() [6/10]

```
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]
```

15.15.1.100 pfspmv() [10/18]

```
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]
```

15.15.1.101 pfspmv() [11/18]

```
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]
```

15.15.1.102 pfspmv() [12/18]

```
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]
```

15.15.1.103 pfspmv_one() [5/8]

```
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]
```

15.15.1.104 pfspmv_mone() [5/8]

```
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]
```

15.15.1.105 pfspmv_one() [6/8]

```
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]
```

15.15.1.106 pfspmv_mone() [6/8]

```
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]
```

15.15.1.107 fspmv() [13/21]

```
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]
```

15.15.1.108 fspmv_simd() [1/4]

```
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]
```

15.15.1.109 fspmv() [14/21]

```
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]
```

15.15.1.110 fspmv_simd() [2/4]

```
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]
```

15.15.1.111 fspmv() [15/21]

```
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]
```

15.15.1.112 fspmv_one() [7/10]

```
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]
```

15.15.1.113 fspmv_mone() [7/10]

```
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]
```

15.15.1.114 fspmv_one() [8/10]

```
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]
```

15.15.1.115 fspmv_mone() [8/10]

```
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]
```

15.15.1.116 fspmv_one_simd() [1/2]

```
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]
```

15.15.1.117 fspmv_mone_simd() [1/2]

```
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]
```

15.15.1.118 pfspmm() [16/18]

```
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]
```

15.15.1.119 pfspmm() [17/18]

```
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]
```

15.15.1.120 pfspmm() [18/18]

```
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]
```

15.15.1.121 pfspmv() [13/18]

```
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]
```

15.15.1.122 pfspmv() [14/18]

```
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]
```

15.15.1.123 pfspmv() [15/18]

```
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]
```

15.15.1.124 fspmm() [13/15]

```
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]
```

15.15.1.125 fspmm() [14/15]

```
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]
```

15.15.1.126 fspmm() [15/15]

```
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]
```

15.15.1.127 fspmv() [16/21]

```
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]
```

15.15.1.128 fspmv() [17/21]

```
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]
```

15.15.1.129 fspmv() [18/21]

```
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]
```

15.15.1.130 pfspmv() [16/18]

```
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]
```

15.15.1.131 pfspmv() [17/18]

```
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]
```

15.15.1.132 pfspmv() [18/18]

```
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]
```

15.15.1.133 pfspmv_one() [7/8]

```
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]
```

15.15.1.134 pfspmv_mone() [7/8]

```
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]
```

15.15.1.135 pfspmv_one() [8/8]

```
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]
```

15.15.1.136 pfspmv_mone() [8/8]

```
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]
```

15.15.1.137 fspmv() [19/21]

```
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]
```

15.15.1.138 fspmv_simd() [3/4]

```
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]
```

15.15.1.139 fspmv() [20/21]

```
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]
```

15.15.1.140 fspmv_simd() [4/4]

```
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]
```

15.15.1.141 fspmv() [21/21]

```
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]
```

15.15.1.142 fspmv_one() [9/10]

```
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]
```

15.15.1.143 fspmv_mone() [9/10]

```
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]
```

15.15.1.144 fspmv_one_simd() [2/2]

```
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]
```

15.15.1.145 fspmv_mone_simd() [2/2]

```
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]
```

15.15.1.146 fspmv_one() [10/10]

```
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]
```

15.15.1.147 fspmv_mone() [10/10]

```
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]
```

15.16 FFLAS::StrategyParameter Namespace Reference

Data Structures

- struct Fixed
- struct Grain
- struct Threads
- struct ThreeD
- struct ThreeDAdaptive
- struct ThreeDInPlace
- struct TwoD
- struct TwoDAdaptive

15.17 FFLAS::StructureHelper Namespace Reference

StructureHelper for ftrsm.

Data Structures

- struct Hybrid
- struct Iterative
- struct Recursive

15.17.1 Detailed Description

StructureHelper for ftrsm.

15.18 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< 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 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)

15.18.1 Function Documentation

15.18.1.1 VEC_ADD()

```
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]
```

15.18.1.2 addp()

```
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]
```

15.18.1.3 VEC_SUB()

```
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]
```

15.18.1.4 subp()

```
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]
```

15.18.1.5 add()

```
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (
            Element * T,
            const Element * TA,
            const Element * TB,
            size_t n )  [inline]
```

15.18.1.6 sub()

```
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (
            Element * T,
            const Element * TA,
            const Element * TB,
            size_t n )  [inline]
```

15.18.1.7 reduce() [1/9]

```
std::enable_if<!std::is_integral< T >::value, T >::type reduce (
            T A,
            T B )  [inline]
```

15.18.1.8 reduce() [2/9]

```
std::enable_if< std::is_integral< T >::value, T >::type reduce (
            T A,
            T B )  [inline]
```

15.18.1.9 reduce() [3/9]

```
Givaro::Integer reduce (
            Givaro::Integer A,
            Givaro::Integer B )  [inline]
```

15.18.1.10 reduce() [4/9]

```
float reduce (
            float A,
            float B,
            float invB,
            float min,
            float max )  [inline]
```

15.18.1.11 reduce() [5/9]

```
double reduce (
            double A,
            double B,
            double invB,
            double min,
            double max )  [inline]
```

15.18.1.12 reduce() [6/9]

```
int64_t reduce (
            int64_t A,
            int64_t p,
            double invp,
            double min,
            double max,
            int64_t pow50rem )  [inline]
```

15.18.1.13 reduce() [7/9]

```
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::MachineIntTag > & H )  [inline]
```

15.18.1.14 reduce() [8/9]

```
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::MachineFloatTag > & H )  [inline]
```

15.18.1.15 reduce() [9/9]

```
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > & H )  [inline]
```

15.18.1.16 modp() [1/2]

```
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]
```

15.18.1.17 modp() [2/2]

```
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]
```

15.18.1.18 scalp() [1/2]

```
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]
```

15.18.1.19 scalp() [2/2]

```
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]
```

15.19 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)

15.19.1 Function Documentation

15.19.1.1 modp() [1/2]

```
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]
```

15.19.1.2 modp() [2/2]

```
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]
```

15.19.1.3 scalp() [1/2]

```
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]
```

15.19.1.4 scalp() [2/2]

```
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]
```

15.20 FFPACK Namespace Reference

Finite **F**ield **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 precondtion 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 x 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_THRESHOL
- 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_
- 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 &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::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 >
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 $(v, Av, ..., A^N v)$.
- 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)
- 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 L X = B or X L = 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∗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∗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=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\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=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\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\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\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\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.
- 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 L X = B or X L = 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 &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)
- 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, 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 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 x 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 x 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 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)
- 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)
- 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 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 $\{0, \ldots, N-1\}$ 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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)

15.20.1 Detailed Description

Finite **F**ield **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.

15.20.2 Typedef Documentation

15.20.2.1 Checker_PLUQ

```
using Checker_PLUQ = FFLAS::Checker_Empty<Field>
```

15.20.2.2 Checker_Det

```
using Checker_Det = FFLAS::Checker_Empty<Field>
```

15.20.2.3 Checker_invert

```
using Checker_invert = FFLAS::Checker_Empty<Field>
```

15.20.2.4 Checker_charpoly

```
using Checker_charpoly = FFLAS::Checker_Empty<Field>
```

15.20.2.5 ForceCheck_PLUQ

```
using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
```

15.20.2.6 ForceCheck_Det

```
using ForceCheck_Det = CheckerImplem_Det<Field>
```

15.20.2.7 ForceCheck_invert

```
using ForceCheck_invert = CheckerImplem_invert<Field>
```

15.20.2.8 ForceCheck_charpoly

```
using ForceCheck_charpoly = CheckerImplem_charpoly<Field,Polynomial>
```

15.20.3 Function Documentation

15.20.3.1 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.

15.20.3.2 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.

15.20.3.3 applyP() [1/4]

```
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 x 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==FFLAS::FflasLeft for row permutation Side==FFLAS::FflasRight for a column permutation Trans==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]

15.20.3.4 applyP() [2/4]

```
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]
```

15.20.3.5 applyP() [3/4]

```
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]
```

15.20.3.6 MonotonicApplyP()

```
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 reprensentation (a sequence of transpositions)
- the remaining iend-ibeg-R values of the permutation are in a monotonically increasing progression Side==FFLAS::FflasLeft for row permutation Side==FFLAS::FflasRight for a column permutation Trans==FFLAS::FflasTrans for the inverse permutation of P

Parameters

F	base field
Side	selects if it is a row (FflasLeft) or column (FflasRight) permutation
Trans	inverse permutation (FflasTrans/NoTrans)
M	
ibeg	
iend	
A	input matrix
lda	leading dimension of A
P	LAPACK permuation
R	first values of P

The non pivot elements, are located in montonically increasing order.

15.20.3.7 fgetrs() [1/4]

```
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

F	base field
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
M	row dimension of `B`
N	col dimension of `B`
R	rank of `A`
A	input matrix
lda	leading dimension of `A`
P	row permutation of the `PLUQ` decomposition of `A`
Q	column permutation of the `PLUQ` decomposition of `A`
B	Right/Left hand side matrix. Initially stores `B`, finally stores the solution `X`.
ldb	leading dimension of `B`
info	Success of the computation: 0 if successfull, $>$0 if system is inconsistent

15.20.3.8 fgetrs() [2/4]

```
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

F	base field
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
M	row dimension of A
N	col dimension of A
NRHS	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
R	rank of A
A	input matrix
lda	leading dimension of A
P	row permutation of the PLUQ decomposition of A
Q	column permutation of the PLUQ decomposition of A
X	solution matrix
ldx	leading dimension of X
B	Right/Left hand side matrix.
ldb	leading dimension of B
info	Succes of the computation: 0 if successfull, $>$0 if system is inconsistent

15.20.3.9 fgesv() [1/4]

```
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

F	The computation domain
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
M	row dimension of B
N	col dimension of B
A	input matrix
lda	leading dimension of A
B	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
ldb	leading dimension of B
info	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

15.20.3.10 fgesv() [2/4]

```
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

F	The computation domain
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
M	row dimension of A
N	col dimension of A
NRHS	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
A	input matrix
lda	leading dimension of A
B	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
ldb	leading dimension of B
X	
ldx	
info	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

15.20.3.11 ftrtri() [1/2]

```
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

F	base field
Uplo	whether the matrix is upper or lower triangular
Diag	whether the matrix is unit diagonal (FflasUnit/NoUnit)
N	input matrix order
A	the input matrix
lda	leading dimension of A

15.20.3.12 trinv_left() [1/2]

```
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 )
```

15.20.3.13 ftrtrm() [1/2]

```
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

F	base field
Side	set to FflasLeft to compute the product UL, FflasRight to compute LU
diag	whether the matrix U is unit diagonal (FflasUnit/NoUnit)
N	input matrix order
A	the input matrix
lda	leading dimension of A

15.20.3.14 ftrstr()

```
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

F	base field
Side	set to FflasLeft to compute U1^-1∗U2 or L1^-1∗L2, FflasRight to compute U1∗U2^-1 or L1∗L2^-1
Uplo	whether the matrix A is upper or lower triangular
diag1	whether the matrix U1 or L2 is unit diagonal (FflasUnit/NoUnit)
diag2	whether the matrix U2 or L2 is unit diagonal (FflasUnit/NoUnit)
N	order of the input matrices
A	the input matrix to be inverted (U1 or L1)
lda	leading dimension of A
B	the input right hand side (U2 or L2)
ldb	leading dimension of B

15.20.3.15 ftrssyr2k()

```
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

	F	base field
	Side	set to FflasLeft to compute U1^-1∗U2 or L1^-1∗L2, FflasRight to compute U1∗U2^-1 or L1∗L2^-1
	Uplo	whether the matrix A is upper or lower triangular
	diagA	whether the matrix A is unit diagonal (FflasUnit/NoUnit)
	N	order of the input matrices
	A	the input matrix
	lda	leading dimension of A
in,out	*B*	the input right hand side where the output is written
	ldb	leading dimension of B

15.20.3.16 fsytrf() [1/3]

```
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.

15.20.3.17 fsytrf() [2/3]

```
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]
```

15.20.3.18 fsytrf() [3/3]

```
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]
```

15.20.3.19 fsytrf_nonunit() [1/3]

```
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: $A = L \times Dinv \times L^T$ 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.

15.20.3.20 PLUQ() [1/6]

```
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
Diag	whether U should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
M	matrix row dimension
N	matrix column dimension
A	input matrix
lda	leading dimension of A
P	the row permutation
Q	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

15.20.3.21 pPLUQ()

```
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]
```

15.20.3.22 PLUQ() [2/6]

```
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]
```

15.20.3.23 PLUQ() [3/6]

```
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 )
```

15.20.3.24 LUdivine() [1/4]

```
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

F	base field
Diag	whether the transformation matrix (U of the CUP, L of the PLE) should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
trans	whether to compute the CUP decomposition (FflasNoTrans) or the PLE decomposition (FflasTrans)
M	matrix row dimension
N	matrix column dimension
A	input matrix
lda	leading dimension of A
P	the factor of CUP or PLE
Q	a permutation indicating the pivot position in the echelon form C or E in its first r positions
LuTag	flag for setting the earling termination if the matrix is singular
cutoff	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

15.20.3.25 ColumnEchelonForm() [1/3]

```
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 \ V] such that AU = C is a column echelon decomposition of A, with U = P^T [V] and C = M + Q [Ir] [0 In-r] [0] If LuTag == FfpackTileRecursive then A = [N \ 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

15.20.3.26 pColumnEchelonForm()

```
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]
```

15.20.3.27 ColumnEchelonForm() [2/3]

```
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]
```

15.20.3.28 RowEchelonForm() [1/3]

```
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 LuTag == FfpackTileRecursive, then after the computation A = [L \ M] such that X A = R is a row echelon decomposition of A, with X = [L 0] P and R = M + [Ir 0] Q^T [In-r] If LuTag == FfpackTileRecursive then A = [L \ N] such that the same holds with M = N Q^T Qt = Q^T If transform=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

15.20.3.29 pRowEchelonForm()

```
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]
```

15.20.3.30 RowEchelonForm() [2/3]

```
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]
```

15.20.3.31 ReducedColumnEchelonForm() [1/3]

```
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 [Ir] [0 In-r] [M 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
	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
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

15.20.3.32 pReducedColumnEchelonForm()

```
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]
```

15.20.3.33 ReducedColumnEchelonForm() [2/3]

```
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]
```

15.20.3.34 ReducedRowEchelonForm() [1/3]

```
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 = [Ir 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
	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

15.20.3.35 pReducedRowEchelonForm()

```
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]
```

15.20.3.36 ReducedRowEchelonForm() [2/3]

```
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]
```

15.20.3.37 Invert() [1/4]

```
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 $A \leftarrow A^{-1}$

15.20.3.38 Invert() [2/4]

```
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 **nullity** $= 0$). It is untouched otherwise.
	ldx	leading dimension of X
	nullity	dimension of the kernel of A

Returns

pointer to $X = A^{-1}$

15.20.3.39 Invert2() [1/2]

```
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 $2n^3 f$ 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

	F	the computation domain
	M	order of the matrix
in,out	*A*	input matrix ($M \times M$). On output, A is modified and represents a "psycological" factorisation LU.
	lda	leading dimension of A
out	*X*	this is the inverse of A if A is invertible (non NULL and $\texttt{nullity} = 0$). It is untouched otherwise.
	ldx	leading dimension of X
	nullity	dimension of the kernel of A

Returns

pointer to $X = A^{-1}$

Todo this init is not all necessary (done after ftrtri)

Todo this init is not all necessary (done after ftrtri)

15.20.3.40 CharPoly() [1/8]

```
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

	R	the polynomial ring of charp (contains the base field)
out	*charp*	the characteristic polynomial of as a list of factors
	N	order of the matrix A
in	*A*	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	lda	leading dimension of A
	CharpTag	the algorithmic variant
	G	a random iterator (required for the randomized variants LUKrylov and ArithProg)

15.20.3.41 CharPoly() [2/8]

```
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

	R	the polynomial ring of charp (contains the base field)
out	*charp*	the characteristic polynomial of `as` a single polynomial
	N	order of the matrix `A`
in	*A*	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	lda	leading dimension of `A`
	CharpTag	the algorithmic variant
	G	a random iterator (required for the randomized variants LUKrylov and ArithProg)

15.20.3.42 CharPoly() [3/8]

```
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

	R	the polynomial ring of charp (contains the base field)
out	*charp*	the characteristic polynomial of `as` a single polynomial
	N	order of the matrix `A`
in	*A*	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	lda	leading dimension of `A`
	CharpTag	the algorithmic variant

15.20.3.43 MinPoly() [1/4]

```
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

	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

15.20.3.44 MinPoly() [2/4]

```
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, .., 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

15.20.3.45 MatVecMinPoly() [1/2]

```
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 $(v, Av, ..., A^N v)$.

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`

15.20.3.46 Rank() [1/3]

```
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

15.20.3.47 pRank()

```
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 )
```

15.20.3.48 Rank() [2/3]

```
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 )
```

15.20.3.49 IsSingular() [1/2]

```
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 != N`, then the matrix is virtually padded with zeros to make it square and it's 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

15.20.3.50 Det() [1/6]

```
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

15.20.3.51 pDet()

```
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]
```

15.20.3.52 Det() [2/6]

```
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 )
```

15.20.3.53 Solve() [1/3]

```
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	*A*	input matrix
	lda	leading dimension of A
out	*x*	output solution vector
	incx	increment of x
	b	input right hand side of the system
	incb	increment of b

15.20.3.54 Solve() [2/3]

```
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 )
```

15.20.3.55 pSolve()

```
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]
```

15.20.3.56 RandomNullSpaceVector() [1/3]

```
*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 L X = B or X L = B in place.

L is M∗M if Side == FFLAS::FflasLeft and N∗N if Side == FFLAS::FflasRight, B is M∗N. Only the R non trivial column of L are stored in the M∗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

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified to its LU version
	lda	leading dimension of A
out	*X*	output vector
	incX	increment of X

15.20.3.57 NullSpaceBasis() [1/2]

```
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

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified
	lda	leading dimension of A
out	*NS*	output matrix of dimension N x NSdim (allocated here)
out	*ldn*	leading dimension of NS
out	*NSdim*	the dimension of the Nullspace (N-rank(A))

15.20.3.58 RowRankProfile() [1/3]

```
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

	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	*rkprofile*	return the rank profile as an array of row indexes, of dimension r=rank(A)
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

Returns

R

15.20.3.59 pRowRankProfile()

```
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]
```

15.20.3.60 RowRankProfile() [2/3]

```
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]
```

15.20.3.61 ColumnRankProfile() [1/3]

```
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=rank(A)

A is modified rkprofile is allocated during the computation.

Returns

R

15.20.3.62 pColumnRankProfile()

```
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]
```

15.20.3.63 ColumnRankProfile() [2/3]

```
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]
```

15.20.3.64 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

15.20.3.65 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

P	the permutation carrying the rank profile information
M	the row dimension of the initial matrix
N	the column dimension of the initial matrix
R	the rank of the initial matrix
LSm	the row dimension of the leading submatrix considered
LSn	the column dimension of the leading submatrix considered
P	the row permutation of the PLUQ decomposition
Q	the column permutation of the PLUQ decomposition
RRP	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.

15.20.3.66 RowRankProfileSubmatrixIndices() [1/2]

```
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*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
	rowindices	array of the row indices of X in A
	colindices	array of the col indices of X in A
	lda	leading dimension of A
out	*R*	list of indices

rowindices and colindices are allocated during the computation. A is modified

Returns

R

15.20.3.67 ColRankProfileSubmatrixIndices() [1/2]

```
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*r X of A whose columns correspond to the column rank profile of A.

Parameters

	F	base field
	M	number of rows

Parameters

	N	number of columns
in	*A*	input matrix of dimension
	rowindices	array of the row indices of X in A
	colindices	array of the col indices of X in A
	lda	leading dimension of A
out	*R*	list of indices

rowindices and colindices are allocated during the computation.

Warning

A is modified

Returns

R

15.20.3.68 RowRankProfileSubmatrix() [1/2]

```
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∗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

15.20.3.69 ColRankProfileSubmatrix() [1/2]

```
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
	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

15.20.3.70 getTriangular() [1/2]

```
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\U of rank R.

if OnlyNonZeroVectors is false, then T and A have the same dimensions Otherwise, T is R x N if UpLo = FflasUpper, else T is M x 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 ?

15.20.3.71 getTriangular() [2/2]

```
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=L\U to reveal a triangular matrix of rank R.

Parameters

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is revealed
	diag	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix
in,out	*A*	input/output matrix
	lda	leading dimension of A

Todo just one triangular fzero+fassign ?

Todo just one triangular fzero+fassign ?

15.20.3.72 getEchelonForm() [1/2]

```
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\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)

15.20.3.73 getEchelonForm() [2/2]

```
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\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
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	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 (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

15.20.3.74 getEchelonTransform()

```
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=L\U of rank R obtained by Row↩ EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower: T is N x N (already allocated) such that A T = C is a transformation of A in Column echelon form Else T is M x M (already allocated) such that T A = E is a transformation of A in Row Echelon form

Parameters

	F	base field
	UpLo	Lower (FflasLower) means Transformation to Column Echelon Form, Upper (FflasUpper), to Row Echelon Form
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix
	P	permutation matrix
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

15.20.3.75 getReducedEchelonForm() [1/2]

```
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\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

	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

Parameters

	ldt	leading dimension of T
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)
	OnlyNonZeroVectors	decides whether the last zero rows/columns should be ignored

15.20.3.76 getReducedEchelonForm() [2/2]

```
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\U of rank R obtained by ReducedRowEchelonForm or ReducedColumn←EchelonForm 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 (FflasUpper) or lower (FflasLower) triangular matrix is returned
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	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 (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

15.20.3.77 getReducedEchelonTransform()

```
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\U of rank R obtained by Row↩ EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower: T is N x N (already allocated) such that A T = C is a transformation of A in Column echelon form Else T is M x M (already allocated) such that T A = E is a transformation of A in Row Echelon form

Parameters

	F	base field
	UpLo	selects Col (FflasLower) or Row (FflasUpper) Echelon Form
	diag	selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix
	P	permutation matrix
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

15.20.3.78 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.

15.20.3.79 LQUPtoInverseOfFullRankMinor() [1/2]

```
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

F	base field
rank	rank of the matrix.
A_factors	matrix containing the L and U entries of the factorization
lda	leading dimension of A
QtPointer	theLQUP->getQ()->getPointer() (note: getQ returns Qt!)
X	desired location for output
ldx	leading dimension of X

15.20.3.80 RandomNullSpaceVector() [2/3]

```
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 L X = B or X L = B in place.

L is M∗M if Side == FFLAS::FflasLeft and N∗N if Side == FFLAS::FflasRight, B is M∗N. Only the R non trivial column of L are stored in the M∗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

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified to its LU version
	lda	leading dimension of A
out	*X*	output vector
	incX	increment of X

15.20.3.81 solveLB() [1/2]

```
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 )
```

15.20.3.82 solveLB2() [1/2]

```
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 )
```

15.20.3.83 Danilevski()

```
std::list< Polynomial > & Danilevski (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda )
```

15.20.3.84 buildMatrix()

```
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 :

15.20.3.85 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]
```

15.20.3.86 CharPoly() [5/8]

```
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]
```

15.20.3.87 Det() [3/6]

```
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]
```

15.20.3.88 Det() [4/6]

```
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]
```

15.20.3.89 fsytrf_BC_Crout()

```
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]
```

15.20.3.90 fsytrf_BC_RL()

```
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]
```

15.20.3.91 fsytrf_UP_RPM_BC_RL()

```
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]
```

15.20.3.92 fsytrf_LOW_RPM_BC_Crout()

```
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]
```

15.20.3.93 fsytrf_UP_RPM_BC_Crout()

```
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]
```

15.20.3.94 fsytrf_UP_RPM()

```
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 |] [I_(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

15.20.3.95 fsytrf_nonunit() [2/3]

```
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]
```

15.20.3.96 fsytrf_nonunit() [3/3]

```
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]
```

15.20.3.97 fsytrf_RPM()

```
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]
```

15.20.3.98 getTridiagonal()

```
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]
```

15.20.3.99 LUdivine_gauss() [1/2]

```
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]
```

15.20.3.100 LUdivine_small() [1/2]

```
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]
```

15.20.3.101 LUdivine() [2/4]

```
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 ?

15.20.3.102 LUdivine() [3/4]

```
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]
```

15.20.3.103 MonotonicCompress()

```
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]
```

15.20.3.104 MonotonicCompressMorePivots()

```
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]
```

15.20.3.105 MonotonicCompressCycles()

```
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]
```

15.20.3.106 MonotonicExpand()

```
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 )
```

15.20.3.107 applyP_block()

```
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]
```

15.20.3.108 doApplyS()

```
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]
```

15.20.3.109 MatrixApplyS() [1/3]

```
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]
```

15.20.3.110 MatrixApplyS() [2/3]

```
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]
```

15.20.3.111 MatrixApplyS() [3/3]

```
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]
```

15.20.3.112 PermApplyS()

```
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]
```

15.20.3.113 doApplyT()

```
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]
```

15.20.3.114 MatrixApplyT() [1/3]

```
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]
```

15.20.3.115 MatrixApplyT() [2/3]

```
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]
```

15.20.3.116 MatrixApplyT() [3/3]

```
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]
```

15.20.3.117 PermApplyT()

```
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]
```

15.20.3.118 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N )  [inline]
```

Computes P1 x 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

15.20.3.119 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 x 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

P1	a LAPACK permutation of size N
P2	a LAPACK permutation of size N-R

15.20.3.120 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	*MathP1*	a MathPermutation of size N
	P2	a LAPACK permutation of size N-R

15.20.3.121 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s ) [inline]
```

15.20.3.122 cyclic_shift_row_col() [1/2]

```
void cyclic_shift_row_col (
            const Field & F,
            typename Field::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda )  [inline]
```

15.20.3.123 cyclic_shift_row() [1/3]

```
void cyclic_shift_row (
            const Field & F,
            typename Field::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda )  [inline]
```

15.20.3.124 cyclic_shift_row() [2/3]

```
void cyclic_shift_row (
            const RNSIntegerMod< T > & F,
            typename T::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda )  [inline]
```

15.20.3.125 cyclic_shift_col() [1/3]

```
void cyclic_shift_col (
            const Field & F,
            typename Field::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda )  [inline]
```

15.20.3.126 cyclic_shift_col() [2/3]

```
void cyclic_shift_col (
            const RNSIntegerMod< T > & F,
            typename T::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda ) [inline]
```

15.20.3.127 PLUQ_basecaseV3()

```
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]
```

15.20.3.128 PLUQ_basecaseV2()

```
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]
```

15.20.3.129 PLUQ_basecaseCrout()

```
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]
```

15.20.3.130 _PLUQ()

```
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]
```

15.20.3.131 PLUQ() [4/6]

```
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]
```

15.20.3.132 threads_fgemm()

```
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 )
```

15.20.3.133 threads_ftrsm()

```
void threads_ftrsm (
            const size_t m,
            const size_t n,
            int nbthreads,
            size_t * t1,
            size_t * t2 )
```

15.20.3.134 PLUQ() [5/6]

```
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]
```

15.20.3.135 fflas_const_cast() [1/3]

```
rns_double_elt_ptr fflas_const_cast (
            rns_double_elt_cstptr x ) [inline]
```

15.20.3.136 fflas_const_cast() [2/3]

```
rns_double_elt_cstptr fflas_const_cast (
            rns_double_elt_ptr x ) [inline]
```

15.20.3.137 cyclic_shift_row_col() [2/2]

```
void cyclic_shift_row_col (
            Base_t * A,
            size_t m,
            size_t n,
            size_t lda )
```

15.20.3.138 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 )
```

15.20.3.139 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 )
```

15.20.3.140 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 )
```

15.20.3.141 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 )
```

15.20.3.142 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 )
```

15.20.3.143 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 )
```

15.20.3.144 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 )
```

15.20.3.145 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 )
```

15.20.3.146 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 ldx )
```

15.20.3.147 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 )
```

15.20.3.148 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 )
```

15.20.3.149 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 )
```

15.20.3.150 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 )
```

15.20.3.151 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 )
```

15.20.3.152 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 )
```

15.20.3.153 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 )
```

15.20.3.154 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 )
```

15.20.3.155 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 )
```

15.20.3.156 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 )
```

15.20.3.157 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 )
```

15.20.3.158 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 )
```

15.20.3.159 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 )
```

15.20.3.160 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 )
```

15.20.3.161 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 )
```

15.20.3.162 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 )
```

15.20.3.163 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 )
```

15.20.3.164 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 )
```

15.20.3.165 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 )
```

15.20.3.166 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 )
```

15.20.3.167 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 )
```

15.20.3.168 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 )
```

15.20.3.169 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 )
```

15.20.3.170 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 )
```

15.20.3.171 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 )
```

15.20.3.172 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 )
```

15.20.3.173 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 )
```

15.20.3.174 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 )
```

15.20.3.175 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 )
```

15.20.3.176 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 )
```

15.20.3.177 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 )
```

15.20.3.178 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 )
```

15.20.3.179 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 )
```

15.20.3.180 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 )
```

15.20.3.181 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 )
```

15.20.3.182 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 )
```

15.20.3.183 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 )
```

15.20.3.184 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 )
```

15.20.3.185 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 )
```

15.20.3.186 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 )
```

15.20.3.187 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 )
```

15.20.3.188 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 )
```

15.20.3.189 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 )
```

15.20.3.190 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 )
```

15.20.3.191 fflas_const_cast() [3/3]

```
T fflas_const_cast (
            CT x )
```

15.20.3.192 failure()

```
Failure & failure ( ) [inline]
```

15.20.3.193 isOdd() [1/3]

```
bool isOdd (
            const T & a ) [inline]
```

15.20.3.194 isOdd() [2/3]

```
bool isOdd (
            const float & a ) [inline]
```

15.20.3.195 isOdd() [3/3]

```
bool isOdd (
            const double & a ) [inline]
```

15.20.3.196 NonZeroRandomMatrix() [1/2]

```
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` x `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 x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

15.20.3.197 NonZeroRandomMatrix() [2/2]

```
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 x 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 x lda field elements)
	lda	leading dimension of A

Returns

A.

15.20.3.198 RandomMatrix() [1/2]

```
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 x 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 x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

15.20.3.199 RandomMatrix() [2/2]

```
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 x 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 x lda field elements)
	lda	leading dimension of A

Returns

A.

15.20.3.200 RandomTriangularMatrix() [1/2]

```
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 x n triangular matrix with random entries. The UpLo parameter defines wether 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 x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

15.20.3.201 RandomTriangularMatrix() [2/2]

```
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 x n triangular matrix with random entries. The UpLo parameter defines wether 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 x lda field elements)
	lda	leading dimension of A

Returns

A.

15.20.3.202 RandInt()

```
size_t RandInt (
            size_t a,
            size_t b )  [inline]
```

15.20.3.203 RandomSymmetricMatrix()

```
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 x n triangular matrix with random entries. The UpLo parameter defines wether it is upper or lower triangular.

Parameters

	F	field
	n	order of A
out	*A*	the matrix (preallocated to at least n x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

15.20.3.204 RandomMatrixWithRank() [1/2]

```
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 x 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 x lda field elements)
lda	leading dimension of A
G	a random iterator

Returns

A.

15.20.3.205 RandomMatrixWithRank() [2/2]

```
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 x 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 x lda field elements)
	lda	leading dimension of A

Returns

A.

15.20.3.206 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 $\{0, \ldots, N-1\}$ 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)

15.20.3.207 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)

15.20.3.208 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 x 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)

15.20.3.209 swapval()

```
void swapval (
            size_t k,
            size_t N,
            size_t * P,
            size_t val ) [inline]
```

15.20.3.210 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` x `N` : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.

Parameters

	N	matrix order
out	*rows*	the row position of each non zero element (pre-allocated)
out	*cols*	the column position of each non zero element (pre-allocated)

15.20.3.211 RandomMatrixWithRankandRPM() [1/2]

```
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` x `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` x `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
G	a random iterator

Returns

`A`.

15.20.3.212 RandomMatrixWithRankandRPM() [2/2]

```
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` x `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` x `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.

15.20.3.213 RandomSymmetricMatrixWithRankandRPM() [1/2]

```
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` x `n` symmetric matrix with random entries and rank `r`.

Parameters

F	field
n	order of `A`
r	rank of `A`
A	the matrix (preallocated to at least `n` x `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
G	a random iterator

Returns

A.

15.20.3.214 RandomSymmetricMatrixWithRankandRPM() [2/2]

```
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` x `n` symmetric matrix with random entries and rank `r`.

Parameters

F	field
n	order of `A`
r	rank of `A`
A	the matrix (preallocated to at least `n` x `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`.

15.20.3.215 RandomMatrixWithRankandRandomRPM() [1/2]

```
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` x `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` x `lda` field elements)
lda	leading dimension of `A`
G	a random iterator

Returns

`A`.

15.20.3.216 RandomMatrixWithRankandRandomRPM() [2/2]

```
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` x `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` x `lda` field elements)
lda	leading dimension of `A`

Returns

`A`.

15.20.3.217 RandomSymmetricMatrixWithRankandRandomRPM() [1/2]

```
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` x `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` x `lda` field elements)
lda	leading dimension of `A`
G	a random iterator

Returns

`A`.

15.20.3.218 RandomSymmetricMatrixWithRankandRandomRPM() [2/2]

```
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` x `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` x `lda` field elements)
lda	leading dimension of `A`

Returns

`A`.

15.20.3.219 RandomMatrixWithDet() [1/2]

```
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` x `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` x `lda` field elements)
lda	leading dimension of `A`

Returns

`A`.

15.20.3.220 RandomMatrixWithDet() [2/2]

```
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 x 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 x lda field elements)
lda	leading dimension of A

Returns

A.

15.20.3.221 maxFieldElt()

```
Givaro::Integer maxFieldElt ( )
```

15.20.3.222 maxFieldElt< Givaro::ZRing< Givaro::Integer > >()

```
Givaro::Integer maxFieldElt< Givaro::ZRing< Givaro::Integer > > ( )
```

15.20.3.223 chooseField()

```
Field * chooseField (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed )
```

15.20.3.224 chooseField< Givaro::ZRing< int32_t > >()

```
Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed )
```

15.20.3.225 chooseField< Givaro::ZRing< int64_t > >()

```
Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed )
```

15.20.3.226 chooseField< Givaro::ZRing< float > >()

```
Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed )
```

15.20.3.227 chooseField< Givaro::ZRing< double > >()

```
Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed )
```

15.21 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 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 >
  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 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 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 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 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)

15.21.1 Function Documentation

15.21.1.1 LUdivine_construct() [1/2]

```
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 )
```

15.21.1.2 GaussJordan()

```
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

	M	row dimension of A
	N	column dimension of A
in,out	*A*	an m x n matrix
	lda	leading dimension of A
	P	row permutation
	Q	column permutation
	LuTag	set the base case to a Tile (FfpackGaussJordanTile) or Slab (FfpackGaussJordanSlab) recursive RedEchelon

| I | A11 | A12 | | |—–|—–|—–|—–| | | |I | ∗| A22 | | | |0 | 0| A22 | | |—–|—–|—–|—–| | | | 0 | A32 | | |—–|—–|—–|—–|
where the transformation matrix is stored at the pivot column position

15.21.1.3 KellerGehrig()

```
std::list< Polynomial > & KellerGehrig (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda )
```

15.21.1.4 KGFast()

```
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 )
```

15.21.1.5 KGFast_generalized()

```
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 )
```

15.21.1.6 fgemv_kgf()

```
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 )
```

15.21.1.7 LUKrylov()

```
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 )
```

15.21.1.8 Danilevski()

```
std::list< Polynomial > & Danilevski (
            const Field & F,
            std::list< Polynomial > & charp,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda )
```

15.21.1.9 RandomKrylovPrecond()

```
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

15.21.1.10 ArithProg()

```
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]
```

15.21.1.11 LUKrylov_KGFast()

```
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 )
```

15.21.1.12 MatVecMinPoly()

```
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]
```

15.21.1.13 Hybrid_KGF_LUK_MinPoly()

```
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 )
```

15.21.1.14 updateD()

```
size_t updateD (
            const Field & F,
```

```
            size_t * d,
            size_t k,
            std::vector< std::vector< typename Field::Element > > & minpt )
```

15.21.1.15 newD()

```
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 )
```

15.21.1.16 CompressRows()

```
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]
```

15.21.1.17 CompressRowsQK()

```
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]
```

15.21.1.18 DeCompressRows()

```
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]
```

15.21.1.19 DeCompressRowsQK()

```
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]
```

15.21.1.20 CompressRowsQA()

```
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]
```

15.21.1.21 DeCompressRowsQA()

```
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]
```

15.21.1.22 LUdivine_construct() [2/2]

```
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 )
```

15.22 Givaro Namespace Reference

Data Structures

- class ModularBalanced
- class Montgomery

15.23 MKL_CONFIG Namespace Reference

15.24 RecInt Namespace Reference

Data Structures

- class rint
- class ruint

Chapter 16

Data Structure Documentation

16.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference

Public Types

- typedef MMHelperAlgo::Winograd value

16.1.1 Member Typedef Documentation

16.1.1.1 value

`typedef MMHelperAlgo::Winograd value`

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference

Public Types

- typedef MMHelperAlgo::Classic value

16.2.1 Member Typedef Documentation

16.2.1.1 value

`typedef MMHelperAlgo::Classic value`

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.3 ArbitraryPrecIntTag Struct Reference

Arbitrary precision integers: GMP.

`#include <field-traits.h>`

16.3.1 Detailed Description

Arbitrary precision integers: GMP.
The documentation for this struct was generated from the following file:

- field-traits.h

16.4 AreEqual< X, Y > Class Template Reference

```
#include <fflas_enum.h>
```

Static Public Attributes

- static const bool value = false

16.4.1 Field Documentation

16.4.1.1 value

```
const bool value = false [static]
```

The documentation for this class was generated from the following file:

- fflas_enum.h

16.5 AreEqual< X, X > Class Template Reference

```
#include <fflas_enum.h>
```

Static Public Attributes

- static const bool value = true

16.5.1 Field Documentation

16.5.1.1 value

```
const bool value = true [static]
```

The documentation for this class was generated from the following file:

- fflas_enum.h

16.6 Argument Struct Reference

```
#include <args-parser.h>
```

Data Fields

- char c
- const char * example
- const char * helpString
- ArgumentType type
- void * data

16.6.1 Field Documentation

16.6.1.1 c

`char c`

16.6.1.2 example

`const char* example`

16.6.1.3 helpString

`const char* helpString`

16.6.1.4 type

`ArgumentType type`

16.6.1.5 data

`void* data`

The documentation for this struct was generated from the following file:

- args-parser.h

16.7 associatedDelayedField< Field > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Field field
- typedef Field & type

16.7.1 Member Typedef Documentation

16.7.1.1 field

`typedef Field field`

16.7.1.2 type

`typedef Field& type`

The documentation for this struct was generated from the following file:

- field-traits.h

16.8 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FFPACK::RNSInteger< RNS > field
- typedef FFPACK::RNSInteger< RNS > type

16.8.1 Member Typedef Documentation

16.8.1.1 field

```
typedef FFPACK::RNSInteger<RNS> field
```

16.8.1.2 type

```
typedef FFPACK::RNSInteger<RNS> type
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.9 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

16.9.1 Member Typedef Documentation

16.9.1.1 field

```
typedef Givaro::ZRing<T> field
```

16.9.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.10 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > field
- typedef Givaro::ZRing< T > type

16.10.1 Member Typedef Documentation

16.10.1.1 field

```
typedef Givaro::ZRing<T> field
```

16.10.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.11 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > field
- typedef Givaro::ZRing< T > type

16.11.1 Member Typedef Documentation

16.11.1.1 field

```
typedef Givaro::ZRing<T> field
```

16.11.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.12 Auto Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.13 Bini Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.14 Block Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.15 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)

16.15.1 Member Function Documentation

16.15.1.1 operator()()

```
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

16.16 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)

16.16.1 Member Function Documentation

16.16.1.1 operator()()

```
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

16.17 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)

16.17.1 Member Function Documentation

16.17.1.1 operator()()

```
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

16.18 CharpolyFailed Class Reference

`#include <ffpack.h>`

The documentation for this class was generated from the following file:

- ffpack.h

16.19 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)

16.19.1 Constructor & Destructor Documentation

16.19.1.1 Checker_Empty()

```
Checker_Empty (
            Params... parameters ) [inline]
```

16.19.2 Member Function Documentation

16.19.2.1 check()

```
bool check (
            Params... parameters ) [inline]
```

The documentation for this struct was generated from the following file:

- checker_empty.h

16.20 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)

16.20.1 Constructor & Destructor Documentation

16.20.1.1 CheckerImplem_charpoly() [1/2]

```
CheckerImplem_charpoly (
            const Field & F_,
            const size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda_ ) [inline]
```

16.20.1.2 CheckerImplem_charpoly() [2/2]

```
CheckerImplem_charpoly (
            typename Field::RandIter & G,
            const size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda_ ) [inline]
```

16.20.1.3 ~CheckerImplem_charpoly()

```
~CheckerImplem_charpoly ( ) [inline]
```

16.20.2 Member Function Documentation

16.20.2.1 check()

```
bool check (
            Polynomial & g ) [inline]
```

The documentation for this class was generated from the following file:

- checker_charpoly.inl

16.21 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.

16.21.1 Constructor & Destructor Documentation

16.21.1.1 CheckerImplem_Det() [1/2]

```
CheckerImplem_Det (
            const Field & F_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda )  [inline]
```

16.21.1.2 CheckerImplem_Det() [2/2]

```
CheckerImplem_Det (
            typename Field::RandIter & G,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda )  [inline]
```

16.21.1.3 ∼CheckerImplem_Det()

```
~CheckerImplem_Det ( )  [inline]
```

16.21.2 Member Function Documentation

16.21.2.1 check()

```
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

LU,storage	for L and U

Parameters

det	
P	
Q	

The documentation for this class was generated from the following file:

- checker_det.inl

16.22 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)

16.22.1 Constructor & Destructor Documentation

16.22.1.1 CheckerImplem_fgemm() [1/2]

```
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]
```

16.22.1.2 CheckerImplem_fgemm() [2/2]

```
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]
```

16.22.1.3 ∼CheckerImplem_fgemm()

```
~CheckerImplem_fgemm ( )  [inline]
```

16.22.2 Member Function Documentation

16.22.2.1 check()

```
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

16.23 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_ftrsm (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_ftrsm ()
- 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)

16.23.1 Constructor & Destructor Documentation

16.23.1.1 CheckerImplem_ftrsm() [1/2]

```
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 )  [inline]
```

16.23.1.2 CheckerImplem_ftrsm() [2/2]

```
CheckerImplem_ftrsm (
            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]
```

16.23.1.3 ~CheckerImplem_ftrsm()

```
~CheckerImplem_ftrsm ( )  [inline]
```

16.23.2 Member Function Documentation

16.23.2.1 check()

```
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_ftrsm.inl

16.24 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)

16.24.1 Constructor & Destructor Documentation

16.24.1.1 CheckerImplem_invert() [1/2]

```
CheckerImplem_invert (
            const Field & F_,
            const size_t m_,
            typename Field::ConstElement_ptr A,
            const size_t lda_ )  [inline]
```

16.24.1.2 CheckerImplem_invert() [2/2]

```
CheckerImplem_invert (
            typename Field::RandIter & G,
            const size_t m_,
            typename Field::ConstElement_ptr A,
            const size_t lda_ )  [inline]
```

16.24.1.3 ~CheckerImplem_invert()

```
~CheckerImplem_invert ( )  [inline]
```

16.24.2 Member Function Documentation

16.24.2.1 check()

```
bool check (
            typename Field::ConstElement_ptr A,
            int nullity )  [inline]
```

The documentation for this class was generated from the following file:

- checker_invert.inl

16.25 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.

16.25.1 Constructor & Destructor Documentation

16.25.1.1 CheckerImplem_PLUQ() [1/2]

```
CheckerImplem_PLUQ (
            const Field & F_,
            size_t m_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda )  [inline]
```

16.25.1.2 CheckerImplem_PLUQ() [2/2]

```
CheckerImplem_PLUQ (
            typename Field::RandIter & G,
            size_t m_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda )  [inline]
```

16.25.1.3 ~CheckerImplem_PLUQ()

```
~CheckerImplem_PLUQ ( )  [inline]
```

16.25.2 Member Function Documentation

16.25.2.1 check()

```
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

A	
r	
P	
Q	

The documentation for this class was generated from the following file:

- checker_pluq.inl

16.26 Classic Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.27 Column Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.28 CompactElement< Element > Struct Template Reference

Public Types

- typedef Element type

16.28.1 Member Typedef Documentation

16.28.1.1 type

```
typedef Element type
```

The documentation for this struct was generated from the following file:

- test-io.C

16.29 CompactElement< double > Struct Reference

Public Types

- typedef int32_t type

16.29.1 Member Typedef Documentation

16.29.1.1 type

`typedef int32_t type`

The documentation for this struct was generated from the following file:

- test-io.C

16.30 CompactElement< float > Struct Reference

Public Types

- typedef int16_t type

16.30.1 Member Typedef Documentation

16.30.1.1 type

`typedef int16_t type`

The documentation for this struct was generated from the following file:

- test-io.C

16.31 CompactElement< int16_t > Struct Reference

Public Types

- typedef int8_t type

16.31.1 Member Typedef Documentation

16.31.1.1 type

`typedef int8_t type`

The documentation for this struct was generated from the following file:

- test-io.C

16.32 CompactElement< int32_t > Struct Reference

Public Types

- typedef int16_t type

16.32.1 Member Typedef Documentation

16.32.1.1 type

`typedef int16_t type`

The documentation for this struct was generated from the following file:

- test-io.C

16.33 CompactElement< int64_t > Struct Reference

Public Types

- typedef int32_t type

16.33.1 Member Typedef Documentation

16.33.1.1 type

```
typedef int32_t type
```

The documentation for this struct was generated from the following file:

- test-io.C

16.34 compatible_data_type< Field > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true

16.34.1 Field Documentation

16.34.1.1 value

```
constexpr bool value = true  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

16.35 compatible_data_type< Givaro::ZRing< double > > Struct Reference

Static Public Attributes

- static constexpr bool value = false

16.35.1 Field Documentation

16.35.1.1 value

```
constexpr bool value = false  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

16.36 compatible_data_type< Givaro::ZRing< float > > Struct Reference

Static Public Attributes

- static constexpr bool value = false

16.36.1 Field Documentation

16.36.1.1 value

```
constexpr bool value = false  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

16.37 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)

16.37.1 Constructor & Destructor Documentation

16.37.1.1 Compose() [1/5]

```
Compose ( )  [inline]
```

16.37.1.2 Compose() [2/5]

```
Compose (
            const Compose< H1, H2 > & other )  [inline]
```

16.37.1.3 Compose() [3/5]

```
Compose (
            const Sequential & S )  [inline]
```

16.37.1.4 Compose() [4/5]

```
Compose (
            size_t th1,
            size_t th2 )  [inline]
```

16.37.1.5 Compose() [5/5]

```
Compose (
            const H1 & o1,
            const H2 & o2 )  [inline]
```

16.37.2 Member Function Documentation

16.37.2.1 first_component()

```
H1 first_component ( ) const  [inline]
```

16.37.2.2 second_component()

```
H2 second_component ( ) const  [inline]
```

16.37.3 Friends And Related Function Documentation

16.37.3.1 operator<<

```
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

16.38 Const_int_t< n > Class Template Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- instrset.h

16.39 Const_uint_t< n > Class Template Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- instrset.h

16.40 Simd128_impl< true, true, false, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.40.1 Field Documentation

16.40.1.1 v

```
vect_t v
```

16.40.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd128_int16.inl

16.41 Simd128_impl< true, true, false, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.41.1 Field Documentation

16.41.1.1 v

`vect_t v`

16.41.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd128_int32.inl

16.42 Simd128_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.42.1 Field Documentation

16.42.1.1 v

`vect_t v`

16.42.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd128_int64.inl

16.43 Simd128_impl< true, true, true, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.43.1 Field Documentation

16.43.1.1 v

`vect_t v`

16.43.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd128_int16.inl

16.44 Simd128_impl< true, true, true, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.44.1 Field Documentation

16.44.1.1 v

`vect_t v`

16.44.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd128_int32.inl

16.45 Simd128_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.45.1 Field Documentation

16.45.1.1 v

`vect_t v`

16.45.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd128_int64.inl

16.46 Simd256_impl< true, true, false, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.46.1 Field Documentation

16.46.1.1 v

```
vect_t v
```

16.46.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int16.inl

16.47 Simd256_impl< true, true, false, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.47.1 Field Documentation

16.47.1.1 v

```
vect_t v
```

16.47.1.2 t

```
scalar_t t
```

The documentation for this union was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

16.48 Simd256_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.48.1 Field Documentation

16.48.1.1 v

```
vect_t v
```

16.48.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int64.inl

16.49 Simd256_impl< true, true, true, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.49.1 Field Documentation

16.49.1.1 v

```
vect_t v
```

16.49.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int16.inl

16.50 Simd256_impl< true, true, true, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.50.1 Field Documentation

16.50.1.1 v

```
vect_t v
```

16.50.1.2 t

```
scalar_t t
```

The documentation for this union was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

16.51 Simd256_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.51.1 Field Documentation

16.51.1.1 v

```
vect_t v
```

16.51.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int64.inl

16.52 Simd512_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.52.1 Field Documentation

16.52.1.1 v

```
vect_t v
```

16.52.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd512_int64.inl

16.53 Simd512_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

16.53.1 Field Documentation

16.53.1.1 v

```
vect_t v
```

16.53.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd512_int64.inl

16.54 ConvertTo< T > Struct Template Reference

Force conversion to appropriate element type of ElementCategory T.

```
#include <field-traits.h>
```

16.54.1 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::FixedPrecIntTag> tries conversion of Modular<Integer> to Modular<RecInt<K> >
- ConvertTo<ElementCategories::ArbitraryPrecIntTag> tries conversion of Modular<Integer> to RNSInteger

The documentation for this struct was generated from the following file:

- field-traits.h

16.55 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

16.55.1 Member Typedef Documentation

16.55.1.1 Self

```
using Self = Coo<ValT, IdxT>
```

16.55.2 Constructor & Destructor Documentation

16.55.2.1 Coo() [1/4]

```
Coo (
            ValT v,
            IdxT r,
            IdxT c )  [inline]
```

16.55.2.2 Coo() [2/4]

```
Coo ( )  [default]
```

16.55.2.3 Coo() [3/4]

```
Coo (
            const Self & )  [default]
```

16.55.2.4 Coo() [4/4]

```
Coo (
            Self && )  [default]
```

16.55.3 Member Function Documentation

16.55.3.1 operator=() [1/2]

```
Self & operator= (
            const Self & )  [default]
```

16.55.3.2 operator=() [2/2]

```
Self & operator= (
            Self && )  [default]
```

16.55.4 Field Documentation

16.55.4.1 val

```
ValT val = 0
```

16.55.4.2 row

```
IdxT row = 0
```

16.55.4.3 col

```
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- csr_hyb_utils.inl

16.56 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

16.56.1 Constructor & Destructor Documentation

16.56.1.1 Coo() [1/4]

```
Coo ( ) [default]
```

16.56.1.2 Coo() [2/4]

```
Coo (
        typename Field::Element v,
        index_t r,
        index_t c ) [inline]
```

16.56.1.3 Coo() [3/4]

```
Coo (
        const Self & ) [default]
```

16.56.1.4 Coo() [4/4]

```
Coo (
        Self && ) [default]
```

16.56.2 Member Function Documentation

16.56.2.1 operator=() [1/2]

```
Self & operator= (
            const Self & ) [default]
```

16.56.2.2 operator=() [2/2]

```
Self & operator= (
            Self && ) [default]
```

16.56.3 Field Documentation

16.56.3.1 val

```
Field::Element val = 0
```

16.56.3.2 col

```
index_t col = 0
```

16.56.3.3 row

```
index_t row = 0
```

16.56.3.4 deleted

```
bool deleted = false
```

The documentation for this struct was generated from the following file:

- read_sparse.h

16.57 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

16.57.1 Member Typedef Documentation

16.57.1.1 Self

```
using Self = Coo<ValT, IdxT>
```

16.57.2 Constructor & Destructor Documentation

16.57.2.1 Coo() [1/4]

```
Coo (
            ValT v,
            IdxT r,
            IdxT c )  [inline]
```

16.57.2.2 Coo() [2/4]

```
Coo ( )  [default]
```

16.57.2.3 Coo() [3/4]

```
Coo (
            const Self & )  [default]
```

16.57.2.4 Coo() [4/4]

```
Coo (
            Self && )  [default]
```

16.57.3 Member Function Documentation

16.57.3.1 operator=() [1/2]

```
Self & operator= (
            const Self & )  [default]
```

16.57.3.2 operator=() [2/2]

```
Self & operator= (
            Self && )  [default]
```

16.57.4 Field Documentation

16.57.4.1 val

```
ValT val = 0
```

16.57.4.2 row

```
IdxT row = 0
```

16.57.4.3 col

```
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- sell_utils.inl

16.58 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

16.58.1 Field Documentation

16.58.1.1 _coo16

```
FFLAS::Sparse<Field,SparseMatrix_t::COO,int16_t>* _coo16 = nullptr
```

16.58.1.2 _coo32

```
FFLAS::Sparse<Field,SparseMatrix_t::COO,int32_t>* _coo32 = nullptr
```

16.58.1.3 _coo64

```
FFLAS::Sparse<Field,SparseMatrix_t::COO,int64_t>* _coo64 = nullptr
```

16.58.1.4 _coo16_zo

```
FFLAS::Sparse<Field,SparseMatrix_t::COO_ZO,int16_t>* _coo16_zo = nullptr
```

16.58.1.5 _coo32_zo

```
FFLAS::Sparse<Field,SparseMatrix_t::COO_ZO,int32_t>* _coo32_zo = nullptr
```

16.58.1.6 _coo64_zo

```
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

16.59 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

16.59.1 Field Documentation

16.59.1.1 _csr16

`FFLAS::Sparse<Field,SparseMatrix_t::CSR,int16_t>* _csr16 = nullptr`

16.59.1.2 _csr32

`FFLAS::Sparse<Field,SparseMatrix_t::CSR,int32_t>* _csr32 = nullptr`

16.59.1.3 _csr64

`FFLAS::Sparse<Field,SparseMatrix_t::CSR,int64_t>* _csr64 = nullptr`

16.59.1.4 _csr16_zo

`FFLAS::Sparse<Field,SparseMatrix_t::CSR_ZO,int16_t>* _csr16_zo = nullptr`

16.59.1.5 _csr32_zo

`FFLAS::Sparse<Field,SparseMatrix_t::CSR_ZO,int32_t>* _csr32_zo = nullptr`

16.59.1.6 _csr64_zo

`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

16.60 DefaultBoundedTag Struct Reference

Use standard field operations, but keeps track of bounds on input and output.
`#include <field-traits.h>`

16.60.1 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

16.61 DefaultTag Struct Reference

No specific mode of action: use standard field operations.
`#include <field-traits.h>`

16.61.1 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

16.62 DelayedTag Struct Reference

Performs field operations with delayed mod reductions. Ensures result is reduced.
`#include <field-traits.h>`

16.62.1 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

16.63 ElementTraits< Element > Struct Template Reference

ElementTraits.
`#include <field-traits.h>`

Public Types

- typedef ElementCategories::GenericTag value

16.63.1 Detailed Description

template<class Element>
struct FFLAS::ElementTraits< Element >

ElementTraits.

16.63.2 Member Typedef Documentation

16.63.2.1 value

`typedef ElementCategories::GenericTag value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.64 ElementTraits< double > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineFloatTag value

16.64.1 Member Typedef Documentation

16.64.1.1 value

```
typedef ElementCategories::MachineFloatTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.65 ElementTraits< FFPACK::rns_double_elt > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::RNSElementTag value

16.65.1 Member Typedef Documentation

16.65.1.1 value

```
typedef ElementCategories::RNSElementTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.66 ElementTraits< float > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineFloatTag value

16.66.1 Member Typedef Documentation

16.66.1.1 value

```
typedef ElementCategories::MachineFloatTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.67 ElementTraits< Givaro::Integer > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::ArbitraryPrecIntTag value

16.67.1 Member Typedef Documentation

16.67.1.1 value

```
typedef ElementCategories::ArbitraryPrecIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.68 ElementTraits< int16_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value

16.68.1 Member Typedef Documentation

16.68.1.1 value

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.69 ElementTraits< int32_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value

16.69.1 Member Typedef Documentation

16.69.1.1 value

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.70 ElementTraits< int64_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value

16.70.1 Member Typedef Documentation

16.70.1.1 value

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.71 ElementTraits< int8_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value

16.71.1 Member Typedef Documentation

16.71.1.1 value

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.72 ElementTraits< RecInt::rint< K > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value

16.72.1 Member Typedef Documentation

16.72.1.1 value

```
typedef ElementCategories::FixedPrecIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.73 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value

16.73.1 Member Typedef Documentation

16.73.1.1 value

```
typedef ElementCategories::FixedPrecIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.74 ElementTraits< RecInt::ruint< K > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value

16.74.1 Member Typedef Documentation

16.74.1.1 value

```
typedef ElementCategories::FixedPrecIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.75 ElementTraits< uint16_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value

16.75.1 Member Typedef Documentation

16.75.1.1 value

```
typedef ElementCategories::MachineIntTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.76 ElementTraits< uint32_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value

16.76.1 Member Typedef Documentation

16.76.1.1 value

`typedef ElementCategories::MachineIntTag value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.77 ElementTraits< uint64_t > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineIntTag value

16.77.1 Member Typedef Documentation

16.77.1.1 value

`typedef ElementCategories::MachineIntTag value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.78 ElementTraits< uint8_t > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineIntTag value

16.78.1 Member Typedef Documentation

16.78.1.1 value

`typedef ElementCategories::MachineIntTag value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.79 EllMat< 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

16.79.1 Field Documentation

16.79.1.1 _ell16

```
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int16_t>* _ell16 = nullptr
```

16.79.1.2 _ell32

```
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int32_t>* _ell32 = nullptr
```

16.79.1.3 _ell64

```
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int64_t>* _ell64 = nullptr
```

16.79.1.4 _ell16_zo

```
FFLAS::Sparse<Field,SparseMatrix_t::ELL_ZO,int16_t>* _ell16_zo = nullptr
```

16.79.1.5 _ell32_zo

```
FFLAS::Sparse<Field,SparseMatrix_t::ELL_ZO,int32_t>* _ell32_zo = nullptr
```

16.79.1.6 _ell64_zo

```
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

16.80 Failure Class Reference

A precondtion 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

16.80.1 Detailed Description

A precondtion 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.

16.80.2 Constructor & Destructor Documentation

16.80.2.1 Failure()

```
Failure ( ) [inline]
```

16.80.3 Member Function Documentation

16.80.3.1 operator()() [1/2]

```
void operator() (
            const char * function,
            int line,
            const char * check ) [inline]
```

A precondtion failed.

Parameters

function	usually `__func__`, the function that threw the error
line	usually `__LINE__`, the line where it happened
check	a string telling what failed.

16.80.3.2 operator()() [2/2]

```
void operator() (
            const char * function,
            const char * file,
            int line,
            const char * check ) [inline]
```

A precondtion 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

function	usually `__func__`, the function that threw the error
file	usually `__FILE__`, the file where this function is
line	usually `__LINE__`, the line where it happened
check	a string telling what failed.

16.80.3.3 setErrorStream()

```
void setErrorStream (
             std::ostream & stream )
```

16.80.3.4 print()

```
std::ostream & print (
             std::ostream & o ) const  [inline]
```

overload the virtual print of LinboxError.

Parameters

o	output stream

16.80.4 Field Documentation

16.80.4.1 _errorStream

```
std::ostream* _errorStream  [protected]
```

The documentation for this class was generated from the following file:

- debug.h

16.81 FailureCharpolyCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- checkers_ffpack.h

16.82 FailureDetCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- checkers_ffpack.h

16.83 FailureFgemmCheck Class Reference

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- checkers_fflas.h

16.84 FailureInvertCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- checkers_ffpack.h

16.85 FailurePLUQCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

16.86 FailureTrsmCheck Class Reference

`#include <checkers_fflas.h>`

The documentation for this class was generated from the following file:

- checkers_fflas.h

16.87 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

16.87.1 Member Typedef Documentation

16.87.1.1 Field

```
using Field = _Field
```

16.87.1.2 Element

```
using Element = typename Field::Element
```

16.87.1.3 simd

```
using simd = Simd<typename _Field::Element>
```

16.87.1.4 vect_t

```
using vect_t = typename simd::vect_t
```

16.87.1.5 scalar_t

```
using scalar_t = typename simd::scalar_t
```

16.87.2 Constructor & Destructor Documentation

16.87.2.1 FieldSimd() [1/3]

```
FieldSimd (
            const Field & f )  [inline]
```

16.87.2.2 FieldSimd() [2/3]

```
FieldSimd (
            const Self & )  [default]
```

16.87.2.3 FieldSimd() [3/3]

```
FieldSimd (
            Self && )  [default]
```

16.87.3 Member Function Documentation

16.87.3.1 operator=() [1/2]

```
Self & operator= (
            const Self & )  [default]
```

16.87.3.2 operator=() [2/2]

```
Self & operator= (
            Self && )  [default]
```

16.87.3.3 init() [1/2]

```
INLINE vect_t init (
            vect_t & x,
            const vect_t a ) const  [inline]
```

16.87.3.4 init() [2/2]

```
INLINE vect_t init (
            const vect_t a ) const  [inline]
```

16.87.3.5 add() [1/2]

```
INLINE vect_t add (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline]
```

16.87.3.6 add() [2/2]

```
INLINE vect_t add (
            const vect_t a,
            const vect_t b )  [inline]
```

16.87.3.7 addin()

```
INLINE vect_t addin (
            vect_t & a,
            const vect_t b ) const  [inline]
```

16.87.3.8 add_r() [1/2]

```
INLINE vect_t add_r (
            vect_t & c,
            const vect_t a,
            const vect_t b ) const  [inline]
```

16.87.3.9 add_r() [2/2]

```
INLINE vect_t add_r (
            const vect_t a,
            const vect_t b ) const [inline]
```

16.87.3.10 addin_r()

```
INLINE vect_t addin_r (
            vect_t & a,
            const vect_t b ) const [inline]
```

16.87.3.11 sub() [1/2]

```
INLINE vect_t sub (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline]
```

16.87.3.12 sub() [2/2]

```
INLINE vect_t sub (
            const vect_t a,
            const vect_t b ) [inline]
```

16.87.3.13 subin()

```
INLINE vect_t subin (
            vect_t & a,
            const vect_t b ) const [inline]
```

16.87.3.14 sub_r() [1/2]

```
INLINE vect_t sub_r (
            vect_t & c,
            const vect_t a,
            const vect_t b ) const [inline]
```

16.87.3.15 sub_r() [2/2]

```
INLINE vect_t sub_r (
            const vect_t a,
            const vect_t b ) const [inline]
```

16.87.3.16 subin_r()

```
INLINE vect_t subin_r (
            vect_t & a,
            const vect_t b ) const [inline]
```

16.87.3.17 zero() [1/2]

```
INLINE vect_t zero (
            vect_t & x ) const  [inline]
```

16.87.3.18 zero() [2/2]

```
INLINE vect_t zero ( ) const  [inline]
```

16.87.3.19 mod()

```
INLINE vect_t mod (
            vect_t & c ) const  [inline]
```

16.87.3.20 mul() [1/2]

```
INLINE vect_t mul (
            vect_t & c,
            const vect_t a,
            const vect_t b ) const  [inline]
```

16.87.3.21 mul() [2/2]

```
INLINE vect_t mul (
            const vect_t a,
            const vect_t b ) const  [inline]
```

16.87.3.22 mulin()

```
INLINE vect_t mulin (
            vect_t & a,
            const vect_t b ) const  [inline]
```

16.87.3.23 mul_r() [1/2]

```
INLINE vect_t mul_r (
            vect_t & c,
            const vect_t a,
            const vect_t b ) const  [inline]
```

16.87.3.24 mul_r() [2/2]

```
INLINE vect_t mul_r (
            const vect_t a,
            const vect_t b ) const  [inline]
```

16.87.3.25 axpy() [1/2]

```
INLINE vect_t axpy (
            vect_t & r,
            const vect_t a,
```

```
             const vect_t b,
             const vect_t c ) const  [inline]
```

16.87.3.26 axpy() [2/2]

```
INLINE vect_t axpy (
             const vect_t c,
             const vect_t a,
             const vect_t b ) const  [inline]
```

16.87.3.27 axpyin()

```
INLINE vect_t axpyin (
             vect_t & c,
             const vect_t a,
             const vect_t b ) const  [inline]
```

16.87.3.28 axpy_r() [1/2]

```
INLINE vect_t axpy_r (
             vect_t & r,
             const vect_t a,
             const vect_t b,
             const vect_t c ) const  [inline]
```

16.87.3.29 axpy_r() [2/2]

```
INLINE vect_t axpy_r (
             const vect_t c,
             const vect_t a,
             const vect_t b ) const  [inline]
```

16.87.3.30 axpyin_r()

```
INLINE vect_t axpyin_r (
             vect_t & c,
             const vect_t a,
             const vect_t b ) const  [inline]
```

16.87.3.31 maxpy() [1/2]

```
INLINE vect_t maxpy (
             vect_t & r,
             const vect_t a,
             const vect_t b,
             const vect_t c ) const  [inline]
```

16.87.3.32 maxpy() [2/2]

```
INLINE vect_t maxpy (
             const vect_t c,
```

```
            const vect_t a,
            const vect_t b ) const  [inline]
```

16.87.3.33 maxpyin()

```
INLINE vect_t maxpyin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) const  [inline]
```

16.87.4 Field Documentation

16.87.4.1 vect_size

```
const constexpr size_t vect_size = simd::vect_size  [static], [constexpr]
```

16.87.4.2 alignment

```
const constexpr size_t alignment = simd::alignment  [static], [constexpr]
```

The documentation for this class was generated from the following file:

- simd_modular.inl

16.88 FieldTraits< Field > Struct Template Reference

FieldTrait.

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false

16.88.1 Detailed Description

template<class Field>
struct FFLAS::FieldTraits< Field >

FieldTrait.

16.88.2 Member Typedef Documentation

16.88.2.1 category

```
typedef FieldCategories::GenericTag category
```

16.88.3 Field Documentation

16.88.3.1 balanced

`const bool balanced = false [static]`

The documentation for this struct was generated from the following file:

- field-traits.h

16.89 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

16.89.1 Member Typedef Documentation

16.89.1.1 category

`typedef FieldCategories::UnparametricTag category`

16.89.2 Field Documentation

16.89.2.1 balanced

`const bool balanced = false [static]`

The documentation for this struct was generated from the following file:

- field-traits.h

16.90 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

16.90.1 Member Typedef Documentation

16.90.1.1 category

```
typedef FieldCategories::ModularTag category
```

16.90.2 Field Documentation

16.90.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.91 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

16.91.1 Member Typedef Documentation

16.91.1.1 category

```
typedef FieldCategories::ModularTag category
```

16.91.2 Field Documentation

16.91.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.92 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

16.92.1 Member Typedef Documentation

16.92.1.1 category

```
typedef FieldCategories::ModularTag category
```

16.92.2 Field Documentation

16.92.2.1 balanced

```
const bool balanced = true [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.93 FieldTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category

Static Public Attributes

- static const bool balanced = false

16.93.1 Member Typedef Documentation

16.93.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.93.2 Field Documentation

16.93.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.94 FieldTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category

Static Public Attributes

- static const bool balanced = false

16.94.1 Member Typedef Documentation

16.94.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.94.2 Field Documentation

16.94.2.1 balanced

```
const bool balanced = false  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.95 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

16.95.1 Member Typedef Documentation

16.95.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.95.2 Field Documentation

16.95.2.1 balanced

```
const bool balanced = false  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.96 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

16.96.1 Member Typedef Documentation

16.96.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.96.2 Field Documentation

16.96.2.1 balanced

```
const bool balanced = false  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.97 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

16.97.1 Member Typedef Documentation

16.97.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.97.2 Field Documentation

16.97.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.98 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

16.98.1 Member Typedef Documentation

16.98.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.98.2 Field Documentation

16.98.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.99 FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category

Static Public Attributes

- static const bool balanced = false

16.99.1 Member Typedef Documentation

16.99.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.99.2 Field Documentation

16.99.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.100 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

16.100.1 Member Typedef Documentation

16.100.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.100.2 Field Documentation

16.100.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.101 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

16.101.1 Member Typedef Documentation

16.101.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.101.2 Field Documentation

16.101.2.1 balanced

```
const bool balanced = false  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.102 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

16.102.1 Member Typedef Documentation

16.102.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

16.102.2 Field Documentation

16.102.2.1 balanced

```
const bool balanced = false  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.103 Fixed Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.104 FixedPrecIntTag Struct Reference

Fixed precision integers above machine precision: Givaro::recInt.

```
#include <field-traits.h>
```

16.104.1 Detailed Description

Fixed precision integers above machine precision: Givaro::recInt.
The documentation for this struct was generated from the following file:

- field-traits.h

16.105 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

16.105.1 Constructor & Destructor Documentation

16.105.1.1 ForStrategy1D() [1/2]

```
ForStrategy1D (
            const blocksize_t n,
            const ParSeqHelper::Parallel< Cut, Param > H )  [inline]
```

16.105.1.2 ForStrategy1D() [2/2]

```
ForStrategy1D (
            const blocksize_t b,
            const blocksize_t e,
            const ParSeqHelper::Parallel< Cut, Param > H )  [inline]
```

16.105.2 Member Function Documentation

16.105.2.1 build()

```
void build (
            const blocksize_t n,
            const ParSeqHelper::Parallel< Cut, Param > H )  [inline]
```

16.105.2.2 initialize()

```
blocksize_t initialize ( )  [inline]
```

16.105.2.3 isTerminated()

```
bool isTerminated ( ) const  [inline]
```

16.105.2.4 begin()

```
blocksize_t begin ( ) const  [inline]
```

16.105.2.5 end()

```
blocksize_t end ( ) const  [inline]
```

16.105.2.6 numblocks()

```
blocksize_t numblocks ( ) const  [inline]
```

16.105.2.7 blockindex()

```
blocksize_t blockindex ( ) const  [inline]
```

16.105.2.8 operator++()

```
blocksize_t operator++ ( )  [inline]
```

16.105.3 Field Documentation

16.105.3.1 ibeg

```
blocksize_t ibeg  [protected]
```

16.105.3.2 iend

```
blocksize_t iend  [protected]
```

16.105.3.3 current

```
blocksize_t current  [protected]
```

16.105.3.4 firstBlockSize

```
blocksize_t firstBlockSize  [protected]
```

16.105.3.5 lastBlockSize

```
blocksize_t lastBlockSize  [protected]
```

16.105.3.6 changeBS

```
blocksize_t changeBS  [protected]
```

16.105.3.7 numBlock

```
blocksize_t numBlock  [protected]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.106 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 numRowBlock
- blocksize_t numColBlock
- blocksize_t BLOCKS

Friends

- std::ostream & operator<< (std::ostream &out, const ForStrategy2D &FS2D)

16.106.1 Constructor & Destructor Documentation

16.106.1.1 ForStrategy2D()

```
ForStrategy2D (
            const blocksize_t m,
            const blocksize_t n,
            const ParSeqHelper::Parallel< Cut, Param > H )  [inline]
```

16.106.2 Member Function Documentation

16.106.2.1 initialize()

```
blocksize_t initialize ( )  [inline]
```

16.106.2.2 isTerminated()

```
bool isTerminated ( ) const  [inline]
```

16.106.2.3 ibegin()

```
blocksize_t ibegin ( ) const  [inline]
```

16.106.2.4 jbegin()

```
blocksize_t jbegin ( ) const  [inline]
```

16.106.2.5 iend()

```
blocksize_t iend ( ) const  [inline]
```

16.106.2.6 jend()

```
blocksize_t jend ( ) const  [inline]
```

16.106.2.7 operator++()

```
blocksize_t operator++ ( )  [inline]
```

16.106.2.8 rownumblocks()

```
blocksize_t rownumblocks ( ) const  [inline]
```

16.106.2.9 colnumblocks()

```
blocksize_t colnumblocks ( ) const  [inline]
```

16.106.2.10 blockindex()

```
blocksize_t blockindex ( ) const  [inline]
```

16.106.2.11 rowblockindex()

```
blocksize_t rowblockindex ( ) const  [inline]
```

16.106.2.12 colblockindex()

```
blocksize_t colblockindex ( ) const  [inline]
```

16.106.3 Friends And Related Function Documentation

16.106.3.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const ForStrategy2D< blocksize_t, Cut, Param > & FS2D )  [friend]
```

16.106.4 Field Documentation

16.106.4.1 _ibeg

```
blocksize_t _ibeg  [protected]
```

16.106.4.2 _iend

```
blocksize_t _iend  [protected]
```

16.106.4.3 _jbeg

```
blocksize_t _jbeg  [protected]
```

16.106.4.4 _jend

```
blocksize_t _jend  [protected]
```

16.106.4.5 rowBlockSize

```
blocksize_t rowBlockSize  [protected]
```

16.106.4.6 colBlockSize

```
blocksize_t colBlockSize [protected]
```

16.106.4.7 current

```
blocksize_t current [protected]
```

16.106.4.8 lastRBS

```
blocksize_t lastRBS [protected]
```

16.106.4.9 lastCBS

```
blocksize_t lastCBS [protected]
```

16.106.4.10 changeRBS

```
blocksize_t changeRBS [protected]
```

16.106.4.11 changeCBS

```
blocksize_t changeCBS [protected]
```

16.106.4.12 numRowBlock

```
blocksize_t numRowBlock [protected]
```

16.106.4.13 numColBlock

```
blocksize_t numColBlock [protected]
```

16.106.4.14 BLOCKS

```
blocksize_t BLOCKS [protected]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.107 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.108 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.109 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.110 ftrmmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.111 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.112 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.113 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.114 ftrmmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.115 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.116 ftrmmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.117 ftrmmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.118 ftrmmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.119 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.120 ftrmmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.121 ftrmmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.122 ftrmmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.123 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.124 ftrsmLeftLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.125 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.126 ftrsmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.127 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

16.127.1 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

16.128 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.129 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.130 ftrsmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.131 ftrsmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.132 ftrsmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.133 ftrsmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.134 ftrsmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.135 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.136 ftrsmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.137 ftrsmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.138 ftrsmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

16.139 GenericTag Struct Reference

default is generic
`#include <field-traits.h>`

16.139.1 Detailed Description

default is generic
The documentation for this struct was generated from the following file:

- field-traits.h

16.140 GenericTag Struct Reference

generic ring.
`#include <field-traits.h>`

16.140.1 Detailed Description

generic ring.
The documentation for this struct was generated from the following file:

- field-traits.h

16.141 Grain Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.142 has_minus_eq_impl< C > Struct Template Reference

`#include <sparse_matrix_traits.h>`

Static Public Attributes

- static constexpr bool value = type::value

16.142.1 Field Documentation

16.142.1.1 value

`constexpr bool value = type::value [static], [constexpr]`

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.143 has_minus_impl< C > Struct Template Reference

`#include <sparse_matrix_traits.h>`

Static Public Attributes

- static constexpr bool value = type::value

16.143.1 Field Documentation

16.143.1.1 value

```
constexpr bool value = type::value  [static], [constexpr]
```
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.144 has_mul_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

16.144.1 Field Documentation

16.144.1.1 value

```
constexpr bool value = type::value  [static], [constexpr]
```
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.145 has_mul_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

16.145.1 Field Documentation

16.145.1.1 value

```
constexpr bool value = type::value  [static], [constexpr]
```
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.146 has_operation< T > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value

16.146.1 Field Documentation

16.146.1.1 value

```
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

16.147 has_plus_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

16.147.1 Field Documentation

16.147.1.1 value

```
constexpr bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.148 has_plus_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

16.148.1 Field Documentation

16.148.1.1 value

```
constexpr bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.149 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

16.149.1 Field Documentation

16.149.1.1 none

```
constexpr uint64_t none = 0_ui64  [static], [constexpr]
```

16.149.1.2 coo

```
constexpr uint64_t coo = 1_ui64  [static], [constexpr]
```

16.149.1.3 csr

```
constexpr uint64_t csr = 1_ui64 << 1  [static], [constexpr]
```

16.149.1.4 ell

```
constexpr uint64_t ell = 1_ui64 << 2  [static], [constexpr]
```

16.149.1.5 aut

```
constexpr uint64_t aut = 1_ui64 << 32  [static], [constexpr]
```

16.149.1.6 pm1

```
constexpr uint64_t pm1 = 1_ui64 << 33  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- fflas_sparse.h

16.150 HelperMod< Field, ElementTraits > Struct Template Reference

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

16.151 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

16.151.1 Constructor & Destructor Documentation

16.151.1.1 HelperMod() [1/2]

```
HelperMod ( )  [inline]
```

16.151.1.2 HelperMod() [2/2]

```
HelperMod (
            const Field & F )  [inline]
```

16.151.2 Field Documentation

16.151.2.1 p

```
Field::Element p
```

16.151.2.2 invp

```
double invp
```

16.151.2.3 min

```
double min
```

16.151.2.4 max

```
double max
```

16.151.2.5 pow50rem

```
int64_t pow50rem
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

16.152 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference

Public Member Functions

- HelperMod ()
- HelperMod (const Field &F)

Data Fields

- Field::Element p

16.152.1 Constructor & Destructor Documentation

16.152.1.1 HelperMod() [1/2]

```
HelperMod ( )  [inline]
```

16.152.1.2 HelperMod() [2/2]

```
HelperMod (
            const Field & F )  [inline]
```

16.152.2 Field Documentation

16.152.2.1 p

```
Field::Element p
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

16.153 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference

Public Member Functions

- HelperMod ()
- HelperMod (const Field &F)

Data Fields

- Field::Element p

16.153.1 Constructor & Destructor Documentation

16.153.1.1 HelperMod() [1/2]

```
HelperMod ( )  [inline]
```

16.153.1.2 HelperMod() [2/2]

```
HelperMod (
            const Field & F )  [inline]
```

16.153.2 Field Documentation

16.153.2.1 p

```
Field::Element p
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

16.154 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

16.154.1 Constructor & Destructor Documentation

16.154.1.1 HelperMod() [1/2]

```
HelperMod ( )  [inline]
```

16.154.1.2 HelperMod() [2/2]

```
HelperMod (
            const Field & F )  [inline]
```

16.154.2 Field Documentation

16.154.2.1 p

```
Field::Element p
```

16.154.2.2 invp

```
Field::Element invp
```

16.154.2.3 min

```
Field::Element min
```

16.154.2.4 max

```
Field::Element max
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

16.155 Hybrid Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.156 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

16.156.1 Constructor & Destructor Documentation

16.156.1.1 Info() [1/4]

```
Info (
        uint64_t it,
        uint64_t s,
        uint64_t p )  [inline]
```

16.156.1.2 Info() [2/4]

```
Info ( )  [default]
```

16.156.1.3 Info() [3/4]

```
Info (
        const Info & )  [default]
```

16.156.1.4 Info() [4/4]

```
Info (
            Info && )  [default]
```

16.156.2 Member Function Documentation

16.156.2.1 operator=() [1/2]

```
Info & operator= (
            const Info & )  [default]
```

16.156.2.2 operator=() [2/2]

```
Info & operator= (
            Info && )  [default]
```

16.156.3 Field Documentation

16.156.3.1 size

```
uint64_t size = 0
```

16.156.3.2 perm

```
uint64_t perm = 0
```

16.156.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- csr_hyb_utils.inl

16.157 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

16.157.1 Constructor & Destructor Documentation

16.157.1.1 Info() [1/4]

```
Info (
            uint64_t it,
            uint64_t s,
            uint64_t p )  [inline]
```

16.157.1.2 Info() [2/4]

```
Info ( )  [default]
```

16.157.1.3 Info() [3/4]

```
Info (
            const Info & )  [default]
```

16.157.1.4 Info() [4/4]

```
Info (
            Info && )  [default]
```

16.157.2 Member Function Documentation

16.157.2.1 operator=() [1/2]

```
Info & operator= (
            const Info & )  [default]
```

16.157.2.2 operator=() [2/2]

```
Info & operator= (
            Info && )  [default]
```

16.157.3 Field Documentation

16.157.3.1 size

```
uint64_t size = 0
```

16.157.3.2 perm

```
uint64_t perm = 0
```

16.157.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- sell_utils.inl

16.158 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

16.158.1 Member Typedef Documentation

16.158.1.1 type

```
using type = std::integral_constant<bool, false>
```

16.158.2 Field Documentation

16.158.2.1 value

```
const constexpr bool value = false [static], [constexpr]
```

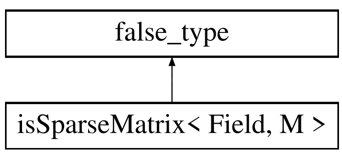
The documentation for this struct was generated from the following file:

- fflas_simd.h

16.159 isSparseMatrix< Field, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, M >:



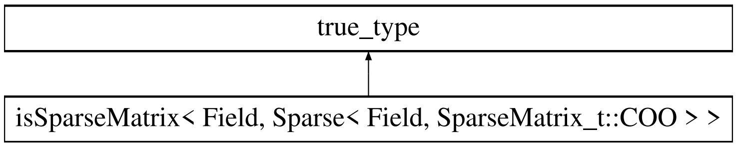
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.160 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 > >:



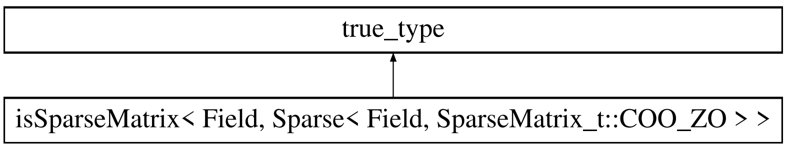
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.161 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 > >:



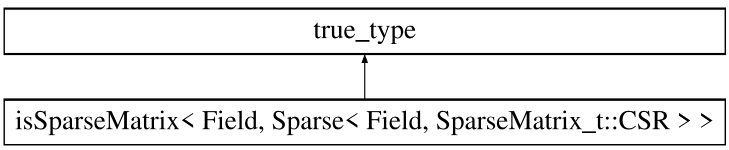
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.162 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 > >:



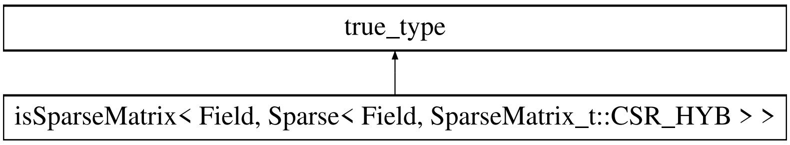
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.163 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 > >:



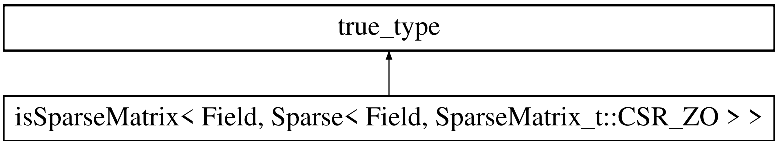
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.164 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 > >:



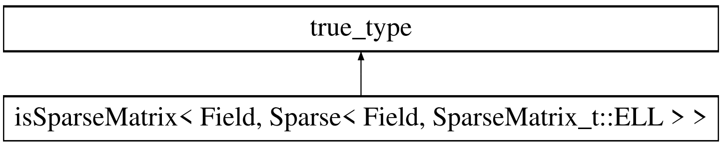
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.165 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 > >:



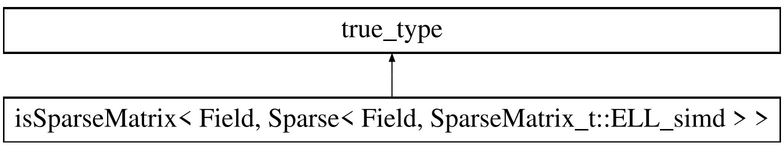
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.166 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 > >:



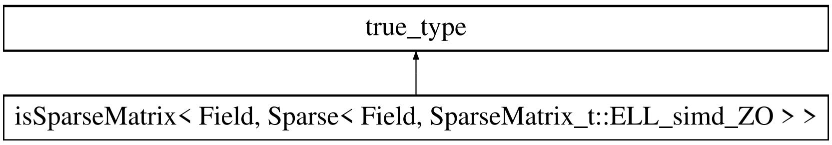
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.167 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 > >:



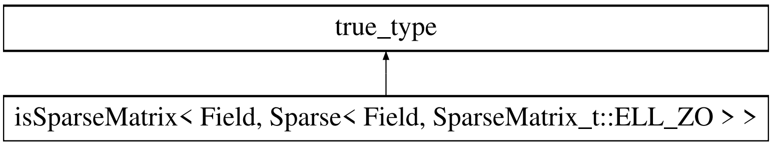
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.168 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 > >:



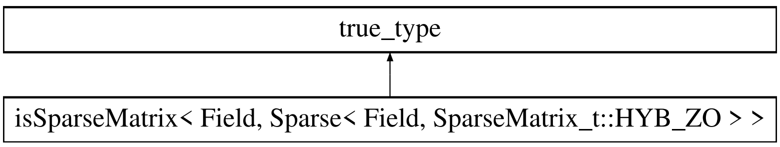
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.169 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 > >:



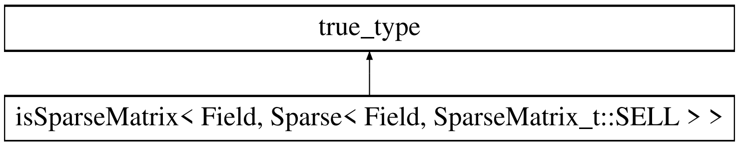
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.170 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 > >:



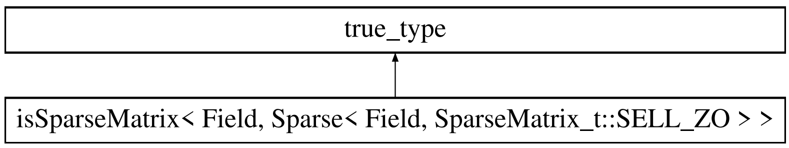
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.171 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 > >:



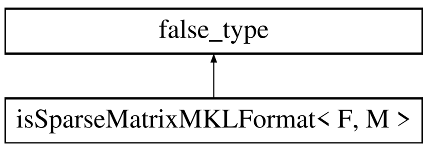
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.172 isSparseMatrixMKLFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixMKLFormat< F, M >:



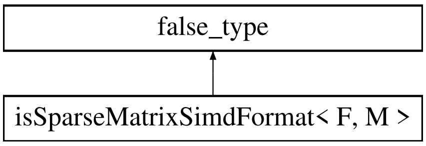
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.173 isSparseMatrixSimdFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixSimdFormat< F, M >:



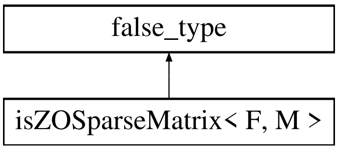
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.174 isZOSparseMatrix< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< F, M >:



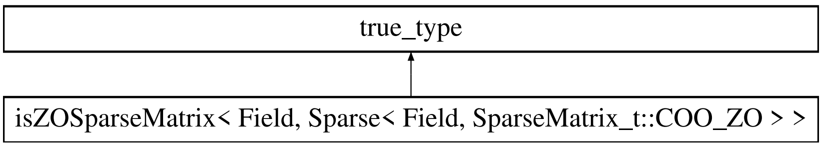
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.175 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 > >:



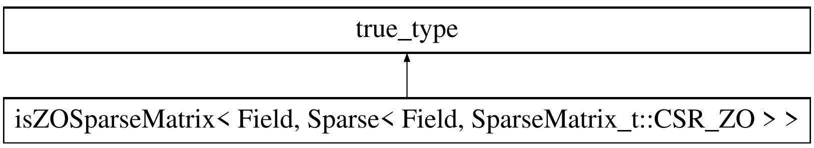
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.176 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 > >:



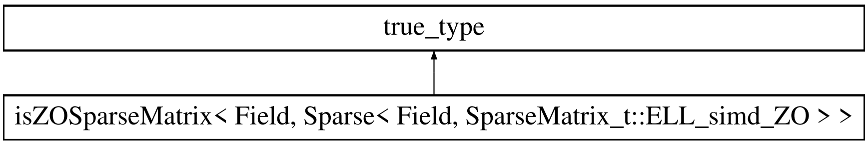
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.177 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 > >:



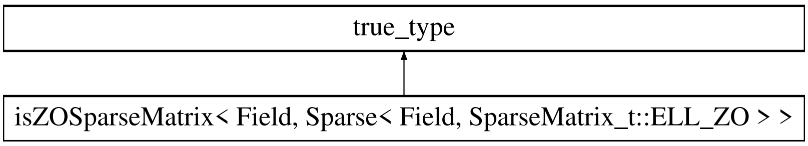
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.178 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 > >:



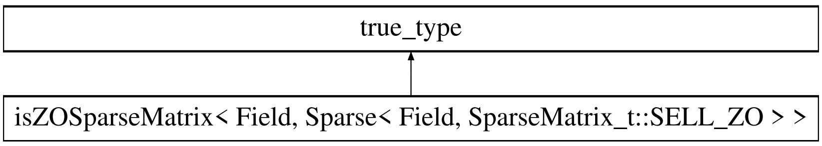
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

16.179 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

16.180 Iterative Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.181 LazyTag Struct Reference

Performs field operations with delayed mod only when necessary. Result may not be reduced.
`#include <field-traits.h>`

16.181.1 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

16.182 limits< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- flimits.h

16.183 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

16.183.1 Member Typedef Documentation

16.183.1.1 T

```
typedef char T
```

16.183.2 Member Function Documentation

16.183.2.1 max()

```
static constexpr char max ( )  [inline], [static], [constexpr], [noexcept]
```

16.183.2.2 min()

```
static constexpr char min ( )  [inline], [static], [constexpr], [noexcept]
```

16.183.2.3 digits()

```
static constexpr int32_t digits ( )  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.184 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

16.184.1 Member Typedef Documentation

16.184.1.1 T

```
typedef double T
```

16.184.2 Member Function Documentation

16.184.2.1 max()

```
static constexpr int64_t max ( )  [inline], [static], [constexpr], [noexcept]
```

16.184.2.2 min()

```
static constexpr int64_t min ( )  [inline], [static], [constexpr], [noexcept]
```

16.184.2.3 digits()

```
static constexpr int32_t digits ( )  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.185 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

16.185.1 Member Typedef Documentation

16.185.1.1 T

```
typedef float T
```

16.185.2 Member Function Documentation

16.185.2.1 max()

```
static constexpr int32_t max ( ) [inline], [static], [constexpr], [noexcept]
```

16.185.2.2 min()

```
static constexpr int32_t min ( ) [inline], [static], [constexpr], [noexcept]
```

16.185.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.186 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

16.186.1 Member Typedef Documentation

16.186.1.1 T

```
typedef Givaro::Integer T
```

16.186.2 Member Function Documentation

16.186.2.1 max()

```
static constexpr int max ( ) [inline], [static], [constexpr], [noexcept]
```

16.186.2.2 min()

```
static constexpr int min ( )  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.187 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

16.187.1 Member Typedef Documentation

16.187.1.1 T

```
typedef int T
```

16.187.2 Member Function Documentation

16.187.2.1 max()

```
static constexpr int max ( )  [inline], [static], [constexpr], [noexcept]
```

16.187.2.2 min()

```
static constexpr int min ( )  [inline], [static], [constexpr], [noexcept]
```

16.187.2.3 digits()

```
static constexpr int32_t digits ( )  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.188 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

16.188.1 Member Typedef Documentation

16.188.1.1 T

```
typedef long T
```

16.188.2 Member Function Documentation

16.188.2.1 max()

```
static constexpr long max ( ) [inline], [static], [constexpr], [noexcept]
```

16.188.2.2 min()

```
static constexpr long min ( ) [inline], [static], [constexpr], [noexcept]
```

16.188.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.189 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

16.189.1 Member Typedef Documentation

16.189.1.1 T

```
typedef long long T
```

16.189.2 Member Function Documentation

16.189.2.1 max()

`static constexpr long long max ( ) [inline], [static], [constexpr], [noexcept]`

16.189.2.2 min()

`static constexpr long long min ( ) [inline], [static], [constexpr], [noexcept]`

16.189.2.3 digits()

`static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- flimits.h

16.190 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

16.190.1 Member Typedef Documentation

16.190.1.1 T

`typedef RecInt::ruint<K> T`

16.190.2 Member Function Documentation

16.190.2.1 max()

`static constexpr RecInt::rint< K > max ( ) [inline], [static], [constexpr], [noexcept]`

16.190.2.2 min()

`static constexpr RecInt::rint< K > min ( ) [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- flimits.h

16.191 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

16.191.1 Member Typedef Documentation

16.191.1.1 T

```
typedef RecInt::ruint<K> T
```

16.191.2 Member Function Documentation

16.191.2.1 max()

```
static constexpr RecInt::ruint< K > max ( ) [inline], [static], [constexpr], [noexcept]
```

16.191.2.2 min()

```
static constexpr RecInt::ruint< K > min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.192 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

16.192.1 Member Typedef Documentation

16.192.1.1 T

```
typedef short int T
```

16.192.2 Member Function Documentation

16.192.2.1 max()

```
static constexpr short int max ( )  [inline], [static], [constexpr], [noexcept]
```

16.192.2.2 min()

```
static constexpr short int min ( )  [inline], [static], [constexpr], [noexcept]
```

16.192.2.3 digits()

```
static constexpr int32_t digits ( )  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.193 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

16.193.1 Member Typedef Documentation

16.193.1.1 T

```
typedef signed char T
```

16.193.2 Member Function Documentation

16.193.2.1 max()

```
static constexpr signed char max ( )  [inline], [static], [constexpr], [noexcept]
```

16.193.2.2 min()

```
static constexpr signed char min ( )  [inline], [static], [constexpr], [noexcept]
```

16.193.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.194 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

16.194.1 Member Typedef Documentation

16.194.1.1 T

```
typedef unsigned char T
```

16.194.2 Member Function Documentation

16.194.2.1 max()

```
static constexpr unsigned char max ( ) [inline], [static], [constexpr], [noexcept]
```

16.194.2.2 min()

```
static constexpr unsigned char min ( ) [inline], [static], [constexpr], [noexcept]
```

16.194.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.195 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

16.195.1 Member Typedef Documentation

16.195.1.1 T

```
typedef unsigned int T
```

16.195.2 Member Function Documentation

16.195.2.1 max()

```
static constexpr unsigned int max ( )  [inline], [static], [constexpr], [noexcept]
```

16.195.2.2 min()

```
static constexpr unsigned int min ( )  [inline], [static], [constexpr], [noexcept]
```

16.195.2.3 digits()

```
static constexpr int32_t digits ( )  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.196 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

16.196.1 Member Typedef Documentation

16.196.1.1 T

```
typedef unsigned long T
```

16.196.2 Member Function Documentation

16.196.2.1 max()

```
static constexpr unsigned long max ( ) [inline], [static], [constexpr], [noexcept]
```

16.196.2.2 min()

```
static constexpr unsigned long min ( ) [inline], [static], [constexpr], [noexcept]
```

16.196.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.197 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

16.197.1 Member Typedef Documentation

16.197.1.1 T

```
typedef unsigned long long T
```

16.197.2 Member Function Documentation

16.197.2.1 max()

```
static constexpr unsigned long long max ( ) [inline], [static], [constexpr], [noexcept]
```

16.197.2.2 min()

```
static constexpr unsigned long long min ( ) [inline], [static], [constexpr], [noexcept]
```

16.197.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.198 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

16.198.1 Member Typedef Documentation

16.198.1.1 T

```
typedef unsigned short int T
```

16.198.2 Member Function Documentation

16.198.2.1 max()

```
static constexpr unsigned short int max ( ) [inline], [static], [constexpr], [noexcept]
```

16.198.2.2 min()

```
static constexpr unsigned short int min ( ) [inline], [static], [constexpr], [noexcept]
```

16.198.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

16.199 MachineFloatTag Struct Reference

float or double

```
#include <field-traits.h>
```

16.199.1 Detailed Description

float or double
The documentation for this struct was generated from the following file:

- field-traits.h

16.200 MachineIntTag Struct Reference

short, int, long, long long, and unsigned variants
`#include <field-traits.h>`

16.200.1 Detailed Description

short, int, long, long long, and unsigned variants
The documentation for this struct was generated from the following file:

- field-traits.h

16.201 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)

16.201.1 Member Typedef Documentation

16.201.1.1 Self_t

```
typedef MMHelper<Field,AlgoTrait,ModeTrait,ParSeqTrait> Self_t
```

16.201.1.2 DelayedField_t

```
typedef associatedDelayedField<constField>::type DelayedField_t
```

16.201.1.3 DelayedField

```
typedef associatedDelayedField<constField>::field DelayedField
```

16.201.1.4 DFElt

```
typedef DelayedField::Element DFElt
```

16.201.2 Constructor & Destructor Documentation

16.201.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

16.201.2.2 MMHelper() [2/5]

```
MMHelper (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            ParSeqTrait _PS )  [inline]
```

16.201.2.3 MMHelper() [3/5]

```
MMHelper (
            const Field & F,
            int w,
            ParSeqTrait _PS = ParSeqTrait() )  [inline]
```

16.201.2.4 MMHelper() [4/5]

```
MMHelper (
            MMHelper< F2, AlgoT2, FT2, PS2 > & WH )  [inline]
```

16.201.2.5 MMHelper() [5/5]

```
MMHelper (
            const Field & F,
            int w,
            DFElt _Amin,
            DFElt _Amax,
            DFElt _Bmin,
            DFElt _Bmax,
            DFElt _Cmin,
            DFElt _Cmax,
            ParSeqTrait _PS = ParSeqTrait() )  [inline]
```

16.201.3 Member Function Documentation

16.201.3.1 initC()

```
void initC ( )  [inline]
```

16.201.3.2 initA()

```
void initA ( )  [inline]
```

16.201.3.3 initB()

```
void initB ( )  [inline]
```

16.201.3.4 initOut()

```
void initOut ( )  [inline]
```

16.201.3.5 MaxDelayedDim()

```
size_t MaxDelayedDim (
            DFElt beta )  [inline]
```

16.201.3.6 Aunfit()

```
bool Aunfit ( )  [inline]
```

16.201.3.7 Bunfit()

```
bool Bunfit ( )  [inline]
```

16.201.3.8 setOutBounds()

```
void setOutBounds (
            const size_t k,
            const DFElt alpha,
            const DFElt beta )  [inline]
```

16.201.3.9 checkA()

```
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]
```

16.201.3.10 checkB()

```
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]
```

16.201.3.11 checkOut()

```
bool checkOut (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda )  [inline]
```

16.201.4 Friends And Related Function Documentation

16.201.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Self_t & M )  [friend]
```

16.201.5 Field Documentation

16.201.5.1 recLevel

`int recLevel`

16.201.5.2 FieldMin

`DFElt FieldMin`

16.201.5.3 FieldMax

`DFElt FieldMax`

16.201.5.4 Amin

`DFElt Amin`

16.201.5.5 Amax

`DFElt Amax`

16.201.5.6 Bmin

`DFElt Bmin`

16.201.5.7 Bmax

`DFElt Bmax`

16.201.5.8 Cmin

`DFElt Cmin`

16.201.5.9 Cmax

`DFElt Cmax`

16.201.5.10 Outmin

`DFElt Outmin`

16.201.5.11 Outmax

```
DFElt Outmax
```

16.201.5.12 MaxStorableValue

```
DFElt MaxStorableValue
```

16.201.5.13 delayedField

```
const DelayedField_t delayedField
```

16.201.5.14 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.202 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)

16.202.1 Member Typedef Documentation

16.202.1.1 Self_t

```
typedef MMHelper<FFPACK::RNSInteger<E>, AlgoTrait,ModeCategories::DefaultTag, ParSeqTrait>
Self_t
```

16.202.2 Constructor & Destructor Documentation

16.202.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

16.202.2.2 MMHelper() [2/5]

```
MMHelper (
            Givaro::Integer Amax,
            Givaro::Integer Bmax ) [inline]
```

16.202.2.3 MMHelper() [3/5]

```
MMHelper (
            const FFPACK::RNSInteger< E > & F,
            size_t m,
            size_t n,
            size_t k,
            ParSeqTrait PS = ParSeqTrait() ) [inline]
```

16.202.2.4 MMHelper() [4/5]

```
MMHelper (
            const FFPACK::RNSInteger< E > & F,
            int wino,
            ParSeqTrait PS = ParSeqTrait() ) [inline]
```

16.202.2.5 MMHelper() [5/5]

```
MMHelper (
            MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

16.202.3 Member Function Documentation

16.202.3.1 setNorm()

```
void setNorm (
            Givaro::Integer p ) [inline]
```

16.202.4 Friends And Related Function Documentation

16.202.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Self_t & M )  [friend]
```

16.202.5 Field Documentation

16.202.5.1 normA

```
Givaro::Integer normA
```

16.202.5.2 normB

```
Givaro::Integer normB
```

16.202.5.3 recLevel

```
int recLevel
```

16.202.5.4 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fgemm_classical_mp.inl

16.203 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)

16.203.1 Member Typedef Documentation

16.203.1.1 Self_t

```
typedef MMHelper<FFPACK::RNSIntegerMod<E>, AlgoTrait,ModeCategories::DefaultTag, ParSeqTrait>
Self_t
```

16.203.2 Constructor & Destructor Documentation

16.203.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

16.203.2.2 MMHelper() [2/5]

```
MMHelper (
        Givaro::Integer Amax,
        Givaro::Integer Bmax ) [inline]
```

16.203.2.3 MMHelper() [3/5]

```
MMHelper (
        const FFPACK::RNSIntegerMod< E > & F,
        size_t m,
        size_t n,
        size_t k,
        ParSeqTrait PS = ParSeqTrait() ) [inline]
```

16.203.2.4 MMHelper() [4/5]

```
MMHelper (
        const FFPACK::RNSIntegerMod< E > & F,
        int wino,
        ParSeqTrait PS = ParSeqTrait() ) [inline]
```

16.203.2.5 MMHelper() [5/5]

```
MMHelper (
        MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

16.203.3 Member Function Documentation

16.203.3.1 setNorm()

```
void setNorm (
        Givaro::Integer p ) [inline]
```

16.203.4 Friends And Related Function Documentation

16.203.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Self_t & M )  [friend]
```

16.203.5 Field Documentation

16.203.5.1 normA

```
Givaro::Integer normA
```

16.203.5.2 normB

```
Givaro::Integer normB
```

16.203.5.3 recLevel

```
int recLevel
```

16.203.5.4 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fgemm_classical_mp.inl

16.204 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)

16.204.1 Member Typedef Documentation

16.204.1.1 Self_t

```
typedef MMHelper<Field,AlgoTrait, ModeCategories::ConvertTo<Dest>,ParSeqTrait> Self_t
```

16.204.2 Constructor & Destructor Documentation

16.204.2.1 MMHelper() [1/4]

```
MMHelper ( ) [inline]
```

16.204.2.2 MMHelper() [2/4]

```
MMHelper (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            ParSeqTrait _PS ) [inline]
```

16.204.2.3 MMHelper() [3/4]

```
MMHelper (
            const Field & F,
            int w,
            ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

16.204.2.4 MMHelper() [4/4]

```
MMHelper (
            MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

16.204.3 Friends And Related Function Documentation

16.204.3.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Self_t & M ) [friend]
```

16.204.4 Field Documentation

16.204.4.1 recLevel

```
int recLevel
```

16.204.4.2 parseq

`ParSeqTrait parseq`

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.205 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)

16.205.1 Member Typedef Documentation

16.205.1.1 Self_t

```
typedef MMHelper<Field, AlgoTrait,ModeCategories::ConvertTo<ElementCategories::RNSElementTag>,
ParSeqTrait> Self_t
```

16.205.2 Constructor & Destructor Documentation

16.205.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

16.205.2.2 MMHelper() [2/5]

```
MMHelper (
            MMHelper< F2, A2, M2, PS2 > H2 )  [inline]
```

16.205.2.3 MMHelper() [3/5]

```
MMHelper (
            Givaro::Integer Amax,
            Givaro::Integer Bmax )  [inline]
```

16.205.2.4 MMHelper() [4/5]

```
MMHelper (
            const Field & F,
            size_t m,
            size_t n,
            size_t k,
            ParSeqTrait PS = ParSeqTrait() )  [inline]
```

16.205.2.5 MMHelper() [5/5]

```
MMHelper (
            const Field & F,
            int wino,
            ParSeqTrait PS = ParSeqTrait() )  [inline]
```

16.205.3 Member Function Documentation

16.205.3.1 setNorm()

```
void setNorm (
            Givaro::Integer p )  [inline]
```

16.205.4 Friends And Related Function Documentation

16.205.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Self_t & M )  [friend]
```

16.205.5 Field Documentation

16.205.5.1 normA

```
Givaro::Integer normA
```

16.205.5.2 normB

```
Givaro::Integer normB
```

16.205.5.3 recLevel

```
int recLevel
```

16.205.5.4 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fgemm_classical_mp.inl

16.206 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)

16.206.1 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.

16.206.2 Member Typedef Documentation

16.206.2.1 Self_t

```
typedef MMHelper<Field,AlgoTrait, ModeCategories::DefaultTag,ParSeqTrait> Self_t
```

16.206.3 Constructor & Destructor Documentation

16.206.3.1 MMHelper() [1/4]

```
MMHelper ( ) [inline]
```

16.206.3.2 MMHelper() [2/4]

```
MMHelper (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            ParSeqTrait _PS ) [inline]
```

16.206.3.3 MMHelper() [3/4]

```
MMHelper (
            const Field & F,
            int w,
            ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

16.206.3.4 MMHelper() [4/4]

```
MMHelper (
            MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

16.206.4 Friends And Related Function Documentation

16.206.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Self_t & M ) [friend]
```

16.206.5 Field Documentation

16.206.5.1 recLevel

```
int recLevel
```

16.206.5.2 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.207 ModeTraits< Field > Struct Template Reference

ModeTraits.
`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DefaultTag value

16.207.1 Detailed Description

template<class Field>
struct FFLAS::ModeTraits< Field >

ModeTraits.

16.207.2 Member Typedef Documentation

16.207.2.1 value

`typedef ModeCategories::DefaultTag value`
The documentation for this struct was generated from the following file:

- field-traits.h

16.208 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DelayedTag value

16.208.1 Member Typedef Documentation

16.208.1.1 value

`typedef ModeCategories::DelayedTag value`
The documentation for this struct was generated from the following file:

- field-traits.h

16.209 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value

16.209.1 Member Typedef Documentation

16.209.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.210 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.210.1 Member Typedef Documentation

16.210.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.211 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.211.1 Member Typedef Documentation

16.211.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.212 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.212.1 Member Typedef Documentation

16.212.1.1 value

`typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.213 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value

16.213.1 Member Typedef Documentation

16.213.1.1 value

`typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.214 ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.214.1 Member Typedef Documentation

16.214.1.1 value

`typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.215 ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.215.1 Member Typedef Documentation

16.215.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.216 ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.216.1 Member Typedef Documentation

16.216.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.217 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DelayedTag value

16.217.1 Member Typedef Documentation

16.217.1.1 value

```
typedef ModeCategories::DelayedTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.218 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value

16.218.1 Member Typedef Documentation

16.218.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.219 ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.219.1 Member Typedef Documentation

16.219.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.220 ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.220.1 Member Typedef Documentation

16.220.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.221 ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value

16.221.1 Member Typedef Documentation

16.221.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.222 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DefaultBoundedTag value

16.222.1 Member Typedef Documentation

16.222.1.1 value

```
typedef ModeCategories::DefaultBoundedTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

16.223 ModeTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DefaultBoundedTag value

16.223.1 Member Typedef Documentation

16.223.1.1 value

`typedef ModeCategories::DefaultBoundedTag value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.224 ModeTraits< Givaro::ZRing< float > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DefaultBoundedTag value

16.224.1 Member Typedef Documentation

16.224.1.1 value

`typedef ModeCategories::DefaultBoundedTag value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.225 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value

16.225.1 Member Typedef Documentation

16.225.1.1 value

`typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value`

The documentation for this struct was generated from the following file:

- field-traits.h

16.226 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

16.227 ModularTag Struct Reference

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`

`#include <field-traits.h>`

16.227.1 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

16.228 Montgomery< T > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

16.229 need_field_characteristic< Field > Struct Template Reference

Static Public Attributes

- static constexpr bool value = false

16.229.1 Field Documentation

16.229.1.1 value

`constexpr bool value = false [static], [constexpr]`

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

16.230 need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true

16.230.1 Field Documentation

16.230.1.1 value

`constexpr bool value = true [static], [constexpr]`

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

16.231 need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true

16.231.1 Field Documentation

16.231.1.1 value

`constexpr bool value = true [static], [constexpr]`

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

16.232 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

16.232.1 Member Typedef Documentation

16.232.1.1 vect_t

`using vect_t = T*`

16.232.1.2 scalar_t

`using scalar_t = T`

16.232.2 Member Function Documentation

16.232.2.1 type_string()

`static const std::string type_string ( ) [inline], [static]`

16.232.2.2 valid()

```
static constexpr bool valid (
            TT p ) [inline], [static], [constexpr]
```

16.232.2.3 compliant()

```
static constexpr bool compliant (
            TT n ) [inline], [static], [constexpr]
```

16.232.3 Field Documentation

16.232.3.1 vect_size

```
const constexpr size_t vect_size = 1 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

16.233 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)

16.233.1 Member Typedef Documentation

16.233.1.1 Cut

```
typedef C Cut
```

16.233.1.2 Param

```
typedef P Param
```

16.233.2 Constructor & Destructor Documentation

16.233.2.1 Parallel()

```
Parallel (
            size_t n = NUM_THREADS ) [inline]
```

16.233.3 Member Function Documentation

16.233.3.1 numthreads()

```
size_t numthreads ( ) const [inline]
```

16.233.3.2 set_numthreads()

```
size_t & set_numthreads (
            size_t n ) [inline]
```

16.233.4 Friends And Related Function Documentation

16.233.4.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Parallel< C, P > & p ) [friend]
```

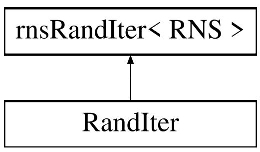
The documentation for this struct was generated from the following file:

- blockcuts.inl

16.234 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

16.234.1 Constructor & Destructor Documentation

16.234.1.1 RandIter()

```
RandIter (
            const RNSInteger< RNS > & F,
            size_t size = 0,
            uint64_t seed = 0 ) [inline]
```

16.234.2 Member Function Documentation

16.234.2.1 random() [1/2]

`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

16.234.2.2 random() [2/2]

`RNS::Element random ( ) const [inline], [inherited]`

16.234.2.3 operator()() [1/2]

`RNS::Element & operator() ( typename RNS::Element & elt ) const [inline], [inherited]`

16.234.2.4 operator()() [2/2]

`RNS::Element operator() ( ) const [inline], [inherited]`

16.234.2.5 ring()

`const RNS & ring ( ) const [inline], [inherited]`

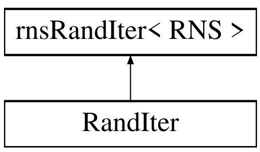
The documentation for this class was generated from the following file:

- rns-integer.h

16.235 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

16.235.1 Constructor & Destructor Documentation

16.235.1.1 RandIter()

```
RandIter (
            const RNSIntegerMod< RNS > & F,
            size_t size = 0,
            uint64_t seed = 0 )  [inline]
```

16.235.2 Member Function Documentation

16.235.2.1 random() [1/2]

```
RNS::Element & random (
            typename RNS::Element & elt ) const  [inline]
```

16.235.2.2 random() [2/2]

```
RNS::Element random ( ) const  [inline], [inherited]
```

16.235.2.3 operator()() [1/2]

```
RNS::Element & operator() (
            typename RNS::Element & elt ) const  [inline], [inherited]
```

16.235.2.4 operator()() [2/2]

```
RNS::Element operator() ( ) const  [inline], [inherited]
```

16.235.2.5 ring()

```
const RNS & ring ( ) const  [inline], [inherited]
```

The documentation for this class was generated from the following file:

- rns-integer-mod.h

16.236 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)

16.236.1 Member Typedef Documentation

16.236.1.1 Element

```
typedef Field::Element Element
```

16.236.1.2 Element_ptr

```
typedef Field::Element_ptr Element_ptr
```

16.236.2 Member Function Documentation

16.236.2.1 operator()()

```
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

16.237 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)

16.237.1 Member Typedef Documentation

16.237.1.1 Element

```
typedef Field::Element Element
```

16.237.1.2 Element_ptr

```
typedef Field::Element_ptr Element_ptr
```

16.237.2 Member Function Documentation

16.237.2.1 operator()()

```
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

16.238 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.239 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.240 rint< K > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

16.241 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 RecInt::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, RecInt::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

16.241.1 Member Typedef Documentation

16.241.1.1 integer

```
typedef Givaro::Integer integer
```

16.241.1.2 ModField

```
typedef Givaro::Modular<double> ModField
```

16.241.1.3 BasisElement

```
typedef double BasisElement
```

16.241.1.4 Element

```
typedef rns_double_elt Element
```

16.241.1.5 Element_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

16.241.1.6 ConstElement_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

16.241.2 Constructor & Destructor Documentation

16.241.2.1 rns_double() [1/4]

```
rns_double (
            const integer & bound,
            size_t pbits,
            bool rnsmod = false,
            long seed = time(NULL) ) [inline]
```

16.241.2.2 rns_double() [2/4]

```
rns_double (
            size_t pbits,
            size_t size,
            long seed = time(NULL) ) [inline]
```

16.241.2.3 rns_double() [3/4]

```
rns_double (
            const Vect & basis,
            bool rnsmod = false,
            long seed = time(NULL) ) [inline]
```

16.241.2.4 rns_double() [4/4]

```
rns_double (
            const RNSIntegerMod< rns_double > & basis,
            bool rnsmod = false,
            long seed = time(NULL) ) [inline]
```

16.241.3 Member Function Documentation

16.241.3.1 precompute_cst()

```
void precompute_cst (
            size_t K = 0 ) [inline]
```

16.241.3.2 init() [1/3]

```
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]
```

16.241.3.3 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]
```

16.241.3.4 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]
```

16.241.3.5 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]
```

16.241.3.6 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]
```

16.241.3.7 reduce()

```
void reduce (
            size_t n,
            double * Arns,
            size_t rda,
            bool RNS_MAJOR = false ) const  [inline]
```

16.241.3.8 init() [3/3]

```
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]
```

16.241.3.9 convert() [2/2]

```
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]
```

16.241.4 Field Documentation

16.241.4.1 _basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

16.241.4.2 _basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

16.241.4.3 _negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

16.241.4.4 _invbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

16.241.4.5 _field_rns

```
std::vector<ModField> _field_rns
```

16.241.4.6 _M

```
integer _M
```

16.241.4.7 _Mi

```
std::vector<integer> _Mi
```

16.241.4.8 _MMi

```
std::vector<double> _MMi
```

16.241.4.9 _crt_in

```
std::vector<double> _crt_in
```

16.241.4.10 _crt_out

```
std::vector<double> _crt_out
```

16.241.4.11 _size

```
size_t _size
```

16.241.4.12 _pbits

```
size_t _pbits
```

16.241.4.13 _ldm

```
size_t _ldm
```

16.241.4.14 _mi_sum

```
integer _mi_sum
```

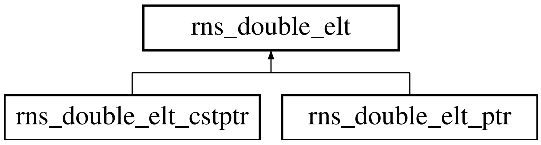
The documentation for this struct was generated from the following files:

- rns-double.h
- rns-double-recint.inl
- rns-double.inl

16.242 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

16.242.1 Constructor & Destructor Documentation

16.242.1.1 rns_double_elt() [1/3]

```
rns_double_elt ( )   [inline]
```

16.242.1.2 ∼rns_double_elt()

```
∼rns_double_elt ( )   [inline]
```

16.242.1.3 rns_double_elt() [2/3]

```
rns_double_elt (
            double ∗ p,
            size_t r,
            size_t a = false )   [inline]
```

16.242.1.4 rns_double_elt() [3/3]

```
rns_double_elt (
            const rns_double_elt & x )   [inline]
```

16.242.2 Member Function Documentation

16.242.2.1 operator&() [1/2]

```
rns_double_elt_ptr operator& ( )   [inline]
```

16.242.2.2 operator&() [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline]
```

16.242.3 Field Documentation

16.242.3.1 _ptr

```
double* _ptr
```

16.242.3.2 _stride

```
size_t _stride
```

16.242.3.3 _alloc

```
bool _alloc
```

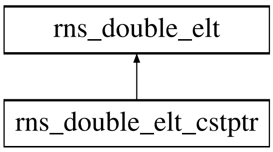
The documentation for this struct was generated from the following file:

- rns-double-elt.h

16.243 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

16.243.1 Constructor & Destructor Documentation

16.243.1.1 rns_double_elt_cstptr() [1/5]

```
rns_double_elt_cstptr ( ) [inline]
```

16.243.1.2 rns_double_elt_cstptr() [2/5]

```
rns_double_elt_cstptr (
            double * p,
            size_t r ) [inline]
```

16.243.1.3 rns_double_elt_cstptr() [3/5]

```
rns_double_elt_cstptr (
            const rns_double_elt_ptr & x ) [inline]
```

16.243.1.4 rns_double_elt_cstptr() [4/5]

```
rns_double_elt_cstptr (
            const rns_double_elt_cstptr & x ) [inline]
```

16.243.1.5 rns_double_elt_cstptr() [5/5]

```
rns_double_elt_cstptr (
            rns_double_elt_cstptr && ) [default]
```

16.243.2 Member Function Documentation

16.243.2.1 operator&() [1/2]

```
rns_double_elt_cstptr * operator& ( ) [inline]
```

16.243.2.2 operator*()

```
rns_double_elt & operator* ( ) const [inline]
```

16.243.2.3 operator[]() [1/2]

```
rns_double_elt operator[] (
            size_t i ) const [inline]
```

16.243.2.4 operator[]() [2/2]

```
rns_double_elt & operator[ ] (
            size_t i ) [inline]
```

16.243.2.5 operator++()

```
rns_double_elt_cstptr operator++ ( ) [inline]
```

16.243.2.6 operator--()

```
rns_double_elt_cstptr operator-- ( ) [inline]
```

16.243.2.7 operator+()

```
rns_double_elt_cstptr operator+ (
            size_t inc ) const [inline]
```

16.243.2.8 operator-()

```
rns_double_elt_cstptr operator- (
            size_t inc ) const [inline]
```

16.243.2.9 operator+=()

```
rns_double_elt_cstptr & operator+= (
            size_t inc ) [inline]
```

16.243.2.10 operator-=()

```
rns_double_elt_cstptr & operator-= (
            size_t inc ) [inline]
```

16.243.2.11 operator=()

```
rns_double_elt_cstptr & operator= (
            const rns_double_elt_cstptr & x ) [inline]
```

16.243.2.12 operator<()

```
bool operator< (
            const rns_double_elt_cstptr & x ) [inline]
```

16.243.2.13 operator"!=()

```
bool operator!= (
            const rns_double_elt_cstptr & x ) [inline]
```

16.243.2.14 operator&() [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```

16.243.3 Field Documentation

16.243.3.1 other

`rns_double_elt other`

16.243.3.2 _ptr

`double* _ptr [inherited]`

16.243.3.3 _stride

`size_t _stride [inherited]`

16.243.3.4 _alloc

`bool _alloc [inherited]`

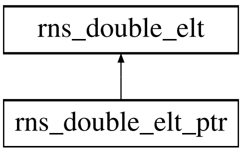
The documentation for this struct was generated from the following file:

- rns-double-elt.h

16.244 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

16.244.1 Constructor & Destructor Documentation

16.244.1.1 rns_double_elt_ptr() [1/5]

```
rns_double_elt_ptr ( ) [inline]
```

16.244.1.2 rns_double_elt_ptr() [2/5]

```
rns_double_elt_ptr (
            double * p,
            size_t r ) [inline]
```

16.244.1.3 rns_double_elt_ptr() [3/5]

```
rns_double_elt_ptr (
            const rns_double_elt_ptr & x ) [inline]
```

16.244.1.4 rns_double_elt_ptr() [4/5]

```
rns_double_elt_ptr (
            const rns_double_elt_cstptr & x ) [inline]
```

16.244.1.5 rns_double_elt_ptr() [5/5]

```
rns_double_elt_ptr (
            rns_double_elt_ptr && ) [default]
```

16.244.2 Member Function Documentation

16.244.2.1 operator&() [1/2]

```
rns_double_elt_ptr * operator& ( ) [inline]
```

16.244.2.2 operator*()

```
rns_double_elt & operator* ( ) [inline]
```

16.244.2.3 operator[]() [1/2]

```
rns_double_elt operator[] (
            size_t i ) const [inline]
```

16.244.2.4 operator[]() [2/2]

```
rns_double_elt & operator[] (
            size_t i ) [inline]
```

16.244.2.5 operator++()

```
rns_double_elt_ptr operator++ ( ) [inline]
```

16.244.2.6 operator--()

```
rns_double_elt_ptr operator-- ( ) [inline]
```

16.244.2.7 operator+()

```
rns_double_elt_ptr operator+ (
            size_t inc ) [inline]
```

16.244.2.8 operator-()

```
rns_double_elt_ptr operator- (
            size_t inc ) [inline]
```

16.244.2.9 operator+=()

```
rns_double_elt_ptr & operator+= (
            size_t inc ) [inline]
```

16.244.2.10 operator-=()

```
rns_double_elt_ptr & operator-= (
            size_t inc ) [inline]
```

16.244.2.11 operator=()

```
rns_double_elt_ptr & operator= (
            const rns_double_elt_ptr & x ) [inline]
```

16.244.2.12 operator<()

```
bool operator< (
            const rns_double_elt_ptr & x ) [inline]
```

16.244.2.13 operator"!=()

```
bool operator!= (
            const rns_double_elt_ptr & x ) [inline]
```

16.244.2.14 operator&() [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```

16.244.3 Field Documentation

16.244.3.1 other

`rns_double_elt other`

16.244.3.2 _ptr

`double* _ptr [inherited]`

16.244.3.3 _stride

`size_t _stride [inherited]`

16.244.3.4 _alloc

`bool _alloc [inherited]`

The documentation for this struct was generated from the following file:

- rns-double-elt.h

16.245 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

16.245.1 Member Typedef Documentation

16.245.1.1 integer

```
typedef Givaro::Integer integer
```

16.245.1.2 ModField

```
typedef Givaro::ModularExtended<double> ModField
```

16.245.1.3 BasisElement

```
typedef double BasisElement
```

16.245.1.4 Element

```
typedef rns_double_elt Element
```

16.245.1.5 Element_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

16.245.1.6 ConstElement_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

16.245.2 Constructor & Destructor Documentation

16.245.2.1 rns_double_extended() [1/3]

```
rns_double_extended (
            const integer & bound,
            size_t pbits,
            bool rnsmod = false,
            long seed = time(NULL) ) [inline]
```

16.245.2.2 rns_double_extended() [2/3]

```
rns_double_extended (
            size_t pbits,
            size_t size,
            long seed = time(NULL) ) [inline]
```

16.245.2.3 rns_double_extended() [3/3]

```
rns_double_extended (
            const Vect & basis,
            bool rnsmod = false,
            long seed = time(NULL) ) [inline]
```

16.245.3 Member Function Documentation

16.245.3.1 precompute_cst()

```
void precompute_cst ( ) [inline]
```

16.245.3.2 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]
```

16.245.3.3 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]
```

16.245.3.4 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]
```

16.245.3.5 init() [3/3]

```
void init (
            size_t m,
            double * Arns,
            const integer * A,
            size_t lda ) const  [inline]
```

16.245.3.6 convert() [2/2]

```
void convert (
            size_t m,
            integer * A,
            const double * Arns ) const  [inline]
```

16.245.3.7 reduce()

```
void reduce (
            size_t n,
            double * Arns,
            size_t rda,
            bool RNS_MAJOR = false ) const  [inline]
```

16.245.4 Field Documentation

16.245.4.1 _basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

16.245.4.2 _basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

16.245.4.3 _negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

16.245.4.4 _invbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

16.245.4.5 _field_rns

```
std::vector<ModField> _field_rns
```

16.245.4.6 _M

```
integer _M
```

16.245.4.7 _Mi

```
std::vector<integer> _Mi
```

16.245.4.8 _MMi

```
std::vector<double> _MMi
```

16.245.4.9 _crt_in

```
std::vector<double> _crt_in
```

16.245.4.10 _crt_out

```
std::vector<double> _crt_out
```

16.245.4.11 _size

```
size_t _size
```

16.245.4.12 _pbits

```
size_t _pbits
```

16.245.4.13 _ldm

```
size_t _ldm
```

The documentation for this struct was generated from the following files:

- rns-double.h
- rns-double.inl

16.246 RNSElementTag Struct Reference

Representation in a Residue Number System.

```
#include <field-traits.h>
```

16.246.1 Detailed Description

Representation in a Residue Number System.
The documentation for this struct was generated from the following file:

- field-traits.h

16.247 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

16.247.1 Member Typedef Documentation

16.247.1.1 BasisElement

`typedef RNS::BasisElement BasisElement [protected]`

16.247.1.2 integer

`typedef Givaro::Integer integer [protected]`

16.247.1.3 Element

`typedef RNS::Element Element`

16.247.1.4 Element_ptr

`typedef RNS::Element_ptr Element_ptr`

16.247.1.5 ConstElement_ptr

`typedef RNS::ConstElement_ptr ConstElement_ptr`

16.247.2 Constructor & Destructor Documentation

16.247.2.1 RNSInteger() [1/2]

```
RNSInteger (
            const RNS & myrns ) [inline]
```

16.247.2.2 RNSInteger() [2/2]

```
RNSInteger (
            const T & F ) [inline]
```

16.247.3 Member Function Documentation

16.247.3.1 rns()

`const RNS & rns ( ) const [inline]`

16.247.3.2 size()

`size_t size ( ) const [inline]`

16.247.3.3 isOne()

```
bool isOne (
            const Element & x ) const  [inline]
```

16.247.3.4 isMOne()

```
bool isMOne (
            const Element & x ) const  [inline]
```

16.247.3.5 isZero()

```
bool isZero (
            const Element & x ) const  [inline]
```

16.247.3.6 characteristic()

```
integer characteristic (
            integer & p ) const  [inline]
```

16.247.3.7 cardinality()

```
integer cardinality (
            integer & p ) const  [inline]
```

16.247.3.8 init() [1/2]

```
Element & init (
            Element & x ) const  [inline]
```

16.247.3.9 init() [2/2]

```
Element & init (
            Element & x,
            const Givaro::Integer & y ) const  [inline]
```

16.247.3.10 reduce() [1/2]

```
Element & reduce (
            Element & x,
            const Element & y ) const  [inline]
```

16.247.3.11 reduce() [2/2]

```
Element & reduce (
            Element & x ) const  [inline]
```

16.247.3.12 convert()

```
Givaro::Integer convert (
            Givaro::Integer & x,
            const Element & y ) const  [inline]
```

16.247.3.13 assign()

```
Element & assign (
            Element & x,
            const Element & y ) const  [inline]
```

16.247.3.14 write() [1/2]

```
std::ostream & write (
            std::ostream & os,
            const Element & y ) const  [inline]
```

16.247.3.15 write() [2/2]

```
std::ostream & write (
            std::ostream & os ) const  [inline]
```

16.247.4 Field Documentation

16.247.4.1 _rns

```
const RNS* _rns  [protected]
```

16.247.4.2 one

```
Element one
```

16.247.4.3 mOne

```
Element mOne
```

16.247.4.4 zero

```
Element zero
```

The documentation for this class was generated from the following files:

- field-traits.h
- rns-integer.h

16.248 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

16.248.1 Member Typedef Documentation

16.248.1.1 Element

```
typedef RNS::Element Element
```

16.248.1.2 Element_ptr

```
typedef RNS::Element_ptr Element_ptr
```

16.248.1.3 ConstElement_ptr

```
typedef RNS::ConstElement_ptr ConstElement_ptr
```

16.248.1.4 BasisElement

```
typedef RNS::BasisElement BasisElement  [protected]
```

16.248.1.5 ModField

```
typedef Givaro::Modular<BasisElement> ModField  [protected]
```

16.248.1.6 integer

```
typedef Givaro::Integer integer  [protected]
```

16.248.2 Constructor & Destructor Documentation

16.248.2.1 RNSIntegerMod()

```
RNSIntegerMod (
            const integer & p,
            const RNS & myrns )  [inline]
```

16.248.3 Member Function Documentation

16.248.3.1 rns()

```
const rns_double & rns ( ) const  [inline]
```

16.248.3.2 delayed()

```
const RNSInteger< RNS > & delayed ( ) const [inline]
```

16.248.3.3 size()

```
size_t size ( ) const [inline]
```

16.248.3.4 isOne()

```
bool isOne (
            const Element & x ) const [inline]
```

16.248.3.5 isMOne()

```
bool isMOne (
            const Element & x ) const [inline]
```

16.248.3.6 isZero()

```
bool isZero (
            const Element & x ) const [inline]
```

16.248.3.7 characteristic() [1/2]

```
integer & characteristic (
            integer & p ) const [inline]
```

16.248.3.8 characteristic() [2/2]

```
integer characteristic ( ) const [inline]
```

16.248.3.9 cardinality() [1/2]

```
integer & cardinality (
            integer & p ) const [inline]
```

16.248.3.10 cardinality() [2/2]

```
integer cardinality ( ) const [inline]
```

16.248.3.11 minElement()

```
integer minElement ( ) const [inline]
```

16.248.3.12 maxElement()

```
integer maxElement ( ) const [inline]
```

16.248.3.13 init() [1/3]

```
Element & init (
            Element & x ) const  [inline]
```

16.248.3.14 init() [2/3]

```
Element & init (
            Element & x,
            const Givaro::Integer & y ) const  [inline]
```

16.248.3.15 reduce() [1/2]

```
Element & reduce (
            Element & x,
            const Element & y ) const  [inline]
```

16.248.3.16 reduce() [2/2]

```
Element & reduce (
            Element & x ) const  [inline]
```

16.248.3.17 init() [3/3]

```
Element & init (
            Element & x,
            const Element & y ) const  [inline]
```

16.248.3.18 convert()

```
Givaro::Integer convert (
            Givaro::Integer & x,
            const Element & y ) const  [inline]
```

16.248.3.19 assign()

```
Element & assign (
            Element & x,
            const Element & y ) const  [inline]
```

16.248.3.20 add()

```
Element & add (
            Element & x,
            const Element & y,
            const Element & z ) const  [inline]
```

16.248.3.21 sub()

```
Element & sub (
            Element & x,
```

```
            const Element & y,
            const Element & z ) const  [inline]
```

16.248.3.22 neg()

```
Element & neg (
            Element & x,
            const Element & y ) const  [inline]
```

16.248.3.23 mul()

```
Element & mul (
            Element & x,
            const Element & y,
            const Element & z ) const  [inline]
```

16.248.3.24 axpyin()

```
Element & axpyin (
            Element & x,
            const Element & y,
            const Element & z ) const  [inline]
```

16.248.3.25 inv()

```
Element & inv (
            Element & x,
            const Element & y ) const  [inline]
```

16.248.3.26 areEqual()

```
bool areEqual (
            const Element & x,
            const Element & y ) const  [inline]
```

16.248.3.27 write() [1/2]

```
std::ostream & write (
            std::ostream & os,
            const Element & y ) const  [inline]
```

16.248.3.28 write() [2/2]

```
std::ostream & write (
            std::ostream & os ) const  [inline]
```

16.248.3.29 reduce_modp() [1/2]

```
void reduce_modp (
            size_t n,
            Element_ptr B ) const  [inline]
```

16.248.3.30 write_matrix()

```
std::ostream & write_matrix (
            std::ostream & c,
            const double * E,
            int n,
            int m,
            int lda ) const  [inline]
```

16.248.3.31 write_matrix_long()

```
std::ostream & write_matrix_long (
            std::ostream & c,
            const double * E,
            int n,
            int m,
            int lda ) const  [inline]
```

16.248.3.32 reduce_modp() [2/2]

```
void reduce_modp (
            size_t m,
            size_t n,
            Element_ptr B,
            size_t lda ) const  [inline]
```

16.248.3.33 reduce_modp_rnsmajor()

```
void reduce_modp_rnsmajor (
            size_t n,
            Element_ptr B ) const  [inline]
```

16.248.4 Field Documentation

16.248.4.1 _p

```
integer _p  [protected]
```

16.248.4.2 _Mi_modp_rns

```
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _Mi_modp_↩
rns  [protected]
```

16.248.4.3 _iM_modp_rns

```
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _iM_modp_↩
rns  [protected]
```

16.248.4.4 _rns

```
const RNS* _rns  [protected]
```

16.248.4.5 _F

```
Givaro::Modular<Givaro::Integer> _F [protected]
```

16.248.4.6 _RNSdelayed

```
RNSInteger<RNS> _RNSdelayed [protected]
```

16.248.4.7 one

```
Element one
```

16.248.4.8 mOne

```
Element mOne
```

16.248.4.9 zero

```
Element zero
```

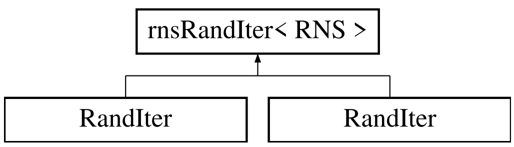
The documentation for this class was generated from the following files:

- field-traits.h
- rns-integer-mod.h

16.249 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

16.249.1 Constructor & Destructor Documentation

16.249.1.1 rnsRandIter()

```
rnsRandIter (
        const RNS & R,
        size_t size = 0,
        uint64_t seed = 0 ) [inline]
```

16.249.2 Member Function Documentation

16.249.2.1 random() [1/2]

```
RNS::Element & random (
            typename RNS::Element & elt ) const  [inline]
```

RNS ring Element random assignement.
Element is supposed to be initialized

Returns

random ring Element

16.249.2.2 operator()() [1/2]

```
RNS::Element & operator() (
            typename RNS::Element & elt ) const  [inline]
```

16.249.2.3 operator()() [2/2]

```
RNS::Element operator() ( ) const  [inline]
```

16.249.2.4 random() [2/2]

```
RNS::Element random ( ) const  [inline]
```

16.249.2.5 ring()

```
const RNS & ring ( ) const  [inline]
```

The documentation for this class was generated from the following file:

- rns-double.h

16.250 Row Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.251 ruint< K > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

16.252 ScalFunctions< Element, Enable > Struct Template Reference

The documentation for this struct was generated from the following file:

- test-simd.C

16.253 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)

16.253.1 Member Function Documentation

16.253.1.1 zero()

```
static Element zero ( )  [inline], [static]
```

16.253.1.2 vand()

```
static Element vand (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.3 vor()

```
static Element vor (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.4 vxor()

```
static Element vxor (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.5 vandnot()

```
static Element vandnot (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.6 ceil()

```
static Element ceil (
            Element x )  [inline], [static]
```

16.253.1.7 floor()

```
static Element floor (
            Element x )  [inline], [static]
```

16.253.1.8 round()

```
static Element round (
            Element x )  [inline], [static]
```

16.253.1.9 add()

```
static Element add (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.10 addin()

```
static Element addin (
            Element & x1,
            Element x2 )  [inline], [static]
```

16.253.1.11 sub()

```
static Element sub (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.12 subin()

```
static Element subin (
            Element & x1,
            Element x2 )  [inline], [static]
```

16.253.1.13 mul()

```
static Element mul (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.14 mulin()

```
static Element mulin (
            Element & x1,
            Element x2 )  [inline], [static]
```

16.253.1.15 div()

```
static Element div (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.16 fmadd()

```
static Element fmadd (
            Element x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.253.1.17 fmaddin()

```
static Element fmaddin (
            Element & x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.253.1.18 fmsub()

```
static Element fmsub (
            Element x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.253.1.19 fmsubin()

```
static Element fmsubin (
            Element & x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.253.1.20 fnmadd()

```
static Element fnmadd (
            Element x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.253.1.21 fnmaddin()

```
static Element fnmaddin (
            Element & x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.253.1.22 lesser()

```
static Element lesser (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.23 lesser_eq()

```
static Element lesser_eq (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.24 greater()

```
static Element greater (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.25 greater_eq()

```
static Element greater_eq (
            Element x1,
            Element x2 )  [inline], [static]
```

16.253.1.26 eq()

```
static Element eq (
            Element x1,
            Element x2 )  [inline], [static]
```

The documentation for this struct was generated from the following file:

- test-simd.C

16.254 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 fnmadd (Element x1, Element x2, Element x3)
- static Element fnmaddin (Element &x1, Element x2, Element x3)
- static Element fnmaddx (Element x1, Element x2, Element x3)
- static Element fnmaddxin (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)

16.254.1 Member Function Documentation

16.254.1.1 zero()

```
static Element zero ( )  [inline], [static]
```

16.254.1.2 round()

```
static Element round (
            Element x )  [inline], [static]
```

16.254.1.3 vand()

```
static Element vand (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.4 vor()

```
static Element vor (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.5 vxor()

```
static Element vxor (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.6 vandnot()

```
static Element vandnot (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.7 add()

```
static Element add (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.8 addin()

```
static Element addin (
            Element & x1,
            Element x2 )  [inline], [static]
```

16.254.1.9 sub()

```
static Element sub (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.10 subin()

```
static Element subin (
            Element & x1,
            Element x2 )  [inline], [static]
```

16.254.1.11 mul()

```
static Element mul (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.12 mullo()

```
static Element mullo (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.13 mulhi()

```
static Element mulhi (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.14 mulx()

```
static Element mulx (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.15 fmadd()

```
static Element fmadd (
            Element x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.254.1.16 fmaddin()

```
static Element fmaddin (
            Element & x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.254.1.17 fmaddx()

```
static Element fmaddx (
            Element x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.254.1.18 fmaddxin()

```
static Element fmaddxin (
            Element & x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.254.1.19 fmsub()

```
static Element fmsub (
            Element x1,
            Element x2,
            Element x3 )  [inline], [static]
```

16.254.1.20 fmsubin()

```
static Element fmsubin (
            Element & x1,
            Element x2,
            Element x3 ) [inline], [static]
```

16.254.1.21 fmsubx()

```
static Element fmsubx (
            Element x1,
            Element x2,
            Element x3 ) [inline], [static]
```

16.254.1.22 fmsubxin()

```
static Element fmsubxin (
            Element & x1,
            Element x2,
            Element x3 ) [inline], [static]
```

16.254.1.23 fnmadd()

```
static Element fnmadd (
            Element x1,
            Element x2,
            Element x3 ) [inline], [static]
```

16.254.1.24 fnmaddin()

```
static Element fnmaddin (
            Element & x1,
            Element x2,
            Element x3 ) [inline], [static]
```

16.254.1.25 fnmaddx()

```
static Element fnmaddx (
            Element x1,
            Element x2,
            Element x3 ) [inline], [static]
```

16.254.1.26 fnmaddxin()

```
static Element fnmaddxin (
            Element & x1,
            Element x2,
            Element x3 ) [inline], [static]
```

16.254.1.27 sra() [1/2]

```
static enable_if<!is_signed< Element >::value &&EnableTrue, Element >::type sra (
            Element x1 ) [inline], [static]
```

16.254.1.28 sra() [2/2]

```
static enable_if< is_signed< Element >::value &&EnableTrue, Element >::type sra (
            Element x1 )  [inline], [static]
```

16.254.1.29 srl()

```
static Element srl (
            Element x1 )  [inline], [static]
```

16.254.1.30 sll()

```
static Element sll (
            Element x1 )  [inline], [static]
```

16.254.1.31 lesser()

```
static Element lesser (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.32 lesser_eq()

```
static Element lesser_eq (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.33 greater()

```
static Element greater (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.34 greater_eq()

```
static Element greater_eq (
            Element x1,
            Element x2 )  [inline], [static]
```

16.254.1.35 eq()

```
static Element eq (
            Element x1,
            Element x2 )  [inline], [static]
```

The documentation for this struct was generated from the following file:

- test-simd.C

16.255 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 &)

16.255.1 Constructor & Destructor Documentation

16.255.1.1 Sequential() [1/3]

```
Sequential ( )  [inline]
```

16.255.1.2 Sequential() [2/3]

```
Sequential (
            size_t nth )  [inline]
```

16.255.1.3 Sequential() [3/3]

```
Sequential (
            Parallel< Cut, Param > &  )  [inline]
```

16.255.2 Member Function Documentation

16.255.2.1 numthreads()

```
size_t numthreads ( ) const  [inline]
```

16.255.3 Friends And Related Function Documentation

16.255.3.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Sequential &  )  [friend]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

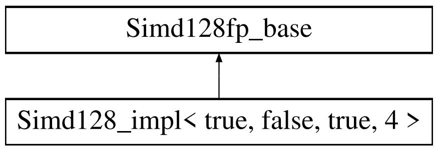
16.256 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd128.inl

16.257 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 ()

16.257.1 Member Function Documentation

16.257.1.1 type_string()

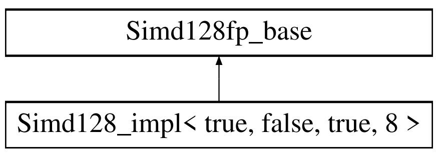
`static const std::string type_string ( ) [inline], [static], [inherited]`

The documentation for this struct was generated from the following file:

- simd128_float.inl

16.258 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 ()

16.258.1 Member Function Documentation

16.258.1.1 type_string()

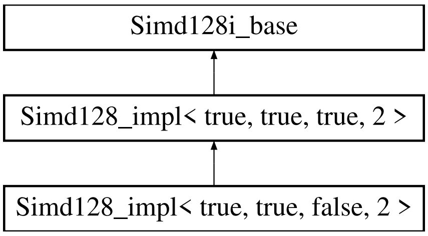
`static const std::string type_string ( ) [inline], [static], [inherited]`

The documentation for this struct was generated from the following file:

- simd128_double.inl

16.259 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

16.259.1 Member Typedef Documentation

16.259.1.1 scalar_t

```
using scalar_t = uint16_t
```

16.259.1.2 vect_t

```
using vect_t = __m128i  [inherited]
```

16.259.2 Member Function Documentation

16.259.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.259.2.2 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]
```

16.259.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.259.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.259.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.259.2.6 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.259.2.7 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.259.2.8 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.259.2.9 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.259.2.10 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b )  [inline], [static]
```

16.259.2.11 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b )  [inline], [static]
```

16.259.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.14 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.15 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.259.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.259.2.23 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr], [inherited]
```

16.259.2.24 compliant()

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr], [inherited]
```

16.259.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static], [inherited]
```

16.259.2.26 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]
```

16.259.2.27 gather() [2/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static], [inherited]
```

16.259.2.28 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static], [inherited]
```

16.259.2.29 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p ) [inline], [static], [inherited]
```

16.259.2.30 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v ) [inline], [static], [inherited]
```

16.259.2.31 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v ) [inline], [static], [inherited]
```

16.259.2.32 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static], [inherited]
```

16.259.2.33 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static], [inherited]
```

16.259.2.34 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static], [inherited]
```

16.259.2.35 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static], [inherited]
```

16.259.2.36 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.37 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.38 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.39 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.40 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.41 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.42 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.43 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.44 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.45 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.46 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.47 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.48 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.259.2.49 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.259.2.50 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.259.2.51 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.259.2.52 round()

```
static INLINE CONST vect_t round (
            const vect_t a ) [inline], [static], [inherited]
```

16.259.2.53 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]
```

16.259.2.54 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

16.259.2.55 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

16.259.2.56 sll128()

```
static INLINE CONST vect_t sll128 (
            const vect_t a ) [inline], [static], [inherited]
```

16.259.2.57 srl128()

```
static INLINE CONST vect_t srl128 (
            const vect_t a ) [inline], [static], [inherited]
```

16.259.2.58 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.259.2.59 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.259.2.60 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.259.2.61 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.259.3 Field Documentation

16.259.3.1 vect_size

```
const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

16.259.3.2 alignment

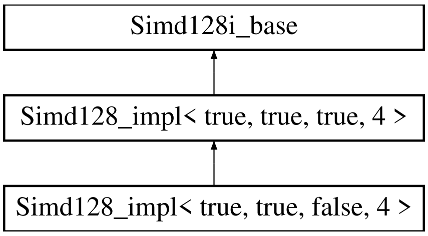
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int16.inl

16.260 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

16.260.1 Member Typedef Documentation

16.260.1.1 scalar_t

```
using scalar_t = uint32_t
```

16.260.1.2 vect_t

```
using vect_t = __m128i  [inherited]
```

16.260.2 Member Function Documentation

16.260.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.260.2.2 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]
```

16.260.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.260.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.260.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.260.2.6 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.260.2.7 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.260.2.8 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.260.2.9 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a ) [inline], [static]
```

16.260.2.10 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.260.2.11 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.260.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.260.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.260.2.14 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.260.2.15 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.260.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.260.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.260.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.260.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.260.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.260.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.260.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.260.2.23 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr], [inherited]
```

16.260.2.24 compliant()

```
static constexpr bool compliant (
            T n )  [inline], [static], [constexpr], [inherited]
```

16.260.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static], [inherited]
```

16.260.2.26 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]
```

16.260.2.27 gather() [2/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static], [inherited]
```

16.260.2.28 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static], [inherited]
```

16.260.2.29 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p ) [inline], [static], [inherited]
```

16.260.2.30 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v ) [inline], [static], [inherited]
```

16.260.2.31 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v ) [inline], [static], [inherited]
```

16.260.2.32 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v ) [inline], [static], [inherited]
```

16.260.2.33 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a ) [inline], [static], [inherited]
```

16.260.2.34 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a ) [inline], [static], [inherited]
```

16.260.2.35 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a ) [inline], [static], [inherited]
```

16.260.2.36 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.37 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.38 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.39 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.40 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.41 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.42 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.43 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.44 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.45 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.46 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.47 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.48 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.49 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.50 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.51 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.260.2.52 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static], [inherited]
```

16.260.2.53 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]
```

16.260.2.54 type_string()

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.260.2.55 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.260.2.56 sll128()

```
static INLINE CONST vect_t sll128 (
            const vect_t a )  [inline], [static], [inherited]
```

16.260.2.57 srl128()

```
static INLINE CONST vect_t srl128 (
            const vect_t a )  [inline], [static], [inherited]
```

16.260.2.58 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.260.2.59 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.260.2.60 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.2.61 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.260.3 Field Documentation

16.260.3.1 vect_size

```
const constexpr size_t vect_size = 4 [static], [constexpr], [inherited]
```

16.260.3.2 alignment

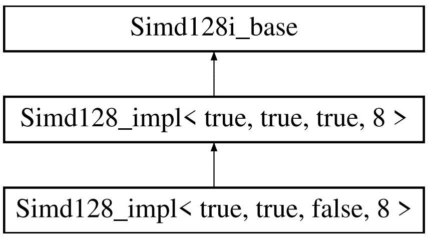
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int32.inl

16.261 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)

16.261.1 Member Typedef Documentation

16.261.1.1 scalar_t

```
using scalar_t = uint64_t
```

16.261.1.2 vect_t

```
using vect_t = __m128i [inherited]
```

16.261.2 Member Function Documentation

16.261.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static]
```

16.261.2.2 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1 ) [inline], [static]
```

16.261.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static]
```

16.261.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static]
```

16.261.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p ) [inline], [static]
```

16.261.2.6 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.261.2.7 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.261.2.8 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.261.2.9 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a ) [inline], [static]
```

16.261.2.10 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.261.2.11 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.261.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.14 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t x0,
            const vect_t x1 ) [inline], [static]
```

16.261.2.15 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.21 fmsubxin() [1/2]

```
static INLINE CONST vect_t fmsubxin (
            vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.261.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a ) [inline], [static]
```

16.261.2.23 valid()

```
static constexpr bool valid (
            T * p ) [inline], [static], [constexpr], [inherited]
```

16.261.2.24 compliant()

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr], [inherited]
```

16.261.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static], [inherited]
```

16.261.2.26 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1 ) [inline], [static], [inherited]
```

16.261.2.27 gather() [2/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static], [inherited]
```

16.261.2.28 get()

```
static INLINE CONST scalar_t get (
            vect_t v ) [inline], [static], [inherited]
```

16.261.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.261.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.261.2.31 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.261.2.32 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.261.2.33 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static], [inherited]
```

16.261.2.34 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static], [inherited]
```

16.261.2.35 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static], [inherited]
```

16.261.2.36 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static], [inherited]
```

16.261.2.37 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.38 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.39 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.40 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.41 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.42 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.43 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.44 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.45 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.46 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.47 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.48 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.49 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.50 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.51 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.52 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.53 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static], [inherited]
```

16.261.2.54 mask_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

16.261.2.55 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y ) [static], [inherited]
```

16.261.2.56 mod()

```
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], [inherited]
```

16.261.2.57 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x ) [inline], [static], [protected], [inherited]
```

16.261.2.58 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

16.261.2.59 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

16.261.2.60 sll128()

```
static INLINE CONST vect_t sll128 (
            const vect_t a ) [inline], [static], [inherited]
```

16.261.2.61 srl128()

```
static INLINE CONST vect_t srl128 (
            const vect_t a ) [inline], [static], [inherited]
```

16.261.2.62 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.261.2.63 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.64 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.2.65 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.261.3 Field Documentation

16.261.3.1 vect_size

```
const constexpr size_t vect_size = 2  [static], [constexpr], [inherited]
```

16.261.3.2 alignment

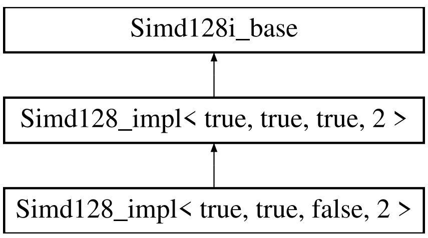
```
const constexpr size_t alignment = 16  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int64.inl

16.262 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 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 __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

16.262.1 Member Typedef Documentation

16.262.1.1 vect_t

```
using vect_t = __m128i
```

16.262.1.2 scalar_t

```
using scalar_t = int16_t
```

16.262.2 Member Function Documentation

16.262.2.1 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr]
```

16.262.2.2 compliant()

```
static constexpr bool compliant (
            T n )  [inline], [static], [constexpr]
```

16.262.2.3 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.262.2.4 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]
```

16.262.2.5 gather()

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.262.2.6 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.262.2.7 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.262.2.8 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.262.2.9 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.262.2.10 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.262.2.11 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static]
```

16.262.2.12 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static]
```

16.262.2.13 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.262.2.14 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static]
```

16.262.2.15 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.16 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.17 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.18 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.19 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.262.2.20 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.21 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.262.2.22 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.23 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.24 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.25 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.26 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.27 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.28 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.29 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.30 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.31 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.32 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.33 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.34 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.35 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.36 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.38 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.39 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.40 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.41 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.42 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.262.2.43 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.262.2.44 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static]
```

16.262.2.45 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]
```

16.262.2.46 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

16.262.2.47 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

16.262.2.48 sll128()

```
static INLINE CONST vect_t sll128 (
            const vect_t a ) [inline], [static], [inherited]
```

16.262.2.49 srl128()

```
static INLINE CONST vect_t srl128 (
            const vect_t a ) [inline], [static], [inherited]
```

16.262.2.50 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.262.2.51 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.262.2.52 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.262.2.53 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.262.3 Field Documentation

16.262.3.1 vect_size

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```

16.262.3.2 alignment

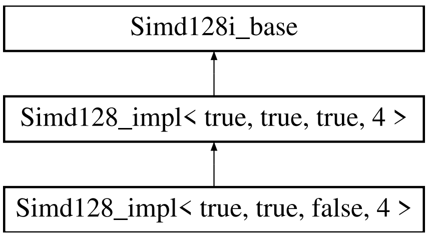
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int16.inl

16.263 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

16.263.1 Member Typedef Documentation

16.263.1.1 vect_t

```
using vect_t = __m128i
```

16.263.1.2 scalar_t

```
using scalar_t = int32_t
```

16.263.2 Member Function Documentation

16.263.2.1 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr]
```

16.263.2.2 compliant()

```
static constexpr bool compliant (
            T n )  [inline], [static], [constexpr]
```

16.263.2.3 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.263.2.4 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]
```

16.263.2.5 gather()

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.263.2.6 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.263.2.7 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.263.2.8 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.263.2.9 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.263.2.10 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.263.2.11 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static]
```

16.263.2.12 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static]
```

16.263.2.13 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.263.2.14 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static]
```

16.263.2.15 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.16 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.17 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.18 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.19 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b ) [inline], [static]
```

16.263.2.20 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.21 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b ) [inline], [static]
```

16.263.2.22 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.23 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.24 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.25 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.26 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.27 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.28 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.29 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.30 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.31 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.32 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.33 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.34 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.35 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.36 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.263.2.38 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
```

```
            const vect_t b )  [inline], [static]
```

16.263.2.39 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.263.2.40 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.263.2.41 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.263.2.42 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.263.2.43 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.263.2.44 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static]
```

16.263.2.45 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]
```

16.263.2.46 type_string()

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.263.2.47 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

16.263.2.48 sll128()

```
static INLINE CONST vect_t sll128 (
            const vect_t a ) [inline], [static], [inherited]
```

16.263.2.49 srl128()

```
static INLINE CONST vect_t srl128 (
            const vect_t a ) [inline], [static], [inherited]
```

16.263.2.50 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.263.2.51 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.263.2.52 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.263.2.53 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.263.3 Field Documentation

16.263.3.1 vect_size

```
const constexpr size_t vect_size = 4 [static], [constexpr]
```

16.263.3.2 alignment

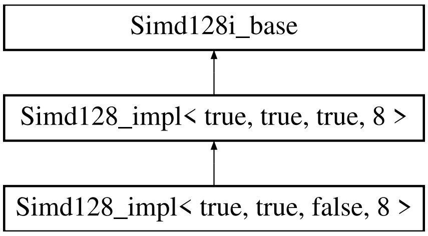
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int32.inl

16.264 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 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 __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)

16.264.1 Member Typedef Documentation

16.264.1.1 vect_t

```
using vect_t = __m128i
```

16.264.1.2 scalar_t

```
using scalar_t = int64_t
```

16.264.2 Member Function Documentation

16.264.2.1 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr]
```

16.264.2.2 compliant()

```
static constexpr bool compliant (
            T n )  [inline], [static], [constexpr]
```

16.264.2.3 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.264.2.4 set()

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1 )  [inline], [static]
```

16.264.2.5 gather()

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.264.2.6 get()

```
static INLINE CONST scalar_t get (
            vect_t v )  [inline], [static]
```

16.264.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.264.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.264.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.264.2.10 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.264.2.11 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.264.2.12 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static]
```

16.264.2.13 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static]
```

16.264.2.14 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.264.2.15 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static]
```

16.264.2.16 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.17 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.18 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.19 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.20 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.264.2.21 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.22 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.264.2.23 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t x0,
            const vect_t x1 )  [inline], [static]
```

16.264.2.24 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.25 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.26 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.27 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.28 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.29 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.30 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.31 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.32 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.33 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.34 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.35 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.36 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.38 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.39 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.40 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.41 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.264.2.42 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.264.2.43 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a ) [inline], [static]
```

16.264.2.44 round()

```
static INLINE CONST vect_t round (
            const vect_t a ) [inline], [static]
```

16.264.2.45 mask_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

16.264.2.46 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y ) [static]
```

16.264.2.47 mod()

```
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]
```

16.264.2.48 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x ) [inline], [static], [protected]
```

16.264.2.49 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

16.264.2.50 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

16.264.2.51 sll128()

```
static INLINE CONST vect_t sll128 (
            const vect_t a )  [inline], [static], [inherited]
```

16.264.2.52 srl128()

```
static INLINE CONST vect_t srl128 (
            const vect_t a )  [inline], [static], [inherited]
```

16.264.2.53 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.264.2.54 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.264.2.55 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.264.2.56 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.264.3 Field Documentation

16.264.3.1 vect_size

```
const constexpr size_t vect_size = 2  [static], [constexpr]
```

16.264.3.2 alignment

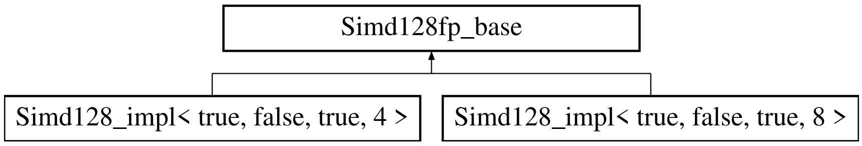
```
const constexpr size_t alignment = 16  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int64.inl

16.265 Simd128fp_base Struct Reference

Inheritance diagram for Simd128fp_base:



Static Public Member Functions

- static const std::string type_string ()

16.265.1 Member Function Documentation

16.265.1.1 type_string()

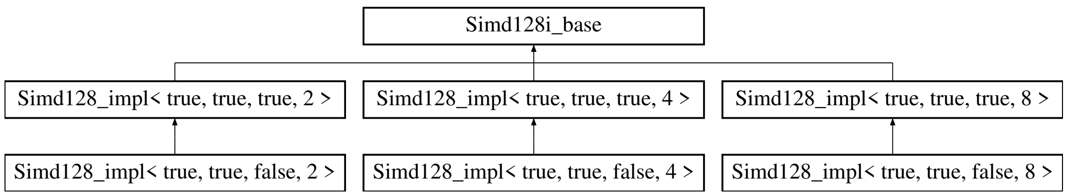
```
static const std::string type_string ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- simd128.inl

16.266 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)

16.266.1 Member Typedef Documentation

16.266.1.1 vect_t

```
using vect_t = __m128i
```

16.266.2 Member Function Documentation

16.266.2.1 type_string()

```
static const std::string type_string ( )  [inline], [static]
```

16.266.2.2 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static]
```

16.266.2.3 sll128()

```
static INLINE CONST vect_t sll128 (
            const vect_t a )  [inline], [static]
```

16.266.2.4 srl128()

```
static INLINE CONST vect_t srl128 (
            const vect_t a )  [inline], [static]
```

16.266.2.5 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.266.2.6 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.266.2.7 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.266.2.8 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

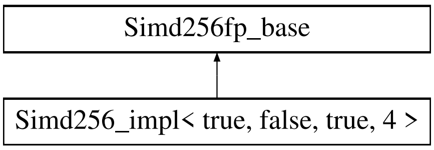
16.267 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd256.inl

16.268 Simd256_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd256_impl< true, false, true, 4 >:

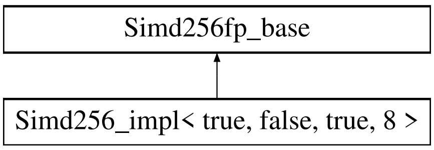


The documentation for this struct was generated from the following file:

- simd256_float.inl

16.269 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

16.269.1 Member Typedef Documentation

16.269.1.1 vect_t

```
using vect_t = __m256d
```

16.269.1.2 scalar_t

```
using scalar_t = double
```

16.269.2 Member Function Documentation

16.269.2.1 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr]
```

16.269.2.2 compliant()

```
static constexpr bool compliant (
            T n )  [inline], [static], [constexpr]
```

16.269.2.3 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static]
```

16.269.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.269.2.5 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]
```

16.269.2.6 gather()

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.269.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.269.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.269.2.9 store()

```
static INLINE void store (
            const scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.269.2.10 storeu()

```
static INLINE void storeu (
            const scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.269.2.11 stream()

```
static INLINE void stream (
            const scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.269.2.12 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.13 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.14 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.15 blendv()

```
static INLINE CONST vect_t blendv (
            const vect_t a,
            const vect_t b,
            const vect_t mask ) [inline], [static]
```

16.269.2.16 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.17 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b ) [inline], [static]
```

16.269.2.18 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.269.2.19 subin()

```
static INLINE CONST vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.269.2.20 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.269.2.21 mulin()

```
static INLINE CONST vect_t mulin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.269.2.22 div()

```
static INLINE CONST vect_t div (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.269.2.23 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.269.2.24 fmaddin()

```
static INLINE CONST vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.269.2.25 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.269.2.26 fnmaddin()

```
static INLINE CONST vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.27 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.28 fmsubin()

```
static INLINE CONST vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.29 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.30 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.31 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.32 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.33 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.34 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.35 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.36 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.37 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.38 floor()

```
static INLINE CONST vect_t floor (
            const vect_t a ) [inline], [static]
```

16.269.2.39 ceil()

```
static INLINE CONST vect_t ceil (
            const vect_t a ) [inline], [static]
```

16.269.2.40 round()

```
static INLINE CONST vect_t round (
            const vect_t a ) [inline], [static]
```

16.269.2.41 hadd()

```
static INLINE CONST vect_t hadd (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.269.2.42 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a ) [inline], [static]
```

16.269.2.43 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]
```

16.269.3 Field Documentation

16.269.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr]
```

16.269.3.2 alignment

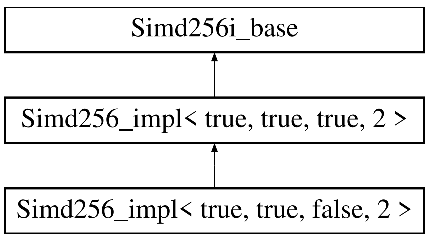
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_double.inl

16.270 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 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 ()

Static Public Attributes

- static const constexpr size_t vect_size = 16
- static const constexpr size_t alignment = 32

16.270.1 Member Typedef Documentation

16.270.1.1 scalar_t

```
using scalar_t = uint16_t
```

16.270.1.2 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

16.270.1.3 vect_t

```
using vect_t = __m256i  [inherited]
```

16.270.1.4 half_t

```
using half_t = __m128i  [inherited]
```

16.270.2 Member Function Documentation

16.270.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.270.2.2 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]
```

16.270.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.270.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.270.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.270.2.6 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.270.2.7 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.270.2.8 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.270.2.9 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a ) [inline], [static]
```

16.270.2.10 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.270.2.11 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.270.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.270.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.270.2.14 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.270.2.15 mulx()

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.270.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.270.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.270.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.270.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.270.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.270.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.270.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.270.2.23 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr], [inherited]
```

16.270.2.24 compliant()

```
static constexpr bool compliant (
            T n )  [inline], [static], [constexpr], [inherited]
```

16.270.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static], [inherited]
```

16.270.2.26 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]
```

16.270.2.27 gather() [2/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static], [inherited]
```

16.270.2.28 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.270.2.29 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.270.2.30 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.270.2.31 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.270.2.32 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static], [inherited]
```

16.270.2.33 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a ) [inline], [static], [inherited]
```

16.270.2.34 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a ) [inline], [static], [inherited]
```

16.270.2.35 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a ) [inline], [static], [inherited]
```

16.270.2.36 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.270.2.37 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.270.2.38 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.270.2.39 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.270.2.40 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & s1,
            vect_t & s2,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.270.2.41 blend_twice()

```
static INLINE CONST vect_t blend_twice (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.270.2.42 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.43 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.44 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.45 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.46 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.47 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.48 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.49 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.50 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.51 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.52 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.53 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.54 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.270.2.55 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static], [inherited]
```

16.270.2.56 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]
```

16.270.2.57 type_string()

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.270.2.58 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.270.3 Field Documentation

16.270.3.1 vect_size

```
const constexpr size_t vect_size = 16  [static], [constexpr], [inherited]
```

16.270.3.2 alignment

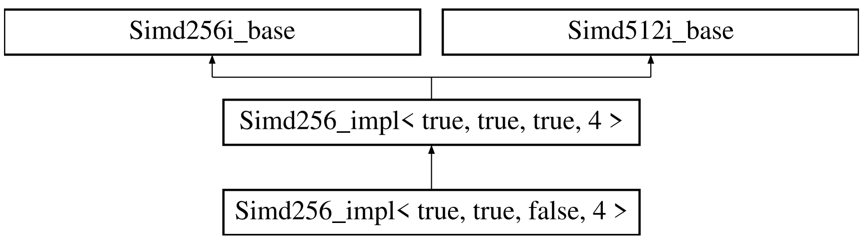
```
const constexpr size_t alignment = 32  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd256_int16.inl

16.271 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 fmadd (const vect_t c, 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 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 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

16.271.1 Member Typedef Documentation

16.271.1.1 scalar_t [1/2]

```
using scalar_t = uint32_t
```

16.271.1.2 simdHalf [1/2]

```
using simdHalf = Simd128<scalar_t>
```

16.271.1.3 scalar_t [2/2]

```
using scalar_t = uint32_t
```

16.271.1.4 simdHalf [2/2]

```
using simdHalf = Simd128<scalar_t>
```

16.271.1.5 vect_t [1/2]

```
using vect_t = __m256i  [inherited]
```

16.271.1.6 vect_t [2/2]

```
using vect_t = __m512i  [inherited]
```

16.271.1.7 half_t [1/2]

```
using half_t = __m128i  [inherited]
```

16.271.1.8 half_t [2/2]

```
using half_t = __m256i  [inherited]
```

16.271.2 Member Function Documentation

16.271.2.1 set1() [1/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.271.2.2 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]
```

16.271.2.3 gather() [1/3]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.271.2.4 load() [1/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.271.2.5 loadu() [1/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.271.2.6 store() [1/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.271.2.7 storeu() [1/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.271.2.8 stream() [1/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.271.2.9 sra() [1/2]

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.271.2.10 greater() [1/2]

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b )  [inline], [static]
```

16.271.2.11 lesser() [1/2]

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b )  [inline], [static]
```

16.271.2.12 greater_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.13 lesser_eq() [1/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.14 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.15 mulx() [1/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.271.2.16 fmaddx() [1/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.17 fmaddxin() [1/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.18 fnmaddx() [1/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.19 fnmaddxin() [1/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.20 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.21 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.22 hadd_to_scal() [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a ) [inline], [static]
```

16.271.2.23 set1() [2/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static]
```

16.271.2.24 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]
```

16.271.2.25 gather() [2/3]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static]
```

16.271.2.26 load() [2/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static]
```

16.271.2.27 loadu() [2/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p ) [inline], [static]
```

16.271.2.28 store() [2/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.271.2.29 storeu() [2/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.271.2.30 stream() [2/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.271.2.31 sra() [2/2]

```
static INLINE CONST vect_t sra (
            const vect_t a ) [inline], [static]
```

16.271.2.32 greater() [2/2]

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.271.2.33 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.271.2.34 greater_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.35 lesser_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.36 mulhi() [2/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.271.2.37 mulx() [2/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.271.2.38 fmaddx() [2/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.39 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.40 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.41 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.42 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.43 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.271.2.44 hadd_to_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.271.2.45 valid() [1/2]

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr], [inherited]
```

16.271.2.46 valid() [2/2]

```
static constexpr bool valid (
            T * p ) [inline], [static], [constexpr], [inherited]
```

16.271.2.47 compliant() [1/2]

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr], [inherited]
```

16.271.2.48 compliant() [2/2]

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr], [inherited]
```

16.271.2.49 set1() [3/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static], [inherited]
```

16.271.2.50 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]
```

16.271.2.51 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]
```

16.271.2.52 gather() [3/3]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static], [inherited]
```

16.271.2.53 load() [3/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.271.2.54 loadu() [3/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.271.2.55 store() [3/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.271.2.56 storeu() [3/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.271.2.57 stream() [3/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static], [inherited]
```

16.271.2.58 sll() [1/2]

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.59 sll() [2/2]

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.60 srl() [1/2]

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.61 srl() [2/2]

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.62 shuffle_twice() [1/2]

```
static INLINE CONST vect_t shuffle_twice (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.63 shuffle_twice() [2/2]

```
static INLINE CONST vect_t shuffle_twice (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.64 shuffle() [1/2]

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.65 shuffle() [2/2]

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.66 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.67 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.68 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.69 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.70 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & s1,
            vect_t & s2,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.71 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.72 add() [1/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.73 add() [2/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.74 addin() [1/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.75 addin() [2/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.76 sub() [1/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.77 sub() [2/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.78 subin() [1/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.79 subin() [2/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.80 mullo() [1/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.81 mullo() [2/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.82 mul() [1/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.83 mul() [2/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.84 fmadd() [1/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.85 fmadd() [2/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.86 fmaddin() [1/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.87 fmaddin() [2/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.88 fnmadd() [1/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.89 fnmadd() [2/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.90 fnmaddin() [1/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.91 fnmaddin() [2/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.92 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.271.2.93 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.94 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.95 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.96 eq() [1/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.97 eq() [2/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.98 round() [1/2]

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.99 round() [2/2]

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static], [inherited]
```

16.271.2.100 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]
```

16.271.2.101 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]
```

16.271.2.102 type_string() [1/2]

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.271.2.103 type_string() [2/2]

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.271.2.104 zero() [1/2]

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.271.2.105 zero() [2/2]

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.271.2.106 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.107 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.108 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.2.109 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.271.3 Field Documentation

16.271.3.1 vect_size

```
static const constexpr size_t vect_size = 8  [static], [constexpr], [inherited]
```

16.271.3.2 alignment

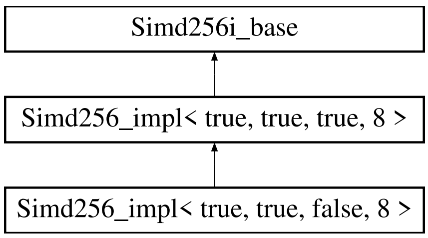
```
static const constexpr size_t alignment = 32  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

16.272 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)

16.272.1 Member Typedef Documentation

16.272.1.1 scalar_t

```
using scalar_t = uint64_t
```

16.272.1.2 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

16.272.1.3 vect_t

```
using vect_t = __m256i  [inherited]
```

16.272.1.4 half_t

```
using half_t = __m128i  [inherited]
```

16.272.2 Member Function Documentation

16.272.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.272.2.2 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]
```

16.272.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.272.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.272.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.272.2.6 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.272.2.7 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.272.2.8 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.272.2.9 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.272.2.10 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b )  [inline], [static]
```

16.272.2.11 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b )  [inline], [static]
```

16.272.2.12 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.272.2.13 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.272.2.14 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.272.2.15 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.272.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.272.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.272.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.272.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.272.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.272.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.272.2.22 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.272.2.23 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr], [inherited]
```

16.272.2.24 compliant()

```
static constexpr bool compliant (
            T n )  [inline], [static], [constexpr], [inherited]
```

16.272.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static], [inherited]
```

16.272.2.26 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]
```

16.272.2.27 gather() [2/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static], [inherited]
```

16.272.2.28 get()

```
static INLINE CONST scalar_t get (
            vect_t v )  [inline], [static], [inherited]
```

16.272.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.272.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.272.2.31 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.272.2.32 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.272.2.33 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static], [inherited]
```

16.272.2.34 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static], [inherited]
```

16.272.2.35 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static], [inherited]
```

16.272.2.36 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static], [inherited]
```

16.272.2.37 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.38 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.39 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.272.2.40 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.272.2.41 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & l,
            vect_t & h,
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.272.2.42 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.272.2.43 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.272.2.44 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b ) [inline], [static], [inherited]
```

16.272.2.45 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.272.2.46 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b ) [inline], [static], [inherited]
```

16.272.2.47 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.48 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.49 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.50 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.51 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.52 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.53 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.54 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.272.2.55 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static], [inherited]
```

16.272.2.56 mask_high()

```
static INLINE CONST vect_t mask_high ( )  [inline], [static], [inherited]
```

16.272.2.57 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y )  [static], [inherited]
```

16.272.2.58 mod()

```
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], [inherited]
```

16.272.2.59 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x )  [inline], [static], [protected], [inherited]
```

16.272.2.60 type_string()

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.272.2.61 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.272.3 Field Documentation

16.272.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr], [inherited]
```

16.272.3.2 alignment

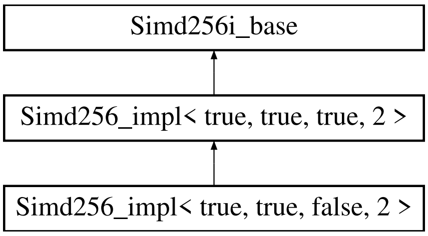
`const constexpr size_t alignment = 32 [static], [constexpr], [inherited]`

The documentation for this struct was generated from the following file:

- simd256_int64.inl

16.273 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

16.273.1 Member Typedef Documentation

16.273.1.1 vect_t

```
using vect_t = __m256i
```

16.273.1.2 half_t

```
using half_t = __m128i
```

16.273.1.3 scalar_t

```
using scalar_t = int16_t
```

16.273.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

16.273.2 Member Function Documentation

16.273.2.1 valid()

```
static constexpr bool valid (
            T * p )  [inline], [static], [constexpr]
```

16.273.2.2 compliant()

```
static constexpr bool compliant (
            T n )  [inline], [static], [constexpr]
```

16.273.2.3 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.273.2.4 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]
```

16.273.2.5 gather()

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.273.2.6 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static]
```

16.273.2.7 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p ) [inline], [static]
```

16.273.2.8 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.273.2.9 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.273.2.10 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.273.2.11 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a ) [inline], [static]
```

16.273.2.12 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a ) [inline], [static]
```

16.273.2.13 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a ) [inline], [static]
```

16.273.2.14 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a ) [inline], [static]
```

16.273.2.15 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.16 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.17 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.18 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.19 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & s1,
            vect_t & s2,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.20 blend_twice()

```
static INLINE CONST vect_t blend_twice (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.21 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.22 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.273.2.23 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.24 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.273.2.25 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.26 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.27 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.28 mulx()

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b )  [inline], [static]
```

16.273.2.29 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.30 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.31 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.32 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.33 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.34 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.35 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.36 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.37 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.38 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.39 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.40 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.41 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.42 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.43 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.44 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.45 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.273.2.46 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.273.2.47 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static]
```

16.273.2.48 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]
```

16.273.2.49 type_string()

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.273.2.50 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.273.3 Field Documentation

16.273.3.1 vect_size

```
const constexpr size_t vect_size = 16  [static], [constexpr]
```

16.273.3.2 alignment

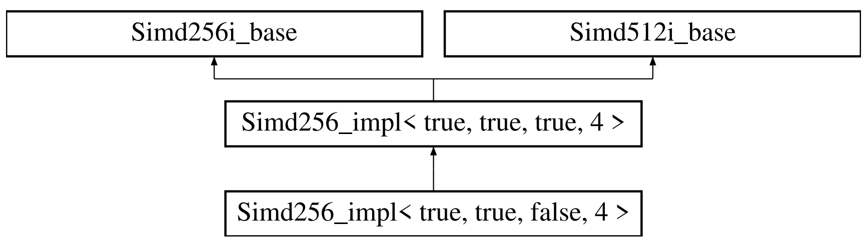
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_int16.inl

16.274 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 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 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

16.274.1 Member Typedef Documentation

16.274.1.1 vect_t [1/2]

```
using vect_t = __m256i
```

16.274.1.2 half_t [1/2]

```
using half_t = __m128i
```

16.274.1.3 scalar_t [1/2]

```
using scalar_t = int32_t
```

16.274.1.4 simdHalf [1/2]

```
using simdHalf = Simd128<scalar_t>
```

16.274.1.5 vect_t [2/2]

```
using vect_t = __m512i
```

16.274.1.6 half_t [2/2]

```
using half_t = __m256i
```

16.274.1.7 scalar_t [2/2]

```
using scalar_t = int32_t
```

16.274.1.8 simdHalf [2/2]

```
using simdHalf = Simd256<scalar_t>
```

16.274.2 Member Function Documentation

16.274.2.1 valid() [1/2]

```
static constexpr bool valid (
            T * p ) [inline], [static], [constexpr]
```

16.274.2.2 compliant() [1/2]

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr]
```

16.274.2.3 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static]
```

16.274.2.4 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]
```

16.274.2.5 gather() [1/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static]
```

16.274.2.6 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static]
```

16.274.2.7 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p ) [inline], [static]
```

16.274.2.8 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.274.2.9 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.274.2.10 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.274.2.11 sll() [1/2]

```
static INLINE CONST vect_t sll (
            const vect_t a ) [inline], [static]
```

16.274.2.12 srl() [1/2]

```
static INLINE CONST vect_t srl (
            const vect_t a ) [inline], [static]
```

16.274.2.13 sra() [1/2]

```
static INLINE CONST vect_t sra (
            const vect_t a ) [inline], [static]
```

16.274.2.14 shuffle_twice() [1/2]

```
static INLINE CONST vect_t shuffle_twice (
            const vect_t a ) [inline], [static]
```

16.274.2.15 shuffle() [1/2]

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static]
```

16.274.2.16 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.17 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.20 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & s1,
            vect_t & s2,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.21 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.22 add() [1/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.23 addin() [1/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b ) [inline], [static]
```

16.274.2.24 sub() [1/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.25 subin() [1/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b ) [inline], [static]
```

16.274.2.26 mullo() [1/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.27 mul() [1/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.28 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.29 mulx() [1/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.274.2.30 fmadd() [1/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.31 fmaddin() [1/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.32 fmaddx() [1/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.33 fmaddxin() [1/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.34 fnmadd() [1/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.35 fnmaddin() [1/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.36 fnmaddx() [1/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.37 fnmaddxin() [1/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.38 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.39 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.40 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.41 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.42 eq() [1/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.43 greater() [1/2]

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.44 lesser() [1/2]

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.45 greater_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.46 lesser_eq() [1/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.47 hadd_to_scal() [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a ) [inline], [static]
```

16.274.2.48 round() [1/2]

```
static INLINE CONST vect_t round (
            const vect_t a ) [inline], [static]
```

16.274.2.49 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]
```

16.274.2.50 valid() [2/2]

```
static constexpr bool valid (
            T * p ) [inline], [static], [constexpr]
```

16.274.2.51 compliant() [2/2]

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr]
```

16.274.2.52 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static]
```

16.274.2.53 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]
```

16.274.2.54 gather() [2/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.274.2.55 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.274.2.56 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.274.2.57 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.274.2.58 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.274.2.59 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.274.2.60 sll() [2/2]

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static]
```

16.274.2.61 srl() [2/2]

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static]
```

16.274.2.62 sra() [2/2]

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.274.2.63 shuffle_twice() [2/2]

```
static INLINE CONST vect_t shuffle_twice (
            const vect_t a )  [inline], [static]
```

16.274.2.64 shuffle() [2/2]

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static]
```

16.274.2.65 add() [2/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.66 addin() [2/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.274.2.67 sub() [2/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.68 subin() [2/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.274.2.69 mullo() [2/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.70 mul() [2/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.71 mulhi() [2/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.72 mulx() [2/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.274.2.73 fmadd() [2/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.74 fmaddin() [2/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.75 fmaddx() [2/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.76 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.77 fnmadd() [2/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.78 fnmaddin() [2/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.79 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.80 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.81 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.82 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.83 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.84 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.274.2.85 eq() [2/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
```

```
            const vect_t b )  [inline], [static]
```

16.274.2.86 greater() [2/2]

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.87 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.88 greater_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.89 lesser_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.274.2.90 hadd_to_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.274.2.91 round() [2/2]

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static]
```

16.274.2.92 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]
```

16.274.2.93 type_string() [1/2]

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.274.2.94 zero() [1/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

16.274.2.95 type_string() [2/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

16.274.2.96 zero() [2/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

16.274.2.97 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.274.2.98 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.274.2.99 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.274.2.100 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b ) [inline], [static], [inherited]
```

16.274.3 Field Documentation

16.274.3.1 vect_size

```
static const constexpr size_t vect_size = 8 [static], [constexpr]
```

16.274.3.2 alignment

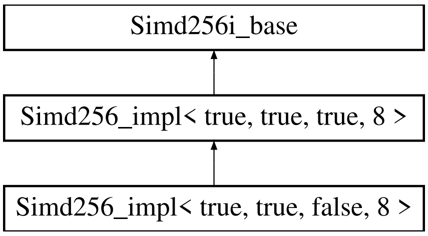
```
static const constexpr size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

16.275 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)

16.275.1 Member Typedef Documentation

16.275.1.1 vect_t

```
using vect_t = __m256i
```

16.275.1.2 half_t

```
using half_t = __m128i
```

16.275.1.3 scalar_t

```
using scalar_t = int64_t
```

16.275.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

16.275.2 Member Function Documentation

16.275.2.1 valid()

```
static constexpr bool valid (
            T * p ) [inline], [static], [constexpr]
```

16.275.2.2 compliant()

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr]
```

16.275.2.3 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static]
```

16.275.2.4 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]
```

16.275.2.5 gather()

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static]
```

16.275.2.6 get()

```
static INLINE CONST scalar_t get (
            vect_t v ) [inline], [static]
```

16.275.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static]
```

16.275.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.275.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.275.2.10 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.275.2.11 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.275.2.12 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static]
```

16.275.2.13 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static]
```

16.275.2.14 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.275.2.15 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static]
```

16.275.2.16 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.17 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.20 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & l,
            vect_t & h,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.21 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.22 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.23 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b ) [inline], [static]
```

16.275.2.24 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.25 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b ) [inline], [static]
```

16.275.2.26 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.275.2.27 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.28 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.29 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.30 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.31 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.32 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.275.2.33 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.34 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.35 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.36 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.37 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.38 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.39 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.40 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.41 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.42 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.43 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.44 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.45 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.275.2.46 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.275.2.47 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static]
```

16.275.2.48 mask_high()

```
static INLINE CONST vect_t mask_high ( )  [inline], [static]
```

16.275.2.49 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y )  [static]
```

16.275.2.50 mod()

```
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]
```

16.275.2.51 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x )  [inline], [static], [protected]
```

16.275.2.52 type_string()

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.275.2.53 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.275.3 Field Documentation

16.275.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr]
```

16.275.3.2 alignment

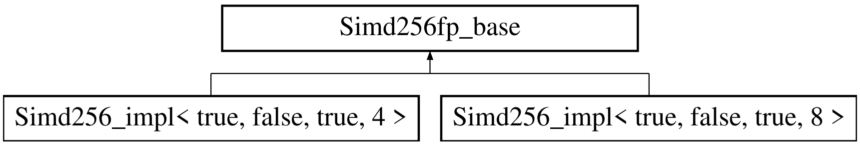
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_int64.inl

16.276 Simd256fp_base Struct Reference

Inheritance diagram for Simd256fp_base:

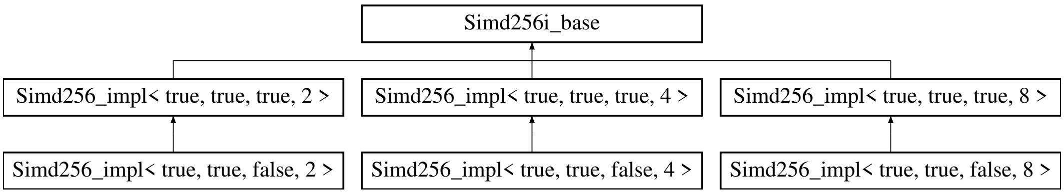


The documentation for this struct was generated from the following file:

- simd256.inl

16.277 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 ()

16.277.1 Member Typedef Documentation

16.277.1.1 vect_t

```
using vect_t = __m256i
```

16.277.2 Member Function Documentation

16.277.2.1 type_string()

```
static const std::string type_string ( ) [inline], [static]
```

16.277.2.2 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- simd256.inl

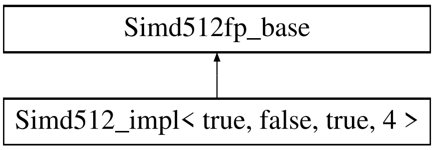
16.278 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd512.inl

16.279 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 ()

16.279.1 Member Function Documentation

16.279.1.1 type_string()

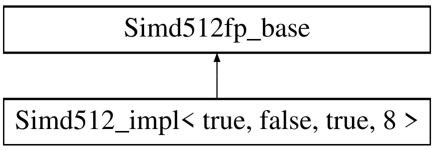
```
static const std::string type_string ( )  [inline], [static], [inherited]
```

The documentation for this struct was generated from the following file:

- simd512_float.inl

16.280 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

16.280.1 Member Typedef Documentation

16.280.1.1 vect_t

```
using vect_t = __m512d
```

16.280.1.2 scalar_t

```
using scalar_t = double
```

16.280.2 Member Function Documentation

16.280.2.1 valid()

```
static constexpr bool valid (
            T * p ) [inline], [static], [constexpr]
```

16.280.2.2 compliant()

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr]
```

16.280.2.3 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

16.280.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static]
```

16.280.2.5 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]
```

16.280.2.6 gather()

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static]
```

16.280.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static]
```

16.280.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p ) [inline], [static]
```

16.280.2.9 store()

```
static INLINE void store (
            const scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.280.2.10 storeu()

```
static INLINE void storeu (
            const scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.280.2.11 stream()

```
static INLINE void stream (
            const scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.280.2.12 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a ) [inline], [static]
```

16.280.2.13 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.14 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.15 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.16 blendv()

```
static INLINE CONST vect_t blendv (
            const vect_t a,
            const vect_t b,
            const vect_t mask ) [inline], [static]
```

16.280.2.17 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.280.2.18 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.280.2.19 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.280.2.20 subin()

```
static INLINE CONST vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.280.2.21 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.280.2.22 mulin()

```
static INLINE CONST vect_t mulin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.280.2.23 div()

```
static INLINE CONST vect_t div (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.280.2.24 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.280.2.25 fmaddin()

```
static INLINE CONST vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.26 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.27 fnmaddin()

```
static INLINE CONST vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.28 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.29 fmsubin()

```
static INLINE CONST vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.30 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.31 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.32 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.33 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.34 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.35 floor()

```
static INLINE CONST vect_t floor (
            const vect_t a ) [inline], [static]
```

16.280.2.36 ceil()

```
static INLINE CONST vect_t ceil (
            const vect_t a ) [inline], [static]
```

16.280.2.37 round()

```
static INLINE CONST vect_t round (
            const vect_t a ) [inline], [static]
```

16.280.2.38 hadd()

```
static INLINE CONST vect_t hadd (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.280.2.39 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a ) [inline], [static]
```

16.280.2.40 type_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

16.280.3 Field Documentation

16.280.3.1 vect_size

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```

16.280.3.2 alignment

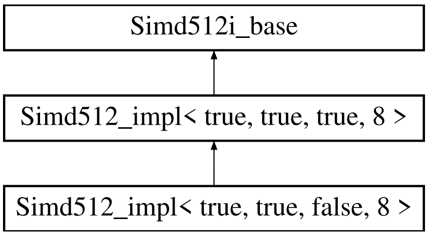
`const constexpr size_t alignment = 64 [static], [constexpr]`

The documentation for this struct was generated from the following file:

- simd512_double.inl

16.281 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 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 __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)

16.281.1 Member Typedef Documentation

16.281.1.1 scalar_t

```
using scalar_t = uint64_t
```

16.281.1.2 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

16.281.1.3 vect_t

```
using vect_t = __m512i  [inherited]
```

16.281.1.4 half_t

```
using half_t = __m256i  [inherited]
```

16.281.2 Member Function Documentation

16.281.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x )  [inline], [static]
```

16.281.2.2 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]
```

16.281.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx ) [inline], [static]
```

16.281.2.4 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p ) [inline], [static]
```

16.281.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p ) [inline], [static]
```

16.281.2.6 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.281.2.7 maskstore() [1/2]

```
static INLINE void maskstore (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.281.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v ) [inline], [static]
```

16.281.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v ) [inline], [static]
```

16.281.2.10 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a ) [inline], [static]
```

16.281.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.281.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.281.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.15 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b ) [inline], [static]
```

16.281.2.16 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.17 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.281.2.23 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a ) [inline], [static]
```

16.281.2.24 valid()

```
static constexpr bool valid (
            T * p ) [inline], [static], [constexpr], [inherited]
```

16.281.2.25 compliant()

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr], [inherited]
```

16.281.2.26 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static], [inherited]
```

16.281.2.27 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]
```

16.281.2.28 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]
```

16.281.2.29 gather() [2/2]

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static], [inherited]
```

16.281.2.30 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.281.2.31 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static], [inherited]
```

16.281.2.32 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.281.2.33 maskstore() [2/2]

```
static INLINE void maskstore (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.281.2.34 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static], [inherited]
```

16.281.2.35 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static], [inherited]
```

16.281.2.36 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static], [inherited]
```

16.281.2.37 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static], [inherited]
```

16.281.2.38 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static], [inherited]
```

16.281.2.39 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.40 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.41 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.42 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.43 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & l,
            vect_t & h,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.44 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.45 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.46 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.47 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.48 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.49 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.50 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.51 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.52 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.53 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.54 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.55 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.56 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.57 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static], [inherited]
```

16.281.2.58 mask_high()

```
static INLINE CONST vect_t mask_high ( )  [inline], [static], [inherited]
```

16.281.2.59 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y )  [static], [inherited]
```

16.281.2.60 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]
```

16.281.2.61 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x )  [inline], [static], [protected], [inherited]
```

16.281.2.62 type_string()

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.281.2.63 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.281.2.64 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.65 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.66 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.2.67 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.281.3 Field Documentation

16.281.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr], [inherited]
```

16.281.3.2 alignment

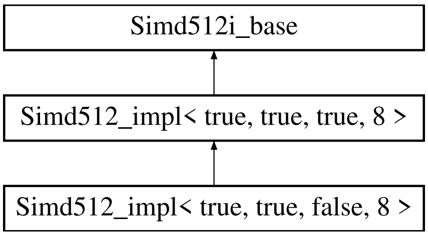
`const constexpr size_t alignment = 64 [static], [constexpr], [inherited]`

The documentation for this struct was generated from the following file:

- simd512_int64.inl

16.282 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 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 __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)

16.282.1 Member Typedef Documentation

16.282.1.1 vect_t

```
using vect_t = __m512i
```

16.282.1.2 half_t

```
using half_t = __m256i
```

16.282.1.3 scalar_t

```
using scalar_t = int64_t
```

16.282.1.4 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

16.282.2 Member Function Documentation

16.282.2.1 valid()

```
static constexpr bool valid (
            T * p ) [inline], [static], [constexpr]
```

16.282.2.2 compliant()

```
static constexpr bool compliant (
            T n ) [inline], [static], [constexpr]
```

16.282.2.3 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x ) [inline], [static]
```

16.282.2.4 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]
```

16.282.2.5 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]
```

16.282.2.6 gather()

```
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx )  [inline], [static]
```

16.282.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p )  [inline], [static]
```

16.282.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p )  [inline], [static]
```

16.282.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.282.2.10 maskstore()

```
static INLINE void maskstore (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.282.2.11 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v )  [inline], [static]
```

16.282.2.12 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v )  [inline], [static]
```

16.282.2.13 sll()

```
static INLINE CONST vect_t sll (
            const vect_t a )  [inline], [static]
```

16.282.2.14 srl()

```
static INLINE CONST vect_t srl (
            const vect_t a )  [inline], [static]
```

16.282.2.15 sra()

```
static INLINE CONST vect_t sra (
            const vect_t a )  [inline], [static]
```

16.282.2.16 shuffle()

```
static INLINE CONST vect_t shuffle (
            const vect_t a )  [inline], [static]
```

16.282.2.17 unpacklo_twice()

```
static INLINE CONST vect_t unpacklo_twice (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.18 unpackhi_twice()

```
static INLINE CONST vect_t unpackhi_twice (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.19 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.20 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.21 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & l,
            vect_t & h,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.22 blend()

```
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.23 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.24 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.282.2.25 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.26 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b )  [inline], [static]
```

16.282.2.27 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b )  [inline], [static]
```

16.282.2.28 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.29 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.30 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.31 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.32 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.33 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.34 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.35 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.36 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.37 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.38 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.282.2.39 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.282.2.40 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.282.2.41 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.282.2.42 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.282.2.43 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.282.2.44 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.282.2.45 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.282.2.46 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b )  [inline], [static]
```

16.282.2.47 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a )  [inline], [static]
```

16.282.2.48 round()

```
static INLINE CONST vect_t round (
            const vect_t a )  [inline], [static]
```

16.282.2.49 mask_high()

```
static INLINE CONST vect_t mask_high ( )  [inline], [static]
```

16.282.2.50 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y )  [static]
```

16.282.2.51 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]
```

16.282.2.52 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x )  [inline], [static], [protected]
```

16.282.2.53 type_string()

```
static const std::string type_string ( )  [inline], [static], [inherited]
```

16.282.2.54 zero()

```
static INLINE CONST vect_t zero ( )  [inline], [static], [inherited]
```

16.282.2.55 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.282.2.56 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.282.2.57 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.282.2.58 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b )  [inline], [static], [inherited]
```

16.282.3 Field Documentation

16.282.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr]
```

16.282.3.2 alignment

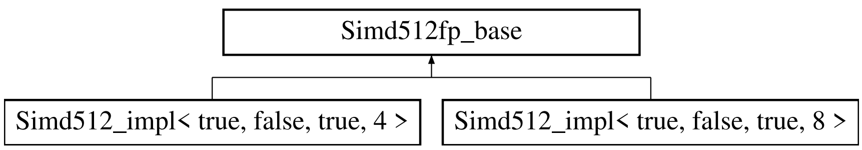
```
const constexpr size_t alignment = 64  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd512_int64.inl

16.283 Simd512fp_base Struct Reference

Inheritance diagram for Simd512fp_base:



Static Public Member Functions

- static const std::string type_string ()

16.283.1 Member Function Documentation

16.283.1.1 type_string()

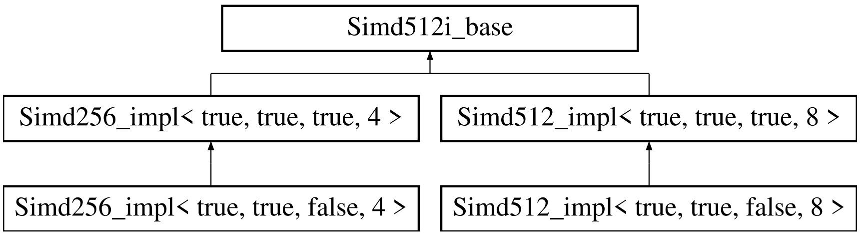
```
static const std::string type_string ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- simd512.inl

16.284 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)

16.284.1 Member Typedef Documentation

16.284.1.1 vect_t

```
using vect_t = __m512i
```

16.284.2 Member Function Documentation

16.284.2.1 type_string()

```
static const std::string type_string ( ) [inline], [static]
```

16.284.2.2 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

16.284.2.3 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.284.2.4 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.284.2.5 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b ) [inline], [static]
```

16.284.2.6 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

16.285 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

16.286 SimdChooser< T, false, b > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using value = NoSimd< T >

16.286.1 Member Typedef Documentation

16.286.1.1 value

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

16.287 SimdChooser< T, true, false > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using value = NoSimd< T >

16.287.1 Member Typedef Documentation

16.287.1.1 value

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

16.288 SimdChooser< T, true, true > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using value = NoSimd< T >

16.288.1 Member Typedef Documentation

16.288.1.1 value

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

16.289 simdToType< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- fflas_simd.h

16.290 Single Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.291 Sparse< Field, SparseMatrix_t, IdxT, PtrT > Struct Template Reference

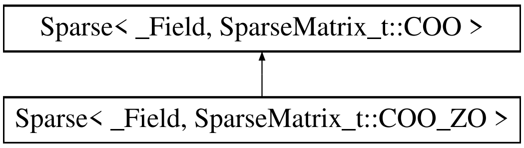
The documentation for this struct was generated from the following file:

- fflas_sparse.h

16.292 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

16.292.1 Member Typedef Documentation

16.292.1.1 Field

```
using Field = _Field
```

16.292.2 Field Documentation

16.292.2.1 col

```
index_t* col = nullptr
```

16.292.2.2 row

```
index_t* row = nullptr
```

16.292.2.3 dat

```
_Field::Element_ptr dat
```

16.292.2.4 delayed

```
bool delayed = false
```

16.292.2.5 kmax

```
uint64_t kmax = 0
```

16.292.2.6 m

```
index_t m = 0
```

16.292.2.7 n

```
index_t n = 0
```

16.292.2.8 nnz

```
uint64_t nnz = 0
```

16.292.2.9 nElements

```
uint64_t nElements = 0
```

16.292.2.10 maxrow

```
uint64_t maxrow = 0
```

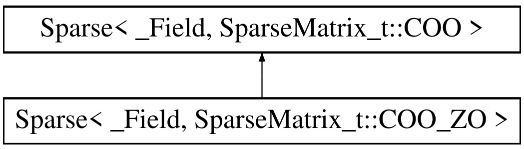
The documentation for this struct was generated from the following file:

- coo.h

16.293 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

16.293.1 Member Typedef Documentation

16.293.1.1 Field

```
using Field = _Field
```

16.293.2 Field Documentation

16.293.2.1 cst

```
_Field::Element cst = 1
```

16.293.2.2 col

```
index_t* col = nullptr  [inherited]
```

16.293.2.3 row

```
index_t* row = nullptr  [inherited]
```

16.293.2.4 dat

```
_Field::Element_ptr dat  [inherited]
```

16.293.2.5 delayed

```
bool delayed = false  [inherited]
```

16.293.2.6 kmax

```
uint64_t kmax = 0  [inherited]
```

16.293.2.7 m

```
index_t m = 0  [inherited]
```

16.293.2.8 n

```
index_t n = 0  [inherited]
```

16.293.2.9 nnz

`uint64_t nnz = 0 [inherited]`

16.293.2.10 nElements

`uint64_t nElements = 0 [inherited]`

16.293.2.11 maxrow

`uint64_t maxrow = 0 [inherited]`

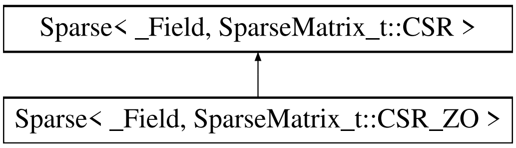
The documentation for this struct was generated from the following file:

- coo.h

16.294 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

16.294.1 Member Typedef Documentation

16.294.1.1 Field

`using Field = _Field`

16.294.2 Field Documentation

16.294.2.1 delayed

```
bool delayed = false
```

16.294.2.2 kmax

```
uint64_t kmax = 0
```

16.294.2.3 m

```
index_t m = 0
```

16.294.2.4 n

```
index_t n = 0
```

16.294.2.5 nnz

```
uint64_t nnz = 0
```

16.294.2.6 nElements

```
uint64_t nElements = 0
```

16.294.2.7 maxrow

```
uint64_t maxrow = 0
```

16.294.2.8 col

```
index_t* col = nullptr
```

16.294.2.9 st

```
index_t* st = nullptr
```

16.294.2.10 stend

```
index_t* stend = nullptr
```

16.294.2.11 dat

```
_Field::Element_ptr dat
```

The documentation for this struct was generated from the following file:

- csr.h

16.295 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

16.295.1 Member Typedef Documentation

16.295.1.1 Field

```
using Field = _Field
```

16.295.2 Field Documentation

16.295.2.1 delayed

```
bool delayed = false
```

16.295.2.2 col

```
index_t* col = nullptr
```

16.295.2.3 st

```
index_t* st = nullptr
```

16.295.2.4 dat

```
_Field::Element_ptr dat
```

16.295.2.5 kmax

```
uint64_t kmax = 0
```

16.295.2.6 m

```
index_t m = 0
```

16.295.2.7 n

```
index_t n = 0
```

16.295.2.8 nnz

```
uint64_t nnz = 0
```

16.295.2.9 nElements

```
uint64_t nElements = 0
```

16.295.2.10 maxrow

```
uint64_t maxrow = 0
```

16.295.2.11 nOnes

```
uint64_t nOnes = 0
```

16.295.2.12 nMOnes

```
uint64_t nMOnes = 0
```

16.295.2.13 nOthers

```
uint64_t nOthers = 0
```

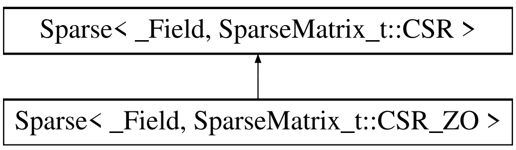
The documentation for this struct was generated from the following file:

- csr_hyb.h

16.296 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

16.296.1 Member Typedef Documentation

16.296.1.1 Field

```
using Field = _Field
```

16.296.2 Field Documentation

16.296.2.1 cst

```
int64_t cst = 1
```

16.296.2.2 delayed

```
bool delayed = false
```

16.296.2.3 kmax

```
uint64_t kmax = 0 [inherited]
```

16.296.2.4 m

```
index_t m = 0 [inherited]
```

16.296.2.5 n

```
index_t n = 0 [inherited]
```

16.296.2.6 nnz

```
uint64_t nnz = 0 [inherited]
```

16.296.2.7 nElements

```
uint64_t nElements = 0 [inherited]
```

16.296.2.8 maxrow

```
uint64_t maxrow = 0 [inherited]
```

16.296.2.9 col

```
index_t* col = nullptr [inherited]
```

16.296.2.10 st

```
index_t* st = nullptr [inherited]
```

16.296.2.11 stend

```
index_t* stend = nullptr [inherited]
```

16.296.2.12 dat

```
_Field::Element_ptr dat [inherited]
```

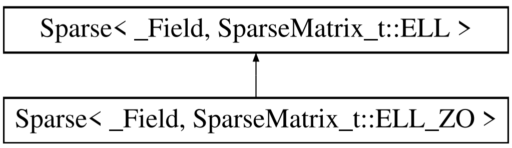
The documentation for this struct was generated from the following file:

- csr.h

16.297 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

16.297.1 Member Typedef Documentation

16.297.1.1 Field

```
using Field = _Field
```

16.297.2 Field Documentation

16.297.2.1 delayed

```
bool delayed = false
```

16.297.2.2 kmax

```
uint64_t kmax = 0
```

16.297.2.3 m

```
index_t m = 0
```

16.297.2.4 n

```
index_t n = 0
```

16.297.2.5 ld

```
index_t ld = 0
```

16.297.2.6 nnz

```
uint64_t nnz = 0
```

16.297.2.7 nElements

```
uint64_t nElements = 0
```

16.297.2.8 maxrow

```
uint64_t maxrow = 0
```

16.297.2.9 col

```
index_t* col = nullptr
```

16.297.2.10 dat

```
_Field::Element_ptr dat
```

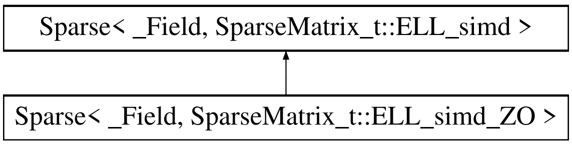
The documentation for this struct was generated from the following file:

- ell.h

16.298 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

16.298.1 Field Documentation

16.298.1.1 delayed

```
bool delayed = false
```

16.298.1.2 chunk

```
int chunk = 0
```

16.298.1.3 m

```
index_t m = 0
```

16.298.1.4 n

```
index_t n = 0
```

16.298.1.5 ld

`index_t ld = 0`

16.298.1.6 kmax

`uint64_t kmax = 0`

16.298.1.7 nnz

`uint64_t nnz = 0`

16.298.1.8 nElements

`uint64_t nElements = 0`

16.298.1.9 maxrow

`uint64_t maxrow = 0`

16.298.1.10 nChunks

`uint64_t nChunks = 0`

16.298.1.11 col

`index_t* col = nullptr`

16.298.1.12 dat

`_Field::Element_ptr dat`

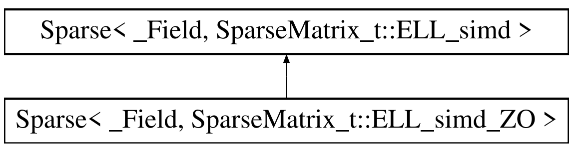
The documentation for this struct was generated from the following file:

- ell_simd.h

16.299 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

16.299.1 Field Documentation

16.299.1.1 cst

`_Field::Element cst = 1`

16.299.1.2 delayed

`bool delayed = false [inherited]`

16.299.1.3 chunk

`int chunk = 0 [inherited]`

16.299.1.4 m

`index_t m = 0 [inherited]`

16.299.1.5 n

`index_t n = 0 [inherited]`

16.299.1.6 ld

`index_t ld = 0 [inherited]`

16.299.1.7 kmax

`uint64_t kmax = 0 [inherited]`

16.299.1.8 nnz

`uint64_t nnz = 0 [inherited]`

16.299.1.9 nElements

```
uint64_t nElements = 0 [inherited]
```

16.299.1.10 maxrow

```
uint64_t maxrow = 0 [inherited]
```

16.299.1.11 nChunks

```
uint64_t nChunks = 0 [inherited]
```

16.299.1.12 col

```
index_t* col = nullptr [inherited]
```

16.299.1.13 dat

```
_Field::Element_ptr dat [inherited]
```

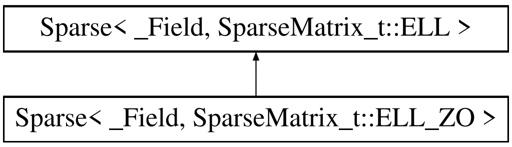
The documentation for this struct was generated from the following file:

- ell_simd.h

16.300 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

16.300.1 Member Typedef Documentation

16.300.1.1 Field

`using Field = _Field`

16.300.2 Field Documentation

16.300.2.1 cst

`_Field::Element cst = 1`

16.300.2.2 delayed

`bool delayed = false [inherited]`

16.300.2.3 kmax

`uint64_t kmax = 0 [inherited]`

16.300.2.4 m

`index_t m = 0 [inherited]`

16.300.2.5 n

`index_t n = 0 [inherited]`

16.300.2.6 ld

`index_t ld = 0 [inherited]`

16.300.2.7 nnz

`uint64_t nnz = 0 [inherited]`

16.300.2.8 nElements

`uint64_t nElements = 0 [inherited]`

16.300.2.9 maxrow

`uint64_t maxrow = 0 [inherited]`

16.300.2.10 col

`index_t* col = nullptr [inherited]`

16.300.2.11 dat

```
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- ell.h

16.301 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

16.301.1 Member Typedef Documentation

16.301.1.1 Field

```
using Field = _Field
```

16.301.1.2 Self_t

```
typedef Sparse<_Field, SparseMatrix_t::HYB_ZO> Self_t
```

16.301.2 Field Documentation

16.301.2.1 delayed

```
bool delayed = false
```

16.301.2.2 kmax

```
uint64_t kmax = 0
```

16.301.2.3 m

```
index_t m = 0
```

16.301.2.4 n

```
index_t n = 0
```

16.301.2.5 nnz

```
uint64_t nnz = 0
```

16.301.2.6 maxrow

```
uint64_t maxrow = 0
```

16.301.2.7 nElements

```
uint64_t nElements = 0
```

16.301.2.8 dat

```
Sparse<_Field, SparseMatrix_t::CSR>* dat = nullptr
```

16.301.2.9 one

```
Sparse<_Field, SparseMatrix_t::CSR_ZO>* one = nullptr
```

16.301.2.10 mone

```
Sparse<_Field, SparseMatrix_t::CSR_ZO>* mone = nullptr
```

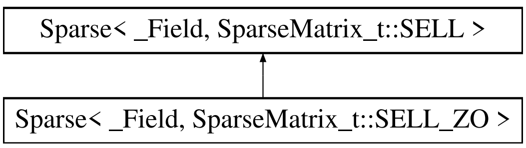
The documentation for this struct was generated from the following file:

- hyb_zo.h

16.302 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

16.302.1 Member Typedef Documentation

16.302.1.1 Field

```
using Field = _Field
```

16.302.2 Field Documentation

16.302.2.1 delayed

```
bool delayed = false
```

16.302.2.2 chunk

```
int chunk = 0
```

16.302.2.3 kmax

```
index_t kmax = 0
```

16.302.2.4 m

```
index_t m = 0
```

16.302.2.5 n

```
index_t n = 0
```

16.302.2.6 maxrow

```
index_t maxrow = 0
```

16.302.2.7 sigma

`index_t sigma = 0`

16.302.2.8 nChunks

`index_t nChunks = 0`

16.302.2.9 nnz

`uint64_t nnz = 0`

16.302.2.10 nElements

`uint64_t nElements = 0`

16.302.2.11 perm

`index_t* perm = nullptr`

16.302.2.12 st

`uint64_t* st = nullptr`

16.302.2.13 chunkSize

`index_t* chunkSize = nullptr`

16.302.2.14 col

`index_t* col = nullptr`

16.302.2.15 dat

`_Field::Element_ptr dat`

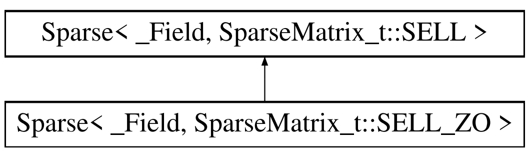
The documentation for this struct was generated from the following file:

- sell.h

16.303 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

16.303.1 Member Typedef Documentation

16.303.1.1 Field

```
using Field = _Field
```

16.303.2 Field Documentation

16.303.2.1 cst

```
_Field::Element cst = 1
```

16.303.2.2 delayed

```
bool delayed = false [inherited]
```

16.303.2.3 chunk

```
int chunk = 0 [inherited]
```

16.303.2.4 kmax

```
index_t kmax = 0 [inherited]
```

16.303.2.5 m

```
index_t m = 0 [inherited]
```

16.303.2.6 n

`index_t n = 0 [inherited]`

16.303.2.7 maxrow

`index_t maxrow = 0 [inherited]`

16.303.2.8 sigma

`index_t sigma = 0 [inherited]`

16.303.2.9 nChunks

`index_t nChunks = 0 [inherited]`

16.303.2.10 nnz

`uint64_t nnz = 0 [inherited]`

16.303.2.11 nElements

`uint64_t nElements = 0 [inherited]`

16.303.2.12 perm

`index_t* perm = nullptr [inherited]`

16.303.2.13 st

`uint64_t* st = nullptr [inherited]`

16.303.2.14 chunkSize

`index_t* chunkSize = nullptr [inherited]`

16.303.2.15 col

`index_t* col = nullptr [inherited]`

16.303.2.16 dat

`_Field::Element_ptr dat [inherited]`

The documentation for this struct was generated from the following file:

- sell.h

16.304 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

16.304.1 Field Documentation

16.304.1.1 _coo

`FFLAS::CooMat<Field>* _coo = nullptr`

16.304.1.2 _csr

`FFLAS::CsrMat<Field>* _csr = nullptr`

16.304.1.3 _ell

`FFLAS::EllMat<Field>* _ell = nullptr`

The documentation for this struct was generated from the following file:

- fflas_sparse.h

16.305 Static_error_check< bool > Class Template Reference

`#include <instrset.h>`

Public Member Functions

- Static_error_check ()

16.305.1 Constructor & Destructor Documentation

16.305.1.1 Static_error_check()

`Static_error_check ( ) [inline]`

The documentation for this class was generated from the following file:

- instrset.h

16.306 Static_error_check< false > Class Reference

`#include <instrset.h>`

The documentation for this class was generated from the following file:

- instrset.h

16.307 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

16.307.1 Field Documentation

16.307.1.1 rowdim

```
uint64_t rowdim = 0
```

16.307.1.2 coldim

```
uint64_t coldim = 0
```

16.307.1.3 nOnes

```
uint64_t nOnes = 0
```

16.307.1.4 nMOnes

```
uint64_t nMOnes = 0
```

16.307.1.5 nOthers

```
uint64_t nOthers = 0
```

16.307.1.6 nnz

```
uint64_t nnz = 0
```

16.307.1.7 maxRow

```
uint64_t maxRow = 0
```

16.307.1.8 minRow

```
uint64_t minRow = 0
```

16.307.1.9 averageRow

```
uint64_t averageRow = 0
```

16.307.1.10 deviationRow

```
uint64_t deviationRow = 0
```

16.307.1.11 maxCol

```
uint64_t maxCol = 0
```

16.307.1.12 minCol

```
uint64_t minCol = 0
```

16.307.1.13 averageCol

```
uint64_t averageCol = 0
```

16.307.1.14 deviationCol

```
uint64_t deviationCol = 0
```

16.307.1.15 minColDifference

```
uint64_t minColDifference = 0
```

16.307.1.16 maxColDifference

```
uint64_t maxColDifference = 0
```

16.307.1.17 averageColDifference

```
uint64_t averageColDifference = 0
```

16.307.1.18 deviationColDifference

```
uint64_t deviationColDifference = 0
```

16.307.1.19 minRowDifference

```
uint64_t minRowDifference = 0
```

16.307.1.20 maxRowDifference

```
uint64_t maxRowDifference = 0
```

16.307.1.21 averageRowDifference

```
uint64_t averageRowDifference = 0
```

16.307.1.22 deviationRowDifference

```
uint64_t deviationRowDifference = 0
```

16.307.1.23 nDenseRows

```
uint64_t nDenseRows = 0
```

16.307.1.24 nDenseCols

```
uint64_t nDenseCols = 0
```

16.307.1.25 nEmptyRows

```
uint64_t nEmptyRows = 0
```

16.307.1.26 nEmptyCols

```
uint64_t nEmptyCols = 0
```

16.307.1.27 nEmptyColsEnd

```
uint64_t nEmptyColsEnd = 0
```

16.307.1.28 denseRows

```
std::vector<uint64_t> denseRows
```

16.307.1.29 denseCols

```
std::vector<uint64_t> denseCols
```

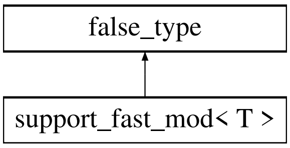
The documentation for this struct was generated from the following file:

- utils.h

16.308 support_fast_mod< T > Struct Template Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< T >:



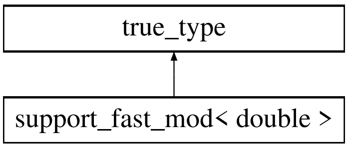
The documentation for this struct was generated from the following file:

- fflas_freduce.h

16.309 support_fast_mod< double > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< double >:



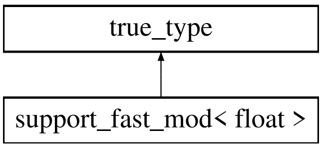
The documentation for this struct was generated from the following file:

- fflas_freduce.h

16.310 support_fast_mod< float > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< float >:



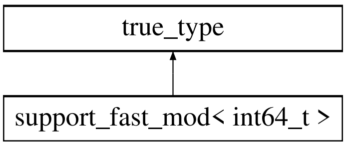
The documentation for this struct was generated from the following file:

- fflas_freduce.h

16.311 support_fast_mod< int64_t > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< int64_t >:



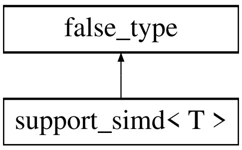
The documentation for this struct was generated from the following file:

- fflas_freduce.h

16.312 support_simd< T > Struct Template Reference

`#include <fflas_simd.h>`

Inheritance diagram for support_simd< T >:



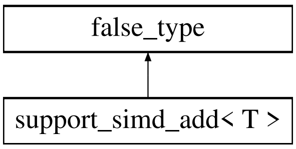
The documentation for this struct was generated from the following file:

- fflas_simd.h

16.313 support_simd_add< T > Struct Template Reference

`#include <fflas_fadd.h>`

Inheritance diagram for support_simd_add< T >:



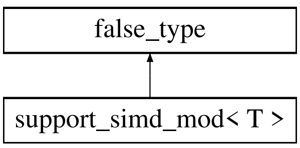
The documentation for this struct was generated from the following file:

- fflas_fadd.h

16.314 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

16.315 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)...))

16.315.1 Member Function Documentation

16.315.1.1 operator()()

```
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

16.316 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)...))

16.316.1 Member Function Documentation

16.316.1.1 operator()()

```
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

16.317 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)...))

16.317.1 Member Function Documentation

16.317.1.1 operator()()

```
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

16.318 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)...))

16.318.1 Member Function Documentation

16.318.1.1 operator()()

```
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

16.319 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)...))

16.319.1 Member Function Documentation

16.319.1.1 operator()()

```
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

16.320 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)...))

16.320.1 Member Function Documentation

16.320.1.1 operator()()

```
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

16.321 Threads Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.322 ThreeD Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.323 ThreeDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.324 ThreeDInPlace Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.325 TRSMHelper< RecIterTrait, 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, Par↩ SeqTrait 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

16.325.1 Detailed Description

template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
struct FFLAS::TRSMHelper< RecIterTrait, ParSeqTrait >

TRSM Helper.

16.325.2 Constructor & Destructor Documentation

16.325.2.1 TRSMHelper() [1/3]

```
TRSMHelper (
            ParSeqHelper::Parallel< Cut, Param > _PS )  [inline]
```

16.325.2.2 TRSMHelper() [2/3]

```
TRSMHelper (
            ParSeqHelper::Sequential _PS )  [inline]
```

16.325.2.3 TRSMHelper() [3/3]

```
TRSMHelper (
            TRSMHelper< RIT, PST > & _TH )  [inline]
```

16.325.3 Member Function Documentation

16.325.3.1 pMMH() [1/2]

```
FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (
            Dom & D,
            size_t m,
            size_t k,
            size_t n,
            ParSeqTrait p ) const  [inline]
```

16.325.3.2 pMMH() [2/2]

```
FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (
            Dom & D,
            size_t m,
            size_t k,
            size_t n ) const  [inline]
```

16.325.4 Field Documentation

16.325.4.1 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.326 TwoD Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.327 TwoDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

16.328 UnparametricTag Struct Reference

If the field uses a representation with infix operators.

```
#include <field-traits.h>
```

16.328.1 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

16.329 Winograd Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

16.330 WinogradPar Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

Chapter 17

File Documentation

17.1 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)

17.1.1 Macro Definition Documentation

17.1.1.1 CUBE

```
#define CUBE(
            x ) ((x)*(x)*(x))
```

17.1.1.2 GFOPS

```
#define GFOPS(
            m,
            n,
            r,
            t ) (2.7*CUBE(double(n)/1000.0))/t
```

17.1.2 Typedef Documentation

17.1.2.1 TTimer

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

17.1.3 Function Documentation

17.1.3.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.2 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 ()

17.2.1 Macro Definition Documentation

17.2.1.1 CUBE

```
#define CUBE(
             x ) ((x)*(x)*(x))
```

17.2.1.2 GFOPS

```
#define GFOPS(
             n,
             t ) (CUBE(double(n)/1000.0)/(3.0*t))
```

17.2.2 Typedef Documentation

17.2.2.1 TTimer

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

17.2.3 Function Documentation

17.2.3.1 main()

```
int main (
            void )
```

17.3 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 ()

17.3.1 Macro Definition Documentation

17.3.1.1 CUBE

```
#define CUBE(
             x ) ((x)*(x)*(x))
```

17.3.1.2 GFOPS

```
#define GFOPS(
             n,
             t ) (CUBE(double(n)/1000.0)/(3.0*t))
```

17.3.2 Typedef Documentation

17.3.2.1 TTimer

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

17.3.3 Function Documentation

17.3.3.1 main()

```
int main (
            void )
```

17.4 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 ()

17.4.1 Macro Definition Documentation

17.4.1.1 CUBE

```
#define CUBE(
            x ) ((x)*(x)*(x))
```

17.4.1.2 GFOPS

```
#define GFOPS(
            n,
            t ) (CUBE(double(n)/1000.0)/(3.0*t))
```

17.4.2 Typedef Documentation

17.4.2.1 TTimer

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

17.4.3 Function Documentation

17.4.3.1 main()

```
int main (
            void )
```

17.5 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 ()

17.5.1 Macro Definition Documentation

17.5.1.1 DOUBLE_TO_FLOAT_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 0
```

17.5.1.2 GFOPS

```
#define GFOPS(
            n,
            t ) (2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)
```

17.5.2 Typedef Documentation

17.5.2.1 TTimer

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

17.5.3 Function Documentation

17.5.3.1 balanced() [1/2]

```
bool balanced (
            const Field & )
```

17.5.3.2 balanced() [2/2]

```
bool balanced (
            const Givaro::ModularBalanced< T > & )
```

17.5.3.3 main()

```
int main (
            void )
```

17.6 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)

17.6.1 Macro Definition Documentation

17.6.1.1 __FFLASFFPACK_FORCE_SEQ

```
#define __FFLASFFPACK_FORCE_SEQ
```

17.6.2 Function Documentation

17.6.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.7 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)

17.7.1 Macro Definition Documentation

17.7.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.7.2 Function Documentation

17.7.2.1 run_with_field()

```
void run_with_field (
            int q,
            size_t bits,
            size_t n,
            size_t d,
            size_t iter,
```

```
            std::string file,
            int variant )
```

17.7.2.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.8 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)

17.8.1 Macro Definition Documentation

17.8.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.8.1.2 _NR_TESTS

```
#define _NR_TESTS 5
```

17.8.1.3 _MAX_SIZE_MATRICES

```
#define _MAX_SIZE_MATRICES 1000
```

17.8.1.4 CUBE

```
#define CUBE(
            x ) ((x)*(x)*(x))
```

17.8.2 Function Documentation

17.8.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.9 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)

17.9.1 Macro Definition Documentation

17.9.1.1 CBLAS_GEMM

```
#define CBLAS_GEMM cblas_dgemm
```

17.9.2 Typedef Documentation

17.9.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

17.9.2.2 Floats

```
typedef double Floats
```

17.9.3 Function Documentation

17.9.3.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.10 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)

17.10.1 Macro Definition Documentation

17.10.1.1 __FFLASFFPACK_HAVE_DGETRF

```
#define __FFLASFFPACK_HAVE_DGETRF 1
```

17.10.2 Typedef Documentation

17.10.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

17.10.3 Function Documentation

17.10.3.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.11 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)

17.11.1 Typedef Documentation

17.11.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

17.11.2 Function Documentation

17.11.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.12 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)

17.12.1 Macro Definition Documentation

17.12.1.1 EFFGFF

```
#define EFFGFF(
              n,
              t,
              i ) ( (double(n)/1000.*double(n)/1000.*double(n)/1000.0) / double(t) * double(i)
/ 3.)
```

17.12.2 Typedef Documentation

17.12.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

17.12.3 Function Documentation

17.12.3.1 main()

```
int main (
             int argc,
             char ** argv )
```

17.13 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)

17.13.1 Typedef Documentation

17.13.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

17.13.2 Function Documentation

17.13.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.14 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)

17.14.1 Macro Definition Documentation

17.14.1.1 __FFLASFFPACK_HAVE_DTRTRI

```
#define __FFLASFFPACK_HAVE_DTRTRI 1
```

17.14.2 Typedef Documentation

17.14.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

17.14.3 Function Documentation

17.14.3.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.15 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)

17.15.1 Macro Definition Documentation

17.15.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.15.2 Function Documentation

17.15.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.16 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)

17.16.1 Macro Definition Documentation

17.16.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.16.2 Function Documentation

17.16.2.1 run_with_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 )
```

17.16.2.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.17 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)

17.17.1 Macro Definition Documentation

17.17.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.17.1.2 MG_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

17.17.1.3 STD_RECINT_SIZE

```
#define STD_RECINT_SIZE 8
```

17.17.2 Function Documentation

17.17.2.1 tmain()

```
int tmain ( )
```

17.17.2.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.18 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[ ])

17.18.1 Macro Definition Documentation

17.18.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.18.2 Typedef Documentation

17.18.2.1 RNS

```
typedef FFPACK::rns_double RNS
```

17.18.2.2 Field

```
typedef FFPACK::RNSInteger<RNS> Field
```

17.18.2.3 Element_ptr

```
typedef Field::Element_ptr Element_ptr
```

17.18.2.4 ConstElement_ptr

```
typedef Field::ConstElement_ptr ConstElement_ptr
```

17.18.2.5 THREADS

```
typedef StrategyParameter::Threads THREADS
```

17.18.2.6 GRAIN

```
typedef StrategyParameter::Grain GRAIN
```

17.18.2.7 TWOD

```
typedef StrategyParameter::TwoD TWOD
```

17.18.2.8 TWODA

```
typedef StrategyParameter::TwoDAdaptive TWODA
```

17.18.2.9 THREED

```
typedef StrategyParameter::ThreeD THREED
```

17.18.2.10 THREEDA

```
typedef StrategyParameter::ThreeDAdaptive THREEDA
```

17.18.2.11 THREEDIP

```
typedef StrategyParameter::ThreeDInPlace THREEDIP
```

17.18.2.12 PSeq

```
typedef ParSeqHelper::Sequential PSeq
```

17.18.3 Function Documentation

17.18.3.1 main()

```
int main (
            int argc,
            char * argv[ ] )
```

17.19 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)

17.19.1 Macro Definition Documentation

17.19.1.1 CLASSIC_HYBRID

```
#define CLASSIC_HYBRID
```

17.19.2 Function Documentation

17.19.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.20 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)

17.20.1 Macro Definition Documentation

17.20.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.20.1.2 MG_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

17.20.1.3 STD_RECINT_SIZE

```
#define STD_RECINT_SIZE 8
```

17.20.2 Function Documentation

17.20.2.1 write_matrix()

```
std::ostream & write_matrix (
            std::ostream & out,
            Givaro::Integer p,
            size_t m,
            size_t n,
            T * C,
            size_t ldc )
```

17.20.2.2 tmain()

```
int tmain ( )
```

17.20.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.21 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 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)
- 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)

17.21.1 Macro Definition Documentation

17.21.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.21.2 Function Documentation

17.21.2.1 fill_value()

```
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 )
```

17.21.2.2 genData()

```
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 )
```

17.21.2.3 check_result()

```
bool check_result (
            Field & F,
            size_t m,
            size_t lda,
            Matrix & A,
            Vector & X,
            size_t incX,
            Vector & Y,
            size_t incY )
```

17.21.2.4 benchmark_with_timer()

```
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 )
```

17.21.2.5 benchmark_disp()

```
void benchmark_disp (
            Field & F,
            bool pass,
            double & time,
            size_t iters,
            int p,
            size_t m,
```

```
            size_t k,
            arg & as )
```

17.21.2.6 benchmark_in_Field()

```
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 )
```

17.21.2.7 benchmark_with_field() [1/2]

```
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 )
```

17.21.2.8 benchmark_with_field() [2/2]

```
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 )
```

17.21.2.9 main()

```
int main (
            int argc,
            char ** argv )
```

17.22 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)

17.22.1 Macro Definition Documentation

17.22.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.22.2 Function Documentation

17.22.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.23 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)

17.23.1 Macro Definition Documentation

17.23.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.23.1.2 CUBE

```
#define CUBE(
              x ) ((x)*(x)*(x))
```

17.23.2 Function Documentation

17.23.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.24 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)

17.24.1 Macro Definition Documentation

17.24.1.1 __FFPACK_FSYTRF_BC_CROUT

```
#define __FFPACK_FSYTRF_BC_CROUT
```

17.24.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.24.1.3 CUBE

```
#define CUBE(
              x ) ((x)*(x)*(x))
```

17.24.2 Function Documentation

17.24.2.1 main()

```
int main (
             int argc,
             char ** argv )
```

17.25 benchmark-ftrsm-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)

17.25.1 Macro Definition Documentation

17.25.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.25.2 Function Documentation

17.25.2.1 main()

```
int main (
             int argc,
             char ** argv )
```

17.26 benchmark-ftrsm.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)

17.26.1 Macro Definition Documentation

17.26.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.26.2 Function Documentation

17.26.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.27 benchmark-ftrsv.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)

17.27.1 Macro Definition Documentation

17.27.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.27.2 Function Documentation

17.27.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.28 benchmark-ftrtri.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)

17.28.1 Macro Definition Documentation

17.28.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.28.1.2 CUBE

```
#define CUBE(
             x ) ((x)*(x)*(x))
```

17.28.2 Function Documentation

17.28.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.29 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)

17.29.1 Macro Definition Documentation

17.29.1.1 CUBE

```
#define CUBE(
             x ) ((x)*(x)*(x))
```

17.29.2 Function Documentation

17.29.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.30 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)

17.30.1 Function Documentation

17.30.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.31 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)

17.31.1 Macro Definition Documentation

17.31.1.1 CUBE

```
#define CUBE(
             x ) ((x)*(x)*(x))
```

17.31.2 Function Documentation

17.31.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.32 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)

17.32.1 Macro Definition Documentation

17.32.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

17.32.1.2 CUBE

```
#define CUBE(
            x ) ((x)*(x)*(x))
```

17.32.2 Typedef Documentation

17.32.2.1 Field

```
typedef Givaro::ModularBalanced<double> Field
```

17.32.3 Function Documentation

17.32.3.1 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 )
```

17.32.3.2 Rec_Initialize()

```
void Rec_Initialize (
            Field & F,
            Field::Element * C,
            size_t m,
            size_t n,
            size_t ldc )
```

17.32.3.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.33 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)

17.33.1 Macro Definition Documentation

17.33.1.1 CUBE

```
#define CUBE(
             x ) ((x)*(x)*(x))
```

17.33.2 Function Documentation

17.33.2.1 launch_wino()

```
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 )
```

17.33.2.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.34 mainpage.doxy File Reference

17.35 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 ()

17.35.1 Macro Definition Documentation

17.35.1.1 CUBE

```
#define CUBE(
             x ) ((x)*(x)*(x))
```

17.35.1.2 GFOPS

```
#define GFOPS(
             m,
             n,
             r,
             t ) (2.7*CUBE(double(n)/1000.0))/t
```

17.35.2 Typedef Documentation

17.35.2.1 TTimer

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

17.35.3 Function Documentation

17.35.3.1 main()

```
int main (
            void )
```

17.36 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.

17.36.1 Function Documentation

17.36.1.1 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.

17.37 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.

17.37.1 Function Documentation

17.37.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

This example computes the determinant of a matrix over a defined finite field.
Outputs the determinant.

17.38 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.

17.38.1 Function Documentation

17.38.1.1 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.

17.39 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 ()

17.39.1 Macro Definition Documentation

17.39.1.1 CUBE

```
#define CUBE(
              x ) ((x)*(x)*(x))
```

17.39.1.2 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*double(r)/1000.0*(m+n)/1000)/t
```

17.39.2 Typedef Documentation

17.39.2.1 TTimer

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

17.39.3 Function Documentation

17.39.3.1 main()

```
int main (
             void )
```

17.40 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)

17.40.1 Function Documentation

17.40.1.1 main()

```
int main (
             int argc,
             char ** argv )
```

17.41 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.

17.41.1 Function Documentation

17.41.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

This example computes the rank of a matrix over a defined finite field.
Outputs the rank.

17.42 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.

17.42.1 Function Documentation

17.42.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

This example solve the quare system defined by the input over a defined finite field.

17.43 checker_charpoly.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class CheckerImplem_charpoly< Field, Polynomial >

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_checker_charpoly_INL

17.43.1 Macro Definition Documentation

17.43.1.1 __FFLASFFPACK_checker_charpoly_INL

```
#define __FFLASFFPACK_checker_charpoly_INL
```

17.44 checker_det.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class CheckerImplem_Det< Field >

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_checker_det_INL

17.44.1 Macro Definition Documentation

17.44.1.1 __FFLASFFPACK_checker_det_INL

```
#define __FFLASFFPACK_checker_det_INL
```

17.45 checker_empty.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
```

Data Structures

- struct Checker_Empty< Field >

Namespaces

- namespace FFLAS

17.46 checker_fgemm.inl File Reference

Data Structures

- class CheckerImplem_fgemm< Field >

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_checker_fgemm_INL

17.46.1 Macro Definition Documentation

17.46.1.1 __FFLASFFPACK_checker_fgemm_INL

```
#define __FFLASFFPACK_checker_fgemm_INL
```

17.47 checker_ftrsm.inl File Reference

Data Structures

- class CheckerImplem_ftrsm< Field >

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_checker_ftrsm_INL

17.47.1 Macro Definition Documentation

17.47.1.1 __FFLASFFPACK_checker_ftrsm_INL

```
#define __FFLASFFPACK_checker_ftrsm_INL
```

17.48 checker_invert.inl File Reference

Data Structures

- class CheckerImplem_invert< Field >

Namespaces

- namespace FFPACK

  *Finite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_checker_invert_INL

17.48.1 Macro Definition Documentation

17.48.1.1 __FFLASFFPACK_checker_invert_INL

```
#define __FFLASFFPACK_checker_invert_INL
```

17.49 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

17.49.1 Macro Definition Documentation

17.49.1.1 __FFLASFFPACK_checker_pluq_INL

```
#define __FFLASFFPACK_checker_pluq_INL
```

17.50 checkers.doxy File Reference

17.51 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 >

17.52 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 >

17.52.1 Macro Definition Documentation

17.52.1.1 FFLASFFPACK_checkers_fflas_inl_H

```
#define FFLASFFPACK_checkers_fflas_inl_H
```

17.53 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 >

17.54 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 >

17.54.1 Macro Definition Documentation

17.54.1.1 FFLASFFPACK_checkers_ffpack_inl_H

```
#define FFLASFFPACK_checkers_ffpack_inl_H
```

17.55 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 ∗)

17.55.1 Macro Definition Documentation

17.55.1.1 CBLAS_INT

```
#define CBLAS_INT int
```

17.55.1.2 CBLAS_ENUM_DEFINED_H

```
#define CBLAS_ENUM_DEFINED_H
```

17.55.1.3 CBLAS_EXTERNALS

```
#define CBLAS_EXTERNALS
```

17.55.1.4 blas_enum

```
#define blas_enum enum
```

17.55.2 Enumeration Type Documentation

17.55.2.1 CBLAS_ORDER

```
enum CBLAS_ORDER
```

Enumerator

CblasRowMajor	
CblasColMajor	

17.55.2.2 CBLAS_TRANSPOSE

```
enum CBLAS_TRANSPOSE
```

Enumerator

CblasNoTrans	
CblasTrans	
CblasConjTrans	
AtlasConj	

17.55.2.3 CBLAS_UPLO

```
enum CBLAS_UPLO
```

Enumerator

CblasUpper	
CblasLower	

17.55.2.4 CBLAS_DIAG

```
enum CBLAS_DIAG
```

Enumerator

CblasNonUnit	
CblasUnit	

17.55.2.5 CBLAS_SIDE

```
enum CBLAS_SIDE
```

Enumerator

CblasLeft	
CblasRight	

17.55.3 Function Documentation

17.55.3.1 daxpy_()

```
void daxpy_ (
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

17.55.3.2 saxpy_()

```
void saxpy_ (
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

17.55.3.3 ddot_()

```
double ddot_ (
            const int * ,
            const double * ,
            const int * ,
            const double * ,
            const int * )
```

17.55.3.4 sdot_()

```
float sdot_ (
            const int * ,
            const float * ,
            const int * ,
            const float * ,
            const int * )
```

17.55.3.5 dasum_()

```
double dasum_ (
            const int * ,
```

```
const double * ,
const int * )
```

17.55.3.6 idamax_()

```
int idamax_ (
            const int * ,
            const double * ,
            const int * )
```

17.55.3.7 dnrm2_()

```
double dnrm2_ (
            const int * ,
            const double * ,
            const int * )
```

17.55.3.8 dgemv_()

```
void dgemv_ (
            const char * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            const double * ,
            const int * ,
            const double * ,
            double * ,
            const int * )
```

17.55.3.9 sgemv_()

```
void sgemv_ (
            const char * ,
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            const float * ,
            const int * ,
            const float * ,
            float * ,
            const int * )
```

17.55.3.10 dger_()

```
void dger_ (
            const int * ,
            const int * ,
            const double * ,
            const double * ,
```

```
            const int * ,
            const double * ,
            const int * ,
            double * ,
            const int *  )
```

17.55.3.11 sger_()

```
void sger_ (
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            const float * ,
            const int * ,
            float * ,
            const int *  )
```

17.55.3.12 dcopy_()

```
void dcopy_ (
            const int * ,
            const double * ,
            const int * ,
            double * ,
            const int *  )
```

17.55.3.13 scopy_()

```
void scopy_ (
            const int * ,
            const float * ,
            const int * ,
            float * ,
            const int *  )
```

17.55.3.14 dscal_()

```
void dscal_ (
            const int * ,
            const double * ,
            double * ,
            const int *  )
```

17.55.3.15 sscal_()

```
void sscal_ (
            const int * ,
            const float * ,
            float * ,
            const int *  )
```

17.55.3.16 dtrsm_()

```
void dtrsm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

17.55.3.17 strsm_()

```
void strsm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

17.55.3.18 dtrmm_()

```
void dtrmm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

17.55.3.19 strmm_()

```
void strmm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const float * ,
            const float * ,
```

```
            const int * ,
            float * ,
            const int *  )
```

17.55.3.20 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 *  )
```

17.55.3.21 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 *  )
```

17.56 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"

17.56.1 Macro Definition Documentation

17.56.1.1 HAVE_BLAS

```
#define HAVE_BLAS 1
```

17.56.1.2 HAVE_CBLAS

```
#define HAVE_CBLAS 1
```

17.56.1.3 HAVE_CXX11

```
#define HAVE_CXX11 1
```

17.56.1.4 HAVE_DLFCN_H

```
#define HAVE_DLFCN_H 1
```

17.56.1.5 HAVE_FLOAT_H

```
#define HAVE_FLOAT_H 1
```

17.56.1.6 HAVE_INTTYPES_H

```
#define HAVE_INTTYPES_H 1
```

17.56.1.7 HAVE_LAPACK

```
#define HAVE_LAPACK 1
```

17.56.1.8 HAVE_LIMITS_H

```
#define HAVE_LIMITS_H 1
```

17.56.1.9 HAVE_LITTLE_ENDIAN

```
#define HAVE_LITTLE_ENDIAN 1
```

17.56.1.10 HAVE_PTHREAD_H

```
#define HAVE_PTHREAD_H 1
```

17.56.1.11 HAVE_STDDEF_H

```
#define HAVE_STDDEF_H 1
```

17.56.1.12 HAVE_STDINT_H

```
#define HAVE_STDINT_H 1
```

17.56.1.13 HAVE_STDIO_H

```
#define HAVE_STDIO_H 1
```

17.56.1.14 HAVE_STDLIB_H

```
#define HAVE_STDLIB_H 1
```

17.56.1.15 HAVE_STRINGS_H

```
#define HAVE_STRINGS_H 1
```

17.56.1.16 HAVE_STRING_H

```
#define HAVE_STRING_H 1
```

17.56.1.17 HAVE_SYS_STAT_H

```
#define HAVE_SYS_STAT_H 1
```

17.56.1.18 HAVE_SYS_TIME_H

```
#define HAVE_SYS_TIME_H 1
```

17.56.1.19 HAVE_SYS_TYPES_H

```
#define HAVE_SYS_TYPES_H 1
```

17.56.1.20 HAVE_UNISTD_H

```
#define HAVE_UNISTD_H 1
```

17.56.1.21 LT_OBJDIR

```
#define LT_OBJDIR ".libs/"
```

17.56.1.22 OPENBLAS_NUM_THREADS

```
#define OPENBLAS_NUM_THREADS 1
```

17.56.1.23 PACKAGE

```
#define PACKAGE "fflas-ffpack"
```

17.56.1.24 PACKAGE_BUGREPORT

```
#define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

17.56.1.25 PACKAGE_NAME

```
#define PACKAGE_NAME "FFLAS-FFPACK"
```

17.56.1.26 PACKAGE_STRING

```
#define PACKAGE_STRING "FFLAS-FFPACK 2.4.3"
```

17.56.1.27 PACKAGE_TARNAME

```
#define PACKAGE_TARNAME "fflas-ffpack"
```

17.56.1.28 PACKAGE_URL

```
#define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

17.56.1.29 PACKAGE_VERSION

```
#define PACKAGE_VERSION "2.4.3"
```

17.56.1.30 SIZEOF_CHAR

```
#define SIZEOF_CHAR 1
```

17.56.1.31 SIZEOF_INT

```
#define SIZEOF_INT 4
```

17.56.1.32 SIZEOF_LONG

```
#define SIZEOF_LONG 4
```

17.56.1.33 SIZEOF_LONG_LONG

```
#define SIZEOF_LONG_LONG 8
```

17.56.1.34 SIZEOF_SHORT

```
#define SIZEOF_SHORT 2
```

17.56.1.35 SIZEOF___INT64

```
#define SIZEOF___INT64 0
```

17.56.1.36 STDC_HEADERS

```
#define STDC_HEADERS 1
```

17.56.1.37 USE_OPENMP

```
#define USE_OPENMP 1
```

17.56.1.38 VERSION

```
#define VERSION "2.4.3"
```

17.57 config.h File Reference

Macros

- #define __FFLASFFPACK_HAVE_BLAS 1
- #define __FFLASFFPACK_HAVE_CBLAS 1
- #define __FFLASFFPACK_HAVE_CXX11 1
- #define __FFLASFFPACK_HAVE_DLFCN_H 1
- #define __FFLASFFPACK_HAVE_FLOAT_H 1
- #define __FFLASFFPACK_HAVE_INTTYPES_H 1
- #define __FFLASFFPACK_HAVE_LAPACK 1
- #define __FFLASFFPACK_HAVE_LIMITS_H 1
- #define __FFLASFFPACK_HAVE_LITTLE_ENDIAN 1
- #define __FFLASFFPACK_HAVE_PTHREAD_H 1
- #define __FFLASFFPACK_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"

17.57.1 Macro Definition Documentation

17.57.1.1 __FFLASFFPACK_HAVE_BLAS

```
#define __FFLASFFPACK_HAVE_BLAS 1
```

17.57.1.2 __FFLASFFPACK_HAVE_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

17.57.1.3 __FFLASFFPACK_HAVE_CXX11

```
#define __FFLASFFPACK_HAVE_CXX11 1
```

17.57.1.4 __FFLASFFPACK_HAVE_DLFCN_H

```
#define __FFLASFFPACK_HAVE_DLFCN_H 1
```

17.57.1.5 __FFLASFFPACK_HAVE_FLOAT_H

```
#define __FFLASFFPACK_HAVE_FLOAT_H 1
```

17.57.1.6 __FFLASFFPACK_HAVE_INTTYPES_H

```
#define __FFLASFFPACK_HAVE_INTTYPES_H 1
```

17.57.1.7 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

17.57.1.8 __FFLASFFPACK_HAVE_LIMITS_H

```
#define __FFLASFFPACK_HAVE_LIMITS_H 1
```

17.57.1.9 __FFLASFFPACK_HAVE_LITTLE_ENDIAN

```
#define __FFLASFFPACK_HAVE_LITTLE_ENDIAN 1
```

17.57.1.10 __FFLASFFPACK_HAVE_PTHREAD_H

```
#define __FFLASFFPACK_HAVE_PTHREAD_H 1
```

17.57.1.11 __FFLASFFPACK_HAVE_STDDEF_H

```
#define __FFLASFFPACK_HAVE_STDDEF_H 1
```

17.57.1.12 __FFLASFFPACK_HAVE_STDINT_H

```
#define __FFLASFFPACK_HAVE_STDINT_H 1
```

17.57.1.13 __FFLASFFPACK_HAVE_STDIO_H

```
#define __FFLASFFPACK_HAVE_STDIO_H 1
```

17.57.1.14 __FFLASFFPACK_HAVE_STDLIB_H

```
#define __FFLASFFPACK_HAVE_STDLIB_H 1
```

17.57.1.15 __FFLASFFPACK_HAVE_STRINGS_H

```
#define __FFLASFFPACK_HAVE_STRINGS_H 1
```

17.57.1.16 __FFLASFFPACK_HAVE_STRING_H

```
#define __FFLASFFPACK_HAVE_STRING_H 1
```

17.57.1.17 __FFLASFFPACK_HAVE_SYS_STAT_H

```
#define __FFLASFFPACK_HAVE_SYS_STAT_H 1
```

17.57.1.18 __FFLASFFPACK_HAVE_SYS_TIME_H

```
#define __FFLASFFPACK_HAVE_SYS_TIME_H 1
```

17.57.1.19 __FFLASFFPACK_HAVE_SYS_TYPES_H

```
#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1
```

17.57.1.20 __FFLASFFPACK_HAVE_UNISTD_H

```
#define __FFLASFFPACK_HAVE_UNISTD_H 1
```

17.57.1.21 __FFLASFFPACK_LT_OBJDIR

```
#define __FFLASFFPACK_LT_OBJDIR ".libs/"
```

17.57.1.22 __FFLASFFPACK_OPENBLAS_NUM_THREADS

```
#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1
```

17.57.1.23 __FFLASFFPACK_PACKAGE

```
#define __FFLASFFPACK_PACKAGE "fflas-ffpack"
```

17.57.1.24 __FFLASFFPACK_PACKAGE_BUGREPORT

```
#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

17.57.1.25 __FFLASFFPACK_PACKAGE_NAME

```
#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"
```

17.57.1.26 __FFLASFFPACK_PACKAGE_STRING

```
#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.4.3"
```

17.57.1.27 __FFLASFFPACK_PACKAGE_TARNAME

```
#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
```

17.57.1.28 __FFLASFFPACK_PACKAGE_URL

```
#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

17.57.1.29 __FFLASFFPACK_PACKAGE_VERSION

```
#define __FFLASFFPACK_PACKAGE_VERSION "2.4.3"
```

17.57.1.30 __FFLASFFPACK_SIZEOF_CHAR

```
#define __FFLASFFPACK_SIZEOF_CHAR 1
```

17.57.1.31 __FFLASFFPACK_SIZEOF_INT

```
#define __FFLASFFPACK_SIZEOF_INT 4
```

17.57.1.32 __FFLASFFPACK_SIZEOF_LONG

```
#define __FFLASFFPACK_SIZEOF_LONG 4
```

17.57.1.33 __FFLASFFPACK_SIZEOF_LONG_LONG

```
#define __FFLASFFPACK_SIZEOF_LONG_LONG 8
```

17.57.1.34 __FFLASFFPACK_SIZEOF_SHORT

```
#define __FFLASFFPACK_SIZEOF_SHORT 2
```

17.57.1.35 __FFLASFFPACK_SIZEOF___INT64

```
#define __FFLASFFPACK_SIZEOF___INT64 0
```

17.57.1.36 __FFLASFFPACK_STDC_HEADERS

```
#define __FFLASFFPACK_STDC_HEADERS 1
```

17.57.1.37 __FFLASFFPACK_USE_OPENMP

```
#define __FFLASFFPACK_USE_OPENMP 1
```

17.57.1.38 __FFLASFFPACK_VERSION

```
#define __FFLASFFPACK_VERSION "2.4.3"
```

17.58 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__)

17.58.1 Detailed Description

Defaults for optimised values.
While `fflas-ffpack-optimise.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-optimise.h` is not empty, then its values preceeds the defaults here.

17.58.2 Macro Definition Documentation

17.58.2.1 GCC_VERSION

```
#define GCC_VERSION (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)
```

17.59 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

17.59.1 Macro Definition Documentation

17.59.1.1 __FFLASFFPACK_WINOTHRESHOLD

```
#define __FFLASFFPACK_WINOTHRESHOLD 1000
```

17.59.1.2 __FFLASFFPACK_WINOTHRESHOLD_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_FLT 2000
```

17.59.1.3 __FFLASFFPACK_WINOTHRESHOLD_BAL

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL 1000
```

17.59.1.4 __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT 2000
```

17.59.1.5 __FFLASFFPACK_PLUQ_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 256
```

17.59.1.6 __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD

```
#define __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD 1000
```

17.59.1.7 __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD

```
#define __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD 16
```

17.59.1.8 __FFLASFFPACK_ARITHPROG_THRESHOLD

```
#define __FFLASFFPACK_ARITHPROG_THRESHOLD 30
```

17.59.1.9 __FFLASFFPACK_FTRTRI_THRESHOLD

```
#define __FFLASFFPACK_FTRTRI_THRESHOLD 32
```

17.59.1.10 __FFLASFFPACK_FSYTRF_THRESHOLD

```
#define __FFLASFFPACK_FSYTRF_THRESHOLD 64
```

17.59.1.11 __FFLASFFPACK_FSYRK_THRESHOLD

```
#define __FFLASFFPACK_FSYRK_THRESHOLD 3000
```

17.60 fflas-ffpack-thresholds.h File Reference

17.61 fflas-ffpack.doxy File Reference

17.62 fflas-ffpack.h File Reference

Includes FFLAS and FFPACK.

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas/fflas.h"
#include "ffpack/ffpack.h"
```

17.62.1 Detailed Description

Includes FFLAS and FFPACK.

17.63 fflas.doxy File Reference

17.64 fflas.h File Reference

Finite **F**ield **L**inear **A**lgebra **S**ubroutines

```
#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.

17.64.1 Detailed Description

Finite **F**ield **L**inear **A**lgebra **S**ubroutines

Author

Clément Pernet.

17.64.2 Macro Definition Documentation

17.64.2.1 WINOTHRESHOLD

```
#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD
```

17.64.2.2 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.

17.65 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.

17.65.1 Macro Definition Documentation

17.65.1.1 __FFLASFFPACK_fflas_bounds_INL

```
#define __FFLASFFPACK_fflas_bounds_INL
```

17.65.1.2 FFLAS_INT_TYPE

```
#define FFLAS_INT_TYPE uint64_t
```

17.66 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)

17.67 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.

17.68 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, Field↩
Categories::UnparametricTag)

17.68.1 Macro Definition Documentation

17.68.1.1 __FFLASFFPACK_fadd_INL

```
#define __FFLASFFPACK_fadd_INL
```

17.69 fflas_fassign.h File Reference

```
#include "fflas_fassign.inl"
```

17.70 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 : $A \leftarrow B$.

17.70.1 Macro Definition Documentation

17.70.1.1 __FFLASFFPACK_fassign_INL

```
#define __FFLASFFPACK_fassign_INL
```

17.71 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 : $y \leftarrow \alpha \cdot x + y$.

17.71.1 Macro Definition Documentation

17.71.1.1 __FFLASFFPACK_faxpy_INL

```
#define __FFLASFFPACK_faxpy_INL
```

17.72 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::ConstElem 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 $x^T y$.

17.72.1 Macro Definition Documentation

17.72.1.1 __FFLASFFPACK_fdot_INL

```
#define __FFLASFFPACK_fdot_INL
```

17.73 fflas_fgemm.inl File Reference

```
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/debug.h"
#include "fflas_fgemm/fgemm_classical.inl"
#include "fflas_fgemm/fgemm_winograd.inl"
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fgemm_INL

Functions

- 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 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)
- 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, 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 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: **F**ield **GE**neral **M**atrix **M**ultiply.*
- 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)

17.73.1 Macro Definition Documentation

17.73.1.1 __FFLASFFPACK_fgemm_INL

```
#define __FFLASFFPACK_fgemm_INL
```

17.74 fgemm_classical.inl File Reference

```
#include <cmath>
#include "fflas-ffpack/field/field-traits.h"
```

Macros

- #define __FFLASFFPACK_fflas_fflas_fgemm_classical_INL

17.74.1 Macro Definition Documentation

17.74.1.1 __FFLASFFPACK_fflas_fflas_fgemm_classical_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_classical_INL
```

17.75 fgemm_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_fgemm_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, Mode↵ Categories::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 >, MMHelper↵ Algo::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 >, MMHelper↵ Algo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- 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)

17.75.1 Detailed Description

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

17.75.2 Macro Definition Documentation

17.75.2.1 __FFPACK_fgemm_classical_INL

```
#define __FFPACK_fgemm_classical_INL
```

17.76 fgemm_winograd.inl File Reference

```
#include <stdint.h>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fgemm_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_fgemm_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)

17.76.1 Macro Definition Documentation

17.76.1.1 __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
```

17.76.1.2 NEWWINO

```
#define NEWWINO
```

17.77 matmul.doxy File Reference

17.78 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)

17.78.1 Detailed Description

Bini implementation.

17.78.2 Macro Definition Documentation

17.78.2.1 __FFLASFFPACK_fgemm_bini_INL

```
#define __FFLASFFPACK_fgemm_bini_INL
```

17.79 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, FieldTrait > &WH)

17.79.1 Macro Definition Documentation

17.79.1.1 __FFLASFFPACK_fgemm_winograd_INL

```
#define __FFLASFFPACK_fgemm_winograd_INL
```

17.80 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)

17.80.1 Macro Definition Documentation

17.80.1.1 __FFLASFFPACK_fgemm_winograd_acc_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_INL
```

17.81 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, Field↩ Trait > &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)

17.81.1 Macro Definition Documentation

17.81.1.1 __FFLASFFPACK_fgemm_winograd_acc_ip_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL
```

17.82 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)

17.82.1 Macro Definition Documentation

17.82.1.1 __FFLASFFPACK_fgemm_winograd_ip_INL

```
#define __FFLASFFPACK_fgemm_winograd_ip_INL
```

17.83 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, 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::Float↩ Domain::Element beta, Givaro::FloatDomain::Element_ptr Y, const size_t incY, MMHelper< Givaro::Float↩ Domain, 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)

17.83.1 Macro Definition Documentation

17.83.1.1 __FFLASFFPACK_fgemv_INL

```
#define __FFLASFFPACK_fgemv_INL
```

17.84 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, Mode↩ Categories::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 >
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)

17.84.1 Macro Definition Documentation

17.84.1.1 __FFLASFFPACK_fgemv_mp_INL

```
#define __FFLASFFPACK_fgemv_mp_INL
```

17.85 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::Double↩ Domain::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, Mode↩ Categories::DefaultBoundedTag > &H)
- void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::Float↩ Domain::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, 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)
- 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)

17.85.1 Macro Definition Documentation

17.85.1.1 __FFLASFFPACK_fger_INL

```
#define __FFLASFFPACK_fger_INL
```

17.86 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 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)

17.86.1 Macro Definition Documentation

17.86.1.1 __FFPACK_fger_mp_INL

```
#define __FFPACK_fger_mp_INL
```

17.87 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)

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

  freduce $x \leftarrow xmodF$.

- 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 Initializes X *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 $A \leftarrow AmodF$.
- 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)

17.88 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 BENCMARK 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::MachineFloat↩Tag > &H)
- template< class Field >
  Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::ArbitraryPrec↩IntTag > &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)

17.88.1 Macro Definition Documentation

17.88.1.1 __FFLASFFPACK_fflas_freduce_INL

```
#define __FFLASFFPACK_fflas_freduce_INL
```

17.88.1.2 FFLASFFPACK_COPY_REDUCE

```
#define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCMARK LATER */
```

17.89 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)

17.89.1 Macro Definition Documentation

17.89.1.1 __FFLASFFPACK_fflas_freduce_mp_INL

```
#define __FFLASFFPACK_fflas_freduce_mp_INL
```

17.90 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.*

17.90.1 Macro Definition Documentation

17.90.1.1 __FFLASFFPACK_freivalds_INL

```
#define __FFLASFFPACK_freivalds_INL
```

17.91 fflas_fscal.h File Reference

```
#include "fflas_fscal.inl"
```

17.92 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 fscalin (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 fscalin (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 fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.

- 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 fscalin (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↩ Element a, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void fscalin (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 fscalin (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)

  fscalin $A \leftarrow a \cdot A$.
- 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 $B \leftarrow a \cdot A$.

17.92.1 Macro Definition Documentation

17.92.1.1 __FFLASFFPACK_fscal_INL

```
#define __FFLASFFPACK_fscal_INL
```

17.93 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 fscalin (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 fscalin (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 fscalin (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 fscalin (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)

17.93.1 Macro Definition Documentation

17.93.1.1 __FFLASFFPACK_fscal_mp_INL

```
#define __FFLASFFPACK_fscal_mp_INL
```

17.94 fflas_fsyr2k.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_fsyr2k_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

17.94.1 Macro Definition Documentation

17.94.1.1 __FFLASFFPACK_fflas_fsyr2k_INL

```
#define __FFLASFFPACK_fflas_fsyr2k_INL
```

17.95 fflas_fsyrk.inl File Reference

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_fsyrk_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

17.95.1 Macro Definition Documentation

17.95.1.1 __FFLASFFPACK_fflas_fsyrk_INL

```
#define __FFLASFFPACK_fflas_fsyrk_INL
```

17.96 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**iangular **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**iangular **M**atrix **M**ultiply with 3 operands Computes* $C \leftarrow \alpha \mathrm{op}(A)B + betaC$ *or* $C \leftarrow \alpha B\mathrm{op}(A) + betaC$.

17.96.1 Macro Definition Documentation

17.96.1.1 __FFLASFFPACK_ftrmm_INL

```
#define __FFLASFFPACK_ftrmm_INL
```

17.97 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)

17.97.1 Macro Definition Documentation

17.97.1.1 __FFLASFFPACK_ftrsm_INL

```
#define __FFLASFFPACK_ftrsm_INL
```

17.98 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_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)

17.98.1 Detailed Description

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

17.98.2 Macro Definition Documentation

17.98.2.1 __FFPACK_ftrsm_mp_INL

```
#define __FFPACK_ftrsm_mp_INL
```

17.99 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 $X \leftarrow \mathrm{op}(A^{-1})X$

17.99.1 Macro Definition Documentation

17.99.1.1 __FFLASFFPACK_ftrsv_INL

```
#define __FFLASFFPACK_ftrsv_INL
```

17.100 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< RecIterTrait, 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 RecInt::rint< 6 > &k)
- template<> size_t min_types (const RecInt::rint< 7 > &k)
- template<> size_t min_types (const RecInt::rint< 8 > &k)
- template<> size_t min_types (const RecInt::rint< 9 > &k)
- template<> size_t min_types (const RecInt::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 (RecInt::rint< K > x)
- template<> bool unfit (RecInt::rint< 6 > x)

17.100.1 Macro Definition Documentation

17.100.1.1 __FFLASFFPACK_fflas_fflas_mmhelper_INL

```
#define __FFLASFFPACK_fflas_fflas_mmhelper_INL
```

17.101 igemm.doxy File Reference

17.102 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)

17.103 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)

17.103.1 Macro Definition Documentation

17.103.1.1 __FFLASFFPACK_fflas_igemm_igemm_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_INL
```

17.104 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 ∗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)

17.105 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)

17.105.1 Macro Definition Documentation

17.105.1.1 __FFLASFFPACK_fflas_igemm_igemm_kernels_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_kernels_INL
```

17.106 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)

17.107 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)

17.107.1 Macro Definition Documentation

17.107.1.1 __FFLASFFPACK_fflas_igemm_igemm_tools_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL
```

17.108 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)

  freduce $x \leftarrow x mod F$.
- 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)

  freduce $x \leftarrow y mod F$.
- 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)

  finit $x \leftarrow y mod F$.
- 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* $F\$$.
- 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 $x \leftarrow y mod F$.
- 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 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<class Field >
void fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

    fscalin $x \leftarrow \alpha \cdot x$.

- 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: 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)

17.108.1 Macro Definition Documentation

17.108.1.1 __FFLASFFPACK_fflas_fflas_level1_INL

```
#define __FFLASFFPACK_fflas_fflas_level1_INL
```

17.109 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 $A \leftarrow AmodF.$
- 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 , 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 , 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 $\mathtt{A}$ *in* $F\$.$
- 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 $A \leftarrow BmodF.$
- 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 fscalin (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)

  fscalin $A \leftarrow a \cdot A.$
- 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 $X \leftarrow \mathrm{op}(A^{-1})X$
- 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 $X \leftarrow \mathrm{op}(A)X$

17.109.1 Macro Definition Documentation

17.109.1.1 __FFLASFFPACK_fflas_fflas_level2_INL

```
#define __FFLASFFPACK_fflas_fflas_level2_INL
```

17.110 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, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)
- template<class Field >
  void MatF2MatFl_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**iangular **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**iangular **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**iangular **M**atrix **M**ultiply with 3 operands Computes* $C \leftarrow \alpha \mathrm{op}(A)B + betaC$ *or* $C \leftarrow \alpha B \mathrm{op}(A) + betaC$.
- 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 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<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 **GE**neral **M**atrix **M**ultiply.*
- 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.

17.110.1 Macro Definition Documentation

17.110.1.1 __FFLASFFPACK_fflas_fflas_level3_INL

```
#define __FFLASFFPACK_fflas_fflas_level3_INL
```

17.110.1.2 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

17.111 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, typenameField::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)

17.111.1 Macro Definition Documentation

17.111.1.1 __FFLASFFPACK_fflas_pfgemm_INL

```
#define __FFLASFFPACK_fflas_pfgemm_INL
```

17.111.1.2 __FFLASFFPACK_SEQPARTHRESHOLD

```
#define __FFLASFFPACK_SEQPARTHRESHOLD 220
```

17.111.1.3 __FFLASFFPACK_DIMKPENALTY

```
#define __FFLASFFPACK_DIMKPENALTY 1
```

17.112 fflas_pftrsm.inl File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_pftrsm_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)

17.112.1 Macro Definition Documentation

17.112.1.1 __FFLASFFPACK_fflas_pftrsm_INL

```
#define __FFLASFFPACK_fflas_pftrsm_INL
```

17.112.1.2 PTRSM_HYBRID_THRESHOLD

```
#define PTRSM_HYBRID_THRESHOLD 256
```

17.113 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, INVP, Q)

Typedefs

- template< class T >
  using Simd = typename SimdChooser< T >::value

17.113.1 Macro Definition Documentation

17.113.1.1 SIMD_INT

```
#define SIMD_INT 1
```

17.113.1.2 INLINE

```
#define INLINE inline
```

17.113.1.3 CONST

```
#define CONST
```

17.113.1.4 PURE

```
#define PURE
```

17.113.1.5 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);
      \
}
```

17.113.1.6 FLOAT_MOD

```
#define FLOAT_MOD(
              C,
              P,
              INVP,
              Q )
```

Value:

```
{
      \
    Q = mul(C, INVP);
      \
    Q = floor(Q);
      \
    C = fnmadd(C, Q, P);
      \
}
```

17.113.2 Typedef Documentation

17.113.2.1 Simd

```
using Simd = typename SimdChooser<T>::value
```

17.114 simd.doxy File Reference

17.115 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)>

17.115.1 Macro Definition Documentation

17.115.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL
```

17.115.2 Typedef Documentation

17.115.2.1 Simd128

```
using Simd128 = Simd128_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↩
::is_signed<T>::value, sizeof(T)>
```

17.116 simd128_double.inl File Reference

Data Structures

- struct Simd128_impl< true, false, true, 8 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL

17.116.1 Macro Definition Documentation

17.116.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL
```

17.117 simd128_float.inl File Reference

Data Structures

- struct Simd128_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL

17.117.1 Macro Definition Documentation

17.117.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL
```

17.118 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

17.118.1 Macro Definition Documentation

17.118.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL
```

17.119 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

17.119.1 Macro Definition Documentation

17.119.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL
```

17.120 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

17.120.1 Macro Definition Documentation

17.120.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
```

17.120.1.2 vect_t

```
#define vect_t Simd128_impl<true,true,true,8>::vect_t
```

17.121 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)>

17.121.1 Macro Definition Documentation

17.121.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL
```

17.121.2 Typedef Documentation

17.121.2.1 Simd256

```
using Simd256 = Simd256_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↩
::is_signed<T>::value, sizeof(T)>
```

17.122 simd256_double.inl File Reference

Data Structures

- struct Simd256_impl< true, false, true, 8 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL

17.122.1 Macro Definition Documentation

17.122.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL
```

17.123 simd256_float.inl File Reference

Data Structures

- struct Simd256_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL

17.123.1 Macro Definition Documentation

17.123.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL
```

17.124 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

17.124.1 Macro Definition Documentation

17.124.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL
```

17.125 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

17.125.1 Macro Definition Documentation

17.125.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL
```

17.126 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

17.126.1 Macro Definition Documentation

17.126.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
```

17.126.1.2 vect_t

```
#define vect_t Simd256_impl<true, true, true, 8>::vect_t
```

17.127 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)>

17.127.1 Macro Definition Documentation

17.127.1.1 __FFLASFFPACK_simd512_INL

```
#define __FFLASFFPACK_simd512_INL
```

17.127.2 Typedef Documentation

17.127.2.1 Simd512

```
using Simd512 = Simd512_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↩
::is_signed<T>::value, sizeof(T)>
```

17.128 simd512_double.inl File Reference

Data Structures

- struct Simd512_impl< true, false, true, 8 >

Macros

- #define __FFLASFFPACK_simd512_double_INL

17.128.1 Macro Definition Documentation

17.128.1.1 __FFLASFFPACK_simd512_double_INL

```
#define __FFLASFFPACK_simd512_double_INL
```

17.129 simd512_float.inl File Reference

Data Structures

- struct Simd512_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_simd512_float_INL

17.129.1 Macro Definition Documentation

17.129.1.1 __FFLASFFPACK_simd512_float_INL

```
#define __FFLASFFPACK_simd512_float_INL
```

17.130 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

17.130.1 Macro Definition Documentation

17.130.1.1 __FFLASFFPACK_simd512_int32_INL

```
#define __FFLASFFPACK_simd512_int32_INL
```

17.131 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

17.131.1 Macro Definition Documentation

17.131.1.1 _simd512_int64_INL

```
#define _simd512_int64_INL
```

17.131.1.2 vect_t

```
#define vect_t Simd512_impl<true, true, true, 8>::vect_t
```

17.132 simd_modular.inl File Reference

Data Structures

- class FieldSimd< _Field >

17.133 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::MachineFloatTag >::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::MachineFloatTag >::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::MachineFloatTag >::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::MachineFloatTag >::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::MachineFloatTag >::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::MachineFloatTag >::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)

17.133.1 Macro Definition Documentation

17.133.1.1 index_t

```
#define index_t uint32_t
```

17.133.1.2 ROUND_DOWN

```
#define ROUND_DOWN(
              x,
              s ) ((x) & ~((s)-1))
```

17.133.1.3 __FFLASFFPACK_CACHE_LINE_SIZE

```
#define __FFLASFFPACK_CACHE_LINE_SIZE 64
```

17.133.1.4 assume_aligned

```
#define assume_aligned(
              pout,
              pin,
              v ) decltype(pin) pout = pin;
```

17.133.1.5 DENSE_THRESHOLD

```
#define DENSE_THRESHOLD 0.5
```

17.134 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::MachineFloatTag >::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::MachineFloatTag >::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 &&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 >::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::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 &&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)
- 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::MachineFloatTag >::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, Element↩ Categories::MachineFloatTag >::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)

17.134.1 Macro Definition Documentation

17.134.1.1 __FFLASFFPACK_fflas_fflas_sparse_INL

```
#define __FFLASFFPACK_fflas_fflas_sparse_INL
```

17.135 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_spmm.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)

17.136 coo_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_coo_spmm_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::Unparametric←Tag)
- 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 block←Size, 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)

17.136.1 Macro Definition Documentation

17.136.1.1 __FFLASFFPACK_fflas_sparse_coo_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmm_INL
```

17.137 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)

17.137.1 Macro Definition Documentation

17.137.1.1 __FFLASFFPACK_fflas_sparse_coo_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

17.138 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)

17.138.1 Macro Definition Documentation

17.138.1.1 __FFLASFFPACK_fflas_sparse_coo_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_utils_INL
```

17.139 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_spmm.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)

17.140 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::Unparametric↩Tag)
- 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)

17.140.1 Macro Definition Documentation

17.140.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL
```

17.141 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)

17.141.1 Macro Definition Documentation

17.141.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL
```

17.142 csr_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_spmm_INL

Functions

- 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, Field↩ Categories::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, Field↩ Categories::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, Field↩ Categories::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)

17.142.1 Macro Definition Documentation

17.142.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmm_INL
```

17.143 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)

17.143.1 Macro Definition Documentation

17.143.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

17.144 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)

17.145 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_spmm.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)

17.146 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 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)

17.146.1 Macro Definition Documentation

17.146.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL
```

17.147 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 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)

17.147.1 Macro Definition Documentation

17.147.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL
```

17.148 csr_hyb_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL

Functions

- 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)

17.148.1 Macro Definition Documentation

17.148.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL
```

17.149 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)

17.149.1 Macro Definition Documentation

17.149.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

17.150 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)

17.150.1 Macro Definition Documentation

17.150.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL
```

17.151 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_spmm.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)

17.152 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 >
  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)
- 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)

17.152.1 Macro Definition Documentation

17.152.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL
```

17.153 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)

17.153.1 Macro Definition Documentation

17.153.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL
```

17.154 ell_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_spmm_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)

17.154.1 Macro Definition Documentation

17.154.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmm_INL
```

17.155 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)

17.155.1 Macro Definition Documentation

17.155.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL
```

17.156 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)

17.156.1 Macro Definition Documentation

17.156.1.1 __FFLASFFPACK_fflas_sparse_ELL_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_utils_INL
```

17.157 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)

17.158 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)

17.158.1 Macro Definition Documentation

17.158.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL
```

17.159 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)

17.159.1 Macro Definition Documentation

17.159.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL
```

17.160 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)

17.160.1 Macro Definition Documentation

17.160.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL
```

17.161 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_spmm.inl"
```

Data Structures

- struct Sparse< _Field, SparseMatrix_t::HYB_ZO >

Namespaces

- namespace FFLAS

17.162 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 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)

17.162.1 Macro Definition Documentation

17.162.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL
```

17.163 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 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)

17.163.1 Macro Definition Documentation

17.163.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL
```

17.164 hyb_zo_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL

Functions

- 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)

17.164.1 Macro Definition Documentation

17.164.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL
```

17.165 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)

17.165.1 Macro Definition Documentation

17.165.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

17.166 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)

17.166.1 Macro Definition Documentation

17.166.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL
```

17.167 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 &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)

17.167.1 Macro Definition Documentation

17.167.1.1 DNS_BIN_VER

```
#define DNS_BIN_VER 0
```

17.167.1.2 mask_t

```
#define mask_t uint64_t
```

17.168 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

17.169 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)

17.169.1 Macro Definition Documentation

17.169.1.1 __FFLASFFPACK_fflas_sparse_sell_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_pspmv_INL
```

17.170 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)

17.170.1 Macro Definition Documentation

17.170.1.1 __FFLASFFPACK_fflas_sparse_sell_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL
```

17.171 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)

17.171.1 Macro Definition Documentation

17.171.1.1 __FFLASFFPACK_fflas_sparse_sell_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_utils_INL
```

17.172 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_copy↩_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

17.173 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)

17.174 ffpack.doxy File Reference

17.175 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 x 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_THRESHOL
- 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_
- 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 &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 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 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 >
  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 Charp↩ Tag=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 > &completed↩ Factors, 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 $(v, Av, ..., A^N v)$.
- 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)
- 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 L X = B or X L = 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∗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∗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=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\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=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\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\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\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\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.

17.175.1 Detailed Description

Set of elimination based routines for dense linear algebra.
Matrices are supposed over finite prime field of characteristic less than 2^26.

17.175.2 Macro Definition Documentation

17.175.2.1 __FFLASFFPACK_FTRSTR_THRESHOLD

```
#define __FFLASFFPACK_FTRSTR_THRESHOLD 64
```

17.175.2.2 __FFLASFFPACK_FTRSSYR2K_THRESHOLD

```
#define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64
```

17.176 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 L X = B or X L = 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 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)

17.176.1 Macro Definition Documentation

17.176.1.1 __FFLASFFPACK_ffpack_INL

```
#define __FFLASFFPACK_ffpack_INL
```

17.177 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)

17.177.1 Macro Definition Documentation

17.177.1.1 __FFLASFFPACK_charpoly_INL

```
#define __FFLASFFPACK_charpoly_INL
```

17.178 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, typename Field::Element_ptr A, const size_t lda)

17.178.1 Macro Definition Documentation

17.178.1.1 __FFLASFFPACK_ffpack_charpoly_danilveski_INL

```
#define __FFLASFFPACK_ffpack_charpoly_danilveski_INL
```

17.179 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, 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)

17.179.1 Macro Definition Documentation

17.179.1.1 __FFLASFFPACK_ffpack_charpoly_kgfast_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfast_INL
```

17.180 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)

17.180.1 Macro Definition Documentation

17.180.1.1 __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL
```

17.181 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)

17.181.1 Macro Definition Documentation

17.181.1.1 __FFLASFFPACK_ffpack_charpoly_kglu_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kglu_INL
```

17.182 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 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)

17.182.1 Macro Definition Documentation

17.182.1.1 __FFPACK_charpoly_mp_INL

```
#define __FFPACK_charpoly_mp_INL
```

17.183 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

  ***F**inite **F**ield **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 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)

17.183.1 Macro Definition Documentation

17.183.1.1 __FFPACK_det_mp_INL

```
#define __FFPACK_det_mp_INL
```

17.184 ffpack_echelonforms.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **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 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.
- 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 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\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 lda, 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\U of rank R obtained by RowEchelon←Form 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 Lu←Tag=FfpackSlabRecursive)

    Extracts a matrix in echelon form from a compact storage A=L\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\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelon←Form 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\U of rank R obtained by RowEchelon←Form or ColumnEchelonForm.

17.184.1 Macro Definition Documentation

17.184.1.1 __FFLASFFPACK_ffpack_echelon_forms_INL

```
#define __FFLASFFPACK_ffpack_echelon_forms_INL
```

17.184.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256
```

17.185 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.

17.185.1 Macro Definition Documentation

17.185.1.1 __FFLASFFPACK_ffpack_fgesv_INL

```
#define __FFLASFFPACK_ffpack_fgesv_INL
```

17.186 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 A X = B or X A = B.

17.186.1 Macro Definition Documentation

17.186.1.1 __FFLASFFPACK_ffpack_fgetrs_INL

```
#define __FFLASFFPACK_ffpack_fgetrs_INL
```

17.187 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 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 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 lda, 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 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)

17.188 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 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 &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)
- 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_THRESHOL
- 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_
- 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)

17.188.1 Macro Definition Documentation

17.188.1.1 __FFLASFFPACK_ffpack_fsytrf_INL

```
#define __FFLASFFPACK_ffpack_fsytrf_INL
```

17.189 ffpack_ftrssyr2k.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **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.

17.189.1 Macro Definition Documentation

17.189.1.1 __FFLASFFPACK_ffpack_ftrssyr2k_INL

```
#define __FFLASFFPACK_ffpack_ftrssyr2k_INL
```

17.190 ffpack_ftrstr.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **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.

17.190.1 Macro Definition Documentation

17.190.1.1 __FFLASFFPACK_ffpack_ftrstr_INL

```
#define __FFLASFFPACK_ffpack_ftrstr_INL
```

17.191 ffpack_ftrtr.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **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 ldx)

17.191.1 Macro Definition Documentation

17.191.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.191.1.2 __FFLASFFPACK_ffpack_ftrtr_INL

```
#define __FFLASFFPACK_ffpack_ftrtr_INL
```

17.192 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.

17.192.1 Macro Definition Documentation

17.192.1.1 __FFLASFFPACK_ffpack_invert_INL

```
#define __FFLASFFPACK_ffpack_invert_INL
```

17.193 ffpack_krylovelim.inl File Reference

Macros

- #define __FFLASFFPACK_ffpack_krylovelim_INL

17.193.1 Macro Definition Documentation

17.193.1.1 __FFLASFFPACK_ffpack_krylovelim_INL

```
#define __FFLASFFPACK_ffpack_krylovelim_INL
```

17.194 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)

17.194.1 Macro Definition Documentation

17.194.1.1 __FFLASFFPACK_ffpack_ludivine_INL

```
#define __FFLASFFPACK_ffpack_ludivine_INL
```

17.195 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)

17.195.1 Macro Definition Documentation

17.195.1.1 __FFPACK_ludivine_mp_INL

```
#define __FFPACK_ludivine_mp_INL
```

17.196 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 $(v, Av, ..., A^N v)$.
- 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)

17.196.1 Macro Definition Documentation

17.196.1.1 __FFLASFFPACK_ffpack_minpoly_INL

```
#define __FFLASFFPACK_ffpack_minpoly_INL
```

17.197 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 x 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 x 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 >
  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 x 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 , 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)

17.197.1 Macro Definition Documentation

17.197.1.1 __FFLASFFPACK_ffpack_permutation_INL

```
#define __FFLASFFPACK_ffpack_permutation_INL
```

17.197.1.2 FFLASFFPACK_PERM_BKSIZE

```
#define FFLASFFPACK_PERM_BKSIZE 32
```

17.198 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, 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)
- 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.

17.198.1 Macro Definition Documentation

17.198.1.1 __FFLASFFPACK_ffpack_pluq_INL

```
#define __FFLASFFPACK_ffpack_pluq_INL
```

17.198.1.2 CROUT

```
#define CROUT
```

17.199 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)

17.199.1 Macro Definition Documentation

17.199.1.1 __FFPACK_pluq_mp_INL

```
#define __FFPACK_pluq_mp_INL
```

17.200 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_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<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)

17.200.1 Macro Definition Documentation

17.200.1.1 __FFLASFFPACK_ffpack_ppluq_INL

```
#define __FFLASFFPACK_ffpack_ppluq_INL
```

17.200.1.2 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

17.200.1.3 PBASECASE_K

```
#define PBASECASE_K 256
```

17.201 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=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 $r*r$ 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 $r*r$ 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.

17.201.1 Macro Definition Documentation

17.201.1.1 __FFLASFFPACK_ffpack_rank_profiles_INL

```
#define __FFLASFFPACK_ffpack_rank_profiles_INL
```

17.202 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::recInt.
- 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< RecInt::rint< K > >
- struct ElementTraits< RecInt::ruint< K > >
- struct ElementTraits< RecInt::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< RecInt::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< RecInt::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 RecInt
- namespace Givaro
- namespace FFPACK

  ***F**inite **F**ield **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

17.202.1 Detailed Description

Field Traits.

17.203 field.doxy File Reference

17.204 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)

17.204.1 Detailed Description

rns elt structure with double support

17.205 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

17.205.1 Macro Definition Documentation

17.205.1.1 __FFLASFFPACK_field_rns_double_recint_INL

```
#define __FFLASFFPACK_field_rns_double_recint_INL
```

17.206 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)

17.206.1 Detailed Description

rns structure with double support

17.206.2 Macro Definition Documentation

17.206.2.1 ROUND_DOWN

```
#define ROUND_DOWN(
              x,
              s ) ((x) & ~((s)-1))
```

17.207 rns-double.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_field_rns_double_INL

17.207.1 Macro Definition Documentation

17.207.1.1 __FFLASFFPACK_field_rns_double_INL

```
#define __FFLASFFPACK_field_rns_double_INL
```

17.208 rns-integer-mod.h File Reference

representation of Z/pZ 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

  ***F**inite **F**ield **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)

17.208.1 Detailed Description

representation of $\mathbb{Z}/p\mathbb{Z}$ using RNS representation (note: fixed precision)

17.209 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

  ***F**inite **F**ield **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)

17.209.1 Detailed Description

representation of $\mathbb{Z}$ using RNS representation (note: fixed precision)

17.210 rns.h File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

17.211 rns.inl File Reference

```
#include "rns-double.h"
#include "rns-integer.h"
#include "rns-integer-mod.h"
```

Macros

- #define __FFLASFFPACK_field_rns_INL

17.211.1 Macro Definition Documentation

17.211.1.1 __FFLASFFPACK_field_rns_INL

```
#define __FFLASFFPACK_field_rns_INL
```

17.212 interfaces.doxy File Reference

17.213 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 , FflasRowMajor =101 , FflasColMajor =102 }

  Storage by row or col ?
- enum FFLAS_C_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 , FflasNoTrans = 111 , FflasTrans = 112 }

  Is matrix transposed ?
- enum FFLAS_C_UPLO { FflasUpper = 121 , FflasLower = 122 , FflasUpper = 121 , FflasLower = 122 }

  Is triangular matrix's shape upper ?
- enum FFLAS_C_DIAG { FflasNonUnit = 131 , FflasUnit = 132 , FflasNonUnit = 131 , FflasUnit = 132 }

  Is the triangular matrix implicitly unit diagonal ?
- enum FFLAS_C_SIDE { FflasLeft = 141 , FflasRight = 142 , 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 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)
- 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)
- 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 ∗B, const size_t ldB, double ∗A, const size_t ldA, 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 ∗B, const size_t ldB, double ∗A, const size_t ldA, 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 ∗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)

17.213.1 Macro Definition Documentation

17.213.1.1 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

17.213.2 Enumeration Type Documentation

17.213.2.1 FFLAS_C_ORDER

`enum FFLAS_C_ORDER`

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major
FflasRowMajor	
FflasColMajor	

17.213.2.2 FFLAS_C_TRANSPOSE

`enum FFLAS_C_TRANSPOSE`

Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.
FflasNoTrans	
FflasTrans	

17.213.2.3 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 $T_{i,j} = 0$)
FflasUpper	
FflasLower	

17.213.2.4 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 ($T_{i,i} = 1$)
FflasNonUnit	
FflasUnit	

17.213.2.5 FFLAS_C_SIDE

enum FFLAS_C_SIDE

On what side ?

Enumerator

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

17.213.2.6 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 precison BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

17.213.3 Function Documentation

17.213.3.1 freducein_1_modular_double()

```
void freducein_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive )
```

17.213.3.2 freduce_1_modular_double()

```
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 )
```

17.213.3.3 fnegin_1_modular_double()

```
void fnegin_1_modular_double (
            const double F,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive )
```

17.213.3.4 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 )
```

17.213.3.5 fzero_1_modular_double()

```
void fzero_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive )
```

17.213.3.6 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 )
```

17.213.3.7 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 )
```

17.213.3.8 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 )
```

17.213.3.9 fscalin_1_modular_double()

```
void fscalin_1_modular_double (
            const double p,
            const size_t n,
            const double alpha,
            double * X,
            const size_t incX,
            bool positive )
```

17.213.3.10 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 )
```

17.213.3.11 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 )
```

17.213.3.12 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 )
```

17.213.3.13 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 )
```

17.213.3.14 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 )
```

17.213.3.15 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 )
```

17.213.3.16 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 )
```

17.213.3.17 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 )
```

17.213.3.18 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 )
```

17.213.3.19 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 )
```

17.213.3.20 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 )
```

17.213.3.21 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 )
```

17.213.3.22 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 )
```

17.213.3.23 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 )
```

17.213.3.24 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 )
```

17.213.3.25 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 )
```

17.213.3.26 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 )
```

17.213.3.27 fscalin_2_modular_double()

```
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 )
```

17.213.3.28 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 )
```

17.213.3.29 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 )
```

17.213.3.30 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 )
```

17.213.3.31 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 )
```

17.213.3.32 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 )
```

17.213.3.33 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 )
```

17.213.3.34 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 )
```

17.213.3.35 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 betA,
            double * Y,
            const size_t incY,
            bool positive )
```

17.213.3.36 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 )
```

17.213.3.37 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 )
```

17.213.3.38 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 )
```

17.213.3.39 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 )
```

17.213.3.40 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 )
```

17.213.3.41 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 )
```

17.214 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

17.214.1 Macro Definition Documentation

17.214.1.1 __FFLAS_L1_INST_C

```
#define __FFLAS_L1_INST_C
```

17.214.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

17.214.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

17.214.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

17.214.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

17.214.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

17.214.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

17.214.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

17.214.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

17.214.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

17.215 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

17.215.1 Macro Definition Documentation

17.215.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

17.215.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

17.215.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

17.215.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

17.215.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

17.215.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

17.215.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

17.215.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

17.215.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

17.216 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)

  freduce $x \leftarrow x mod F$.
- 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)

  freduce $x \leftarrow y mod F$.
- 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)

  finit $x \leftarrow y mod F$.
- 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)

  fconvert $x \leftarrow y mod F$.
- template INST_OR_DECL void fnegin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT ∗X, const size_t incX)

  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)

  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 fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT ∗X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- 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)

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

17.217 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

17.217.1 Macro Definition Documentation

17.217.1.1 __FFLAS_L2_INST_C

```
#define __FFLAS_L2_INST_C
```

17.217.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

17.217.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

17.217.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

17.217.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

17.217.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

17.217.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

17.217.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

17.217.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

17.217.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

17.218 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

17.218.1 Macro Definition Documentation

17.218.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

17.218.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

17.218.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

17.218.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

17.218.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

17.218.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

17.218.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

17.218.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

17.218.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

17.219 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 $A \leftarrow AmodF$.
- 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 $A \leftarrow BmodF$.
- 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 $A \leftarrow BmodF$.
- 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 fscalin (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)

  fscalin $A \leftarrow a \cdot A$.
- 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 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 $X \leftarrow \mathrm{op}(A^{-1})X$

17.220 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

17.220.1 Macro Definition Documentation

17.220.1.1 __FFLAS_L3_INST_C

```
#define __FFLAS_L3_INST_C
```

17.220.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

17.220.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

17.220.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

17.220.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

17.220.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

17.220.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

17.220.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

17.220.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

17.220.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

17.221 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

17.221.1 Macro Definition Documentation

17.221.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

17.221.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

17.221.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

17.221.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

17.221.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

17.221.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

17.221.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

17.221.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

17.221.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

17.222 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: **TR**iangular **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: **TR**iangular **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: **F**ield **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.

17.222.1 Macro Definition Documentation

17.222.1.1 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

17.223 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 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 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)

17.223.1 Detailed Description

C functions calls for level 1 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level1.inl

17.223.2 Function Documentation

17.223.2.1 freducein_1_modular_double()

```
void freducein_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive )
```

17.223.2.2 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 )
```

17.223.2.3 fnegin_1_modular_double()

```
void fnegin_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive )
```

17.223.2.4 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 )
```

17.223.2.5 fzero_1_modular_double()

```
void fzero_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive )
```

17.223.2.6 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 )
```

17.223.2.7 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 )
```

17.223.2.8 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 )
```

17.223.2.9 fscalin_1_modular_double()

```
void fscalin_1_modular_double (
            const double p,
            const size_t n,
            const double alpha,
            double * X,
            const size_t incX,
            bool positive )
```

17.223.2.10 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 )
```

17.223.2.11 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 )
```

17.223.2.12 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 )
```

17.223.2.13 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 )
```

17.223.2.14 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 )
```

17.223.2.15 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 )
```

17.223.2.16 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 )
```

17.223.2.17 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 )
```

17.224 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)

17.224.1 Detailed Description

C functions calls for level 2 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level2.inl

17.224.2 Function Documentation

17.224.2.1 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 )
```

17.224.2.2 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 )
```

17.224.2.3 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 )
```

17.224.2.4 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 )
```

17.224.2.5 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 )
```

17.224.2.6 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 )
```

17.224.2.7 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 )
```

17.224.2.8 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 )
```

17.224.2.9 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 )
```

17.224.2.10 fscalin_2_modular_double()

```
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 )
```

17.224.2.11 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 )
```

17.224.2.12 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 )
```

17.224.2.13 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 )
```

17.224.2.14 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 )
```

17.224.2.15 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 )
```

17.224.2.16 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 )
```

17.224.2.17 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 )
```

17.224.2.18 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 )
```

17.224.2.19 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 )
```

17.224.2.20 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 )
```

17.225 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)

17.225.1 Detailed Description

C functions calls for level 3 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level3.inl

17.225.2 Function Documentation

17.225.2.1 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 )
```

17.225.2.2 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 )
```

17.225.2.3 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 )
```

17.225.2.4 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 betA,
              double * C,
              const size_t ldC,
              bool positive )
```

17.226 fflas_sparse.C File Reference

C functions calls for level 1.5 and 2.5 FFLAS in fflas-c.h.

17.226.1 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

17.227 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 ldx, 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 ldx, 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 ldx, 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 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)
- 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)
- 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 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 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 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 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_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 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)
- 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)
- 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)
- 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)
- 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)
- 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)
- 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)
- 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)
- 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)
- 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)
- 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 ldx, 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 ldx, 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)

17.227.1 Detailed Description

C functions calls for FFPACK in ffpack-c.h.

Author

Brice Boyer

See also

ffpack/ffpack.h

17.227.2 Function Documentation

17.227.2.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
            size_t * MathP,
            const size_t * LapackP,
            const size_t N )
```

17.227.2.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
            size_t * LapackP,
            const size_t * MathP,
            const size_t N )
```

17.227.2.3 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 )
```

17.227.2.4 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 )
```

17.227.2.5 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 )
```

17.227.2.6 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 )
```

17.227.2.7 composePermutationsLLM()

```
void composePermutationsLLM (
            size_t * MathP,
            const size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N )
```

17.227.2.8 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N )
```

17.227.2.9 composePermutationsMLM()

```
void composePermutationsMLM (
            size_t * MathP1,
            const size_t * P2,
            const size_t R,
            const size_t N )
```

17.227.2.10 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s )
```

17.227.2.11 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 )
```

17.227.2.12 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 )
```

17.227.2.13 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 )
```

17.227.2.14 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 )
```

17.227.2.15 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 )
```

17.227.2.16 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 )
```

17.227.2.17 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 )
```

17.227.2.18 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 )
```

17.227.2.19 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 )
```

17.227.2.20 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 )
```

17.227.2.21 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 )
```

17.227.2.22 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 )
```

17.227.2.23 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 )
```

17.227.2.24 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 )
```

17.227.2.25 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 )
```

17.227.2.26 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 )
```

17.227.2.27 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 )
```

17.227.2.28 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 )
```

17.227.2.29 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 )
```

17.227.2.30 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 )
```

17.227.2.31 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 )
```

17.227.2.32 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 )
```

17.227.2.33 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 )
```

17.227.2.34 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 )
```

17.227.2.35 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 )
```

17.227.2.36 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 )
```

17.227.2.37 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 )
```

17.227.2.38 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 )
```

17.227.2.39 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 )
```

17.227.2.40 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 )
```

17.227.2.41 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 )
```

17.227.2.42 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 )
```

17.227.2.43 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 )
```

17.227.2.44 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 )
```

17.227.2.45 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 )
```

17.227.2.46 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 )
```

17.227.2.47 Invertin_modular_double()

```
double * Invertin_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            int * nullity,
            bool positive )
```

17.227.2.48 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 )
```

17.227.2.49 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 )
```

17.227.2.50 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 )
```

17.227.2.51 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 )
```

17.227.2.52 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 )
```

17.227.2.53 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 )
```

17.227.2.54 Det_modular_double()

```
double Det_modular_double (
            const double p,
```

```
            const size_t N,
            double * A,
            const size_t lda,
            bool positive )
```

17.227.2.55 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 )
```

17.227.2.56 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 )
```

17.227.2.57 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 )
```

17.227.2.58 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 )
```

17.227.2.59 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 )
```

17.227.2.60 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 )
```

17.227.2.61 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 )
```

17.227.2.62 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 )
```

17.227.2.63 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 )
```

17.227.2.64 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 )
```

17.227.2.65 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 )
```

17.227.2.66 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 )
```

17.227.2.67 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 )
```

17.227.2.68 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 )
```

17.227.2.69 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 )
```

17.227.2.70 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 )
```

17.227.2.71 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 )
```

17.227.2.72 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 )
```

17.227.2.73 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 )
```

17.227.2.74 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 )
```

17.227.2.75 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 )
```

17.227.2.76 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
            const size_t N,
            const size_t R,
            const size_t * P,
            size_t * outPerm )
```

17.228 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 , FflasRowMajor =101 , FflasColMajor =102 }
- enum FFLAS_C_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 , FflasNoTrans = 111 , FflasTrans = 112 }
- enum FFLAS_C_UPLO { FflasUpper = 121 , FflasLower = 122 , FflasUpper = 121 , FflasLower = 122 }
- enum FFLAS_C_DIAG { FflasNonUnit = 131 , FflasUnit = 132 , FflasNonUnit = 131 , FflasUnit = 132 }
- enum FFLAS_C_SIDE { FflasLeft = 141 , FflasRight = 142 , 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 ldx, 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 ldx, 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)

17.228.1 Macro Definition Documentation

17.228.1.1 FFPACK_COMPILED

```
#define FFPACK_COMPILED
```

17.228.2 Enumeration Type Documentation

17.228.2.1 FFLAS_C_ORDER

```
enum FFLAS_C_ORDER
```

Enumerator

FflasRowMajor	row major
FflasColMajor	col major
FflasRowMajor	
FflasColMajor	

17.228.2.2 FFLAS_C_TRANSPOSE

```
enum FFLAS_C_TRANSPOSE
```

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.
FflasNoTrans	
FflasTrans	

17.228.2.3 FFLAS_C_UPLO

```
enum FFLAS_C_UPLO
```

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 $T_{i,j} = 0$)
FflasUpper	
FflasLower	

17.228.2.4 FFLAS_C_DIAG

```
enum FFLAS_C_DIAG
```

Enumerator

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

17.228.2.5 FFLAS_C_SIDE

```
enum FFLAS_C_SIDE
```

Enumerator

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

Enumerator

FflasLeft	
FflasRight	

17.228.2.6 FFPACK_C_LU_TAG

```
enum FFPACK_C_LU_TAG
```

Enumerator

FfpackSlabRecursive	
FfpackTileRecursive	
FfpackSingular	

17.228.2.7 FFPACK_C_CHARPOLY_TAG

```
enum FFPACK_C_CHARPOLY_TAG
```

Enumerator

FfpackLUK	
FfpackKG	
FfpackHybrid	
FfpackKGFast	
FfpackDanilevski	
FfpackArithProg	
FfpackKGFastG	

17.228.2.8 FFPACK_C_MINPOLY_TAG

```
enum FFPACK_C_MINPOLY_TAG
```

Enumerator

FfpackDense	
FfpackKGF	

17.228.3 Function Documentation

17.228.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
            size_t * MathP,
            const size_t * LapackP,
            const size_t N )
```

17.228.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
            size_t * LapackP,
            const size_t * MathP,
            const size_t N )
```

17.228.3.3 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 )
```

17.228.3.4 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 )
```

17.228.3.5 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 )
```

17.228.3.6 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 )
```

17.228.3.7 composePermutationsLLM()

```
void composePermutationsLLM (
            size_t * MathP,
            const size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N )
```

17.228.3.8 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N )
```

17.228.3.9 composePermutationsMLM()

```
void composePermutationsMLM (
            size_t * MathP1,
            const size_t * P2,
            const size_t R,
            const size_t N )
```

17.228.3.10 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s )
```

17.228.3.11 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 )
```

17.228.3.12 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 )
```

17.228.3.13 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 )
```

17.228.3.14 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 )
```

17.228.3.15 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 )
```

17.228.3.16 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 )
```

17.228.3.17 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 )
```

17.228.3.18 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 )
```

17.228.3.19 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 )
```

17.228.3.20 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 )
```

17.228.3.21 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 )
```

17.228.3.22 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 )
```

17.228.3.23 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 )
```

17.228.3.24 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 )
```

17.228.3.25 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 )
```

17.228.3.26 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 )
```

17.228.3.27 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 )
```

17.228.3.28 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 )
```

17.228.3.29 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 )
```

17.228.3.30 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 )
```

17.228.3.31 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 )
```

17.228.3.32 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 )
```

17.228.3.33 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 )
```

17.228.3.34 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 )
```

17.228.3.35 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 )
```

17.228.3.36 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 )
```

17.228.3.37 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 )
```

17.228.3.38 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 )
```

17.228.3.39 Invertin_modular_double()

```
double * Invertin_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            int * nullity,
            bool positive )
```

17.228.3.40 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 )
```

17.228.3.41 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 )
```

17.228.3.42 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 )
```

17.228.3.43 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 )
```

17.228.3.44 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 )
```

17.228.3.45 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 )
```

17.228.3.46 Det_modular_double()

```
double Det_modular_double (
            const double p,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive )
```

17.228.3.47 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 )
```

17.228.3.48 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 )
```

17.228.3.49 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 )
```

17.228.3.50 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 )
```

17.228.3.51 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 )
```

17.228.3.52 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 )
```

17.228.3.53 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 )
```

17.228.3.54 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 )
```

17.228.3.55 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 )
```

17.228.3.56 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 )
```

17.228.3.57 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 )
```

17.228.3.58 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 )
```

17.228.3.59 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 )
```

17.228.3.60 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 )
```

17.228.3.61 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 )
```

17.228.3.62 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 )
```

17.228.3.63 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 )
```

17.228.3.64 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 )
```

17.228.3.65 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 )
```

17.228.3.66 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 )
```

17.228.3.67 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 )
```

17.228.3.68 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
            const size_t N,
            const size_t R,
            const size_t * P,
            size_t * outPerm )
```

17.229 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

17.229.1 Macro Definition Documentation

17.229.1.1 __FFPACK_INST_C

```
#define __FFPACK_INST_C
```

17.229.1.2 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

17.229.1.3 INST_OR_DECL

```
#define INST_OR_DECL
```

17.229.1.4 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

17.229.1.5 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

17.229.1.6 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

17.229.1.7 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

17.229.1.8 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

17.229.1.9 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

17.229.1.10 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

17.229.1.11 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

17.230 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

17.230.1 Macro Definition Documentation

17.230.1.1 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

17.230.1.2 INST_OR_DECL

```
#define INST_OR_DECL <>
```

17.230.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

17.230.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

17.230.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

17.230.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

17.230.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

17.230.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

17.230.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

17.230.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

17.231 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 x 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 composePermutationsLLL (size_t ∗P1, const size_t ∗P2, const size_t R, const size_t N)

  Computes P1 x 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 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<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)
- 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 ∗ 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)

17.232 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)

17.232.1 Macro Definition Documentation

17.232.1.1 __FFLASFFPACK_fflas_blockcuts_INL

```
#define __FFLASFFPACK_fflas_blockcuts_INL
```

17.232.1.2 __FFLASFFPACK_MINBLOCKCUTS

```
#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)
```

17.233 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)

17.234 kaapi_routines.inl File Reference

Macros

- #define __FFLASFFPACK_KAAPI_ROUTINES_INL

17.234.1 Macro Definition Documentation

17.234.1.1 __FFLASFFPACK_KAAPI_ROUTINES_INL

```
#define __FFLASFFPACK_KAAPI_ROUTINES_INL
```

17.235 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, I) {I;}
- #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__)

17.235.1 Macro Definition Documentation

17.235.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.235.1.2 index_t

```
#define index_t size_t
```

17.235.1.3 TASK

```
#define TASK(
             M,
             I ) {I;}
```

17.235.1.4 WAIT

```
#define WAIT
```

17.235.1.5 CHECK_DEPENDENCIES

```
#define CHECK_DEPENDENCIES
```

17.235.1.6 BARRIER

```
#define BARRIER
```

17.235.1.7 PAR_BLOCK

```
#define PAR_BLOCK
```

17.235.1.8 SYNCH_GROUP

```
#define SYNCH_GROUP(
             Args... ) {{Args};}
```

17.235.1.9 NUM_THREADS

```
#define NUM_THREADS 1
```

17.235.1.10 MAX_THREADS

```
#define MAX_THREADS 1
```

17.235.1.11 READ

```
#define READ(
             Args... )
```

17.235.1.12 WRITE

```
#define WRITE(
            Args... )
```

17.235.1.13 READWRITE

```
#define READWRITE(
            Args... )
```

17.235.1.14 CONSTREFERENCE

```
#define CONSTREFERENCE(
            ... )
```

17.235.1.15 VALUE

```
#define VALUE(
            ... )
```

17.235.1.16 BEGIN_PARALLEL_MAIN

```
#define BEGIN_PARALLEL_MAIN(
            Args... ) int main(Args) {
```

17.235.1.17 END_PARALLEL_MAIN

```
#define END_PARALLEL_MAIN(
            void ) return 0; }
```

17.235.1.18 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;} }
```

17.235.1.19 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; })
```

17.235.1.20 PARFORBLOCK1D

```
#define PARFORBLOCK1D(
                iter,
                m,
                Helper,
                Args... )
```

Value:

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter)      \
{ Args; }
```

17.235.1.21 PARFOR1D

```
#define PARFOR1D(
                iter,
                m,
                Helper,
                Args... )
```

Value:

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter)      \
{ Args; }
```

17.235.1.22 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; } }
```

17.235.1.23 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; })
```

17.235.1.24 PARFORBLOCK2D

```
#define PARFORBLOCK2D(
                iter,
                m,
                n,
                Helper,
                Args... ) FORBLOCK2D(iter, m, n, Helper, Args)
```

17.235.1.25 PARFOR2D

```
#define PARFOR2D(
            i,
            j,
            m,
            n,
            Helper,
            Args... ) FOR2D(i, j, m, n, Helper, Args)
```

17.235.1.26 COMMA

```
#define COMMA ,
```

17.235.1.27 MODE

```
#define MODE(
            ... ) __VA_ARGS__
```

17.235.1.28 RETURNPARAM

```
#define RETURNPARAM(
            f,
            P1,
            Args... ) P1=f(Args)
```

17.235.1.29 NUMARGS

```
#define NUMARGS(
            ... ) PP_NARG_(__VA_ARGS__,PP_RSEQ_N())
```

17.235.1.30 PP_NARG_

```
#define PP_NARG_(
            ... ) PP_ARG_N(__VA_ARGS__)
```

17.235.1.31 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
```

17.235.1.32 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
```

17.235.1.33 NOSPLIT

```
#define NOSPLIT( ) FFLAS::ParSeqHelper::Sequential()
```

17.235.1.34 splitting_0

```
#define splitting_0( ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::
```

17.235.1.35 splitting_1

```
#define splitting_1(
              a ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threa
```

17.235.1.36 splitting_2

```
#define splitting_2(
              a,
              c ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,c>(a)
```

17.235.1.37 splitting_3

```
#define splitting_3(
              a,
              b,
              c ) FFLAS::ParSeqHelper::Parallel<b,c>(a)
```

17.235.1.38 splitt

```
#define splitt(
              _1,
              _2,
              _3,
              NAME,
              ...   ) NAME
```

17.235.1.39 SPLITTER

```
#define SPLITTER(
              ...   ) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0)(←
__VA_ARGS__)
```

17.236 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)

17.237 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, 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)

17.238 align-allocator.h File Reference

```
#include "fflas-ffpack/config.h"
```

17.239 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

17.239.1 Macro Definition Documentation

17.239.1.1 TYPE_BOOL

```
#define TYPE_BOOL TYPE_NONE
```

17.239.1.2 END_OF_ARGUMENTS

```
#define END_OF_ARGUMENTS { '\0', "\0", "\0", TYPE_NONE, NULL }
```

17.239.1.3 type_integer

```
#define type_integer long int
```

17.239.2 Enumeration Type Documentation

17.239.2.1 ArgumentType

```
enum ArgumentType
```

Enumerator

TYPE_NONE	
TYPE_INT	
TYPE_UINT64	
TYPE_LONGLONG	
TYPE_INTEGER	
TYPE_DOUBLE	
TYPE_INTLIST	
TYPE_STR	

17.239.3 Function Documentation

17.239.3.1 printHelpMessage()

```
void printHelpMessage (
            const char * program,
            Argument * args,
            bool printDefaults = false )
```

17.239.3.2 findArgument()

```
Argument * findArgument (
            Argument * args,
            char c )
```

17.239.3.3 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

outlist	list once converted
instring	list to be converted

Returns

status message.

17.240 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)

17.240.1 Macro Definition Documentation

17.240.1.1 __has_builtin

```
#define __has_builtin(
              x ) 0
```

17.240.2 Function Documentation

17.240.2.1 clz() [1/2]

```
int32_t clz (
            uint64_t val ) [inline]
```

17.240.2.2 clz() [2/2]

```
int32_t clz (
            uint32_t val ) [inline]
```

17.240.2.3 ctz() [1/2]

```
int32_t ctz (
            uint32_t val ) [inline]
```

17.240.2.4 ctz() [2/2]

```
int32_t ctz (
            uint64_t val ) [inline]
```

17.241 cast.h File Reference

Namespaces

- namespace FFPACK

  F*inite* ***F****ield* ***PACK*** *Set of elimination based routines for dense linear algebra.*

Functions

- template<class T , class CT = const T>
  T fflas_const_cast (CT x)

17.242 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 precondtion failed.

Namespaces

- namespace FFPACK

  F*inite* ***F****ield* ***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)

17.242.1 Detailed Description

Various utilities for debugging.

Todo we should put vector printing elsewhere.

17.242.2 Macro Definition Documentation

17.242.2.1 FFLASFFPACK_check

```
#define FFLASFFPACK_check(
                check )
```

Value:

```
if (!(check)) {\
    FFPACK::failure()(__func__, __FILE__, __LINE__, #check); \
    throw std::runtime_error(#check); \
}
```

17.242.2.2 FFLASFFPACK_abort

```
#define FFLASFFPACK_abort(
                msg )
```

Value:

```
{\
    FFPACK::failure()(__func__, __FILE__, __LINE__, msg); \
    throw std::runtime_error(msg); \
}
```

17.243 fflas_intrinsic.h File Reference

17.244 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 &ifs, FFLAS_FORMAT &format)

- 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.
- 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.

17.245 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 ()

17.246 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

  ***F**inite **F**ield **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 $\{0,\ldots,N-1\}$ 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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.

17.247 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< RecInt::ruint< K > >
- struct limits< RecInt::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)

17.247.1 Function Documentation

17.247.1.1 in_range() [1/3]

```
std::enable_if< std::is_signed< T >::value==std::is_signed< E >::value, bool >::type in_↩
range (
            E e )
```

17.247.1.2 in_range() [2/3]

```
std::enable_if<(std::is_signed< T >::value)&&!(std::is_signed< E >::value), bool >::type
in_range (
            E e )
```

17.247.1.3 in_range() [3/3]

```
std::enable_if<!(std::is_signed< T >::value)&&(std::is_signed< E >::value), bool >::type
in_range (
            E e )
```

17.248 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 mapleFormat=false, bool column_major=false)

17.248.1 Function Documentation

17.248.1.1 read_field()

```
Field::Element_ptr read_field (
            const Field & F,
            const char * mat_file,
            size_t * tni,
            size_t * tnj )
```

17.248.1.2 write_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 )
```

17.249 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)

17.250 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

17.251 cblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

Macros

- #define __FFLASFFPACK_CONFIGURATION
- #define __FFLASFFPACK_HAVE_CBLAS 1

Functions

- int main ()

17.251.1 Macro Definition Documentation

17.251.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

17.251.1.2 __FFLASFFPACK_HAVE_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

17.251.2 Function Documentation

17.251.2.1 main()

```
int main (
            void )
```

17.252 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 ()

17.252.1 Macro Definition Documentation

17.252.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

17.252.1.2 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

17.252.1.3 __FFLASFFPACK_HAVE_CLAPACK

```
#define __FFLASFFPACK_HAVE_CLAPACK 1
```

17.252.2 Function Documentation

17.252.2.1 main()

```
int main (
            void )
```

17.253 cuda.C File Reference

```
#include <stdio.h>
#include <cuda_runtime.h>
#include <cusparse.h>
```

Functions

- int main ()

17.253.1 Function Documentation

17.253.1.1 main()

```
int main (
            void )
```

17.254 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 ()

17.254.1 Macro Definition Documentation

17.254.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

17.254.2 Function Documentation

17.254.2.1 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 *  )
```

17.254.2.2 main()

```
int main (
          void  )
```

17.255 gmp.C File Reference

`#include <gmpxx.h>`

Functions

- int main ()

17.255.1 Function Documentation

17.255.1.1 main()

```
int main (
          void  )
```

17.256 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)

17.256.1 Macro Definition Documentation

17.256.1.1 INSTRSET_H

```
#define INSTRSET_H 125
```

17.256.1.2 INSTRSET

```
#define INSTRSET 0
```

17.256.1.3 const_int

```
#define const_int(
              n ) (Const_int_t <n>())
```

17.256.1.4 const_uint

```
#define const_uint(
              n ) (Const_uint_t<n>())
```

17.256.2 Typedef Documentation

17.256.2.1 int8_t

```
typedef signed char int8_t
```

17.256.2.2 uint8_t

```
typedef unsigned char uint8_t
```

17.256.2.3 int16_t

```
typedef signed short int int16_t
```

17.256.2.4 uint16_t

```
typedef unsigned short int uint16_t
```

17.256.2.5 int32_t

```
typedef signed int int32_t
```

17.256.2.6 uint32_t

```
typedef unsigned int uint32_t
```

17.256.2.7 int64_t

```
typedef long long int64_t
```

17.256.2.8 uint64_t

```
typedef unsigned long long uint64_t
```

17.256.2.9 intptr_t

```
typedef int32_t intptr_t
```

17.256.3 Function Documentation

17.256.3.1 instrset_detect()

```
int instrset_detect (
             void )
```

17.256.3.2 hasFMA3()

```
bool hasFMA3 (
             void )
```

17.256.3.3 hasFMA4()

```
bool hasFMA4 (
             void )
```

17.256.3.4 hasXOP()

```
bool hasXOP (
             void )
```

17.256.3.5 hasAVX512ER()

```
bool hasAVX512ER (
             void )
```

17.257 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)

17.257.1 Function Documentation

17.257.1.1 instrset_detect()

```
int instrset_detect (
            void )
```

17.257.1.2 hasFMA3()

```
bool hasFMA3 (
            void )
```

17.257.1.3 hasFMA4()

```
bool hasFMA4 (
            void )
```

17.257.1.4 hasXOP()

```
bool hasXOP (
            void )
```

17.257.1.5 hasF16C()

```
bool hasF16C (
            void )
```

17.257.1.6 hasAVX512ER()

```
bool hasAVX512ER (
            void )
```

17.258 lapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

Macros

- #define __FFLASFFPACK_CONFIGURATION
- #define __FFLASFFPACK_HAVE_LAPACK 1

Functions

- int main ()

17.258.1 Macro Definition Documentation

17.258.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

17.258.1.2 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

17.258.2 Function Documentation

17.258.2.1 main()

```
int main (
            void  )
```

17.259 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 ()

17.259.1 Function Documentation

17.259.1.1 check1()

```
bool check1 ( )
```

17.259.1.2 check2()

```
bool check2 ( )
```

17.259.1.3 check3()

```
bool check3 ( )
```

17.259.1.4 check4()

```
bool check4 ( )
```

17.259.1.5 checkZeroDimCharpoly()

```
bool checkZeroDimCharpoly ( )
```

17.259.1.6 checkZeroDimMinPoly()

```
bool checkZeroDimMinPoly ( )
```

17.259.1.7 gf2ModularBalanced()

```
bool gf2ModularBalanced ( )
```

17.259.1.8 main()

```
int main (
            void )
```

17.260 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)

17.260.1 Macro Definition Documentation

17.260.1.1 ENABLE_CHECKER_charpoly

```
#define ENABLE_CHECKER_charpoly 1
```

17.260.1.2 TIME_CHECKER_CHARPOLY

```
#define TIME_CHECKER_CHARPOLY 1
```

17.260.2 Function Documentation

17.260.2.1 printPolynomial()

```
void printPolynomial (
            const Field & F,
            Polynomial & v )
```

17.260.2.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.261 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)

17.261.1 Function Documentation

17.261.1.1 launch_test()

```
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 )
```

17.261.1.2 run_with_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 )
```

17.261.1.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.262 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)

17.262.1 Typedef Documentation

17.262.1.1 Field

```
typedef Givaro::Modular<double> Field
```

17.262.2 Function Documentation

17.262.2.1 printvect()

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

Bug does not belong here

17.262.2.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.263 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)

17.263.1 Macro Definition Documentation

17.263.1.1 ENABLE_CHECKER_Det

```
#define ENABLE_CHECKER_Det 1
```

17.263.1.2 TIME_CHECKER_Det

```
#define TIME_CHECKER_Det 1
```

17.263.2 Function Documentation

17.263.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.264 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)

17.264.1 Function Documentation

17.264.1.1 test_det()

```
bool test_det (
            Field & F,
            size_t n,
            int iter,
            RandIter & G )
```

Todo test with stride

17.264.1.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.265 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 Lu↩
  Tag, 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)

17.265.1 Macro Definition Documentation

17.265.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.265.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25
```

17.265.1.3 __FFLASFFPACK_PLUQ_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 25
```

17.265.2 Function Documentation

17.265.2.1 test_colechelon()

```
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 lda

17.265.2.2 test_rowechelon()

```
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 lda

17.265.2.3 test_redcolechelon()

```
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 lda

17.265.2.4 test_redrowechelon()

```
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 lda

17.265.2.5 run_with_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 )
```

17.265.2.6 main()

```
int main (
            int argc,
            char ** argv )
```

17.266 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)

17.266.1 Function Documentation

17.266.1.1 test_fadd()

```
bool test_fadd (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed )
```

17.266.1.2 test_faddin()

```
bool test_faddin (
            const Field & F,
            size_t m,
            size_t k,
```

```
            size_t n,
            bool timing,
            uint64_t seed )
```

17.266.1.3 test_fsub()

```
bool test_fsub (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed )
```

17.266.1.4 test_fsubin()

```
bool test_fsubin (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed )
```

17.266.1.5 main()

```
int main (
            int ac,
            char ** av )
```

17.267 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)

17.267.1 Macro Definition Documentation

17.267.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.267.2 Function Documentation

17.267.2.1 check_fdot()

```
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 )
```

17.267.2.2 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            size_t BS,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.267.2.3 run_with_Integer()

```
bool run_with_Integer (
            size_t BS,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.267.2.4 main()

```
int main (
            int argc,
            char ** argv )
```

17.268 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)

17.268.1 Macro Definition Documentation

17.268.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.268.2 Function Documentation

17.268.2.1 launch_MM_dispatch()

```
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 ldX equal

Bug

Bug test for transpo

Bug

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

17.268.2.2 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            int m,
            int n,
            int k,
            size_t iters,
            uint64_t seed )
```

17.268.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.269 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)

17.269.1 Macro Definition Documentation

17.269.1.1 ENABLE_CHECKER_fgemm

```
#define ENABLE_CHECKER_fgemm 1
```

17.269.2 Function Documentation

17.269.2.1 check_MM()

```
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 )
```

17.269.2.2 launch_MM()

```
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 )
```

17.269.2.3 launch_MM_dispatch()

```
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 ?) ?

17.269.2.4 run_with_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 )
```

17.269.2.5 main()

```
int main (
            int argc,
            char ** argv )
```

17.270 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)

17.270.1 Function Documentation

17.270.1.1 check_MV()

```
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 )
```

17.270.1.2 launch_MV()

```
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 )
```

17.270.1.3 launch_MV_dispatch()

```
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 )
```

17.270.1.4 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            int m,
            int k,
            size_t iters,
            bool par,
            uint64_t seed )
```

17.270.1.5 main()

```
int main (
            int argc,
            char ** argv )
```

17.271 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)

17.271.1 Macro Definition Documentation

17.271.1.1 TIME

```
#define TIME 1
```

17.271.2 Function Documentation

17.271.2.1 check_fger()

```
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 )
```

17.271.2.2 launch_fger()

```
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 )
```

17.271.2.3 launch_fger_dispatch()

```
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 ?) ?

17.271.2.4 run_with_field()

```
bool run_with_field (
            int64_t q,
            uint64_t b,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.271.2.5 main()

```
int main (
            int argc,
            char ** argv )
```

17.272 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)

17.272.1 Function Documentation

17.272.1.1 test_square_fgesv()

```
bool test_square_fgesv (
            Field & F,
            FFLAS_SIDE side,
            string fileA,
            string fileB,
            size_t m,
            size_t k,
            size_t r,
            RandIter & G )
```

17.272.1.2 test_rect_fgesv()

```
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 )
```

17.272.1.3 run_with_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 )
```

17.272.1.4 main()

```
int main (
            int argc,
            char ** argv )
```

17.273 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)

17.273.1 Function Documentation

17.273.1.1 test_freduce()

```
bool test_freduce (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed )
```

17.273.1.2 run_with_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 )
```

17.273.1.3 main()

```
int main (
            int ac,
            char ** av )
```

17.274 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)

17.274.1 Function Documentation

17.274.1.1 test_fscal() [1/2]

```
bool test_fscal (
            const Field & F,
            const typename Field::Element & alpha,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            RandIter & G )
```

17.274.1.2 test_fscal() [2/2]

```
bool test_fscal (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed )
```

17.274.1.3 test_fscalin() [1/2]

```
bool test_fscalin (
            const Field & F,
            const typename Field::Element & alpha,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            RandIter & G )
```

17.274.1.4 test_fscalin() [2/2]

```
bool test_fscalin (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed )
```

17.274.1.5 main()

```
int main (
            int ac,
            char ** av )
```

17.275 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)

17.275.1 Macro Definition Documentation

17.275.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.275.2 Function Documentation

17.275.2.1 check_fsyr2k()

```
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 )
```

17.275.2.2 run_with_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 )
```

17.275.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.276 test-fsyrk.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_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)
- 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)
- 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)
- 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)

17.276.1 Macro Definition Documentation

17.276.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.276.2 Function Documentation

17.276.2.1 check_fsyrk()

```
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 )
```

17.276.2.2 check_fsyrk_diag()

```
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 )
```

17.276.2.3 check_fsyrk_bkdiag()

```
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 )
```

17.276.2.4 run_with_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 )
```

17.276.2.5 main()

```
int main (
            int argc,
            char ** argv )
```

17.277 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)

17.277.1 Function Documentation

17.277.1.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & os,
            const std::vector< T > & x )
```

17.277.1.2 test_RPM_fsytrf()

```
bool test_RPM_fsytrf (
            Field & F,
            FFLAS_UPLO uplo,
            string file,
            size_t n,
            size_t r,
            RandIter & G,
            size_t threshold )
```

17.277.1.3 test_generic_fsytrf()

```
bool test_generic_fsytrf (
            Field & F,
            FFLAS_UPLO uplo,
            string file,
            size_t n,
```

```
RandIter & G,
size_t threshold )
```

17.277.1.4 run_with_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 )
```

17.277.1.5 main()

```
int main (
            int argc,
            char ** argv )
```

17.278 test-ftrmm.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_ftrmm (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)

17.278.1 Macro Definition Documentation

17.278.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.278.2 Function Documentation

17.278.2.1 check_ftrmm()

```
bool check_ftrmm (
            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 )
```

17.278.2.2 run_with_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 )
```

17.278.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.279 test-ftrmv.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)

17.279.1 Macro Definition Documentation

17.279.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.279.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.279.2 Function Documentation

17.279.2.1 check_ftrmv()

```
bool check_ftrmv (
            const Field & F,
            size_t n,
            FFLAS_UPLO uplo,
            FFLAS_TRANSPOSE trans,
            FFLAS_DIAG diag,
            RandIter & Rand )
```

17.279.2.2 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.279.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.280 test-ftrsm-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)

17.280.1 Macro Definition Documentation

17.280.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.280.2 Function Documentation

17.280.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.281 test-ftrsm.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_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)

- 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)

17.281.1 Macro Definition Documentation

17.281.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.281.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.281.2 Function Documentation

17.281.2.1 check_ftrsm()

```
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 )
```

17.281.2.2 run_with_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 )
```

17.281.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.282 test-ftrssyr2k.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_ftrssyr2k (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)

17.282.1 Macro Definition Documentation

17.282.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.282.2 Function Documentation

17.282.2.1 check_ftrssyr2k()

```
bool check_ftrssyr2k (
            const Field & F,
            size_t n,
            FFLAS::FFLAS_UPLO uplo,
            FFLAS::FFLAS_DIAG diagA,
            RandIter & Rand )
```

17.282.2.2 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.282.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.283 test-ftrstr.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)

17.283.1 Macro Definition Documentation

17.283.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.283.2 Function Documentation

17.283.2.1 check_ftrstr()

```
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 )
```

17.283.2.2 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.283.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.284 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_ftrsv (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)

17.284.1 Macro Definition Documentation

17.284.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.284.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.284.2 Function Documentation

17.284.2.1 check_ftrsv()

```
bool check_ftrsv (
            const Field & F,
            size_t n,
            FFLAS_UPLO uplo,
            FFLAS_TRANSPOSE trans,
            FFLAS_DIAG diag,
            RandIter & Rand )
```

17.284.2.2 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.284.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.285 test-ftrtri.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_ftrtri (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)

17.285.1 Macro Definition Documentation

17.285.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.285.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.285.2 Function Documentation

17.285.2.1 check_ftrtri()

```
bool check_ftrtri (
            const Field & F,
            size_t n,
            FFLAS_UPLO uplo,
            FFLAS_DIAG diag,
            RandIter & Rand )
```

17.285.2.2 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.285.2.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.286 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 ()

17.286.1 Function Documentation

17.286.1.1 main()

```
int main (
            void )
```

17.287 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)

17.287.1 Macro Definition Documentation

17.287.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.287.2 Function Documentation

17.287.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.288 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)

17.288.1 Function Documentation

17.288.1.1 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            size_t m,
            size_t n,
            size_t iters,
            uint64_t seed )
```

17.288.1.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.289 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 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<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)
- 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

17.289.1 Macro Definition Documentation

17.289.1.1 BASECASE_K

```
#define BASECASE_K 37
```

17.289.1.2 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.289.1.3 __LUDIVINE_CUTOFF

```
#define __LUDIVINE_CUTOFF 1
```

17.289.2 Function Documentation

17.289.2.1 test_LUdivine()

```
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

Field	Field
Diag	Unit diagonal in U
Trans	

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

17.289.2.2 verifPLUQ()

```
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 Parameters

Field	Field
Diag	Unit diagonal in U

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

17.289.2.3 test_pluq()

```
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

Field	Field
Diag	Unit diagonal in U
Trans	

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

17.289.2.4 launch_test()

```
bool launch_test (
                const Field & F,
                size_t r,
                size_t m,
                size_t n,
                RandIter & G )
```

17.289.2.5 run_with_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 )
```

17.289.2.6 main()

```
int main (
                int argc,
                char ** argv )
```

17.289.3 Variable Documentation

17.289.3.1 tperm

```
Givaro::Timer tperm
```

17.289.3.2 tgemm

```
Givaro::Timer tgemm
```

17.289.3.3 tBC

```
Givaro::Timer tBC
```

17.289.3.4 ttrsm

```
Givaro::Timer ttrsm
```

17.289.3.5 trest

```
Givaro::Timer trest
```

17.289.3.6 timtot

```
Givaro::Timer timtot
```

17.289.3.7 mvcnt

```
size_t mvcnt = 0
```

17.290 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 ()

17.290.1 Macro Definition Documentation

17.290.1.1 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 )
```

17.290.2 Function Documentation

17.290.2.1 test()

```
bool test (
             Givaro::Integer p,
             size_t kmax )
```

17.290.2.2 main()

```
int main (
             void )
```

17.291 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/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/test-utils.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)

17.291.1 Function Documentation

17.291.1.1 check_minpoly()

```
bool check_minpoly (
             const Field & F,
```

```
              size_t n,
              RandIter & G )
```

17.291.1.2 run_with_field()

```
bool run_with_field (
              Givaro::Integer q,
              size_t b,
              size_t n,
              size_t iters,
              uint64_t seed )
```

17.291.1.3 main()

```
int main (
              int argc,
              char ** argv )
```

17.292 test-multifile1.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

17.293 test-multifile2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int main (void)

17.293.1 Function Documentation

17.293.1.1 main()

```
int main (
              void )
```

17.294 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)

17.294.1 Function Documentation

17.294.1.1 checkingMessage()

```
std::string checkingMessage (
            const Field & F )
```

17.294.1.2 readOrRandomMatrixWithRankAndRandomRPM()

```
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.

17.294.1.3 test_nullspace()

```
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 )
```

17.294.1.4 run_with_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 )
```

17.294.1.5 main()

```
int main (
            int argc,
            char ** argv )
```

17.295 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

17.295.1 Function Documentation

17.295.1.1 checkMonotonicApplyP()

```
bool checkMonotonicApplyP (
            FFLAS_SIDE Side,
            FFLAS_TRANSPOSE trans,
            size_t * P,
            size_t N,
            size_t R )
```

17.295.1.2 main()

```
int main (
            void )
```

17.295.2 Variable Documentation

17.295.2.1 tperm

```
Givaro::Timer tperm
```

17.295.2.2 tgemm

```
Givaro::Timer tgemm
```

17.295.2.3 tBC

```
Givaro::Timer tBC
```

17.295.2.4 ttrsm

```
Givaro::Timer ttrsm
```

17.295.2.5 trest

```
Givaro::Timer trest
```

17.295.2.6 timtot

```
Givaro::Timer timtot
```

17.296 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)

17.296.1 Macro Definition Documentation

17.296.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

17.296.2 Function Documentation

17.296.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.297 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)

17.297.1 Macro Definition Documentation

17.297.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

17.297.2 Function Documentation

17.297.2.1 run_with_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 )
```

17.297.2.2 main()

```
int main (
            int argc,
            char ** argv )
```

17.298 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)

17.298.1 Function Documentation

17.298.1.1 checkRPM()

```
bool checkRPM (
            size_t M,
            size_t N,
            size_t R )
```

17.298.1.2 checkSymmetricRPM()

```
bool checkSymmetricRPM (
            size_t N,
            size_t R )
```

17.298.1.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.299 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 fname)
- 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[ ])

17.299.1 Macro Definition Documentation

17.299.1.1 REGISTER_TYPE_NAME

```
#define REGISTER_TYPE_NAME(
              type )  template<> const char *TypeName<type>(){return #type;}
```

17.299.1.2 TEST_ONE_OP

```
#define TEST_ONE_OP(
              name )  btest &= test_op<simd> (simd::name, Scal::name, #name);
```

17.299.2 Typedef Documentation

17.299.2.1 integer

```
typedef Givaro::Integer integer
```

17.299.3 Function Documentation

17.299.3.1 TypeName()

```
const char * TypeName ( )
```

17.299.3.2 REGISTER_TYPE_NAME() [1/8]

```
REGISTER_TYPE_NAME (
             float  )
```

17.299.3.3 REGISTER_TYPE_NAME() [2/8]

```
REGISTER_TYPE_NAME (
             double  )
```

17.299.3.4 REGISTER_TYPE_NAME() [3/8]

```
REGISTER_TYPE_NAME (
             int16_t  )
```

17.299.3.5 REGISTER_TYPE_NAME() [4/8]

```
REGISTER_TYPE_NAME (
            int32_t )
```

17.299.3.6 REGISTER_TYPE_NAME() [5/8]

```
REGISTER_TYPE_NAME (
            int64_t )
```

17.299.3.7 REGISTER_TYPE_NAME() [6/8]

```
REGISTER_TYPE_NAME (
            uint16_t )
```

17.299.3.8 REGISTER_TYPE_NAME() [7/8]

```
REGISTER_TYPE_NAME (
            uint32_t )
```

17.299.3.9 REGISTER_TYPE_NAME() [8/8]

```
REGISTER_TYPE_NAME (
            uint64_t )
```

17.299.3.10 generate_random_vector() [1/2]

```
enable_if< is_integral< Element >::value >::type generate_random_vector (
            vector< Element, Alloc > & a )
```

17.299.3.11 generate_random_vector() [2/2]

```
enable_if< is_floating_point< Element >::value >::type generate_random_vector (
            vector< Element, Alloc > & a )
```

17.299.3.12 check_eq() [1/2]

```
enable_if< is_integral< Element >::value, bool >::type check_eq (
            Element x,
            Element y )
```

17.299.3.13 check_eq() [2/2]

```
enable_if< is_floating_point< Element >::value, bool >::type check_eq (
            Element x,
            Element y )
```

17.299.3.14 eval_func_on_array() [1/2]

```
Ret eval_func_on_array (
            function< Ret()> f,
            array< T, 0 > arr )
```

17.299.3.15 eval_func_on_array() [2/2]

```
Ret eval_func_on_array (
            function< Ret(T, TArgs...)> f,
            array< typename remove_reference< T >::type, sizeof...(TArgs)+1 > & arr )
```

17.299.3.16 test_op()

```
enable_if< sizeof...(AScal)==sizeof...(ASimd), bool >::type test_op (
            RSimd(&)(ASimd...)  FSimd,
            RScal(&)(AScal...)  FScal,
            string fname )
```

17.299.3.17 test_impl() [1/2]

```
enable_if< is_floating_point< Element >::value, bool >::type test_impl ( )
```

17.299.3.18 test_impl() [2/2]

```
enable_if< is_integral< Element >::value, bool >::type test_impl ( )
```

17.299.3.19 test() [1/2]

```
enable_if< is_integral< Element >::value, bool >::type test ( )
```

17.299.3.20 test() [2/2]

```
enable_if< is_floating_point< Element >::value, bool >::type test ( )
```

17.299.3.21 main()

```
int main (
            int argc,
            char * argv[] )
```

17.300 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)

17.300.1 Function Documentation

17.300.1.1 check_solve()

```
bool check_solve (
            const Field & F,
            size_t m,
            RandIter & Rand,
            bool isParallel )
```

17.300.1.2 run_with_field()

```
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t m,
            size_t iters,
            uint64_t seed )
```

17.300.1.3 main()

```
int main (
            int argc,
            char ** argv )
```

17.301 101-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)

17.301.1 Function Documentation

17.301.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.302 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)

17.302.1 Function Documentation

17.302.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.303 2x2-ftrsv.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)

17.303.1 Function Documentation

17.303.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.304 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)

17.304.1 Function Documentation

17.304.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.305 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)

17.305.1 Function Documentation

17.305.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.306 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)

17.306.1 Function Documentation

17.306.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.307 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)

17.307.1 Function Documentation

17.307.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.308 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)

17.308.1 Function Documentation

17.308.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.309 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)

17.309.1 Function Documentation

17.309.1.1 main()

```
int main (
            int argc,
            char ** argv )
```

17.310 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)

17.310.1 Function Documentation

17.310.1.1 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*x will not be equal to b for the later verification stage

Index