Knowing your ARM from your ARSE

(or, how to identify the CPU features in your phone)

Embedded Recipes, Paris

Will Deacon <will@kernel.org>

September, 2019

Disclaimer/excuses:

I no longer work at Arm!

Arm are sponsoring this conference and have kindly paid for my travel to be here.

I usually give fairly deep, low-level technical presentations, so this is a change for me.
Feel free to stop/interrupt me during the talk.

This presentation is based on work done at Arm and is not part of my current employment

Introduction

- Linux kernel co-maintainer of `arm64` architecture, `ARM` perf backends, `SMMU` drivers, atomics, locking, memory model, TLB invalidation…
- Worked for a decade in the Open-Source Software group at Arm
- Contributed significantly to the Armv8 architecture
- Heavy exposure to the development of the Arm ecosystem
- Tend to be CPU centric

Lots of insider knowledge required to overcome hurdles and develop successful system software for Arm…

Spot the difference

Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)

Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)
- The `ARM7TDMI` variant was particularly popular. The ‘T’ means ‘Thumb’

Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)
- The `ARM7TDMI` variant was particularly popular. The ‘T’ means ‘Thumb’
- Nowadays, most 32-bit Arm CPUs tend to implement `ARMv7-A`…

Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)
- The `ARM7TDMI` variant was particularly popular. The ‘T’ means ‘Thumb’
- Nowadays, most 32-bit Arm CPUs tend to implement `ARMv7-A`…
- …but usually people just say `ARMv7`

Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)
- The `ARM7TDMI` variant was particularly popular. The ‘T’ means ‘Thumb’
- Nowadays, most 32-bit Arm CPUs tend to implement `ARMv7-A`…
- …but usually people just say `ARMv7`
- For example, the `Cortex-A7`, or `A7`, which implements ‘Thumb-2’

Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)
- The `ARM7TDMI` variant was particularly popular. The ‘T’ means ‘Thumb’
- Nowadays, most 32-bit Arm CPUs tend to implement `ARMv7-A`…
- …but usually people just say `ARMv7`
- For example, the `Cortex-A7`, or `A7`, which implements ‘Thumb-2’
- Not be confused with the ‘Apple A7’, which is 64-bit according to wikipedia

Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)
- The `ARM7TDMI` variant was particularly popular. The ‘T’ means ‘Thumb’
- Nowadays, most 32-bit Arm CPUs tend to implement `ARMv7-A`…
- …but usually people just say `ARMv7`
- For example, the `Cortex-A7`, or `A7`, which implements ‘Thumb-2’
- Not be confused with the ‘Apple A7’, which is 64-bit according to wikipedia
- `Cortex-A7` also implements virtualisation and SMP, unlike the `Cortex-A8` which came out years before

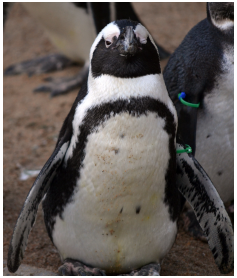
Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)
- The `ARM7TDMI` variant was particularly popular. The ‘T’ means ‘Thumb’
- Nowadays, most 32-bit Arm CPUs tend to implement `ARMv7-A`…
- …but usually people just say `ARMv7`
- For example, the `Cortex-A7`, or `A7`, which implements ‘Thumb-2’
- Not be confused with the ‘Apple A7’, which is 64-bit according to wikipedia
- `Cortex-A7` also implements virtualisation and SMP, unlike the `Cortex-A8` which came out years before
- Recently, the `Cortex-A77` was announced, which implements `Armv8` and is 64-bit

Everything is fine

- `ARM7` was a 32-bit ARM CPU implementing `ARMv3` (around 20 years ago)
- The `ARM7TDMI` variant was particularly popular. The ‘T’ means ‘Thumb’
- Nowadays, most 32-bit Arm CPUs tend to implement `ARMv7-A`…
- …but usually people just say `ARMv7`
- For example, the `Cortex-A7`, or `A7`, which implements ‘Thumb-2’
- Not be confused with the ‘Apple A7’, which is 64-bit according to wikipedia
- `Cortex-A7` also implements virtualisation and SMP, unlike the `Cortex-A8` which came out years before
- Recently, the `Cortex-A77` was announced, which implements `Armv8` and is 64-bit
- Although the `Cortex-A32` also implements `Armv8` but is 32-bit only

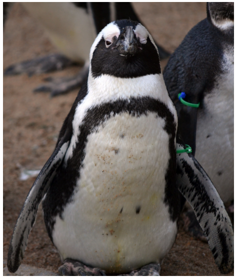
Even Pinhead is confused (July, 2012)



*"They have separate versions for their 'architecture' (ex: arm v8) and for their 'implementation' (ex: ARM11), and maybe it all makes sense if you have drunk the ARM cool-aid and have joined the ARM cult, but to sane people and outsiders, it is just a f*cking mess."*

*"Christ. Seriously. The insanity is so strong in the ARM version names that it burns. If it really makes sense to anybody that 'ARM A9' (technically 'Cortex-A9', but nobody seems to use the 'Cortex' part at least in cellphones) is an 'ARM-v7' architecture microprocessor which is *completely* different from the ARM9 family, which in turn is different from ARMv9 that hasn't happened yet, you need to have your head examined."*

Even Pinhead is confused (July, 2012)

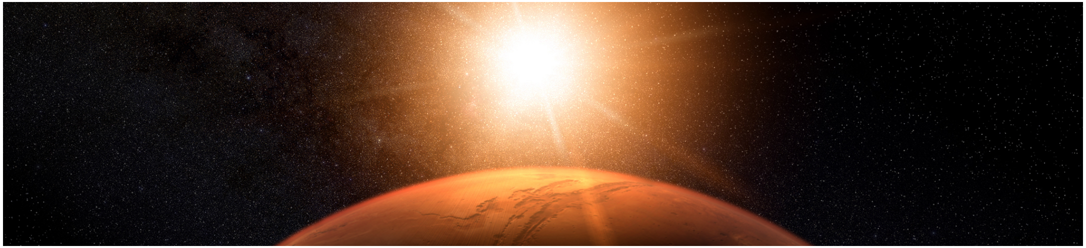


*"They have separate versions for their 'architecture' (ex: arm v8) and for their 'implementation' (ex: ARM11), and maybe it all makes sense if you have drunk the ARM cool-aid and have joined the ARM cult, but to sane people and outsiders, it is just a f*cking mess."*

*"Christ. Seriously. The insanity is so strong in the ARM version names that it burns. If it really makes sense to anybody that 'ARM A9' (technically 'Cortex-A9', but nobody seems to use the 'Cortex' part at least in cellphones) is an 'ARM-v7' architecture microprocessor which is *completely* different from the ARM9 family, which in turn is different from ARMv9 that hasn't happened yet, you need to have your head examined."*

So let's drink the 'ARM cool-aid' and figure out what's going on…

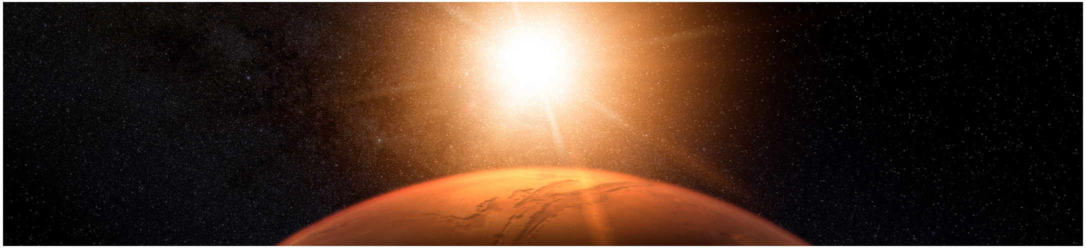
Documentation: off to a good start



Everything you need to know is described in a handy book!

- Hilariously called the 'Arm ARM' (miraculously, you can search for this)
- Almost 8000 pages long
- Sporadic releases; usually not quite up-to-date (arch changes weekly)
- Acts as a reference, not as a learning resource. Deliberately avoids CPU details.
- Did you know?

Documentation: off to a good start



Everything you need to know is described in a handy book!

- Hilariously called the ‘Arm ARM’ (miraculously, you can search for this)
- Almost 8000 pages long
- Sporadic releases; usually not quite up-to-date (arch changes weekly)
- Acts as a reference, not as a learning resource. Deliberately avoids CPU details.
- Did you know? If you printed a copy for every shipped CPU, it would reach Mars and back 500x!

The Arm ARM: Maybe not so handy after all?

“Any interrupt that is pending before a Context synchronization event in the following list, is taken before the first instruction after the context synchronizing event, provided that the pending interrupt is not masked:

- *Execution of an ISB instruction.*
- *Exception entry, if ARMv8.5-CSEH is not implemented, or if ARMv8.5-CSEH is implemented and the appropriate SCTLR_ELx.EIS bit is set.*
- *Exception return, if ARMv8.5-CSEH is not implemented or if ARMv8.5-CSEH is implemented and the appropriate SCTLR_ELx.EOS bit is set.*
- *Exit from Debug state.”*

The Arm ARM: Maybe not so handy after all?

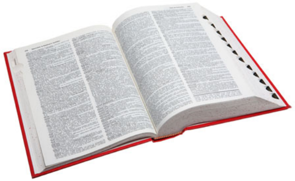
"Any interrupt that is pending before a Context synchronization event in the following list, is taken before the first instruction after the context synchronizing event, provided that the pending interrupt is not masked:

- *Execution of an ISB instruction.*
- *Exception entry, if ARMv8.5-CSEH is not implemented, or if ARMv8.5-CSEH is implemented and the appropriate SCTLR_ELx.EIS bit is set.*
- *Exception return, if ARMv8.5-CSEH is not implemented or if ARMv8.5-CSEH is implemented and the appropriate SCTLR_ELx.EOS bit is set.*
- *Exit from Debug state."*

This is not the 'cool-aid' you are looking for.

Hurdle 1: Arm is mostly inaccessible to developers on the outside.

Important terminology



- Architecture vs micro-architecture
- Core license vs architecture license
- Diversity vs fragmentation
- CPU vs SoC
- ISA vs system architecture

Understanding these differences is crucial to picking apart and understanding the Arm ecosystem.

Architecture vs Micro-architecture

Architecture is the contract between hardware and software. It describes the portable guarantees provided by the hardware, without dictating how they are implemented, and is typically specified in English. Arm has three architecture *profiles*: A, R and M (more hilarity). Approx. 5yr lag.

Micro-architecture relates to the design of a specific piece of hardware, and is typically specified in a hardware description language such as Verilog.

Architecture vs Micro-architecture

Architecture is the contract between hardware and software. It describes the portable guarantees provided by the hardware, without dictating how they are implemented, and is typically specified in English. Arm has three architecture *profiles*: A, R and M (more hilarity). Approx. 5yr lag.

Micro-architecture relates to the design of a specific piece of hardware, and is typically specified in a hardware description language such as Verilog.

Software analogy: The C language standard is the architecture, whereas the toolchain source code (e.g. GCC + GLIBC) is the micro-architecture.

Arm develop both architectures and micro-architectures as licensable products!

Architecture vs Micro-architecture

Architecture is the contract between hardware and software. It describes the portable guarantees provided by the hardware, without dictating how they are implemented, and is typically specified in English. Arm has three architecture *profiles*: A, R and M (more hilarity). Approx. 5yr lag.

Micro-architecture relates to the design of a specific piece of hardware, and is typically specified in a hardware description language such as Verilog.

Software analogy: The C language standard is the architecture, whereas the toolchain source code (e.g. GCC + GLIBC) is the micro-architecture.

Arm develop both architectures and micro-architectures as licensable products!

Hurdle 2: This distinction is far less significant on other architectures such as x86.

Hysterical raisins

<=ARMv4	*When dinosaurs roamed the Earth*
ARMv5	2000, Baseline, Xscale
ARMv6	2002, SMP (sort of), ARM11
ARMv7-A	2005, Virtualisation, VMSA architecture, Cortex-A8
Armv8-A	2011, 64-bit ISA (AArch64), Cortex-A53

Armv8.1-A	2014, LSE atomics, Cavium ThunderX
Armv8.2-A	2015, RAS extensions, Cortex-A77
Armv8.3-A	2016, Pointer authentication, Vortex?
Armv8.4-A	2017, MPAM
Armv8.5-A	2018, MTE

Hysterical raisins

<=ARMv4	*When dinosaurs roamed the Earth*
ARMv5	2000, Baseline, Xscale
ARMv6	2002, SMP (sort of), ARM11
ARMv7-A	2005, Virtualisation, VMSA architecture, Cortex-A8
Armv8-A	2011, 64-bit ISA (AArch64), Cortex-A53

Armv8.1-A	2014, LSE atomics, Cavium ThunderX
Armv8.2-A	2015, RAS extensions, Cortex-A77
Armv8.3-A	2016, Pointer authentication, Vortex?
Armv8.4-A	2017, MPAM
Armv8.5-A	2018, MTE

Hurdle 3: Many features optional! Discoverable at runtime using ID registers.

Licensing

Why is the product naming so inconsistent? *Partly* due to the licensing model:

Core license to include a specific Arm CPU design in your SoC. Minor modification may be permitted.

Architecture license to design your own CPU from scratch which implements the Arm architecture and allow your marketing team to call it whatever they like

This means that Arm's architecture partners are also indirect competitors!

Licensing

Why is the product naming so inconsistent? *Partly* due to the licensing model:

Core license to include a specific Arm CPU design in your SoC. Minor modification may be permitted.

Architecture license to design your own CPU from scratch which implements the Arm architecture and allow your marketing team to call it whatever they like

This means that Arm's architecture partners are also indirect competitors!

Hurdle 4: Important that Arm doesn't produce production silicon as an SoC

The Arm ecosystem

Playing ‘Switzerland’

Arm’s influence is in the power of the partnership and underlying platform; on its own, it’s relatively tiny in revenue terms:

Qualcomm	22B USD (2018)
Intel	70B USD (2018)
Samsung	211B USD (2017)

Playing ‘Switzerland’

Arm’s influence is in the power of the partnership and underlying platform; on its own, it’s relatively tiny in revenue terms:

Qualcomm	22B USD (2018)
Intel	70B USD (2018)
Samsung	211B USD (2017)
Arm	<1B GBP (2015)

Greater than the sum of its parts, but driven by consensus.

Playing ‘Switzerland’

Arm’s influence is in the power of the partnership and underlying platform; on its own, it’s relatively tiny in revenue terms:

Qualcomm	22B USD (2018)
Intel	70B USD (2018)
Samsung	211B USD (2017)
Arm	<1B GBP (2015)

Greater than the sum of its parts, but driven by consensus.

Hurdle 5: Limited ability to mandate standards compliance or move goalposts

Diversity vs Fragmentation

‘Pick ‘n’ mix’ architecture reigned in slightly by desire for software re-use:

PSCI	to bring CPUs up and down
SBSA	certification for server system designs
Trusted firmware	open-source boiler-plate
Linux/KVM	upstream-first approach

Diversity vs Fragmentation

‘Pick ‘n’ mix’ architecture reigned in slightly by desire for software re-use:

PSCI	to bring CPUs up and down
SBSA	certification for server system designs
Trusted firmware	open-source boiler-plate
Linux/KVM	upstream-first approach

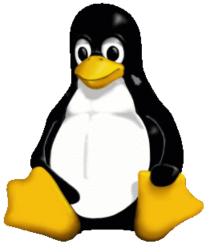
Hurdle 6: Still a tendancy towards fragmentation between SoCs

`https://www.mono-project.com/news/2016/09/12/arm64-icache/`

How does Linux cope?

The `arm64` Linux kernel is written to the *architecture* and aims to abstract away the underlying bag of bits baked into silicon:

- Single `defconfig` binary image for all AArch64 CPUs
- Heavy use of ‘alternative’ instruction patching
- uarch-specific features not worth the effort…
- …apart from errata workarounds and side-channels
- Heterogeneous systems are a headache
- Exposed to userspace via `proc`, `sys`, `ELF HWCAP` and `MRS` instructions



Details in `Documentation/arm64/cpu-feature-registers.rst`

Email the list if you need more: `linux-arm-kernel@lists.infradead.org`

arm

Fire up the simulation

Where do I start? / What’s in the box?

1. Ignore all the marketing junk
2. Identify the SoC
3. Identify the CPU(s) – Arm derived?
4. Identify architecture version
5. Identify features
6. `https://wikichip.org`

Some example devices

Example 1: Libre Computer ‘La Frite’

They say…

AML-S805X-AC is the perfect development platform for projects that require highly performant ARM Cortex-A class CPUs, small compact form factor, secure and non-secure 1080P media delivery and playback, high reliability, and low power.

Example 1: Libre Computer ‘La Frite’

They say…

AML-S805X-AC is the perfect development platform for projects that require highly performant ARM Cortex-A class CPUs, small compact form factor, secure and non-secure 1080P media delivery and playback, high reliability, and low power.

- Soc is Amlogic S805x
- 4x ARM Cortex-A53 @ 1.2GHz
- Cortex-A53 is a 64-bit core implementing ARMv8.0
- Usually the ‘Little’ in early ‘big.Little’ designs

ARM says…

The most widely-used processor with balanced performance and efficiency

Example 2: Samsung Galaxy S9

They say…

10nm 64-bit Octa-Core Processor 2.8GHz + 1.7GHz (Maximum Clock Speed, Performance Core + Efficiency Core)

Example 2: Samsung Galaxy S9

They say…

10nm 64-bit Octa-Core Processor 2.8GHz + 1.7GHz (Maximum Clock Speed, Performance Core + Efficiency Core)

- SoC appears to be Samsung Exynos 9810 (in Europe)
- Wikichip claims this features 4xMongoose 3 plus 4xCortex-A55 cores
- Cortex-A55 designed by Arm and implements ARMv8.2
- Mongoose 3 designed by Samsung and implements ARMv8.0

Example 2: Samsung Galaxy S9

They say…

10nm 64-bit Octa-Core Processor 2.8GHz + 1.7GHz (Maximum Clock Speed, Performance Core + Efficiency Core)

- SoC appears to be Samsung Exynos 9810 (in Europe)
- Wikichip claims this features 4xMongoose 3 plus 4xCortex-A55 cores
- Cortex-A55 designed by Arm and implements ARMv8.2
- Mongoose 3 designed by Samsung and implements ARMv8.0

The Cortex-A55 implements more instructions than the Mongoose 3!

`https://medium.com/@jadr2ddude/a-big-little-problem-a-tale-of-big-little-gone-wrong-e7778ce744bb`

Example 3: Motorola moto z4

They say…

Feel the speed of a Qualcomm® Snapdragon™ 675 processor that's 57% faster than moto z3 play. It's a phone built to keep up with you.

Example 3: Motorola moto z4

They say…

Feel the speed of a Qualcomm® Snapdragon™ 675 processor that's 57% faster than moto z3 play. It's a phone built to keep up with you.

It's an SoC not a bloody processor!

Qualcomm say…

CPU Cores: Qualcomm® Kryo™ 460 CPU, Octa-core CPU…built on Arm Cortex technology…

Example 3: Motorola moto z4

They say…

Feel the speed of a Qualcomm® Snapdragon™ 675 processor that's 57% faster than moto z3 play. It's a phone built to keep up with you.

It's an SoC not a bloody processor!

Qualcomm say…

CPU Cores: Qualcomm® Kryo™ 460 CPU, Octa-core CPU…built on Arm Cortex technology…

- Wikichip to the rescue!
- Kryo 460 Silver => Cortex-A55
- Kryo 460 Gold => Cortex-A76
- Both implement ARMv8.2

'The two big disclosed changes are an increase of the core's reorder buffer from 128 entries to a higher, unspecified amount, as well as tuning the prefetchers to better work with floating point workloads.' https://www.anandtech.com/show/14072/the-samsung-galaxy-s10plus-review/3

arm

Questions?

Please contact `support@arm.com`!

Image references

1. Colin Davis, 'Veins in my Right Arm'. Cropped from `https://live.staticflickr.com/3382/3554305017_053fee6793_b_d.jpg` - CC BY 2.0 (`https://creativecommons.org/licenses/by/2.0/`)
2. Marco Verch, 'Bauarbeiterdekoltee'. Unmodified from `https://live.staticflickr.com/4357/36221532814_779c61e968_k_d.jpg` - CC BY 2.0 (`https://creativecommons.org/licenses/by/2.0/`)
3. DJRphoto36, 'Angry Penguin'. Unmodified from `https://live.staticflickr.com/5056/5496520090_e9aa83dc0b_b_d.jpg` - CC BY 2.0 (`https://creativecommons.org/licenses/by/2.0/`)
4. Kevin Gill, 'Mars'. Unmodified from `https://live.staticflickr.com/2107/32789125872_f0184cc247_k_d.jpg` - CC BY 2.0 (`https://creativecommons.org/licenses/by/2.0/`)
5. Pablo Cabezos, 'King College´s chapel (Cambridge)'. Unmodified from `https://live.staticflickr.com/8300/29112354294_3c0f166ec1_b_d.jpg` - CC BY 2.0 (`https://creativecommons.org/licenses/by/2.0/`)
6. kecia85, 'Dictionary-007'. Unmodified from `https://live.staticflickr.com/8322/7983928912_f18c81fb31_o_d.jpg` - CC BY-SA 2.0 (`https://creativecommons.org/licenses/by-sa/2.0/`)
7. Images Money, 'Money'. Unmodified from `https://live.staticflickr.com/6029/5929622407_e474db378d_k_d.jpg` - CC BY 2.0 (`https://creativecommons.org/licenses/by/2.0/`)
8. wisegie, 'Swiss flag'. Unmodified from `https://live.staticflickr.com/6082/6078034891_ca23bb39a8_b_d.jpg` - CC BY 2.0 (`https://creativecommons.org/licenses/by/2.0/`)
9. The DLC, 'Buffet line'. Cropped from `https://live.staticflickr.com/1369/4734507240_57b0244541_h_d.jpg` - CC BY-SA 2.0 (`https://creativecommons.org/licenses/by-sa/2.0/`)
10. `lewing@isc.tamu.edu` Larry Ewing and The GIMP, 'Tux penguin'. Unmodified from `https://commons.wikimedia.org/wiki/File:Tux.png`
11. Curtis Alan Jackson, 'After this, I give up. Probably.'. Unmodified from `https://live.staticflickr.com/5097/5418579238_d28022927c_h_d.jpg` - CC BY 2.0 (`https://creativecommons.org/licenses/by/2.0/`)