QEMU for System Software Development

a gentle introduction

Scott Tsai

Why Use QEMU and Target ARMv7-A?

Why Use QEMU for Development?

- There’ll always be demand for a simulator that can run the latest Linux kernel and work with the latest compilers & ABI
- No, “I’m stuck on an old kernel due to my simulator being old” or ...
- “The C library now depends on this new instruction that my simulator doesn't support” problems

Why Use QEMU for Development?

- QEMU kept up with hardware advancement for the last 10+ years
  - It was released in **2003**!

[Qemu-devel] Welcome to qemu-devel (and test)

From: Rusty Russell
Subject: [Qemu-devel] Welcome to qemu-devel (and test)
Date: Fri, 11 Apr 2003 14:37:50 +1000

```
Hi.

        I voluteered to do the Savannah legwork for Fabrice.
Hopefully this list will speed development.

Cheers,
Rusty.
--
  Anyone who quotes me in their sig is an idiot. -- Rusty Russell.
```

Why Use QEMU for Development?

- QEMU kept up with hardware advancement for the last 10+ years
  - It was released in **2003** and still going strong
- In 5 years time, the most popular architecture might be different
  - RISC-V? 
  - ARMv(N+1)?
- … but QEMU will support it, and you'll be able to use the latest kernel and compiler toolchain to develop against it
- Most academic simulators stop advancing once the developers move on
- QEMU has enough developers & companies contributing to form critical mass

Why target ARMv7-A in this tutorial?

- I put up a poll in our FB group

Scott Tsai created a poll.
September 23 at 9:31am

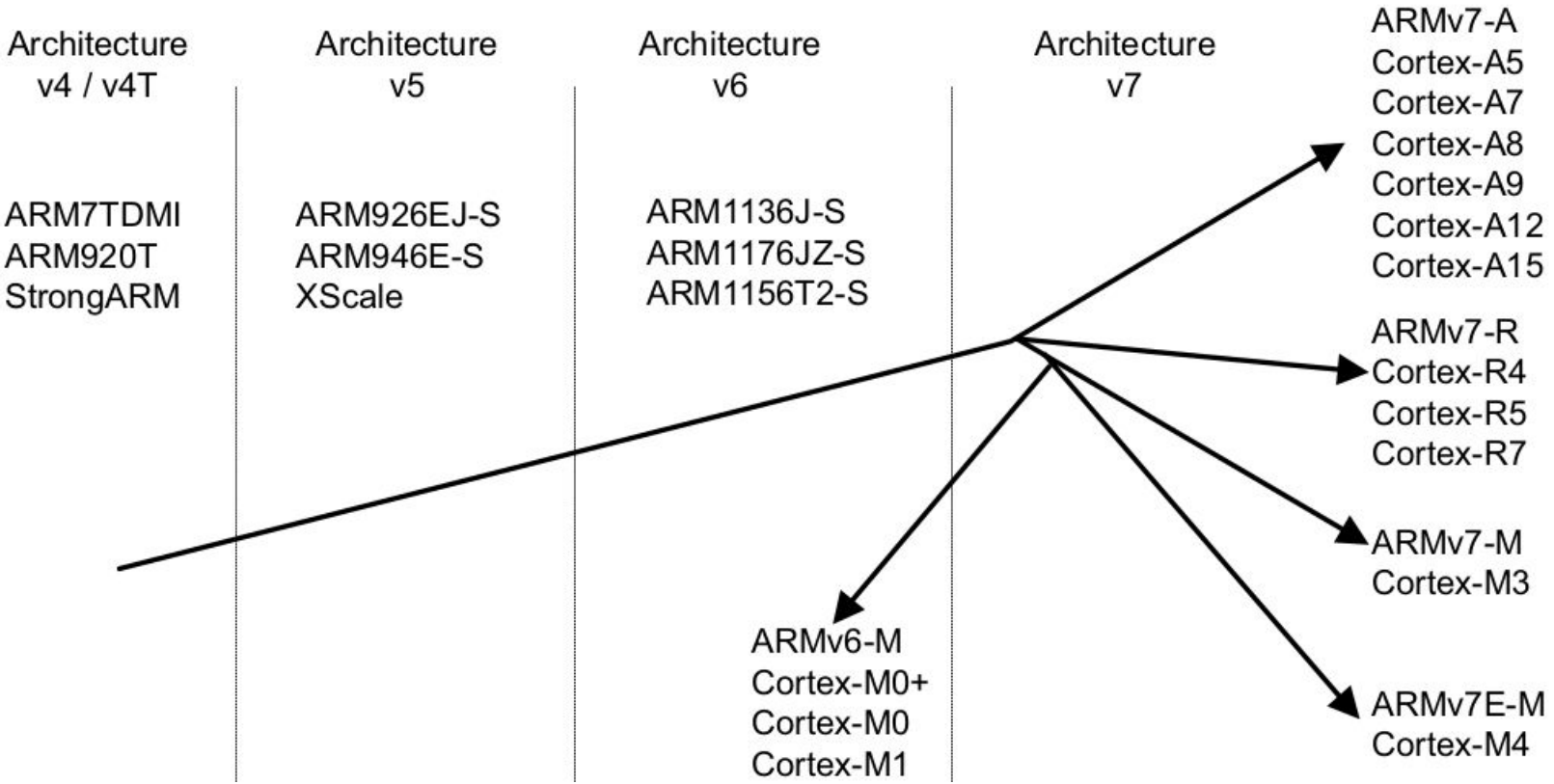
想問一下大家對 CPU 架構組語的偏好、熟悉程度。尤其是有參加《Linux环境编程》讀書會的朋友，因為之後幾個主題：
ELF 動態連結
AddressSanitizer
setjmp()/longjmp()
Unix signal 與 sigsetjmp()/siglongjmp()
選定一種 CPU 架構，具體看組語討論會比較清楚。
選項我只列現在還記得的 😉

Why target ARMv7-A in this tutorial?

- I put up a poll in our FB group a month ago
- ARMv7-A was the clear favorite

- Frankly, I was expecting x86-64 to be most popular
- Results perhaps reflects embedded focus of Taiwanese tech industry?

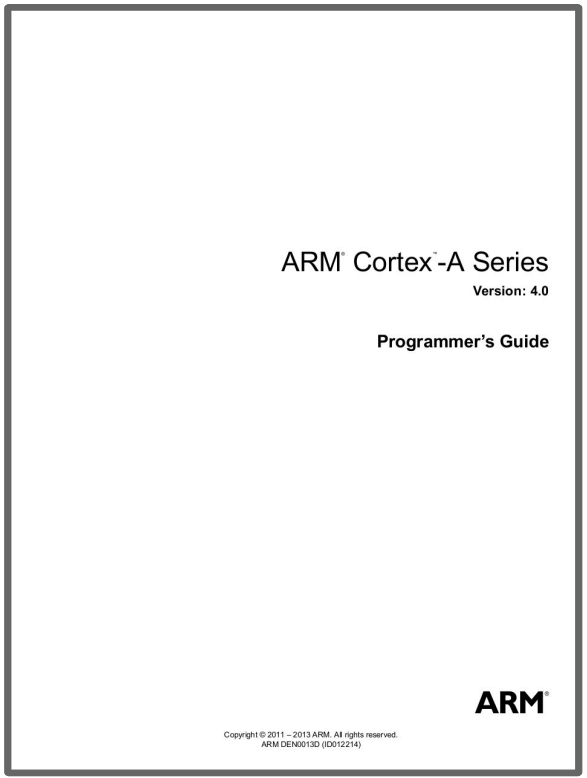
ARM is a good e.g. of major arch changes over time



ARMv5→v6: memory system changed significantly

- The TLB gained Address Space IDs (ASID)
- Caches VIVT⟶VIPT
- ARMv5 “subpage” support marked deprecated
  - … then removed in ARMv7
  - See: ARMv6 page table translation subpage AP bits disabled
- TLB & cache management during context switch greatly affected
- See: AN425 Migrating a software application from ARMv5 to ARMv7-A/R
  - and The ARM Architecture Version 6 White Paper
  - For the details.

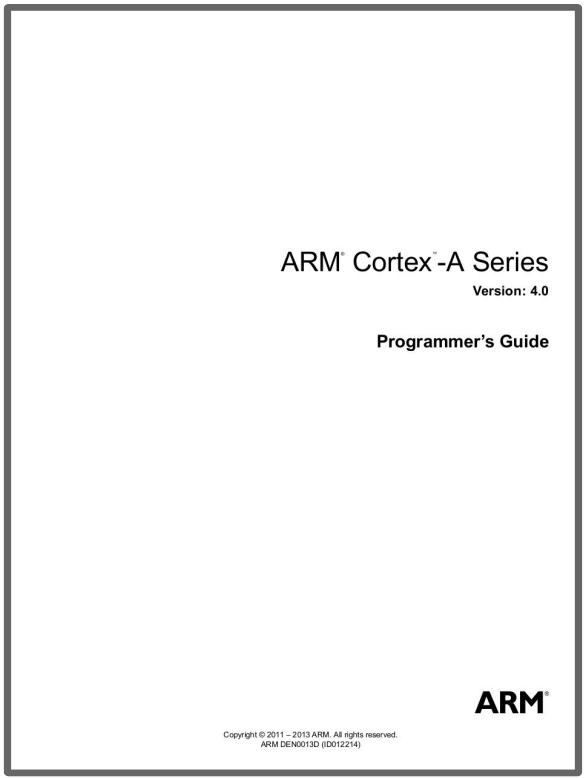
For ARMv7-A, you want the Programmer's Guide



[Cortex-A Series Programmer's Guide]

Freely downloadable after registration.

For ARMv7-A, you want the Programmer's Guide



Cortex-A Series Programmer's Guide

- Good diagrams
- Clear exposition
- Helpful references to Linux kernel code
- Not enough code snippets

Pro: doesn't bother with the pre v7 stuff.

Docs for ARMv7-A System Software Development

- Cortex-A Series Programmer's Guide for ARMv7-A
- ARMv7 Architecture Reference Manual
- ARM Generic Interrupt Controller Version 2.0
- Cortex-A15 Technical Reference Manual

  - The trick is to ignore the ARMv4 stuff you find on the net
  - Also, most SOC families have different interrupt controllers and timers
    - E.g. Raspberry Pi 1 vs. Raspberry Pi 2 vs. ARM Vexpress