Embedded FreeBSD on a five-core RISC-V processor using LLVM
How hard can it be?

Jeremy Bennett
Target Hardware

- RV64IMAC
- Memory
- RV64IMAC
- RV64IMAC
- RV32IMAC
- Memory
Pre-prototype: QEMU
FreeBSD

- Build user land
  - create RAM disc
- Build kernel
  - supply RAM disc details
- Build BBL wrapper
  - adds device tree
- Result: image to load
Compiler Tool Chain

- Clang/LLVM
- GNU binutils
- GNU Debugger
  - including remote debugger
- C libraries from FreeBSD
Incremental Project Stages

Repeat for 32-bit

64-bit
What Could Possibly Go Wrong?

• FreeBSD is already ported to RV64
  - thank you to Ruslan Bukin

• Clang/LLVM is FreeBSD default tool chain
  - thanks to all the RISC-V LLVM team
What Could Possibly Go Wrong? (2)

- FreeBSD build documentation (RISC-V FreeBSD Wiki)
  - recommends HEAD, which often breaks
  - tries to validate users against build machine!
  - tries to use build machine user database to build file image
- With this corrected builds and runs multicore on QEMU
  - these were trivial glitches quickly fixed
What Could Possibly Go Wrong? (3)

- Would not boot on HiFive Unleashed
  - CPU 0 is E51, which is masked out
  - modify FreeBSD, to not assume running on CPU 0
- With this fixed, boots single core
What Could Possibly Go Wrong? (4)

• FreeBSD only works single core on HiFive Unleashed
  - multicore boot cause kernel panic
    • most of the time!
  - seems to be due to random ordering in which CPUs come up
    • needs an abstraction layer to map CPUs to consistent order.
    • connected to enabling of interrupts

• Work in progress...
What Could Possibly Go Wrong? (5)

• Clang/LLVM for 64-bit RISC-V is brand new
  - first patches published October 2018
• 7 patches needed to get FreeBSD to build
  - PC-relative addressing
  - PIC addressing
  - thread local storage (TLS) support
  - 9 more patches to fix bugs, 1 (?) bug left
What Could Possibly Go Wrong? (6)

- Still would not run
- CompilerRT was locking up in floating point emulation
  - float->int uses hardware floating point opcodes!
- With this fixed FreeBSD runs when built with LLVM
  - but tends to lock up after around 10 minutes
  - kernel issue: stable if just kernel is built with GCC
    - LLVM compiler bug
What Could Possibly Go Wrong? (7)

• The FreeBSD test system is Kyua
  - has to be built native – no cross-compile
  - therefore needed to build a native GCC
  - the QEMU emulation of RISC-V is slow...

• We now have Kyua running
  - nightly testing is possible
  - around 5½ hours on 20 core Xeon server
What Could Possibly Go Wrong? (8)

• There is no longer a FreeBSD gdbserver
  – currently recreating this
  – trying to reuse as much of the native GDB code
    • generic improvement to GDB
    – still several more weeks to go

• For now we have native GDB
  – but relies on a lot of unneeded system infrastructure
  – stripped down embedded system will need gdbserver
Current Status: FreeBSD for 64-bit RISC-V

- We can run embedded FreeBSD for RV64 built with LLVM
  - reference implementation for HiFive Unleashed
- Work in progress with some limitations
  - single core only (HiFive specific issue)
  - unstable when built with LLVM (compiler bug)
- Reference implementation will be available shortly
  - just needs to be organized & documented
FreeBSD for 32-bit RISC-V

• Issues to address
  - FreeBSD not yet ported to RV32
  - building with GCC doesn’t work
    • CompilerRT library needs \texttt{int128} type
  - we don't have a suitable RV32 hardware platform
    • RV32IMAC + MMU

• Watch this space...