FreeBSD/RISC-V progress

Ruslan Bukin

University of Cambridge Computer Laboratory

August 15, 2016

Approved for public release; distribution is unlimited. This research is sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL), under contract FA8750-10-C-0237. The views, opinions, and/or findings contained in this article/presentation are those of the author(s)/presenter(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

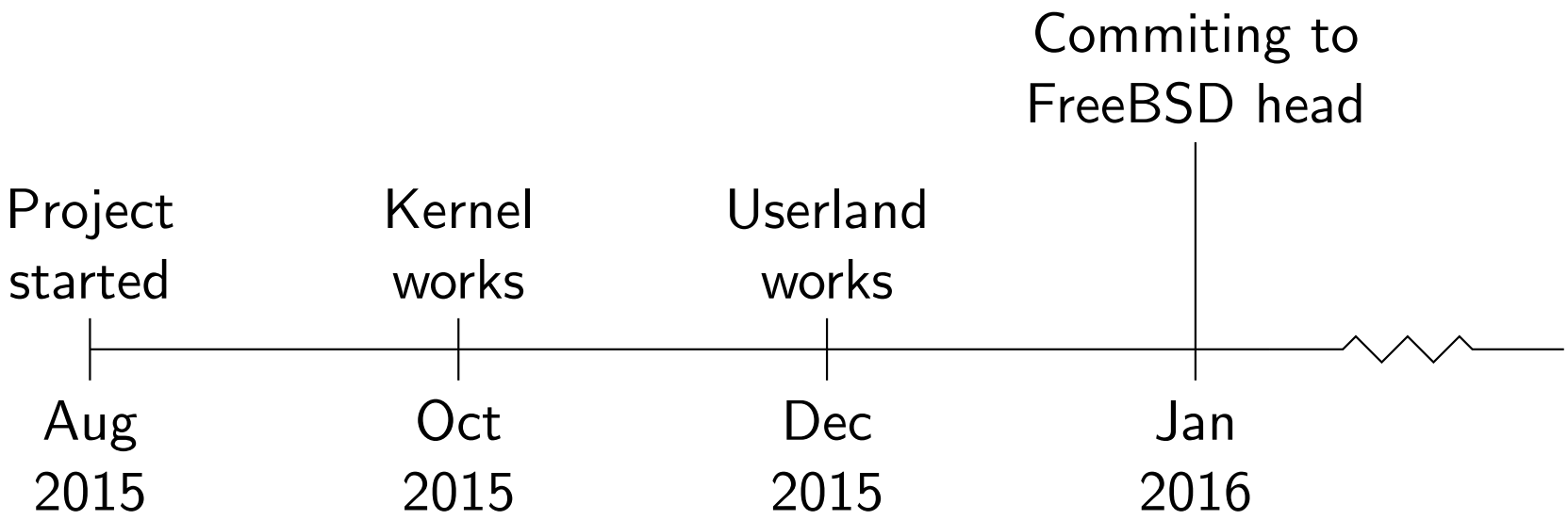
RISC-V v1.9 privileged ISA

- RV32 (32 bit, 32 registers)
- RV64 (64 bit, 32 registers)
- RV128 (128 bit, 32 registers)
- RV32E (embedded, x0-x15 registers)
- RV64G - General extensions, 'G' is combination IMAFD
- extensions
  - "A" atomic
  - "C" compressed, 2-byte instruction size
  - "E" embedded
  - "F" Single precision FPU
  - "M" integer multiplication
  - ...

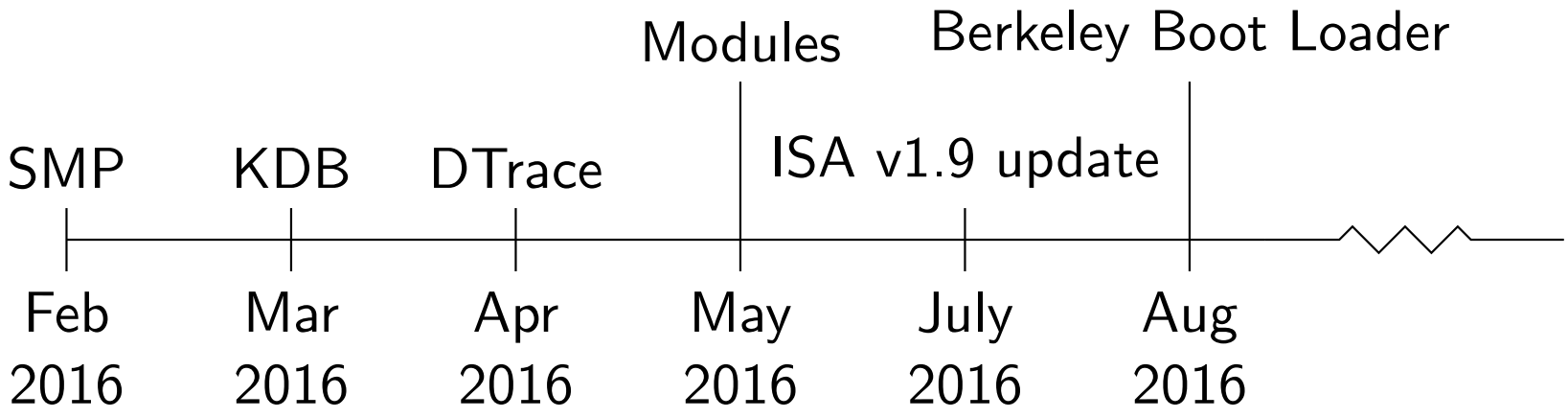
Some reasons to use FreeBSD/RISC-V

- Full stack BSD license (RISC-V, FreeBSD, LLVM/Clang)
- Technology transition
- Growing community, 42 Universities, 63 companies attended latest RISC-V workshop

FreeBSD/RISC-V: first 6 months



FreeBSD/RISC-V: next 6 months



Challenges

- Single page table base register
- Single tp (thread pointer) register
- Single sscratch register
- Single exceptions entry

FreeBSD/RISC-V: single page table base register problem

- Install kernel pagetables to each user pmap
- Update/delete on kernel pmap requires to update all the user pmaps

FreeBSD/RISC-V: single sscratch and tp registers

- Store PCPU on supervisor stack first
- Then store supervisor stack to sscratch register

FreeBSD/RISC-V: Exceptions entry problem

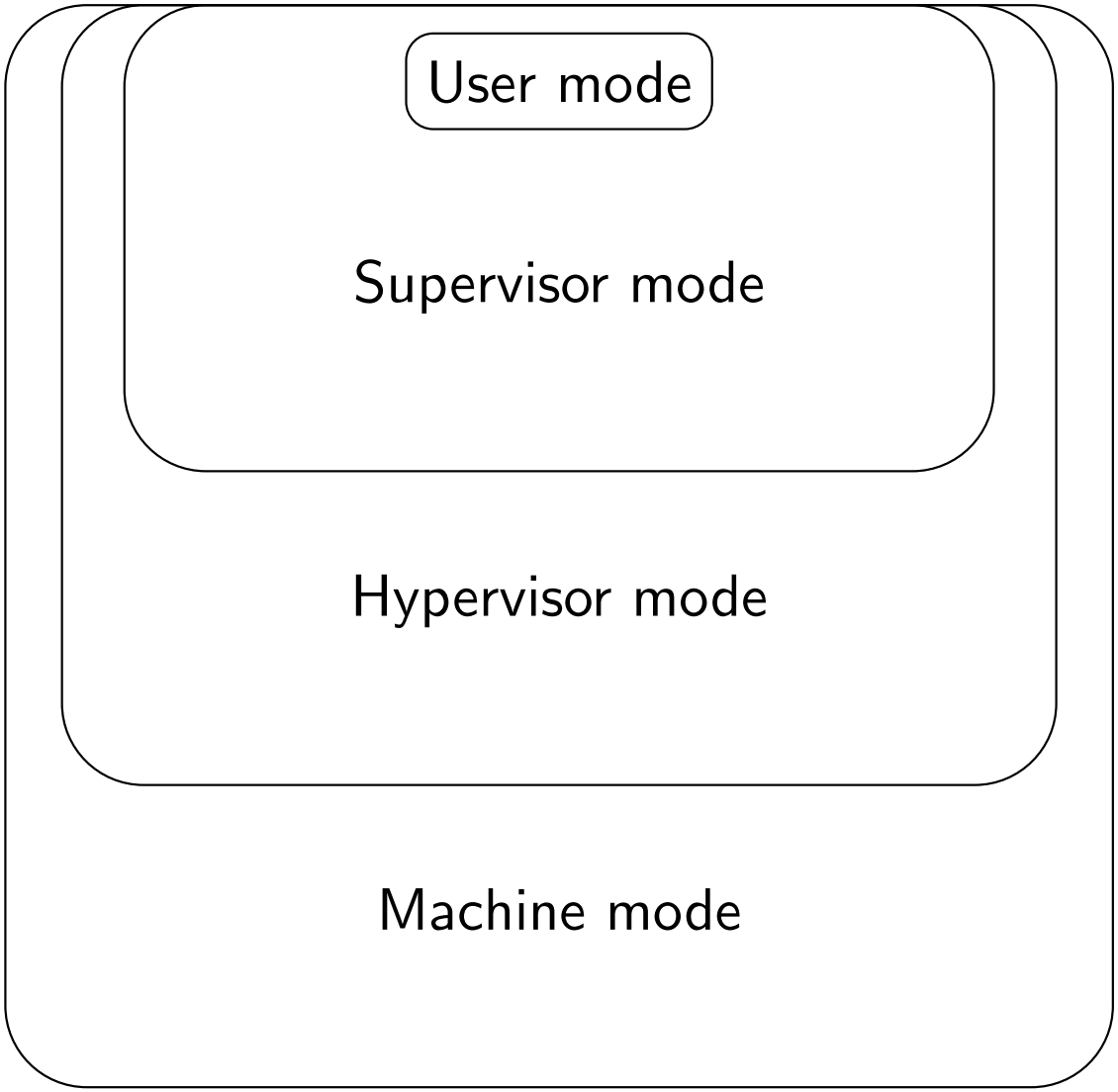
```
ENTRY(cpu_exception_supervisor)
        [..]
        call    _C_LABEL(do_trap_supervisor)
        [..]
END(cpu_exception_supervisor)

ENTRY(cpu_exception_user)
        [..]
        call    _C_LABEL(do_trap_user)
        [..]
END(cpu_exception_user)
```

FreeBSD/RISC-V: Exceptions entry solution

```
ENTRY(cpu_exception_handler)
        csrrw   sp, sscratch, sp
        beqz    sp, 1f
        /* User mode detected */
        csrrw   sp, sscratch, sp
        j       cpu_exception_handler_user
1:
        /* Supervisor mode detected */
        csrrw   sp, sscratch, sp
        j       cpu_exception_handler_supervisor
END(cpu_exception_handler)
```

Privilege levels



FreeBSD/RISC-V: Berkeley Boot Loader

- Original firmware from RISC-V project

FreeBSD/RISC-V: facts

- Based on ARMv8 port
- Patch to head 25k lines (200 new files)
- External GNU toolchain used (GCC 6.1)
- 6 months from scratch

Thanks

People involved (Thanks!)

- Robert Watson (University of Cambridge)
- David Chisnall (University of Cambridge)
- Andrew Turner (ABT Systems)
- Arun Thomas (BAE Systems)
- Ed Maste (The FreeBSD Foundation)
- Yukishige Shibata (Sony Japan)

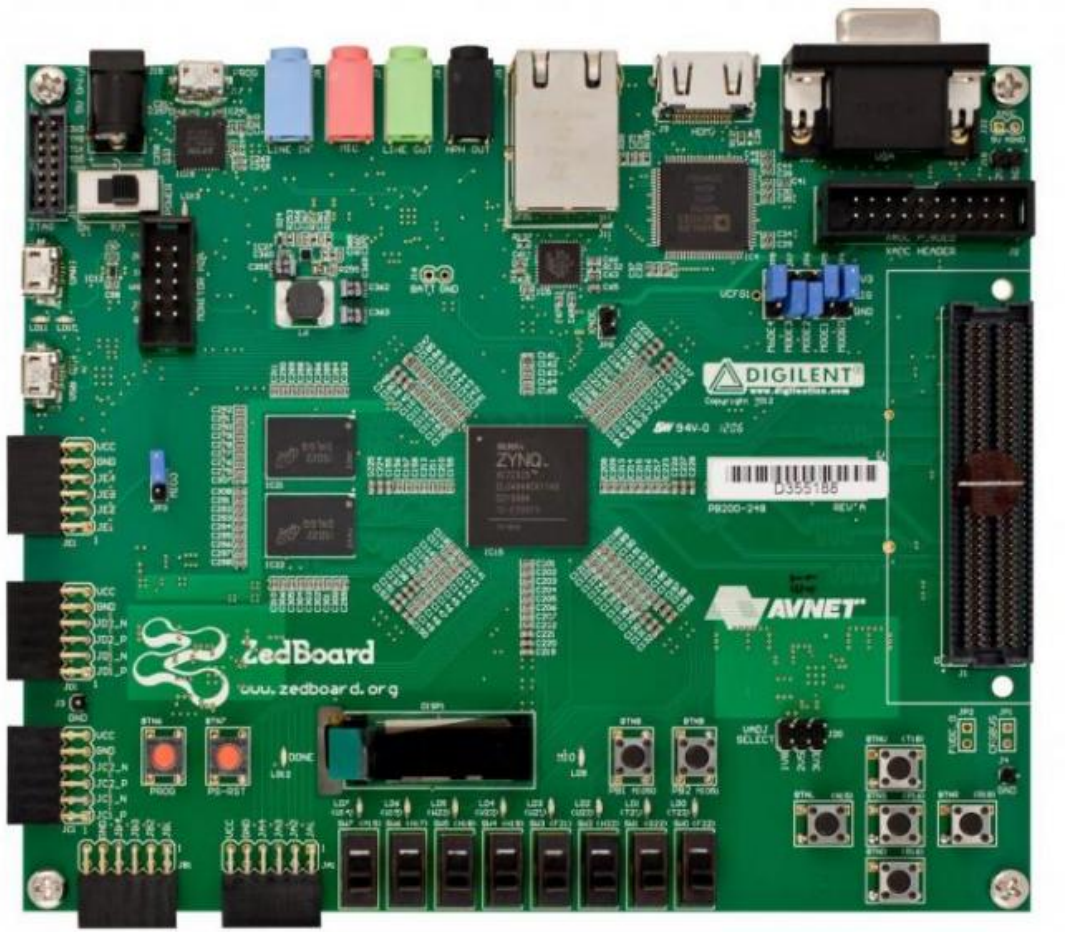
Tools

Table: Simulator/Emulator supported

	Wrotten in	FreeBSD support
Spike	C++	Yes
QEMU	C	Yes, v1.7 ISA
RocketCore FPGA	Chisel	Yes
lowRISC FPGA	Chisel	Not yet

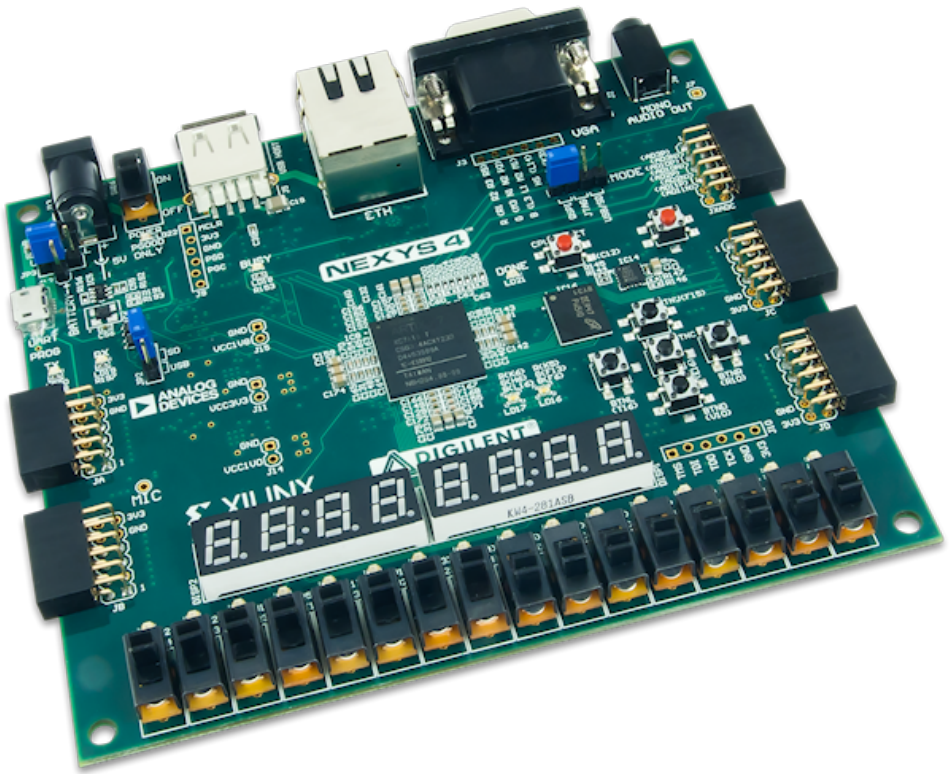
RocketChip

- 6-stage single-issue in-order pipeline
- Chisel $\rightarrow$ verilog $\rightarrow$ bitfile
- Synthesized on ZedBoard
- also SiFive Freedom U500 Platform



lowRISC

- RocketChip fork
- University of Cambridge
- Chisel $\rightarrow$ verilog $\rightarrow$ bitfile
- Synthesized on Nexys4



FreeBSD/RISC-V: Current port status

- Everything works (tm)

TODO

- Ports/packages for toolchain, for tools, for BBL
- QEMU integration for ports (usermode QEMU)
- FPU (Floating Point Unit)
- LLVM/Clang support
- FDT/non-FDT GENERIC kernel

Questions

Project home:
https://wiki.freebsd.org/riscv