



UNIVERSITY OF  
CAMBRIDGE

Computer Laboratory

# FreeBSD/RISC-V and Device Drivers

Ruslan Bukin

University of Cambridge Computer Laboratory

BCS London 2017

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.

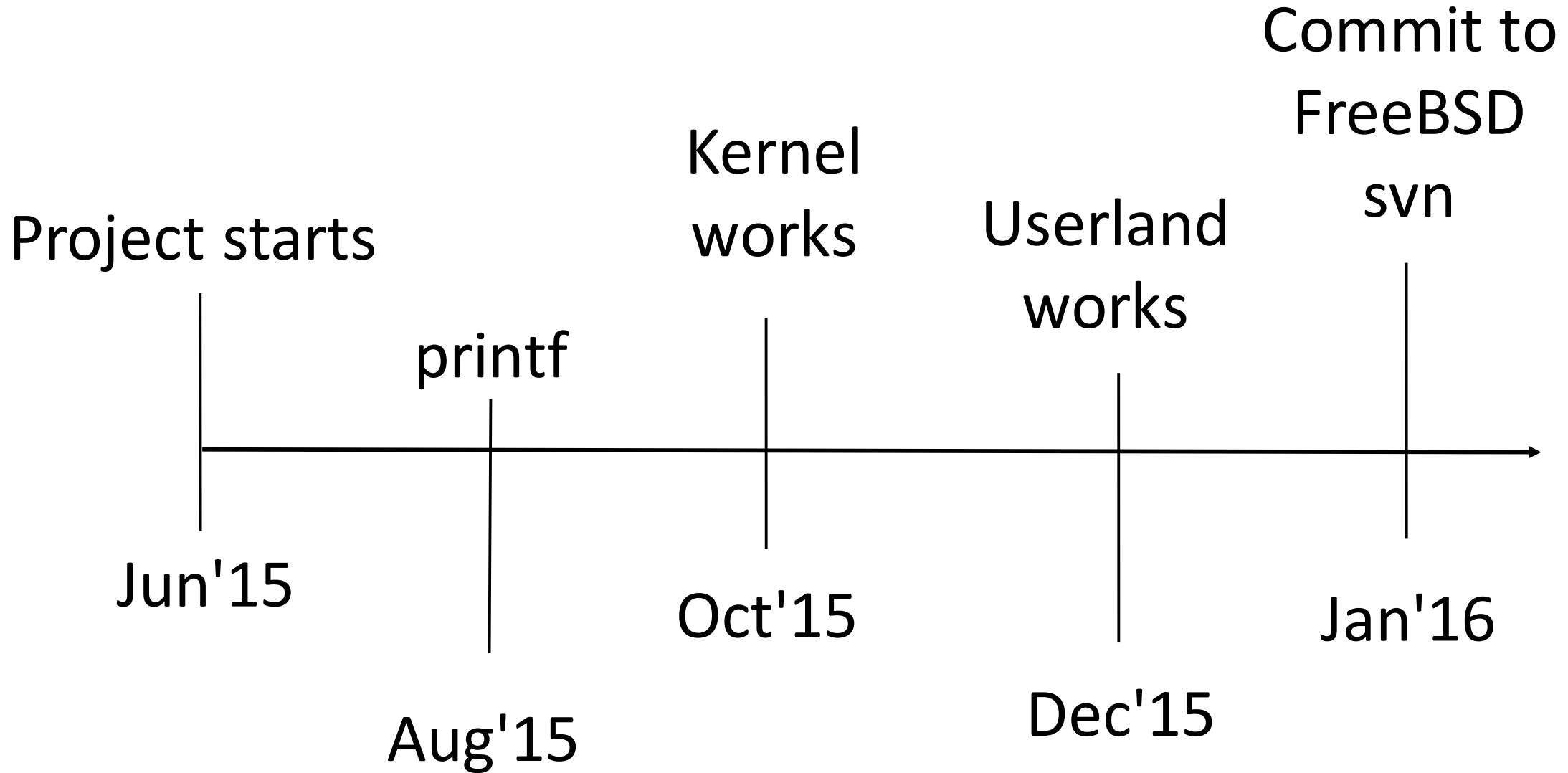
# About the presenter

- FreeBSD architectural ports (ARM, MIPS, RISC-V, x86)
- Device Drivers (Sound, DMA, Ethernet, MMC, ...)
- DTrace support (ARMv8, RISC-V)
- HWPMC (ARMv7, ARMv8)
- Intel® Software Guard Extensions (SGX)
- Intel® Processor Tracing Technology (under review)

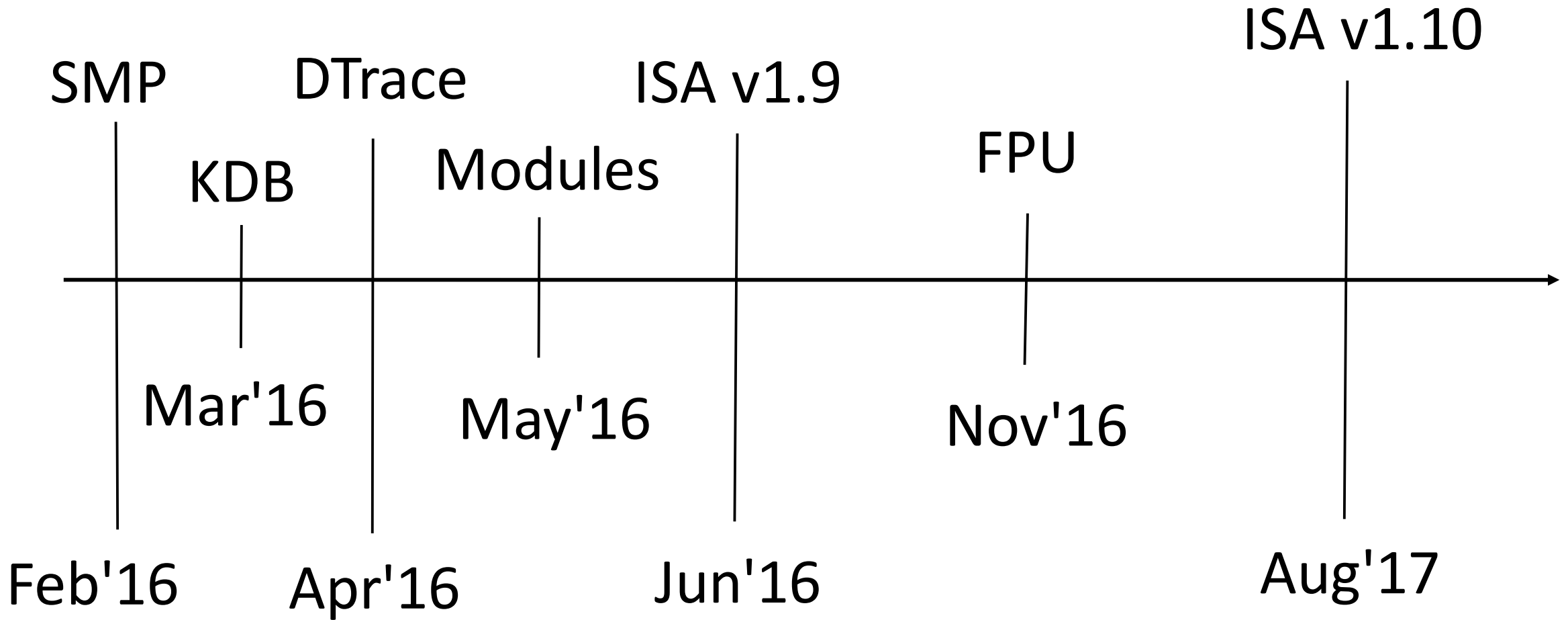
# Overview

- History
- Porting process
- Challenges experienced
- Status updates on v1.10 privilege spec
- Drivers support

# Timeline



# Timeline (cont'd)



# RISC-V privilege levels

❑ Machine (M-mode) – Berkeley BootLoader (BBL)

❑ Hypervisor (H-mode)

❑ Supervisor (S-mode) – FreeBSD OS kernel

❑ User (U-mode) – /sbin/init, libc and userspace

- Supported combination of modes:

- M                      simple embedded

- M, U                  simple embedded with protection

- M, S, U              UNIX like OS (FreeBSD, Linux)

# Berkeley Boot Loader (BBL)

- Firmware/bootloader
- Operates in M-mode
- Provides access to console, timers, IPI
- All traps switch mode to M
  - However we can delegate some traps directly to S-mode using mtdeleg instruction

# Start of work

- Toolchain:
  - GNU toolchain (GCC, binutils, as, ...)
  - LLVM (no support yet)

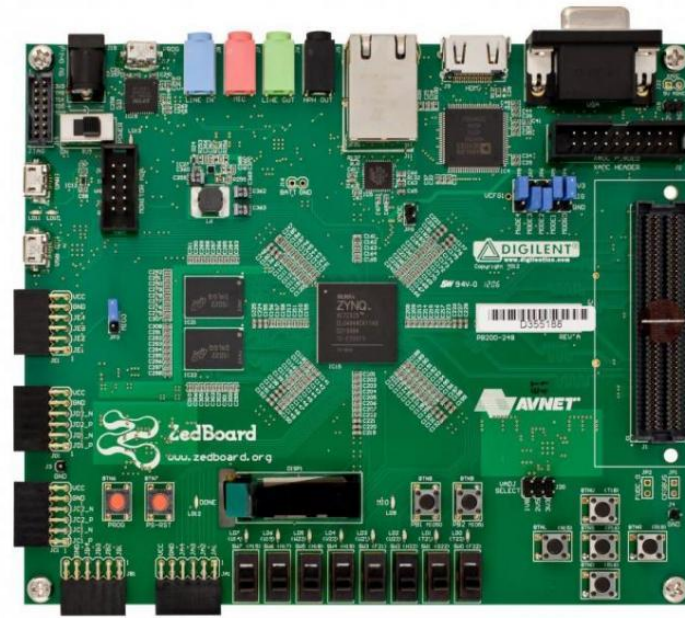


# Template

```
$ cp -R sys/arm64 sys/riscv
$ do {
$   vi sys/riscv/...
$   make TARGET_ARCH=riscv64
$ } while ($?)
```

# Prepare hardware

- Emulators:
  - Spike
  - QEMU
- FPGA
- root disk
  - ROOTFS embedded to kernel
  - kernel is embedded to BBL



# Porting: early assembly code

- `_start` in `locore.S`
- Put machine into known state
- Build initial page tables
- Enable MMU and branch to virtual addressing
- Setup stack pointer
- Jump to C-code

# Porting: kernel (1)

- Console
- Atomics
  - `atomic_add()`, `atomic_readandclear()`, ...
- Initialize physmap table
- PMAP implementation

# PMAP

- 40 functions to implement:
  - pmap\_enter()
  - pmap\_extract()
  - pmap\_remove()
  - pmap\_invalidate()
  - pmap\_protect()
  - pmap\_activate()
  - pmap\_unwire()
  - ...

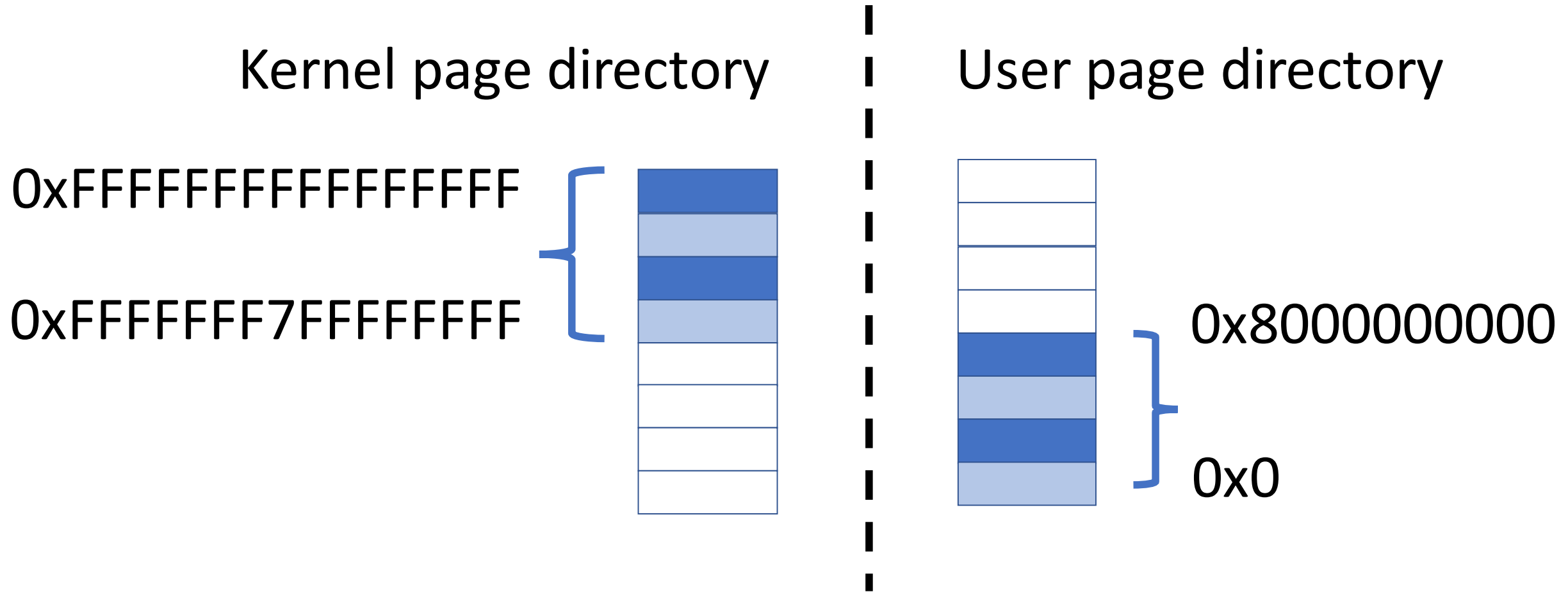
# Porting: kernel (2)

- Exceptions
- `cpu_switch()`
- `fork_trampoline()`
- Kernel VA  $\leftrightarrow$  user VA copy functions
  - `copyin`
  - `copyout`
  - ...

# Porting: kernel (3)

- Timer driver
- Interrupt controller driver
- Disk driver
- Done

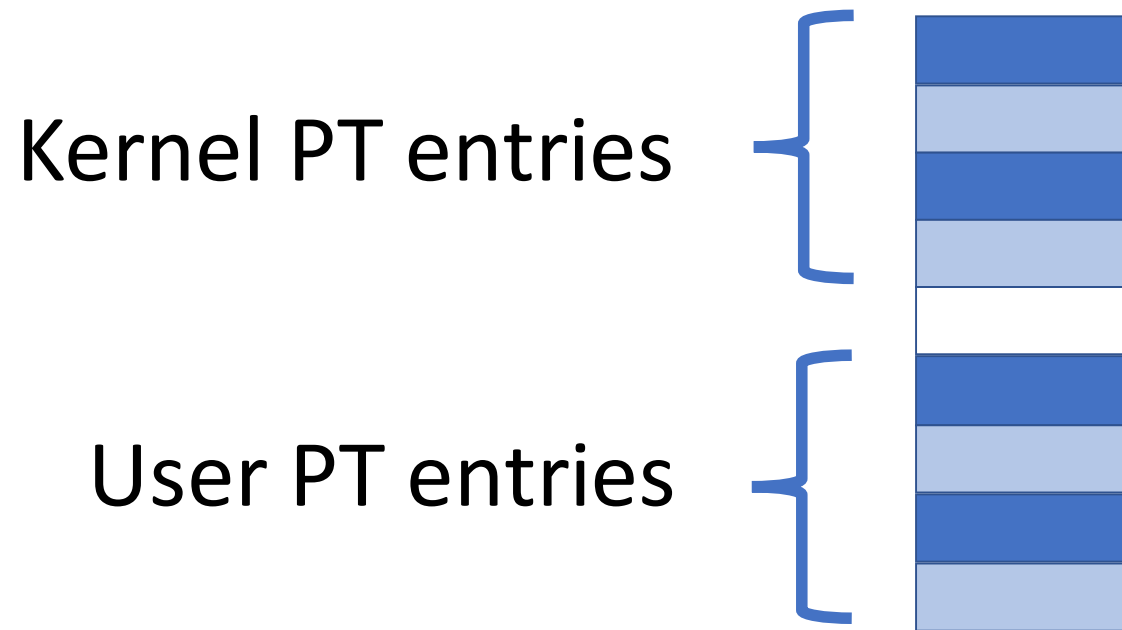
# Challenge #1: page table base





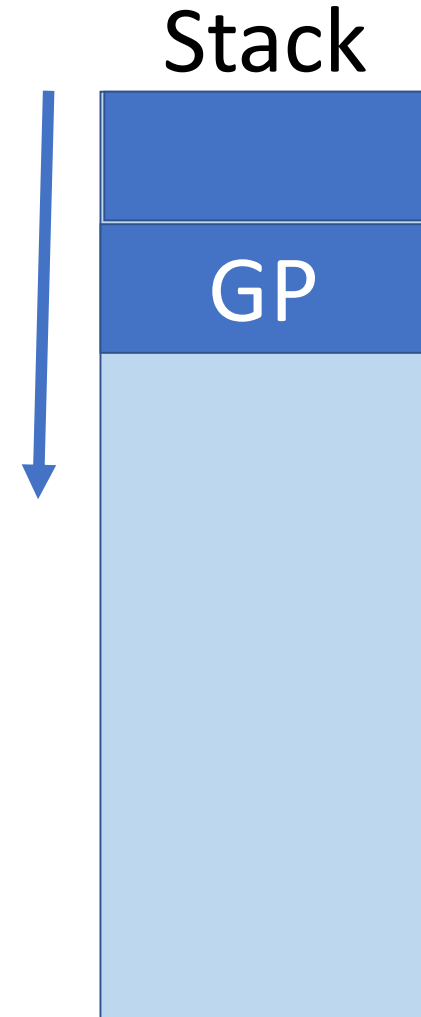
# Challenge #1: page table base (cont'd)

User page directory



# Challenge #2: single TP register

- PCPU pointer
- Solution:
  - Store GP to supervisor stack
  - Reload GP on return from user thread



# Porting: userspace (1)

- jemalloc
- csu
  - crt1.S, crtN.S, crti.S
- libc
  - syscalls
  - setjmp, longjmp, \_set\_tp
- msun

# Porting: userspace (2)

- Compile world
- Try to run `/bin/sh`
- `rtld` (run-time linker)

# Porting: finalize

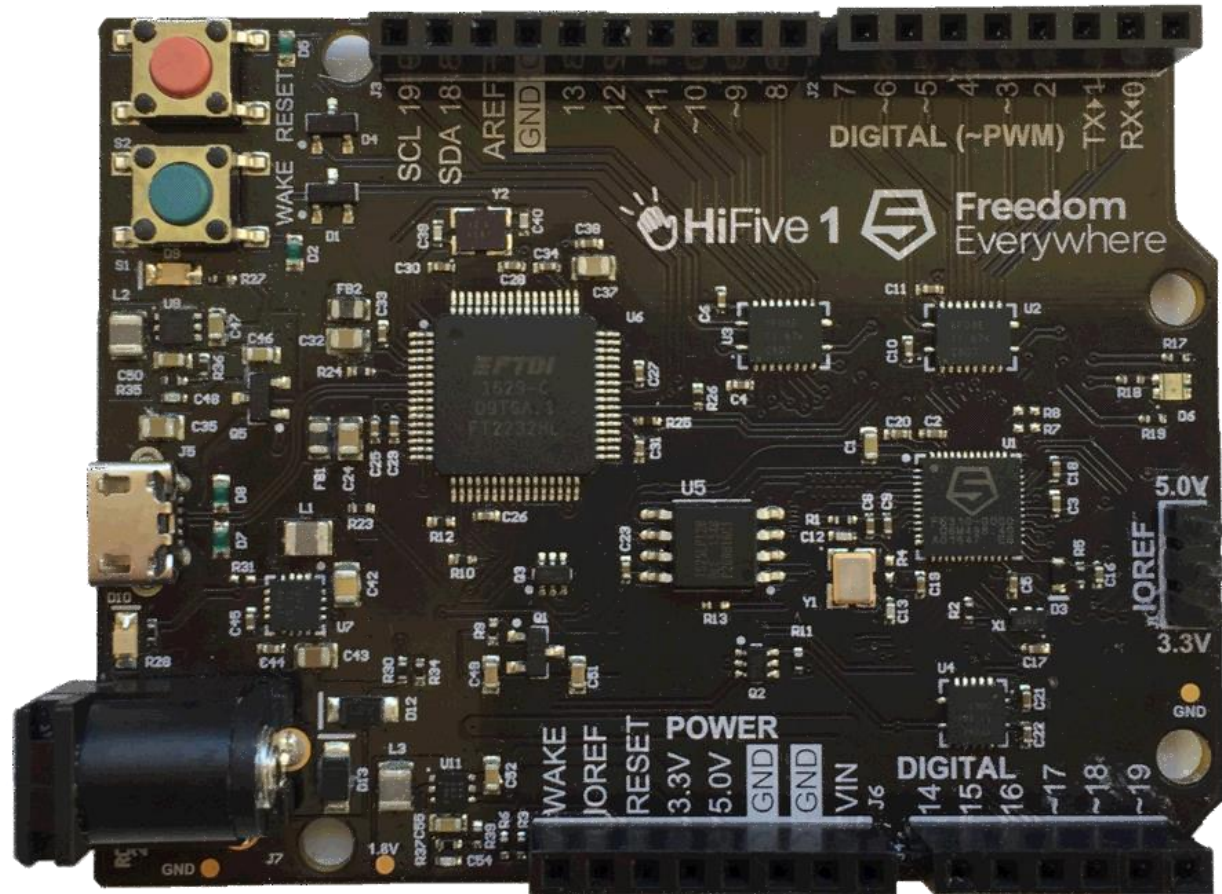
- Most challenging parts:
  - pmap in kernel
  - Run-time linker in userspace
- 6 months from scratch
- 25k lines diff (200 new files)
- Thanks to people involved: Arun Thomas, Andrew Turner, Robert Watson, Ed Maste, David Chisnall, Yukishige Shibata (Sony Japan), Li-Wen Hsu

# Status update

- v1.10 of Privilege Specification
- HiFive1 board release
- FreeBSD board is planned on Quarter 1 2018
- GCC 7 target upstream
- NVIDIA is to ship all of their GPUs with RISC-V processor
- Next RISC-V workshop November 2017 in San Jose

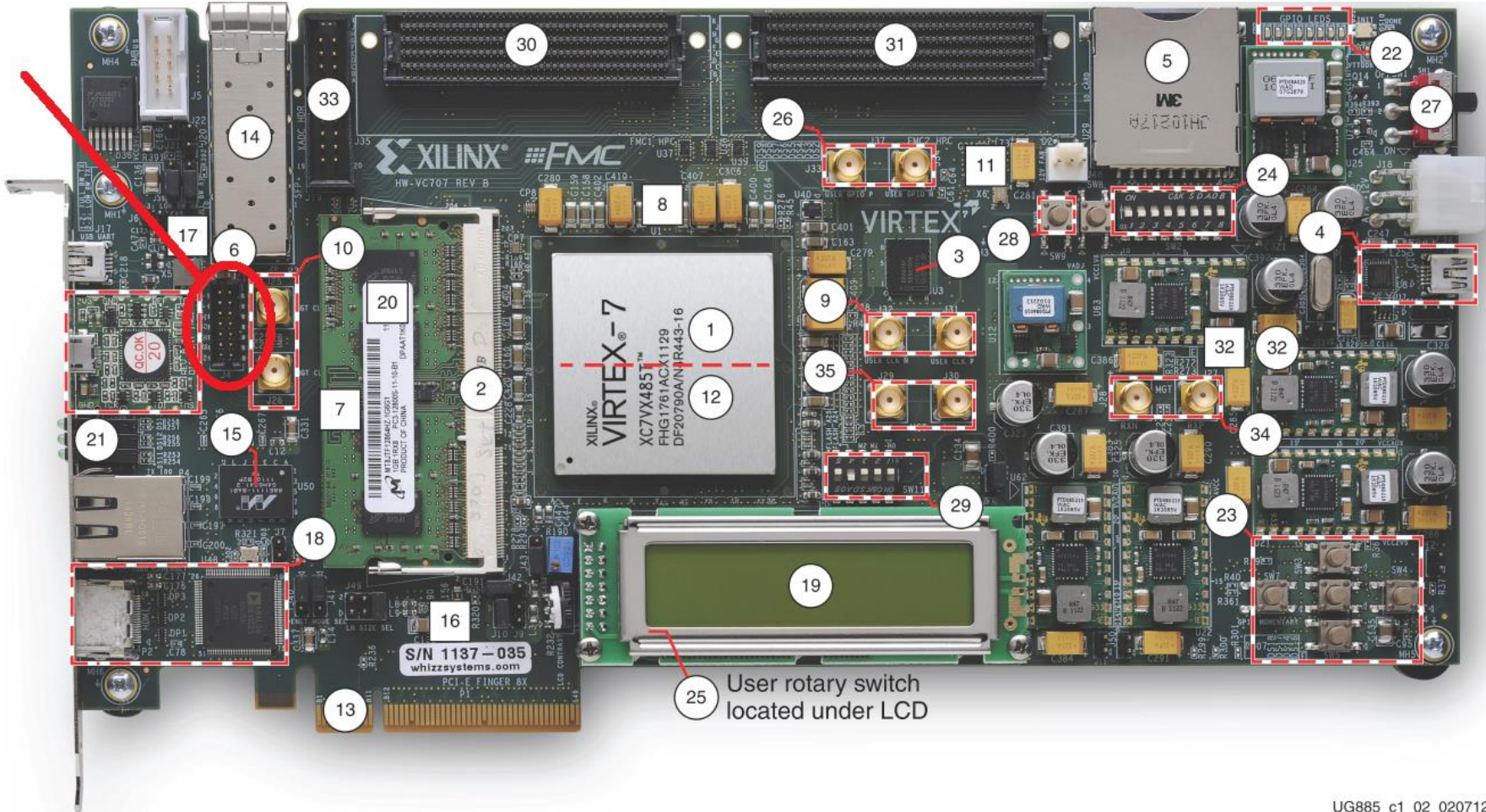
# SiFive HiFive1

- TSMC 180nm
- 320+ MHZ
- 16kb SRAM
- 16kb I-cache
- Chisel, open source
- Board available \$59





## VC707



UG885\_c1\_02\_020712

**Figure 1-2: VC707 Board Component Locations**



# Privileged Architecture v1.10: changes

- Not compatible on S-mode with v1.9
  - Next versions “should” be compatible with v1.10 on S-mode
- Changes:
  - Built-in macros and compiler arguments changed
  - SBI interface, VM changes, BBL changes
  - Physical Memory Protection (PMP) Unit introduced
  - Support for FDT

# Compiler arguments changed

-mno-float, -msoft-float removed

-march=rv64imafdc -mabi=lp64

“a” – atomic

“m” – multiplication

“fd” – double precision floating point unit

“c” – compressed

# Compiler built-in defines

- `__riscv__`, `__riscv64` removed!

`__riscv`

`__riscv_compressed`

`__riscv_atomic`

`__riscv_mul`

`__riscv_div`

`__riscv_muldiv`

`__riscv_fdiv`

`__riscv_fsqrt`

`__riscv_float_abi_soft`

`__riscv_float_abi_single`

`__riscv_float_abi_double`

`__riscv_cmodel_medlow`

`__riscv_cmodel_medany`

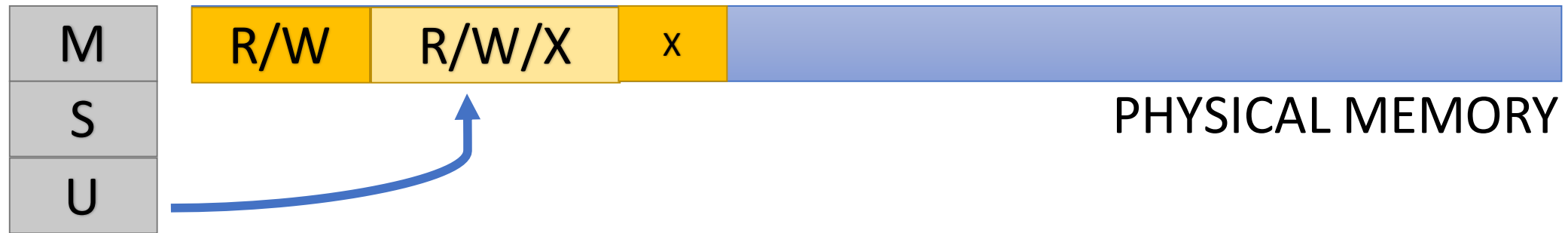
`__riscv_cmodel_pic`

`__riscv_xlen == 64`

# SBI interface changed

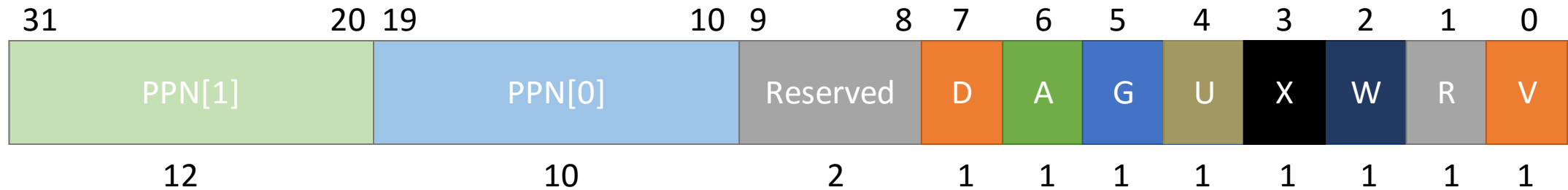
- Old way: pre-defined function address in physical memory
- New way: ecall to upper (machine) privileged level
- Required for:
  - Timer
  - Console
  - IPI
  - Shutdown
  - Hart ID

# PMP unit



- R/W/X on 4-byte granularity, 16 regions
- Composable with MMU
- Can be locked on M-mode
- When enabled, modes below M have no memory permissions

# Virtual Memory changes

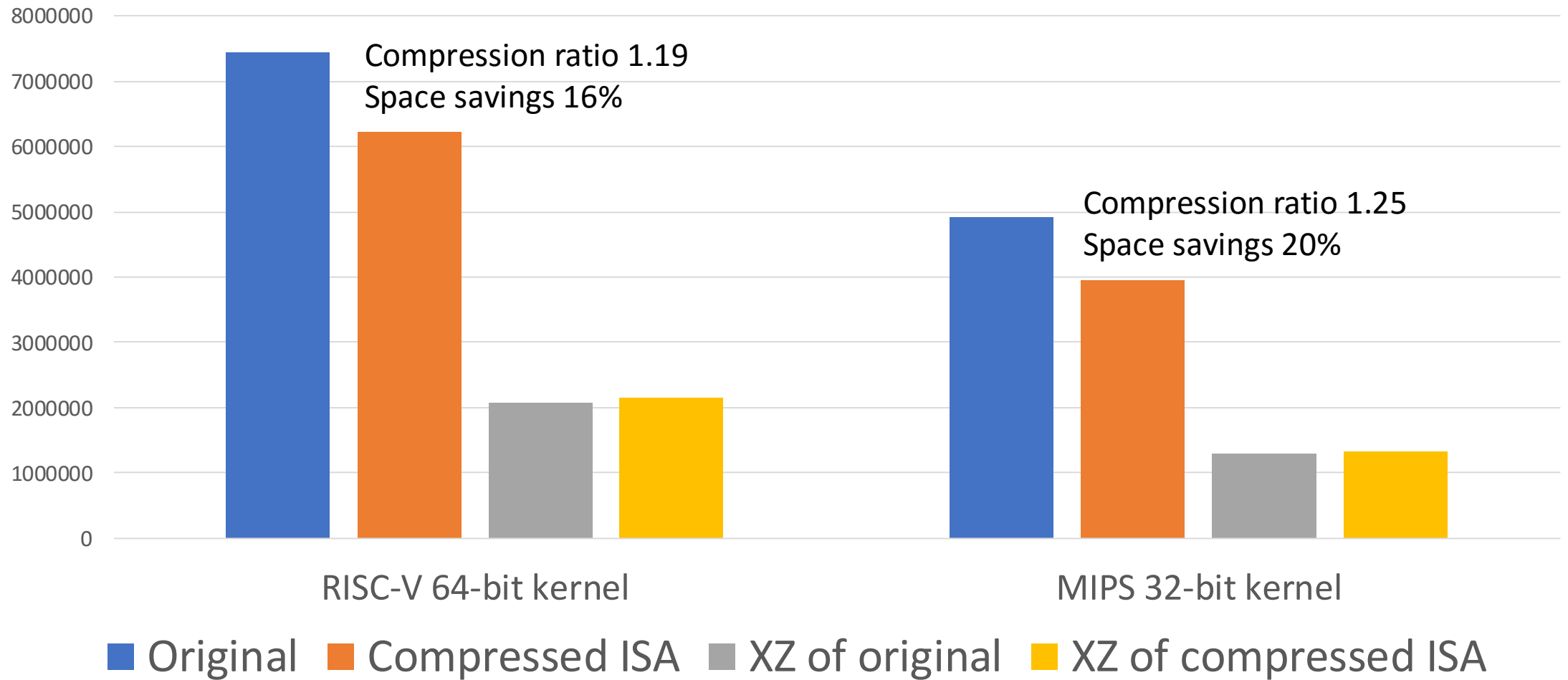


- VM turn on/off using `satp` register
- Supervisor can't access user pages by default
  - Syscalls: set SUM bit in `sstatus` register
- R, W, X controlled separately
  - Support for X-only pages
  - Combination W &  $\sim$ R reserved

# “C”- Compressed Extension

- Saves space for low-end deeply embedded
- Saves cache footprint for high-end workload
- Compresses some instructions to 2 bytes
- Declared 25-30% smaller code with extension turned on

# “C”- Compressed Extension (continued)





# Compressed extension assembly

```
/home/br/obj/riscv.riscv64/usr/home/br/dev/freebsd-riscv/sys/GENERIC/kernel:      file format elf64-little

Disassembly of section .text:

ffffffc000200000 <_start>:
ffffffc000200000:      000402b7      lui      t0,0x40
ffffffc000200004:      1002a073      csrs     sstatus,t0
ffffffc000200008:      8d2a         mv       s10,a0
ffffffc00020000a:      8dae         mv       s11,a1
ffffffc00020000c:      000d0463      beqz     s10,ffffffc000200014 <_start+0x14>
ffffffc000200010:      1b00406f      j       fffffffc0002041c0 <mpentry>
ffffffc000200014:      00461497      auipc    s1,0x461
ffffffc000200018:      fec48493      addi     s1,s1,-20 # fffffffc000661000 <pagetab
ffffffc00020001c:      00462917      auipc    s2,0x462
ffffffc000200020:      fe490913      addi     s2,s2,-28 # fffffffc000662000 <pagetab
ffffffc000200024:      00c95913      srli     s2,s2,0xc
ffffffc000200028:      fff0079b      addiw    a5,zero,-1
ffffffc00020002c:      02679793      slli     a5,a5,0x26
ffffffc000200030:      83f9         srli     a5,a5,0x1e
ffffffc000200032:      1ff7f793      andi     a5,a5,511
ffffffc000200036:      4e85         li       t4,1
ffffffc000200038:      00a91f13      slli     t5,s2,0xa
ffffffc00020003c:      01eeefb3      or       t6,t4,t5
ffffffc000200040:      4821         li       a6,8
ffffffc000200042:      030787bb      mulw     a5,a5,a6
ffffffc000200046:      00f482b3      add      t0,s1,a5
ffffffc00020004a:      01f2b023      sd       t6,0(t0) # 40000 <kernbase-0xffffffffbf
ffffffc00020004e:      00462497      auipc    s1,0x462
ffffffc000200052:      fb248493      addi     s1,s1,-78 # fffffffc000662000 <pagetab
:[]
```

# FDT support

- GENERIC kernel
- timer/console moved from OFW bus to nexus bus



```
# ./spike -dump-dts
```

```
# ./spike -m2048 -p8 /path/to/bbl
```

# Hardware support for v1.10 privilege spec

Implementation	Status
Spike	Fully supported
RocketChip	Unknown
lowRISC	Unknown
QEMU	Unknown
Gem5	Unknown
Real hardware	N/A

# Future plans

- ASIC chips and boards
- LLVM support
- Ports and packages
  - QEMU user (syscall emulation) mode
- bhyve support ?

# Device driver

- How to attach
- Resources
  - How to access
  - Where to request

# Device driver frameworks

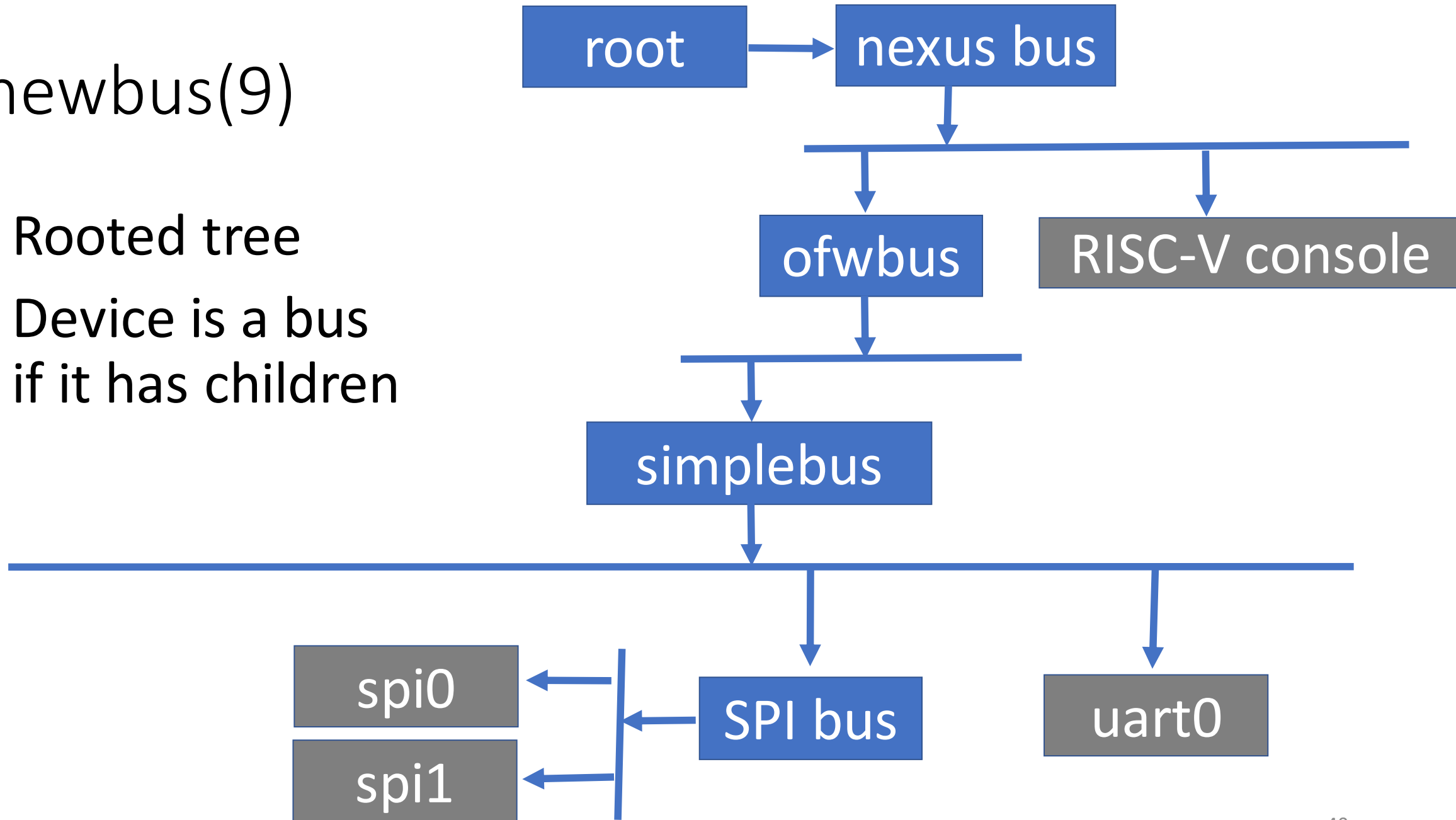
- kld
- newbus(9)
- rman(9)
- cdevsw
- ofw and FDT
- bus\_space
- bus\_dma
- sysctl
- SYSINIT

# SYSINIT

```
static int
test_handler(module_t mod, int what, void *arg)
{
    switch (what) {
        case MOD_LOAD:
        case MOD_UNLOAD:
    }
}
DECLARE_MODULE(test, SI_SUB_VM, ...)
```

newbus(9)

- Rooted tree
- Device is a bus if it has children





# newbus(9) methods

```
static device_method_t test_methods[] = {  
    DEVMETHOD(device_probe,      test_probe),  
    DEVMETHOD(device_attach,     test_attach),  
}
```

```
DRIVER_MODULE(test, simplebus, test_methods, ...)
```

# Resources

- Device tree source (DTS)
- `bus_alloc_resources(dev, ...)`
- `bus_setup_intr(dev, intr_func, ...)`

# cdevsw

- open(2)
- read(2)
- ioctl(2)
- mmap(2)
- poll(2)

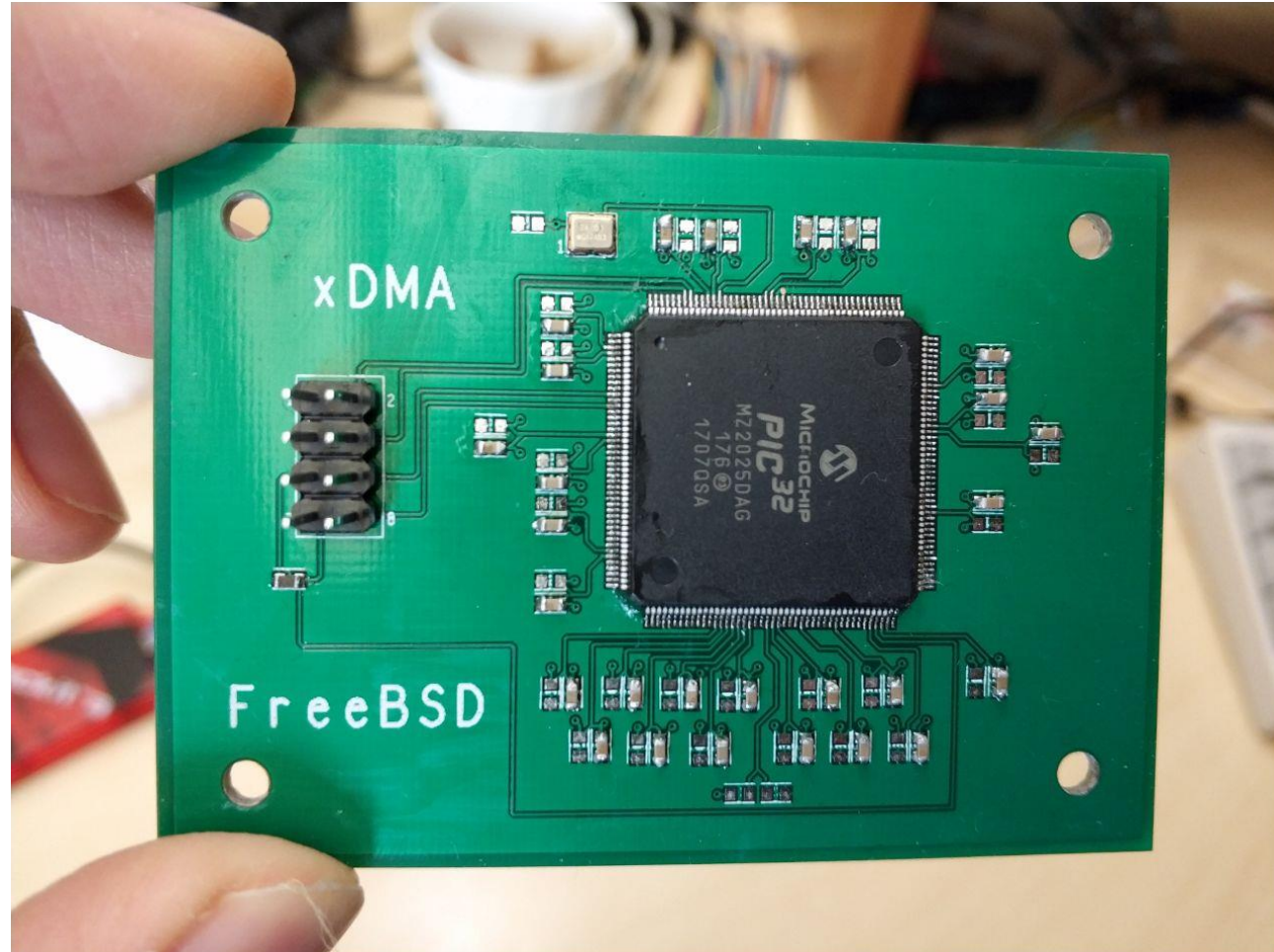
```
fd = open("/dev/test", ...)
vaddr = mmap(fd, ...)
```

# Frameworks / subsystems

- SPI bus
- I2C bus
- xdma(9)
- Ifnet(9)
- callout(9)
- pci
- sound(9)
- Wi-Fi stack
- vt(4)
- taskqueue(9)

# FreeBSD/MIPS board

- 27 capacitors
- 1 resistor
- 1 pin header
- 1 crystal oscillator
- 1 MIPS CPU



# Questions ?

Project home:  
<http://wiki.freebsd.org/riscv>