Slides: tinyurl.com/riscv-kr-22

Kernel Recipes

June 1-3, 2022

Baylibre

Linux on RISC-V

Drew Fustini <dfustini@baylibre.com>

$ whoami

- Linux kernel developer, BayLibre
  - embedded software consultancy based in Nice, France, with ~50 engineers around the world contributing to open source projects like Linux, U-Boot, Android and Zephyr
- Board of Directors, BeagleBoard.org Foundation
- Board of Directors, Open Source Hardware Association (OSHWA)
  - OSHW Certification Program
- Ambassador, RISC-V International

RISC-V: a Free and Open ISA

- Started by a computer architecture research group at University of California Berkeley in 2010 led by Krste Asanovic
- **V** as in the roman numeral five, because it is the 5th **RISC** instruction set to come out of UC Berkeley
- **Free and Open** because the specifications are published under an open source license: Creative Commons Attribution 4.0 International
  - Volume 1, Unprivileged Spec v. 20191213 [PDF]
  - Volume 2, Privileged Spec v. 20211203 [PDF]

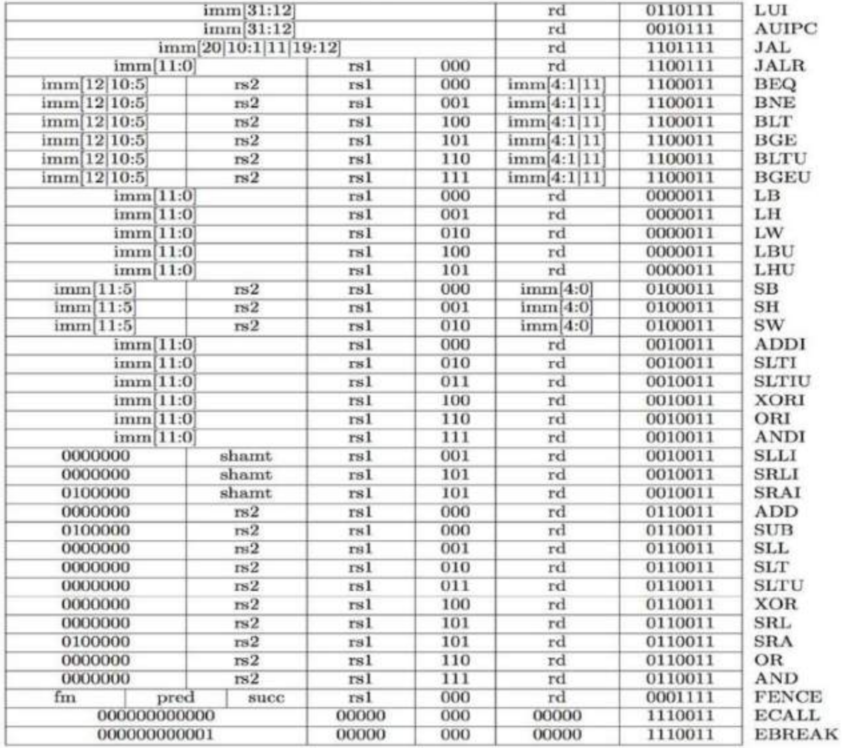
What is different about RISC-V?

- Simple clean-slate design
  - Avoids any dependencies on microarchitecture style *(in-order, out-of-order, etc)*
- Modular design
  - Suitable for everything from microcontrollers to supercomputers
- Stable base
  - Base integer ISAs and standard extensions are frozen
  - Additions via optional extensions, not new versions

(source: Instruction Sets Want to be Free, Krste Asanovic)

RISC-V base integer ISAs

- RV32I: 32-bit
  - less than 50 instructions needed!
- RV64I: 64-bit
  - Most important for Linux
- RV128I: 128-bit
  - Future-proof address space

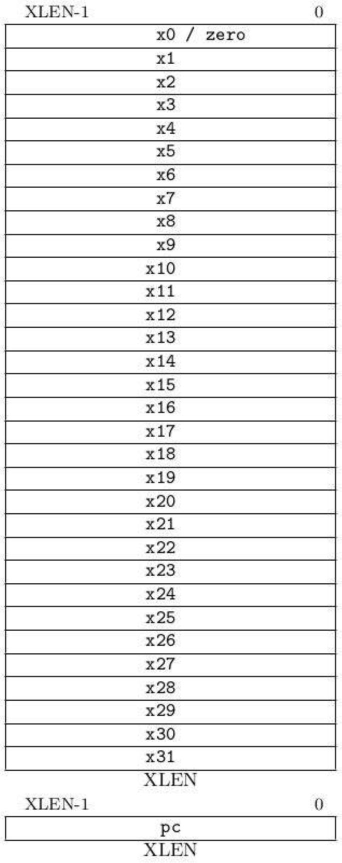


imm[31:12]				rd	0110111	LUI
imm[31:12]				rd	0010111	AUIPC
imm[20|10:1|11|19:12]				rd	1101111	JAL
imm[11:0]		rs1	000	rd	1100111	JALR
imm[12|10:5]	rs2	rs1	000	imm[4:1|11]	1100011	BEQ
imm[12|10:5]	rs2	rs1	001	imm[4:1|11]	1100011	BNE
imm[12|10:5]	rs2	rs1	100	imm[4:1|11]	1100011	BLT
imm[12|10:5]	rs2	rs1	101	imm[4:1|11]	1100011	BGE
imm[12|10:5]	rs2	rs1	110	imm[4:1|11]	1100011	BLTU
imm[12|10:5]	rs2	rs1	111	imm[4:1|11]	1100011	BGEU
imm[11:0]		rs1	000	rd	0000011	LB
imm[11:0]		rs1	001	rd	0000011	LH
imm[11:0]		rs1	010	rd	0000011	LW
imm[11:0]		rs1	100	rd	0000011	LBU
imm[11:0]		rs1	101	rd	0000011	LHU
imm[11:5]	rs2	rs1	000	imm[4:0]	0100011	SB
imm[11:5]	rs2	rs1	001	imm[4:0]	0100011	SH
imm[11:5]	rs2	rs1	010	imm[4:0]	0100011	SW
imm[11:0]		rs1	000	rd	0010011	ADDI
imm[11:0]		rs1	010	rd	0010011	SLTI
imm[11:0]		rs1	011	rd	0010011	SLTIU
imm[11:0]		rs1	100	rd	0010011	XORI
imm[11:0]		rs1	110	rd	0010011	ORI
imm[11:0]		rs1	111	rd	0010011	ANDI
0000000	shamt	rs1	001	rd	0010011	SLLI
0000000	shamt	rs1	101	rd	0010011	SRLI
0100000	shamt	rs1	101	rd	0010011	SRAI
0000000	rs2	rs1	000	rd	0110011	ADD
0100000	rs2	rs1	000	rd	0110011	SUB
0000000	rs2	rs1	001	rd	0110011	SLL
0000000	rs2	rs1	010	rd	0110011	SLT
0000000	rs2	rs1	011	rd	0110011	SLTU
0000000	rs2	rs1	100	rd	0110011	XOR
0000000	rs2	rs1	101	rd	0110011	SRL
0100000	rs2	rs1	101	rd	0110011	SRA
0000000	rs2	rs1	110	rd	0110011	OR
0000000	rs2	rs1	111	rd	0110011	AND
fm pred	succ	rs1	000	rd	0001111	FENCE
000000000000		00000	000	00000	1110011	ECALL
000000000001		00000	000	00000	1110011	EBREAK

(source: RISC-V Summit 2019: State of the Union, Krste Asanovic)

RISC-V base integer registers

- XLEN defines the register width
  - XLEN=32 for RV32I
  - XLEN=64 for RV64I
- 32 registers named x0 to x31
- Dedicated PC register
- Base ISA talk by Andrew Waterman explains the instruction encoding scheme



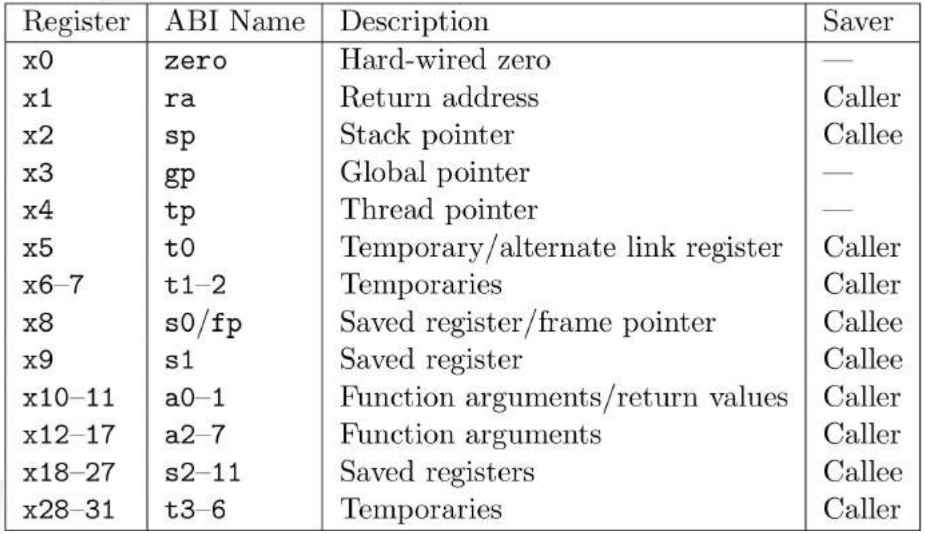
XLEN-1 0
x0 / zero
x1
x2
x3
x4
x5
x6
x7
x8
x9
x10
x11
x12
x13
x14
x15
x16
x17
x18
x19
x20
x21
x22
x23
x24
x25
x26
x27
x28
x29
x30
x31
XLEN

XLEN-1 0
pc
XLEN

(source: Figure 2.1: RISC-V base unprivileged integer register state)

RISC-V ABI

- x1 to x31 are all equally general-use registers as far as the processor is concerned
- RISC-V psABI defines standard functions for these registers *[PDF]*
  - s0 to s11 are preserved across function calls
  - argument registers a0 to a7 and the temporary registers t0 to t6 are not



Register	ABI Name	Description	Saver
x0	zero	Hard-wired zero	—
x1	ra	Return address	Caller
x2	sp	Stack pointer	Callee
x3	gp	Global pointer	—
x4	tp	Thread pointer	—
x5	t0	Temporary/alternate link register	Caller
x6–7	t1–2	Temporaries	Caller
x8	s0/fp	Saved register/frame pointer	Callee
x9	s1	Saved register	Callee
x10–11	a0–1	Function arguments/return values	Caller
x12–17	a2–7	Function arguments	Caller
x18–27	s2–11	Saved registers	Callee
x28–31	t3–6	Temporaries	Caller

(source: RISC-V Assembly Programmer's Manual)

RISC-V Standard Extensions

- **M**: integer multiply/divide
- **A**: atomic memory operations
- **F**, **D**, **Q**: floating point, double-precision, quad-precision
- **G**: “general purpose” ISA, equivalent to **IMAFD**
- **C**: compressed instructions conserve memory and cache
- Most Linux distributions target **RV64GC**

(source: RISC-V Summit 2019: State of the Union, Krste Asanovic)

Ratified in 2021

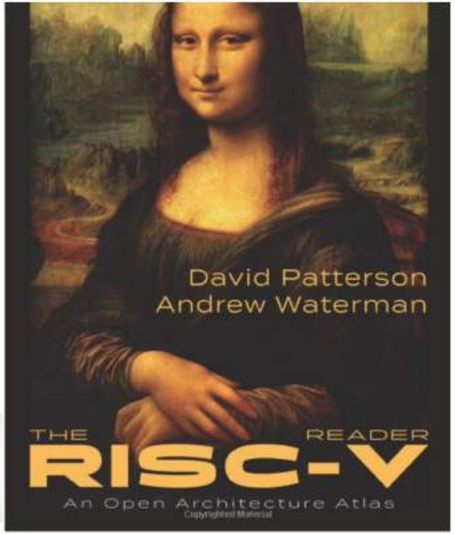
- 15 new specifications representing more than 40 extensions
- Vector
- Hypervisor
- Scalar Cryptography
- Bit Manipulation

RISC-V Profiles

- RISC-V is a highly modular and extensible architecture
    - Flexibility to pick and choose what is right for your processor design, but that flexibility creates a large number of possible combinations
- RISC-V Profiles specify a much smaller set of ISA choices that represent the most common use-cases
    - RVM for microcontrollers intended to run bare-metal code or an RTOS
    - RVA for application processors designed to run full operating system like Linux
- RISC-V Summit talk by Greg Favor [slides]

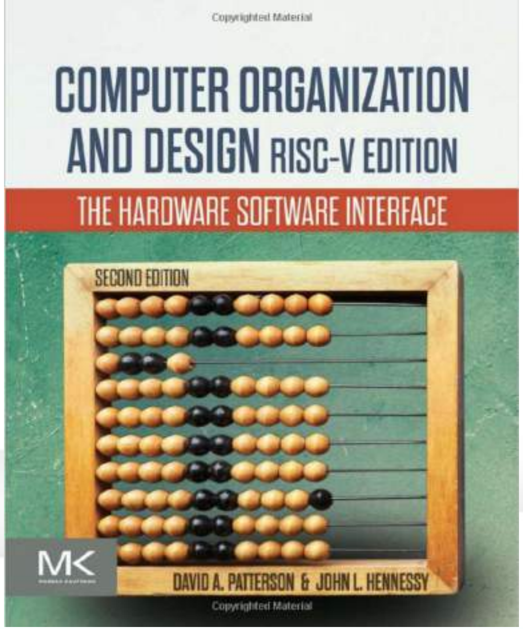
Learn more about RISC-V

- Get up-to-speed quick with the RISC-V Reader



Learn more about RISC-V

- Textbook: Computer Organization and Design, RISC-V Edition


Copyrighted Material
COMPUTER ORGANIZATION AND DESIGN RISC-V EDITION
THE HARDWARE SOFTWARE INTERFACE
SECOND EDITION
MK
DAVID A. PATTERSON & JOHN L. HENNESSY
Copyrighted Material

RISC-V and Industry

- RISC-V International now controls the specifications: riscv.org
  - Non-profit with 2,700+ members including companies & universities from 70 countries
  - Become a member - free of cost to individuals and non-profits!
  - RISC-V International YouTube channel has hundreds of talks
- Companies have already shipped billions of RISC-V cores
  - Nvidia GPUs have RISC-V cores for system management tasks
  - Seagate and Western Digital are using RISC-V cores in storage controllers

(source: State of the Union, Krste Asanovic)

RISC-V and Industry

- No ISA licensing fees or royalties
    - Avoid legal costs and delays caused by complex licensing agreements
- Freedom to choose microarchitecture implementation
    - An open ISA means that everyone has an architecture license
- Freedom to leverage existing open source implementations
    - Broad range of open source cores already available from small embedded cores to high-performance out-of-order superscalar designs

“Is RISC-V an Open Source processor?”

- RISC-V is a set of specifications under an open source license
- RISC-V implementations can be open source or proprietary
- Open specifications make open source implementations possible
- **An open ISA makes it possible to design an open source processor**

RISC-V open source cores

- Academia
  - Rocket and BOOM from Berkeley, PULP family of cores from ETH Zurich
- Industry
  - SweRV created by Western Digital and now developed by CHIPS Alliance
  - OpenHW Group creating proven verified IP like their Core-V designs
  - Google OpenTitan silicon root of trust project uses LowRISC Ibex core

RISC-V open source cores

- FPGA soft-cores
  - PicoRV32, RVfpga, SERV, and VexRiscV
- FOSSi Foundation
  - El Correo Libre monthly newsletter for the latest on open source cores
- Build your own open source SoC with an open source silicon toolchain
  - Google worked with Skywater to open source their 130 nm PDK (process development kit). Google offers free-of-cost MPW (multi-project wafer) runs to open source projects. Learn to design your own using open source design tools with Zero to ASIC.

RISC-V software ecosystem

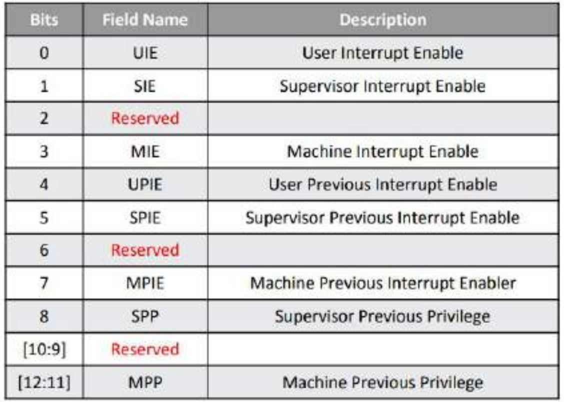
- RISC-V already has a well supported software ecosystem
  - RISC-V International software committee coordinates efforts of member organizations
  - RISC-V extension and feature support
  - PLCT Lab led by Wei Wu at ISCAS does a lot of compiler and runtime work
- Operating systems: Linux, BSDs, FreeRTOS, Zephyr
- Toolchains & libraries: gcc, glibc, gdb, binutils, clang/llvm, newlib
- Languages and Runtimes: V8, Node.js, Rust, Go, OpenJDK, Python

RISC-V Privileged Architecture

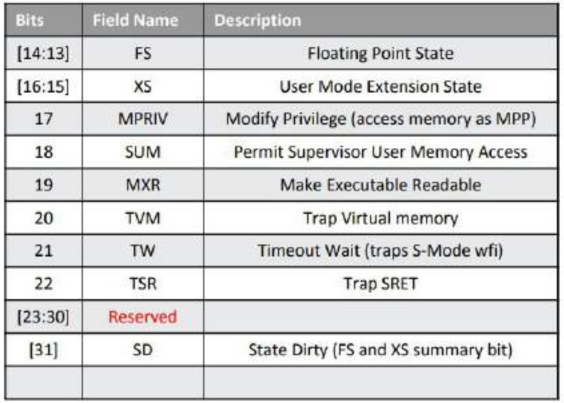
- Three privilege modes
  - User (U-mode): application
  - Supervisor (S-mode): OS kernel
  - Machine (M-mode): firmware
- Environment Call (`ECALL`) instruction
  - Transfer control to a higher privileged mode
  - Userspace program (u-mode) uses ECALL to make system call into OS kernel (s-mode)

Control and Status Registers (CSRs)

- CSR have their own dedicated instructions to read and write
- CSR are specific to a mode (e.g. m-mode and s-mode)
- Machine Status (`mstatus`) is an important CSR



Bits	Field Name	Description
0	UIE	User Interrupt Enable
1	SIE	Supervisor Interrupt Enable
2	Reserved	
3	MIE	Machine Interrupt Enable
4	UPIE	User Previous Interrupt Enable
5	SPIE	Supervisor Previous Interrupt Enable
6	Reserved	
7	MPIE	Machine Previous Interrupt Enabler
8	SPP	Supervisor Previous Privilege
[10:9]	Reserved	
[12:11]	MPP	Machine Previous Privilege

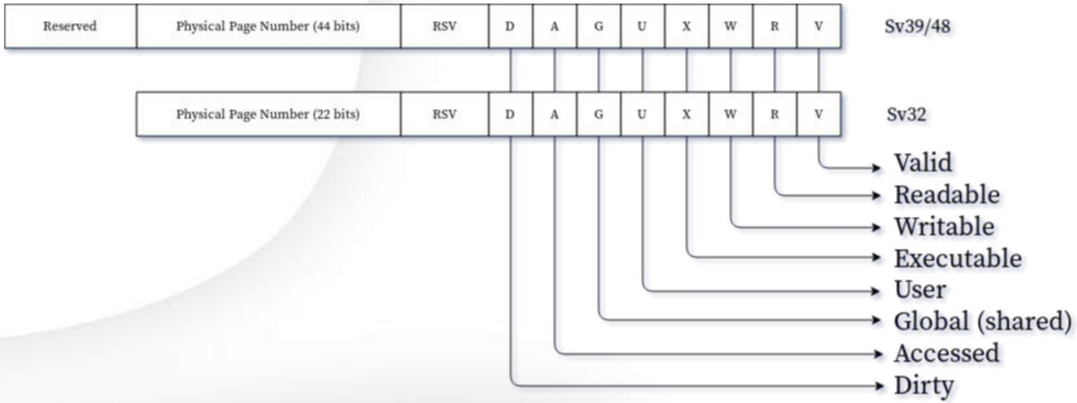


Bits	Field Name	Description
[14:13]	FS	Floating Point State
[16:15]	XS	User Mode Extension State
17	MPRIV	Modify Privilege (access memory as MPP)
18	SUM	Permit Supervisor User Memory Access
19	MXR	Make Executable Readable
20	TVM	Trap Virtual memory
21	TW	Timeout Wait (traps S-Mode wfi)
22	TSR	Trap SRET
[23:30]	Reserved	
[31]	SD	State Dirty (FS and XS summary bit)

(source: Introduction to the RISC-V Architecture [PDF],Drew Barbier)

RISC-V Virtual Memory

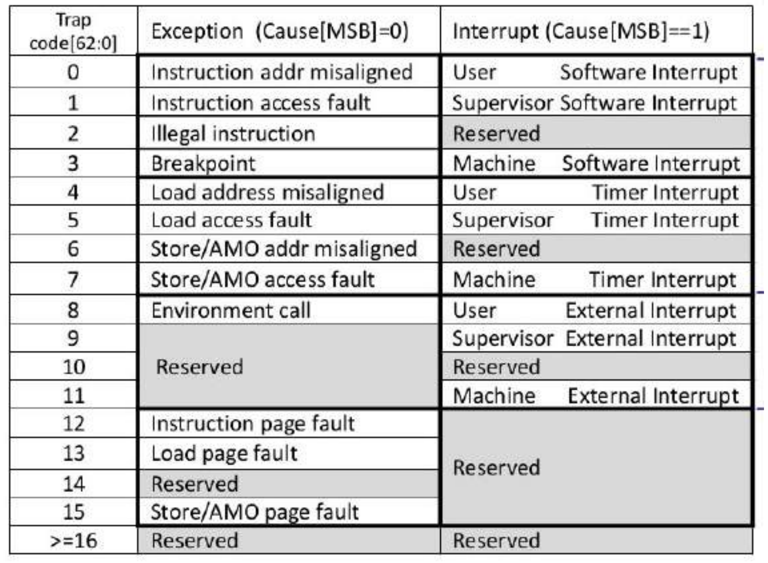
- `satp` CSR (Supervisor Address Translation and Protection) controls supervisor-mode address translation and protection
- Sv32: 3 level page table
- Sv39: 3 level page table
- Sv48: 4 level page table
- Sv57: 5 level page table



(source: Demystifying the RISC-V Linux software stack, Nick Kossifidis)

RISC-V Trap Handling

- Exceptions occur synchronously
- Interrupts occur asynchronously
- <x>cause CSR indicates which interrupt or exception occurred
    - mcause for m-mode / scause for s-mode
- Corresponding bit is set in <x>E/IP CSR



Trap code[62:0]	Exception (Cause[MSB]=0)	Interrupt (Cause[MSB]==1)
0	Instruction addr misaligned	User Software Interrupt
1	Instruction access fault	Supervisor Software Interrupt
2	Illegal instruction	Reserved
3	Breakpoint	Machine Software Interrupt
4	Load address misaligned	User Timer Interrupt
5	Load access fault	Supervisor Timer Interrupt
6	Store/AMO addr misaligned	Reserved
7	Store/AMO access fault	Machine Timer Interrupt
8	Environment call	User External Interrupt
9	Reserved	Supervisor External Interrupt
10		Reserved
11		Machine External Interrupt
12	Instruction page fault	Reserved
13	Load page fault	
14	Reserved	
15	Store/AMO page fault	
>=16	Reserved	Reserved

(source: RISC-V Privileged Architecture [PDF], Allen Baum)

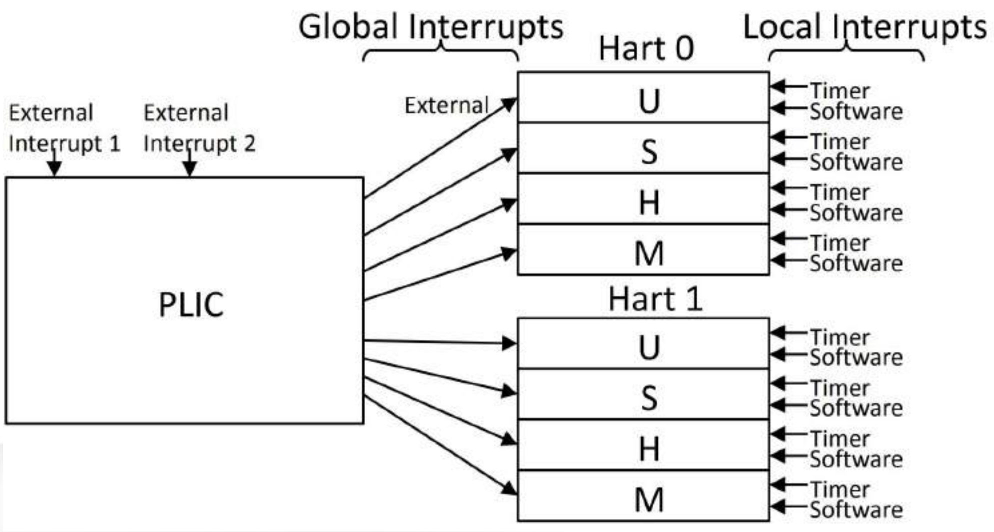
What is a Hart?

- Hart is a **har**dware **t**hread
- Each RISC-V core contains an independent instruction fetch unit
- A RISC-V core with multi-threading (SMT) would contain multiple harts
- Each hart is a processor from the perspective of Linux
  - Imagine a RISC-V laptop which has 2 cores with 2 harts per core
  - Linux would see 4 processors

(source: Section 1.1 in RISC-V Unprivileged spec)

RISC-V Interrupts

- Local per-hart interrupts
  - CLINT (Core Local Interruptor)
  - CLIC (Core Local Interrupt Controller)
- Global interrupts
  - PLIC (Platform Level Interrupt Controller)

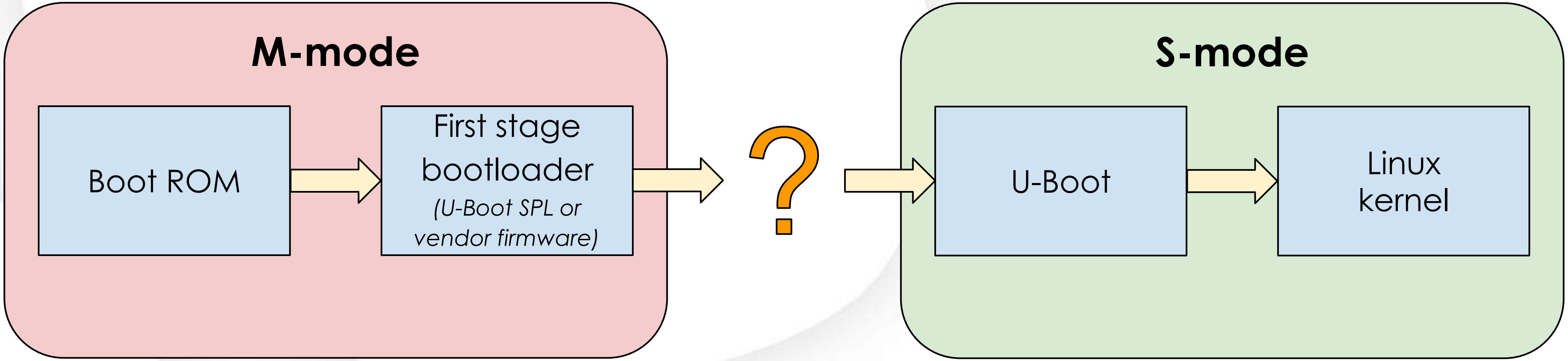


(source: RISC-V Fast Interrupts [PDF], Krste Asanovic)

Advanced Interrupt Architecture (AIA)

- Developed on the AIA SIG mailing list: tech-aia
- APLIC (Advanced Platform-Level Interrupt Controller) replaces PLIC
- Adds IMSIC (Incoming Message-Signaled Interrupt Controller) for PCIe
- AIA is complimented by ACLINT (Advanced Core Local Interruptor)
    - Developed on the tech-unixplatformspec mailing list
    - Backwards compatible with the SiFive CLINT but restructured to be more efficient
    - RISC-V Summit talk by Anup Patel and John Hauser [slides]

RISC-V Boot Flow



M-mode

Boot ROM

First stage bootloader

(U-Boot SPL or vendor firmware)

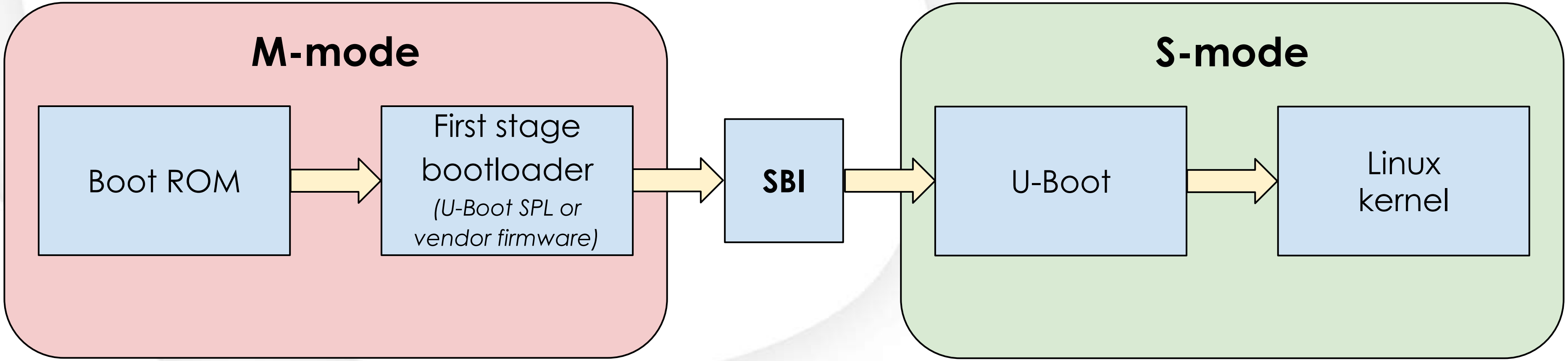
?

S-mode

U-Boot

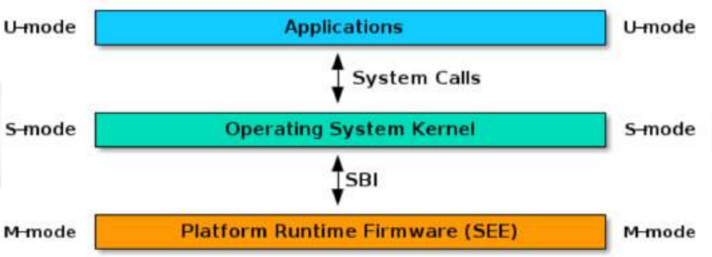
Linux kernel

RISC-V Boot Flow



Supervisor Binary Interface (SBI)

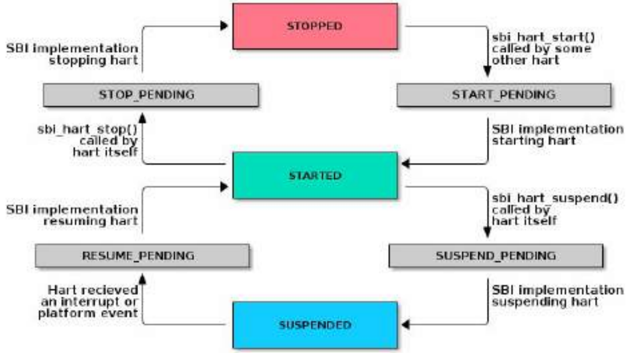
- Non-ISA RISC-V specification
  - This means it does not add or modify and RISC-V instructions
- The calling convention between S-mode and M-mode
  - Allows supervisor-mode (s-mode) software like the Linux to be portable across RISC-V implementations by abstracting platform specific functionality



Supervisor Binary Interface (SBI)

- Required by the UNIX-Class Platform Specification
  - Mailing list: tech-unixplatformspec
  - This will be replaced by the upcoming RISC-V Platform Specification
- Small core along with a set of optional modular extensions
  - Base extension - query basic information about the machine
  - Timer extension - program the clock for the next event
  - IPI extension - send an inter-processor interrupt to harts defined in mask
  - RFENCE extension - instructs remote harts to execute FENCE.I instruction

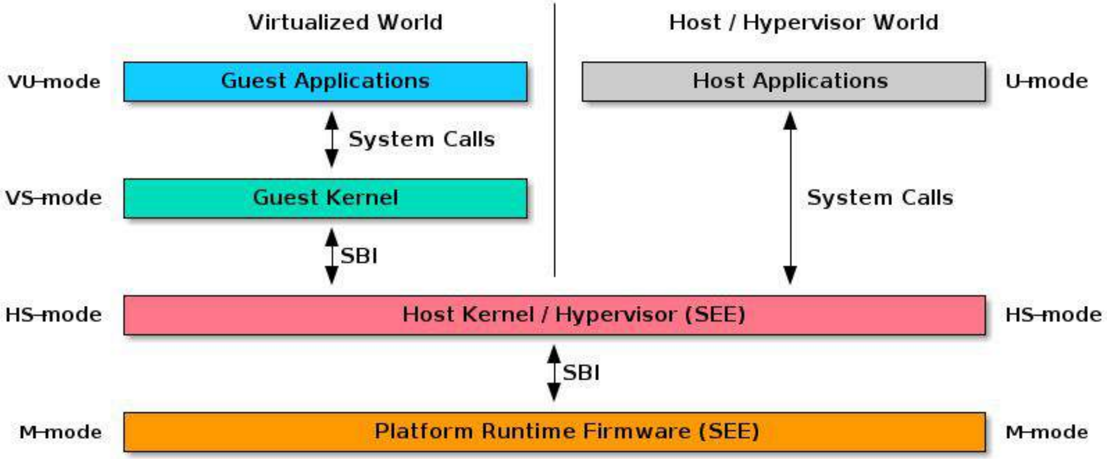
SBI Extensions


STOPPED
SBI implementation stopping hart
sbi_hart_start() called by some other hart
STOP_PENDING
START_PENDING
sbi_hart_stop() called by hart itself
SBI implementation starting hart
STARTED
SBI implementation resuming hart
sbi_hart_suspend() called by hart itself
RESUME_PENDING
SUSPEND_PENDING
Hart recieved an interrupt or platform event
SBI implementation suspending hart
SUSPENDED

- Hart State Management (HSM)
    - S-mode can request to stop, start or suspend a hart
- System Reset
    - Supervisor-mode software can request system-level reboot or shutdown
- Performance Monitoring Unit
    - Interface for supervisor-mode to configure and use the RISC-V hardware performance counters with assistance from the machine-mode
    - ”Performance Monitoring in RISC-V using perf” by Atish Patra

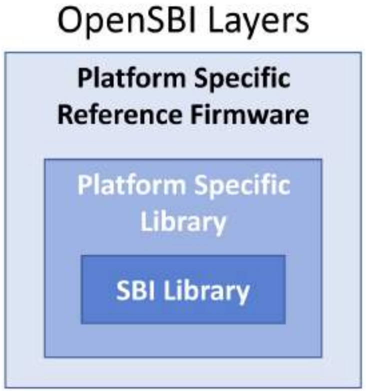
Hypervisor extension

- Hypervisor Supervisor mode (HS-mode) where host kernel runs
- Virtualized Supervisor mode (VS-mode) where the guest kernel runs



OpenSBI

- Open source implementation of SBI
  - Core library
  - Platform specific libraries
  - Full reference firmware for some platforms
- Provides runtime services to S-mode software
  - SBI extensions present on a platform define the available runtime services
  - Unimplemented instructions will trap and OpenSBI can emulate


OpenSBI Layers
Platform Specific Reference Firmware
Platform Specific Library
SBI Library

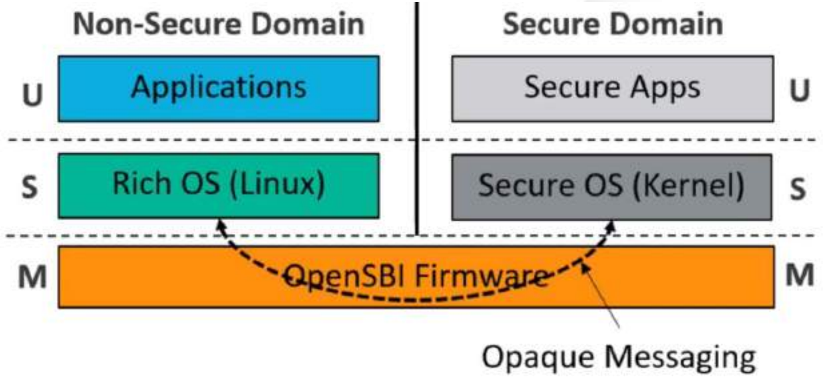
(source: OpenSBI Deep Dive, Anup Patel)

OpenSBI Generic Platform

- No need to add code to OpenSBI for each new platform
  - First-stage bootloader, like U-Boot SPL, is expected to pass a Device Tree to OpenSBI which describes all the platform specific functionality
- The same OpenSBI binary can be used across platforms
  - Many RISC-V boards and emulators now use Generic Platform
  - Linux distros do not need to ship a different OpenSBI build for each board

OpenSBI Domain Support

- An OpenSBI domain is a system-level partition of underlying hardware having its own memory regions and HARTs
- Talk by Anup Patel



UEFI Support

- UEFI is a standard interface between firmware and operating systems, and it is used on most x86 and arm64 platforms
- U-Boot and TianoCore EDK2 both have UEFI implementations on RISC-V
- Grub2 can be used as an UEFI payload on RISC-V
- UEFI support for RISC-V added in Linux 5.10

UEFI Support

- Boot hart ID is known only at boot and it is needed before ACPI tables or DT properties can be parsed
- Hart ID is passed in the a0 register on non-UEFI systems, but the UEFI application calling conventions do not allow this
- RISC-V EFI Boot Protocol allows the OS to discover the boot hart ID
- The public review process has completed, and Sunil V L has added support to the Linux kernel for RISCV_EFI_BOOT_PROTOCOL

RISC-V Platform Specification

- Goal is to support “off-the-shelf” software by standardizing the interface between hardware platforms and operating systems
- Created by the Platforms Horizontal Subcommittee (HSC)
  - Bi-weekly meetings chaired by Kumar Sankaran
  - Mailing list: tech-unixplatformspec
- Platforms talk at RISC-V Summit 2021
  - Philipp Tomsich, Chair of Software HSC, and Mark Himelstein CTO RISC-V International

RISC-V Platform Specification

- OS-A Platform
  - “A” as in application, this is a category of platforms that support full OS like Linux
  - OS-A Common Requirements
  - OS-A Embedded Platform
  - OS-A Server Platform
- RVM-CSI Platform
  - Bare-metal applications or RTOS running on RISC-V microcontrollers
  - CSI is common software interface; goal is to ease porting, not binary compatibility

RISC-V Platform Specification

- OS-A common requirements for Embedded and Server
  - Must comply with the RVA22U and RVA22S ISA profiles as defined in RISC-V ISA Profiles
  - Common requirements for Debug, Timers, Interrupt Controllers
  - Requires serial console with UART 8250 or UART 16550
  - Requirements for runtime services such as SBI extensions
  - Software components must comply with the RISC-V Calling Convention specification and the RISC-V ELF specification

RISC-V Platform Specification

- OS-A Embedded Platform
  - Target might be a single board computer or mobile device
  - PMU counters and events for performance monitoring
  - Boot process must comply with Embedded Base Boot Requirements (EBBR) spec
  - EBBR requires a subset of the UEFI spec which U-Boot has implemented
  - Device Tree (DT) is the required mechanism for the hardware discovery and config
  - GPT partitioning required for shared storage

RISC-V Platform Specification

- OS-A Server Platform
  - Goal is for an enterprise Linux distro like RHEL to “just work” on server-class hardware that complies with this
  - System peripheral requirements like PCIe, watchdog timer, system date/time
  - RAS (Reliability, Availability, and Serviceability) requirements like ECC RAM
  - ACPI is the required mechanism for the hardware discovery and configuration
  - Must comply with the RISC-V ACPI Platform Requirements Specification

RISC-V ACPI Platform Specification

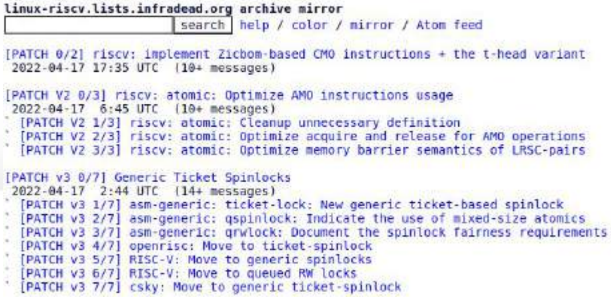
- Defines mandatory ACPI tables and objects for RISC-V server platforms
- New tables are needed for RISC-V
  - RISC-V Hart Capabilities Table (RHCT)
  - RISC-V Timer Description Table (RTDT)
- More details in ‘ACPI for RISC-V: Enabling Server Class Platforms’
  - Sunil V L from Ventana Microsystems at RISC-V Summit [slides]

RISC-V emulation in QEMU

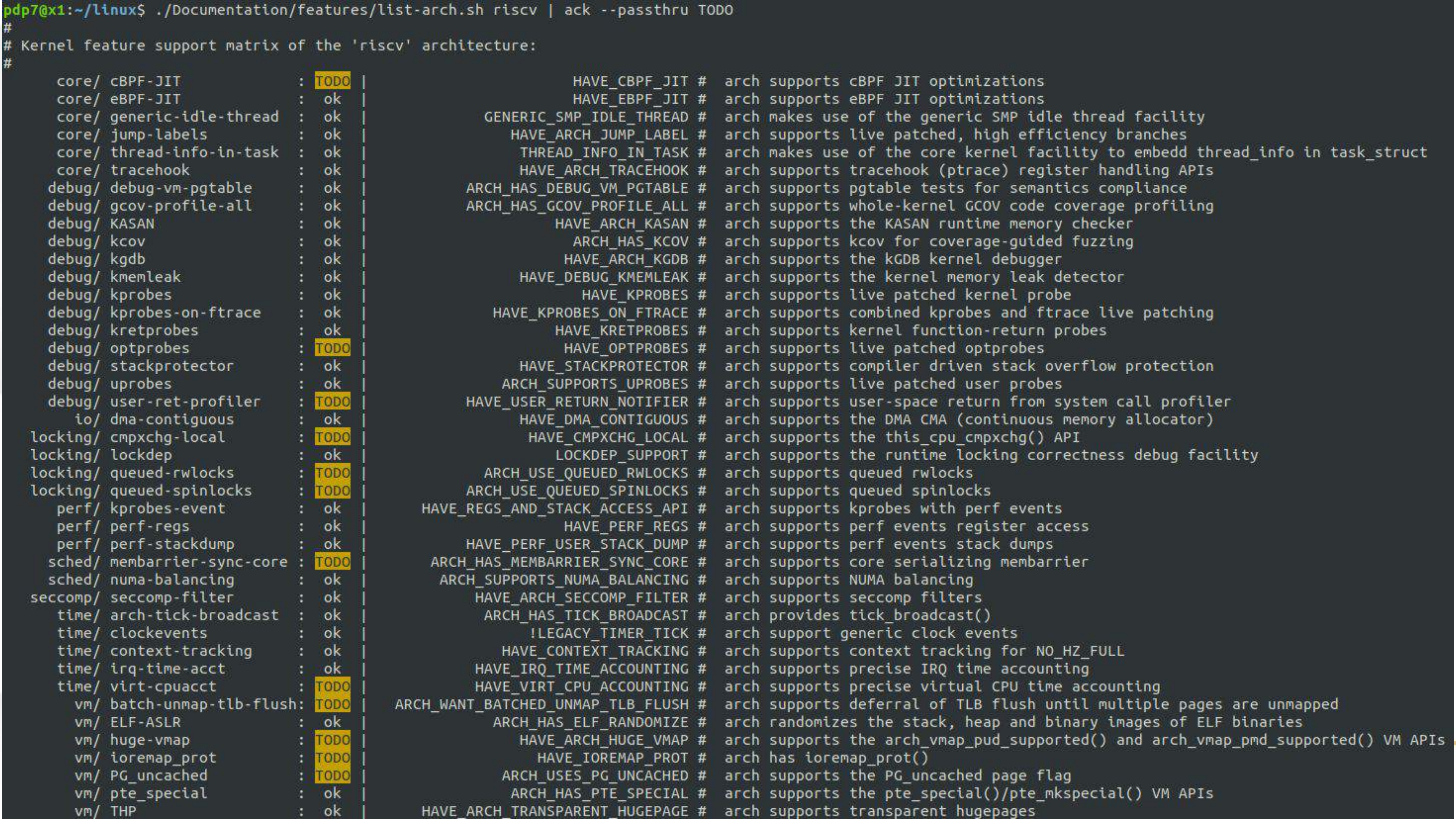
- Support for RISC-V in mainline QEMU
- Boots 32-bit and 64-bit mainline Linux kernel
- Machine configs to boot same binaries as some RISC-V dev boards
- Supports the new Hypervisor and Vector extensions

RISC-V in the Linux kernel

- Initial port by Palmer Dabbelt was merged into Linux 4.15 back in 2018
- “It’s a fun, friendly, and still pretty small community” - *Björn Töpel* [1]
- Palmer continues to maintain the riscv tree
- Development happens on the linux-riscv mailing list
- View the archive on lore.kernel.org
- IRC: #riscv on libera.chat



```
linux-riscv.lists.infradead.org archive mirror
[          ] search  help / color / mirror / Atom feed

[PATCH 0/2] riscv: implement Zicbom-based CMO instructions + the t-head variant
 2022-04-17 17:35 UTC  (10+ messages)

[PATCH V2 0/3] riscv: atomic: Optimize AMO instructions usage
 2022-04-17  6:45 UTC  (10+ messages)
` [PATCH V2 1/3] riscv: atomic: Cleanup unnecessary definition
` [PATCH V2 2/3] riscv: atomic: Optimize acquire and release for AMO operations
` [PATCH V2 3/3] riscv: atomic: Optimize memory barrier semantics of LRSC-pairs

[PATCH v3 0/7] Generic Ticket Spinlocks
 2022-04-17  2:44 UTC  (14+ messages)
` [PATCH v3 1/7] asm-generic: ticket-lock: New generic ticket-based spinlock
` [PATCH v3 2/7] asm-generic: qspinlock: Indicate the use of mixed-size atomics
` [PATCH v3 3/7] asm-generic: qrwlock: Document the spinlock fairness requirements
` [PATCH v3 4/7] openrisc: Move to ticket-spinlock
` [PATCH v3 5/7] RISC-V: Move to generic spinlocks
` [PATCH v3 6/7] RISC-V: Move to queued RW locks
` [PATCH v3 7/7] csky: Move to generic ticket-spinlock
```



```
pdp7@x1:~/linux$ ./Documentation/features/list-arch.sh riscv | ack --passthru TODO
#
# Kernel feature support matrix of the 'riscv' architecture:
#
      core/ cBPF-JIT             : TODO |                       HAVE_CBPF_JIT #  arch supports cBPF JIT optimizations
      core/ eBPF-JIT             :   ok |                       HAVE_EBPF_JIT #  arch supports eBPF JIT optimizations
      core/ generic-idle-thread  :   ok |             GENERIC_SMP_IDLE_THREAD #  arch makes use of the generic SMP idle thread facility
      core/ jump-labels          :   ok |                HAVE_ARCH_JUMP_LABEL #  arch supports live patched, high efficiency branches
      core/ thread-info-in-task  :   ok |                 THREAD_INFO_IN_TASK #  arch makes use of the core kernel facility to embedd thread_info in task_struct
      core/ tracehook            :   ok |                 HAVE_ARCH_TRACEHOOK #  arch supports tracehook (ptrace) register handling APIs
     debug/ debug-vm-pgtable     :   ok |           ARCH_HAS_DEBUG_VM_PGTABLE #  arch supports pgtable tests for semantics compliance
     debug/ gcov-profile-all     :   ok |           ARCH_HAS_GCOV_PROFILE_ALL #  arch supports whole-kernel GCOV code coverage profiling
     debug/ KASAN                :   ok |                     HAVE_ARCH_KASAN #  arch supports the KASAN runtime memory checker
     debug/ kcov                 :   ok |                       ARCH_HAS_KCOV #  arch supports kcov for coverage-guided fuzzing
     debug/ kgdb                 :   ok |                      HAVE_ARCH_KGDB #  arch supports the kGDB kernel debugger
     debug/ kmemleak             :   ok |                 HAVE_DEBUG_KMEMLEAK #  arch supports the kernel memory leak detector
     debug/ kprobes              :   ok |                        HAVE_KPROBES #  arch supports live patched kernel probe
     debug/ kprobes-on-ftrace    :   ok |              HAVE_KPROBES_ON_FTRACE #  arch supports combined kprobes and ftrace live patching
     debug/ kretprobes           :   ok |                     HAVE_KRETPROBES #  arch supports kernel function-return probes
     debug/ optprobes            : TODO |                      HAVE_OPTPROBES #  arch supports live patched optprobes
     debug/ stackprotector       :   ok |                 HAVE_STACKPROTECTOR #  arch supports compiler driven stack overflow protection
     debug/ uprobes              :   ok |               ARCH_SUPPORTS_UPROBES #  arch supports live patched user probes
     debug/ user-ret-profiler    : TODO |           HAVE_USER_RETURN_NOTIFIER #  arch supports user-space return from system call profiler
        io/ dma-contiguous       :   ok |                 HAVE_DMA_CONTIGUOUS #  arch supports the DMA CMA (continuous memory allocator)
   locking/ cmpxchg-local        : TODO |                  HAVE_CMPXCHG_LOCAL #  arch supports the this_cpu_cmpxchg() API
   locking/ lockdep              :   ok |                     LOCKDEP_SUPPORT #  arch supports the runtime locking correctness debug facility
   locking/ queued-rwlocks       : TODO |             ARCH_USE_QUEUED_RWLOCKS #  arch supports queued rwlocks
   locking/ queued-spinlocks     : TODO |           ARCH_USE_QUEUED_SPINLOCKS #  arch supports queued spinlocks
      perf/ kprobes-event        :   ok |      HAVE_REGS_AND_STACK_ACCESS_API #  arch supports kprobes with perf events
      perf/ perf-regs            :   ok |                      HAVE_PERF_REGS #  arch supports perf events register access
      perf/ perf-stackdump       :   ok |           HAVE_PERF_USER_STACK_DUMP #  arch supports perf events stack dumps
     sched/ membarrier-sync-core : TODO |       ARCH_HAS_MEMBARRIER_SYNC_CORE #  arch supports core serializing membarrier
     sched/ numa-balancing       :   ok |        ARCH_SUPPORTS_NUMA_BALANCING #  arch supports NUMA balancing
   seccomp/ seccomp-filter       :   ok |            HAVE_ARCH_SECCOMP_FILTER #  arch supports seccomp filters
      time/ arch-tick-broadcast  :   ok |             ARCH_HAS_TICK_BROADCAST #  arch provides tick_broadcast()
      time/ clockevents          :   ok |                  !LEGACY_TIMER_TICK #  arch support generic clock events
      time/ context-tracking     :   ok |               HAVE_CONTEXT_TRACKING #  arch supports context tracking for NO_HZ_FULL
      time/ irq-time-acct        :   ok |            HAVE_IRQ_TIME_ACCOUNTING #  arch supports precise IRQ time accounting
      time/ virt-cpuacct         : TODO |            HAVE_VIRT_CPU_ACCOUNTING #  arch supports precise virtual CPU time accounting
        vm/ batch-unmap-tlb-flush: TODO |   ARCH_WANT_BATCHED_UNMAP_TLB_FLUSH #  arch supports deferral of TLB flush until multiple pages are unmapped
        vm/ ELF-ASLR             :   ok |              ARCH_HAS_ELF_RANDOMIZE #  arch randomizes the stack, heap and binary images of ELF binaries
        vm/ huge-vmap            : TODO |                 HAVE_ARCH_HUGE_VMAP #  arch supports the arch_vmap_pud_supported() and arch_vmap_pmd_supported() VM APIs
        vm/ ioremap_prot         : TODO |                   HAVE_IOREMAP_PROT #  arch has ioremap_prot()
        vm/ PG_uncached          : TODO |               ARCH_USES_PG_UNCACHED #  arch supports the PG_uncached page flag
        vm/ pte_special          :   ok |                ARCH_HAS_PTE_SPECIAL #  arch supports the pte_special()/pte_mkspecial() VM APIs
        vm/ THP                  :   ok |      HAVE_ARCH_TRANSPARENT_HUGEPAGE #  arch supports transparent hugepages
```

Recently added to Linux

- KVM RISC-V support by Anup Patel added in Linux 5.16
  - Add KVM support for the Hypervisor specification
- SBI SRST extension support by Anup Patel added in Linux 5.17
  - Support for the SBI SRST (system reset) extension which allows systems that do not have an explicit driver in Linux to reboot

New in Linux 5.18

- Add Sv57 page table support by Qinglin Pan
  - use 5-level page table to support Sv57 which expands the virtual address space to 57 bits (128 petabytes)
- Improve RISC-V Perf support by Atish Patra
  - Recent talk: Perf on RISC-V: The Past, the Present and the Future (slides)

New in Linux 5.18

- RISC-V CPU Idle support by Anup Patel
  - cpuidle and suspend drivers now support the SBI HSM extension
- Provide framework for RISC-V ISA extensions by Atish Patra
  - Linux was no longer correctly parsing the RISC-V ISA string as the number of RISC-V extensions has grown and extension names are no longer a single character
  - This series implements a generic framework to parse multi-letter ISA extensions.
  - Based on initial work by Tsukasa OI

Coming in Linux 5.19…

- RISC-V Patches for the 5.19 Merge Window, Part 1 - Palmer *(2022-05-31)*
  - Support for page-based memory attributes <*we'll dive into that topic in a few slides*>
  - Support for running rv32 binaries on rv64 systems via the compat subsystem
  - Support for kexec_file()
  - Support new generic ticket-based spinlocks, which allows us to also move to qrwlock
  - A handful of cleanups and fixes, include some larger ones around atomics and XIP
- Part 2? Follow linux-riscv and look at Palmer's riscv/for-next branch

Work in progress

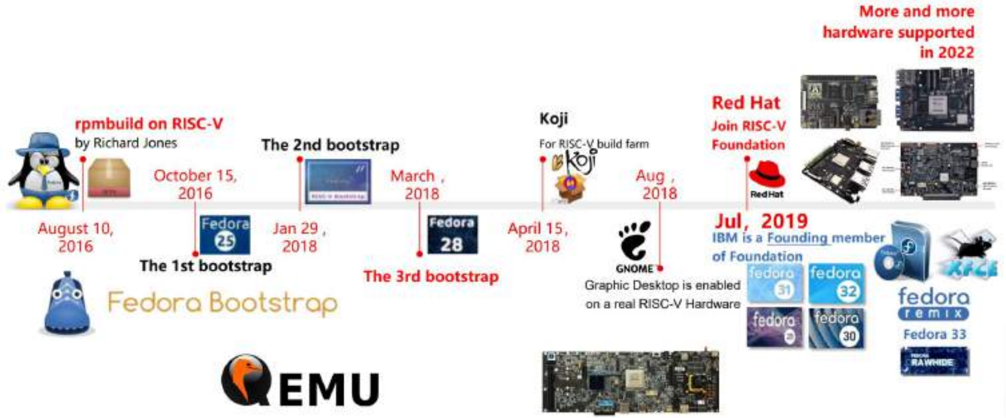
- [PATCH v10 00/16] riscv: Add vector ISA support by Greentime Hu
  - Vector ISA support based on the ratified Vector 1.0 extension
  - Defines new structure `__riscv_v_state` in struct `thread_struct` to save/restore the vector related registers. It is used for both kernel space and user space.
- [PATCH v6 0/7] RISC-V IPI Improvements by Anup Patel
  - Traditionally, RISC-V S-mode software like the Linux kernel calls to into M-mode runtime firmware like OpenSBI to issue IPIs (inter-processor interrupts)
  - AIA (advanced interrupt architecture) provides the ability for S-mode to issue IPIs without any assistance from M-mode. This improves efficiency.

Work in progress

- [PATCH v4 0/4] Add Sstc extension support by Atish Patra
  - Traditionally, an SBI call is necessary to generate timer interrupts as S-mode does not have access to the M-mode time compare registers.
  - This results in significant latency for the kernel to generate timer interrupts at kernel
  - For virtualized environments, it’s even worse as the KVM handles the SBI call and uses a software timer to emulate the time compare register.
  - Sstc extension allows kernel to program a timer and receive interrupt without supervisor execution environment (M-mode/HS mode) intervention.

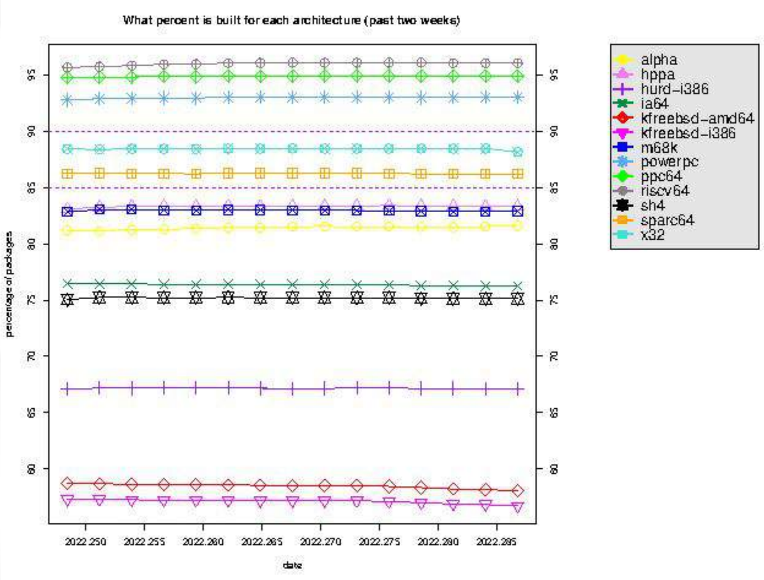
Linux distro: Fedora

- Aims to provide a complete Fedora experience on RISC-V
- Talk by Wei Fu, RISC-V Ambassador and Red Hat engineer [slides]
- Installation instructions for QEMU and RISC-V dev boards



Linux distro: Debian

- riscv64 is port of Debian to RISC-V
    - “a port in Debian terminology means to provide the software normally available in the Debian archive (over 20,000 source packages) ready to install and run”
- 95% of packages are built for RISC-V
    - The Debian port uses RV64GC as the hardware baseline and the lp64d ABI



Linux distro: Ubuntu

- riscv64 supported since the release of Ubuntu 20.04 LTS.
- Ubuntu 22.04 pre-installed SD-card image for SiFive boards and QEMU
- Starting with Ubuntu 22.04, a server install image is made available to install Ubuntu on NVMe drive of the SiFive Unmatched board

Linux distros

- OpenSuSE
    - RISC-V support is still under development with Tumbleweed images for some boards
- Arch Linux
    - Community effort has 95% of core packages building for RISC-V
- Gentoo
    - riscv64 stages are available on the Gentoo download page

OpenEmbedded and Yocto

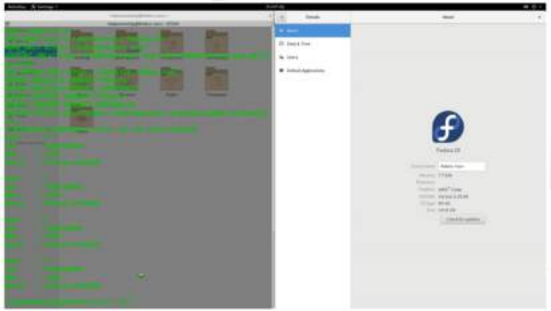
- meta-riscv: general hardware-specific BSP overlay for RISC-V devices
  - works with different OpenEmbedded/Yocto distributions and layer stacks
  - Supports both QEMU and RISC-V dev boards

BuildRoot

- RISC-V port is now supported in the upstream BuildRoot project
- “Embedded Linux from scratch in 45 minutes (on RISC-V)”
  - Tutorial by Michael Opdenacker at FOSDEM 2021
  - Use Buildroot to compile OpenSBI, U-Boot, Linux and BusyBox
  - Boot the system in QEMU

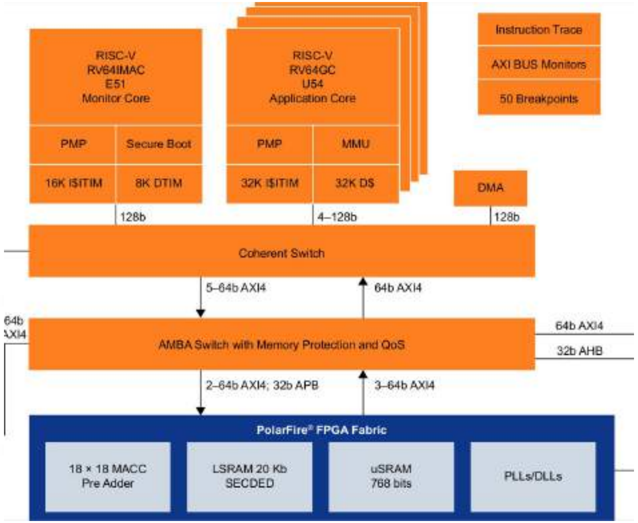
SiFive Freedom Unleashed

- Launched in 2018 as first Linux-capable RISC-V dev board
- Exciting to see Fedora GNOME desktop on RISC-V
- $999 was too expensive for widespread adoption
- FU540 SoC chip was never sold separately



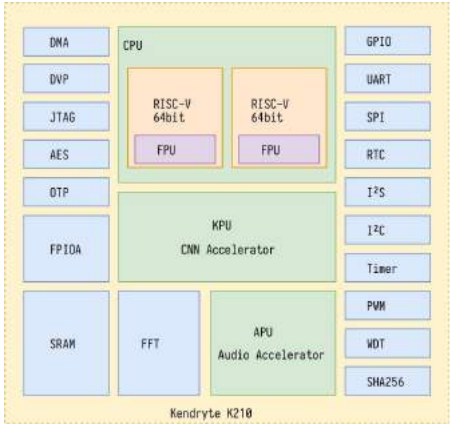
Microchip PolarFire SoC

- Same RISC-V cores as the SiFive FU540 SoC but adds a FPGA fabric
    - FPGA with 25k to 460k logic elements (LEs)
    - Supports DDR4 and PCIe Gen2
- Full commercial product family
    - Parts will be available from distributors
- Microchip Icicle dev board ($499)



Kendryte K210

- 400 MHz dual core RV64GC
- 8MB SRAM but **no DRAM**
- Dev boards starting at $14
- Upstream support in Linux and u-boot
- Buildroot support by Damien Le Moal can create a busybox-based rootfs



SiFive Unmatched

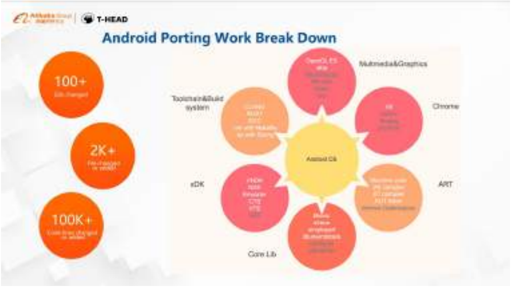
- SiFive Freedom FU740 SoC
  - 4x U74 RV64GC application cores
- Mini-ITX PC form factor
  - 16GB DDR4, 4x USB 3.2, one x16 PCIe slot
  - M.2 for NVMe SSD and WiFi/Bluetooth
- Shipped in 2021 for $665
  - Discontinued in 2022 to focus on next-gen

T-Head XuanTie C910

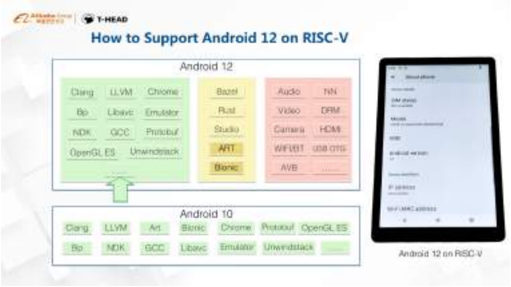
- T-Head ("píng tóu gē") is part of Alibaba
- High performance RV64GC with up 16 cores
  - 12-stage pipeline, out-of-order, multi-issue architecture
  - comparable to Arm Cortex-A73
- 2 core 'ICE' SoC made in low qty for evaluation
- T-Head ported Android 10 (AOSP) to RISC-V and showed a demo on the ICE in early 2021

Porting Android to RISC-V

- Mao Han presented why and how Alibaba T-Head ported Android to RISC-V *(jump to 4:32)*
  - PDF of slides
- Update on Android 12 from April *(jump to 13:23)*
  - PDF of slides
- How Alibaba is Porting RISC-V to the Android OS blog post with more technical details


T-HEAD

Android Porting Work Break Down


T-HEAD

How to Support Android 12 on RISC-V

Android 12 on RISC-V

T-HEAD

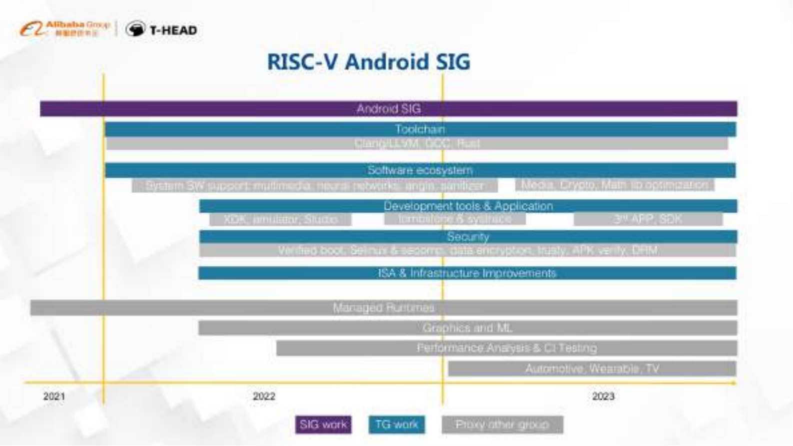
Running TF lite on RISC-V

T-Head RVB-ICE dev board

- ‘ICE’ SoC featuring dual C910 core at 1.2GHz
- 4GB LPDDR4, 16GB eMMC, 7 inch touchscreen, WiFi, Gigabit Ethernet
- Produced in limited quantity and available on AliExpress for $399

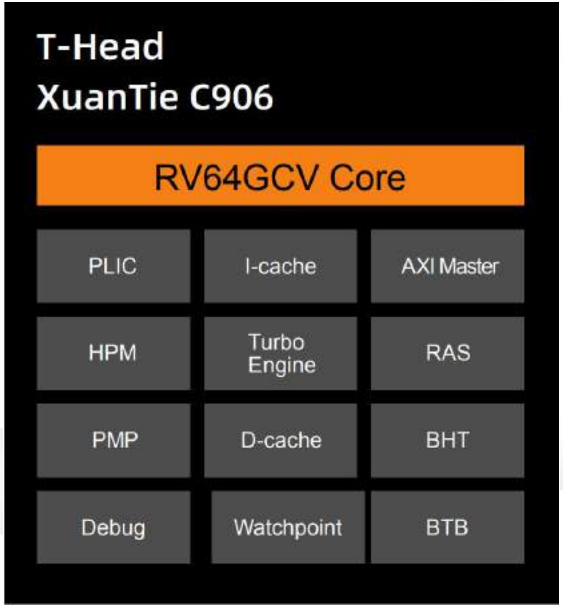
RISC-V Android SIG *(Special Interest Group)*

- GitHub organization riscv-android-src “contains all the modified AOSP(Android open source project) repositories with RISC-V support”
- Instruction to build and run Android 12 on RISC-V



T-Head XuanTie C906

- single-core RV64GC, only up to 1GHz, simpler 5-stage in-order pipeline



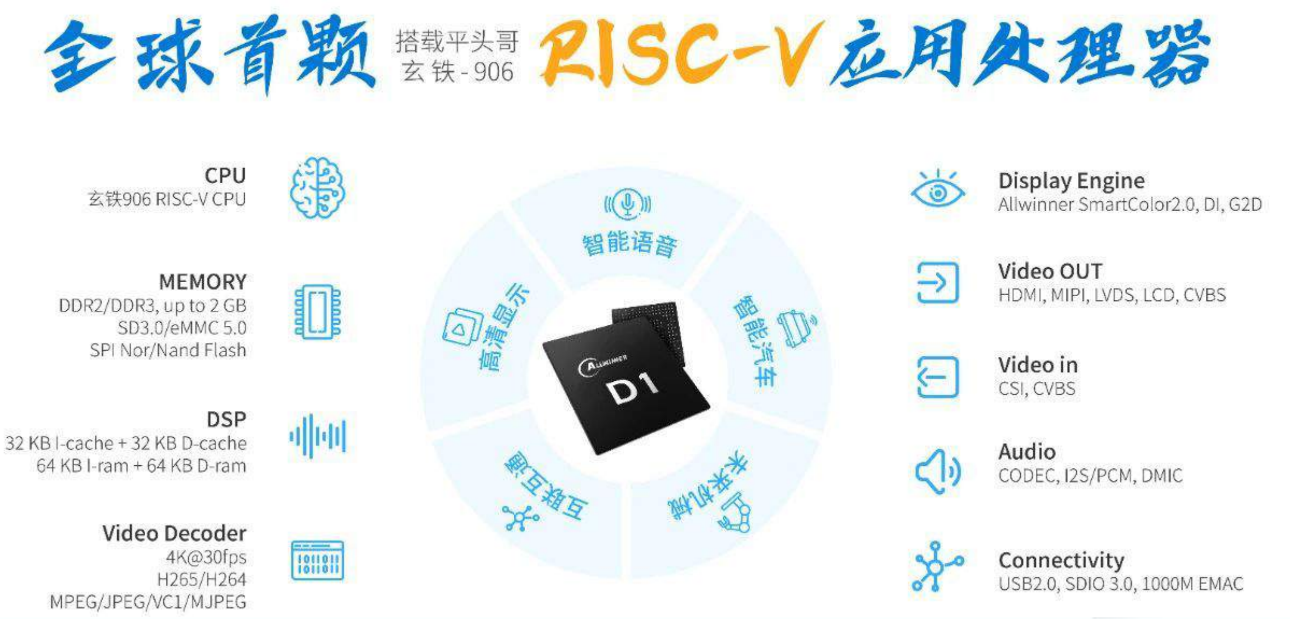
T-Head
XuanTie C906

RV64GCV Core

PLIC	I-cache	AXI Master
HPM	Turbo Engine	RAS
PMP	D-cache	BHT
Debug	Watchpoint	BTB

Allwinner D1 SoC

- mass production low cost SoC with a single T-Head C906 core at 1 GHz



全球首颗 搭载平头哥 玄铁-906 RISC-V应用处理器

CPU
玄铁906 RISC-V CPU

MEMORY
DDR2/DDR3, up to 2 GB
SD3.0/eMMC 5.0
SPI Nor/Nand Flash

DSP
32 KB I-cache + 32 KB D-cache
64 KB I-ram + 64 KB D-ram

Video Decoder
4K@30fps
H265/H264
MPEG/JPEG/VC1/MJPEG

Display Engine
Allwinner SmartColor2.0, DI, G2D

Video OUT
HDMI, MIPI, LVDS, LCD, CVBS

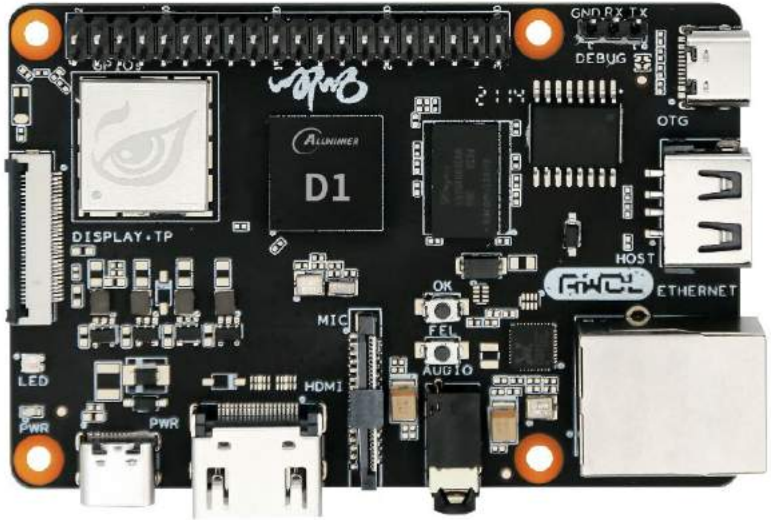
Video in
CSI, CVBS

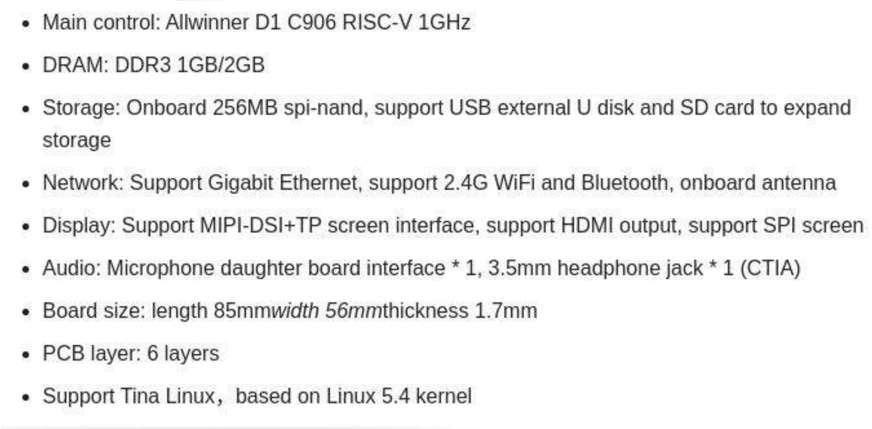
Audio
CODEC, I2S/PCM, DMIC

Connectivity
USB2.0, SDIO 3.0, 1000M EMAC

Allwinner Nezha D1 dev board

- Official D1 board made by Allwinner Online, $115 bundle on AliExpress





- Main control: Allwinner D1 C906 RISC-V 1GHz
- DRAM: DDR3 1GB/2GB
- Storage: Onboard 256MB spi-nand, support USB external U disk and SD card to expand storage
- Network: Support Gigabit Ethernet, support 2.4G WiFi and Bluetooth, onboard antenna
- Display: Support MIPI-DSI+TP screen interface, support HDMI output, support SPI screen
- Audio: Microphone daughter board interface * 1, 3.5mm headphone jack * 1 (CTIA)
- Board size: length 85mm*width 56mm*thickness 1.7mm
- PCB layer: 6 layers
- Support Tina Linux，based on Linux 5.4 kernel

RISC-V Developer Boards

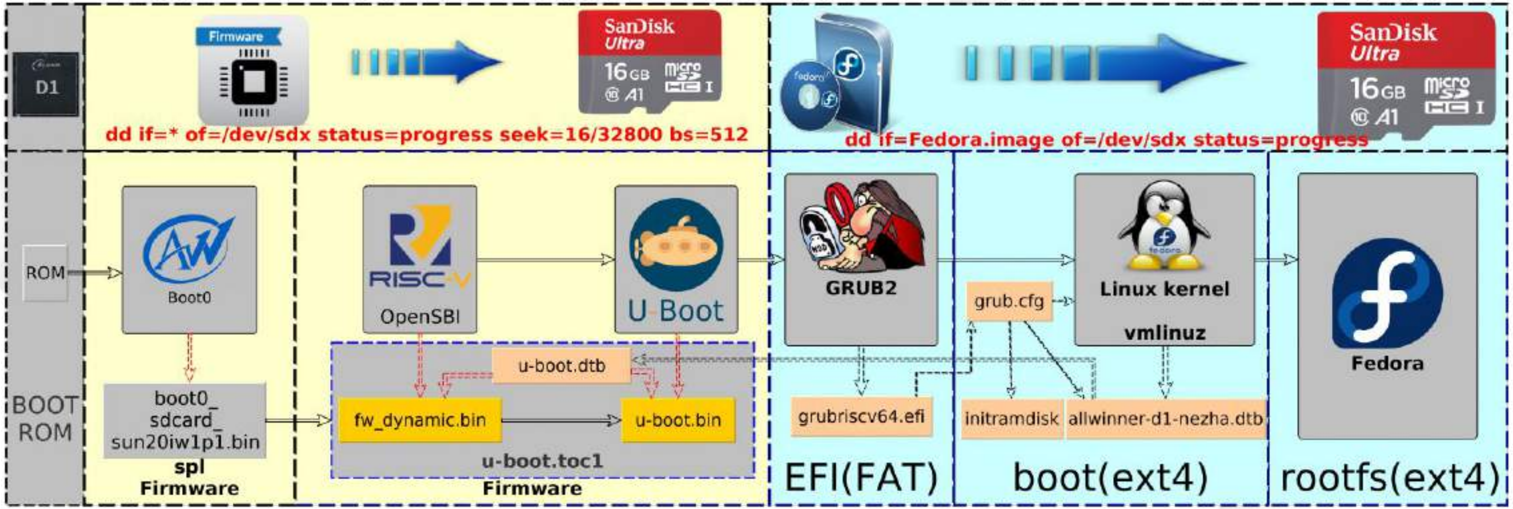
- Initiative from RISC-V International to get Linux-capable boards into the hands of open source developers
    - Launched in 2021 with the Allwinner D1 Nezha and SiFive Unmatched
- Fill out this form to apply
    - Need to be RISC-V International member (or part of a member organization), but remember that individuals can join RISC-V International free of cost
    - Explain why you are interested in RISC-V and what you plan to do with dev board
    - To improve your chances, don’t overestimate your hardware requirements like RAM

Allwinner D1 open source community

- linux-sunxi: strong open source community for Allwinner SoCs
  - D1 wiki page and Allwinner Nezha board wiki page
- Telegram group: Mainline Linux for D1 (190 members)
- Samuel Holland has been working on getting mainline to run
  - U-Boot SPL: https://github.com/smaeul/sun20i_d1_spl
  - OpenSBI: https://github.com/smaeul/opensbi
  - U-Boot: https://github.com/smaeul/u-boot/tree/d1-wip
  - Linux: https://github.com/smaeul/linux/tree/riscv/d1-wip

Fedora on Allwinner D1

- Wei Fu has created a Fedora “Rawhide” image for the Allwinner D1 Nezha dev board that includes the XFCE desktop environment

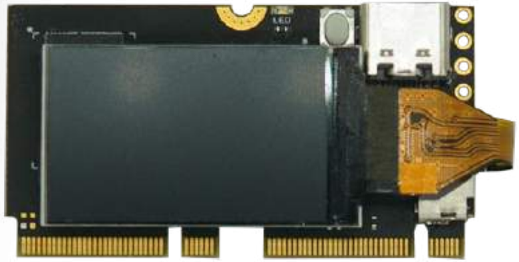


```
dd if=* of=/dev/sdx status=progress seek=16/32800 bs=512
```

```
dd if=Fedora.image of=/dev/sdx status=progress
```

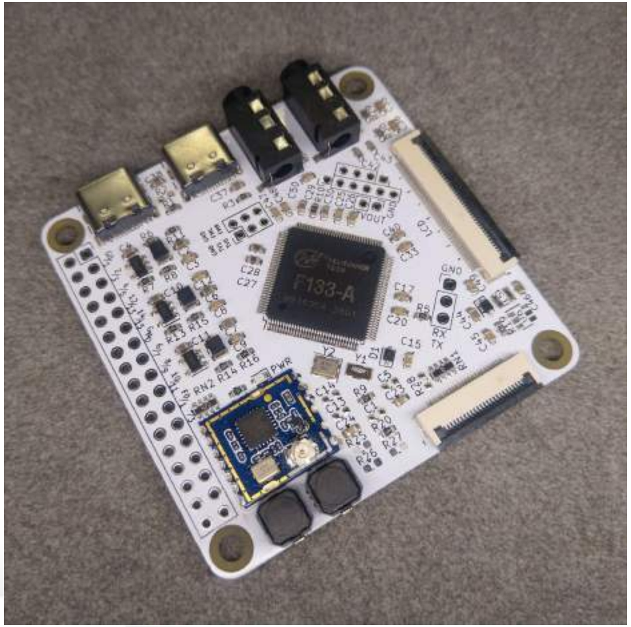
Lichee RV-Nezha CM

- Allwinner D1 with DDR3 RAM of either 512MB ($22) or 1GB ($32)
- 1.14" SPI LCD, USB Type-C OTG, uSD
- Lichee RV Dock: HDMI out, USB-A host port, WiFi+BT, mic, audio out
- Lichee RV 86 Panel: 8 inch screen
- More details on sunxi wiki



Xassette Asterisk

- Allwinner F133 combines the Allwinner D1 with 64MB DDR2 in a single package
- Board design published as Open Source Hardware under the CERN OHL-w v2 licence
- Designed with KiCad (open source CAD sw)
- Not available commercially but possible to be hand assembled by hobbyists



MangoPi-Nezha MQ

- Allwinner F133 (also known as D1s)
- WiFi, USB Type C, mic, audio out
- DSI and RGB display connectors
- Open source hardware: KiCad files on GitHub
- $39 on Crowd Supply

Allwinner D1 mainline Linux support

- Allwinner reused peripheral IP from their existing ARM SoCs and the Linux kernel already has drivers for most of them
- T-Head cores like C910 and C906 do have some non-standard functionality for performance but it’s not needed to boot
  - Instructions to accelerate I-cache and TLB synchronization
- T-Head MMU has a non-standard ‘enhanced’ mode that is needed to support DMA with devices on non-coherent interconnects
  - Linux needs to enable that ‘enhanced’ MMU mode to function properly

How to handle non-coherent interconnects?

- The original RISC-V Privileged spec stated that “the use of hardware-incoherent regions is discouraged due to software complexity, performance, and energy impacts”
- However, non-coherent interconnects are important for low cost SoCs
- T-Head designed the C9xx cores in 2019, and there were no RISC-V extensions that provided ability to handle non-coherent devices

T-Head PTE format

- T-Head used bits in the PTE (Page Table Entry) to specify memory type

```
| 63 | 62 | 61 | 60 | 59-8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0
  SO   C    B    SH   RSW    D   A   G   U   X   W   R   V
  ^    ^    ^    ^

BIT(63): SO - Strong Order
BIT(62): C  - Cacheable
BIT(61): B  - Bufferable
BIT(60): SH - Shareable

0000 - NC   Weakly-ordered, Non-cacheable, Non-bufferable, Non-shareable
0111 - PMA  Weakly-ordered, Cacheable, Bufferable, Shareable
1000 - IO   Strongly-ordered, Non-cacheable, Non-bufferable, Non-shareable
```

… but those bits were already marked reserved in RISC-V priv spec

Page-Based Memory Types extension

- **Svpbmt** proposed by Virtual Memory TG and ratified at the end of 2021
  - *"S" = supervisor-mode (privileged architecture), "v" for virtual memory*

```
Here is the svpbmt PTE format:
| 63 | 62-61 | 60-8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0
  N      MT      RSW    D   A   G   U   X   W   R   V
         ^

RISC-V
Encoding &
MemType     RISC-V Description
----------  ------------------------------------------------
00 - PMA    Normal Cacheable, No change to implied PMA memory type
01 - NC     Non-cacheable, idempotent, weakly-ordered Main Memory
10 - IO     Non-cacheable, non-idempotent, strongly-ordered I/O memory
11 - Rsvd   Reserved for future standard use
```

Svpbmt support in Linux

- [PATCH v10 00/12] riscv: support for Svpbmt and D1 memory types
  - by Heiko Stuebner based on initial work by Alibaba kernel engineer Guo Ren
  - Implements the official RISC-V Svpbmt extension
  - The standard Svpbmt and custom T-Head PTE formats both use the highest bits to determine memory type but the encoding and semantics are different
  - The custom T-Head PTE format is supported through boot-time code patching using the Linux Alternatives Framework
  - Expected to land in Linux 5.19 as it is in Palmer's pull request

Cache Management Operations

- Instructions to manage cache are important for SoCs which that lack cache coherent interconnects

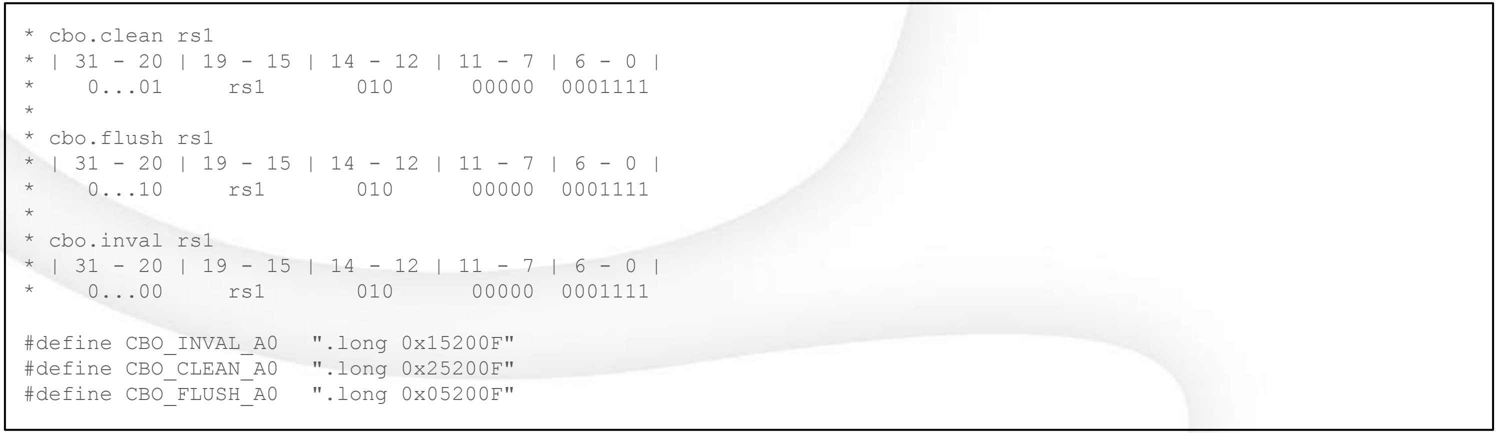
- Zicbom extension *("Z" prefix means Unpriv spec)* was ratified at the end of 2021, and it defines cache-block management (CBO) instructions:
  - CBO.CLEAN guarantee store by hart can be read from mem by non-coherent device
  - CBO.INVAL guarantee hart can load data written to memory by non-coherent device
  - CBO.FLUSH guarantees both

- Support for Non-Coherent I/O Devices in RISC-V from RV Summit [slides]

CMO support in Linux

- riscv: implement Zicbom-based CMO instructions + the t-head variant by Heiko Stuebner

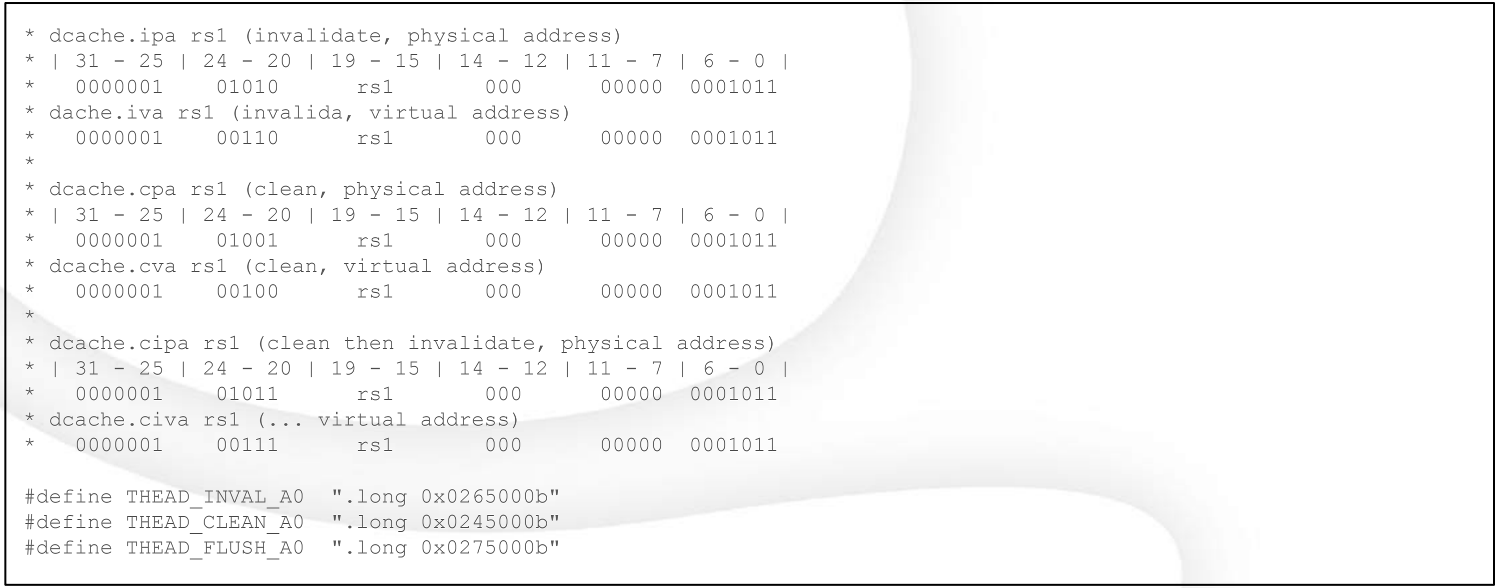
- Implements Zicbom-extension to handle cache clean, invalidate, flush



```
* cbo.clean rs1
* | 31 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*    0...01     rs1       010      00000   0001111
*
* cbo.flush rs1
* | 31 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*    0...10     rs1       010      00000   0001111
*
* cbo.inval rs1
* | 31 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*    0...00     rs1       010      00000   0001111

#define CBO_INVAL_A0   ".long 0x15200F"
#define CBO_CLEAN_A0   ".long 0x25200F"
#define CBO_FLUSH_A0   ".long 0x05200F"
```

CMO support in Linux

- T-Head implemented custom cache instructions before Zicbom existed



```
* dcache.ipa rs1 (invalidate, physical address)
* | 31 - 25 | 24 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*   0000001    01010      rs1       000      00000  0001011
* dache.iva rs1 (invalida, virtual address)
*   0000001    00110      rs1       000      00000  0001011
*
* dcache.cpa rs1 (clean, physical address)
* | 31 - 25 | 24 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*   0000001    01001      rs1       000      00000  0001011
* dcache.cva rs1 (clean, virtual address)
*   0000001    00100      rs1       000      00000  0001011
*
* dcache.cipa rs1 (clean then invalidate, physical address)
* | 31 - 25 | 24 - 20 | 19 - 15 | 14 - 12 | 11 - 7 | 6 - 0 |
*   0000001    01011      rs1       000      00000  0001011
* dcache.civa rs1 (... virtual address)
*   0000001    00111      rs1       000      00000  0001011

#define THEAD_INVAL_A0  ".long 0x0265000b"
#define THEAD_CLEAN_A0  ".long 0x0245000b"
#define THEAD_FLUSH_A0  ".long 0x0275000b"
```

CMO support in Linux

- While the Zicbom and T-Head instructions are different, they provide the same functionality, so the T-Head variant handled with the existing alternatives mechanism

- Allwinner D1 needs these cache instructions for peripherals like MMC (SD card), USB, and Ethernet to work

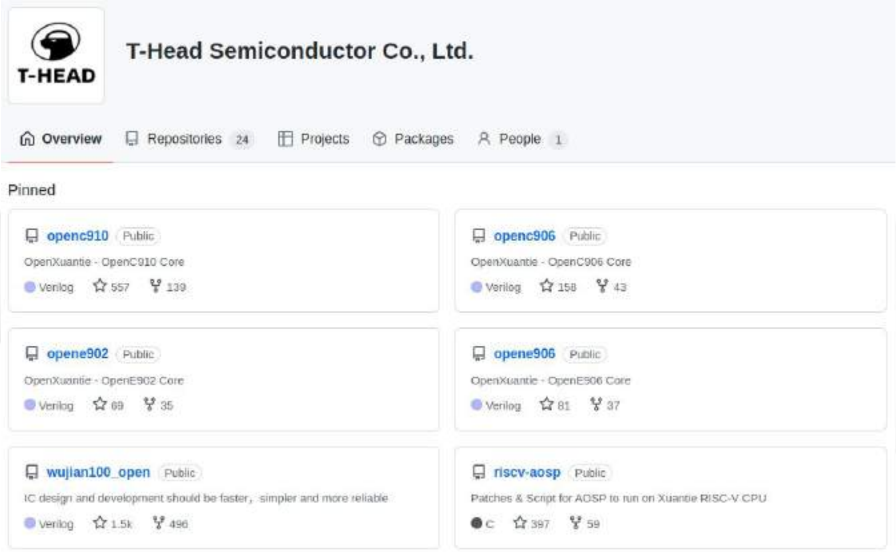
- Unfortunately, these patches use pre-coded CMO instructions and Palmer would prefer that Linux support wait until the instructions are added to gcc and binutils

Allwinner D1 IOMMU support

- [PATCH 0/5] iommu/sun50i: Allwinner D1 support by Samuel Holland
- IOMMU is not needed for boot
- Optional feature for the display engine and video decoder
- Without IOMMU support, video/frame buffers have to be contiguous in physical memory, and that requires the user to know how much memory to reserve for them at boot

T-Head released RISC-V cores as open source!

- OpenE902, OpenE906, OpenC906, and OpenC910 cores on GitHub under permissive Apache 2.0 licence



T-Head Semiconductor Co., Ltd.

Pinned

openc910 Public
OpenXuantie - OpenC910 Core
Verilog ☆ 557 139

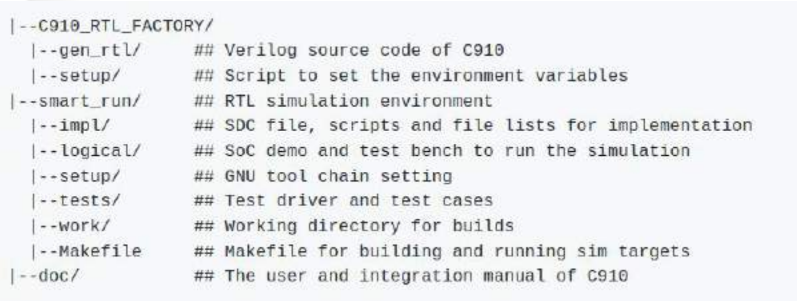
openc906 Public
OpenXuantie - OpenC906 Core
Verilog ☆ 158 43

opene902 Public
OpenXuantie - OpenE902 Core
Verilog ☆ 69 35

opene906 Public
OpenXuantie - OpenE906 Core
Verilog ☆ 81 37

wujian100_open Public
IC design and development should be faster，simpler and more reliable
Verilog ☆ 1.5k 496

riscv-aosp Public
Patches & Script for AOSP to run on Xuantie RISC-V CPU
● C ☆ 397 59



```
|--C910_RTL_FACTORY/
  |--gen_rtl/      ## Verilog source code of C910
  |--setup/        ## Script to set the environment variables
|--smart_run/      ## RTL simulation environment
  |--impl/         ## SDC file, scripts and file lists for implementation
  |--logical/      ## SoC demo and test bench to run the simulation
  |--setup/        ## GNU tool chain setting
  |--tests/        ## Test driver and test cases
  |--work/         ## Working directory for builds
  |--Makefile      ## Makefile for building and running sim targets
|--doc/            ## The user and integration manual of C910
```

XiangShan (香山)

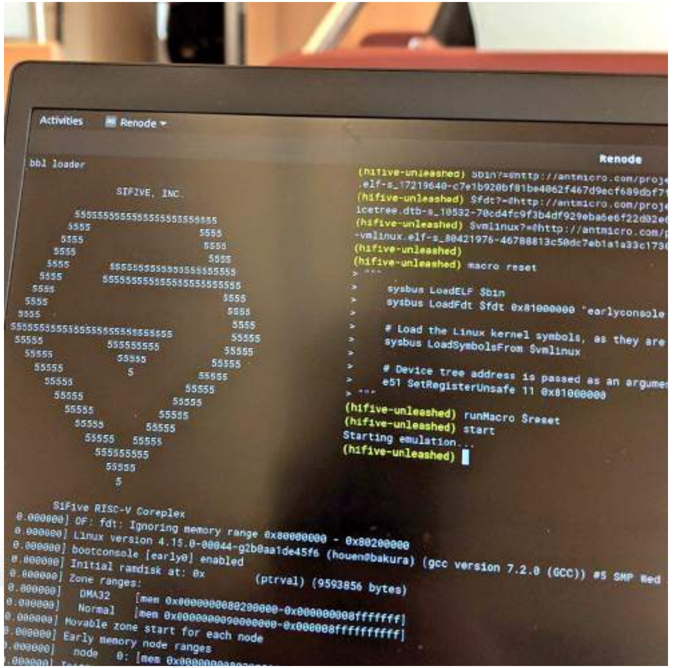
- open source high-performance RISC-V processor project from the Chinese Academy of Science
- RISC-V Summit 2021 talk by Professor Yungang Bao (slides)
  - “Contribute to XiangShan and realize your ideas on real chips! The open-source XiangShan will be taped-out every ~6 months”
- Nanhu is the 2nd generation microarchitecture
  - Target: 2GHz@14nm, SPEC CPU2006 20 marks; 7.81 CoreMark/MHz

RISC-V Lab

- PLCT Lab at Chinese Academy of Sciences
    - Status report from lab director Wei Wu
- Continuous Integration (CI) for open source software projects on RISC-V hardware
    - 70+ SiFive Unmatched boards
    - 100+ Allwinner Nezha D1 boards
    - Open source devs can request access

No hardware? Try Renode!

- Emulate physical hardware systems including CPU, peripherals, sensors, and networking
- Run the same binaries as the real hardware for over 30 supported dev boards
  - ○ Microchip PolarFire SoC Icicle Kit
  - ○ Kendryte K210
  - ○ SiFive HiFive Unleashed



```
bbl loader

SIFIVE, INC.

SiFive RISC-V Coreplex
0.000000] OF: fdt: Ignoring memory range 0x80000000 - 0x80200000
0.000000] Linux version 4.15.0-00044-g2b0aa1de45f6 (houen@bakura) (gcc version 7.2.0 (GCC)) #5 SMP Wed
0.000000] bootconsole [early0] enabled
0.000000] Initial ramdisk at: 0x              (ptrval) (9593856 bytes)
0.000000] Zone ranges:
0.000000]   DMA32    [mem 0x0000000080200000-0x00000000ffffffff]
0.000000]   Normal   [mem 0x0000000090000000-0x000008ffffffffff]
0.000000] Movable zone start for each node
0.000000] Early memory node ranges
.000000]   node   0: [mem 0x00000000
```

```
(hifive-unleashed) $bin?=@http://antmicro.com/proje
.elf-s_17219640-c7e1b920bf81be4062f467d9ecf689dbf7
(hifive-unleashed) $fdt?=@http://antmicro.com/proje
icetree.dtb-s_10532-70cd4fc9f3b4df929eba6e6f22d02e
(hifive-unleashed) $vmlinux?=@http://antmicro.com/p
-vmlinux.elf-s_80421976-46788813c50dc7eb1a1a33c173
(hifive-unleashed)
(hifive-unleashed) macro reset
> """
>     sysbus LoadELF $bin
>     sysbus LoadFdt $fdt 0x81000000 "earlyconsole
>
>     # Load the Linux kernel symbols, as they are
>     sysbus LoadSymbolsFrom $vmlinux
>
>     # Device tree address is passed as an argume
>     e51 SetRegisterUnsafe 11 0x81000000
> """
(hifive-unleashed) runMacro $reset
(hifive-unleashed) start
Starting emulation...
(hifive-unleashed) █
```

How to get involved with RISC-V International?

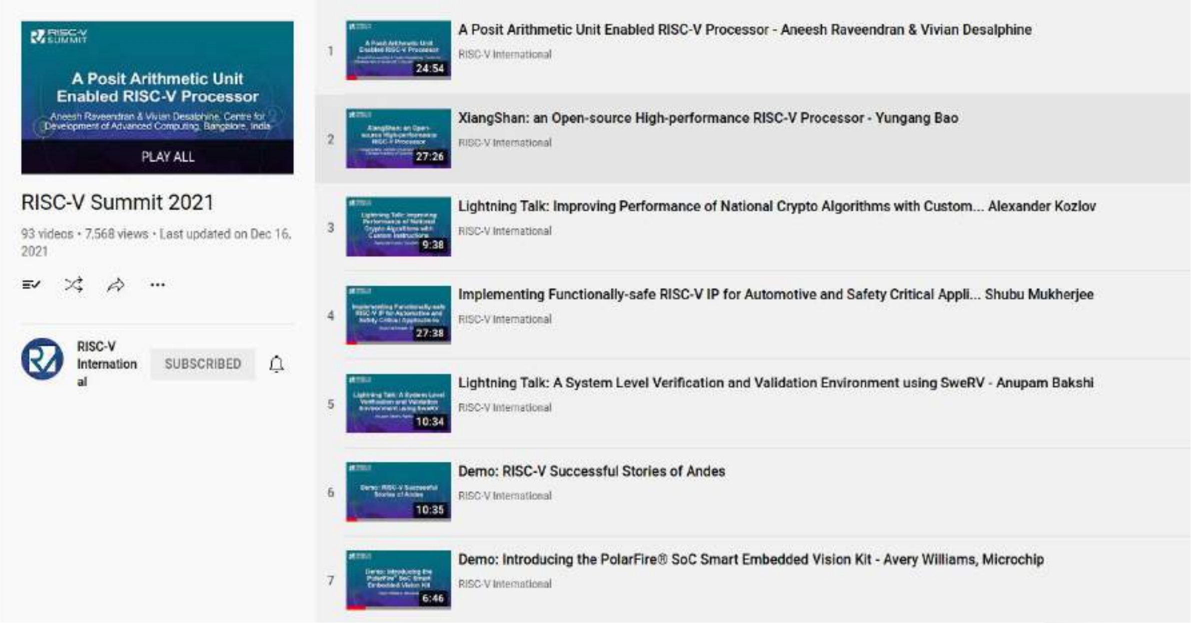
- Become a member
  - Individuals and non-profits can join free of cost
- RISC-V Technical wiki landing page is the single best place to visit
  - Technical Working Groups
  - Recently Ratified Extensions

How to get involved with RISC-V International?

- RISC-V mailing list server
  - Only RISC-V members can participate
  - Archives of all the lists are public
- Technical Meetings Calendar
  - Many groups have bi-weekly or monthly meetings
  - ICS File / Google Calendar

RISC-V Summit 2021

- YouTube playlist with 93 talks



A Posit Arithmetic Unit Enabled RISC-V Processor

Aneesh Raveendran & Vivian Desalphine, Centre for Development of Advanced Computing, Bangalore, India

PLAY ALL

RISC-V Summit 2021

93 videos • 7,568 views • Last updated on Dec 16, 2021

RISC-V International SUBSCRIBED

1 A Posit Arithmetic Unit Enabled RISC-V Processor - Aneesh Raveendran & Vivian Desalphine 24:54
RISC-V International

2 XiangShan: an Open-source High-performance RISC-V Processor - Yungang Bao 27:26
RISC-V International

3 Lightning Talk: Improving Performance of National Crypto Algorithms with Custom... Alexander Kozlov 9:38
RISC-V International

4 Implementing Functionally-safe RISC-V IP for Automotive and Safety Critical Appli... Shubu Mukherjee 27:38
RISC-V International

5 Lightning Talk: A System Level Verification and Validation Environment using SweRV - Anupam Bakshi 10:34
RISC-V International

6 Demo: RISC-V Successful Stories of Andes 10:35
RISC-V International

7 Demo: Introducing the PolarFire® SoC Smart Embedded Vision Kit - Avery Williams, Microchip 6:46
RISC-V International

RISC-V Spring Week 2022

- Videos on the RISC-V YouTube channel
- State of the Union & the Road Ahead
- Maturing the RISC-V Ecosystem
- Evolving the Role of Software in the RISC-V Ecosystem
- RISC-V IOMMU Architecture Overview

RISC-V®

State of the Union
&
The Road Ahead
Spring 2022 RISC-V Week

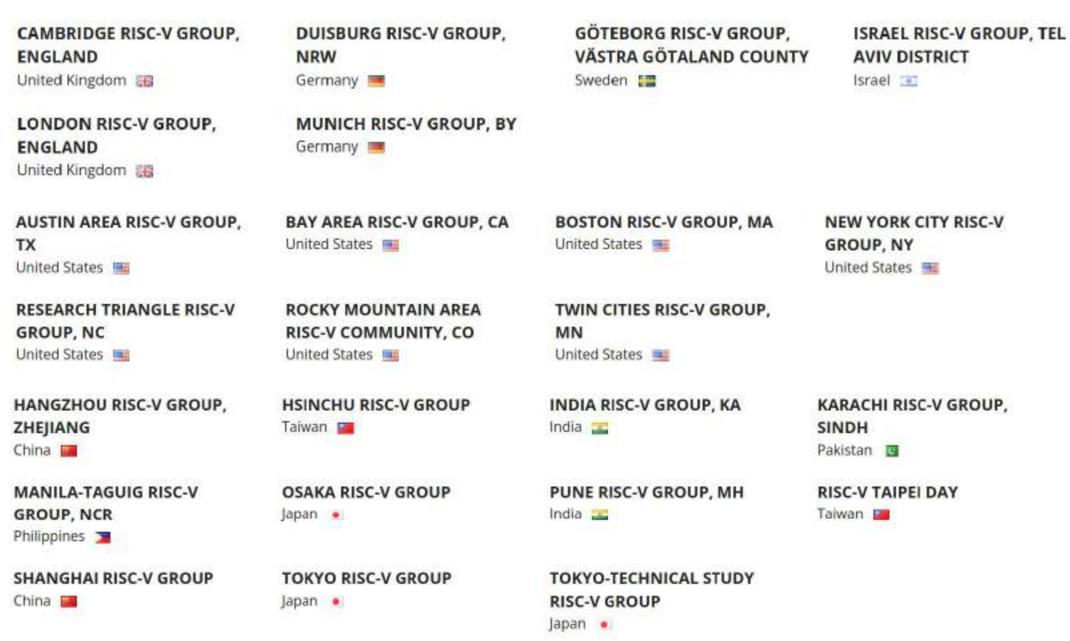
Mark Himelstein
CTO, RISC-V

@risc_v

Embedded Linux Conf 2021

- Initializing RISC-V A Guided Tour for ARM Developers
  - Ahmad Fatoum & Rouven Czerwinski, Pengutronix
- Building a Low-key XIP-enabled RISC-V Linux System
  - Vitaly Vul, Konsulko AB
- Perf on RISC-V: The Past, the Present and the Future
  - Atish Patra & Anup Patel, Western Digital
- "A New user(space): Adding RISC-V Support to Zephyr RTOS" [slides]
  - Kevin Hilman & Alexandre Mergnat, BayLibre

RISC-V meetups around the world



CAMBRIDGE RISC-V GROUP, ENGLAND
United Kingdom

DUISBURG RISC-V GROUP, NRW
Germany

GÖTEBORG RISC-V GROUP, VÄSTRA GÖTALAND COUNTY
Sweden

ISRAEL RISC-V GROUP, TEL AVIV DISTRICT
Israel

LONDON RISC-V GROUP, ENGLAND
United Kingdom

MUNICH RISC-V GROUP, BY
Germany

AUSTIN AREA RISC-V GROUP, TX
United States

BAY AREA RISC-V GROUP, CA
United States

BOSTON RISC-V GROUP, MA
United States

NEW YORK CITY RISC-V GROUP, NY
United States

RESEARCH TRIANGLE RISC-V GROUP, NC
United States

ROCKY MOUNTAIN AREA RISC-V COMMUNITY, CO
United States

TWIN CITIES RISC-V GROUP, MN
United States

HANGZHOU RISC-V GROUP, ZHEJIANG
China

HSINCHU RISC-V GROUP
Taiwan

INDIA RISC-V GROUP, KA
India

KARACHI RISC-V GROUP, SINDH
Pakistan

MANILA-TAGUIG RISC-V GROUP, NCR
Philippines

OSAKA RISC-V GROUP
Japan

PUNE RISC-V GROUP, MH
India

RISC-V TAIPEI DAY
Taiwan

SHANGHAI RISC-V GROUP
China

TOKYO RISC-V GROUP
Japan

TOKYO-TECHNICAL STUDY RISC-V GROUP
Japan

Find more at: community.riscv.org

RISC-V Open Hours

- Bi-weekly virtual meetup for the community to interact in real-time
  - Primary focus on RISC-V support in open source software and RISC-V dev boards
  - Call for participation is open! No prepared talk or slides required!
- Schedule
  - Wednesday, June 8, 7:00 PM (US PDT) which is Thursday morning in Asia
  - Wednesday, June 29, 9:00 AM (US PDT) which is early evening in Europe

Slides: tinyurl.com/riscv-kr-22

Kernel Recipes

June 1-3, 2022

Linux on RISC-V

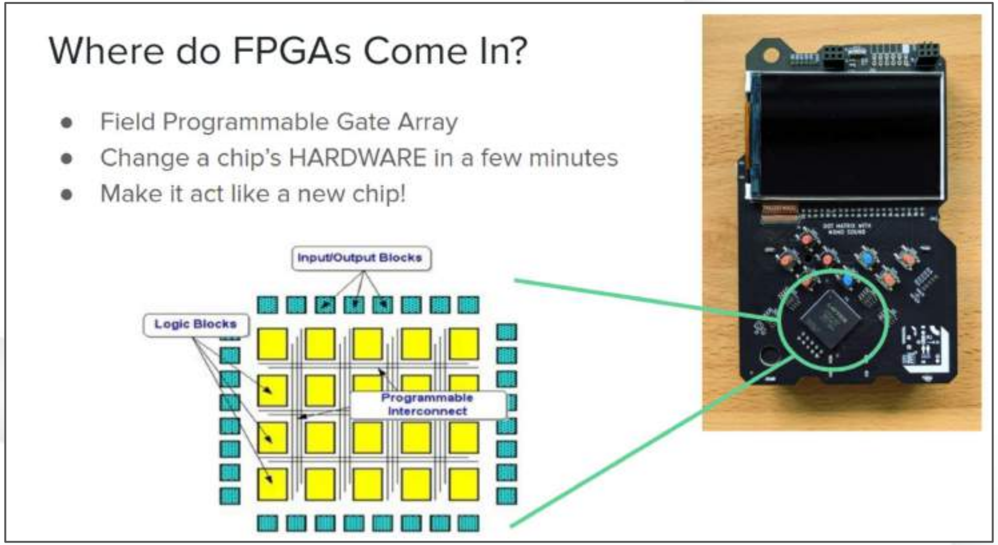
Drew Fustini <dfustini@baylibre.com>

Baylibre

<u>BONUS:</u>
What about RISC-V on FPGAs?

Introduction

- “RISC-V and FPGAs: Open Source Hardware Hacking” keynote at Hackaday Supercon 2019 by Dr. Megan Wachs



Where do FPGAs Come In?

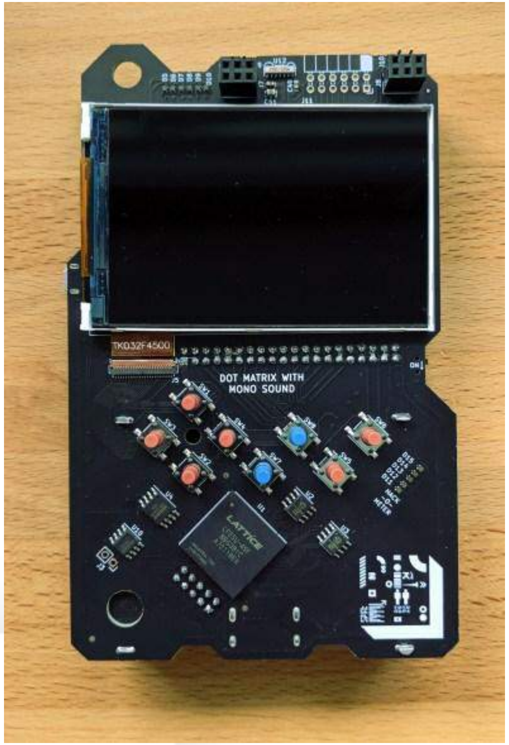
- Field Programmable Gate Array
- Change a chip’s HARDWARE in a few minutes
- Make it act like a new chip!

Open source FPGA toolchains

- Project IceStorm for Lattice iCE40 FPGA
    - "A Free and Open Source Verilog-to-Bitstream Flow for iCE40 FPGAs" by Claire Wolf
- Project Trellis for the more capable Lattice ECP5 FPGA
    - "Project Trellis and nextpnr FOSS FPGA flow for the Lattice ECP5" by Myrtle Shah
- Project X-Ray and SymbiFlow for much more capable Xilinix Series 7
    - "Xilinx Series 7 FPGAs Now Have a Fully Open Source Toolchain!" by Tim Ansell
    - "Open Source Verilog-to-Bitstream FPGA synthesis flow, currently targeting Xilinx 7-Series, Lattice iCE40 and Lattice ECP5 FPGAs. Think of it as the GCC of FPGAs"

Hackaday Supercon badge

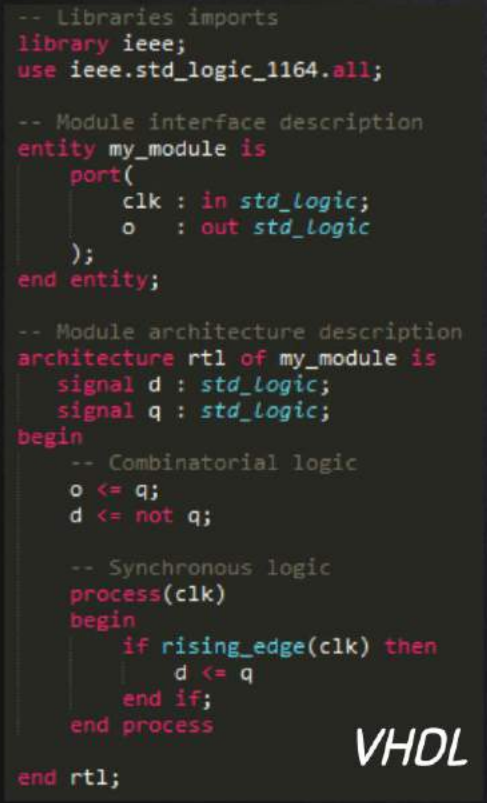
- RISC-V “soft” core on ECP5 FPGA
- Gigantic FPGA In Game Boy Form Factor



Why design an SoC in Python?

- Python has advantages over traditional HDL like VHDL and Verilog
  - Many people are familiar with Python than HDL (hardware description languages)
  - There are currently more software developers than hardware designers
- Migen is a Python framework that can automate chip design
  - Leverages the object-oriented, modular nature of Python
  - Produces Verilog code so it can be used with existing chip design workflows
- “Using Python for creating hardware to record FOSS conferences!”

What is Migen?

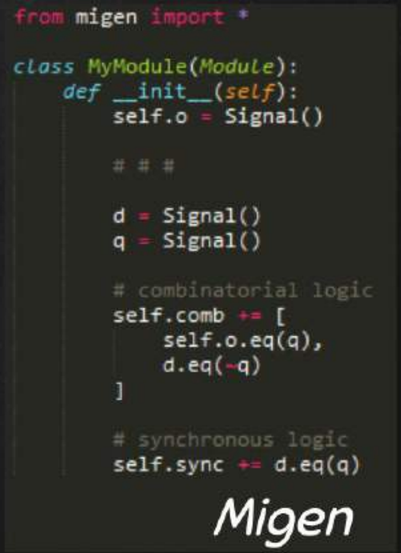


```vhdl
-- Libraries imports
library ieee;
use ieee.std_logic_1164.all;

-- Module interface description
entity my_module is
    port(
        clk : in std_logic;
        o   : out std_logic
    );
end entity;

-- Module architecture description
architecture rtl of my_module is
   signal d : std_logic;
   signal q : std_logic;
begin
    -- Combinatorial logic
    o <= q;
    d <= not q;

    -- Synchronous logic
    process(clk)
    begin
        if rising_edge(clk) then
            d <= q
        end if;
    end process

end rtl;
```

VHDL

An alternative HDL

based on Python



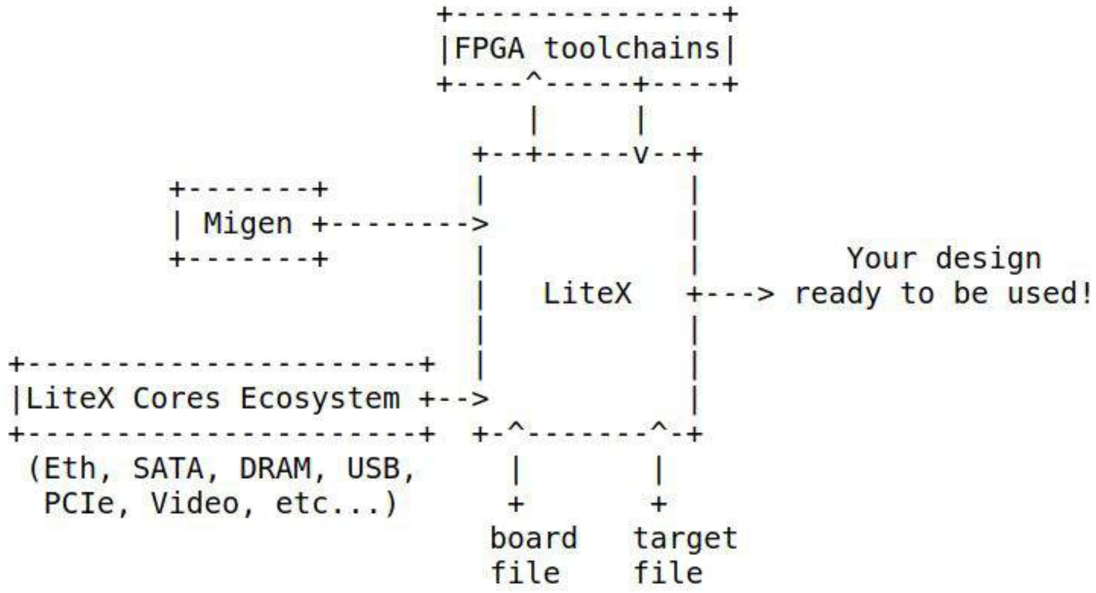
```python
from migen import *

class MyModule(Module):
    def __init__(self):
        self.o = Signal()

        # # #

        d = Signal()
        q = Signal()

        # combinatorial logic
        self.comb += [
            self.o.eq(q),
            d.eq(~q)
        ]

        # synchronous logic
        self.sync += d.eq(q)
```

Migen

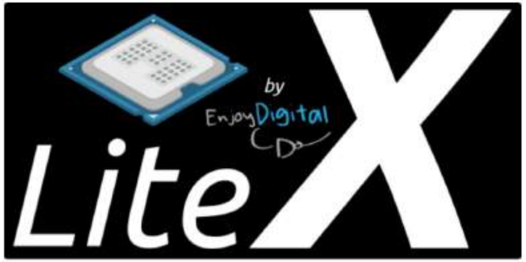
source: http://goo.gl/mZJvFQ

LiteX

- Based on Migen, builds full SoC that can be loaded into an FPGA



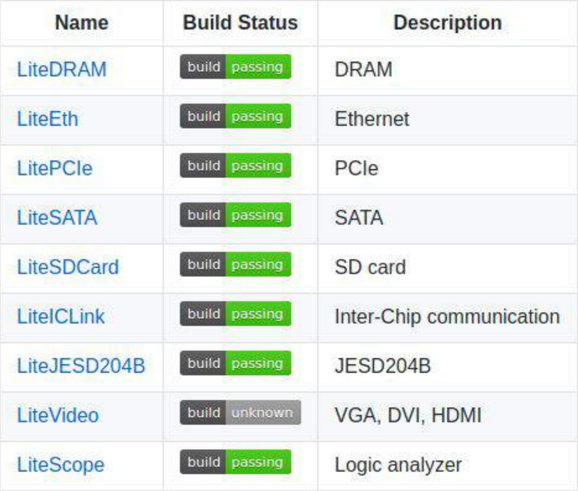
```
                        +---------------+
                        |FPGA toolchains|
                        +----^-----+----+
                             |     |
                          +--+-----v--+
         +-------+        |           |
         | Migen +-------->           |
         +-------+        |           |        Your design
                          |   LiteX   +---> ready to be used!
                          |           |
+----------------------+  |           |
|LiteX Cores Ecosystem +-->           |
+----------------------+  +-^-------^-+
 (Eth, SATA, DRAM, USB,     |       |
  PCIe, Video, etc...)      +       +
                           board   target
                           file    file
```



Build your hardware, easily!

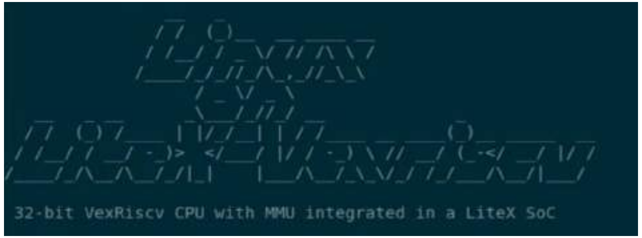
LiteX

- “LiteX vs. Vivado: First Impressions”
- Collection of open cores for DRAM, Ethernet, PCIe, SATA and more...



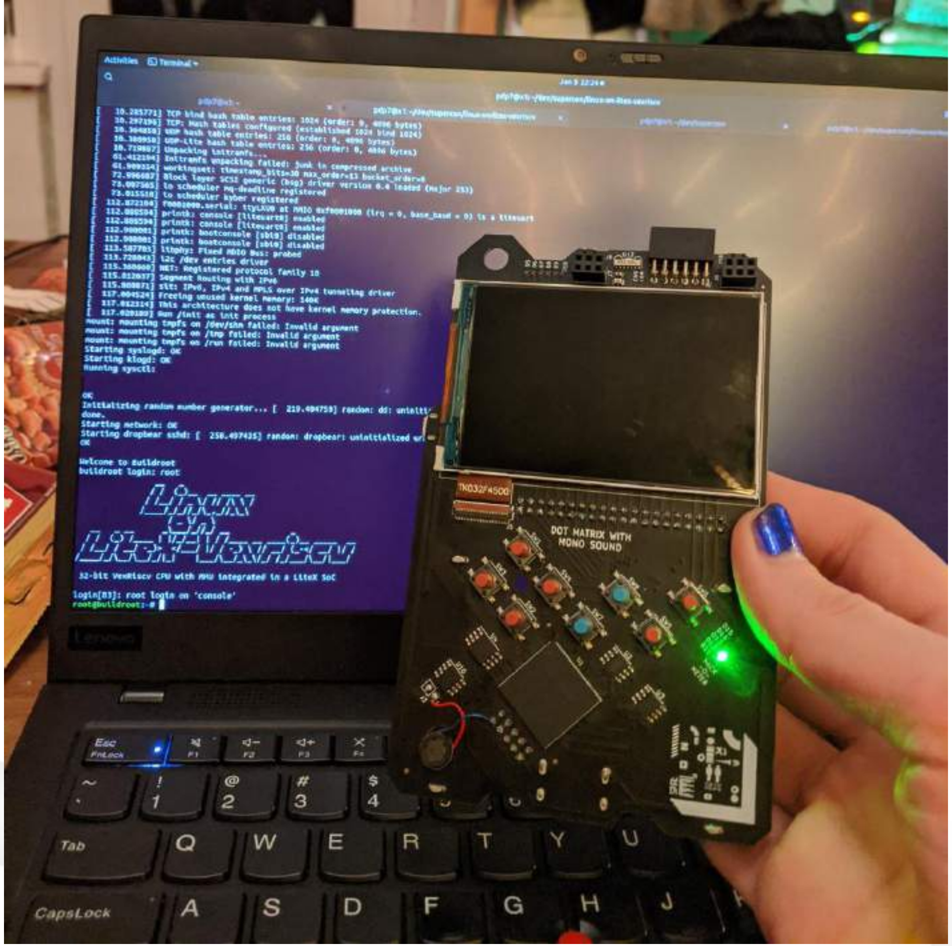
Name	Build Status	Description
LiteDRAM	build passing	DRAM
LiteEth	build passing	Ethernet
LitePCIe	build passing	PCIe
LiteSATA	build passing	SATA
LiteSDCard	build passing	SD card
LiteICLink	build passing	Inter-Chip communication
LiteJESD204B	build passing	JESD204B
LiteVideo	build unknown	VGA, DVI, HDMI
LiteScope	build passing	Logic analyzer

Linux on LiteX-VexRiscv

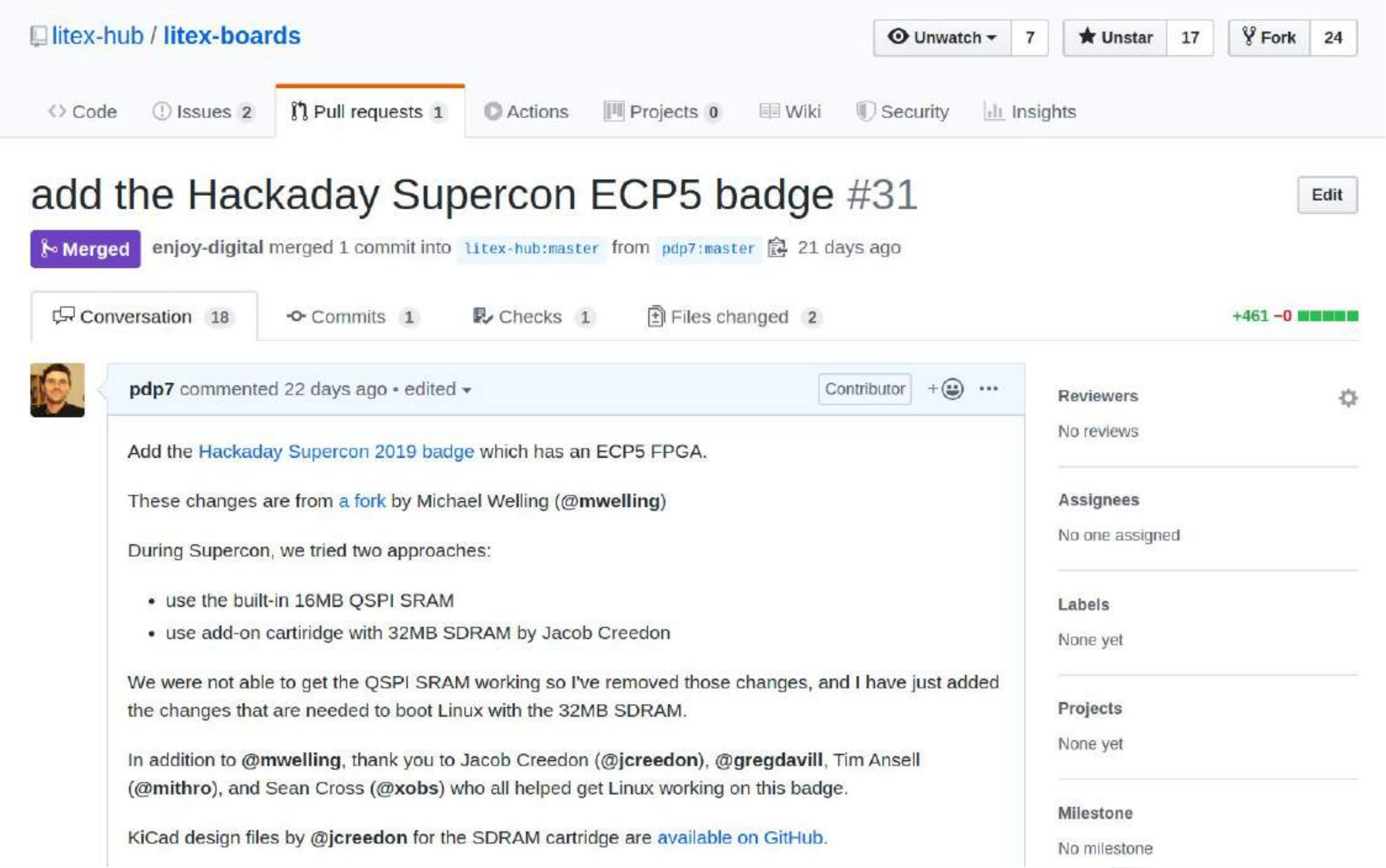


```
32-bit VexRiscv CPU with MMU integrated in a LiteX SoC
```

- VexRiscv: 32-bit Linux-capable RISC-V core
  - Designed to be FPGA friendly
  - Written in Spinal HDL (based on Scala)
- Builds an SoC using VexRiscv core and LiteX modules
  - Such as LiteDRAM, LiteEth, LiteSDCard, LitePCIe
  - “This project demonstrates how high level HDLs (Spinal HDL, Migen) enable new possibilities and complement each other. Results shown here are the results of a productive collaboration between open-source communities”
- Supports large number of FPGA dev boards including Digilent Arty A7



```
[   10.285771] TCP bind hash table entries: 1024 (order: 0, 4096 bytes)
[   10.297196] TCP: Hash tables configured (established 1024 bind 1024)
[   10.304858] UDP hash table entries: 256 (order: 0, 4096 bytes)
[   10.309958] UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
[   10.719887] Unpacking initramfs...
[   61.412194] Initramfs unpacking failed: junk in compressed archive
[   61.909314] workingset: timestamp_bits=30 max_order=13 bucket_order=0
[   72.996687] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 253)
[   73.007565] io scheduler mq-deadline registered
[   73.015518] io scheduler kyber registered
[  112.872104] f0002000.serial: ttyLXU0 at MMIO 0xf0001000 (irq = 0, base_baud = 0) is a liteuart
[  112.888504] printk: console [liteuart0] enabled
[  112.888504] printk: console [liteuart0] enabled
[  112.900081] printk: bootconsole [sbi0] disabled
[  112.900081] printk: bootconsole [sbi0] disabled
[  113.587703] litephy: Fixed MDIO Bus: probed
[  113.720043] i2c /dev entries driver
[  115.380000] NET: Registered protocol family 10
[  115.812037] Segment Routing with IPv6
[  115.800871] sit: IPv6, IPv4 and MPLS over IPv4 tunneling driver
[  117.004524] Freeing unused kernel memory: 140K
[  117.012314] This architecture does not have kernel memory protection.
[  117.020189] Run /init as init process
mount: mounting tmpfs on /dev/shm failed: Invalid argument
mount: mounting tmpfs on /tmp failed: Invalid argument
mount: mounting tmpfs on /run failed: Invalid argument
Starting syslogd: OK
Starting klogd: OK
Running sysctl:


OK
Initializing random number generator... [  219.404759] random: dd: uninit
done.
Starting network: OK
Starting dropbear sshd: [  250.497435] random: dropbear: uninitialized ur
OK

Welcome to Buildroot
buildroot login: root

           Linux
             on
   LiteX-VexRiscv

 32-bit VexRiscv CPU with MMU integrated in a LiteX SoC

login[83]: root login on 'console'
root@buildroot:~#
```



add the Hackaday Supercon ECP5 badge #31

enjoy-digital merged 1 commit into `litex-hub:master` from `pdp7:master` 21 days ago

pdp7 commented 22 days ago • edited

Add the Hackaday Supercon 2019 badge which has an ECP5 FPGA.

These changes are from a fork by Michael Welling (**@mwelling**)

During Supercon, we tried two approaches:

- use the built-in 16MB QSPI SRAM
- use add-on cartiridge with 32MB SDRAM by Jacob Creedon

We were not able to get the QSPI SRAM working so I've removed those changes, and I have just added the changes that are needed to boot Linux with the 32MB SDRAM.

In addition to **@mwelling**, thank you to Jacob Creedon (**@jcreedon**), **@gregdavill**, Tim Ansell (**@mithro**), and Sean Cross (**@xobs**) who all helped get Linux working on this badge.

KiCad design files by **@jcreedon** for the SDRAM cartridge are available on GitHub.

Reviewers

No reviews

Assignees

No one assigned

Labels

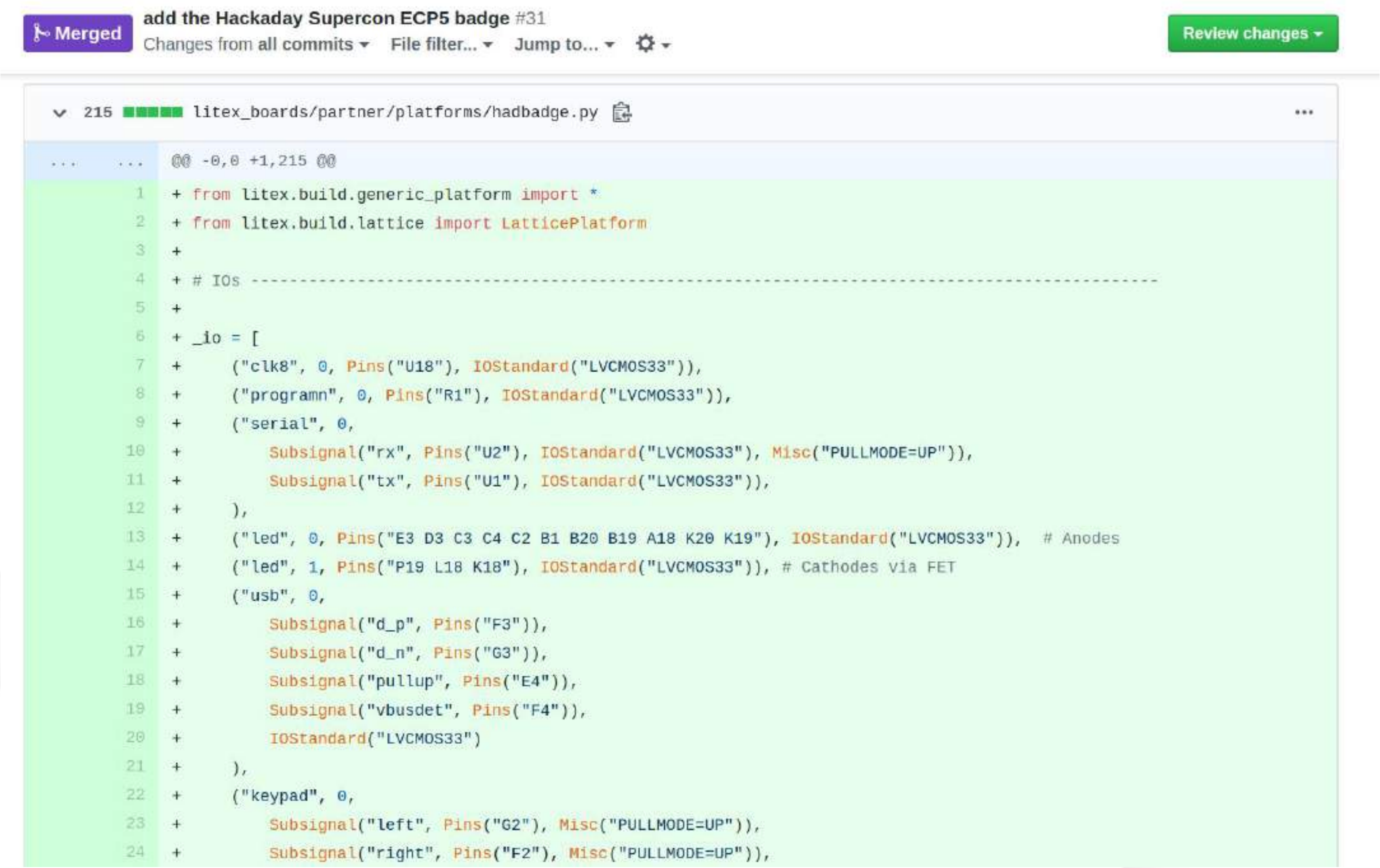
None yet

Projects

None yet

Milestone

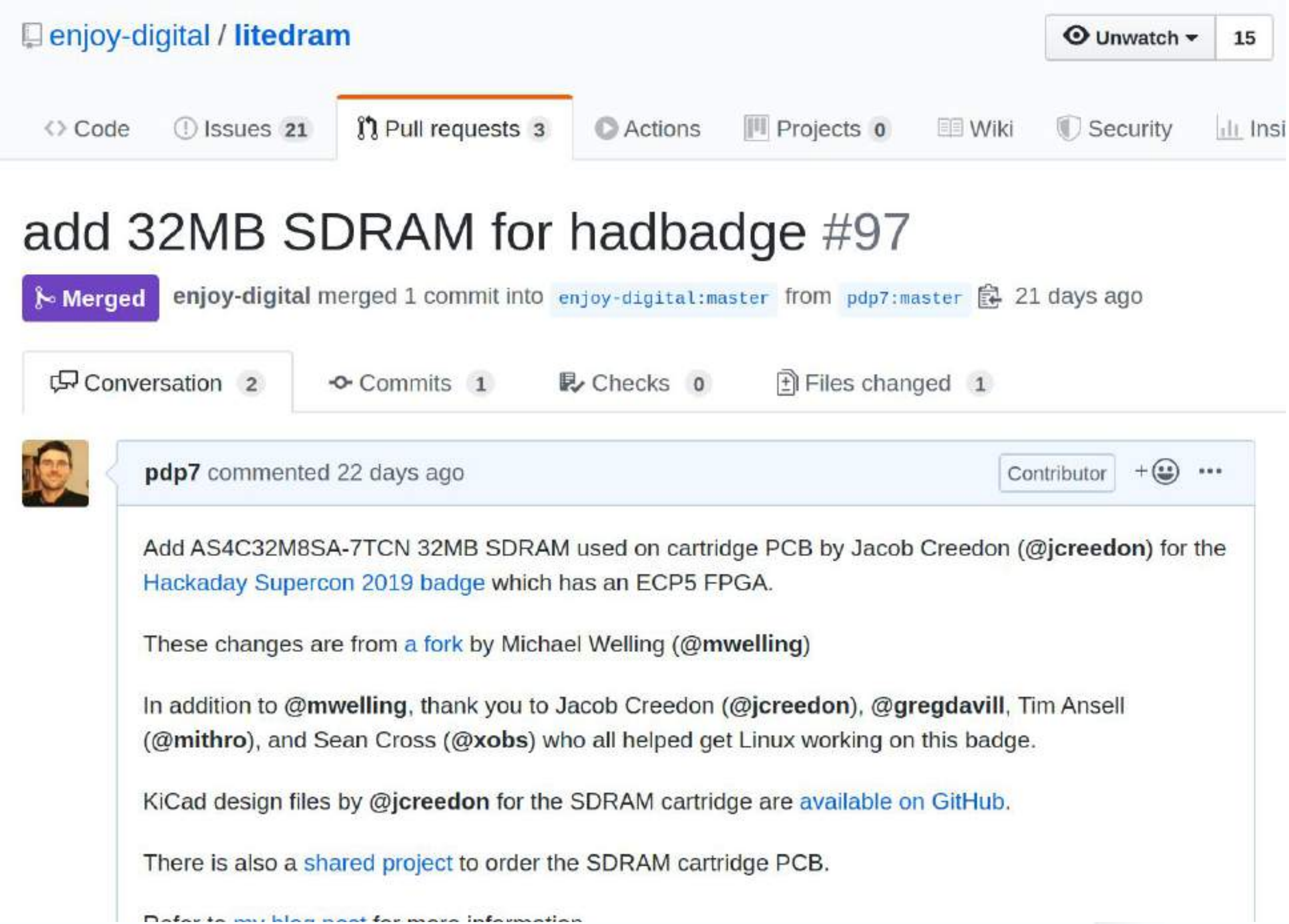
No milestone



add the Hackaday Supercon ECP5 badge #31

215 🟩🟩🟩🟩🟩 litex_boards/partner/platforms/hadbadge.py

```
@@ -0,0 +1,215 @@
+ from litex.build.generic_platform import *
+ from litex.build.lattice import LatticePlatform
+
+ # IOs ----------------------------------------------------------------------------------------------
+
+ _io = [
+     ("clk8", 0, Pins("U18"), IOStandard("LVCMOS33")),
+     ("programn", 0, Pins("R1"), IOStandard("LVCMOS33")),
+     ("serial", 0,
+         Subsignal("rx", Pins("U2"), IOStandard("LVCMOS33"), Misc("PULLMODE=UP")),
+         Subsignal("tx", Pins("U1"), IOStandard("LVCMOS33")),
+     ),
+     ("led", 0, Pins("E3 D3 C3 C4 C2 B1 B20 B19 A18 K20 K19"), IOStandard("LVCMOS33")),  # Anodes
+     ("led", 1, Pins("P19 L18 K18"), IOStandard("LVCMOS33")), # Cathodes via FET
+     ("usb", 0,
+         Subsignal("d_p", Pins("F3")),
+         Subsignal("d_n", Pins("G3")),
+         Subsignal("pullup", Pins("E4")),
+         Subsignal("vbusdet", Pins("F4")),
+         IOStandard("LVCMOS33")
+     ),
+     ("keypad", 0,
+         Subsignal("left", Pins("G2"), Misc("PULLMODE=UP")),
+         Subsignal("right", Pins("F2"), Misc("PULLMODE=UP")),
```



add 32MB SDRAM for hadbadge #97

Merged **enjoy-digital** merged 1 commit into `enjoy-digital:master` from `pdp7:master` 21 days ago

pdp7 commented 22 days ago

Add AS4C32M8SA-7TCN 32MB SDRAM used on cartridge PCB by Jacob Creedon (**@jcreedon**) for the Hackaday Supercon 2019 badge which has an ECP5 FPGA.

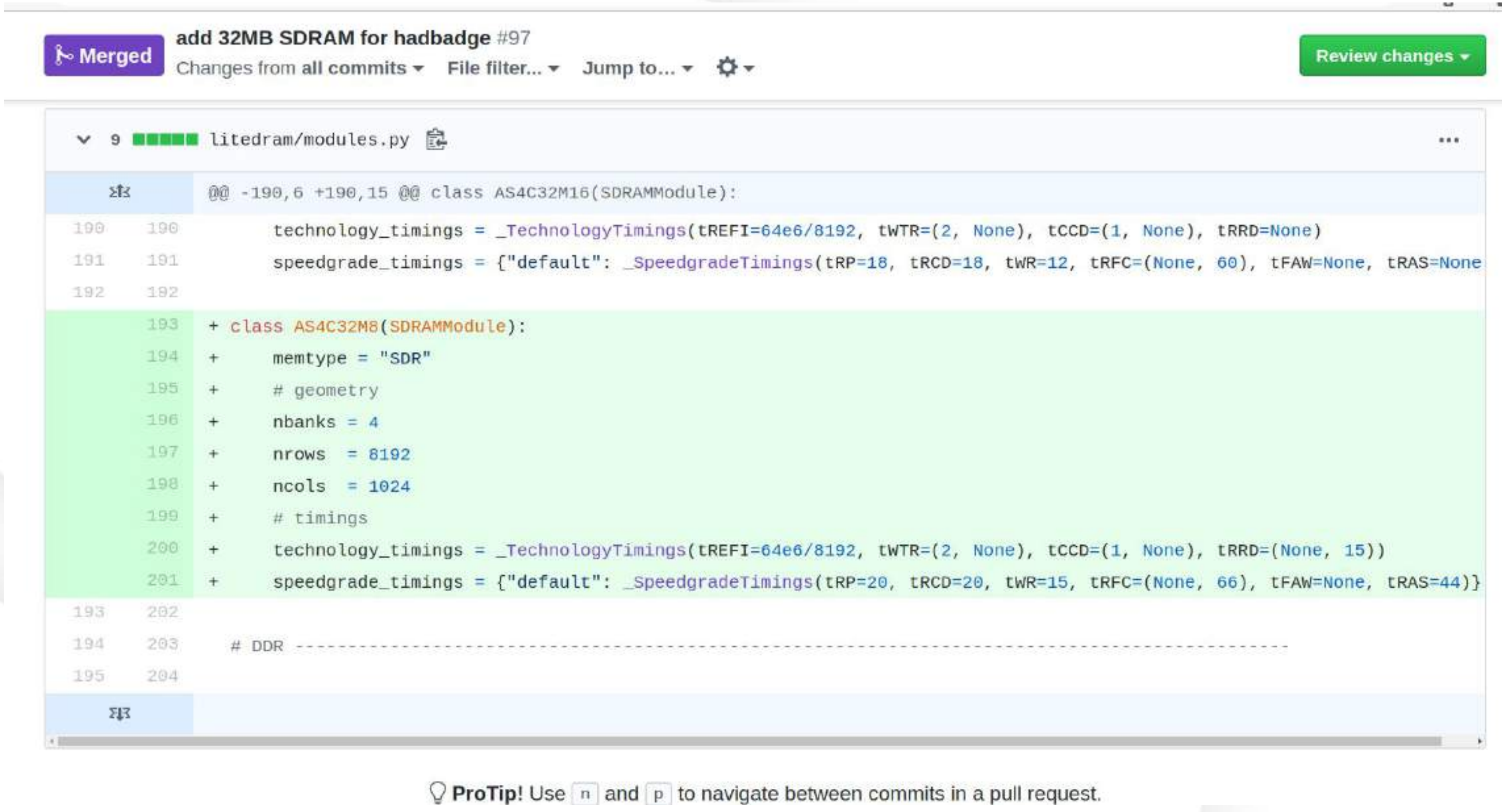
These changes are from a fork by Michael Welling (**@mwelling**)

In addition to **@mwelling**, thank you to Jacob Creedon (**@jcreedon**), **@gregdavill**, Tim Ansell (**@mithro**), and Sean Cross (**@xobs**) who all helped get Linux working on this badge.

KiCad design files by **@jcreedon** for the SDRAM cartridge are available on GitHub.

There is also a shared project to order the SDRAM cartridge PCB.

Refer to my blog post for more information.

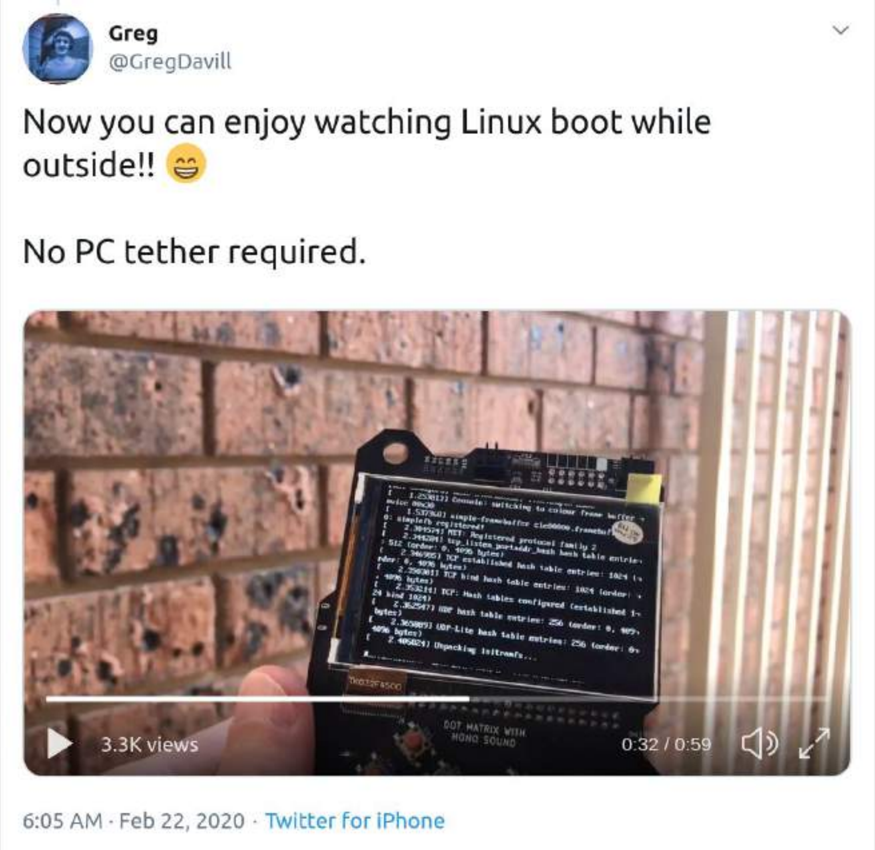


add 32MB SDRAM for hadbadge #97

9 🟩🟩🟩🟩🟩 litedram/modules.py

```
@@ -190,6 +190,15 @@ class AS4C32M16(SDRAMModule):
      technology_timings = _TechnologyTimings(tREFI=64e6/8192, tWTR=(2, None), tCCD=(1, None), tRRD=None)
      speedgrade_timings = {"default": _SpeedgradeTimings(tRP=18, tRCD=18, tWR=12, tRFC=(None, 60), tFAW=None, tRAS=None

+ class AS4C32M8(SDRAMModule):
+     memtype = "SDR"
+     # geometry
+     nbanks = 4
+     nrows  = 8192
+     ncols  = 1024
+     # timings
+     technology_timings = _TechnologyTimings(tREFI=64e6/8192, tWTR=(2, None), tCCD=(1, None), tRRD=(None, 15))
+     speedgrade_timings = {"default": _SpeedgradeTimings(tRP=20, tRCD=20, tWR=15, tRFC=(None, 66), tFAW=None, tRAS=44)}

  # DDR -----------------------------------------------------------------------------------------------
```

ProTip! Use n and p to navigate between commits in a pull request.



Greg
@GregDavill

Now you can enjoy watching Linux boot while outside!! 😁

No PC tether required.

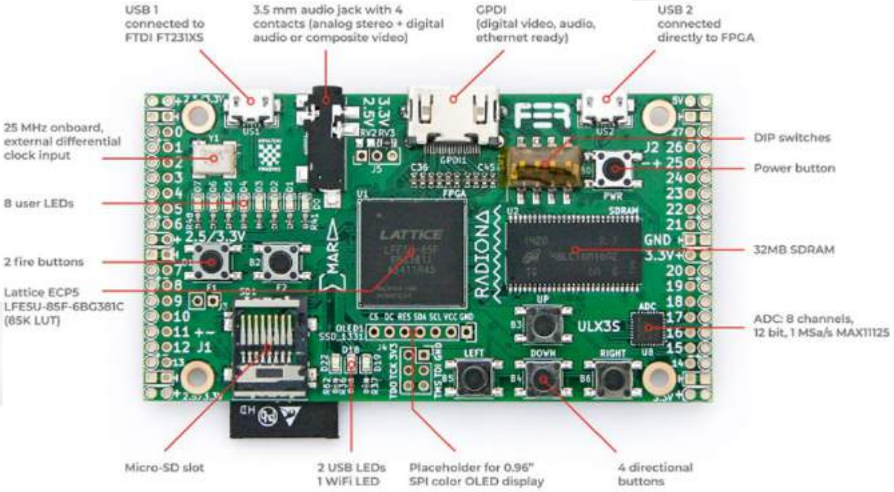
```
[    1.253012] Console: switching to colour frame buffer -
evice 80x30
[    1.537960] simple-framebuffer c1e00000.framebuf-
0: simplefb registered!
[    2.304574] NET: Registered protocol family 2
[    2.344280] tcp_listen_portaddr_hash hash table entrie-
s: 512 (order: 0, 4096 bytes)
[    2.346985] TCP established hash table entries: 1024 (o-
rder: 0, 4096 bytes)
[    2.350381] TCP bind hash table entries: 1024 (order: -
, 4096 bytes)
[    2.353214] TCP: Hash tables configured (established 1-
24 bind 1024)
[    2.362547] UDP hash table entries: 256 (order: 0, 409-
bytes)
[    2.365089] UDP-Lite hash table entries: 256 (order: 0,
4096 bytes)
[    2.405024] Unpacking initramfs...
```

6:05 AM · Feb 22, 2020 · Twitter for iPhone

https://twitter.com/GregDavill/status/1231082623633543168

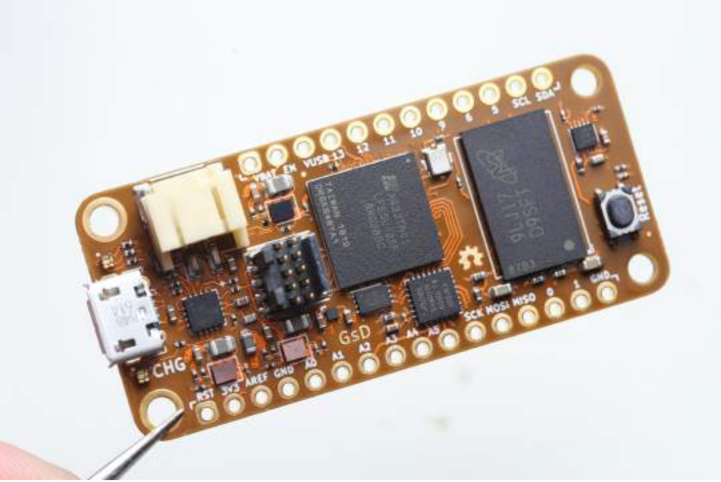
Open Source ECP5 FPGA boards

- Radiona.org ULX3S
  - 32MB SDRAM; ESP32 on-board for WiFi and Bluetooth; $115 on CrowdSupply or Mouser


USB 1 connected to FTDI FT231XS
3.5 mm audio jack with 4 contacts (analog stereo + digital audio or composite video)
GPDI (digital video, audio, ethernet ready)
USB 2 connected directly to FPGA
25 MHz onboard, external differential clock input
8 user LEDs
2 fire buttons
Lattice ECP5 LFE5U-85F-6BG381C (85K LUT)
DIP switches
Power button
32MB SDRAM
ADC: 8 channels, 12 bit, 1 MSa/s MAX11125
Micro-SD slot
2 USB LEDs 1 WiFi LED
Placeholder for 0.96" SPI color OLED display
4 directional buttons

Open Source ECP5 FPGA boards

- OrangeCrab by Greg Davill
    - 128MB DDR RAM; Adafruit Feather form factor; available for $129



Want to learn FPGAs? Try Fomu!

- Online workshop from Tim Ansell and Sean Cross
- $50 on CrowdSupply
- Fits inside USB port!
- Learn how to use:
  - MicroPython
  - Verilog
  - LiteX