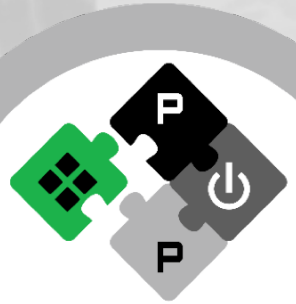


PULP RISC-V Cores: Ariane, RI5CY and friends

HPCA 2018 - Vienna

25.02.2018

Florian Zaruba

Frank K. Gürkaynak

Andreas Kurth

Francesco Conti

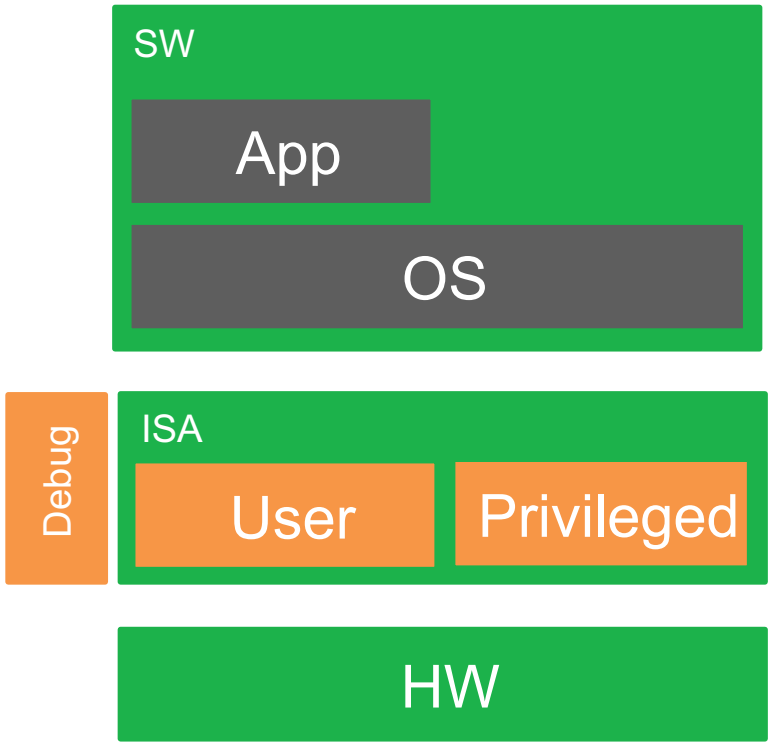
http://pulp-platform.org

[1]Department of Electrical, Electronic and Information Engineering

[2]Integrated Systems Laboratory

RISC-V Instruction Set Architecture

- Started by UC-Berkeley in 2010
- Contract between SW and HW
  - Partitioned into user and privileged spec
  - External Debug
- Open Standard governed by RISC-V foundation
  - **ETHZ is a founding member** of the foundation
  - Necessary for the continuity
- Defines 32, 64 and 128 bit ISA
  - No implementation, just the ISA
  - Different RISC-V implementations (both open and close source) are available
- At IIS we specialize in **efficient implementations of RISC-V cores**



RISC-V ISA

- Generally kept very simple and extendable
- From deeply-embedded devices to HPC
- ISA support is given by RV + word-width + extensions supported (RV32I)
- User specification:
  - Separated into extensions, only I is mandatory
- Privileged Specification (WIP):
  - Governs OS functionality: Exceptions, Interrupts
  - Virtual Addressing
  - Privilege Levels

Spec separated into "extensions"

I	Integer instructions (frozen)
E	Reduced number of registers
M	Multiplication and Division (frozen)
A	Atomic instructions (frozen)
F	Single-Precision Floating-Point (frozen)
D	Double-Precision Floating-Point (frozen)
C	Compressed Instructions (frozen)
X	Non Standard Extensions

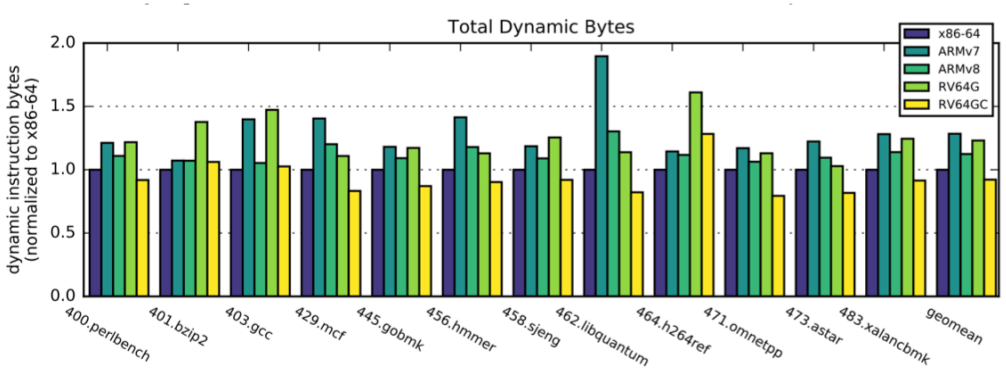
RISC-V Planned Extensions

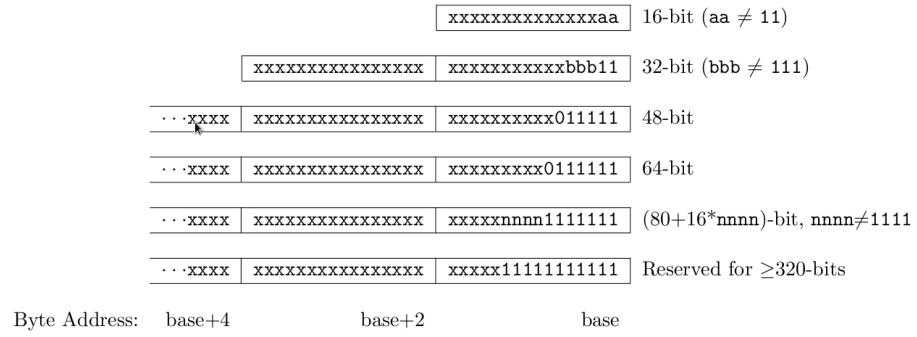
- No or partial work done yet on those extensions
- Possible to contribute as a foundation member in task-groups
- Dedicated task-groups
  - Formal specification
  - Memory Model
  - Marketing
  - External Debug Specification
- For Bit-manipulation we provide our own solution → part of the task group

Q	Quad-Precision Floating-Point
L	Decimal Floating-Point (IEEE 754-2008)
B	Bit-Manipulation
T	Transactional Memory
P	Packed-SIMD
J	Dynamically Translated Languages
V	Vector Operations
N	User-Level Interrupts

Compressed Instructions “C”

- 16-bit instructions covering most used ops, code size reduction by 34 %
- Length-encoded in LSBs, variable length instructions
- Compressed instructions increase fetch-bandwidth
- Allow for macro-op fusion of common patterns

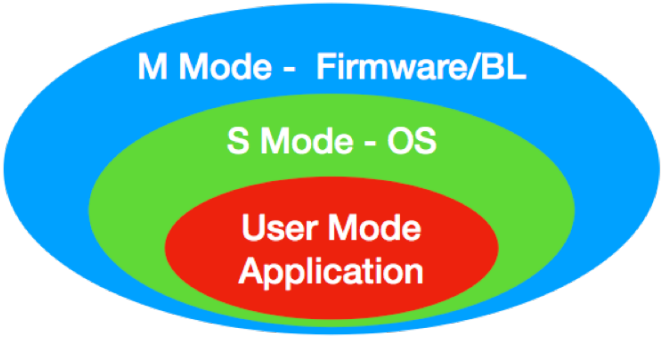




x86-64: 3.71 bytes / instruction RV64IC: 3.00 bytes / instruction

Privileged Specification 1.11 (WIP)

- Defines Control and Status Registers (CSR)
- Defines instructions to RMW CSR
- 3 Rings of operation
  - Machine-, Supervisor- and User-Mode
  - Hypervisor WIP
- Exceptions + IRQ support
  - IRQ/Exception stack
  - Wait for Interrupt (WFI) instruction
  - Specification for platform level interrupt controller
  - Instructions to enter/return from exceptions
- Virtual Memory
  - Page-based, 32-bit PA, 39-bit PA and/or 48-bit PA
- Platform Configuration String (DTS)



RISC-V Ecosystem[1]

- Binutils – upstream
- GCC – upstream
- LLVM – in development
- Simulator:
  - ”Spike” - reference
  - QEMU, Gem5
- OpenOCD
- OS: Linux, sel4, freeRTOS, zephyr
- Runtimes: Jikes, Ocaml, Go

- SW maintained by different parties
  - Binutils and GCC by Sifive a Berkeley start-up

[1] https://github.com/riscv/riscv-wiki/wiki/RISC-V-Software-Status

RISC-V cores under development at IIS

32 bit

Low Cost Core

- **Zero-riscy**
  - RV32-ICM
- **Micro-riscy**
  - RV32-CE

ARM Cortex-M0+

Core with DSP enhancements

- **RI5CY**
  - RV32-ICMX
  - SIMD
  - HW loops
  - Bit Man
  - Fixed point

ARM Cortex-M4

Floating-point capable Core

- **RI5CY + FPU**
  - RV32-ICMFX

ARM Cortex-M4F

64 bit

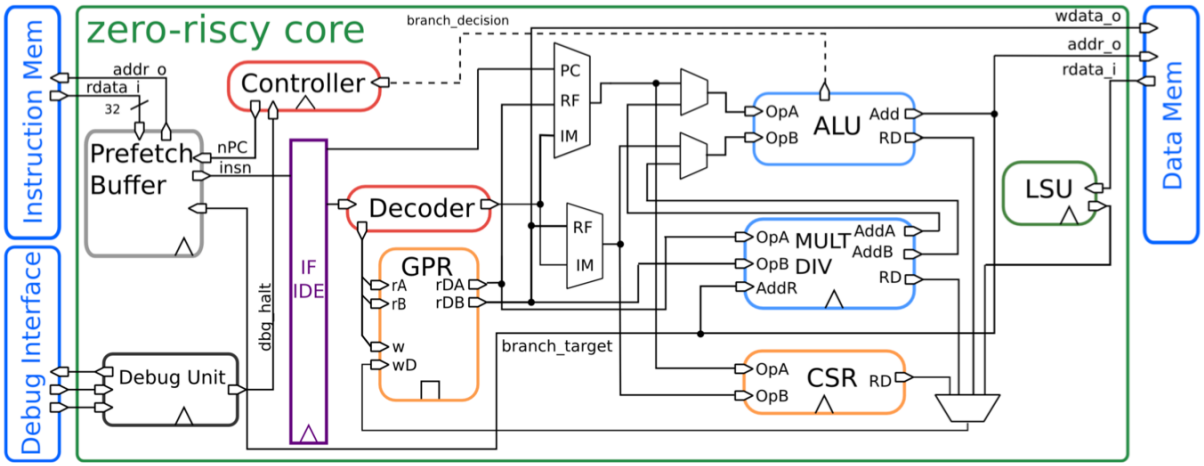
Linux capable Core

- **Ariane**
  - RV64-IC(MA)
  - Full privileged specification

ARM Cortex-A5

Zero/Micro-riscy, small area core for control applications

- **2-stage pipeline**
- **Optimized for area**
  - Area:
    19 kGE (Zero-riscy)
    12 kGE (Micro-riscy)
  - Critical path:
    ~ 30 logic levels
- Plans for tape-out
  - Early version taped out in UMC65
  - Controller core of Mr. Wolf (PULP systems in TSMC40 LP)

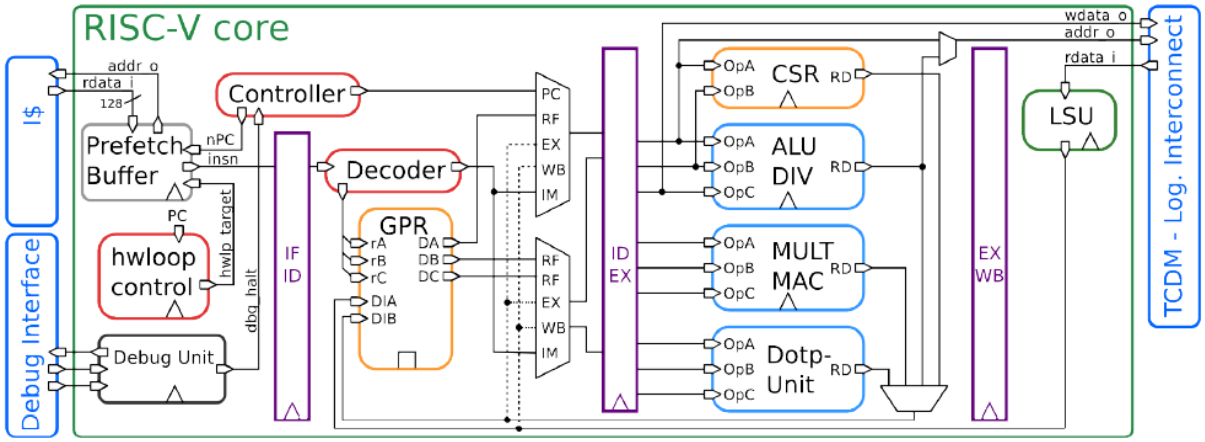


- **Two Configurations:**
  - **Zero-riscy**: RV32-ICM (2,44 Coremark/MHz)
    - 32 registers, hardware multiplier
  - **Micro-riscy** : RV32-CE (0,91 Coremark/MHz)
    - 16 registers (E), software emulated multiplier

PULP Multitherman ExaNoDe PRECOMP Open Transprecision Computing ETH

RI5CY: Our 32-bit workhorse

- **4-stage pipeline**
  - 41 kGE
  - 30 logic levels of critical path
  - Coremark/MHz 3.19
- **Includes various extensions**
  - SIMD
  - Fixed point
  - Bit manipulations
  - HW loops
- **Versions taped out in:**
  - GF28
  - UMC65
  - TSMC40LP (with FPU, in preparation)



- **Different Options:**
  - **FPU**: IEEE 754 single precision
    - Including hardware support for FDIV, FSQRT, FMAC, FMUL
  - **Privilege support**:
    - Supports privilege mode **M** and **U**

PULP Multitherman ExaNoDe PRECOMP Open Transprecision Computing ETH

A new perspective: Application class processor

- Currently: Bare metal/simple run-time
- Virtual Memory
  - Multi-program environment
  - Efficient sharing and protection
- Operating System
  - Highly sequential code
  - Increase frequency to gain performance
- Large software infrastructure
  - Drivers for hardware (PCIe, ethernet)
  - Application SW (e.g.: Tensorflow, …)

- **Larger address space** (64-bit)
- **Requires more hardware support**
  - MMU (TLBs, PTW)
  - Privilege Levels
  - More Exceptions (page fault, illegal access)

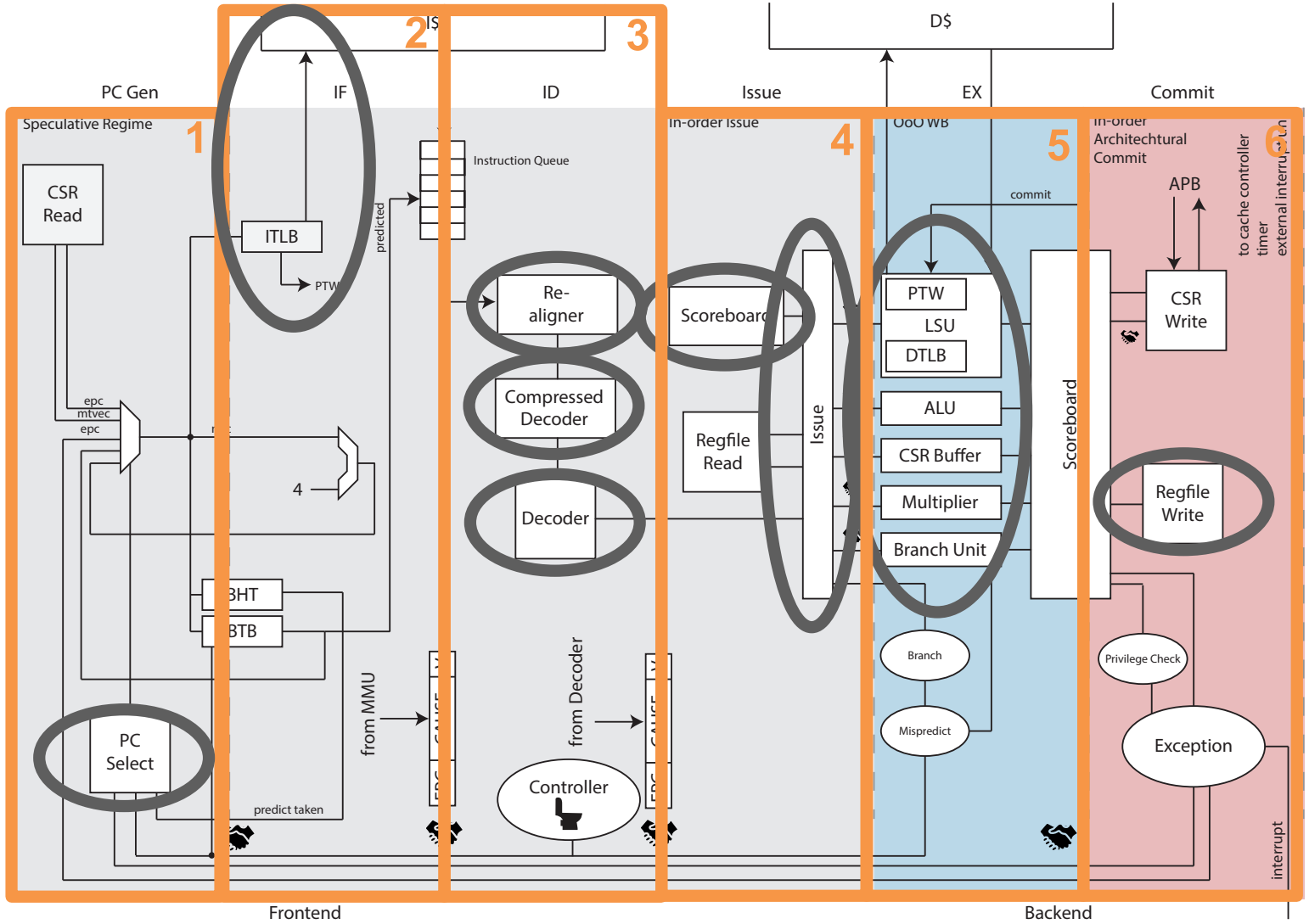
→ **Ariane** an application class processor

PULP Multitherman ExaNoDe PRECOMP Open Transprecision Computing ETH

ARIANE: Linux Capable 64-bit core

- Application class processor
- Linux Capable
  - M, S and U privilege modes
  - TLB
  - Tightly integrated D$ and I$
  - Hardware PTW
- Optimized for performance
  - Frequency: > 1.5 GHz (22 FDX)
  - Area: 185 kGE
  - Critical path: ~ 25 logic levels
- 6-stage pipeline
  - In-order issue
  - Out-of-order write-back
  - In-order commit
- Branch-prediction
- Scoreboarding
- Taped-out in GF22FDX
- Designed for extendability

1. PC Gen
   - Select PC
2. Instr. Fetch
   - TLB
   - Query I$
3. Instr. Decode
   - Re-align
   - De-compress
   - Decode
4. Issue
   - Select FU
   - Issue
5. Execute
6. Commit
   1. Write state



PULP Multitherman ExaNoDe PRECOMP Open Transprecision Computing ETH

Frequency-IPC trade-off

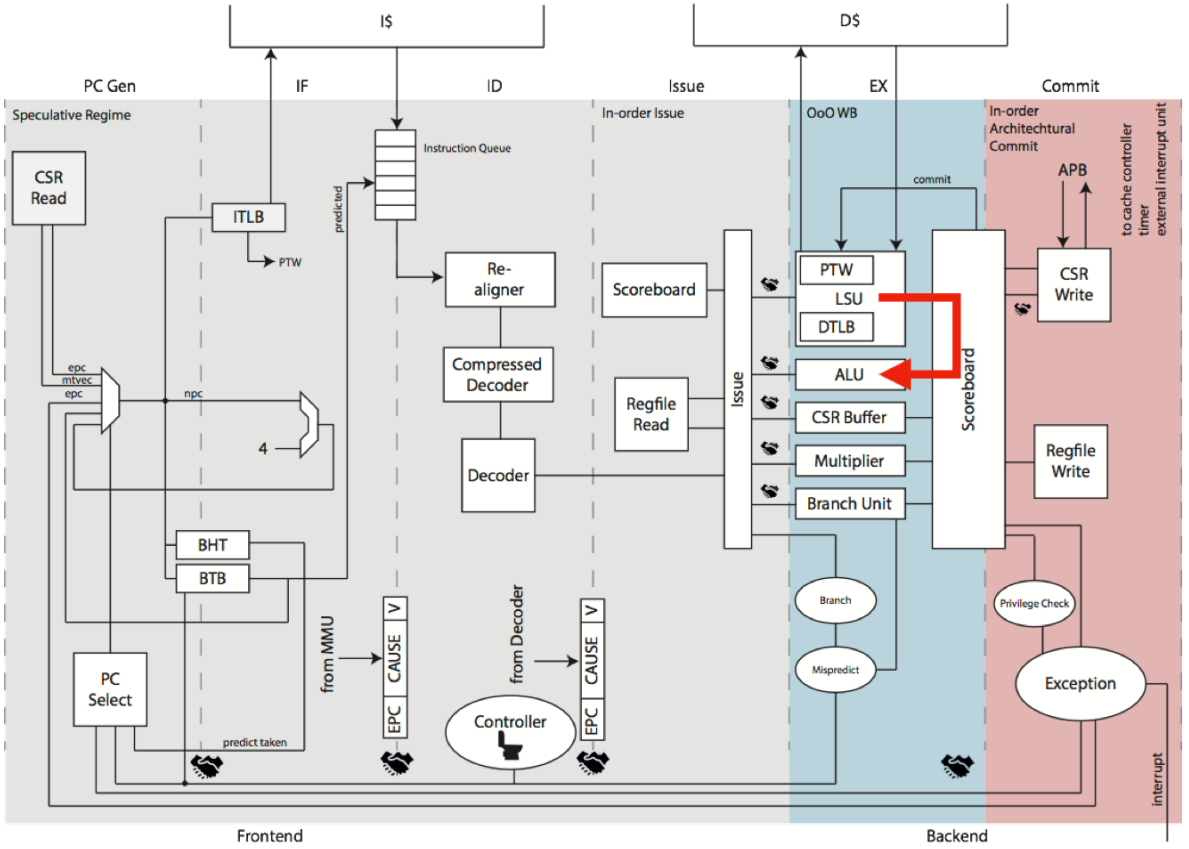
- Frequency:
  - Increase frequency through pipelining
  - Modern Intel CPUs have around 10 - 20 pipeline stages
- Adds significant complexity on the cache interfaces

- Increased bubbles due to:
  - Data Hazards ➔ **Forwarding**
  - Structural Hazards ➔ **Scoreboard**
  - Control Hazards ➔ **Branch Prediction**

Data Hazards - Forwarding

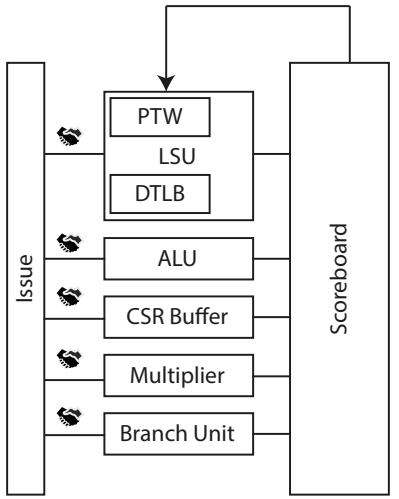
Data Hazards ➔ **Forwarding**

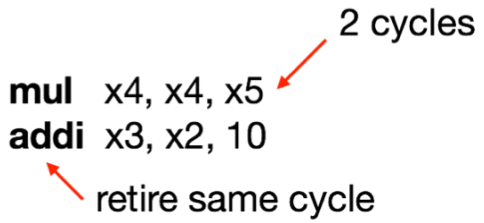
```
ld    x2, 0(x7)
addi  x3, x2, 10
```



Scoreboarding

- Hide latency of multi-cycle instructions
- Clean and modular interface to functional units ➔ scalability (FPU)
- Add issue port: Dual-Issue implementation
- Split execution into four steps:
  - **Issue**: Relatively complex issue logic (extra pipeline-stage)
  - **Read Operands**: From register file or forwarded
  - **Execute**
  - **Write Back**: Mitigate structural hazards on write-back path
- Implemented as a circular buffer





Scoreboard

Decode:

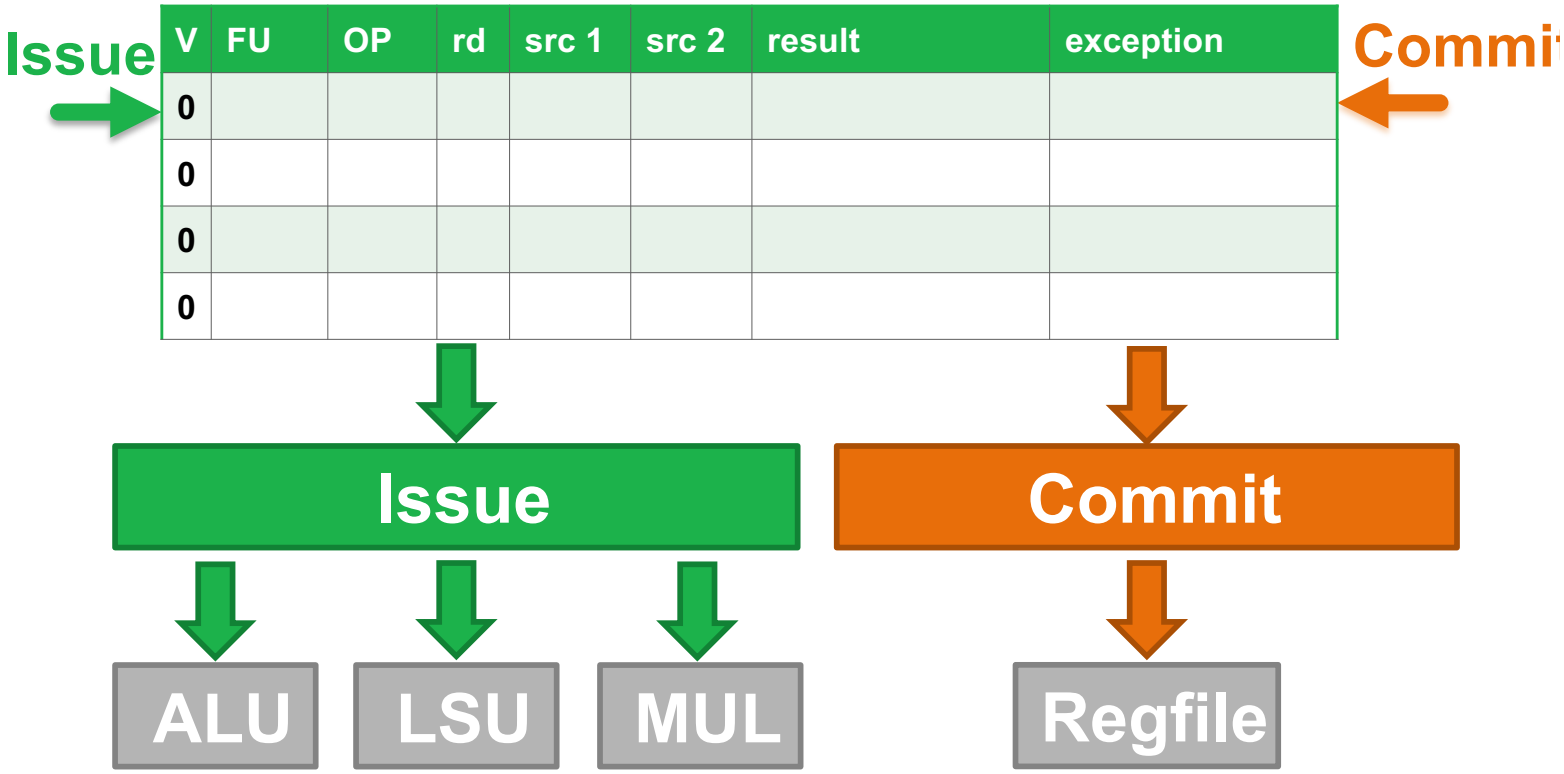
ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

addi x2 ← x3, x1

ld x1 ← x2(128)



Scoreboard – 3 cycle instruction (LD)

Decode:

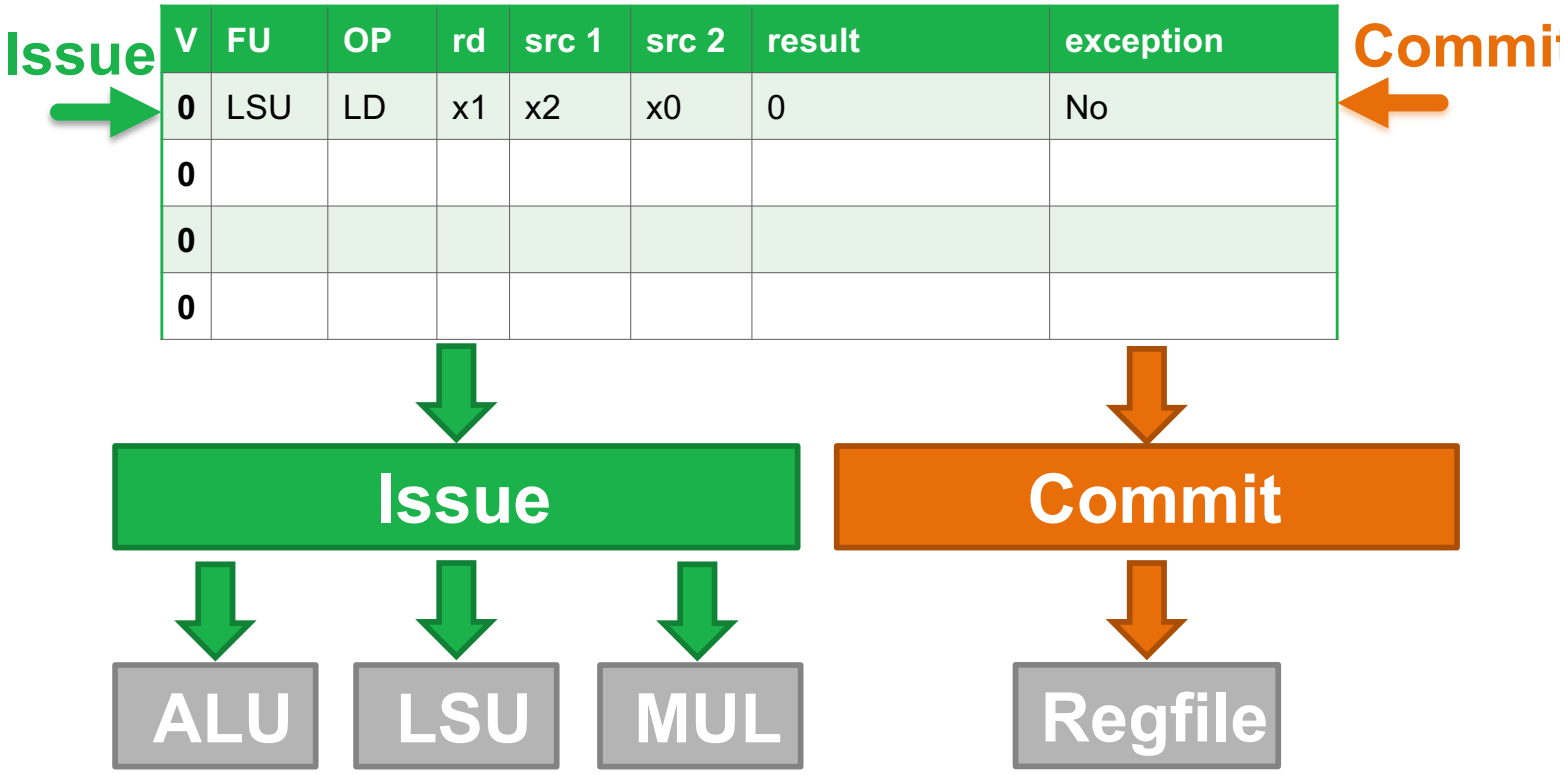
ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

addi x2 ← x3, x1

ld x1 ← x2(128)



Scoreboard – 2 cycle instruction (MUL)

Decode:

ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

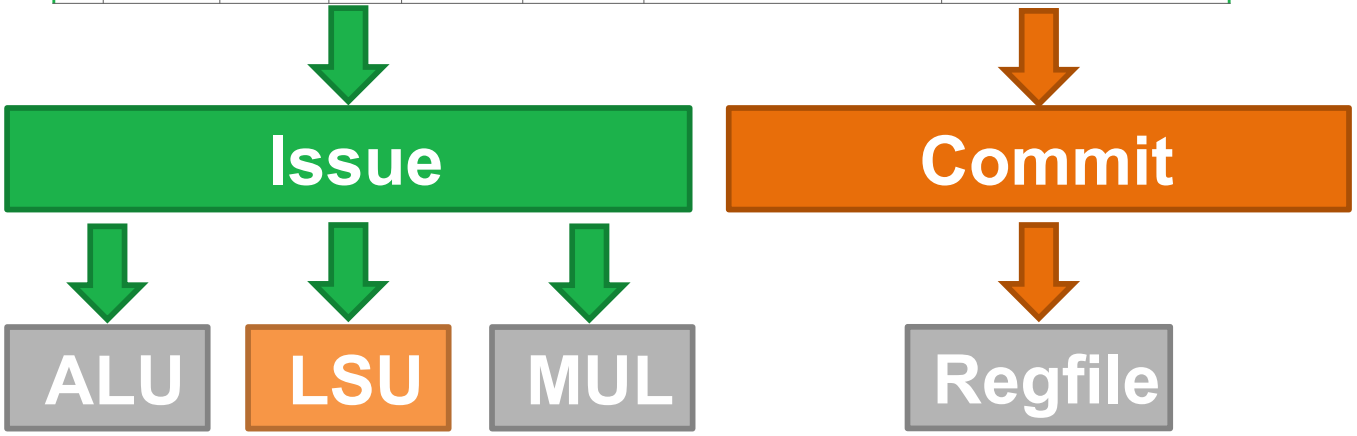
addi x2 ← x3, x1

ld x1 ← x2(128)

Issue

V	FU	OP	rd	src 1	src 2	result	exception
0	LSU	LD	x1	x2	x0	0	No
0	MUL	MUL	x2	x2	x2	0	No
0							
0							

Commit



Scoreboard - single cycle instruction (ALU)

Decode:

ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

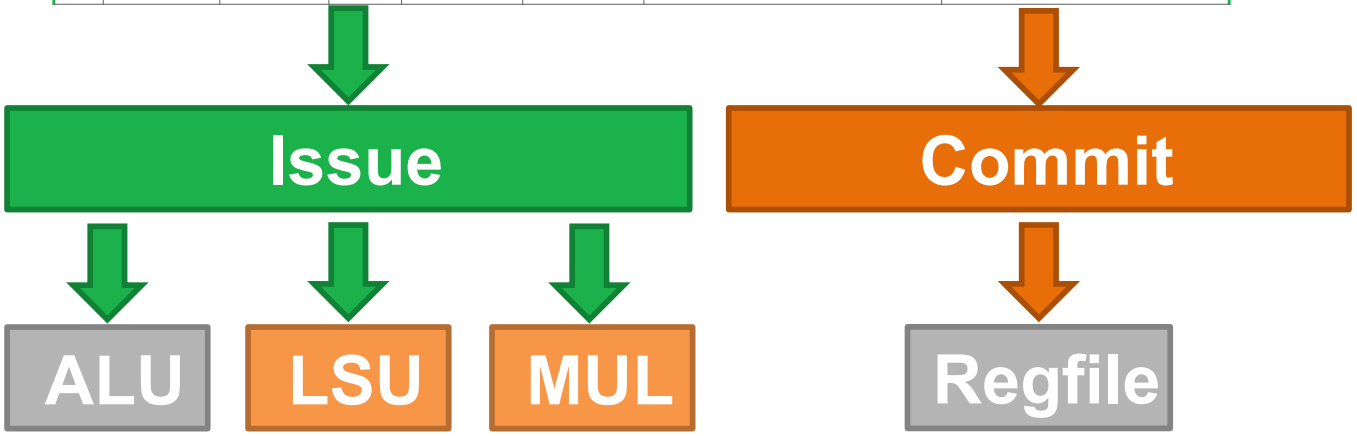
addi x2 ← x3, x1

ld x1 ← x2(128)

Issue

V	FU	OP	rd	src 1	src 2	result	exception
0	LSU	LD	x1	x2	x0	0	No
0	MUL	MUL	x2	x2	x2	0	No
0	ALU	ADD	x3	x0	x0	16	No
0							

Commit



Scoreboard – Multiple Write Back

Decode:

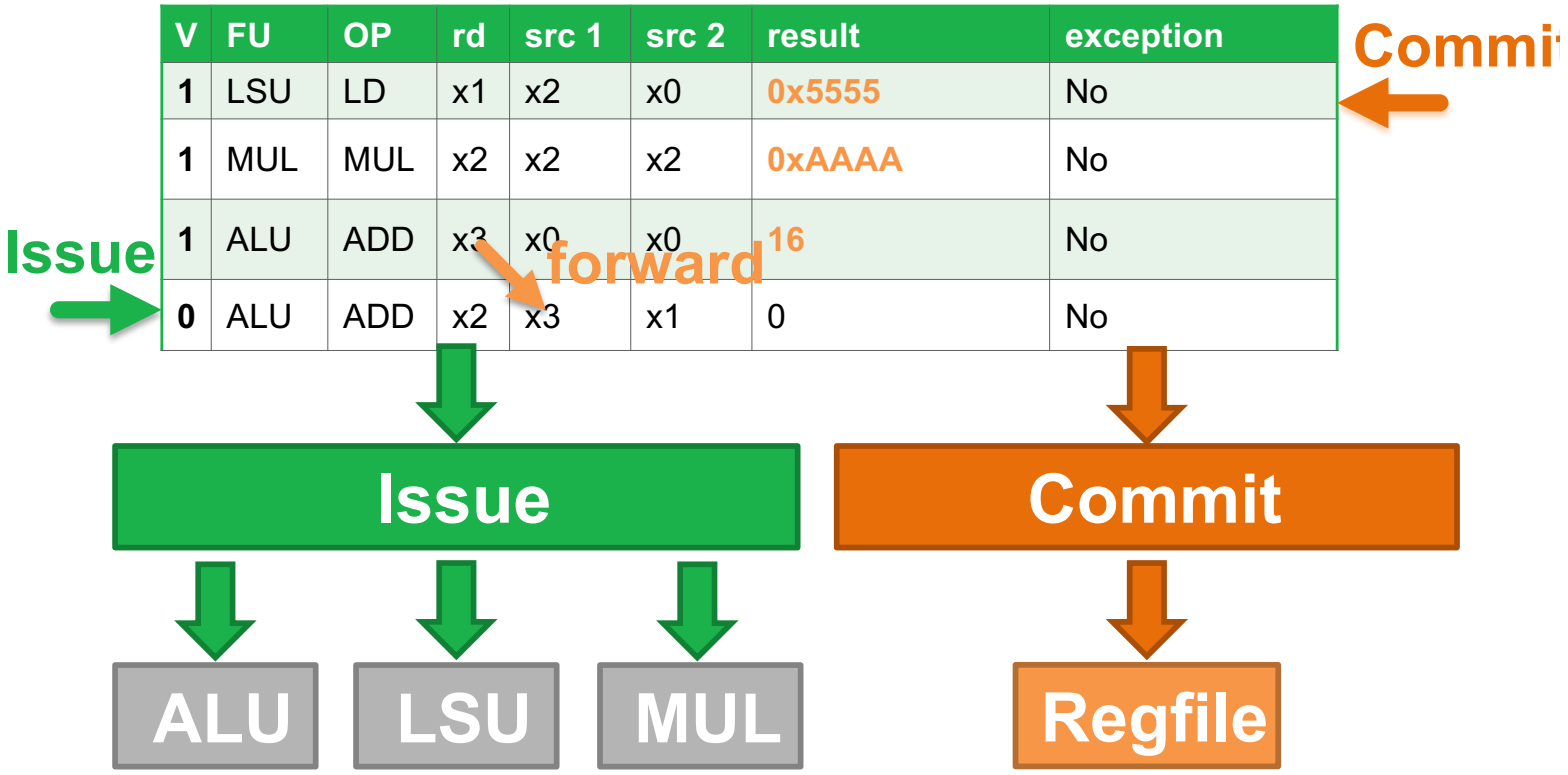
ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

addi x2 ← x3, x1

ld x1 ← x2(128)



Scoreboard - Commit

Decode:

ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

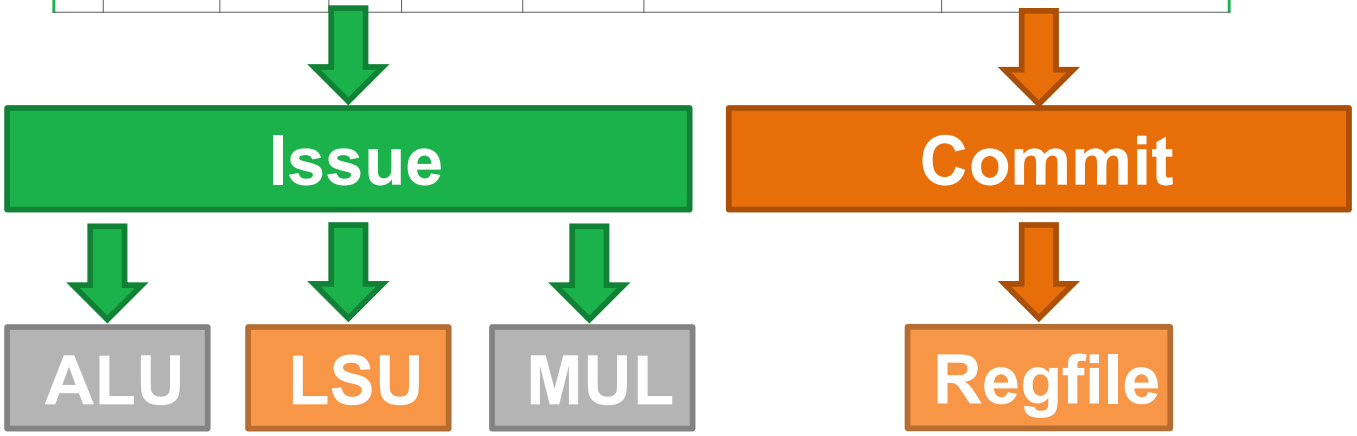
addi x2 ← x3, x1

ld x1 ← x2(128)

Issue

V	FU	OP	rd	src 1	src 2	result	exception
0	LSU	LD	x1	x2	x0	128	No
1	MUL	MUL	x2	x2	x2	0xAAAA	No
1	ALU	ADD	x3	x0	x0	16	No
1	ALU	ADD	x2	x3	x1	0xAABA	No

Commit



Scoreboard – Exception

Decode:

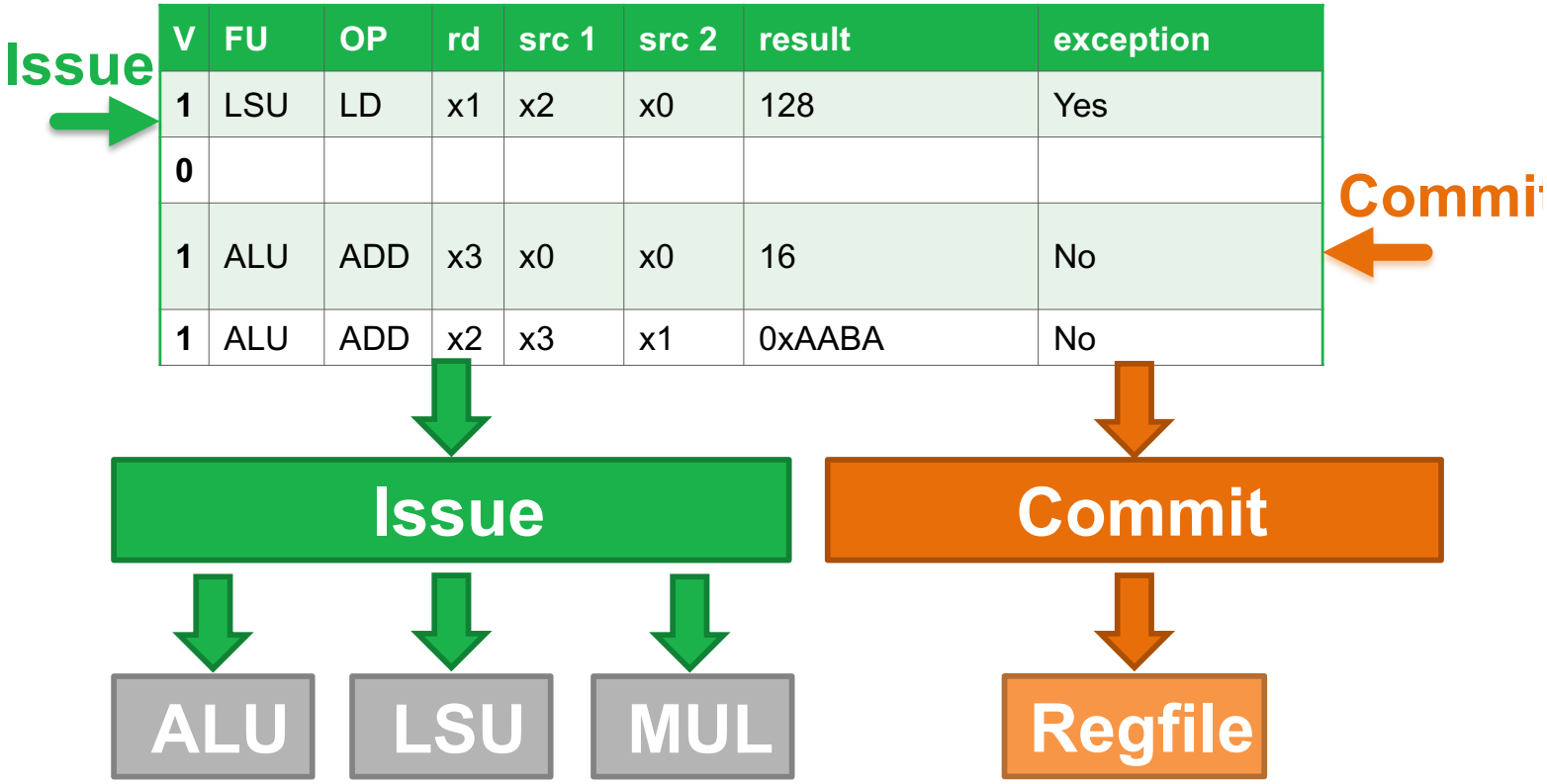
ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

addi x2 ← x3, x1

ld x1 ← x2(128)



Scoreboard - Commit

Decode:

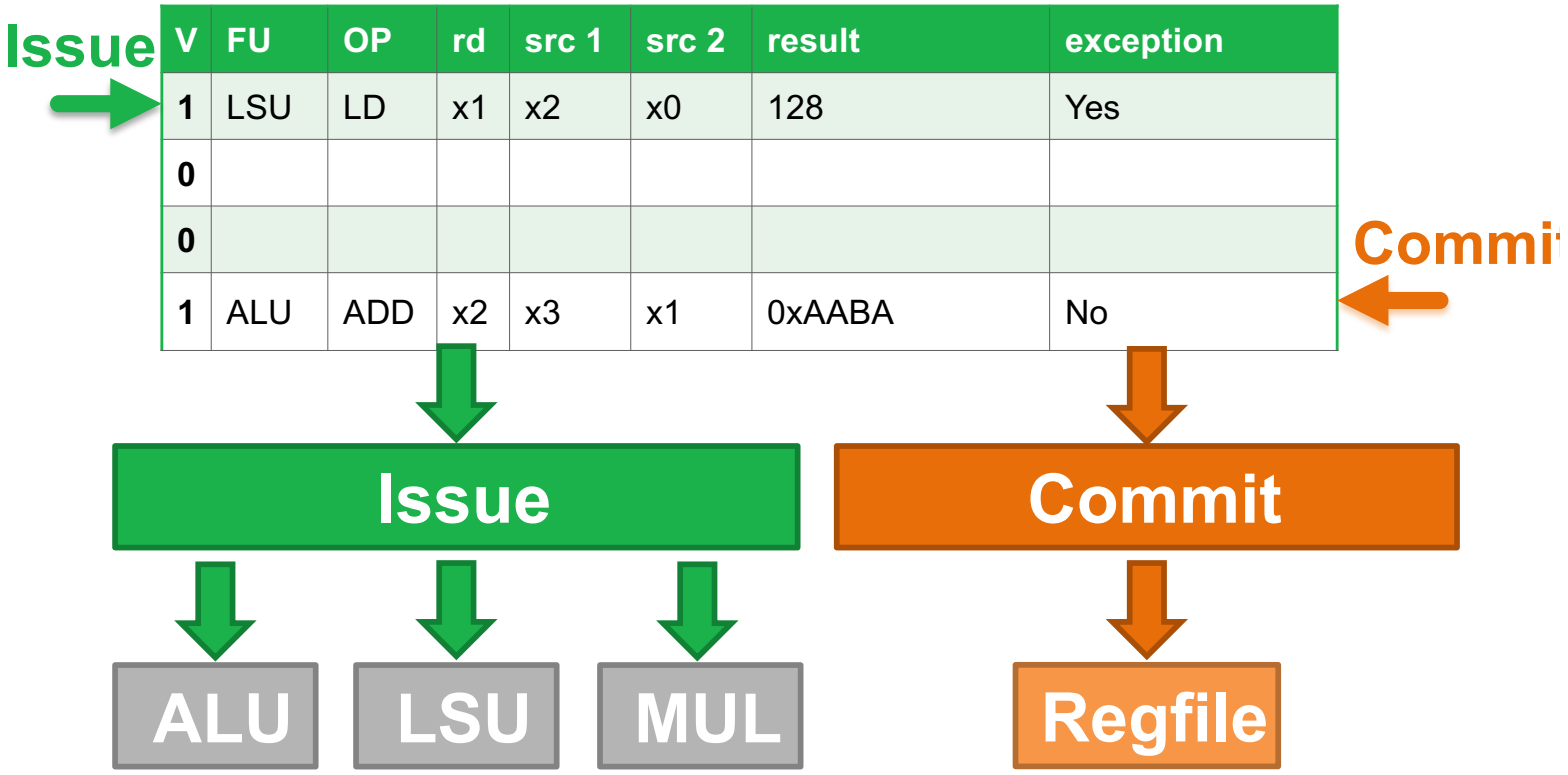
ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

addi x2 ← x3, x1

ld x1 ← x2(128)



Scoreboard - Commit

Decode:

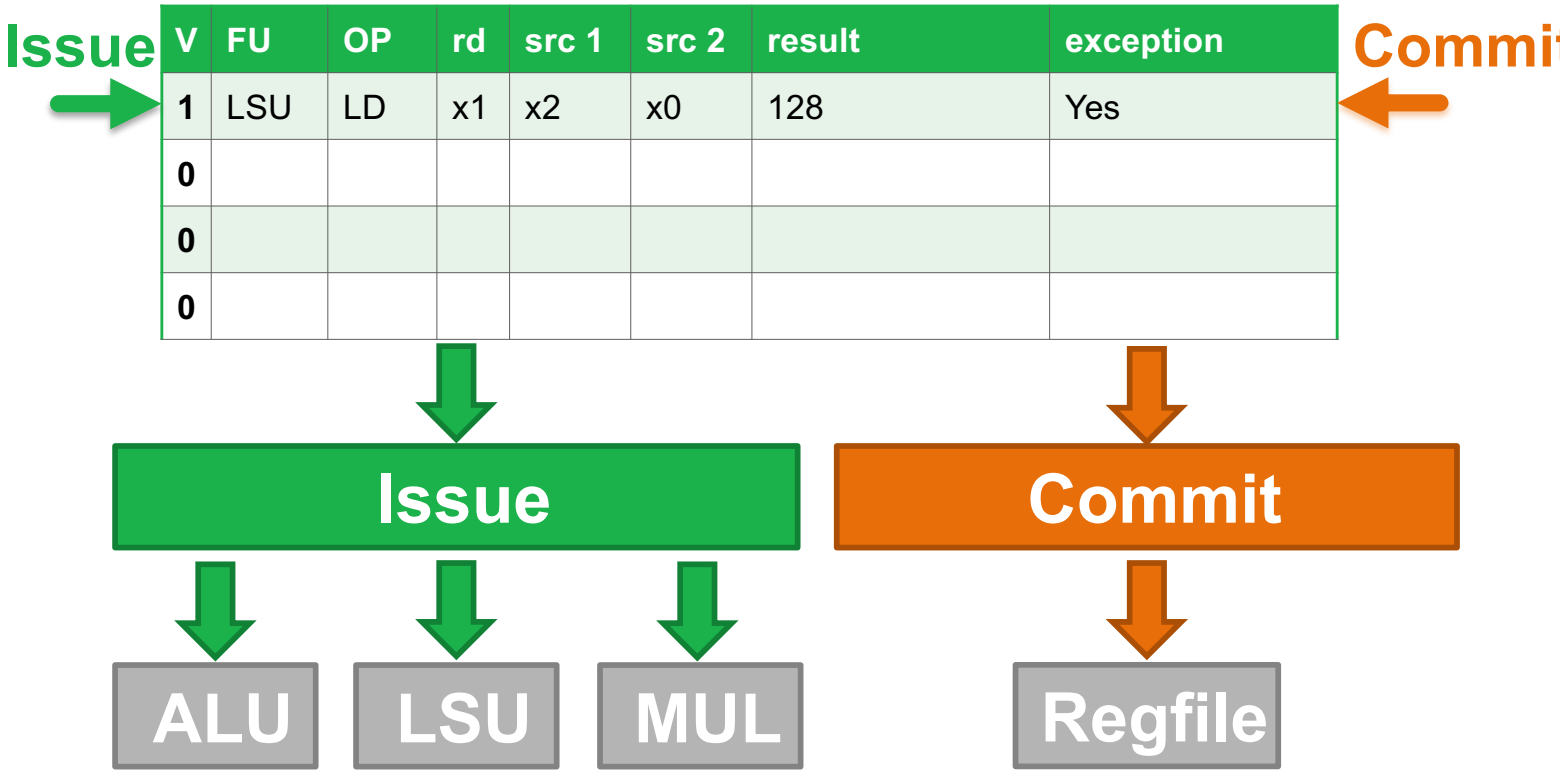
ld x1 ← x2(0)

mul x2 ← x2, x2

addi x3 ← x0, 16

addi x2 ← x3, x1

ld x1 ← x2(128)



Scoreboard - Commit

Decode:

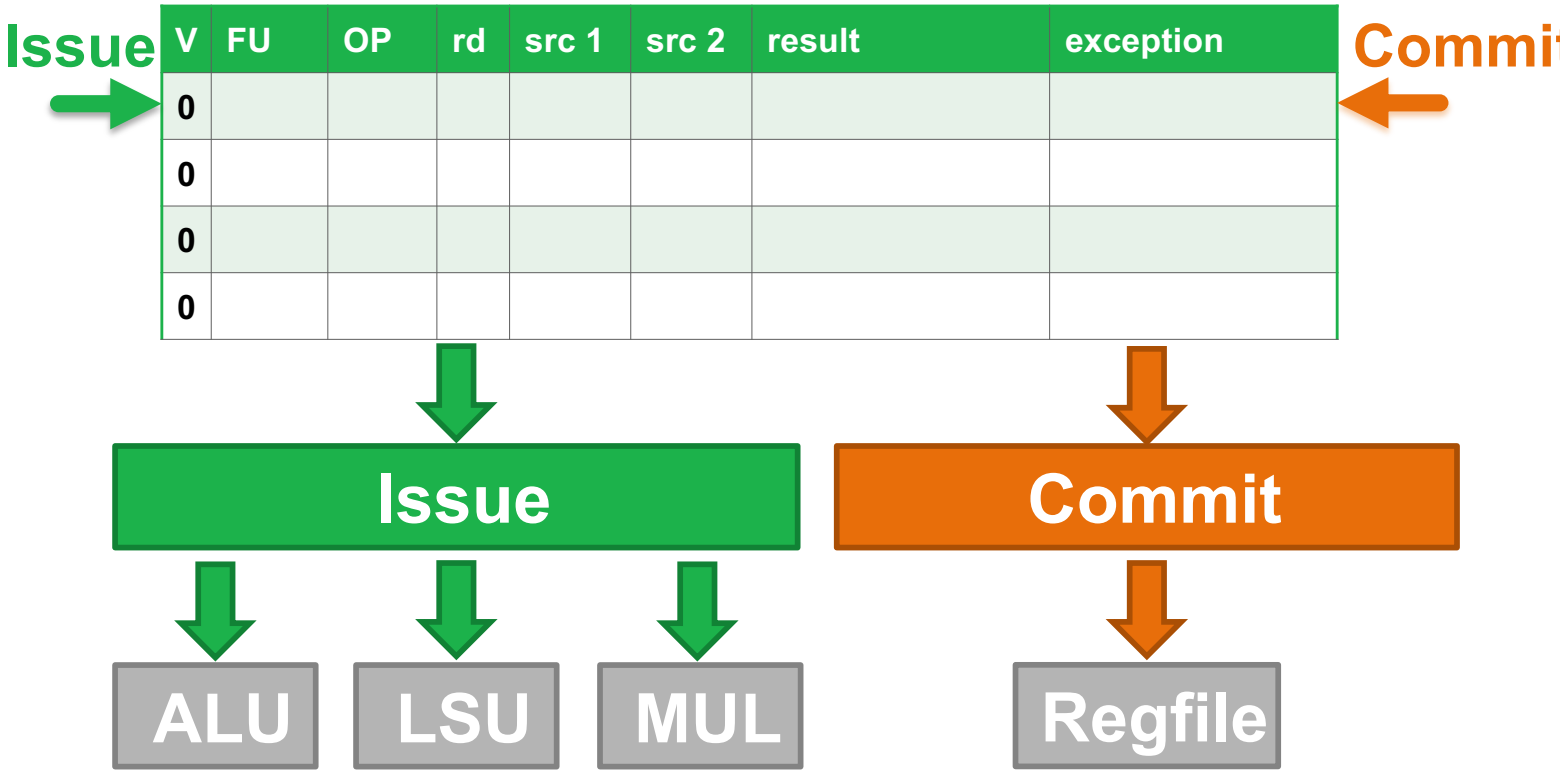
ld x1 ← x2(0)

mul x2 ← x2, x2

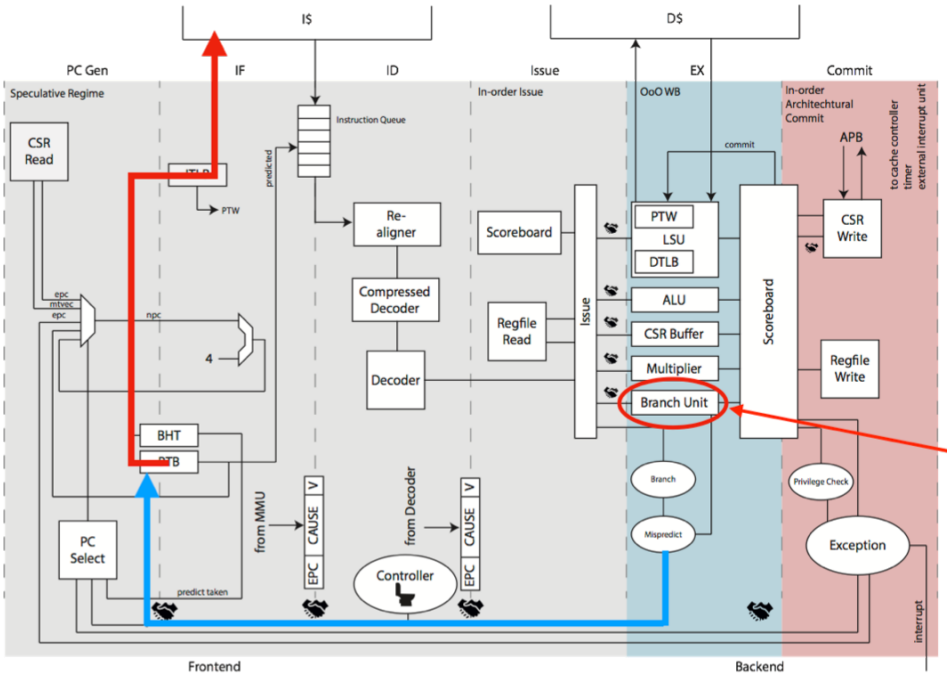
addi x3 ← x0, 16

addi x2 ← x3, x1

ld x1 ← x2(128)



Branch-prediction



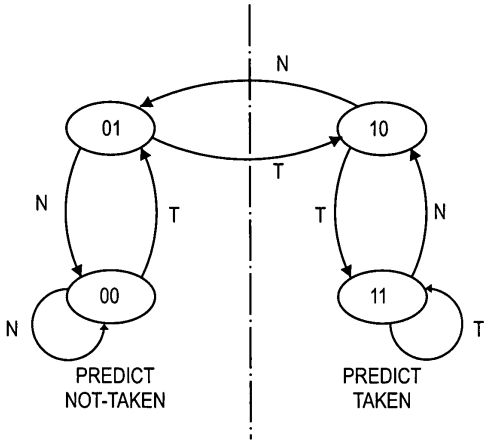
Control Hazards

→ **Branch Prediction**

taken?

bneq x4, x6, pc + 16
addi x3, x2, 10

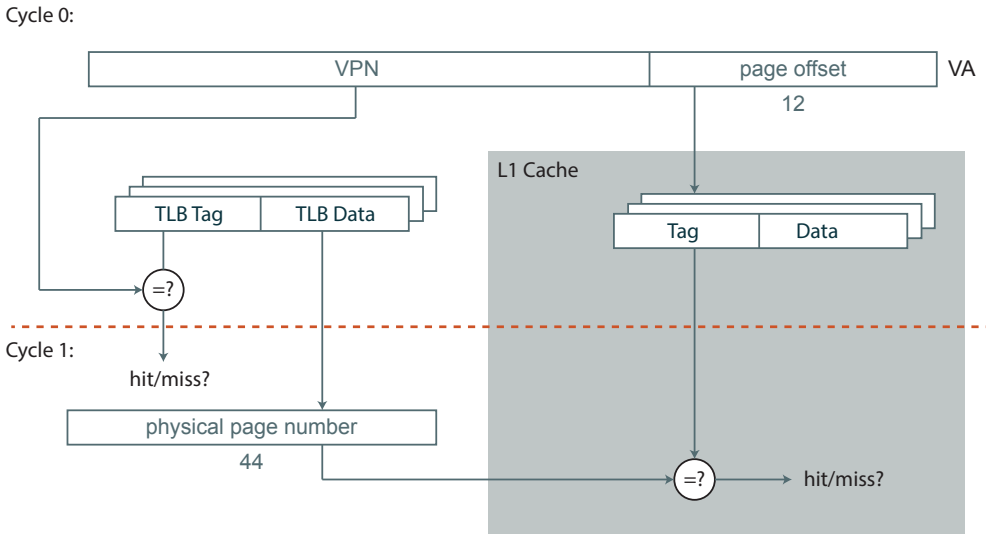
- **Branch Target Buffer**
- **Branch History Table**
  - 2-bit saturation counter
- Anti-aliasing bits



Caches

- Caches are a necessity for larger systems
- Private L1 caches
  - **I$ (16 kByte, 32 entries, 16 byte cache line, 4 way) - 1,41% MR** (Linux Boot)
  - **D$ (32 kByte, 32 entries, 16 byte cache line, 8 way) - 3,17% MR** (Linux Boot)
- Currently no (shared) L2 cache

- SRAMs (cache memories) are slow compared to regular logic
- Virtually indexed, physically tagged data cache



PULP

Parallel Ultra Low Power

[1]*Department of Electrical, Electronic and Information Engineering*

ETH zürich

[2]*Integrated Systems Laboratory*

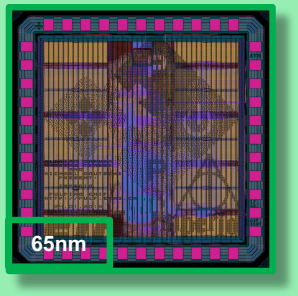
From Cores to (ULP) SoC Platforms

PULP

Open sourced

Standalone µC

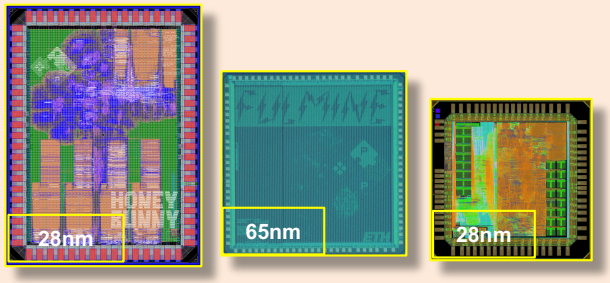
- PULPino
- Ease of use



To be Opened

Multicore Cluster

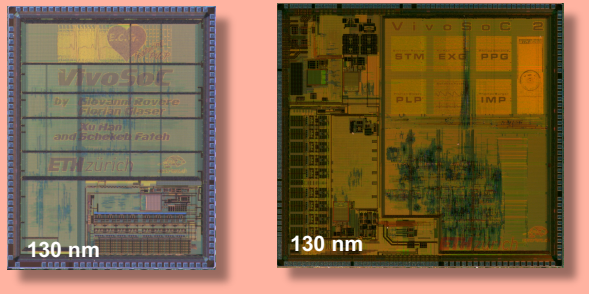
- PULP
- Research



Closed

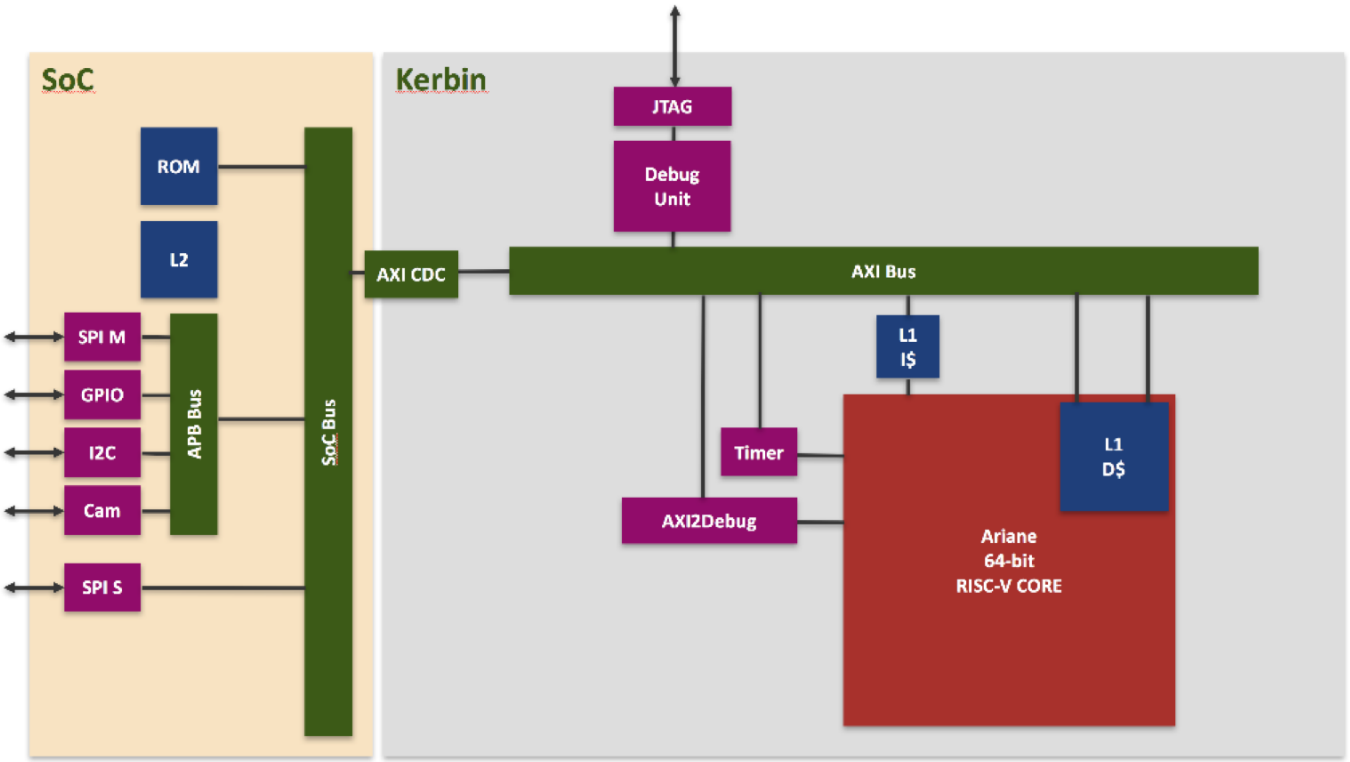
Mixed signal

- VivoSoC
- Healthcare



PULP Multitherman ExaNoDe PRECOMP Open Transprecision Computing ETH

Kerbin: Proof of concept SoC for Ariane



SoC

- PULP Peripherals

CLUSTER

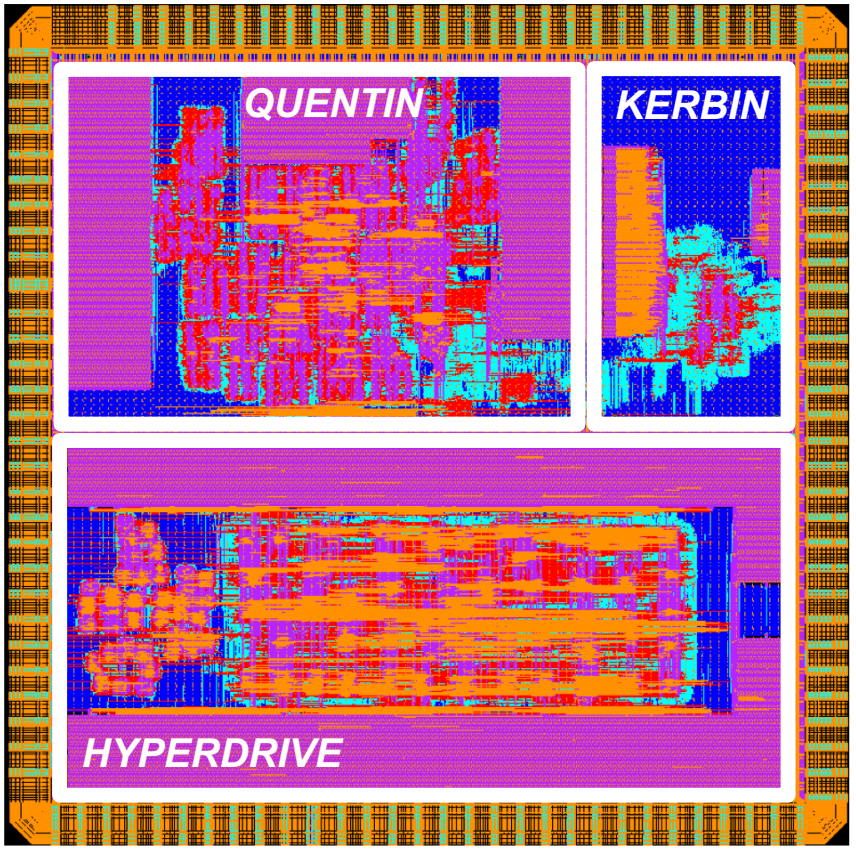
- 64-bit interconnect
- Debug support

→ **Taped out in Globalfoundries 22 nm FDX in January**

Poseidon

- First 22FDX Testchip of the labs
- Goals:
  - Explore performance/energy efficiency:
    - In the low-power domain → LVT
    - In the high-performance domain → SLVT
  - Explore the voltage range of the technology (LVT + SLVT)
  - Explore body biasing
- Test SoCs:
  - **Poseidon**: ultra-low-power 32-bit microcontroller
  - **Kerbin**: high-performance 64-bit processor
  - **Hyperdrive**: Convolutional Neural Networks Accelerator

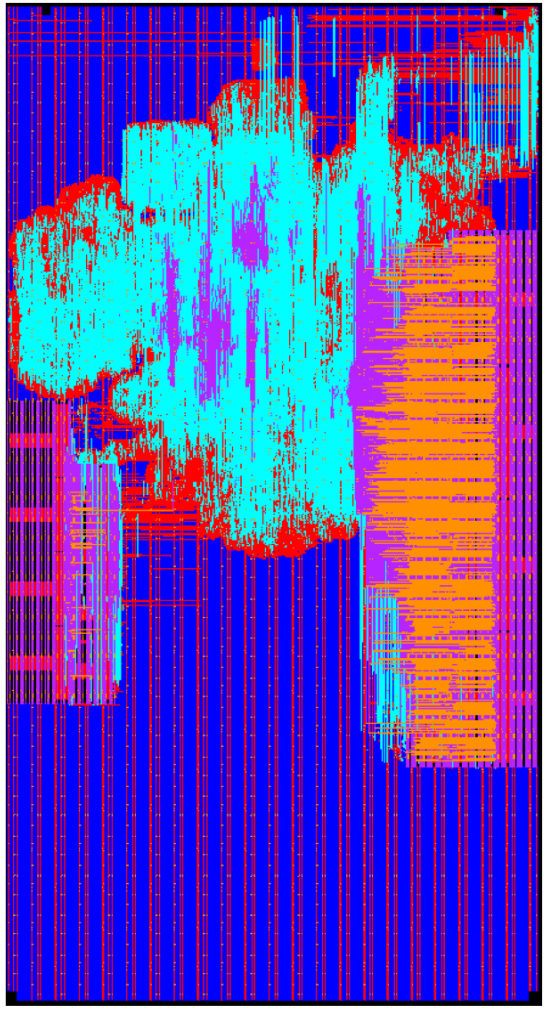
Poseidon layout



PULP Multitherman ExaNoDe PRECOMP Open Transprecision Computing ETH

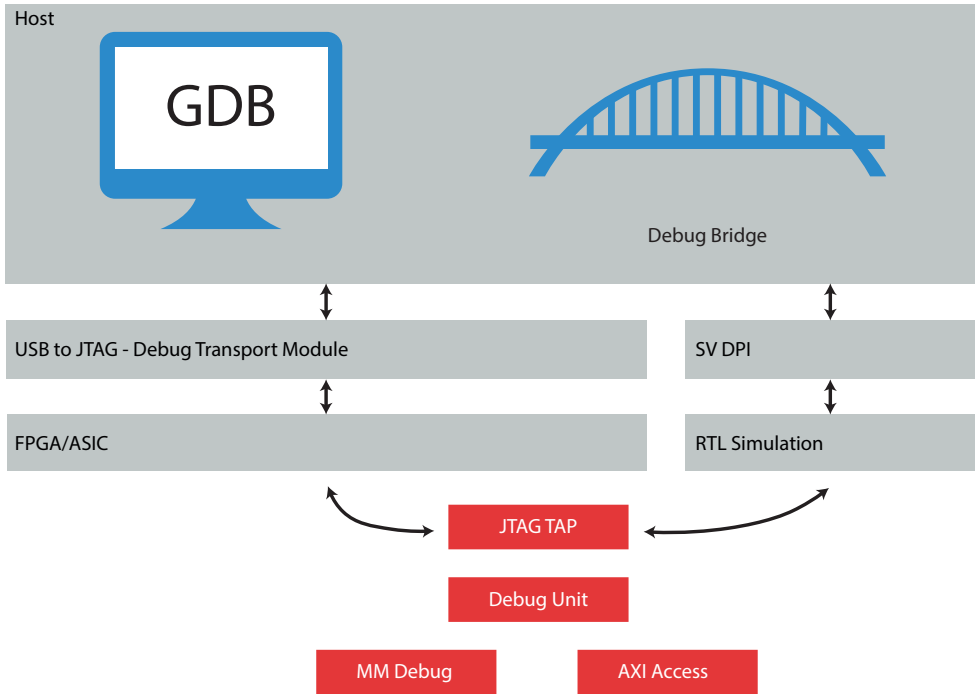
Kerbin Implementation Results

- Silicon implementation in GF22FDX, mixed LVT and SLVT libraries.
- The system features 32 kByte of instruction and 32 kByte of data cache.
  - D$: 8-way, 4 kByte
  - I$: 4-way, 8 kByte
- Timing closure: 910 MHz @ SSG, 125/-40 °C, 0.72V – NO BB
- Area: 0.23 mm2 – 175 kGE



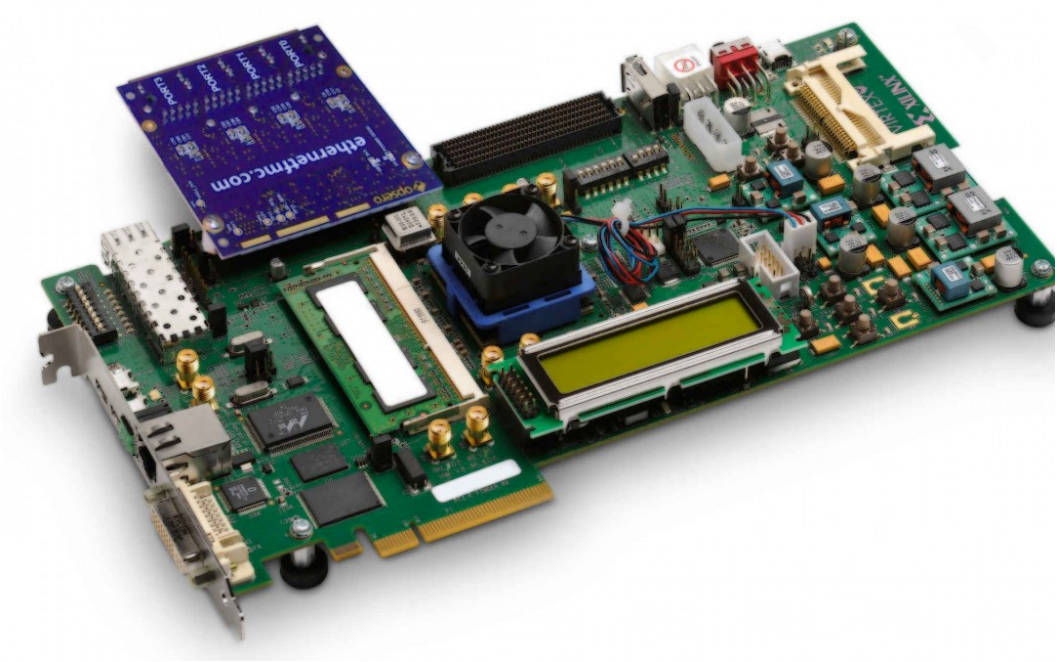
Full Debug support

- Leveraging existing infrastructure: RISC-V GDB
- Debug Bridge to communicate with hardware
- Allows for:
  - run-control
  - single-step
  - inspection
  - (hardware) breakpoints
- Essential for SW debug and hardware bring-up



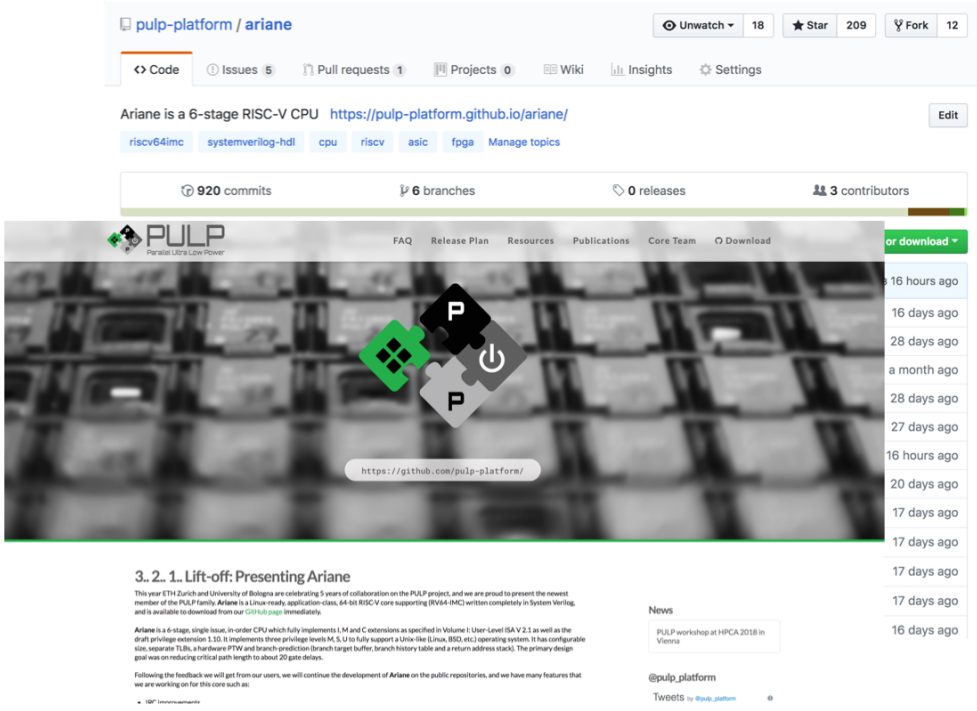
Status Quo

- Full FPGA implementation
  - Xilinx Vertex 7 – VC707
  - **Core:** 50 – 100 MHz
  - **Core:** 15 kLUTs
  - 1 GB DDR3
  - Booting Linux (Demo)
- FPGA implementation allows for fast prototyping – HW/SW codesign
- Area allows for exploration of multi-processor systems



It is Open Source!

- The whole core has been open-sourced
- **OpenSource toolflow for easy exploration**: Verilator
- QuestaSim (commercial simulator) toolflow for more thorough use-cases
- Further development entirely in the open:
  - Further IPC improvements
  - Floating point support
  - Atomic Support
- FPGA mapping following soon!



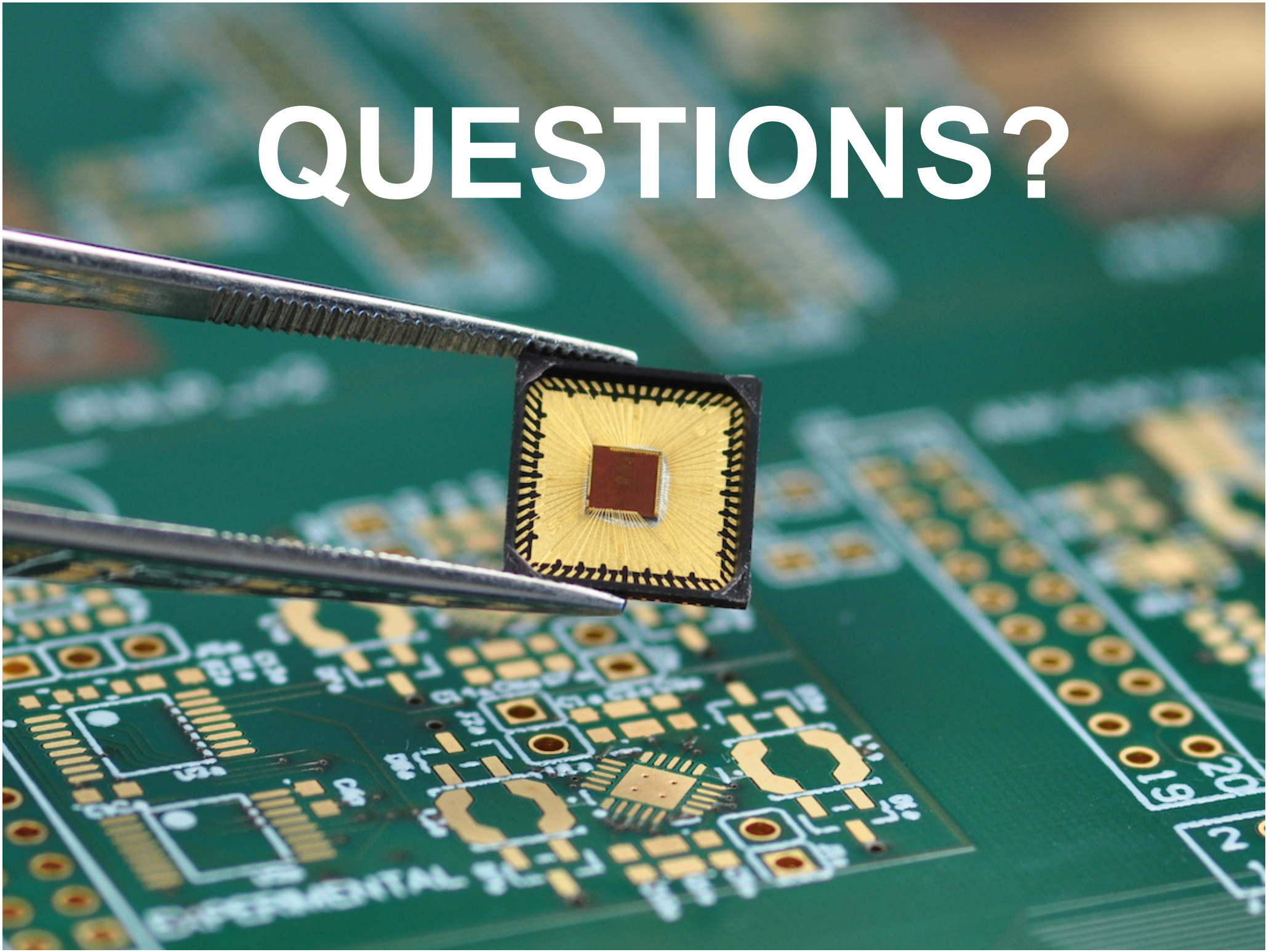
→ **https://github.com/pulp-platform/ariane**

Structure of this workshop

- Open source hardware and our role (Frank)
- The PULP family tree (Frank)
- Our RISC-V cores: Ariane, RI5CY and friends (Florian)
- **Break – Demos**
- **Accelerators in PULP (Francesco)**
- **Our Programmable Multi-Core Accelerator – HERO (Andreas)**
- **Programming PULP (Andreas)**

Please interrupt at any time to ask questions

PULP Multitherman ExaNoDe PRECOMP Open Transprecision Computing ETH


QUESTIONS?