RISC-V System-on-Chip Design

Wally Open-Source RISC-V Core & SoC with Accompanying Textbook

Professor Sarah L. Harris, Dept. of Electrical & Computer Engineering

UNLV | HOWARD R. HUGHES COLLEGE OF ENGINEERING

Overview

- Background: Why RISC-V?
- Why Wally?
- Wally Details

Background: Why RISC-V?

First major non-proprietary computer architecture

- 5th generation Reduced Instruction Set Computer (thus, RISC-V)
- Developed at Berkeley in 2010 – development ongoing
- Simple instructions
  - Avoids many “cute tricks” (like conditional execution, branch delay slot, etc.) that proved overly clever in older architectures
- No patents or licensing agreements required
- Driven by non-profit RISC-V International

Huge momentum

- Likely to displace all architectures besides x86 and ARM
- 10 billion cores shipped by 2022

Why Wally?

Goals:

- The main goal of developing the Wally SoC and its accompanying textbook is:
  - To take users through **building a real full-featured SoC** containing a RISC-V processor, memories, and peripherals.
- Most architecture textbooks focus on high-level principles
- This textbook will show you **both principles and detailed implementation**

Wally features:

- Configurable, open-source core & SoC
- 32 or 64 bit
- Bus, cache, memory management, branch prediction
- Floating-point unit
- Supports other standard extensions
- Peripherals
- Boots Linux on an FPGA

Advantages of Wally over Other Cores/SoCs

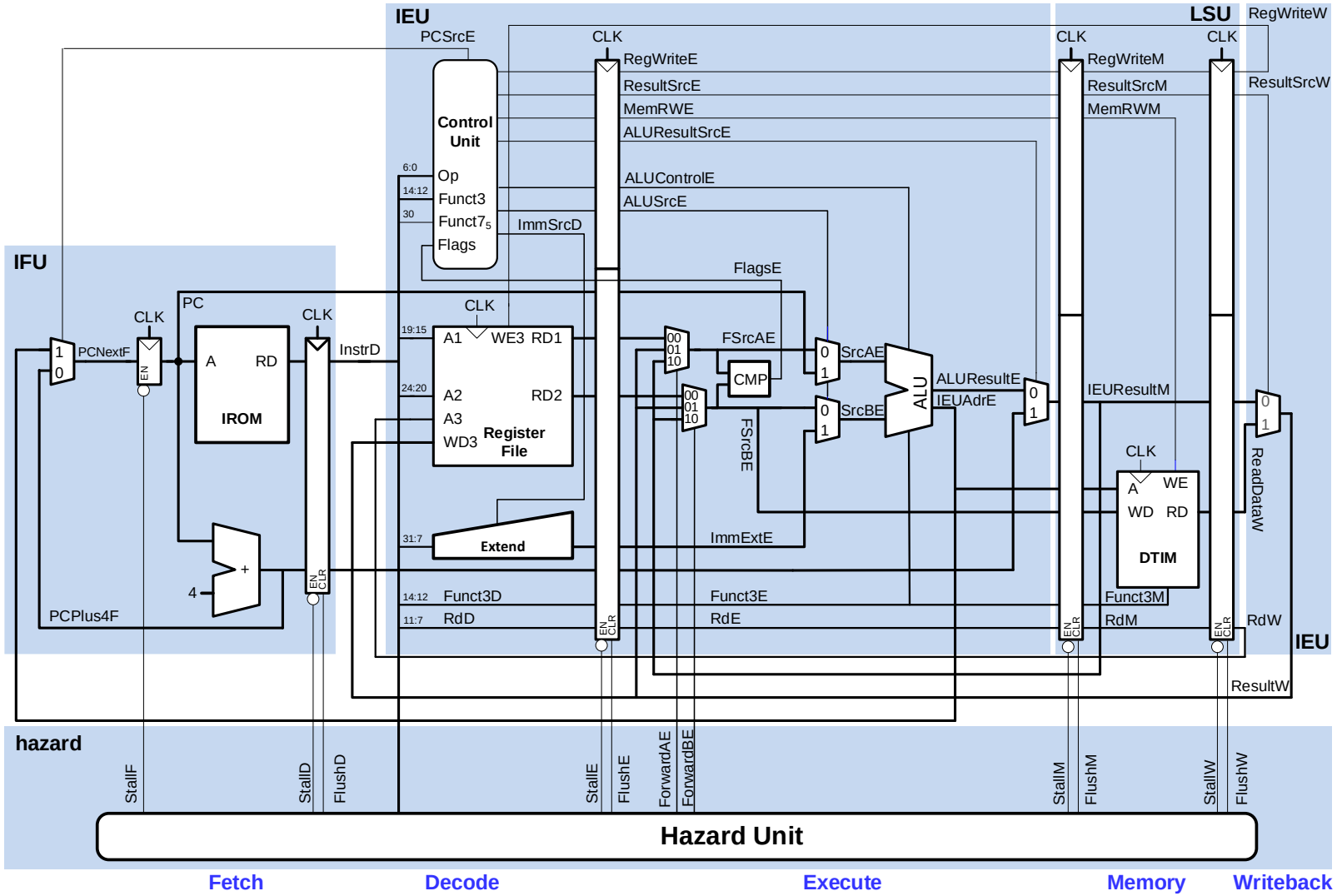
- **Configurable**
  - From a small, embedded core (rv32e) to high-performance system (rv64gc)
- **Boots Linux** (some other open-source cores do not, including SweRV/VeeR cores)
- The Wally open-source SoC is accompanied by a **textbook** that provides:
  - Detailed explanations about the **principles** – and how to put them into **practice**
  - Detailed **schematics** of Wally building blocks
  - Wally **source code** (SystemVerilog): https://github.com/openhwgroup/cvw
    - Open-source
    - Written to be understood by users
    - Signals and module names correspond to those in the textbook diagrams
  - Publication date: ~Early 2025

Wally Details

- Textbook Table of Contents
- Wally SoC Overview and Schematics

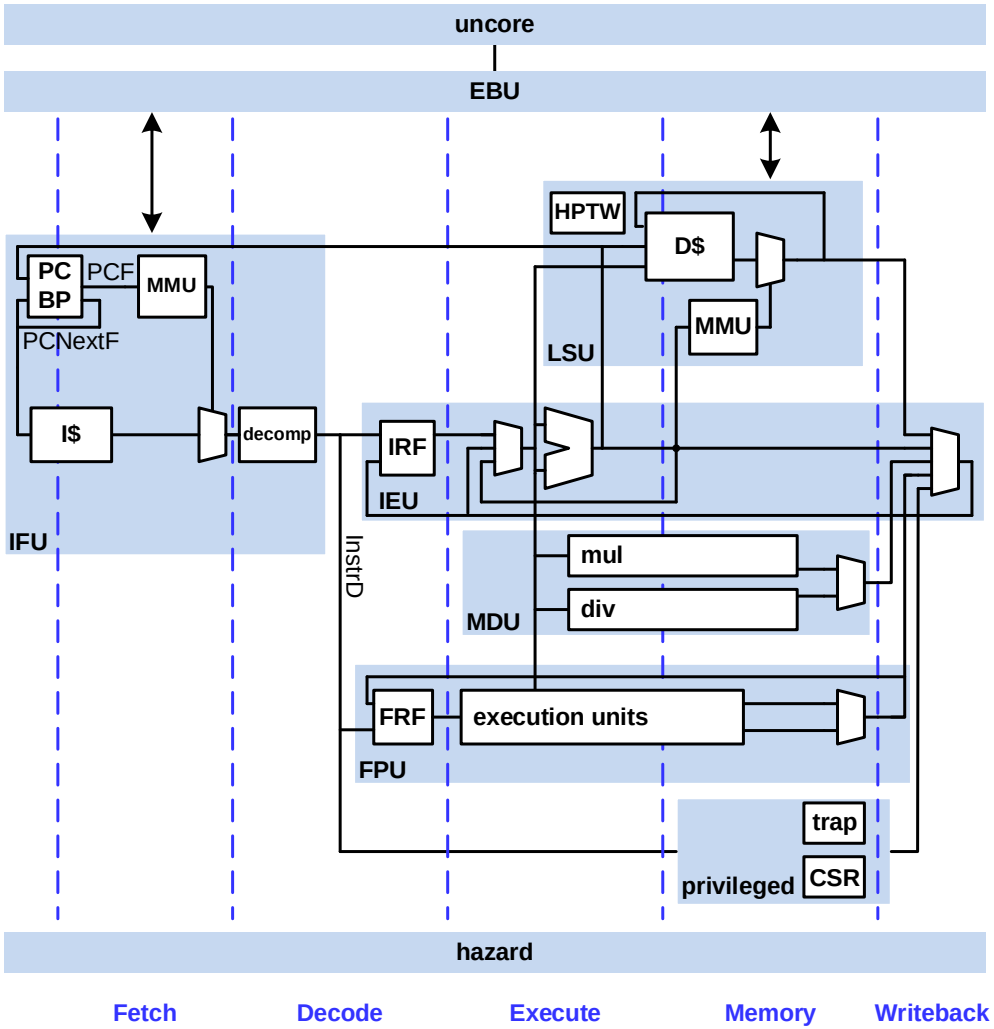
RISC-V SoC Design: Table of Contents

Wally SoC: Pipelined Processor



- **5-stage pipeline:**
  - Fetch
  - Decode
  - Execute
  - Memory
  - Writeback
- Looks similar to the simplified pipelined RISC-V processor in *Digital Design and Computer Architecture: RISC-V Edition* textbook

Wally SoC: Top-Level Schematic



- **Traditional 5-stage pipeline:**
  - Fetch, Decode, Execute, Memory, Writeback
- **Main units:**
  - Fetch & Decode Stages:
    - IFU: Instruction Fetch Unit
  - Execute Stage (parallel paths)
    - IEU: Instruction Execute Unit
    - MDU: Multiply/Divide Unit
    - FPU: Floating-Point Unit
  - Memory Stage:
    - LSU: Load/Store Unit
    - priv: Privileged Unit
  - Writeback Stage writes to:
    - IRF: Integer Register File (in IEU) or
    - FRF: Floating-Point Register File (in FPU)

Wally Supported Features

- RV32I & RV64I (XLEN = 32 or 64)
- Supported Extensions
  - A: Atomic
  - C: Compressed
  - D: Double-precision floating-point
  - E: Embedded 16 registers
  - F: Single-precision floating-point
  - M: Multiplication and division
  - Q: Quad-precision floating-point
  - Zb: Bit manipulation
  - Zicsr: CSRs
  - Zfencei: FENCE.I instruction synchronization
- Privileged Modes and Features
  - Machine, Supervisor, User (M, S, U)
  - Reset vector address
  - Virtual memory (ITLB, DTLB)
  - Physical memory protection
  - Vectored interrupts
  - Counters: performance counting
- Caches and Branch Prediction
- Peripherals
  - Core Local Interrupt Controller with timers (CLINT)
  - Platform Level Interrupt Controller (PLIC)
  - UART
  - GPIO
  - Timers

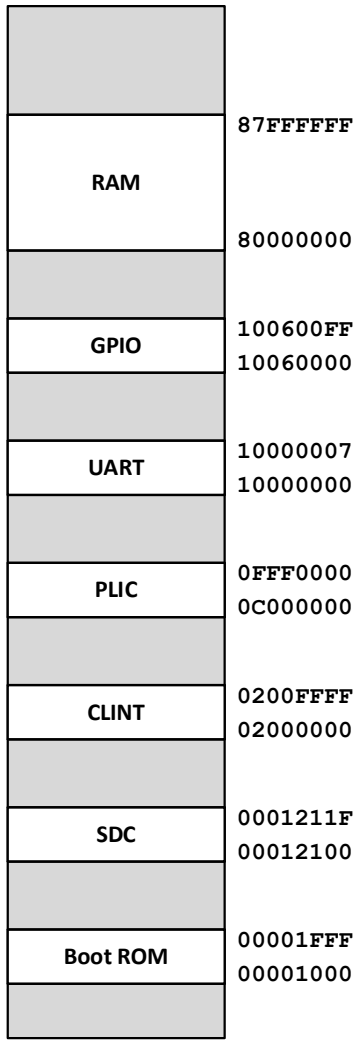
Not Supported in Wally

- Hypervisor mode
- Advanced microarchitecture: superscalar, out-of-order, multithreading, multicore
- Unfrozen extensions
  - E: Embedded (16 registers)
  - L: Decimal floating-point
  - J: Dynamically translated languages
  - T: Transactional memory
  - P: Packed SIMD
  - V: Vector instructions
  - H: Hypervisor

Wally Configurations

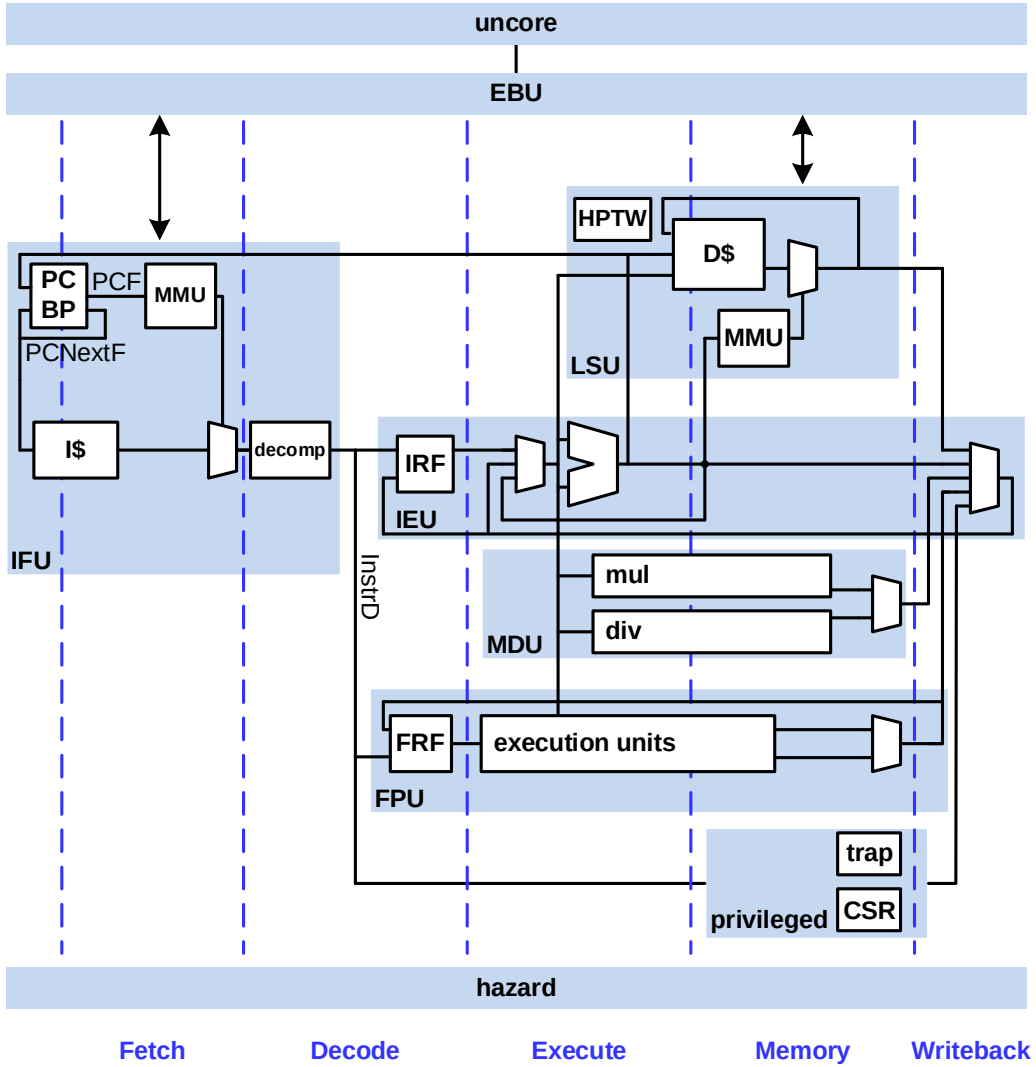
Configuration	Config	XLEN	DTIM / IROM Size	Bus	Periph	I$/D$ Size	Priv. Modes	Virt. Mem
Embedded	rv32e	32 bits	n/a	YES	NO	n/a	none	NO
Simple CPU	rv32i	32 bits	2K / 2K	NO	NO	n/a	none	NO
µcontroller	rv32ic	32 bits	4K / 16K	YES	YES	n/a	MU	NO
Apps Proc	rv32gc	32 bits	n/a	YES	YES	16K	MSU	YES
Simple CPU	rv64i	64 bits	2K / 2K	NO	NO	n/a	none	NO
Apps Proc	rv64gc	64 bits	n/a	YES	YES	16K	MSU	YES

Wally Memory Map



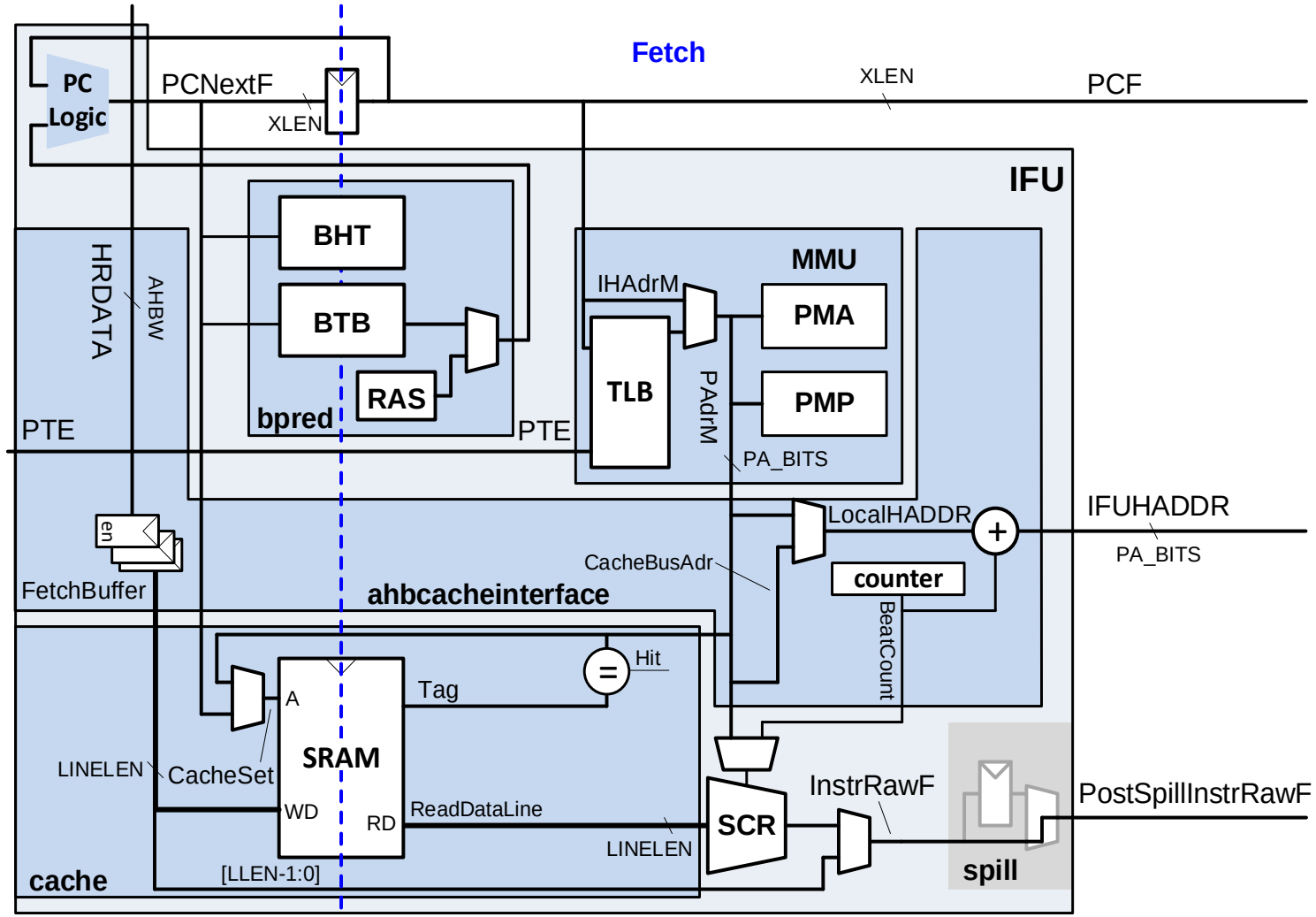
Region	Size	Range
Boot ROM	4 KiB	0x00001000-0x00001FFF
RAM	128 KiB	0x80000000-0x87FFFFFF
CLINT	64 KiB	0x02000000-0x0200FFFF
PLIC	64 MiB	0x0C000000-0x0FFF0000
UART	8 B	0x10000000-0x10000007
GPIO	256 B	0x10060000-0x100600FF
SDC	32 B	0x00012100-0x0001211F

Wally SoC: Dive down into Hierarchy



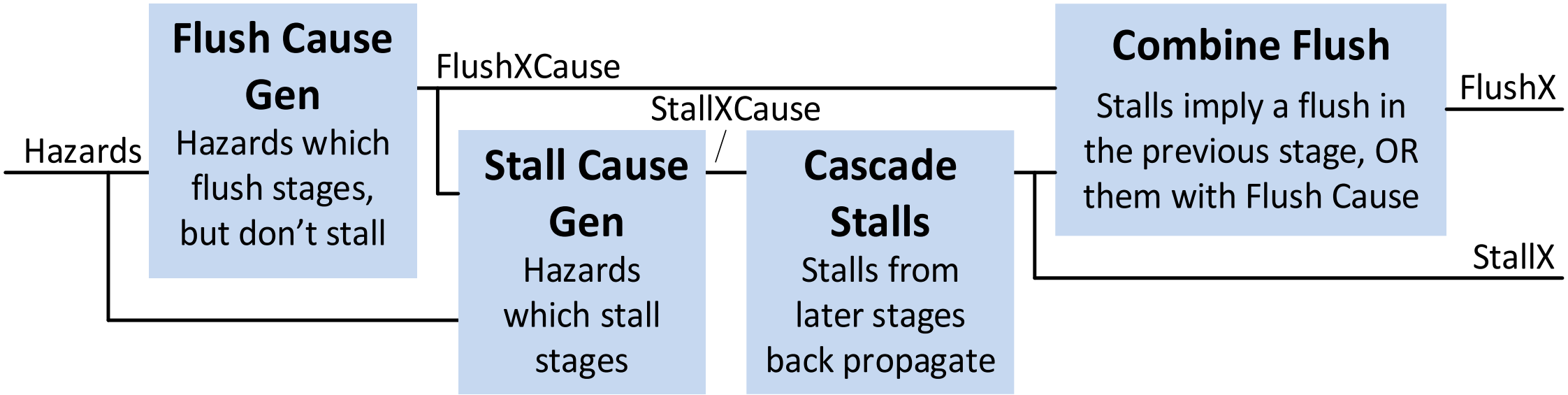
- **IFU:** Instruction Fetch Unit
- **hazard:** Hazard Unit
- **IEU:** Instruction Execute Unit
- **LSU:** Load/Store Unit
- **EBU:** External Bus Unit (interface and arbitration for AHB bus)
- **uncore:** everything except the core (i.e., AHB bus, peripherals, etc.)
- **MDU:** Multiply/Divide Unit
- **FPU:** Floating-Point Unit
- **priv:** Privileged Unit
- **decompress:** Decompress Unit

Wally SoC: IFU (Instruction Fetch Unit)

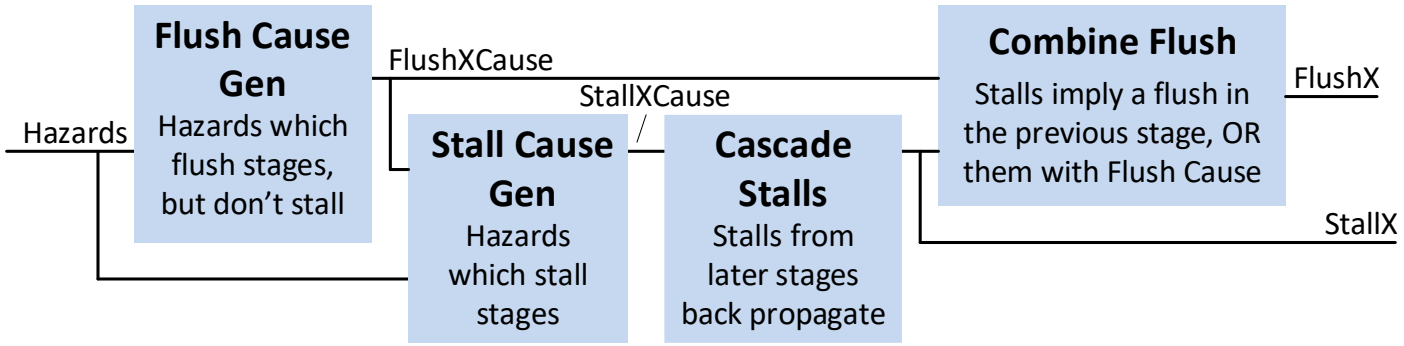


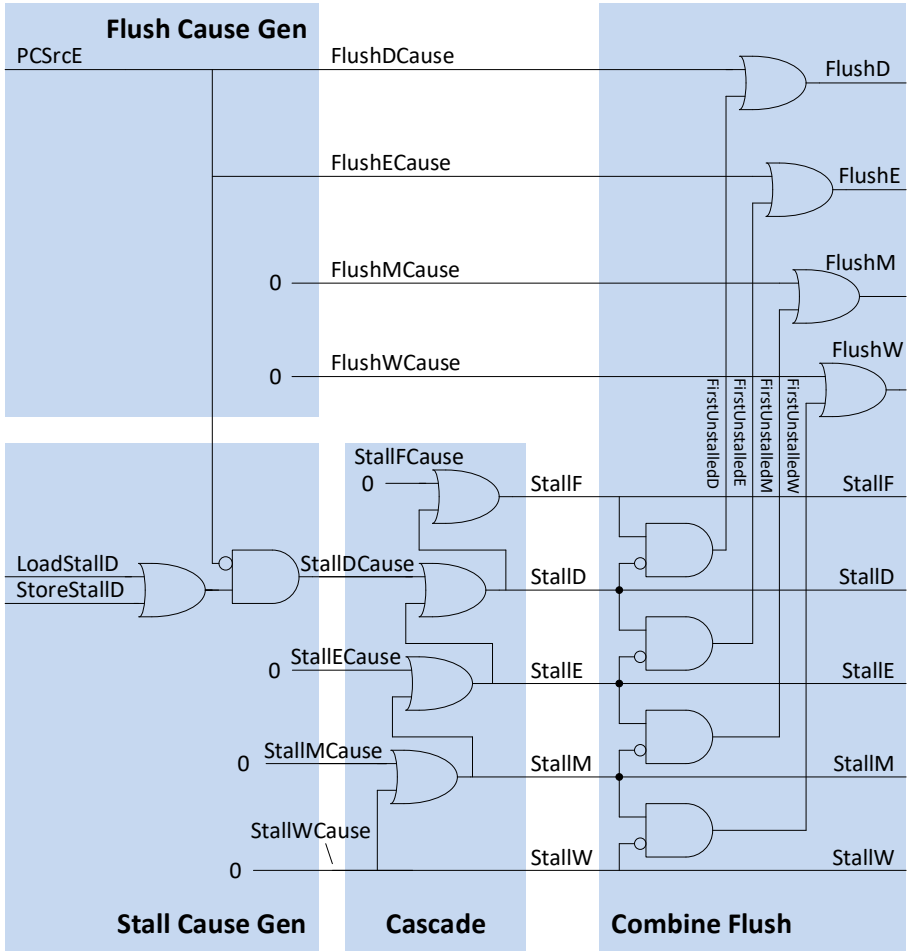
- **bpred:** Branch Prediction
- **cache:** Instruction Cache
- **ahbcacheinterface:** AHB Cache Interface for cache misses
- **PC Logic**

Wally SoC: Hazard Unit Architecture

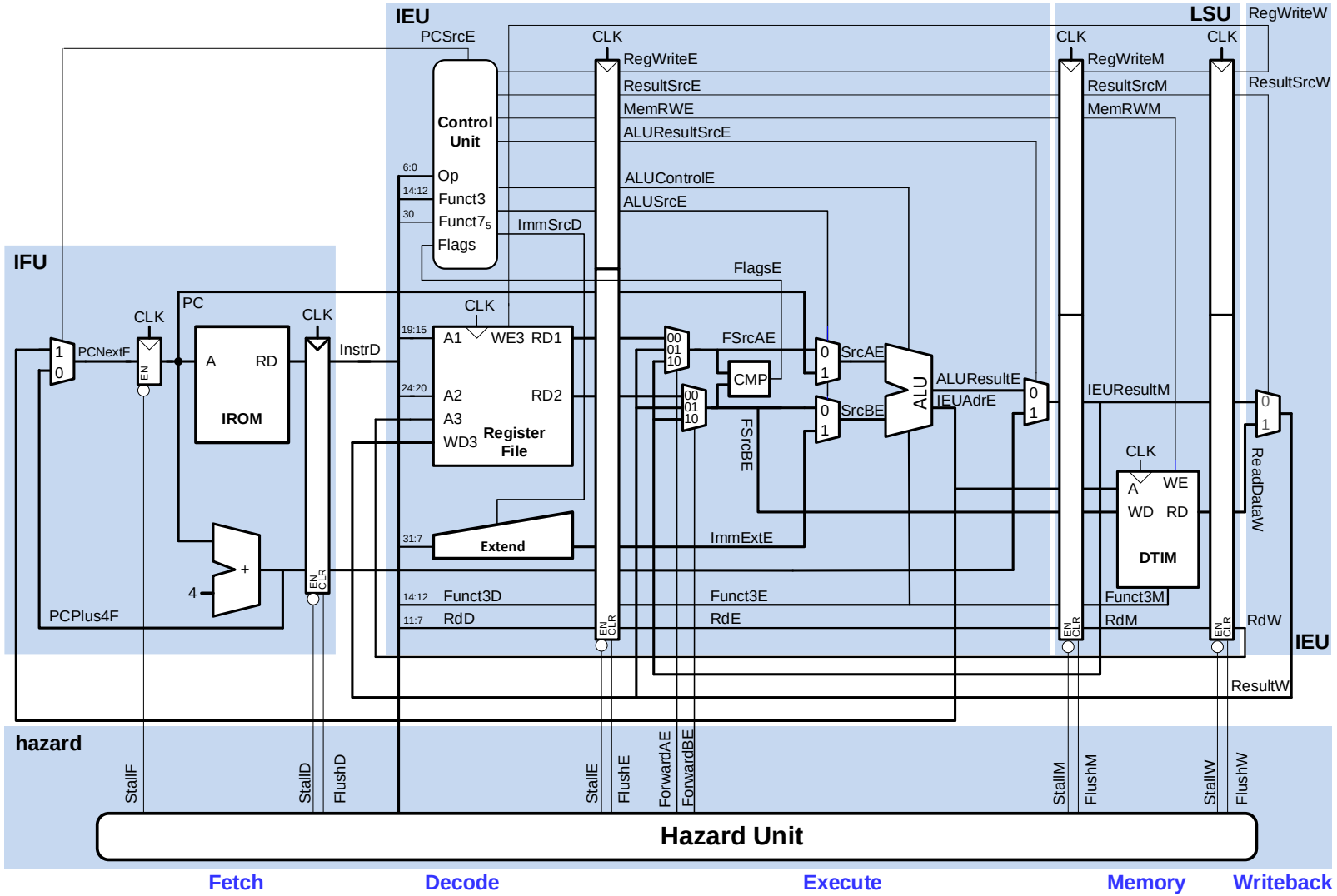


Wally SoC: Hazard Unit



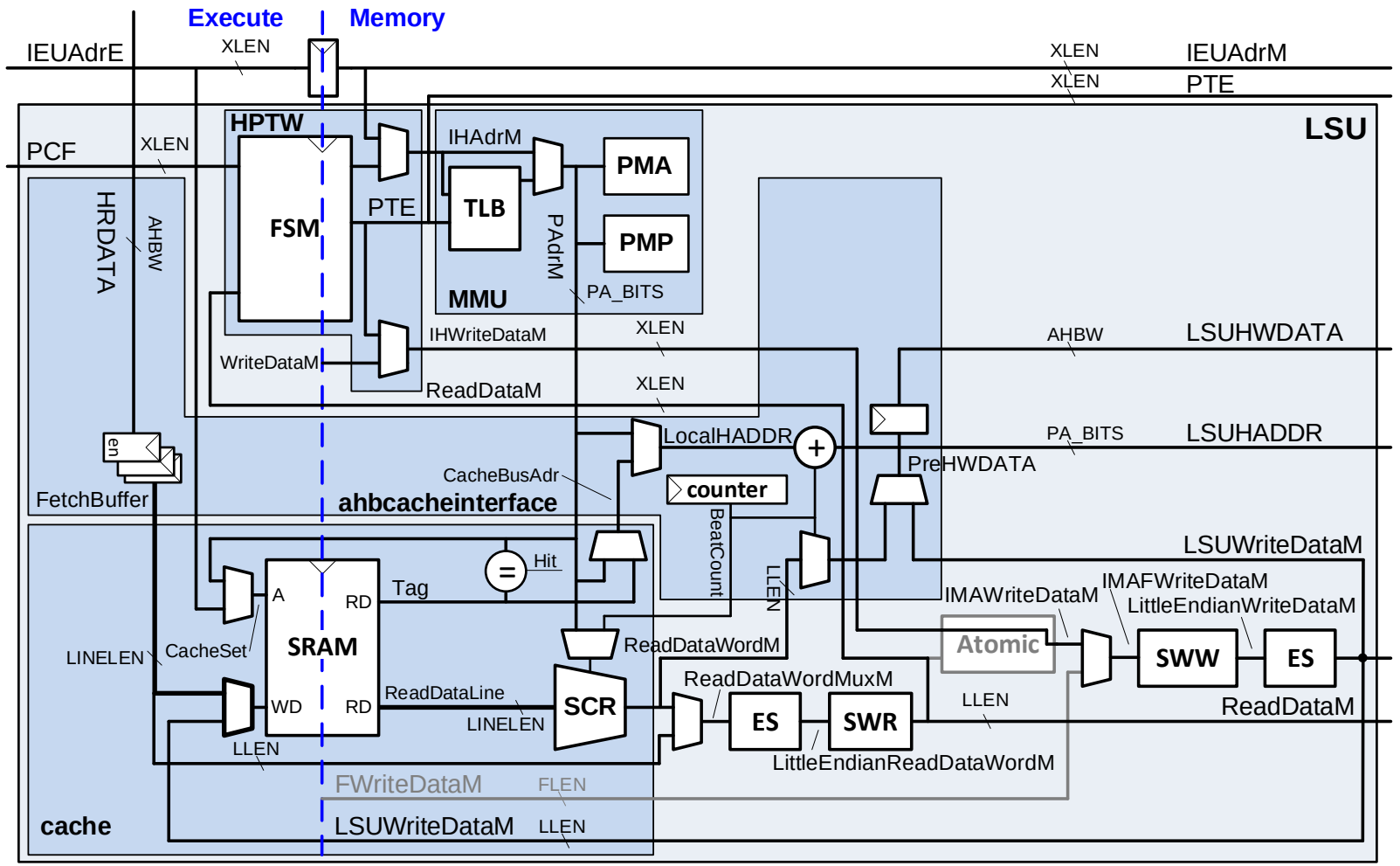


Wally SoC: IEU (Instruction Execute Unit)



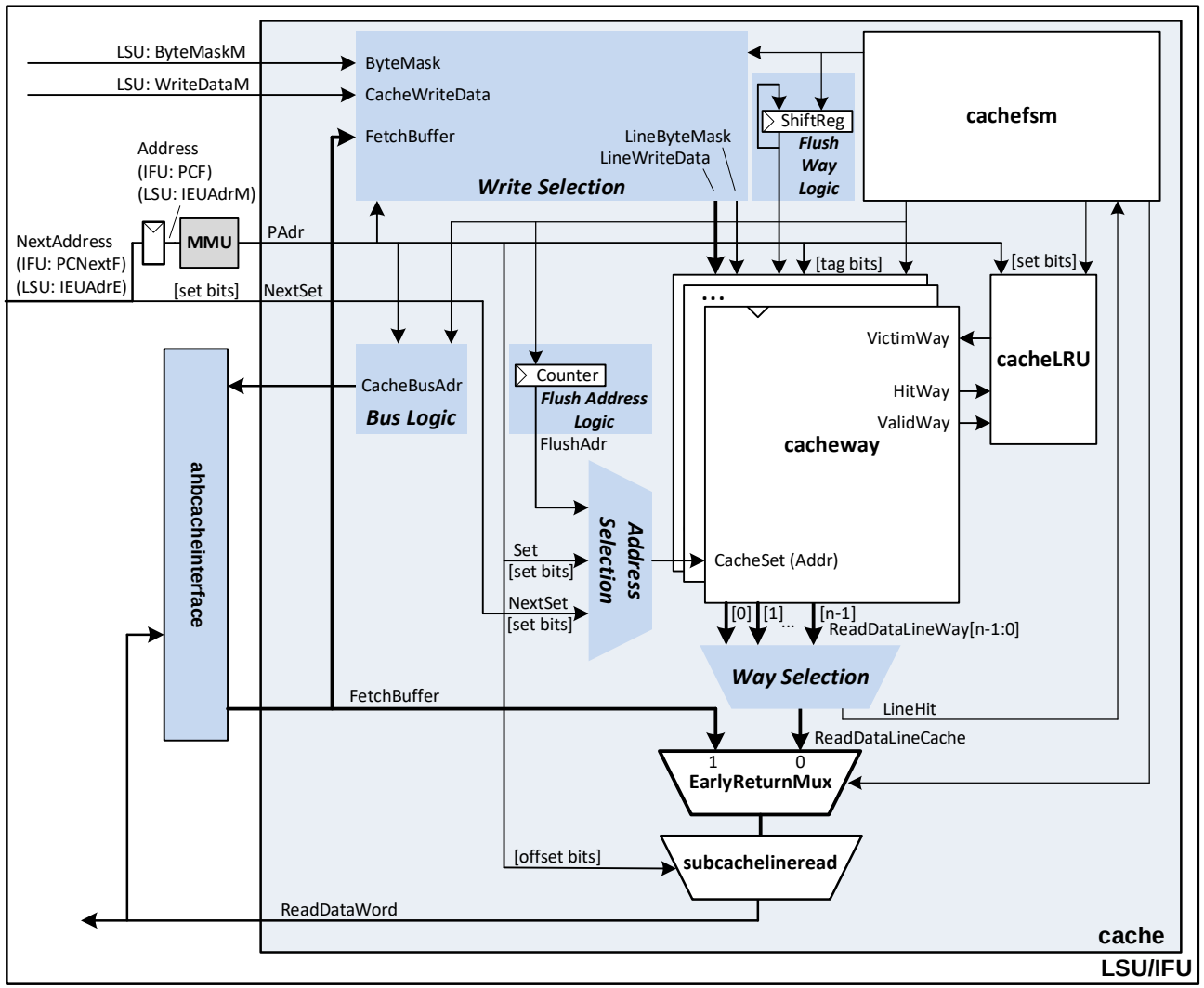
- ALU
- Compare unit (CMP)
- Hazard Unit
- Multiplexers for forwarding

Wally SoC: LSU (Load/Store Unit)



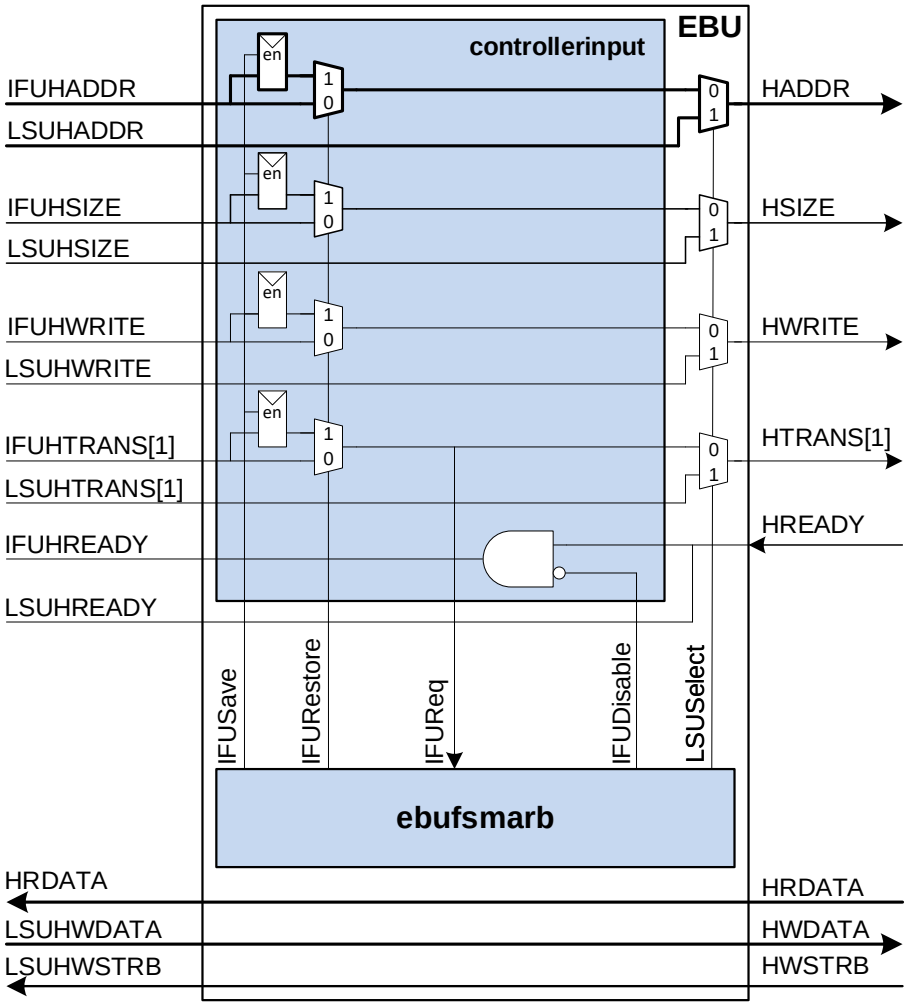
- MMU (Memory Management Unit)
- Cache
- Bus interface (ahbcacheinterface)
- Modules for dealing with endianness and subword accesses (ES: endian swap, SWW: subword write, and SWR: subword read)

Wally SoC: Cache



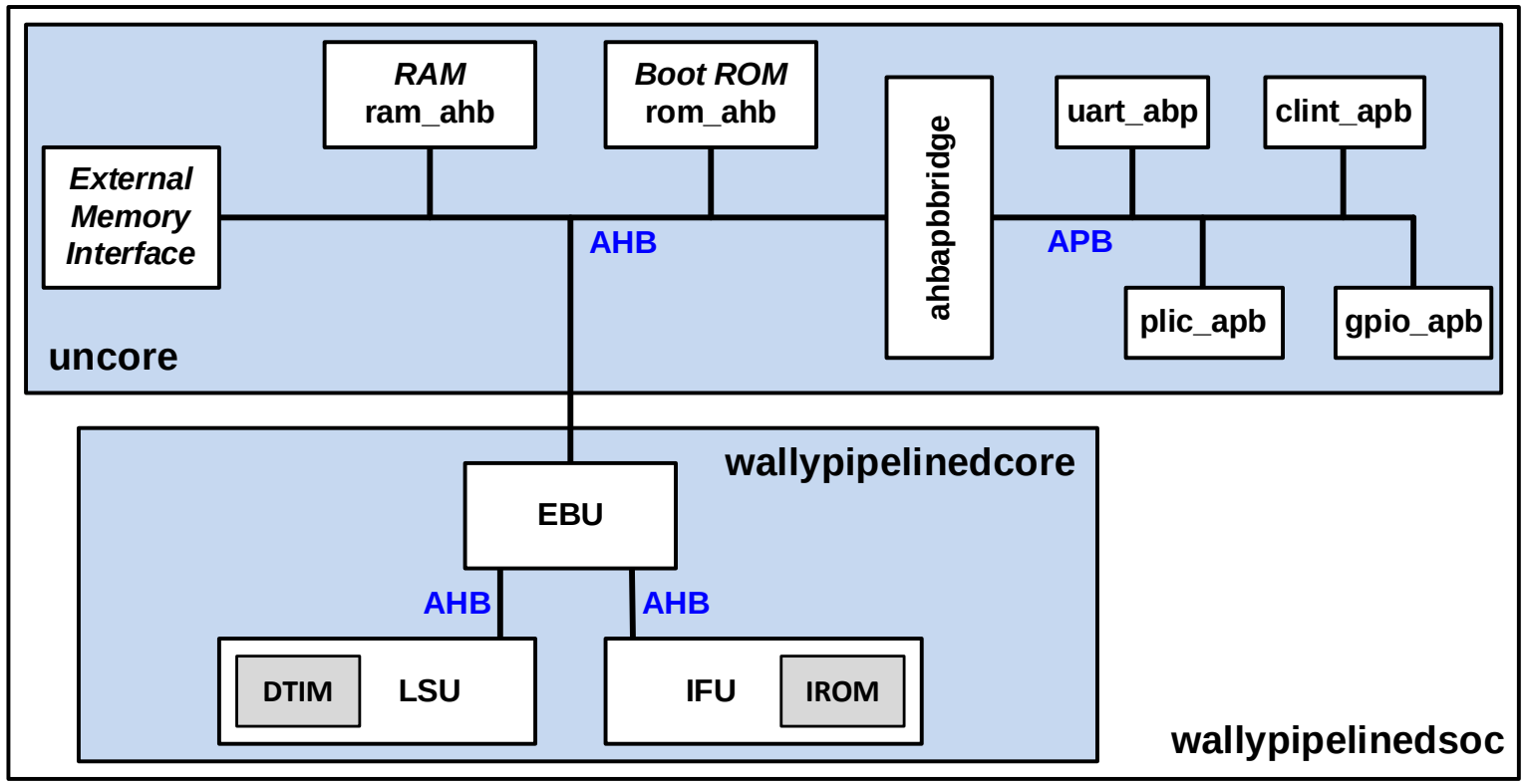
- Reads tag bits in Execute stage
- Fetches data and compares tags in Memory stage
- Configurable
- Line Size: 64 KiB

Wally SoC: EBU (External Bus Unit)



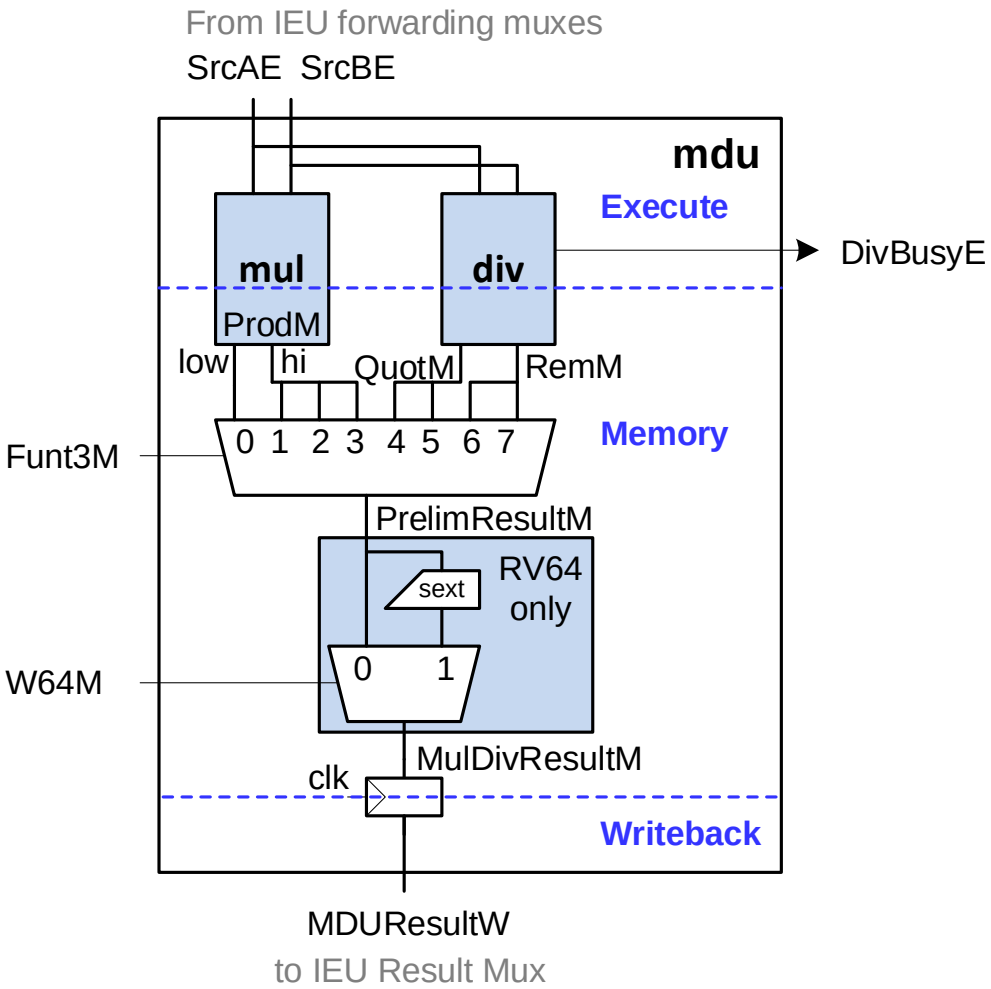
- AHB-Lite bus interface
  - All AHB signals start with “H”
- IFU or LSU read data
- EBU FSM Arbiter (ebufsmarb) arbitrates between IFU or LSU for reads
- Only LSU writes data
  - So, HWData = LSUHWData

Wally SoC: Uncore



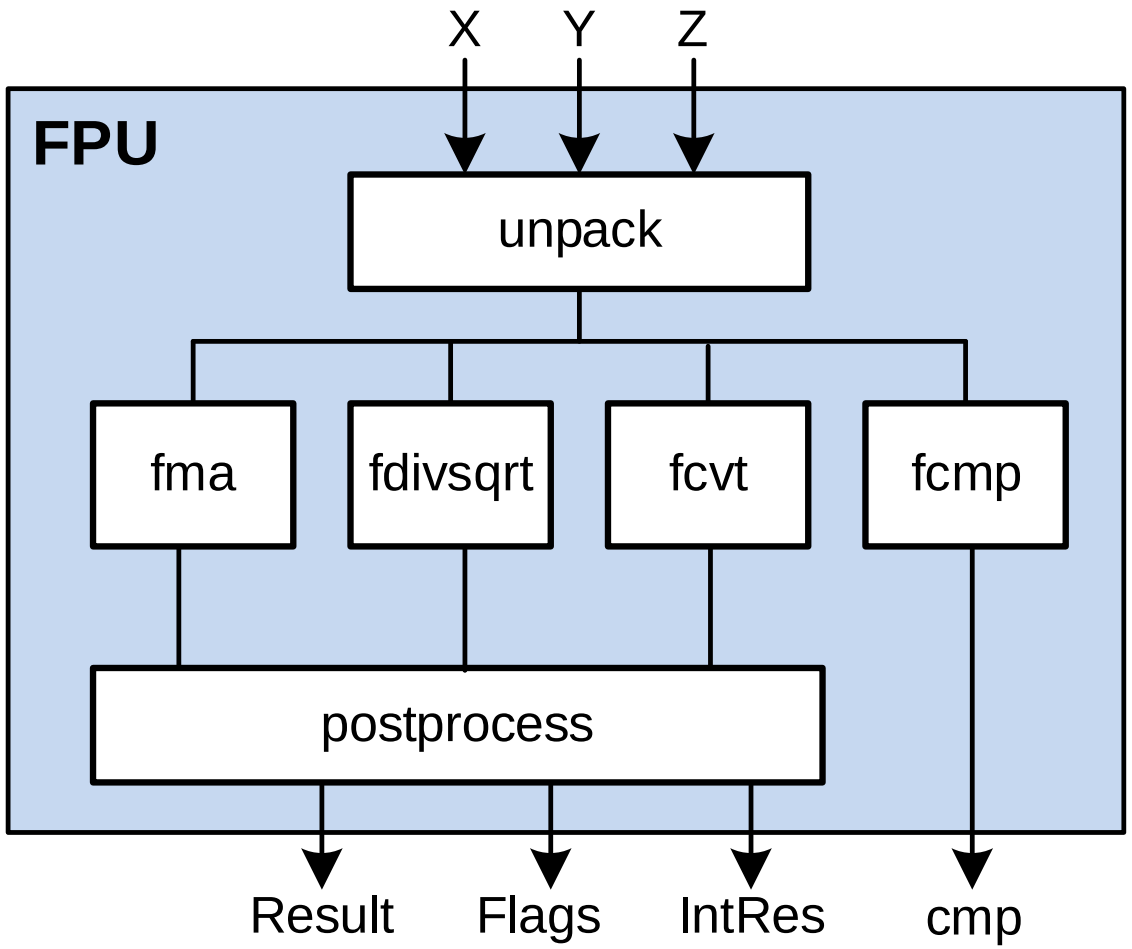
- **Uncore:** everything other than the core
  - AHB bus for interfacing with the core and memories
  - APB bus for interfacing with
    - Peripherals: UART, GPIO
    - PLIC: Platform-level interrupt controller
    - CLINT: Core-local interrupt controller
  - AHB-to-APB bridge

Wally SoC: MDU (Multiply/Divide Unit)



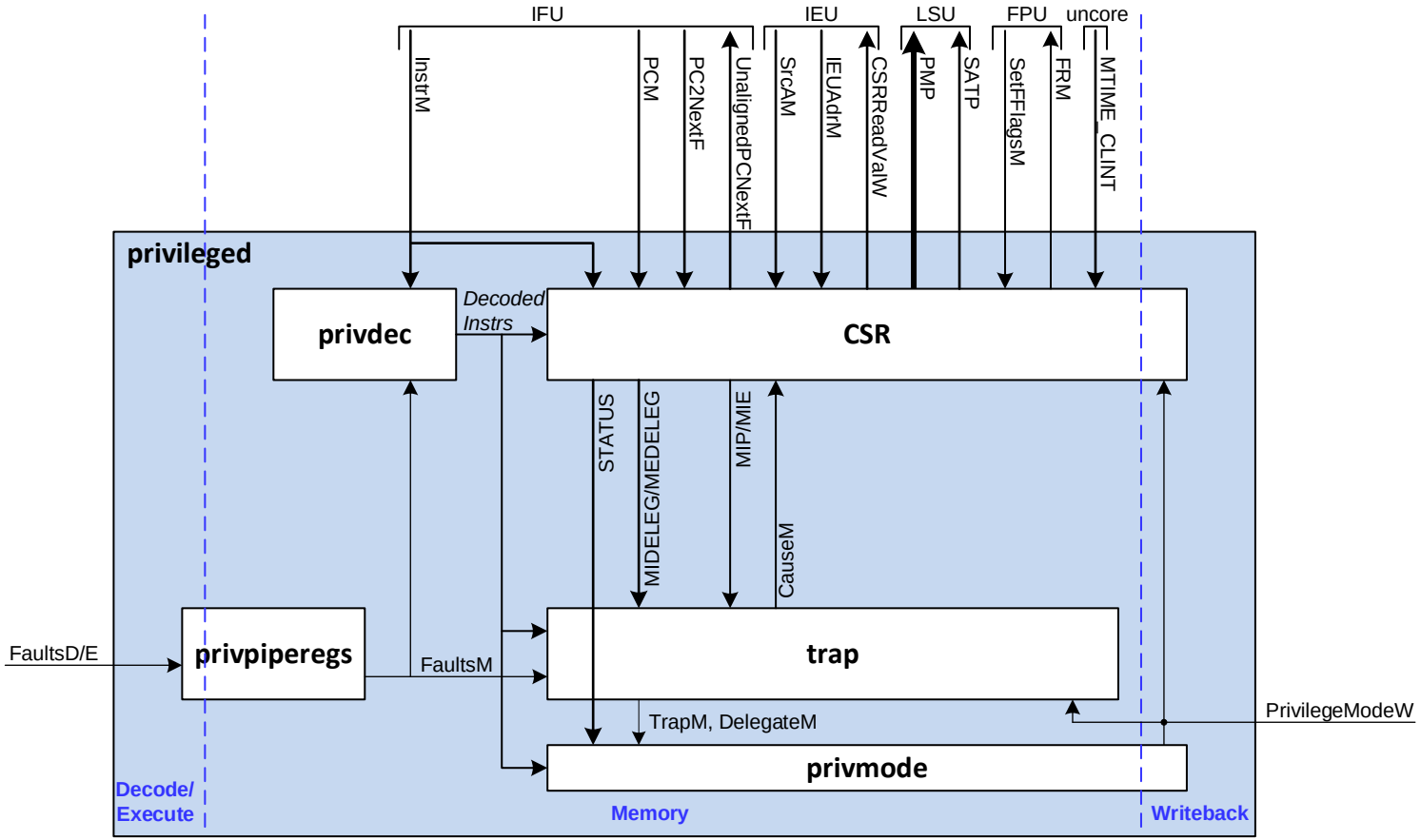
- Multiplication and division take multiple cycles in Execute stage:
  - **Multiplier:** pipelined: 2 cycles (Execute and Memory stages)
  - **Divider:** up to 9 cycles (8 cycles in Execute stage, 1 in Memory stage). Divider may terminate early upon completion
- May need to sign-extend a 32-bit result into 64-bit register (RV64 only)

Wally SoC: FPU (Floating-Point Unit)



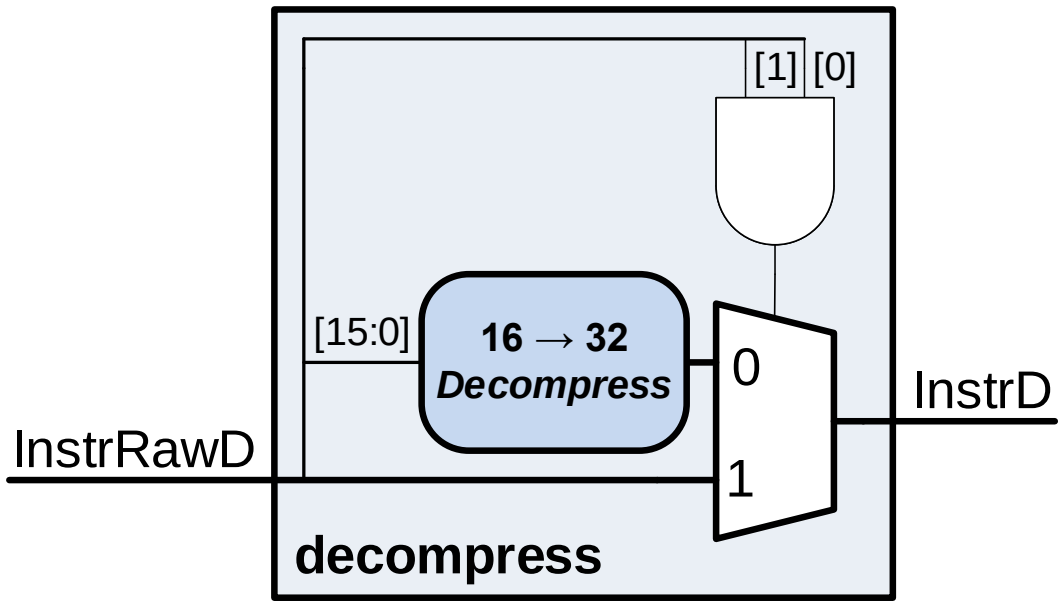
- **unpack:** unpack fields of operands and preprocess (prepend leading 1, shift to make exponents equal, etc.)
- **Computation units:**
  - fma: Floating-point multiply/add: Add and potentially multiply
  - fdivsqrt: Perform division or square root
  - fcvt: Convert to another format (single-to-double, etc.)
  - Fcmp: Compare floating-point numbers
- **postprocess:** produce results

Wally SoC: Privileged Unit



- **privdec:**
  - Decodes instructions, receives fault information
- **CSR (control & status registers):**
  - Holds CSRs – they are read & written
- **trap:**
  - Determines if take trap, produces: TrapM, CauseM, DelegateM
- **privmode:**
  - Stores and changes privilege mode (PrivilegeModeW)

Wally SoC: Decompress Unit



- Decompresses 16-bit (compressed) instructions into their 32-bit equivalents
- 32-bit instruction detected when lower-two bits of the instruction (i.e., opcode) are both 1s

Benchmarking: Wally Performance Coremark

Configuration	CoreMark CM/MHz	CPI
rv64i	0.99	1.14
rv64im	2.55	1.13
rv64imc	2.51	1.15
rv64imafdc	2.51	1.15
rv64imafdc_zba_zbb_zbc_zbs	2.79	1.18

Benchmarking: Wally Performance Embench

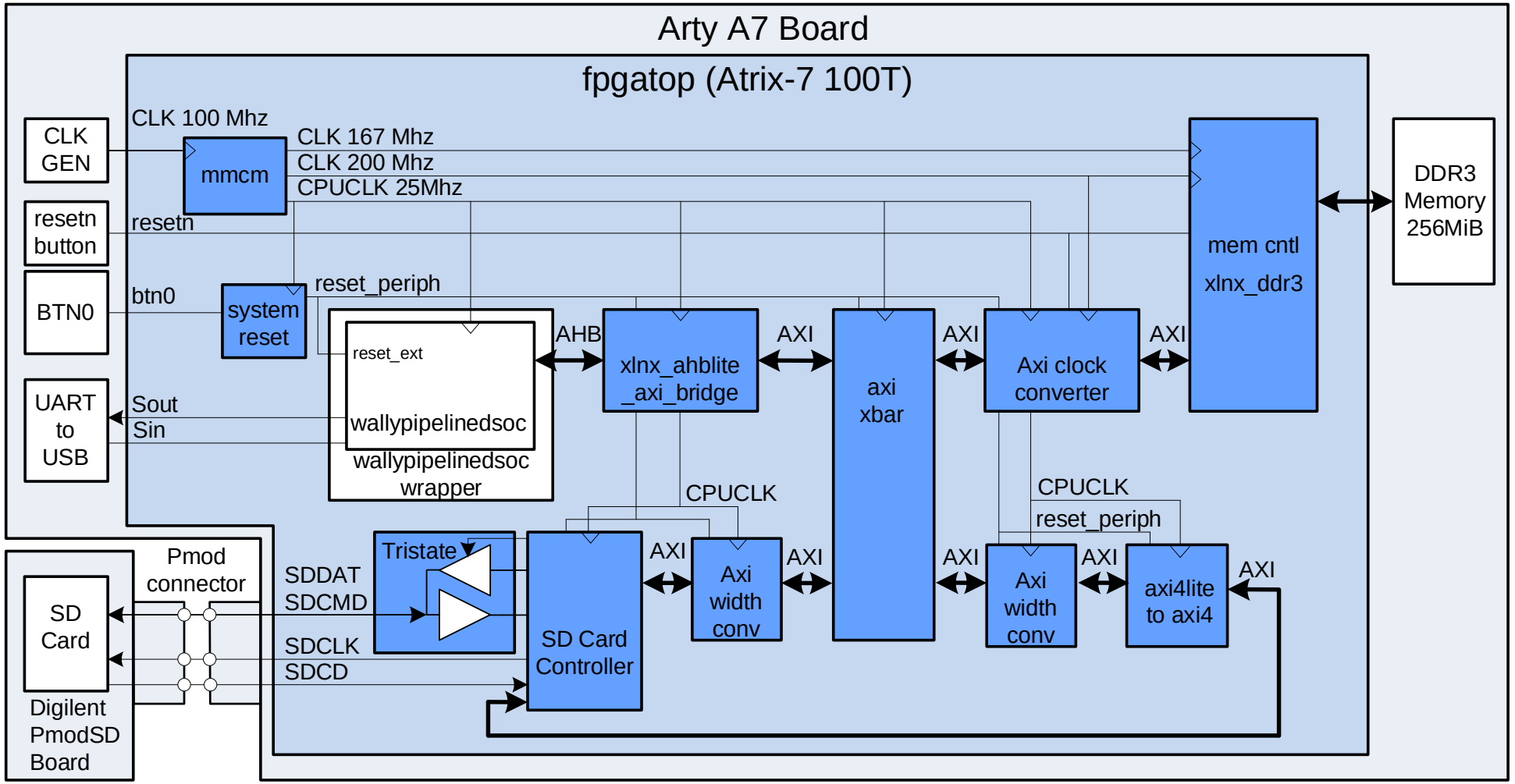
Configuration	Embench (speed opt)		Embench (size opt)	
	Performance (higher better)	Code Size (smaller better)	Performance (higher better)	Code Size (smaller better)
rv32i	0.51	1.75	0.44	1.52
rv32im	1.05	1.68	1.01	1.46
rv32imc	1.05	1.21	1.00	1.04
rv32imc_zba_zbb_zbc_zbs	1.10	1.17	1.04	1.01
rv32imafdc_zba_zbb_zbc_zbs	2.41	1.12	1.54	0.96
Cortex-M4 (reference)	1	1	1	1

Performance and Code Size scaled relative to Cortex-M4
Speed opt = -O2, size opt = -OS

Linux on Wally

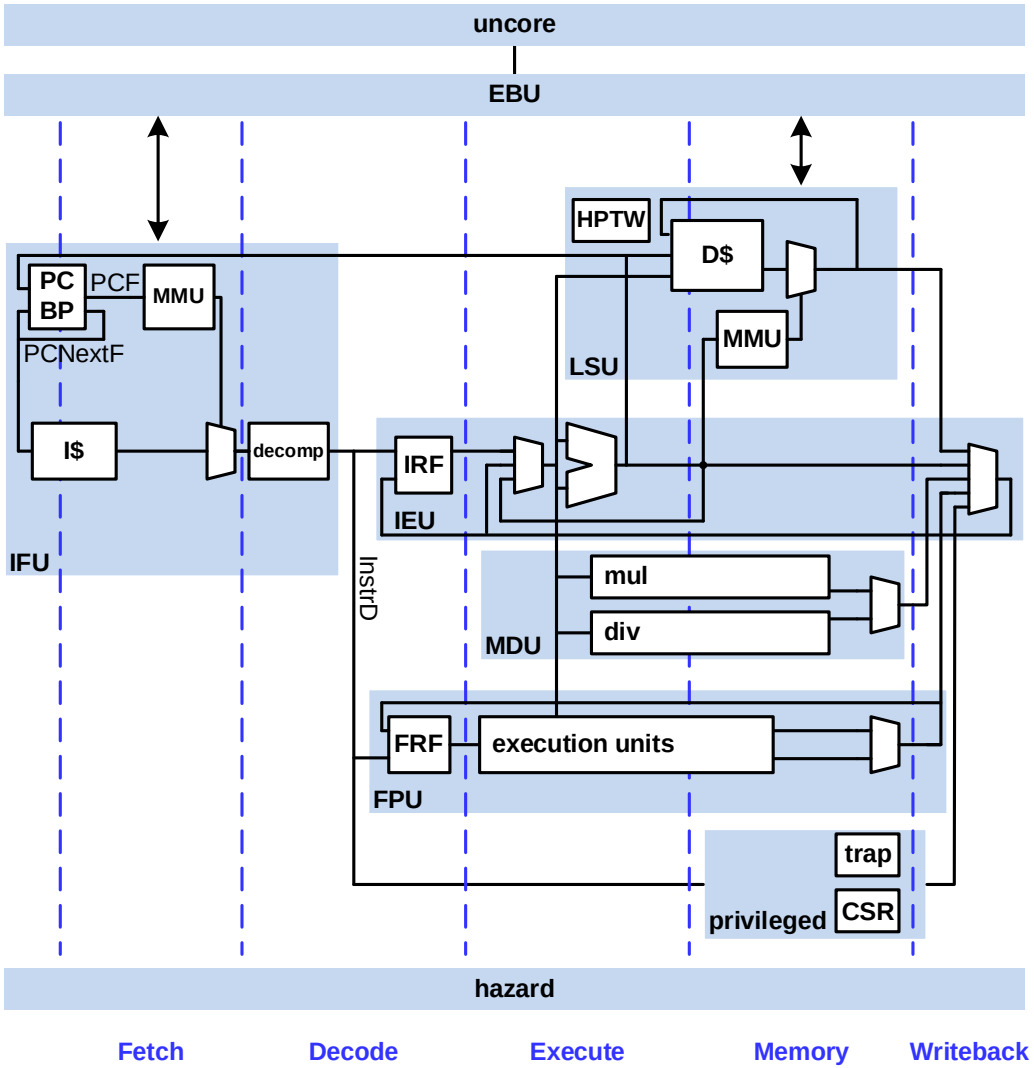
- Wally uses buildroot
- Linux OS does the following:
  - **Boots** the computer when it turns on
  - **Protects** hardware and other processes from faulty or malicious processes
  - **Provides resources** to processes:
    - Process and thread management and scheduling
    - Memory management
    - File system
    - Peripheral device drivers
    - Security
    - User interface

Wally on an FPGA (Arty-A7)



- Wally runs at **25 MHz**
- External memory:
  - On-board **DDR3** memory (256 MiB)
  - **SD card**
  - Both use AXI interface (via AHB-to-AXI bridge)

Wally SoC Recap



- 5-stage pipeline
- Configurable
- Supports:
  - RV32I & RV64I
  - Extensions: A, C, D, E, F, M, Q, Zb, Zicsr, Zfencei
- Boots Linux

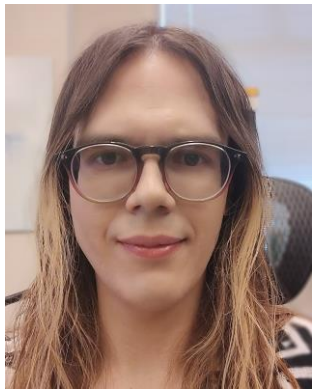
How to Use Wally

- **In courses:**
  - Textbook and SoC can be used as basis of upper-division / master's / PhD course
  - Supporting resources available:
    - Labs
    - Exercises at the end of each chapter and exercise solutions
    - Slides
    - All source code for Wally SoC
- **In research:**
  - Wally can be extended to support other features
  - Wally can be used as the basis for performing architectural tests / benchmarking for novel architecture and microarchitecture algorithms and hardware
- **In collaborations with industry:**
  - Wally could be implemented on a chip or as part of a larger system

Thank you

D. Harris

J. E. Stine



R. Thompson

S. L. Harris

HARVEY MUDD COLLEGE

OSU

UNLV