CORES TG – September 6 2021

Arjan Bink

Jérôme Quevremont

Davide Schiavone

Agenda

- CV32E20 status
- CV32E40P status
- CV32E40X / CV32E40S status
- CVA6 status
- Ara Vector Coprocessor (load/store handling)

CV32E20 Status

Lee Hoff, Joe Circello

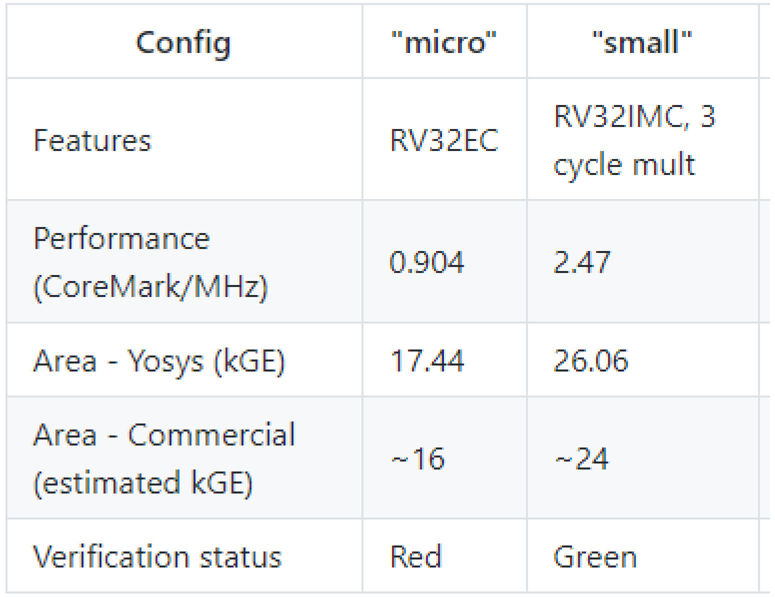
CV32E20: Status

- Project Gates
  - PC (former PPL) passed
  - Currently preparing for PL
    - Intended technical details are fairly well defined
    - Development of initial high-level development timeline
  - PA TBD
- Project meetings
  - Every other Monday 14:00 EDT (next on 9/13)
- TWG : Cores : CVE20 Mattermost channel
  - Channel created
  - Will upload all documents there

Status Continued

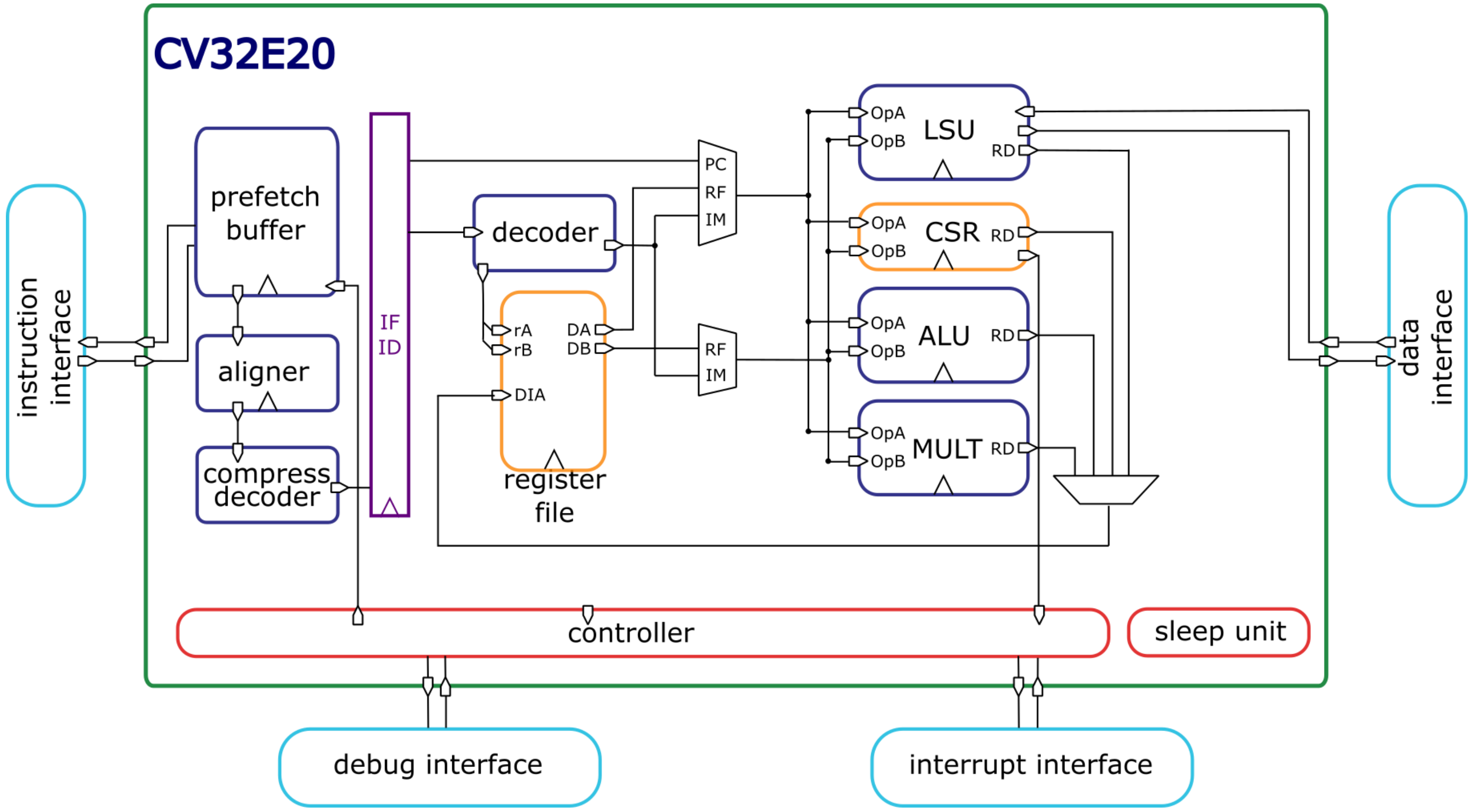
- Project Summary
  - Co-sponsored by NXP and Intrinsix
  - Ultra-low-end core based on ETH-Zurich ZeroRISCY
    - Use the IBEX core RTL design as the starting point
      - Intended as “minimum size” 32-bit RISC-V processor
      - Support for “micro” and “small” configurations
  - Optimized for minimum gate count and lowest power
  - Targeted at the low-end of RISC-V MCU roadmaps
    - Any “compute constrained socket”
    - Processor element in embedded SoC subsystems

https://github.com/lowRISC/ibex/blob/master/README.md



Config	"micro"	"small"
Features	RV32EC	RV32IMC, 3 cycle mult
Performance (CoreMark/MHz)	0.904	2.47
Area - Yosys (kGE)	17.44	26.06
Area - Commercial (estimated kGE)	~16	~24
Verification status	Red	Green

CV32E20 Core Block Diagram



- CV32E20 "CorePlex" (aka Core Complex)
  - Core + Debug + Harvard 32-bit AHB bus gaskets + OpenTitan CLIC

CV32E20: Current Activities

- Agreement on desired technical core details
- Ongoing discussions with multiple stakeholders
  - OpenHW Verification (Mike Thompson)
  - Imperas (Larry Lapides, Kat Hsu)
  - Embecosm (Jeremy Bennett)
- Working to identify RTL parameters to be supported vs. removed
- Early Design Exploration
  - Very preliminary synthesis experiments by NXP on the Ibex core
    - Process node is TSMC 16-nm FFC, 100 MHz Fmax target, "minimum" core configuration
      - `Ibex RV32E, no debug configuration = 15 Kgates`
      - `Ibex RV32I, no debug configuration = 19 Kgates`
      - `Point of reference: Arm Cortex-M0+ = 11 Kgates`

CV32E40Pv2 status

Pascal Gouédo

Yoann Pruvost

CV32E40Pv2

- Gates
  - PC (former PPL) passed on June 28th 2021
  - PL to prepare for next TWG Monthly on September 27 2021
- Project meetings
  - None right now
- TWG : Cores : CV32E4*P Mattermost channel

CV32E40Pv2: current activity

- Verification
  - On-going contract negotiation with Imperas about re-encoded PULP instructions support in Reference Model
  - Core-v-verif
    - Core test-bench up and running fine with Questa (1 pull request)
    - UVM test-bench up and running fine
      - Custom tests (Coremark,…), Google RISCV-DV generated tests, Embench (some updates in python scripts)

- SW
  - On-going contract negotiation with Embecosm about re-encoded PULP instructions support in GNU Toolchain

CV32E40Pv2: next

- Specification
  - Description format for re-encoded instructions?
  - Micro-architecture for new PULP features (HWLOOP,…)
- Design
  - v1 issue list to review
  - Working technical meeting with Davide Schiavone about correcting one v1 issue and make a pull request
- Verification Plan
  - Re-encoded PULP instructions
  - Simulation (Core & FPU)
  - Formal (Core & FPU)
  - FPU parts (FPnew & wrapped div_sqrt_top_mvp)

CV32E40X / CV32E40S status

Øystein Knauserud

CV32E40X

- Bugfixes
- RVFI updates
  - Debug, interrupt and exception handling
  - Doc
- LSU PMA support
- Misaligned load/store fully handled by LSU
  - PMA updated to detect misaligned access to I/O regions
- All stages now follow common valid/ready rules
- Moved CSR decoding to cs_registers (removed duplicate decoding)
- Removed any traces of PMP

CV32E40S

- Weekly merges from E40X repository
- User mode and related CSRs
  - Partly implemented
  - ci_check failing one test (debug, support not implemented yet)

Onespin status

- Amazon AWS cloud up and running
- Attempts to get E40P running with Onespin
  - Encountered issues with the tool and will require a new release
- Will target E40X once the tool is working with 40P

CVA6 status

Jérôme Quévremont

CVA6

- Gates
  - PC (former PPL) passed
  - PL passed
  - PA (former PPA) to come:
    - Specification presented at August TWG meeting; goal to approve it at October meeting
    - Detailed plan later (expected grants and new member)
- Project meetings
  - Every Friday 14:00 CEST
  - Alterning technical and progress meetings
  - https://calendar.google.com/calendar/u/0/embed?src=meetings@openhwgroup.org
- TWG : Cores : CVA6 Mattermost channel
  - Also spans on verification, SW topics…

CVA6: current activity

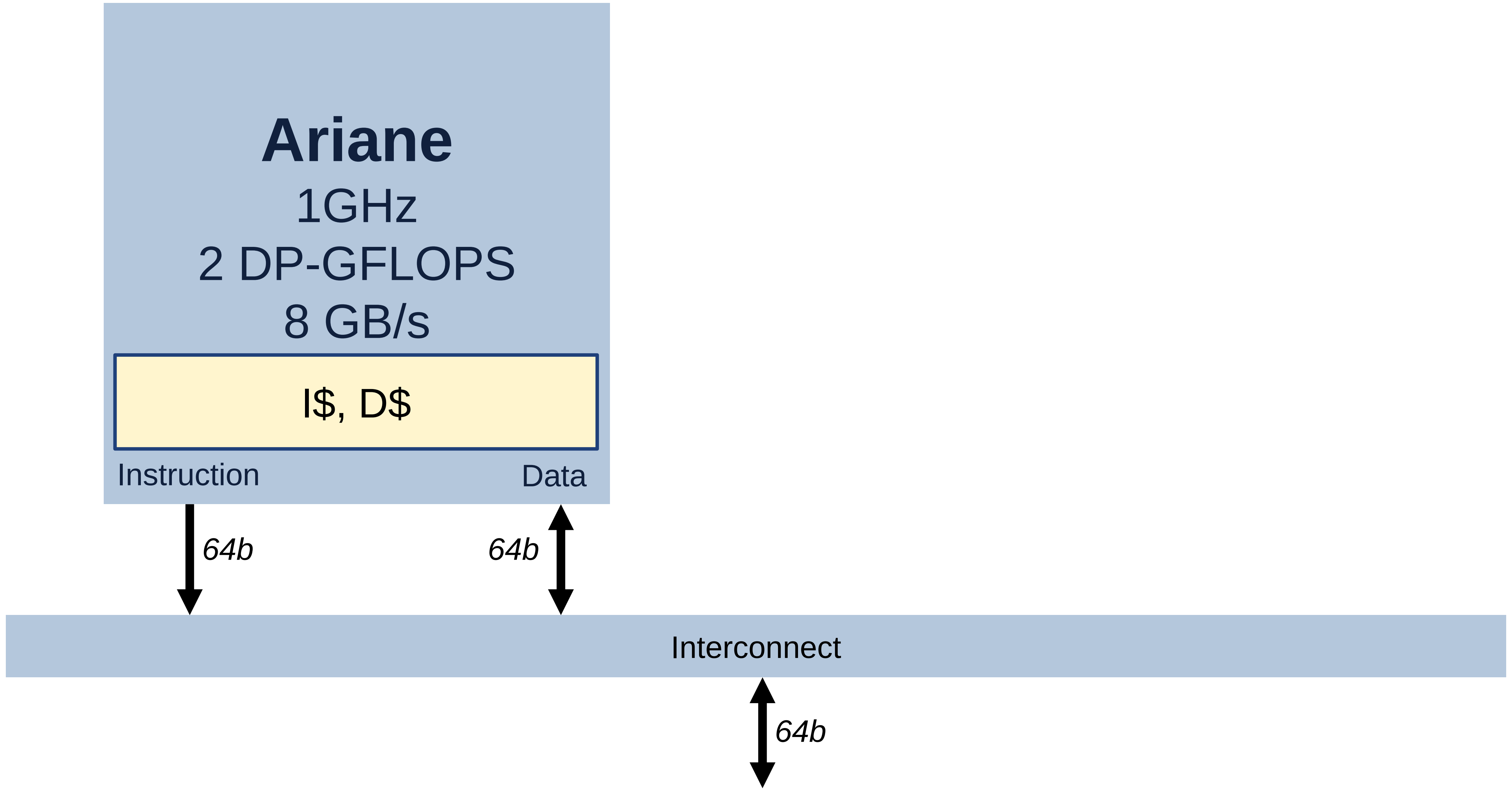
- Specification (link)
  - Google Docs → ReadTheDocs ongoing
  - Good progress on OVPSim questionnaire
- CV-X-IF: investigating a 2nd implementation
- Opportunity to share fpnew verification with CV2E40Pv2
- Design:
  - Sv32 MMU merged to master
  - Investigating divider speedup
  - Upcoming: implementing configurable reset (sync/async…)
  - ETH working on backend optimization
- Verification:
  - PR in the pipeline
  - CI tests running on cva6_reorg branch
  - Opportunity to share fpnew verification with CV2E40Pv2
- SW:
  - FreeRTOS: progressing
  - Linux 32b
    - Port available (BuildRoot)
    - Working on a common repo to generate a Linux image for CV32A6 and CV64A6
    - Preparing PR
  - LLVM activity

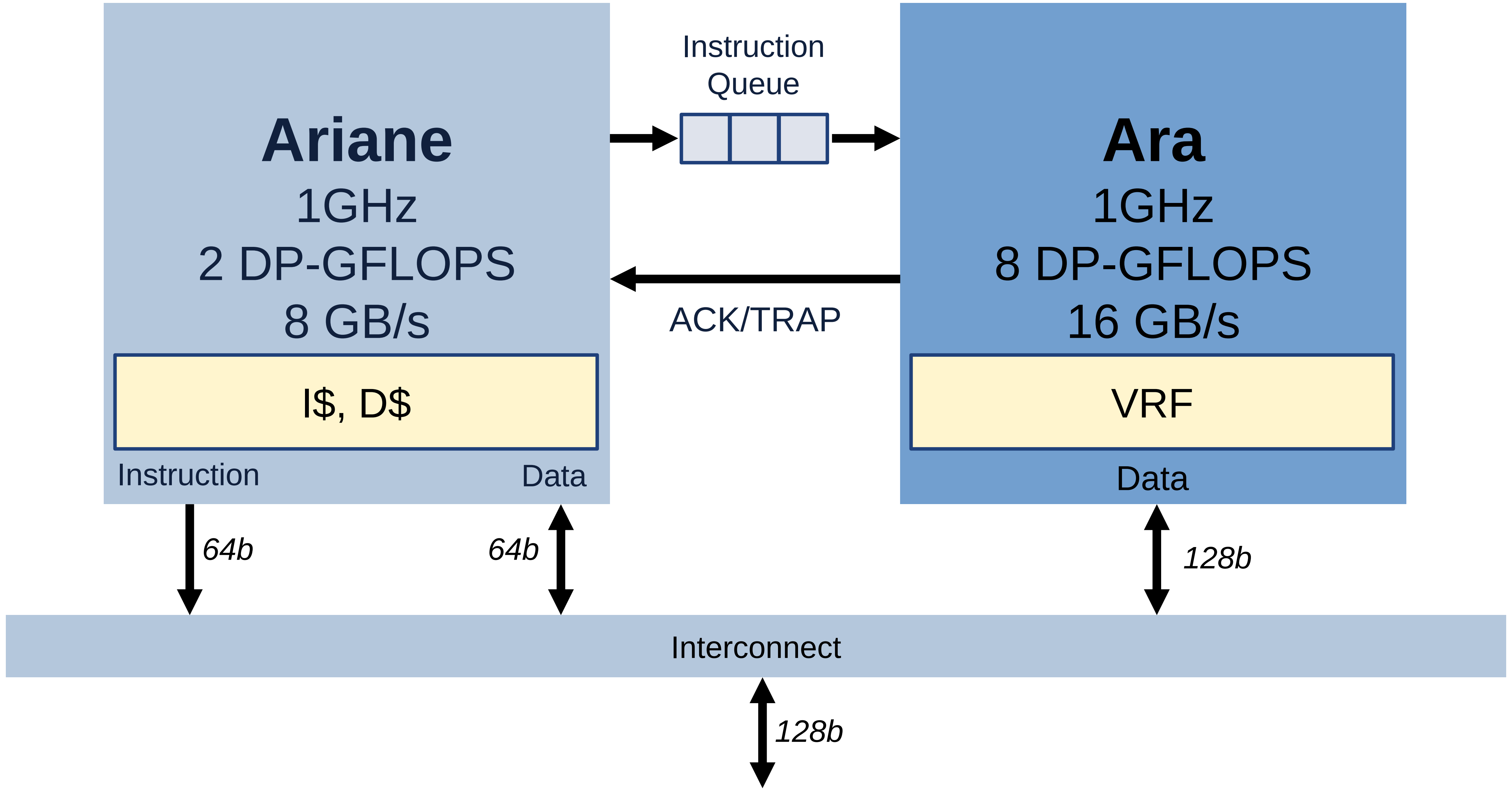
ETH zürich

D-ITET
Integrated Systems Laboratory

Ara

Matheus Cavalcante
Ph.D. Student
Integrated Systems Laboratory

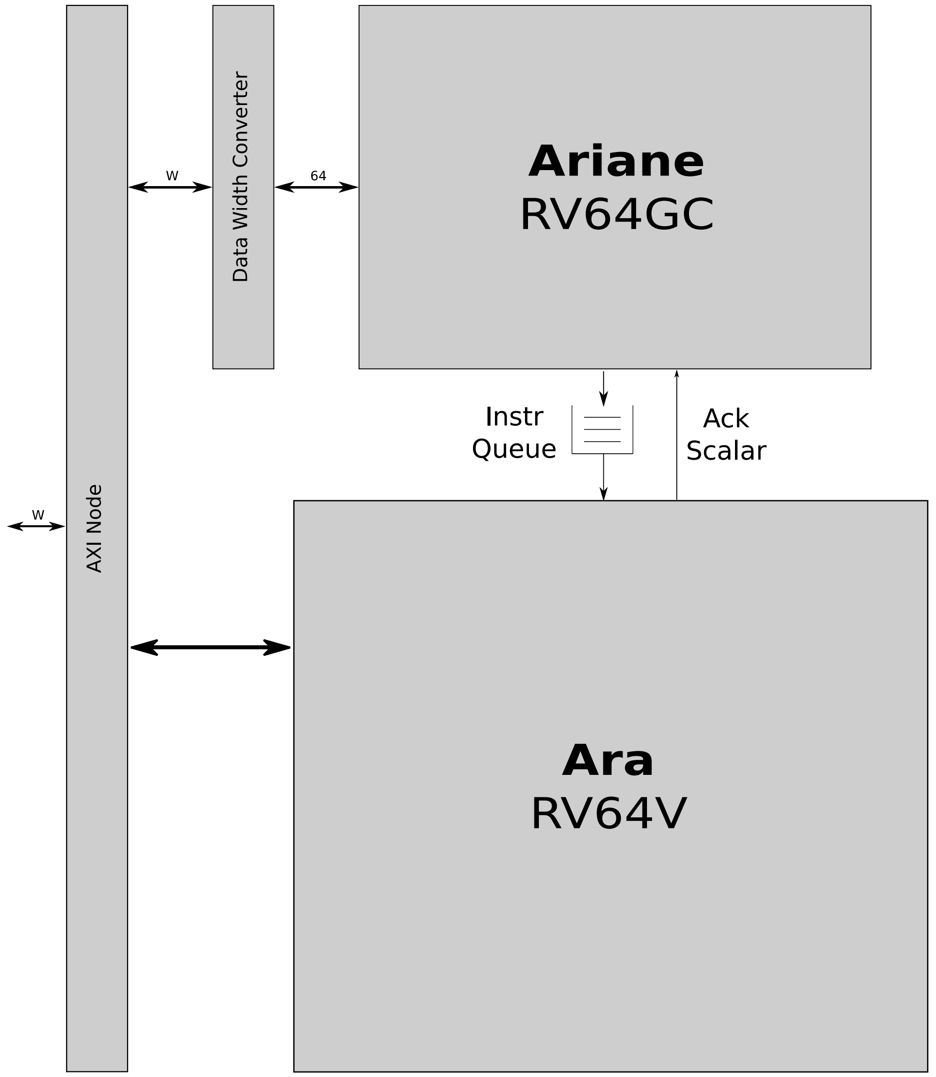

Ariane
1GHz
2 DP-GFLOPS
8 GB/s
I$, D$
Instruction
Data
64b
64b
Interconnect
64b


Ariane
1GHz
2 DP-GFLOPS
8 GB/s
I$, D$
Instruction
Data
Instruction
Queue
ACK/TRAP
Ara
1GHz
8 DP-GFLOPS
16 GB/s
VRF
Data
64b
64b
128b
Interconnect
128b

ETH zürich

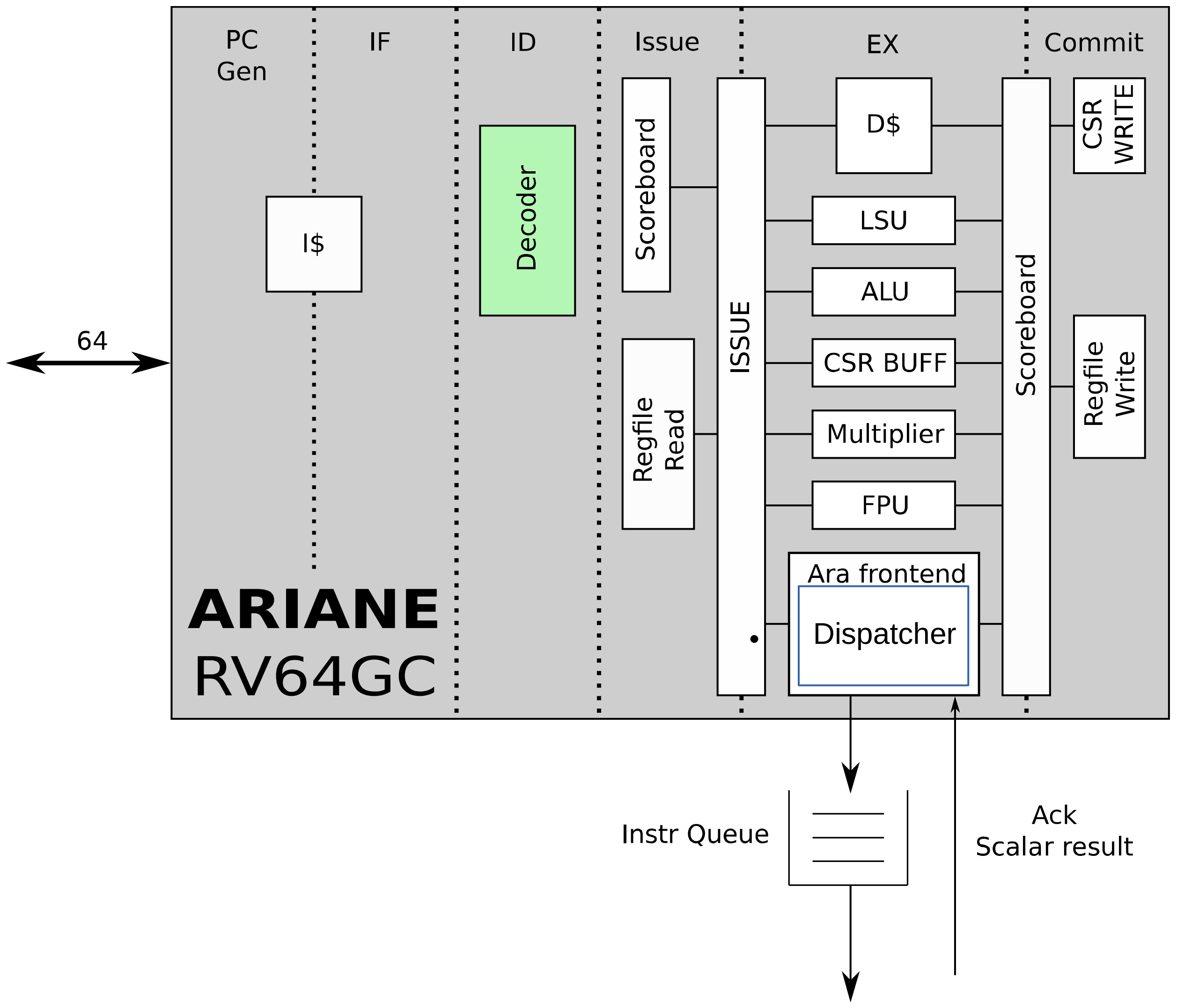
Ara - High level architecture

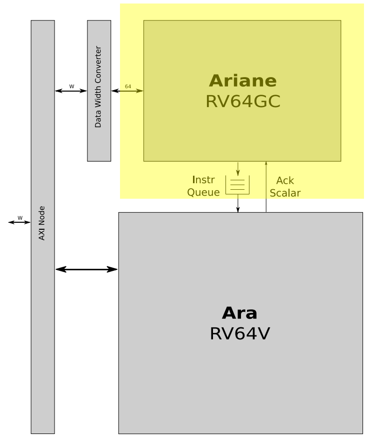
- **Coupled with Ariane**
- Ariane dispatches RVV instructions to Ara
- **Ara acknowledges Ariane**
  - Any errors?
  - Scalar result?
- Private memory access through AXI



Ariane – RVV instruction dispatch

- Fetch + Decode in Ariane
- Scoreboard + **Issue to Ara frontend**
- Dispatch to Ara at **commit-time only**
- **Wait for Ara‘s answer to commit**





Ara's memory interface requirements

- Ara is developed for a memory-throughput-to-performance ratio of 2 B/FLOP
  - Even a "small" Ara instance with 4 lanes (8 FLOP/cycle) asks for a 128-bit memory interface

- Throughput is essential, latency is (to a certain extent) tolerable
  - The smaller the throughput, the larger the problem has to be to achieve good performance on Ara

- Ara uses wide and long AXI bursts for most of the data transfers
  - The longest AXI bursts, of 4 KiB, are a common occurrence in vector code for "large" problems

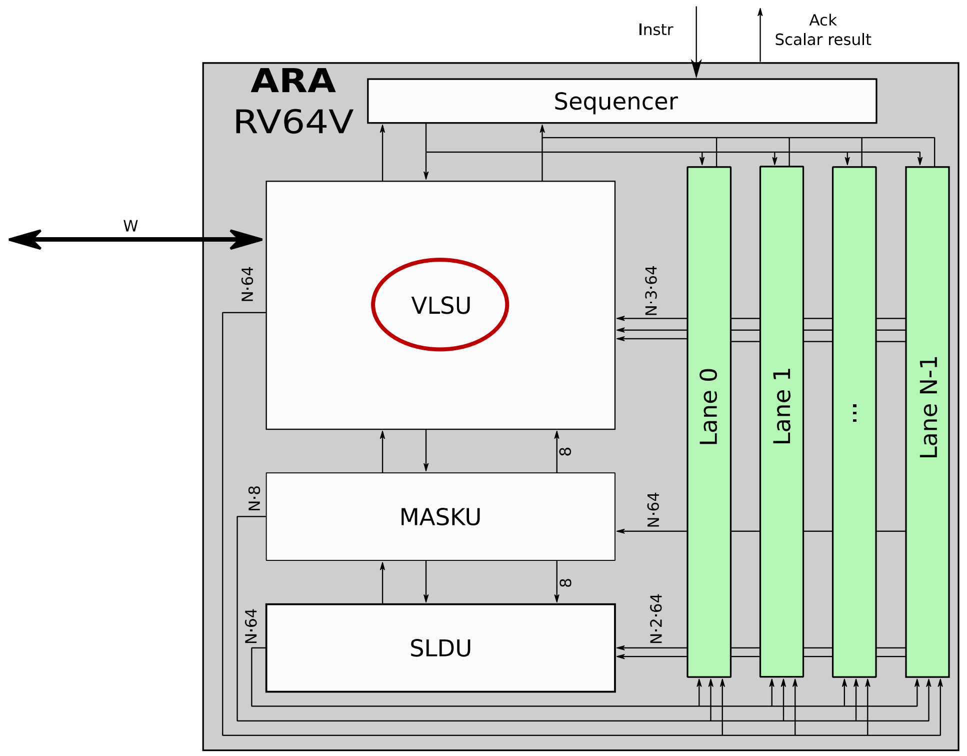
Vector Memory Operations

- Unit-strided memory vector operations
  - Access `@baseaddr`, `@(baseaddr + vlew)`, up to `@(baseaddr + (vl – 1) * vlew)`
  - Translated into an AXI INCR burst
  - Known access pattern

- Constant-strided memory vector operations
  - Access `@baseaddr`, `@(baseaddr + stride)`, up to `@(baseaddr + (vl – 1) * stride)`
  - Translated into a series of AXI accesses
  - Known access pattern

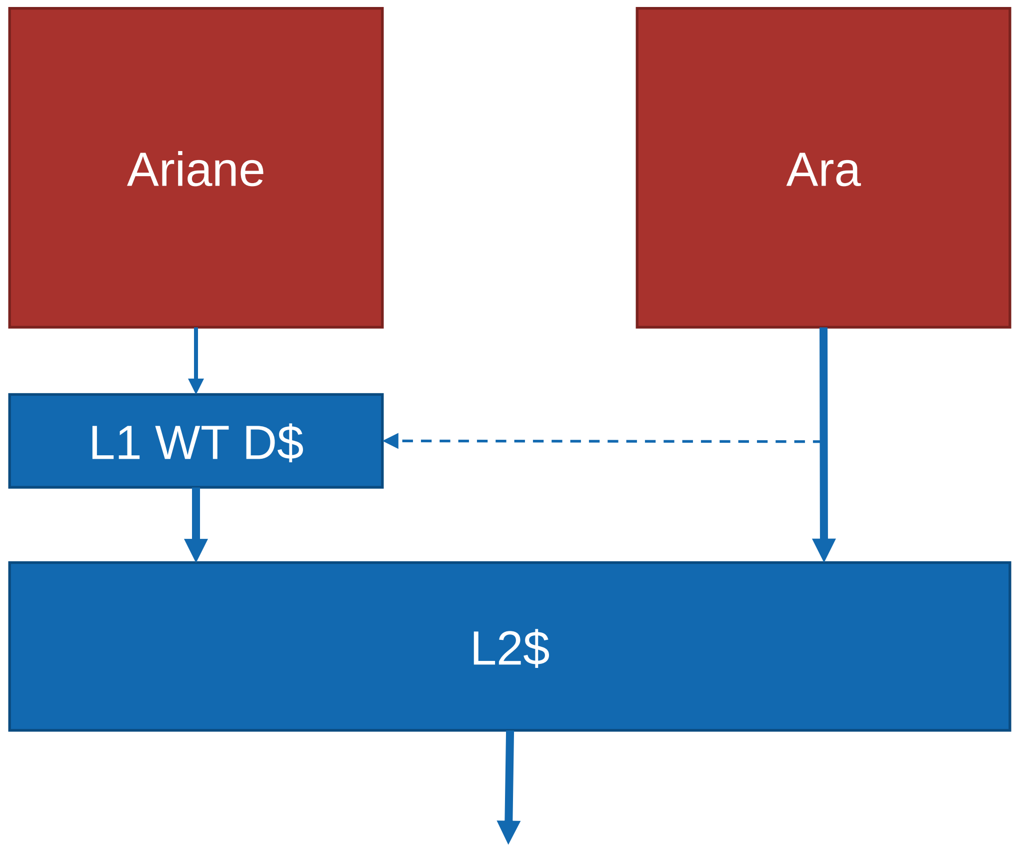
- Scatter/gather memory vector operations
  - Access `@(vaddr[0])`, `@(vaddr[1])`, up to `@(vaddr[vl - 1])`
  - Translated into a series of AXI accesses
  - Requires access to the vector register file: unknown access pattern

Vector memory instructions on Ara



- Memory instructions are handled by the VLSU
- Upon receiving a vector memory instruction, the *address generator* translates the instruction into a series of AR/AW beats
  - Bursts, when possible
  - All requests have the same AxID: no response reordering done inside Ara
- The *load unit* and the *store unit* are responsible for receiving and producing the R, W, and B beats, and communicating with the VRF inside the lanes

Coherency



- The following mechanisms ensure coherency between Ara and Ariane:
  - Ariane's L1 cache is write-through
  - Ara's vector stores cause invalidation requests in Ariane's L1 D$
  - No vector memory operation in allowed to go on if there is a scalar store ongoing
  - No scalar memory operation in allowed to go on if there is a vector store ongoing

Traps

- Memory operations can throw an exception in the middle of their execution
  - Specification allows for precise or imprecise traps, depends on the environment
  - In idempotent memory regions, vector stores are allowed to have updated elements past the element causing a trap

- Fault-only-first vector memory operations
  - Those instructions only throw an exception if it happened at the *first* element
  - Otherwise, the **`vl`** is trimmed to the one minus the first faulty element
  - Useful for the parallelization of while loops

```
vle64.v v0, 0(a0) // vl = 128, fault @elem. #20
EXCEPTION!
vadd.vi v0, v0, 1 // Not executed

vle64ff.v v0, 0(a0) // vl = 128, fault @elem. #20
(No Exception)
vadd.vi v0, v0, 1   // vl = 19
```

Exception Handling in Ara

- Ara currently does not do exception handling
  - Mechanism that stalls scalar memory operations when Ara does a vector store can be extended
    - Prevent *any* instruction to execute on Ariane when Ara is doing a memory operation

  - No support for virtual memory
  - No support for access protection
    - Those could be offloaded to Ariane through the X Interface

  - Can the exception be determined before we receive the R/B responses?
    - Otherwise we have to stall Ariane for a long period

Thank you!