# 32-bit RISC-V CPU Design

Review and discussion session – August 27th 2023

Updates:

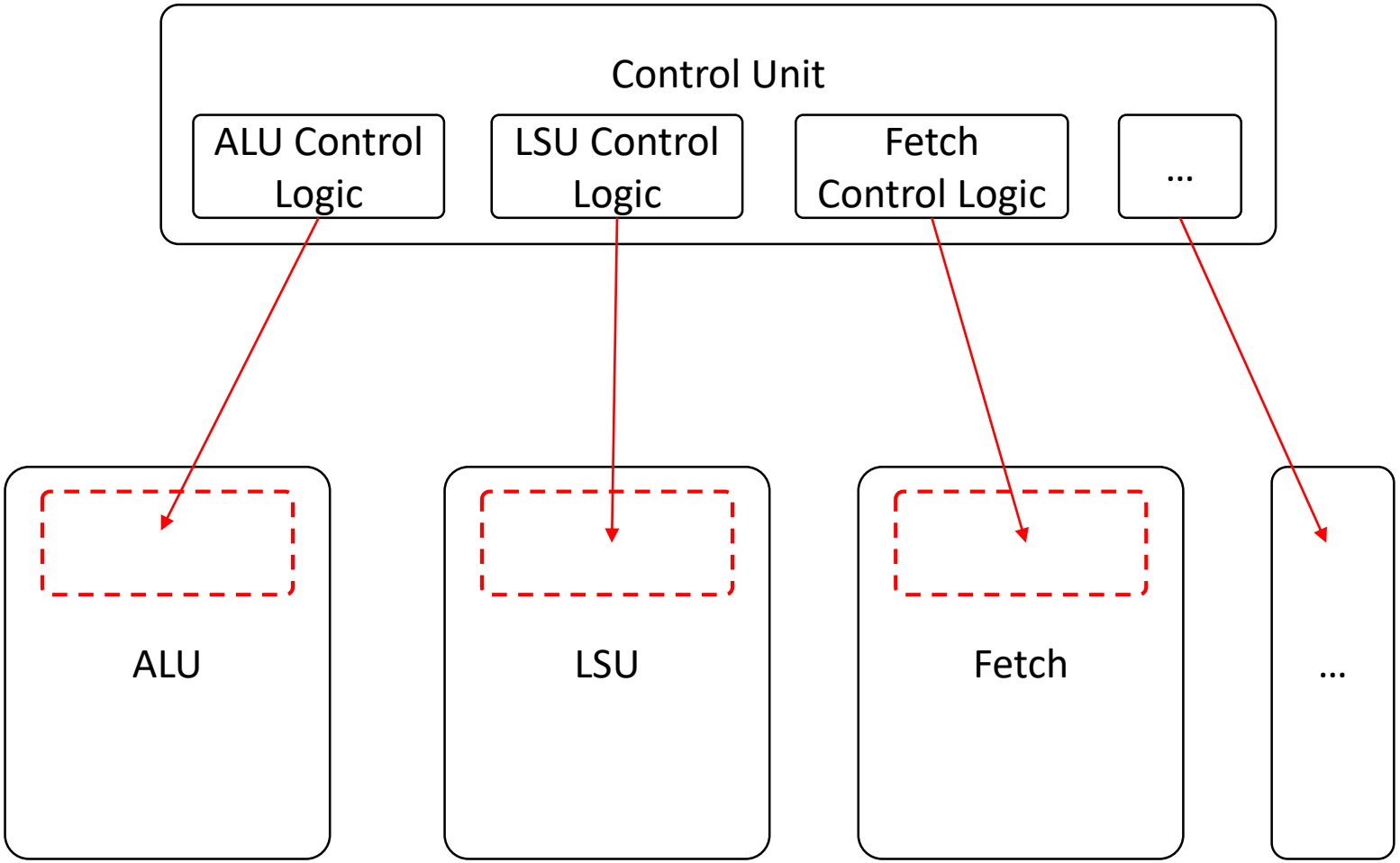
- RV32I (Integer) extension RISC-V ISA support
- 5 stage pipelined processor
- Hazard detection and data forwarding
- Modular and extensive design
- Self-control logic on modules

# Technical Specifications:

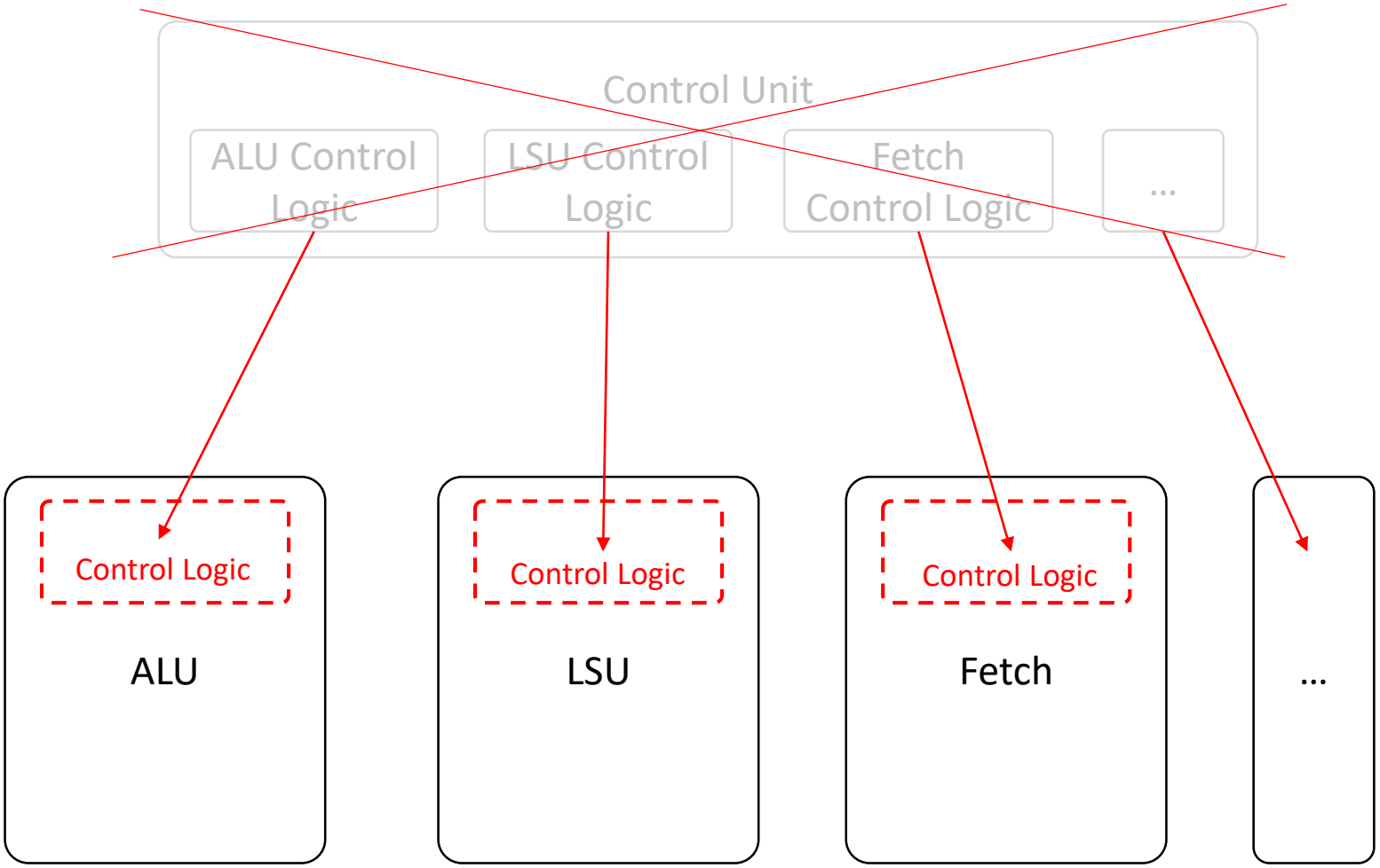
| Module | Max Delay (ps) |
|---|---|
| Address Generator | 3844.84 |
| Arithmetic Logic Unit | 3099.01 |
| Control Status Registers | 747.689 |
| Hazard Forward Unit | 1131.73 |
| Immediate Generator | 1016.44 |
| Instruction Decoder | 716.437 |
| Jump Branch Unit | 243.115 |
| Register File | - |
| Memory Interface (8 cells - Logic) | 265.648 |
| Normalized Memory Access Time | 10000 - 40000 |
| Fetch Unit (8 cells - Logic) | 308.907 |
| Load Store Unit (8 cells - Logic) | 569.903 |
| Total | 11678.071 |

| Core specifications | |
|---|---|
| Clock Cycle Time | 4 ns |
| Memory Operation Time | 3 clock cycles |
| CPI (R,I-TYPE) | 1.13 |
| Frequency | 250MHz |
| Memory address space | 4KB |

Self Control Logic:



# Self Control Logic:



Testing processor and benchmarking:

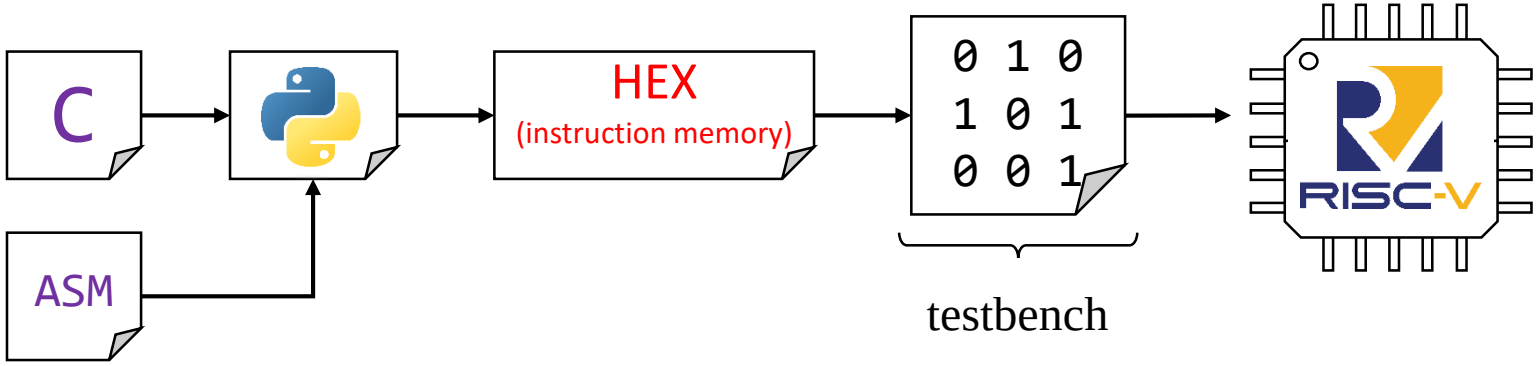
There are 2 applications developed in Python for automation of core programming process. These code executant applications can be executed in both Windows and Linux environments.

- Windows:

Venus Simulator (assembly code) “Visual Studio Code” extension
Test flow:
Assembly output (.txt) → Python Script → instruction memory HEX file → Testbench

- Linux:

RISC-V GCC compiler toolchain (C code)
Test flow:
C code → Python script → Generated shell script to run C code by GCC toolchain → instruction memory HEX file → Testbench



To do list:

- Run GCC compiler output HEX file on RV32I core
- Add Multiplier Unit to core (Approximate Multiplier)
- Run GCC compiler output HEX file on RV32IM core

- Run time analysis tests on final core
  - CPI calculation
  - Critical path and clock width calculation
  - Frequency calculation

- Final report on core specifications
- Documentation and publishing repository

Issues:

- Memory management system (simulation limitations)

- Standard synthetize and static time analyze (STA) tool
- Siliconcompiler out of reach (remote run not supported anymore)
  - Current tools: Yosys, Qflow
  - TSMC 180nm technology (osu018)
  - FreePDK45: STA wrong output + DRC error
  - Standard tools: Cadence, Synopsys