Student

Betsaleel W. D. Henry

Advisor

Prof. Bo-Cheng Charles Lai

RISC-V

processor design meeting

National Yangming Chiaotung University

Electrical & Electronics Engineering department

2023

Overview

- Brief talk of specifics related to the RISC-V project so far.
- Objective and scope of the project so far.
- Highlights of the issues encountered.
- Q&A and suggestions from the professor.

Introduction

- RISC-V stands for Reduced Instruction Set Computer - V.
- It is an open-source ISA that follows the RISC (Reduced Instruction Set Computer) design philosophy.
- RISC-V provides a foundation for designing and implementing processors for a wide range of applications, from embedded systems to high-performance computing.

Advantages of RISC-V

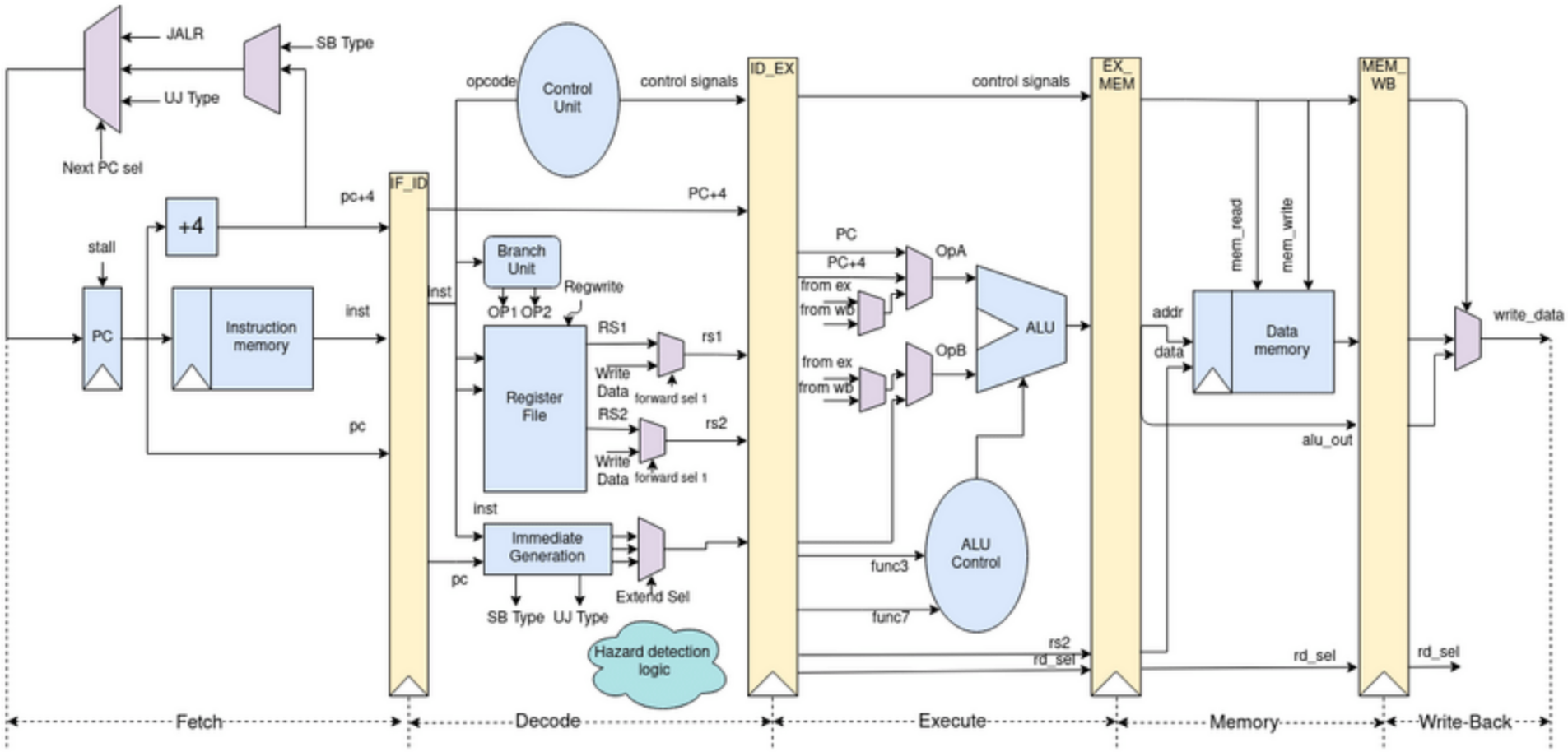
- Open source
- Modularity
- Simplicity
- Portability
- Industry adoption etc.

Methodology/Approach for project

- RISC-V has 32 integer registers and can have 32 floating point registers.
- The memory is addressed by 8-bit bytes.
- The instructions are assumed to be 32-bit addresses.
- RISC-V, a "load-store" machine, has different types.
- Thus this project implements instructions from ISA RISC 32I -> (RV32I)
- I'm using Verilog to implement ALU, IMEM, and Decoder ...

https://five-embeddev.com/riscv-isa-manual/latest/rvc-instr-table.html

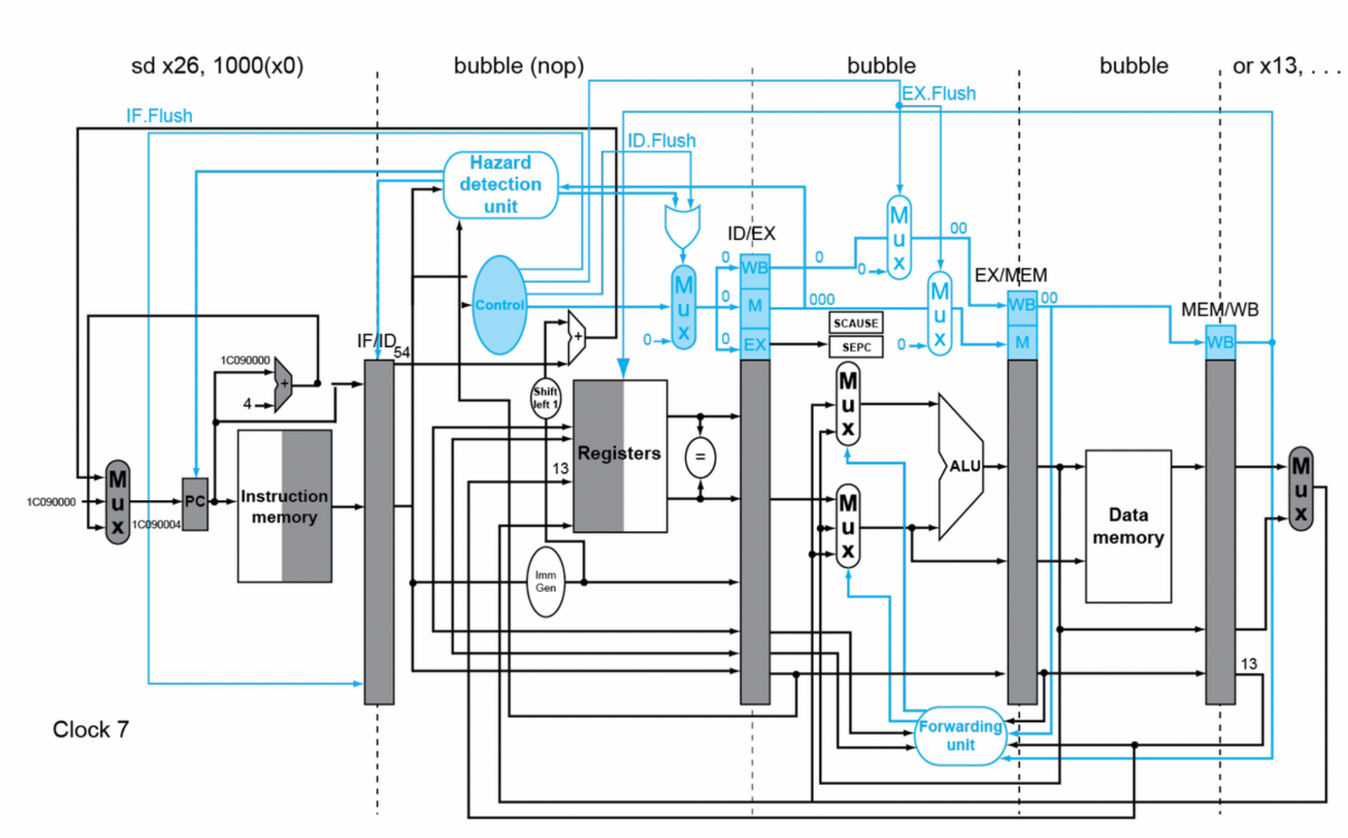
Overall system layout



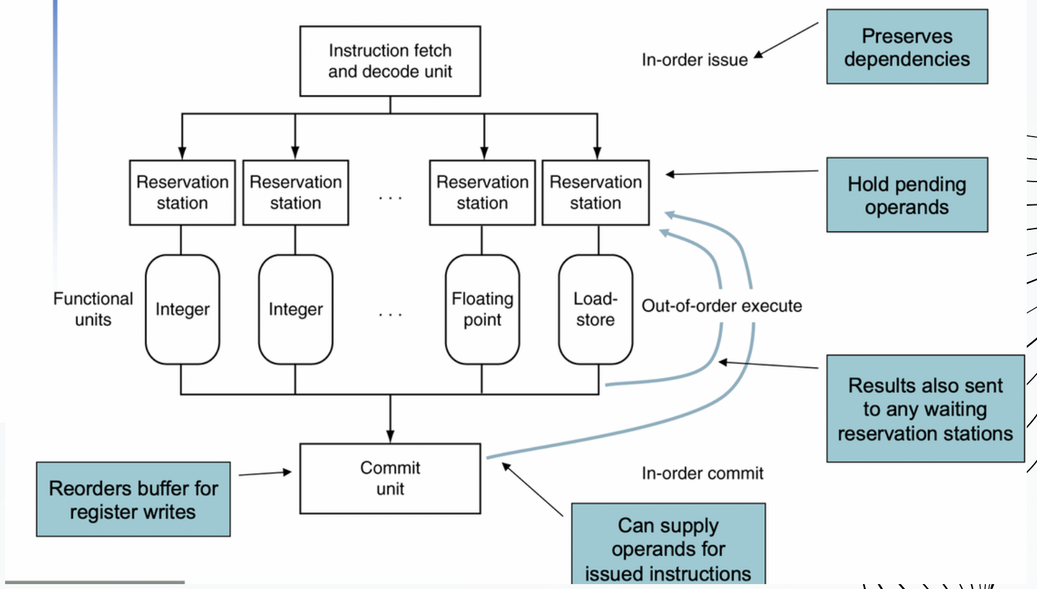
ps. hazard detection logic to be implemented

Observations

Mux control	Source	Explanation
ForwardA = 00	ID/EX	The first ALU operand comes from the register file.
ForwardA = 10	EX/MEM	The first ALU operand is forwarded from the prior ALU result.
ForwardA = 01	MEM/WB	The first ALU operand is forwarded from data memory or an earlier ALU result.
ForwardB = 00	ID/EX	The second ALU operand comes from the register file.
ForwardB = 10	EX/MEM	The second ALU operand is forwarded from the prior ALU result.
ForwardB = 01	MEM/WB	The second ALU operand is forwarded from data memory or an earlier ALU result.



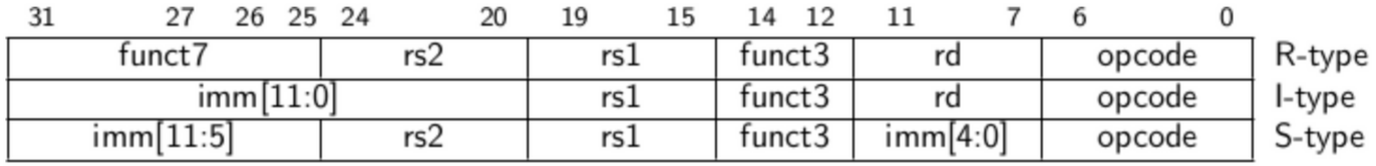
Dynamically Scheduled CPU



observations

- **beq**
- **sd**
- **ld**
- **blt**
- **addi**
- **add**
- **sub**
- **xor**
- **or**
- **and**
- **jal**
- **jalr**

- Pipelining stages in the RISC-V processor, sequential and combinational blocks separated, how they contribute to improved performance and efficiency.
- Not very clear yet, the organization and structure of the register file, RISC-V processor does it need to have floating point arithmetic modules.
- No forwarding unit yet, handling the hazards part that influences overall design, will need more research on that. detecting data hazards and implementing code.
- The project scope goes over the most popular instructions, fetching, decoding, executing, and writing back to register. listed on the page.
- Left to do: the forwarding unit, assembler to machine code translator better testbench for corner cases and other instruction handling support in the decoder.



Q&A

Thanks for suggestions