# Computer Architecture

Hw4: Single Cycle CPU

Due: 2020/1/3

## 1. Goal

Utilizing the ALU in hw3 to implement a 32-bit single cycle CPU supporting RV64I. The CPU reads 32-bit instructions and operates on 64-bit registers and data memory.

## 2. Requirements

a. Please use Modelsim or Xinlinx as your HDL simulator.

b. Extend your ALU in hw3 to **64-bit** to support 64-bit registers in RV64I.

c. Cout and Overflow ports of ALU wouldn't be used in this assignment.

d. ProgramerCounter, InstructionMemory, DataMemory, RegisterFile, Adder and testbench are supplied.

e. Instruction set: the following instructions need to be supported by your CPU

| | |
|---|---|
| R-type: add, sub, and, or, slt | (50%) |
| I-type: addi, slti, ld | (30%) |
| S-type: sd | (10%) |
| B-type: beq | (10%) |
| J-type: jal | (20%) (bonus) |

f. All x1 - x31 registers are assumed to be general-purpose registers. (The register x0 is hardwired to the constant 0)

g. Your CPU should read **machine code** rather than assembly code.

h. We provide two testcases for testing. If you want to change the testcase, please modify the 38-th line of InstructionMemory.v

```
$readmemb("test/test1.txt", Instr_Mem);  //Read instruction from "test1.txt"
```

- beq rs1, rs2, offset
  if rs1 == rs2, then PC = PC + (signed_extend(offset) << 1)
- jal rd, offset
  PC = PC + (signed_extend(offset) << 1), rd = PC + 4

| 31 27 26 25 | 24 20 | 19 15 | 14 12 | 11 7 | 6 0 | |
|---|---|---|---|---|---|---|
| funct7 | rs2 | rs1 | funct3 | rd | opcode | R-type |
| imm[11:0] | | rs1 | funct3 | rd | opcode | I-type |
| imm[11:5] | rs2 | rs1 | funct3 | imm[4:0] | opcode | S-type |
| imm[12\|10:5] | rs2 | rs1 | funct3 | imm[4:1\|11] | opcode | B-type |
| imm[31:12] | | | | rd | opcode | U-type |
| imm[20\|10:1\|11\|19:12] | | | | rd | opcode | J-type |

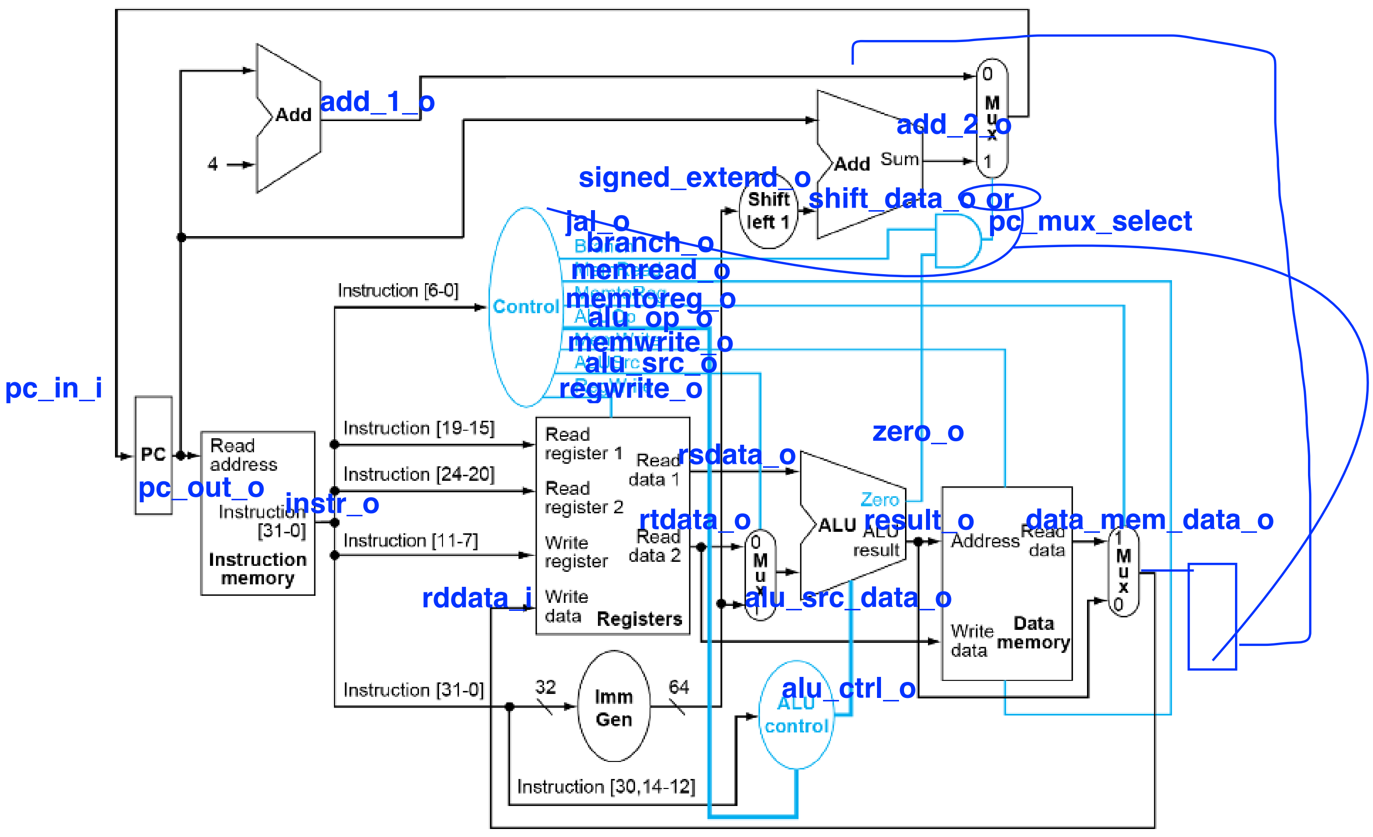
**RV64I Base Instruction Set (in addition to RV32I)**

| | | | | | | |
|---|---|---|---|---|---|---|
| imm[11:0] | | rs1 | 110 | rd | 0000011 | LWU |
| imm[11:0] | | rs1 | 011 | rd | 0000011 | LD |
| imm[11:5] | rs2 | rs1 | 011 | imm[4:0] | 0100011 | SD |

**RV32I Base Instruction Set**

| | | | | | | |
|---|---|---|---|---|---|---|
| imm[31:12] | | | | rd | 0110111 | LUI |
| imm[31:12] | | | | rd | 0010111 | AUIPC |
| imm[20\|10:1\|11\|19:12] | | | | rd | 1101111 | JAL |
| imm[11:0] | | rs1 | 000 | rd | 1100111 | JALR |
| imm[12\|10:5] | rs2 | rs1 | 000 | imm[4:1\|11] | 1100011 | BEQ |
| imm[12\|10:5] | rs2 | rs1 | 001 | imm[4:1\|11] | 1100011 | BNE |
| imm[12\|10:5] | rs2 | rs1 | 100 | imm[4:1\|11] | 1100011 | BLT |
| imm[12\|10:5] | rs2 | rs1 | 101 | imm[4:1\|11] | 1100011 | BGE |
| imm[12\|10:5] | rs2 | rs1 | 110 | imm[4:1\|11] | 1100011 | BLTU |
| imm[12\|10:5] | rs2 | rs1 | 111 | imm[4:1\|11] | 1100011 | BGEU |
| imm[11:0] | | rs1 | 000 | rd | 0000011 | LB |
| imm[11:0] | | rs1 | 001 | rd | 0000011 | LH |
| imm[11:0] | | rs1 | 010 | rd | 0000011 | LW |
| imm[11:0] | | rs1 | 100 | rd | 0000011 | LBU |
| imm[11:0] | | rs1 | 101 | rd | 0000011 | LHU |
| imm[11:5] | rs2 | rs1 | 000 | imm[4:0] | 0100011 | SB |
| imm[11:5] | rs2 | rs1 | 001 | imm[4:0] | 0100011 | SH |
| imm[11:5] | rs2 | rs1 | 010 | imm[4:0] | 0100011 | SW |
| imm[11:0] | | rs1 | 000 | rd | 0010011 | ADDI |
| imm[11:0] | | rs1 | 010 | rd | 0010011 | SLTI |
| imm[11:0] | | rs1 | 011 | rd | 0010011 | SLTIU |
| imm[11:0] | | rs1 | 100 | rd | 0010011 | XORI |
| imm[11:0] | | rs1 | 110 | rd | 0010011 | ORI |
| imm[11:0] | | rs1 | 111 | rd | 0010011 | ANDI |
| 0000000 | shamt | rs1 | 001 | rd | 0010011 | SLLI |
| 0000000 | shamt | rs1 | 101 | rd | 0010011 | SRLI |
| 0100000 | shamt | rs1 | 101 | rd | 0010011 | SRAI |
| 0000000 | rs2 | rs1 | 000 | rd | 0110011 | ADD |
| 0100000 | rs2 | rs1 | 000 | rd | 0110011 | SUB |
| 0000000 | rs2 | rs1 | 001 | rd | 0110011 | SLL |
| 0000000 | rs2 | rs1 | 010 | rd | 0110011 | SLT |
| 0000000 | rs2 | rs1 | 011 | rd | 0110011 | SLTU |
| 0000000 | rs2 | rs1 | 100 | rd | 0110011 | XOR |
| 0000000 | rs2 | rs1 | 101 | rd | 0110011 | SRL |
| 0100000 | rs2 | rs1 | 101 | rd | 0110011 | SRA |
| 0000000 | rs2 | rs1 | 110 | rd | 0110011 | OR |
| 0000000 | rs2 | rs1 | 111 | rd | 0110011 | AND |
| fm | pred succ | rs1 | 000 | rd | 0001111 | FENCE |
| 000000000000 | | 00000 | 000 | 00000 | 1110011 | ECALL |
| 000000000001 | | 00000 | 000 | 00000 | 1110011 | EBREAK |

## 3. Architecture Diagram



- The components for J-type instructions are not included in this diagram. You need to design by yourself.

## 4. Grade

a. Total: 100 points + 20 bonus (plagiarism will get 0 point)

b. No late submission

## 5. Hand in

a. These files don't need to submit

- Data_Memory.v
- Instr_Memory.v
- Reg_File.v
- ProgramCounter.v
- testbench.v
- Adder.v
- test directory, testcast1.txt, testcase2.txt

b. Please zip the archive, name it as "ID.zip" and upload the assignment to ceiba

## 6. Q&A

For any questions regarding hw4, please contact 范航熏 (rr1155001100@gmail.com)

## 7. Reference

RISC-V spec