CENG3420
Lab 2-2: RISC-V Simulator

Chen BAI

Department of Computer Science and Engineering
The Chinese University of Hong Kong

cbai@cse.cuhk.edu.hk

Spring 2021

香港中文大學
The Chinese University of Hong Kong

Overview

Background

RV32I Instructions

Lab Assigment

Overview

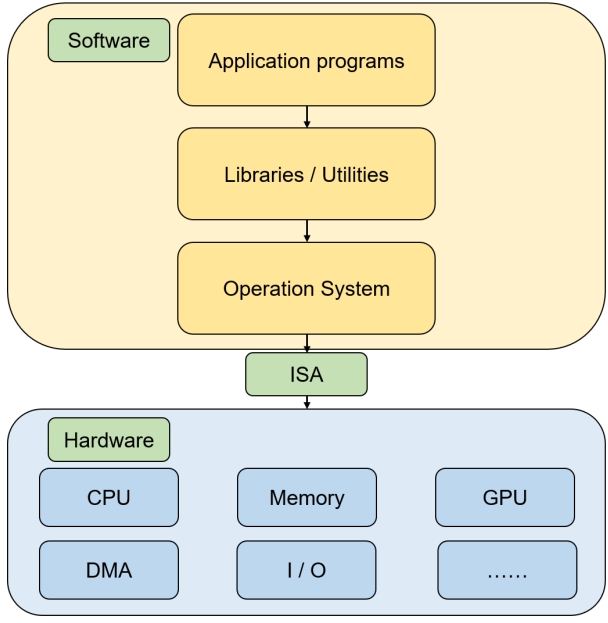
RISC-V assembler

- In lab2-1, we should implement a naive RISC-V assembler, although it may not observe the RISC-V specifications strictly!
- In lab2-2, we are going to implement a naive RISC-V simlulator with binaries output by our RISC-V assembler

RISC-V simulator

- Spike, QEMU, rv8...
- Spike is the official simulator provided by the RISC-V community

Recap our computer architecture



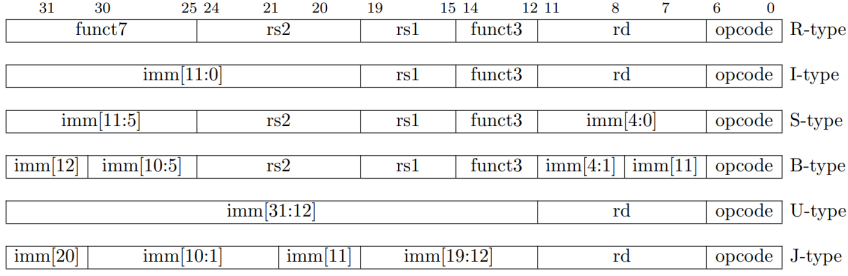
Overview

Background

RV32I Instructions

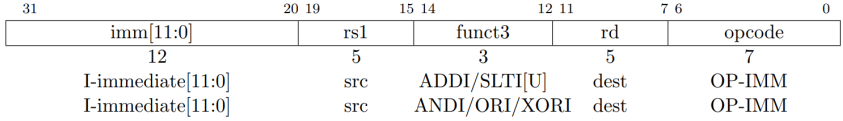
Lab Assigment

RV32I instructions



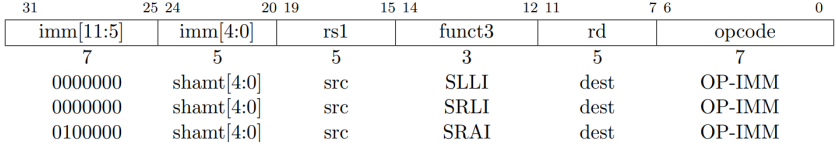
RV32I base instructions

Integer Register-Immediate Instructions



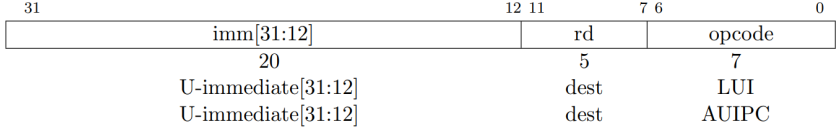
31 — 20	19 — 15	14 — 12	11 — 7	6 — 0
imm[11:0]	rs1	funct3	rd	opcode
12	5	3	5	7
I-immediate[11:0]	src	ADDI/SLTI[U]	dest	OP-IMM
I-immediate[11:0]	src	ANDI/ORI/XORI	dest	OP-IMM

addi, andi, ori, xori



31 — 25	24 — 20	19 — 15	14 — 12	11 — 7	6 — 0
imm[11:5]	imm[4:0]	rs1	funct3	rd	opcode
7	5	5	3	5	7
0000000	shamt[4:0]	src	SLLI	dest	OP-IMM
0000000	shamt[4:0]	src	SRLI	dest	OP-IMM
0100000	shamt[4:0]	src	SRAI	dest	OP-IMM

slli, srli, srai



31 — 12	11 — 7	6 — 0
imm[31:12]	rd	opcode
20	5	7
U-immediate[31:12]	dest	LUI
U-immediate[31:12]	dest	AUIPC

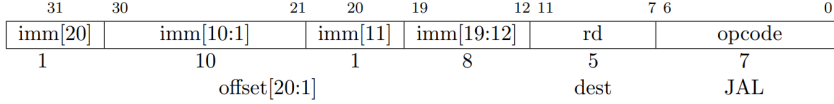
lui

Integer Register-Register Instructions

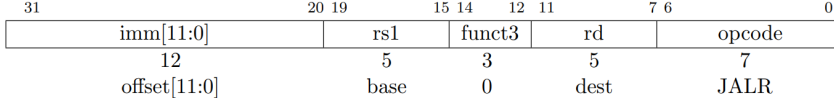
31 25	24 20	19 15	14 12	11 7	6 0
funct7	rs2	rs1	funct3	rd	opcode
7	5	5	3	5	7
0000000	src2	src1	ADD/SLT/SLTU	dest	OP
0000000	src2	src1	AND/OR/XOR	dest	OP
0000000	src2	src1	SLL/SRL	dest	OP
0100000	src2	src1	SUB/SRA	dest	OP

add, and, or, xor, sll, srl, sub, sra

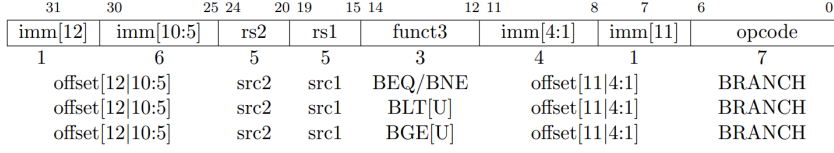
Control Transfer Instructions



jal

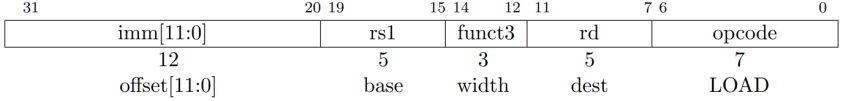


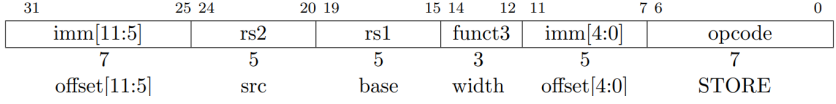
jalr



beq, bne, blt, bge

Load and Store Instructions





lb, lh, lw, sb, sh, sw

Other Instructions

- LA (translate to lui and addi)
- .FILL & HALT

Overview

Background

RV32I Instructions

Lab Assigment

Lab2.2 assignment

A framework of the naive RV32I simluator (released on Mar. 17)

- git clone https://github.com/baichen318/ceng3420.git
- git checkout lab2.2

Compile (Linux environment is suggested)

- make

Run the simulator

- ./sim isa.bin # simulate with isa.bin

Lab2.2 assignment

Finish the RV32I simulator including 26+ instructions and other operations as follows

- Pseudo instruction: la
- Integer Register-Immediate Instructions: slli, xori, srli, srai, ori, andi, lui
- Integer Register-Register Operations: sub, sll, xor, srl, sra, or, and
- Unconditional Jumps: jalr, jal
- Conditional Branches: bne, blt, bge
- Load and Store Instructions: lb, lh, lw, sb, sh, sw
- Other operations: Read the memory, Sign extension, PC update, Instruction decoding, etc.

These unimplemented codes are annotated with Lab2-2 assignment

Lab2.2 assignment

Verify your codes with binary machine codes suffixed with .bin

- isa.bin
- count10.bin
- swap.bin

Submission Method:

Prepare a package onto blackboard, including

- All source codes (C source files, header files, Makefile, etc.)
- A lab report (`<name-sid>-lab2.pdf`) with all console results.
- Deadline: 23:59, 2 April