CENG3420

Lab 3-1: RISCV-LC Datapath

Chen BAI

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

cbai@cse.cuhk.edu.hk

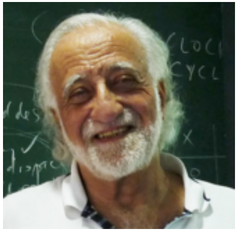
Spring 2021

香 港 中 文 大 學
The Chinese University of Hong Kong

Overview

LC-3b

- LC-3b: **Little Computer 3, b** version.
- Relatively simple instruction set
- Most used in teaching for CS & CE
- Developed by Yale Patt@UT & Sanjay J. Patel@UIUC



RISCV-LC

- Inspired from LC-3b
- RISC-V version (RV32I)
- Programming language: C
- Compatible with Lab2 (RV32I assembler & RV32I simulator)

RISCV-LC

- RISC-V 32 general-purpose registers
- 32-bit data and address
- 28+ instructions (including pseudo instructions)

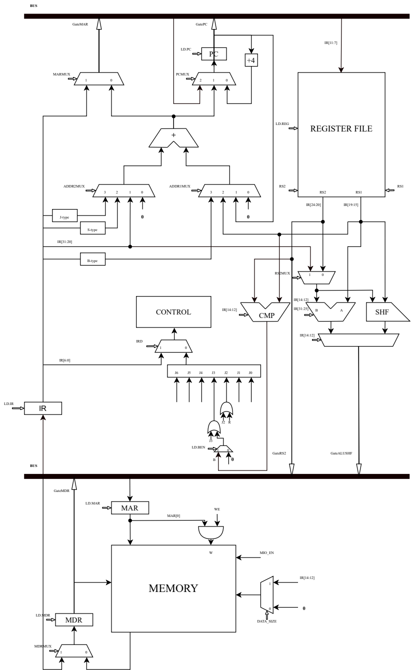
Plus 4 special-purpose registers:

- Program Counter (PC)
- Instruction Register (IR)
- Memory Access Register (MAR)
- Memory Data Register (MDR)

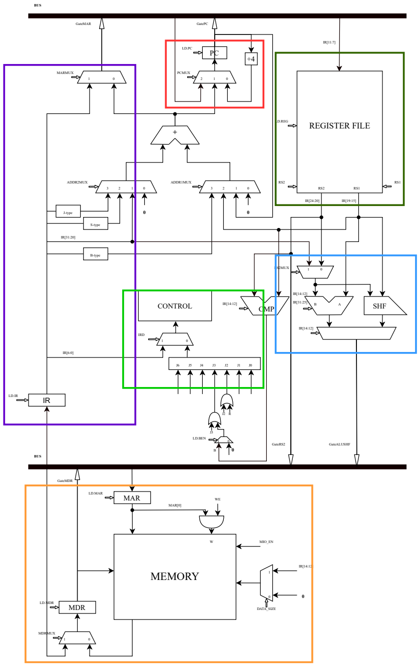
In order to make labs easy, I have modified some definitions of instructions, and they do not oberseve RISC-V specifications strictly!

Overview

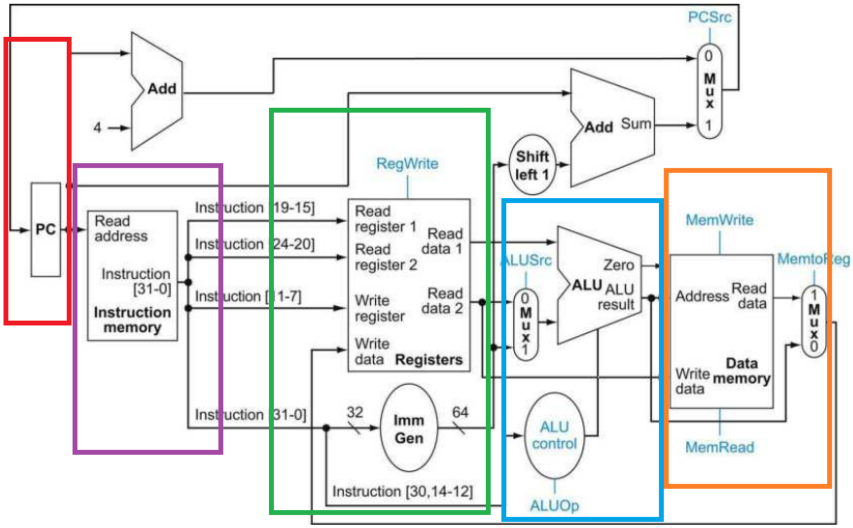
Datapath


BUS
GateMAR
GatePC
LD.PC
PC
+4
MARMUX
PCMUX
IR[11:7]
LD.REG
REGISTER FILE
+
ADDR2MUX
ADDR1MUX
RS2
RS1
IR[24:20]
IR[19:15]
J-type
S-type
IR[31:20]
B-type
RS2MUX
IR[14:12]
CMP
IR[31:25]
SHF
CONTROL
IRD
IR[6:0]
J6 J5 J4 J3 J2 J1 J0
LD.IR
IR
LD.BEN
GateRS2
GateALUSHF
BUS
GateMDR
LD.MAR
MAR
MAR[0]
WE
W
MIO_EN
MEMORY
IR[14:12]
DATA_SIZE
LD.MDR
MDR
MDRMUX

Datapath Annotation



Datapath Annotation

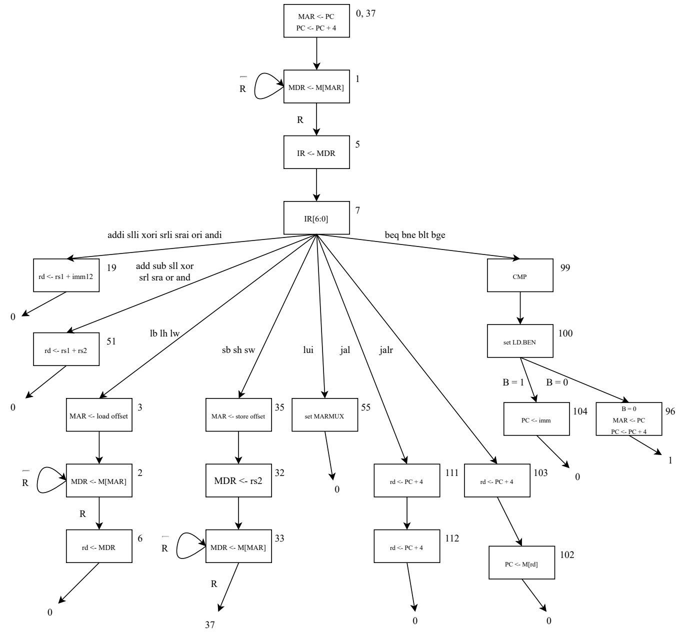


RISCV datapath annotation based on the textbook

FSM

- Clock cycle: The cycle time of a microprocessor is the duration of a clock cycle
- Finite State Machine: The behavior of the RISCV-LC microarchitecture during a given clock cycle is completely determined by the 33 control signals. These 33 control signals specify the state of the control structure of the RISCV-LC microarchitecture.

FSM



MAR <- PC
PC <- PC + 4
0, 37
MDR <- M[MAR]
1
$\overline{R}$
R
IR <- MDR
5
IR[6:0]
7
addi slli xori srli srai ori andi
beq bne blt bge
rd <- rs1 + imm12
19
0
add sub sll xor srl sra or and
rd <- rs1 + rs2
51
0
lb lh lw
sb sh sw
lui
jal
jalr
CMP
99
set LD.BEN
100
B = 1
B = 0
MAR <- load offset
3
MAR <- store offset
35
set MARMUX
55
PC <- imm
104
B = 0
MAR <- PC
PC <- PC + 4
96
1
0
MDR <- M[MAR]
2
$\overline{R}$
R
MDR <- rs2
32
0
rd <- PC + 4
111
rd <- PC + 4
103
rd <- MDR
6
$\overline{R}$
MDR <- M[MAR]
33
rd <- PC + 4
112
PC <- M[rd]
102
0
R
37
0
0

Overview

Operations in One Clock Cycle

In “riscv-lc.c”:

```
void cycle() {
    eval_micro_sequencer();
    cycle_memory();
    eval_bus_drivers();
    drive_bus();
    latch_datapath_values();

    CURRENT_LATCHES = NEXT_LATCHES;

    CYCLE_COUNT++;
}
```

Lab3.1 Assignment

RISCV-LC codes will be released on 21 Apr.

- git clone https://github.com/baichen318/ceng3420.git; cd ceng3420
- git checkout lab3.1

Compile (Linux + x86_64 environment is suggested)

- make

Run the simulator

- ./lc uop benchmarks/isa.bin # RISCV-LC runs isa.bin

Lab3.1 assignment

- Complete uop (i.e., fill x with right 1s or 0s)
- Finish eval_micro_sequencer

These unimplemented codes are annotated with Lab3-1 assignment

Lab3.1 assignment

Verify your codes with binary machine codes suffixed with .bin

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

Submission Method:

Submit the source code and report after the whole Lab3, onto blackboard.

Thanks. For any question:

byu@cse.cuhk.edu.hk

cbai@cse.cuhk.edu.hk

ybai@cse.cuhk.edu.hk