

CSC 411

Computer Organization (Spring 2023) Lecture 7: Executing Instructions

Prof. Marco Alvarez, University of Rhode Island

RISC-V interpreter

RISC-V Interpreter

Input your RISC-V code here:

```
1 lui t0, 0x0
2 lui t1, 0xFFFF
3
4
5
6
7
8
9
10
11
12
13
14
15
16
```

Reset Stop Run CPU: 32 Hz

The most recent instructions will be shown here when stepping.

Features

- Reset to load the code, Stop one instruction, or Run all instructions
- Set a breakpoint by clicking on the line number (only for Run)
- View registers on the right, memory on the bottom of this page

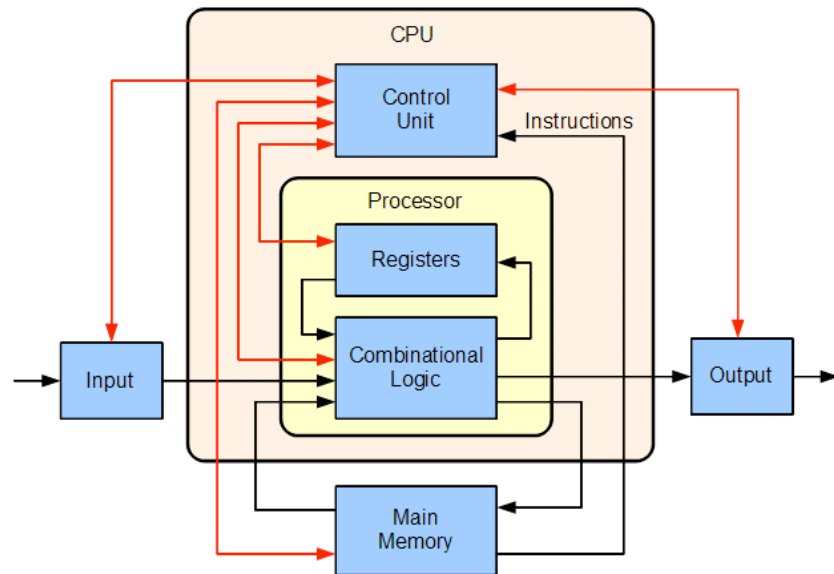
Supported Instructions

- Arithmetic: ADD, ADDI, SUB
- Logical: AND, ANDI, OR, ORI, XOR, XORI
- Sets: SLT, SLTI, SLTU, SLTIU
- Shifts: SRAI, SRAI, SRL, SRLI, SLLI
- Memory: LW, SW, LB, SB
- PC: LUI, AUIPC
- Jumps: JAL, JALR
- Branches: BEQ, BEQZ, BNE, BNEZ, BLT, BLTZ, BLTU, BLTZU

RISC-V Reference: [riscv-spec-v2.2.pdf](https://riscv.org/specs/riscv-spec-v2.2.pdf)

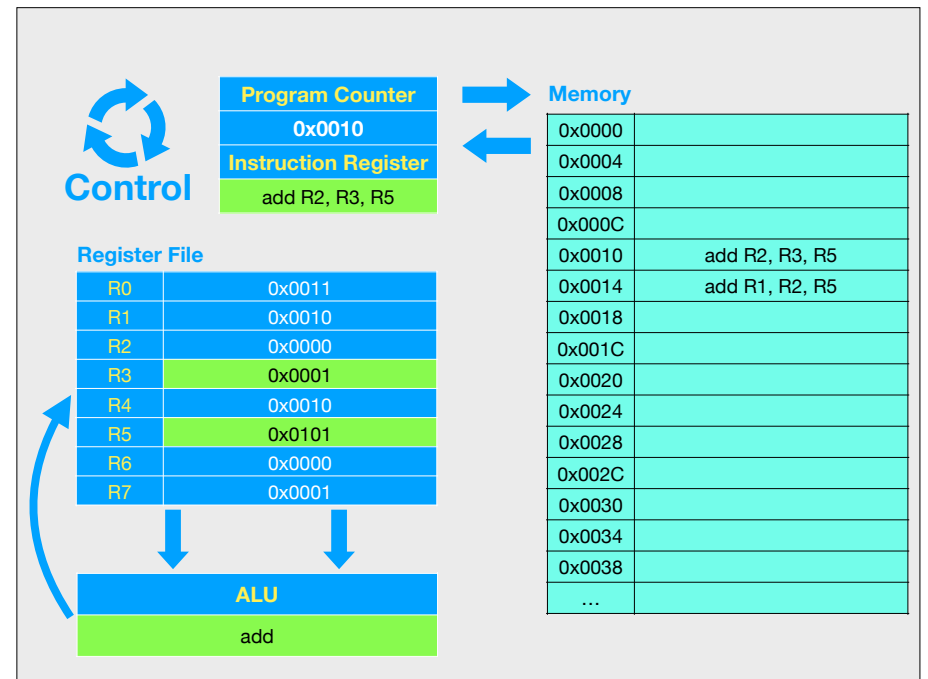
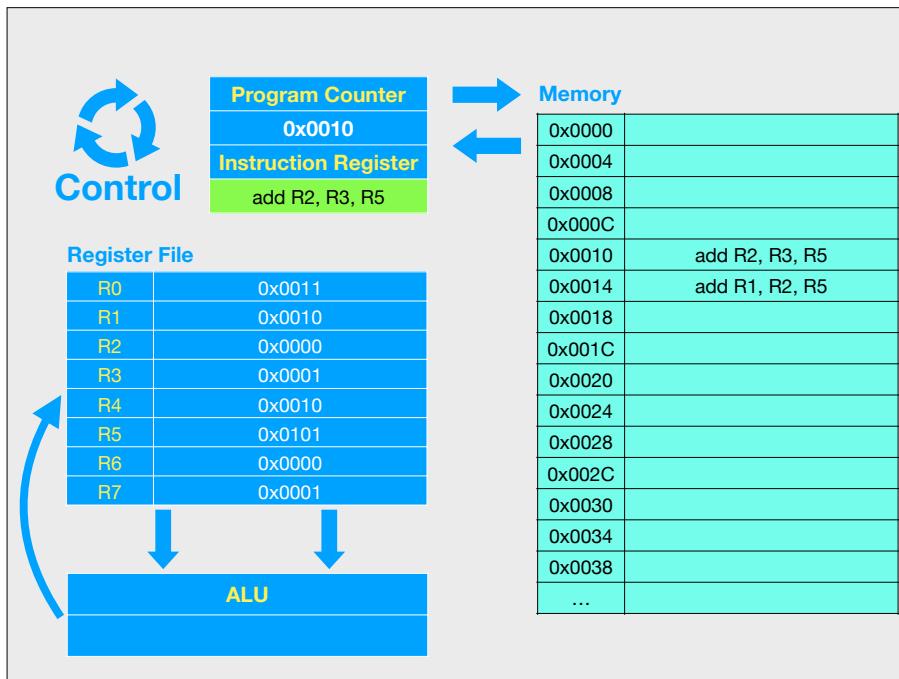
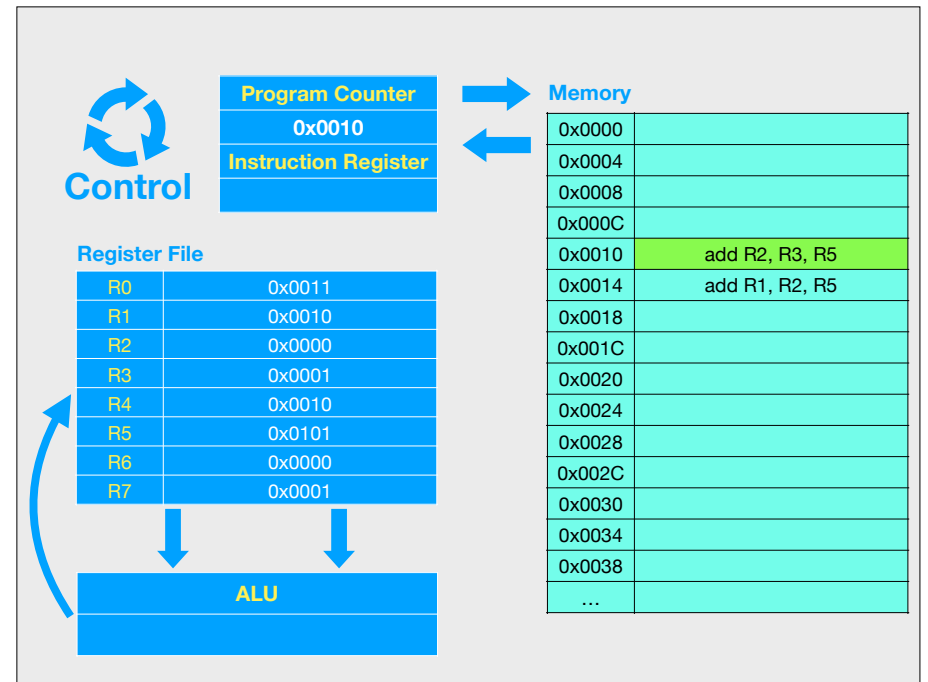
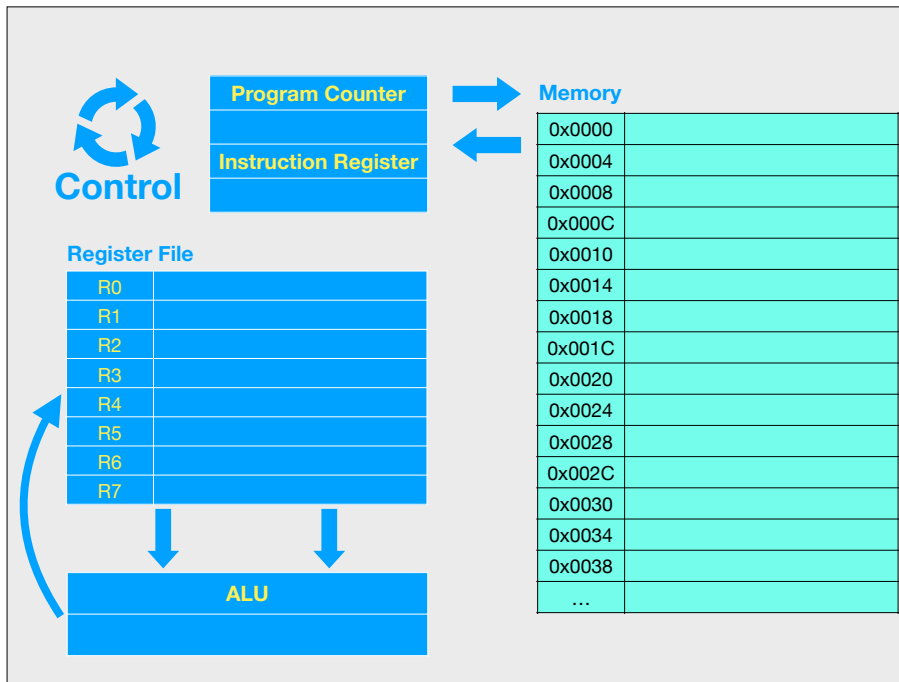
Init Value	Register	Decimal	Hex	Binary
0	x0 (zero)	0	0x00000000	0x00000000000000000000000000000000
0	x1 (ra)	0	0x00000000	0x00000000000000000000000000000000
0	x2 (sp)	0	0x00000000	0x00000000000000000000000000000000
0	x3 (gp)	0	0x00000000	0x00000000000000000000000000000000
0	x4 (tp)	0	0x00000000	0x00000000000000000000000000000000
0	x5 (t0)	0	0x00000000	0x00000000000000000000000000000000
0	x6 (t1)	0	0x00000000	0x00000000000000000000000000000000
0	x7 (t2)	0	0x00000000	0x00000000000000000000000000000000
0	x8 (a0/rp)	0	0x00000000	0x00000000000000000000000000000000
0	x9 (a1)	0	0x00000000	0x00000000000000000000000000000000
0	x10 (a2)	0	0x00000000	0x00000000000000000000000000000000
0	x11 (a3)	0	0x00000000	0x00000000000000000000000000000000
0	x12 (a4)	0	0x00000000	0x00000000000000000000000000000000
0	x13 (a5)	0	0x00000000	0x00000000000000000000000000000000
0	x14 (a6)	0	0x00000000	0x00000000000000000000000000000000
0	x15 (a7)	0	0x00000000	0x00000000000000000000000000000000
0	x16 (a8)	0	0x00000000	0x00000000000000000000000000000000
0	x17 (a9)	0	0x00000000	0x00000000000000000000000000000000
0	x18 (a10)	0	0x00000000	0x00000000000000000000000000000000
0	x19 (a11)	0	0x00000000	0x00000000000000000000000000000000
0	x20 (a12)	0	0x00000000	0x00000000000000000000000000000000
0	x21 (a13)	0	0x00000000	0x00000000000000000000000000000000
0	x22 (a14)	0	0x00000000	0x00000000000000000000000000000000
0	x23 (a15)	0	0x00000000	0x00000000000000000000000000000000
0	x24 (a16)	0	0x00000000	0x00000000000000000000000000000000
0	x25 (a17)	0	0x00000000	0x00000000000000000000000000000000
0	x26 (a18)	0	0x00000000	0x00000000000000000000000000000000
0	x27 (a19)	0	0x00000000	0x00000000000000000000000000000000
0	x28 (a20)	0	0x00000000	0x00000000000000000000000000000000
0	x29 (a21)	0	0x00000000	0x00000000000000000000000000000000
0	x30 (a22)	0	0x00000000	0x00000000000000000000000000000000
0	x31 (a23)	0	0x00000000	0x00000000000000000000000000000000

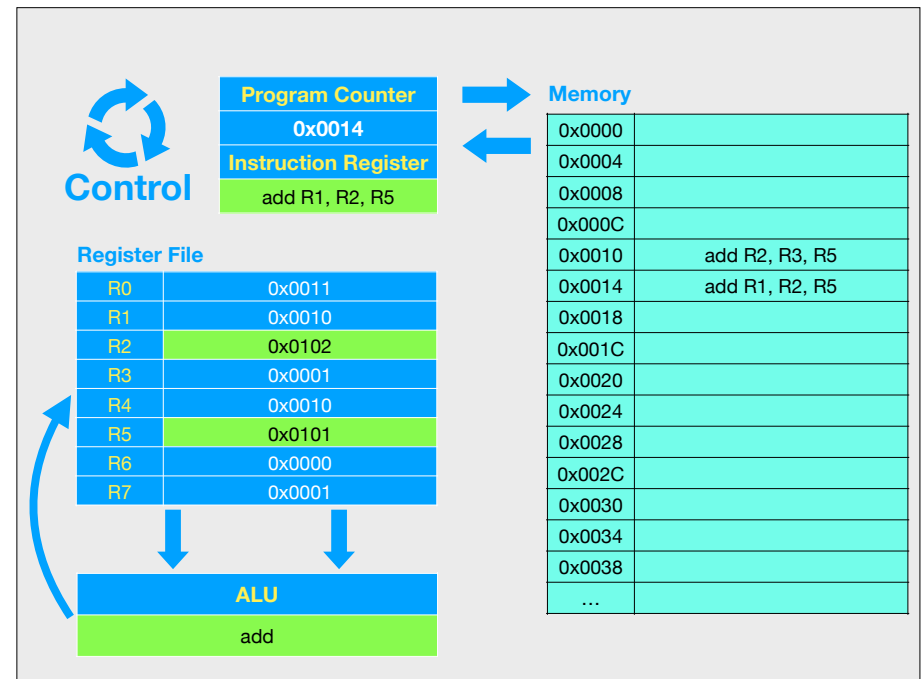
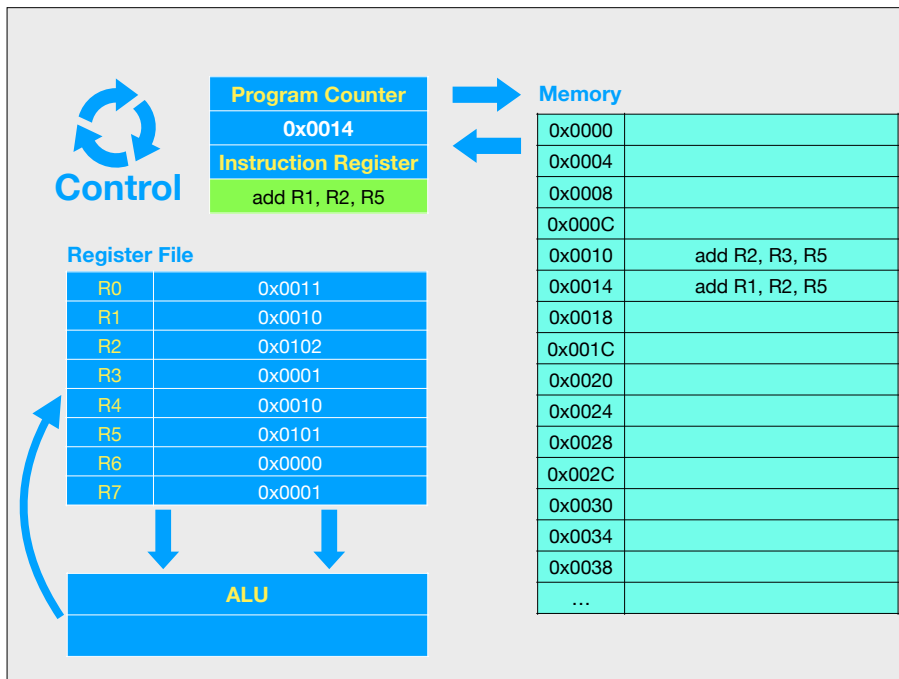
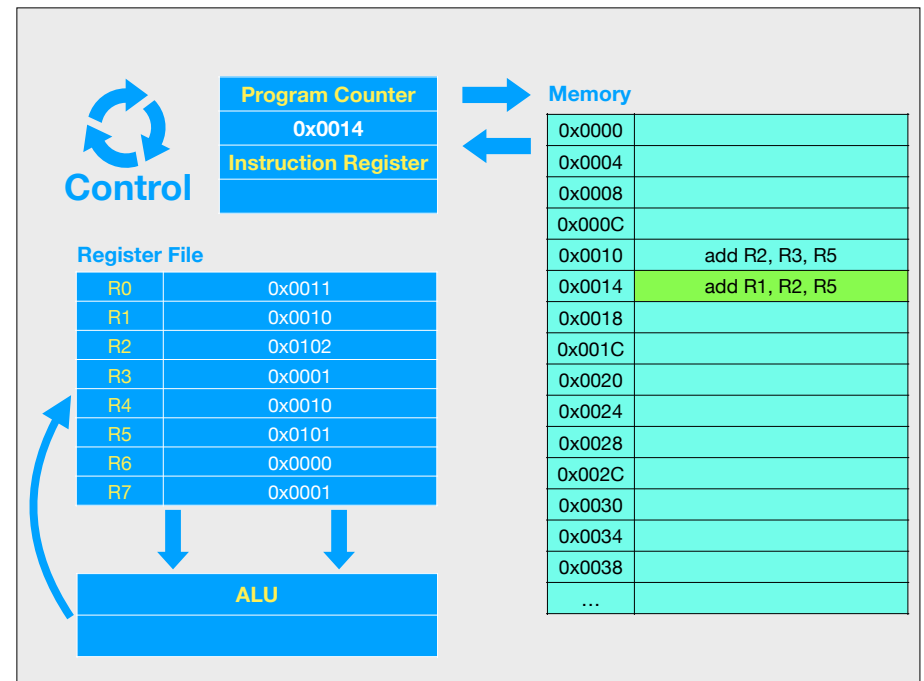
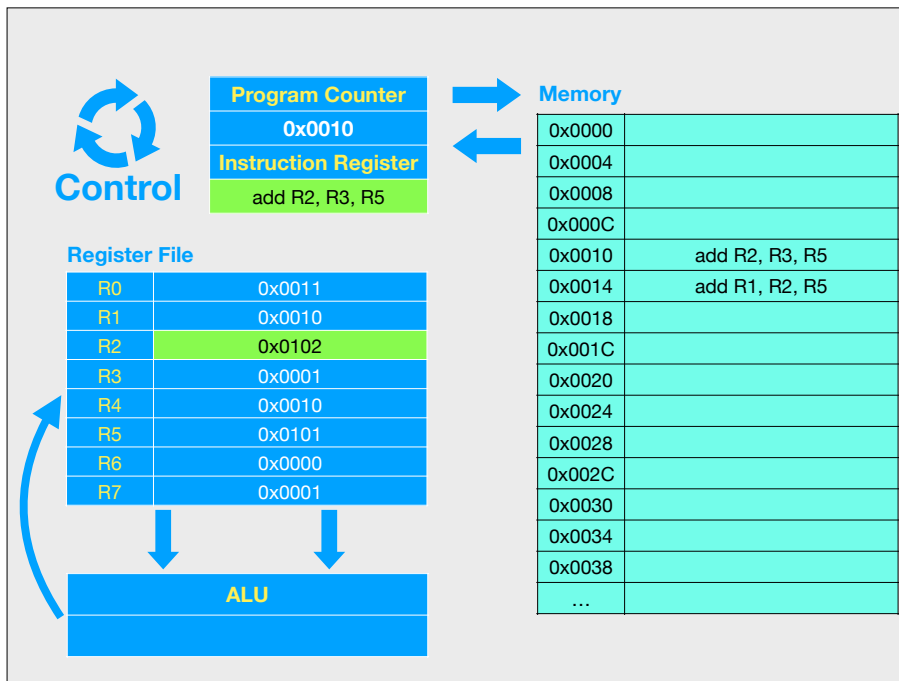
<https://www.cs.cornell.edu/courses/cs3410/2019sp/riscv/interpreter/>

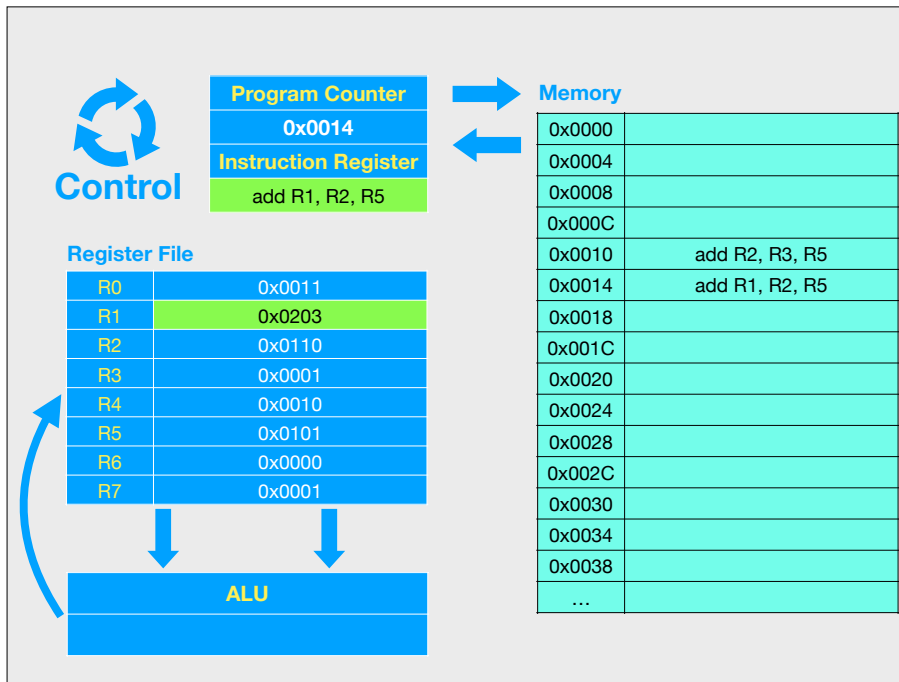


https://en.wikipedia.org/wiki/Computer_architecture

Executing instructions







Executing memory instructions

