RISC-V Simulator

Educational CPU Architecture Simulator

20 Example Programs Available

Execution Controls

Run | Step | Cycle | Reset

7	36
Instructions	Cycles

Machine Cycle

Current Phase: DECODE

Decoding instruction opcode and operands

1 FETCH	2 DECODE	3 EXECUTE	4 MEMORY	5 WRITEBACK

Machine Status

Status:	STOPPED
Current Phase:	DECODE

Instructions	Cycles
7	36

CPI (Cycles per Instruction): 5.14

Program Counter & Addressing

Program Counter (PC) Current Address

0x0000001C

Instruction #8 of 8

Next PC

0x0000001C

Execution Progress 87.5%

Byte Address
28

Word Address
7

Assembly Code

Memory Pattern

```
# Store pattern in memory
addi x1, x0, 1      # Start with 1
sw x1, 0(x0)        # mem[0] = 1
add x1, x1, x1      # x1 = 2
sw x1, 4(x0)        # mem[4] = 2
add x1, x1, x1      # x1 = 4
sw x1, 8(x0)        # mem[8] = 4
add x1, x1, x1      # x1 = 8
sw x1, 12(x0)       # mem[12] = 8
```

Parsed Instructions

0x0000	667F9F88	addi x1, x0, 1 # Start with 1
0x0004	56E4F2FF	sw x1, 0(x0) # mem[0] = 1
0x0008	8CF9F378	add x1, x1, x1 # x1 = 2
0x000c	E0968275	sw x1, 4(x0) # mem[4] = 2
0x0010	C4AF3F51	add x1, x1, x1 # x1 = 4

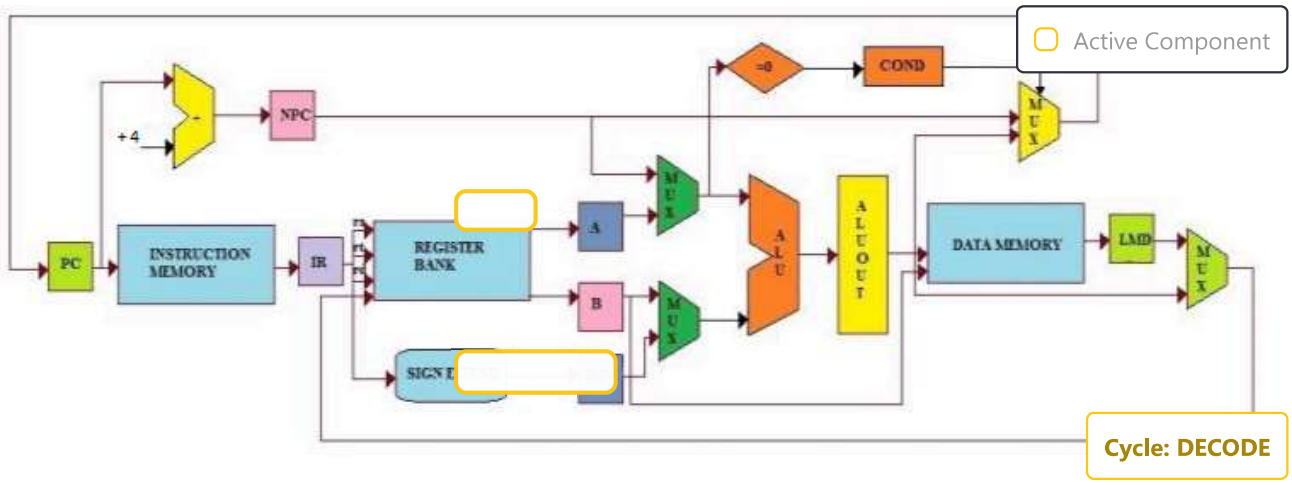
Registers

x0 (zero)	00000000	x1 (ra)	00000008
x2 (sp)	7FFFFFFC	x3 (gp)	00000000
x4 (tp)	00000000	x5 (t0)	00000000
x6 (t1)	00000000	x7 (t2)	00000000
x8 (s0/fp)	00000000	x9 (s1)	00000000
x10 (a0)	00000000	x11 (a1)	00000000

Memory

0x00000000	00000001
0x00000004	00000002
0x00000008	00000004
0x0000000C	00000000
0x00000010	00000000
0x00000014	00000000
0x00000018	00000000
0x0000001C	00000000
0x00000020	00000000

RISC-V CPU Architecture



Current Operations:

```
Decode: SW (S-type)
Control signals generated
```

Instruction Decoder

Current Instruction

S-Type

```
sw x1, 12(x0) # mem[12] = 8
```

Machine Code: 7F1E5ADB

32-bit Binary Representation

```
0000000 00000 00000 010 00000 0100011
```

31 0

Decoded Fields

Opcode SW	Source 1 (rs1) x0
Source 2 (rs2) x1	Immediate 12

S-Type Format

Fields: immediate[11:5] | rs2 | rs1 | funct3 | immediate[4:0] | opcode

Address Calculations

No active address calculations

Execute memory/branch instructions to see addresses

Address Types

- Memory Access
- Branch Target
- Jump Target
- Stack Operation

🚀 Step through instructions to see live register and memory updates

💡 Use the example programs to learn RISC-V assembly language