Design and Implementation of a Single-Cycle RISC-V Processor

Ahmed Mohamed Abdelmaboud
Yahia Hany Gaber
Omar Hesham Desouky
Mahmoud Khaled
Mazen Ahmed

Computer Architecture Project

Project Overview

Objective

Design a 64-bit single-cycle RISC-V processor that executes each instruction in one clock cycle.

- Based on RV64I instruction set
- Complete single-cycle implementation
- Modular Verilog design
- Verified with Vivado simulations

Key Features

Architecture

- 64-bit data paths
- 32 registers
- Single-cycle execution

Implementation

- Verilog HDL
- Vivado simulation
- Modular design
- Comprehensive testing

Instruction Set Categories

Supported Instruction Types

- R-Type: Register-register operations
- I-Type: Register-immediate operations
- Load/Store: Memory access
- Branch: Control flow
- Jump: Unconditional jumps
- U-Type: Upper immediate

R-Type Instructions

Opcode: 0110011

Category	**Instructions**
Arithmetic	add, sub
Logical	and, or, xor
Shift	sll, srl, sra
Comparison	slt, sltu

I-Type Instructions

Opcode: 0010011

Category	**Instructions**
Arithmetic	addi
Logical	andi, ori, xori
Shift	slli, srli, srai
Comparison	slti, sltiu

Memory Instructions

Load Instructions

Opcode: 0000011

- ld (load doubleword)

Store Instructions

Opcode: 0100011

- sd (store doubleword)

Control Flow Instructions

Branch Instructions

Opcode: 1100011

- beq (branch if equal)

Jump Instructions

- jal (Opcode: 1101111)
- jalr (Opcode: 1100111)

Upper Immediate Instructions

LUI

Opcode: 0110111

- Load Upper Immediate

AUIPC

Opcode: 0010111

- Add Upper Immediate to PC

ALU Operations

ALU Control	Operation
0000	ADD
0001	SUB
0010	AND
0011	OR
0100	XOR
0101	SLT (signed)
0110	SLL
0111	SRL
1000	SRA
1001	SLTU (unsigned)

ALU Features

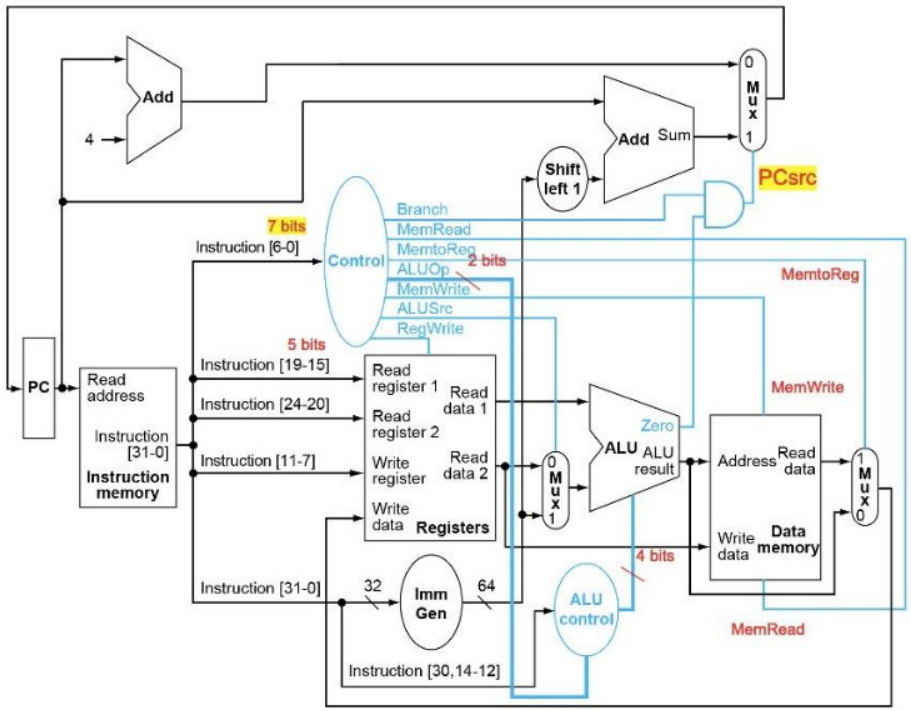
Key Characteristics

- 10 distinct operations
- 64-bit data paths
- Zero flag output
- Signed/unsigned operations

Usage

- All arithmetic/logical operations
- Memory address calculation
- Branch comparisons

Datapath Overview



Datapath Components

- Program Counter (PC)
- Instruction Memory
- Register File
- ALU
- Data Memory
- Immediate Generator
- Control Unit

Control Signals

Signal	Purpose
RegWrite	Register write enable
MemRead	Memory read enable
MemWrite	Memory write enable
MemToReg	Memory to register
ALUSrc	ALU source select
ALUOp	ALU operation class
pc_src	Next PC source
imm_type	Immediate format

Immediate Formats

Type	Format
000	I-type (12-bit)
001	S-type (12-bit)
010	B-type (13-bit)
011	U-type (20-bit)
100	J-type (21-bit)

Control Unit Design

Merged Control Unit

- Combines main and ALU control
- Decodes opcode, funct3, funct7
- Generates all control signals

Inputs

- opcode[6:0]
- funct3[2:0]
- funct7[6:0]

ALU Control Logic

ALUOp	Operation
00	Force ADD
01	Force SUB
10	Use funct3/funct7 (R-type)
11	Use funct3 (I-type)

Register File

Specifications

- 32 general-purpose registers
- 64-bit registers
- Register x0 hardwired to zero
- 2 read ports, 1 write port

Memory Units

Instruction Memory

- Stores program code
- Read-only during execution
- 64-bit access

Data Memory

- Stores data
- Read/write access
- 64-bit load/store

Test Program Example

Sample Code

```
 addi x0, x0, 7
addi x11, x0, 196
addi x3, x0, 4
add x2, x11, x3
and x24, x2, x11
srl x27, x24, x3
sd x27, 0(x0)
ld x19, 0(x0)
beq x19, x27, L1
addi x13, x27, 7
beq x0, x0, END
L1:  addi x13, x0, 17
END: addi x0, x0, 0
```

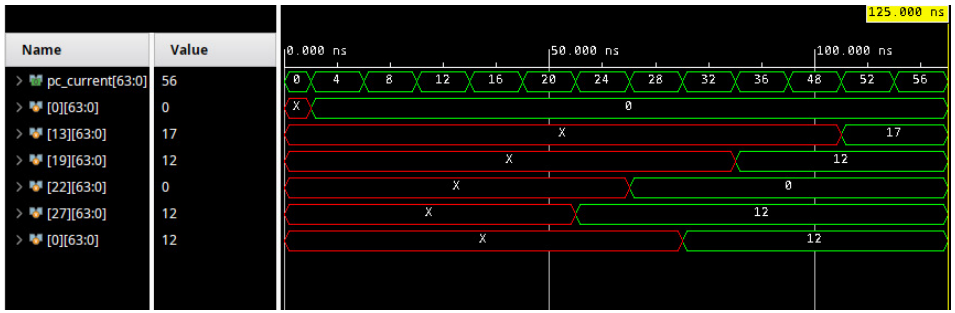
Expected Results

Register Values

- x0 = 0 (always)
- x27 = 12
- x19 = 12
- x13 = 17

Memory State

- mem[0] = 12



Simulation Environment

Tools Used

- Verilog HDL
- Vivado Design Suite
- Testbenches

Verification Results

Verified Components

- PC updates
- Instruction decoding
- Register operations
- ALU operations
- Memory access
- Branch execution

Project Achievements

- Complete 64-bit RISC-V processor
- 10 ALU operations implemented
- All instruction types supported
- Clean modular design
- Successful simulation verification

Conclusion

Successful Implementation

- Single-cycle RISC-V processor
- Complete RV64I support
- Verified functionality
- Modular and scalable design

Thank You!

Questions?