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

Abstract

Objective

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

- Supports arithmetic, logic, memory access, and branching
- Follows standard single-cycle RISC-V datapath
- Merged Main Control Unit and ALU Control Unit
- Implemented in Verilog HDL with simulation verification

Index Terms: RISC-V, Single-Cycle Processor, Computer Architecture, Verilog, CPU Design

Introduction

Project Goal

Educational implementation of a single-cycle RISC-V processor for a Computer Architecture course.

- **Single-cycle architecture:** Each instruction completes in one clock cycle
- Not performance-optimized, but educationally valuable
- Clear demonstration of datapath and control design
- Supports:
  - R-type and I-type arithmetic/logic instructions
  - Memory access (load/store)
  - Conditional branching

Supported Instruction Set

R-Type Instructions

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

I-Type Arithmetic Instructions

- addi, andi, ori, xori

Shift Instructions

- Immediate shifts: slli, srli
- Register-based: sll, srl

Memory Instructions

- ld (load doubleword)
- sd (store doubleword)

Branch Instruction

- beq (branch if equal)

Datapath Design

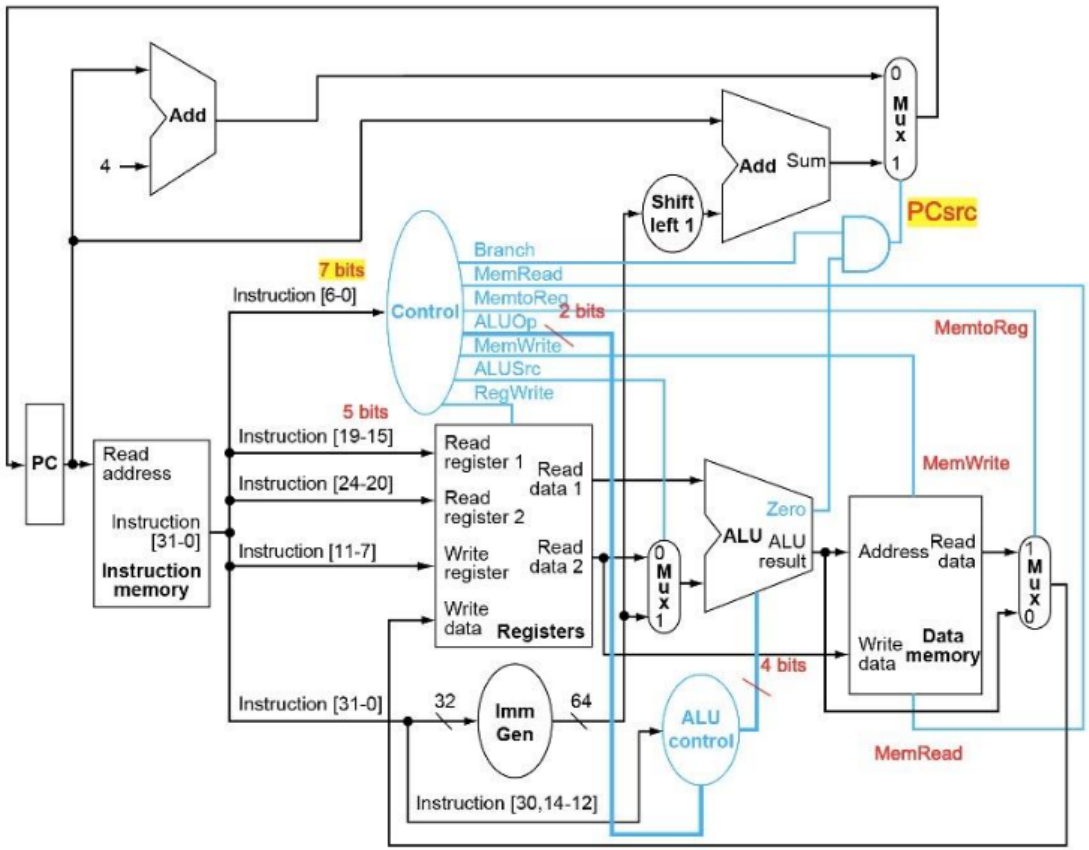

Fig. 1: Single-Cycle RISC-V Datapath

Dataapath Components

- **Program Counter (PC):** Holds address of next instruction
- **Instruction Memory:** Stores program instructions
- **Register File:** 32 registers for data storage and manipulation
- **ALU:** Performs arithmetic and logic operations
- **Data Memory:** For load/store operations
- **Sign/Zero Extender:** Extends immediate values to 64 bits
- **Multiplexers (MUXes):** Select between different data sources

Control Unit Signals (Selected Instructions)

Instruction	RegWrite	MemRead	MemWrite	MemToReg	Branch	ALUSrc	AL
R-type (add, sub, etc.)	1	0	0	0	0	0	
addi, andi, etc.	1	0	0	0	0	1	
ld	1	1	0	1	0	1	
sd	0	0	1	X	0	1	
beq	0	0	0	X	1	0	

ALUOp	funct3	funct7	ALU Operation
00	XXX	XXXXXXX	ADD (for loads/stores)
01	XXX	XXXXXXX	SUB (for branches)
10	000	0000000	ADD
10	000	0100000	SUB
10	111	0000000	AND
10	110	0000000	OR
10	100	0000000	XOR

Test Program

Assembly Test Program

```
 addi x0, x0, 7
addi x11, x0, 196
addi x3, x0, 4
add x2, x11, x3
and x24, x2, x11
srl x27, x24, x3
sub x22, x27, x27
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

- Register x0 = 0 (hardwired to zero)
- Register x27 = 12
- Register x22 = 0
- Register x19 = 12
- Register x13 = 17

Memory State

- Memory location mem[0] = 12

Verification Method

- Values derived analytically
- Used as reference for simulation
- Tests all instruction types

Simulation Results

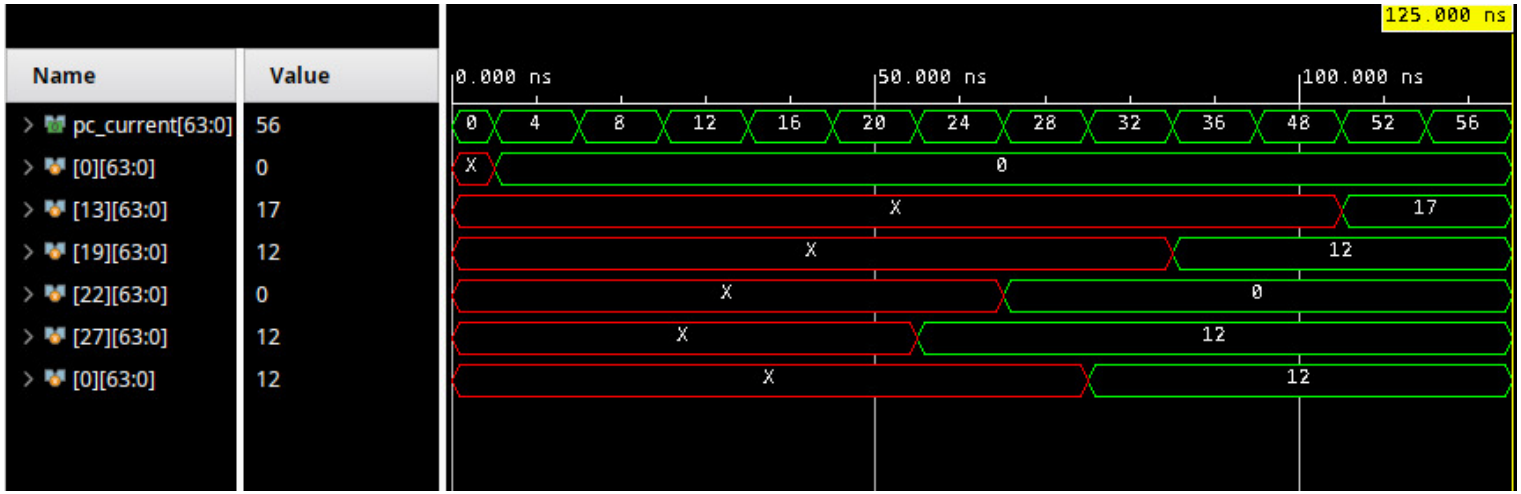

Fig. 2: Vivado Simulation Waveform

Verified Components

- Correct PC updates and control flow
- Instruction execution and decoding
- Register file operations
- ALU operations
- Memory access operations
- Branch behavior

Conclusion

Project Achievements

- Successfully designed and implemented 64-bit single-cycle RISC-V processor
- Comprehensive control unit with truth table
- All instruction categories supported and verified
- Modular Verilog implementation
- Complete simulation verification with Vivado

Educational Value

- Provides clear understanding of processor datapath
- Demonstrates control unit design principles
- Hands-on experience with CPU implementation
- Foundation for advanced processor architectures

References

Thank You

Questions?