|  |  |  |
| --- | --- | --- |
| Qatar University College of Engineering  Depart. of Computer Science & Engineering | Description: QU_Logo_03 | CMPE 263: COA-1  Course Project Spring 2024 |

Course Project

Announcement Date : 10 – March –2024 Submission Deadline : 25 – April – 2024

Team size : Max of four members per team

# Project Title

Design of 7-bit CPU using Logisim by integrating ALU, Registers and ROM.

# Introduction

The goal of this project is to design a programmable 7-bit CPU in Logisim using ALU, Registers, Memory and implement instructions MOV, ADD, SUB, Podd, Peven, LSL & LSR. Use the build-in components like Resisters, Adders, Subtractors, Shifters, Decoders and ROM Memory from Logisim. Use the four 7-bit general purpose registers (e.g., R1, R2, R3, R4).

# Objectives

* Design a 7-bit programmable CPU architecture with 8 instructions.
* 8 bit address and 7 bit data path using 7-bit general-purpose registers (R0 to R4).
* Add ROM from Logisim to the design controlled by 8 bit address decoder.
* An ALU to execute the required instructions.

# Methodology

1. Design a CPU architecture using *Logisim* simulator.
2. Design the instruction set including the opcode and operands for each instruction.
3. Implement the CPU using 7 bit general-purpose registers, ALU and ROM.
4. Connect all registers with ALU for arithmetic and logical operations controlled by ROM. The addresses of ROM are generated by the Program counter.

# Expected Outcome

This project is expected to produce a programmable 7-bit CPU with 8 instructions, 8-bit fixed- length instructions, and 7-bit registers numbered R1 through R4. In addition, an 8-bit addressable memory with a 16-bit width will be developed as part of the project.

* Write sample program for addition of two numbers
* Write sample program for substraction of two numbers
* Write sample program for multiplication (R1 x 5) using Shift Left Operation
* Write sample program for division (R1 / 8) using shift Right Operation
* Write sample program for insertion of Even Parity (Peven) and Odd Parity (Podd) with 7 bits data where the LSB will be Parity bit. *Store the data with Parity in a register for transmission.*

The Program Counter stops once all the instructions are executed. Add a Green LED to indicate program termination.

**Table 1: - Design Requirements**

|  |  |
| --- | --- |
| Specifications | Description |
| Number of instructions | 8 |
| Width of the instruction | 16-bits fixed-length |
| CPU registers | R1 to R4, each of 7-bits (7 data + 1 Parity) |
| Memory Size | 8-bits address × 16-bits width |
| Data Type | 7-bit unsigned integer |

**Table 2:- Basic Instruction Set Examples**

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Operation | Syntax | In RTL |
| Transfer | Register-to-register | MOV Rd, Rs | Rd  Rs |
| Imm-to-register | MOV Rd, imm | Rd  imm |
| Arithmetic | Add | ADD Rd, Rs | Rd  Rd + Rs |
| Subtract | SUB Rd, Rs | Rd  Rd  Rs |
| Parity | Podd  Peven | Podd Rd, 0/1  Peven Rd, 0/1 | 𝑅𝑑 ← Rd[PXXXXXXX]  𝑅𝑑 ← Rd[PXXXXXXX] |
| Shift | Logical Shift Right | LSR Rd, Rs | Rd  LSR (Rs, 1) |
| Logical Shift Left | LSL Rd, Rs | Rd  LSL (Rs, 1) |

Notes:-

* + Imm  Immediate data, e.g. MOV R2, 5 ; in RTL:- R2  5
  + P is Parity bit where Peven is Data Packet having even parity bit.
  + P is Parity bit where Podd is Data Packet having odd parity bit.
  + The hardware can only execute a single bit LSR / LSL operation

# Design Operation

1. Based on the given instructions, write the assembly code.
2. Based on your instruction format, convert the assembly code to machine code
3. Upload the machine code into the memory (ROM)
4. Hit on the RUN button. The CPU should execute the code and saves the result as instructed by the code
5. When program execution is complete, the results will be available in registers.

# Sample of Assembly Code

With the given instructions, we can write the assembly code to multiply the contents of R1 by 5 and save the result in R3 without changing the value of R1.

|  |  |  |
| --- | --- | --- |
| MOV | R1, R0 | ; R1 R0 |
| LSL | R0, R0 | ; R0  2 × R0 |
| LSR | R0, R0 | ; R0  R0 /2 |
| ADD | R0, R1 | ; R0  R0 + R1 |

# Conclusion

This course project will provide a hands-on experience in designing an 7-bit CPU architecture and execute various instructions on ALU. This project will enhance students' understanding of computer architecture and hardware design, preparing them for future careers in computer engineering and related fields.