Faculty of Engineering and Technology

Electrical and Computer Engineering Department

ENCS3310: Advanced Digital Systems Design

Course Project

Summer Course 2023/2024

Dr. Ayman Hroub

Deadline: August 19, 2024 at 23:59

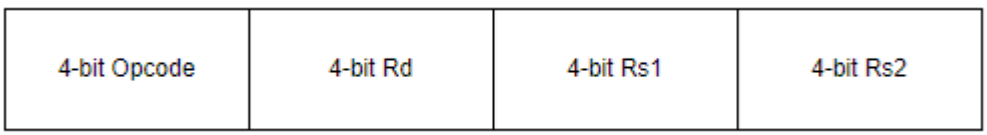
Late submission is penalized at a rate of 10% per day for up to 3 days

Objective:

In this project, you will design and verify a very simple multi-cycle processor in Verilog. A multi-cycle processor is a processor that executes an instruction in multiple clock cycles. Moreover, in such processors, each instruction is given only the number of clock cycles required to execute it.

Architectural and Microarchitectural Specifications:

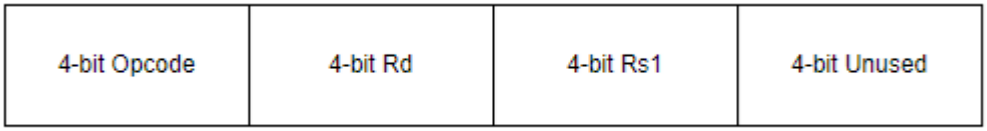
1. The processor has a register file of 16 general-purpose registers. Each register is 16-bit wide. This register file has two read ports, and one write port.
2. The processor has an instruction memory to store the instructions. The width of this instruction memory is 16-bit
3. The processor has a data memory to store the data. The width of this data memory is 16-bit
4. For simplicity, this processor supports only two types of instructions. Each instruction is 16-bit wide.
    a. R-Type ALU instructions with the following format



- **Opcode:** operation code, i.e., it determines which operation the processor should execute

- **Rd:** the number of the destination register, i.e., the register where the result is stored
- **Rs1:** the number of the first source register
- **Rs2:** the number of the second source register

b. M-Type memory instructions with the following format



- If the instruction is **load**: DataMemory [Reg[Rs1]] ➔ Reg [Rd]
- If the instruction is **store**: DataMemory [Reg[Rs1]] ← Reg [Rd]

5. ALU R-Type instructions require fetch, decode, execute, and write-back stages
6. Load instruction require fetch, decode, memory, and write-back stages
7. Store instruction requires fetch, decode, and memory stages.
8. The fetch stage implies reading the instruction from the instruction memory at the address stored in the special purpose 32-bit register called (PC: Program Counter). Then, stores this instruction in a 16-bit register called the instruction register (IR). Finally, it increments the PC.
9. The decode stage generates the control signals of the instruction, and it determines the next stage the instruction should go to
10. The execute stage comprises a simple ALU to perform the operation of the instruction.
11. The memory stage implies reading the data memory in the case of a load instruction, and writing the data memory in the case of a store instruction.
12. The write-back stage implies writing the result on the destination register.
13. The table below lists the supported instructions by this processor with their opcodes. The other opcodes are unused.

Instruction	Meaning	Opcode
ADD Rd, Rs1, Rs2	Reg [Rd] = Reg [Rs1] + Reg [Rs2]	0000
SUB Rd, Rs1, Rs2	Reg [Rd] = Reg [Rs1] - Reg [Rs2]	0001
AND Rd, Rs1, Rs2	Reg [Rd] = Reg [Rs1] & Reg [Rs2]	0010
OR Rd, Rs1, Rs2	Reg [Rd] = Reg [Rs1] | Reg [Rs2]	0011
XOR Rd, Rs1, Rs2	Reg [Rd] = Reg [Rs1] ^ Reg [Rs2]	0100
LOAD Rd, Rs1	DataMemory [Reg[Rs1]] ➔ Reg [Rd]	0101
STORE Rd, Rs1	DataMemory [Reg[Rs1]] ← Reg [Rd]	0110

14. The control signals generated by the control unit in the decode stage include

Instruction	RegWr	ALUOP	MemRd	MemWr
ADD Rd, Rs1, Rs2	1	000	0	0
SUB Rd, Rs1, Rs2	1	001	0	0
AND Rd, Rs1, Rs2	1	010	0	0
OR Rd, Rs1, Rs2	1	011	0	0
XOR Rd, Rs1, Rs2	1	100	0	0
LOAD Rd, Rs1	1	xxx	1	0
STORE Rd, Rs1	0	xxx	0	1

- **RegWr:** Register file write enable
- **ALUOP:** ALU operation code
- **MemRd:** Data memory read enable
- **MemWr:** Data memory write enable

Report:

This project report should be written as **formal report.** The report should include sections on the following:

- Brief introduction and background
- Design philosophy
- Block diagram
- Test cases: the test case implies writing a program using this processor language (using the given instructions), storing this program in the processor's instruction memory, executing the program, and checking whether the execution results are correct.

- Simulation results
- Conclusion and Future works

The report must be submitted as pdf file, and the code (design and TestBench) should be submitted as one zipped file.

Key Points:

- Project grading is done through discussion. Any team does not discuss their project, will get the grade of **zero**.
- Any type of plagiarism or cheating will be penalized by **0** mark, and the cheaters will be treated according to the university laws.
- The design description should include diagrams of the design, and give a justification of the decisions made.
- Technical achievement in design is linked to the degree of functionality that was attempted, as explained below.
- Technical achievement in implementation is based on the quality of your Verilog code. This includes issues such as legibility of code, use of meaningful variable names, good comments, clear structure, and modifiability of the design.
- Technical achievement in evaluation is based on the quality of your simulation results.
- **Hint:** use FSM (Finite State Machine) to implement the control logic.

Assessment Criteria

Report	10%
The way of discussion	10%
Code organization and documentation	10%
Processor Design	20%
Features Implementation (RTL code)	25%
Features Verification (TestBench)	25%