BIRZEIT UNIVERSITY

Department of Electrical and Computer Engineering
Second Semester, 2023/2024

Project No. 2
ENCS4370| Computer Architecture
Deadline: June 10, 2024 at 23:59

---

1. Objectives:

To design and verify a simple pipelined RISC processor in Verilog

2. Processor Specifications

1. The instruction size and the word size is 16 bits
2. 8 16-bit general-purpose registers: from R0 to R7.
3. R0 is hardwired to zero. Any attempt to write it will be discarded.
4. 16-bit special purpose register for the program counter (PC)
5. Four instruction types (R-type, I-type, J-type, and S-type).
6. Separate data and instruction memories
7. Byte addressable memory
8. Little endian byte ordering
9. You need to generate the required signals from the ALU to calculate the condition branch outcome (taken/ not taken). These signals might include zero, carry, overflow, etc.

3. Instruction Types and Formats

As mentioned above, this Instruction Set Architecture (ISA) has four instruction types, namely, R-type, I-type, J-type, and S-type. These four types have a common **opcode** field, which determines the specific operation of the instruction

R-Type (Register Type)

Opcode[4]	Rd[3]	Rs1[3]	Rs2[3]	Unused[3]

- **3-bit Rd:** destination register
- **3-bit Rs1:** first source register
- **3-bit Rs2:** second source register
- **3-bit unused**

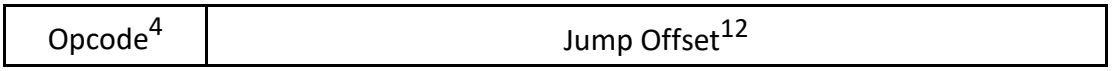
I-Type (Immediate Type)

- **3-bit Rd:** destination register
- **3-bit Rs1:** first source register
- **5-bit immediate:** unsigned for logic instructions, and signed otherwise.
- **1-bit mode:** this is used with load and branch instructions, such that:
  For the load:
  0: LBs load byte with zero extension
  1: LBu load byte with sign extension
  For the branch:
  0: compare Rd with Rs1
  1: compare Rd with R0

J-Type (Jump Type)
This type includes the following instruction formats. The opcode is used to distinguish each instruction

```
jmp    L  # Unconditional jump to the target L.
call   F  # Call the function F.  F is a label.
          # The return address is pushed on r15.
```

The target address is calculated by concatenating the most significant 7-bit of the current PC with the 12-bit offset after multiplying offset by 2.




```
ret    # return from a function.
       # the next PC will be the value stored in r7
```

S-Type (Store)

This format support only one instruction as flowing

Sv rs, imm # M[rs] = imm

4. Instructions' Encoding

For simplicity, you are required to implement a subset only of this processor's ISA. The table below shows the different instructions you are required to implement. It shows their type, the opcode value, and their meaning in RTN (Register Transfer Notation). Although the instruction set is reduced, it is still rich enough to write useful programs.

No.	Instr	Format	Meaning	Opcode Value	m
1	AND	R-Type	Reg(Rd) = Reg(Rs1) & Reg(Rs2)	0000	
2	ADD	R-Type	Reg(Rd) = Reg(Rs1) + Reg(Rs2)	0001	
3	SUB	R-Type	Reg(Rd) = Reg(Rs1) - Reg(Rs2)	0010	
4	ADDI	I-Type	Reg(Rd) = Reg(Rs1) + Imm	0011	
5	ANDI	I-Type	Reg(Rd) = Reg(Rs1) + Imm	0100	
6	LW	I-Type	Reg(Rd) = Mem(Reg(Rs1) + Imm)	0101	
7	LBu	I-Type	Reg(Rd) = Mem(Reg(Rs1) + Imm)	0110	0
8	LBs	I-Type	Reg(Rd) = Mem(Reg(Rs1) + Imm)	0110	1
9	SW	I-Type	Mem(Reg(Rs1) + Imm) = Reg(Rd)	0111	
10	BGT	I-Type	if (Reg(Rd) > Reg(Rs1)) Next PC = PC + sign_extended (Imm) else PC = PC + 2	1000	0
11	BGTZ	I-Type	if (Reg(Rd) > Reg(0)) Next PC = PC + sign_extended (Imm) else PC = PC + 2	1000	1
12	BLT	I-Type	if (Reg(Rd) < Reg(Rs1)) Next PC = PC + sign_extended (Imm) else PC = PC + 2	1001	0
13	BLTZ	I-Type	if (Reg(Rd) < Reg(R0)) Next PC = PC + sign_extended (Imm) else PC = PC + 2	1001	1
14	BEQ	I-Type	if (Reg(Rd) == Reg(Rs1)) Next PC = PC + sign_extended (Imm) else PC = PC + 2	1010	0
15	BEQZ	I-Type	if (Reg(Rd) == Reg(R0)) Next PC = PC + sign_extended (Imm) else PC = PC + 2	1010	1
16	BNE	I-Type	if (Reg(Rd) != Reg(Rs1)) Next PC = PC + sign_extended (Imm)	1011	0

			else PC = PC + 2		
17	BNEZ	I-Type	if (Reg(Rd) != Reg(Rs1)) Next PC = PC + sign_extended (Imm) else PC = PC + 2	1011	1
18	JMP	J-Type	Next PC = {PC[15:10], Immediate}	1100	
19	CALL	J-Type	Next PC = {PC[15:10], Immediate} PC + 4 is saved on r15	1101	
20	RET	J-Type	Next PC = r7	1110	
21	Sv	S-Type	M[rs] = imm	1111	

5. RTL Design

You are required to design a **5-stage pipelined** processor (fetch, decode, ALU, memory access, and write back). The design should include a datapath and control path that support all of the aforementioned instructions.

6. Verification

To verify the RTL design, write a testbench around it. Moreover, you need to write multiple code sequences in the given ISA and show how the processor you designed executes these code sequences.

7. Project Report

The report document must contain sections highlighting the following:

1 – Design and Implementation

1. Detailed description of the datapath, its components, and the assembly of these components.
2. A complete description of the control signals, description, truth table, and Boolean equations
3. The implementation details, and the design choices you made with justification
4. Datapath and control path block diagrams.
5. A list of sources for any parts of your design that are not entirely yours (if any).
6. Carry out the design and implementation with the following aspects in mind:
   - Correctness of the individual components
   - Correctness of the overall design when connecting these components together
   - Completeness: all instructions were implemented properly.

2 – Simulation and Testing

1. Carry out the simulation of the processor developed
2. Describe the test programs that you used to test your design with enough comments describing the program, its inputs, and its expected output. List all the instructions that were tested and work correctly. List all the instructions that do not run properly.
3. Also, provide snapshots of the simulator window with your test program loaded and showing the

simulation output results.

3 – Teamwork

1. Work in groups of up to three students.
2. Group members are required to coordinate the work equally among themselves so that everyone is involved in all the following activities:
   - Design and implementation
   - Simulation and testing
   - Report writing
   - Project demonstration and discussion
3. Clearly show the work done by each group member.

8. Submission Guidelines

Attach one zip file containing all the design circuits, the programs source code and binary instruction files that you have used to test your design, their test data, as well as the report document.

9. Grading Criteria

Regarding the grade, this project weight is 15 points.

- If you implement a pipelined processor, the project will be graded out of 18, i.e., you will get a bonus of 5 points
- If you implement a multi-cycle processor, the project will be graded out of 15, i.e., you will get a full mark
- If you implement a single-cycle processor, the project will be graded out of 12, i.e., you will lose 3 points

Note: each of the following items is a prerequisite to the next one, e.g., we cannot consider the TL code if there is no design, and we cannot consider the verification part if there is no RTL code.

Item	**Weight**
Control signals generation (truth tables and Boolean equations)	10
Processor Microarchitecture Design (complete datapath and control path) that supports all instructions	20
Complete **modular** RTL design of the above microarchitecture that supports all instructions	20
Verification environment (testbench) around the RTL design such that you can write code sequences in the ISA, store them in the instruction memory, execute them and show results using waveform diagrams.	20
Code organization and documentation	5
Detailed report that includes control signals truth tables, Boolean equations, detailed and clear microarchitecture design schematics, test cases, simulation screenshots, etc.	15
Discussion	10
Total	**100**