Assignment: ALU Design for the SimpleRISC Core

Background

You are provided with the Verilog RTL of a single-cycle SimpleRISC processor core. The current version of the processor uses a placeholder ALU that is not optimized for performance. Your task is to design and implement a high-performance ALU and integrate it into the existing core.

Objectives

• Design a synthesizable Arithmetic Logic Unit (ALU) supporting all arithmetic and logical operations defined in the SimpleRISC ISA.

• Integrate the ALU into the given SimpleRISC RTL core by replacing the existing placeholder ALU module.

• Meet a target operating frequency of 250 MHz under typical synthesis constraints.

• Verify the correctness of your ALU by running the provided assembler program (program.asm) and testbench.

ALU Specification

The ALU must support the following operations (selected by a 4-bit alu_op control signal):

Opcode	Operation	Description
0000	ADD	Signed/unsigned addition
0001	SUB	Signed/unsigned subtraction
0010	AND	Bitwise AND
0011	OR	Bitwise OR
0100	XOR	Bitwise XOR
0101	SLT	Set to 1 if A < B (signed), else 0
0110	SLL	Logical shift left
0111	SRL	Logical shift right
1000	SRA	Arithmetic shift right
1001	PASS	Pass operand B
1010	NOT	Bitwise NOT of operand B
1011	MUL	Signed multiplication
1100	DIV	Signed division
1101	MOD	Signed modulus

Inputs: a[31:0], b[31:0] (operands), op[3:0] (control signal).

Outputs: y[31:0] (result), zero (flag = 1 if result == 0).

Design Requirements

• Synthesis Performance: The ALU must be designed to meet a 250 MHz clock frequency target.

• Area and Power Trade-offs: Solutions should balance logic utilization and performance.

• Integration: Replace the placeholder ALU in the given SimpleRISC RTL (alu.v).

• Verification: Use the provided assembler and testbench; add at least 5 custom test cases.

Deliverables

• ALU Design Report (4–6 pages) with design description, algorithms, synthesis results, and trade-offs.

• Verilog Code: alu.v (optimized ALU) and any helper modules.

• Simulation Proof: waveforms and register dumps.

• Test Cases: program.asm + program.hex for at least 5 new tests.

Evaluation Criteria

Component	Weight
Correct functional behavior	30%
Achieving 250 MHz target	30%
Code quality & readability	15%
Design report & analysis	15%
Custom test cases & proof	10%