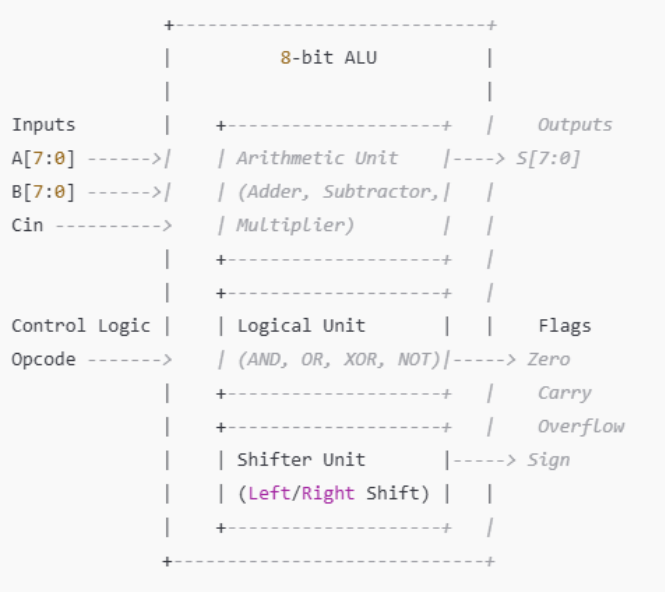
Due Date: 10.01.2025

ELEC 451/540 VLSI Design Project: Design and Layout of an 8-Bit ALU Unit Using Skywater 130nm Technology

1. Objective

The project involves the design, simulation, and physical layout of an **8-bit Arithmetic Logic Unit (ALU)** using Skywater 130nm technology. The ALU will perform a variety of arithmetic and logical operations, commonly used in processors. Students will work collaboratively to divide the circuit into sub-blocks, design each block, and integrate them into the final ALU. You are allowed to use your own IP that you designed at the previous homework. It is highly suggested to write an HDL to obtain the RTL schematic.

2. Block Diagram



The 8-bit ALU consists of the following sub-blocks:

2.1 Arithmetic Unit:

- Adder
- Subtractor
- Multiplier

2.2 Logical Unit:

- AND Gate
- OR Gate
- XOR Gate

- NOT Operation

2.3 Shifter Unit:
- Left Shift
- Right Shift

2.4 Control Logic:
- Operation Selector (Based on opcode)

2.5 Output Unit:
Output, S [7:0]

Flags (Zero, Carry, Overflow, Sign)

3. Sub-Block Descriptions

3.1 Arithmetic Unit

- **Adder**:
  - An 8-bit **Ripple Carry Adder (RCA)** is used for simplicity. It computes the sum of two 8-bit operands.
  - Inputs: A[7:0],B[7:0], Cin
  - Output: S[7:0],Cout
  - Implementation: Full adder chains connected serially.
- **Subtractor**:
  - Utilizes the adder by taking the 2's complement of the second operand.
  - Inputs: A[7:0], B[7:0]
  - Output: D[7:0]
  - 2's Complement: Invert B[7:0] and add 1 using the carry-in.
- **Multiplier**:
  - Implements an **array multiplier** for 8-bit operands.
  - Partial products are computed and added hierarchically.
  - Inputs: A[7:4],B[7:4]
  - Output: M[8:0]

3.2 Logical Unit

- **AND/OR/XOR Operations**:
    - Each logical operation is implemented using gate-level combinational circuits.
    - Inputs: A[7:0],B[7:0]
    - Outputs: Logical[7:0] (AND, OR, XOR based on control logic).
- **NOT Operation**:
    - Performs bitwise negation on the operand A[7:0].
    - Output: NotA[7:0].

3.3 Shifter Unit

- **Left Shift**:
    - Shifts the bits of A[7:0] to the left by 1 or more positions.
    - Input: A[7:0]
    - Output: ShiftL[7:0].
- **Right Shift**:
    - Shifts the bits of A[7:0] to the right.
    - Can be implemented as arithmetic (sign-extended) or logical (zero-filled).
    - Output: ShiftR[7:0].

3.4 Control Logic

- The control logic decodes a 4-bit **opcode** to select the operation performed by the ALU.
    - Example Opcode Mapping:
        - 0000: Add
        - 0001: Subtract
        - 0010: Multiply
        - 0011: AND
        - 0100: OR
        - 0101: XOR
        - 0110: NOT

- 0111: Left Shift
- 1000: Right Shift

3.5 Output Unit

- **Output**
  - **S[7:0]**
- **Flags**:
  - **Zero (Z)**: Set if the output of an operation is all zeros.
  - **Carry (C)**: Set if there's a carry-out from the most significant bit.
  - **Overflow (V)**: Set if signed overflow occurs.
  - **Sign (S)**: Set if the result is negative in signed arithmetic.

4. Design Approach

4.1 Circuit Design:

- Design each sub-block using behavioral Verilog.
- Simulate functionality using a simulator (e.g., NGSPICE or Xyce).

4.2 Schematic and Layout:

- Create a transistor-level schematic for each block.
- Generate physical layouts ensuring adherence to **Design Rules (DRC)** and verifying with **Layout vs. Schematic (LVS)** checks.

5. Integration:

- Combine all sub-blocks into a hierarchical ALU design.
- Optimize area, power, and performance.

6.Deliverables

1. Verilog/Spice code for all sub-blocks.
2. Simulated waveforms demonstrating ALU functionality.
3. Transistor-level schematics of each sub-block.
4. Complete physical layout of the 8-bit ALU.
5. Report detailing:
   - Design process.

- Challenges and solutions.
- Performance metrics (area, delay, power).

Evaluation Criteria

1. **Correctness**: Proper functionality of each sub-block and the integrated ALU.
2. **Optimization**: Efficient use of area and power in the layout.
3. **Team Collaboration**: Clear division of tasks and integration.
4. **Documentation**: Completeness and clarity of the final report.

This project provides a challenging, collaborative learning experience with a mix of arithmetic, logical, and layout design tasks.