CSE 460
VLSI Design Laboratory
Brac University

Project:

Design a 4-bit ALU capable of performing 4 different arithmetic or logical operations using Quartus, implement it using Verilog HDL, and verify it using a timing diagram.

Specifications:
The ALU takes two **four-bit inputs**: **A**, **B,** and a **three-bit operation code** (opcode). Based on the opcode, it performs **five** different operations on A and B and produces a **four-bit output**, **C**. Depending on the result of a particular operation, the ALU also produces **three flags**: carry, zero and sign flag. Fig. 1 illustrates the block diagram of the ALU.

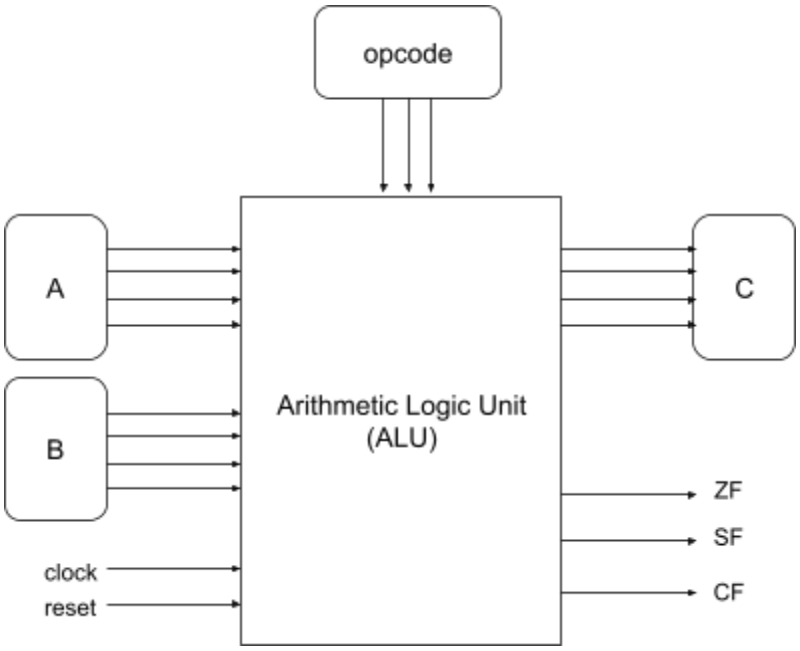

Fig. 1: Block diagram of the ALU.

The following table shows the details of the flags followed by an operation.

Flags	Description
Zero Flag (ZF)	ZF is set to 1 when the output C is 0.
Sign Flag (SF)	SF is set to 1 when the MSB of the output C is 1.
Carry Flag (CF)	CF is set to 1 when output carry/borrow is 1.

Example:

Suppose we need to design an ALU with three opcodes performing **three** operations (Op = 00, 01, 10 for RESET, AND, OR operation respectively) on **A**, **B**. We are skipping the flag outputs for this example which you need to include in your project. The state diagram of the system is illustrated in the following figure. As depicted, the system should be in an **Idle state** initially. Depending on the opcode, it will move to the definite state and eventually perform the desired operation. The opcode should be kept unchanged during an operation. After the required clock cycles to perform a particular operation, the opcode will be set to 00 manually from the input (timing diagram) to reset the system.

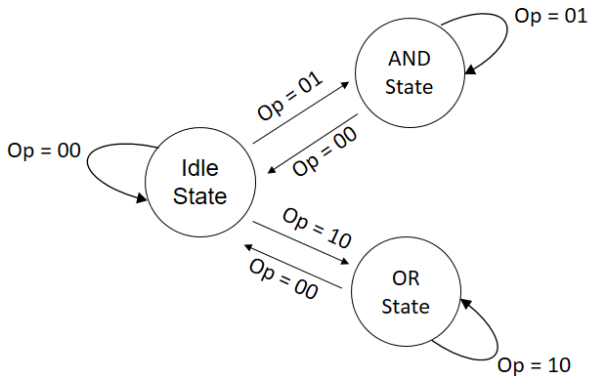

Fig. 2: Simplified state diagram of an ALU with three operations: RESET, AND, OR.

The following figure describes the output waveform of the above-described system for AND operation.

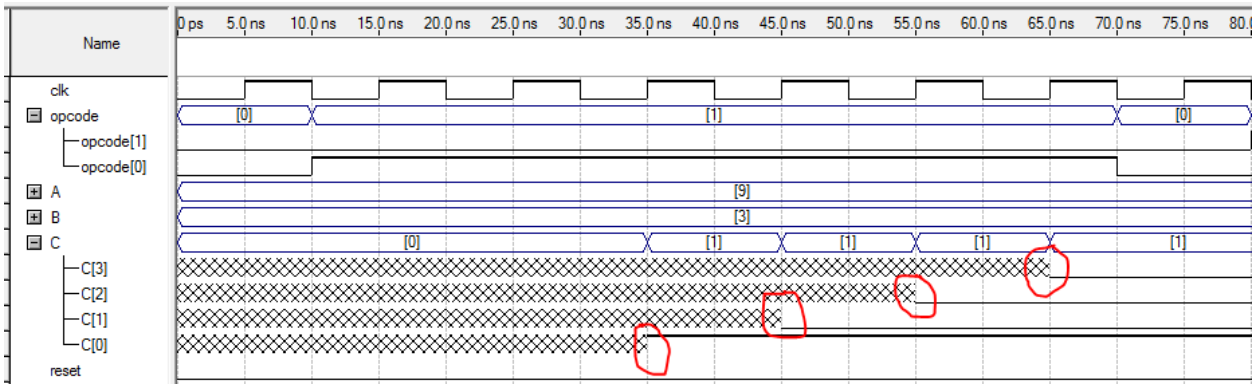

Fig. 3: Timing diagram of an ALU for **AND** operation.

As depicted, when **opcode = 01** from 10ns to 70 ns, the ALU performs bitwise **AND** operation between contents A (1001) and B (0011). At first, C[0] is assigned with the value of A[0] & B[0] at 35ns. Then, C[1] is assigned with A[1] & B[1] at the next positive edge of the clk (at 45ns) and so on. At 75ns, the Op=00, and thus the ALU is in idle state holding the previous operation's result in C.

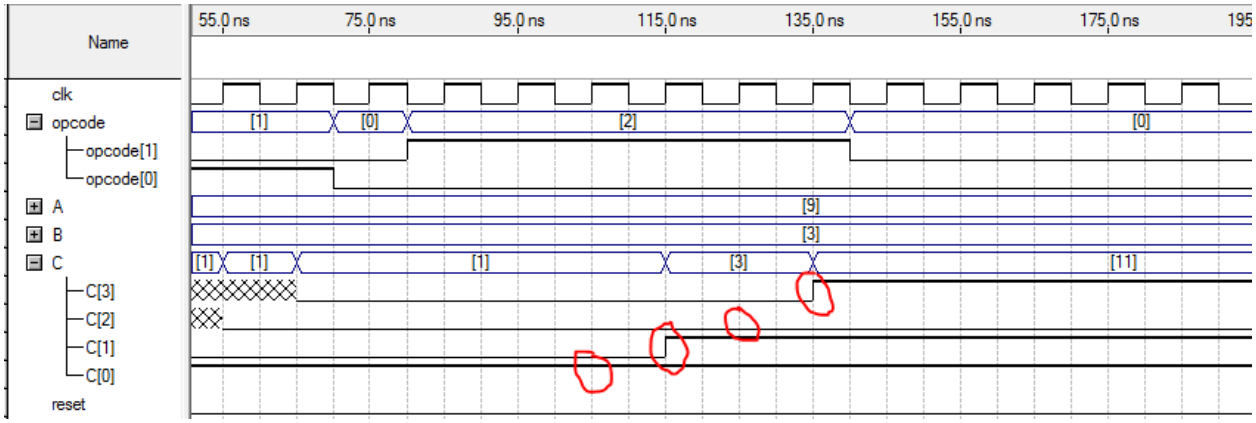

Fig. 4: Timing diagram of an ALU for **OR** operation.

After AND operation, there must be a **mandatory reset** from opcode input (**Op=00**) at 75ns. From, 80ns to 140ns, **Op=10**, and thus, the **serial OR** operation is performed. The operation will overwrite the previous output value of C bit by bit (from LSB to MSB). Accordingly, C is updated with **1011** resulting from the OR operation of A (**1001**), B (**0011**) and C holds this value until further opcode comes in.

Opcode distribution:

Group 6

opcode	operation	description
000	RESET	The ALU will be at the idle state (no operation state) and the output C will retain the result of the last operation performed.
001	XOR	The ALU will perform bitwise XOR operation on A, B. The bitwise operation will be performed serially at the positive edge of input clock i.e. during the first clock cycle, the operation will be performed on the LSBs (A_0, B_0) storing the result in C_0 and in the next clock cycle the operation will be performed on A_1, B_1 updating C_1 and finally, on A_3 and B_3 saving it to C_3 as long as the opcode is active.
010	ADD	The ALU will perform ADD operation on A, B. The operation will be performed serially at the positive edge of input clock i.e. during the first clock cycle, the operation will be performed on the LSBs (A_0, B_0) storing the result in C_0 and in the next clock cycle the operation will be performed on A_1, B_1 updating C_1 and finally, on A_3 and B_3 saving it to C_3 as long as the opcode is active.
011	AND	The ALU will perform bitwise AND operation on A, B. The bitwise operation will be performed serially at the positive edge of input clock i.e. during the first clock cycle, the operation will be performed on the LSBs (A_0, B_0) storing the result in C_0 and in the next clock cycle the operation will be performed on A_1, B_1 updating C_1 and finally, on A_3 and B_3 saving it to C_3 as long as the opcode is active.
100	SUB	The ALU will SUBTRACT B from A.. The operation will be performed serially at the positive edge of input clock i.e. during the first clock cycle, the operation will be performed on the LSBs (A_0, B_0) storing the result in C_0 and in the next clock cycle the operation will be performed on A_1, B_1 updating C_1 and finally, on A_3 and B_3 saving it to C_3 as long as the opcode is active. During SUB operation, the MSBs of both A and B should be treated as sign bits i.e. A_3 will be the sign bit for A and correspondingly B_3 will be the sign bit for B. So, the subtraction will occur between two signed binary numbers where the MSBs are the sign bits and the remaining bits are the values of the operands.

Report: Report must be in IEEE format and should contain the following:

- Abstract
- Introduction
- Operation (State diagram, timing diagram explanations of all the instructions).
- Conclusion
- Appendix (Verilog Code)

The report should be three pages long at most except Appendix.

Demonstration: Full working project presentation in Verilog simulation and verification by timing diagram. Bring the hard copy of the group report on the day of the demonstration.

Deadline: Last week of Fall'22 (Dec 24 ~ Dec 29)