BANGLADESH UNIVERSITY OF ENGINEERING & TECHNOLOGY

DHAKA, BANGLADESH

Course No. CSE 306

Course Title: Computer Architecture Sessional

Experiment No. 01

Name of the Experiment: Design and Implementation of a 4-bit ALU (Arithmetic and Logical Unit)

Date of Submission: 19/12/2022

Lab Section: A2

Lab Group: 04

Group Members: 1905035

1905043

1905045

1905050

1905060

Introduction:

An arithmetic and logical unit is a digital circuit used in computers to perform arithmetic and logical operations. It is a combinational logic circuit, which can perform a selected amount of arithmetic and logical operation. If there are k control bits, the ALU can perform at most 2^k operations. An n-bit ALU has n-bit inputs and n-bit outputs.

For the 4-bit ALU implementation, we were given 3 selection bits. To determine which operation to perform, the selection bits were input in a decoder, and the received 8 output lines were used for these operations. As we were given 6 ALU operation to simulate, three decoder output lines were not used.

To implement the circuit, we used 4 4x1 MUX in Arithmetic side and 4 2x1 MUX in logical side. Further 4 more 2x1 MUX were used to distinguish between the Arithmetic and logical output line.

The MUX used in logical operation were 2x1 MUX.

Finally, 4 more 2x1 MUX was used to choose between the output lines of Arithmetic and Logical operation.

Also, four flags were implemented for both logical and arithmetic operations, namely Carry (C), Sign (S), Overflow (V), and Zero (Z) flags. In case of logical operations, C and V were required to be cleared, which was also handled as needed in the circuit.

Problem Specification:

Design a 4-bit ALU with three selection bits CS_0, CS_1 and CS_2 for performing the following operations:

Control Signals			Functions
CS_2	CS_1	CS_0	
0	0	0	AND
0	0	1	Sub
0	1	X	Decrement A
1	0	0	Complement A

1	0	1	XOR
1	1	X	Add

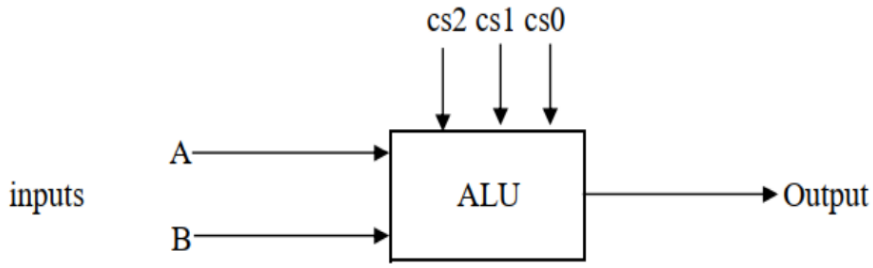

Figure 01: Simplified Block diagram of a 4-bit ALU

Truth Table:

Table 1: Control bit to Decoder

CS_2	CS_1	CS_0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0

Table 2: Arithmetic 4x1 MUX

D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	Y_i	S_1	S_0	C_{in}
0	1	1	1	1	1	1	1	X	X	X	X
1	0	1	1	1	1	1	1	B_i'	0	0	1
1	1	0	1	1	1	1	1	1	0	1	0
1	1	1	0	1	1	1	1	1	0	1	0
1	1	1	1	0	1	1	1	X	X	X	X
1	1	1	1	1	0	1	1	X	X	X	X
1	1	1	1	1	1	0	1	B_i	1	1	0
1	1	1	1	1	1	1	0	B_i	1	1	0

$$S_0 = D_1$$

$$S_1 = D_6 \oplus D_7$$

$$C_{in} = D_1{}'$$

Table 3: Logical 2x1 MUX

D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	Y_i	S
0	1	1	1	1	1	1	1	X	0
1	0	1	1	1	1	1	1	X	X
1	1	0	1	1	1	1	1	X	X
1	1	1	0	1	1	1	1	X	X
1	1	1	1	0	1	1	1	1	1
1	1	1	1	1	0	1	1	B_i	1
1	1	1	1	1	1	0	1	X	X
1	1	1	1	1	1	1	0	X	X

$$Y_i = D_5 + B_i$$

$$S = D_0$$

Table 4: Combined Output

D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	S
0	1	1	1	1	1	1	1	0
1	0	1	1	1	1	1	1	1
1	1	0	1	1	1	1	1	1
1	1	1	0	1	1	1	1	1
1	1	1	1	0	1	1	1	0
1	1	1	1	1	0	1	1	0
1	1	1	1	1	1	0	1	1
1	1	1	1	1	1	1	0	1

$$S = D_0 D_4 D_5$$

Necessary K-maps:

Arithmetic 4x1 MUX:

$$S_0 = D_1$$

$$S_1 = D_6 \oplus D_7$$

$$C_{in} = D_1{}'$$

Logical 2x1 MUX:

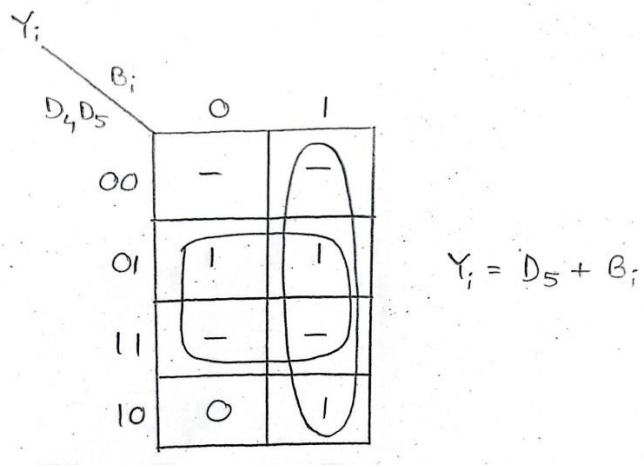

Figure 02: K-Map for selection between complement and XOR

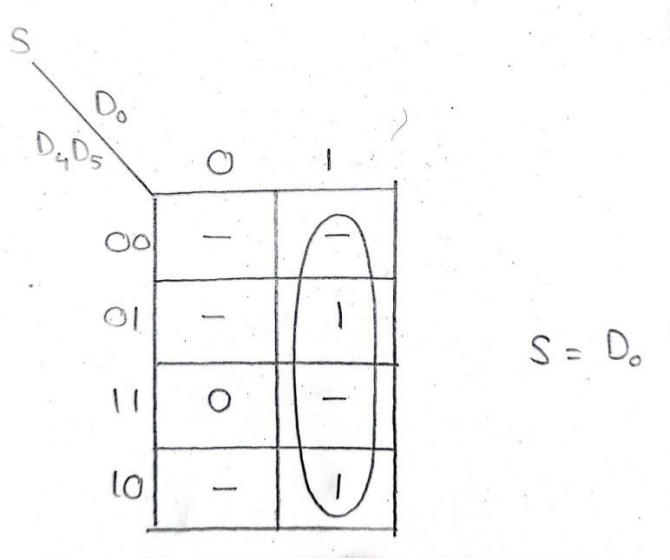

Figure 03: K-Map for selection bit of the 2x1 MUX

Block Diagram:

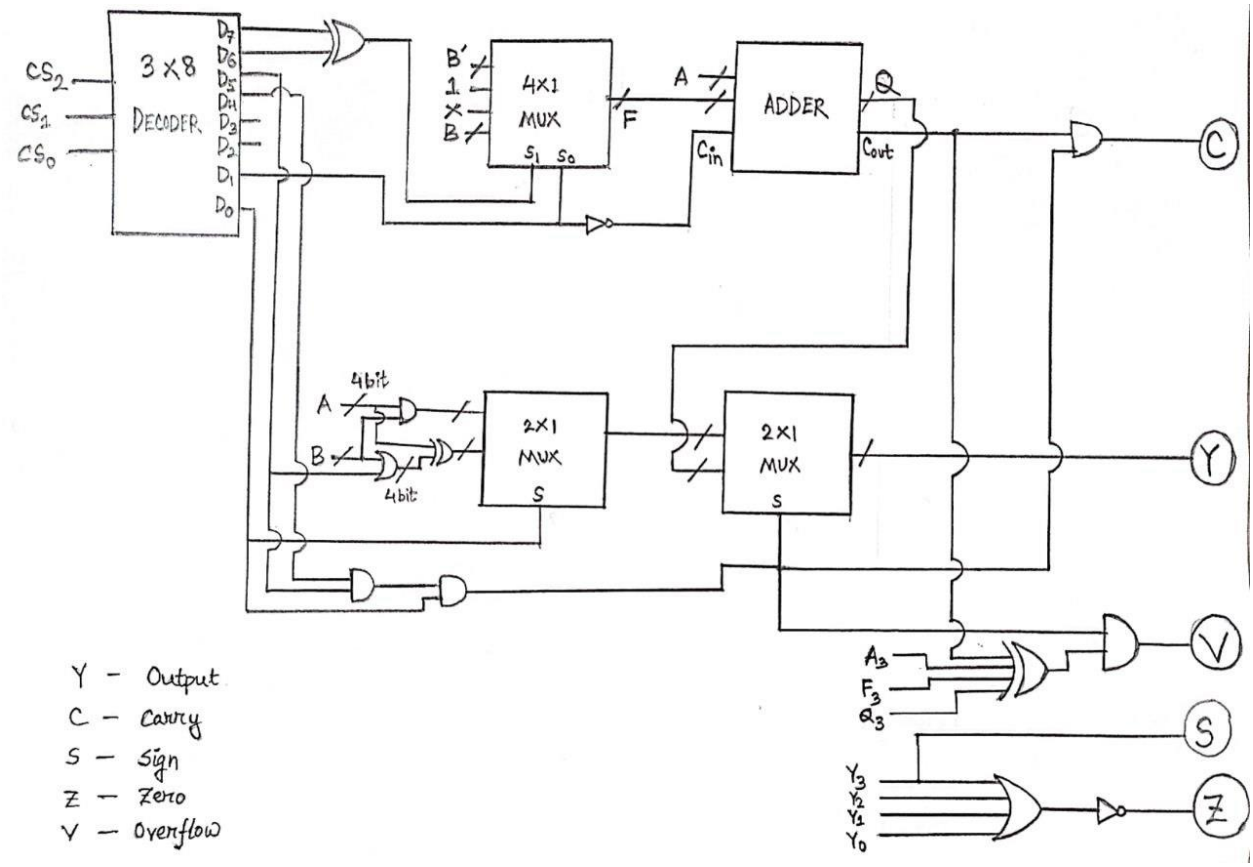

Figure 02: Detailed Block Diagram

Complete Circuit Diagram:

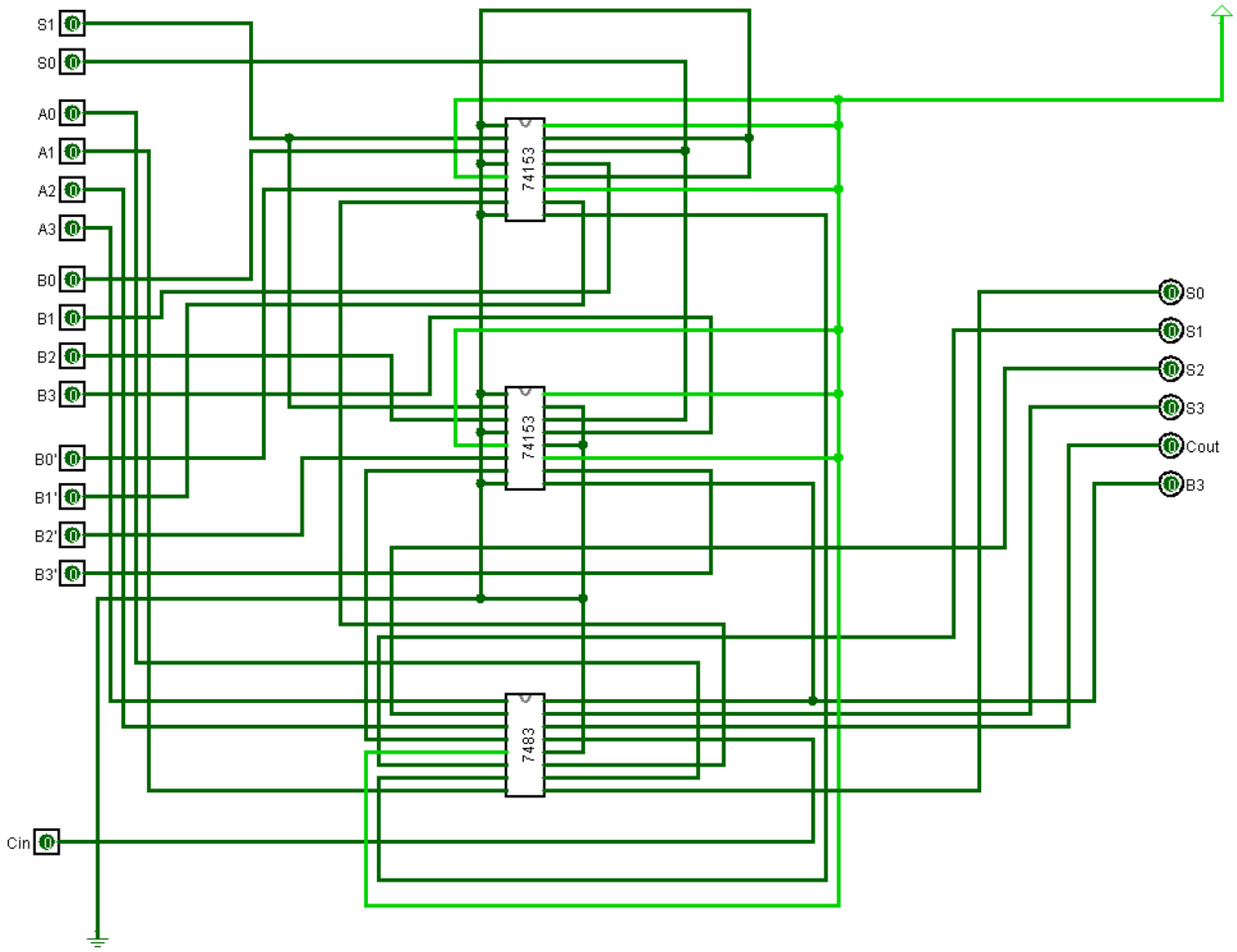

Figure 03: Arithmetic Unit

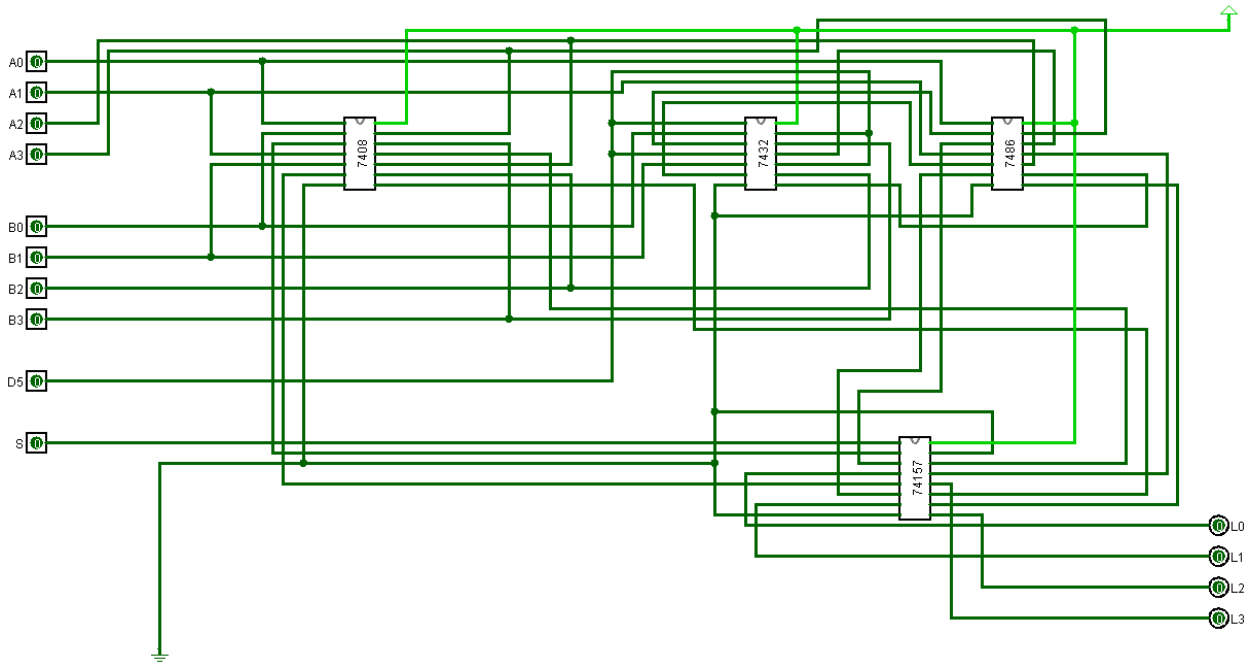

Figure 04: Logical Unit

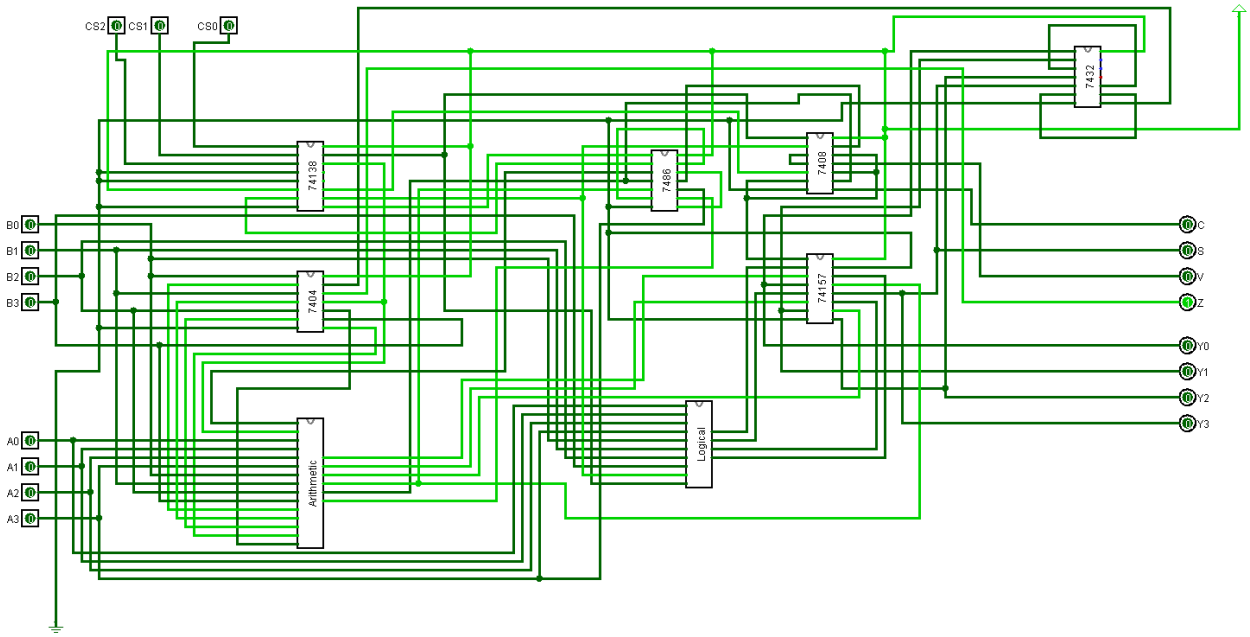

Figure 05: Main unit

Required ICs:

IC Name	Count
IC 74138 (3 by 8 decoder)	1
IC 7483 (4-bit full adder)	1
IC 74157 (2 by 1 MUX)	2
IC 74153 (4 by 1 MUX)	2
IC 7408 (AND)	2
IC 7486 (XOR)	2
IC 7432 (OR)	2
IC 7404 (NOT)	1

Simulator Used: Logisim version 2.7.1

Discussion:

- The purpose of the 4 4x1 MUX in arithmetic is to select one from the second input (For Add operation) or it's complement (For Sub operation) or 1 (For Decrement operation), which will be passed to the 4-bit parallel adder. All the 4x1 MUX has one garbage input line, due to the control bit pattern.

- To optimize the circuit, we adopted a way to combine the complement and XOR operation. We know, to get the complement of a bit, we XOR the bit with 1. The control bits for XOR were 1 0 1. We get 101 from the decoder through the D_5 line. So, when the control bits are 1 0 1, D_5 will be 1. As the 74LS138 IC is active low, the output will be 1, and the second input to 74LS86 IC will be B_i. Otherwise, 1. So, taking the OR of B_i with D_5, then passing the output to the 74LS86 IC will yield XOR or complement respectively. AND operation was done normally.

- The carry (C) flag and overflow (V) flag were needed to be cleared in case of any logical operations. This was taken care of by doing AND with the selection bit from the arithmetic/logical 2x1 MUX to make the flags 0 when any logical operation is chosen.