Lab 3: ALU (Arithmetic Logic Unit)

By

Nisarg Patel (**300277261**)

Sailendhar Vippu (**300299480**)

CEG2136 Section B03 (B4O2)

Group 30

Facilitating TA : Surbhi

Course Professor: V.Groza

Date of Experiment: October, 17th 2023

Date of Submission: November, 3rd 2023

Department of Computer Engineering

University of Ottawa

Lab Objectives:

- To initiate the students who are not familiar with the Altera Quartus II Design Software and the Altera FPGA based DE2-115 platform.
- To design, build and test an ALU (Arithmetic Logic Unit).
- To understand the operation of control signals and switching between micro operations.
- To compile, simulate, debug, and test their design.

Required Equipment:

- Quartus II Software
- Altera DE2-115 Board
- USB Blaster Cable

Preface:

Within this lab, we are tasked with designing, simulating and also building an ALU (Arithmetic Logic Unit). This ALU will have to execute multiple operations (16), the results will be given back in the format of 4-bits, which are Overflow, Sign, Zero and Carry.

Circuit	Control Word S3 S2 S1 S0	ALU Data Output	Micro-operation Description
Arithmetic circuit (AC)	0 0 0 0	$C \leftarrow A + B$	Addition
	0 0 0 1	$C \leftarrow A + B + 1$	Add with carry
	0 0 1 0	$C \leftarrow A$	Transfer A
	0 0 1 1	$C \leftarrow A + 1$	Increment A
	0 1 0 0	$C \leftarrow A + \overline{B}$	Subtraction A - B with borrow (using 1's complement of B gives a result lower by 1 than a conventional subtraction).
	0 1 0 1	$C \leftarrow A + \overline{B} + 1$	Subtraction A - B (use 2's complement of B)
	0 1 1 0	$C \leftarrow \overline{A}$	NOT A (complement A)
	0 1 1 1	$C \leftarrow \overline{A} + 1$	2's complement of A
Logic and shift circuit (LSC)	1 0 0 0	$C \leftarrow$"0000"	Reset C
	1 0 0 1	$C \leftarrow$"1111"	Set C
	1 0 1 0	$C \leftarrow A \wedge B$	A AND B
	1 0 1 1	$C \leftarrow A \vee B$	A OR B
	1 1 0 0	$C \leftarrow A \oplus B$	A EXCLUSIVE-OR B
	1 1 0 1	$C \leftarrow A \wedge \overline{B}$	Reset A bits selected by "mask" B
	1 1 1 0	$C \leftarrow$ ashl A	Shift A left (signed multiplication by 2)
	1 1 1 1	$C \leftarrow$ ashr A	Shift A right (signed division by 2)

Figure PF - Alu Micro-Ops

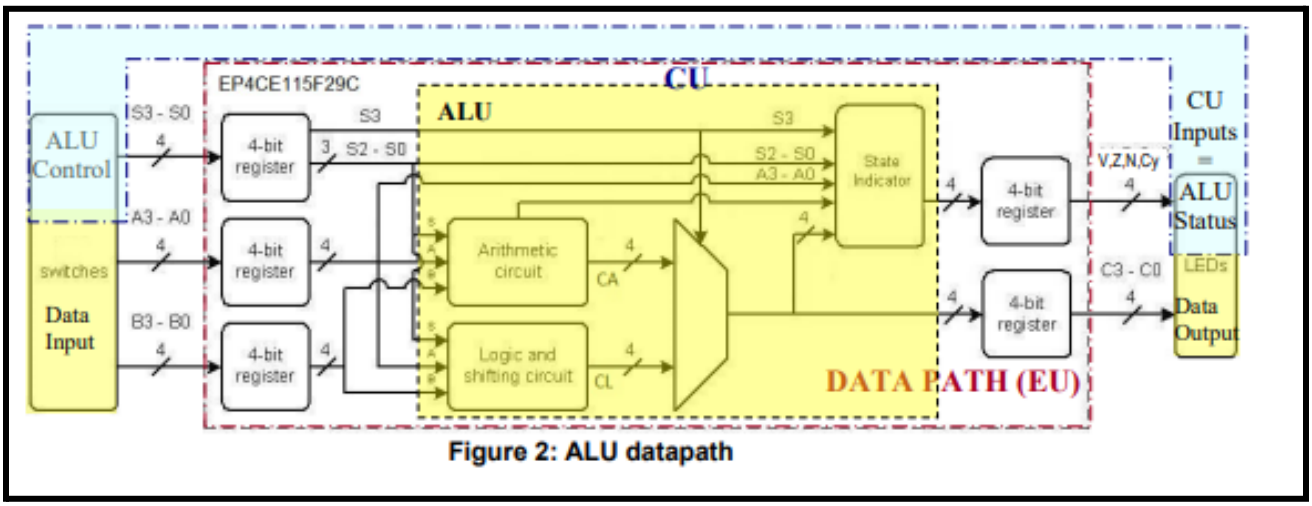
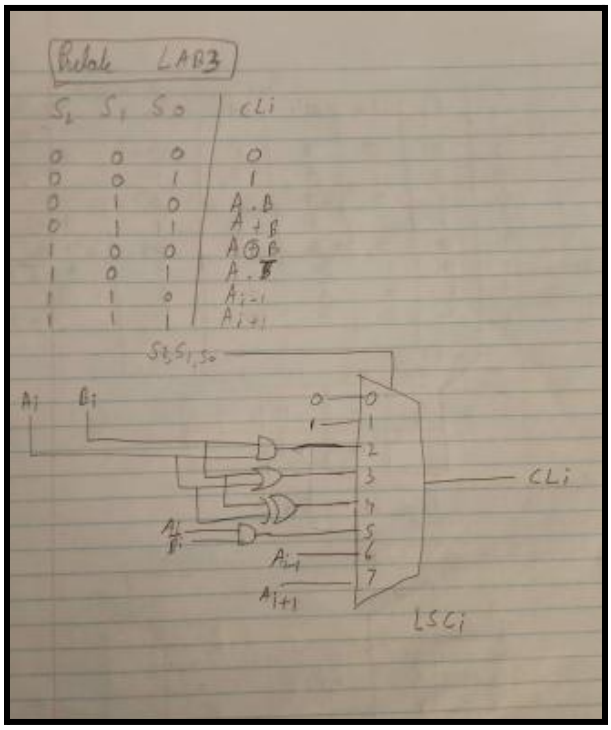


Figure 2: ALU datapath

ALU datapath

Prelab 5.1-5.4

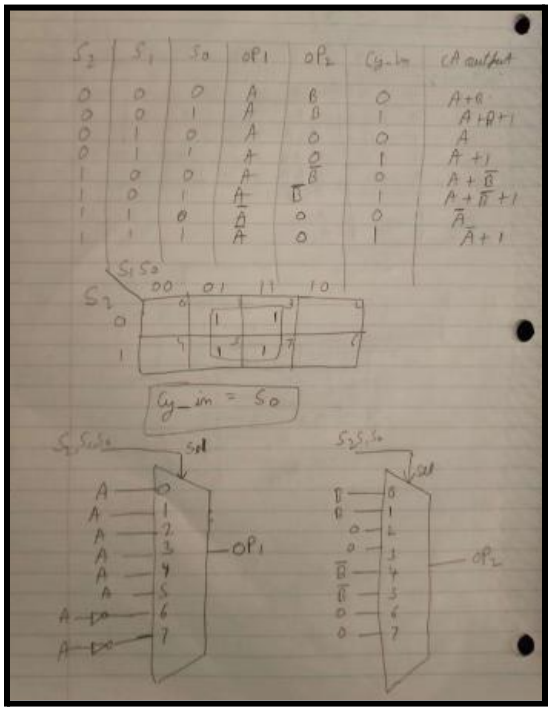
5.3: Design of logic circuit(LSC)

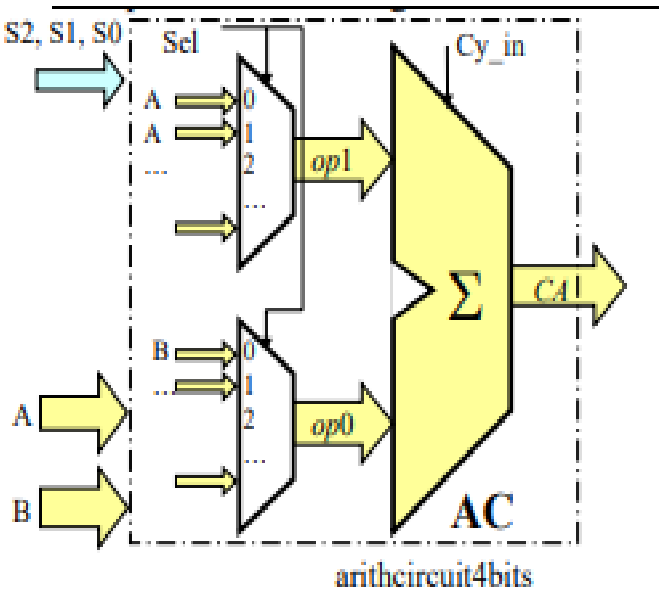


Equations for LSC

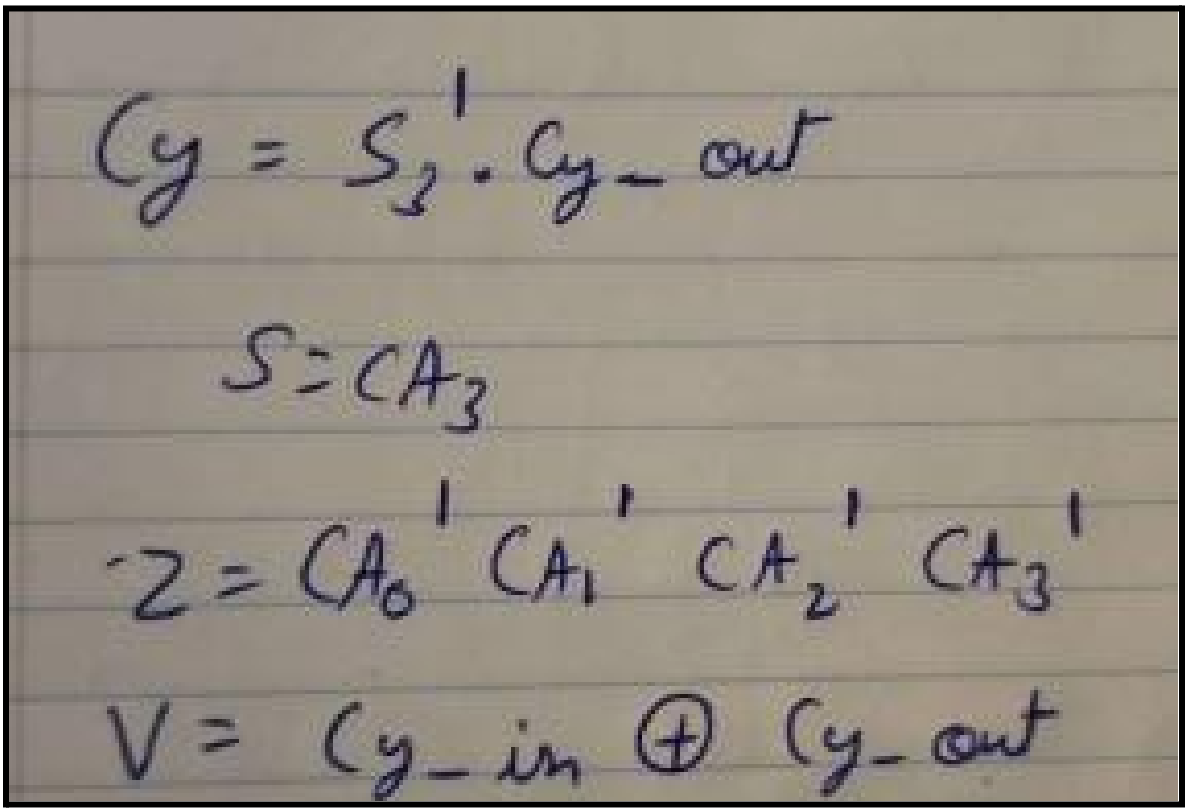
S2	S1	S0	CL_i
0	0	0	$CL_i \leftarrow 0$
0	0	1	$CL_i \leftarrow 1$
0	1	0	$CL_i \leftarrow A_i \cdot B_i$
0	1	1	$CL_i \leftarrow A_i + B_i$
1	0	0	$CL_i \leftarrow A_i \oplus B_i$
1	0	1	$CL_i \leftarrow A_i \cdot B_i'$
1	1	0	$CL_i \leftarrow A_{i-1}$
1	1	1	$CL_i \leftarrow A_{i+1}$

5.4: Design of Arithmetic Circuit(AC)





5.5: ALU state register



Part I (Design) sections (6.1. – 6.6)

6.1: Register 4-bit

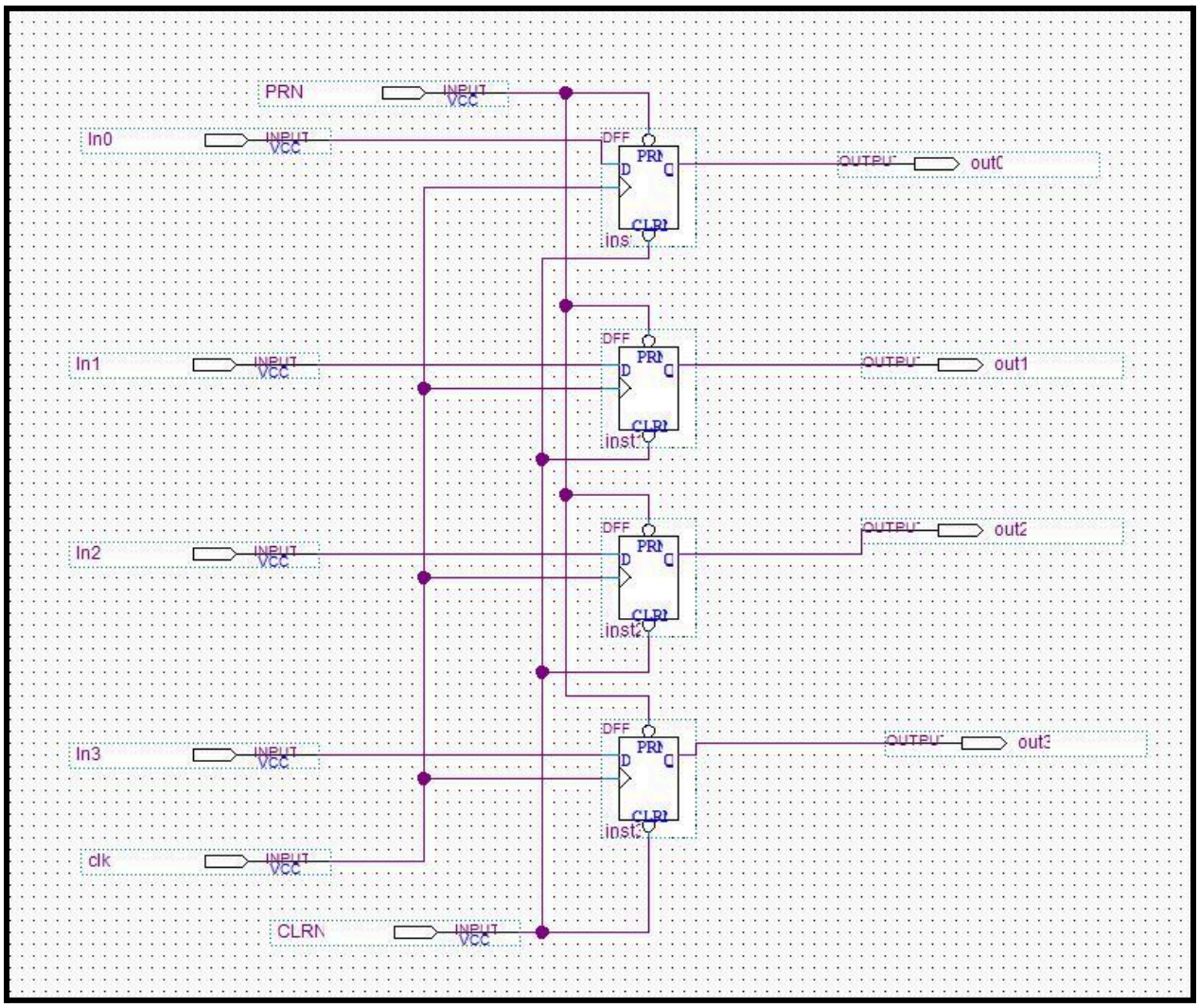


Figure 4 - Resister4bit

Fig. 4 Analysis:

In figure 4, we designed a circuit for the 4-bit register, this was implemented using four of the D flip-flops.

6.3: logic circuit 1-bit

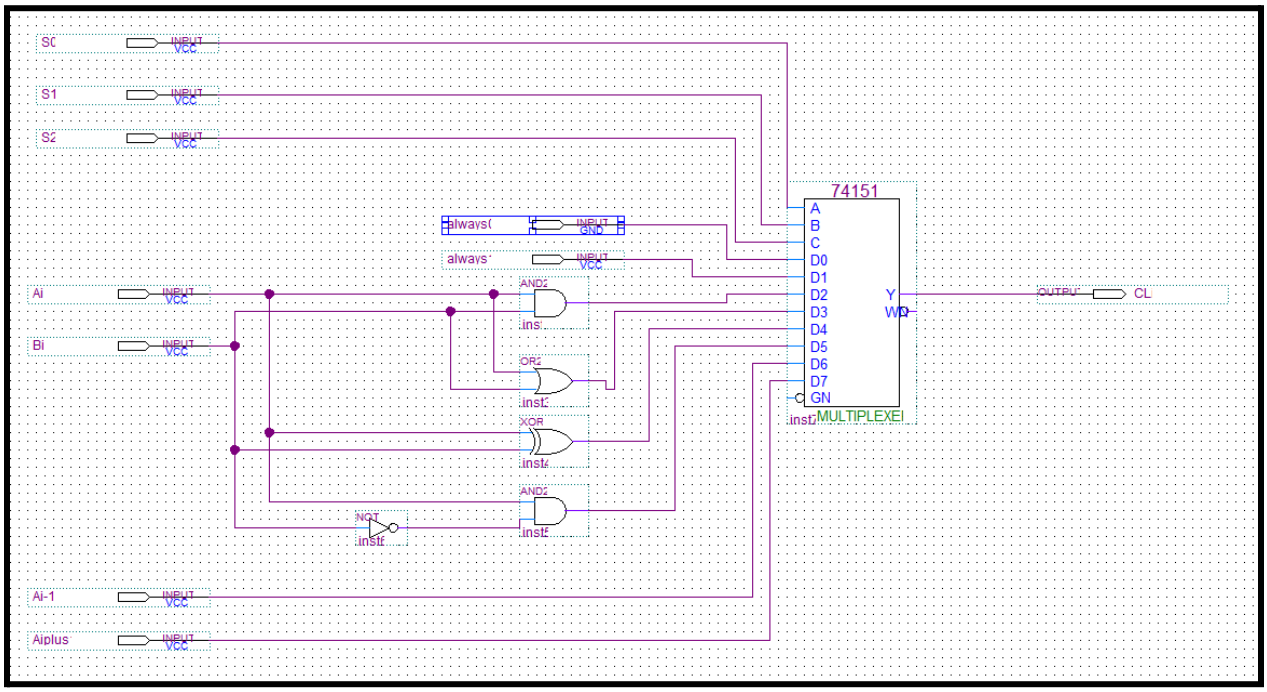


Figure 5 - Logic circuit_1bit

Fig. 5 Analysis:

Here, we have designed a 1-bit logic circuit constructed using an 8-to-1 multiplexer (IC 74151, as shown in Fig. 5). The circuit takes three inputs, denoted as (S0, S1, S2), and two data inputs, A and B. We have implemented the logic equation obtained from the truth table in our prelab to configure the operation of the multiplexer.

6.4: logic circuit 4-bit

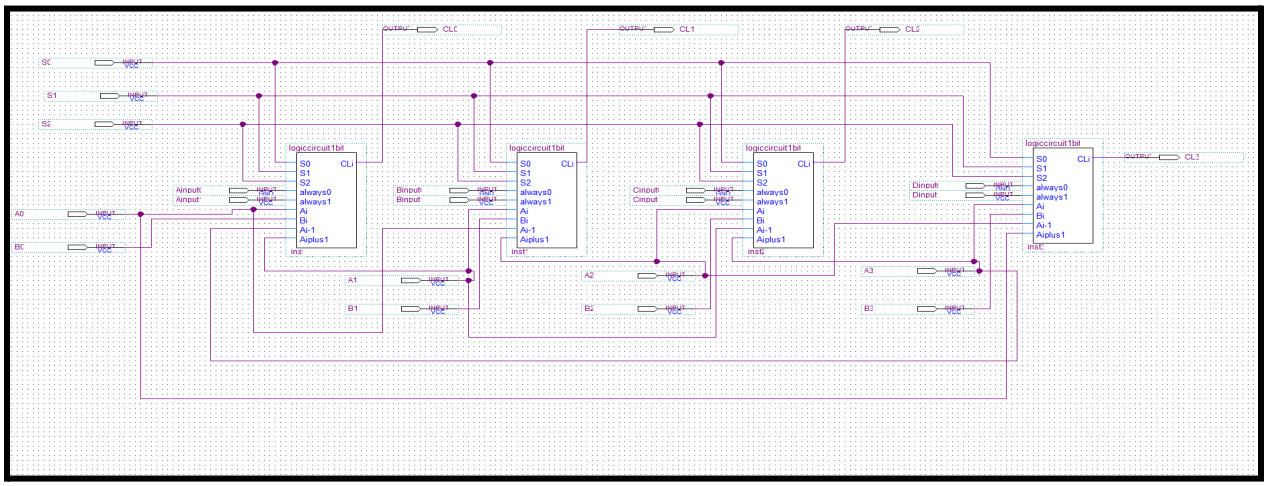


Figure 6 - Logic circuit_4bit

Fig. 6 Analysis:

As seen in Figure 6, it comprises four 1-bit Logic State Circuit (LSCs) interconnected to function as a 4-bit LSC. The circuit receives three control inputs designated as (S0, S1, S2) and two data inputs, A and B, for each bit from (A0 - Ai) and (B0 - Bi), resulting in four output signals (C0 - Ci). It executes the identical logic equation implemented for the 1-bit LSC, allowing the application of the same logic operation to a 4-bit binary number.

6.5: Full Adder 1-Bit

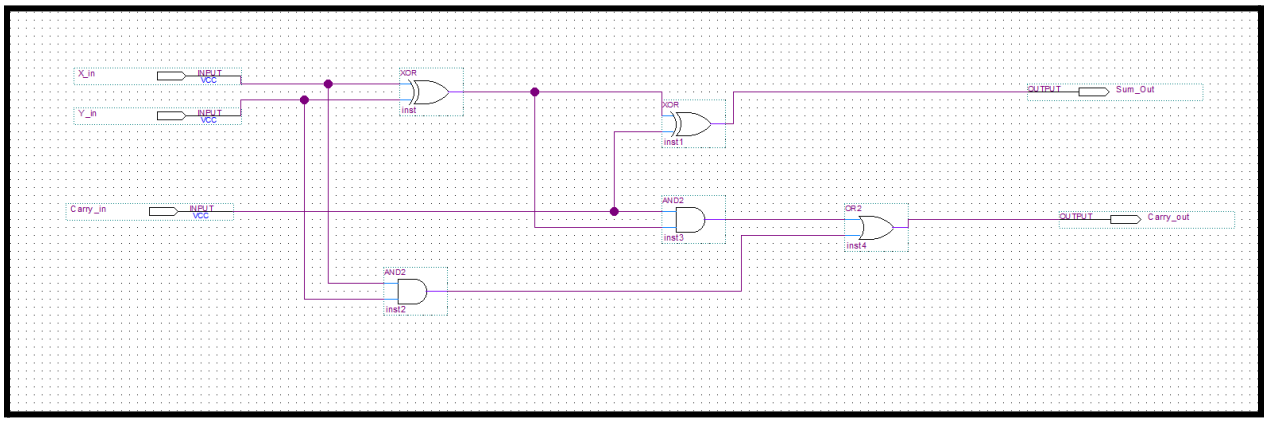


Figure 7 - Full Adder 1 Bit

Fig. 7 Analysis:

This is a depiction of a 1-bit full adder, designed to accept three data inputs (A, B, and Carry_in) and produce two outputs (Sum_out and Carry_out). The purpose of this circuit is to execute binary addition for two bits and provide a carry-out signal in cases where an addition results in a carry. Additionally, it plays a crucial role in building multi-bit adders for more extensive arithmetic operations as we will see in upcoming steps.

6.6: Full Adder 1-Bit with (Op1 and Op2)

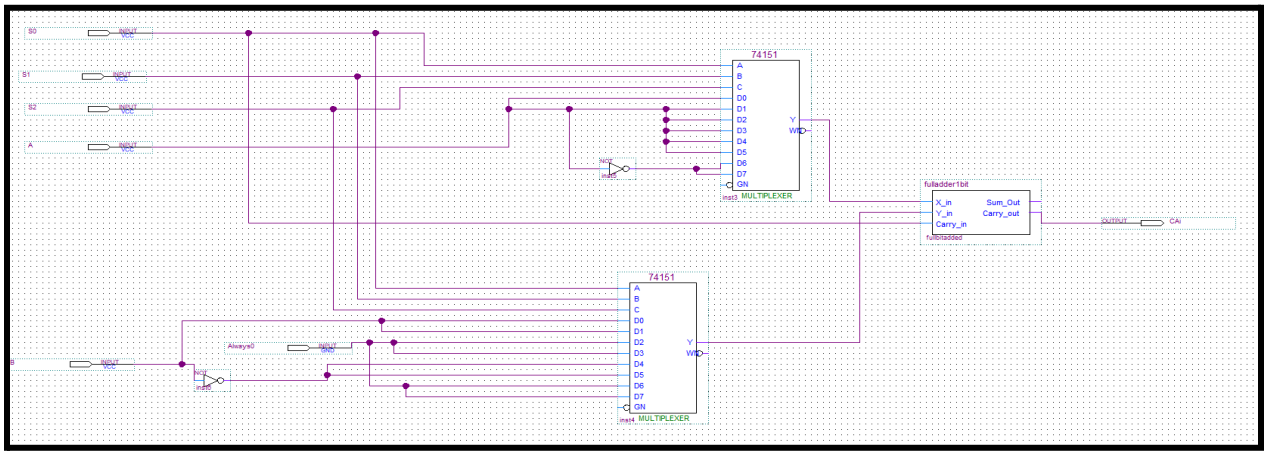


Figure 8 - Full Adder with Op1 & Op2

Fig. 8 Analysis:

At this stage, it remains a 1-bit full adder; however, it incorporates an additional two 8-to-1 multiplexer (MUX) setup, serving as a micro-operation selector with inputs (S0, S1, S2). The specific combination of S0-S2 dictates the values of data inputs A and B, which are subsequently passed to the 1-bit full adder as inputs op1 and op2. This enables the final execution of binary addition on these bits.

6.6: Full Adder 4-Bit

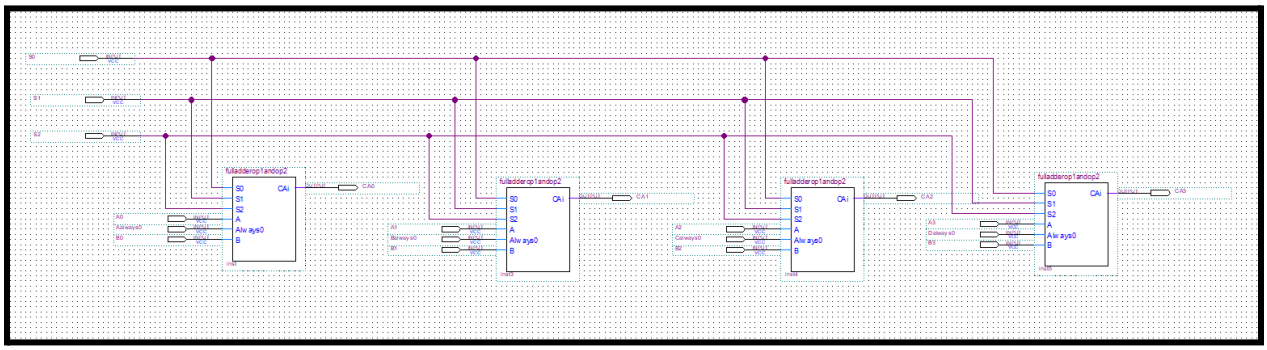


Figure 9 - Arithmetic Adder 4 bit

Fig. 9 Analysis:

The circuit depicted in Figure 9 constitutes a comprehensive representation of a 4-bit full adder. It consists of four 1-bit full adders interconnected in sequence, from the most significant bit to the least significant bit. The circuit takes three control inputs labeled as (S0, S1, S2) and two data inputs, A and B, for each bit from (A0 - Ai) and (B0 - Bi), generating four output signals (C0 - Ci). Its primary purpose is to carry out binary addition for 4-bit numbers and deliver carry-out results from each individual bit operation.

6.7: State register

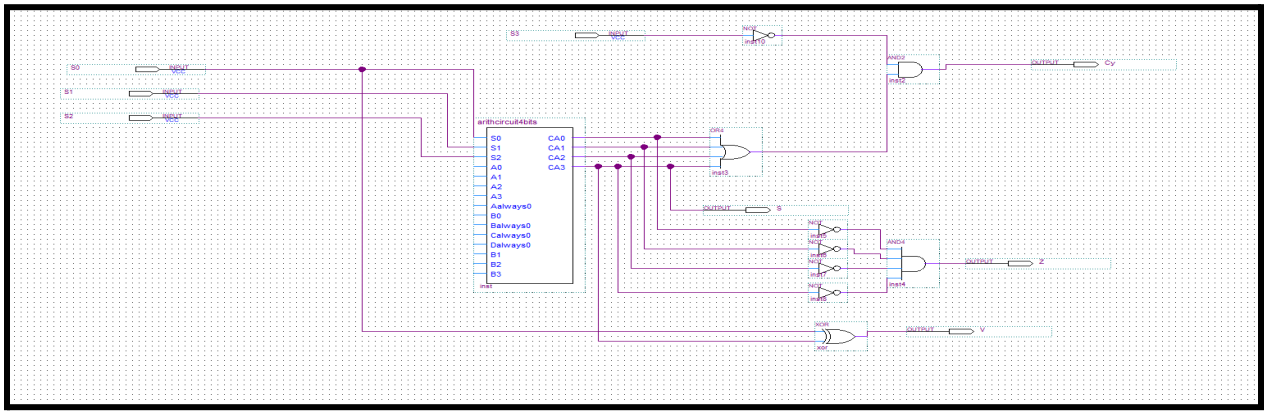


Figure 10 - ALU State register

Fig. 10 Analysis:

This is an ALU state register featuring four distinct states (Cy, S, Z, V), each determined by the logic equations derived in the prelab. Each state signifies a unique logic condition and represents the specific operational mode of the circuit.

6.8: Lab3 Top circuit

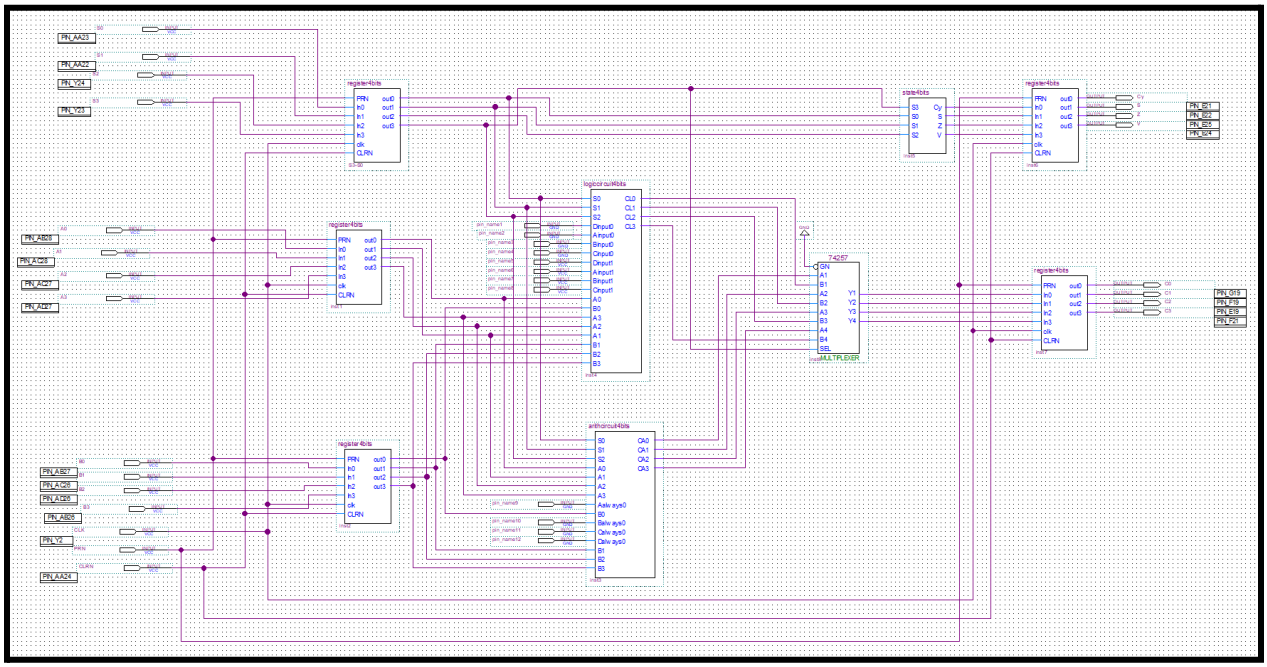


Figure 11 - Lab3_Top (FULL CIRCUIT)

Fig. 11 Analysis:

This circuit is obtained by combining all the circuits from the figures above i.e. 4-Bit (LSC), 4-Bit (AC) and State register. This gives us the full diagram of the ALU datapath that is provided in the lab manual. The circuit takes three control inputs labeled as (S0, S1, S2) and two data inputs, A and B, for each bit from (A0 - Ai) and (B0 - Bi), generating four output signals (C0 - Ci) and 4 state of the circuit resulting from State register. This circuit implements the functionality given in the table 1 given below:

Cycle	S	A	B	C	Cy,S,Z,V
1	1101	1010	0011	1000	1010
2	1110	0110	0000	1100	0010
3	0000	0011	0101	1000	1010
4	0011	1100	0000	1101	0010
5	1100	0011	0101	0110	0000
6	0110	1010	0000	0101	0000
7	1000	****	0000	0000	0100
8	0101	0101	0011	0010	0001
9	0010	1110	0000	1110	0010
10	0111	0110	0000	1010	0010
11	1010	0101	0011	0001	0000
12	0001	0001	0010	0100	0000
13	1111	1101	0000	0110	0010
14	0100	0110	0101	000	0101
15	1001	****	0000	1111	0010
16	1011	1100	1010	1110	0010

Table 1 : Sequence of M-O Executed

Discussion:

Within this lab, we were tasked with creating an ALU (Arithmetic logic unit). First the pre-lab was completed with the given operations. Subsequently, we designed and implemented the arithmetic and logic circuits. We created an ALU with the components of Figures 4-9.

The logic circuits play the role of assisting us with doing the shift operations, while the arithmetic circuits assist with binary addition of 4-bit numbers using the 4-bit (AC).

To verify the compliance of our circuit with the truth table provided in Table 1, we conducted simulations for each row of the table. This involved applying data inputs A and B, along with control inputs S0-Si, and observing the corresponding circuit outputs, including state and carry_out (C0-Ci). We successfully completed this task, and the results for every combination within the circuit were accurate. We have included one of these experiments in this report, with accompanying waveforms and brief explanations, which can be found below.

Experiment example 1 (Row -1)

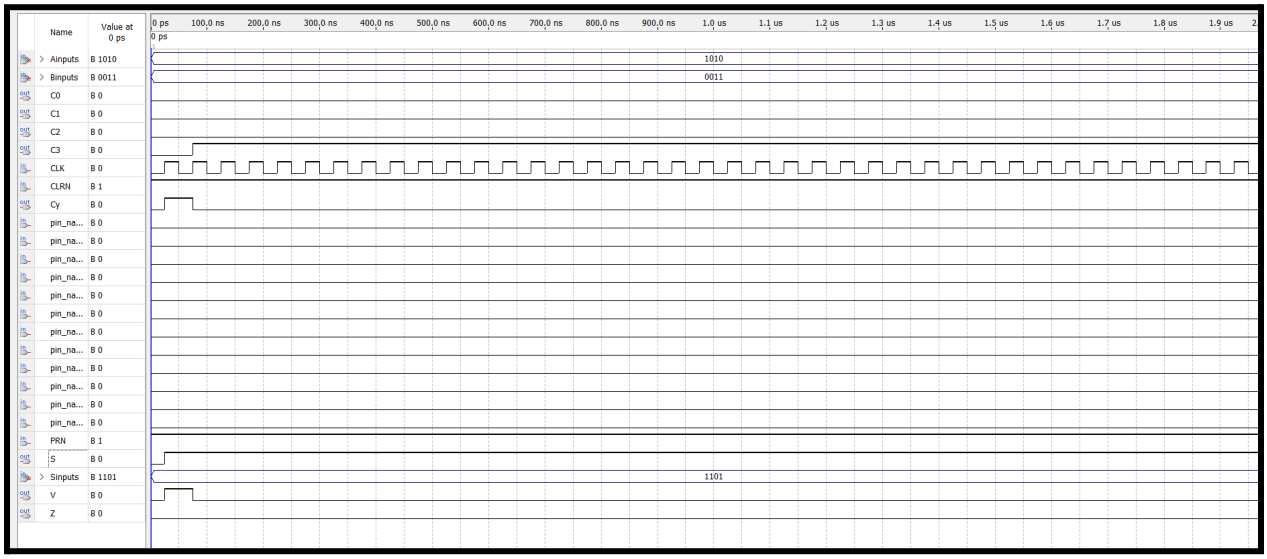


Figure 12 - Waveform for row #1 from truth table

During our simulation, we generated the waveform provided above, which was cross-verified to validate the correctness of our circuit. This validation was achieved by comparing it with the carry_out combinations from row 1 of Table 1. The comparison between Table 1 and Figure 12 substantiates that the actual outputs align with the expected results, confirming the flawless execution of this lab.

Using our final block schematic file, we used this to test on our Altera DE2 115 board. Using the pins and buttons and LED we could utilize this with the usb blaster on seeing our ALU perform.

Conclusion:

In this lab it was possible to understand how to design, build and also simulate an Arithmetic Logic Unit. We used Quartus II to develop our files and an Altera board to experiment with our ALU. Within our block schematic file, we used adders, registers and multiplexers. Our ALU matched with the truth table shown in figure 12. We did not encounter any major issues while executing this lab. In conclusion of this report, our ALU was built successfully and was functional while tested on the Altera board.