North South University

Department of Electrical & Computer Engineering

Lab Report

Experiment No:	**05**
Experiment Title:	**Design of a 2-bit Arithmetic Unit**
Course Code:	**CSE332L**
Course Name:	**Computer Organization & Architecture Lab**
Name & ID:	**Ishrat Jahan,1921909042**
Date of Experiment:	**02/12/2021**
Date of Submission:	**07/12/2021**

Objective:

1: To construct a 2-bit arithmetic Unit which will perform arithmetic operations: Add, Add with carry, Subtract, Subtract with borrow, Increment A, Decrement A and Transfer A.

2: Understanding how each of these arithmetic operations work on a 2-bit arithmetic unit.

3: Learning how to vary the inputs and Cin of the full adder to make the circuit perform the desired operation.

Theory:

2-bit Arithmetic Unit: It is a part of an ALU. As it's 2-bit, there will be two inputs, each of 2-bit. This Arithmetic unit performs operations like addition, subtraction, increment, decrement or transfer of any of the inputs.
As it's 2-bit, the number system will have 4-combinations of inputs: 00, 01, 10, 11.

Arithmetic Operations:

i) Add: Each bit of input A is added with the corresponding bit of input B. Here corresponding bit means the LSB of A and B are added, then the MSB of A and B are added. The sum will appear at the output of each full adder along with the carry out if any. As it is just addition, we will consider there is no carry from before and Cin will be 0.

$$A = A_1 A_0.$$
$$+\; B = B_1 B_0$$
$$S = D_1 D_0$$

Therefore : $\therefore A+B+0$

ii) Add with carry: Each bit of input A and B are added with the input carry and the sum will appear at the output of each full adder with any carry out. Here we consider $C_{in} = 1$, that's why it is Add with carry.

$A = A_1 A_0$ $\quad\quad\therefore A + B + 1 (C_{in})$.

$B = B_1 B_0$

iii) Subtract: Each bit of input B is subtracted from the corresponding bit of input A. The result after subtraction appears at the output of each full adder with any borrow out.

We know, $A - B$

$= A + (-B)$. $\quad\therefore -B$ stand's for 2's complement of B

For calculating $-B$ from B:-

i) We first find 1's complement of B by toggling the respective bit's

ii) Then we add 1, with 1's complement of B to find the 2's complement of B.

Therefore, $A - B$

$= A + (-B)$

$= A +$ 2's complement of B

$= A +$ 1's complement of $B + 1$.

$= A +$ 1's complement

$= A + B' + 1$.

Here, 1 is the C_{in} of the full adder.

iv) Subtract with borrow: Each bit of input B is subtracted from A with borrow. The result after subtraction appears at the output of each full adder with any borrow out.

A subtraction with borrow means it will take away one 1.

For subtraction we know,

$$A - B$$
$$= A + (-B) \; [2\text{'s complement of } B]$$
$$= A + 1\text{'s complement of } B + 1$$
$$= A + B' + 1.$$

After subtraction with borrow:-

We are taking away one 1

$$\therefore A + B' + 0.$$

Here, 0 is the C_{in} of the full adder.

Incrementing one Input: For incrementing/decrementing, we will be doing it for only one input. because we have a limitation as we are constructing a 2-bit Arithmetic Unit. If I increment both inputs A and B at the same time, it's not possible to see a 4 bit output in a full adder with 2-bit output.

For example:- $A = 00.$
$B = 11.$

If we increment A by 1: $A = 01$
If we increment B by 1: $B = 100.$

We can't see both of these output together in the full adder's 2 bit output port. That's why we increment / decrement only one of the input at a time.

v) Increment A: Each bit of A is increased by 1 and the output appears at the output port of each full adder. We can increment by 2 as well. As we want to see the increment of A in the output of each full adder we will be keeping B's input bit to 00 and the Cin as 1.

$$\therefore A + 00 + 1$$

This equation will show the increment output of A in the full adder's output port.

vi) Decrement A: Each bit of A is decreased by 1 and the result appears at the output of each full adder.

If A is decremented by 1

$$\therefore A - 1$$
$$= A + (-1).$$
$$= A + (-01)$$
$$= A + 2\text{'s complement of } 01$$
$$= A + 1\text{'s complement of } 01 + 1$$
$$= A + 10 + 1$$

After adding 1:-

$$= A + 11 + 0$$

Here 11 is the input of B and 0 is the Cin.

vii) Transfer A: Each bit of A appears at the output of each full adder without any modification. Therefore, the input bit of B and the C_{in} will be 0.

$\therefore A + B + C_{in}$

$\therefore A + 00 + 0$.

<u>Decrementing</u>: The 2-bit number system has:—

00
01
10
11

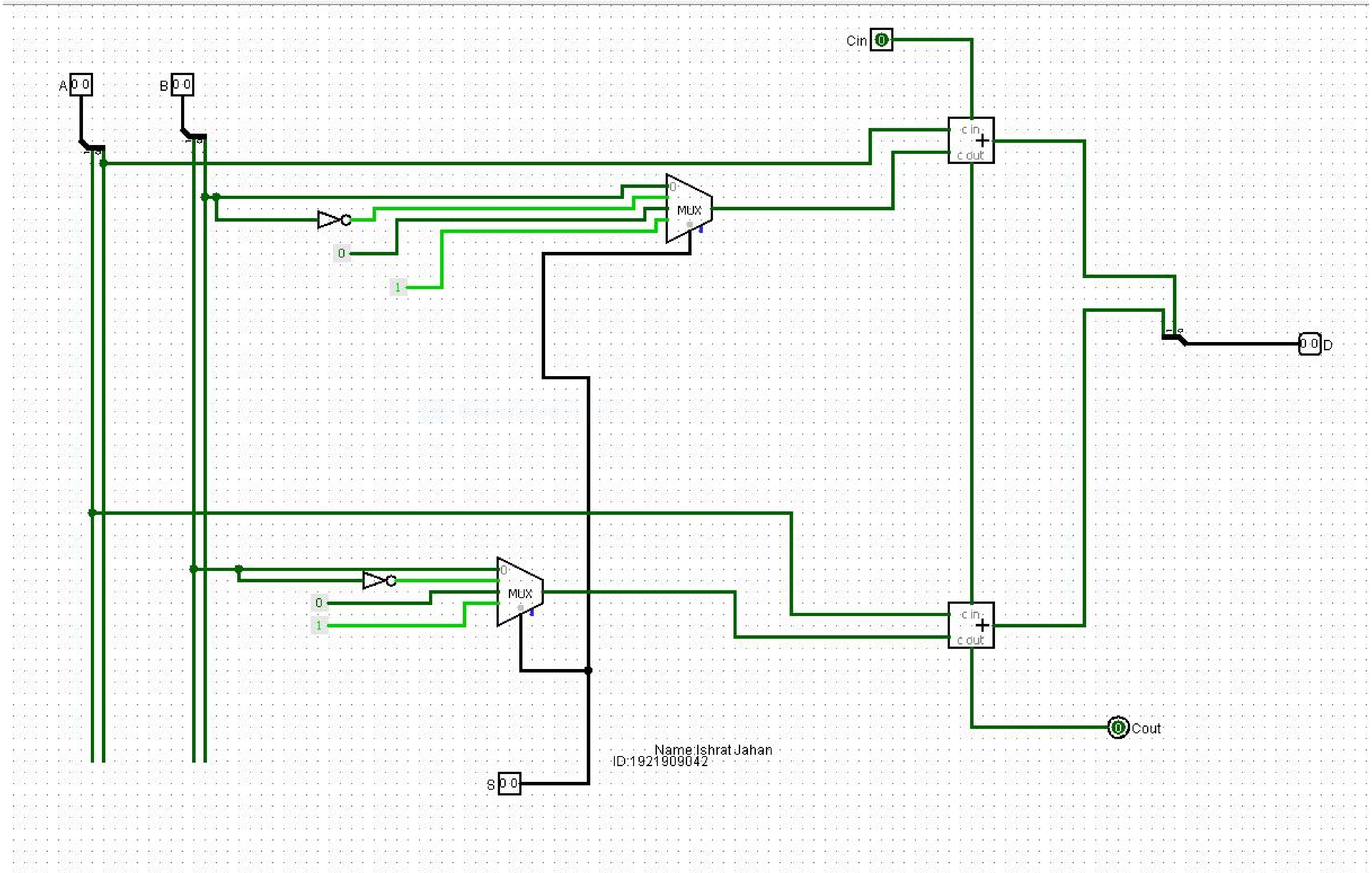
∴ If I want to decrement from 01 by 1, the output will be

$$\begin{array}{r} 01 \\ -\ 1 \\ \hline 00 \end{array}$$

If it is looked in the way of addition, by adding 3(11) with 01, I will get the same result I get by subtracting 1 from 01.

$$\begin{array}{r} 01 \\ +\,11 \\ \hline 00 \end{array}$$

Therefore for 2 bit, we need to add 3(11) for decrementing an input by 1. Likewise for 3, we need to add 7(111) to decrement an input by 1.



Cin
A
B
MUX
0
1
c in
c out
D
Cout
S
Name:Ishrat Jahan
ID:1921909042

Function Table:

S1	S0	Cin	A1	A0	B1	B0	D1	D0	Cout	Micro operation
0	0	0	0	0	0	1	0	1	0	Add
0	0	1	1	0	0	1	0	0	1	Add with Carry
0	1	0	0	1	0	0	0	0	1	Subtract With Borrow
0	1	1	1	1	0	1	1	0	1	Subtract.
1	0	0	1	1	0	1	1	1	0	Transfer A A1A0+00+0
1	0	1	1	0	1	0	1	1	0	Increment A A1A0+00+1
1	1	0	1	1	0	0	1	0	1	Decrement A A1A0+11+0.
1	1	1	1	0	0	0.	1	0	1	Transfer A. A1A0+11+1

Discussion

The experiment of Lab 05 was based on "Design of a 2-bit arithmetic unit which is a part of an ALU. In this lab, we learned how to design a 2-bit arithmetic unit in Logisim and how to perform arithmetic operations like Add, Subtract, Increment, Decrement and Transfer of any of the inputs using it.

At first, we studied the concept of 2 bit arithmetic unit. We knew arithmetic unit is a part of ALU and it performs arithmetic operations. Then we proceeded to learn what 2 bits mean in an arithmetic unit. For a 2-bit arithmetic unit, we need two inputs A and B each of them being 2 bit; $A = A_1A_0$, $B = B_1B_0$ (where A_0 and B_0 is the least significant bit and A_1 and B_1 is the most significant bit). As we were asked to perform seven specific arithmetic operations with the arithmetic unit, we learned how each of them works and how we need to vary the inputs (A and B) and what equipments we need and adjusting of the equipments as well.

The first arithmetic operation we discussed was Add (Addition). For addition, each bit of input A is added with the corresponding bit of input B; The LSB of A and B are added, likewise the MSB of A and B are added together. For example: If A=00 and B=01, the LSB of A is 0 and the LSB of B is 1. So they are added. On the other hand, the MSB of A is 0 and the MSB of B is also 0. So, they will added. For addition, we know we need to used a full adder. Therefore, the output of the addition of A and B will appear at the output of each full adder with any carry out if any. As it's just addition, we will consider their is no carry from before and Cin will be 0. in the full adder. Next we discussed Add with Carry where each bit of A and B are added with the input carry. Here we considered Cin as 1, as there will be a carry. The output of this operation will also be displayed at the output port of the full adders. As a result we got the following equation :

$$Add = A + B + 0$$

$$Add\ with\ Carry = A + B + 1.$$

The third operation that we discussed was Subtract. For subtraction we learned, each bit of input B is subtracted from the corresponding bit of A. Then we saw how to perform subtraction by finding the 2's complement of B. and adding that with A. We know, A − B can be written as A + (−B). We can find (−B) by first finding it's 1's complement and then by adding 1. That is,

$$A + (-B)$$
$$= A + 2\text{'s complement of } B.$$
$$= A + 1\text{'s complement of } B + 1.$$
$$= A + B' + 1$$

Here 1's complement of B is B′ as it's calculated just by toggling each bit of B. and the 1 is the Cin of the full adder. The output will be displayed using the full adder's output,

Scanned with CamScanner

The fourth operation we learned about was subtract with borrow. A subtraction with borrow means it will take away one 1. As we know for normal subtraction the equation is:-

$\therefore A+B'+1$

If we take away the 1, we will get the equation for subtract with borrow.

Therefore

$\therefore A+B'+0$ is the equation of subtract with borrow where Cin will be zero. These results will be displayed at the output port of the full adder.

The fifth operation we learned was Transfer A. As it was already specified in the lab manual. Here, we discussed the Transfer of input A only. each bit of input A appears at the output of the full adder without any modification. Therefore, it suggests input B and Cin will be 00 and 0 respectively. So the equation becomes A+00+0.

The last two operations were Increment A and Decrement A. For increment and decrement we learned, we can do this arithmetic operation for only one input at a time. The reason is if I am designing a 2-bit arithmetic ~~operation~~ unit, the output is suppossed to be 2 bit. If I am incrementing/decrementing both the inputs, in total they boths are 4 bit and in a 2 bit output we won't be able to see 4 bit output result. Therefore, we proceeded with Increment A and Decrement A as mentioned in the lab manual.

For increment A, each bit of A is increased by 1. It can be increased by 2 as well. As we want to see the increment of A in the output of the full adder we will keep the input B 00 and the Cin as 1 for increment. The equation therefore becomes: A + 00 + 1. Whereas for decrement we decrease each bit of A by 1. For example: If I decrease A by 1 then:- A - 1

= A + (-1)

= A + (-01)

= A + 2's complement of 01

= A + 1's complement of 01 + 1

= A + 10 + 1

∴ After adding 1 = A + 11 + 0

Here 11 is the input of B and 0 will be the Cin of full adder. The outputs will appear at the output of each full adder.

Afterwards we learent how many combinations are there in a 2 bit Number system. It has 4 combinations 00, 01, 10 and 11. Then we learned how we can add a specific number with the combinations to decrement it by 1. For example, if I want to decrement 01 by 1 I will get 00 as the result. If I add 3(11) with any of the combinations in the 2-bit number system it will be decremented by 1.

$$
\begin{array}{r}
\therefore\ 01 \\
+\ 11 \\
\hline
\boxed{1}00
\end{array}
$$

Carry out.

Likewise for 3-bit number system we need to add 7(111) and we will get a decrement.

Following this, we used our theoretical knowledge and completed the function table.

Then we proceeded to design the 2-bit ~~to~~ arithmetic unit in logisim. At first I took two input pins, changed there data bits to 2 and labelled them as 'A' and 'B'. Then I took two splitter from the wiring section, with fan out and bit width changed to 2 and connected each of them to A and B. The 0 part in the splitter stands for A_0 and B_0 whereas the 1 part is for A_1 and B_1. As we will be performing one arithmetic operation at a time for LSB and MSB individually we will be using two MUX for 2-bits. Therefore, I took two MUX from the plexers section, changed their select bits to 2 and data bit's to 1 for each of them. Then I took an input pin, changed it's data bits to 2 and connected it with the select bit port of both Multiplexer. As we have noticed for all equations A is unchanged and only B's bits are changed for complements. Therefore, ~~we~~ I connected B_0 and B_0' (with NOT gate) with the first two inputs of the first multiplexer. The other two inputs had two constants 0 and 1. The first multiplexer did the operations for LSB. So, I repeated the same steps for MSB of A and B.

After that, I took two Adder from the Arithmetic section and changed there data bits to 1. The data bits was changed to 1 because each adder calculated the sum of LSB and MSB respectively. The first adder was for the LSB. I connected A_0 to one of the adders input and the output of the first mux to the second input port. The C_{out} of the first adder was connected to the C_{in} of the second adder because if the LSB generated a carry it is transferred to the MSB. For the second adder, it carried out the operations of the MSB of A and B so the inputs were A_1 and the MUX output of the second multiplexer.

For displaying the output, I took a output pin and connected it to the C_{out} of the second adder. This will display the carry. Then I took another output pin, changed it's data bits to 2 and labelled it as D. This output pin will display the result from the adders. There I took a splitter with fan out and Bit width of 2 and connected it with the output pin. The 0 portion of the splitter was connected with the adder output that carries out operation on the LSB and the 1 portion was connected to the other adder. The whole circuit was completed.

Finally, after completion I checked the outputs and matched them with my truth table. The circuit worked correctly and the experiment was successful.

The experiment was pretty easy to carry out after understanding the theoretical part. Therefore, I didn't face any limitation in this experiment.