North South University

CSE231L

Experiment # 5

Name of Experiment: **Binary Arithmetic**

Date of Performance: 6 November, 2019

Date of Submission: 20 November, 2019

Section: 13

Group: 3

Submitted To: Farhana Saleh

Submitted By:

ID	Name
1530486042	Md. Abdul Zabbar
1711038042	MD. ASHRAFUL KABIR
1712747042	Ashik Iqbal
1731046042	Nahian -Al Sabri
1530187042	Md.Ahasun kamal

Lab 5: Binary Arithmetic

A. Objectives :

* understand the concept of binary addition and subtraction.
* Learn about half and full binary adder.
* perform binary addition and subtraction using IC 74283.
* understanding the concept of BCD addition and implement a BCD adder using IC74283

B. Theory: Digital computers perform a variety of information of information-processing tasks. Among the functions encountered are the various arithmetic operations, the most basic arithmetic operation is the operation is the addition of two binary digits. The

simple addition consists of four possible elementary operations: $0+0=0$; $0+1=1$; $1+0=1$ and $1+1=10$. The first three operations produce a sum of one digit, but when both augend and added bits are produce a sum of one digit, but when both augend and added bits are equal to 1, the binary sum consists of two digits. The higher significant bit of this result is called a carry. when the augend and added numbers contain more significant digits, the carry obtained from the addition of two bits is added to the next higher order pair of significant bits.

A combinational circuit that performs the addition of two bits is called a

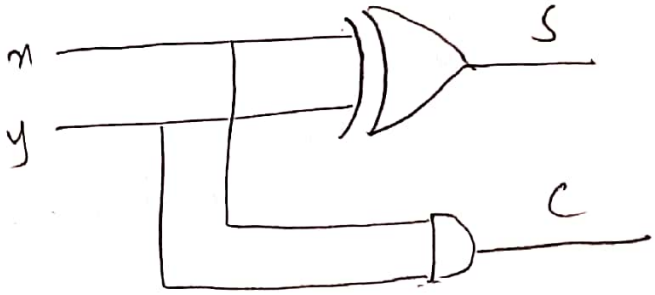
half adder. one that performs the addition of three bits (two significant bits and a previous carry) is a full adder. The names of the circuit stem from the fact that two half adders can be employed to implement a full adder.

In practice, binary addition is usually performed using ICs that contain several full adders chained and can be used to add together groups of bits. These ICs themselves can be chained to form even larger adders. Since binary subtraction is performed by complement addition, the adder ICs can also be used for subtraction by using some extra logical operations to perform

The complement calculation.

Half Adder

x	y	c	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



$S = x \oplus y$

$C = xy$

Fig B.1: Logic Diagram & Truth table of a half adder.

Full Adder:

x	y	z	c	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

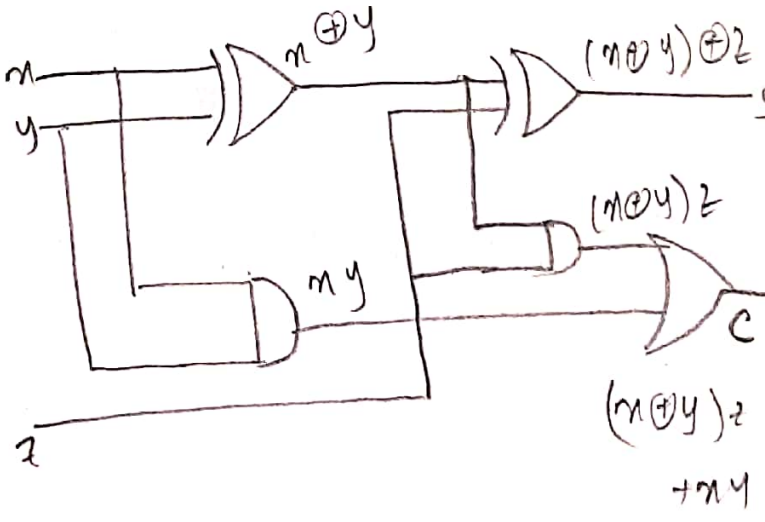


Figure B.2: Logic Diagram and Truth table of a full adder.

Experiment 1: Binary Adder Subtractor.

6.1 Apparatus

* Trainer board
* 1 × IC 7483 4-bit binary adder.
* 2 × IC 7486 quadruple 2-input XOR gates.

New Apparatus: IC 7483: The 16 pin 7483 IC is a 4 bit full adder. That means, it can take two 4 bit binary numbers ($A_4A_3A_2A_1$ and $B_4B_3B_2B_1$) and calculate the sum ($S_4S_3S_2S_1$). The input carry (if any) is connected to Cin and the output carry is obtained from Cout.

Two 7483 ICs can be calculated to form an 8-bit ripple through carry adder. The lower 4 bits of each

number is used as input for first 74283 and the output carry is connected to the input Carry of the next 74283. The higher 4 bits of each number is used as inputs for the second 74283. The first IC provides the lower 4 bits of the sum and the second one provides the upper 4 bits.

D.1 Procedure:

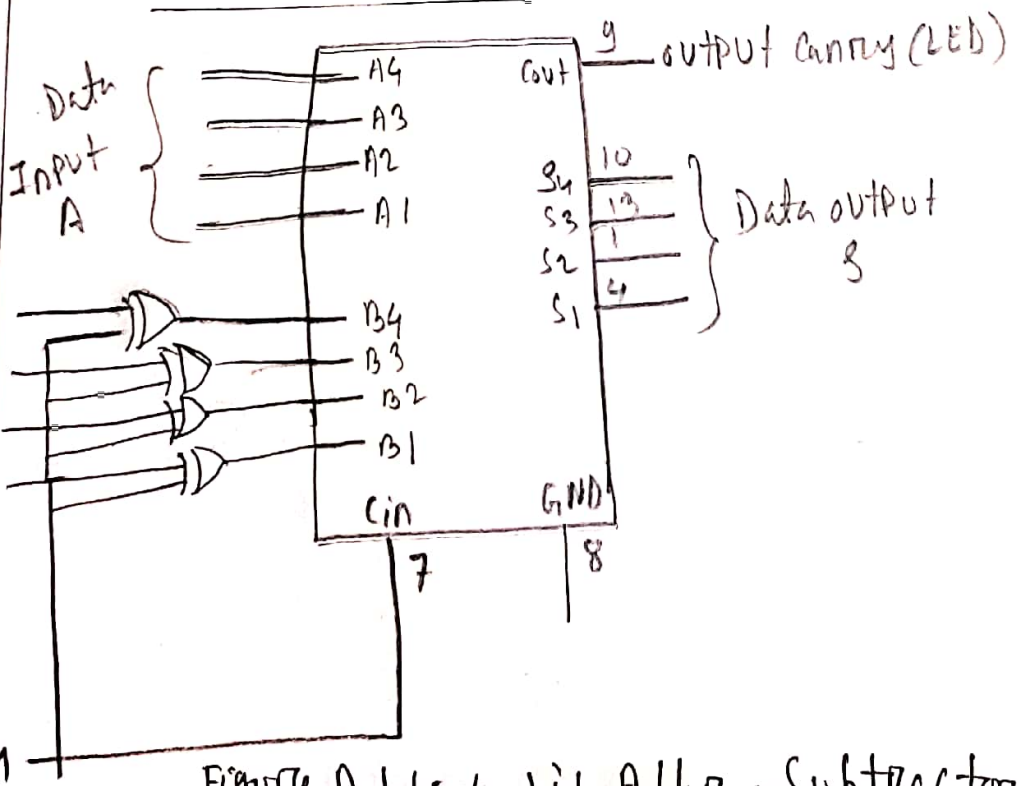


Figure D.1.1: 4 bit Adder - Subtraction.

Experiment 2: BCD Adder

C.2 Apparatus

* Trainer Board
* 2 × IC 74283 4-bit binary adder.
* 2 × IC 7408 quadruple 2 input AND gates.
* 1 × IC 7432 quadruple 2 input OR gates.

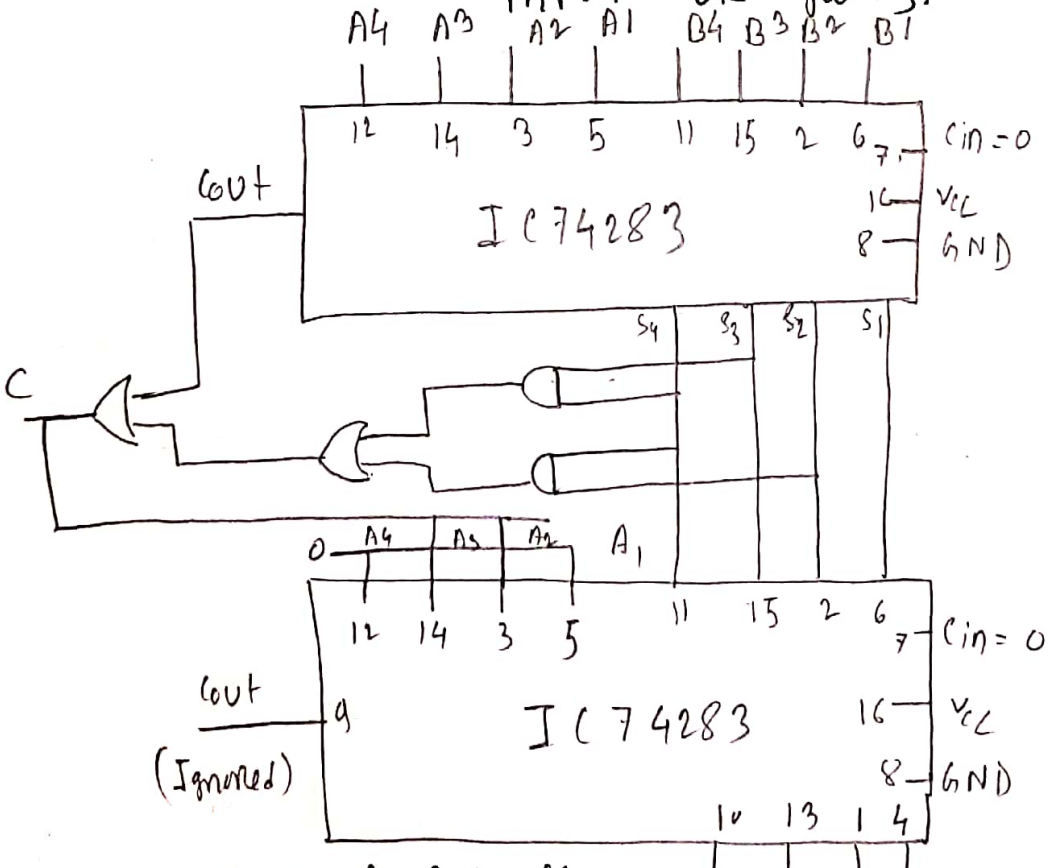


Fig D-2-1: BCD adder circuit

Question Answer:

(2) In the circuit we set mode control such that when the mode control is zero, addition is performed and subtraction is performed when the mode is one. we use XOR gates to feed the input, so that when mode control is one, the complement of each of the four bits are fed and when mode control is zero, the input as such is fed.

(3) (a) The AND gate and OR gate implementation connected at the B input of the 4 bit Adder is used to allow complement on un-complemented B input to be connected to the Adder input. Adding of two 4 bit numbers A and B can be performed by selecting the Add/Subtract = 0. The AND gates marked 0 (un-complemented) are enabled.

allowing $B^{3:3}$ to be passed on to the OR gates and the B input of the Adder. Subtraction is performed by selecting the Add/Subtract = 1. The AND gates marked C (Complemented) are enabled allowing complemented B^{3-3} to be passed on to the OR gates and the B input of the Adder. The Carry In is also set to 1 when the Add/Substract is set to 1.

(b) Here to get the output in BCD form, we will use BCD adder. Example: Input; A = 0111 B = 1000 output Y = 1010. Explanation; we are adding A(=7) and B(=8). But the BCD sum will be 1010, where 1 is 0001 in binary and 5 is 0101 in binary.

© The AND gate and OR gate implementation connected at the B input of the 4-bit Adder is used to allow complemented or UN-complemented B input to be connected to the Adder input. Adding of two 4 bit numbers A and B can be performed by selecting the Add/Subtract = 0. The AND gates are marked U (UN-complemented) are enabled allowing B^{0-3} to be passed on to the OR gates and B input of the Adder. Subtracting is performed by selecting the Add/Subtract = 1, The AND gates marked C (Complemented) are enabled allowing complemented B^{0-3} to be passed on the OR gates and the B input of the Adder. The Carry I_n is also set to 1 when Add/Subtract is set to 1.

(d) The addition of two decimal digits in BCD, will create a possible carray bit 1 which needs to be added to the next group of 4 bits. If the binary sum with the added canny bit is equal to or less than 9 (1001), the corresponding BCD digit is correct.

Discussion:

(i) we did face problem with IC's.

(ii) 5v Power source wasn't working. So we used ancient ~~~~ conventional way.

Data Sheet for Lab 5 :

F.1 (Experimental data (4 bit Binary adder - Subtractor)

operation	M	A	B	Cout	S_4 S_3 S_2 S_1
7+5	0	0111	0101	0	1 1 0 0
4+6	0	0100	0110	0	1 0 1 0
9+11	0	1001	1011	1	0 1 0 0
15+15	0	1111	1111	1	1 1 1 0
7-5	0	0111	0101	1	0 0 1 0
4-6	1	0100	0110	0	1 1 1 0
11-2	1	1011	0010	0	1 0 0 1
15-15	1	1111	1111	1	0 0 0 0
	1			1	

Table F.1.1

F2. Experimental Data (BCD Adder):

Decimal value	Binary sum					BCD sum				
	Cout	Z_3	Z_2	Z_1	Z_0	C	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
2	0	0	0	1	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	1	1
4	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	1	0	1
6	0	0	1	1	0	0	0	1	1	0
7	0	0	1	1	1	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0	0
9	0	1	0	0	1	0	1	0	0	1
10	0	1	0	1	0	1	0	0	0	0
11	0	1	0	1	1	1	0	0	0	1
12	0	1	1	0	0	1	0	0	1	0
13	0	1	1	0	1	1	0	0	1	1
14	0	1	1	1	0	1	0	1	0	0
15	0	1	1	1	1	1	0	1	0	1
16	1	0	0	0	0	1	0	1	1	0
17	1	0	0	0	1	1	0	1	1	1
18	1	0	0	1	0	1	1	0	0	0

19	1	0	0	1	1	1	1	0	0	1

Table F.2.1.

operation	A	B	overflow Carry	Sum
9+0	1001	0000	x	1001
9+1	1001	0001	x	0000
9+2	1001	0010	x	0001
9+3	1001	0011	x	0010
9+4	1001	0100	x	0011
9+5	1001	0101	x	0100
9+6	1001	0110	x	0101
9+7	1001	0111	x	0110
9+8	1001	1000	x	0111
9+9	1001	1001	x	1000

Table: F.2.2.

CSE231L/EEE211L
Lab 5 – Binary Arithmetic

Data Sheet: Instructor's Signature: .....

Section: 13	Group No.: 3	Date: 6 November, 2019

F.1 Experimental data (4-bit Binary Adder-Subtractor):

Operation	M	A	B	C_{out}	S4 S3 S2 S1
7 + 5	0	0111	0101	0	1100
4 + 6	0	0100	0110	0	1010
9 + 11	0	1001	1011	1	0100
15 + 15	0	1111	1111	1	1110
7 – 5	1	0111	0101	1	0010
4 – 6	1	0100	0110	0	1110
11 – 2	1	1011	0010	1	1001
15 – 15	1	1111	1111	1	0000

Table F.1.1

F.2 Experimental Data (BCD Adder):

Decimal Value	Binary Sum					BCD Sum				
	C_{out}	Z_3	Z_2	Z_1	Z_0	C	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1
2	0	0	0	1	0	0	0	0	1	0
3	0	0	0	1	1	0	0	0	1	1
4	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	1	0	0	1	0	1
6	0	0	1	1	0	0	0	1	1	0
7	0	0	1	1	1	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0	0
9	0	1	0	0	1	0	1	0	0	1
10	0	1	0	1	0	1	0	0	0	0
11	0	1	0	1	1	1	0	0	0	1
12	0	1	1	0	0	1	0	0	1	0
13	0	1	1	0	1	1	0	0	1	1
14	0	1	1	1	0	1	0	1	0	0
15	0	1	1	1	1	1	0	1	0	1
16	1	0	0	0	0	1	0	1	1	0
17	1	0	0	0	1	1	0	1	1	1
18	1	0	0	1	0	1	1	0	0	0
19	1	0	0	1	1	1	1	0	0	1

Table F.2.1

Operation	A	B	Overflow Carry	Sum
9 + 0	1001	0000	X	1001
9 + 1	1001	0001	X	0000
9 + 2	1001	0010	X	0001
9 + 3	1001	0011	X	0010
9 + 4	1001	0100	X	0011
9 + 5	1001	0101	X	0100
9 + 6	1001	0110	X	0101
9 + 7	1001	0111	X	0110
9 + 8	1001	1000	X	0111
9 + 9	1001	1001	X	1000

Table F.2.2

16.11.19

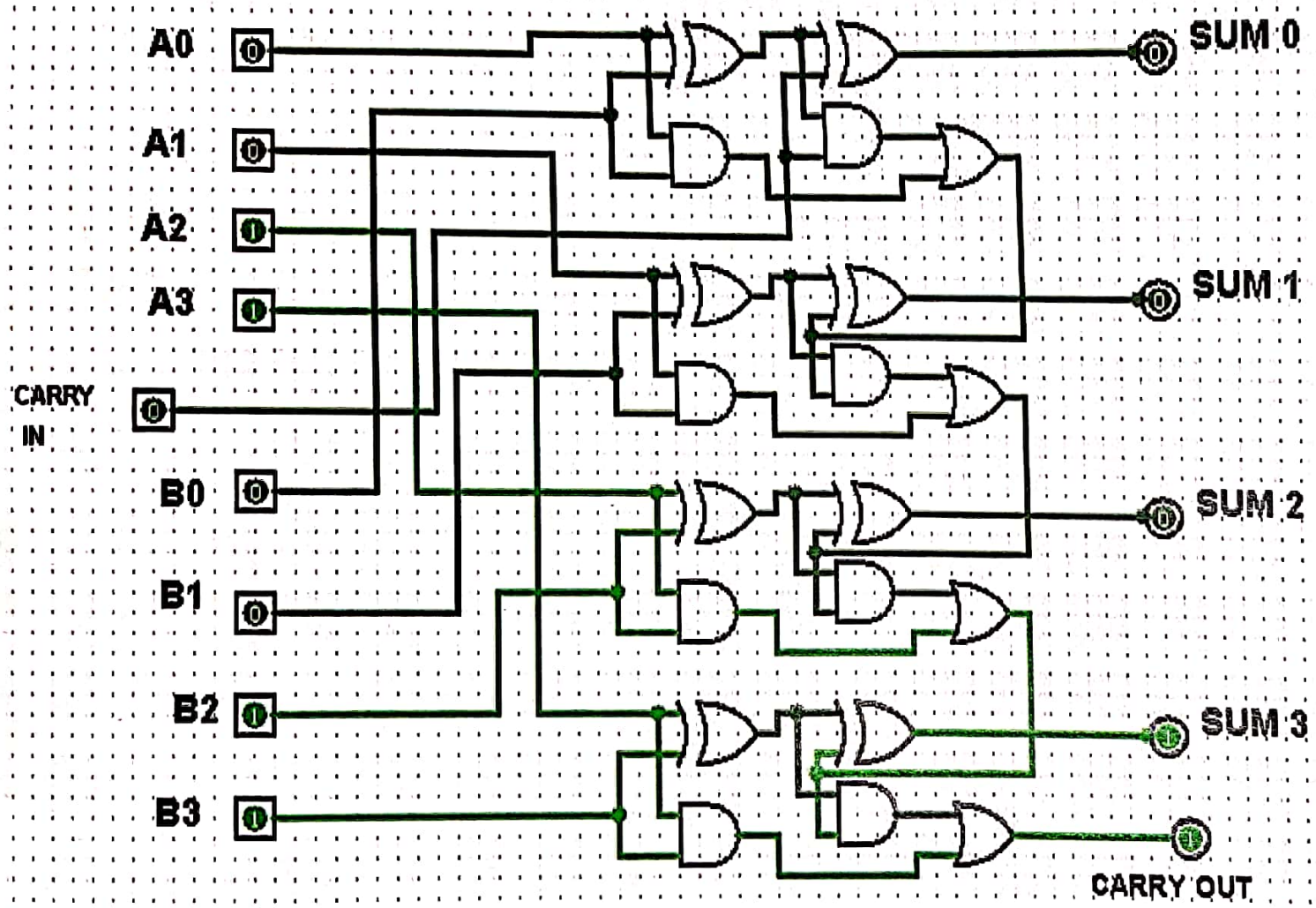


Figure: 4-bit adder using basic logic gates

Our group no: 3

so, the inputs are:

0011

0011