جامعة بيرزيت

BIRZEIT UNIVERSITY

ENCS2110

DIGITAL ELECTRONICS AND COMPUTER ORGANIZATION LABORATORY

Name: Sara Totah

Student ID: 1181779

Instructor: Khader Mohammed

T.A: Enas Jawabreh

Date: 03/03/2021

Table of Contents

Table of Figures: 3

Pre-Lab: 4

Three-Bit Comparator: 4

Abstract: 5

Theory: 5

First: Comparator Circuit: 5

1-Bit Comparator: 5

4-Bit Comparator: 6

Second: Half- and Full- Adder Circuits: 7

Half-Adder: 7

Full-Adder: 7

Third: Half- and Full- Subtractor Circuits: 8

Half-Subtractor: 8

Full-Subtractor: 9

Procedure: 10

Comparator Circuits: 10

A. Constructing Comparator with Basic Logic Gates: 10

B. Constructing Comparator with an IC: 10

Half- and Full-Adder Circuits: 11

A. Constructing Half- and Full-Adders with Basic logic Gates: 11

B. Full-Adder with Basic logic Gates: 11

C. Constructing BCD Adder: 12

Half- and Full-Subtractor Circuits: 12

A. Half-Subtractor 12

B. Full-Subtractor 13

Conclusion: 14

References: 14

Table of Figures:

Figure 1: Three-Bit comparator-- 4

Figure 2: N - bit Comparator-- 5

Figure 3: 1-bit comparator circuit --- 5

Figure 4: 4-bit Comparator Circuit diagram --- 6

Figure 5: 4-bit comparator constructed with1-bit comparator -- 7

Figure 6: Half Adder Logic circuit --- 7

Figure 7: Full-Adder circuit diagram -- 7

Figure 8:Full-Adder logic diagram --- 8

Figure 9:Half-Subtractor Logic Circuit -- 8

Figure 10:Full-Subtraction Circuit Diagram -- 9

Figure 11: Full-Subtractor Logic Diagram--- 9

Figure 12: Implemented 1-bit Comparator ---10

Figure 13: Comparator constructed with an IC --10

Figure 14: Implemented Half-Adder---11

Figure 15: Implemented Full-Adder ---11

Figure 16: 4-bit Full Adder constructed with IC--12

Figure 17: Half-Subtractor ---12

Figure 18: Full-Subtractor --13

Pre-Lab:

Three-Bit Comparator:

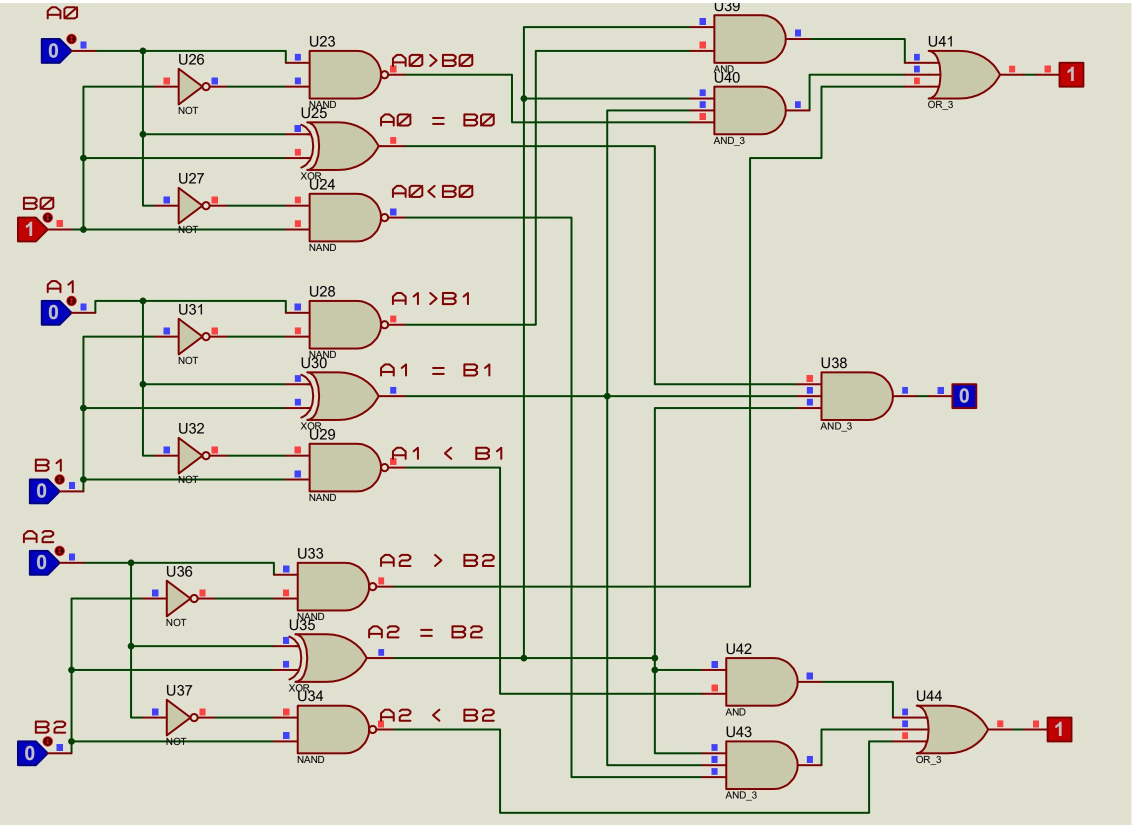

Figure 1: Three-Bit comparator

A2	A1	A0	B2	B1	B0	A>B	A<B	A=B
0	0	0	0	0	0	0	0	1
0	0	0	0	0	1	0	1	0
0	0	0	0	1	0	0	1	0
0	0	0	0	1	1	0	1	0
0	0	0	1	0	0	0	1	0
0	0	0	1	0	1	0	1	0
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
1	1	1	0	1	1	1	0	0
1	1	1	0	0	0	1	0	0
1	1	1	1	0	1	1	0	0
1	1	1	1	1	0	1	0	0
1	1	1	1	1	1	0	0	1

Abstract:

The Aim of the experiment: To understand the principle of the digital comparators, half-adders, full-adders, half-subtractor and full- subtractor, and how to implement each one of them.

Equipment Used in the experiment:

*Since the experiment was implemented online, we used a simulator called **Proteus.**

Theory:

First: Comparator Circuit:

Comparator Circuits are made of basic gates such as AND, NOR & NOT, these circuit should be able to determine whether the value of the input A is greater than, equal or smaller than the value of the input B.

The magnitude comparator has three output: One for equality A = B, another for greater than A > B, and the last one for less than A < B.

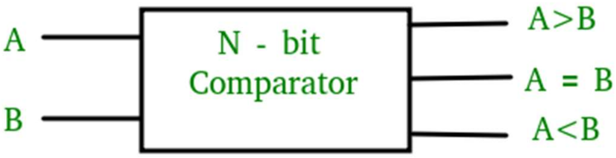

Figure 2: N - bit Comparator

1-Bit Comparator:

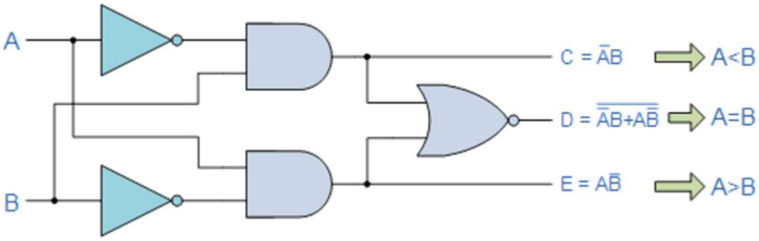

Figure 3: 1-bit comparator circuit

Inputs		Outputs		
B	A	A>B	A=B	A<B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

Table 1: 1-bit comparator Truth table

K-map for 1-bit comparator:

A > B

A B	0	1
0	0	0
1	1	0

A > B equation = A.B’

A = B

A B	0	1

0	1	0
1	0	1

A = B equation = A’B’ + AB

A < B

A B	0	1
0	0	1
1	0	0

A < B equation = A’B

4-Bit Comparator:

In order to design a 4 – bit comparator, each bit of the 4 – bit numbers should be compared on it’s own, and based on this comparison we can explain how to design the 4-bit comparator.

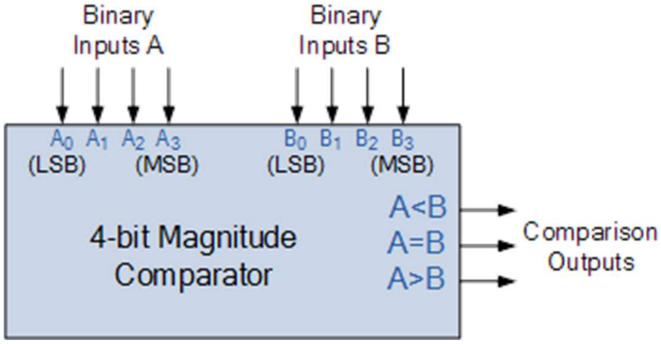

Figure 4: 4-bit Comparator Circuit diagram

A = B:

In order to equal the two inputs, each bit of the first number A should be equal to same bit of the second number B such as;

$$A_0 = B_0,\ A_1 = B_1,\ A_2 = B_2,\ A_3 = B_3$$

A > B:

There are multiple occasions where this output is true:

1. When $A_3 > B_3$
2. When $A_3 = B_3$ AND $A_2 > B_2$
3. When $A_3 = B_3$ AND $A_3 = B_3$ AND $A_2 > B_2$
4. When $A_3 = B_3$ AND $A_3 = B_3$ AND $A_2 = B_2$ AND $A_1 > B_1$
5. When $A_3 = B_3$ AND $A_3 = B_3$ AND $A_2 = B_2$ AND $A_1 = B_1$ AND $A_0 > B_0$

A > B:

There are multiple occasions where this output is true:

1. When $A_3 < B_3$
2. When $A_3 = B_3$ AND $A_2 < B_2$
3. When $A_3 = B_3$ AND $A_3 = B_3$ AND $A_2 < B_2$
4. When $A_3 = B_3$ AND $A_3 = B_3$ AND $A_2 = B_2$ AND $A_1 < B_1$
5. When $A_3 = B_3$ AND $A_3 = B_3$ AND $A_2 = B_2$ AND $A_1 = B_1$ AND $A_0 < B_0$

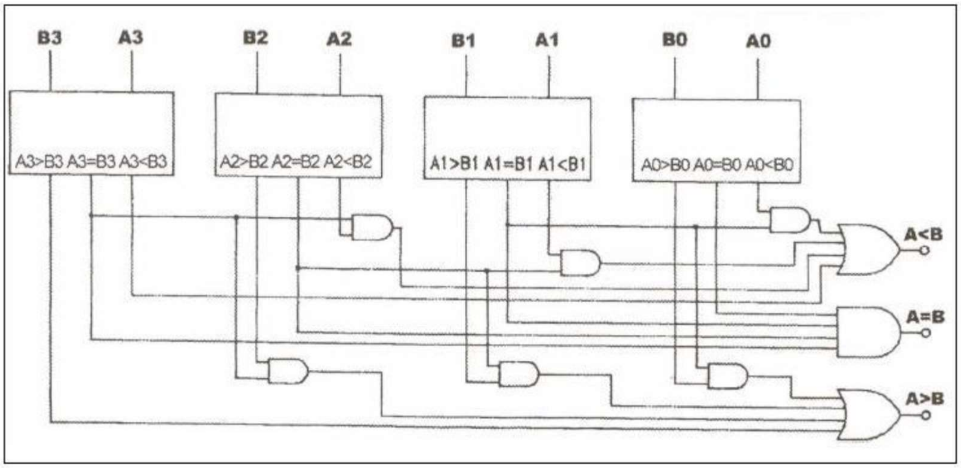

Figure 5: 4-bit comparator constructed with1-bit comparator

Second: Half- and Full- Adder Circuits:

Half-Adder:

The addition of two bits is done using a combinational circuit called Half-Adder:

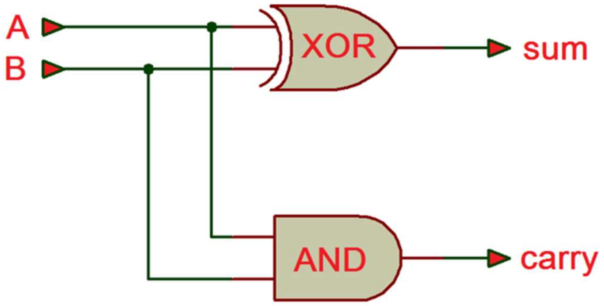

Figure 6: Half Adder Logic circuit

Inputs		Outputs	
B	A	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Table 2: Half-Adder Truth Table

Full-Adder:

The full-adder can perform addition or subtraction, it adds together two binary digits and carry in digit, which means it has three inputs and comes up with two outputs.

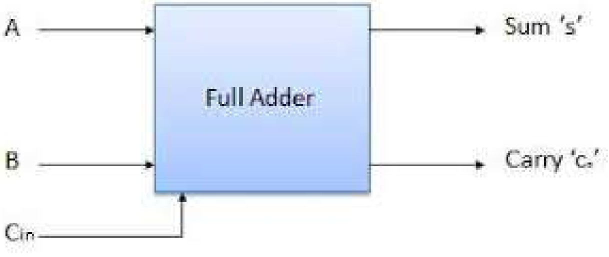

Figure 7: Full-Adder circuit diagram

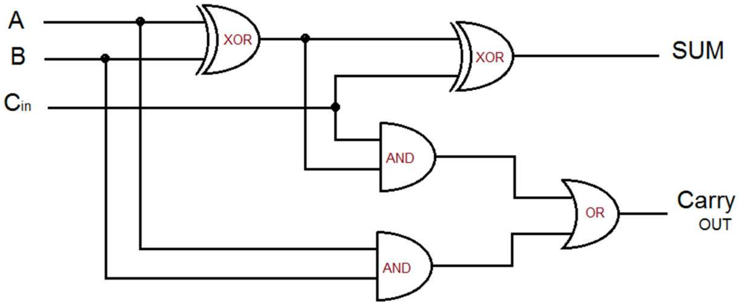

Figure 8:Full-Adder logic diagram

Inputs			Outputs	
A	B	C_{in}	Sum	Carry Out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table 3: Full-Adder Truth Table

Third: Half- and Full- Subtractor Circuits:

Half-Subtractor:

The circuit of half-subtractor can be built with NAND and XOR gates

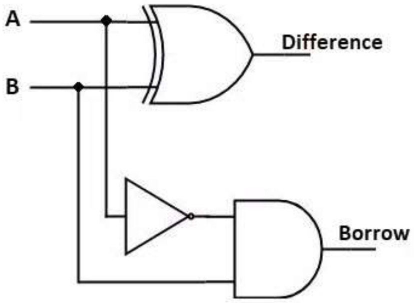

Figure 9:Half-Subtractor Logic Circuit

Inputs		Outputs	
B	A	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Full-Subtractor:

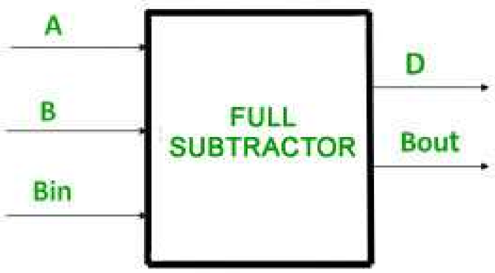

Figure 10:Full-Subtraction Circuit Diagram

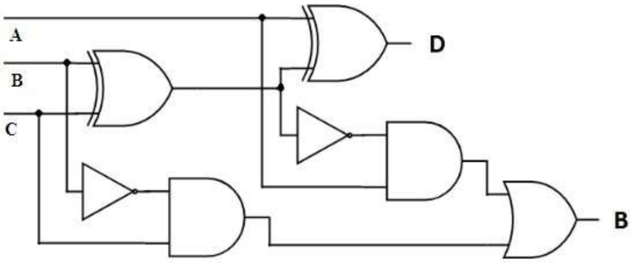

Figure 11: Full-Subtractor Logic Diagram

Inputs			Outputs	
A	B	B_{in}	D	B_{OUT}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Table 4: Full-Subtractor Truth Table

Procedure:

Comparator Circuits:

A. Constructing Comparator with Basic Logic Gates:

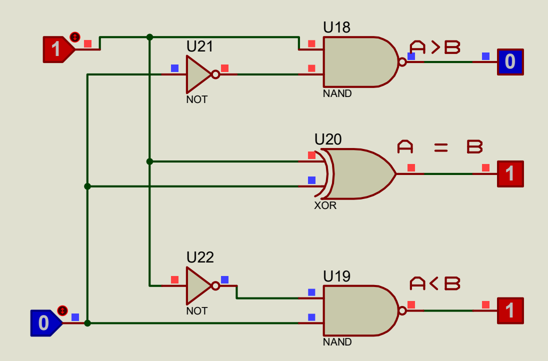

Figure 12: Implemented 1-bit Comparator

The circuit above was implemented using Proteus, and the output was recorded.

Inputs			Outputs		
B	A		F1	F2	F5
(SW2)	(SW1)		(L1)	(L2)	(L3)
0	0	A = B	1	1	0
0	1	A > B	0	1	1
1	0	A < B	1	0	1
1	1	A = B	1	1	0

Table 5: 1-bit comparator Truth Table

After comparing the truth table in the theory with this one, it's clear that the are exactly the opposite since we used NAND and XOR gates instead of NOT, AND & NOR.

B. Constructing Comparator with an IC:

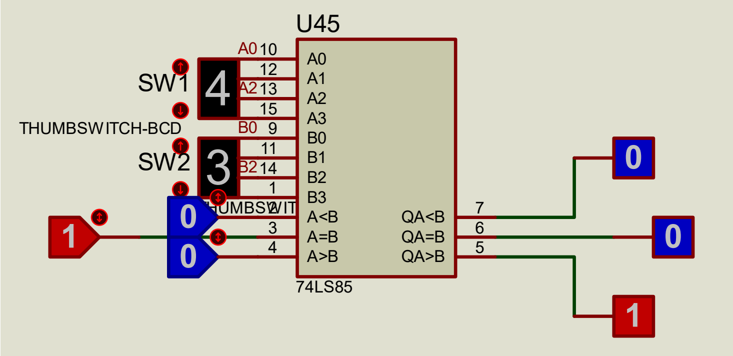

Figure 13: Comparator constructed with an IC

The circuit above was connected and A was set to equal B and the output was recorded

INPUTS			OUTPUTS		
A > B	A = B	A < B	A > B	A = B	A < B
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	0	0	0
1	1	1	0	1	0

Half- and Full-Adder Circuits:

A. Constructing Half- and Full-Adders with Basic logic Gates:

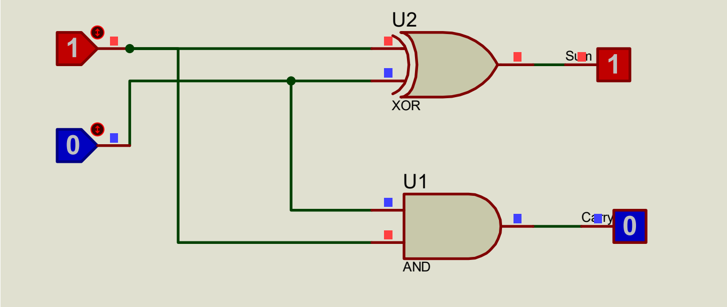

Figure 14: Implemented Half-Adder

INPUTS		OUTPUTS	
SW1 (B)	SW0 (A)	CARRY (F1)	SUM (F2)
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Table 6: Half-Adder Truth table

A B	0	1
0		1
1	1	

Sum

A B	0	1
0	0	0
1	0	1

Carry

After implementing the k-map on this truth table the functions of the sum and the carry were:

Carry = AB = A AND B

Sum = AB’ + A’B = AXOR B

B. Full-Adder with Basic logic Gates:

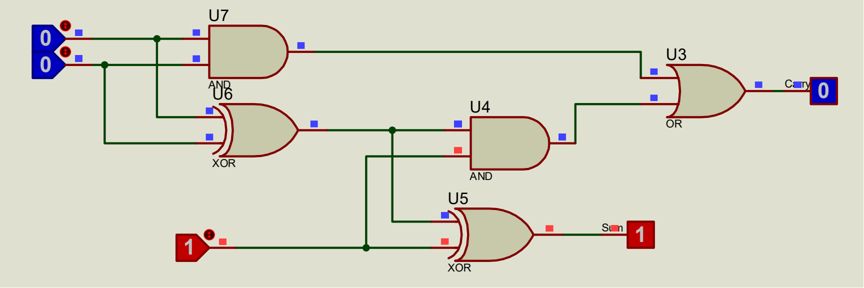

Figure 15: Implemented Full-Adder

INPUTS			OUTPUTS	
SW3 (C)	SW2 (B)	SW1 (A)	CARRY	SUM
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

Table 7: Full-Adder Truth Table

C_{in} \ AB	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Sum

C_{IN} \ AB	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Carry

After implementing the k-map on this truth table the functions of the sum and the carry were:

Carry = AC + BC + AB

Sum = CA'B' + C'AB' + CBA + C'BA'

C. Constructing BCD Adder:

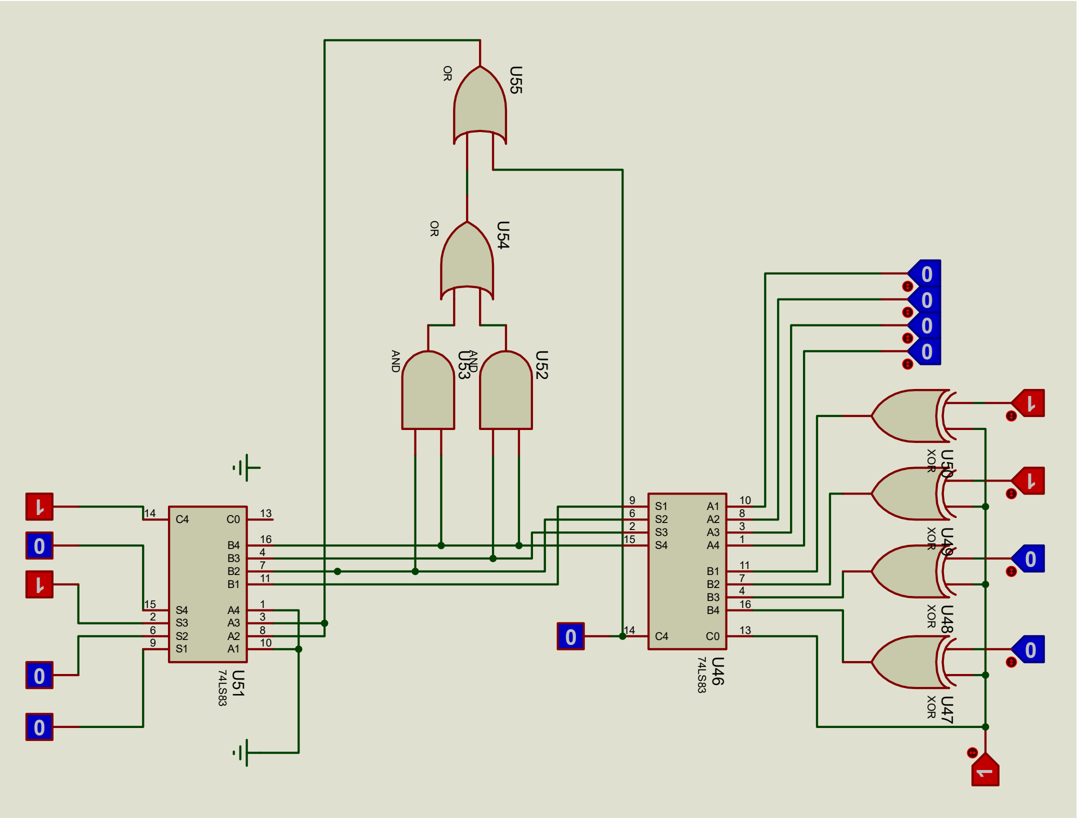

Figure 16: 4-bit Full Adder constructed with IC

Half- and Full-Subtractor Circuits:

A. Half-Subtractor

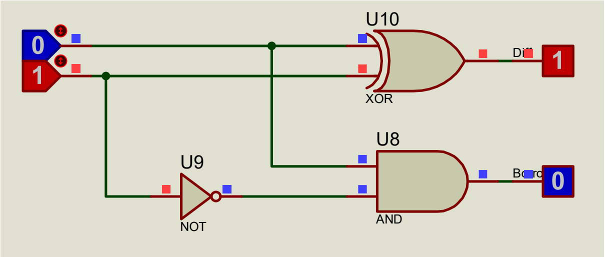

Figure 17: Half-Subtractor

Inputs		Outputs	
A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

A B	0	1
0		1
1	1	

Diff

A B	0	1
0	0	0
1	1	0

Carry

After implementing the k-map on this truth table the functions of the sum and the carry were:

Carry = AB = A AND B

Diff = AB’ + A’B = AXOR B

B. Full-Subtractor

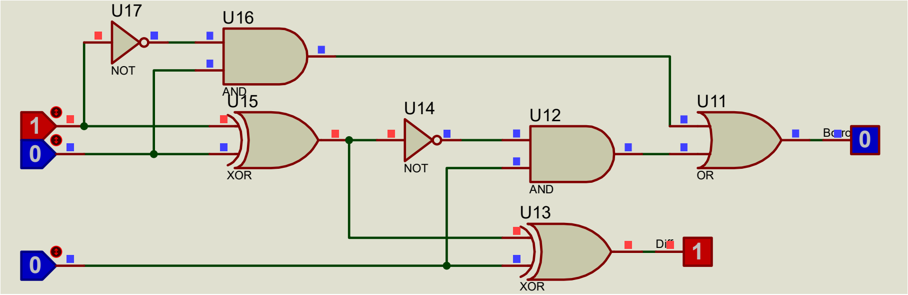

Figure 18: Full-Subtractor

Inputs			Outputs	
A	B	C_{in}	Diff	B_{orrow}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

A BC_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Sum

A BC_{in}	00	01	11	10
0	0	1	1	1
1	0	0	1	0

Carry

After implementing the k-map on this truth table the functions of the sum and the carry were:

Borrow = A’C_{in} + A’B + BC_{in}

Diff= A’B’C_{in} + AB’C_{in}’ + A’BC_{in} + A’BC_{in}

Conclusion:

The experiment went smoothly with no complications, it took me about hour and a half to finish it, and the results satisfied the theory part of the report, the experiment helped with understanding more about comparators and adders and subtractors, but I couldn’t solve the post lab on my own I was able to write the truth table and the k-maps but I couldn’t build a proper design.

Text size: 12

References:

https://www.geeksforgeeks.org/full-adder-in-digital-logic/

https://www.electronics-tutorials.ws/combination/comb_8.html

Digital Lab manual