SWITCHING CIRCUITS AND LOGIC DESIGN
ASSIGNMENT 1

GROUP 17

Hardik Soni
20CS30023

Shivam Raj
20CS10056

Kushaz Sehgal
20CS30030

Jay Kumar Thakur
20CS30024

September 18, 2024

Contents

Chapter 1

Part 1(BCD)

1.1 Problem Statement

Develop circuits to convert from 4-bit binary to 2-digit BCD

1.2 Solution Description

Below is the Explanation of how we designed a 4-bit binary to 2-digit BCD circuit.

1.3 Truth Table

The Input is a 4-bit binary code(A B C D) so

$$N = 16(2^4) \tag{1.1}$$

Combinations. Hence the Output should have 8-bit, but the first three will all be 0 for all combinations of input, the output may be treated as a 5-bit BCD (A B C D E). Here we are considering only 2-bit BCD (A B).

Binary Code (Input)				BCD Code (Output)				
A	B	C	D	V	W	X	Y	Z
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	0
0	0	1	1	0	0	0	1	1
0	1	0	0	0	0	1	0	0
0	1	0	1	0	0	1	0	1
0	1	1	0	0	0	1	1	0
0	1	1	1	0	0	1	1	1
1	0	0	0	0	1	0	0	0
1	0	0	1	0	1	0	0	1
1	0	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	1
1	1	0	0	1	0	0	1	0
1	1	0	1	1	0	0	1	1
1	1	1	0	1	0	1	0	0
1	1	1	1	1	0	1	0	1

Table 1.1: **BCD to Binary Converter**

1.4 Logic Expression

For V:

$$V = A\bar{B}C\bar{D} + A\bar{B}CD + AB\bar{C}\bar{D} + AB\bar{C}D + ABC\bar{D} + ABCD$$

$$V = A\bar{B}C(\bar{D} + D) + AB\bar{C}(\bar{D} + D) + ABC(\bar{D} + D)$$

$$V = A\bar{B}C + AB\bar{C} + ABC$$

$$V = A\bar{B}C + AB(\bar{C} + C)$$

$$V = A\bar{B}C + AB$$

$$V = A(\bar{B}C + B)$$

$$V = A(B + C)$$

$$V = AB + AC \tag{1.2}$$

For W:

$$W = A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D$$

$$W = A\bar{B}\bar{C}(\bar{D} + D)$$

$$W = A\bar{B}\bar{C}$$

$$W = A\bar{B}\bar{C} \tag{1.3}$$

For X:

$$X = \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D + \bar{A}BC\bar{D} + \bar{A}BCD + ABC\bar{D} + ABCD$$
$$X = \bar{A}B\bar{C}(\bar{D} + D) + \bar{A}BC(\bar{D} + D) + ABC(\bar{D} + D)$$
$$X = \bar{A}B\bar{C} + \bar{A}BC + ABC$$
$$X = \bar{A}B + ABC$$
$$X = \bar{A}B + BC \tag{1.4}$$

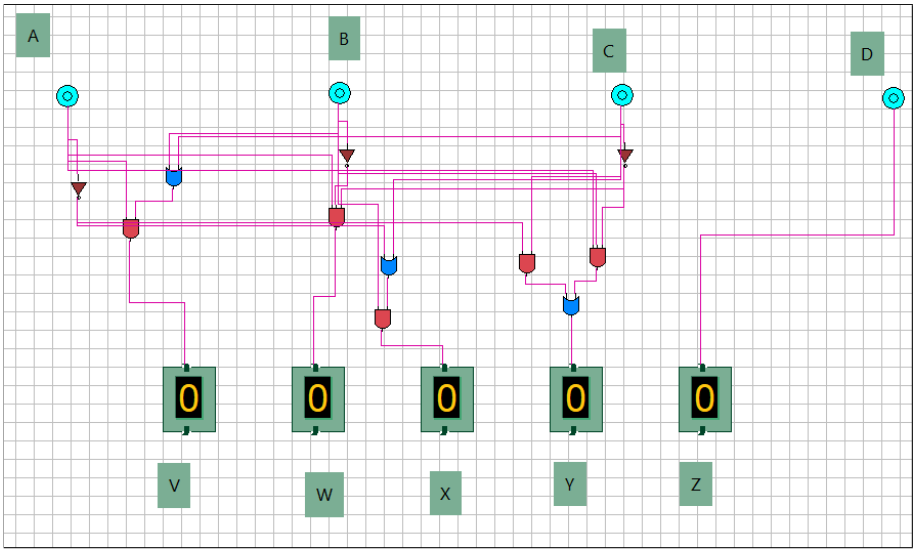
For Y:

$$Y = \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}CD + \bar{A}BC\bar{D} + \bar{A}BCD + AB\bar{C}D + AB\bar{C}\bar{D}$$
$$Y = \bar{A}\bar{B}C + \bar{A}BC + AB\bar{C}$$
$$Y = \bar{A}C + AB\bar{C} \tag{1.5}$$

For Z:

$$Z = D \tag{1.6}$$

1.5 Virtual Lab File



§ 4-bit Binary to 2-digit BCD §

:::: END OF PART 1 ::::

Chapter 2

Part 2(Gray)

2.1 Problem Statement

Develop circuits to convert from 4-bit Gray to 4-bit binary and vice-versa

2.2 Solution Description

Below is the Explanation of how we designed the Circuits to Convert 4-bit Gray to 4-bit Binary and Vice Versa:

2.3 Truth Table

:::::Part 1: 4-bit Gray-Code to 4-bit Binary Code:::::

The Conversion of 4-Bit input Gray Code(A B C D) into the Binary Code Output(X Y Z W) The Input is a 4-bit binary code(X Y Z W) so

$$N = 16(2^4) \tag{2.1}$$

:::::Part 2: 4-bit Binary to 4-bit Gray Code:::::

The Conversion of 4-Bit input Gray Code(A B C D) into the Binary Code Output(X Y Z W) The Input is a 4-bit binary code(X Y Z W) so

$$N = 16(2^4) \tag{2.2}$$

Gray Code (Input)				**Binary (Output)**			
A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	1
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	0	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

Table 2.1: **Gray code to Binary Converter**

Binary Code (Input)				**Gray Code (Output)**			
A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Table 2.2: **Binary to 4-bit Gray Code Converter**

2.4 Logic Expression

:::::Part 1: 4-bit Gray-Code to 4-bit Binary Code:::::

Logical Expression for 4-bit Gray to 4-bit Binary: Simplifying the Truth Table using K-Maps we get:
For W:

$$W = A \tag{2.3}$$

For X:

$$X = A\bar{B} + B\bar{A}$$
$$X = A \oplus B \tag{2.4}$$

For Y:

$$Y = ABC + A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}C$$
$$Y = \bar{C}X + C\bar{X}$$

where

$$X = A \oplus B$$
$$Y = A \oplus B \oplus C \tag{2.5}$$

For Z:

$$Z = \bar{A}\bar{B}\bar{C}D + A\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + \bar{A}BCD + A\bar{B}CD + AB\bar{C}D + ABC\bar{D}$$
$$Z = X\bar{Y} + \bar{X}Y$$

where

$$X = A \oplus B$$

and

$$Y = C \oplus D$$
$$Z = A \oplus B \oplus C \oplus D \tag{2.6}$$

:::::Part 2: 4-bit Binary to 4-bit Gray Code:::::

Logical Expression for 4-bit Binary to 4-bit Gray: Simplifying the Truth Table using K-Maps we get:
For X:

$$W = A \tag{2.7}$$

For Y:

$$X = A \oplus B \tag{2.8}$$

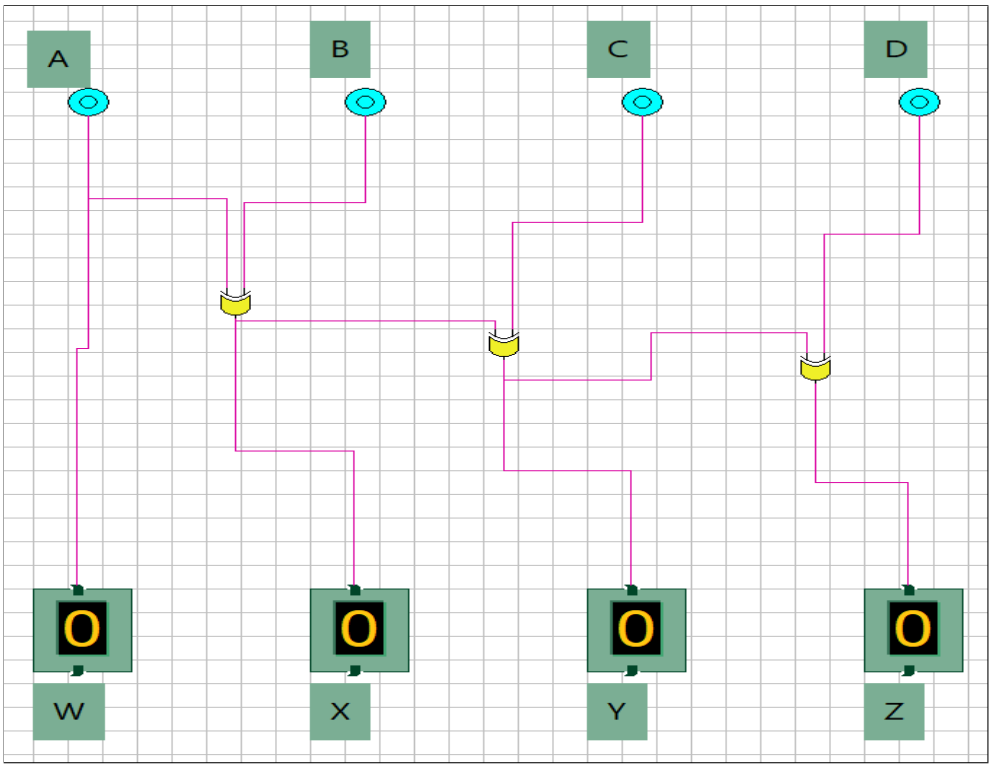
For Z:

$$Y = B \oplus C \tag{2.9}$$

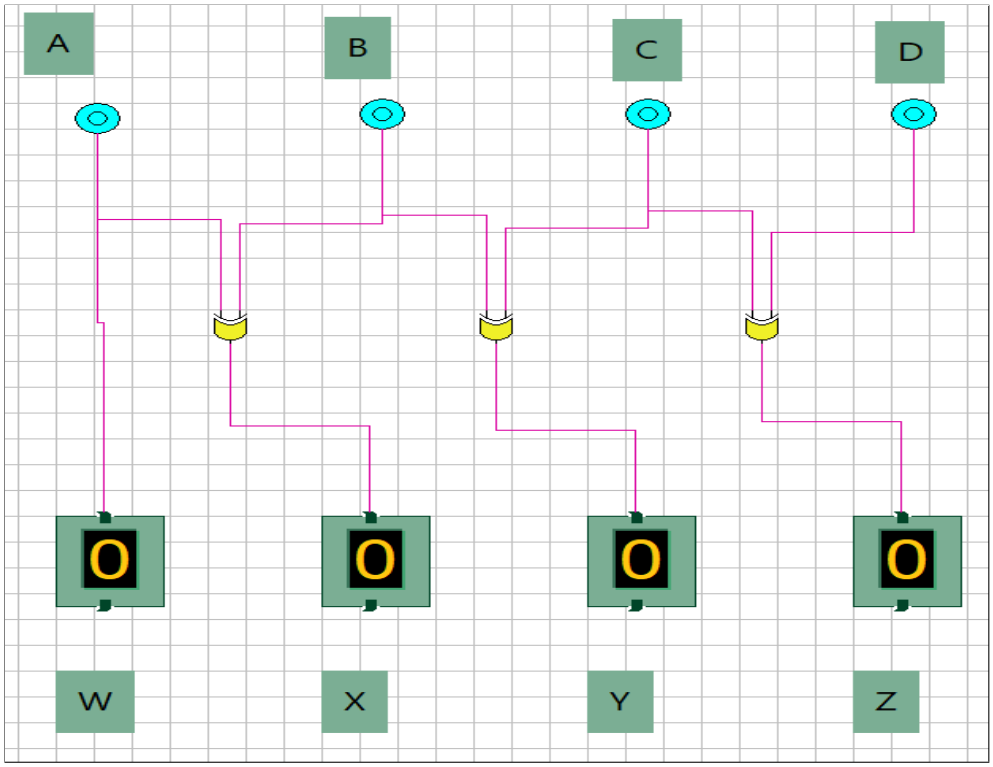
For W:

$$Z = C \oplus D \tag{2.10}$$

2.5 Virtual Lab File



§ 4-bit Gray to 4-digit Binary §



§ 4-digit Binary to 4-digit Gray §

:::: END OF PART 2 ::::

Chapter 3

Part 2(Excess-3)

3.1 Problem Statement

- Develop a half adder for handling two bits
- Develop a full adder using half adders and any additional logic
- Develop a ripple carry adder needed for this assignment using full adders
- Develop circuits to convert from excess-3 to 4-bit binary and vice-versa

3.2 Solution Description

Below is the Explanation of how we designed half adder for handling two bits, full adder using half adders, ripple carry adder using full adders, excess-3 to 4-bit binary and vice versa.

3.3 Truth Table

:::::Part 1: Half Adder for Handling Two Bits:::::

Logical Expression for Half Adder for Handling two Bits: The truth table in X shows that the outputs S and C are simply binary functions on X and Y. This circuit can be designed and implemented in as shown in Figure below.

Input		Output	
A	B	Sum	$Carry$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Table 3.1: Half Adder Truth Table

Input			Output	
A	B	Cin	Sum	$Carry$
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 3.2: Full Adder using Half Adder Truth Table

:::::Part 2: Full Adder using Half Adders :::::

Logical Expression for Full Adder using Half Adders: A Full Adder can also be implemented using two half adders and one OR gate.The Truth Table is shown in Table 3.2. This circuit can be designed and implemented in as shown in Figure below.

:::::Part 3: Ripple Carry Adder using Full Adders:::::

Logical Expression for Ripple Carry Adder using Full Adders: Ripple Carry Adder is a combinational logic circuit. It is used for the purpose of adding two n-bit binary numbers. It requires n full adders in its circuit for adding two n-bit binary numbers. However, in our case we will be considering two 4-bit numbers and a carry-in bit.

:::::Part 4: Excess-3 to 4-bit Binary :::::

An excess 3 code, is an excess of three of the binary number. We add 3 to a binary number and represent that in binary form, that will be your excess 3/xs3 code. We need W, X, Y and Z as inputs. We need 7 AND Gates, 3 NOT Gates and 3 OR Gates.

Excess-3 Code (Output)				Binary Code (Input)			
A	B	C	D	W	X	Y	Z
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1
1	1	0	1	1	0	1	0
1	1	1	0	1	0	1	1
1	1	1	1	1	1	0	0

Table 3.3: **Excess-3 to Binary Converter**

:::::Part 5: 4-bit Binary to Excess-3 :::::

The Conversion of 4-Bit input Binary(A B C D) into the Excess-3 Output(X Y Z W). The Input to the Circuit is: A, B, C, D. The Components we need are: 4 AND gates, 3 NOT gates, and 4 OR Gates.

3.4 Logic Expression

:::::Part 1: Half Adder for Handling Two Bits:::::
The result of the Sum has a Sum Component, the value placed at the resultant bit: The Input is X and Y:

$$S = X \oplus Y \tag{3.1}$$

The Carry Forward Bit for the half Adder Circuit is:

$$C = X.Y \tag{3.2}$$

:::::Part 2: Full Adder using Half Adders :::::
The Input for the Full Adder Circuit using Half Adders is X,Y and the Carry-In Bit:- c
The Carry-Out Bit will be:

$$C = P.Q + P.c + Q.c \tag{3.3}$$

Binary Code (Input)				Excess-3 Code (Output)			
A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	1	1	0	1
1	0	1	1	1	1	1	0
1	1	0	0	1	1	1	1
1	1	0	1	x	x	x	x
1	1	1	0	x	x	x	x
1	1	1	1	x	x	x	x

Table 3.4: **Binary to Excess-3 Converter**

The Sum Carried will be:

$$Sum = c \oplus (A \oplus B) \tag{3.4}$$

:::::Part 3: Ripple Carry Adder using Full Adders:::::

The Input to the Circuit will be two 4-bit Binary Numbers namely, X and Y. We also need the carry-in bit as Input:(c).
We will need 4 Full Adders as Components:-

* The carry-out obtained from the first full adder is used as carry-in for the second full adder and so on. *

$$S_0 = A_0 \oplus B_0 \oplus c \tag{3.5}$$

$$S_1 = A_1 B_1 \oplus C_0 \tag{3.6}$$

$$S_2 = A_2 B_2 \oplus C_1 \tag{3.7}$$

$$S_3 = A_3 B_3 \oplus C_2 \tag{3.8}$$

$$C_{out} = A_3 B_3 \oplus B_3 C_2 \oplus C_2 A_3 \tag{3.9}$$

:::::Part 4: Excess-3 to 4-bit Binary :::::

Input: W,X,Y,Z:-

$$A_{out} = W.X + W.Y.Z \tag{3.10}$$

$$B_{out} = \bar{X}.\bar{Y} + \bar{X}.\bar{Z} + X.Y.Z \tag{3.11}$$

$$C_{out} = Y \oplus Z \tag{3.12}$$

$$D_{out} = \bar{Z} \tag{3.13}$$

:::::Part 5: 4-bit Binary to Excess-3 :::::

Input:- A,B,C,D

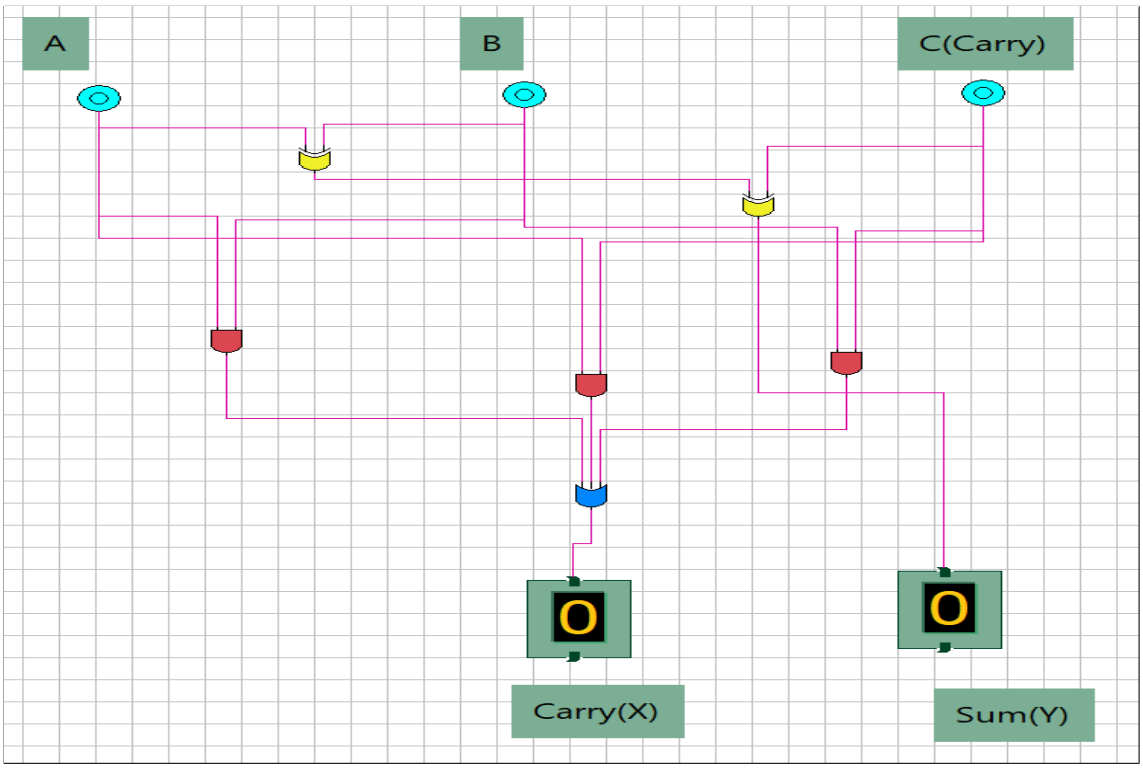
$$X = A + B.C + B.D \tag{3.14}$$

$$Y = \bar{B}C + \bar{B}D + B\bar{C}\bar{D} \tag{3.15}$$

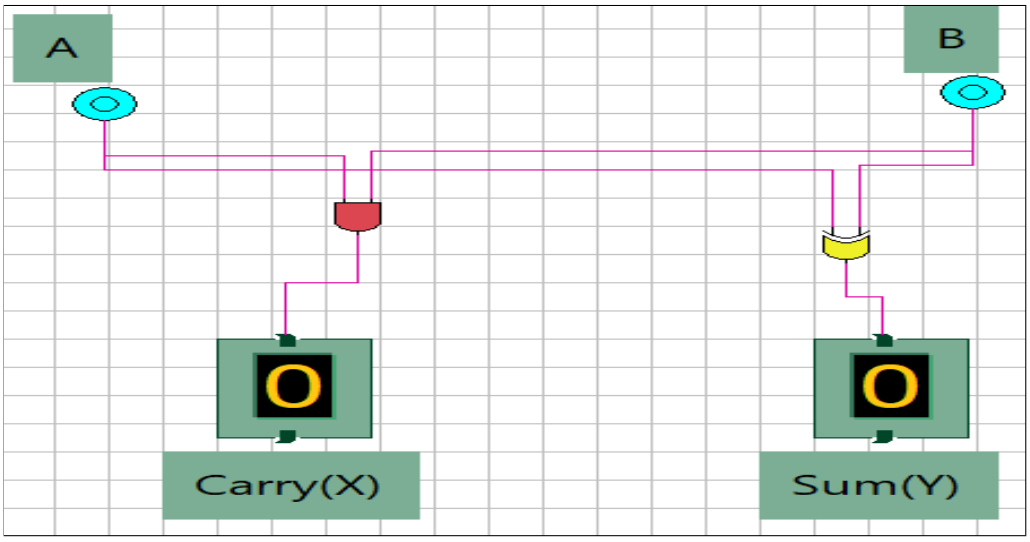
$$Z = CD + \bar{C}\bar{D} \tag{3.16}$$

$$W = \bar{D} \tag{3.17}$$

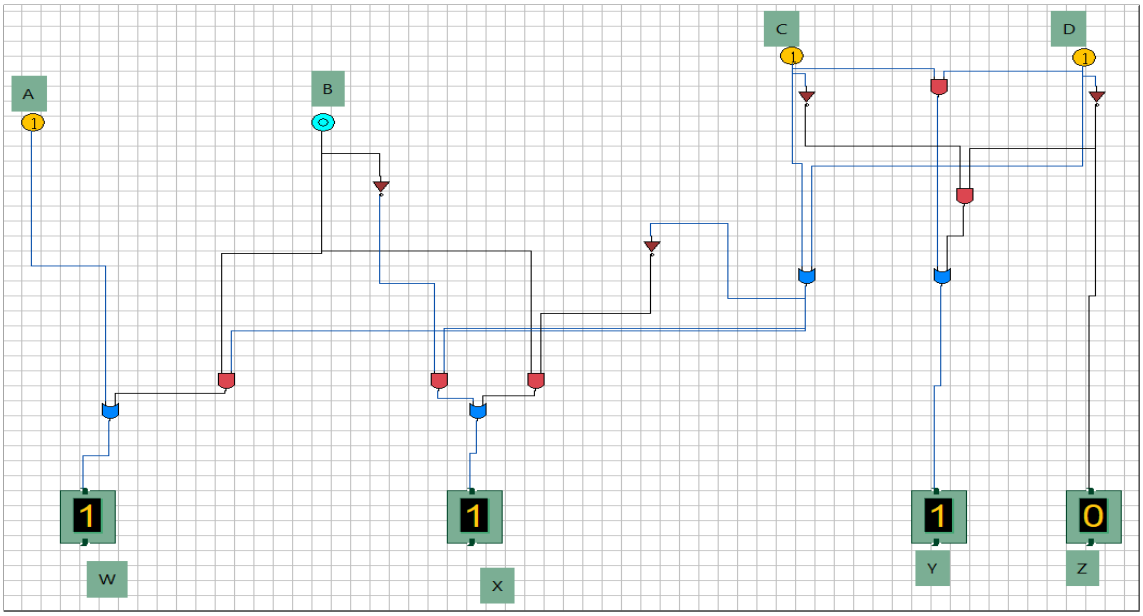
3.5 Virtual Lab File



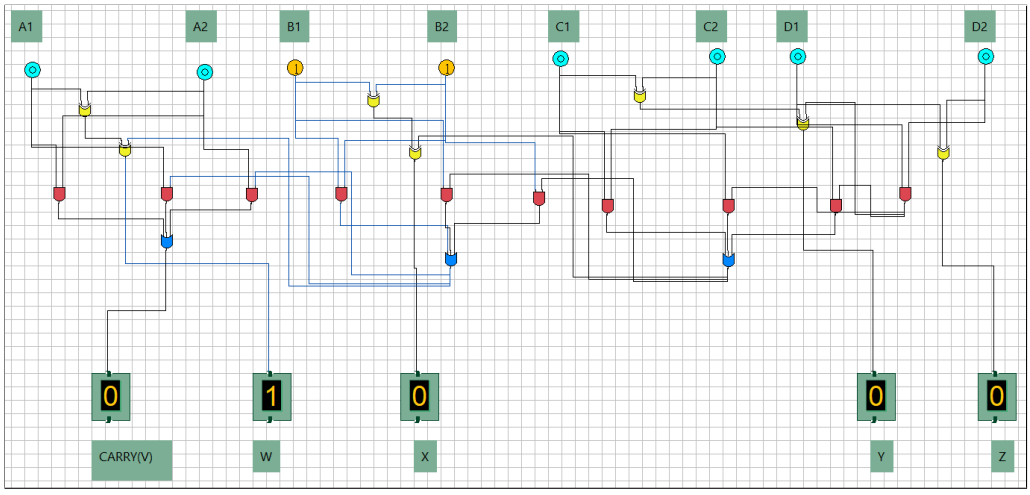
§ Full adder using half adders and any additional logic §



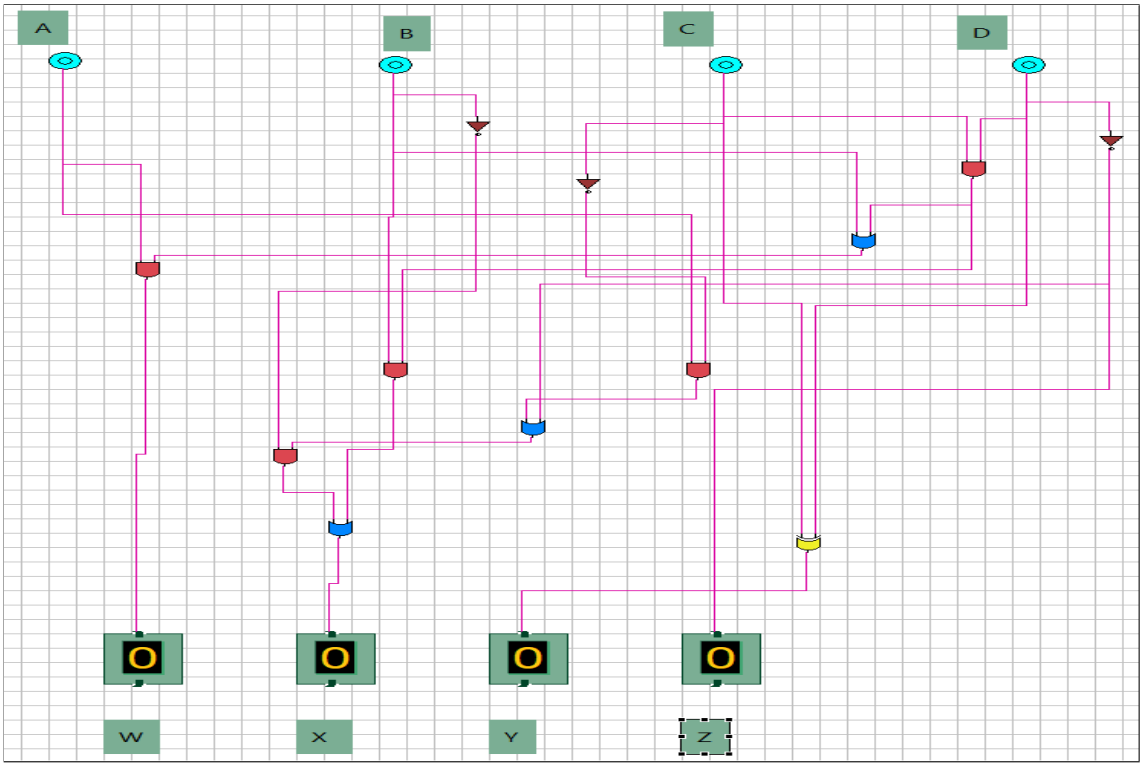
§ Half-Adder for handling two bits§



§ Binary to Excess 3 §



§ Ripple Carry Adder Circuit §



§ Excess-3 to Biniary Converter Circuit§