Indian Institute Of Technology

Kharagpur, West Bengal, India

CS29002 SWITCHING CIRCUITS LABORATORY

LABORATORY REPORT 2

Student Name	Roll Number
1. Aman Sharma	20CS30063
2. Deepsikha Behera	20CS10023
3. Umika Agrawal	20CS30056

Group Number: 2

Submission Date : 03/02/2022

1 Problem Statement (BCD)

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

Solution

- The input in this is a 4-bit binary code and it needs to be converted to a 2 digit, i.e. 8 bit BCD code.
- As the range of a 4 bit binary code is from 0 to 15, the BCD output would be of the form $000B_4B_3B_2B_1B_0$
- In this solution, we'll determine the values of B_4, B_3, B_2, B_1 and B_0.

1. **Truth Table**
   The input is a 4-bit binary code, so the input has 16 possible combinations. Hence, the output should have 8-bit, but because the first three bit will all be 0 for all combinations of inputs, the output can be treated as 5-bit BCD code(B_4 B_3 B_2 B_1 B_0). The conversion of binary code into BCD code as shown as follows:

Decimal	Binary code				BCD Code				
	A	B	C	D	B_4	B_3	B_2	B_1	B_0
0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	1
2	0	0	1	0	0	0	0	1	0
3	0	0	1	1	0	0	0	1	1
4	0	1	0	0	0	0	1	0	0
5	0	1	0	1	0	0	1	0	1
6	0	1	1	0	0	0	1	1	0
7	0	1	1	1	0	0	1	1	1
8	1	0	0	0	0	1	0	0	0
9	1	0	0	1	0	1	0	0	1
10	1	0	1	0	1	0	0	0	0
11	1	0	1	1	1	0	0	0	1
12	1	1	0	0	1	0	0	1	0
13	1	1	0	1	1	0	0	1	1
14	1	1	1	0	1	0	1	0	0
15	1	1	1	1	1	0	1	0	1

2. Analysis

- **$\mathbf{B_0}$**
  As B_0 is the 1st bit of the ones digit, it would be the same as the first bit of the input.
  $B_0 = D$
- **$\mathbf{B_1}$**
  From the truth table above, we can write the expression for B_1 as follows:

$$B_1 = \overline{AB}C\overline{D} + \overline{AB}CD + \overline{A}BC\overline{D} + AB\overline{CD} + AB\overline{C}D$$
$$B_1 = \overline{AB}C(\overline{D} + D) + \overline{A}BC(\overline{D} + D) + AB\overline{C}(\overline{D} + D)$$

  As

$$\overline{D} + D = 1$$
$$B_1 = \overline{AB}C + \overline{A}BC + AB\overline{C}$$
$$B_1 = AB\overline{C} + \overline{A}C(\overline{B} + B)$$

$$B_1 = AB\overline{C} + \overline{A}C$$

- **$\mathbf{B_2}$**
  From the truth table above, we can write the expression for Y as follows:

$$B_2 = \overline{A}B\overline{CD} + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + ABC\overline{D} + ABCD$$
$$B_2 = \overline{A}B\overline{C}(\overline{D} + D) + \overline{A}BC(\overline{D} + D) + ABC(\overline{D} + D)$$

  As

$$\overline{D} + D = 1$$
$$B_2 = \overline{A}B\overline{C} + \overline{A}BC + ABC$$
$$B_2 = \overline{A}B(\overline{C} + C) + ABC$$
$$B_2 = \overline{A}B + ABC$$
$$B_2 = B(\overline{A} + AC)$$

  As

$$A{+}BC = (A{+}B)(A{+}C)$$
$$B_2 = B(\overline{A} + A)(\overline{A} + C)$$
$$B_2 = B(\overline{A} + C)$$
$$B_2 = \overline{A}B + CB$$

- **$\mathbf{B_3}$**
  From the truth table above, we can write the expression for X as follows:

$$B_3 = A\overline{BCD} + A\overline{BC}D$$
$$B_3 = A\overline{BC}(\overline{D} + D)$$

  As

$$\overline{D} + D = 1$$
$$B_3 = A\overline{BC}$$

- **B_4**
  From the truth table above, we can write the expression for W as follows:

$$B_4 = A\bar{B}C\bar{D} + A\bar{B}CD + AB\bar{C}\bar{D} + AB\bar{C}D + ABC\bar{D} + ABCD$$
$$B_4 = A\bar{B}C(\bar{D} + D) + AB\bar{C}(\bar{D} + D) + ABC(\bar{D} + D)$$

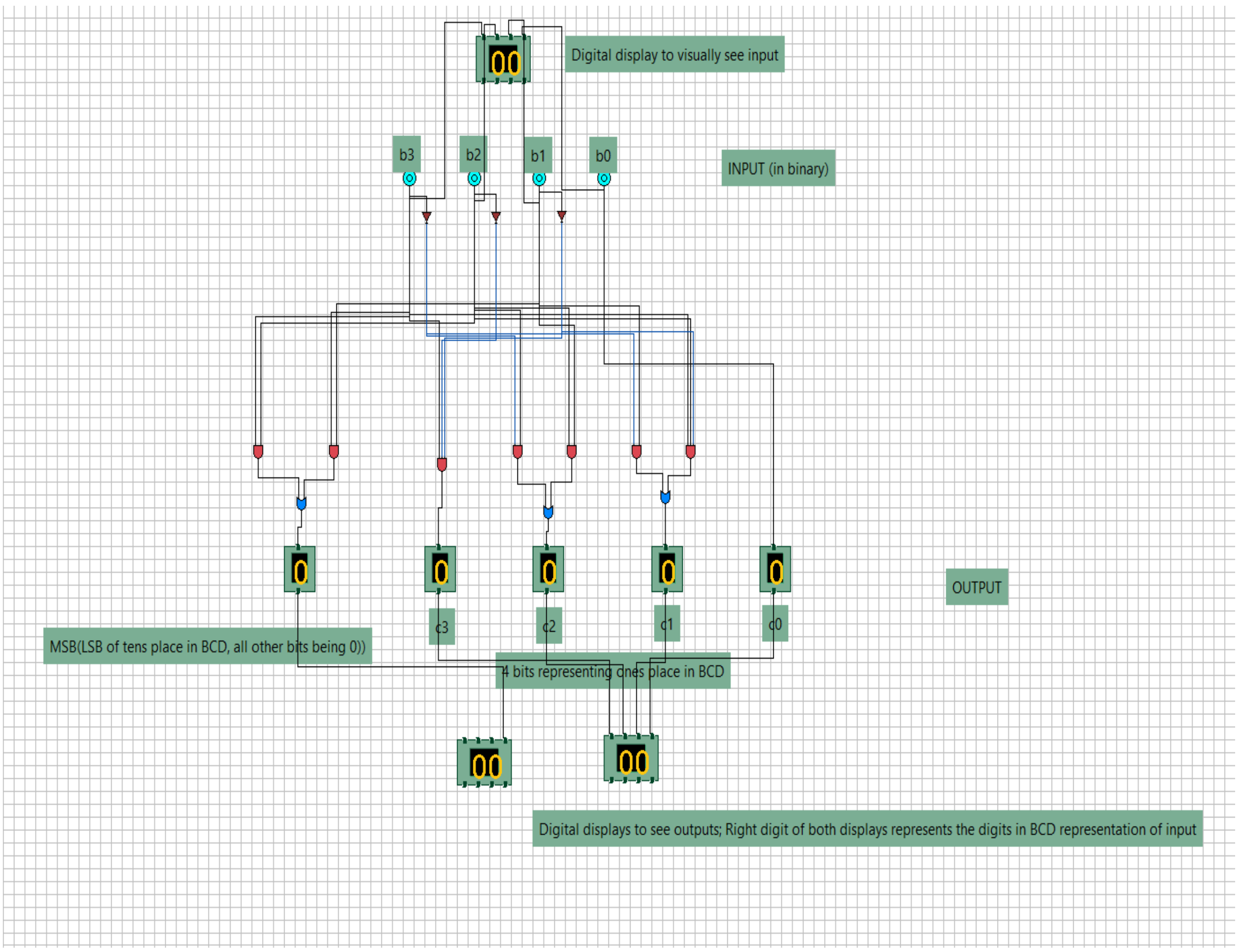
  As

$$\bar{D} + D = 1$$
$$B_4 = A\bar{B}C + AB\bar{C} + ABC$$
$$B_4 = A\bar{B}C + AB(\bar{C} + C)$$
$$B_4 = A(\bar{B}C + B)$$

  As

$$A{+}BC = (A{+}B)(A{+}C)$$
$$B_4 = A(\bar{B} + B)(B + C)$$
$$B_4 = A(B{+}C)$$
$$B_4 = AB + AC$$

3. Circuit Diagram



2 Problem Statement (Gray)

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

1. **Truth Table**

Decimal numbers	Binary code				Gray Code			
	B_1	B_2	B_3	B_4	G_1	G_2	G_3	G_4
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

2. **Binary to Gray:**

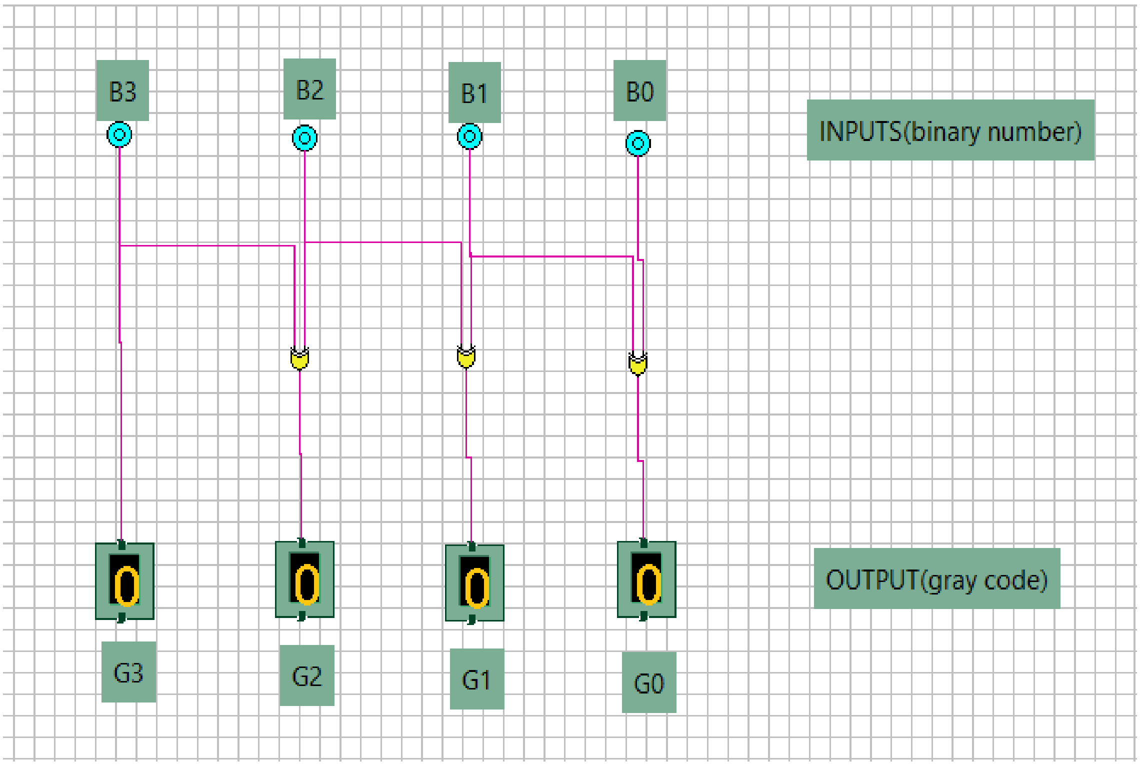
Theory:

- Gray code is made such that the consecutive numbers differ in just 1 bit.
- The last bit from the right (MSB) is always the same in gray as the binary code.

- The further bits(n^{th} bit) are obtained by the method that nth bit of gray is the XOR of n^{th} and $(n+1)^{th}$ bit of the binary code.

$$G_i = B_i \text{ for } i{=}n$$
$$G_i = B_i \text{ XOR } B_{i+1} \text{ for } i \in [0, n-1]$$

Circuit Diagram:



3. **Gray to Binary:**

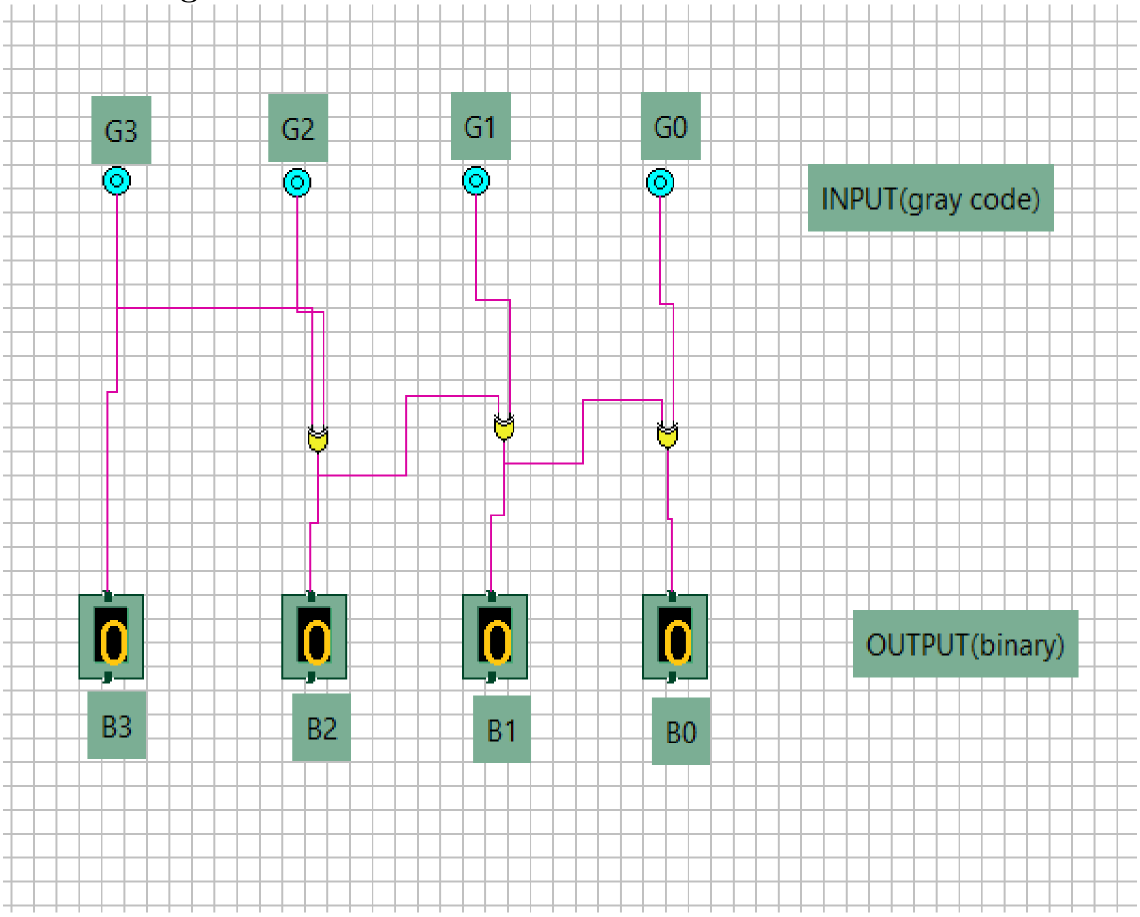
Theory:

- The MSB of the binary number is the same as the MSB of the gray code.
- The 2^{nd} bit(from the left) of the binary number is the same as the 1^{st} bit of the binary number if the 2^{nd} bit of the Gray code is 0. Else, the 2^{nd} bit is the opposite of the 1^{st} bit of the binary number.

- The same follows for all the further bits of the binary number(B_i and G_i are the i^{th} bits from the right)

$$B_i = G_i \text{ for } i = n$$
$$B_i = G_i \text{ XOR } B_{i+1} \text{ for } i \in [0, n-1]$$

Circuit Diagram:



3 Problem Statement (Excess-3)

- 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.

Solution

- ## Half Adder

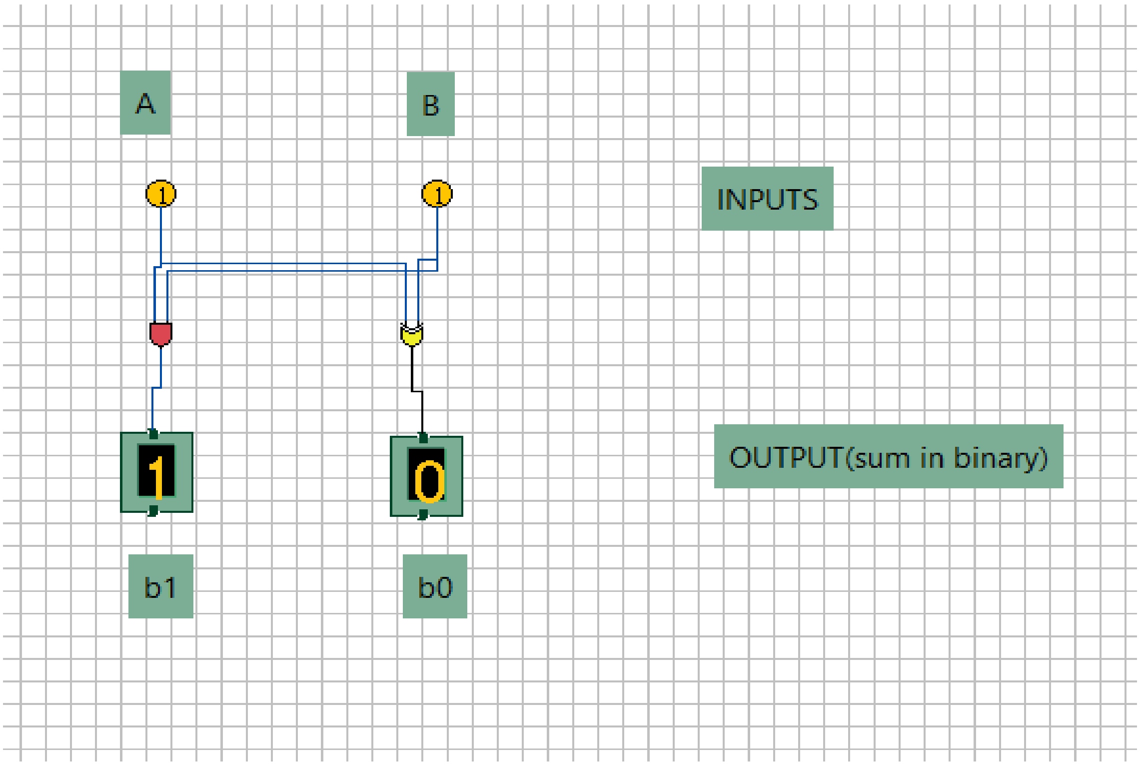
  A half adder is a logical circuit which finds the sum of two binary digits, and outputs a Sum and a Carry value.

  Truth Table

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

Sum = A XOR B
Carry = A AND B

Circuit Diagram



- ## Full Adder

A full adder is a logical circuit that finds the sum of three binary digits, and outputs a SUM and a Carry-out(C-OUT) value. The first two inputs are A and B, and the third input is C-IN.

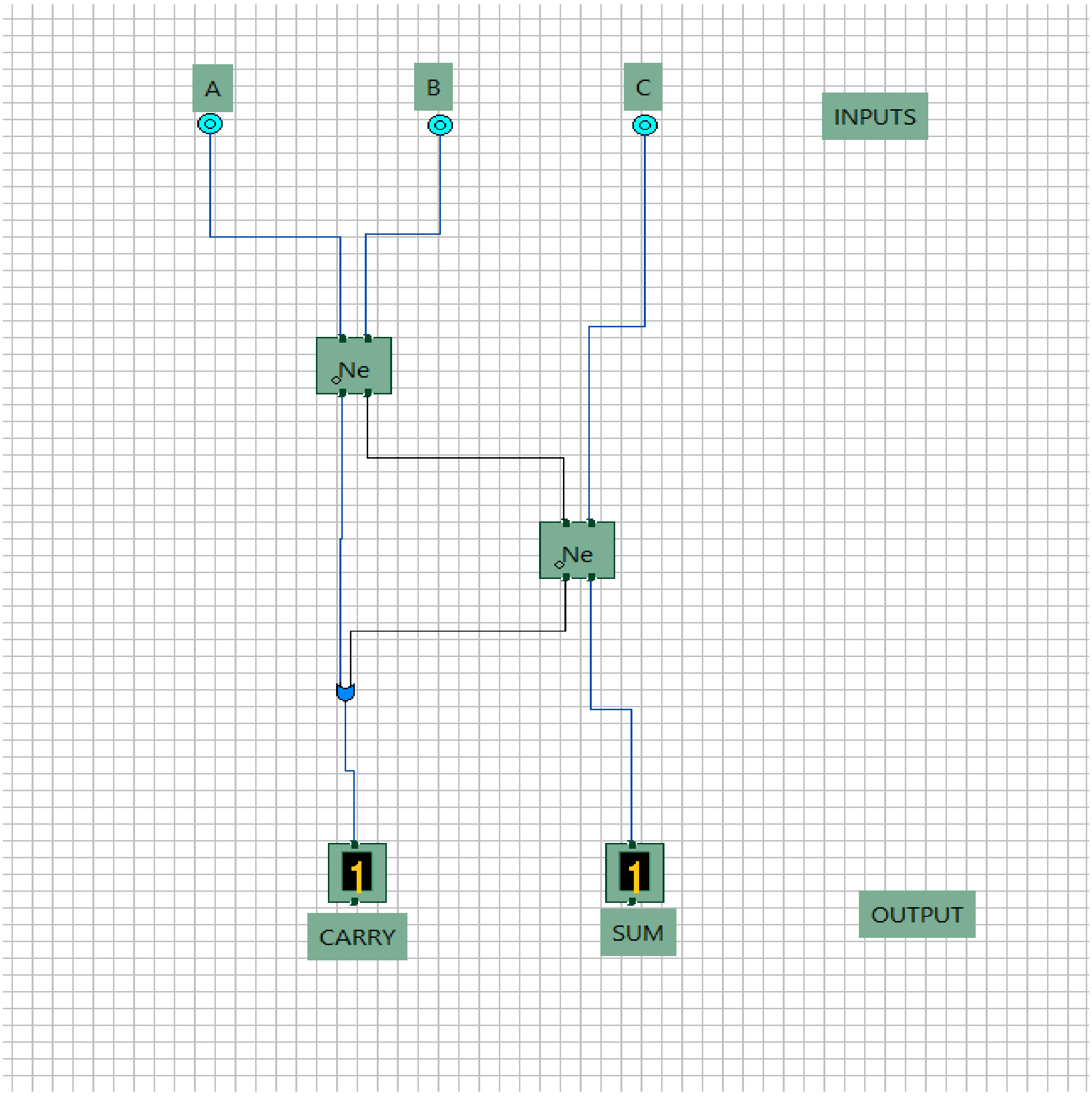
Truth Table

Input			Output	
A	B	C(Cin)	Sum	Carry(Cout)
0	0	0	0	0
1	0	0	1	0
0	1	0	1	0
0	0	1	1	0
1	1	0	0	1
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

Logical Expression for SUM:
$= \overline{AB}\text{C-IN} + \overline{A}B\overline{\text{C-IN}} + A\overline{B\text{C-IN}}$
$= \text{C-IN}(\overline{AB} + AB) + \overline{\text{C-IN}}(\overline{A}B + A\overline{B})$
$= \text{C-IN XOR (A XOR B)}$

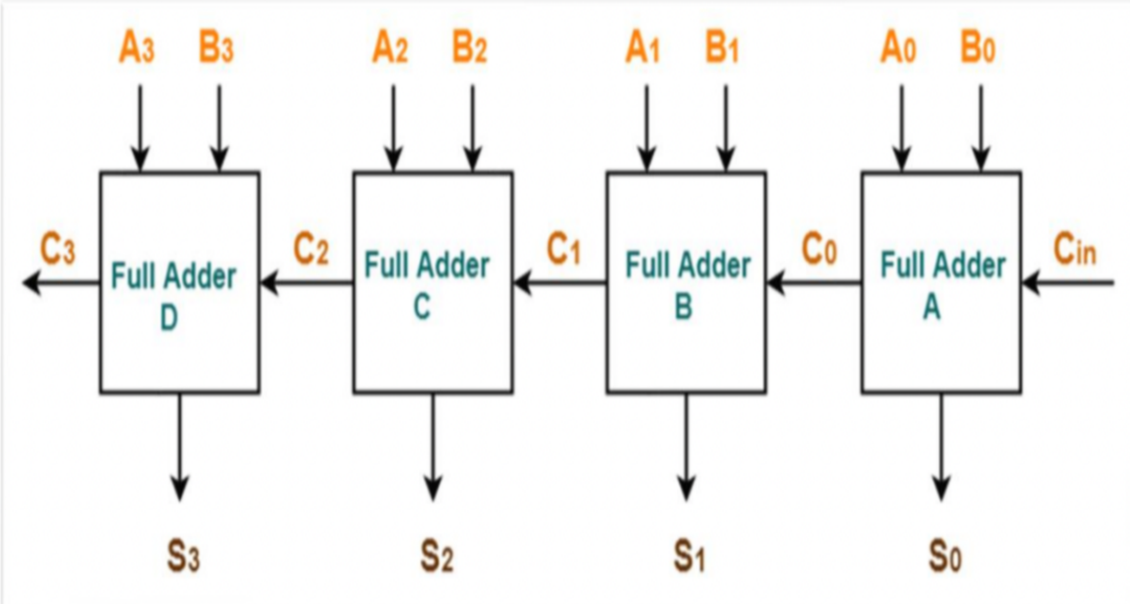
Logical Expression for C-OUT:
$= \overline{A}B\text{C-IN} + A\overline{B}\text{C-IN} + AB\overline{\text{C-IN}} + AB\text{C-IN}$
$= AB + B\text{C-IN} + A\text{C-IN}$

Circuit Diagram



• Ripple Carry Adder

A ripple carry adder is a logical circuit which finds the sum of two n-bit binary numbers. It is made by using n full adders. Each full adder acts as a single weighted column in a long binary addition.

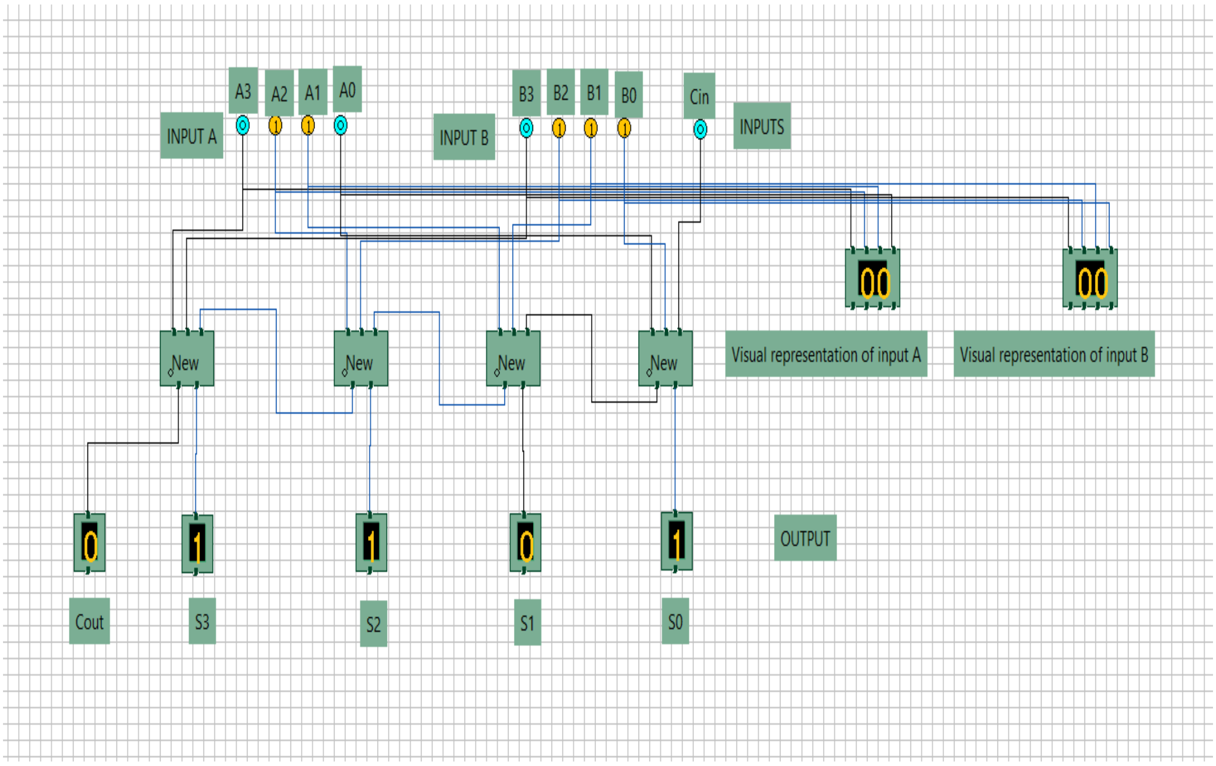


The minimised expression for each output using k-map is:

- $S_0 = A_0 \oplus B_0 \oplus C_{in}$
- $C_0 = A_0 \cdot B_0 \oplus B_0 \cdot C_{in} \oplus C_{in} \cdot A_0$
- $S_1 = A_1 \oplus B_1 \oplus C_0$
- $C_1 = A_1 \cdot B_1 \oplus B_1 \cdot C_0 \oplus C_0 \cdot A_1$
- $S_2 = A_2 \oplus B_2 \oplus C_1$
- $C_2 = A_2 \cdot B_2 \oplus B_2 \cdot C_1 \oplus C_1 \cdot A_2$
- $S_3 = A_3 \oplus B_3 \oplus C_2$
- $C_3 = A_3 \cdot B_3 \oplus B_3 \cdot C_2 \oplus C_2 \cdot A_3$

However, using full adders we just provide the Carry out of the last adder to the next adder as Carry in. Thus, we get an n-bit adder using n full adders.

Circuit Diagram

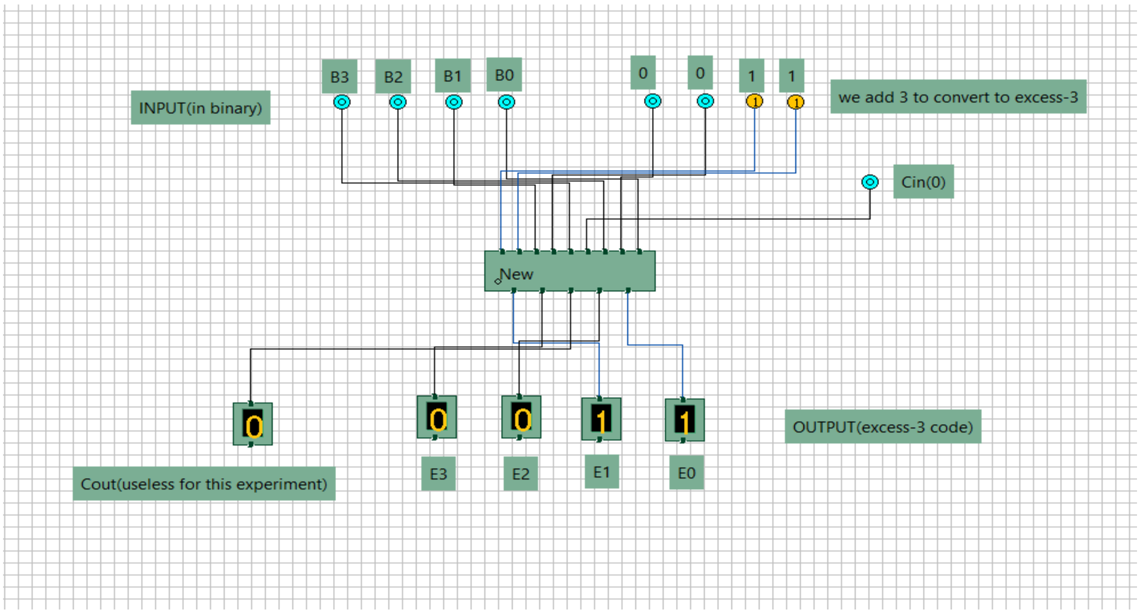


- ## Binary to Excess-3 code

  In Excess-3 code, each digit of the decimal number is represented by adding 3 to it.
  To convert from binary code to excess-3 code:

  - Convert Binary to decimal
  - Add 3 to every digit
  - Find binary representation of each digit
  - We can also convert Binary to BCD first, and add 0011 to each digit in the BCD code.

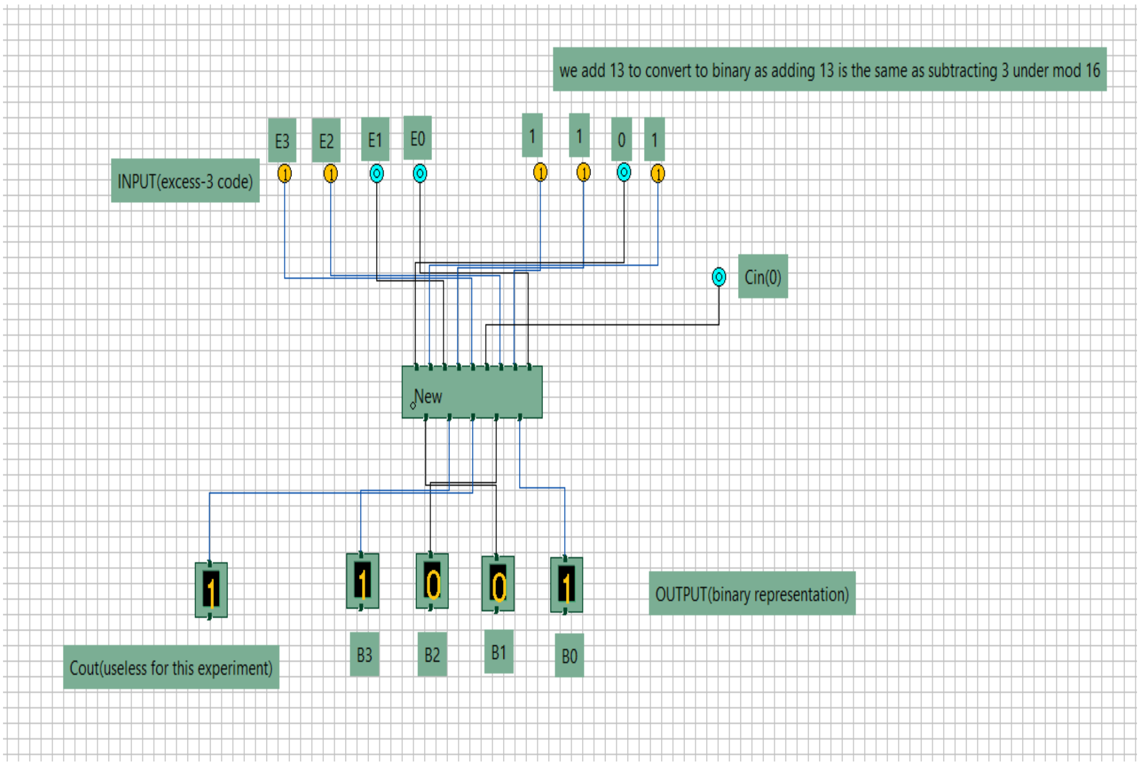
Circuit Diagram



- **Excess-3 to Binary code**

  - Convert Excess-3 to BCD(by subtracting 0011) from every digit.
  - Convert the BCD code to Binary code.

Circuit Diagram



Conversion Table between EXCESS-3 code and binary

EXCESS-3 INPUT				BCD OUTPUT			
E3	E2	E1	E0	B3	B2	B1	B0
0	0	0	0	X	X	X	X
0	0	0	1	X	X	X	X
0	0	1	0	X	X	X	X
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	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X