Department of Computer Science and Engineering
BRAC University
CSE 260: Digital Logic Design

Experiment # 1: Familiarization with *Fundamental Logic Gates*

Objective:

- To get familiarized with fundamental logic gates and demonstrate the input-output relationship of 2-input **AND** (IC – 7408), **OR** (IC – 7432) **and NOT/Inverter** (IC – 7404) gates by constructing their truth tables.
- To get familiar with other logic gates like **NAND** (IC – 7400), **NOR** (IC – 7402), **XOR** (IC – 7486) and **XNOR** (IC – 4077)

Required Components:

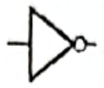
1. IC 7408 × 1
2. IC 7432 x 1
3. IC 7404 × 1
4. IC 7400 x 1
5. IC 7402 × 1
6. IC 7486 x 1
7. IC 4077 x 1

Procedure:

- For each of the ICs, place the IC correctly on the trainer board
- Remember to connect each IC's VCC pin to the "+5V" position of the DC Power Supply of the trainer board, and the GND or 0V pin to the "GND" position of the trainer board.
- Connect the inputs to the data switches and the output to any position on the LED display.
- Find out the outputs for all possible combinations of input states.
- Write down the input-output in tabular form.

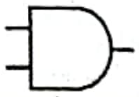
Logic gate symbols with corresponding truth tables:

NOT



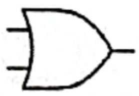
INPUT	OUTPUT
A	
0	1
1	0

AND



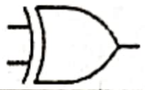
INPUT		OUTPUT
A	B	
0	0	0
1	0	0
0	1	0
1	1	1

OR



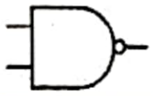
INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	1

XOR



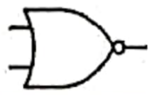
INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	0

NAND



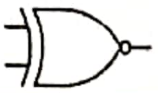
INPUT		OUTPUT
A	B	
0	0	1
1	0	1
0	1	1
1	1	0

NOR



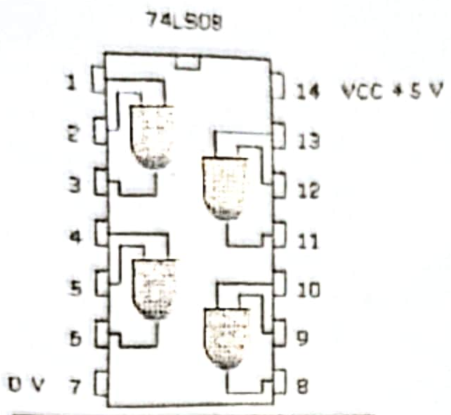
INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	0

XNOR

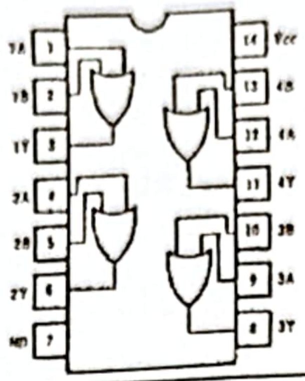


INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	1

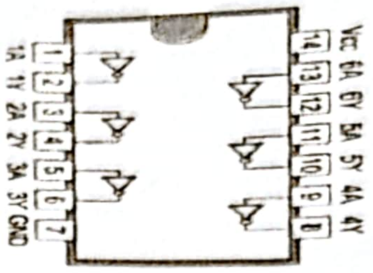
Pin Diagrams of ICs:



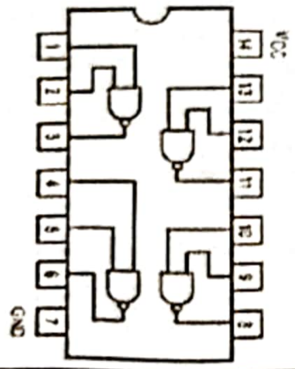
Pin layout of 7408



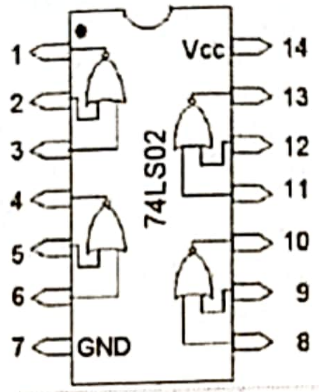
Pin layout of 7432



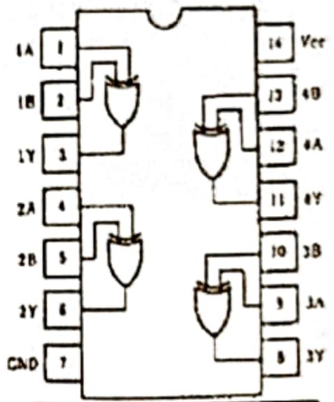
Pin layout of 7404



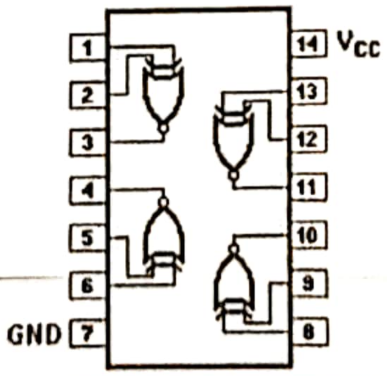
Pin layout of 7400



Pin Layout of 7402



Pin layout of 7486



Pin layout of 4077

Department of Computer Science and Engineering
BRAC University
CSE 260: Digital Logic Design

Experiment # 2: Universal Gates, Applications of Boolean Algebra

Objective:

- To investigate the rules of Boolean algebra.
- To gain experience working with practical circuits.
- To simplify a complex function using Boolean algebra.

Required Components:

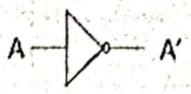
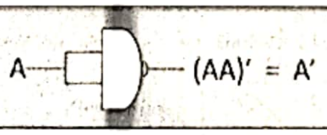
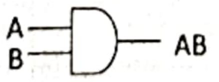
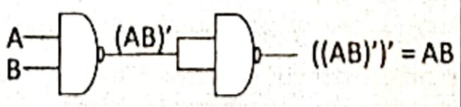
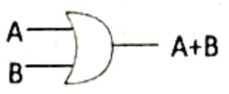
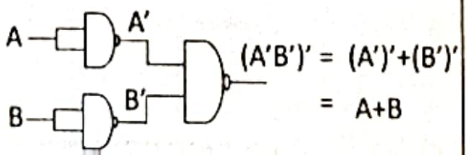
1. IC 7400 × 1
2. IC 7402 x 2

Boolean Postulates & Theorems:

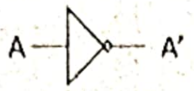
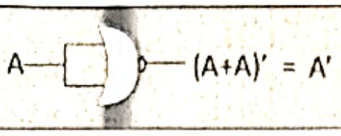
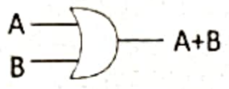
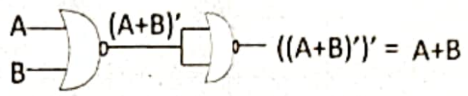
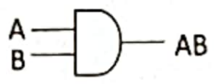
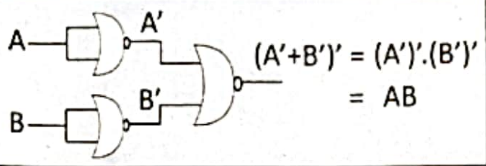
Postulate 2:	(a) x + 0 = x	(b) x . 1 = x
Postulate 3: Commutative	(a) x + y = y + x	(b) x . y = y . x
Postulate 4: Distributive	(a) x(y+z) = xy + xz	(b) x + yz = (x+y).(x+z)
Postulate 5:	(a) x + x' = 1	(b) x . x' = 0
Theorem 1:	(a) x + x = x	(b) x . x = x
Theorem 2:	(a) x + 1 = 1	(b) x . 0 = 0
Theorem 3: Involution	(x')' = x	
Theorem 4: Associative	(a) x+(y+z) = (x+y) +z	(b) x.(y.z) = (x.y).z
Theorem 5: DeMorgan	(a) (x + y)' = x' . y'	(b) (xy)' = x' + y'
Theorem 6: Absorption	(a) x + xy = x	(b) x. (x + y) = x

Diagrams:

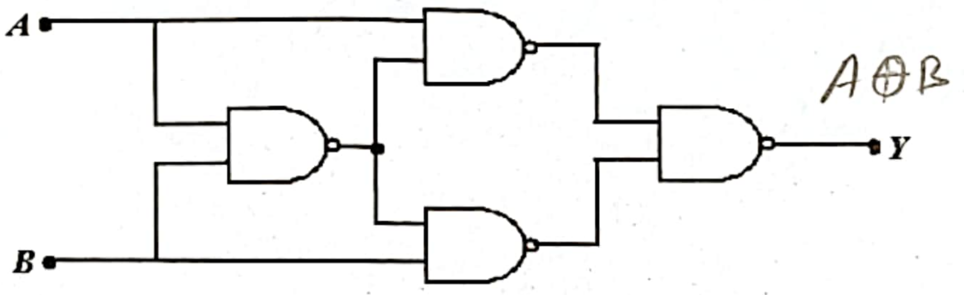
Building basic gates using Universal (NAND) Gate(s):

Not		
And		
Or		

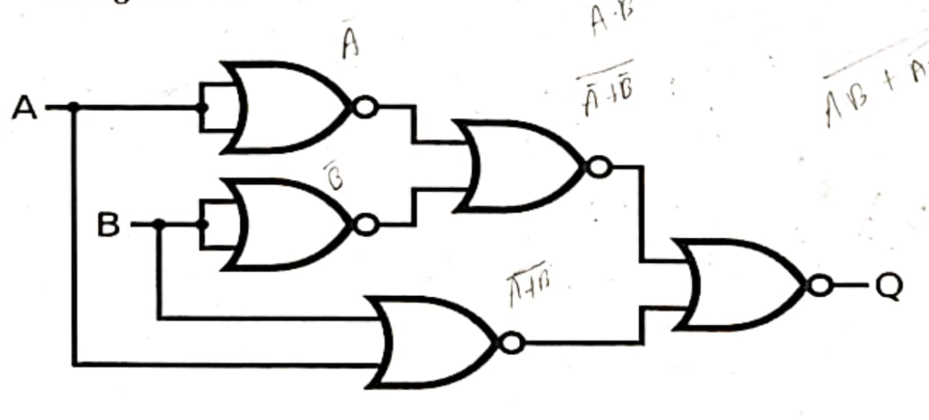
Building basic gates using Universal (NOR) Gate(s):

Not		
Or		
And		

Circuit Diagram - 1:



Circuit Diagram - 2:



$\overline{AB + A+B} = \overline{AB} \cdot (\overline{\overline{A+B}})$

$= \overline{AB}\,(A+B)$

Procedure:

- Construct the Circuit Diagram - 1 on the breadboard.
- Remember to connect each IC's VCC pin to the "+5V" position of the DC Power Supply of the trainer board, and the GND or 0V pin to the "GND" position of the trainer board.
- Connect the inputs to the Data switches and outputs to any position of the LED Display.
- Find out the outputs for all possible combinations of input states.
- Write down the input-output in tabular form.

Department of Computer Science and Engineering
BRAC University
CSE 260: Digital Logic Design

Experiment # 3: Parity Bit Checker and Generator

Objective:

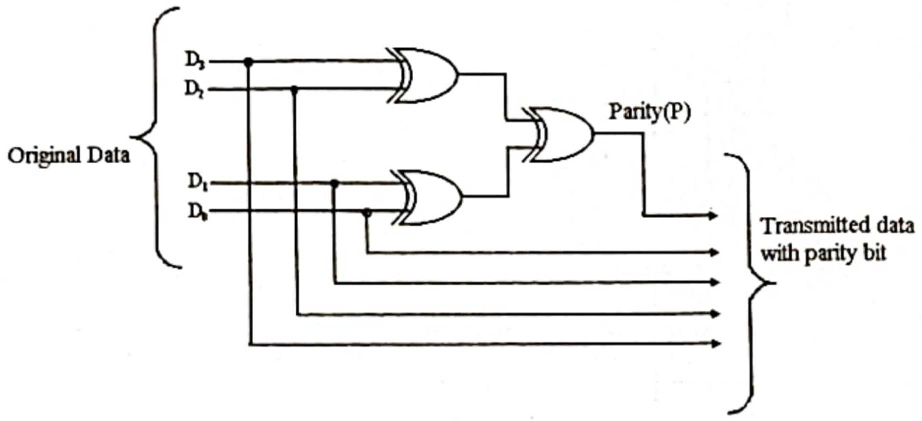
- To design and implement an even parity Generator and even parity checker using XOR gates. (IC-7486).

Required Components:

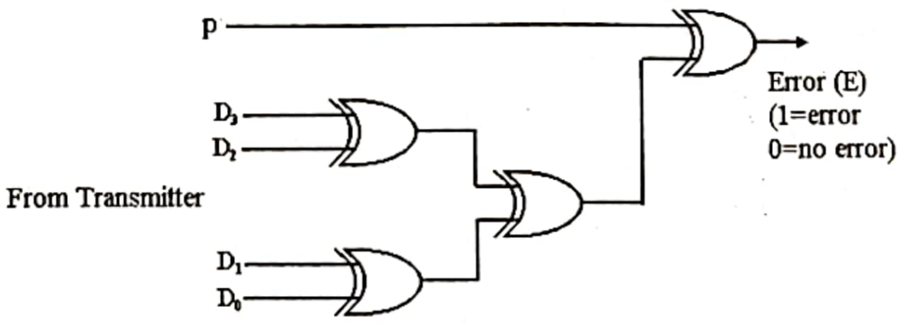
1. IC 7486 × 1

Diagrams:

Building an even parity generator using XOR Gates for 4-bit Data:



Building an even parity checker using XOR Gates for 4-bit Data:



Procedure:

- Construct the Circuits of both the parity bit generator and checker on the breadboard.
- Remember that each IC's pin 14 is connected to the "+5V" position of the DC Power Supply of AT-700, and pin 7 is connected to the "GND" position.
- Connect the inputs to the Data switches and outputs to any position of the LED Display.

Result:

Complete both of the following truth tables.

Truth Table for Parity Bit Generator:

	Data				Parity
	D_3	D_2	D_1	D_0	
a.	1	0	0	1	
b.	0	0	0	1	
c.	1	1	1	1	
d.	0	0	0	0	

Truth Table for Parity Bit Checker:

	Data					Error
	Parity	D_3	D_2	D_1	D_0	
a.	1	1	0	0	1	
b.	0	0	0	0	1	
c.	0	1	1	1	1	
d.	1	0	0	0	0	

Department of Computer Science and Engineering
BRAC University
CSE 260: Digital Logic Design

Experiment # 4: Design and Implementation of 4-bit Parallel Binary Adder

Objective:

- To investigate how the Half adder and Full adder circuits work
- To gain experience working with practical circuits.
- To investigate how the IC 7483(4-bit parallel adder) works

Required Components:

1. IC 7408
2. IC 7432
3. IC 7486
4. IC 7483

Theory:

The addition of two binary numbers is performed in exactly the same manner as the addition of decimal numbers.
Let us first review the decimal addition ***LSB***

```
3    7    6
4    6    1
---------------
8    3    7
```

The least significant digit position is operated on first, producing a sum of 7. The digits in the second position are then added to produce a sum of 13, which produces a ***carry*** *of 1 into the third position. This produces a sum of 8 in the third position.*

The same general steps are followed in binary addition. However, only four cases can occur in adding the two binary digits (bits) in any position. They are

0+0=0
1+0=1
1+1=10=0+carry of 1 into the next position
1+1+1=11=1+carry of 1 into the next position

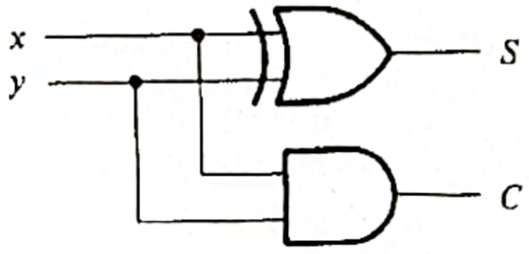
Here are several examples of the addition of two binary numbers:

```
 1001       1101
 1111       0110
-------    --------
11000      10011
```

Half Adder Circuit:

A half adder is a combinational circuit that forms the arithmetic sum of two input bits. It consists of two inputs and two outputs. Two of the input variables, denoted by x and y represent the two significant bits to be added. The two outputs are designed by the symbols S and C. The binary S gives the value of the least significant bit of the sum. The binary variable C gives the output carry. The truth table of the full adder is as follows:

X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



(e) $S = x \oplus y$

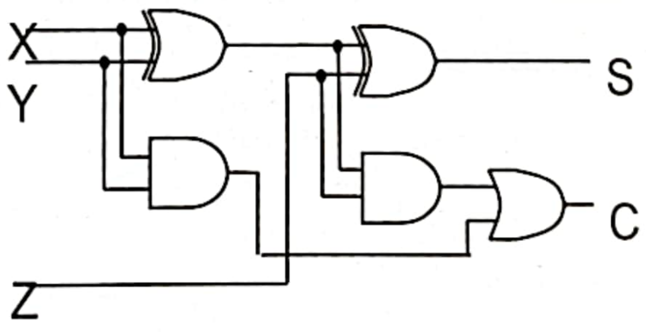
$C = xy$

Figure: Half Adder circuit

Full Adder Circuit:

A full adder is a combinational circuit that forms the arithmetic sum of three input bits. It consists of three inputs and two outputs. Two of the input variables, denoted by x and y represent the two significant bits to be added. The third input z represents the carry from the previous lower significant position. The two outputs are designed by the symbols S and C. The binary S gives the value of the least significant bit of the sum. The binary variable C gives the output carry. The truth table of the full adder is as follows:

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



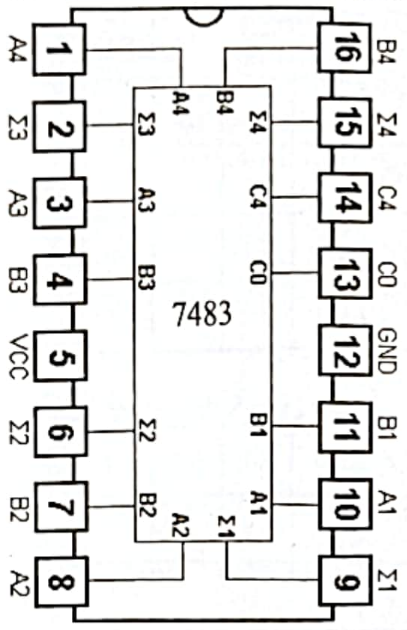
$C = XY + (X \oplus Y)Z$

$S = (X \oplus Y) \oplus Z$

Figure: Full Adder Circuit

A four-bit parallel adder Circuit (IC - 7483):

A binary parallel adder is a digital function that produces the arithmetic sum of two binary numbers in parallel. It consists of full adders connected in cascade, with the output carry from one full adder connected to the input of the next full adder.



A four bit parallel adder cum subtractor:

IC: 7486(XOR) 7483(4bit parallel adder):

To implement addition and subtraction together:

1. B1 xor C_0 , B2 xor C_0, B3 xor C_0 and B4 xor C_0
2. Connect the output from step 1 to the input of the 7483 IC's B inputs.
3. Keep C_0 common for all steps
4. give C_0= 0 to perform addition, C_0=1 to perform subtraction

We use XOR gate as it produces the invert output of one operand when the other operand is equal to 1.

A	B	Output
1	0	1 (invert of B)
1	1	0 (invert of B)
0	1	1
0	0	0

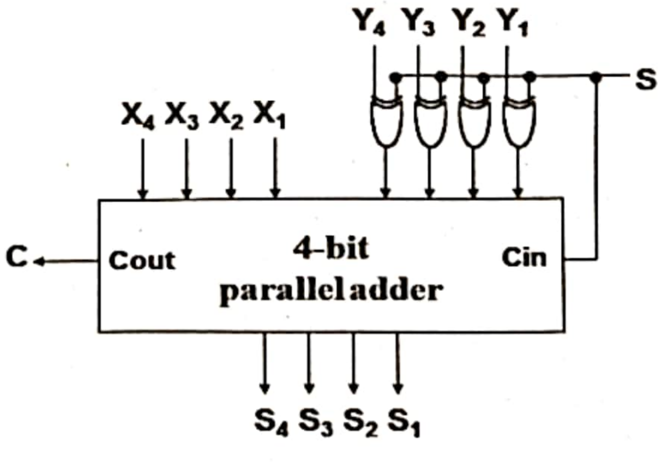


Figure: 4-bit Parallel Adder cum Subtractor

Result:

Fill in the following tables:

(i) Truth table for 4-bit Parallel Adder:

	A	B	Cin	C4	S4	S3	S2	S1
a.	0000	1000	0	0	1	0	0	0
b.	1100	1101	0	1	1	0	0	1
c.	1110	0011	0	1	0	0	0	1
d.	1111	1111	0	1	1	1	1	0

(ii) Truth table for 4-bit Parallel Adder cum Subtractor:

	A	B	Cin	C4	S4	S4	S2	S1
a.	1100	1000	1					
b.	1100	1101	0					
c.	1110	0011	1					
d.	1111	1111	0					

Department of Computer Science and Engineering
BRAC University
CSE 260: Digital Logic Design

Experiment # 5: Implementation of 4-bit Magnitude Comparator

Objective:

- Designing a 4-bit Magnitude Comparator circuit with proper truth tables.

Required Components:

1. IC 7408
2. IC 7432
3. IC 7404
4. IC 4077

Theory:

The comparison of two numbers is an operation that determines if one number is greater than, less than or equal to the other number. A magnitude comparator is a combinational circuit that compares two numbers A and B and determines their relative magnitudes. The outcome of the comparison is specified by three binary variables that indicate A>B, A=B, or A<B.

The algorithm is a direct application of the procedure a person uses to compare the relative magnitudes of two numbers. Consider the two numbers A, and B, with four digits each. Write the coefficients of the numbers with descending significance as follows:

$$A=A_3A_2A_1A_0$$
$$B=B_3B_2B_1B_0$$

where each subscripted digit represents one of the digits in the number.

The two numbers are equal if all pairs of significant digits are equal i.e., if $A_3=B_3$, $A_2=B_2$, $A_1=B_1$ and $A_0=B_0$. When the numbers are binary the digits are either 1 or 0 and the equality relation of each pair of bits can be expressed logically with an equivalence function:

$$x_i=A_iB_i+A_i'B_i', \quad i=0, 1, 2, 3$$

where $x_i=1$ only if the pair of bits in position i are equal, i.e., if both are 1's or both are 0's.

The equality of the two numbers A and B is displayed in a combinational circuit by an output binary. This binary variable is equal to 1 if the input numbers A and B are equal and it is equal to zero otherwise. For an equality condition to exist, all x_i variables must be equal to 1. This indicates an AND operation of all variables:

$$(A=B)=x_3x_2x_1x_0$$

To demonstrate if A is greater than or less than B, we inspect the relative magnitude of pairs of significant digits starting from the most significant position. If the two digits are equal, we compare the next lower significant pair of digits. This comparison continues until a pair of unequal digits is reached. If the corresponding digit of A is 1 and that of is 0, we conclude that A>B. If the corresponding digit of A is 0 and B is 1, we have that A<B. The sequential comparison can be expressed logically by the following two Boolean functions:

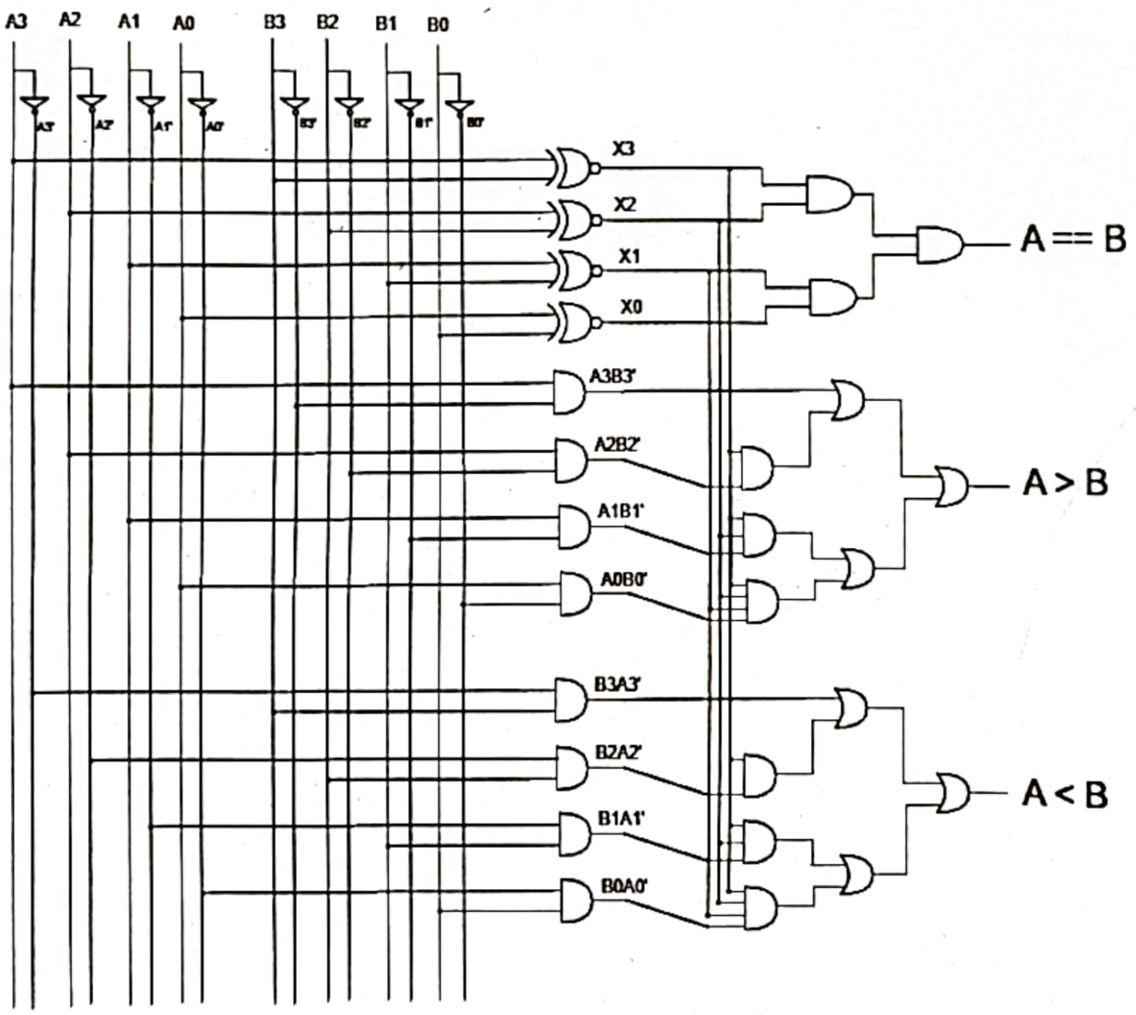
A= 00
B= 10

$$(A > B) = A_3 B_3{}' + x_3 A_2 B_2{}' + x_3 x_2 A_1 B_1{}' + x_3 x_2 x_1 A_0 B_0{}'$$

$$(A < B) = A_3{}' B_3 + x_3 A_2{}' B_2 + x_3 x_2 A_1{}' B_1 + x_3 x_2 x_1 A_0{}' B_0$$

Diagram:

Building a 4-bit magnitude comparator using and, or, not, xnor gate(s):



Procedure:

- Construct the Circuit Diagram on the breadboard.
- Remember to connect each IC's VCC pin to the "+5V" position of the DC Power Supply of the trainer board, and the GND or 0V pin to the "GND" position of the trainer board.
- Connect the inputs to the Data switches and outputs to any position of the LED Display.

Department of Computer Science and Engineering
BRAC University
CSE 260: Digital Logic Design

Experiment # 6: Design circuit using encoder & decoder.

Objective:

- To get familiarized with ***74ls138 [decoder]; 74ls148 [encoder]***
- To gain experience working with practical circuits.

Required Components:

1. IC 74138
2. IC 74148

Theory:

Enable Pin: An enable pin is a special input in a digital circuit that works like an ON/OFF switch. It controls whether the circuit or a part of it is active or inactive.

Active Low Circuit: Active-low means the circuit turns on or performs its function when the input is 0 (low voltage).
For example, if an enable pin is active-low, it means you must give a 0 (low voltage) to turn the circuit ON.

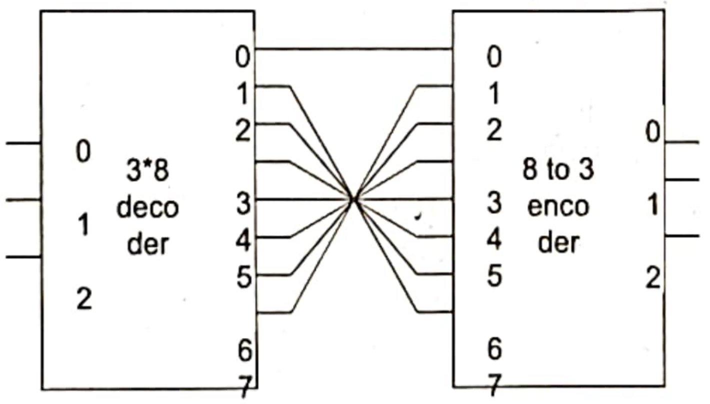
Task - 1: Design a circuit that outputs the 2's complement of a 3-bit number using encoder & decoder.

Truth Table:

Inputs				Expected Outputs				Active Low Outputs		
Minterm	C	B	A	Minterm	D_2	D_1	D_0	D2	D1	D0
0	0	0	0	0	0	0	0	1	1	1
1	0	0	1	7	1	1	1	0	0	0
2	0	1	0	6	1	1	0	0	0	1
3	0	1	1	5	1	0	1	0	1	0
4	1	0	0	4	1	0	0	0	1	1
5	1	0	1	3	0	1	1	1	0	0
6	1	1	0	2	0	1	0	1	0	1
7	1	1	1	1	0	0	1	1	1	0

Diagram:

Building a 3-bit 2's complement converter using encoder and decoder:



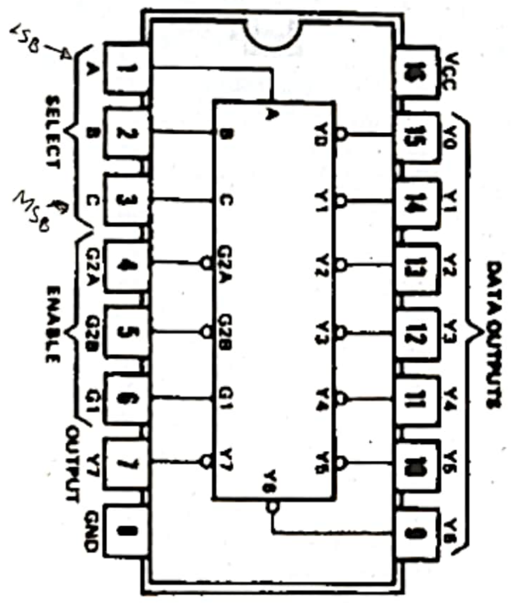
Output line Connection	
Decoder	Encoder
0	0
1	7
2	6
3	5
4	4
5	3
6	2
7	1

Procedure:

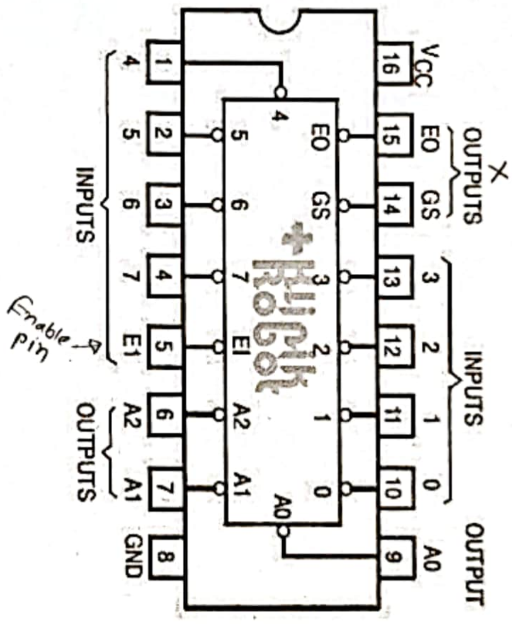
- Construct the Circuit Diagram on the breadboard.
- Remember to connect each IC's VCC pin to the "+5V" position of the DC Power Supply of the trainer board, and the GND or 0V pin to the "GND" position of the trainer board.
- Connect the inputs to the Data switches and outputs to any position of the LED Display.

Pin Diagrams of 74138 [decoder] and 74148 [encoder] ICs:

74138



74148



Decoder to Encoder Connection:

15 - 10
14 - 4
13 - 3
12 - 2
11 - 1
10 - 13
9 - 12
7 - 11

Department of Computer Science and Engineering
BRAC University
CSE 260: Digital Logic Design

Experiment # 7: Function Implementation Using a 4x1 MUX.

Objective:

- To get familiarized with *74153[MUX];*
- To gain experience working with practical circuits.

Required Components:

1. IC 74153
2. IC 7408
3. IC 7432
4. IC 7404

Function:

F(A, B, C, D) = Σ(1, 2, 3, 4, 5, 6, 7, 9, 11, 12, 13)

Procedure:

We will implement a given logic function using a 4×1 multiplexer. The experiment is divided into two checkpoints:

1. **Checkpoint 1** – Test a 4×1 MUX to ensure it works correctly.
2. **Checkpoint 2** – Implement the given logic function using the MUX.

Checkpoint 1: Testing a 4x1 mux
We will use the 74153 IC, which contains two 4×1 MUXes in one package. For this experiment, we will use MUX-1 (the left one).

MUX-1 Pin Configuration:

Inputs:	Selectors:	Output:	Power:	Enable (Strobe) Pins:
Pin 3 → I3 Pin 4 → I2 Pin 5 → I1 Pin 6 → I0	Pin 2 → S1 Pin 14 → S0	Pin 7 → Y	Pin 16 → +5v Pin 8 → GND	Pin 1 (MUX-1 Enable) → GND Pin 15 (MUX-2 Enable) → GND

Setup:

- Connect Pin 16 to +5V and Pin 8 to GND.
- Ground both Enable pins → Pin 1 and 15.
- Connect four input switches to the **MUX input pins** → Pin 6, 5, 4, 3.
- Connect another two switches to the **selector pins** → Pin 2, 14.
- Connect Output (Pin 7) to an LED.

Testing the MUX:

MUX input lines				Selectors		Output
I3	I2	I1	I0	S1	S0	Y
0	0	0	1	0	0	1
1	1	1	0	0	0	0
0	0	1	0	0	1	1
1	1	0	1	0	1	0
0	1	0	0	1	0	1
1	0	1	1	1	0	0
1	0	0	0	1	1	1
0	1	1	1	1	1	0

Check if the MUX output matches the Expected Output in the table. If there is any mismatch, re-check wiring and ensure enable pins are grounded.
Checkpoint 1 - Completed.

Checkpoint 2: Implementing the function

Find the mux input logics:

- The given function has four input variables: A, B, C, D.
- We are using a 4×1 MUX which requires two selector pins.
- We will use:
  C → S1 (Selector 1)
  D → S0 (Selector 0)
- This means the MUX input lines (I0, I1, I2, I3) will be defined in terms of only A and B.

F(A, B, C, D) = Σ(1, 2, 3, 4, 5, 6, 7, 9, 11, 12, 13)
Now complete the table for variable A and B.

	I0	I1	I2	I3
A'B'	0	①	②	③
A'B	④	⑤	⑥	⑦
AB'	8	⑨	10	⑪
AB	⑫	⑬	14	15

Find the equations:

I0 = B
I1 = 1
I2 = A'
I3 = A'+B'

Setup:

Once you have derived the Boolean equations for all four MUX input lines (I0–I3):

- Take two new inputs from two switches — these will be A and B.
- Using A and B, build the logic circuits for each equation you found for I0, I1, I2, and I3.
- Connect the outputs of these logic circuits to their corresponding MUX input pins.

Testing the circuit:

	A	B	C(S1)	D(S0)	F
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	0
15	1	1	1	1	0

Check if the MUX output matches the Expected Output in the table. If there is any mismatch, re-check wiring and ensure the enable pins are grounded.

Checkpoint 2 - Completed.

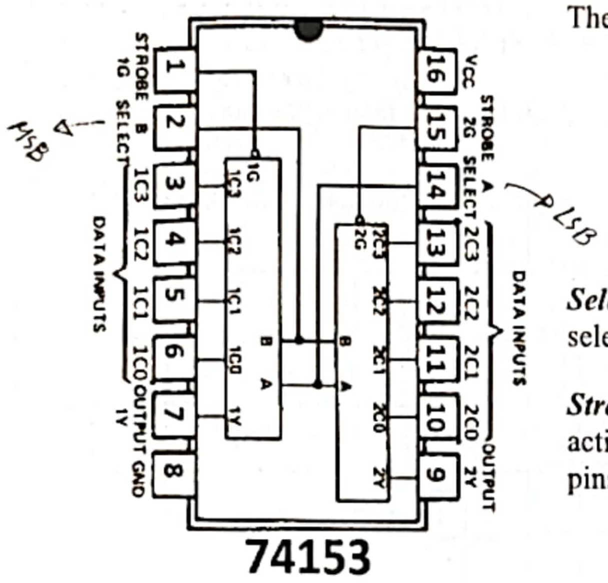
Explanation:

Lets take an input = 12. So, ABCD = 1100

CD is basically the selector bits S1, S0. So, the input line I0 will be selected.

So, check the Logic equation for the I0 input. If it evaluates to 0, the MUX output will be 0 else 1.

Pin Diagrams of 74153[MUX] IC:



IC Description:

The IC has two identical 4:1 multiplexers:

- MUX 1: Uses inputs 1C0 to 1C3, output 1Y
- MUX 2: Uses inputs 2C0 to 2C3, output 2Y

Selection Lines: Both MUXs share the same select lines: A and B

Strobe Pins (1G and 2G): Both of these pins are active-low enable pins; So, keep both 1G and 2G pins connected with 0V or "GND"