Laboratory Manual

For

Design of Digital Circuits

(IT 301)

B.Tech

SEM III (IT)

June 2010

Faculty of Technology
Dharamsinh Desai University
Nadiad.
www.ddu.ac.in

Table of Contents

List of Figures:

Sample experiment

1 AIM: Implement half adder and full adder circuits

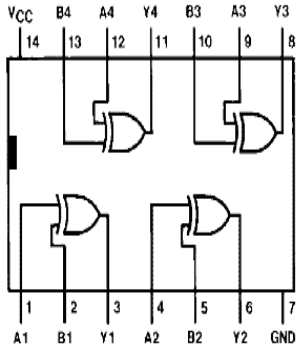
2 TOOLS/APPARATUS: IC 7486, IC 7432, IC 7408, IC 7400, Power supply, Connecting wires, Multimeter etc.

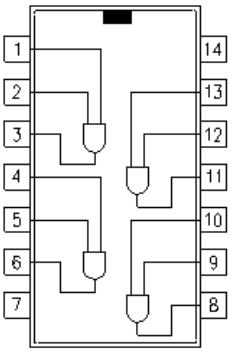
3 STANDARD PROCEDURES:

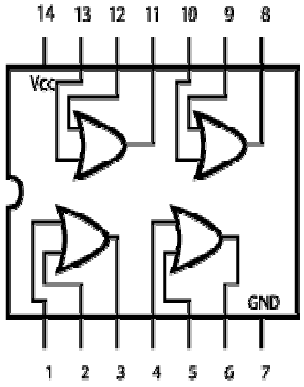
3.1 Analyzing the Problem:

To implement half and full adder we require X-OR gates, AND gates, OR Gate. Pin Diagrams of these gates are as below.

Pin Diagrams of Basic gates ICs used in experiment:







XOR Gate(IC 7486) **AND Gate(IC 7408)** **OR Gate(IC 7432)**

Truth Tables of Basic gates used in experiment:

XOR

xy	F(x)
00	0
01	1
10	1
11	0

AND

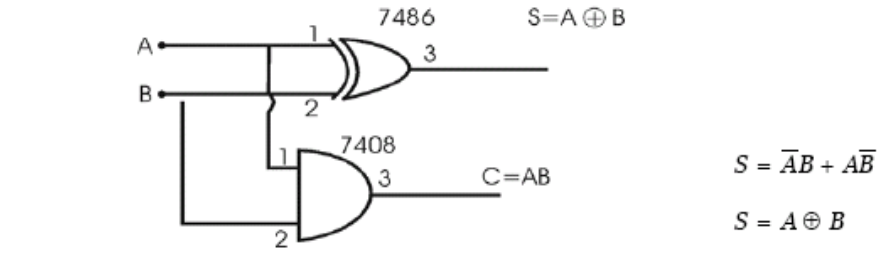
xy	F(x)
00	0
01	0
10	0
11	1

OR

xy	F(x)
00	0
01	1
10	1
11	1

3.2 Designing the Solution:

Half Adder using basic gates:-

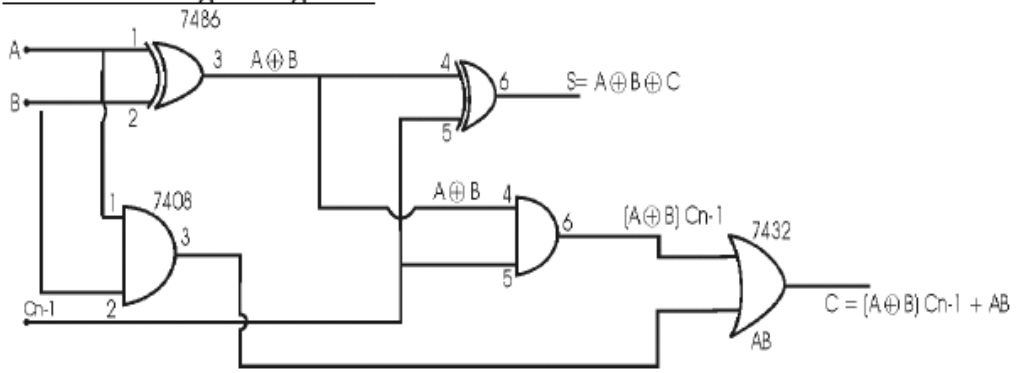


$S = \overline{A}B + A\overline{B}$

$S = A \oplus B$

$C = AB$

Full Adder using basic gates:-



3.3 Implementing the Solution

- Plug the chips you will be using into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
- Connect +5V and GND pins of each chip to the power and ground bus strips on the breadboard.
- Make the connections as per the circuit diagram.
- Switch on VCC and apply various combinations of input according to truth table.
- Note down the output readings for half/full adder and sum and the carry bit for different combinations of inputs in following Tables where S & V indicating logic value of the output. And fill your result in S (V) and C (V) in voltage. Where 5V indicating logic 1 and 0V indicating logic 0.

Half Adder					
A	B	S	C	S(V)	C(V)
0	0				
0	1				
1	0				
1	1				

Full Adder						
A	B	Cn-1	S	C	S(V)	C(V)
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

- Derive the Equation for Sum and Carry.

3.4 Testing the Solution

Half Adder			
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Truth table for half adder

Full Adder				
A	B	Cn-1	S	C
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

Truth table for full adder

4 Conclusions

By using various logic gate ICs we can perform the full or half adder and check the truth table.

EXPERIMENT -1

1 AIM: To make 5 volt DC power supply

2 TOOLS/APPARATUS: Connecting wires, Multimeter, Step-down Transformer, Capacitors ,LED Diode etc.

3 STANDARD PROCEDURES:

Designing the Solution:

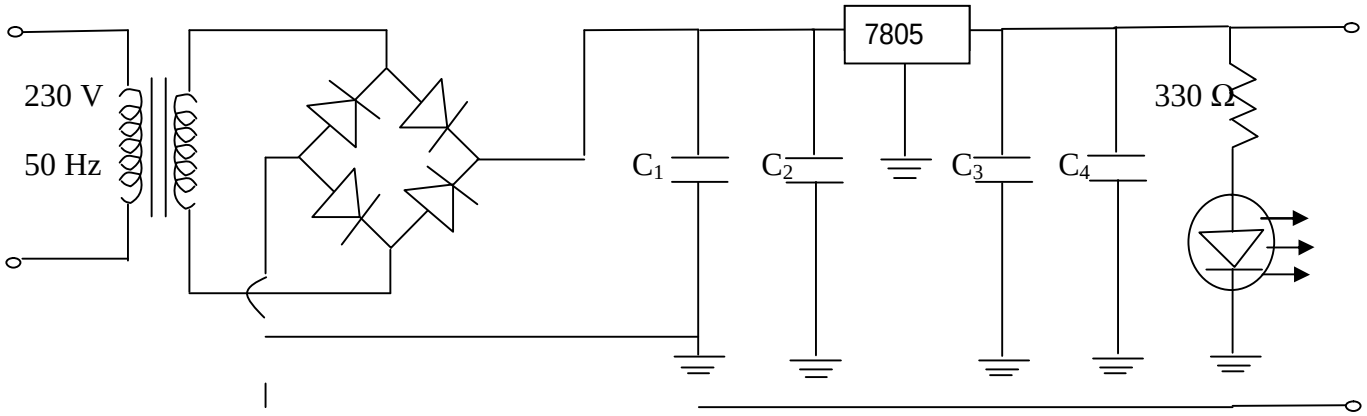


Fig:1.1 Power Supply

Stepdown Transformer:

It is used to step down the voltage of A.C signal. Here we are using 9 V stepdown transformers. It converts 230 V A.C signal into 9 V A.C signal.

Bridge Rectifier:

It converts A.C signal into D.C signal. It is full wave rectifier. Here it converts 9 V A.C signal into 9 V D.C signal.

Capacitors:

After rectification, we don't get the perfect D.C signal; there is some ripple in it. To avoid this kind of ripple, we use capacitors. It works as a filter.

Here C_1=220 μF, 16 V, C_2=0.1 μF, 10 V, C_3=100 μF, 10 V , C_4=0.1 μF, 10 V.

LED Diode:

It is used to check whether the circuit is getting power or not. LED turns on when the circuit gets power.

4 CONCLUSION:

Finally we get 5 V D.C voltages across LED. We can use this circuit as a 5 V D.C power supply.

EXPERIMENT -2

1 AIM: To verify basic logic gates.

2 TOOLS/APPARATUS: IC 7404, IC 7432, IC 7408, IC 7400, IC 7402, IC 7486, IC 74266,Power supply, Connecting wires, Multimeter, Bread Board etc.

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

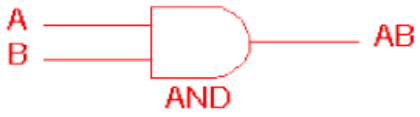
Logic gates are the basic components in digital electronics. They are used to create digital circuits and even complex integrated circuits. For example, complex integrated circui may bring already a complete circuit ready to be used – microprocessors and microcontrollers are the best example – but inside them they were projected using several logic gates.

A gate is a digital electronic circuit having only one output but one or more inputs. The output or a signal will appear at the output of the gate only for certain input-signal combinations.

There are many types of logic gates; such as AND, OR and NOT, which are usually called the three basic gates. Other popular gates are the NAND and the NOR gates; which are simply combinations of an AND or an OR gate with a NOT gate inserted just before the output signal. Other gates include the XOR “Exclusive-OR” and the XNOR "Exclusive NOR" gates.

All the logic gates used in the exercises below are known as TTL (transistor-to-transistor) logic. These have the convenient property that the output of any gate can be used directly as input to another gate. All these TTL circuits are operated from a 5 V power supply, and the binary digits 0 and 1 are represented by low and high voltages on the gate terminals.

- **AND gate**

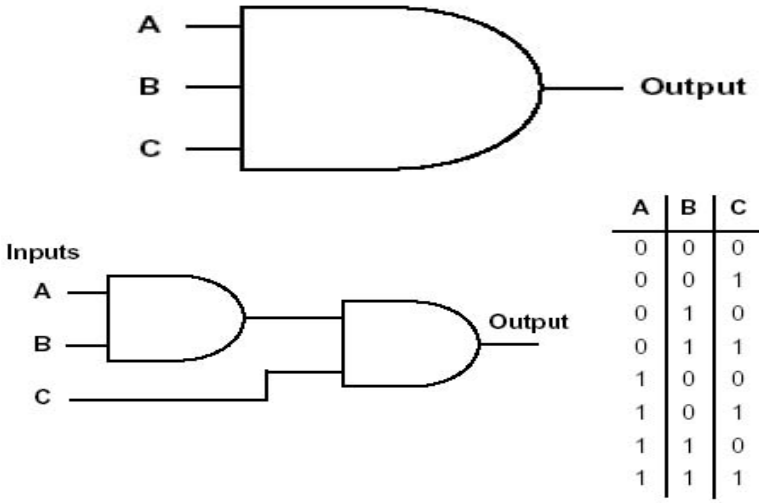


2 Input AND gate		
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

Fig:2.1 (A) 2 input AND Gate

The AND gate is an electronic circuit that gives a high output (1) only if all its inputs are high. A dot (.) is used to show the AND operation i.e. A.B. Bear in mind that this dot is sometimes omitted i.e. AB

- **3-input AND gate**



A	B	C	Output
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Fig:2.1 (B) 3 input AND Gate

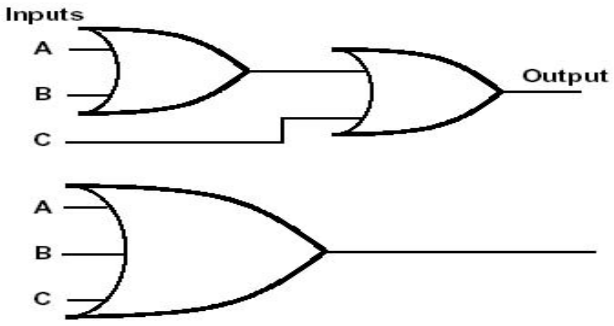
- **OR gate**

2 Input OR gate		
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

Fig:2.2 (A) 2 input OR Gate

The OR gate is an electronic circuit that gives a high output (1) if one or more of its inputs are high. A plus (+) is used to show the OR operation.

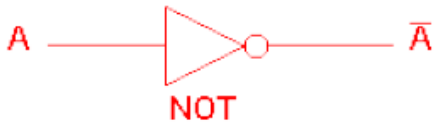
- **3-input OR gate**



A	B	C	Output
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Fig:2.2 (B) 3 input OR Gate

- **NOT gate**



NOT gate	
A	$\overline{A}$
0	1
1	0

Fig:2.3 NOT Gate

The NOT gate is an electronic circuit that produces an inverted version of the input at its output. It is also known as an inverter. If the input variable is A, the inverted output is known as NOT A. This is also shown as A', or A with a bar over the top, as shown at the outputs. The diagrams below show two ways that the NAND logic gate can be configured to produce a NOT gate. It can also be done using NOR logic gates in the same way.

- **NAND gate**

2 Input NAND gate		
A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

Fig:2.4 NAND Gate

This is a NOT-AND gate which is equal to an AND gate followed by a NOT gate. The outputs of all NAND gates are high if any of the inputs are low. The symbol is an AND gate with a small circle on the output. The small circle
represents inversion.

- **NOR gate**

2 Input NOR gate		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Fig:2.5 Nor Gate

This is a NOT-OR gate which is equal to an OR gate followed by a NOT gate.
The outputs of all NOR gates are low if any of the inputs are high. The symbol is an OR gate with a small circle on the output. The small circle represents inversion.

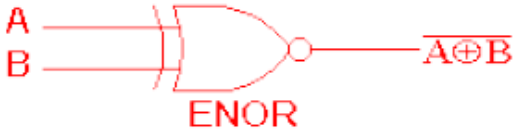
- **EXOR gate**

2 Input EXOR gate		
A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Fig:2.6 EX OR Gate

The 'Exclusive-OR' gate is a circuit which will give a high output if either, but not both, of its two inputs are high. An encircled plus sign ( ) is used to + show the EOR operation.

- **EXNOR gate**

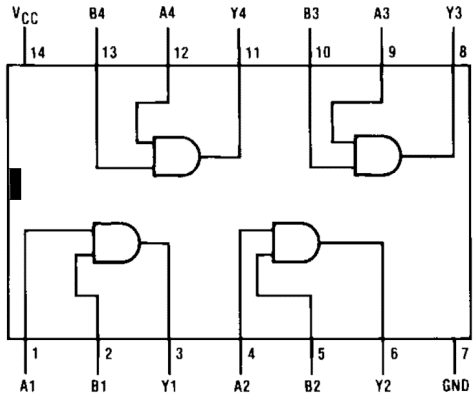


2 Input EXNOR gate		
A	B	$\overline{A⊕B}$
0	0	1
0	1	0
1	0	0
1	1	1

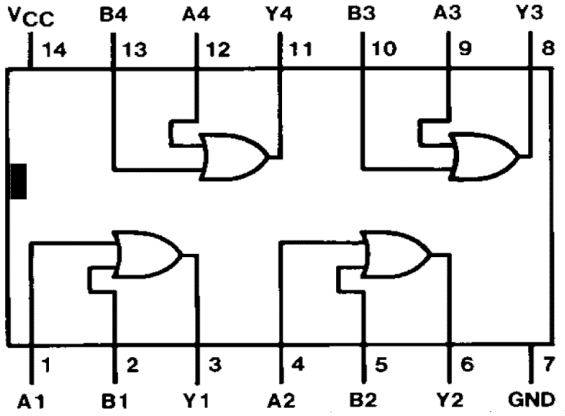
The 'Exclusive-NOR' gate circuit does the opposite to the EOR gate. It will give a low output if either, but not both, of its two inputs are high. The symbol is an EXOR gate with a small circle on the output. The small circle represents inversion.

3.2 Designing the Solution:

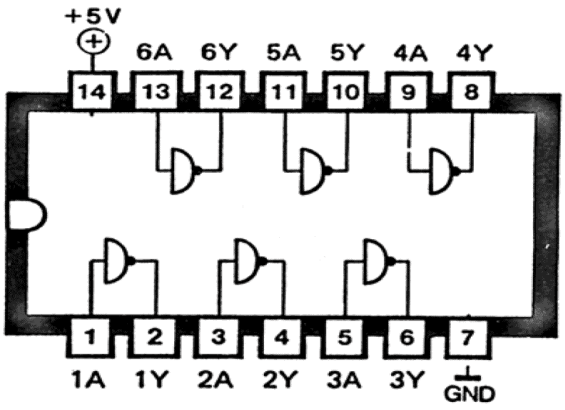
Pin diagram of all the above gates are:



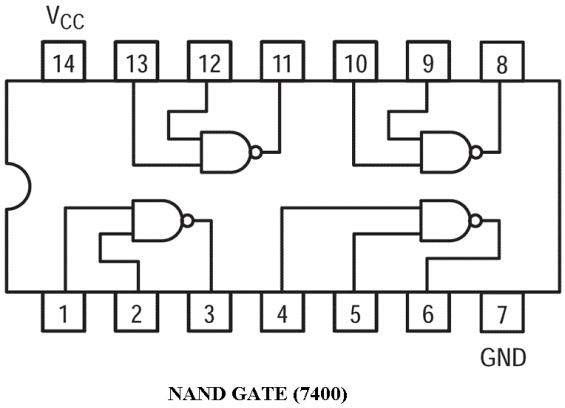
AND GATE (7408)



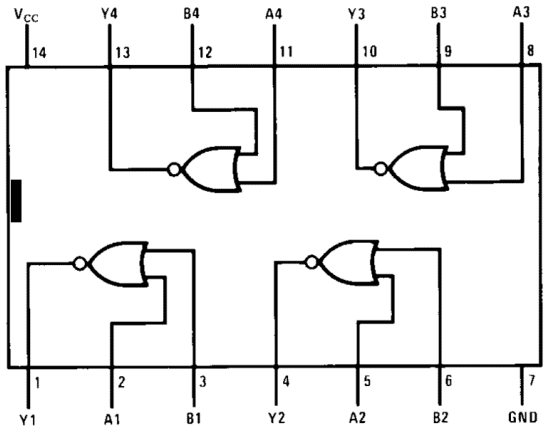
OR GATE (7432)



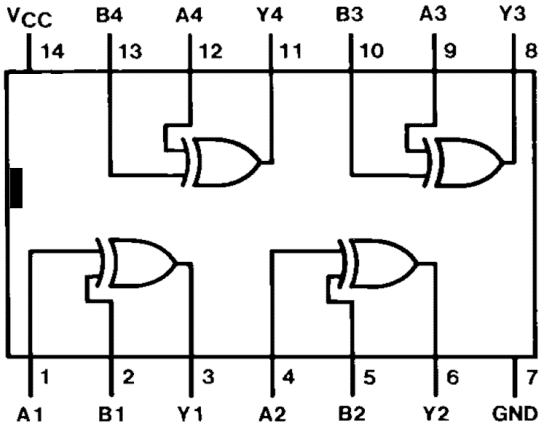
NOT GATE (7404)



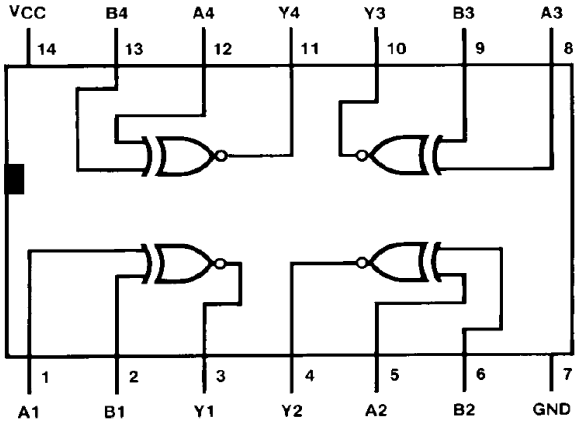
NAND GATE (7400)



NOR GATE (7402)



XOR GATE (7486)



XNOR GATE (74266)

Fig:2.7 Pin Diagrams of IC 7404,7402,7400,7486,74266

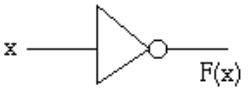
3.3 Implementing the Solution

- Plug the chips you will be using into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
- Connect +5V and GND pins of each chip to the power and ground bus strips on the breadboard.
- Make the connections as per the circuit diagram.
- Switch on VCC and apply various combinations of input according to truth table.
- Note down the output readings for half/full adder and sum and the carry bit for different combinations of inputs in following Tables where S & V indicating logic value of the output. And fill your result in S (V) and C (V) in voltage. Where 5V indicating logic 1 and 0V indicating logic 0.

3.4 Testing the Solution

Observation Tables For all the Gates

NOT



x	F(x)
0	
1	

AND

xy	F(x)
00	
01	
10	
11	

NOR

xy	F(x)
00	
01	
10	
11	

OR

xy	F(x)
00	
01	
10	
11	

XOR

xy	F(x)
00	
01	
10	
11	

NAND

xy	F(x)
00	
01	
10	
11	

XNOR

xy	F(x)
00	
01	
10	
11	

4 Conclusions

All the truth tables are verified.

EXPERIMENT - 3

1 AIM: To verify universal gates NAND and NOR.

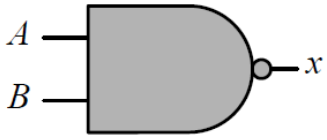
2 TOOLS/APPARATUS: IC 7402, IC 7400, Power supply, Connecting wires, Multimeter, Breadboard etc.

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

- **NAND GATE:**

graphic symbol



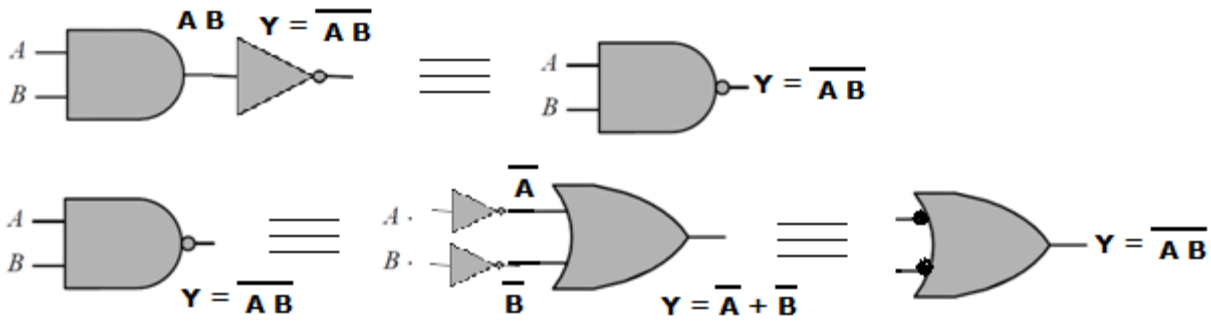
truth table

A	B	x
0	0	1
0	1	1
1	0	1
1	1	0

algebraic function

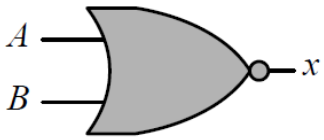
$x = (AB)'$

- NAND is the contraction of AND – NOT gates.
- It has two or more inputs and only one output i.e. Y = A · B.
- When all the inputs are HIGH, the output is LOW. If any one or both the inputs are LOW, then the output is HIGH.
- The Logic symbol and the truth table of NAND gate is as shown here.
- The small circle (or ***bubble***) represents the operation of inversion.
- The NAND gate is equivalent to an OR gate with the bubble at its inputs which are as shown.



- **NOR GATE:**

graphic symbol



truth table

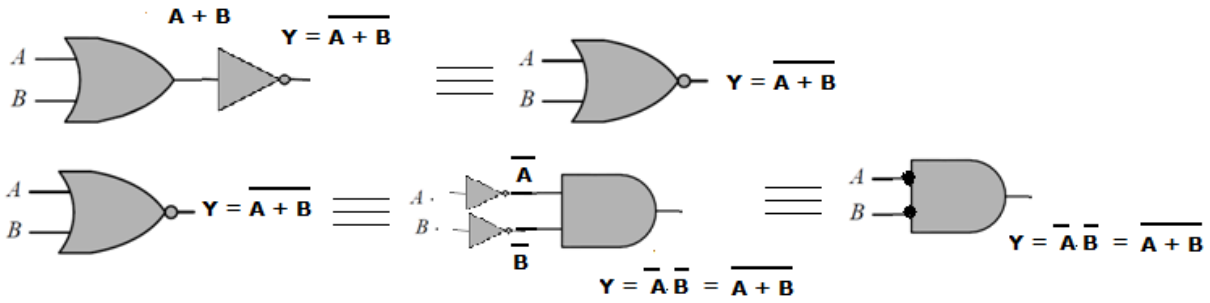
A	B	x
0	0	1
0	1	0
1	0	0
1	1	0

algebraic function

$x = (A + B)'$

Fig:3.2 NOR GATE

- NOR is the contraction of OR – NOT gates.
- It has two or more inputs and only one output i.e. Y = A + B.
- When all the inputs are LOW, the output is HIGH. If any one or both the inputs are HIGH, then the output is LOW.
- The Logic symbol and the truth table of NOR gate is as shown here.
- The small circle (or ***bubble***) represents the operation of inversion.
- The NOR gate is equivalent to an AND gate with the bubble at its inputs which are as shown.



- A universal gate is a gate which can implement any Boolean function without need to use any other gate type.
- The NAND and NOR gates are universal gates.
- In practice, this is advantageous since NAND and NOR gates are economical and easier to fabricate and are the basic gates used in all IC digital logic families.

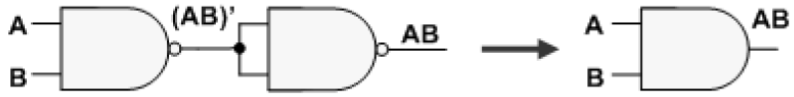
3.2 Designing the Solution:

- **NAND GATE AS A UNIVERSAL GATE :**

To prove that any Boolean function can be implemented using only NAND gates, we will show that the AND, OR, and NOT operations can be performed using only these gates.

- **IMPLEMENTING INVERTER USING NAND GATE :**
  The figure shows two ways in which a NAND gate can be used as an inverter (NOT gate).
  1. All NAND input pins connect to the input signal A gives an output A'.
  2. One NAND input pin is connected to the input signal A while all other input pins are connected to logic 1. The output will be A'.

- **IMPLEMENTING AND USING NAND GATE :**
  An AND gate can be replaced by NAND gates as shown in the figure (The AND is replaced by a NAND gate with its output complemented by a NAND gate inverter).



- **IMPLEMENTING OR USING NAND GATE :**
  An OR gate can be replaced by NAND gates as shown in the figure (The OR gate is replaced by a NAND gate with all its inputs complemented by NAND gate inverters).

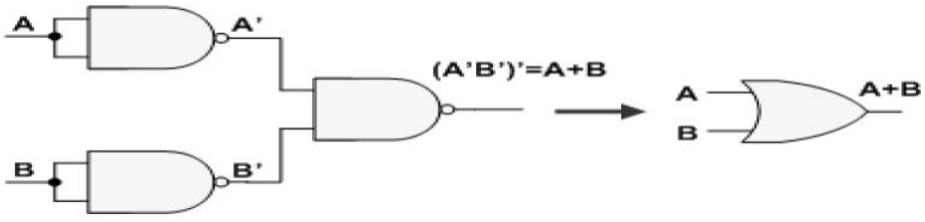
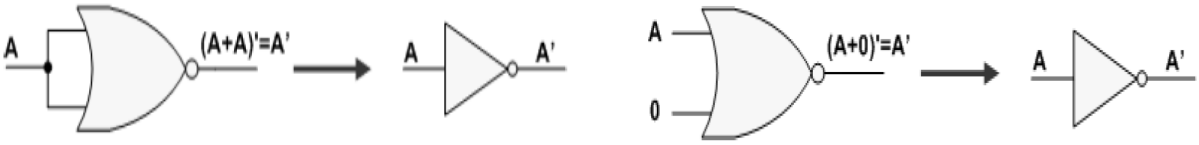


Fig:3.1 IMPLEMENTING BASIC GATES NAND GATE

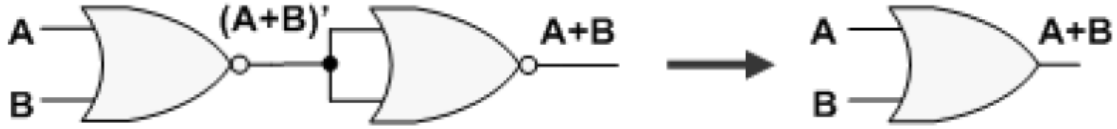
- **NOR GATE AS A UNIVERSAL GATE:**

To prove that any Boolean function can be implemented using only NOR gates, we will show that the AND, OR, and NOT operations can be performed using only these gates.

- **IMPLEMENTING INVERTER USING NOR GATE :**
  - The figure shows two ways in which a NOR gate can be used as an inverter (NOT gate).
  - All NOR input pins connect to the input signal A gives an output A'.
  - One NOR input pin is connected to the input signal A while all other input pins are connected to logic 0. The output will be A'.



- **IMPLEMENTING OR USING NOR GATE :**
  - An OR gate can be replaced by NOR gates as shown in the figure (The OR is replaced by a NOR gate with its output complemented by a NOR gate inverter)



- **IMPLEMENTING AND USING NOR GATE :**
  - An AND gate can be replaced by NOR gates as shown in the figure (The AND gate is replaced by a NOR gate with all its inputs complemented by NOR gate inverters)

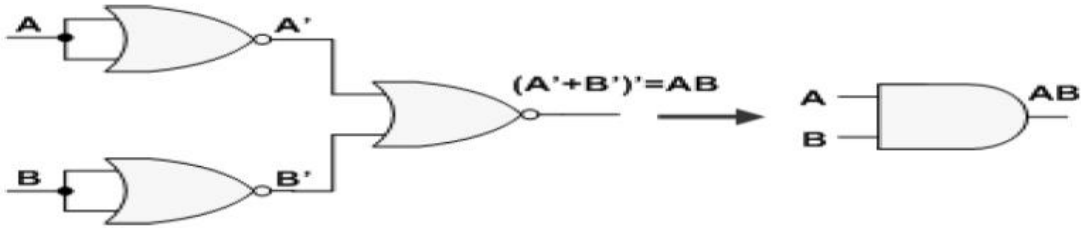


Fig:3.2 IMPLEMENTING BASIC GATES NOR GATE

- Basic IC needed are NAND gate and NOR gate.
- **IC diagram are given as below**

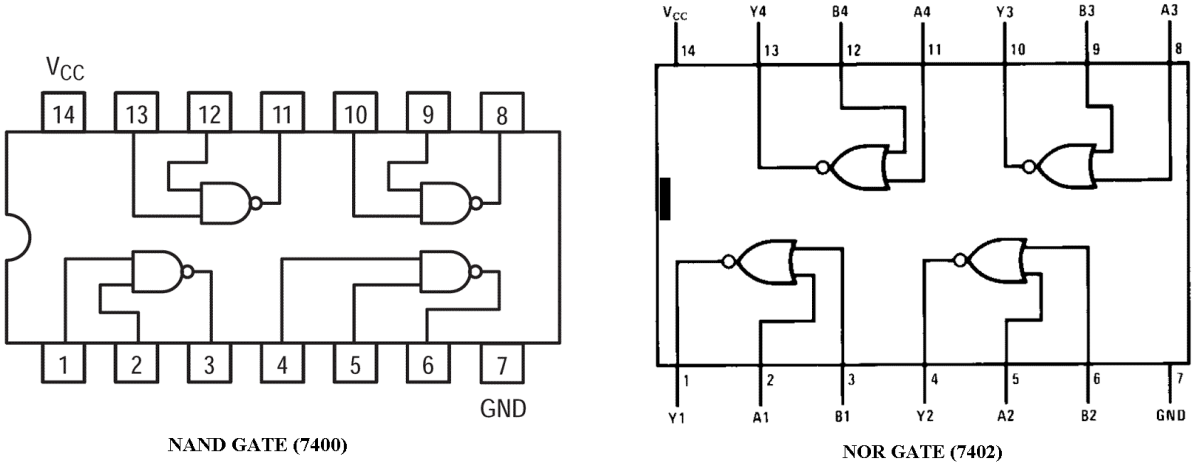


Fig:3.3 Pin Diagram of NAND & NOR GATES

3.3 Implementing the Solution

- Plug the chips you will be using into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
- Connect +5V and GND pins of each chip to the power and ground bus strips on the breadboard.
- Make the connections as per the circuit diagram.
- Switch on VCC and apply various combinations of input according to truth table.
- Note down the output readings for half/full adder and sum and the carry bit for different combinations of inputs in following Tables where S & V indicating logic value of the output. And fill your result in S (V) and C (V) in voltage. Where 5V indicating logic 1 and 0V indicating logic 0.

3.4 Testing the Solution:

Truth Tables:

NAND Gate

A	*B*	Output
0	0	
0	1	
1	0	
1	1	

NOR Gate

A	*B*	Output
0	0	
0	1	
1	0	
1	1	

Inverter Using Nand Gate

A	*B*	Output
0	0	
0	1	
1	0	
1	1	

And Using Nand Gate

A	*B*	Output
0	0	
0	1	
1	0	
1	1	

Or Using Nand Gate

A	*B*	Output
0	0	
0	1	
1	0	
1	1	

Inverter Using Nor Gate

A	*B*	Output
0	0	
0	1	
1	0	
1	1	

Or Using Nor Gate

A	*B*	Output
0	0	
0	1	
1	0	
1	1	

And Using Nor Gate

A	*B*	Output
0	0	
0	1	
1	0	
1	1	

4 Conclusions

A universal gate is a gate which can implement any Boolean function without need to use any other gate type. The NAND and NOR gates are universal gates.

EXERCISE

- Construct Ex-OR gate using a NAND gate.
- Construct Ex-NOR gate using a NOR gate.

EXPERIMENT -4

1 AIM: Implement half adder and full adder circuits

2 TOOLS/APPARATUS: IC 7486, IC 7432, IC 7408, IC 7400, Power supply, Connecting wires, Multimeter etc.

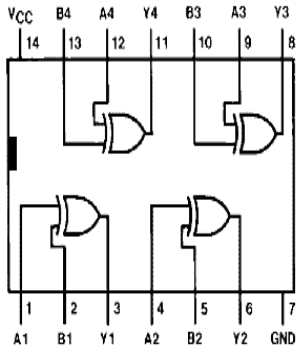
3 STANDARD PROCEDURES:

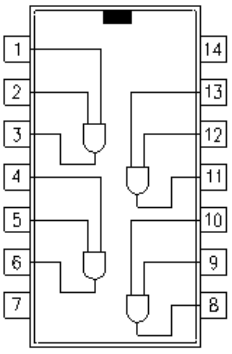
3.1 Analyzing the Problem:

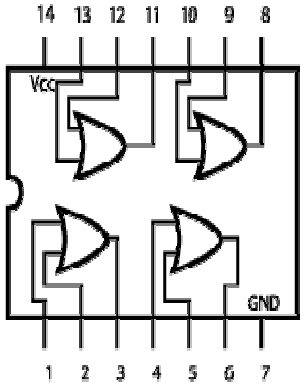
To implement half and full adder we require X-OR gates, AND gates, OR Gate. Pin Diagrams of these gates are as below.

- Derive the Equation for Sum and Carry.

Pin Diagrams of Basic gates ICs used in experiment:







XOR Gate(IC 7486) **AND Gate(IC 7408)** **OR Gate(IC 7432)**

Truth Tables of Basic gates used in experiment:

XOR

xy	F(x)
00	0
01	1
10	1
11	0

AND

xy	F(x)
00	0
01	0
10	0
11	1

OR

xy	F(x)
00	0
01	1
10	1
11	1

3.2 Designing the Solution:

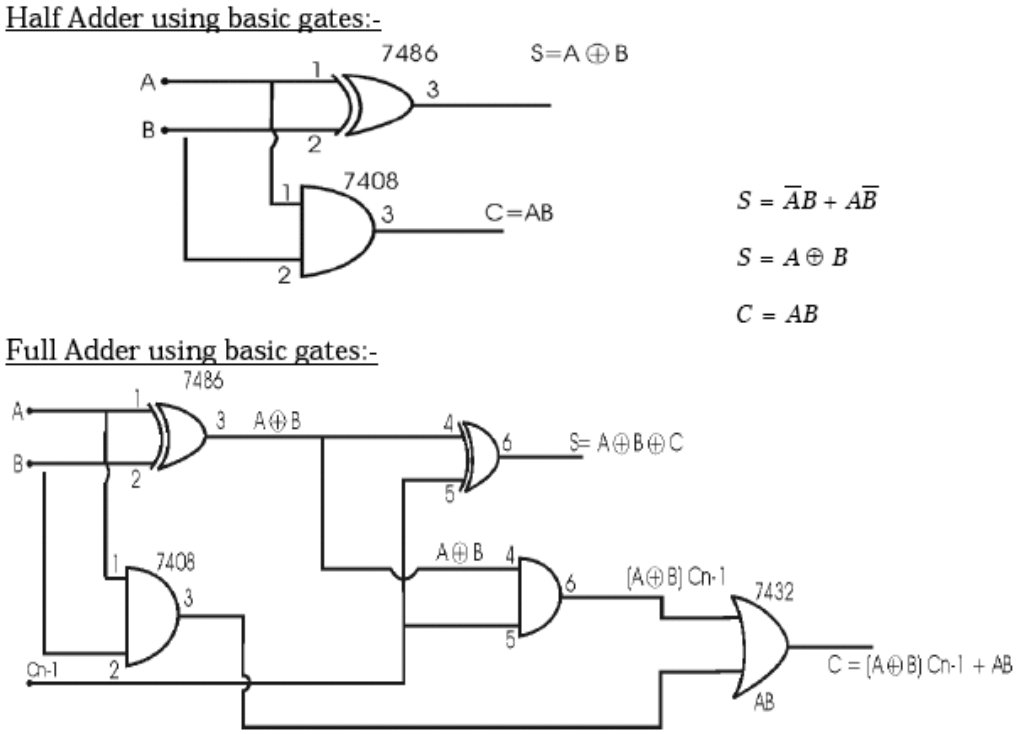


Fig:4.1 adder and full adder circuits

3.3 Implementing the Solution

- Plug the chips you will be using into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
- Connect +5V and GND pins of each chip to the power and ground bus strips on the breadboard.
- Make the connections as per the circuit diagram.
- Switch on VCC and apply various combinations of input according to truth table.
- Note down the output readings for half/full adder and sum and the carry bit for different combinations of inputs in following Tables where S & V indicating logic value of the output. And fill your result in S (V) and C (V) in voltage. Where 5V indicating logic 1 and 0V indicating logic 0.

3.4 Testing the Solution:

Write Observations in following tables and check the output:

Half Adder					
A	B	S	C	S(V)	C(V)
0	0				
0	1				
1	0				
1	1				

Full Adder						
A	B	Cn-1	S	C	S(V)	C(V)
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

4 Conclusions

By using various logic gate ICs we can perform the full or half adder and check the truth table.

EXERCISE:

- Implement half adder and full adder circuits

EXPERIMENT -5

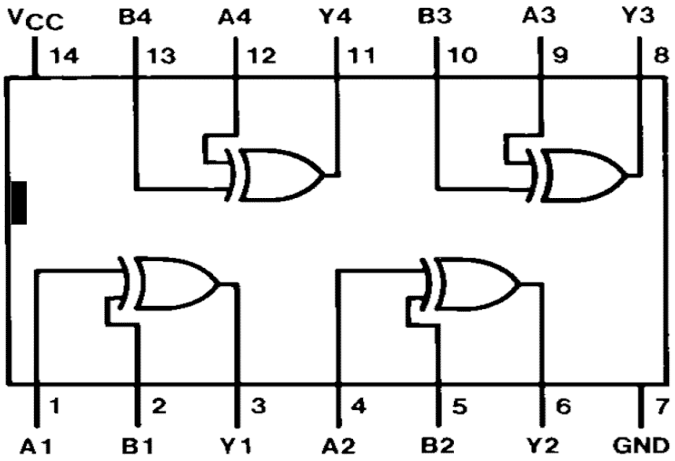
1 AIM: To implement circuit that converts binary to gray and gray to binary.

2 TOOLS/APPARATUS: IC 7486, Power supply, Connecting wires, Multimeter, BreadBoard, etc.

3 STANDARD PROCEDURES:

3.1 Designing the Solution:

IC Diagram of XOR GATE:



XOR GATE (7486)

- **BINARY TO GRAY CODE CONVERTOR**

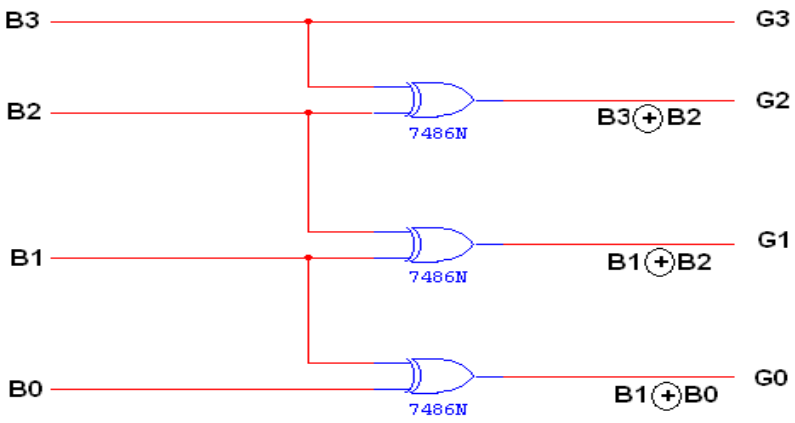


Fig:5.1 BINARY TO GRAY CODE CONVERTOR

- **GRAY CODE TO BINARY CONVERTOR**

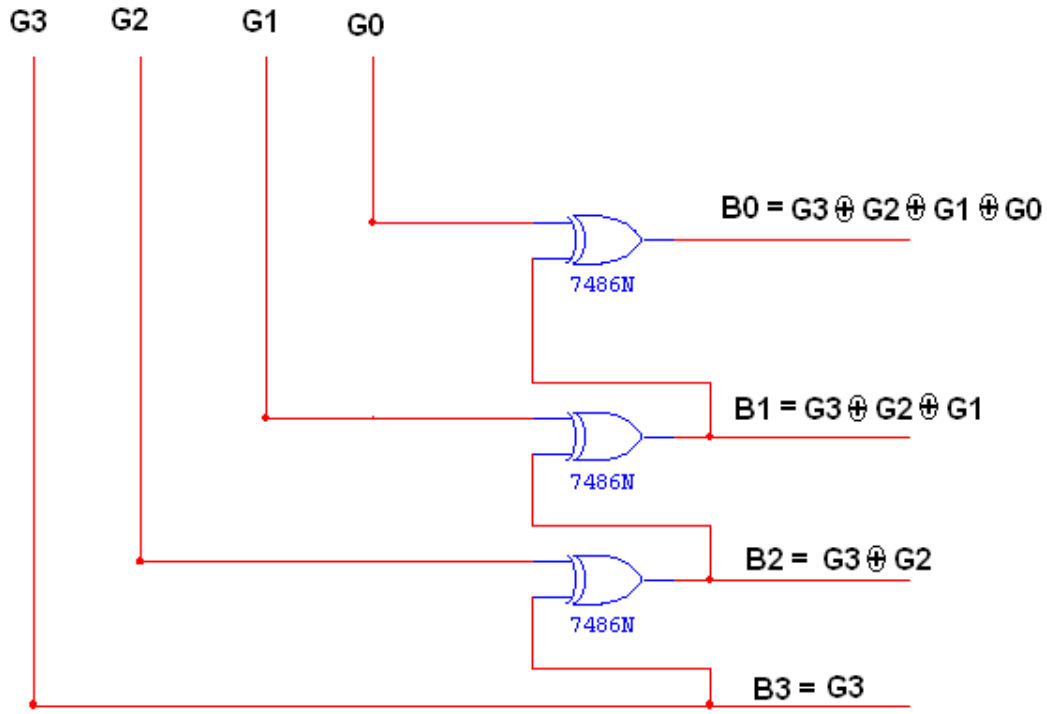


Fig:5.2 GRAY CODE TO BINARY CONVERTOR

3.3 Implementing the Solution

- Plug the chips you will be using into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
- Connect +5V and GND pins of each chip to the power and ground bus strips on the breadboard.
- Make the connections as per the circuit diagram.
- Switch on VCC and apply various combinations of input according to truth table.
- In the case of binary to gray conversion, the inputs B0, B1, B2 and B3 are given at respective pins and outputs G0, G1, G2, G3 are taken for all the 16 combinations of the input.
- In the case of gray to binary conversion, the inputs G0, G1, G2 and G3 are given at respective pins and outputs B0, B1, B2, and B3 are taken for all the 16 combinations of inputs.
- The values of the outputs are tabulated.

3.4 Testing the Solution:

- **BINARY TO GRAY CODE CONVERTOR**

- **OBSERVATION TABLE:**

BINARY INPUT				GRAY OUTPUT			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

- **GRAY CODE TO BINARY CONVERTOR**

  - **OBSERVATION TABLE:**

GRAY INPUT				BINARY OUTPUT			
G3	G2	G1	G0	B3	B2	B1	B0
0	0	0	0				
0	0	0	1				
0	0	1	1				
0	0	1	0				
0	1	1	0				
0	1	1	1				
0	1	0	1				
0	1	0	0				
1	1	0	0				
1	1	0	1				
1	1	1	1				
1	1	1	0				
1	0	1	0				
1	0	1	1				
1	0	0	1				
1	0	0	0				

4 Conclusions

We designed and simulated a Gray code converter that converts binary coded numbers to Gray coded numbers and vice versa.

EXERCISE: 84-2-1 to BCD Code Converter.

EXPERIMENT -6

1 AIM: (A) To implement 3 X 8 Decoder.

(B) Using 3 X 8 Decoder, implement 4 X 16 Decoder.

2 TOOLS/APPARATUS: Power supply, Digital Trainer kit, ICs – 74138, NOT GATE, connecting wires, Multimeter, etc.

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

Decoder:

A decoder is a device which does the reverse of an encoder, undoing the encoding so that the original information can be retrieved. The same method used to encode is usually just reversed in order to decode.

In digital electronics, a decoder can take the form of a multiple-input, multiple-output logic circuit that converts coded inputs into coded outputs, where the input and output codes are different. e.g. n-to-2^n , binary-coded decimal decoders. Enable inputs must be on for the decoder to function, otherwise its outputs assume a single "disabled" output code word. Decoding is necessary in applications such as data multiplexing, 7 segment display and memory address decoding.

Example: A 2-to-4 Line Single Bit Decoder

A slightly more complex decoder would be the n-to-2n type binary decoders. These types of decoders are combinational circuits that convert binary information from 'n' coded inputs to a maximum of 2n unique outputs. We say a maximum of 2n outputs because in case the 'n' bit coded information has unused bit combinations, the decoder may have less than 2n outputs. We can have 2-to-4 decoder, 3-to-8 decoder or 4-to-16 decoder. We can form a 3-to-8 decoder from two 2-to-4 decoders (with enable signals).

Similarly, we can also form a 4-to-16 decoder by combining two 3-to-8 decoders. In this type of circuit design, the enable inputs of both 3-to-8 decoders originate from a 4th input, which acts as a selector between the two 3-to-8 decoders. This allows the 4th input to enable either the top or bottom decoder, which produces outputs of D(0) through D(7) for the first decoder, and D(8) through D(15) for the second decoder.

A decoder that contains enable inputs is also known as a decoder-demultiplexer. Thus, we have a 4-to-16 decoder produced by adding a 4th input shared among both decoders, producing 16 outputs.

- To implement 3X8 and 4X16 decoder we require ICs 74138 and NOT Gate.

- Pin Diagrams of these ICs are as follows:

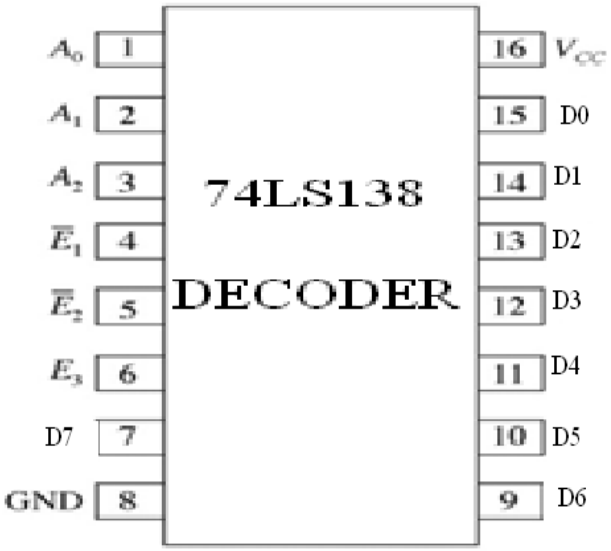


Fig:6.1 Pin Diagram for Decoder

- **Logic diagram of IC- 74LS138**

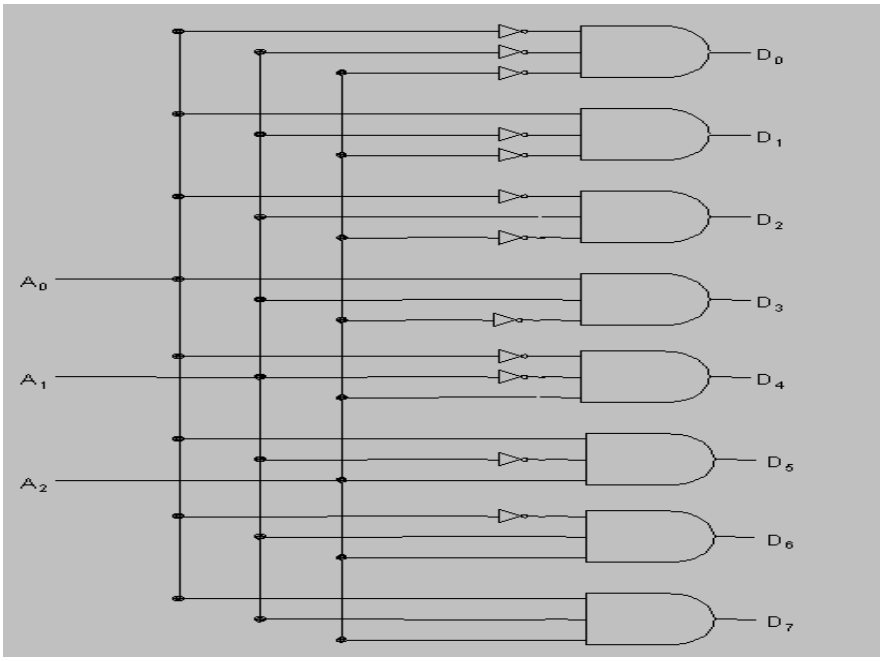
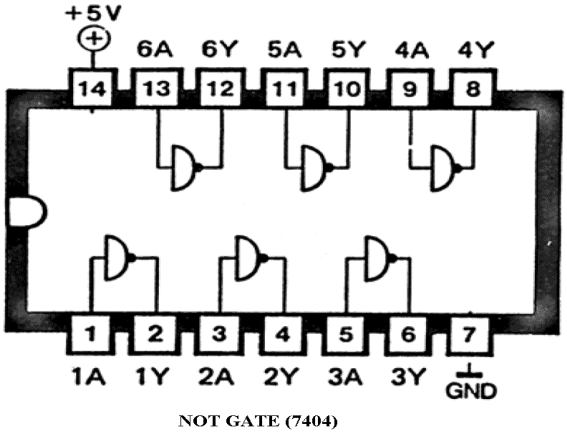


Fig:6.2 Logic diagram of IC- 74LS138

- **Logic diagram of IC- 74LS138**



IC -7404 (NOT GATE)

2 Input NOR gate		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

Truth table NOT Gate

3.2 Designing the Solution:

- **For 3X8 Decoder :**

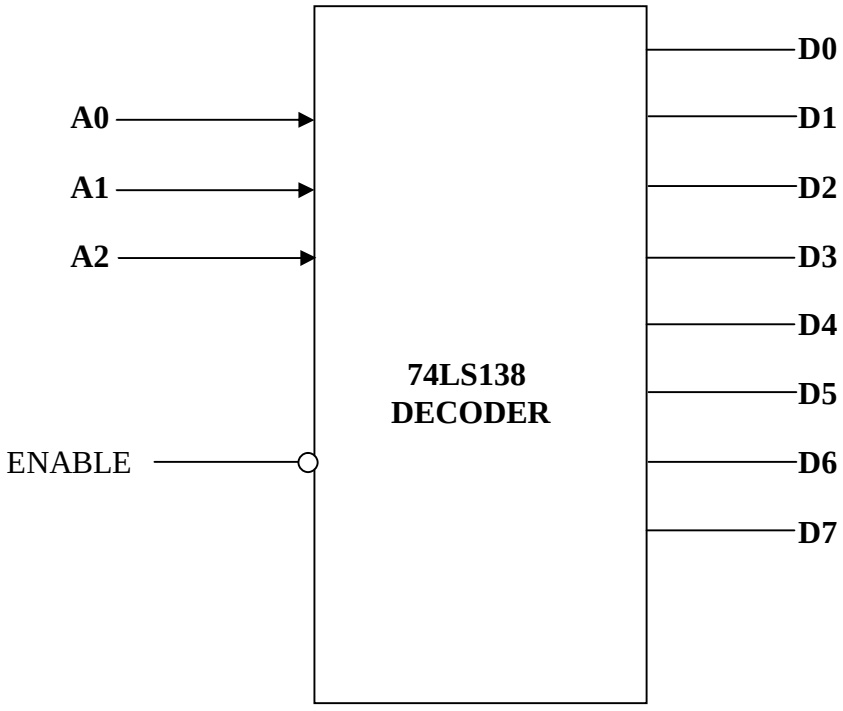


Fig:6.3 3X8 Decoder

- **For 4X16 Decoder:**

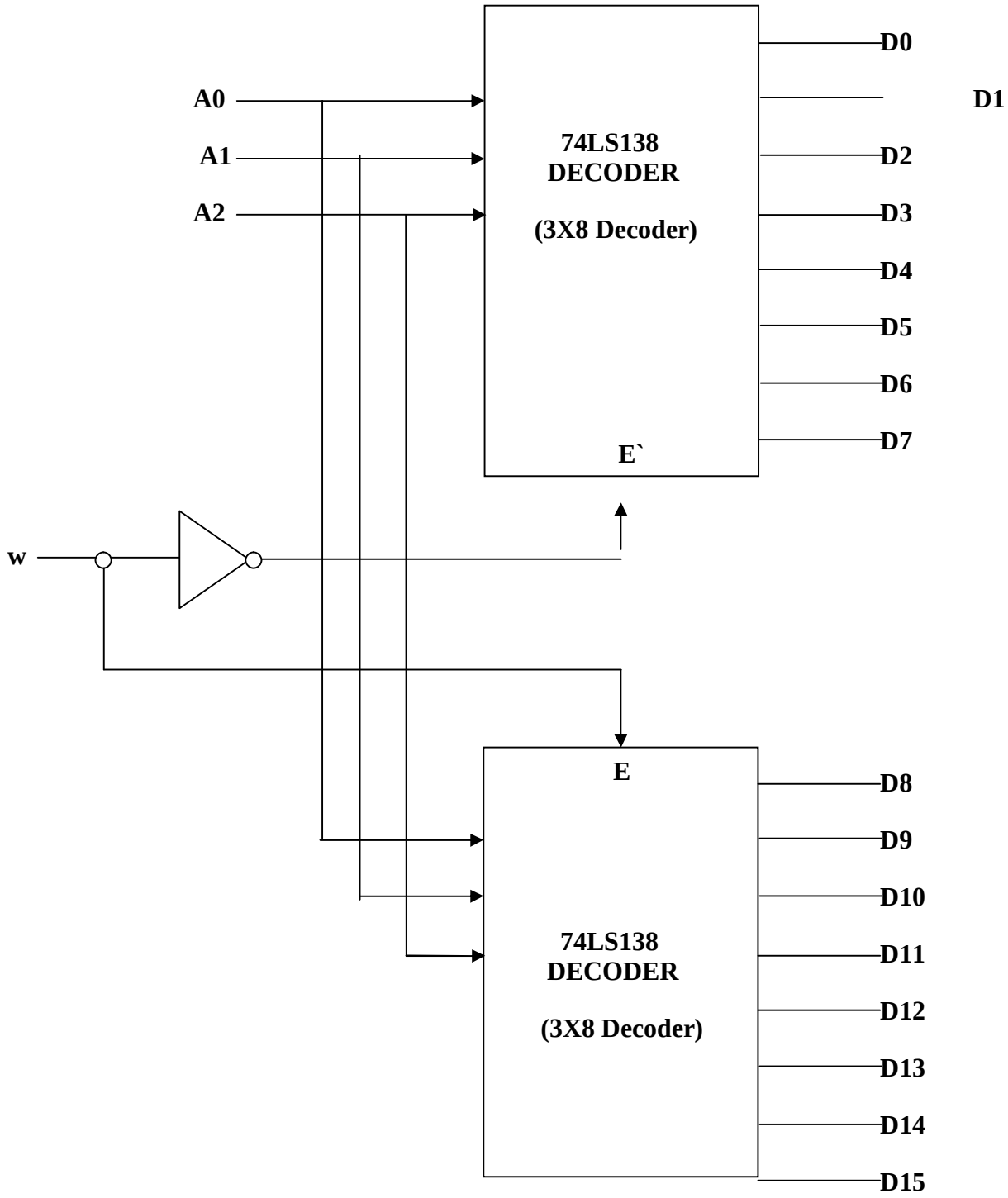


Fig:6.4 4X16 Decoder

3.3 Implementing the Solution:

- A decoder is a logic circuit an n-bit binary input code into M output lines such that only one is output is activated at a time for possible combinations of inputs.
- Consider the decoder on which three inputs and eight outputs are present.
- For active HIGH circuit AND gate is used. For active low circuit the NAND gate is used.
- The decoder is called as 3 to 8 line decoder because it has three input lines and eight output lines.
- And the decoder is called as 4 to 16 line decoder because it has four input lines and sixteen output lines
- Some decoders have one or more ENABLE inputs that are used to control the operation of the decoder.

Procedure:-

- Draw block diagram, truth table and pin diagram for 74138 IC and 7404 IC.
- According to pin assignment, connect the circuit as shown in fig. by using connecting wires.
- Switch 'ON' the power supply.
- Apply the corresponding inputs and verify the truth table.
- For 4 X 16 Decoder, use two 74138 ICs and a 7402 IC. Take any one enable from three enables as fourth input and eight outputs of two ICs will act as sixteen outputs to act as 4 X 16 Decoder.
- Note down the output reading on the multimeter according to the truth tables for 3 X 8 and 4X16 Decoders.

3.4 Testing the Solution

Observation Table:

- **For 3X8 Decoder :**

E	A2	A1	A0	D0	D1	D2	D3	D4	D5	D6	D7
1	0	0	0								
1	0	0	1								
1	0	1	0								
1	0	1	1								
1	1	0	0								
1	1	0	1								
1	1	1	0								
1	1	1	1								

- **For 4X16 Decoder :**

w	w`	A2	A1	A0	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D 10	D 11	D 12	D 13	D 14	D 15
1	0	0	0	0																
1	0	0	0	1																
1	0	0	1	0																
1	0	0	1	1																
1	0	1	0	0																
1	0	1	0	1																
1	0	1	1	0																
1	0	1	1	1																

4 Conclusions:

By using IC-74LS138 we can perform the 3X8 Decoder and by using 2 3X8 Decoder and also the NOT Gate we can perform 4X16 Decoder. We can also check the truth table.

Exercise:

- Construct a 5X32 decoder with four 3X8 decoder and 2X4 decoder.

EXPERIMENT -7

1 AIM: To implement 8X1 Multiplexer.

2 TOOLS/APPARATUS: IC 74151, IC 7404, Power supply, Connecting wires, Multimeter, Breadboard etc.

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

A multiplexer (MUX) is a digital switch which connects data from one of n sources to the output. A number of select inputs determine which data source is connected to the output. The block diagram of MUX with n data sources of b bits wide and s bits wide select line is shown in below figure

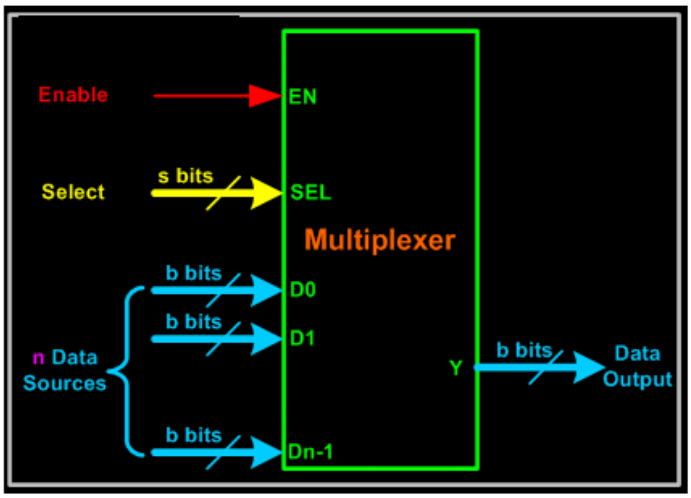


Fig:.7.1 MULTIPLEXER

MUX acts like a digitally controlled multi-position switch where the binary code applied to the select inputs controls the input source that will be switched on to the output as shown in the figure below. At any given point of time only one input gets selected and is connected to output, based on the select input signal.

Example:

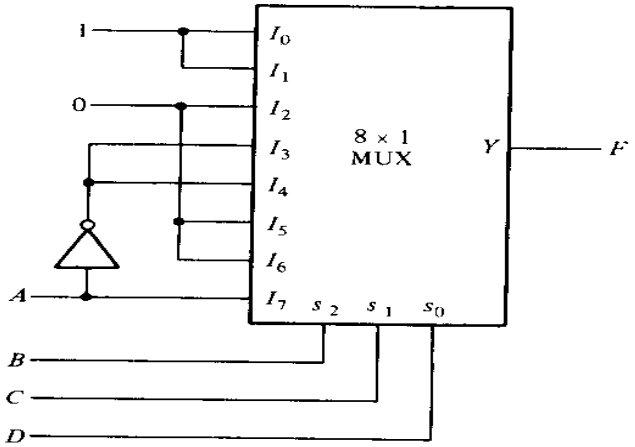
Implement following function with 8x1 multiplexer.

$$F(A, B, C, D) = \Sigma(0, 1, 3, 4, 8, 9, 15)$$

Function Table:

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	⓪	①	2	③	④	5	6	7
A	⑧	⑨	10	11	12	13	14	⑮
	1	1	0	A'	A'	0	0	A

Circuit diagram:



3.2 Designing the Solution:

IC 74LS151 is used as 8x1 multiplexer IC & pinout for IC 74151 is as below:

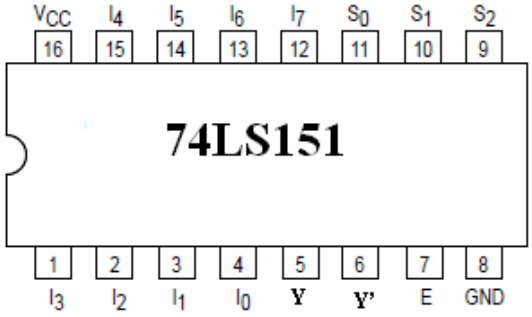


Fig:.7.2 Pin Diagram MULTIPLEXER IC 74151

3.3 Implementing the Solution

- Plug the chips you will be using into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
- Connect +5V and GND pins of each chip to the power and ground bus strips on the breadboard.
- Make the connections as per the circuit diagram.
- Switch on VCC and apply various combinations of input according to truth table.
- Derive your conclusion.

3.4 Testing the Solution:

Observation table for example:

INPUT				**OUTPUT**	
A	B	C	D	Y	Y’
1	×	×	×		
0	0	0	0		
0	0	0	1		
0	0	1	0		
0	0	1	1		
0	1	0	0		
0	1	0	1		
0	1	1	0		
0	1	1	1		

4 Conclusions

We can conclude that to implement any function we can use multiplexer.

EXPERIMENT -8

1 AIM: (A) To implement 4-bit comparator.
(B) Using 4-bit comparator implements 8-bit comparator.

2 TOOLS/APPARATUS: IC 7486, IC 7432, IC 7408, IC 7400, Power supply, Connecting wires, Multimeter etc.

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

The comparison of two numbers is an operator that determines 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 magnitude. The outcome of the comparator is specified by three binary variables that indicate whether A>B, A=B (or) A<B.

$A = A_3\ A_2\ A_1\ A_0$

$B = B_3\ B_2\ B_1\ B_0$

The equality of the two numbers and B is displayed in a combinational circuit designated by the symbol (A=B).

This indicates A greater than B, then inspect the relative magnitude of pairs of significant digits starting from most significant position. A is 0 and that of B is 0.

We have A<B, the sequential comparison can be expanded as

$A>B = A3B_3^{\ 1} + X_3A_2B_2^{\ 1} + X_3X_2A_1B_1^{\ 1} + X_3X_2X_1A_0B_0^{\ 1}$

$A<B = A_3^{\ 1}B_3 + X_3A_2^{\ 1}B_2 + X_3X2A_1^{\ 1}B_1 + X_3X_2X_1A_0^{\ 1}B_0$

The same circuit can be used to compare the relative magnitude of two BCD digits.

Where, A = B is expanded as,

$A = B = (A_3 + B_3)\ (A_2 + B_2)\ (A_1 + B_1)\ \ (A_0 + B_0)$

↓ ↓ ↓ ↓

x_3 x_2 x_1 x_0

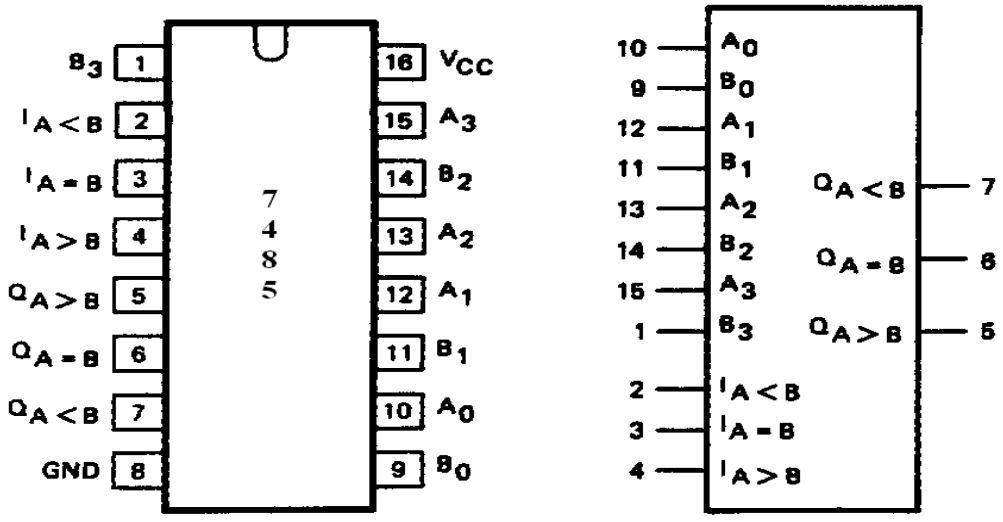
3.2 Designing the Solution:

- **4 BIT MAGNITUDE COMPARATOR**

Pin Configuration of 7485:

PIN NO.	SYMBOL	NAME AND FUNCTION
2	$I_{A<B}$	A < B expansion input
3	$I_{A=B}$	A = B expansion input
4	$I_{A>B}$	A > B expansion input
5	$Q_{A>B}$	A > B output
6	$Q_{A=B}$	A = B output
7	$Q_{A<B}$	A < B output
8	GND	ground (0 V)
9, 11, 14, 1,	B_0 to B_3	word B inputs
10, 12, 13, 15	A_0 to A_3	word A inputs
16	V_{CC}	positive supply voltage

Pin Diagram & Logic Diagram:



Pin configuration. Logic symbol.

Fig:.8.1 Pin Diagram & Logic Diagram of IC 7485

- **8 BIT MAGNITUDE COMPARATOR**

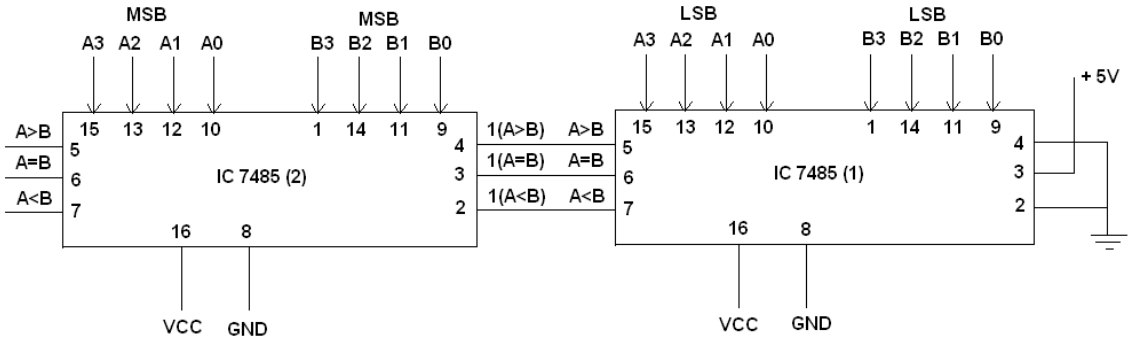


Fig:.8. 2 8 BIT MAGNITUDE COMPARATOR

3.3 Implementing the Solution

- Plug the chips you will be using into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
- Connect +5V and GND pins of each chip to the power and ground bus strips on the breadboard.
- Make the connections as per the circuit diagram.
- Switch on VCC and apply various combinations of input according to truth table.
- Take 2 4bit nos & Write output in observation table.
- For 8 bit Magnitude Comparator connect two 4 bit Magnitude Comparator as shown in diagram and compare 8 bit nos.

3.4 Testing the Solution:

Observation Table for 4 bit Comparator:

A_3 B_3	A_2 B_2	A_1 B_1	A_0 B_0	A>B	A=B	A<B	A>B	A=B	A<B
$A_3>B_3$	X	X	X	X	X	X			
$A_3<B_3$	X	X	X	X	X	X			
$A_3=B_3$	$A_2>B_2$	X	X	X	X	X			
$A_3=B_3$	$A_2<B_2$	X	X	X	X	X			
$A_3=B_3$	$A_2=B_2$	$A_1>B_1$	X	X	X	X			
$A_3=B_3$	$A_2=B_2$	$A_1<B_1$	X	X	X	X			
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0>B_0$	X	X	X			
$A_3=B_3$	$A_2=B_2$	$A_1=B_1$	$A_0<B_0$	X	X	X			

Observation Table for 8 bit Comparator:

A	B	A>B	A=B	A<B
0 0 0 0 0 0 0 0	**0 0 0 0 0 0 0 0**			
0 0 0 1 0 0 0 1	**0 0 0 0 0 0 0 0**			
0 0 0 0 0 0 0 0	**0 0 0 1 0 0 0 1**			

4 Conclusions

We conclude that using 4 bit comparator we can compare 2 4-bit no. & using 2 4 bit comparator we can implement 8 bit comparator which compares 2 8-bit nos.

EXPERIMENT - 9

1 AIM: To verify various flip-flops like D, T, and JK.

2 TOOLS/APPARATUS: Power supply, Digital Trainer kit, ICs – 7474, Connecting wires, Multimeter, CRO, Clock Pulse Generator, Patch Chords, IC 7400 NAND gate IC, IC 7402 NOR gate IC, IC 7404 NOT gate IC, LED.

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

FLIP-FLOP:

"Flip-flop" is the common name given to two-state devices which offer basic memory for sequential logic operations. Flip-flops are heavily used for digital data storage and transfer and are commonly used in banks called "register" for the storage of binary numerical data.

- **D-Type flip-flop (Toggle switch) :**

- The operations of a D flip-flop are much more simpler.
- It has only one input addition to the clock. It is very useful when a single data bit (0 or 1) is to be stored.
- If there is a HIGH on the D input when a clock pulse is applied, the flip-flop SETs and stores a 1.
- If there is a LOW on the D input when a clock pulse is applied, the flip-flop RESETs and stores a 0.
- To implement D flip-flop we require NAND gates and NOR gates.
- The truth table for D flip flop is shown below:

Q D	Q(t+1)
0 0	0
0 1	1
1 0	0
1 1	1

- **J-K flip-flop:**

- The J-K flip-flop works very similar to S-R flip-flop.
- The only difference is that this flip-flop has NO invalid state.
- The truth table is shown below.

Q	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

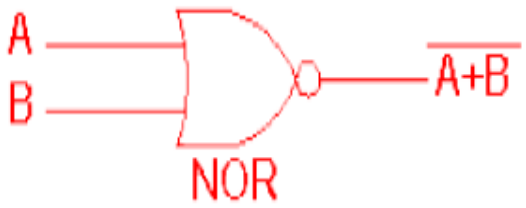
- **T flip-flop:**

  - This type of flip-flop is a simplified version of the JK flip-flop.
  - It is not usually found as an IC chip by itself, but is used in many kinds of circuits, especially counter and dividers.
  - Its only function is that it toggles itself with every clock pulse (on either the leading edge, on the trailing edge) it can be constructed from the RS flip-flop.
  - The truth table is shown below.

Q	T	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

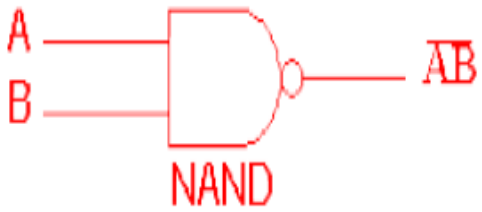
Truth Tables of Basic gates used in experiment:

- **NOR gate**



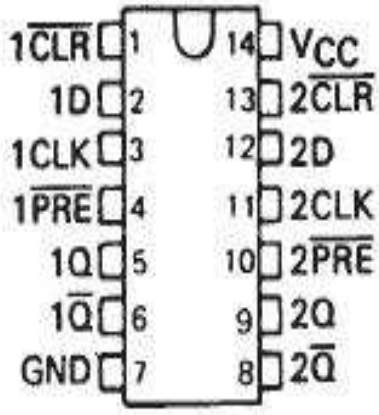
2 Input NOR gate		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

- **NAND gate**

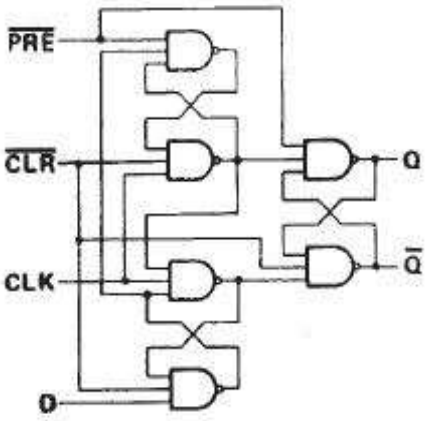


2 Input NAND gate		
A	B	$\overline{A.B}$
0	0	1
0	1	1
1	0	1
1	1	0

Pin Diagrams of Basic gates ICs used in experiment:



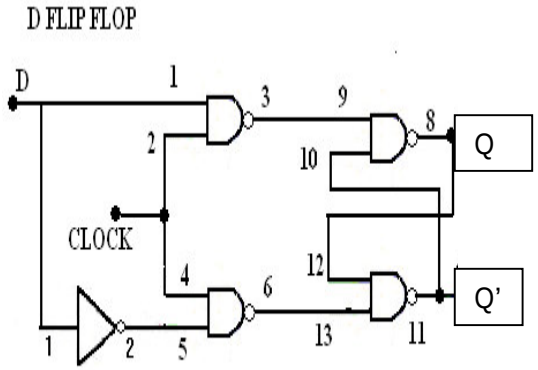
IC – 7474



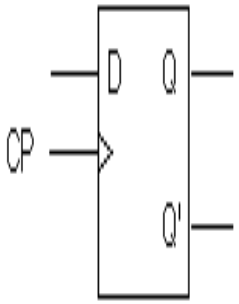
Logic diagram of IC-7474

Fig:.9.1 Pin Diagrams IC – 7474

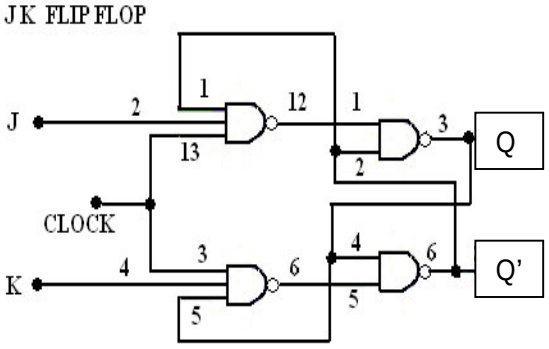
3.2 Designing the Solution:



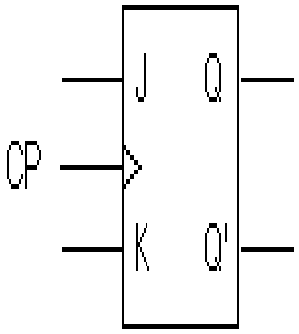
Logic diagram D Flip flop



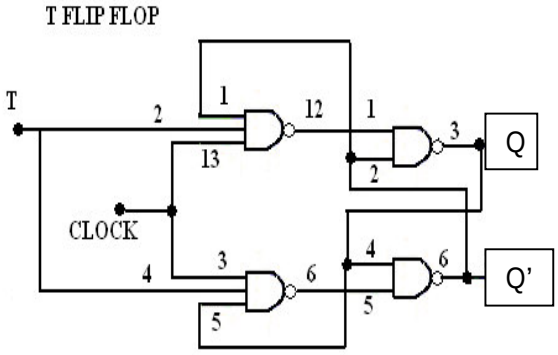
Graphical symbol



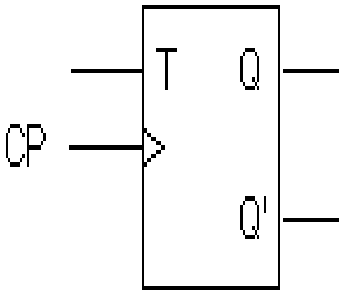
Logic diagram JK Flip flop



Graphical symbol



Logic diagram T Flip flop



Graphical symbol

Fig:.9.2 Logic diagram and Graphical symbol of D ,T & JK FlipFlop

3.3 Implementing the Solution:

D Flip-Flop: -

- The R –S Flip flop has two data inputs R & S.
- Generation of two signals to drive a flip flop is a disadvantage in much application.
- Furthermore, the forbidden condition of both R and S high may occur inadvertently.
- This has lid to the D Flip Flop a circuit that needs only a single data input.
- Fig shows the simple diagram of D Flip- Flop using NOR Gate.
- In this circuit the D input is just transferred to the output e.g. If D =0 then output Q is
- also & If D = 1 output is also 1, as shown in the truth table.

J-K Flip-flop:-

- JK Flip-Flop is the most versatile binary strange element.
- It can perform all the functions of SR and D flip-flop. The uncertainty in the State of SR Flip- Flop when S = R = 1 can be eliminated by using JK Flip-Flop

➢ **Procedure: -**

1. Study the circuit diagram.
2. Connect the circuit as shown in fig i.e. JK Flip Flop by using connecting wires.
3. Switch 'ON' the power supply.
4. Apply proper I/P to J & K I/Ps of Flip-Flop from Logic I/P
5. Check the O/P on Logic O/P Section.
6. Change the I/P & Verify the Truth Table.

T Flip –Flop: -

- The basic digital memory circuit is known as flip flop.
- It two stable states which are known as the 1 state 0 state. It can be obtained by using NAND or NOR gates.
- Generally there are two inputs to the flip flops (R, S or J K) and two outputs Q and Q.
- The outputs Q and Q are always complementary. The circuit has two stable state Q=1which is referred to as the 1 state(or set state) whereas in the other stable state Q=0 which is referred to as the 0 sate (or reset state)
- If the circuit is in 1 state. It continues to remain in this state and similarly if it is in 0 state, it continues to remain in this state.
- This property of the circuit is referred to as memory, that is it can store 1 bit of digital information.
- In a JK flip flop, if J=K the resulting flip flop is referred to as a T Flip Flop, as shown in fig.
- It has only input, referred to as T input. Its truth table is given in table 1. If T=1 it acts as a toggle witch for every clock pulse the output Q changes.

3.4 Testing the Solution:

(1) Observation Table for D Flip flop:

Input	Output
D	**Q**
0	
1	

(2) Observation Table for JK Flip flop:

Input		Output
J	**K**	**Q**
0	**0**	
0	**1**	
1	**0**	
1	**1**	

(3) Observation Table for T Flip flop:

Input	Output
T	**Q**
0	
1	

4 Conclusions: so using the ICs-7474 and gates we can study and verify the different flip flops like D, JK and T.

EXPERIMENT -10

1 AIM: To implement 3-bit and 4-bit binary counters.

2 TOOLS/APPARATUS: Power supply, Digital Trainer kit, ICs – 7493, Connecting wires, Multimeter, CRO, Clock Pulse Generator, Breadboard etc.

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

A 3 bit binary counter count from 0 to 7 and a 4-bit binary counter count from 0 to 15
To implement binary counter we require 7493 IC.Pin Diagrams of these IC is as below.

Pin Diagrams of Basic gates ICs used in experiment:

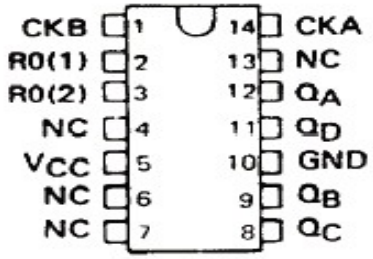
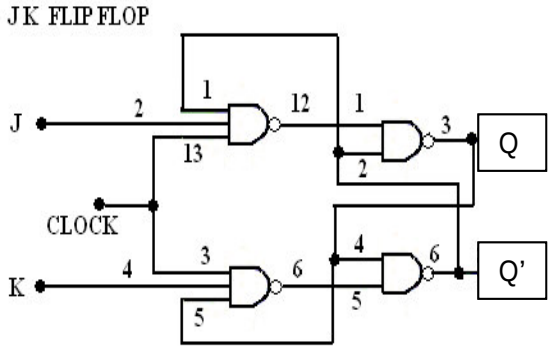
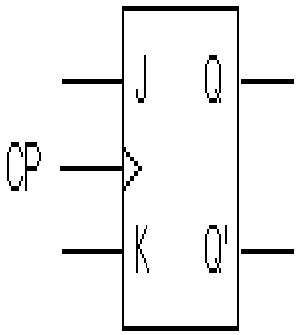


Fig:.10.1 Pin Diagram of IC 7493

Truth Tables of Basic flip-flop used in experiment:



Logic diagram JK Flip flop



Graphical symbol

Q	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Transition table

3.2 Designing the Solution

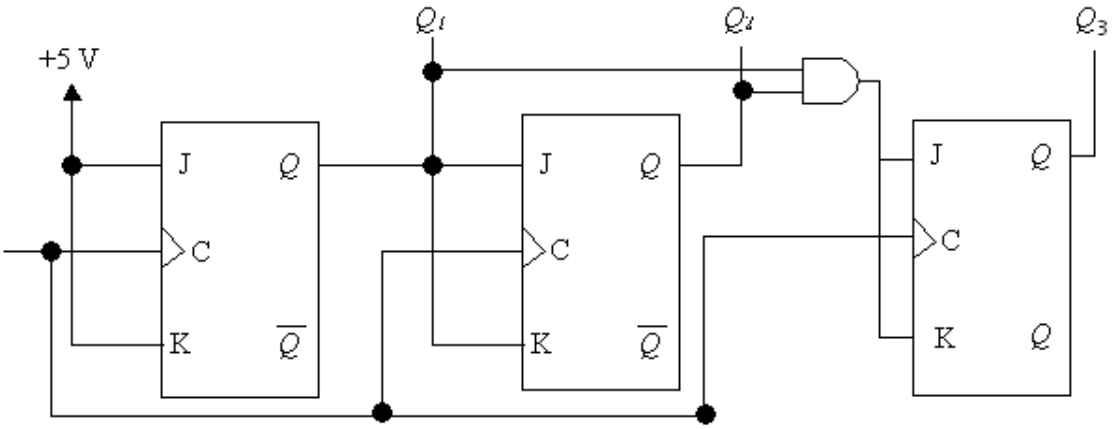


Fig:.10.2 Logic Diagram of 3-bit Counter

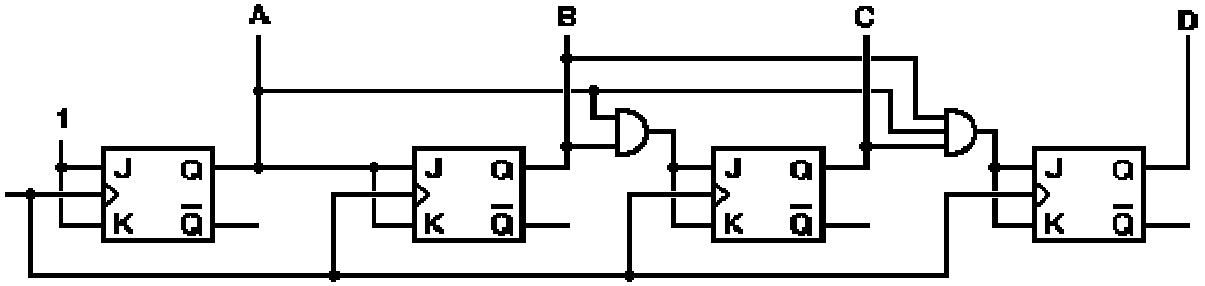


Fig:.10.3 Logic Diagram of 4-bit Counter

3.3 Implementing the Solution

1. Draw block diagram, truth table and pin diagram of 7493 IC.
2. According to pin assignment, connections are done to above mentioned ICs.
3. Note down the output reading on the multimeter according to the truth tables for the 3-Bit Counter.
4. For 3-Bit counter, clock pulse is applied at C input and outputs are measured from Q1, Q2, Q3 of 7493 IC.
5. For 4-Bit counter, clock pulse is applied at A input, B input and Qa is shorted, outputs are measured from Qa, Qb, Qc, Qd of 7493 IC.
6. Wave forms can be observed on CRO for the outputs of Counter.

3.4 Testing the Solution

Observation Table:

Count	3-bit Counter		
	Q3	Q2	Q1
0			
1			
2			
3			
4			
5			
6			
7			

Count	4-bit Counter			
	QD	QC	QB	QA
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				

4 Conclusions

By using 7493 ICs we can perform 3-bit and 4-bit binary counter and check the truth table.

EXPERIMENT: 11

1AIM: To implement BCD counter.

2 TOOLS/APPARATUS: Power supply, Digital Trainer kit, ICs – 7490, Connecting wires, Multimeter, CRO, Clock Pulse Generator, Breadboard etc.

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

A BCD counter is nothing but a mod-10 counter. It requires four FFs.States 0000 through 1001 are stable. After the tenth clock pluse, the counter resets. States 1010 through 1111are Invalid. The excitation requirements are as follows

PS				NS				Required excitation							
Q4	Q3	Q2	Q1	Q4	Q3	Q2	Q1	J4	K4	J3	K3	J2	K2	J1	K1
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	1	X
1	0	0	1	0	0	0	0	X	1	0	X	0	X	X	1

To implement BCD counter we require 7490 IC.Pin Diagrams of these IC is as below.

Pin Diagrams of Basic gates ICs used in experiment:

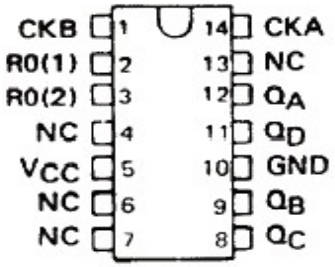
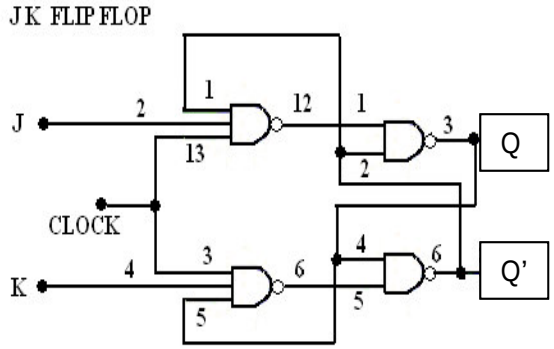
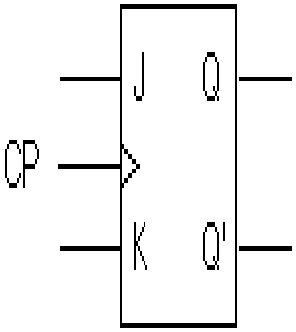


Fig.11.1 Pin Diagram of 7490 IC(BCD Counter)

Truth Tables of Basic flip-flop used in experiment:



Logic diagram JK Flip flop



Graphical symbol

Q	J	K	Q(t+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Transition table

3.2 Designing the Solution:

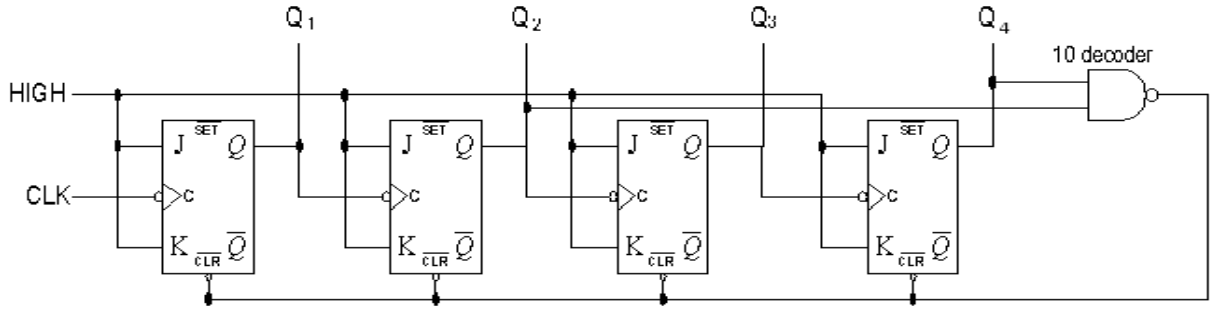


Fig.11.2 BCD Counter

3.3 Implementing the Solution

1. Draw block diagram, truth table and pin diagram of 7490 IC.
2. According to pin assignment, connections are done to above mentioned ICs.
3. Note down the output reading on the multimeter according to the truth tables for the 4-Bit BCD Counter.
4. Wave forms can be observed on CRO for the outputs of Counter.

3.4 Testing the Solution

Truth table for BCD counter

Count	Output			
	Q4	Q3	Q2	Q1
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				

4 Conclusions

By using 7490 ICs we can perform BCD counter and check the truth table.

EXPERIMENT-12

Aim: Design of the 11011 Sequence Detector.

Tools / Apparatus: Power supply, Digital Trainer kit, ICs, Connecting wires, Multimeter, CRO, Clock Pulse Generator, etc.

Procedure:

1. Derive the state diagram and state table for the circuit.
2. Count the number of states in the state diagram (call it N) and calculate the number of flip-flops needed (call it P) by solving the equation $2^{P-1} < N \square 2^{P}$. This is best solved by guessing the value of P.
3. Assign a unique P-bit binary number (state vector) to each state.
   Often, the first state = 0, the next state = 1, etc.
4. Derive the state transition table and the output table.
5. Separate the state transition table into P tables, one for each flip-flop.
   WARNING: Things can get messy here; neatness counts.
6. Decide on the types of flip-flops to use. When in doubt, use all JK's.
7. Derive the input table for each flip-flop using the excitation tables for the type.
8. Derive the input equations for each flip-flop based as functions of the input and current state of all flip-flops.
9. Summarize the equations by writing them in one place.
10. Draw the circuit diagram. Most homework assignments will not go this far,

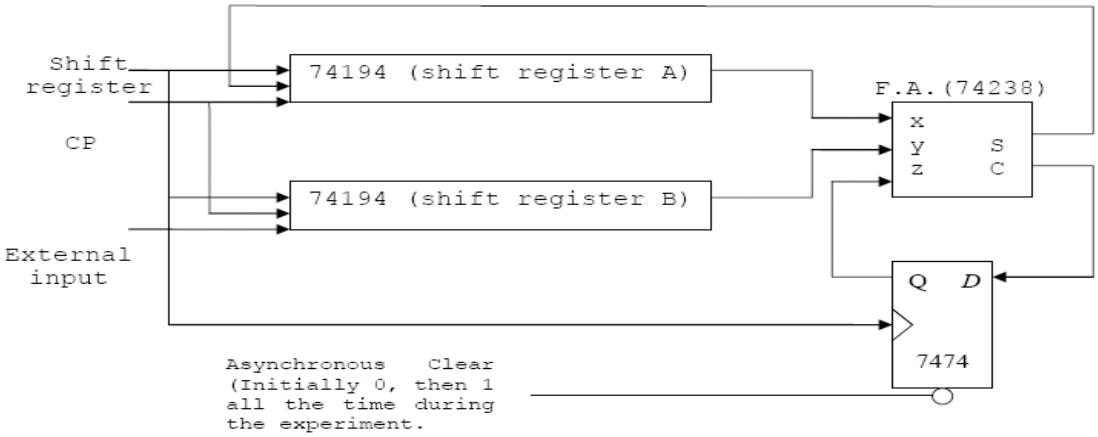
EXPERIMENT-13

Aim: To Design a Serial Adder.

Tools / Apparatus: Power supply, Digital Trainer kit, ICs-74194, 7474,7493, Connecting wires, Multimeter, CRO, Clock Pulse Generator, etc.

Procedure:

1. The function diagram is:



2. Note that the shift register works as the following:

S1	So	Function
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel Load

3. Derive the wiring diagram.
4. Connect the circuit given by the function diagram according to the derived wiring diagram.

References:

Reference Books

- Design of Digital circuits by Morris Mano
- Fundamentals of Digital Circuits by Anand Kumar