LAB MANUAL

FOR

DIGITAL ELECTRONICS

&

MICROPROCESSOR LAB

For

5th sem, Electrical Engg. (Diploma)

GOVERNMENT POLYTECHNIC, BARGARH

Prepared by:

Lect. Rashmita Gouda (Instrumentation & Control Engg.)

&

Lab Asst. Tophan Jena (Instrumentation & ControlEngg.)

COURSE CODE: PR.3

LIST OF EXPERIMENT

DIGITAL ELECTRONICS

1. Verification of truth tables of AND, OR, NOT, NOR, NAND, XOR, XNOR gates.
2. Implementation of various gates by using universal NAND & NOR gates and verification of truth table.
3. a) Implementation half adder and Full adder using logic gates.

   b) Implementation half subtractor and Full subtractor using logic gates.
4. Implementation a 4-bit Binary to Gray code converter.
5. Implementation a Single bit digital comparator.
6. Study Multiplexer and demultiplexer.
7. Study of flip-flops. i) S-R flip flop ii) J-K flip flop iii) D-flip flop iv) T flip flop
8. a) Realization of a 4-bit asynchronous UP/Down counter with a control for up/down counting.

   b) Realization a 4-bit synchronous UP/Down counter with a control for up/down counting.

   c) Implementation Mode-10 asynchronous counters.
9. Study of shift registers.

MICROPROCESSOR GENERAL PROGRAMMING USING 8085A DEVELOPMENT BOARD.

10. a) Programming to find 1'S Complement of a number for 8085MP.

    b) Programming to find 2'S Complement of a number for 8085MP.
11. a) Programming for Addition of 8-bit number.

    b) Programming for Subtraction of 8-bit number resulting 8/16 bit number.
12. a) Programming for decimal Addition of 8-bit numbers.

    b) Programming for decimal Subtraction 8-bit number
13. a) Programming for comparison between two numbers.

    b) Programming for finding the largest in an Array
14. Programming for Block Transfer in 8085MP.
15. Programming for Traffic light control using 8255.
16. Programming for Generation of square wave using 8255.

EXPERIMENT NO: 1

Aim of the Exp.: To verify truth tables of AND, OR, NOT, NOR, NAND, XOR, XNOR gates.

Apparatus Required:

Sl.No.	Name of the Equipment	Specification	Quantity
1	Analog Digital Trainer	DL-ADBT	1
2	Logic gate panel		1
3	Patch Cord		As per requirement

Theory:

Logic Gate

A logic gate is defined as a logic circuit with one output and two (or) more logic inputs. The output signal occurs only for combination of input variables.

There are basically three logic gates

a. AND gate

b. OR gate

c. NOT gate

These three gates are called as fundamental gates, because these gates form building block for most probably all digital circuits. From these fundamental gates other three gates are derived. They are called derived gates.

a. NAND gate

b. NOR gate

c. XOR gate

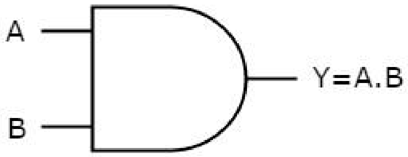
These logic gates are generally terms as decision making elements.

Basic Gates

AND Gate

An AND gate has two (or) more logic inputs and single output. The AND operation is defined as the output as (1) one if and only if all the inputs are (1) one. **7408** is the two inputs AND gate IC. A and B are the Input terminals & Y is the Output terminal.

Its logical equation is **Y = A.B**



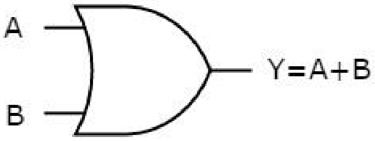
The truth table of two input AND gate is given as

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

1	0	0
1	1	1

OR Gate

An OR gate has two (or) more logic inputs and single output. The OR operation is defined as the output as (1) one if one or more than inputs are (1) one. **7432** is the two Input OR gate IC. A & B are the input terminals & Y is the Output terminal. Its logical equation is **Y = A + B**



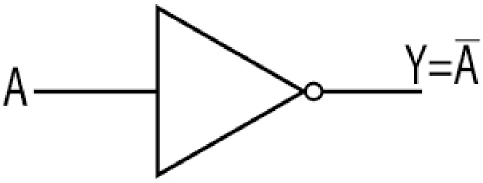
The truth table of two input OR gate is given as

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

NOT Gate

NOT gate has single input and single output. The NOT gate is also known as Inverter. It has one input (A) & one output (Y). IC No. is **7404**. Its logical equation is,

Y = A NOT B, Y = A'



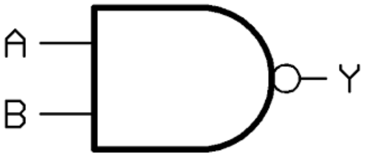
The truth table of NOT gate is given as

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

NAND Gate

An NAND gate has two (or) more logic inputs and single output. The combination of NOT gate and an AND gate is called as NAND gate. The IC no. for NAND gate is **7400**. The NOT-AND operation is known as NAND operation. If all inputs are 1 then output produced is 0. NAND gate is inverted AND gate.

Its logical equation is **Y = (A. B)'**



The truth table of two input NAND gate is given as

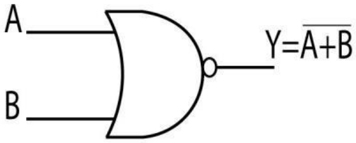
INPUT A	INPUT B	OUTPUT Y
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate

An NAND gate has two (or) more logic inputs and single output. The combination of NOT gate and an OR gate is called as NOR gate. IC **7402** is two I/P NOR gate IC. The NOT-OR operation is known as NOR operation. If all the inputs are 0 then the O/P is 1. NOR gate is inverted OR gate.

Its logical equation is

Y = (A+B)'



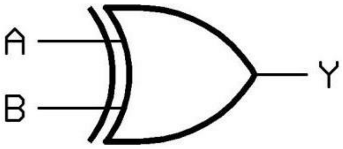
The truth table of two input NAND gate is given as

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

XOR Gate

The XOR gate has two (or) more logic inputs and single output. **7486** is two inputs XOR gate IC. A high output (1) results if both of the inputs to the gate are different. So it is also called as inequality detector.

Y=A'B+AB'



The truth table of two input XOR gate is given as

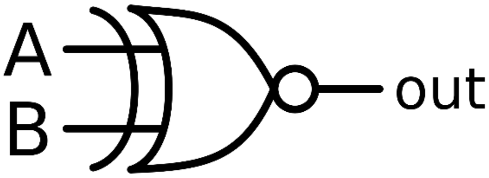
INPUT A	INPUT B	OUTPUT Y
0	0	0
0	1	1
1	0	1
1	1	0

XNOR Gate

The **XNOR gate** is a digital logic gate whose function is the logical complement of the Exclusive OR (XOR) gate. A high output (1) results if both of the inputs to the gate are the same. So it is also called as equality detector.

Its logical equation is

Y=AB+A’B’



The truth table of two input XOR gate is given as

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

Procedure:

1. The logic gate panel is placed over the slot on Analog Digital Trainer Kit.

2. +5V and GND is connected from the panel to the Analog Digital Trainer Kit using patch cord.

3. The inputs & outputs are connected to the logic gate from trainer kit input section and output section respectively through patch cords.

4. Using toggle switches of input section, different input combinations are given to the logic gate and output is verified from the truth table.

5. Step 3 & 4 are repeated for all the logic gates.

Observation:

AND Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

OR Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

NOT Gate

Input A	Output Y
L	
H	

NAND Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

NOR Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

X-OR Gate

Input A	Input B	Output Y
L	L	
L	H	

H	L	
H	H	

X-NOR Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

Conclusion:

EXPERIMENT NO: 2

Aim of the Exp.: To implement various gates by using universal NAND & NOR gates and verification of truth table.

Apparatus Required:

Sl.No.	Name of the Equipment	Specification	Quantity
1	Digital IC Trainer		1
2	Patch Cord		As per requirement

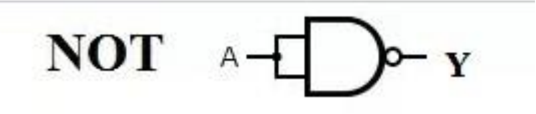
Theory:

Universal gate

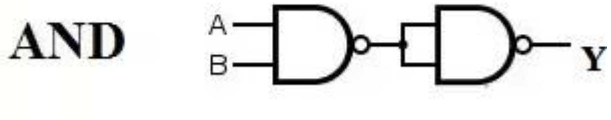
The NAND and NOR gates are called Universal gates, because all the logic gates andlogic functions can be implemented using NAND and NOR gates.

NAND Gate as Universal gate

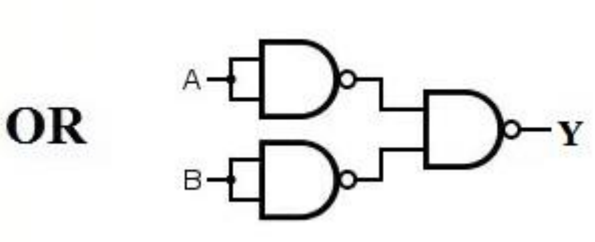
a. **NAND GATE AS INVERTER:** The circuit diagram of implementation of NANDgate as inverter is shown below:



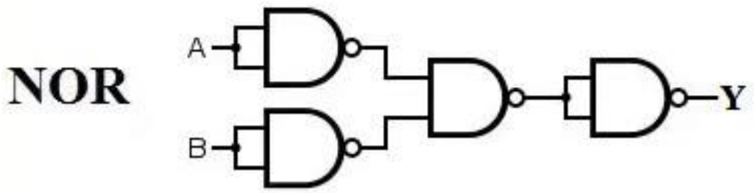
b. **NAND GATE AS AND GATE:** The circuit diagram of implementation of NANDgate as AND gate is shown below:



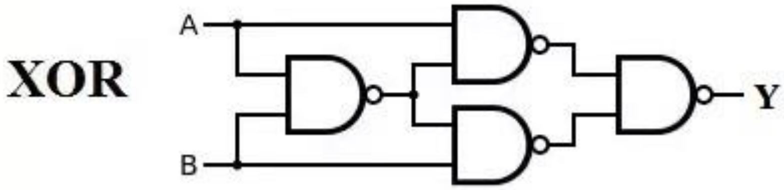
c. **NAND GATE AS OR GATE:** The circuit diagram of implementation of NAND gateas OR gate is shown below:



d. **NAND GATE AS NOR GATE:** The circuit diagram of implementation of NANDgate as is shown below:



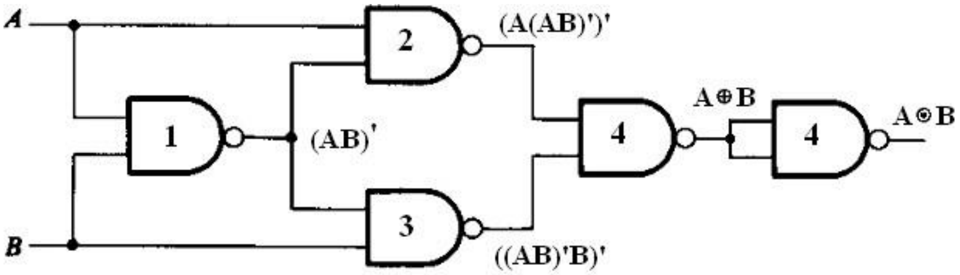
e. **NAND GATE AS EX-OR GATE** The circuit diagram of implementation of NANDgate as is shown below:



NAND gates as X-NOR gate:

X- NOR gate is actually X-OR gate followed by NOT gate. So give the output of X-OR gate to a NOT gate, overall output is that of an X-NOR gate.

Y = AB+ A’B’



X-NOR

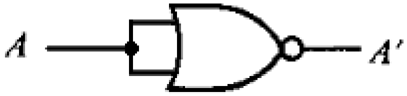
NOR Gate as Universal gate

NOR gates as NOT gate:

A NOT produces complement of the input. It can have only one input, tie the inputs of a NOR gate together. Now itwill work as a NOT gate. Its output is

Y = (A+A)’

Y = (A)’



NOT

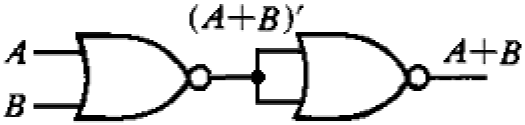
NOR gates as OR gate:

A NOR produces complement of OR gate. So, if the output of a NOR gate is inverted, overall output will be that ofan OR gate.

Y =

((A+B)’)’Y

= (A+B)



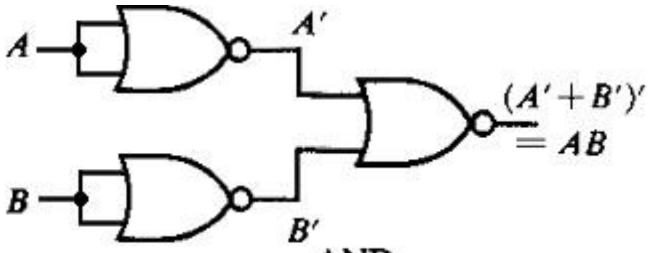
OR

NOR gates as AND gate:

From DeMorgan’s theorems: (A+B)’ = A’B’

(A’+B’)’ = A’’B’’ = AB

So, give the inverted inputs to a NOR gate, obtain AND operation at output.

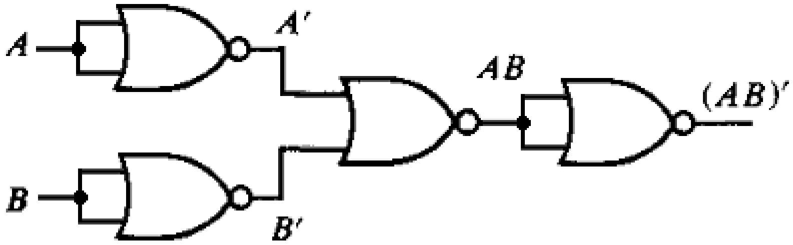


AND

NOR gates as NAND gate:

A NAND gate is an AND gate followed by NOT gate. So connect the output of AND gate to a NOT gate, overalloutput is that of a NAND gate.

Y = (AB)’

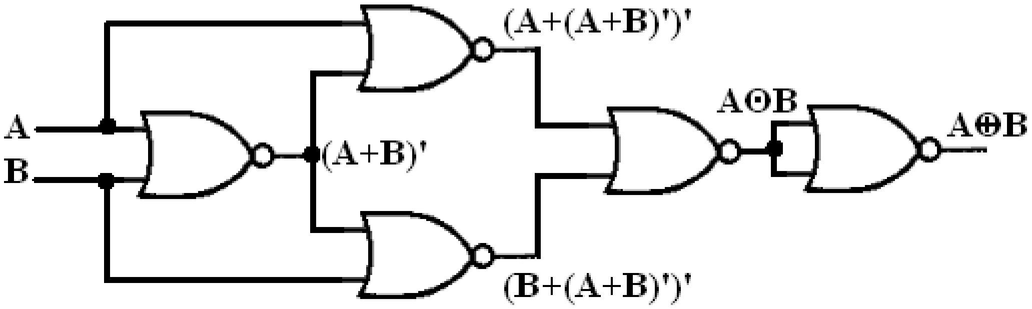


NAND

NOR gates as X-OR gate:

X- OR gate is actually X-NOR gate followed by NOT gate. So give the output of X-NOR gate to a NOT gate, overall output is that of an X-OR gate.

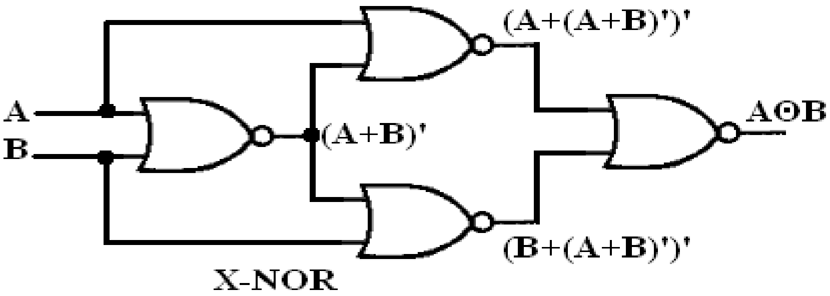
Y = A’B+ AB’



X- OR

NOR gates as X-NOR gate:

The output of a two input X-NOR gate is shown by: Y = AB + A'B'.



Procedure:

1. Connect the trainer kit to ac power supply.
2. Connect the NOR gates for any of the logic functions to be realised.
3. Connect the inputs of first stage to logic sources and output of the last gate to logic indicator.
4. Apply various input combinations and observe output for each one.
5. Verify the truth table for each input/ output combination.
6. Repeat the process for all logic functions.
7. Switch off the ac power supply.

Observation:

<u>NAND gate as Universal Gate</u>

NAND gate as AND Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

NAND gate as OR Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

NAND gate as NOT Gate

Input A	Output Y
L	
H	

NAND gate as NOR Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

NAND gate as X-OR Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

NAND gate as X-NOR Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

NOR gate as Universal Gate

NOR gate as AND Gate

Input A	Input B	Output Y
L	L	
L	H	
H	L	
H	H	

NOR gate as OR Gate

Input A	Input B	Output Y
L	L	

L	H	
H	L	
H	H	

NOR gate as NOT Gate

Input	Output
A	Y
L	
H	

NOR gate as X-OR Gate

Input	Input	Output
A	B	Y
L	L	
L	H	
H	L	
H	H	

NOR gate as X-NOR Gate

Input	Input	Output
A	B	Y
L	L	
L	H	
H	L	
H	H	

Conclusion:

EXPERIMENT NO: 3(a)

Aim of the Exp.: To Implement half adder and Full adder using logic gates.

Apparatus Required:

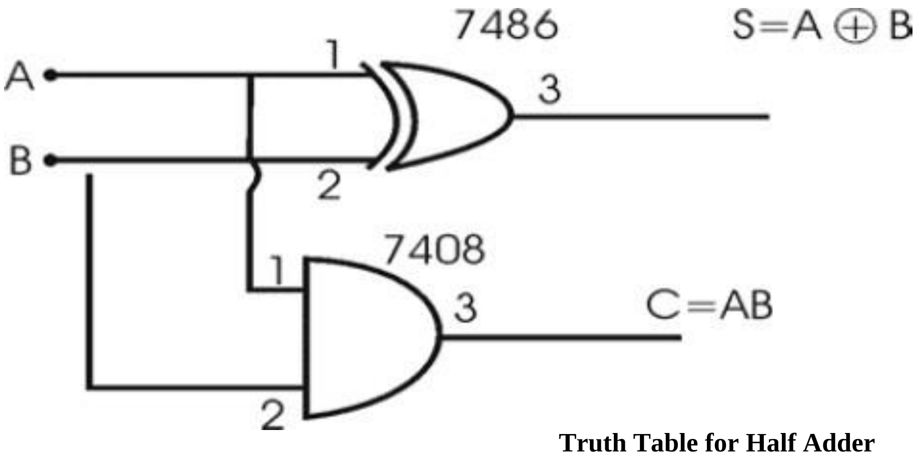
Sl.No.	Name of the Equipment	Specification	Quantity
1	Analog Digital Trainer	DL-ADBT	1
2	HA,FA/HS,FS panel		1
3	Patch Cord		As per requirement

Theory:

Half-Adder: A combinational logic circuit that performs the addition of two data bits, A and B, is called a half-adder. Addition will result in two output bits; one of which is the sum bit S, and the other is the carry bit, C. The Boolean functions describing the half-adder are:

$$S = A \oplus B$$

$$C = A.B$$

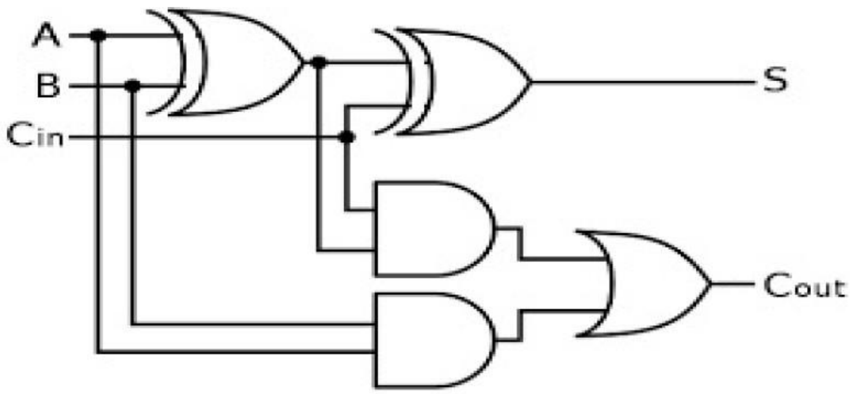


Truth Table for Half Adder

INPUT		OUTPUT	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Full Adder: The half-adder does not take the carry bit from its previous stage into account. This carry bit from its previous stage is called carry-in bit. A combinational logic circuit that adds two data bits, A and B, and a carry-in bit,$_{in}$ is called a full-adder. The Boolean functions describing the full-adder are:

$$S = A \oplus B \oplus C_{in}$$
$$C = A.B + C_{in}(A \oplus B)$$



INPUT A	INPUT B	INPUT C_{in}	$S = A \oplus B \oplus C_{in}$	$C = A.B + C_{in}(A \oplus B)$
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

Procedure:

1. The HA,FA/HS,FS panel is placed over the slot on Analog Digital Trainer Kit.

2. +5V and GND is connected from the panel to the Analog Digital Trainer Kit using patch cord.

3. The inputs & outputs are connected to the Half adder circuit from trainer kit input section and output section respectively through patch cords.

4. AC Supply is given to the kit.

5. Using toggle switches of input section, different input combinations are given to the logic gate and output is verified from the truth table.

6. Switch off the AC power supply.

5. Steps 3 to 6 are repeated for Full Adder.

Observation:

Truth Table for Half Adder

INPUT		OUTPUT	
A	**B**	**SUM**	**CARRY**
L	L		
L	H		
H	L		
H	H		

INPUT A	INPUT B	INPUT C_{in}	$S = A \oplus B \oplus C_{in}$	$C = A.B + C_{in}(A \oplus B)$
L	L	L		
L	L	H		
L	H	L		
L	H	H		
H	L	L		
H	L	H		
H	H	L		
H	H	H		

Conclusion:

EXPERIMENT NO: 3(b)

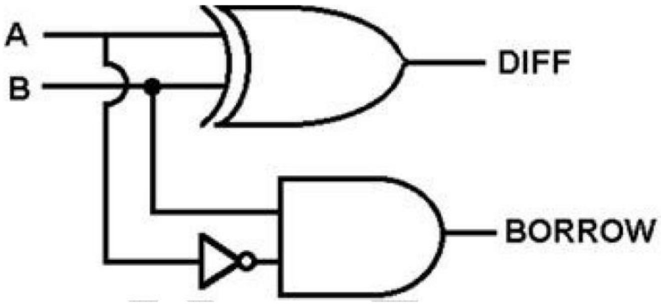
Aim of the Exp.: To Implement half subtractor and Full subtractor using logic gates.

Apparatus Required:

Sl.No.	Name of the Equipment	Specification	Quantity
1	Analog Digital Trainer	DL-ADBT	1
2	HA,FA/HS,FS panel		1
3	Patch Cord		As per requirement

Theory:

Half- subtractor: Subtracting a single-bit binary value B from another A (i.e., A-B) produces a difference bit D and a borrow out bit B_0. This operation is called half subtraction and the circuit to realize it is called a half subtractor. The Boolean functions describing the half-subtractor are:



$$D = A \oplus B$$
$$B_0 = A'B$$

INPUT A	INPUT B	$D = A \oplus B$	$B_0 = A'B$
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	0

Full-subtractor: Subtracting two single-bit binary values, B, C from a single-bit value A produces a difference bit D and a borrow out B_0 bit. This is called full subtraction. The Boolean functions describing the full subtractor.

$$D = A \oplus B \oplus C$$
$$B_0 = A'B + A'C + BC$$

INPUT A	INPUT B	INPUT C_{in}	$D = A \oplus B \oplus C$	$B_0 = A'B + A'C + BC$
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	0	1

Procedure:

1. The HA,FA/HS,FS panel is placed over the slot on Analog Digital Trainer Kit.

2. +5V and GND is connected from the panel to the Analog Digital Trainer Kit using patch cord.

3. The inputs & outputs are connected to the Half subtractor circuit from trainer kit input section and output section respectively through patch cords.

4. AC Supply is given to the kit.

5. Using toggle switches of input section, different input combinations are given to the logic gate and output is verified from the truth table.

6. Switch off the AC power supply.

5. Steps 3 to 6 are repeated for Full Subtractor.

Observation:

Truth Table for Half Subtractor

INPUT		**OUTPUT**	
A	**B**	**DIFFERENCE**	**BORROW**
L	L		
L	H		
H	L		
H	H		

Truth Table for Full Subtractor

INPUT A	INPUT B	INPUT C_{in}	$D = A \oplus B \oplus C$	$B_0 = A'B + A'C + BC$
L	L	L		
L	L	H		
L	H	L		
L	H	H		
H	L	L		
H	L	H		
H	H	L		
H	H	H		

Conclusion:

EXPERIMENT NO: 4(a)

Aim of the Exp.: To Implement a 4-bit Binary to Gray code converter.

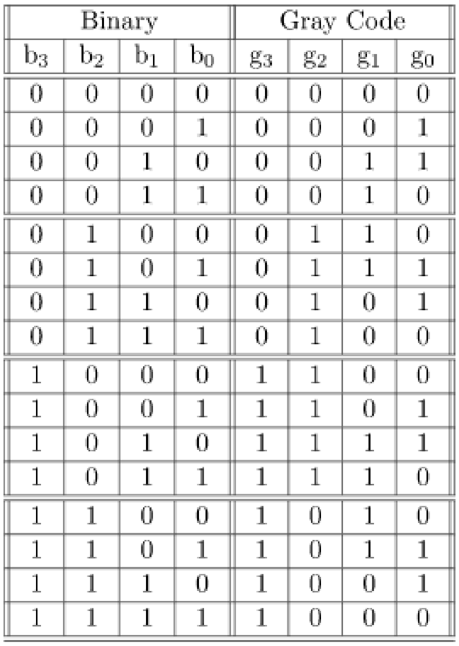
Apparatus Required:

Sl.No.	Name of the Equipment	Specification	Quantity
1	Digital IC Trainer		1
2	Patch Cord		As per requirement

Theory:

Gray code is also known as **Cyclic Code**, **Reflected Binary Code** (RBC), **Reflected Binary** (RB).

Grey code – It is defined as an ordering of the binary number system such that each incremental value can only differ by one bit. In gray code, while traversing from one step to another step only one bit in the code group changes. That is to say that two adjacent code numbers differ from each other by only one bit.



Binary				Gray Code			
b_3	b_2	b_1	b_0	g_3	g_2	g_1	g_0
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

Fig: Truth Table for Binary to gray code converter

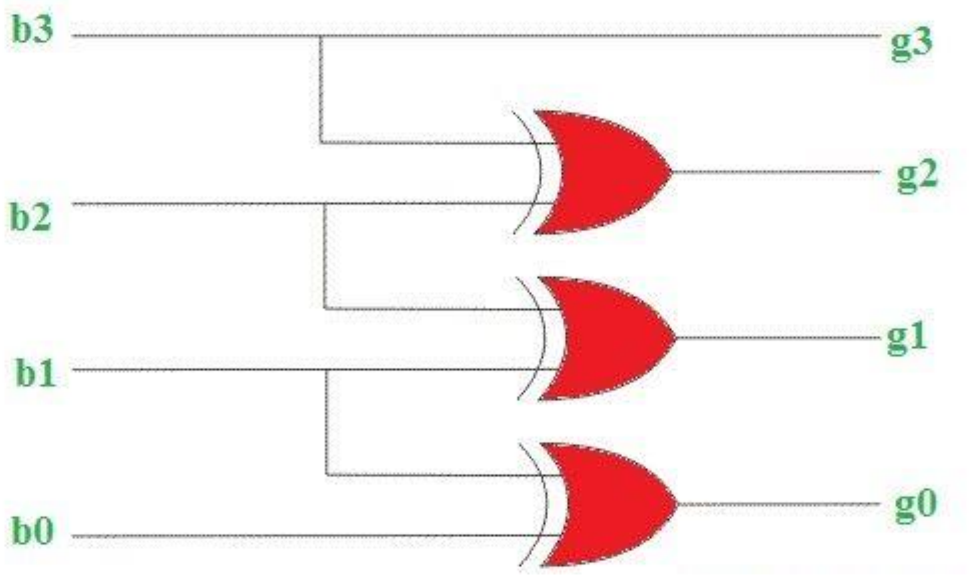


Fig: Circuit diagram of 4-bit binary to gray code converter

Procedure:

1. Connect the trainer kit to ac power supply.
2. Connect the ckt according to the ckt diagram with the help of patch cord.
3. Connect the inputs of the ckt and output to the input and output section of the trainer kit.
4. Switch on the AC Power supply.
5. Apply various input combinations and observe output for the ckt.
6. Verify the truth table for each input/ output combination.

Observation:

Binary Code				Gray Code			
b3	**b2**	**b1**	**b0**	**g3**	**g2**	**g1**	**g0**
L	L	L	L				
L	L	L	H				
L	L	H	L				
L	L	H	H				
L	H	L	L				
L	H	L	H				
L	H	H	L				
L	H	H	H				
H	L	L	L				
H	L	L	H				
H	L	H	L				
H	L	H	H				

H	**H**	**L**	**L**				
H	**H**	**L**	**H**				
H	**H**	**H**	**L**				
H	**H**	**H**	**H**				

Conclusion:

EXPERIMENT NO: 4(b)

Aim of the Exp.: To Implement a Single bit digital comparator.

Apparatus Required:

Sl.No.	Name of the Equipment	Specification	Quantity
1	Digital IC Trainer		1
2	Patch Cord		As per requirement

Theory:

Single Bit Comparator: A comparator used to compare two bits is called a single-bit comparator. It consists of two inputs each for two single-bit numbers and three outputs to generate less than, equal to, and greater than between two binary numbers.

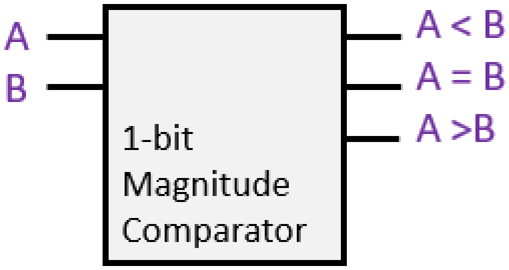
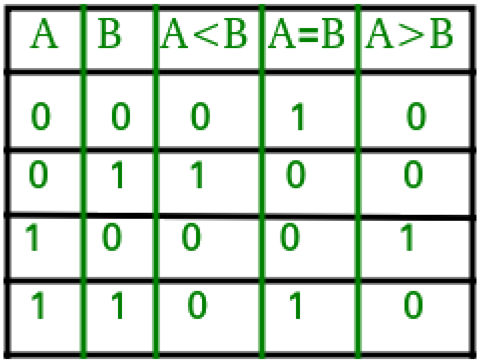


Fig: Block diagram of 1-bit Comparator



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

Fig: Truth Table of 1-bit comparator

From the above truth table logical expressions for each output can be expressed as follows:

A > B : AB'

A < B : A'B

A = B : A'B' + AB

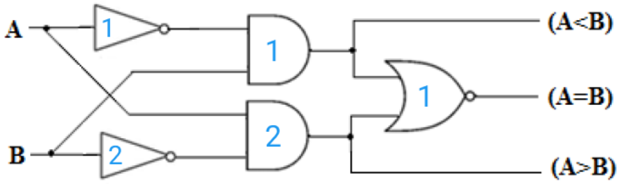


Fig. 1: One-bit Comparator

Procedure:

1. Connect the trainer kit to ac power supply.
2. Connect the ckt according to the ckt diagram with the help of patch cord.
3. Connect the inputs of the ckt and output to the input and output section of the trainer kit.
4. Switch on the AC Power supply.
5. Apply various input combinations and observe output for the ckt.
6. Verify the truth table for each input/ output combination.

Observation:

A	B	A<B	A=B	A>B
L	L			
L	H			
H	L			
H	H			

Conclusion:

EXPERIMENT NO: 6

Aim of the Exp.: To Study Multiplexer and demultiplexer

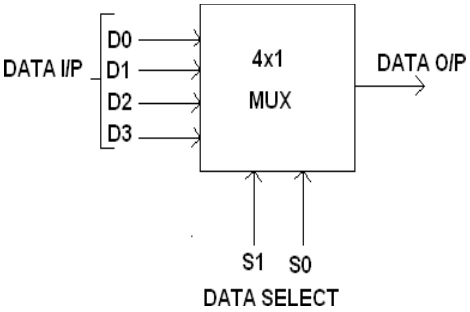
Apparatus Required:

Sl.No.	Name of the Equipment	Specification	Quantity
1	Analog Digital Trainer	DL-ADBT	1
2	MUX/DEMUX panel		1
3	Patch Cord		As per requirement

Theory:

Multiplexer

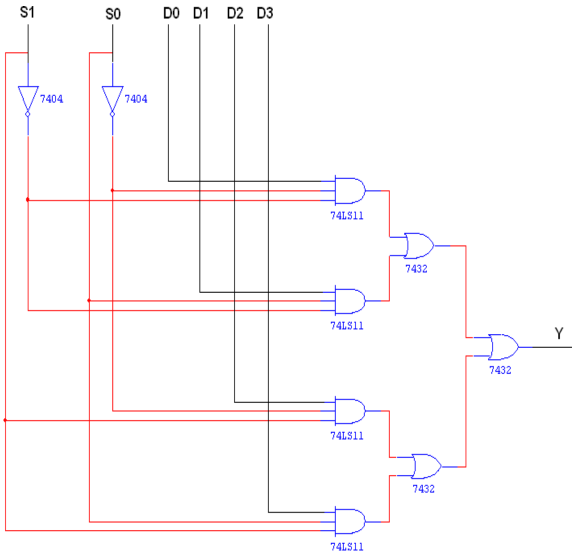
- A Multiplexer (or a data selector) is a logic circuit that accepts several data inputs and allows only one of them at a time to get through to the output. The selection of the desired data input is controlled by the SELECT (orADDRESS) INPUTS.
- Normally there are 2n input line and n selection lines whose bit combination determine which inputis selected. Figure below shows the block diagram of a Multiplexer.



In this diagram the inputs and outputs are indicated by means of broad arrows to indicate that there may be one or more lines. Depending upon the digital code applied at the SELECT inputs, one out of the data sources is selected and transmitted to the single output channel.

Function Table

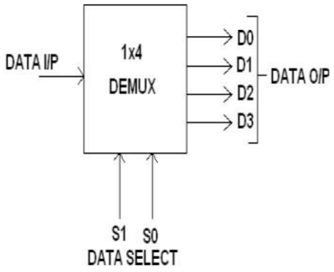
INPUT		OUTPUT
S1	**S0**	**Y**
0	0	D0
0	1	D1
1	0	D2
1	1	D3
Y = D0 S1’ S0’ +D1 S1’ S0 +D2 S1 S0’ +D3 S1 S0		



Demultiplexer

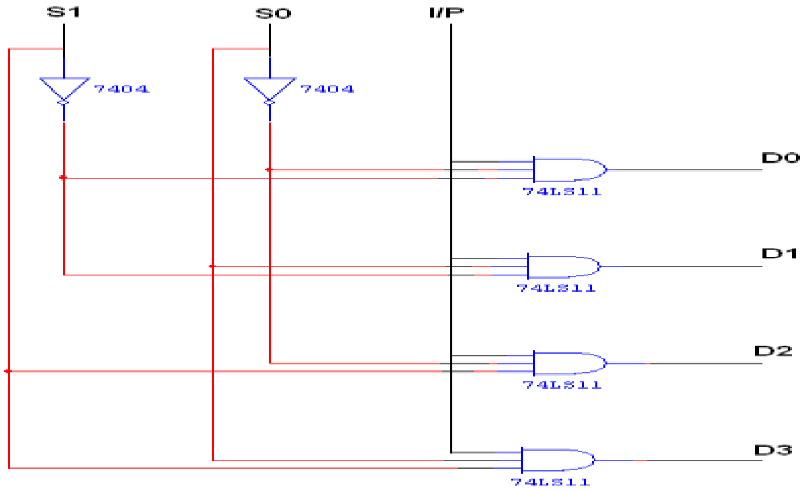
A Demultiplexer performs the reverse operation of a Multiplexer. It accepts a single input and distributes it over several outputs. The SELECT input code determines to which output the data input will be transmitted. The Demultiplexer becomes enabled when the strobe signal is active LOW.

BLOCK DIAGRAM FOR 1:4 DEMULTIPLEXER:



FUNCTION TABLE:

S1	S0	OUTPUT
0	0	D0 = X S1’ S0
0	1	D1 = X S1’ S0
1	0	D2 = X S1 S0’
1	1	D3 = X S1 S0
Y = X S1’ S0 + X S1’ S0 + X S1 S0’ + X S1 S0		



TRUTH TABLE

INPUT			OUTPUT			
S1	S0	I/P	D0	D1	D2	D3
0	0	0	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	0	0	0
0	1	1	0	1	0	0
1	0	0	0	0	0	0
1	0	1	0	0	1	0
1	1	0	0	0	0	0
1	1	1	0	0	0	1

Procedure:

1. The MUX/DEMUX panel is placed over the slot on Analog Digital Trainer Kit.

2. +5V and GND is connected from the panel to the Analog Digital Trainer Kit using patch cord.

3. The inputs & outputs are connected to the MUX circuit from trainer kit input section and output section respectively through patch cords.

4. AC Supply is given to the kit.

5. Using toggle switches of input section, different input combinations are given to the logic gate and output is verified from the truth table.

6. Switch off the AC power supply.

5. Steps 3 to 6 are repeated for DEMUX.

Observation:

MUX

INPUT		**OUTPUT**
S1	**S0**	**Y**
L	L	
L	H	
H	L	
H	H	

DEMUX

INPUT			OUTPUT			
S1	S0	I/P	D0	D1	D2	D3
L	L	L				
L	L	H				
L	H	L				
L	H	H				
H	L	L				
H	L	H				
H	H	L				
H	H	H				

Conclusion:

EXPERIMENT NO: 7

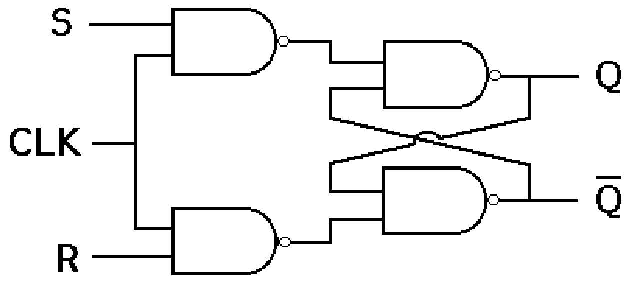
Aim of the Exp.: To Study flip-flops. i) S-R flip flop ii) J-K flip flop iii) D-flip flop iv) T-flip flop

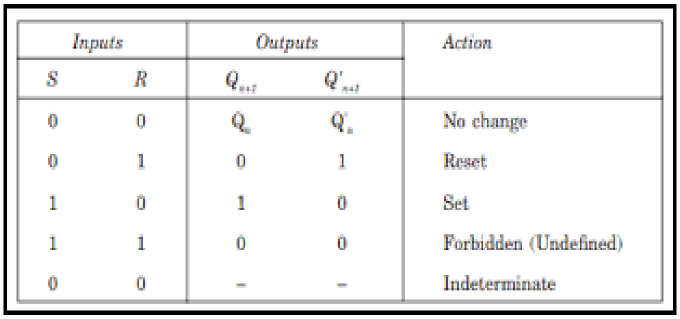
Apparatus Required:

Sl.No.	Name of the Equipment	Specification	Quantity
1	Analog Digital Trainer	DL-ADBT	1
2	Flipflop panel		1
3	Patch Cord		As per requirement

Theory:

SR FLIP-FLOP: There are two inputs to the flip-flop defined as R and S. When I/Ps R = 0 and S = 0 then O/P remains unchanged. When I/Ps R = 0 and S = 1 the flip-flop is switches to the stable state where O/P is 1 i.e. SET. The I/P condition is R = 1 and S = 0 the flip-flop is switched to the stable state where O/P is 0 i.e. RESET. The I/P condition is R = 1 and S = 1 the flip-flop is switched to the stable state where O/P is forbidden.

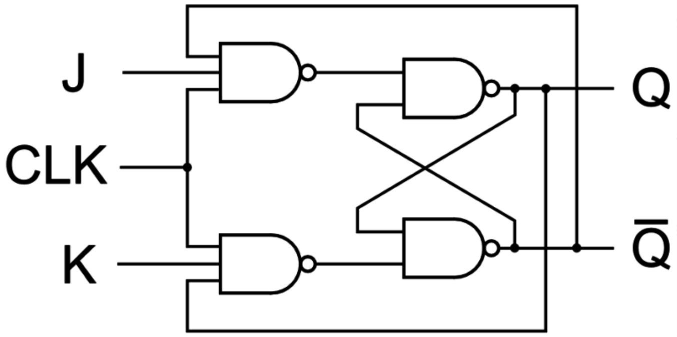




Inputs		Outputs		Action
S	R	Q_{n+1}	Q'_{n+1}	
0	0	Q_n	Q'_n	No change
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Forbidden (Undefined)
0	0	–	–	Indeterminate

Fig: Circuit diagram & Truth Table

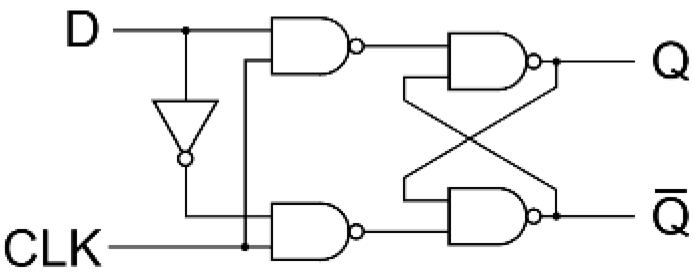
JK FLIP-FLOP: For purpose of counting, the JK flip-flop is the ideal element to use. The variable J and K are called control I/Ps because they determine what the flip- flop does when a positive edge arrives. When J and K are both 0s, both AND gates are disabled and Q retains its last value.



J	K	Q_n	Q_{n+1}	State
0	0	0	0	Q_n (Hold)
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	1	Toggel
1	1	1	0	

Fig: Circuit diagram & Truth Table

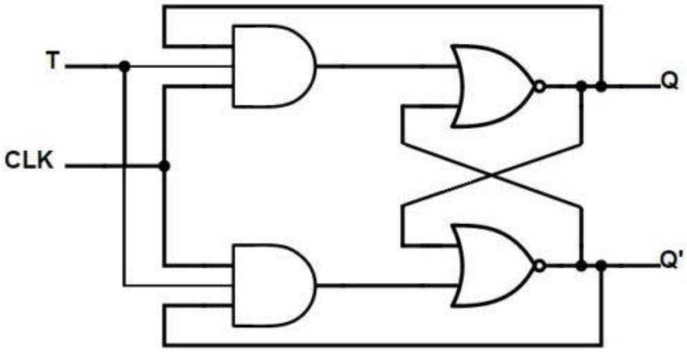
D FLIP-FLOP: This kind of flip flop prevents the value of D from reaching the Q output until clock pulses occur. This flipflop is designed by connecting S-R flipflop inputs through a NOT gate.



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

Fig: Circuit diagram & Truth Table

T FLIP-FLOP: The T or "toggle" flip-flop changes its output on each clock edge, giving an output which is half the frequency of the signal to the T input. It is useful for constructing binary counters, frequency dividers, and general binary addition devices. It can be made from a J-K flip-flop by tying both of its inputs high.



CLK	T	Q n	Q n+1
↓	0	0	0
↓	0	1	1
↓	1	0	1
↓	1	1	0

Fig: Circuit diagram & Truth Table

Procedure:

1. The Flipflop panel is placed over the slot on Analog Digital Trainer Kit.

2. +5V and GND is connected from the panel to the Analog Digital Trainer Kit using patch cord.

3. The inputs & outputs are connected to the S-R Flipflop circuit from trainer kit input section and output section respectively through patch cords.

4. AC Supply is given to the kit.

5. Using toggle switches of input section, different input combinations are given to the logic gate and output is verified from the truth table.

6. Switch off the AC power supply.

5. Steps 3 to 6 are repeated for J-K, D & T Flipflops.

Observation:

S-R Flipflop:

CLK	S	R	Q_n	Q_{n+1}
↑	L	L	L	
↑	L	L	H	
↑	L	H	L	
↑	L	H	H	
↑	H	L	L	
↑	H	L	H	
↑	H	H	L	
↑	H	H	H	

J-K Flipflop:

CLK	J	K	Q_n	Q_{n+1}
↑	L	L	L	
↑	L	L	H	
↑	L	H	L	
↑	L	H	H	
↑	H	L	L	
↑	H	L	H	
↑	H	H	L	
↑	H	H	H	

D Flipflop

CLK	D	Q_n	Q_{n+1}
↑	L	L	
↑	L	H	
↑	H	L	
↑	H	H	

T Flipflop

CLK	D	Q_n	Q_{n+1}
↑	L	L	
↑	L	H	
↑	H	L	
↑	H	H	

Conclusion:

EXPERIMENT NO: 8(a)

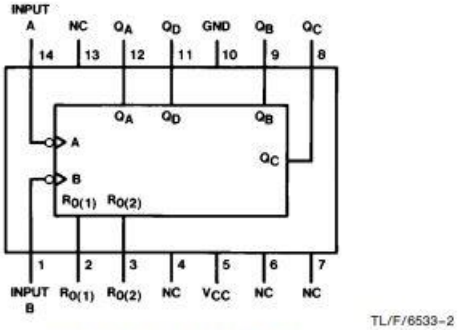
Aim of the Exp.: To Realize a 4-bit asynchronous UP/Down counter with a control for up/down counting.

Apparatus Required:

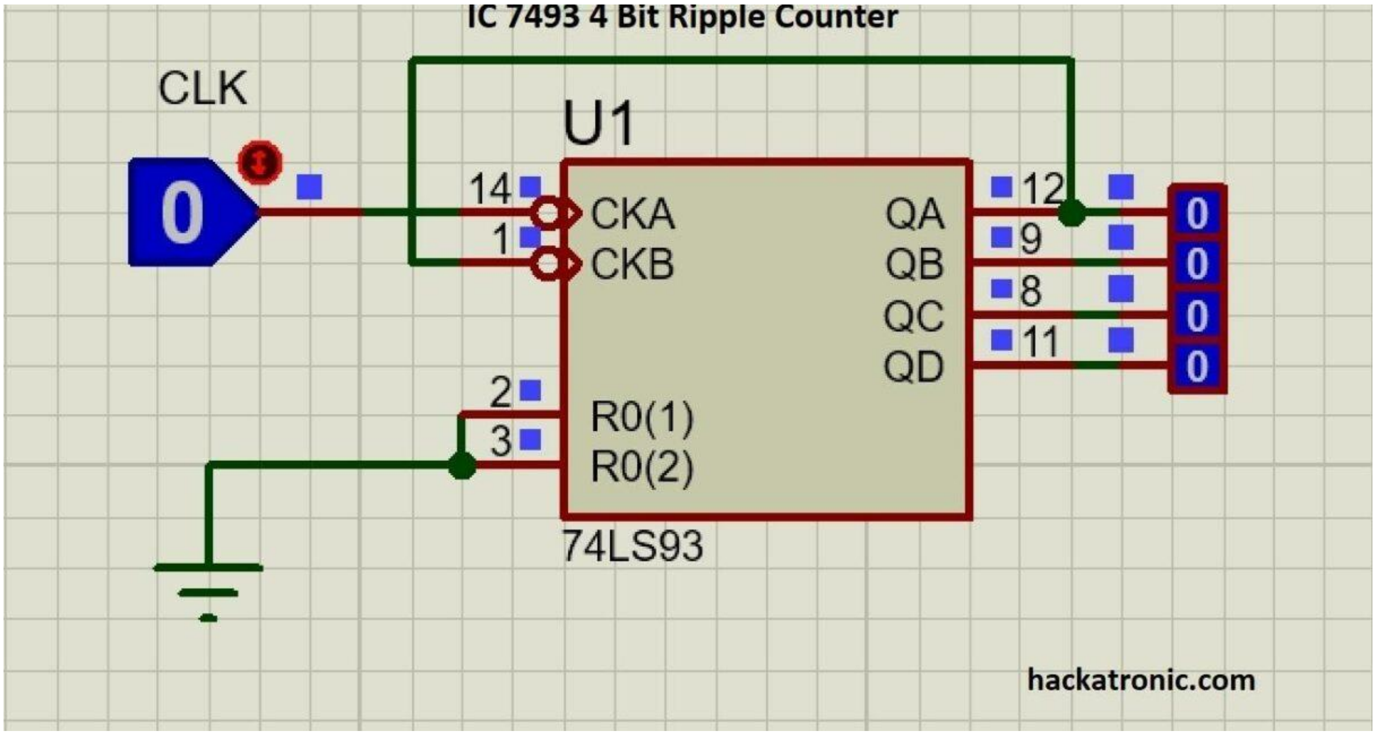
Theory:

IC 7493: 4-bit Asynchronous up/down counter

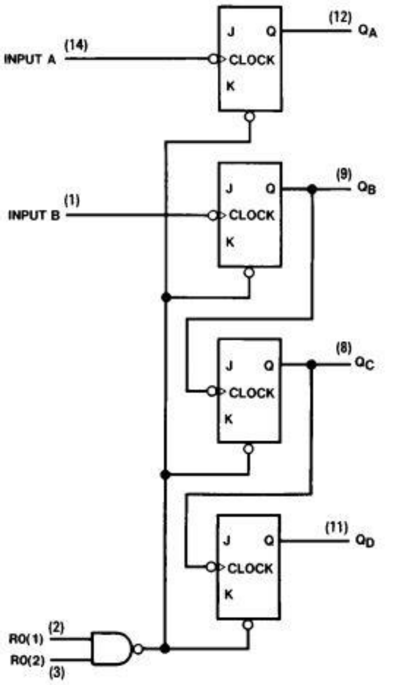
Dual-In-Line Package



Order Number DM7493AN
See NS Package Number N14A



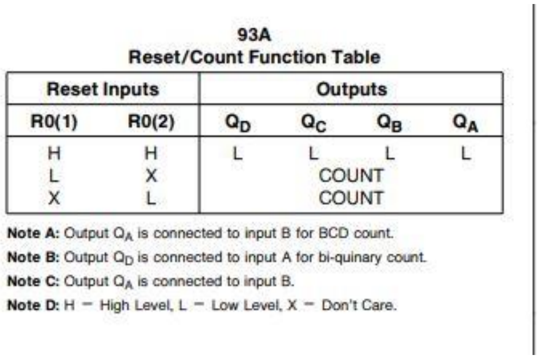
93A



TL/F/6533–4

93A
Count Sequence
(See Note C)

Count	Outputs			
	Q_D	Q_C	Q_B	Q_A
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H
8	H	L	L	L
9	H	L	L	H
10	H	L	H	L
11	H	L	H	H
12	H	H	L	L
13	H	H	L	H
14	H	H	H	L
15	H	H	H	H



93A
Reset/Count Function Table

Reset Inputs		Outputs			
R0(1)	R0(2)	Q_D	Q_C	Q_B	Q_A
H	H	L	L	L	L
L	X	COUNT			
X	L	COUNT			

Note A: Output Q_A is connected to input B for BCD count.
Note B: Output Q_D is connected to input A for bi-quinary count.
Note C: Output Q_A is connected to input B.
Note D: H — High Level, L — Low Level, X — Don't Care.

Procedure:

(i) IC 7493 Connections are given as per circuit diagram on the kit.
(ii) R_0 and R_1 combination is given for COUNT operation
(iii) Apply the clock pulse input and verify the output of the counter for each clock pulse.

Conclusion:

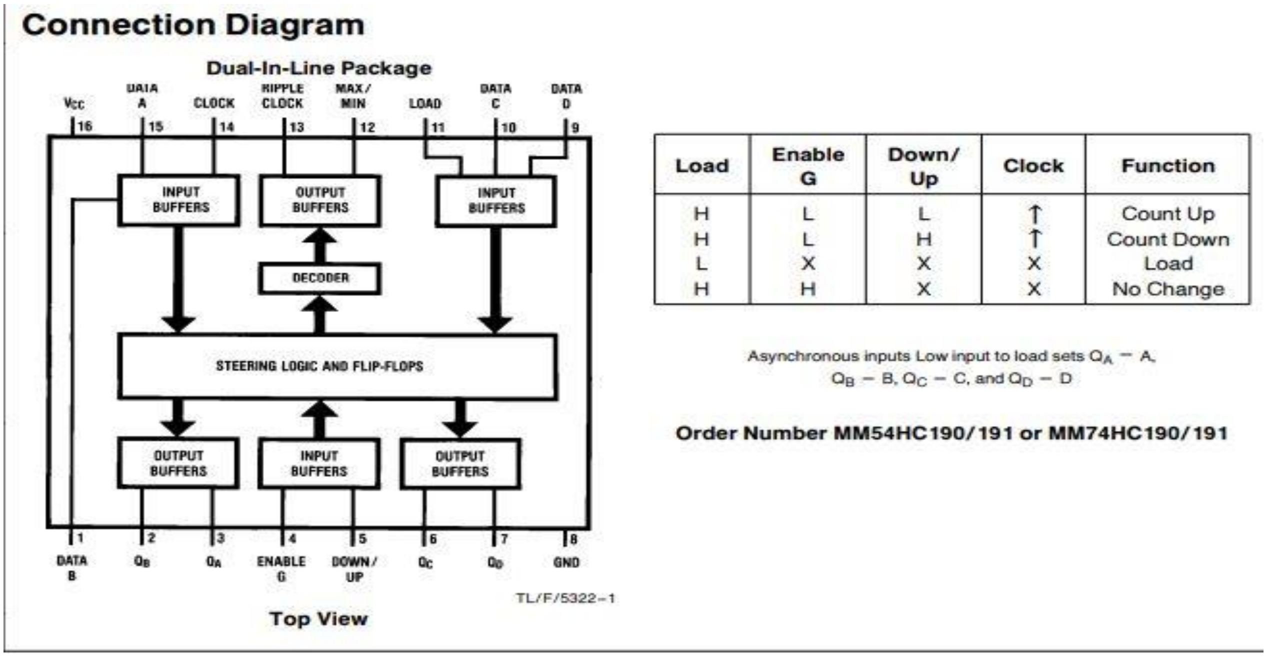
EXPERIMENT NO: 8(b)

Aim of the Exp.: To realize a 4-bit synchronous UP/Down counter with a control for up/down counting.

Apparatus Required:

Theory:

IC 74910: 4-bit Synchronous UP/DOWN Counter



Connection Diagram

Dual-In-Line Package

Top View

Load	Enable G	Down/ Up	Clock	Function
H	L	L	↑	Count Up
H	L	H	↑	Count Down
L	X	X	X	Load
H	H	X	X	No Change

Asynchronous inputs Low input to load sets Q_A = A, Q_B = B, Q_C = C, and Q_D = D

Order Number MM54HC190/191 or MM74HC190/191

Synchronous operation is provided by having all flip-flops clocked simultaneously, so that the outputs change simultaneously when so instructed by the steering logic. This mode of operation eliminates the output counting spikes normally associated with asynchronous (ripple clock) counters.

The outputs of the four master-slave flip-flops are triggered on a low-to-high level transition of the clock input, if the enable input is low. A high at the enable input inhibits counting.

The direction of the count is determined by the level of the down/up input. When low, the counter counts up and when high, it counts down. These counters are fully programmable; that is, the outputs may be preset to either level by placing a low on the load input and entering the desired data at the data inputs.

The output will change independent of the level of the clock input. This feature allows the counters to be used as modulo N dividers by simply modifying the count length with the preset inputs.

Procedure:

Conclusion:

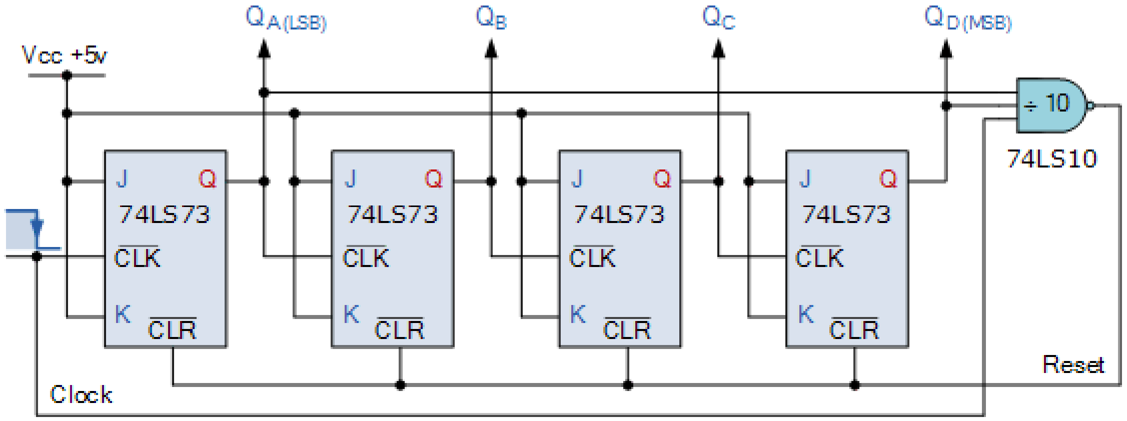
EXPERIMENT NO: 8(c)

Aim of the Exp.: To implement Mode-10 asynchronous counters.

Apparatus Required:

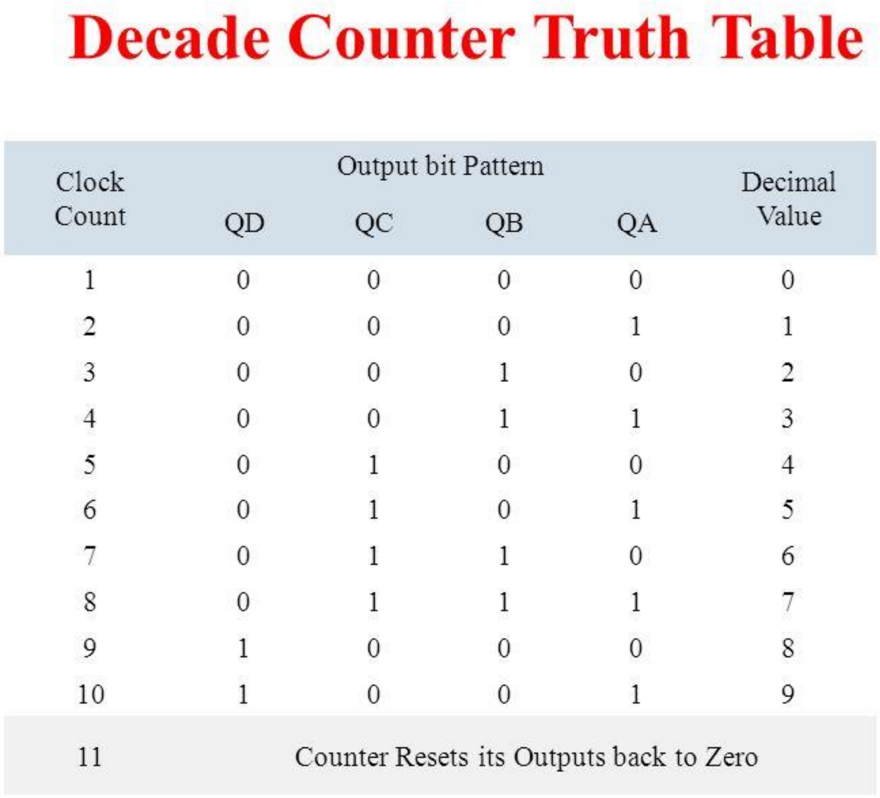
Theory:

Asynchronous Decade Counter



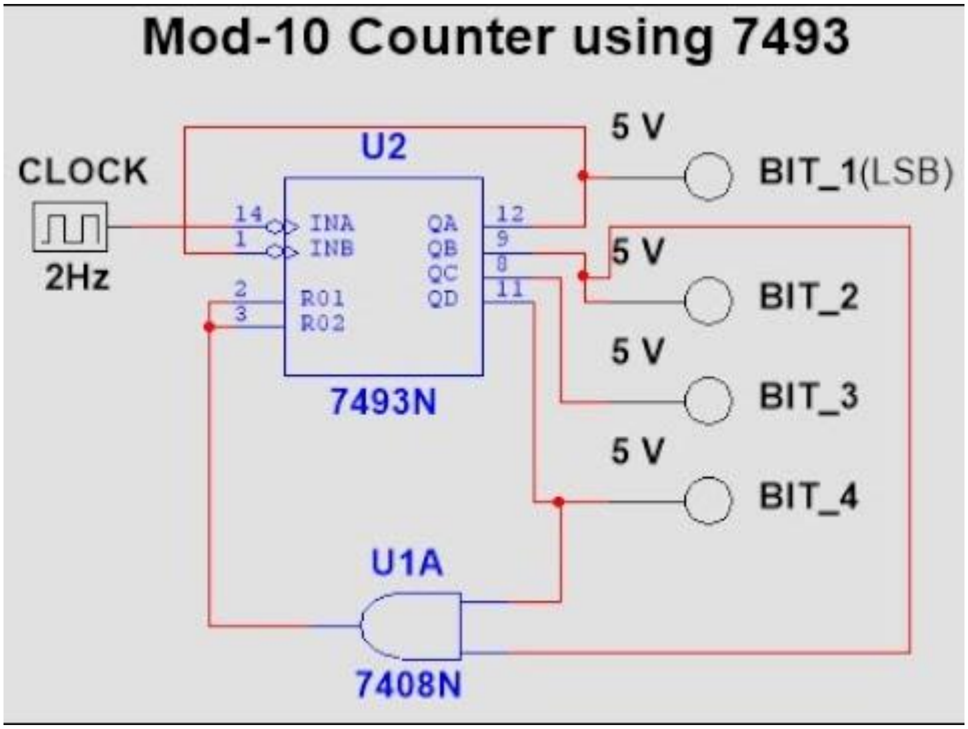
This type of asynchronous counter counts upwards on each trailing edge of the input clock signal starting from 0000 until it reaches an output 1001 (decimal 9). Both outputs QA and QD are now equal to logic “1”. On the application of the next clock pulse, the output from NAND gate changes state from logic “1” to a logic “0” level.

As the output of the NAND gate is connected to the CLEAR (CLR) inputs of all the J-K Flip-flops, this signal causes all of the Q outputs to be reset back to binary 0000 on the count of 10. As outputs QA and QD are now both equal to logic “0” as the flip-flop’s have just been reset, the output of the NAND gate returns back to a logic level “1” and the counter restarts again from 0000. We now have a decade or Modulo-10 up-counter.



Decade Counter Truth Table

Clock Count	Output bit Pattern				Decimal Value
	QD	QC	QB	QA	
1	0	0	0	0	0
2	0	0	0	1	1
3	0	0	1	0	2
4	0	0	1	1	3
5	0	1	0	0	4
6	0	1	0	1	5
7	0	1	1	0	6
8	0	1	1	1	7
9	1	0	0	0	8
10	1	0	0	1	9
11	Counter Resets its Outputs back to Zero				



Procedure:

Conclusion:

EXPERIMENT NO: 9

Aim of the Exp.: To Study shift registers.

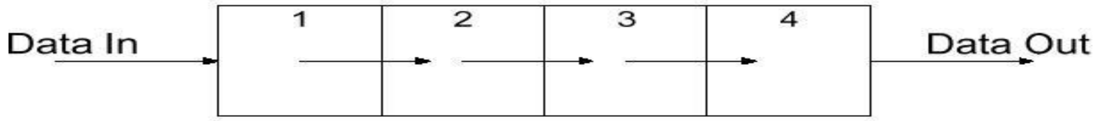
Apparatus Required:

Theory:

Serial In/Shift Right/Serial Out Operation

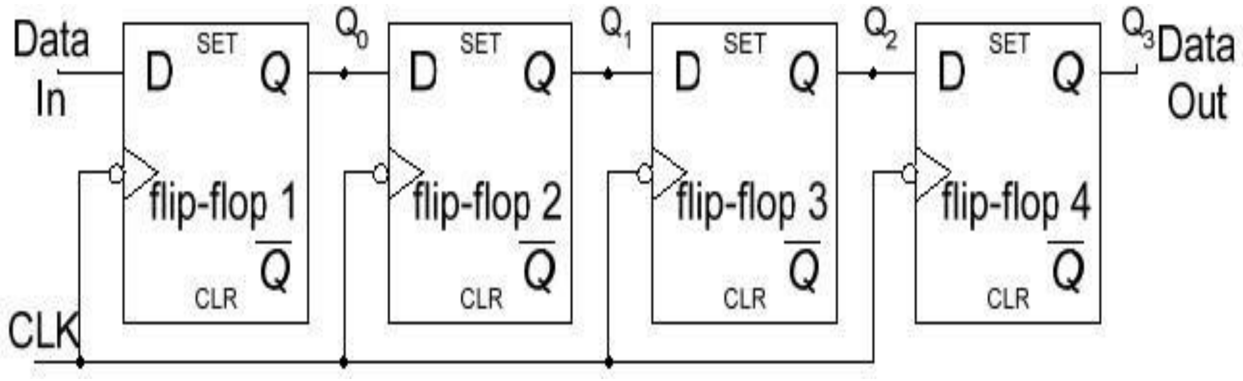
Data is shifted in the right hand direction one bit at a time with each transition of the clock signal. The data enters the shift register serially from the left hand side and after four clock transitions the 4-bit registers has 4-bbits of data.The data is shifted out serially one bit at a time from the right hand side of the register if clock signals are continuously applied. Thus after 8 clock signals the 4-bit data is completely shifted out of the shift register.

Serial In/Serial Right/Serial Out Operation

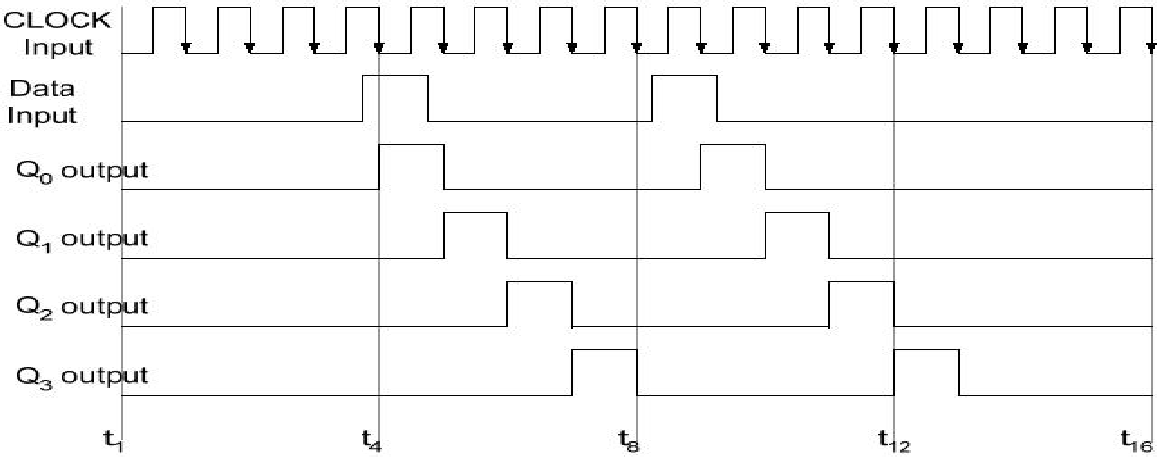


Serial shift registers can be implemented using any type of flip-flops. A serial shift register implemented using D flip-flops with the serial data applied at the D input of the first flip-flop and serial data out obtained at the Q output of the last flip-flop is shown in figure. At each clock transition 1 bit of serial data is shifted in and at the same instant 1-bit of serial data is shifted out. For a 4-bit shift register, 8 clock transitions are required to shift in 4-bit data and completely shift out the 4-bit data. As the data shifted out 1-bit at a time, a logic 0 value is usually shifted in to fillup the vacant bits in the shift register.

Serial In/Shift Right/Serial Out Register



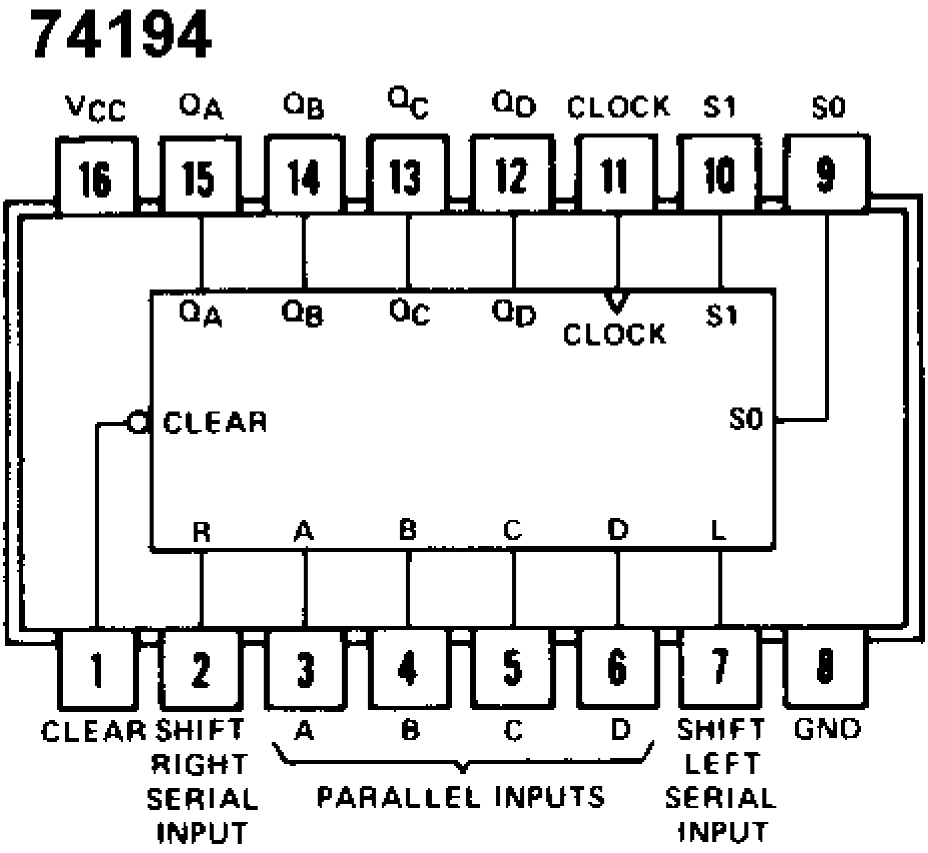
Timing diagram of a Serial In/Shift Right/Serial Out Register



Shift Register Truth Table

Outputs	Q_0	Q_1	Q_2	Q_3
Reset	0	0	0	0
CK Pulse 1	1	0	0	0
CK Pulse 2	0	1	0	0
CK Pulse 3	0	0	1	0
CK Pulse 4	0	0	0	1

Shift Register using IC74194



- Synchronous parallel loading is accomplished by applying the four bits of data and taking both mode control inputs, S0 and S1, high.
- The data are loaded into the associated flip-flops and appear at the outputs after the positive transition of the clock input.
- Shift right is accomplished synchronously with the rising edge of the clock pulse when S0 is high and S1 is low. Serial data for this mode is entered at the shift-right data input.
- When S0 is low and S1 is high, data shifts left synchronously and new data is entered at the shift-left serial input.
- Clocking of the shift register is inhibited when both mode control inputs are low.

Procedure:

(iv) Connections are given as per circuit diagram.
(v) Logical inputs are given as per circuit diagram.
(vi) Observe the output and verify the truth table.

Conclusion:

EXPERIMENT NO: 10(a)

Aim of the Exp.: Write an assembly language program to find 1'S Complement of a number for 8085MP.

Apparatus Required:

Sl. No	Equipment	Specification	Quantity
1	8085 Microprocessor Kit		1
2	Keybpard		1

Program:

Address	Label	Mnemonics	Operand	Hexcode	Remark
2000		MOV	A,67		[A] ← 67
2002		CMA			Complement A reg content
2003		STA	6500		Store the result in 6500 memory
2006		HLT			End of Program

Procedure:

1. Connect the keyboard to Microprocessor kit.

2. Switch on the kit and write the command to start the assembler

3. Enter the instructions of the program one by one.

4. Enter the command to execute the program

5. Check the result.

Result/Output:

Memory location	Content
6500	

Conclusion:

EXPERIMENT NO: 10(b)

Aim of the Exp.: Write an assembly language program to find 2'S Complement of a number for 8085MP.

Apparatus Required:

Theory:

Program:

Address	Label	Mnemonics	Operand	Hexcode	Remark
2000		MOV	A,67		[A] ← 67

2002		CMA			Complement A reg content
2003		INR	A		[A] ← [A]+1
2004		STA	6500		Store the result in 6500 memory
2007		HLT			End of Program

Procedure:

1. Connect the keyboard to Microprocessor kit.

2. Switch on the kit and write the command to start the assembler

3. Enter the instructions of the program one by one.

4. Enter the command to execute the program

5. Check the result.

Result/Output:

Memory location	**Content**
6500	

Conclusion:

EXPERIMENT NO: 11(a)

Aim of the Exp.: Write an assembly language program for Addition of 8-bit number.

Apparatus Required:

Theory:

Procedure:

Conclusion:

EXPERIMENT NO: 11(b)

Aim of the Exp.: Write an assembly language program for Subtraction of 8-bit number resulting 8/16 bit number.

Apparatus Required:

Theory:

Procedure:

Result/Output:

Conclusion:

EXPERIMENT NO: 12(a)

Aim of the Exp.: Write an assembly language program for decimal Addition of 8-bit numbers

Apparatus Required:

Theory:

Procedure:

Result/Output:

Conclusion:

EXPERIMENT NO: 12(b)

Aim of the Exp.: Write an assembly language program for decimal Subtraction of 8-bit numbers

Apparatus Required:

Theory:

Procedure:

Result/Output:

Conclusion:

EXPERIMENT NO: 13(a)

Aim of the Exp.: Write an assembly language program for comparison between two numbers

Apparatus Required:

Theory:

Procedure:

Result/Output:

Conclusion:

EXPERIMENT NO: 13(b)

Aim of the Exp.: Write an assembly language program for finding the largest in an Array

Apparatus Required:

Theory:

Procedure:

Result/Output:

Conclusion:

EXPERIMENT NO: 14

Aim of the Exp.: Write an assembly language program for Block Transfer

Apparatus Required:

Theory:

Procedure:

Result/Output:

Conclusion:

EXPERIMENT NO: 15

Aim of the Exp.: Write an assembly language program for Traffic light control using 8255 .

Apparatus Required:

Theory:

Procedure:

Conclusion:

EXPERIMENT NO: 16

Aim of the Exp.: Write an assembly language program to Generate square wave using 8255 for 8085MP

Apparatus Required:

Theory:

Procedure:

Conclusion: