LABORATARY MANUAL

Of

Digital System Design Lab

For 2nd Year , 1st Semester B.Tech (R18) ECE

19261A0470

Department of Electronics and Communication Engineering

Mahatma Gandhi Institute of Technology

Chaitanya Bharathi (P.O), Gandi Pet, Hyderabad -75

Useful IC Pin details

IC NUMBER	Description of IC
7400	Quad 2 input NAND GATE
7401	Quad 2input NAND Gate (open collector)
7402	Quad 2 input NOR Gate
7403	Quad 2 input NOR Gates (open collector)
7404	Hex Inverts
7421	Dual 4 input AND Gates
7430	8 input NAND Gate
7432	Quad 2 input OR Gates
7486	Quad 2 input EX-OR Gate
74107	Dual j-k Flip Flop
74109	Dual j-k Flip Flop
74174	Hex D Flip Flop
74173	Quad D Flip Flop
7473	Dual j-k Flip Flop
7474	Dual D Flip Flop
7475	Quad Bi-stable latch

EXPERIMENT 1 : Realization Of Boolean Expressions Using Gates

AIM

1. To verify the Truth Tables for different logic gates.

2 To verify the basic laws of Boolean Algebra.

THEORY

The truth tables of different logic gates are given in Table 1 .1

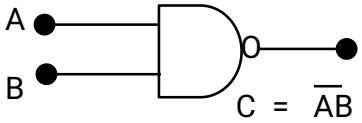
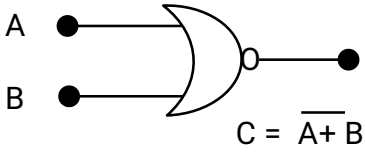
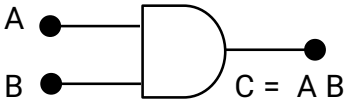
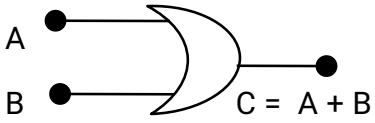
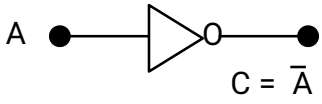
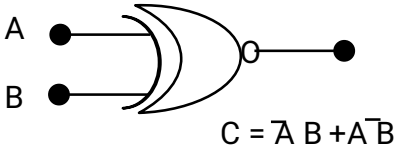
SL No	GATE	SYMBOL	INPUTS A B	OUTPUT Y
1.	NAND		0 0 0 1 1 0 1 1	1 1 1 0
2.	NOR		0 0 0 1 1 0 1 1	1 0 0 0
3.	AND		0 0 0 1 1 0 1 1	0 0 0 1
4.	OR		0 0 0 1 1 0 1 1	0 1 1 1
5.	NOT		0 1	1 0
6.	EX-OR		0 0 0 1 1 0 1 1	0 1 1 0

TABLE 1.1 Truth Tables of Logic Gates

PROCEDURE

1. Wire the logic gates as shown in the Fig 1.1.
2. Feed the Logic Signals (0 or 1) form the Logic Input switches at the inputs A and B.
3. Monitor the outputs using Logic Output LED indicators.
4. Record the inputs and outputs. Verify the Truth Tables for NOT, AND, OR and NOR operations.
5. Wire the Logic gates as shown in Fig-1.2 and repeat the step 2 through 4
6. Wire the Logic gates as shown in Fig - 1.3 and repeat the steps 2 and 3. Verify the Truth table for EX-OR gate Table (1.1)

Boolean Algebra is a mathematical system useful for the design and analysis of switching circuits. Modern digital computers are designed, maintained and their operation is analyzed using techniques and their operation is analyzed using techniques and symbology from Boolean Algebra. A boolean variable takes two values : 0 or 1. The laws of boolean algebra are listed in Table 2.1 These laws can be classified into two categories:

1. Laws of Multiplication
2. Laws of Addition Both are dual to each other.

SL No.	*LAW / THEORM*	*for AND*	*for OR*
1.	Zero Law	$0 . A = 0$	$0 + A = A$
2.	Unit Law	$1 . A = A$	$1 + A = 1$
3.	Idempotent Law	$A . A = A$	$A + A = A$
4.	Complementary Law	$A . \bar{A} = 0$	$A + \bar{A} = 1$
5.	Involution Law	$\bar{\bar{A}} = A$	
6.	Commutative Law	$AB = BA$	$A + B = B + A$
7.	Associative Law	$A(BC) = (AB) C$	$A + (B+C) = (A+B) +C$
8.	Distributive Law	$A(B+C) = AB + AC$	$A + BC = (A + B) (A + C)$
9.	Absorption Law	$A + AB = A$	$A(A+B) = A$
10.	Transposition Theorem	$AC + \bar{A}B = A+B$	$A (A+B) = AB$
11.	Included Term Theorem	$AB + \bar{A}C + BC$ $= AB + \bar{A}C$	$(A+B) (\bar{A}+C) (B+C)$ $= (A+B) (\bar{A}+C)$
12.	Demorgan's law	$\overline{A B} = \bar{A} + \bar{B}$	$\overline{(A+B)} = \bar{A}\,\bar{B}$

Table 2.1 Basic Laws of Boolean Algebra

The Commutative, Associative and Distributive laws may be extended to include any number of terms and variables. Most of the laws of Boolean Algebra are different from those of normal algebra and can be proved by perfect induction method.

CIRCUIT DIAGRAMS

The circuit diagrams using logic gates to verify various laws of boolean algebra (for multiplication) are given in Fig- 2.1 through 2.8

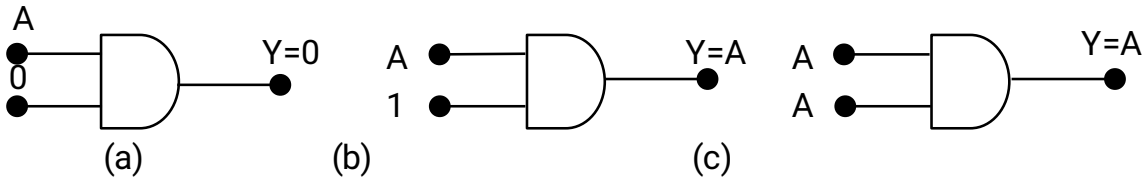


Fig 2.1 Circuits to verify the Zero, Unit and Idempotent Laws where A is 0 or 1.

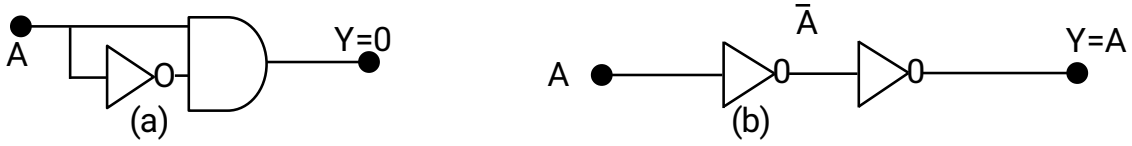


Fig 2.2 Circuits to verify the Complementary and Involution Laws where A is 0 or 1.

Fig 2.3 Circuits to verify the Commutative Law.

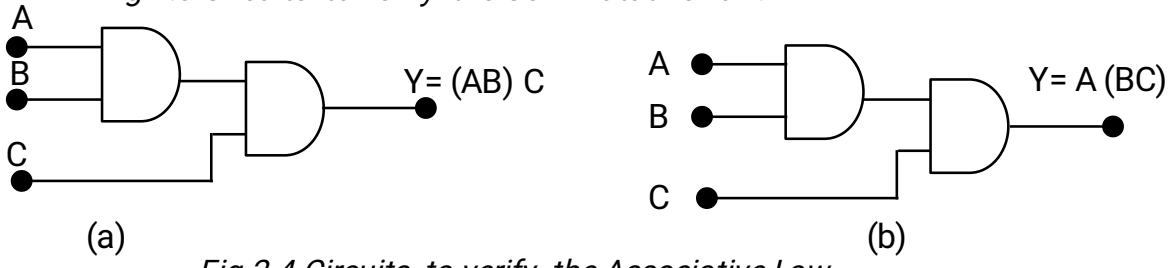


Fig 2.4 Circuits to verify the Associative Law.

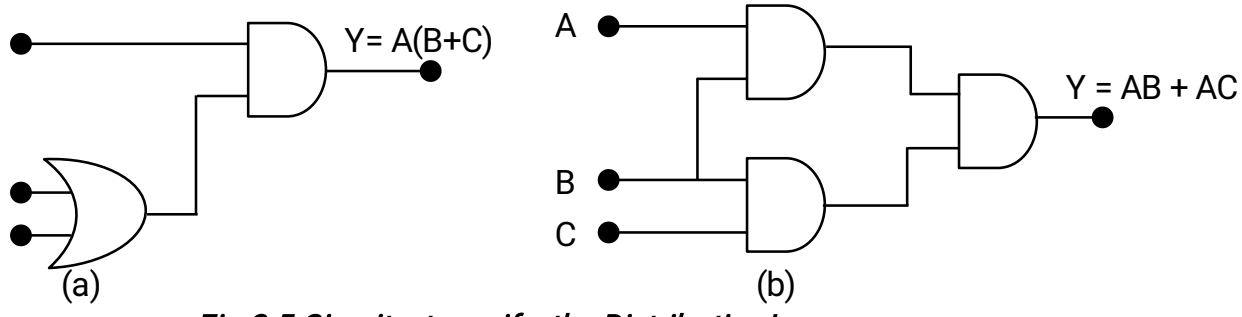


Fig 2.5 Circuits to verify the Distributive Law.

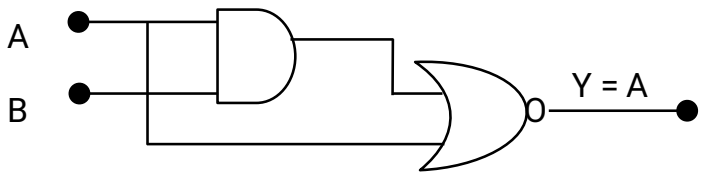


Fig 2.6 Circuits to verify the Absorption Law.

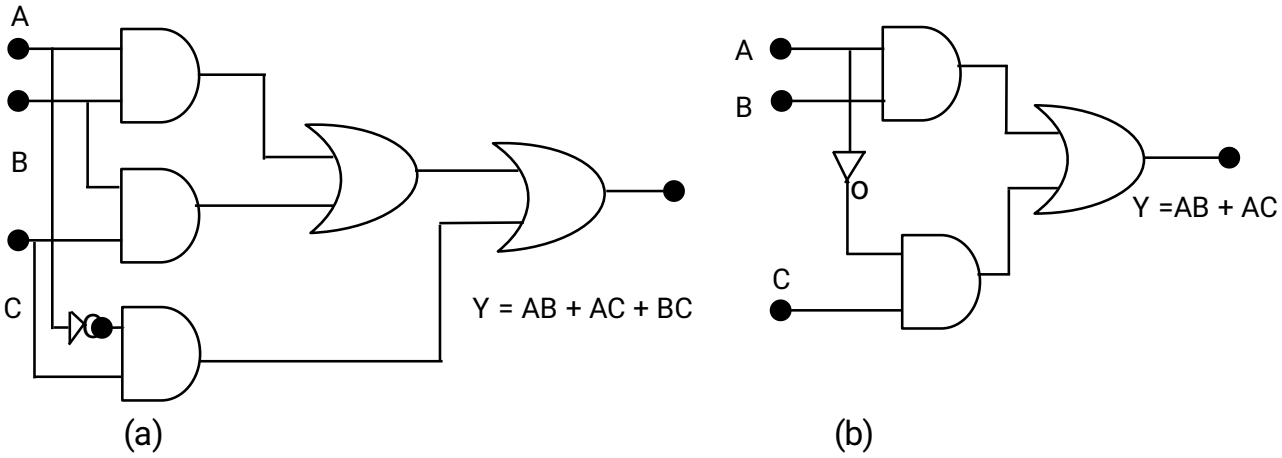


Fig 2.7 Circuits to verify the Included Term Theorm.

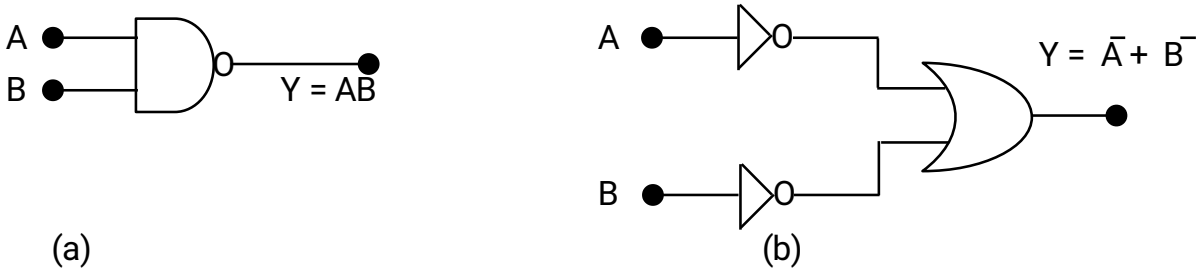


Fig 2.8 Circuits to verify the Demorgan's Law.

PROCEDURE

1. Wire the circuits as shown in Fig- 2.1 through 2.8.
2. Feed the Logic Signals (0 or 1) from the Logic Input switches as per the diagrams.
3. Observe and record the logic outputs using logic output LED indicators,
4. Verify the Boolean laws from the recorded data .

AIM

1. To realize the " Basic Half Subtractor" and verify its truth table.
2. To realize the "Full Subtractor" and verify its truth table

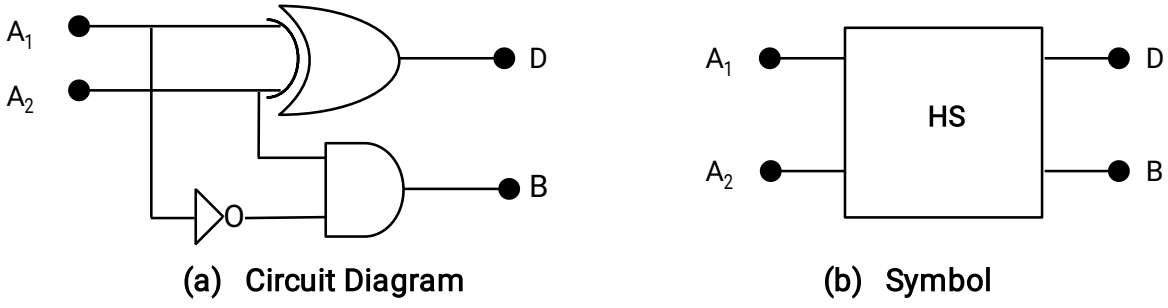
THEORY

1. **Half Subtractor :** It is a binary arithmetic circuit which can subtract one binary digit from another binary digit. The rules of binary subtraction are given table 4.1.

INPUTS A_1	A_2	DIFFERENCE D	BORROW B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Table 4.1 Binary Subtraction rules

The Half Subtractor is shown in Fig-4.1



(a) Circuit Diagram (b) Symbol

Fig 4.1. HALF SUBTRACTOR.

2. **Full Subtractor :** When a multi-bit binary number is to be subtracted from another multi-bit binary number, the Half subtractor is not adequate, since half subtractor has no input to propagate the borrow bit after each single bit subtraction. The circuit which does this is called Full Subtractor. The Full Subtractor can be constructed from two half subtractors as shown in Fig- 4.2.

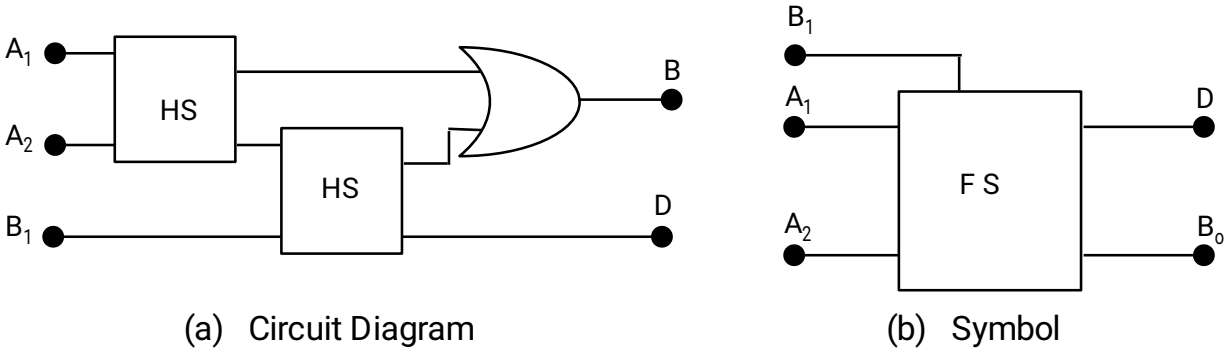


Fig 4.2. FULL SUBTRACTOR.

PROCEDURE

1. Wire the circuit as shown in Fig - 4.1a.
2. Feed the Logic Input switches to the circuit inputs A_1 and A_2.
3. Observe the outputs D and B using the output logic indicators.
4. Record the inputs and outputs. Verify the truth table of half subtractor.
5. Repeat steps 1 through 4 for full subtractor.

AIM

1. To construct the half adder and full adder circuits and to verify their truth tables, &
2. To construct a two-bit parallel adder & to verify its Truth Table .

INTRODUCTION

The EX-OR gate can be used for binary addition as the sum is equal to AB + AB and the carry term is obtained using AND gate. The realisation of half adder circuit is as shown in fig-4(a). This circuit is called a half-adder. It has two inputs for the two bits to be added and two outputs, one for the sum and the other for the carry-out. But, there is no provision to accept the carry from the less significant bits.

INPUT		OUTPUT	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

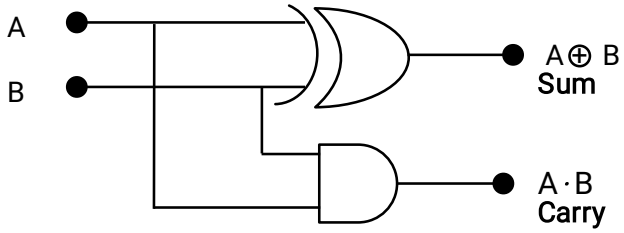


Fig - 4a.. HALF ADDER CIRCUIT.

A Full-Adder accepts the carry signal from the less significant bits. It consists of two half adders And an Or gate. The realisation of a full-adder is as shown in fig 4(b).

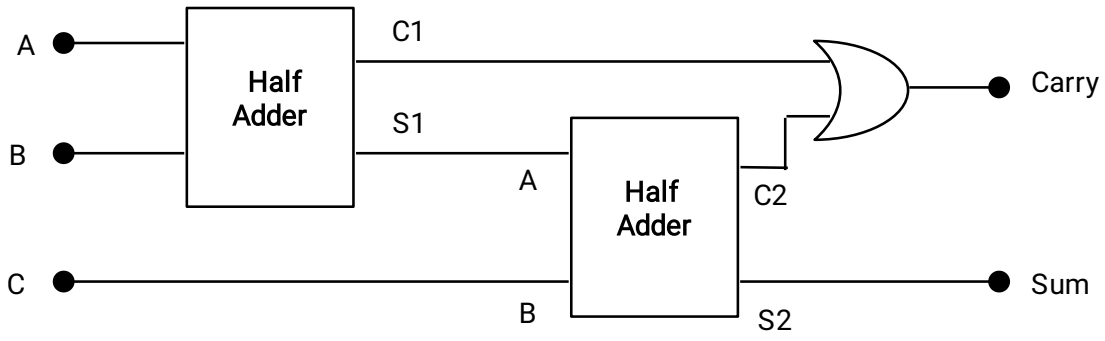


Fig-4 (b) FULL - ADDER CIRCUIT

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

TRUTH TABLE FOR A FULL- ADDER

Experiment 2- Design and realization logic gates using universal gates

Aim: Realization of logic functions with the help of universal gates-NAND Gate, NOR Gate

Apparatus: logic trainer kit, NAND gates (IC 7400), wires. NOR gates (IC 7402)

Theory:

NAND gate is actually a combination of two logic gates: AND gate followed by NOT gate. So its output is complement of the output of an AND gate.

This gate can have minimum two inputs, output is always one. By using only NAND gates, we can realize all logic functions: AND, OR, NOT, X-OR, X-NOR, NOR. So this gate is also called universal gate.

NAND gates as NOT gate

A NOT produces complement of the input. It can have only one input, tie the inputs of a NAND gate together. Now it will work as a NOT gate. Its output is

$Y = (A.A)'$

=> $Y = (A)'$

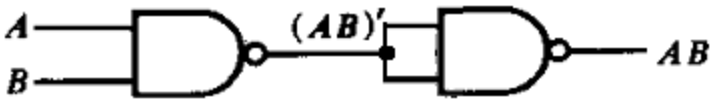
NOT (inverter)

NAND gates as AND gate

A NAND produces complement of AND gate. So, if the output of a NAND gate is inverted, overall output will be that of an AND gate.

$Y = ((A.B)')'$

=> $Y = (A.B)$



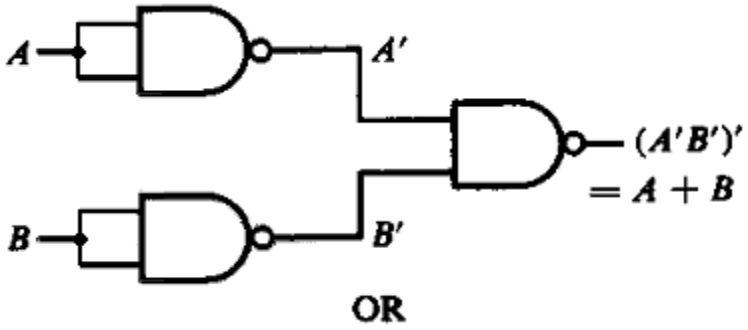
AND

NAND gates as OR gate

From DeMorgan's theorems: $(A.B)' = A' + B'$

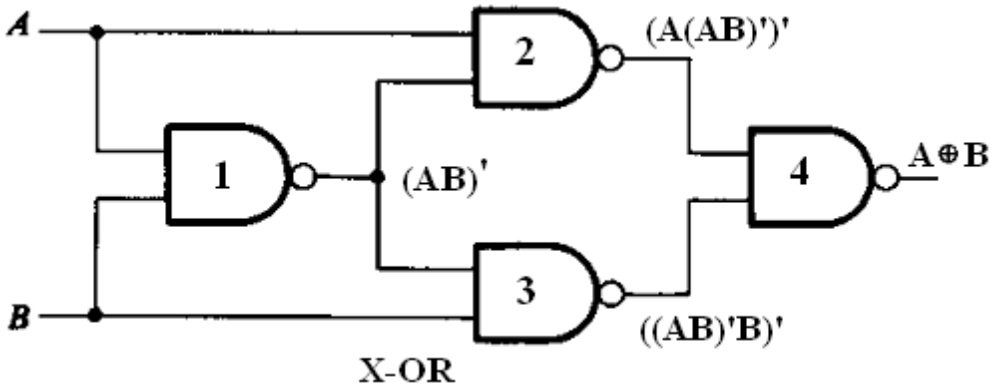
=> $(A'.B')' = A'' + B'' = A + B$

So, give the inverted inputs to a NAND gate, obtain OR operation at output.



NAND gates as X-OR gate

The output of a to input X-OR gate is shown by: Y = A'B + AB'. This can be achieved with the logic diagram shown in the left side.



Gate No.	Inputs	Output
1	A, B	(AB)'
2	A, (AB)'	(A (AB)')'
3	(AB)', B	(B (AB)')'
4	(A (AB)')', (B (AB)')'	A'B + AB'

Now the ouput from gate no. 4 is the overall output of the configuration.

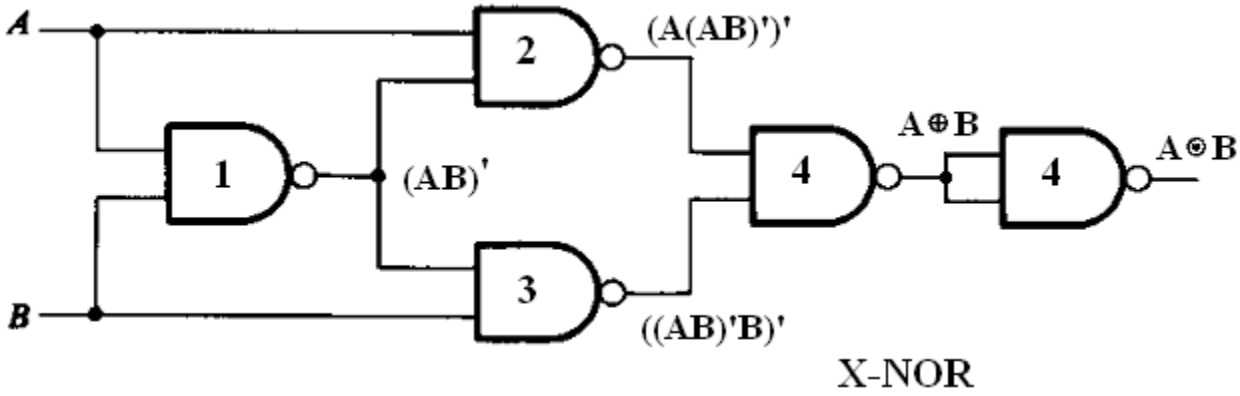
Y = ((A (AB)')' (B (AB)')')'
= (A(AB)')'' + (B(AB)')''
= (A(AB)') + (B(AB)')
= (A(A' + B)') + (B(A' + B'))
= (AA' + AB') + (BA' + BB')
= (0 + AB' + BA' + 0)
= AB' + BA'

=> Y = AB' + A'B

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 ouput is that of an X-NOR gate.

Y = AB+ A'B'

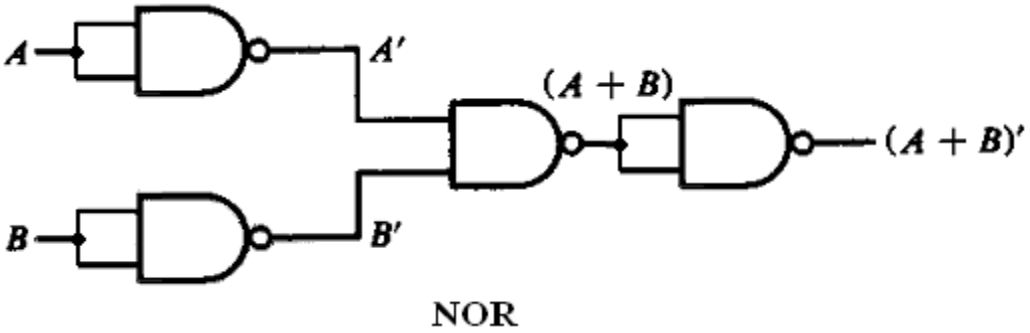


X-NOR

NAND gates as NOR gate

A NOR gate is an OR gate followed by NOT gate. So connect the output of OR gate to a NOT gate, overall output is that of a NOR gate.

Y = (A + B)'



NOR

NOR gate is actually a combination of two logic gates: OR gate followed by NOT gate. So its output is complement of the output of an OR gate.

This gate can have minimum two inputs, output is always one. By using only NOR gates, we can realize all logic functions: AND, OR, NOT, X-OR, X-NOR, NAND. So this gate is also called 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 it will work as a NOT gate. Its output is

Y = (A+A)'

=> Y = (A)'

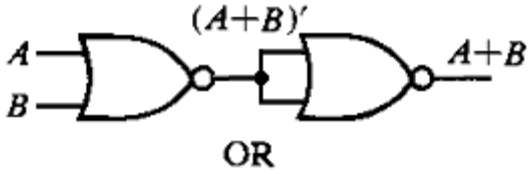
NOT

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 of an OR gate.

Y = ((A+B)')'

=> Y = (A+B)

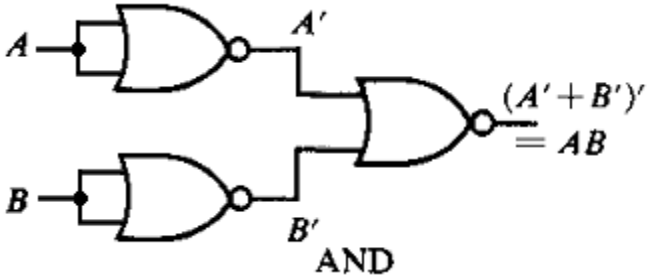


NOR gates as AND gate

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

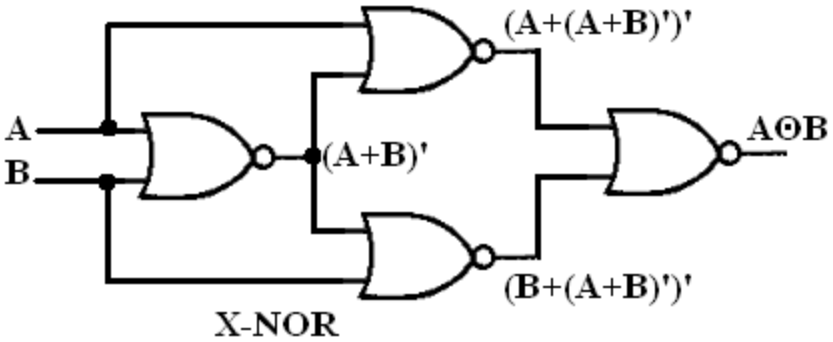
=> (A'+B')' = A''B'' = AB

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



NOR gates as X-NOR gate

The output of a two input X-NOR gate is shown by: Y = AB + A'B'. This can be achieved with the logic diagram shown in the left side.



Gate No.	Inputs	Output
1	A, B	(A + B)'
2	A, (A + B)'	(A + (A+B)')'
3	(A + B)', B	(B + (A+B)')'
4	(A + (A + B)')', (B + (A+B)')'	AB + A'B'

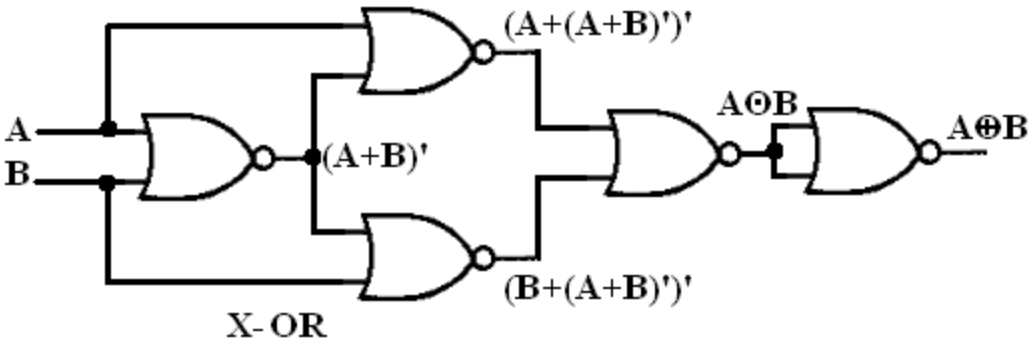
Now the ouput from gate no. 4is the overall output of the configuration.

Y = ((A + (A+B)')' (B +(A+B)')')'
= (A+(A+B)')".(B+(A+B)')"
= (A+(A+B)').(B+(A+B)')
= (A+A'B').(B+A'B')
= (A + A').(A + B').(B+A')(B+B')
= 1.(A+B').(B+A').1
= (A+B').(B+A')
= A.(B + A') +B'.(B+A')
= AB + AA' +B'B+B'A'
=> Y = AB + 0 + 0 + B'A'
= AB + B'A'
= AB + A'B'

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 ouput is that of an X-OR gate.

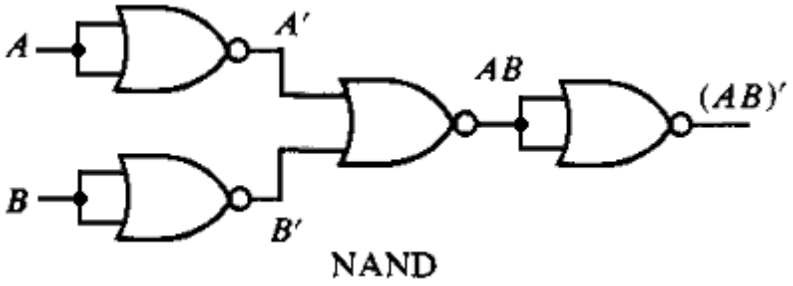
Y = A'B+ AB'



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, overall output is that of a NAND gate.

Y = (AB)'



Procedure:

1. Connect the trainer kit to ac power supply.
2. Connect the NAND gates / 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.

Result:

Experiment 3 - Generation of clock using NAND / NOR gates

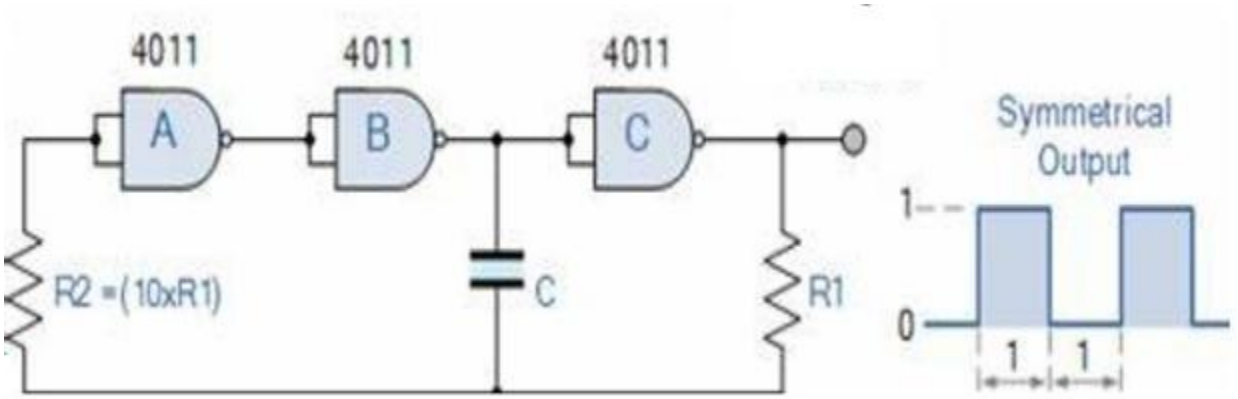
Objective:

Design a 450KHz clock using NAND Gate.

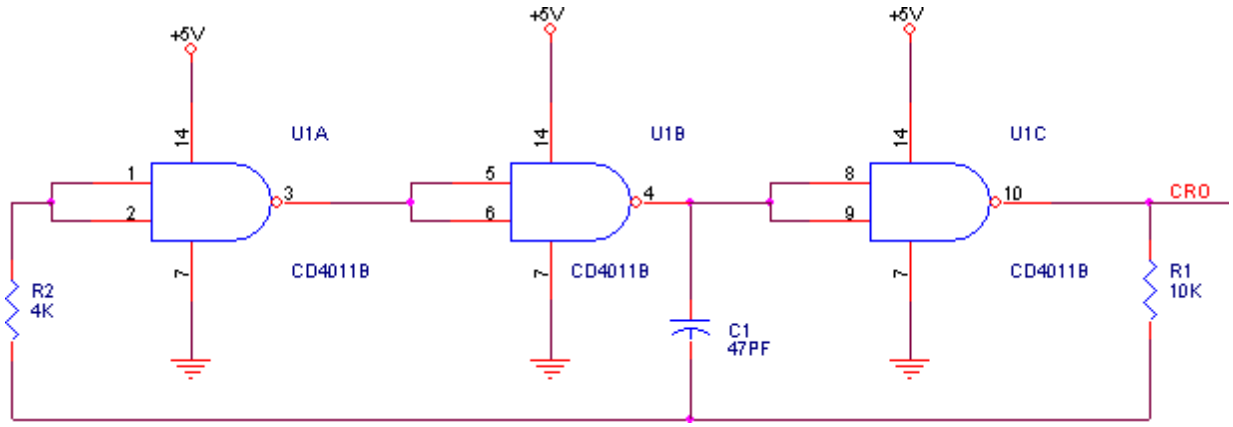
Equipments Needed:

- Patch Cords
- Nvis 6550A & IC CD 4011, 4k Resistance , 10K Resistance , 47pF/25V Electrolytic capacitor

Block Diagram:



Circuit Diagram:



Procedure:

- Make the connection as per the above connection diagram.
- Switch On the power supply.
- Observed the output on the CRO.
- BY changing the value of R1, R2 and C1 to change the Clock Output.

Data Sheet:

TEXAS INSTRUMENTS

Data sheet acquired from Harris Semiconductor
SCHS021D – Revised September 2003

CD4011B, CD4012B, CD4023B Types

CMOS NAND GATES

High-Voltage Types (20-Volt Rating)

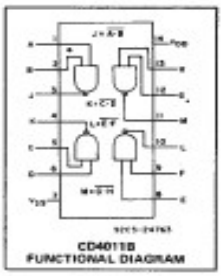
Quad 2 Input – CD4011B
Dual 4 Input – CD4012B
Triple 3 Input – CD4023B

■ CD4011B, CD4012B, and CD4023B NAND gates provide the system designer with direct implementation of the NAND function and supplement the existing family of CMOS gates. All inputs and outputs are buffered.

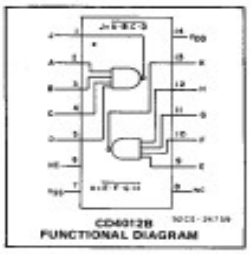
The CD4011B, CD4012B, and CD4023B types are supplied in 14-lead hermetic dual-in-line ceramic packages (F3A suffix), 14-lead dual-in-line plastic packages (E suffix), 14-lead small-outline packages (M, MT, M96, and NSR suffixes), and 14-lead thin shrink small-outline packages (PWR suffix). The CD4011B and CD4023B types also are supplied in 14-lead thin shrink small-outline packages (PW suffix).

Features:

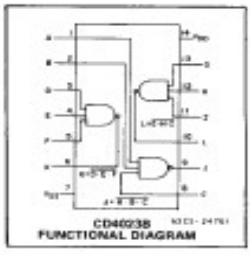
- Propagation delay time = 60 ns (typ.) at C_L = 50 pF, V_{DD} = 10 V
- Buffered inputs and outputs
- Standardized symmetrical output characteristics
- Maximum input current of 1 μA at 18 V over full package temperature range; 100 nA at 18 V and 25°C
- 100% tested for quiescent current at 20 V
- 5-V, 10-V, and 15-V parametric ratings
- Noise margin (over full package temperature range):
  - 1 V at V_{DD} = 5 V
  - 2 V at V_{DD} = 10 V
  - 2.5 V at V_{DD} = 15 V
- Meets all requirements of JEDEC Tentative Standard No. 13B, "Standard Specifications for Description of "B" Series CMOS Devices"



CD4011B
FUNCTIONAL DIAGRAM



CD4012B
FUNCTIONAL DIAGRAM



CD4023B
FUNCTIONAL DIAGRAM

MAXIMUM RATINGS, *Absolute-Maximum Values:*

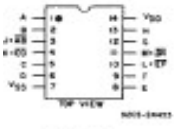
DC SUPPLY-VOLTAGE RANGE, (V_{DD})
(Voltages referenced to V_{SS} Terminal) ..... −0.5V to +20V
INPUT VOLTAGE RANGE, ALL INPUTS ..... −0.5V to V_{DD} +0.5V
DC INPUT CURRENT, ANY ONE INPUT ..... ±10mA
POWER DISSIPATION PER PACKAGE (P_D):
For T_A = −55°C to +100°C 500mW
For T_A = +100°C to +125°C ..... Derate Linearity at 12mW/°C to 200mW
DEVICE DISSIPATION PER OUTPUT TRANSISTOR
FOR T_A = FULL PACKAGE-TEMPERATURE RANGE (All Package Types) 100mW
OPERATING-TEMPERATURE RANGE (T_A) ..... −55°C to +125°C
STORAGE TEMPERATURE RANGE (T_{stg}) ..... −65°C to +150°C
LEAD TEMPERATURE (DURING SOLDERING):
At distance 1/16 ± 1/32 inch (1.59 ± 0.79mm) from case for 10s max ..... +265°C

RECOMMENDED OPERATING CONDITIONS

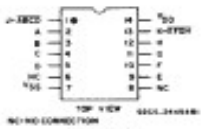
For maximum reliability, nominal operating conditions should be selected so that operation is always within the following ranges:

CHARACTERISTIC	LIMITS		UNITS
	MIN.	MAX.	
Supply-Voltage Range (For T_A = Full Package Temperature Range)	3	18	V

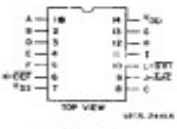
TERMINAL ASSIGNMENTS



CD4011B



CD4012B



CD4023B

Experiment 4- Design a 4 – bit Adder / Subtractor

AIM

To study 4-bit BINARY addition and subtraction using with IC7483.

INTRODUCTION

The Arithmetic Logic Unit (ALU) is the number crunching part of a computer. This means not only arithmetic operations but logic as well (OR, AND, NOT and so forth).

BINARY ADDITION

ALU don't process decimal numbers, they process binary numbers. Before you can understand the circuit inside an ALU, you must learn how add binary numbers. There are five basic cases that must be understood before going on.

CASE 1:

When no pebbles are added to no pebbles, the total is no pebbles. As a word equation

None + None = None

With binary numbers 0 + 0 = 0

CASE 2:

If no pebbles are added to one pebble, the total is one pebble,

None + ● = ●

In terms of binary numbers 0 + 1 = 1

CASE 3:

Addition is commutative. This means you can transpose the number of the preceding case to get

● + None = ●

In terms of binary numbers 1 + 0 = 1

CASE 4:

Next, one pebble added to one pebble to get gives two pebbles

● + ● = ● ●

In terms of binary numbers 1 + 1 = 1 0

To avoid confusion with decimal numbers, read this as an "one plus one equals one-zero". An alternative way of reading the equation is "one plus one equals zero, carry one.

CASE 5:

one pebble plus one pebble plus one gives a total of three pebbles

● + ● + ● = ● ● ●

In terms of binary numbers 1 + 1 + 1 = 1 1

Reads this as “one plus one plus one equals one-one”.
Alternately, “one plus one plus one equals one.”

BINARY SUBTRACTION

To subtract binary numbers, we need to discuss for cases

Case 1 0 - 0 = 0

Case 2 1 - 0 = 1

Case 3 1 - 1 = 0

Case 4 10 - 0 = 1

The last result represents ● ● - ● = ●
Which makes sense.

To subtract larger binary number, subtract column by column, borrowing from the next higher column when necessary. For instance, in subtracting 1 0 1 from 1 1 1, proceed like this

7	1 1 1
-5	-1 0 1
----	--------
2	0 1 0

Starting on the right, 1-1 gives 0, then 1 – 0 is finally, 1 – 1 is 0.
Here is another example: subtract 1 0 1 0 from 1 1 0 1.

13	1 1 0 1
-10	- 1 0 1 0
----	----------
3	0 0 1 1

In last significant column, 1 – 0 is 1. in the second column, we have to barrow from the next higher column then, 1 0 – 1 is 1. in the third column, 0(after borrow) – 0 is 0, in the fourth column, 1 – 1 = 0.

Direct subtraction like the foregoing has been used in computers, however, it is possible to subtract in a different way.

CIRCUIT DESCRIPTION

fig-1 shows the circuit of 4-bit full adder/subtract or, that means it can add or subtract nibbles (4-bits). A_3 A_2 A_1 A_0 is one input and B_3 B_2 B_1 B_0 is another input A_3 A_2 A_1 A_0 given are directly given to the full adder, but B_3 B_2 B_1 B_0 are not given directly through a controlled inverter constructed with Ex-OR gates. Second inputs of all Ex-OR gates in the controlled inverter as combined and given to one SPDT switch. If the switch changes between '0' and '1'. Same switch is connected to pin 13 of 7483 which commands the addition or subtraction operation. When this switch is in '0' position B_3 B_2 B_1 B_0 inputs are directly going to the full adder and the output of the full adder is addition of both A and B inputs. When the switch is in '1' position complements of B_3 B_2 B_1 B_0 are going to the full adder and the full adder output is subtraction of A and B inputs.

HARDWARE SPECIFICATIONS

1. 4-bit binary full adder and subtract or circuit arrangement.
2. IC 7483----------1 No.
3. IC 7486----------1 No.
4. Built in fixed DC source +5V@350mA.
5. Patch chords (1 set) and user Manual.

EQUIPMENT REQUIRESD

1. 4-bit binary full adder and subtractor trainer
2. Patch chords (1 set) and user Manual.

EXPERIMENTAL PROCEDURE

1. Switch ON the experimental board by connecting the power chard to the AC mains.
2. Set C_{IN} to low for binary addition where

$$S_3\ S_2\ S_1\ S_0 = A_3\ A_2\ A_1\ A_0 + B_3\ B_2\ B_1\ B_0 \text{--------- Eq1}$$

3. 2. Set C_{IN} to High for binary subtraction where

$$S_3\ S_2\ S_1\ S_0 = A_3\ A_2\ A_1\ A_0 - B_3\ B_2\ B_1\ B_0 \text{--------- Eq2}$$

4. Verify the table-1 and table-2.

RESULT:

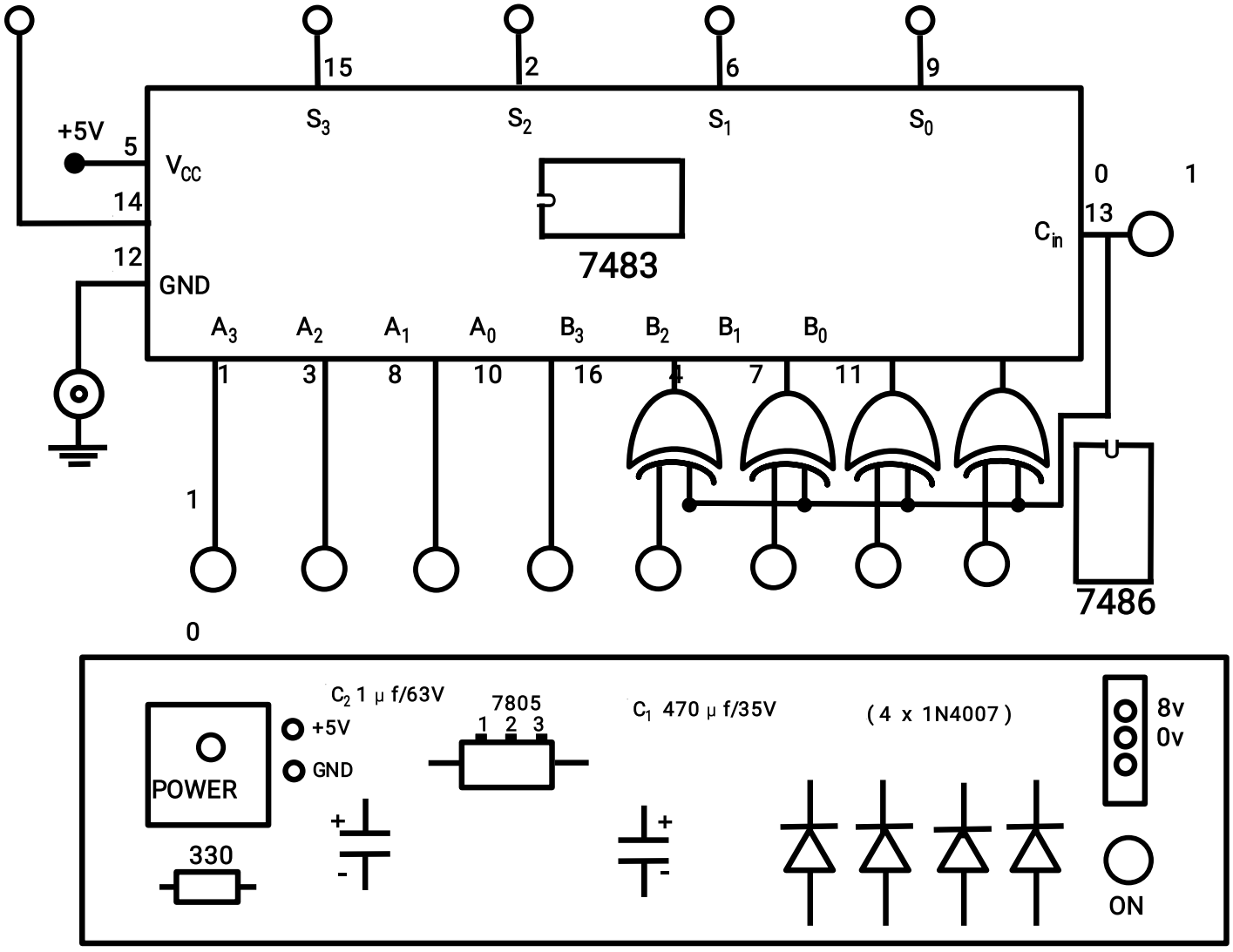
TABLE-1 FOR ADDITION

C_{IN}	A_3	A_2	A_1	A_0	B_3	B_2	B_1	B_0	C_0	S_3	S_2	S_1	S_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	1	0
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
0	1	1	1	1	1	1	1	1	1	1	1	1	0

TABLE-1 FOR SUBTRACTION

C_{IN}	A_3	A_2	A_1	A_0	B_3	B_2	B_1	B_0	C_0	S_3	S_2	S_1	S_0
1	0	0	0	0	0	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	0	1	0	1	1	1	1
1	0	0	0	1	0	0	0	0	1	0	0	0	11
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
1	1	1	1	1	1	1	1	1	1	0	0	0	0

4- BIT BINARY FULL ADDER / SUBTRACTOR



Panel diagram

Experimetnt 5 -- Design and realization a 4 – bit gray to Binary and Binary to Gray Converter

AIM

To study Binary code to Gray code conversion & Gray code to Binary code conversion.

INTRODUCTION

Code conversion from one form to another is often used in digital communication or control areas primarily to enhance the error detection capability and also some times to conceal the original massage during encryption.

We are normally conversant with decimal numbers in day to day life. But the computers recognise and operate only on Binary numbers. If we have n bits of Binary word we can represent 2^n states or decimal numbers.

For a 3 bit code we can represent 8 states, for a 4 Bit code we can represent upto 16 states. Therefore for 10 states (<16 states and > 8 states) also we require 4 bits of binary code.

A decimal zero 0_{10} is represented by 0000_2 binary number. The subsequent increment in count will add a logical 1 at the least significant place.

DECIMAL	BINARY
0	0000
1 (0+1)	0000 + 1 -------- 0001 -------
2 (1+1)	0001 + 1 ------- 0010 ------
3 (2+1)	0010 + 1 -------- 0011 -------- AND So on

The Binary equivalence of Decimal numbers is shown in Table -1.

DECIMAL No.	Binary Code B_3 B_2 B_1 B_0
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

Table -1

BINARY TO GRAY CODE CONVERSION

Gray code is also an ensemble of Binary digits with a different weightage. If G_n is an n^{th} bit in the Gray code word it is related to the bits of Binary code in the following manner.

$$G_n = B_n \oplus B(n+1)$$

For eg. $5_{10} = 0\,1\,0\,1_2$ - Binary code

$B_0 = 1$

$B_1 = 0$

$B_2 = 1$

$B_3 = 0$ then we compute the Gray code

$G_3 = B_3 = 0$

$G_2 = B_3 \oplus B_2$

$= 0 \oplus 1 = 1$

$G_1 = B_2 \oplus B_1$

$= 1 \oplus 0 = 1$

$G_0 = B_1 \oplus B_0$

$= 0 \oplus 1 = 1$

$B_3 \quad B_2 \quad B_1 \quad B_0$

$0 \rightarrow 1 \rightarrow 0 \rightarrow 1$

$\oplus \quad \oplus \quad \oplus$

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

$0 \quad 1 \quad 1 \quad 1$

$G_3 \quad G_2 \quad G_1 \quad G_0$

$\therefore$ The Gray code equivalent to $5_{10} = 0\,1\,1\,1$

The Table 2 gives thus converted Gray codes for the digits 0 through 9_{10}.

Decimal No	Gray Code G_3 G_2 G_1 G_0
0	0 0 0 0
1	0 0 0 1
2	0 0 1 1
3	0 0 1 0
4	1 0
5	0 1 1 1
6	0 1 0 1
7	0 1 0 0
8	1 1 0 0
9	1 1 0 1

Table - 2

Thus we need 3 EX- OR gates to convert Binary code to Gray code.

EXPRIMENTAL PROCEDURE

To verify the Conversion from Binary to Gray Code.

1. Connect the EX-OR gates as shown in Fig -1
2. The input switches will simulate the input Binary code.
3. Connect the outputs to the four logic state indicators.
4. Set the code on the switches as per the Binary code in Table -3 and verify the output code as per the corresponding code in Table -3.

DECIMAL	BINARY CODE B_3 B_2 B_1 B_0	Gray CODE G_3 G_2 G_1 G_0
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

Table - 3

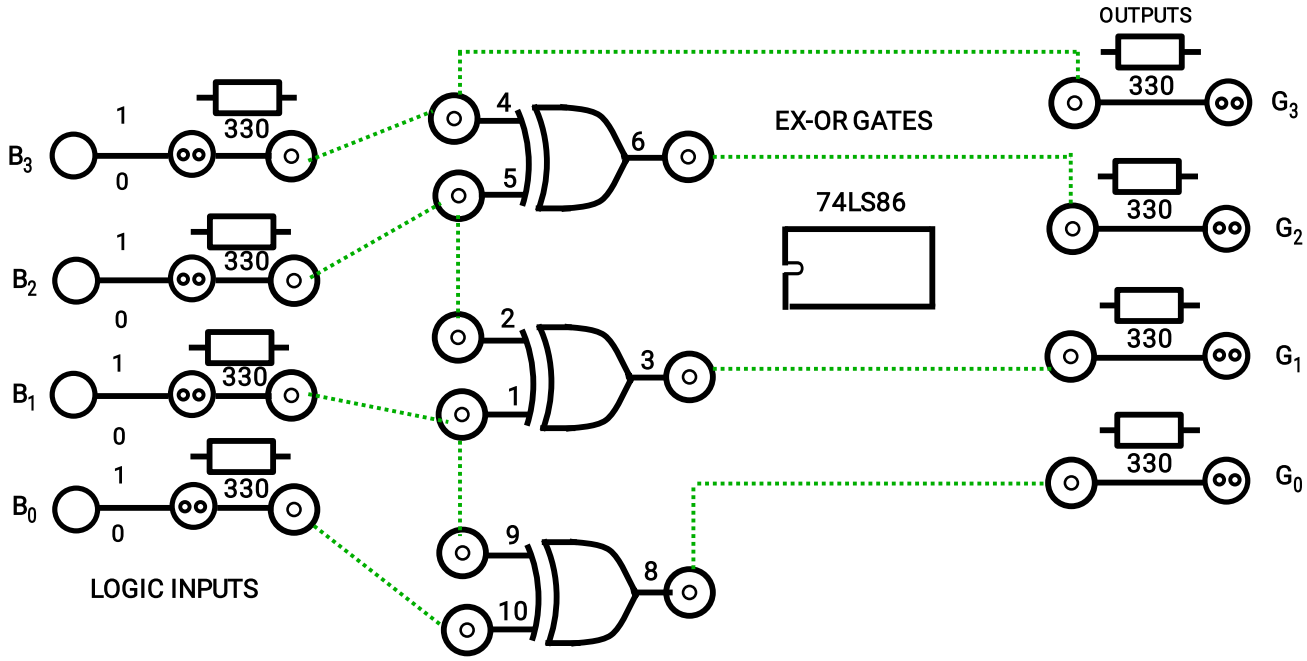
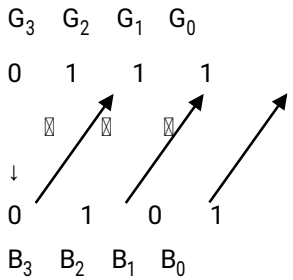


Fig -1 Binary to Gray Code Conversion

GRAY CODE TO BINARY CODE CONVERSION

If G_n is a Bit in the Gray code and B_n is a bit in the Binary code the code conversion follows as per the following formula.

$B_n = G_n \oplus B_{n+1}$



for eg. 5_{10} = 0 1 1 1 in Gray code

i.e. $G_0 = 1, G_1 = 1, G_2 = 1, G_3 = 0$

Then the equivalent Binary code is computed as

$B_3 = G_3 = 0$

$B_2 = G_2 \oplus B_3 = 1 \oplus 0 = 1$

$B_1 = G_1 \oplus B_2 = 1 \oplus 1 = 0$

$B_0 = G_0 \oplus B_1 = 1 \oplus 0 = 1$

$B_3\ B_2\ B_1\ B_0$ = 0 1 0 1

The equivalent Binary codes for all the numbers 0 to 9 is shown in the table 1.
Thus we need three EX-OR gates for code conversion.

EXPERIMENTAL PROCEDURE :

To Study The Code Conversion From Gray Code To Binary Code.

1. Connect three EX-OR gates as shown in fig-2.
2. Connect the outputs to the four logic state indicators.
3. Set the logic input switches as per the code in Table-2 (Gray code).
4. Observe the output on the indicators and verify as per table-4 to its equivalent value.
5. Check all the codes one by one in the table-4

RESULT:

Table 4

DECIMAL	Gray CODE G_3 G_2 G_1 G_0	BINARY CODE B_3 B_2 B_1 B_0
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 1	0 0 1 0
3	0 0 1 0	0 0 1 1
4	0 1 1 0	0 1 0 0
5	0 1 1 1	0 1 0 1
6	0 1 0 1	0 1 1 0
7	0 1 1 1	0 1 0 0
8	1 1 0 0	1 0 0 0
9	1 1 0 1	1 0 0 1
10	1 1 1 1	1 0 1 0
11	1 1 1 0	1 0 1 1
12	1 0 1 0	1 1 0 0
13	1 0 1 1	1 1 0 1
14	1 0 0 1	1 1 1 0
15	1 0 0 0	1 1 1 1

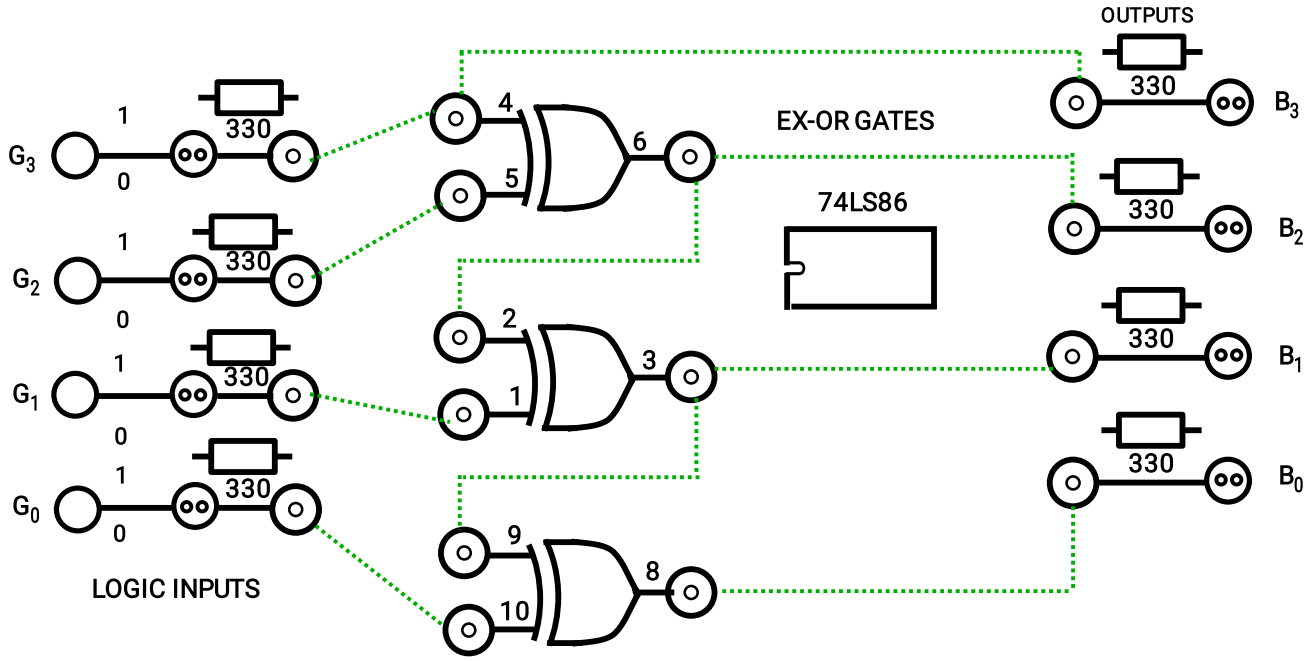
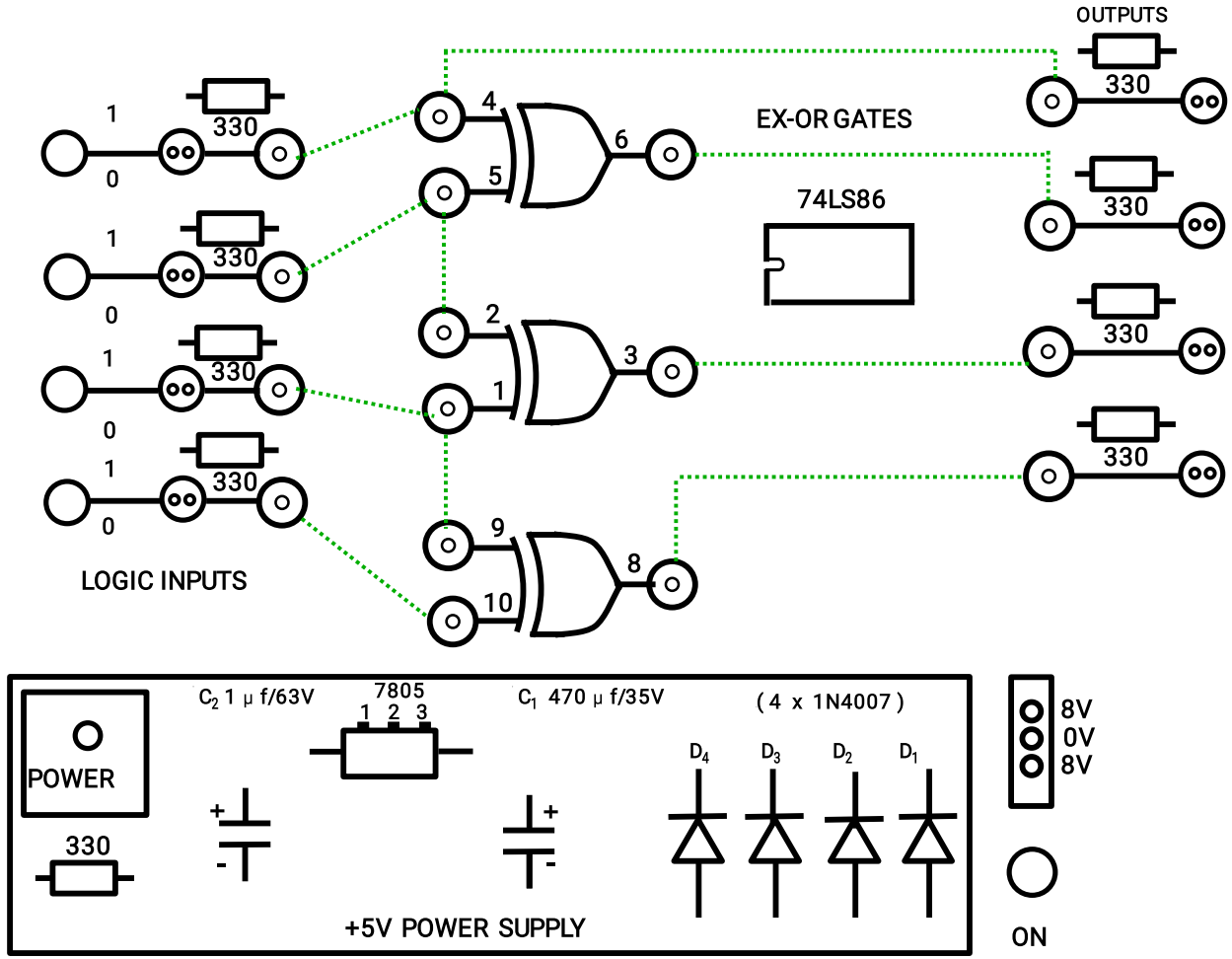


Fig 2 Gray Code to Binary Code Conversion

q4 BIT BINARY TO GRAY & GRAY TO BINARY CONVERTER



Panel Diagram

Experiment 6 - Design and realization of an 8 bit parallel load and serial out shift register using flip-flops.

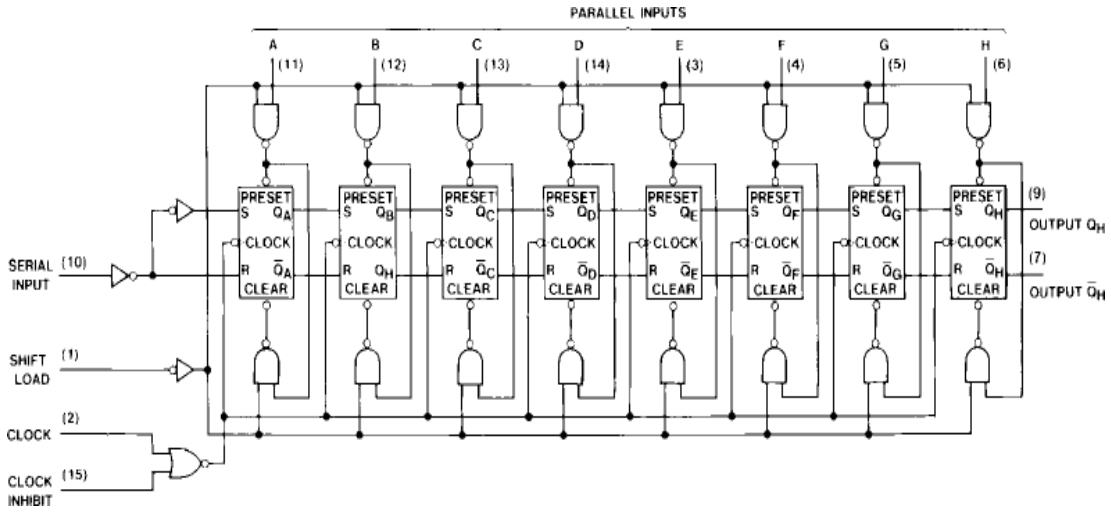
Objective:

Design an 8 bit parallel in and serial Out shift register using two 4 –bit shift register.

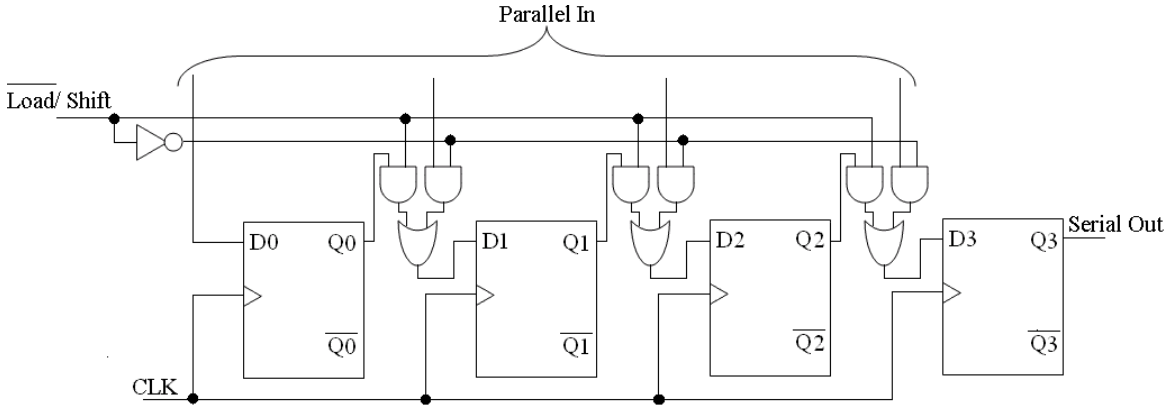
Equipments Needed:

- Patch Cords
- Nvis 6550A & IC 74LS165

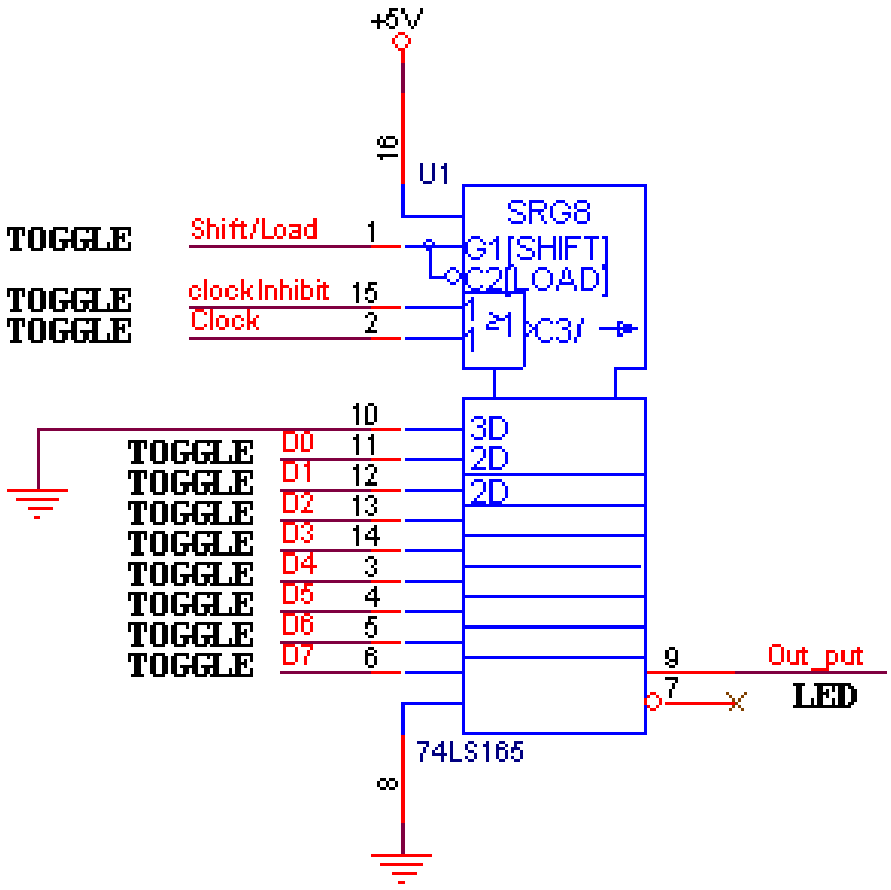
Block Diagram:



Parallel in Serial out Shift resister


Parallel In
Load/ Shift
D0
Q0
Q0
D1
Q1
Q1
D2
Q2
Q2
D3
Q3
Q3
Serial Out
CLK

Circuit Diagram:



Procedure:

- Make the connection as per the above connection diagram.
- Switch On the power supply.
- Set the data by the toggle switch.
- Set the Clock by the toggle switch.
- Set the Preset by the toggle switch.
- Set the Clear by the toggle switch.
- Observed the output on the LEDs
- Make the selection inputs as per the given below table and get the Outputs as per

below table

Clock Pulse No.	Shift/Load	CLK INH	CLK	D0	D1	D2	D3	D4	D5	D6	D7	Dout
1	0	0	X	1	0	1	0	1	0	1	0	0
2	1	0	0-1	1	0	1	0	1	0	1	0	1
3	1	0	0-1	1	0	1	0	1	0	1	0	0
4	1	0	0-1	1	0	1	0	1	0	1	0	1
5	1	0	0-1	1	0	1	0	1	0	1	0	0
6	1	0	0-1	1	0	1	0	1	0	1	0	1
7	1	0	0-1	1	0	1	0	1	0	1	0	0
8	1	0	0-1	1	0	1	0	1	0	1	0	1

Data Sheet:

FAIRCHILD SEMICONDUCTOR™

August 1986
Revised March 2000

DM74LS165
8-Bit Parallel In/Serial Output Shift Registers

General Description

This device is an 8-bit serial shift register which shifts data in the direction of Q_A toward Q_H when clocked. Parallel-in access is made available by eight individual direct data inputs, which are enabled by a low level at the shift/load input. These registers also feature gated clock inputs and complementary outputs from the eighth bit.

Clocking is accomplished through a 2-input NOR gate, permitting one input to be used as a clock-inhibit function. Holding either of the clock inputs HIGH inhibits clocking, and holding either clock input LOW with the load input HIGH enables the other clock input. The clock-inhibit input should be changed to the high level only while the clock input is HIGH. Parallel loading is inhibited as long as the load input is HIGH. Data at the parallel inputs are loaded directly into the register on a HIGH-to-LOW transition of the shift/load input, regardless of the logic levels on the clock, clock inhibit, or serial inputs.

Features

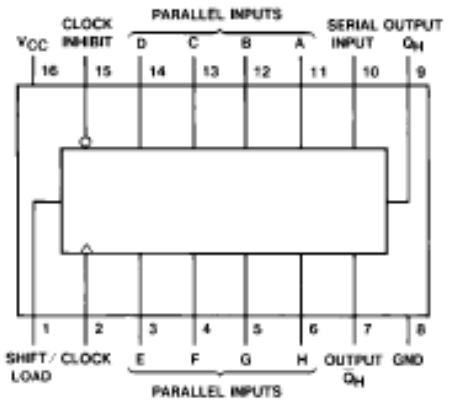
- Complementary outputs
- Direct overriding (data) inputs
- Gated clock inputs
- Parallel-to-serial data conversion
- Typical frequency 35 MHz
- Typical power dissipation 105 mW

Ordering Code:

Order Number	Package Number	Package Description
DM74LS165M	M16A	16-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-012, 0.150 Narrow
DM74LS165WM	M16B	16-Lead Small Outline Intergrated Circuit (SOIC), JEDEC MS-013, 0.300 Wide
DM74LS165N	N16E	16-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Function Table

Inputs					Internal Outputs		Output
Shift/ Load	Clock Inhibit	Clock	Serial	Parallel A...H	Q_A	Q_B	Q_H
L	X	X	X	a...h	a	b	h
H	L	L	X	X	Q_{A0}	Q_{B0}	Q_{H0}
H	L	↑	H	X	H	Q_{An}	Q_{Gn}
H	L	↑	L	X	L	Q_{An}	Q_{Gn}
H	H	X	X	X	Q_{A0}	Q_{B0}	Q_{H0}

H = HIGH Level (steady state)
L = LOW Level (steady state)
X = Don't Care (any input, including transitions)
↑ = Transition from LOW-to-HIGH level
a...h = The level of steady-state input at inputs A through H, respectively.
Q_{A0}, Q_{B0}, Q_{H0} = The level of Q_A, Q_B, or Q_H, respectively, before the indicated steady-state input conditions were established.
Q_{An}, Q_{Gn} = The level of Q_A or Q_G, respectively, before the most recent ↑ transition of the clock.

Experiment 7 - Design and realization a Synchronous counters using flip-flops

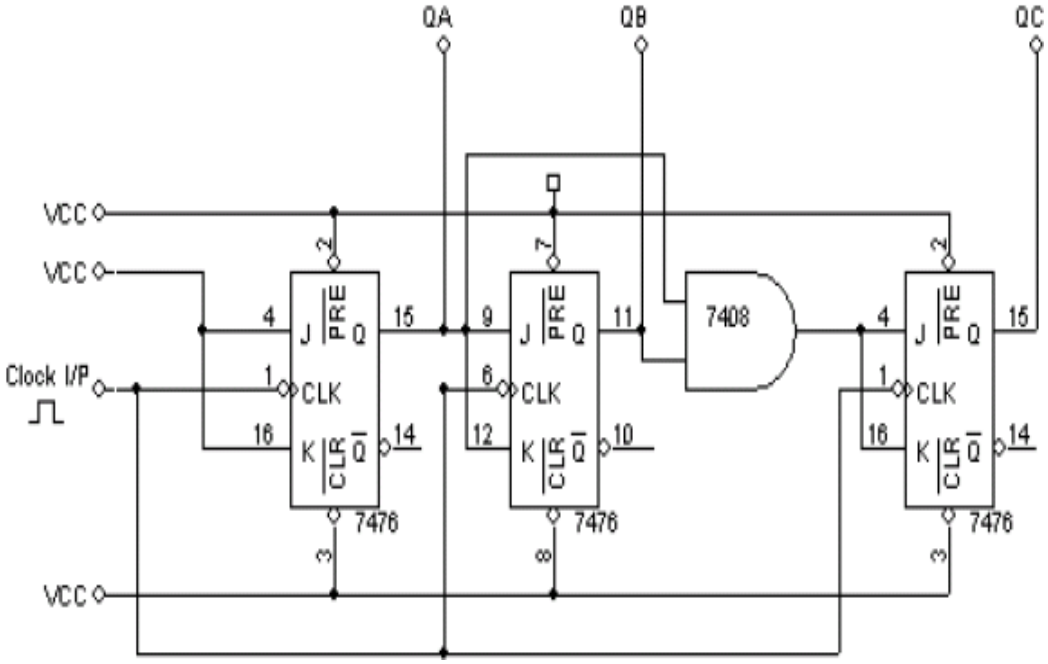
Aim:- To design and construct of 3-bit Synchronous up and down counters,2-bit up/down counter.

Apparatus:

1. IC's - 7408,7476,7400,7432
2. Electronic circuit designer
3. Connecting patch chords

Circuit Diagram:

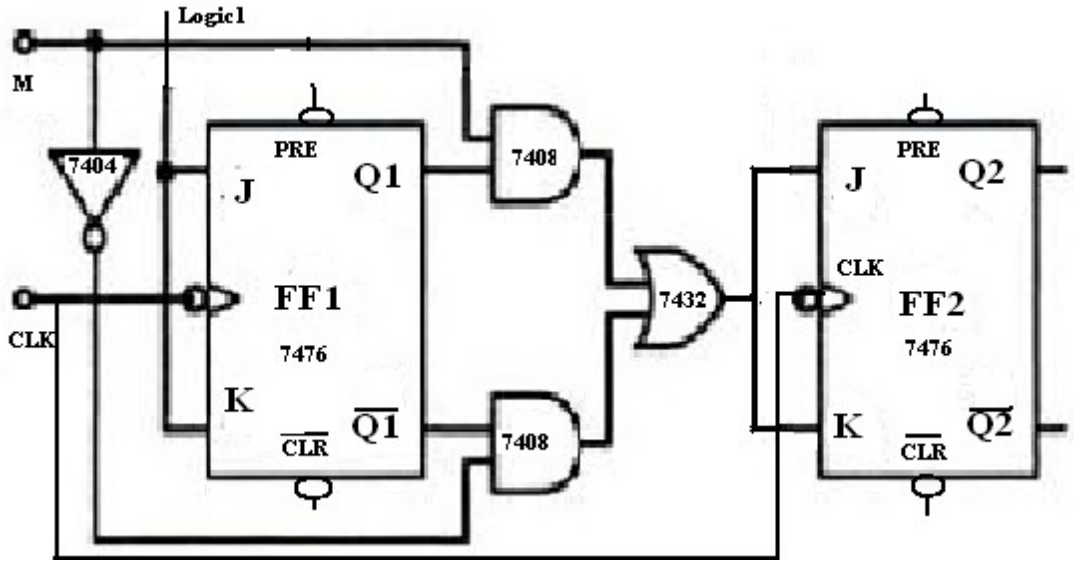
3-bit Synchronous Counter:-



Truth Table

3-bit synchronous up counter			
Clock	QC	QB	QA
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

Two Bit up/Down Counter using negative edge-triggered flip-flops



WHEN M=1

CLK	Q2	Q1
0	0	0
1	0	1
2	1	0
3	1	1

WHEN M=0

CLK	Q2	Q1
0	1	1
1	1	0
2	0	1
3	0	0

Procedure:

1. Connections are made as per the circuit diagram
2. Switch on the power supply.
3. Apply clock pulses and note the outputs after each clock pulse and note done the out puts.

Precautions:

1. All the connections should be made properly.
2. IC should not be reversed.

Result: 3-bit Synchronous up and down counters,2-bit up/down counter are designed and truth tables are verified.

Experiment 8- Design and realization Asynchronous counters using flip-flops

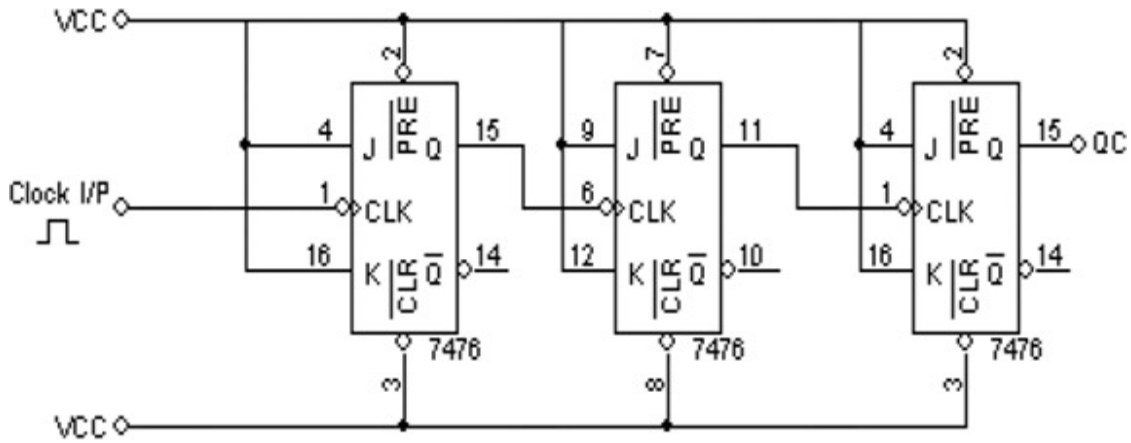
Aim:- To design and construct of 3-bit Asynchronous up and down counters,2-bit up/down counter.

Apparatus:

1. IC's - 7408,7476,7400,7432
2. Electronic circuit designer
3. Connecting patch chords

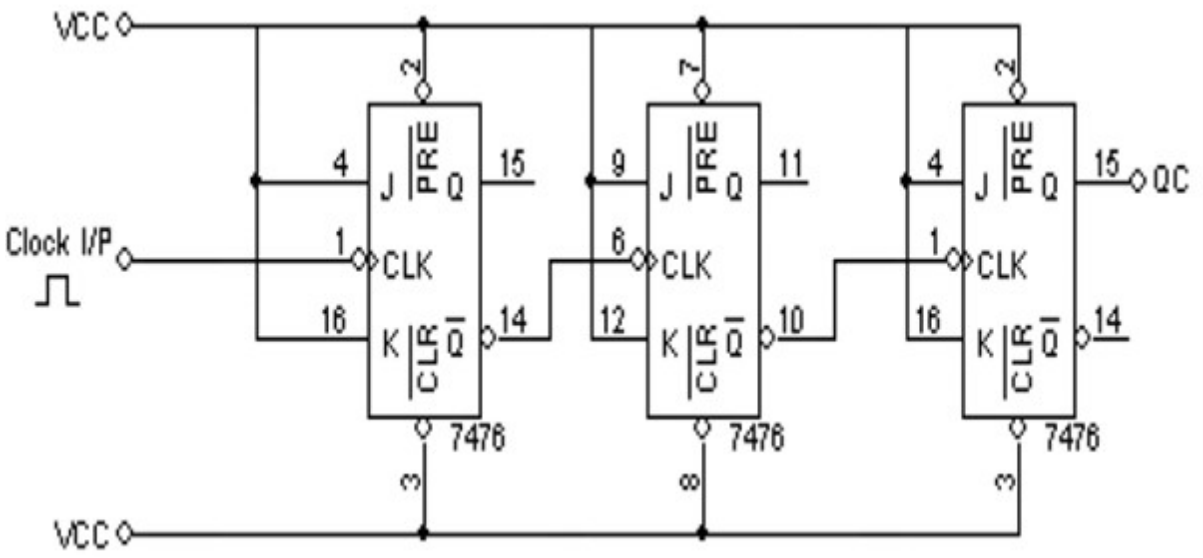
Circuit Diagram:

3-bit Asynchronous up counter:



3-bit Asynchronous up counter			
Clock	QC	QB	QA
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

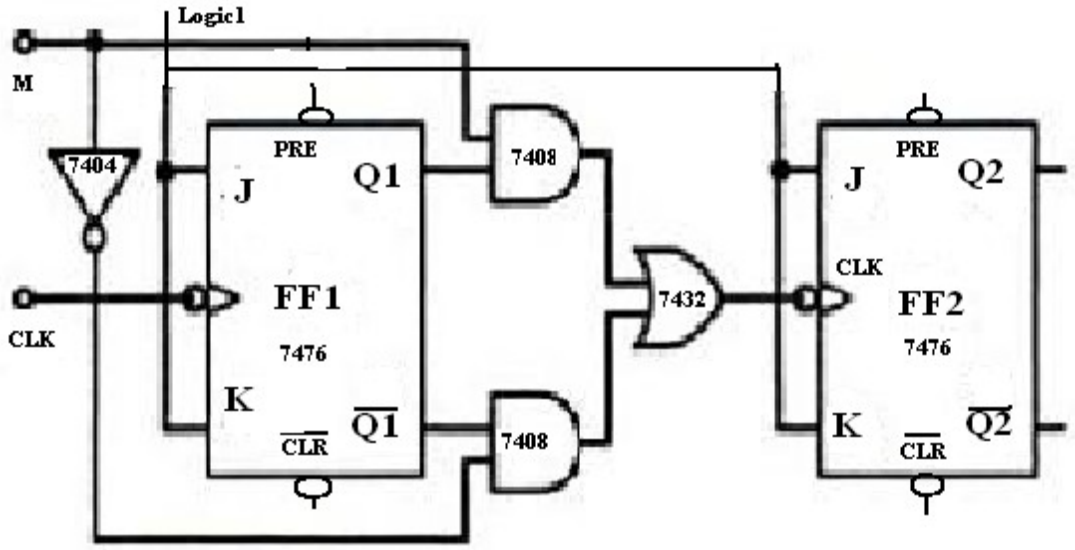
3-bit Asynchronous down counter:



TRUTH TABLE

3-bit Asynchronous down counter			
Clock	QC	QB	QA
0	1	1	1
1	1	1	0
2	1	0	1
3	1	0	0
4	0	1	1
5	0	1	0
6	0	0	1
7	0	0	0
8	1	1	1
9	1	1	0

Two Bit up/Down Counter using negative edge-triggered flip- flops



WHEN M=0

CLK	Q2	Q1
0	1	1
1	1	0
2	0	1
3	0	0

WHEN M=1

CLK	Q2	Q1
0	0	0
1	0	1
2	1	0
3	1	1

Procedure:

1. Connections are made as per the circuit diagram
2. Switch on the power supply.
3. Apply clock pulses and note the outputs after each clock pulse and note done the out puts.

Precautions:

1. All the connections should be made properly.
2. IC should not be reversed.

Result: 3-bit Asynchronous up and down counters,2-bit up/down counter are designed and truth tables are verified.

Experiment 9- Design and realization 8x1 using 2x1 mux

Objective: To realize and implement

Eight-to-one-line Multiplexer Using IC two to one line Multiplexer.

Components Required:

- Mini Digital Training and Digital Electronic Sets.
  - IC 7404, IC 7408, IC 7432, IC 74153,

Theory:

A multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line. The selection of a particular input line is contro1led by a set of selection lines. Normally, there are 2^n input lines and n selection lines whose bit combinations determine which input is selected.

Two-to-One-line Multiplexer (2x1 MUX): A two-to-one-line multiplexer connects one of two 1-bit sources to a common destination, as shown in Fig 1. The circuit has two data input lines, one output line, and one selection line S.

When S = 0, the upper AND gate is enabled and I_0 has a path to the output. When S = 1, the lower AND gate is enabled and I_1 has a path to the output. Multiplexer is often labeled as MUX in the block diagram

Truth table for 2x1 MUX

S	Y
0	I_0
1	I_1

$Y = S'I_0 + SI_1$

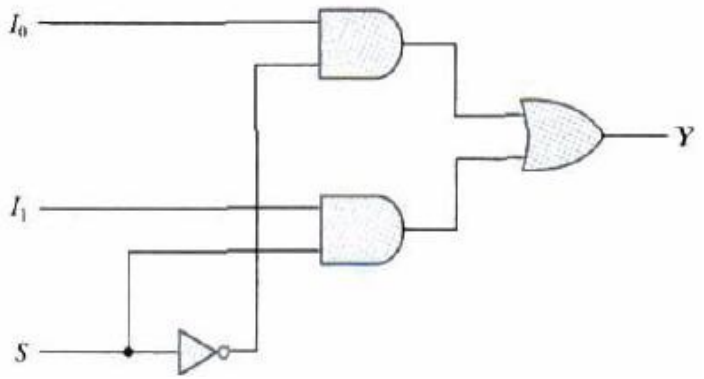


Fig1: Logic diagram of 2x1 MUX

Eight-to-one-line multiplexer (8x1 MUX): An eight-to-one-line multiplexer is a combinational circuit where one of the eight inputs is connected to one output. D_0, D_1, D_2… D_7 are the eight inputs. There are three select lines S_0, S_1, and S_2 to select input.

Truth table for 8-1 multiplexer:

S2	S1	S0	Y
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7

Each of the eight inputs, D_0 through D_7 is applied to one input of an AND gate. Selection lines S_0, S_1, and S_2 are decoded to select a particular AND gate.

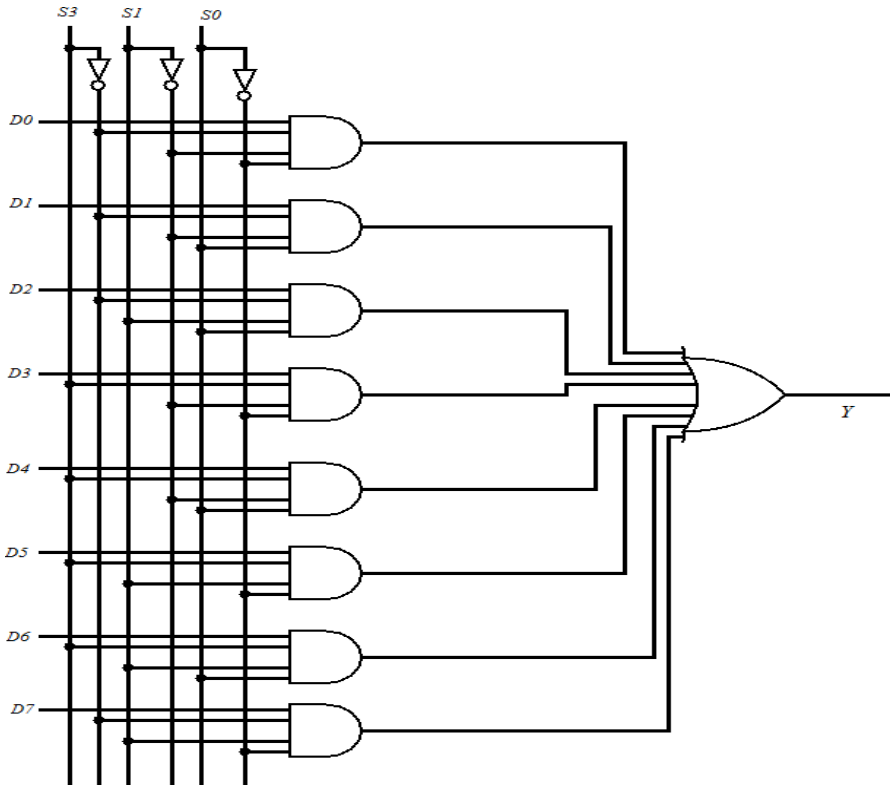
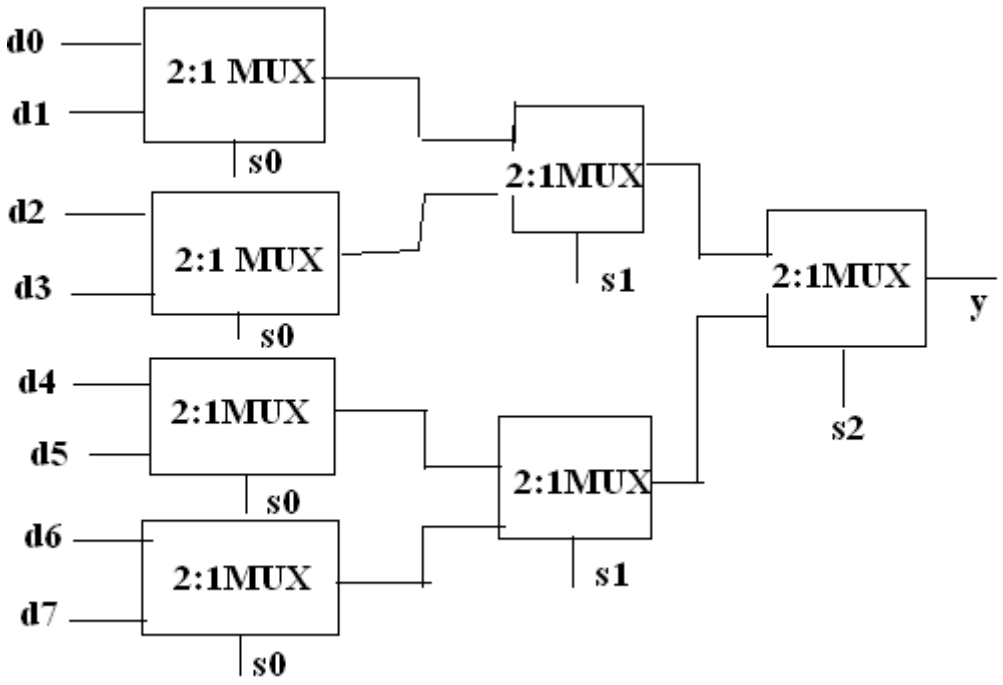


Fig: Logic diagram of 8-1 multiplexer



Procedure:

1. Implement 2x1 multiplexer using logic gates and breadboard, and verify the truth table.
2. Implement 8x1 multiplexer using IC 4051.

Result:

Experiment 10 - Design and realization of 4 bit comparator

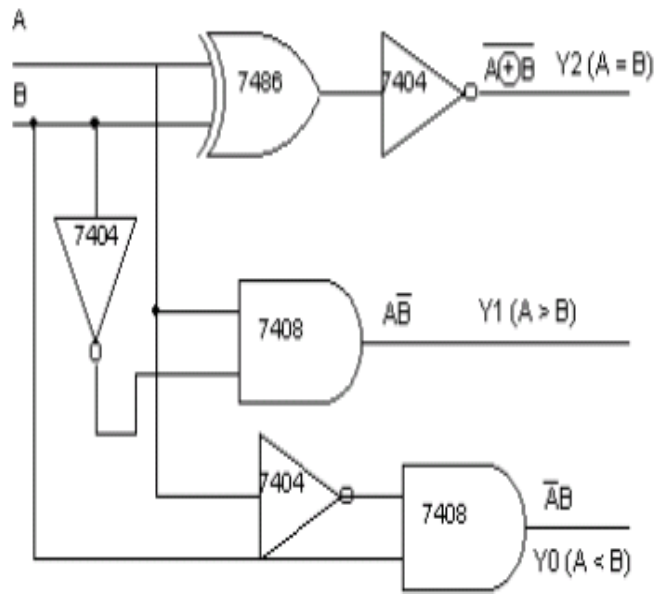
Aim: - To verify the truth table of one bit and four bit comparators using logic Gates and IC 7485

Apparatus : -

IC's -7486, 7404, 7408 and 7485

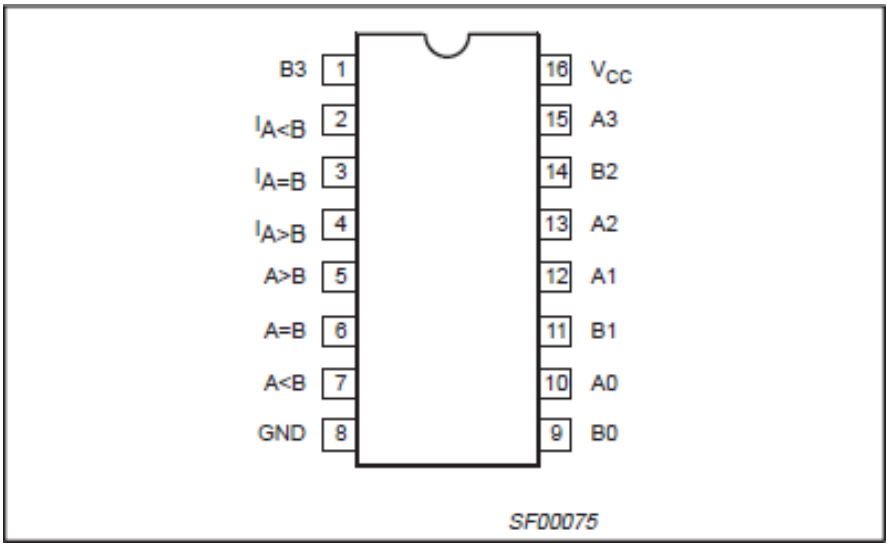
Circuit diagrams:

One Bit Comparator: -

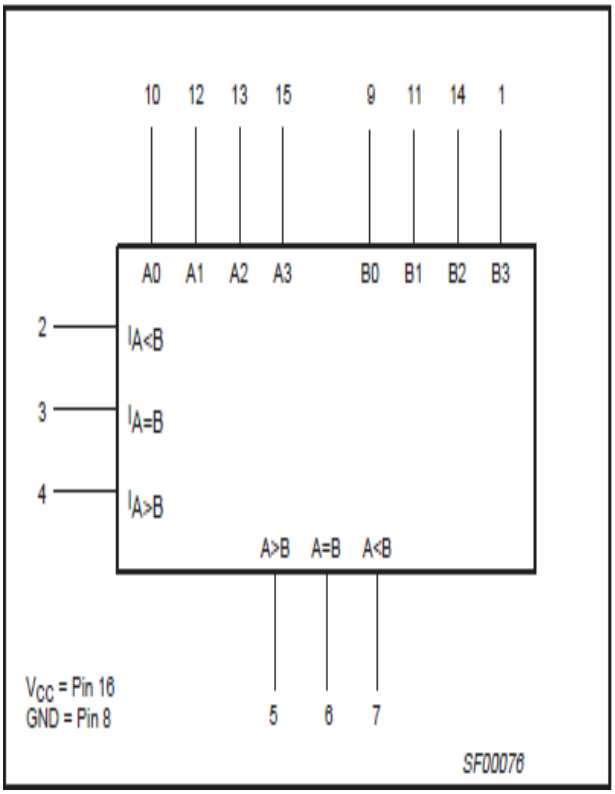


A	B	Y1 (A>B)	Y2 (A = B)	Y3 (A < B)
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

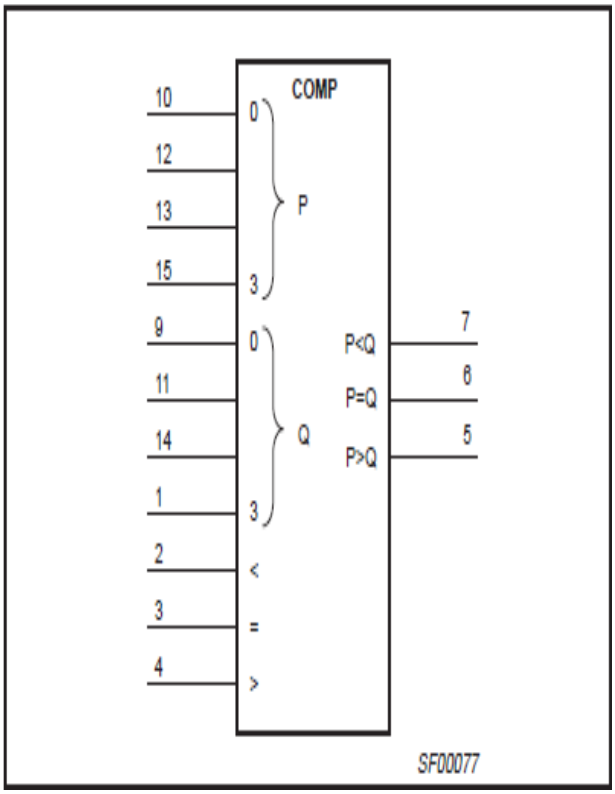
IC 7486 (4 bit Magnitude comparator) pin configuration:



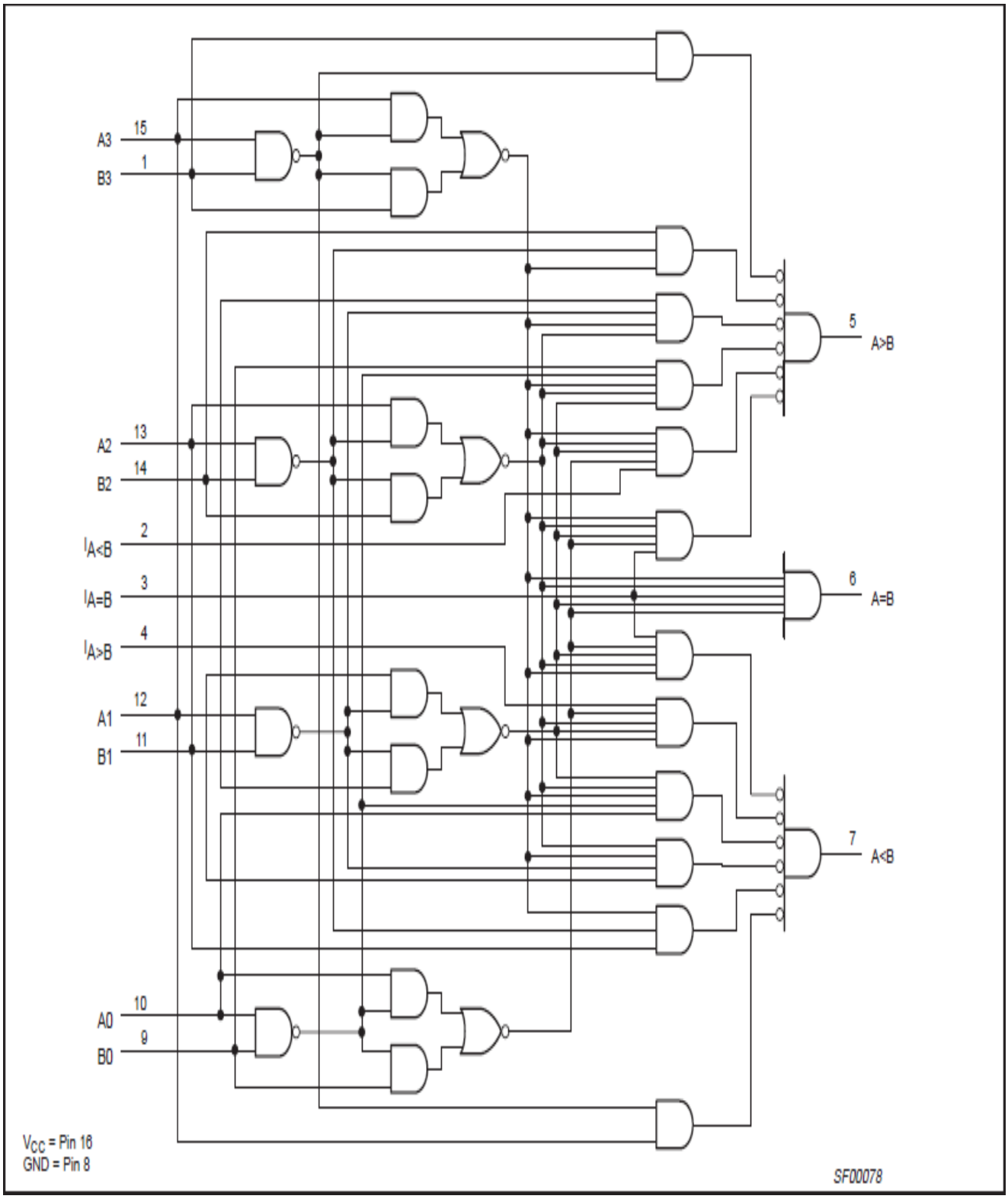
LOGIC SYMBOL



IEC/IEEE SYMBOL



IC 7486 (4 bit Magnitude comparator) Logic diagram:



FUNCTION TABLE

COMPARING INPUTS				EXPANSION INPUTS			OUTPUTS		
A3,B3	A2,B2	A1,B1	A0,B0	$I_{A>B}$	$I_{A<B}$	$I_{A=B}$	A>B	A<B	A=B
A3>B3	X	X	X	X	X	X	H	L	L
A3<B3	X	X	X	X	X	X	L	H	L
A3=B3	A2>B2	X	X	X	X	X	H	L	L
A3=B3	A2<B2	X	X	X	X	X	L	H	L
A3=B3	A2=B2	A1>B1	X	X	X	X	H	L	L
A3=B3	A2=B2	A1<B1	X	X	X	X	L	H	L
A3=B3	A2=B2	A1=B1	A0>B0	X	X	X	H	L	L
A3=B3	A2=B2	A1=B1	A0<B0	X	X	X	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	H	L	L	H	L	L
A3=B3	A2=B2	A1=B1	A0=B0	L	H	L	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	H	L	L	H
A3=B3	A2=B2	A1=B1	A0=B0	X	X	H	L	L	H
A3=B3	A2=B2	A1=B1	A0=B0	H	H	L	L	L	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	L	H	H	L

H = High voltage level
L = Low voltage level
X = Don't care

Verification Table :

A3 A2 A1 A0	B3 B2 B1 B0	OUTPUT
1 1 1 1	1 1 1 0	A>B
1 0 0 0	1 0 0 0	A=B
0 0 0 0	1 1 1 1	A<B

DESCRIPTION:

The 74F85 is a 4-bit magnitude comparator that can be expanded to almost any length. It compares two 4-bit binary, BCD, or other monotonic codes and presents the three possible magnitude results at the outputs. The 4-bit inputs are weighted (A0–A3) and (B0–B3) where A3 and B3 are the most significant bits. The operation of the74F85 is described in the Function Table, showing all possible

logic conditions. The upper part of the table describes the normal operation under all conditions that will occur in a single device or in a series expansion scheme. In the upper part of the table the three outputs are mutually exclusive. In the lower part of the table, the outputs reflect the feed-forward conditions that exist in the parallel expansion scheme.

The expansion inputs IA>B, and IA=B and IA<B are the least significant bit positions. When used for series expansion, the A>B, A=B and A<B outputs of the lease significant word are connected to the corresponding IA>B, IA=B and IA<B inputs of the next higher stage. Stages can be added in this manner to any length, but a propagation delay penalty of about 15ns is added with each additional stage. For proper operation, the expansion inputs of the least significant word should be tied as follows: IA>B = Low,

Procedure: -

1. Connect the circuit as shown in fig. Feed the 4-bit binary words A0, A1, A2 , A3 and B0, B1 , B2 , B3 from the logic input switches.

2. Pin 3 of IC 7485 should be at logic 1 to enable compare operation.

3. Observe the output A>B, A=B , and A<B on logic indicators. The outputs must be 1 or 0 respectively.

4. Repeat the steps 1 ,2 and 3 for various inputs A0 ,A1 , A2 , A3 and B0 , B1 , B2 , B3 and observe the outputs at A>B , A=B and A<B .

Precautions:

1. All the connections should be made properly.

2. IC should not be reversed.

Result: The truth tables of one bit and four bit magnitude comparators are verified.

Experiment 11 - Design and Realization of a sequence detector-a finite state machine

Aim: Design of a Moore's State Machine for 101 sequence detection using ICs.

Component/Equipment Required:

Sl. No.	COMPONENT	SPECIFICATION	QTY.
1.	D FLIP FLOP	IC 7474	2
2.	2 INPUT OR GATE	IC 7432	1
3.	2 INPUT AND GATE	IC 7408	1
4.	NOT GATE	IC 7404	1
4.	IC TRAINER KIT	-	1
5.	PATCH CORDS	-	20

Theory:

A Moore Machine is a Finite State Machine (FSM) whose output depends only on the present state. The block diagram of a Moore FSM is shown in Fig. 2.1

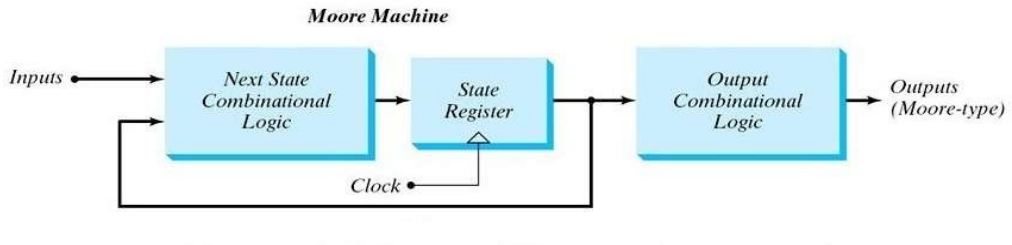


Fig. 2.1: Block diagram of Moore State Machine

The logic diagram of 101 sequence detector is shown in Fig 2.2.

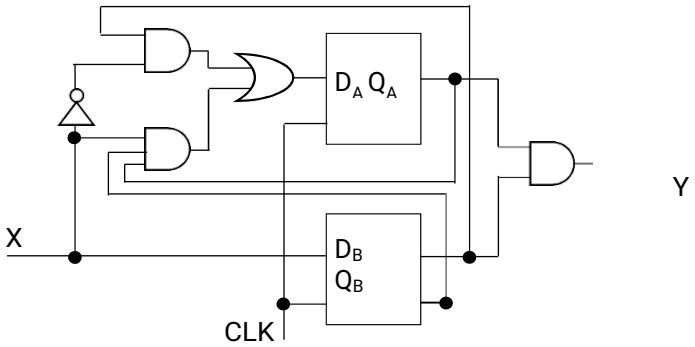


Fig. 2.2: Logic diagram of Moore machine for 2's complement computation

Pin diagram of IC 7474 is given in Fig. 2.3

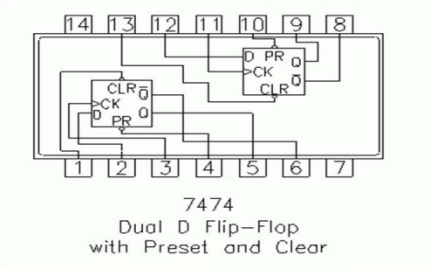


Fig. 2.3: Pin diagram of IC 7474 (Dual D Flipflop with preset and clear)

Pin diagram of IC 7432 is given in Fig. 2.4

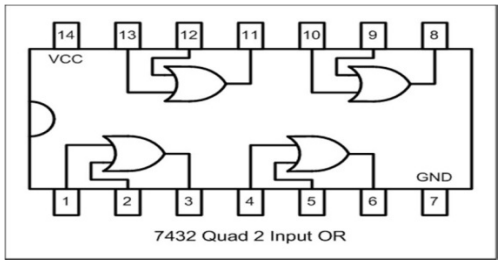


Fig. 2.4: Pin diagram of IC 7432 (two input OR Gate)

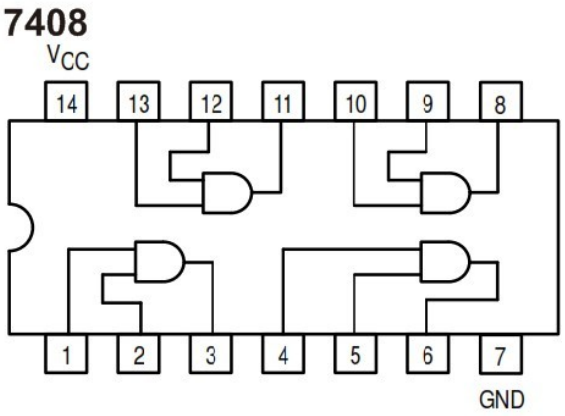
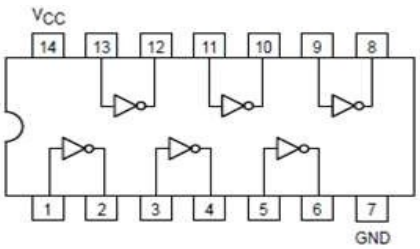


Fig. 2.5: Pin diagram of IC 7408 (Two input AND



Gate) Pin diagram of IC 7404 is given in Fig. 2.6

Fig. 2.6: Pin diagram of IC 7404 (NOT gate)

State Diagram

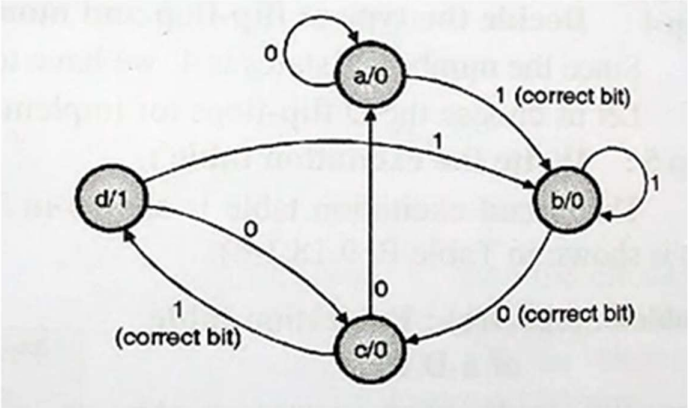


Fig. 2.7: State Diagram of 101 Sequence Detector

Procedures:

1. Make connections as per the logic diagram shown in Fig. 2.2.
2. Connect the input to the switch of the trainer kit.
3. Connect the output to the LEDs of the trainer kit.
4. Connect the power supply.
5. Verify the connections again.
6. Start the experiment by turning on the power of trainer kit.
7. Record the observation as per the observation table.

Observation:

Input X	Flip Flop Outputs		Output Y
	A	B	

Aim: Design of a Mealy's State Machine for serial 2's complement using ICs.

Component / Equipment Required:

Sl. No.	COMPONENT	SPECIFICATION	QTY.
1.	D FLIP FLOP	IC 7474	1
2.	2 INPUT OR GATE	IC 7432	1
3.	2 INPUT XOR GATE	IC 7486	1
4.	IC TRAINER KIT	-	1
5.	PATCH CORDS	-	20

Theory:

A Mealy Machine is a Finite State Machine (FSM) whose output depends on the present state as well as the present input. The block diagram representation is given in Fig. 3.1

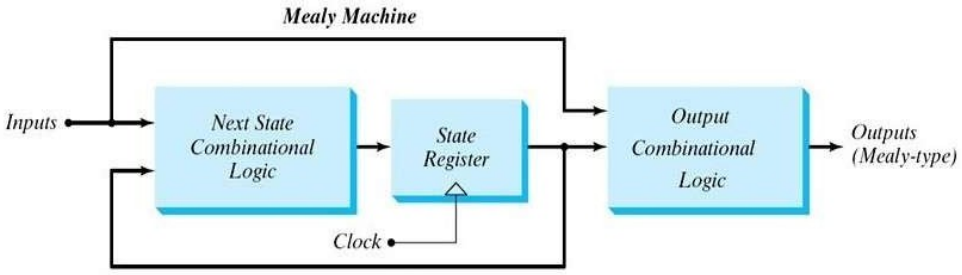


Fig. 3.1: Block diagram of Mealy State

Machine The logic diagram of sequential 2's complement is shown

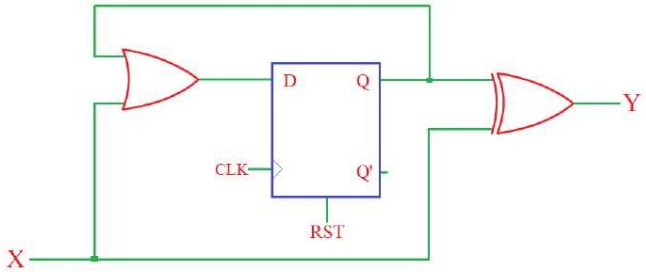


Fig. 3.2: Logic diagram of Mealy machine for 2's complement computation

Pin diagram of IC 7474 is given in Fig. 3.3

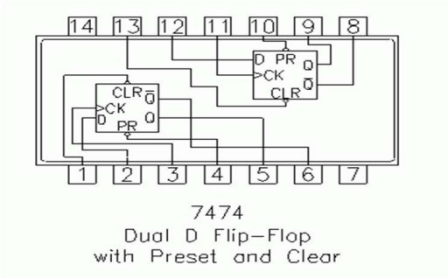


Fig. 3.3: Pin diagram of IC

7474 Pin diagram of IC 7432 is given in Fig. 3.4

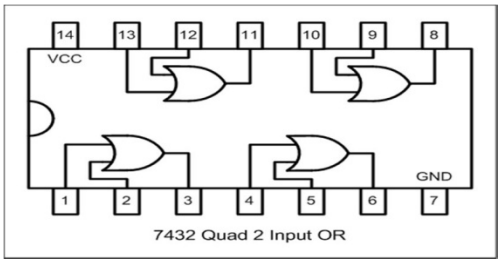


Fig. 3.4: Pin diagram of IC

7432 Pin diagram of IC 7486 is given in Fig. 3.5

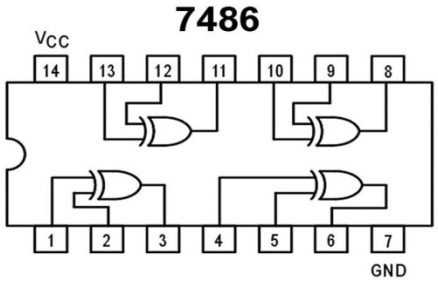


Fig. 3.5: Pin diagram of IC 7486

Procedures:

1. Make connections as per the logic diagram shown in Fig. 3.2.
2. Connect the input to the switch of the trainer kit.
3. Connect the output to the LEDs of the trainer kit.
4. Connect the power supply.
5. Verify the connections again.
6. Start the experiment by turning on the power of trainer kit.
7. Record the observation as per the observation table.

Observation:

Input X	Flip Flop Output	Output Y