M.Sc. (Computer Science)
Master of Computer Application

MS-03 / MCA-203

Course Curriculum for

Digital Electronics

125001

: 1

Author :
Dr. Devendra Mohan

Vetter :
Dr. Sib Krishna Ghoshal

Introduction

All of us are familiar with the impact of digital calculators, watches, modern communication systems and computers in every day life. All persons working in various fields related to electronics must understand the performance of Digital Electronic Circuits. All sizes of computers, as we know, perform complicated task with fantastic speed and accuracy. At stores, the cash register read out digital display digital clock and watches flash the time in all city shops and restaurants. Most automobiles use microprocessors to control engine functions. Aircraft's defense sectors, factory machines and modern diagnostic in medical science are controlled by digital circuits.

Therefore, one asks obvious questions like:

- What is a digital circuit?
- How digital circuits work?
- Why use digital circuits?
- How one makes a digital signal? How does one test a digital signal? And so on, a long list of queries.

This revolution took place with the advent of integrated circuits (IC) which is an offspring of semiconductor technology. The inexpensive fabrication of ICs has made the subject Digital Electronics easy to study. One small IC can perform the task of thousands of Transistors Diodes and Resistors. Many ICs are used to construct Digital Circuits. This is an exciting and rapidly growing field, which uses several principles for the working of computers, Communication systems, Digital machinery's etc. The basic idea is to let the beginners understand the operation of the Digital system and many other systems based on the principles of Digital Techniques. Any

device working under Digital Techniques are called Digital Systems and the Electronic Network used to make them operational are called Digital Circuits. The subject as a whole is often referred as **Modern Digital Electronics.**

Electronic circuits use two kinds of signals. They are **Analog Signals** (continuous supply of voltages and currents) and **Digital Signals** (discrete voltages and current). For example, Circuits (electronic network) using Analog signals are known as **Linear or Analog Circuits**. Similarly, the electronic network of an electronic calculator or digital watch that uses Digital signals are called **Digital Circuits**.

An analog device, then, is one that has a signal, which varies continuously in time with the input, whereas, a digital device operates with a digital signal that varies discontinuously. As a result, the Digital Electronics is the world of **ZEROS'** (OFF/ LOW/DOWN/FALSE) and '**ONES**' (ON/HIGH/UP/TRUE). Figure1.1 shows the behavior of Analog and Digital signal and the possibility of conversion from Analog to Digital. Figure 1.2 represents a symbol of some devices using Analog and Digital world. This example is set to explain how real life problem like movement of a pivot on a water tank that indicates the level of water can be translated to analog/digital form.

Figure

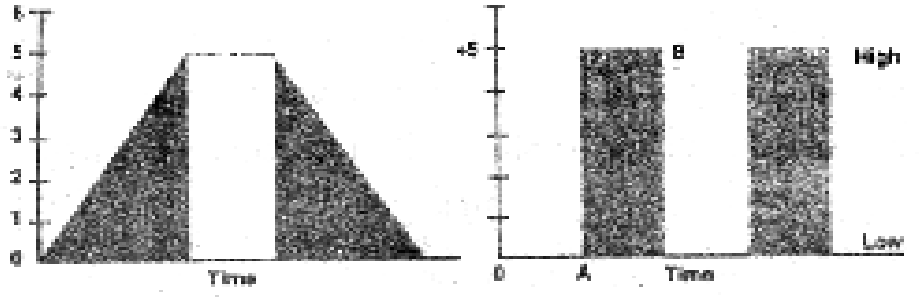


Figure 1.1 (a) Analog Signal Waveform

Figure 1.1 (b) Digital Signal Waveform

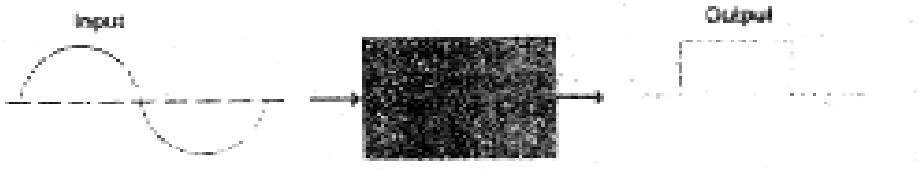


Figure 1.1 (c) Block Diagram of Electronic Circuit Shaping a Sine Wave into a Square Wave

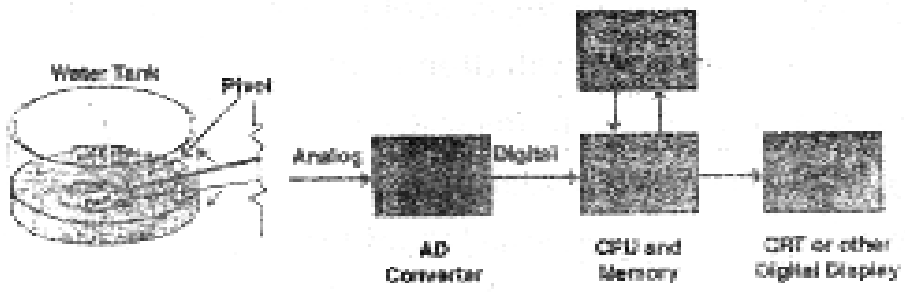


Figure 1.2 Digital System used to Interpret Float Level in Water Tank

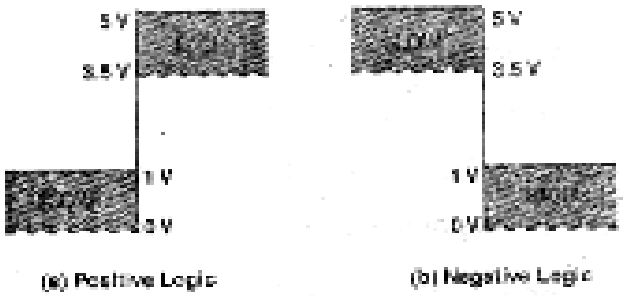


Figure 1.3 Digital Signal Representation

Digital system works under logic and hence they are called **Logic Circuits**, whose building blocks are known as **Gates.** These circuits employ two different representations of digital signal known as Positive Logic Systems and Negative Logic Systems as shown in Figure 1.3. The two discrete signal levels HIGH and LOW are generally represented by **Binary Digits** 1 and 0 respectively is referred to as **bit** and binary number with 8 bits is known as a **byte**.

Since a digital signal can have only one of the two possible level 1 and 0, the **Binary Number System** can be used for the analysis and design of digital system, which was introduced by George Boolean in 1854 and the corresponding algebra is known as **Boolean Algebra**. These logic concepts have been adopted for the design of digital circuit.

The number system that we use in day-to-day life is called **Decimal Number System**. In this system, one works with 10 different Digits, (0 through 9) and is known as based-ten system. But digital electronic devices used a 'strange' number system called **binary**. Digital computers and microprocessor-based systems use other strange

number systems called **Hexadecimal** and **Octal**. One who works in electronics must know how to convert numbers from the everyday decimal system to binary, to Hexadecimal, and to Octal system.

The hexadecimal number system uses the 16 symbol: 0 through 9,A, B, C, D, E, and F and is referred to as base-sixteen system. The letter 'A' stands for decimal 10, 'B' for decimal 11, and so on. The Octal number system uses the 8 symbols: 0 through 7 and are referred to as base-eight system

Any **Decimal number** of any magnitude can be expressed by using the system of positional weighting in which the right most digit of the number called the 'least significant digit' is multiplied by 10^0 to the number, the digit left to the least significant digit is multiplied by 10^1. Similarly, as we move in the number towards left side the power increases in steps of 1.

For example in decimal number $(386)_{10}$ the weightage of digit 6 is $6x10^0=6$, the weightage of digit 8 is $8x10^1= 80$ and for digit 3 is $3x10^2=300$.

Summing all three values 6, 80, and 300 we get

386 =3 x hundred + 8 x ten + 6 x unity

$=3x10^2 + 8x10^1 + 6x10^0$

$=3x100 + 8x10 + 6x1$

$=300 + 80 + 6=(386)_{10}$

The **binary number system** is exactly like the Decimal system except that the base is 2 instead of 10. Again each position in a binary number represent a power of the base 2. In this system, the right most position is the unit 2^o position, the second position from the right is the 2's (2^1), and proceeding in this way, we have 4's (2^2), 8's (2^3) position, and so on.

Thus, the decimal equivalent of the binary number 10101 (written as $10101)_2$ is

$1x2^4+0x2^3+1x2^2+0x2^1+1x2^0$

or $16+0+4+0+1$ or 21

Thus, we write $(10101)_2 = (21)_{10}$

In the **octal system** the largest single digit is 7 (one less than the base). Again each position an octal number represent a power of the base 8.

Thus the decimal equivalent of the octal number 943 (written as $943)_8$ is

$9x8^2+4x8^1+3x8^0$

or $9x64+32+3$

or $576+32+3$

or 611

so we have $(943)_8 = (6!!)_{10}$

In hexadecimal system, the largest single digit is F or 15 (one less than the base). Again, each position in a Hexadecimal system represents a power of the base 16.

Thus, the decimal equivalent of the Hexadecimal number 3AF written as $(3AF)_{16}$ or H is

$3x16^2+Ax16^1+Fx16^0$

or $3x256+10x16+15x1$

or $768+160+15$

or 943

Thus $(3AF)_{16} = (943)_{10}$

And $(3AF)_{16} = (1110101111)_2$

The **Binary Coded Decimal (BCD)** Code is one of the early memory codes. It is based on idea of converting each digit of a decimal number into its binary equivalent rather than converting the entire decimal value into a pure binary form.

Converting $(943)_{10}$ into BCD, results the following

$(943)_{10}$ = 1001 0100 0011

9 4 3

or 100101000011 in BCD

Table 1.1 represents binary, hexadecimal, BCD equivalence to decimal numbers and Table1.2 represent alphabetic and numeric characters in BCD along with their octal equivalent

Decimal	*Binary*	*Hexadecimal*	*BCD Equivalent*
0	***0000***	***0***	***0000***
1	***0001***	***1***	***0001***
2	***0010***	***2***	***0010***
3	***0011***	***3***	***0011***
4	***0100***	***4***	***0100***
5	***0101***	***5***	***0101***
6	***0110***	***6***	***0110***
7	***0111***	***7***	***0111***
8	***1000***	***8***	***1000***
9	***1001***	***9***	***1001***
10	***1010***	***A***	***00010000***
11	***1011***	***B***	***00010001***
12	***1100***	***C***	***00010010***
13	***1101***	***D***	***00010011***
14	***1110***	***E***	***00010100***
15	***1111***	***F***	***00010101***
16	***10000***	***10***	***00010110***
17	***10001***	***11***	***00010111***

Table 1.1

Characters	*Code Digit (BCD)*	*Octal Equivalent*
A	0001	61
B	0010	62
C	0011	63
D	0100	64
E	0101	65
F	0110	66
G	0111	67
H	1000	70
I	1001	71

J	0001	41
K	0010	42
L	0011	43
M	0100	44
N	0101	45
O	0110	46
P	0111	47
Q	1000	50
R	1001	51
S	0010	22
T	0011	23
U	0100	24
V	0101	25
W	0110	26
X	0111	27
Y	1000	30
Z	1001	31
1	0001	01
2	0010	02
3	0011	03
4	0100	04
5	0101	05
6	1001	06
7	0111	07
8	1000	10
9	1001	11
0	1010	12

Table 1.2

Characters	***Digit***	***ASCII-7 code Hexadecimal equivalent***	***Digit***	***ASCII-7 code Hexadecimal equivalent***
0	0	30	0	50
1	1	31	1	51
2	10	32	10	52
3	11	33	11	53
4	100	34	100	54

5	101	35	101	55
6	110	36	110	56
7	111	37	111	57
8	1000	38	1000	58
9	1001	39	1001	59
A	1	41	1	A1
B	10	42	10	A2
C	11	43	11	A3
D	100	44	100	A4
E	101	45	101	A5
F	110	46	110	A6
G	111	47	100	A7
H	1000	48	1000	A8
I	1001	49	1001	A9
J	1010	4A	1010	AA
K	1011	4B	1011	AB
L	1100	4C	1100	AC
M	1101	4D	1101	AD
N	1110	4E	1110	AE
O	1111	4F	1111	AF
P	0	50	0	B0
Q	1	51	1	B1
R	10	52	10	B2
S	11	53	11	B3
T	100	54	100	B4
U	101	55	101	B5
V	110	56	110	B6
W	111	57	111	B7
X	1000	58	1000	B8
Y	1001	59	1001	B9
Z	1010	5A	1010	BA

Table 1.3 Represents Numeric and Alphabetic Characters in ASCII- and ASCII-8 Notation along with their Hexadecimal Equivalent

Another important code that is very widely used in computer is the American Standard Code for Information Interchange (ASCII). This code is popular in data communications. ASCII is of two types: ASCII-7 and ASCII-8. ASCII-7 is a 7 bit code that allows 2^7(128) different characters. ASCII-8 (8bit code) is an extended version of ASCII-7 that allows 28(256) different characters as shown in Table 1.3.

The binary code for the word BOY in ASCII-7 can be represented as

1000010	1001111	1011001
B	O	Y

The first 3 bits in each of the character (for example 100 for B, 100 for O and 101 for Y), are used as 'zone' bits which is internal code for ASCII. In ASCII-8, the representation of BOY will be

1000010	1001111	1011001
B	O	Y

The first 4 bit in each of the character are used as zone bits.

It is therefore important to highlight the superiority of digital circuits and systems over the analog circuits. The 'Real-world' information deals with time, speed, weight, pressure, light intensity, and position measurement and is all analog in nature. Digital systems are required when data must be stored, used for calculations, or displayed as numbers/or letters. They are valuable when calculations, data manipulations, and alphanumeric outputs are required. The "**Central Processing Unit**"(CPU) of a computer can manipulate the input data, output the information, store the information and so forth.

Some of the **advantages** highlighted for the widespread use of digital circuitry in over analog are as follows

1. Inexpensive ICs can be used with few external components.
2. Operate in one of the two **states**, known as ON and OFF makes it very simple.
3. Only a few basic operations are required and are very easy to understand.

4. Digital techniques deal with simple logic mathematics called Boolean algebra.
5. Operation and network analysis of digital circuitry require simple basic concepts like switching speed and loading on the other hand, analysis of analog circuitry (Involved frequency and time domain) are quite complicated.
6. Information can be stored for short periods or indefinitely.
7. Data can be used for precise calculations.
8. Systems can be designed more easily using compatible digital logic families.
9. Systems can be programmed which show some manner of 'intelligence'. A number of programmable ICs are also available.
10. The display of data and other information is very convenient, accurate and elegant by using digital techniques.
11. Digital circuits have capability of memory, which makes these circuits highly suitable for computers, calculators, watches, telephones, medical diagnostics, etc.
12. To learn programming of digital computers it is worth knowing, the way the digital hardware works.

The limitations of digital circuitry are as follows:

Most 'real-world' events are analog in nature.

Analog processing is usually simpler and faster.

Digital circuits are appearing in more and more products primarily because of low-cost, reliable digital ICs. Other reasons for their growing popularity are accuracy, added stability, computer compatibility, memory, ease of use, simplicity of design, and compatibility with at alphanumeric displays.

: 2

Author :
Dr. Devendra Mohan

Vetter :
Dr. Sib Krishna Ghoshal

- **Boolean Algebra**
- **De Morgan's Theorem**
- **Representation of Boolean Algebra**
- **De Morgan's Theorem through Logic Circuits**
- **Karnaugh Map Mathod**
- **Binary/BCD Subtraction and Addition**

Boolean Algebra

Boolean Algebra is Algebra of logic. This is an algebra that deals with logical propositions, which are either true or false. This algebra is suitable for binary number system and is very useful in designing digital circuits, which operates under logic.

For example,

A+A =A, not 2A

Also 1+1 = 1 not 2 as it is logical expression.

One can visualize this as,

TRUE+TRUE=TRUE

FALSE+FALSE=FALSE

A Boolean algebraic expression is composed of variables, constants and operators. The **variables are** generally represented by the letters of the Alphabet (say A) which can have two possible values 1 or 0. The interpretation of 1 may be that

the variable is presented input signal is ON, is TRUE, and is a positive voltage. If A is 0, then it mean that the variable is absent, input signal if OFF, is FALSE, and is a negative voltage. Similarly, the **Boolean Constant** can have any two values, either 1 or 0.

Boolean Operators are used in Boolean Algebra where a mathematical function called **Boolean Function** is constructed. These operators are the Symbols
PLUS (+) meaning an OR operation
DOT (.) meaning and AND operation
BAR ‘A read as COMPLEMENT meaning a NOT operation.

Postulates of Boolean Algebra

A set of Boolean postulates are the following

(a) $A = 0$ iff $A \neq 1$

$A = 1$ iff $A \neq 0$

(b) $0.0=0$

(c) $1+1=1$

(d) $0+0=0$

(e) $1.1=1$

(f) $1.0=0.1=0$

(g) $1+0=0+1=1$

The realization of any Boolean Expression can be obtained with the help of a table called **TRUTH TABLE**. To simplify a Boolean Expression, one requires certain laws of Boolean Algebra.

Laws of Boolean Algebra and their Truth Table

a) Commutative Law

i) $A+B=B+A$

ii) $A.B=B.A$

Truth Table

A	*B*	*A+B*	*A.B*	*B+A*	*B.A*
0	0	0	0	0	0
0	1	1	0	1	0
1	0	1	0	1	0
1	1	1	1	1	1

b) Associative Law

i) A+(B+C)=(A+B)+C

ii) A.(B.C)=(A.B).C

Truth Table

A	*B*	*C*	*B+C*	*A+B*	*A+(B+C)*	*(A+B)+C*	*B.C*	*A.B*	*A.(B.C)*	*(A.B).C*
0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	1	0	0	0	0
0	1	0	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0
1	0	0	0	1	1	1	0	0	0	0
1	0	1	1	1	1	1	0	0	0	0
1	1	0	1	1	1	1	0	1	0	0
1	1	1	1	1	1	1	1	1	1	1

C) Distributive Law

i) A.(B.C)=(A.B)+(A.C)

ii) A+(B.C)=(A+B).(A+C)

Truth Table

A	*B*	*C*	*B+C*	*A.B*	*A.C*	*A.(B+C)*	*(A.B+A.C)*	*B.C*	*A+B*	*A+C*	*A+(B.C)*	*(A+B).(A+C)*
0	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
0	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**
0	**1**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**
0	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**
1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**
1	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**
1	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**0**	**1**	**1**	**1**	**1**
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**

De Morgan's Theorem

This is a useful theorem in Boolean Algebra which states how to complement a Boolean expression. They allow us to convert back and forth from minterm to maxterm forms of Boolean expression. It helps to eliminate long over-bars that cover several variables.

a) First Theorem

$\overline{A + B} = \overline{A} \cdot \overline{B}$ (For two variables)

In general $\overline{A + B + C} + \ldots\ldots = \overline{A} \cdot \overline{B} \cdot \overline{C} \ldots$. (For many variables)

b) Second Theorem

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

In general $\overline{A} \cdot \overline{B} \cdot \overline{C} = \overline{A} + \overline{B} + \overline{C} + \ldots\ldots$

Truth Table

A	B	$\overline{A}$	$\overline{B}$	AB	A+B	$\overline{A+B}$	$\overline{A.B}$
0	0	1	1	0	0	1	1
0	1	1	0	0	1	0	0
1	0	0	1	0	1	0	0
1	1	0	0	1	1	0	0

Representation of Boolean Algebra & De Morgan's Theorem through Logic Circuits

There are nine basic identities commonly used in converting complex Boolean expression to their simple forms. A Boolean identity usually consists of one variable and one constant equates two expressions, which are equal for all possible combinations of the variables. The equivalence of two expressions is presented through Truth Table and representative circuits symbol.

Double Complementation

A double complementation is the complementation of a single complement. The single complement is called **NOT** operation.

Expression : $\overline{A} = A$

Truth Table

A	$\overline{A}$	A
0	1	1
1	0	1

The triangle with a circle at the vertex is known as **inverter.** This circle is called '**Bubble'** and the expression for output A is read as 'A complement' or 'A NOT'.

In the language of logic gate, this is called NOT Gate. The equivalent inverter symbol looks like

AND Function Identities

The identity states that

A .1 = A

Corresponding **Truth Table** and circuit are given below

A	1	Y
0	**1**	**0**
1	**1**	**1**

The Boolean expression for the output is A.1, which is read as 'A and 1". In genral this is A.B.

Similarly, there are other identities using AND operator.

i) A.0=0

A	0	Y
0	0	0
1	0	0

i) A.A=A

A	A	Y
0	0	0
1	1	1

i) A.$\overline{A}$=0

A	$\overline{A}$	Y
0	1	0
1	0	0

OR Function Identities

This identity can be realized with one input permanently tied to logic 1 and the other is varying

A+1=1

A	1	Y
0	1	1
1	1	1

Other three identities using OR operator are

i) **A+0 = A**

A	0	Y
0	0	0
1	0	1

ii) **A+A = A**

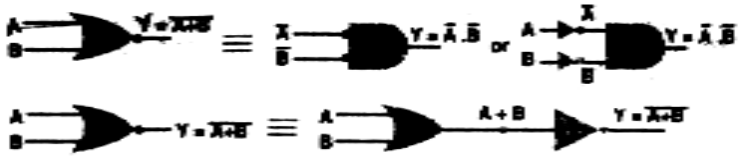
A	0	Y
0	0	0
1	0	1

iii) **$A + \overline{A} = 1$**

A	0	Y
0	0	0
1	0	1

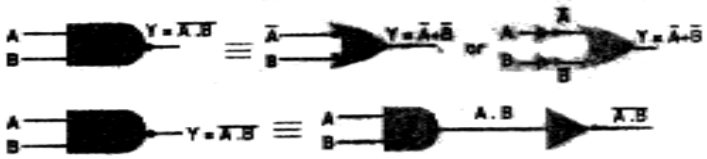
The Circuit Representation of De Morgan's Theorem

1. A+B = A.B known as NOR circuit, which is equivalent to Bubbled input AND circuit.



A	B	A+B	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

2. = $\overline{A} + \overline{B}$ known as NAND circuit, which is equivalent to Bubbled input OR circuit.



A	B	A.B	A.B		$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
0	0	0	1	$\overline{A.B}$	1	1	1
0	1	0	1		1	0	1
1	0	0	1		0	1	1
1	1	1	0		0	0	0

Karnaugh Map Method

There are two methods for developing the required logic diagram from a given Truth Table. The first method requires "**Boolean Algebra"** and "**De Morgan's Theorem**" to reduce the expressions produced to lowest term (Minimal expressions). The second method is a variation of the first and uses a tool called the "**Karnaugh Map (K-Map)**". The K-Map is the simple3st and most commonly used method. It is a graphical method (in the form of table) extensively used to simplify Boolean equation.

The K-Map method uses a table or map to reduce its expressions. Each position in the table is called a "CELL". CELLs are filled with ones and zeros according to the expressions to be reduced.

Adjacent ones are grouped together in clusters, called “subcobes”, following definite rules: a subcube must be of size 1,2,4,8,16, etc. All 1s must be included in a subcube of maximum size. These rules are explained through examples below

Assignment 1

Designed a circuit that will behave according to this Truth Table

Inputs			**Output**
C	**B**	**A**	**Y**
0	**0**	**0**	**0**
0	**0**	**1**	**0**
0	**1**	**0**	**1**
0	**1**	**1**	**1**
1	**0**	**0**	**0**
1	**0**	**1**	**0**
1	**1**	**0**	**1**
1	**1**	**1**	**1**

Solution

Step 1. Draw the table. Choose two of the variables to use as column headings across the top. We will chose C and B. From all combinations of C and ‘C with B and ‘B. Each column heading should differ from all adjacent column by one variable only.

Part 1

$\overline{C}\,\overline{B}$	$\overline{C}\,B$	$C\,B$	

Start with $\overline{C}\,\overline{B}$ and change $\overline{B}$ to B to from the heading for column 2, .

Then change $\overline{C}$ to C for the third column CB, and finally . The fourth

column wraps around to the first column and should differ by one variable only, which it does.

Part 1

$\overline{C}\,\overline{B}$	$\overline{C}\,B$	$C\,B$	

Use the third variable, A for row headings $\overline{A}$ and A.

Step 2. Fill the table with ones and zeros from the Truth Table. The output Y is 1 in line 3 when we have and B and . Place a 1 in the table in cell B . The output Y is also 1 on line 4, which is represented by BA, on line 6, which is C A, on line 7, whichis CB , and on line 8, which is CBA. Fill those cells with ones and the remaining cells with zeros.

Part 1

	$\overline{C}\,B$	$C\,B$	
0	**1**	**1**	**0**
0	**1**	**1**	**1**

Step 3. Combine adjacent cells that contain ones in sub cubes of maximum size. The four ones in the centre of the table compose a sub cube of size 4.

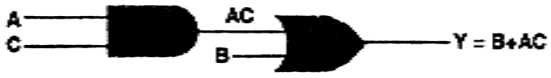
Part 1

$\overline{C}\,\overline{B}$	$\overline{C}\,B$	$C\,B$	
0	**1**	**1**	**0**
0	**1**	**1**	**1**

The 1 in cell $C\overline{B}A$ has not been included in a sub cube so it is used with its adjacent 1 in a sub cube of size 2.

Step 4. Write the expression that 3ach sub cube represents. In the sub cube of size 4, find the variable that occurs in all four cells. In these case B is the only variable that appears in all four cells. The sub cube of size 4 represents B. In the sub cube of size two, A and C appear in each cell, so the sub cube represents AC.

Steps 5. From the output expression. The output Y is the expression from each sub cube ORed together. In this case Y=B+AC.



The Truth Table can be implemented by the above logic diagram.

Verification

In the circuit, when A and C are both 1s, the output of the AND will also be 1. A 1 into an OR gives a 1 out. In the Truth Table, A and C are both 1s on line6 and 8, and the required output is 1. In the circuit, any time B is 1 the output is 1. In the Truth Table, B is 1 on lines 3, 4, 7 and 8, and the required output is 1. The rest of the time both inputs into the OR gate will be 0, and the result will be 0. This occur on lines of 1, 3 and 5 of the Truth Table where the output is 0. In all cases the circuit produces the results required by the Truth Table.

Assignment 2

Use a Karnaugh map to design a logic diagram to implement the following Truth Table.

Inputs Output

Inputs			Output
C	**B**	**A**	**Y**
0	**0**	**0**	**0**
0	**0**	**1**	**1**
0	**1**	**0**	**0**
0	**1**	**1**	**0**
1	**0**	**0**	**0**
1	**0**	**1**	**1**
1	**1**	**0**	**1**
1	**1**	**1**	**1**

Solution

Step 1. Draw the table.

$\overline{C}\,\overline{B}$	$\overline{C}\,B$	$C\,B$	
0	**1**	**1**	**0**
0	**1**	**1**	**1**

Step 2. Fill the table with 1s and 0s from the Truth Table.

Step 3. Combination adjacent cells that contain 1s into sub cubes (size 1, 2, 4, or 8).

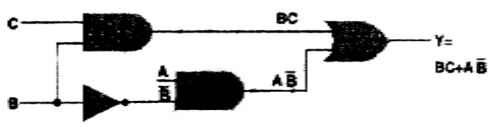
$\overline{C}\,\overline{B}$	$\overline{C}\,B$	$C\,B$	
0	**0**	**1**	**0**
1	**0**	**1**	**1**

The right side of the table **"wraps around"** to the other side so that the table is continuos. The 1s in the lower corners from a sub cube of size 2. The two sub cube **"cover"** the map in that all 1s are contained in sub cube. Any additional sub cube drawn would add un-needed terms to the final expression.

Step 4. Write the expression that each sub cube represents. In the vertical sub cube, C and B remain constant. In the horizontal sub cube, 'B and A are constant.

Step 5. From the output expression.

$Y = BC +$ $Y = BC + A\overline{B}$



Assignment 3

Use a Karnaugh map to design a logic diagram to implement the following Truth Table.

Inputs				Output
D	**C**	**B**	**A**	**Y**
0	**0**	**0**	**0**	**1**
0	**0**	**0**	**1**	**0**
0	**0**	**1**	**0**	**1**
0	**0**	**1**	**1**	**0**
0	**1**	**0**	**0**	**0**
0	**1**	**0**	**1**	**1**
0	**1**	**1**	**0**	**0**
0	**1**	**1**	**1**	**0**
1	**0**	**0**	**0**	**1**
1	**0**	**0**	**1**	**0**
1	**0**	**1**	**0**	**1**
1	**0**	**1**	**1**	**0**
1	**1**	**0**	**0**	**1**
1	**1**	**0**	**1**	**1**
1	**1**	**1**	**0**	**1**
1	**1**	**1**	**1**	**1**

Solution

Step 1. Draw the table. Since four variables are needed, use two across the top and down the side.

$\overline{D}\,\overline{C}$	$\overline{D}\,C$	$D\,C$	
1	**0**	**1**	**1**
0	**1**	**1**	**0**
0	**0**	**1**	**0**
1	**0**	**1**	**1**

Step 2. Fill the table with 1s and 0s from the truth Table.

Step 3. Combine adjacent cells that contain 1s into subcubes of size 1, 2, 4, 8 or 16.

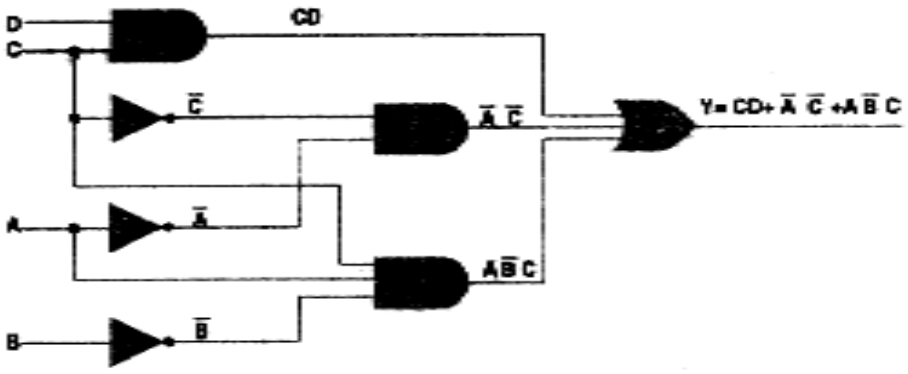
$\overline{D}\,\overline{C}$	$\overline{D}\,C$	$D\,C$	
1	**0**	**1**	**1**
0	**1**	**1**	**0**
0	**0**	**1**	**0**
1	**0**	**1**	**1**

Since the map is continuous top to bottom and side to side, the four corners are adjacent and form a sub cube of size 4. The DC column forms another sub cube of size 4. One cell remains uncovered. ‘DC‘BA forms a sub cube of size 2 with the cell on its right.

Step 4. Write the expression that each sub cube represents. The sub cube formed by the four corners represents the term ‘A‘C. The vertical sub cube represents the expression CD, and the sub cube of size 2 represents the expression A‘BC.

Step 5. From the output expression.

$$Y = CD + \overline{A}\,\overline{C} + A\overline{B}C$$



Binary/BCD Subtraction and Addition

Write working with digital equipment, one has to convert from the binary code to decimal numbers. If a binary number, say 110011 is given, what would be it equals in decimal? First write down the binary numbers as

Binary

1	1	0	0	1	1

Decimal

$1x2^5+1x2^4+0x2^3+0x2^2+1x2^1+1x2^0$

or 32+16+0+0+2+1

or 51

Shortly, it is written as

$(110011)_2=(51)_{10}$

In another example, $(101010)_2=(?)_{10}$

Binary

1	0	1	0	1	0

Decimal

$1x2^5+0x2^4+1x2^3+0x2^2+1x2^1+0x2^0$

or 32+0+8+0+2+0

or 42

Thus, $(101010)_2=(42)_{10}$

What about $(1101010.101)_2=(?)_{10}$

Binary

1	1	0	1	0	1	0	.	1	0	1

Position

6	5	4	3	2	1	0	-1	-2	-3

Decimal

$1x2^6+1x2^5+0x2^4+1x2^3+0x2^2+1x2^1+0x2^0.1x2^{-1}+0x2^{-2}+1x2^{-3}$

or $64+32+0+8+0+2+0.1/_2+0/_4+1/_8$

or $106(5/_8)$ or 106.625

Thus $(1101010.101)_2=(106.625)_{10}$

Now, let us look at the method of converting decimal number to binary number.

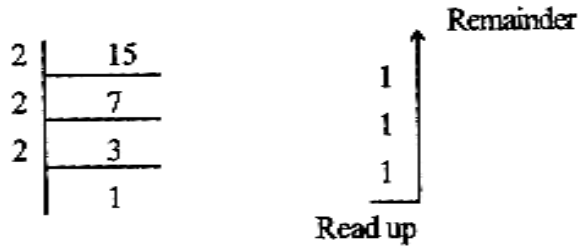
The most popular method to convert decimal number to binary number is the "DOUBLE-DABBLE" method. In this method, one progressively divides the given decimal number by 2 and writes down the remainder after each division. The remainder is read in reverse order.

However, to convert a fraction number into binary number, multiply the decimal number by 2 and record the carry in the integer position and down ward read these carries. Let us take few examples to understand the method of conversion.

Assignment 4

$(15)_{10}=(?)_2$

Solution



$(15)_{10} = (1111)_2$

Assignment 5

$(.35)_{10}=(?)_2$

Solution

0.35x2=0.70 with a carry 0

0.70x2=1.40 with a carry 1

0.40x2=0.80 with a carry 0

0.80x2=1.60 with a carry 1

0.60x2=1.20 with a carry 1

0.20x2=0.40 with a carry 0 $\rightarrow$ **Stop when the number started repeating**

$(.35)_{10}=(.010110\ldots.)_2$

Binary Addition

Binary addition is performed in the same manner as decimal Addition. However, since Binary system has only two digits, the addition table for Binary Arithmetic is very simple consisting of only four entries.

The complete table for Binary addition is as follows;

0+0=0

0+1=1

1+0=1

1+1=0 plus a carry of 1 to next higher column

Alternately, 1+1=10 (sum 0 with carry 1)

Carry-overs are performed in the same way as in decimal arithmetic.

Since 1 is the largest digit in Binary system, any sum greater than 1 requires that a digit be carried over. For instance, 10 plus 10 binary requires the addition of two 1's in the second position. Since 1+1=0 plus a carry over1, the sum of 10+10 is 100 in binary.

Assignment 6

$(1010)_2+(101)_2=(1111)_2$

Solution

Binary		Decimal
$(1010)_2$		$(10)_{10}$
$\underline{+(101)_2}$		$\underline{+(5)_{10}}$
$(1111)_2$	=	$(15)_{10}$

Assignment 7

$(1011)_2+(111)_2=(10010)_2$

Solution

Binary		Decimal
$(1011)_2$	Carry bits(1+1+1=11)	$(11)_{10}$
$+(111)_2$		$+(7)_{10}$
$(10010)_2$	=	$(18)_{10}$

Assignment 8

$(100011)_2+(11011)_2=(1000010)_2$

Solution

Binary		Decimal
1111	Carry bits	$(39)_{10}$
$(100011)_2$		$(27)_{10}$
$+(11011)_2$		
$(1000010)_2$	=	$(66)_{10}$

Binary Subtraction

The principle of binary subtractions consists of two steps. The first step is to determine if it is necessary to borrow. If the subtrahend (the lower digit) is larger than the minuend (the upper digit), it is necessary to borrow from the column to the left. It is important to note here that the value borrowed depends on the base of the number and is always the decimal equivalent of the base. Thus, in decimal, 10 is borrowed; in binary, 2 is borrowed. The second step is simply to subtract the lower value from the upper value.

The complete table for binary subtraction is as follows

0-0=0

1-0=1

1-1=0

0-1=1 with a borrow from the next column.

Alternately, 10-1=1

Note that the only case in which it is necessary to borrow is when 1 is subtracted from 0. Let us take few more examples to make the operation more clear

Assignment 9

$(10101)_2-(01110)_2=(0011)_2$

Solution

Binary		Decimal
12		
0202 borrow		
$(10101)_2$		$(21)_{10}$
$-(01110)_2$		$(14)_{10}$
$(00111)_2$	=	$(7)_{10}$

Assignment 10

$(10100)_2-(1111)_2=(00101)_2$

Solution

Binary		Decimal
01212 borrow		
$(10100)_2$		$(20)_{10}$
$-(1111)_2$		$(15)_{10}$
$(00101)_2$	=	$(5)_{10}$

Assignment 11

$(101.01)_2 - (010.11)_2 = 010.10$

Solution

Binary		Decimal
0202 borrow		
$(101.01)_2$		$(5.25)_{10}$
$-(010.11)_2$		$(2.75)_{10}$
$(010.10)_2$	=	$(2.50)_{10}$

Exercise

Truth Table

C	B	A	Y
0	**0**	**0**	**0**
0	**0**	**1**	**1**
0	**1**	**0**	**0**
0	**1**	**1**	**0**
1	**0**	**0**	**0**
1	**0**	**1**	**1**
1	**1**	**0**	**1**
1	**1**	**1**	**1**

1. Use the Boolean algebra method to develop a circuit to implement the truth table above.
2. Use the Karnaugh map method to develop a circuit to implement the truth table above.

❖ ❖ ❖

: 3

Author :
Dr. Sib Krishna Ghoshal

Vetter :
Dr. Devendra Mohan

3.1 AND, OR and NOT Gates

3.2 NAND, NOR, EXOR and EXNOR Gates

3.3 Application : Logic Implementation using Gates

3.4 Problems and solutions

Introduction

The term "Logic" is generally used to refer a decision-making process. A logic gate, then is a circuit that can decide to say Yes or No at the output based upon the inputs. Gates are circuits that are used to combine digital logic levels (ones and zeros0 in specific ways. The basic building block of any digital circuit is a **logic gate**. A system called **Boolean Algebra** and the corresponding *tabular* representation called Truth Table is used to express the output in terms of the inputs.

Gate is a digital circuit with one or more input signals but with only one output signal. Gate is an electronic switching circuit, which allows passing of the applied input signal under certain specified logical conditions.

The basic gates are AND, OR and NOT gates. An AND gate, universal logic gates NAND and NOR are made. The NAND gate is a NOT gate followed by an AND gate. Similarly, a NOR gate is a NOT gate followed by a OR gate. There are two other basic logic gates called Exclusive OR gate (EXOR) and Exclusive NOR gate (EXNOR).

3.1 AND, OR and NOT Gates

AND Gate

The AND gate is sometimes called the "All or nothing gate". It has N inputs ($N^3 2$) and one output. It generates an output signal of 1 only if all input signals are

also1. These basic idea of the AND gate can be realised using simple switches called electrical analog circuit of logical digital AND gate.

Figure 3.1 AND Gate Circuit using Switches

Here two switches A and B are connected in series. One must close both switches A and N to get the lamp to light. There will be no output (that is the bulb will not glow) if either one or both switches are in the OFF (zero) state. So, two or more switches connected in series behave as an AND gate. The AND gates are constructed of diodes and transistors and package inside an IC. The logic symbol of AND gate is shown in Figure 3.2 and the corresponding Truth Table is in Table 3.1.

Inputs				Output	
B		A		Y	
Switch Voltage	Binary	Switch Voltage	Binary	Light	Binary
Low	0	Low	0	No	0
Low	0	High	1	No	0
High	1	Low	0	No	0
High	1	High	1	Yes	1

Table 3.1 : AND Gate Truth Table

The Boolean expression for a two-input AND gate, with input A and B and output Y is written as

Y=A.B

Which is read “A AND B”. The output Y is one only when both A and B are ones. All possible input combinations are listed in the Truth table by counting in binary from 0.0 to 1.1.

The practical AND gate circuit is shown in Figure 3.3.

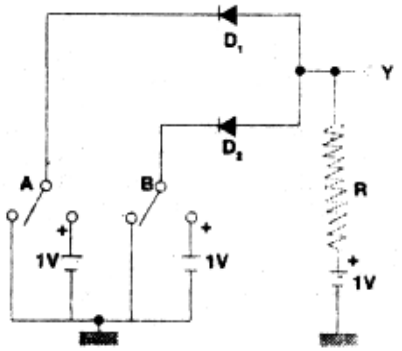


Figure 3.3 AND Gate using Diodes

A and B represent the two inputs and Y the output of the AND gate. Suppose that the diodes are ideal and that the two input voltages are either 0 or 1 volt.

Then four cases may arise

i) Both A= 0V, and B=0V. Under this condition, both the inputs are short-circuited to ground. The 1V battery in the output side, however, biases the two diodes in the forward direction. Hence the two diodes are on, i.e., shorted. The output is also shorted to ground through the diodes. Thus the output Y=0V.

ii) A=0V, B=1V. In this the upper diode conducts and the output is short-circuited to ground through this diode. Thus Y=0V.

iii) A=1V, B=0V. In this case, the output is short-circuited to ground through the lower diode and Y=0V.

iv) Both A=1V and B=1V. Under this condition, none of diodes conduct. Hence no current flows through R. The output is thus held at 1V. Therefore, Y=1V

OR Gate

The OR gate is sometimes called the "any or all gate". This gate is the physical realization of the logical addition operation. It has N inputs (N^32) and one output. The output of an OR gate is 1 only if one or more inputs are 1.The basic idea can be illustrated using simple switches called electrical analog of digital OR operation (Figure 3.4). The Figure shows that the output lamp will light when either or both of the input switches are closed but not when both open.

Inputs				Output	
B		A		Y	
Switch Voltage	Binary	Switch Voltage	Binary	Light	Binary
Low	0	Low	0	No	0
Low	0	High	1	No	0
High	1	Low	0	No	0
High	1	High	1	Yes	1

Table 3.2: OR Gate Truth Table

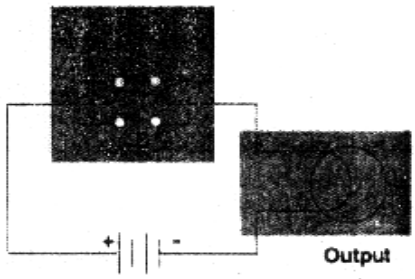


Figure 3.4 (a) OR Circuit using Switches

The logic symbol (Boolean expression) for the two input OR gates can be represented as:

Y=A+B

Which is read “A OR B”. The Truth Table (Table 3.2) is said to describe **inclusion OR function**.

The output Y is 1 when A is 1 or B is 1 or both. The practical OR gate circuit is shown in Figure 3.5.

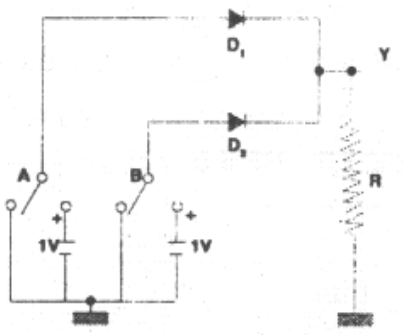


Figure 3.5 OR Gate using Diodes

A and B represent the two inputs and Y the output of the OR gate. The resistor R is the output load resistor. Suppose that the diodes are ideal and that the two input voltages are either 0 or 1V.

Thus, there are four possibilities in which the input voltages may appear. These are,

i) Both A=0V, and B=0V. Under this condition, none of the diodes conduct and the outputY-0V.

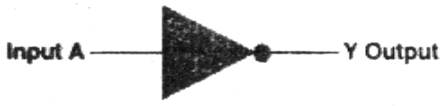
ii) A=0V, B = 1V. Under this condition, the lower diode is forward biased and hence it conducts. The whole of the input voltage appears at the output, since the diode forward resistance is assumed to be negligible. Therefore, Y=1V.

iii) A=1V or B=0V. In this case, the upper diode conducts and the output Y=1V

iv) Both A=1V and B=1V. Under this condition bothdiodes are forward-biased and they conduct. The output is held at Y=1V since the voltages are in parallel.

NOT Gate

The NOT gate is the physical realisation of the complementation operation. This is, an electronic circuit that generates an output signal which is the reverse or inverse of the input signal. A NOT gate is also known as inverter because it inverts the input. The NOT circuit, however, has only one input and one output. The logic symbol for the inverter (NOT gate) is shown in Figure 3.6(a).



Inputs		Output	
A		Y	
Voltage	Binary	Voltage	Binary
Low	0	High	1
High	1	Low	0

Table 3.3 Truth Table for an Inverter

The Boolean expression for NOT gate is written as

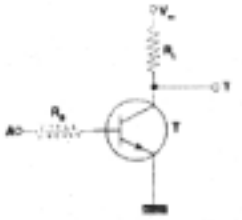
Y = NOT A

= That read as “Y equals not A” or “Y equals complement of A”

The Truth Table is given in Table 3.3

The small circle on the output of the symbol is called a bubble. The bubble on the output indicates that the output is **Active Low,** and the absence of a bubble in the input indicates that the input is active high. The input is "looking for" a 1 level to produce a 0, **Active Low** output.

Therefore, one can say that when a signal passes through an inverter or complimented. We can also say it is **negated**. The terms "**negated**". "**Complemented**", and "**inverted**", then used in the same connotations. The practical NOT gate circuit is shown in Figure 3.6 (b).



If the input voltage is high enough to saturate the transistor, the output is held at a low value. On the other hand, if the input voltage is low enough, the transistor becomes cut-off and the output is high.

3.1 NAND, NOR, EXOR and EXNOR Gates

NAND Gate

A NAND gate is a complemented AND gate. The NOT-AND operation is known as the NAND operation. It is an AND gate followed by a NOT gate. It has N-inputs (N³2) and one output. The output of NAND gate will be a 1 if any one of the input is 0 and will be 0 only when all the inputs are 1. The operation of this circuit can be described by the Boolean expression,

$$Y = \overline{A} \cdot \overline{B}$$

$$= \overline{A} + \overline{B} \text{ (using De Morgan's law).}$$

The expression is read as "Y equals NOT) A AND B)"

The Truth Table and the circuit symbol is projected below in Table 3.4 and Figure 3.7 respectively.

The little bubble (called invert bubble) on the right end of the symbol means to invert the AND.

Inputs		Output	
B	A	AND	NAND
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Table 3.4 : Truth Table for AND and NAND Gate

NOR Gate

A NOR gate is a complemented OR gate. It is an OR gate followed by a NOT gate. The NOR gate is actually a NOT-OR operation. In other words, the output of a OR gate is inverted to form a NOR gate. Figure 3.8 shows an 2 input (N=2) OR gate followed by a NOT gate, that is NOR gate. The Boolean expression for NOR operation is given by.

$Y + \overline{A + B}$ = A .B (using De Morgan's Law)

The expression is read as "Y equals NOT (A OR B)"

Inputs		Output	
B	A	OR	NOR
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

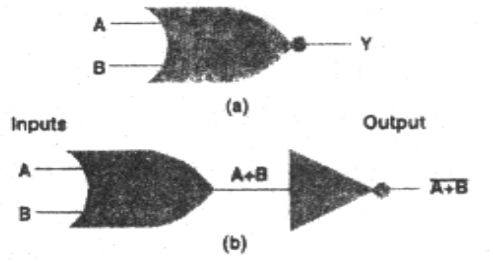


Table 3.5: Truth Table for OR and NOR Gate

The Truth Table of a Two-input NOR gate is shown in Table 3.5. The output of a NOR gate will be a 1 only when all inputs are 0 and it will be a 0 if any input represents a 1.

The logic symbol of a two input NOR gate circuit is shown in Figure 3.8 (a).

EX-OR Gate

The EXCLUSIVE-OR (EX-OR or XOR) gate is sometimes referred to as the

"Any but not all gate". It is not a basic operation and can be performed using the basic gates-AND, OR and NOR or Universal gates-NAND or NOR.

The logic symbol for the two-input XOR gate and its EQUIVALENCE are diagrammed in Figures 3.9 and 3.10 respectively.

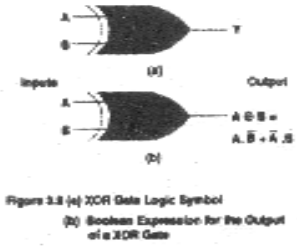


Figure 3.9 (a) XOR Gate Logic Symbol
(b) Boolean Expression for the Output of a XOR Gate

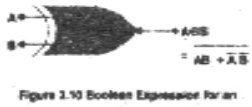


Figure 3.10 Boolean Expression for an Equivalence Operation

The Boolean expression for the XOR function is

and it is EQUIVALENCE is denoted by

$Y = A.B$

$Y = A \oplus B$

$A.\overline{B} + \overline{A}.B$

The Truth Table for the XOR and its equivalence operations are represented in Table 3.6.

Inputs				Output	
A	$\overline{A}$	B			
0	1	0	1	0	0
0	1	1	0	1	1
1	0	0	1	1	1
1	0	1	0	0	0

Table 3.6: Truth Table for OR and XOR Gate

EX-NOR Gate

The EXCLUSIVE-NOR gate is often shortened to EX-NOR or XNOR gate. The two-input EXNOR gate in shown in Figure 3.11. This is the EX-OR symbol with the added invert bubble on the output side.

The Boolean expression for the EX-NOR function is

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

$$= \overline{(\overline{A}.B)(A\overline{B})}$$

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

$$= \overline{A}.\overline{B} + A\,B$$

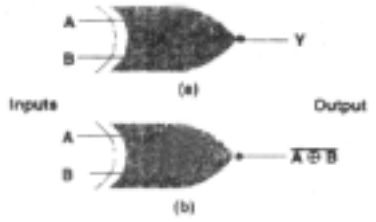


Figure 3.11 (a) XNOR Gate Logic Symbol

Figure 3.11 (b) Boolean Expression for the Output of a XNOR Gate

The bar over $A \oplus B$ expression tells us we have inverted the output of XOR gate. The Truth Table is examined in Table 3.7. Notice that the output of the XNOR gate is the complement of the XOR Truth Table.

Inputs		Output	
B	A	XOR	XNOR
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

$Y = \overline{A \oplus B}$

Table 3.7: Truth Table for XOR and XNOR Gate

CMOS Gate

The CMOS (Complementary Metal Oxide Semiconductor) is actually a complementary metal oxide Semiconductor Field Effect Transistor (MOSFET) which is obtained by connecting a p-channel and an n-channel MOSFET in series. The circuit symbol for FET and MOSFET is shown in Figure 3.12(a) and 3.12(b) respectively. The drains are tied together and the output is taken as common drain point as shown in the Figure 3.12(c). Input is applied at the common gate connection formed by connecting the two gates together. In this circuit, when $V_1 = V_{cc}$, T_1 turns ON ($V_{gs1} > V_t$)

and T_2 is OFF since V_{gs2}=0 volt. Therefore, the quiescent power dissipation, which is the product of the OFF leakage current and V_{cc} is very small. CMOS have extremely low power dissipation and is very useful for remote applications where power is expensive. It has high noise immunity, large Fan-out, full power supply, logic swings etc,

The basic gates can be realised using CMOS ICs as described below

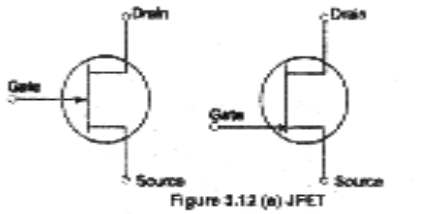


Figure 3.12 (a) JFET

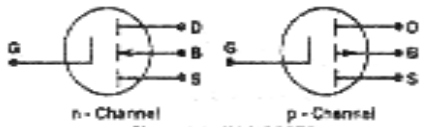


Figure 3.12 (b) MOSFET

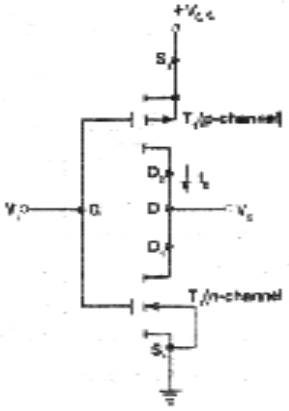


Figure 3.12 (c) A CMOS Switch

CMOS Inverter

Figure 3.13 shows the circuit diagram for CMOS inverter. If low (0V) is applied to the input, Q_1 which is an n-type MOSFET is OFF and Q_2 which is p-type MOSFET is O. Since MOS transistors consumes negligible power the output Y is almost equal to V_{DD} (or HIGH).

When the inverter input is high i.e. at $+V_{DD}$ then Q_1 (n-type MOS transistor) is ON and Q_2 (p-type MOS transistor) is OFF. The output point Y is connected to the ground point, thus, the output is LOW (or 0V).

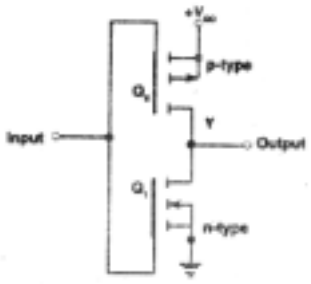


Figure 3.13 CMOS Inverter

CMOS NAND

The Figure 3.14(a) shows the circuit of CMOS NAND, It can be seen that p-type Q_1 and n-type Q_3 MOS transistors forms one complementary pair, p-type Q_2 and n-type Q_4 forms another. A LOW at the gate input turns n-type MOS transistor OFF (switch open) and p-type MOS transistor ON (switch closed).

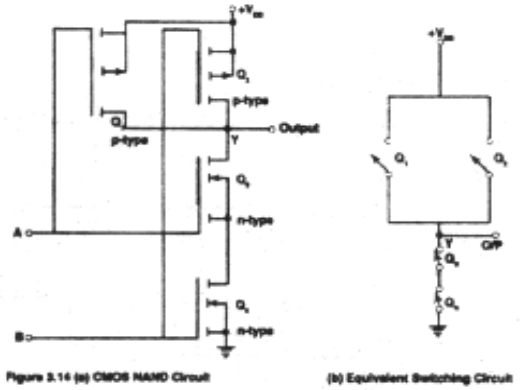


Figure 3.14 (a) CMOS NAND Circuit (b) Equivalent Switching Circuit

Keeping these things in mind, take up the input conditions as if A=B=0 then Q1 and Q2 will be ON and Q3 and Q4 OFF, pulls the VDD to the output point Y and thus getting high output.

If A=0 and B=1Q1 and Q4 are on and Q2 and Q3 are OFF thus point Y is at high potential.

If A=0 and B=1Q2 and Q3 are on and Q1 and Q4 are OFF thus point Y is at high potential.

If A=B=1 then Q1 and Q2 are OFF and Q3 and Q4 are ON, the output terminal Y is connected to ground terminal thus getting low output.

Truth Table		
B	A	Y
0	0	$+V_{DD}$
0	1	$+V_{DD}$
1	0	$+V_{DD}$
1	1	**O**

CMOS NOR

I A=B=0, then Q1 and Q2 are ON and Q3 and Q4 are OFF thus connecting the output point Y to V_{DD} (high).

If A=0 and B=1, in this condition Q1 and Q3 are ON and Q2 and Q4 are OFF, thus connecting the output terminal to ground which is at 0v.

If A=1 and B=0 Q1 and Q3 are OFF and Q2 and Q4 areON, connecting the output terminal Y to ground.

If A=B=1, Q1, Q2 are OFF and Q3 and Q4 are ON thus connecting the output terminal to ground.

Truth Table		
B	A	Y
0	0	**$+V_{DD}$**
0	1	**0**
1	0	**0**
1	1	**0**

Table 3.9: Truth Table for CMOS NOR Gate

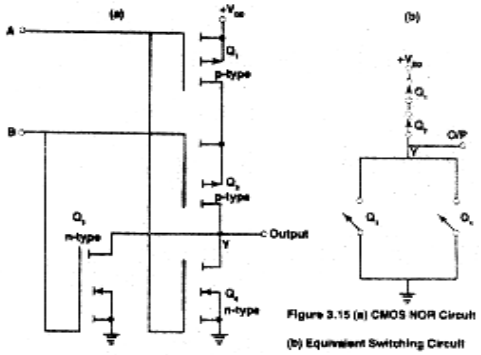


Figure 3.15 (a) CMOS NOR Circuit
(b) Equivalent Switching Circuit

Applications: Logic Implementations using Gates

Any Boolean (or logic) expression can be realised by using the AND, OR and NOT gate. These three gates are called the basic gates. However, the NAND and NOR gate, is said to be universal gate because any of them alone is sufficient to implement any Boolean function. Because of this reason NAND and NOR gates are known as Universal gates.

Realization of NOT, AND, and OR Gate using NAND Gate(s)

The three basic logic operation NOT, AND, and OR can be performed by using NAND gates. This is shown in Figure 3.16.

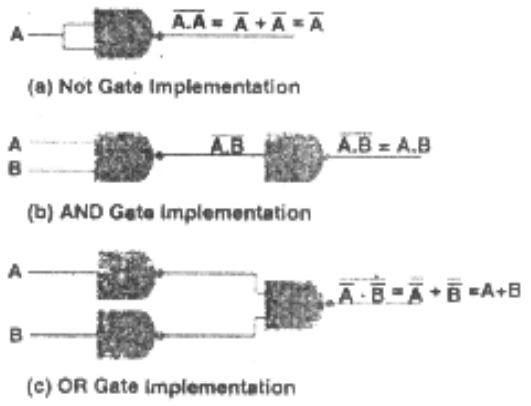


Figure 3.16 Implementation of NOT, AND, and OR Gates by NAND Gate(s)

A NOT operation is obtained from a 1-input NAND gate. Thus, we find that a single input NAND gate behaves as an inverter. The AND operation requires two NAND gates. The first one produces the inverted AND and the second one acts as an inverter to obtain the normal AND output.

For the OR operation the normal inputs A and B are first complemented using two single input NAND gates. Now, the complemented variables are fed as input to another NAND gate, which produces the normal OR output.

Realization of NOT, AND and OR Gate using NOR Gate(s)

The NOR function is the dual of the NAND function. For this reason, all procedures and rules for NOR logic form a dual of the corresponding procedures and rules developed from NAND logic. The logical operational NOT, AND and OR gates can be implemented solely with NOR gates as shown below in Figure 3.17

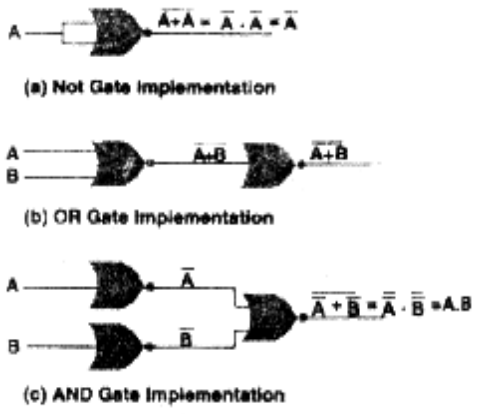


Figure 3.17 Implementation of NOT, OR, and AND Gates by NOR Gate(s)

The NOR operation is obtained from one-input NOR gate. Thus a single-input NOR gate is yet another inverter circuit. The OR operation requires two NOR gate. The first one produces the inverted OR and the second one being a single input NOT gate, acts as an inverter to obtain the normal OR output. The AND operation is achieved through a NOR gate with additional inverters in each input.

Realization of XOR and XBIR Gate using Basic Gate(s)

Figure 3.18 (b) and 3.19(b) realizes the implementation of XOR and XNOR gate using three basic gates. Figure 3.18(a) and 3.19(a) are the symbols for XOR and XNOR.

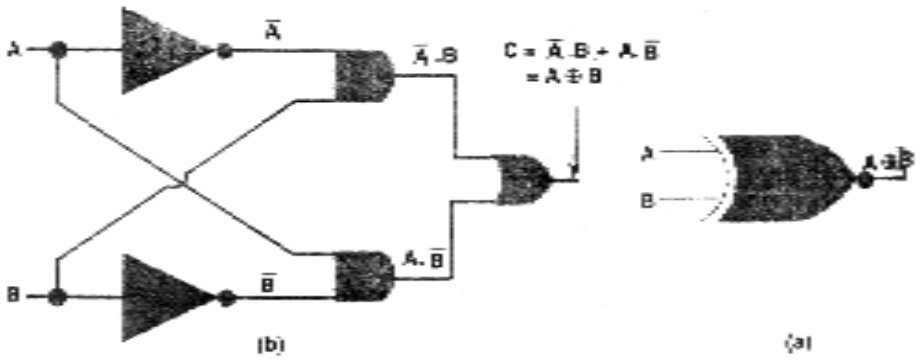


Figure 3.18 (b) Implementation of XOR Function with AND/OR/NOT Gate(s) (a) XOR Gate

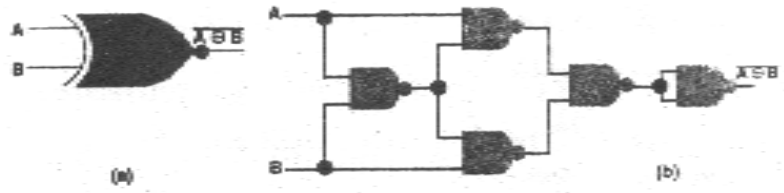
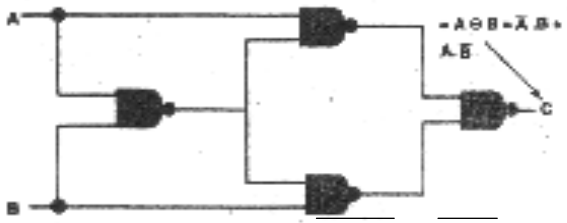


Figure 3.19 (a) XNOR Gate (b) Implementation of XNOR using Three Basic Gate(s)

Realization of XOR and XNOR Gate using NAND and NOR Gates

XOR and XNOR implementation is possible through universal NAND and NOR gates. Figure 3.20 and 3.21 shown the realization of XOR and XNOR through NAND gate. Similarly, XNOR gate can be represented by using NOR gates.



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

Figure 3.20 Implementation of XOR Function with NAND Gate(s)

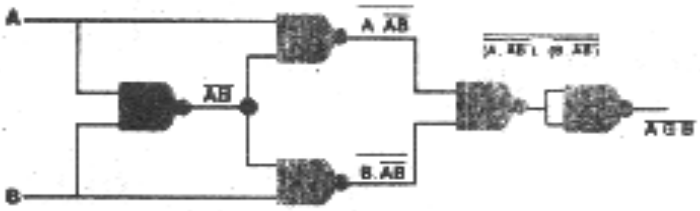


Figure 3.21 Implementation of XNOR Function with NAND Gate(s)

Realization of De Morgan's Theorem through Basic Gates

Theorem1:

Theorem2: $\overline{A + B} = \overline{A}.\overline{B}$

Assignment 1

Prove the following rules and draw their respective circuits

a) AC+ABC=AC

b) A+AB=A

c) ABC+ABC+ABC=A(B+C)

d) A+A=1

e) (A+B)(A+C)=AC+AB

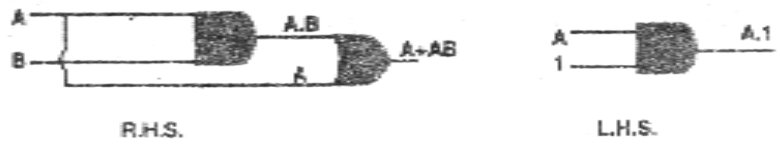
f) AB+AC+BC=AC+BC

g) ABC+ABC+ABC=C(A+B)

Solutions

a) AC+ABC=AC(1+B)=AC, since 1+B=1

b) A+AB=A(1+B)=A $(C + B\overline{C} = C + B)$

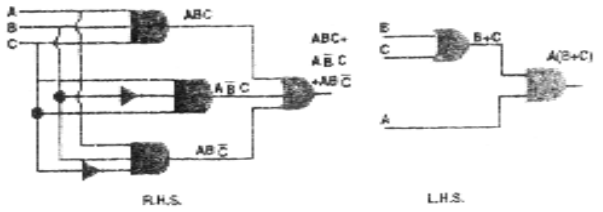


c) $ABC + A\overline{B}C + AB\overline{C} = AC(B + \overline{B}) + AB\overline{C}$

$= AC + AB\overline{C} \qquad (B + \overline{B} = 1)$

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

$= A + (B + C)$

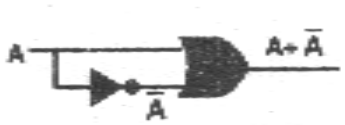


d) A can have only two values, 0 and 1.

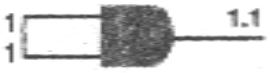
When A=0, A+A=0+0=0+1=1

When A=1, A+A=1+1=1+0=0

Hence A+A=1 for all possible values of A.



R.H.S.



L.H.S.

e) $A\overline{A} + AC + B\overline{A} + BC = 0 + AC + B\overline{A} + BC(A + \overline{A})$

$= AC(B+1) + B\overline{A}(C+1) = AC + \overline{A}B$

Finding the truth tables for the left and right hand expressions as shown in following table can prove this identity. From the table it is seen that for all possible values of A, B and C

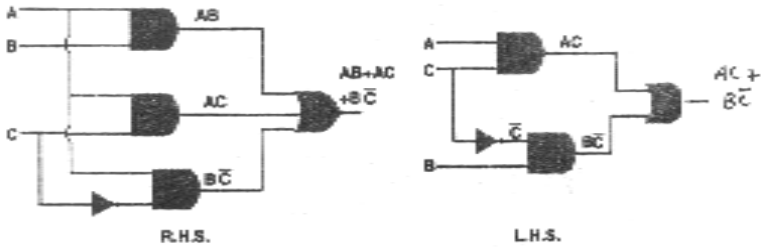
$(A + B)(\overline{A} + C) = AC + \overline{A}B$

A	*B*	*C*	$\overline{A}$	*A+B*	$\overline{A}+C$	*(A+B)*	*A.C*	$A\overline{B}$	$AC+\overline{A}$
0	0	0	1	0	1	0	0	0	0
0	0	1	1	0	1	0	0	0	0
0	1	0	1	1	1	1	0	1	1
0	1	1	1	1	1	1	0	1	1
1	0	0	0	1	0	0	0	0	0
1	0	1	0	1	1	1	1	0	1
1	1	0	0	1	0	0	0	0	0
1	1	1	0	1	1	1	1	0	1

f) $AB + AC + B\overline{C} = AB(C + \overline{C}) + AC + B\overline{C} = ABC + AB\overline{C} + AC + B\overline{C}$

$= AC(B + 1) + B\,\overline{C}\,(A + 1) + AC + B\overline{C}$

This identity can be proved by finding the Truth Tables for left and right hand expressions.



g) $\overline{ABC} + \overline{A}B\overline{C} + A\overline{BC}$

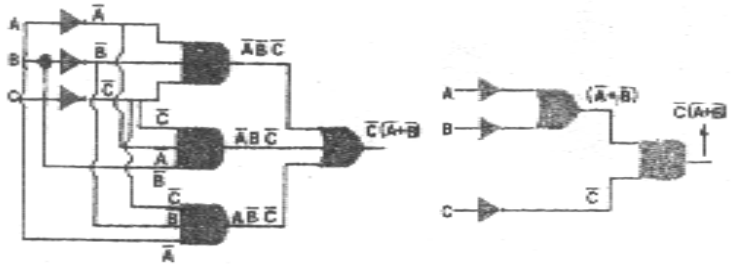
$= \overline{A}.\overline{C}(B + \overline{B}) + A\overline{BC}$

$= \overline{A}.\overline{C}(1) + A\overline{BC}$

$= \overline{AC} + A\overline{BC}$

$= \overline{C}(\overline{A} + AB)$

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

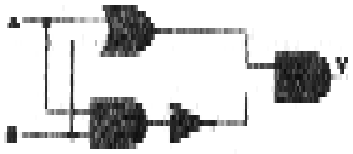


Assignment 2

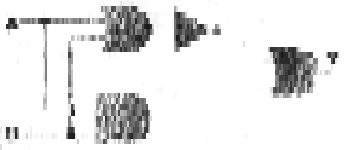
What will be the outputs for the following logic circuits?

What will be the outputs for the following logic circuits?

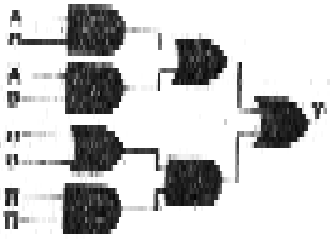
(a)



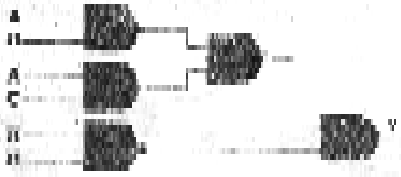
(b)



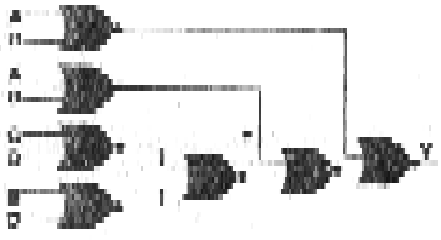
(c)



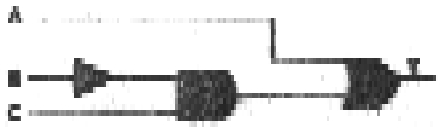
(d)



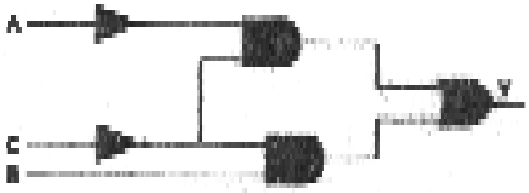
(e)



(f)

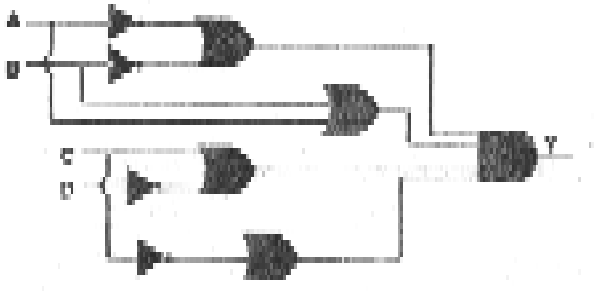


(g)

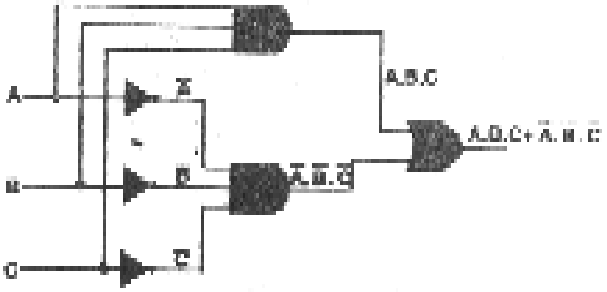


(h)

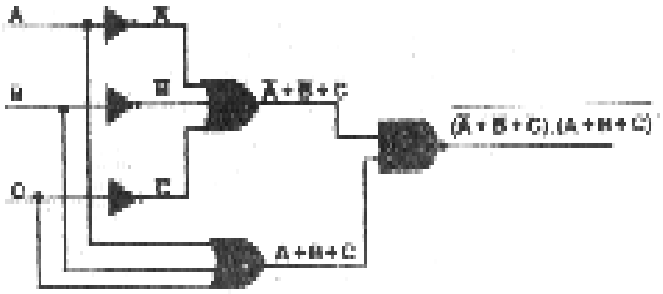
(i)



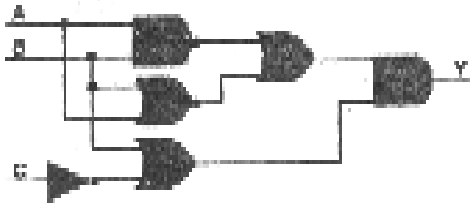
(j)



(k)



(l)



Solutions

(a) $Y = (A + B)(\overline{A}B)$

(b) $Y = (\overline{A + B}) + (A.B)$

(c) $Y = (AC + \overline{A}B) + C.\overline{B}.\overline{D}$

(d) $Y = \overline{C} + \overline{A}B$

(e) $Y = A\overline{B}C = A\overline{B}D + AC\overline{D} + BC\overline{D} + A\overline{B}$

(f) $Y = A + \overline{B}C$

(g) $Y = \overline{C}(\overline{A} + B)$

(h) $Y = (\overline{A} + B)\overline{C}$

(i) $Y = \overline{A}BCD + A\overline{B}CD + A\overline{B}\,\overline{C}\,\overline{D} + \overline{A}B\overline{C}\,\overline{D}$

(j) $Y = ABC + \overline{A}\,\overline{B}\,\overline{C}$

(k) $Y = (\overline{A} + \overline{B} + \overline{C})(A + B + C)$

(l) $\overline{A}B + \overline{A}\,\overline{C} + \overline{B}\,\overline{C}$

Assignment 3

(a) Design a circuit that will implement the Truth Table given below

Solution

Input		Output
A	**B**	**Y**
0	**0**	**0**
0	**1**	**1**
1	**0**	**1**
1	**1**	**1**

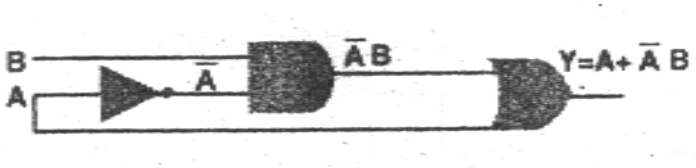
First line **A=0** **B=0** $Y = 0 \Rightarrow \overline{A}.B$

Second line **A=0** **B=1** $Y = 1 \Rightarrow \overline{A}.B$

Third line **A=1** **B=0** $Y = 1 \Rightarrow A + \overline{A}.B$

Fourth line **A=1** **B=1** $Y = 1 \Rightarrow A + \overline{A}B$

Therefore, $Y = A \Rightarrow \overline{A}B$



(b)

Solution

Inputs		Output	
A	**B**	**C**	**Y**
0	**0**	**0**	**0**
1	**0**	**0**	**0**
0	**1**	**0**	**0**
1	**1**	**0**	**1**
0	**0**	**1**	**0**
1	**0**	**1**	**0**
0	**1**	**1**	**1**
1	**1**	**1**	**1**

First line **A=0** **B=0** **C=0** $Y = 0 \Rightarrow A.B.C$

Second line **A=1** **B=0** **C=0**

Third line **A=0** **B=1** **C=0**

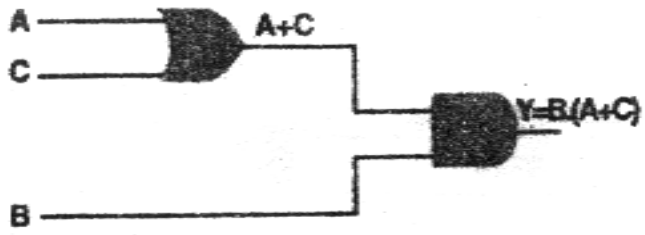
Fourth line **A=1** **B=1** **C=0**

Fifth and Sixth line $\Rightarrow$ **ABC**

Seventh line **A=0** **B=1** **C=1**

Eighth line **A=1** **B=1** **C=1** $Y = 1 \Rightarrow ABC$

Therefore



(c)

Solution

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

Inputs			Output
A	B	C	Y
0	0	0	1
1	0	0	1
0	1	0	0
1	1	0	1
0	0	1	1
1	0	1	0
0	1	1	1
1	1	1	1

First line = $\overline{A}\,\overline{B}\,\overline{C}$

Second line = $A\,\overline{B}\,\overline{C}$

Third line = **ABC**

Fourth line =

Fifth = $\overline{A}\,\overline{B}C$

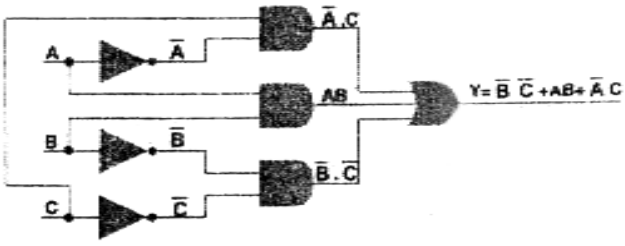
Sixth line = **ABC**

Seventh line =

Eighth line =

Therefore,

Y =



(d)

Solution $\overline{A}\overline{B}\overline{C} + A\,\overline{B}\,\overline{C} + ABC + AB\overline{C} + \overline{A}\,\overline{B}\,C + ABC + \overline{A}BC + AB$

Inputs			Output
A	**B**	**C**	**Y**
0	**0**	**0**	**0**
1	**0**	**0**	**1**
0	**1**	**0**	**0**
1	**1**	**0**	**0**
0	**0**	**1**	**1**
1	**0**	**1**	**1**
0	**1**	**1**	**1**
1	**1**	**1**	**0**

First line = **ABC**

Second line =

Third line = $A\,B\,C$

Fourth line =

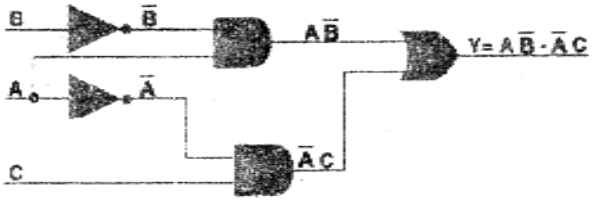
Fifth =

Sixth line = $A\,\overline{B}\,C$

Seventh line = $\overline{A}\,B\,C$

Eighth line = $\overline{A}\,\overline{B}\,\overline{C}$

Therefore, $Y = A\overline{B} + \overline{A}C$



(e)

Solution

Inputs		Output $A\,B\,\overline{C}$	
A	**B**	**C**	**Y**
0	0	0	0
1	0	0	1
0	1	0	0
1	1	0	0
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	0

First line = $A\,B\,C$

Second line =

Third line =

Fourth line = $A\,B\,\overline{C}$

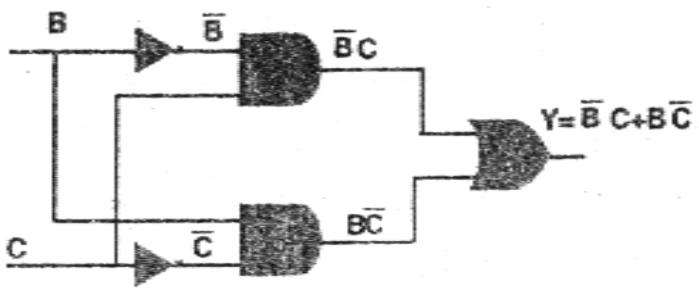
Fifth = $\overline{A}\,\overline{B}\,C$

Sixth line = $A\,\overline{B}\,C$

Seventh line = $B\,C$

Eighth line =

Therefore, Y = $B\,\overline{C} + \overline{B}\,C$



(f)

Solution

Inputs			Output
A	**B**	**C**	**Y**
0	**0**	**0**	**1**
0	**0**	**1**	**1**
0	**1**	**0**	**0**
0	**1**	**1**	**1**
1	**0**	**0**	**1**
1	**0**	**1**	**0**
1	**1**	**0**	**1**
1	**1**	**1**	**1**

A B C

Try yourself and get $Y = \overline{B}\,\overline{C} + \overline{A}\,C$.

❖ ❖ ❖

: 4

Author :
Dr. Sib Krishna Ghoshal

Vetter :
Dr. Devendra Mohan

- **Characteristics: Fan-in, Fan-out, Propagation Delay, Noise Margin, Power Dissipation, Speed of Operation, Figure of Merit and Operating Temperature**
- **Logic Families: RTL, DTL, TTL and MOS Families**
- **Comparison of Logic Families**
- **Tristate Logic/Buffers**

Introduction

An Integrated circuit is a tiny electrical device that built into a single package and performs a complete circuit function. It replaces a given number of transistors, diodes, resistors and capacitors that would be needed to perform the equivalent function.

Basically, there are two types of semiconductor devices

1. Bipolar (Two kind of charge carriers are involved, e. g., transistors)
2. Unipolar (One type of charge carrier is involved, e. g., JFET and MOSFET)

Based on these devices, digital integrated circuits have been fabricated on a single chip of silicon crystal, which are commercially available.

ICs are broadly classified into two categories

1. Linear/Analog ICs (e. g. Operational Amplifier shortly called OP AMP, whose symbolic diagram is shown in the beginning)

2. Digital ICs: this is further classified into two classes depending on their operation, and they are

a) Combinational digital IC (logic gates are examples of this class)

b) Sequential digital IC (Flip flops, multivibrator are in this category)

A group of compatible ICs (Using Bipolar and Unipolar technologies) with the same logic levels and supply voltages for performing various logic functions have been fabricated using a specific circuit configuration which is referred to as a "logic family". Each logic family uses different logic circuits, any two logic families are generally not compatible with each other. In order to use these digital ICs for a specific purpose, it is necessary to be familiar with the operational characteristics of IC logic families and their relative advantage and disadvantage. The various characteristics of digital ICs are used to compare their performances and the performance is very much dependent on the number of components fabricated on the chip (sometime called Scale of Integration see Table 4.1)

IC Classification	*Equivalent Basic Gates*	*Individual Number of Components*
Small–Scale Integration (SSI)	Less than 12	Up to 99
Medium-Scale Integration (MSI)	12-99	100-999
Large-Scale Integration (LSI)	100-999	1,000-9,999
Very Large-Scale Integration(VLSI)	1,000-9,999	10,000-99,990
Ultra Large-Scale Integration(ULSI)	10,000 or more	100,000 and above

Table 4.1: Classification of Digital ICs

Characteristics:

- **Fan-in and Fan-out**
- **Propagation Delay**
- **Noise Margin**
- **Power Dissipation**
- **Speed of Operation**
- **Figure of Merit**
- **Operating Temperature**

These are some commonly accepted performance characteristic that IC manufacturers supply for their products.

These parameters are as follows

- **Fan-In**

It is the number of inputs connected to a gate which measures how much load can be connected to the input of a gate. Thus, if a logic element is said to have a fan-in 10, it means that 10 logic elements can be connected to its input.

- **Fan-Out**

This refers to the maximum number of loads connected to the output of the gate. A logic element having fan-out 20 means 20 logic elements can be connected to its output. High Fan-out is advantageous because it reduces the need for additional drivers to drive more gates.

- **Propagation Delay**

The propagation delay is a measure of how rapidly a change of logic level at the input of a gate or flip-flop appears as a corresponding change at its output. It is the time difference between the application of a signal to a logic gate and its appearance at the output. The delay is measured in nano second (10^{-9} sec).

- **Noise Margin**

The noise margin sometimes called noise-immunity, is a measure of how much noise a logic signal can have superimposed on it, before the noise causes a gate to change its output incorrectly. It is the amount of voltage of extraneous signal, which can be tolerated without any deviation at the output. Noise-immunity is usually specified in mV (10^{-3} volts).

- **Power Dissipation**

This is the amount of power dissipated in an IC, which is a measure of how much power a gate uses and how much heat it generates. This power is specified in milli Watts. It is determined by the current (I), that it draws from the supply voltage (V) and it given by VxI.

- **Speed of Operation**

Speed of a digital circuit, is specified in terms of the propagation delay time. The delay times are measured between the 50% voltage level of input and output waveforms. If the output make a transition from the low state to high state, the delay time is t_{PHL} and for the reverse transition, (low to high), it is denoted by t_{PHL}. The average as shown in Figure 4.1 of these two-delay time is a measure of propagation delay time. Lower the difference, higher is the speed.

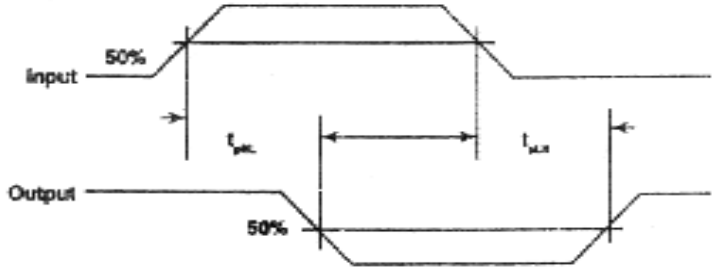


Figure 4.1 Input and Output Voltage Waveforms showing Propagation Delay Times

- **Figure of Merit**

The Figure of Merit of a digital IC is defined as the product of speed and power. The speed is expressed in terms of $t_P = (t_{PHL} + t_{PHL})/2$ (in nanosecond) and the power is expressed in milli Watt, so that the Figure of merit is specified in picojoules (10^{-12} J). A low value of speed-power product is desirable.

- **Operating Temperature**

The temperature range in which an IC functions properly is known as Operating Temperature. For consumer and industrial applications, the accepted temperature range are 0 to $+70^0$C and for military purposes, this range is -55^0C to 125^0C.

Logic Families

Logic families are broadly classified into two categories

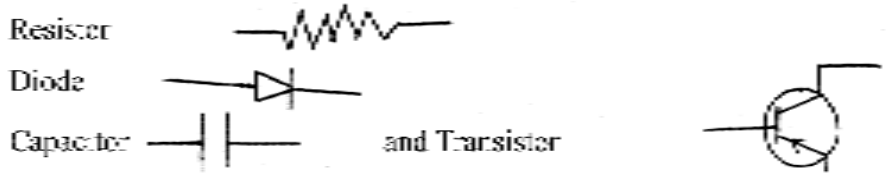
1. Bipolar Logic Families.
2. Unipolar Logic Families

Bipolar logic families are further classified into two types depending upon their operation and they are

a) Saturated Bipolar Logic Family.

b) Unsaturated Bipolar Logic Family.

The elements of a Bipolar Logic Family are



In Saturated Logic, the transistors in the IC are driven to saturation. The Saturated Bipolar Logic Families are

- Resistor-Transistor Logic (RTL)
- Diode-Transistor Logic (DTL)

- Transistor-Transistor Logic(TTL)
- High-Threshold Logic (HTL)
- Integrated-Injection Logic (I^2L)
- Direct-Coupled-Transistor Logic (DCTL)

The Non-Saturated Bipolar Logic Families are

- Emitter-Coupled Logic (ECL)
- Schottky Transistor-Transistor Logic (STTL)

MOS based devices are unipolar device in which the Metal Oxide Semiconductor Field Effect Transistors (MOSFETs) are employed to build-up MOS logic families.

There are three types of MOS logic families

- PMOS (p-Channel MOSFETs, Figure 3.12(b))
- NMOS (n-Channel MOSFETs)
- CMOS (Complementary MOS, both p-channel and n-channel MOSFETs)

Resistor Transistor Logic (RTL)

RTL logic is the first family of logic circuit, it offers high performance but low noise margin. The basic circuits of RTL family of NOR gate as shown in Figure 4.2.

- **Operation**

When inputs are low, neither of the transistors is conducting and the output at Y is high. When either or both the inputs are high, the respective transistors will conduct and output at Y is low.

- **Parameters**

Parameters of the RTL logic circuit are

Fan Out	:	5
Noise Margin	:	0.2 Volts
Propagation Delay	:	12 nano seconds
Power Dissipation	:	12 mW/gate
Power Supply Voltage	:	3.8 Volts

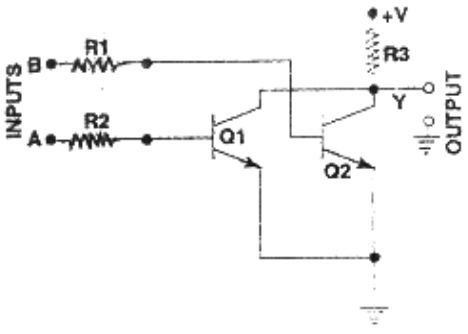


Figure 4.2 RTL NOR Circuit

- **Advantages**

  RTL has only advantage that it has low power dissipation.

- **Disadvantages**

  It has poor noise immunity.

  It is relatively slow in speed.

Diode Transistor Logic (DTL)

The next family, which was introduced after RTL, is DTL, which have better noise margin though slow in speed. The basic DTL gates are AND, OR and NAND.DTL AND circuit is shown in Figure 4.3.

- **Operation**

If any of one both inputs are at logic 0, That diode will be forward biased through resistor R_1, which will make the voltage at point X equal to zero and the transistor goes in cut-off state. No collector current results to low output (logic 0) at output point Y.

If all inputs are at logic 1(+5V), the diodes are reverse (+5V) biased, the transistor conducts heavily to saturation and output becomes high (logic1).

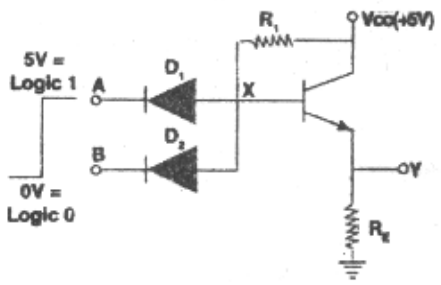


Figure 4.3 DTL AND Circuit

DTL OR Gate

DTL OR Circuit is shown in Figure 4.4. If any or all input is at logic 1(+5V)

that diode conducts and point X is clamped to +5V which saturates the transistor and the output is high (at logic1)

If all inputs are at ground (logic 0), the diodes remain reverse biased and voltage at point X is zero which drive the transistor in cutoff region and the output is low (at logic 0).

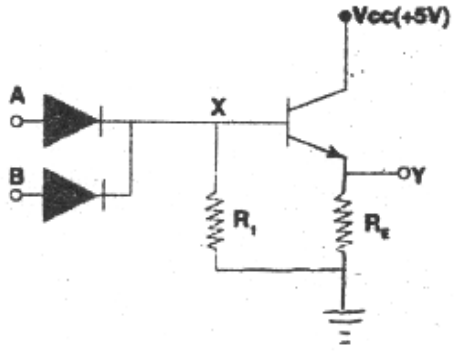


Figure 4.4 DTL OR Circuit

DTL NAND Gate

DTL NAND Circuit is shown in Figure4.5. If both the inputs are high (logic 1), the diodes D_1 and D_2 are reverse biased and passes no current and point X is clamped to +5V; the diode D_3 is forward biased and current flows through R_1 and D_3 to the base of Q_1 thus saturating the transistor, this causes the output point-Y to go low (logic0).

If either input change to its low logic (logic 0) the corresponding input diode conducts and drives current from base of Q_1, thus the transistor is off and the output voltage rise to logic1.

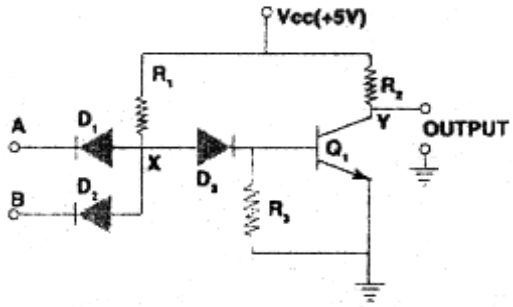


Figure 4.5 DTL NAND Circuit

- **Parameters**

Fan Out	:	8
Noise Margin	:	0.7 V
Propagation Delay	:	30 n sec
Power Dissipation	:	8-12 mW/gate
Power Supply Voltage	:	+5V

- **Advantages**

It has better noise immunity than RTL circuit.

It is much more economical because of the use of diode in place of resistors and capacitors.

It has low power dissipation.

- **Disadvantages**

It has power noise margin.

It has high propagation delay.

Transistor-Transistor Logic

TTL logic is modified form of DTL logic, the input diodes in the case of DTL logic are replaced by transistor with more than one emittor with more than one emitter as input. Figure 4.6 shows the diode and transistor AND gate circuits.

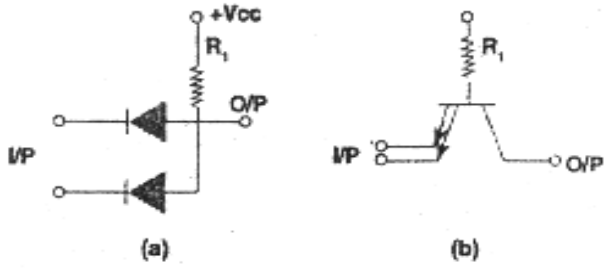


Figure 4.6 (a) Diode AND Gate (b) Transistor AND Gate

- **TTL NAND Gate with Totempole Output**

The basic TTL logic element is a NAND gate and is shown in Figure4.7. The circuit consists of three states one input stage associated with multiple emitter transistor Q_1, a switching stage consisting of transistor Q_2 and output stage having transistor Q_3 and Q_4.

If either or both the inputs A and B are at logic 0, Q_1 conducts and no current flows into the base of Q_2. Therefore, Q_2 is OFF, the voltage at the collector of Q_2 rises to Vcc.

If both inputs A and B are at logic 1 level, no current flows out of the emitter of Q_1 but it does flow through the base of Q_2 through R_1 thus turning on Q_2. As Q_2 is ON, it supplies base current to Q_4causing it to turn ON, when Q_4 is ON its collector voltage and thereby output Y is at zero level. Note that because Q_2 is ON, it collector voltage drops and keeps Q_3 in the OFF State.

The function of diode D in the circuit is to prevent both Q_3 and Q_4 from being turned ON simultaneously because if both transistors becomes ON they will offer low impedance to the supply which will draw excessive current and produce large noise spike at the output.

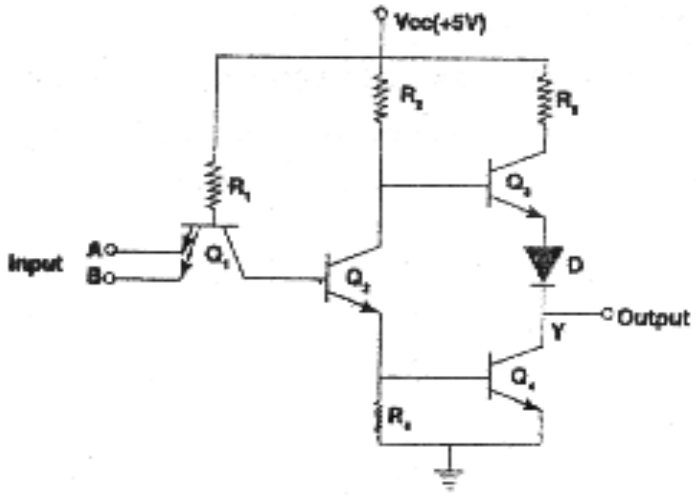


Figure 4.7 TTL NAND Gate Circuit

- **Open Collector TTL**

In open collector TTL logic (Figure 4.8) in the output stage, the emitter follower transistor Q_4 and diode D are absent as compared to the totempole logic. The collector of Q_3 is not connected anywhere internally in the circuit and is open which is connected to common external collector resistor RL. Such IC's are used for wired-output* connection when using resistive loads and for operations with non-standard loads.

With any input LOW, Q_1 is ON thus making Q_2 and Q_3 OFF drawing output HIGH.

With all inputs HIGH resulting in Q_1 OFF and Q_2 and Q_3 ON which makes the output LOW.

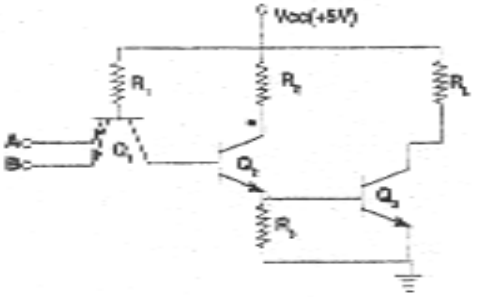


Figure 4.8 TTL NAND Gate Circuit

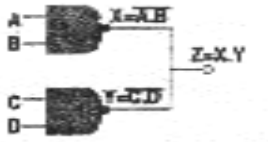


Figure 4.9 Wired-AND Operation

- **TTL Tristate Logic**

A Tristate Logic has three logic. LOW, HIGH and IMPEDANCE state. When the enable line as shown in Figure 4.10 is high the circuit works as normal TTL NAND gate as explained above.

When enable line is LOW the diode D1 becomes forward biased which keeps Q_2 ON, Q_2 Q_3 and Q_4 OFF, with both output transistors Q_3 and Q_4 OFF, the output impedance is very HIGH, this is the HIGH impedance state.

- **TTL Sub-families**

74L XX Series

The letter 'L' denoted that this is a low power TTL IC. A low power TTL gate

has a power dissipation of 1mW, which is made possible by increasing the internal resistance of the IC. This affect the speed of operation of the gate i.e. its propagation delay increases due to increased time constant, the propagation delay of such IC's is about 35 nano seconds.

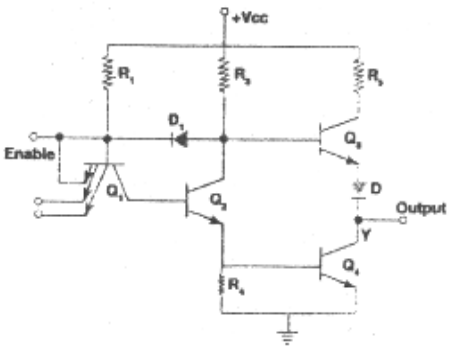


Figure 1.10 TTL Tristate Logic Gate Circuit

74H XX Series

The letter 'H' stands for high speed. The speed can be increased or to say the propagation delay of gate can be decreased by decreased by decreasing the time constant, which can be done by reducing the internal resistance, this costs in increase in the power dissipation per gate. 74 H 00 series has propagation delay of 6 n seconds and power dissipation of 23 MW/gate.

74S XX Series

The letter 'S' stands for Schottky and this series has the highest speed available in TTL series. The Low propagation delay is achieved by use of low voltage-drop diode, called **Schottky diode.** The diodes have low forward voltage drop about 0.4V and have fast switching speed, the reason for its is that this type of diode have a metal to silicon junction and not silicon to silicon junction as in the ordinary p-n junction diodes. There are therefore, no minority carriers and no storage charge. Stored charges are the minority carriers in ordinary p-n junction diodes. Absence of stored charges means that the diode can switch faster than diode which do have stored charges. The Schottky diode is connected between base and collector of each transistor and it prevents the circuit transistor from saturating which results in low switching time.

Schottky TTL has a propagation delay of 3-nano seconds and power dissipation of 23 mW/gate.

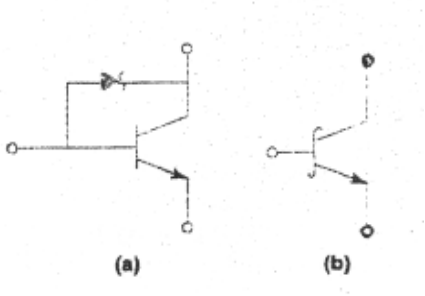


Figure 4.11 (a) Transistor with Schottky Diode (b) Symbol of Schottky Transistor

74LS XX Series

The letter 'LS' stands for the low power Schottky. A low power Schottky TTL series is compromise between the propagation delay and power dissipation. The power dissipation is reduced by increasing the internal resistance and speed is increased by considering Schottky diode. For almost all-commercial purpose instruments this LS series is most suitable choice.

Low-power Schottky series has power dissipation of 2mW and propagation delay of about 10n seconds/gate.

The major parameters, advantages and disadvantages of TTL IC's are listed below :

- **Parameters**

Fan Out	:	10
Propagation Delay	:	07-10 n sec
Power Dissipation	:	10 mW/gate
Noise Margin	:	0.4 V
Power Supply Voltage	:	+5V

- **Advantages**

The main advantage of TTL IC's is that they are compatible with other IC's

It has high speed of operation.

Its smaller size yields more function on an IC.

It has low output impedance which improves the fan out capability.

It has good noise immunity, in worst case it is 0.4V and typically it touches 1V.

It is less expensive.

- **Disadvantages**

It generates switching transients, which can be eliminated by the use of bypass capacitors.

Wired output capability is not possible except with low level and open collector IC's.

CMOS Families

A Complementary Metal Oxide Semiconductor (CMOS) is obtained by connecting a p-channel and n-channel MOSFET in series, with drains tied together and the output is taken at the common drain. Input is applied at the common gate formed by connecting the two gates together (Figure3.12). This type of IC's are noted for their exceptionally low power consumption.

- **Advantages**

The CMOS family of ICs has the following advantages:

Low cost.

Simplicity of design.

Low heat dissipation.

Superior fan-out and Wide logic swings.

Good noise margin performance and Wide-range operation.

- **Disadvantages**

Slower than Bipolar digital ICs such as TTL devices.

Careful handling for protects from static discharges is needed.

Transient voltages can damage the oxide layer in the chip.

To prevent the damage, CMOS ICs are stored in special conducting foam or static shielding bags. The extremely low power consumption makes them ideal for battery operated portable device. They are widely used in electronic wrist-watches, calculators, portable computers and space vehicles.

A typical (CMOS) device is shown in Figure 4.12. Both are enhancement-mode MOSFETs. Then the input voltage V_{in} is LOW, the to MOSFET is ON and the bottom is OFF. The output voltage V_{out} is then HIGH. However, if V_{in} is HIGH, then V_{out} is LOW and the device therefore, acts as an inverter.

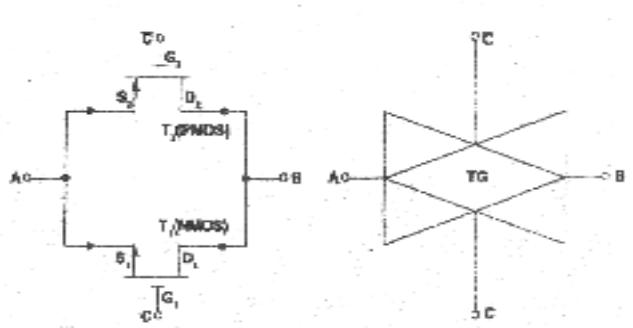


Figure 4.12 (a) A CMOS Transmission Gate (b) Symbol

A two input CMOS –NAND gate and NOR gate are shown in Figure 3.14 (a) and 3.15 (a). A CMOS can be used as a Transmission Gate or Bilateral Switches as shown in Figures. 4.12(a) and 4.12(b). The gates can conduct or allow a signal to pass in either direction like relay conducts. There are two gate voltages C and 'C that controls the transmission gate.

Let C=1 and if A=V(1), which means T_1 is OFF and T_2 is ON. Therefore, $_{T2}$ behaves as a small resistance connecting the input to the output (conducts in the Ohmic region) and B=A=V(1).

Similarly, if A=V(0),which means T_2 is OFF and T_1 is ON and B=A=V(0).

This means the signal is transmitted from A to B when C=1. On the other hand, ifC=0, transmission is not possible. The CMOS technology is used in making several

families of digital ICs. The most popular are the 4000, 74C00, 74H00, and FACT series IC.A 7400 TTL IC is designated as a quadruple (Quad) two-input NAND gate. The FACT (Fairchild Advanced CMOS Technology) logic IC series includes many sub families and was designed to perform existing CMOS bipolar logic families. It is the best at the moment and has very low power consumption (0-1mW/FACT at 1MHz). It has out-standing noise immunity and propagation delay.

Interfacing TTL to CMOS Switches

One of the most common means of entering information into a digital system is the use of switches or a keyboard. Three simple interface switch circuits are shown in Figures 4.13 (a), (b) and (c). In either case, TTL is capable of sinking sufficient current to drive an unlimited number of CMOS gates at low frequency. For the active low switch interface with pull up resistor (Figure 4.13(a)), a 10 KW external resistor is used at the output called **pull-up** resistor. Its purpose is to pull the input voltage up to +5V. To interface TTL with CMOS that is operating at levels, one of the high voltage open collector gates can be used (Figure 4.13(b)). The open collector output is pulled up to the operating voltage of the CMOS gate.

Figure 4.13(c) shows a CMOS NAND gate that could be operated from 5V to 18V. in each case, the resistance value of the pull up and pull down resistors is much greater than those in TTL interface circuits. This is because the input loading currents are much greater in TTL than in CMOS.

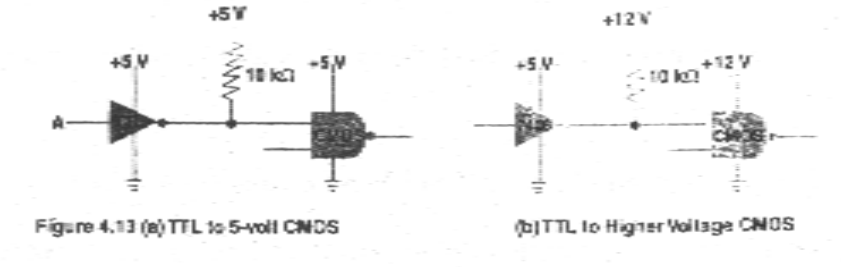


Figure 4.13 (a) TTL to 5-volt CMOS (b) TTL to Higher Voltage CMOS

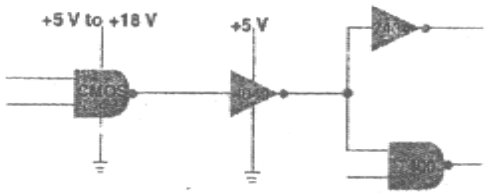


Figure 4.13 (c) CMOS to TTL interface

Comparison of Logic Families

The salient features of all the logic families and the comparative study of their performances are listed below:

- TTL devices are faster than CMOS ICs.
- RTL and DTL families are less efficient because of low speed, high power dissipation and low Fan-out. Modern digital system does not use RTL and DTL families.
- The most widely used logic family is TTL. They are very popular because of wide range of operating speed, power dissipation and large Fan-out. TTL is available is seven different series with large number of functions.
- TTL ICs are available with

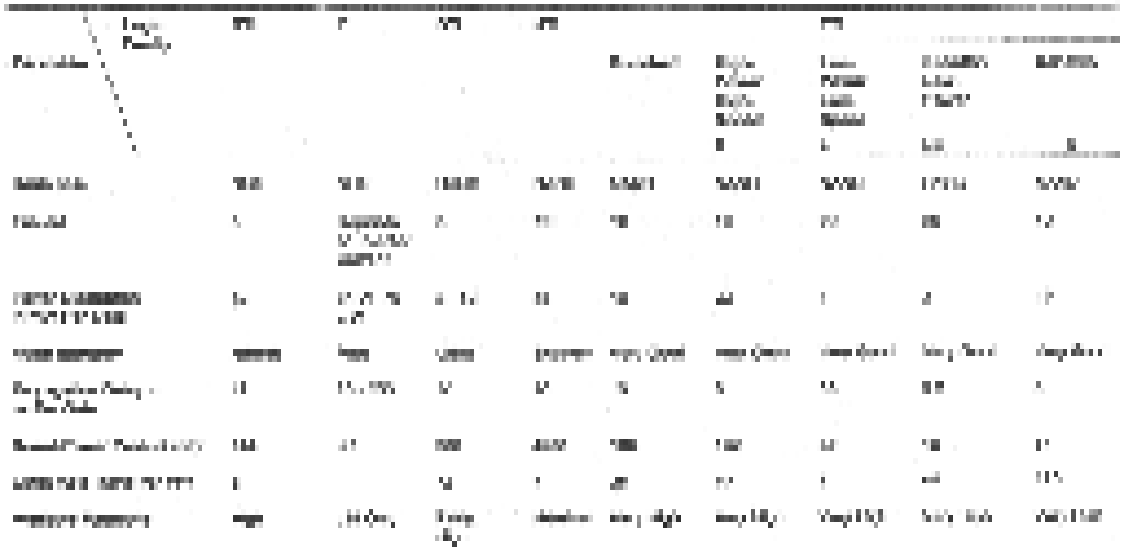
a) Totempole output, in which figure of merit is low.

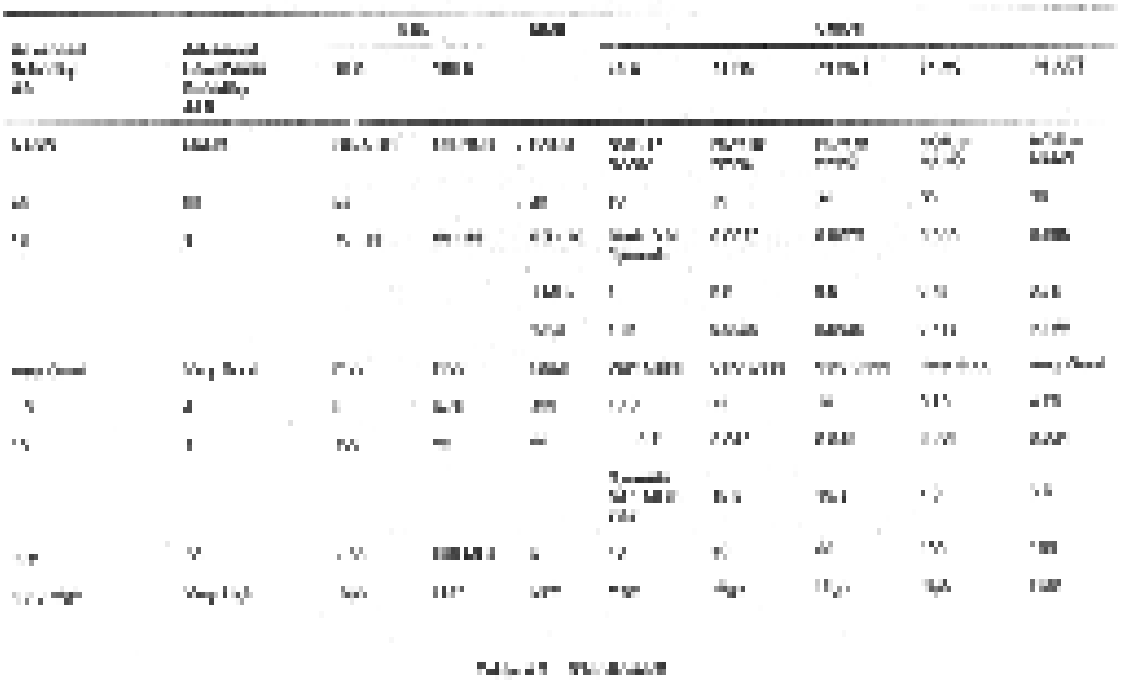
b) Open collector output which is available with wired-AND connection and BUS operation.

c) Tristate logic outputs that are ideal for BUS operation.

- The MOS logic family is the most popular logic for large-scale integration because of low power consumption and small sizes. Its main draw back is slow speed. However PMOS and a variety of NMOS has speed comparable to Bipolar logic families.
- The Figure of merit of CMOS is very low.

The CMOS family is quite comfortable with various TTL series and has the same numbering scheme and pin-outs. Table 4.2 represents a detailed comparison of various logic families.





Tristate Logic/Buffers

The non-inverting buffer serves no logical purpose. It does not invert but is used to supply greater drive current at its output than is normal for a regular gate. Since regular digital ICs have limited drive current capabilities, the non-inverting buffer/driver is very important while interfacing ICs with other devices such as LEDs, LAMPs and others. Buffer/drivers are available in both non-inverting and inverting form.

In normal logic circuits there are two states of the output, LOW and HIGH. If the output is not in the LOW State, it is definitely in the HIGH state and vice versa. In complex digital systems like micro-computer, microprocessors and signal processors, a number of gate outputs requires a common line called BUS, which in turn, may be required to drive a number of gate inputs. There are three logic levels associated with such device and is therefore, called tristate logic or TSL.

There are some difficulties with the connection of gate outputs to the BUS like:

1. Totem-pole outputs cannot be connected together because it causes heating of ICs.
2. Open collector outputs cannot be connected together that causes problem of loading and speed of operation.

A special circuit is designed with one more state of the output to over-come such difficulties, referred to as the **third state or high – impedance** state called ENABLE in addition to the LOW and HIGH states. These circuits are known as TRI-STATE, tri-state-logic (TSL) or three-state logic.

Figure 4.14 represents the logic symbol of a TSL inverter and the corresponding Truth Table is realized in Table 4.3.

Data Input	*Control*	*Data Output*
0	0	HIGH-Z
1	0	HIGH-Z
0	1	1
1	1	0

Table 4.3: Truth Table of a TSL inverter

A TSL inverter circuit with tri-state output is shown in Figure 4.15. When the control input is LOW, the drive is removed from T_3 and T_4. The output is in the third state because both T_3 and T_4 are cut-off. On the other hand, if the control input is HIGH the output Y is at logic 1 or 0 depending on the data input.

Figure 4.14 Logic Symbol of a TSL Inverter

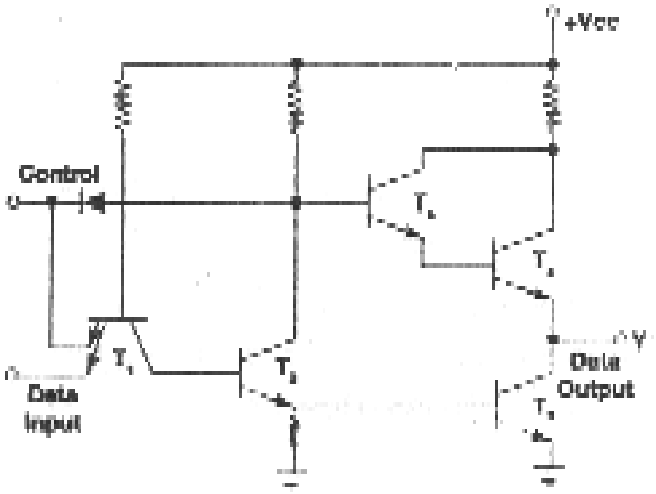


Figure 4.15 A TSL Inverter

❖ ❖ ❖

: 5

Author :
Dr. Dharminder Kumar

Vetter :
Dr. Sib Krishna Ghoshal

- **General concepts**
- **Operation of RS, JK, D and T Flip-Flops**
- **Synchronous and Asynchronous Counters**
- **Ring Counters**
- **Shift Registers**

Introduction

As mentioned earlier, in previous chapters, the logic circuits are classified into two major groups

1. Combinational Logic Circuits.
2. Sequential Logic Circuits.

Logic gates are in the category of combinational logic circuits. Sequential circuits involve timing, counting and memory devices. The basic building block of sequential logic circuits is the **Flip-flop (FF)** like logic gate, the building block of combinational logic circuits. There are several types of Flip-Flop circuit and the control inputs vary with each type. FF are wired to form counters, shift registers and various memory devices. They are capable of storing binary information.

A flip-flop has two stable states 0 or 1. When it is said to one of these states, it remains in that state until the application of a control signal causes it to FLIP or 'FLOP'to the other state. It is a bistable multivibrator circuit. It is a digital circuit has two outputs Q and $\overline{Q}$ which are always in opposite state if Q is 1 then `Q is 0 the flip-

flop is said to be SET, ON or PRESET. If Q is 0 then `Q is 1 and the flip-flop is said to be RESET, OFF or CLEARED. The logic levels on the flip-flop inputs will determine the state of the Q and `Q outputs according to the truth-table for the type of flip-flop.

Unlike the gates, the flip-flops can in some states maintain its output state (ON or OFF) after the input signals which produce the output. Thus the flip-flop can store a bit of information or one place of a larger binary number.

The basic types of Flip-Flops are

- J-K Flip-Flop
- R-S Flip-Flop (R-S stand for Reset-set)
- D-Flip-Flop (D stand for Delay)
- T-Flip-Flop (T stand for Toggle)

The block diagram of a sequential circuit is shown in Figure 5.1. The role of the combinational circuit is to accept digital signals from external inputs and from outputs of memory elements. Then it generates signals for external outputs and for inputs to memory elements. The output of a sequential circuit is a function of the time sequence of inputs and the internal states.

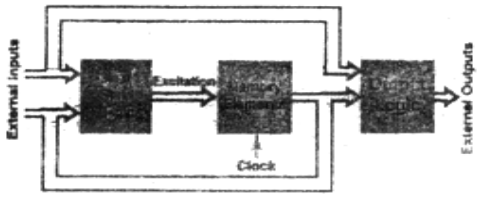


Figure 5.1 Block Diagram of a Sequential Circuit

Depending on timing of their signals, sequential circuits are classified in two main categories.

- Synchronous
- Asynchronous

A sequential circuit whose performance depends upon the sequence in which the input signals change is referred to as an asynchronous sequential circuits. The

commonly used memory elements in these circuits are time delay devices. A sequential circuit whose operation can be defined from the knowledge of its signal at discrete instance of time is referred to as synchronous sequential circuit. In these systems, the memory elements are affected only at discrete instance of time. Synchronous circuits are also known as *Clocked-sequential Circuits.* The synchronisation is achieved by a timing device known as a S*ystem-clock,* which generates a periodic train of clock pulses as shown in Figure 5.2.

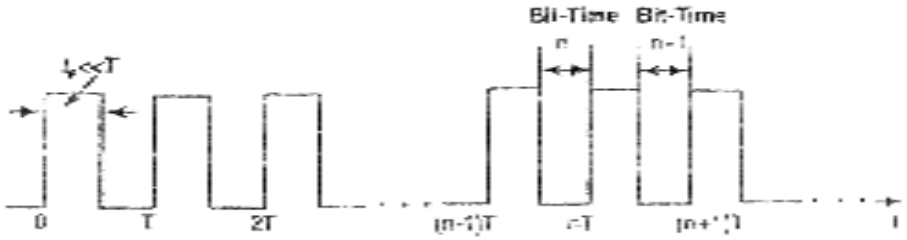


Figure 5.2 A Train of Pulses

- **Clock Flip-Flop**

To push the inputs to the outputs at any desired time and then to hold the outputs for the desired time we use clock and flip flop, now the flip-flop has three inputs e.g. S,R and Clock sometime it is also called **R-S-T Flip Flop.**

- **Clock**

A pulse generator within a digital system to which all operations are synchronized is called clock. The square wave in Figure 5.3 is a typical clock waveform used in a digital system and it is not necessary that the clock to be perfectly a symmetrical square wave as shown in Figure 5.3. It could simply be a series of positive (or negative) pulses but the main requirement is that the clock should be perfectly periodic.

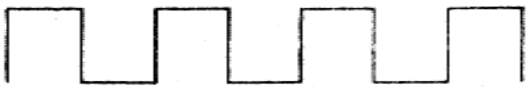


Figure 5.3 Ideal Clock Waveform

For a clock to be ideal it is necessary that

1. The clock level remains absolutely stable, when the clock is HIGH, its level must hold a steady value of +5V, and when it is LOW the level must remain at 0V.
2. The second necessity for an ideal clock is that transition time for the clock should be zero, i.e. the change occurring from 0V to +5V should take no time and vice versa but this is not possible in real practice, the waveform take sometime to make the change.

The practical square wave looks like as shown in Figure 5.4.

The time required for transition from LOW to HIGH is defined as **Rise Time (t_r)**. The time requires for transition from HIGH to LOW is defined as the **Fall Time (t_r)**.

- **Rise Time (t_r)**

Time required for the waveform to travel from 10% or its initial value to 90% of the final value is known as rise time.

- **Fall Time (t_r)**

Time required for the waveform to travel from 90%of its final value to 10% of the initial value is known as fall time.

Some other definitions that will come across in the following topics are:

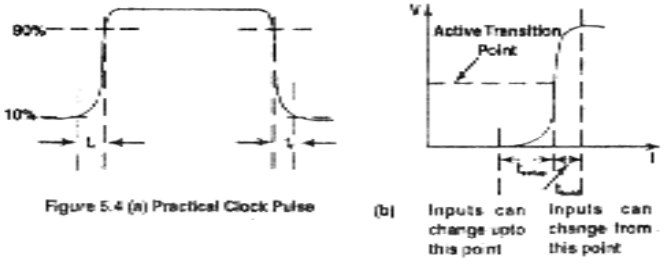


Figure 5.4 (a) Practical Clock Pulse (b)

- **Propagation Delay Time**

The amount of time taken by the gate or flip-flop to cause a change at the output after input has changed. It is around 10 nano seconds for LSTTL logic ICs.

- **Set UP Time**

It is minimum time over which the data bit must be present before the clock edge hits

- **Hold Time**

It is the minimum amount of time over which data bit must be present after the clock edge arrives.

OPERATION OF R S, JK, D AND T FLIP-FLOP

Clocked S R Flip-Flop

The addition of two AND gates at the S-R inputs are shown in Figure 5.5 (a). When the CLK input is low the AND gate outputs will be LOW and changes in neither S or R will have be transmitted to the outputs. When the ENABLE inputs goes LOW, the output will retain the information that was present on the input when HIGH to LOW transition takes place. The Truth Table 5.1 represents its detailed working principle.

CLK	*S*	*R*	*Q*
0	0	0	No change
0	0	1	No change
0	1	0	No change
0	1	1	No change
1	0	0	No change
1	0	1	0
1	1	0	1
1	1	1	? (Forbidden)

Table 5.1: Truth Table for Clocked SR Flip-Flop.

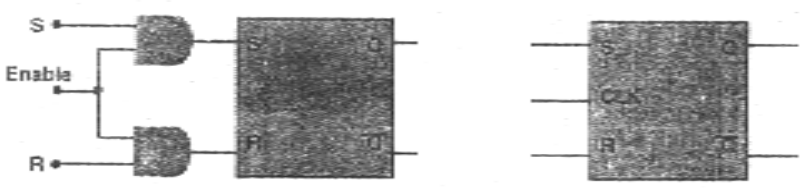


Figure 5.5 (a) Logic Diagram (b) Symbol

Clocked S-R Flip Flop using NAND Gate Only

Before discussing the condition let us first know the meaning of Sn, Rn and Qn+1:Sn and Rn means inputs applied during nth clock cycle and Qn means output during Nth clock cycle. Whereas, Qn+1 means output during (n+1)th clock cycle (as depicted in Figures 5.6 (a) and (b)).

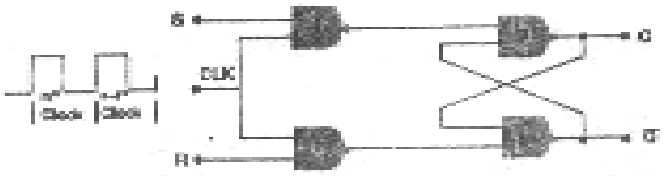


Figure 5.6 (a) Clocked NAND S-R Flip Flop

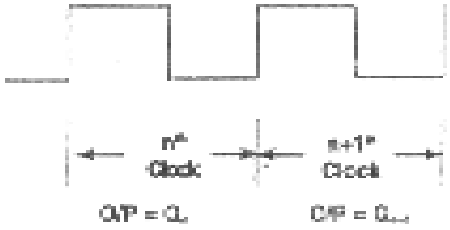


Figure 5.6 (b)

CLK	S	R	Q(n+1)
X	0	0	Q_n X means 0 or 1
1	0	0	0
1	1	0	1
1	1	1	? (Forbidden)

Truth Table 5.2

Condition 1: S_n=0, R_n=0

Suppose when n^{th} cycle was commencing at that time output was Qn=1 and when the next clock cycle i.e. $(n+1)^{th}$ cycle comes, the output remains the same as it was during the $(n+1)^{th}$ cycle is Q_n+1=1 (No change).

If the output during nth cycle was 0 i.e. Q_n=0 then it will remain 0 during $(n+1)^{th}$ cycle, so for S_n = R_n=0, output is not changed.

Condition 2: S_n=0, R_n=1

Suppose when nth cycle was commencing, at that time the output was say Q=1 and we apply the output S_n=0, R_n=1 and when $(n+1)^{th}$ cycle comes, the output becomes Q_n+1=0

Condition 3: S_n=1, R_n=0

Now when S_n=1 and R_n=0 is applied during the low state of n^{th} cycle then whatever be the output during n^{th} cycle the output become 1 during $(n+1)^{th}$ cycle i.e. Q_n+1=1

Condition 4: S_n=1, R_n=1

As discussed earlier, this condition is forbidden as both the outputs try to become 1, which violates the definition of flip-flop, that one output is the complement of the other. The Truth Table 5.2 explains its working.

Edge Triggered S-R Flip Flop using NAND Gates

The Figure 5.7(a) shows as R-S circuit at the input of a S-R flip flop, which is acting as a differentiator whose R-C time constant is kept very low, because of this the capacitor charged fully when the clock goes high, this exponential charging produces a narrow positive spike across the resistor, the trailing edge of the pulse result in a narrow negative spike. This narrow positive spike enables the NAND gate for an instant and the negative spike has no role. This type of edge triggering allows to make the transition at a fixed time interval.

If the triggering is occurring at the positive edge it is called positive edge triggered S R flip-flop and is represented by an upward arrow in the truth table (Table5.3) and by a notch ">" in the symbol (Figure 5.7 (b).

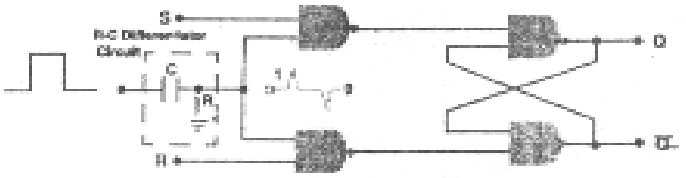
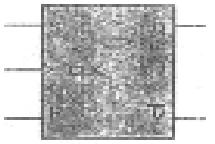
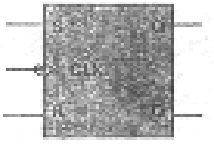


Figure 5.7 (a) Edge Triggered S R Flip Flop with RC Differentiator Circuit



(b) Positive Edge Triggered Flip Flop



(c) Negative Edge Triggered Flip Flop

CLK	S_n	R_n	Q_{n+1}
X	0	0	Q_n
↑	0	1	0
↑	1	0	1
↑	1	1	?(Forbidden)

Truth Table 5.3

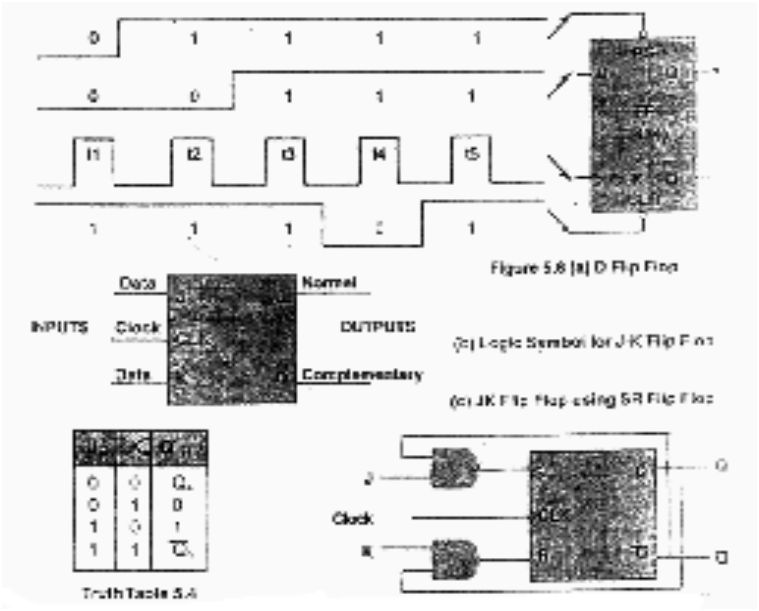
If the triggering is occurring at the negative edge pulse, then it is called negative edge triggered S R flip-flop and is to be shown by a downward arrow in the truth table and by a bubble "O" at the clock input as shown in Figure 5.7 ©.

J K FLIP FLOP

J K Flip Flop from S R Flip Flop

J K Flip Flop is the most widely used flip flop and it eliminates the uncertainty in the fourth row of the truth table of the S R flip flop when S=R condition is applied. One way of constructing J K flip flop is by using S R flip flop and AND gates as shown in Figure 5.8(c). The upper AND gates has two inputs J and the other one connection to the 'Q output.

When input are J=K=1 then output is complemented on the application of clock pulse i.e. if before application of clock pulse the output was 0 then after application of the clock the output becomes 1. The working principle is explained in the truth Table 5.4 and 5.5.



CLK	J	K	Q	$\bar{Q}$	Effect on output Q
	0	0	No change		No change — disable
	0	1	0	1	Reset or cleared to 0
	1	0	1	0	Set to 1
	1	1	Toggle		Change to opposite state

Truth Table 5.5 J-K Flip Flop

J K Flip Flop using NAND Gates

The same function can be realized using NAND gates also and is shown below in Figure 5.9 (a).

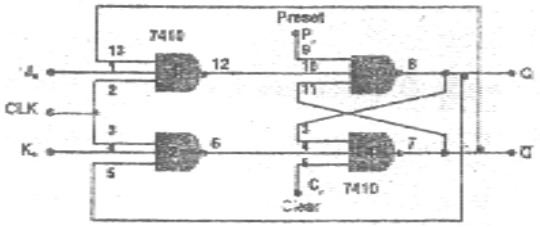


Figure 5.9 (a) J K Flip Flop using NAND Gates only

In the Figure, two new inputs are shown they are Preset (P_r) and Clear (C_r). These inputs are used for assigning initial state for the flip-flop. Suppose it is necessary to have clear outputs i.e. Q=0 when clock goes to zero, this can be done by giving C_r=0 and Pr=1.

Similarly, Q can be set to 1 by choosing P_r=0 and C_r=1 when clock is at zero state. It should be noted that these inputs should only be given when the clock is at zero and once the state of flip flop is established it is necessary to have P_r=C_r=1 before the next clock pulse arrives and at no time P_r=C_r=0 should be given as it leads to uncertainty in the state of flip flop.

The first three conditions in the truth table of J K flip flop is same as S R flip flop but the last condition when J=K=1 is the subject of interest.

Let Q_n=0 and J_n and K_n=1, when the clock is made HIGH, then the output changes to Q_{n+1}=$Q_{n=1}$ this change at the output takes place after a time period corresponding to the propagation delay through two NAND GATES, which is around 20-25 n sec for each gate for TTL logic, therefore, during the presence of clock the output again change to 0 and it go on changing from 0 to 1 and 1 to 0 until the clock is HIGH therefore, it is difficult to predict the output at the end of the clock as the 1 output state is dependent on the pulse width. This is known as **RACE AROUND CONDITION.** The Figure 5.9 (b) shows the output toggles four times for the given

width of the clock pulse and the final output is 0 where as in the second pulse of Figure 5.9(b) for given clock pulse T_2 the output toggles five times and the final output is 1 this is an ambiguous situation where outputs are unpredictable, thereby, giving different outputs every time.

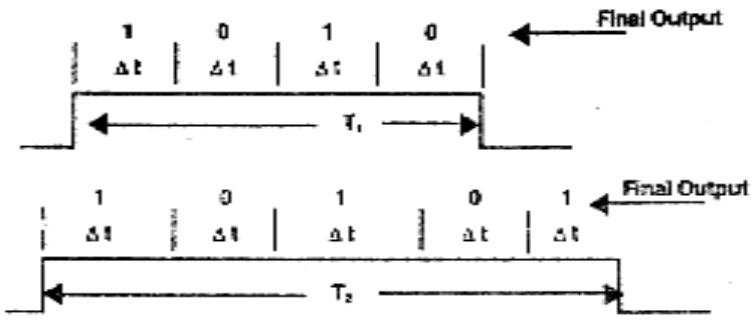


Figure 5.9 (b) Train of Pulses

One easy way to eliminate the race around condition is by making the time period of the clock less than the propagation delay time of the gate but practically this is not possible because the propagation delay is very small, the problem is solved by master slave J K flip flop. (Figure 5.10 (a) and (b)).

Master Slave Flip Flop provides a way to avoid race around problem which is caused because of the changing inputs while the clock is HIGH as the inputs are connected to the output, the racing problem can be checked some arrangement is made so that inputs are disabled while outputs are changing, this can be easily achieved by dividing the flip flop into two parts one is named **MASTER** and the other **SLAVE** as it follows the master (Figure 5.10 (b)).

The MASTER flip flop is positive edge triggered J K flip flop and the SLAVE is negative edge triggered flip flop, therefore, when clock is HIGH the MASTER flip flop is enabled and passes the inputs to its output, during this time the SLAVE flip flop is disabled and do not function. When the clock goes LOW during the time the MASTER is disabled and SLAVE flip flop is enabled and passes the outputs of the MASTER which were produced when the clock was HIGH to its output.

Let us study the last condition of M/S JK flip-flop which is the subject of interest for us when $J_n=K_n=1$, let us assume the output $Q_n=0$, the various conditions are shown n Figure 5.10 (b).

When the clock is HIGH then the output of MASTER flip flop toggles from 0 to 1, i.e. Q=1 now the inputs of the SLAVE flip flop becomes 1 and 0. When the clock goes LOW these inputs are on the output thus changing the output from 0 to 1 i.e. Q_{n+1}=1 which is complement of the previous output this known as toggling. When the SLAVE flip flop output is toggling at the time the MASTER flip flop is disabled and accepts no change at its input thus avoiding the problem of racing. The Figure 5.10(c) shows the M/S flip-flop using NAND gates only.

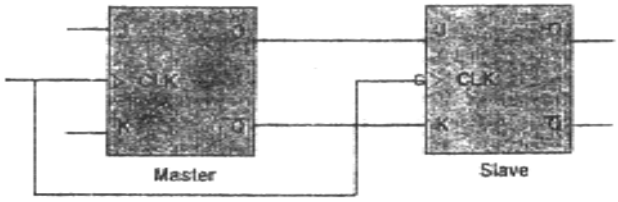


Figure 5.10 (a) Master and Slave Flip Flop

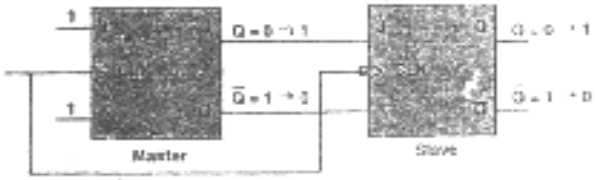
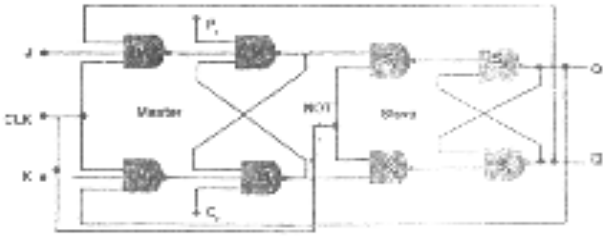


Figure 5.10 (b)



Edge Triggered M/S JK Flip Flop

Now the question comes in the mind that what is the advantage of edge triggering over level triggering. The reason for that is the output of level triggered flip flop can change at any instant when the clock is HIGH, one flip flop may change after DT1 time and the other at some DT_2 time so the exact time when the output appears is uncertain. Therefore, if one wants to cascade two or more flip flops together it will not work satisfactorily, for that reason edge triggered flip flop is preferred in which flip flop is made to trigger only at positive or negative edge of the pulse thus

ascertaining the time of toggling of the flip flop. The Table 5.6 explains its working details.

CLK	J_n	K_n	Q_{n+1}
X	0	0	Q_n
↓	0	1	0
↓	1	0	1
↓	1	1	$\overline{Q}_n$ (Toggle)

Table 5.6: Truth Table of Edge Triggered M/S J K Flip-Flop

The Pin Out of IC 7476 is shown here, which is a negative edge triggered dual M/S JK flip flop (Figure 5.11).

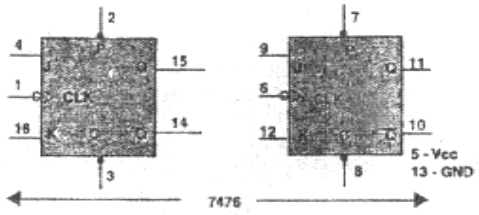


Figure 5.11 Pin Out of 7476 IC

D Flip Flop (DFF)

It is essentially a delay flip flop and it is so called because the bit at the input is transferred at the output of the flip flop when the next clock pulse comes. The D flip flop can be constructed using either SR flip flop of JK flip flop. If we look at the middle two conditions of these flip flops we find that the inputs are dissimilar if one is 0 then the other is 1 and when clock is applied the dissimilar inputs are passed on to the output. This can be done here by placing an inverter in between the two inputs of the AND gate as shown in Figure 5.12 (a).

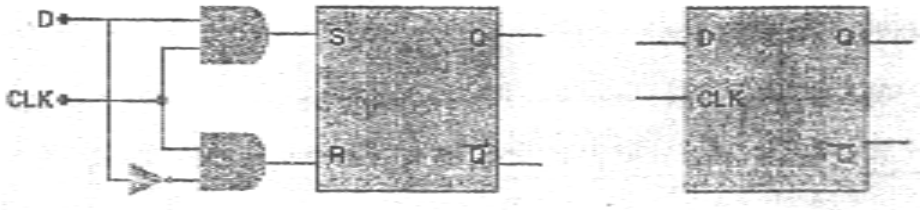


Figure 5.12 (a) D Flip Flop Circuit (b) Symbol

Let us analyses the working of the flip flop

Condition 1: D=0 (0 or 1) and CLK=0

When the clock is zero and the input is either 0 or 1, the input of the AND gates are disabled thus giving 0-0 to the input of say SR flip flop and for that condition the output is non-change condition.

Condition 2: D=0 and CLK=1

For this condition the upper AND gate is disabled and the lower one enabled thus giving say S=0 and R=1, for this condition of SR flip flop the output is 0.

Condition 3: D=1 and CLK=1

Here the upper AND gate is enabled and the lower one disabled thus producing S=1 and R=0, for this condition of SR flip flop the output is 1.

Thus we see that the output flows the input on the application of the clock pulse. All these conditions are tabulated in Table 5.7.

CLK	K_n	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1

Table 5.7: Truth table for DFF

Edge Triggered D Flip Flop

The circuit symbol and the truth table of edge triggered D flip flop is shown in Figure 5.13 and Table 5.8. D flip flop finds its application in making registers.

CLK	D	Q_{n+1}
0	X	Q_n
↑	1	1

Table 5.8: Truth Table for Edge Triggered D Flip-Flop

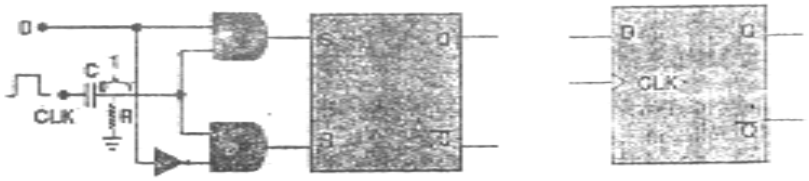


Figure 5.13 (a) Edge Triggered Flip Flop Circuit (b) Symbol

5.1.1 T Flip Flop (TFF)

Toggle flip flop is an extension of JK flip flop. When both J and K inputs of the flip flop is tied or shorted, it become T flip flop. The flip flop to work as toggle flip flop its input is at 1 and when the clock pulses are applied, the output toggles. The Figure 5.14 shows the symbol and truth table of positive edge triggered T flip flop. T flip flop is used in making counters.

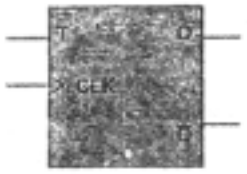


Figure 5.14 Symbol of Positive Edge Triggered T Flip Flop

CLK	K_n	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1

Table 5.9: Truth Table for Edge Triggered T Flip-Flop

SYNCHRONOUS AND ASYNCHRONOUS COUNTERS

Synchronous Counter

A synchronous parallel or clocked counter is one in which all stages are triggered simultaneously. The resulting action of each stage depends on the getting inputs (synchronous-inputs) of each respective stage. This type of counter is faster than ripple or asynchronous counter since higher order stages don't have to wait for lower order changes to occur.

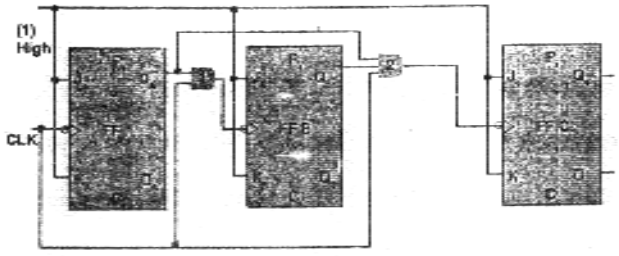


Figure 5.15 Synchronous Up Counter

The circuit of a three bit synchronous Up-counter is shown in the Figure 5.15. in the circuit, JK flip flops are made to work as T flip flop. It can be seen that all the flip flops are getting the clock at the same instant, the FFA gets direct clock and, FFB and FFC through AND gates. The AND gate at the CLK input of FFB passes every second clock to FFB and every fourth clock to FFC and so on. The circuit of synchronous counter can be constructed using T FF.

Working Principle (Table 5.10):

1. Before applying the CLK pulses all FF's are RESET.
2. When first clock pulse comes at the negative edge of the negative edge of the clock FF A toggles from 0 to 1 and the state of FF B and FF C remains unchanged as disabled AND gates do not pass the clock.

   $Q_C\ Q_B\ Q_A$=001
3. After first clock pulse the 1st AND gate is enabled, therefore, when second clock arrives then
4. Simultaneously FFA and FF B toggles. The outputs at the end of second pulse is:

   $Q_C\ Q_B\ Q_A$=010
5. Because Q_A=0, the upper pin of 1st AND disabled and due to Q_B one of the pin of 2nd AND gate is high but, its one pin is also connected to Q_A output which disables the 2nd AND gate. When third clock pulse hits only FF A responds and toggles from to 0 to 1. Thus the outputs at the end of third clock pulse is:

   $Q_C\ Q_B\ Q_A$=011

6. After third clock passes, 1st and 2nd AND gates are enabled and when fourth clock pulse arrives it passes to all FF's which results the output:

$Q_C\ Q_B\ Q_A$=100

7. Like this the count advances, at the end of the 7th clock pulse the count sequence is $Q_C\ Q_B\ Q_A$=111 and 8th clock all the FF's resets to 000 and the cycle repeats.

CLK	Q_C	Q_B	Q_A
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

Table 5.10: Truth Table for Synchronous Up Counter

Down Counter

Here minor changes are made in the UP-COUNTER circuit to make the DOWN COUNTER. The first AND gate is connected to 1_A and the second AND gate is connected to 1_B and outputs are considered from $Q_A\ Q_B\ Q_C$. Initially, FF's are SET to QC, QB, QA=111 and then clock pulses are applied to it and with each clock the counting is reduced by one count and on the seventh clock the count sequence is QC,

QB, QA=111 and when eight clock arrives they all are again set to QC, QB, QA=111. The count sequence is shown in the Truth Table 5.11

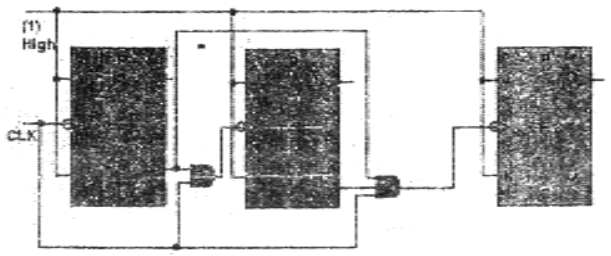


Figure 5.16 Synchronous Down Counter

CLK	Q_C	Q_B	Q_A
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

Table 5.11: Truth Table for Synchronous Down Counter

Up-Down Counter

This circuit (figure 5.17) has both the features of Up-and Down counter when clock is applied to UP input and Down input is kept zero then all the A-AND gates are enabled and all B-AND gates are disabled the circuit works as Up counter as discussed earlier.

When clock is applied at DOWN input and Up input is kept at zero then all B-

AND gates are enabled and A-AND gates are disabled now the circuit works as normal down counters as discussed above.

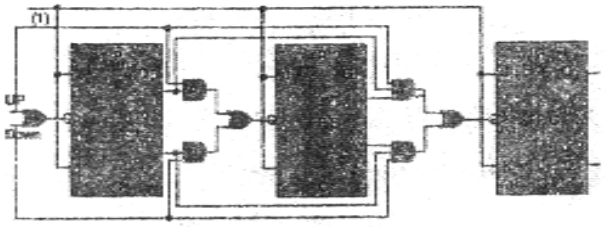


Figure 5.17 Synchronous Up Down Counter

ASYNCHRONOUS COUNTERS

Ripple Counter or Up Counter or Mod-16 Counter

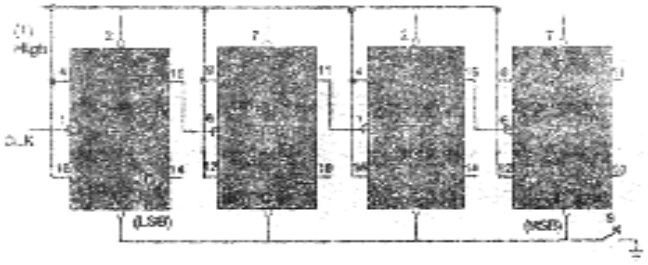


Figure 5.18 Asynchronous Ripple Counter

The binary ripple counter using IC 7476 is shown in Figure 5.18. The four M/S J-K FFs (or TFF) are connected in cascade where FFA represents the least significant bit (LS) and FF D the most significant bit (MSB). The flip flops in the Figure are negative edge triggered which is shown by notch and a bubble at the input of clock in input. All the J and K inputs are tied to +Vcc (HIGH) which means that each flip flop toggles with a negative transition at it clock input. The output of each flip flop is used as the clock input for the next flip flop. Since the clock applied to flip flop's is not in synchronism with the CLK applied to FF A it is called asynchronous counter.

Working Principle (Truth Table5.12):

a) Clear all Q outputs of each FF by momentarily closing switch S and then releasing which give zero at preset inputs that clears all outputs

$Q_D\ Q_C\ Q_B\ Q_A$=0000

b) When the first clock pulse comes then at its negative edge or trailing edge FF A toggles from 0 to 1. Since 0 to 1 is positive trigger pulse for FF B it does not respond because it is negative edge triggered and responds only when FF A's output goes from 1 to 0. Therefore the output at the end of first clock cycle is

$$Q_D\ Q_C\ Q_B\ Q_A = 0001$$

c) When second clock cycle comes then at it negative edge again FF A toggles (as FF A toggles every time clock pulse comes) from 1 to 0. Since this is negative trigger pulse for FF and now it responds by toggling from 0 to 1. The change at the output of FF B is from 0 to 1 which is positive going this change does not triggers FF C as it responds only when output of FF B goes from 1 to 0 therefore the output at the end of second clock cycle is

$$Q_D\ Q_C\ Q_B\ Q_A = 0010$$

d) When third cycle comes FF A toggles from 0 to 1, this positive transition makes no change in FF B and the output of FF B remains 1. Therefore, at the end of third clock cycle the output is

$$Q_D\ Q_C\ Q_B\ Q_A = 0011$$

e) When the fourth clock cycle comes FF A toggles 1 to 0 this negative change triggers FF B and its output also toggles from 1 to 0, this is turn toggle FF C from 0 to 1 as each flip flop toggles when the previous FF's output goes from 1 to 0. Now at the end of the fourth clock cycle the output is

$$Q_D\ Q_C\ Q_B\ Q_A = 0100$$

Like this, the counting advance upwards up to $Q_D\ Q_C\ Q_B\ Q_A = 1$ and when 16^{th} clock pulse is applied all FF's toggles and the O/P output becomes 0000. The truth table shows the count sequence of the counter from 0000 to 1111.

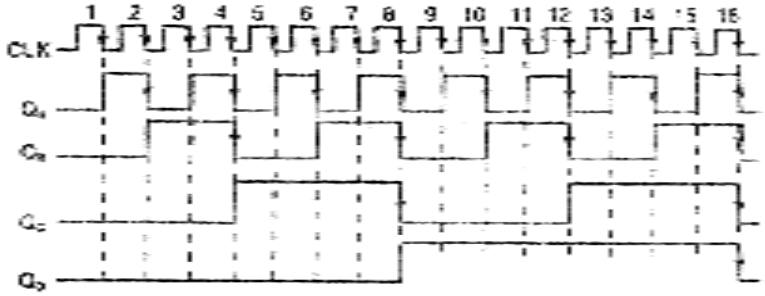


Figure 5.19 Waveform of Ripple Up Counter

CLK	Q_D	Q_C	Q_B	Q_A
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
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Table 5.12: Truth Table for Asynchronous Ripple Counter

The wave from shows the action of the counter as the clock advance. Every time there is negative transition of clock FF A (LSB Counter) toggles and the other toggles when the previous FF's output goes from HIGH to LOW. Thus the trigger moves through the flip flop like a ripple in water.

It can be seen from the waveform that the input clock is divided by each flip flop by a value of 2. Suppose clock applied to FF A has the frequency of 16 kHz. The FF A will divide this clock by 2 and it output will generate a clock frequency of 8kHz now the input of FFB is a clock of 8kHz, it will divide it by 2 and thus the output of FF B will produces a clock of 4 kHz. FF C will further divide thus clock and its output

will generate a clock cycle of 2 kHz. Lastly, FF D will further divide this 2kHz by 2 and it output will generate a clock frequency of 1kHz. This division of frequency can easily be seen from the waveform. Figure 5.19 shows the waveform.

Down Counter

Here, little modification has been made from the previous counter, in stead of applying the clock from the Q output here we apply from the 1 output and the output is taken from Q_C Q_B and Q_B Q_A. The Figure 5.20 shows the circuit arrangement.

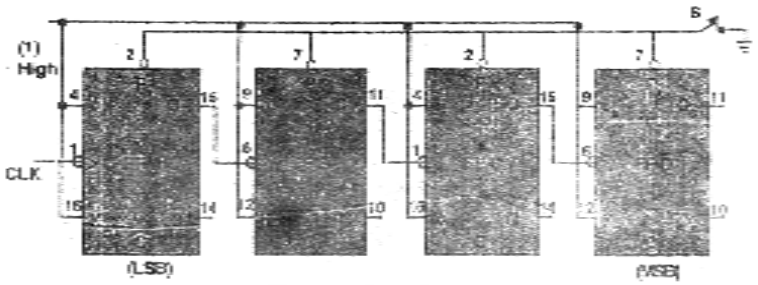


Figure 5.20 Down Counter

Working Principle (Table 5.13):

a) The switch S is momentarily pressed for few seconds which gives 0's at the preset input, which presets the output to

Q_D Q_C Q_B Q_A=1111

b) When first clock pulse is applied then its negative edge FF A toggles Q_A =0 and Q_A and 'Q_A =1. Since 'Q_A has changed from 0 to 1 which is positive clock for FF B it does not toggles and the output of counter at the end of first clock cycle is

Q_D Q_C Q_B Q_A=1110

c) When second clock pulse is applied than FF B toggles and its output becomes Q_B =0, the reason for that is its clock input receives negative transition of the clock as produced by the output of FF A (FFA toggles every time when clock pulse arrives) and the rest of the flip flops do not respond. Therefore, the output at the end of the second clock pulse is

Q_D Q_C Q_B Q_A=1101

d) When third clock pulse arrives only FFA toggles and the rest of lip flops remain unchanged. Therefore, the output at the end of third clock pulse is

$Q_D\ Q_C\ Q_B\ Q_A$=1100

e) When fourth clock pulse arrives at its negative edge FF A toggles and its output becomes Q_A =1 and 'Q_A changes from 1 to 0 which is negative going pulse for the FF B. Due to which the state of FF B is changed from 0 to 1 and that of 'Q_B 1 to 0 is connected to the clock input of FF C it toggles on the negative transition from 1 to 0.

$Q_D\ Q_C\ Q_B\ Q_A$=1011

Like this we go on applying the clock pulses the output go on decreasing by one count and at the end of 15th clock pulse the output becomes $Q_D\ Q_C\ Q_B\ Q_A$=0000. And when 16th clock pulse is applied the output becomes $Q_D\ Q_C\ Q_B\ Q_A$=1111. The count sequence of a down counter is shown in the truth Table 5.13.

CLK	Q_D	Q_C	Q_B	Q_A
0	1	1	1	1
1	1	1	1	0
2	1	1	0	1
3	1	1	0	0
4	1	0	1	1
5	1	0	1	0
6	1	0	0	1
7	1	0	0	0
8	0	1	1	1
9	0	1	1	0
10	0	1	0	1
11	0	1	0	0
12	0	0	1	1
13	0	0	1	0
14	0	0	0	1
15	0	0	0	0
16	1	1	1	1

Table 5.13 : Truth Table for Down Counter

Up Down Counter

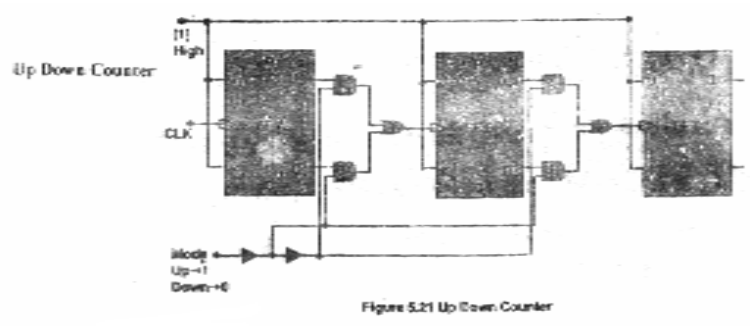


Figure 5.21 Up Down Counter

Here (Figure 5.21) additional hardware is connected between the outputs of FF A and clock input of FF B and the output of FF B and input of FF C in order to make the counter an up-down counter.

Up Counter

To make the circuit work as Up-Counter MODE input is kept at 1, this enables all A AND gates and disables all B AND gate. So any change (negative transition0 occuring at the Q outputs is transmitted to the next FF's clock input and counter works as normal UP-Counter as explained in asynchronous counters.

Down counter

To make the circuit work as Down counter 0 is given at the MODE input which enables all B AND gate and disables all A AND gate. So any change at the 1 outputs is transmitted to the next FF's clock input and the counter works as normal down counter as explained in case of asynchronous counters.

Mod 10 Counter or Decade Counter or BCD 8421 Counter

A counter that recycles in 10 pulses is called a MOD-10 or DECADE or BCD COUNTER. There are many ways to design this counter but we will discuss the method, which resets all FFs after a desired count is reached. The circuit shown in Figure 5.22 is a binary ripple counter that could count up to 16 but some modification are made which allows the circuit to count up to 9 and on the application of 10^{th} pulse the counter is reset to 0000.

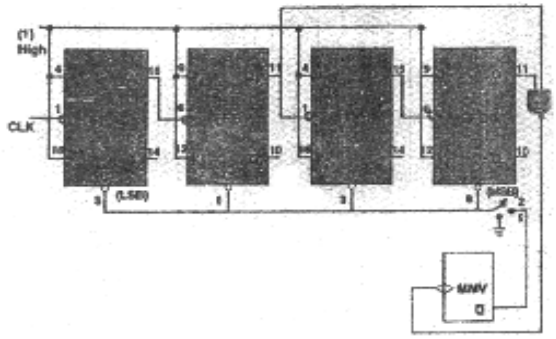


Figure 5.22 Decade Counter

Working Principle (Table 5.14):

a) Switch S is momentarily connected to point 1 and then kept permanently at 2. Which RESET the counter to

$$Q_D\ Q_C\ Q_B\ Q_A=0000$$

b) Input pulses advances counter as binary up counter up to 9 (1001) as a normal binary ripple counter as shown in the truth table of decade counter.

c) The next count pulse advance the count to 10 (Count=1010) but since the Q_D and Q_B outputs are given to the NAND gate which provides 0 at its output providing a 1 to 0 logic change to trigger the monostable multivibrator (MMV). This provides a short pulse to reset all counter's. The 'Q output signal is sued since it is normally high and goes low during the one shot timing period, the flip flop in this circuit being reset to $Q_D\ Q_C\ Q_B\ Q_A=0000$ by low signal level. The output of the NAND gate could have been directly connected to the CLEAR input to reset the Counter, but this is not reliable because propagation delay from CLEAR input to Q output of the FF varies from one flip flop to other. For example if Q_D out put takes longer time to reset than $Q_{A,}$ then the output of the NAND gate again goes to 1 when Q_A returns to 0, this make CLEAR =1 with the result that FF D will not reset. This problem makes importance when the counter outputs are loaded unevenly. So the use of one shot (Mono Stable Multivibrator) memorises the output of the NAND gate at the 10^{th} pulse and eliminates the problem.

CLK	Q_D	Q_C	Q_B	Q_A
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
10	1	0	1	0
	0	0	0	0

Table 5.14: Truth Table BCD Decade Counter

These type of circuits has a disadvantage that the circuit uses a single shot unit and special timing adjustment on pulse duration. To overcome this difficulty direct reset counters are used.

Direct Reset Decade Counters

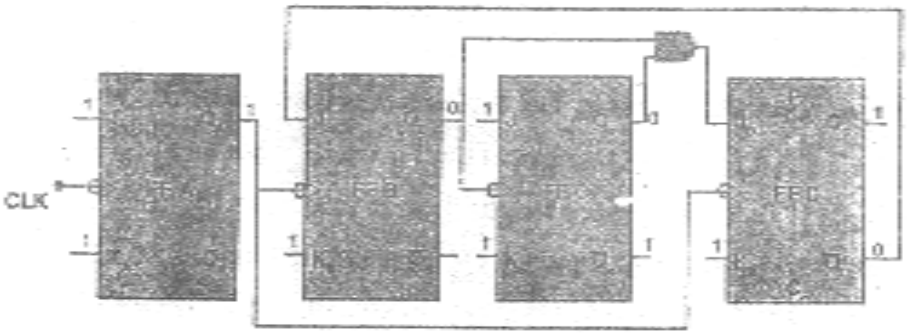


Figure 5.23 Direct Reset Decade Counter

Working Principle:

a) FFM A toggles at each input pulse.

b) FF B toggles when FF A output change from 1 to 0 except when Q_D=1and 'Q_D=0 because during this condition FF B output is Q_B =0 and its input J_A and K_A are 0 and respectively.

c) FF C toggles when FF B output goes from 1 to 0.

d) FF D is reset each time FFA goes from 1 to 0 except when FF B and FF C are both are logical 1.

Let us see the operation of the circuit during 9th & 10th clock pulses. After 9th clock pulses hits the output are

$$Q_D\ Q_C\ Q_B\ Q_A=0000$$

These conditions are shown in the Figure 5.23 for better understanding of the condition. Now the 'Q_D has become 0 which is given to the J_B input as the output of FF B i.e. J_B =0 K_B =1 of FF B and FFC are 0. When 10th clock pulse arrives then FF A toggles from 1 to 0 this shifts the 0 input of J_B to the output i.e. no change in the output of Q_B due to which FF C do not gets any clock at its clock input and its output remains 0 and the input of FF D i.e. J_D =0 and K_D =0 are passed to the output as FF D gets the clock pulse when FF A output changes from 1 to 0. This makes all the flip flops to RESET to $Q_D\ Q_C\ Q_B\ Q_A$=0000. Like MOD-10 counter MOD-9 counter can be easily realised.

Mod 5 Counter

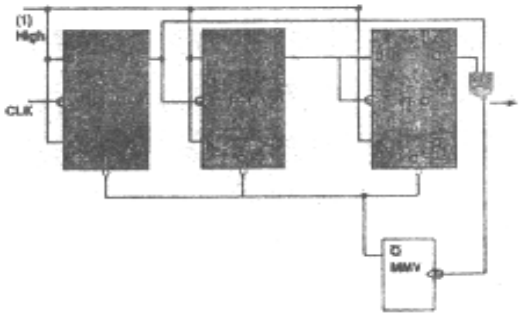


Figure 5.24 MOD 5 Counter

For MOD-5 counter we require 3-FF's. It works as normal ripple counter up to 4th clock pulses and on the 5th clock pulse all FF's are RESET to ZERO the working of the counter is similar to that of MOD –10 counter, this is shown in the truth table given below (Table 5.15).

CLK	Q_D	Q_C	Q_B	Q_A
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
	0	0	0	0

Table 5.15: Truth Table for MOD 5 Counter

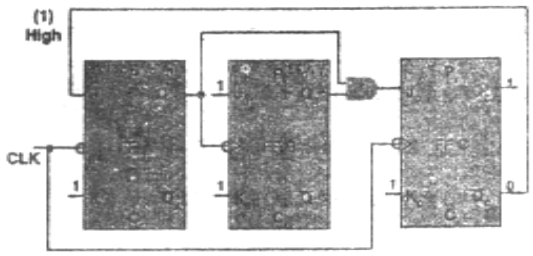


Figure 5.25 Direct Reset MOD-5 Counter

When 4th clock hits flip flops the output condition is Q_C Q_B Q_A=000. Q_C is connected to J_A input. So input of FF A is J_A =0, K_A =1, FF B acts as normal TFF and the input to FF C is J_C and K_C =1. When 5th clock pulse comes 0 at the input at FF A is passed to output.

Since there is no change at the of FF A, FF B do not respond and its output remains at 0 FF C input J_c =0, K_c =1 are passed to the output and thus MOD 5 counters output becomes - Q_C Q_B Q_A=000.

Ring Counters

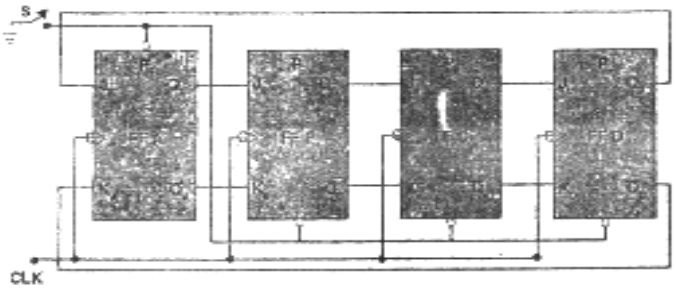


Figure 5.26 Ring Counter

Ring Counter is the simple serial shift register in which a single bit shifts from one flip-flop to the another in the form of a ring as shown in Figure 5.26.

Working Principle (Table 5.16):

1. The switch S is pressed which clear the FF B, FF C and FF D and sets FF A.

   $Q_D Q_C Q_B Q_A$=0001

2. Before the 1st clock hits input to FF B is 1 to rest FF's having 0 inputs, at the end of clock pulse the state of the FF's is

   $Q_D Q_C Q_B Q_A$=0010

3. The FF C has 1 input and the rest of the flip-flop have 0 inputs and when the succeeding clock arrives, the input data's are stored in the respective flip-flops.

   $Q_D Q_C Q_B Q_A$=0100

4. Now the FF D has 1 input and the rest of the flip-flops have 0 input and when 3rd clock pulse comes this 1 is shifted into the FF D.

   $Q_D Q_C Q_B Q_A$=1000

5. Since the Q_D output is directly connected to the input of FF A. When 4th clock pulse comes this 1 is shifted into the FF A and the process repeats till the clock pulses are applied.

CLK	Q_D	Q_C	Q_B	Q_A
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	1	0	0	0
4	0	0	0	1
5	0	0	1	0
6	0	1	0	0
7	1	0	0	0

Table 5.16: Truth table for Ring Counter

Self- Correcting Ring Counter

Sometimes due to false triggering the count sequence may be disrupted and we might not get the actual sequence of the ring counter. To avoid such conditions, self-correcting ring counter is used which is modified form of Ring counter.

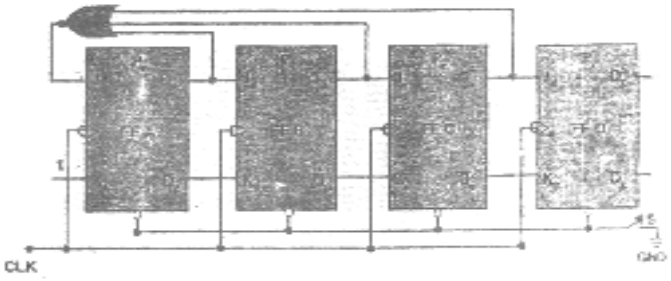


Figure 5.27 Self Correcting Ring Counter

Working principle:

1. Press the switch S to GND and then keeping it high initially resets all the FF;s.

   $Q_D Q_C Q_B Q_A$=0000

2. The Q_A Q_B and Q_C output are connected to the input of NOR gate whose output become 1, then the input of FF A is now 1 and reset all other FF's have 0 inputs.

3. When the 1st clock is applied the 1 the input of FF A is transferred to the output.

$Q_D Q_C Q_B Q_A$=0001

4. Now the input of NOR is 1 and 0 and 0 which give 0 at the output (if any input of NOR gate is high the output is always low) which is the input to FF A, when 2nd clock pulse comes the 1 at the input of FFB, i.e. 1 is transferred to its output.

$Q_D Q_C Q_B Q_A$=1000

and in this way the process goes on like normal ring counter.

Advantages

No FF's is set before the first clock comes.

If by change due to false triggering or other reasons two FF's are set to 1 then in simple Ring counter, this extra one will keep on rotating in the loop and there is no option to correct it while counting is going on but on self- correcting ring counter this problem is eliminated after 2 or 3 cycles.

Johnson or Twisted Ring Counter

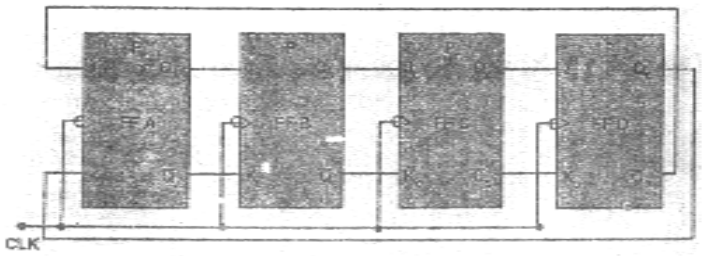


Figure 5.28 Johnson Ring Counter

In this circuit the output of last FF in cross connected to FF A, this technique is called inverse feedback due to each the sequence obtained is very different from the other count sequence which looks like as twisted one (Figure 5.28).

Working principle (Table 5.17):

1. Initially all FF's are RESET.
2. Now the input of FF a has 1 input and the rest flip-flops have 0 input on the

application of 1st clock pulse these inputs are stored in the respective FF's and can be seen at the outputs.

$Q_D Q_C Q_B Q_A$=0001

3. Again FF A is 1 and FF B is 1 rest have 0 input, when 2nd clock pulse comes these inputs are stored by the respective FF's and the count sequence is:

$Q_D Q_C Q_B Q_A$=0011

4. Now the input to FF A and FFB is 1 rest have 0 input, when 3rd clock pulse comes then input are stored in the respective FF's and the count sequence is

$Q_D Q_C Q_B Q_A$=0111

5. All flip-flops now have 1 input and when next clock pulse comes these inputs are stored in the respective FF's

$Q_D Q_C Q_B Q_A$=1111

6. Now the input to FF A has 0 input and the rest other has 1 input thus on the arrival of the 5th clock pulse the outputs are stored in the input of the respective flip-flop

$Q_D Q_C Q_B Q_A$=1110

The Truth Table 5.17 shows the count sequence of the Johnson Counter.

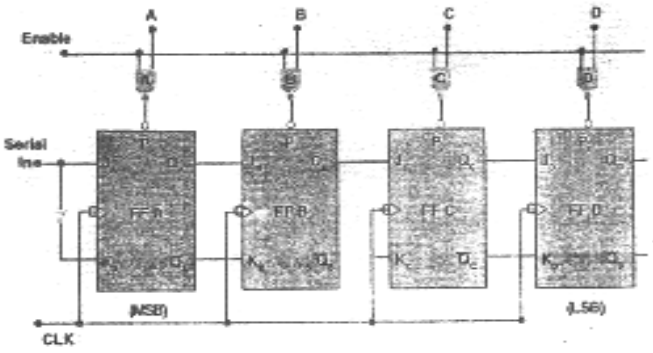


Figure 5.29 Register

CLK	Q_D	Q_C	Q_B	Q_A	*Decimal Equivalent*
0	0	0	0	1	0
1	0	0	0	1	1
2	0	0	1	1	3
3	0	1	1	1	7
4	1	1	1	1	15
5	1	1	1	0	14
6	1	1	0	0	12
7	1	0	0	0	8
8	0	0	0	0	0

Table 5.17: Truth Table for Johnson Ring Counter

5.4 Registers

A group of flip flops used for temporary data storage is known as registers. The features of registers are listed below:

- DFF-is used for storing a single bit data.
- To store n-bit data n-registers are required.
- Data can be entered in the register serial or parallel form and can be taken out serially or in parallel form.
- Shift registers are extensively used in arithmetic operations like multiplication (left shift) and division (right shift) of binary numbers.

The four basic type of registers are

- Serial In Serial Out (SISO)
- Serial In Parallel Out (SIPO)
- Parallel In Serial Out(PISO)
- Parallel In Parallel Out (PIPO)

Serial In Serial Out

Working principle:

1. Initially all FF's are reset by apply a clear pulse at clear input.
2. LSB of data 1011 is applied to the serial input terminal.
3. When the first clock pulse comes at the negative edge of the clock this LSB bit i.e. 1 is stored into the FF A and the register outputs look like

   $Q_D Q_C Q_B Q_A$=1000
4. Now input to FF B is 1 because it is directly connected to the output of FF A and when second clock pulse comes input of FF A is stored in it and also the previous output of FF A which was 1 is stored in FF B.

   $Q_D Q_C Q_B Q_A$=1100
5. Input to FF C is 1 input to FF B is 1, now third data i.e. 0 is applied to FFA's input and when third clock arrive, these data are stored in two respective FF's as shown below

   $Q_D Q_C Q_B Q_A$=0110
6. The last input 1 is applied to the FF A and input to FFC is 1, input to FF B is 0 now when fourth clock comes these input data are stored in the respective FF's.

   $Q_D Q_C Q_B Q_A$=1011

This method of storing the data is called **Serial Data Storage.** Further if we apply fourth clock pulses. The data is shifted out of the right end of the register and lost after four clock times.

Serial In Parallel Out

In this register the data is shifter serially into the register as explained in case of SISO but the data is shifted out in parallel form. In order to get the data out in parallel form it is necessary to have all data bits available as the output at same time. This is done by connecting four parallel wires to get from $Q_D Q_C Q_B Q_{A.}$

Parallel In Serial Out

To enter the data (1011) in parallel form, inputs are given at A,B,C,D terminal. Initially, all registers are cleared, therefore, when we have to shift the data into the register we apply to ENABLE pulse at ENABLE input, now the output of

A-NADN gate is $\overline{1.1} = 0$

B-NAND gate is $\overline{0.1} = 0$

C-NAND gate is $\overline{1.1} = 0$

D-NAND gate is $\overline{1.1} = 0$

To clear input of FF A is 0so the output of FF A will be 1, the clear input of FF B is 1 which keeps the output of FF B at 0 the clear input of FF C is 0 so the output of FF C becomes 1 and the clear inputs of FF D which sets the output of FF D to 1. The register contents are

$$Q_A\, Q_B\, Q_C\, Q_D = 1011$$

By applying a single pulse we have stored all four bit sat same time this is called parallel storing data. To retrieve the serially we apply four clock pulses at the CLK input and data is out of the register.

Parallel in Parallel Out

The data is stored in parallel form as described in case of PISO and retrieved at the output in parallel form through four wires connected at each output.

Right – Left Shift Register

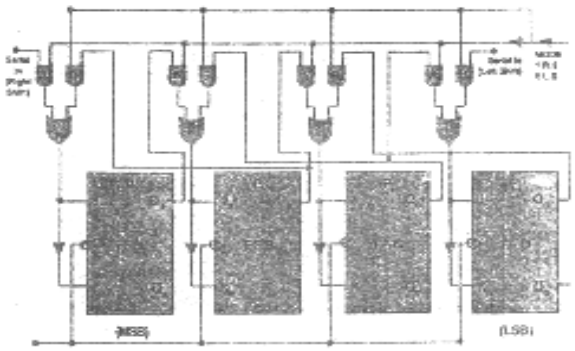


Figure 5.36 Right - Left Shift Register

Right-Left Shift Register find its extensive use in arithmetic unit for multiplication and division purpose. When the number has to be divided then with the shift of data toward right divides the number by two with each shift and for multiplication the data is left shifted, with each shift of the data towards left the number is multiplied by two.

Right Shift

To shift the data towards right MODE CONTROL line is kept HIGH which enables all A AND gates and disables all B AND gates. Let data 1011 is to be shifted towards right.

1. We give 1st bit of data i.e. 1 at the SERIAL IN terminal, since all A AND gates are enabled this data reaches to the input of the FF A, when negative edge of th first clock comes this 1 is stored in FF A and is received at QA output. The state of the output is

   $Q_D Q_C Q_B Q_{A.} = 1000$

2. The QA output passed by A2 AND and respective OR gate, the input of FF B becomes JA=1 and KA=0. Now we give the next data i.e. 1 at the SERIAL IN terminal which reaches to the input of FF A, when negative edge of 2nd clock hits the input data FF A and FF B are stored in the respective FF's

   $Q_D Q_C Q_B Q_{A.} = 1100$

3. The output of FF A and FF B reaches to the input of FF B and FF C respectively after passing through their respective AND and OR gates, now the input to FF B is J_B=1 K_B=0, FF C is J_C=1, K_C=0. Now the third data bit i.e. 0is applied at the SERIAL IN Terminal which ultimately reaches to the input of FA A therefore, when negative edge of 3rd clock comes all the inputs are stored in the respective FF's. The output state of the flip flop are

$$Q_D\, Q_C\, Q_B\, Q_{A.}=0110$$

4. The output of FF A reaches to the input of FF B through A2 AND and OR gate, the output of FF B reaches to the input of FF C through A3 AND gate and respective OR gate and finally the output of FF C i.e. 1 reaches to the input of FF D through A4 AND gate and respective OR gate.

   When last data bit 1 is applied to the SERIAL IN reaches at the input of FF A. When negative edge of 4th clock arrives all the inputs at each FF is stored in the respective flip flop.

$$Q_D\, Q_C\, Q_B\, Q_{A.}=1011$$

This is right shifting of data, the data can be taken out in parallel form through four parallel wires or serially from Q_D by applying four clock pulses.

Left Shift

To shift the data towards left the MODE CONTROL line is placed at ZERO which enables all B AND gates and disables A AND gates. The data is applied from one of the pin of B_4 AND gate shown in Figure 5.30.

For example the data 1001 has to be shifted towards left then following operations follows:

1. Giving 1 at the SERIAL IN (Left Shift) terminal, which reaches to the input of FF D as B4 AND gate OR gates are enable therefore, when the negative edge of the 1st clock comes the 1 at the input is stored in the FF D, the output of FF D is connected to B_3 AND gate which passes this output to the input of FF C.

2. The second data i.e. 0 at the SERIAL IN (Left Shift) terminal reaches to the input of FF D and when the 2nd clock pulse arrives then these data's at the input of the FF D and FF C are stored in the respective FF's, the register contents are:

$Q_A Q_B Q_C Q_{D.}$=0010

3. The output of the flip flops reaches to the next flip flop, third data i.e. 0 is given at the input terminal which reaches to the input of FF D and when the third clock pulse hits the FF C are stored in the respective FF's, the register contents are:

$Q_A Q_B Q_C Q_{D.}$=0100

4. The output Q_B is connected to one of the pin of B_1 AND gate so the 1 at the output of Q_B is passed to the input of FF A, like wise all other data are passed to their next FF's the last data bit which is 1 is given to the input terminal which reaches at the input of FF D and when the 4th clock pulse comes at the negative edge of it the data present at the input to each FF are stored in them

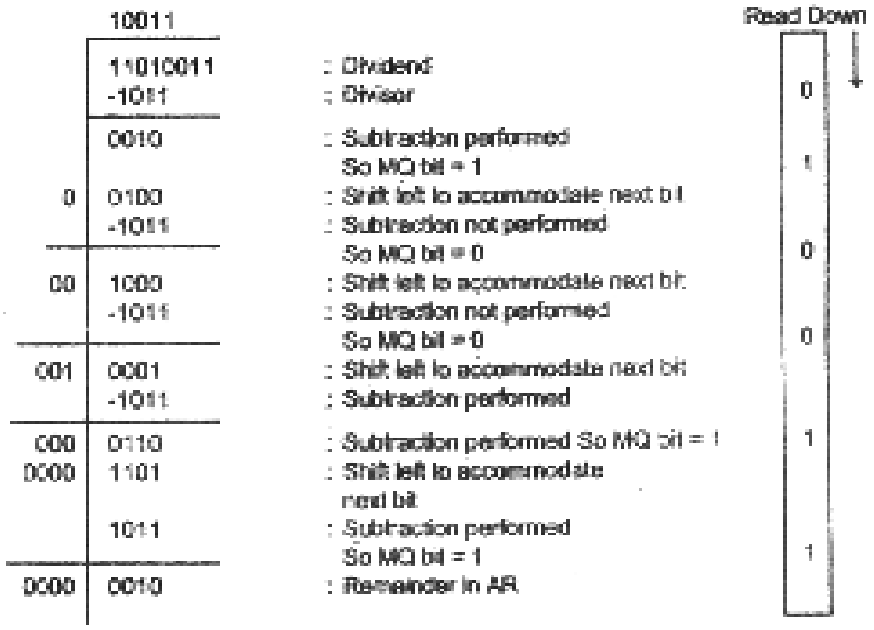
$Q_A Q_B Q_C Q_{D.}$=1011

The data can be collected in parallel by four parallel wires or serially from Q_A by applying four clock pulses. This is left shifting of data.

Assignment

Show the machine division of $(11010011)_2$ by $(1011)_2$

Solution : In this example we tried to show how a binary division can be performed in a manner similar to multiplication using Register. In binary division repeated subtraction and shift left is used.



The answer is in quotient: 10011 with remainder 0010 in AR.

The basic cycle can be summarized as follows:

1. Shift the data into subtractor unit by applying shift pulses (SP).
2. If subtraction is possible MQ=1, if not MQ=0.
3. Shift left the subtracted value left by one place to accommodate next bit.
4. If all dividend data is accessed, if YES output MQ data, else Go To number 1.

❖ ❖ ❖

: 6

Author :
Dr. Dharminder Kumar

Vetter :
Dr. Devendra Mohan

APPLICATIONS OF LOGIC CIRCUITS - I

- **Adders (Half and Full)**
- **Subtractors (Half and Full)**
- **Analog-to-Digital and Digital-to-Analog Convertors**
- **Multiplexers**
- **De Multiplexers**

Prelude

Our imagination has been captured by computers and modern digital electronic devices such as calculators, watches, keyboard, camera, television etc., probably because these machines perform arithmetic tasks with such fantastic speed and accuracy. This chapter deals with some logic circuits that can add and subtract, of course, the adding and subtracting is done in binary. Regular basic gates are wired together to form adders and subtractors. Basic adder and subtractor circuits are combinational logic circuits, they are commonly used with various latches and resistors to hold data.

In the Central Processing Unit (CPU) of a computer, arithmetic is handled in a section commonly called the Arithmetic Logic Unit (ALU). This section within the CPU can usually add and subtract, multiply and divide, complement, compare, shift and rotate, increment and decrement, and perform logic operations such as AND, OR and EXOR. A short description of ALU Chip is presented in Chapter VII in the context of ALU chip.

Most of the data entering of leaving a digital signal processing unit has digital informations. Many digital systems, however, have analog inputs that vary continuously between two voltage levels. In many applications of digital systems, such as control, communication, computers, instrumentation etc., the signals are not available in the digital form and, their signal to digital signal is referred to as an analog-to-digital conversion. The system used for realising this conversion is referred to as an analog-to-digital converter (A/D Converter or ADC) or Encoder.

The output of the system may be required to be in analog form and, therefore, the digital output need to be converted back to the analog form. The reverse process is called digital-to-analog conversion and the digital system is referred to as digital-to-analog conversion and the digital system is referred to as digital-to-analog converter (D/A Converter or DAC) or Decoder. In this chapter, the interfacing of analog devices to digital systems are projected. A block diagram representing a digital system with analog input and analog output (called a hybrid system) is represented in Figure 6.1.

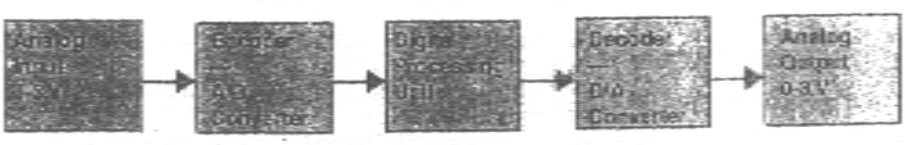
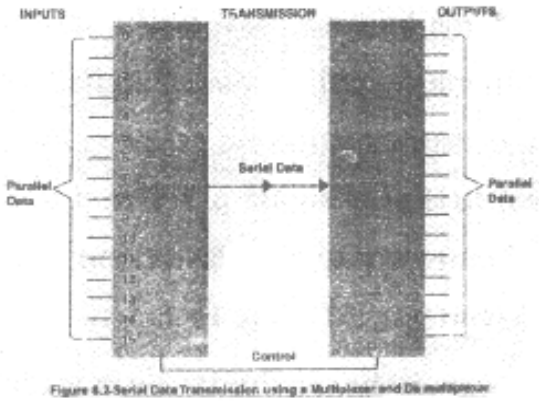


Figure 6.1 A Digital System with Analog Input and analog Output

The data in digital systems is transmitted directly through wires and PC keyboards. Many times bits of data must be transmitted from one place to another. Sometimes the data must be transmitted over telephone lines or cables to points far away. If all the data were sent at time over parallel wires, the cost and size of these cables would be too expensive and large instead, the data is send over a single wire in serial form and reassembled into parallel data at the receiving end. The devices used for sending and receiving serial data are called multiplexers (MUX) and de-multiplexer is shown in Figure 6.2.



The most important characteristic that a digital system has over an analog system is its ability to store data for short or long periods. The availability and use of memory and digital storage devices has fueled what is presently known as '**information revolution'.** The sub-system of a digital-processing unit, which provides the storage facility, is referred to as the '**memory**'.

The entire Internet system is dependent on the transfer of data from one storage/ memory device to another. Of course, computers and telecommunication systems are dependent on large amounts on digital storage. With unprecedented developments in semiconductor technology, it has become possible to make semiconductor memories of various types and sizes. The flip-flop forms a basic memory cell in some semiconductor memories. A simple shift registers latches and counters also uses the flip-flop as a temporary memory. Description of several more types of semiconductor memory cell (SRAM, DRAM, ROM, EPROM, EEPROM, and FLASH memory) will be presented in this chapter.

Any digital system require a display unit as a input or output indicator. The Light Emitting Diode (LED) is perfect for this job because it operates at low current and voltages. The LED actually generates light, where as the Liquid Crystal Display (LED) simply controls available light. The LCD has gained wide acceptance because of its very low power consumption. It is also well suited for use in sun light or in other brightly lit areas. The LCD is also suited for more compact displays than just seven-segment decimal. A very common output device used to display decimal numbers

is the seven-segment display. In this chapter the performance of various display unit using LED and LCD will be discussed at length.

ADDERS (HALF AND FULL)

Half Adder

Addition is the most basic arithmetic operation of any computer system. To design adder circuit, the following rules of binary addition are used.

0+0=0

0+1=1

1+0=1

1+1=10

The first three operations produces sum whose length is one digit, but when augend and addend bits are equal to 1, the binary sum consists of two digits.

The higher significant bit of these results is called a CARRY. When the augend and addend numbers content more significant digit, the carry obtained form the addition of two bits is added to the next higher order pairs of significant bits. A combinational circuit that performs the addition of two bits is called a half adder. A half adder is a circuit that has two inputs, A and B and two outputs, SUM (S) and CARRY©. The block diagram and Truth Table for a half adder are shown in Figure 6.3 and Table 6.1

Inputs		Outputs	
A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Table 6.1

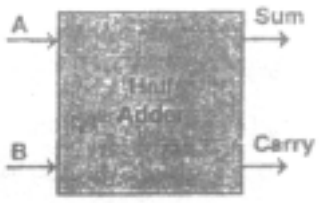


Figure 6.3 Half Adder

The sum output has a Truth Table identical to the Exclusive OR, and the carry output has the truth-table identical to AND gate. The logical expression for S and C outputs are given by

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

$C = AB$

The realisation of an half adder using gates is shown in Figure 6.4. The logic diagram of an half adder using an AND gate and two NOR gates are represented in Figure 6.5 (a) and Figure 6.5(b) represents an half adder using AND, OR and NOT gate.

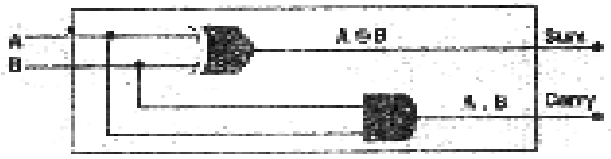


Figure 6.4 Logic Diagram of a Half Adder

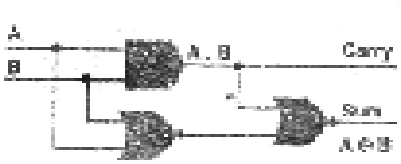
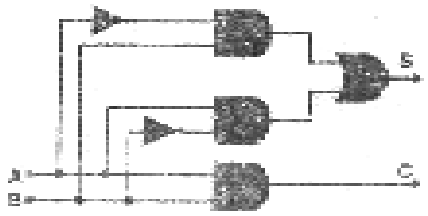


Figure 6.5 (a) Using an AND Gate and two NOR Gates to Construct a Half Adder



(b) Logic Circuit using AND, OR and NOT Gate

Full Adder

In order to perform multi bit addition an half-adder has no provision to add a carry coming from the lower order bits. This difficulty is overcome by adding a third input terminal. The circuit is used to add An, B_n and C_{n-1}. Here, A_n and B_n are the Nth order bits of the number A and B respectively. Cn-1 is the carry generated from the addition of $(n-1)^{th}$ order bits. A combinational circuit that performs the addition of three bits (two significant bits and previous carry) is called a Full Adder. Therefore, a full adder adds three inputs together, A_n, B_n, and a carry from a previous addition (C_{n-1}) and outputs a sum (S_n) and carry (C_n). The block diagram and truth table for full adder are shown in Figure 6.6 (a) and Table 6.2. Figure 6.6 (b) represents

the construction of full adder using X-OR gate. The logical expression for S and C outputs are

$$S_n = \overline{A}_n.B_n.C_{n-1} + \overline{A}_n.B_n.\overline{C}_n - 1 + A_n.\overline{B}_n.\overline{C}_n - 1 + A_n.B_n.C_{n-1}$$

$$C_n = A_n.B_n + A_n.C_{n-1} + B_n.C_{n-1}$$

Inputs		Output		
A_n	B_n	C_{n-1}	S_n	C_n
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

Table 6.2

The NAND-NAND realisations of S_n and C_n in a full adder circuit is shown in Figure 6.7. Another representation using Exclusive-Or gate, AND and NOR gate are shown in Figure 6.8. The full adder in Figure 6.9 (a) and 6.9 (b) is constructed from two half adders and an OR gate.

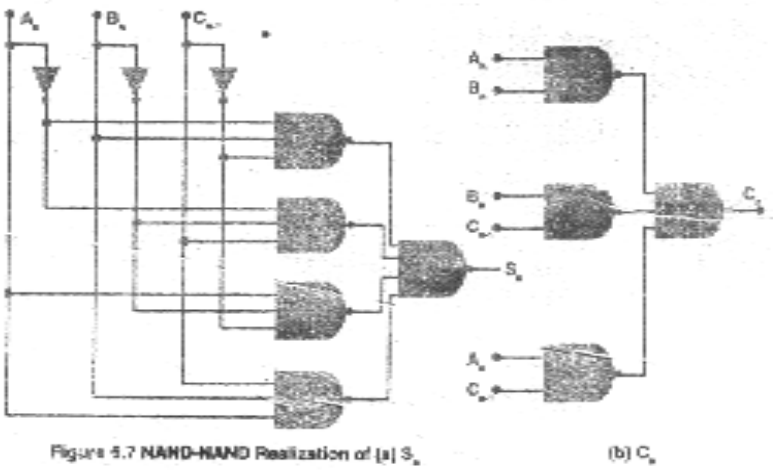


Figure 6.7 NAND-NAND Realization of (a) S_n (b) C_n

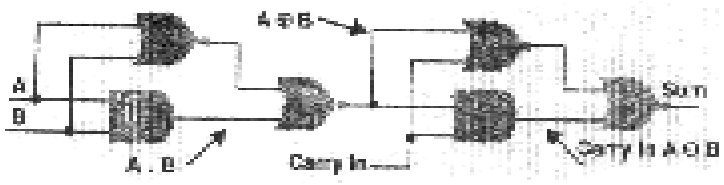
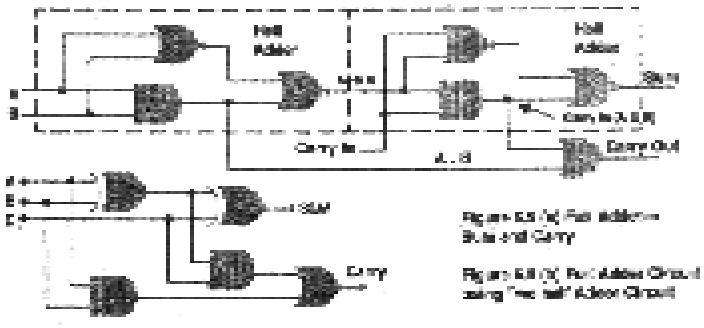


Figure 6.8 A Second Construction for a Full Adder Sum



SUBTRACTORS

Half-Subtractor

The half-subtractors and full subtractors are very similar to that of half-adders. A logic circuit for the subtraction of B (Subtrahend) from A (Minuend), where A and B are 1 –bit numbers is referred to as a half – substractor. The rules for binary subtraction are given in Table 6.3. Here A and B are the two inputs and D (Difference) and C (Burrow) are the two outputs. The Truth table of a half – subtractor is shown in Table 6.4.

Minuend	Subtrahend	Difference	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Table 6.3 : Rules of Binary Substractions

Inputs		Outputs	
A	B	D	C
0	0	0	0
0	1	1	1
0	0	1	0
1	1	0	0

Table 6.4: Half Substractor

The logical expressions for D and Care obtained as

$C = \overline{A}.B$

The block diagram of a half – subtractor is represented in Figure 6.10. The realisation of half-subtractor using gates is shown in Figure 6.11 (a). Figure 6.11 (b) shows the construction of an half-subtractor using AND, OR and NOT gate.

$D = \overline{A}B + A\overline{B} = A \oplus +B$

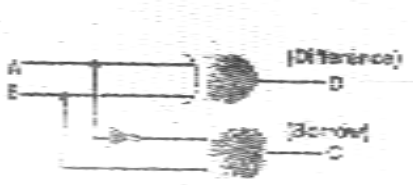


Figure 6.11 (a) Logic Diagram for Half Subtractor using Ex-OR, AND and NOT Gate

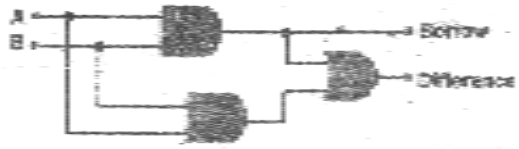


Figure 6.11 (b) Half Subtractor using AND, OR and NOT Gate

Full-Subtractor

Just like a full-adder, a full-subtractor circuit performs multibit subtractions wherein a borrow from the previous bit position may also be there. A Full-subtractor has three inputs, An (Minuend), B_n (Subtrahend) and C_{n-1} (Borrow from the previous stage) and two outputs. D_n (Difference) and C_n (Borrow). The block diagram and the Truth Table for a Full-subractor are shown in Figure 6.12 and Table 6.5

Inputs		Outputs		
A_n	B_n	C_{n-1}	S_n	C_n
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	1	1

Table 6.5

$$C_n = \overline{A}_n.B_n + \overline{A}_n.C_{n-1} + B_n.C_n - 1$$

Figure 6.13

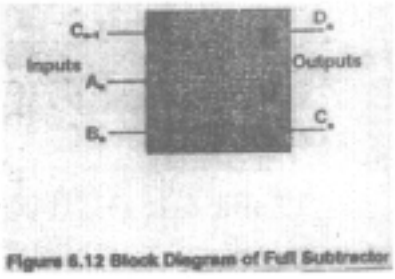


Figure 6.12 Block Diagram of Full Subtractor

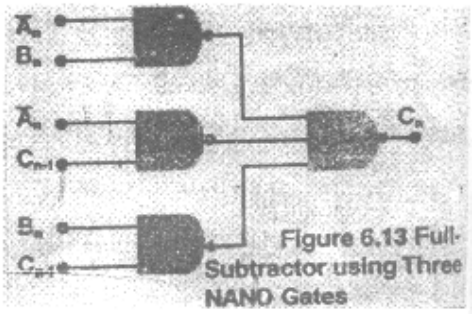


Figure 6.13 Full-Subtractor using Three NAND Gates

Like the full-adder full-subtractor can be wired using two half-subtractors and an OR gate. Figure 6.14 is a full-subtractor showing how half-subtractors are used. A logic diagram for a full-subtractor using gates is shown in Figure 6.15.

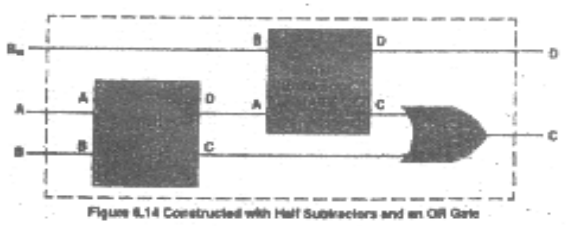


Figure 6.14 Constructed with Half Subtractors and an OR Gate

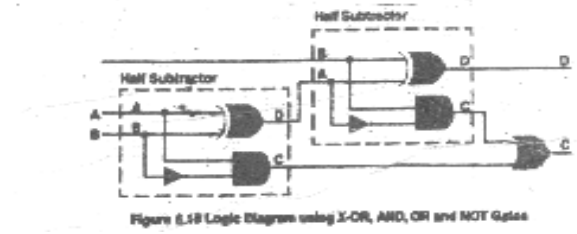


Figure 6.15 Logic Diagram using X-OR, AND, OR and NOT Gates

ANALOG TO DIGITAL CONVERTERS AND DIGITAL TO ANALOG CONVERTERS

Analog to Digital Converter

Conversion of analog signals to digital signals can be carried out by using various types of A to D converters. An analog –to-digital converter produces a binary number which is in direct proportion to an analog voltage input. For example, in an A to D converter, the input analog voltage can have any value in a range but the digital output can have only 2^N discrete values for an N-bit A/D converter.

Therefore, the whole range of analog voltage is required to be represented suitably in 2^N intervals.

The most common types of A/D converters are:

- Stair case Ramp ADC
- Linear Ramp ADC
- Successive approximation ADC
- Simultaneously type ADC
- Dual Slope ADC

The voltage comparator can be used to make a very fast A/D converter because of its speed, this type of A/D converter is often called a Flash Converter. An A/D converter using voltage comparators is shown in Figure 6.16

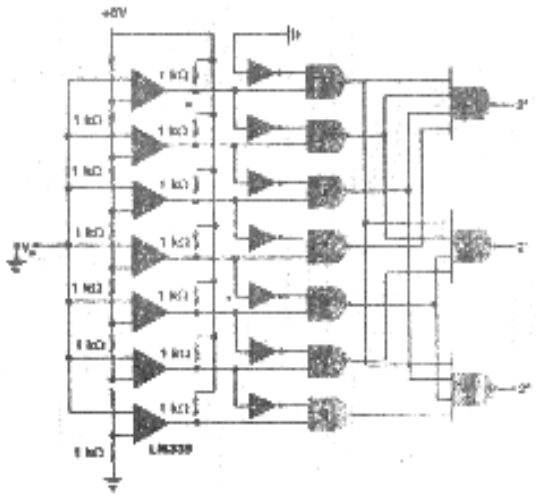


Figure 6.16 An A-to-D Converter using Voltage Comparators

Figure 6.17 shows a counter and compare A to D converter using an LM339 voltage comparator and a 2R resistor network.

To give an overall idea for the performance of an A to D converter, Staircase ramp ADC and Linear Ramp ADC is described below.

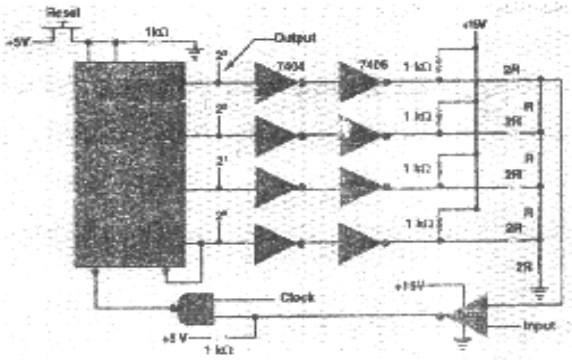


Figure 6.17 A Count-Up and Compare A-to-D Converter

Stair-Case Ramp ADC

The Signal start is given when conversion is to be started which reset the counter to all zero's hence the ladder output voltage becomes 0V. The start pulse to pass through the gate control logic circuit and counter advances thus the ladder network voltage proceeds to rise a fixed amount of step count pulse. This staircase voltage is compared to the input analog voltage by a comparator.

When the staircase voltage just exceeds the analog voltage, the comparator output provides a signal to stop the count action the digital value of the counter can then be used as output. The circuit diagram of staircase Ramp ADC has been shown in Figure 6.18. This ADC has slow conversion rate and is not very accurate.

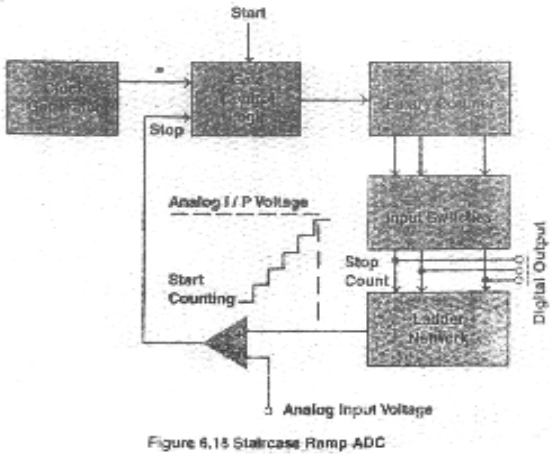


Figure 6.18 Staircase Ramp ADC

Figure 6.19 shows the circuit diagram for linear ramp ADC, this circuit is relatively simple and easy. When start pulse is given the counter is reset to all zero's and also a linear ramp waveform is initiated. The clock advances the counter till the ramp voltage is less than the input voltage and when the ramp voltage exceeds the input voltage, the comparator provides a stop count signal to the counter. The value in the counter represents the analog input in the formed of digital signal. The ADC has the disadvantage that it has poor conversion accuracy. The dual slop type ADC is better replacement for this type of ADC.

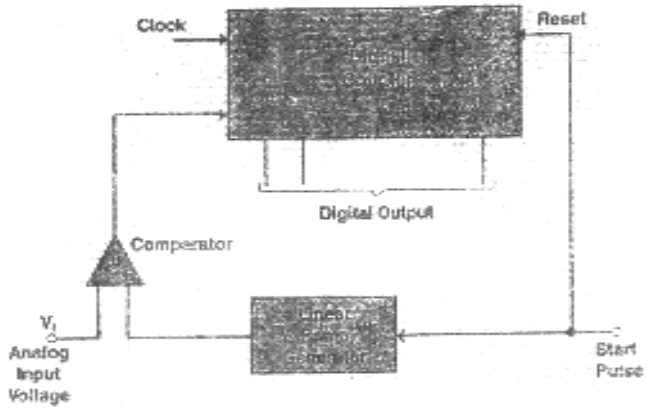


Figure 6.19 Linear Ramp ADC

Parallel-Comparator A/D Converter

One of the most simplest and fastest A/D converter is known as parallels-comparator A/D converter. In such converter, there is a rapid increase in the number of comparators with the number of bits. A 3-bit parallels comparator A-to-D converter with the analog voltage V_a to be converted into digital form is shown in Figure 6.20. Here, V_{R1}, V_{R2}…. are the reference voltages generated using the resistance network. The decoder circuit is used to convert the 7-bit digital signal to a 3-bit output. The converter gives a 7-bit output which is stored in Latches. The comparator outputs and the 3-bit digital output for each interval of the analog voltage are given in Table 6.6. The whole range of the full scale analog voltage V is divided in 8 intervals (3-bit output). Each interval is assigned a unique digital value referred to as 'quantisation'.

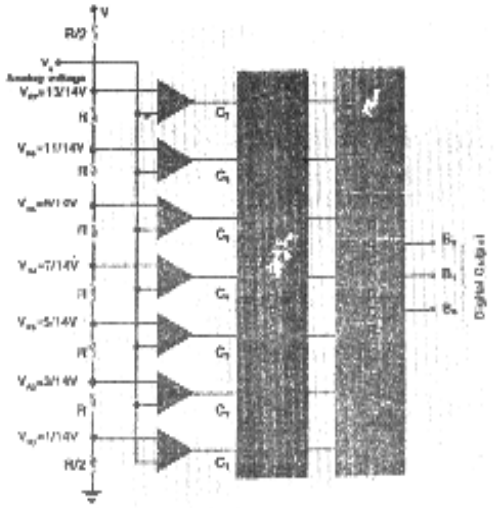


Figure 6.20 A 3-Bit Parallel-Comparator A/D Converter

Analog Input	Comparator Outputs							Digital Output		
V_a	C_7	C_6	C_5	C_4	C_3	C_2	C_1	B_2	B_1	B_0
$0 = V_a < V_{R1}$	0	0	0	0	0	0	0	0	0	0
$V_{R1} < V_a < V_{R2}$	0	0	0	0	0	0	1	0	0	1
$V_{R2} < V_a < V_{R3}$	0	0	0	0	0	1	1	0	1	0
$V_{R3} < V_a < V_{R4}$	0	0	0	0	1	1	1	0	1	1
$V_{R4} < V_a < V_{R5}$	0	0	0	1	1	1	1	1	0	0
$V_{R5} < V_a < V_{R6}$	0	0	1	1	1	1	1	1	0	1
$V_{R6} < V_a < V_{R7}$	0	1	1	1	1	1	1	1	1	0
$V_{R7} < V_a = V$	1	1	1	1	1	1	1	1	1	1

Table 6.6

Digital to Analog Converter

D-to-A converter is an important building block of digital communication system and instrumentation. The D/A converter is made in two sections-the resistor network and a summing amplifier using operational amplifier.

The commonly used D/A converter are:

1. Weighted – Resistor Network.
2. R-2R Ladder Network.

The input to a D/A converter is an N-bit binary signal, available in parallel form and the digital signals are available at the output of the latches or resistors. The voltages corresponding to logic 0 and logic 1 are used to operate through digitally controlled switches.

The analog output voltage V_0 of an N-bit D/A converter has the digital input given by

$$V_0 = K\,(2^{N-1}.b_{N-1}+2^{N-2}.b_{N-2}+\ldots\ldots+2b_1+b_0\,)$$

Where K is a proportionality factor, b_N is the binary coefficient can take two values 1 or 0 depending on the N^{th} bit of the digital input is at 1 or 0, respectively.

Weighted-Resistor D/A converter

Figure 6.21 is used for converting the digital output to analog output in which the resistance values are weighted in accordance with the binary weights.

Weighted Resistor Network

The input digital bits are given at D, C, B and A input which are then given to the logic switch, the logic switch has two inputs one a constant reference voltage which is provided to the resistive network when input data is HIGH. The second input to the logic switch is connected to ground, therefore when input data bit is LOW the switch is connected to ground input.

The output equation of the resistive network is given by equation

$$V = V_{ref}.(2^0+2^1+2^2+\text{——————}+2^{N-1}\,) / (2^{N-1}\,)$$

Where n is number of input bits. The output developed at the input of OP AMP according to the conditions shown in Figure 6.22 is 0101.

$$V = 10\frac{(2^2+2^0)}{(2^4-1)}$$

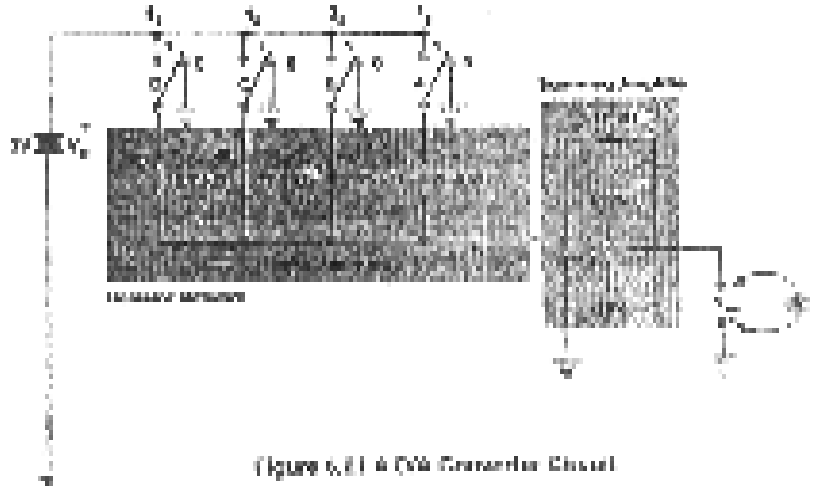


Figure 6.21 A D/A Converter Circuit

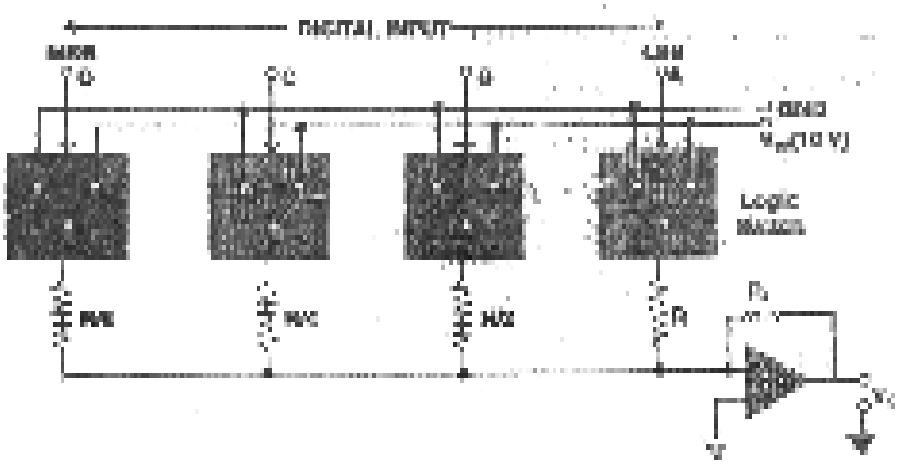


Figure 6.22 4-Bit Weighted Resistor Network

$$V = \frac{10.(4+1)}{15} = \frac{10.5}{15} = 3.33\,\text{Volts}$$

In this network the MSB resistor will be very small if we use a D/A converter of higher bit. It is very difficult to manufacture a precise value of small resistor and due to this fact Binary Weighted resistor network is less popular. This drawback is overcome in R-2R ladder type network described below in Figure 6.23 and Figure 6.24 (a).

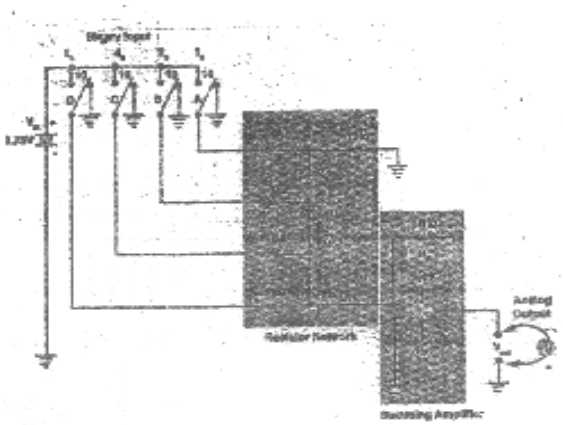


Figure 6.23 A D/A Converter Circuit using an R-2R Ladder Resistor Network

R-2R Ladder Network

Digital-to-analog converters using a R-2R ladder network that provides the proper weighting for the binary inputs is shown in Figure 6.23.

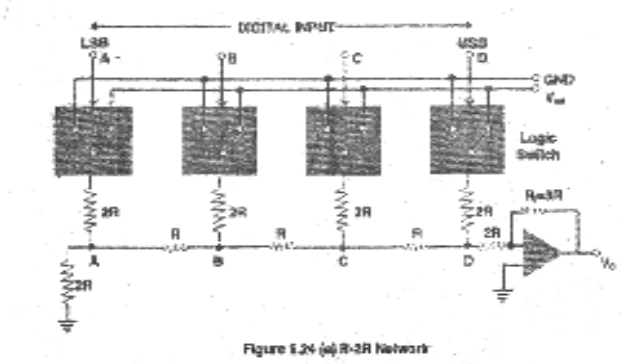


Figure 6.24 (a) R-2R Network

The advantage of this arrangement of resistors is that only two values of resistors are use. Resistors R_1, R2, R_3, R_4 and R_5 are 20KW each and resistor R_6, R_7, R_8 and R_F are each 10KW. Notice that all the horizontal resistors on the ladder are exactly twice the value of the vertical resistors, hence the name R-2R ladder network.

The summing amplifier is same as that the weighted resistor network with a dual power supply ±10V. The operation of this converter is similar to the earlier one and is represented in Table 6.7. An input voltage of 3.75 volt is used and each binary count increases the analog output by 0.25V as shown in the last column of the table. Each 0 on the input side corresponds to 0V applied to the input and each 1 corresponds to an input voltage of 3.75V. The input voltage of 3.75V is used because this is very close to the output of TTL counters and other commercially available.

Binary Inputs				**Analog Outputs**
8_s	**4_s**	**2_s**	**1_s**	
D	**C**	**B**	**A**	**Volts**
0	0	0	0	0
0	0	0	1	0.25
0	0	1	0	0.50
0	0	1	1	0.75
0	1	0	0	1.00
0	1	0	1	1.25
0	1	1	0	1.50
0	1	1	1	1.75
1	0	0	0	2.00
1	0	0	1	2.25
1	0	1	0	2.50
1	0	1	1	2.75
1	1	0	0	3.00
1	1	0	1	3.25
1	1	1	0	3.50
1	1	**1**	**1**	3.75

Table 6.7

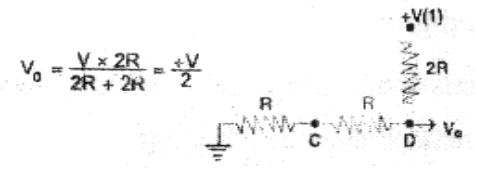


Figure 6.24 (b)

It can be seen from the Figure 6.24 (b) binary ladder that it has only two values of resistors R and 2R thus overcoming the disadvantage of resistive divider network. In R-2R ladder looking back from any node of ladder terminates in equivalent resistance of 2R, thus giving the output voltage equal to V/2.

$V_0 = V.2R/(2R+2R) = + V/2$

This is the output when MSB bit is 1, and as we move towards the LSB the output voltage is divided by 2 each time. For example a 4 bit binary ladder will have 1st MSB voltage=V/2, 2nd MSB=V/4, 3rd MSB=V/8 and LSB bit =V/16.

Characteristics of a D/A converter are specified by the following parameters.

- Resolution
- Linearity
- Accuracy
- Setting time
- Temperature sensitivity

A measure of smallest possible change in output voltage as a fraction of full scale output range. An 8-bit D/A converter has an 8-bit resolution. The resolution of a 8-bit converter is described as one part in 2^8-1=255, or 0.4 percent.

The linearity of a converter is a measure of the precision with which the linear input output relationship is satisfied.

The accuracy of a D/A converter is a measure of the difference between an actual output voltage and the expected output voltage.

The setting time is the time required for the analog output to settle to within ±1/2 LSB of the final value after a change in the digital output. The setting time impose a limit on the frequency of which the digital input can change.

The temperature sensitivity of a D/A converter is specified as ± ppm/degree centigrade. The temperature dependence of the analog output voltage for a fixed digital inputs originate due to temperature sensitivities of the reference voltage source, resistors, operational amplifiers.

Multiplexers

A digital multiplexer is a combinational circuit that selects binary information from one of many inputs and direct it to a single output line. It is one of the most widely used standard circuit in digital design used for sending serial data. The input selected is controlled by a set of select inputs. The receiving of serial data is obtained through de-multiplexers. Figure 6.25 shows the block diagram of a multiplexers with n input lines and one output line.

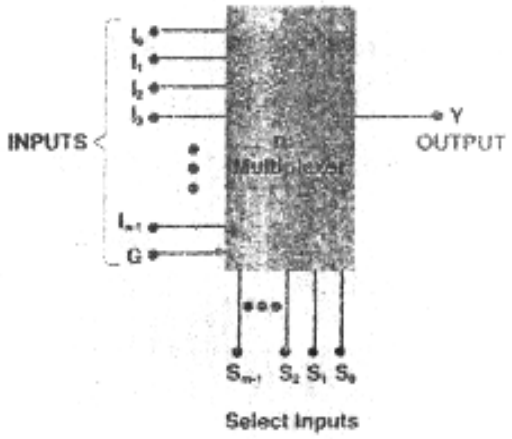


Figure 6.25 Block Diagram of a Digital Multiplexer

For selecting one output of n inputs for connections to the output, a set of m select inputs is required, where 2m=n. The input G (called a store or enable) is incorporated for cascading and is in a active-low state. If a multiplexer has n-select lines, then there can be a maximum of 2n input lines.. Figure 6.26 represent a 4-to-1 multiplexer that has 4-input lines with 2-select lines. The corresponding Truth Table of 4:1 multiplexer is configured in Table 6.8.

The corresponding Boolean expression for the output with LOW input at G is expressed as:

$$Y = \overline{S}_1 S_0 I_0 + \overline{S}_1 S_0 I_1 + S_1 \overline{S}_0 I_2 + S_1 S_0 I_3$$

Select Lines	Output	
S_1	S_0	V
0	0	1_0
0	1	1_1
1	0	1_2
1	1	1_3

Table 6.8

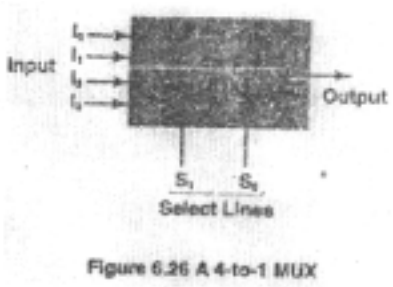
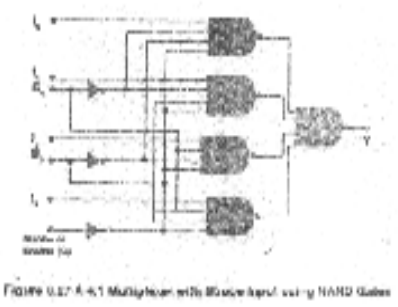
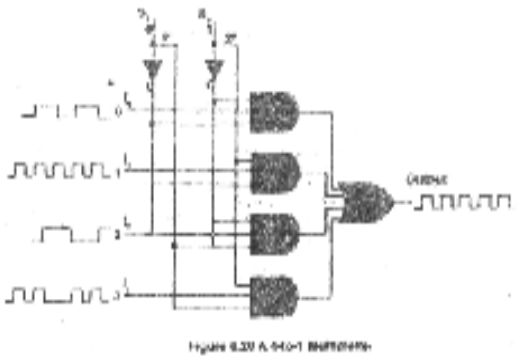


Figure 6.26 A 4-to-1 MUX

This expression can be realised using NAND gates as shown in Figure 6.27. Figure 6.28 represent combinational circuit of a full 4:1 multiplexer using three basic gate: AND, OR and NOT gate.





The realization of 4 variable truth-table using a 8:1 multiplexer is represented in Table 6.9 and the block diagram of the corresponding 8:1 MUX is shown in Figure 6.29. There are 4 possible values of Y and these are : 0, 1, D and ‘D. From this table, it is clear that the output Y is obtained for each of the combinations of A, B and C and this connections has to be made.

Here the inputs A, B and C are to be connected to S_2, S_1 and S_0 selects input respectively. The relationship between input D and output Y for each group of two rows are observed.

Inputs			**Output**
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	$\overline{D}$
0	1	0	
0	1	1	1
1	0	0	D
1	0	1	1 $\overline{D}$
1	1	0	
1	1	1	D

Table 6.9: Table for 4-to-1 Multiplexer

There are many advantages of using multiplexers as logic element. These are

- One does not require to simplify logic expression.
- The IC package count is minimized.
- Logic design is simplified.

Demultiplexers

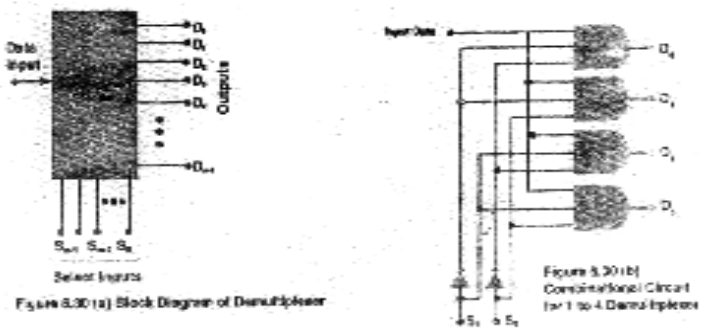
A demultiplexer is a digital switch which allows us to switch one input to one of many possible output lines. It performs the reverse operation of a multiplexer. A demultiplexer accepts a single input and distributes it over several outputs. The block diagram is shown in Figure 6.30 (a). If it has n-selected lines, then there can be

maximum 2n output lines. The select lines are used to select an output on which the input data is present. Figure 6.30 (b) is the realisation of the truth table 6.10 through combinational circuit.

Select Lines		**Output Lines**			
A	**S_0**	**D_3**	**D_2**	**D_1**	**D_0**
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Table 6.10

The combinational circuit of a 1:4 demultiplexer is shown in Figure 6.31. The line which the input is to be connected to is determined by a binary number which is input to the demultiplexer. A demultiplexer looks very much like decoder. A demultiplexer use as the enable line as a data input. The data appears on the select output when the corresponding binary number is input to the select inputs. The input bit or data is given to all the AND gates input and this input is available at a particular output according to the select line input. The truth table for a demultiplexer is given in Table 6.10. Each row of the truth table is read as follows: when the select line is S_1=S_0=1, then the input data is present on D_0. Similarly, when select line is S_1=0 and S_0=1, then input is present on D_1 and so on.



❖ ❖ ❖

: 7

Author :
Dr. Devendra Mohan

Vetter :
Dr. Sib Krishna Ghoshal

APPLICATIONS OF LOGIC CIRCUITS - II

- **Encoders**
- **Decoders**
- **ROM and RAM Cells and Organisations**
- **LED/LCD Displays**
- **ALU Chip**

Encoders

The commonly used encoders are Decimal to BCD encoder Octal to Binary encoder. The encoder is a logic circuit that provides the appropriate code (e.g. binary, BCD) as output for each input voltage signal. The process of encoding is reverse of decoding.

A set of ten switches, one for each numeral 0 and9 is one of the most commonly used input device for a digital system. The logic levels 1 or 0 is generated by these switches in response to turning them OFF or ON. To feed a number in BCD code, the switch corresponding to that number is pressed. The block diagram of 74147 IC for performing this function is shown in Figure 7.1.

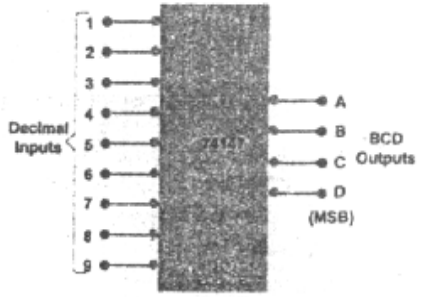


Figure 7.1 Block Diagram of 74147 Decimal-BCD Priority Encoder

Decimal to BCD Encoder

The truth table of decimal to BCD encoder lists the input and output conditions are shown below in table 7.1 (a).

Inputs										**Outputs**			
D_9	**D_8**	**D_7**	**D_6**	**D_5**	**D_4**	**D_3**	**D_2**	**D_1**	**D_0**	**D**	**B**	**C**	**A**
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7.1 (a)

The Boolean expression can be written from the truth table

$A=D_1+D_3+D_5+D_7+D_9$

$B=D_2+D_3+D_6+D_7$

$C=D_4+D_5+D_6+D_7$

$D=D_8+D_9$

Realization these equations are shown in the Figure 7.2.

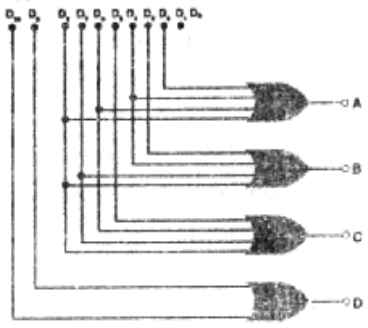


Figure 7.2 Decimal to BCD Encoder Circuit

The IC used for performing the function is 47147IC which, is decimal to BCD priority encoder. It has active Low input and outputs. The priority encoder means the highest number input has priority over the low numbers priority for e.g. if simultaneously inputs are available at D_4 and D_7, then the output will be corresponding to D_7 as it has higher priority than D_4.

Octal-to-Binary Encoder

This kind of encoder is useful in digital circuits for entering long binary words in the form of octal code. The block diagram of a priority encoder 74148 IC that performs such operation is shown in Figure 7.3. These encoders are widely used for handling priority interrupts in computers, microprocessors, etc.

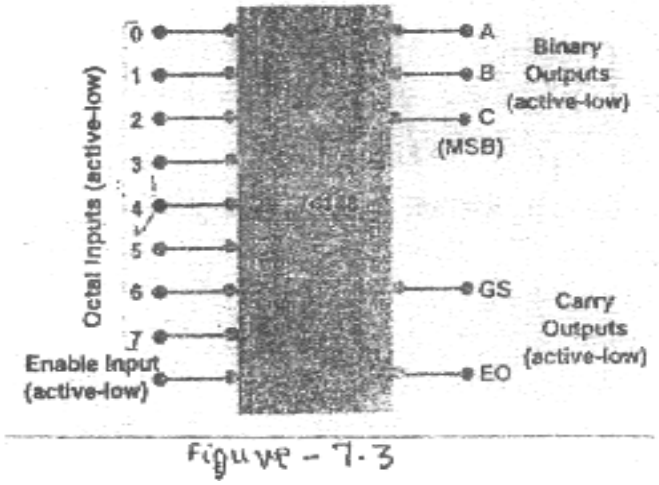


Figure - 7.3

The truth table of Octal to binary encoder is shown below in table 7.1 (b).

The Boolean expressions can be written from truth table as

$X=D_4+D_5+D_6+D_7$

$Y=D_2+D_3+D_6+D_7$

$Z=D_1+D_3+D_5+D_7$

Inputs								Outputs		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	X	Y	Z
0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0	0	0	1
0	0	0	0	1	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	1	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	1	0	0	1
0	1	0	0	0	0	0	1	1	1	0
1	0	0	0	0	0	0	1	1	1	1

Table 7.1 (b)

To realize these equations one can design a priority encoder circuit using gates as shown in Figure 7.4.

As mentioned earlier, the IC used for octal to binary encoding is 74148 IC. This IC has active low input and output and is also a priority encoder IC.

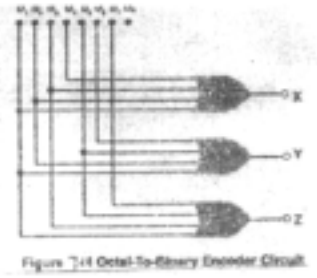


Figure 7.4 Octal-To-Binary Encoder Circuit

A digital system using an encoder is shown in Figure 7.5, which is used to translate from decimal to binary number.

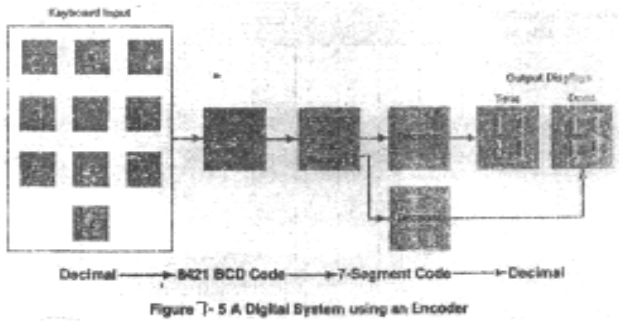


Figure 7- 5 A Digital System using an Encoder

Decoders

We are in the world of decimal numbers and therefore, prefer to see output of digital system in a decimal format. The output can be obtained in two ways, either display (using LEDs or to actuate, some indicators. There are decoder/driver ICs available to perform such job (e.g. 7441, 7442, 7443, etc.) and may halve active high inputs and active low outputs.

A decoder like an encoder is a code translator. Figure 7.3 shows two decoders being used in a display system. The decoders are translating the 8421 BCD code to a seven-segment number. Figure 7.6 shows the BCD number 0101 at the input of the BCD-to-seven-segment decoder/driver. The decoder activates outputs A, C, D, F and G to light the segments as shown. This is the most popular display device used in digital system.

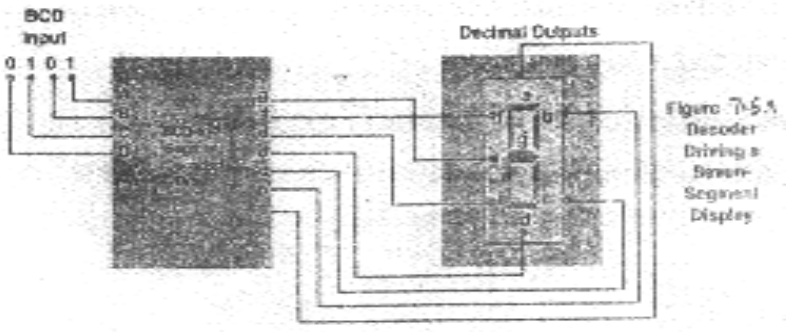


Figure 7-6 A Decoder Driving a Seven-Segment Display

The decimal number five lights up on the display. The block diagram used for the 8421 BCD is shown in Figure 7.7.

The decimal number five lights up on the display. The block diagram used for the 8421 BCD is shown in Figure 6.37.

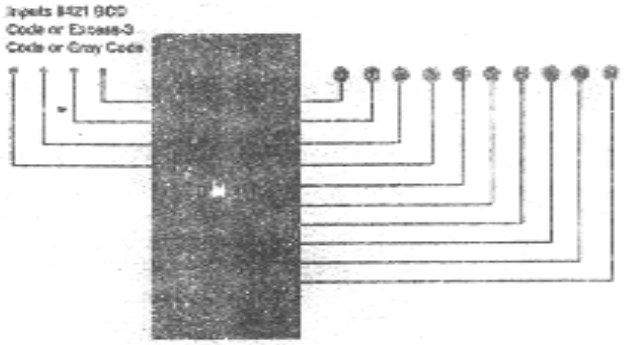


Figure 7-7 A Typical Decoder Block Diagram. Note that Inputs may be 8421 BCD, Excess-3, or Gray Code

In other words, a decoder is a combinational circuit that converts binary information from an input lines to a maximum of 2^n unique lines. For example, if the input to a decoder has two binary lines, the decoder will have four output lines.

The block diagram of 2 to 4 decoder is shown in Figure 7.8. Left if input data is 01, then the output line 1 will be at logic 1 and other will remain at logic 0.

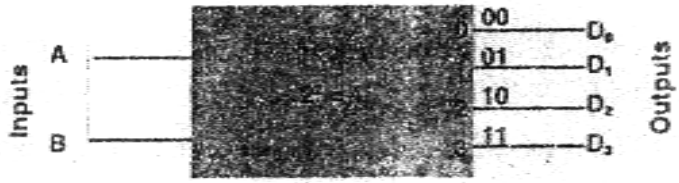


Figure 7.8 Block Diagram of 2 to 4 Decoder

The input/output table for 2 to 4 decoder is represented in Table 7.2.

Input		**Output**			
A	**B**	**D_3**	**D_2**	**D_1**	**D_0**
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	$D_0 = \overline{A}.B$ 0	0	0	0

Table 7.2

From the truth table the equation for D_0, D_1 D_2 and D_3 can be written as,

$D_1 = \overline{A}.B$

$D_2 = A.\overline{B}$

$D_3 = A.B$

The equations can be implemented by using basic gates to design a 2 to 4 decoder as framed in Figure 7.9.

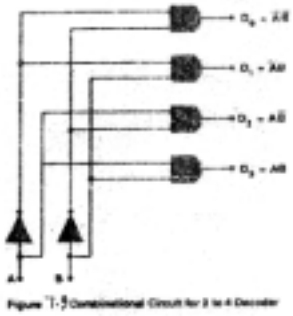


Figure 7.9 Combinational Circuit for 2 to 4 Decoder

The widely used decoders are:

(a) BCD to Decimal Decoder

(b) BCD to Seven Segment Decoder.

BCD to Decimal Decoder

The BCD decimal decoder has 4-bit binary input from 0000 to 1001 and has ten discrete outputs form decimal 0 to 9. The commonly used ICs for BCD to decimal decoder are IC 7441 and IC 7442. Both these ICs have active high input and active low outputs. OC 7441 can directly drive Nixie tube as the output transistor of this IC has high voltage rating. 7442 IC on the other hand easily drive LED's. The pin configuration of these IC's are shown in the figure 7.10 (a) and (b).

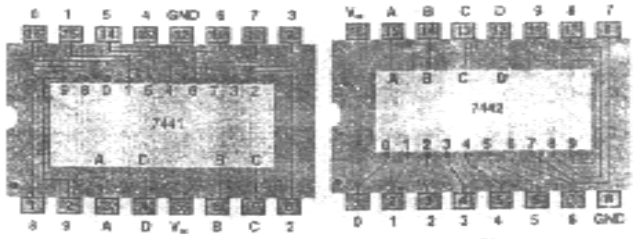


Figure 7.10 (a) Pin Configuration of (a) 7441 IC (b) 7442 IC

BCD to Seven Segment Decoder

BCD to seven segment decoder has standard 8421 BCD input code and generates a special 7 bit output code which is used to operate 7 segment read out. The IC's used for the decoding purpose is 7446 and 7447, the pin configuration for both the IC's is same with minor difference that the output transistor in 7446 can stand up 30V while in 7447 can stand up to 15V. The pin configuration for 7446 and 7447 is shown n Figure 7.11. Now we describe its working.

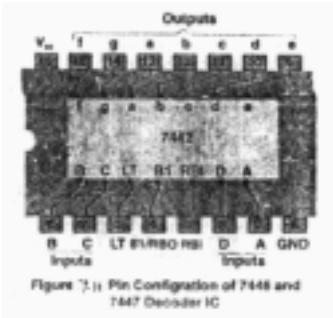


Figure 7.11 Pin Configration of 7446 and 7447 Decoder IC

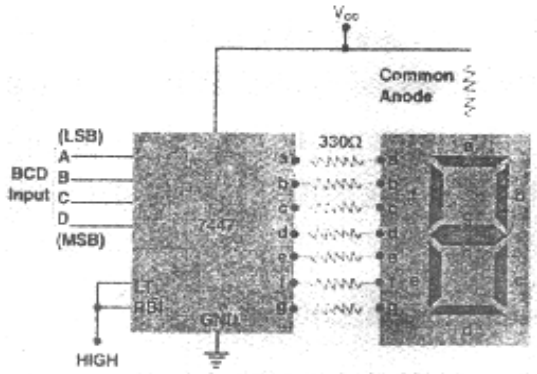


Figure 7.12 Display System

In the pin configuration BI/RBO represent Blanking Input/Ripple Blanking Output. For normal operation this input is held High or kept open. When this input is kept Low all output goes High regardless of any other input condition. Thus, switching off the 7-segment, this input is used for conserving power in multiplexed display.

When RBI, and A,B,C,D, inputs are kept low and Lamp Test (LT) at High level, all segment output go High and RBO goes to Low level, and the 7-segment is switched off, this is used for blanking out leading zero,s in multi-digit display.

When BI/RBO input is kept High and LT input low then whatever may be the condition of the input all output segments go low. For normal operation LT input is kept high, this input is used for cascading purpose and is connected to RBI of the succeeding IC. The function of LT, RBI, RBO and BI are summarized Table 7.3.

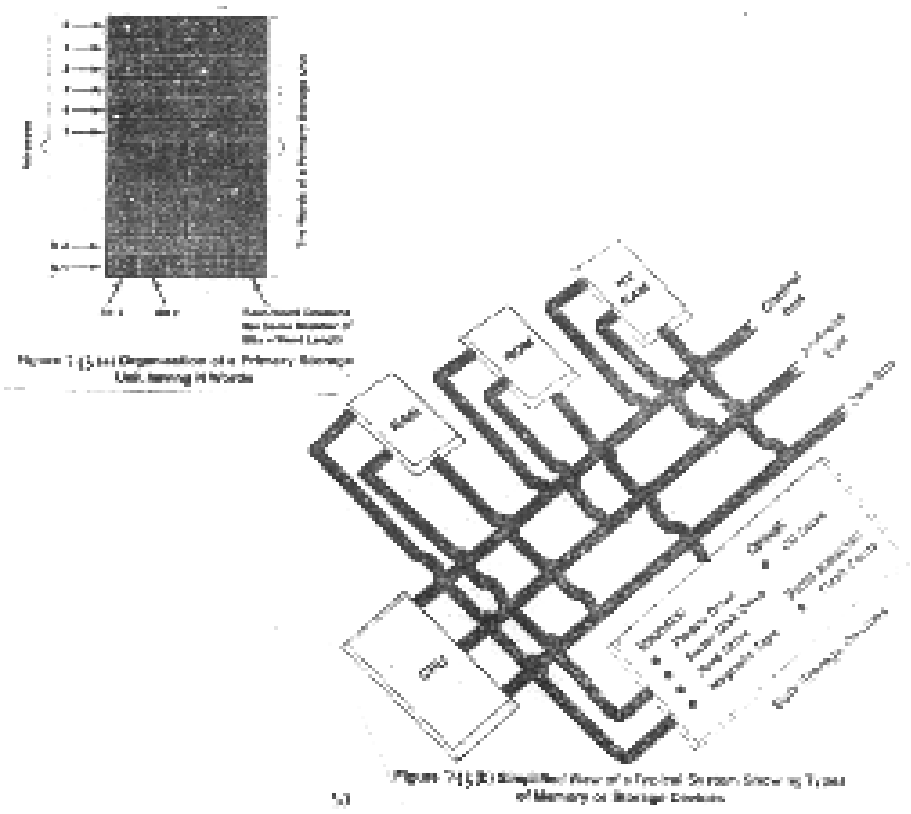
LT	RBI	BI/RBO	BCD Inputs	Display Mode
0	X	1 Output	X	Lamp Test
1	X	0 Output	X	Display Blank
1	1	1 Output	Any Number	Normal Decoding
1	0	Normally at logic Goes to Logic 0 during zero blanking interval	Any Number	Normal Decoding

Table 7.3: Summary of BCD to 7-segment Decoder Functions.

The set up for single 7-segment LED display using 7447 IC is shown in Figure 7.12. LT is used to check the segments of LED. For normal decoding operation, this terminal is connected to logic 1 level. BI is used to conserve power in multiplexed displays. RBO is used for cascading RBO is connected to RBI of the succeeding stage.

ROM and RAM Cells and Organizations

The most important characteristic that a digital system has over an analog system is its ability to store data for short or long periods. The availability and use of memory and digital storage devices has fuelled the information revolution. These storage devices are magnetic, mechanical, optical, or semiconductor in nature. Memory components commonly associated with a modern microcomputer are shown in Figure 7.13 (b).



Semiconductor memory is rapidly becoming the most popular type of medium for the construction of main memory. This is mainly due to two primary reasons:

- Low Cost
- High Speed

Low cost memories are flip-flops called "Scartch pad memories". They range from sixteen to several hundred words in size with speed of the order of few nanoseconds.

Semiconductor memories are basically two types

- ROM
- RAM

The primary storage or the main memory of a computer system is always a challenge to device better and better material for high capacity storage units. The storage locations, addresses and the organizations related to the main memory of a computer system are one of the important aspects.

Any storage unit of a computer system is ranked according to three basic criteria:

1. **Access Time:** A fast time preferred (time takes to retrieve a piece of data from storage).
2. **Storage Capacity:** A large capacity is always desired.
3. **Cost Per Bit of Storage:** Aim is to minimize the cost.

Storage units are basically of two types – primary and secondary. Primary storage units have faster access time, smaller storage capacity and higher bit of storage. Figure 6.43 (a) shows the organizations of a primary storage unit having N words (called word length), which is generally some power of 2. Each word or location has built-in and unique number assigned to it. This number is called the address of the location and is used to identify the location. The primary storage is usually referred as random access memory to directly store and retrieve data and instructions. RAM cell or RAM chip is also referred to as Read/Write memory because the information can be read from it and can also be written into it.

A read only memory (ROM) is one in which information is permanently stored. The information from the memory can only be read and it is not possible to write fresh information into it. This is the reason why it is called ROM. In case of ROM cell, the information is stored inside and is not lost, as the power supply is switched

off. Whereas, in case of RAM cell the information is lost and is therefore, called volatile memory. ROM cells are also known as field stores, permanent stores or dead stores.

To avoid the implementation of complicated electronic circuits, several higher-level operations are performed through special programs called micro program. The micro programs deal with low level machine functions and substitutes for additional hardware. Computer manufacturers for storing this micro-program so those users cannot modify them mainly use ROMs.

A variation of ROM chip is programmable read only memory (PROM). It is possible for a user to customize a system by converting his own programs to micro-programs and storing them in a PROM chip. PROM is a non-volatile storage and the stored information remains intact even if power is switched off.

Once information is stored in a ROM chip, in a PROM chip it cannot be changed or altered. However, there is another type of memory chip called erasable, programmable read only memory (EPROM) that overcomes this problem. It is therefore, possible to erase information stored in an EPROM chip and the chip can be programmed to store new information using a special prom-programmer facility. EPROMs are mainly used by research and development personal to test the efficiency of the computer system with new programmes. A computer program is typically referred to as software. However, when a computer program is stored in a ROM it is called firmware because of the difficulty of making changes.

PIN NOMENCLATURE	
A_0 - A_{12}	**Address Inputs**
E/E/S/S	**Chip enable/Power Down or Chip Select**
Vcc	**Date Out**
Vss	**5-V Supply Ground**

Table 7.4 : Pin Nomenclature

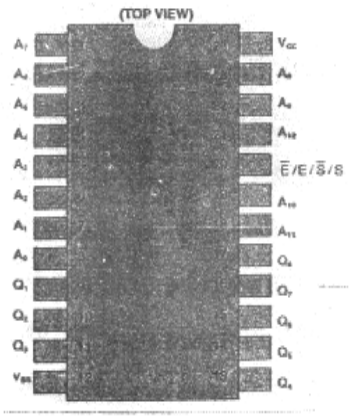


Figure 7.14 Pin Diagram of TMS 4764 ROM IC

A pin diagram for the TMS 4764 ROM is shown in Figure 7.14. The ROM is housed in a 24-pin DIP. The names and functions of the pins are given in Table 7.4. There are total 13 address lines from A0 to A12 to address the 8192(213) memory locations. A0 is the LSB and nanosecond depending on the version of the chip. The pins labeled Q1 (LSB) through Q8 (MSB) are used to store the data permanently. The output pins Q1 to Q8 are enabled by pin 20. Pin 20 is programmed to be in active HIGH or active LOW input.

The EEPROM is a third variation of PRIN. The EEPROM is a electrically erasable PROM also referred to as an E2PROM. There is also a fourth variation of PROM called flash EEPROM. This is having simplest storage cell and have more memory cells on a single chip with a much greater density. Flash-EEPROMs can be erased or programmed faster than EEPROMs.

The basic idea of a PROM, before and after programming is illustrated in Figures 7.15 and 7.16. This simplified 16 bit PROM is similar to the diode ROM. In the diagram (Figure 7.15) each memory cell contains a diode and a good fuse indicating that all of the memory cells are storing a logical1. This is how the PROM might look before programming.

The PROM in Figure 7.16 has been programmed with seven zeros, as a result the tiny fuses are blown. A blown fuse in this case disconnects the diode and means a logical 0 is permanently stored in this cell. Because of the permanent nature of burning of a PROM, the unit cannot be reprogrammed.

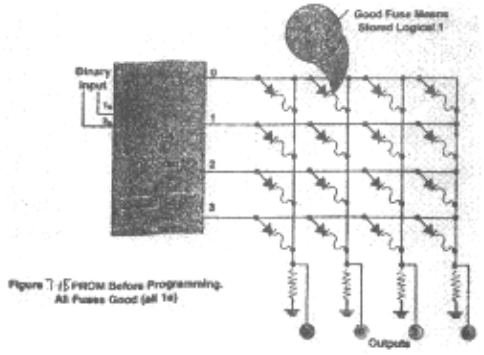


Figure 7.15 PROM Before Programming.
All Fuses Good (all 1s)

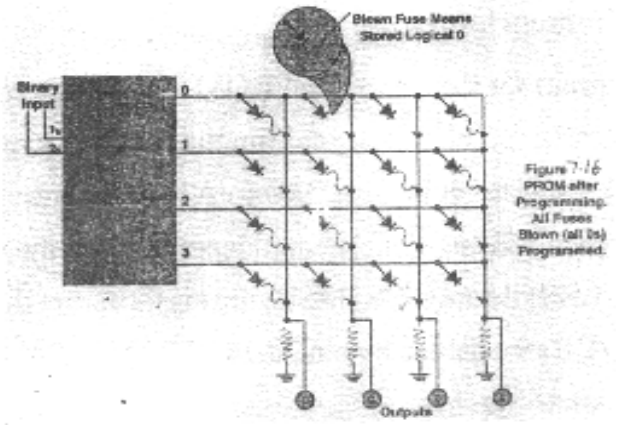


Figure 7.16 PROM after Programming. All Fuses Blown (all 0s) Programmed.

ROM (Read Only Memory)

As mentioned earlier, a Read Only Memory (ROM) is a data storage facility into which information is normally written only once, and once the data is written into it, it produces the same output when addressed.

Figure 7.17 represents the block diagram of 2^{nxm} ROM has n-input lines which can address 2^n words of m-bit each, for example a 16x5 bit ROM means that it has 4 address lines which can access 16 words and each word is of 5 bit.

The basic ROM structure is a matrix of elements each of which is accessed by a random address code allowing approximately equal access time to all bits. The simplest ROM structure is a network of diodes where the presence (1) or absence (0) of diode determines the logic-state of a particular location. Such network is shown in Figure 7.18.

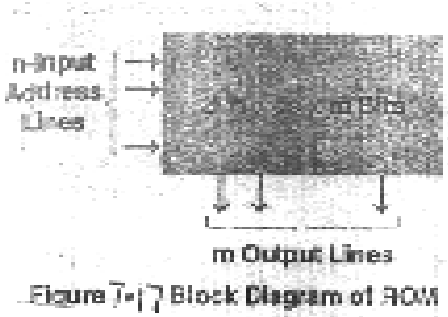
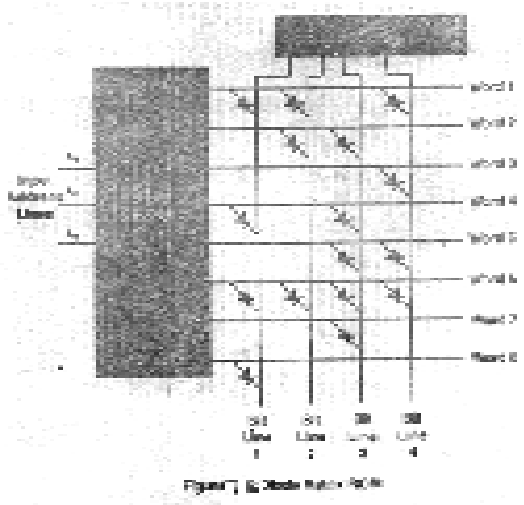


Figure 7.17 Block Diagram of ROM



The row address decoder (3 to 8 decoder) selects a line at a time by raising its voltage to a high level, forward biasing the diodes attached to that line. When the diode begins conducting, they force their associated bit lines to a higher (logic 1) level, while bit lines not connected to diodes remain low (logic 0). For example, if the address bits are 100 then the word line 5 is selected and it will be at higher potential and diodes connected to that line conducts (are at logic1) thus giving output 0011.

The common applications of ROM include, sequence generator, waveform generator and seven-segment display as shown in the previous sections.

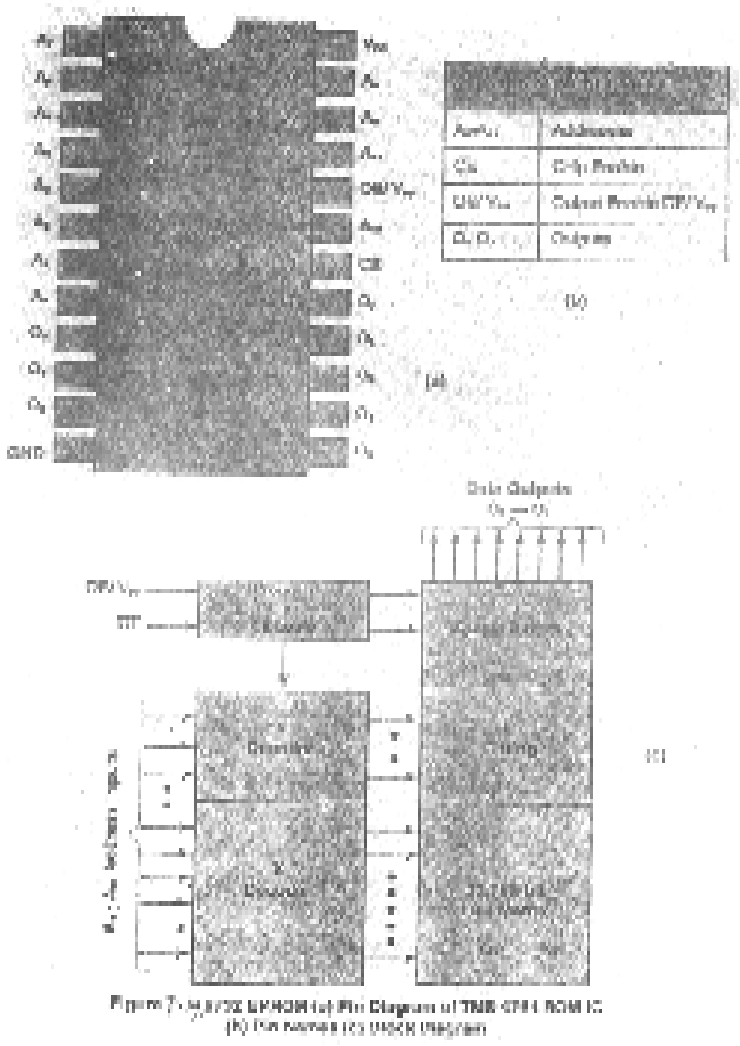
PROM (Programmable ROM)

As shown in the Figure 7.15 and Figure 7.16, a PROM is a user programmable type, which is delivered unprogrammed and then the user can select and enter data from a special programmer. The programmer is an instrument on which a programme is composed and the PROM integrated circuit is plugged into it for the actual writing process. The regular PROM can only be programmed once like a ROM, but its advantage is that it can be made in limited quantities and can be programmed in the local lab or shop.

EPROM (Erasable PROM)

It is a user programmable type which can be erased but only by special means. A popular EPROM family in the 27XXX series. A sample IC from the 27XXX series EPROM family is illustrated in Figure 7.19. The Pin diagram in Figure 7.19 (a) represents the 2732 A 32K (4Kx8) ultraviolet-erasable PROM. It has 12 address pins (A_0 to A_{11}) which can access 4096(2^{12}) byte wide words in the memory. This uses a 5-V power supply and can be erased using UV light. The OE/Vpp pin serves a dual purpose. It has one purpose during reading and another during writing. Under normal use the EPROM is being read.

The eight outputs pin are labeled O_0 to O_7 on the 2732 EPROM indicated in the Table 7.19 (b). Figure 7.19 (c) shows the organization of 2732 EPROM chip. When the EPROM is erased, all memory cells are returned to logical1. Data is introduced by changing selected memory cells to Os. During programming (writing), the input data is applied to the data output pins (O_0 to O_7). The word to be programmed into the EPROM is addressed using the 12 address lines. A very short TTL level LOW pulse is then applied to the CE input to complete the write process. Direct sunlight takes about one week to erase EPROM.

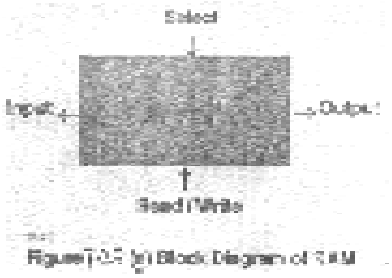


RAM (Random Access Memory)

As discussed in the previous section temporary data storage RAM is used. RAMs are used for calculator memories, buffer memories, cache memories, and micro-computer user memories.

RAM is a type of memory from which data can be read out and can be written (stored) into. The basic storage element in RAM is flip-flop can store one bit of information electronically. RAM is a volatile memory, when power is off all its contents are lost.

Block diagram of a RAM cell is shown in Figure 7.20 (a). A RAM cell, has four terminals one input which is used to input the data into it, another is output which outputs the stored data, a select line which enables or selects particular RAM cell for reading or writing purpose according to the signal present at Read/Write input. If Read input is high (or 1) data can be written into the cell. Figure 7.20 (b) shows the internal structure of a Binary cell.



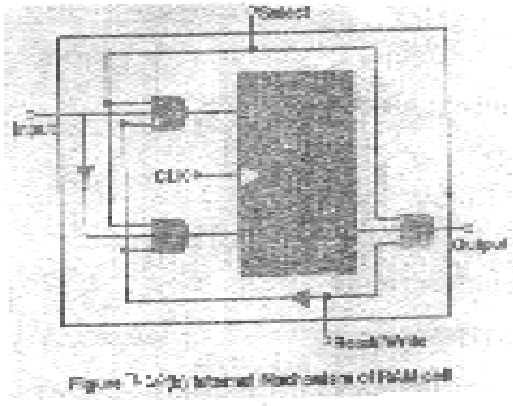


Figure 7.20 (b) shows the internal mechanism of a RAM cell which can store a single bit. For example, if we want to store bit 1 into the cell, then first we select the cell by giving high (1) at select terminal this enables the cell we give 0 to Read/Write terminal which enables the two AND gates at the input of the flip-flop on the positive edge of the clock pulse. This is how a bit is stored into a RAM Cell and when it has to be read out, then select and read/write terminal is made high. This enables the output AND gate and data can be read out from the cell.

These RAM cells are arranged into arrays to store large amount of data. To describe its operation let us consider a 4X3 bit RAM array as shown in Figure 7.21. It has two address lines, which can address maximum of 4 words and here each word is of 3 bits. Lets data at the input is 101, and address lines A_1=1, A_0=0 and Read/Write is low As address is A_1=1, A_0=0, the third i.e. Word–3 is selected for reading and writing operation. As the Read/Write input is 0 the data will be written into the RAM cell at word number 3. When this data has to be read out address of the WORD-3 is given and Read/Write is given high and the data will be present at the output.

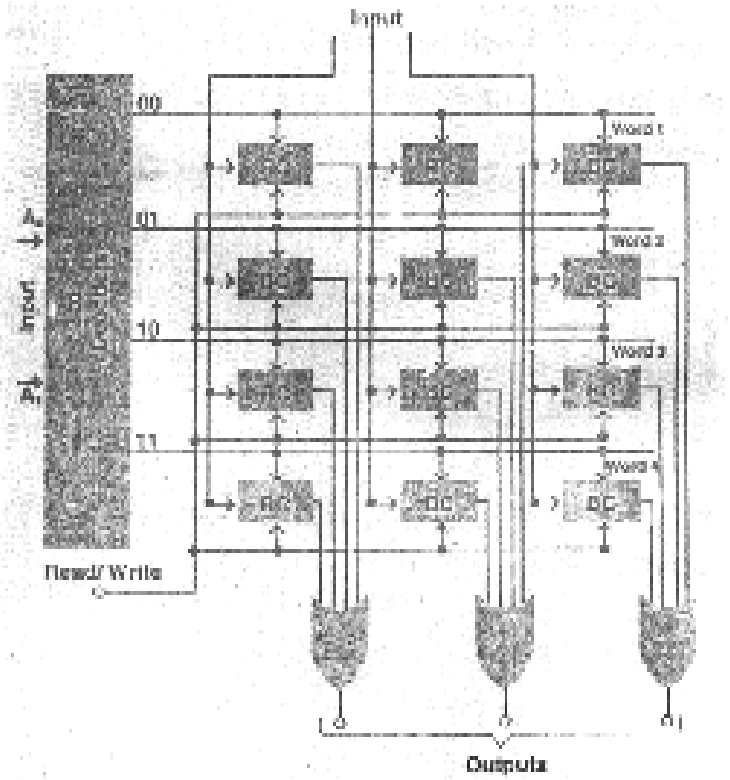


Figure 7.21 4 X 3 Bit RAM Array

Metal Oxide Semiconductor Memory (MOS-RAM)

In the earlier section, most of the description of RAM is done as diode an element i.e. we learnt about bipolar-RAM. Now we look forward to the MOS-RAMS. The basic device used in the construction of a MOS semiconductor memory is MOSFET. Both P-channel and N-channel device are available.

MOS devices are some what simpler than bipolar devices, as a result MOS memories can be constructed with m ore bits on a chip, and they are less expensive than bipolar memories. The intrinsic capacitance associated with the MOS device generally means that MOS memories are slower than bipolar memories, but this capacitance can be used to have advantages.

A MOS cell may be either static or dynamic in nature. A static cell retains its stored data as long as power is supplied to the circuit. A dynamic cell depends upon capacitative charge storage to hold its data and must receive a "refresh" input to counter-act the effect of leakage. The leaky capacitance associated with a MOSFET can be used to store charge and this then the basic unit used to form a dynamic memory.

Static MOS RAM'S

The basic static MOS RAM cell is a bistable multivibrator closed resembling the bipolar flip-flop used in TTL memories.

Dynamic MOS RAM'S

The basic storage element in a dynamic MOS RAM cell is a capacitor which holds and releaves a stored charge in response to read and write commands. While the capacitor could be external device, it is much more common for dynamic RAM's to utilise the capacitance existing between gate and source of MOSFET itself. This capacitance in due to the isolation of the gate from rest of the structure by a dielectric material. Charging the gate source capacitance sufficiently to turn the transistor on represents a logic 1 state in most applications. While a lower charge or no charge at all serves as a logic 0.

As it is with all capacitative devices, the charge stored in the gate-source region drains off due to leakage current. If the charge is allowed to deteriorate too much, the

data bit is lost. Some means must be provided to periodically restore, or refresh, the charge. A common requirement is that every cell in a memory matrix be refreshed in every 2 milliseconds.

LED/ LCD Displays

"More we see more we remember". Therefore, a digital system needs display of the results. A-7 segment display is the most popular device used in display systems. A digital display that consists of seven LED segments is commonly used to display decimal numerals in digital systems. Most familiar examples are electronic calculators and watches, where one seven –segment display device is used for displaying one numeral 0 through 9. For using this display device, the data has to be converted from some binary code to the code required for the display. Usually, the binary code used is a natural BCD. The seven-segments of the display are labeled 'a' through 'g' as shown in Figure 7.22 (a). The displays representing decimal digits 0 through 9 are illustrated in Figure 7.22 (b). For instance if segments a, b and c are lit, a decimal 7 is displayed. If, however, all segments a through g are lit, a decimal 8 is displayed. Here, in the Figure a, f, g, c and d are lit to display five.

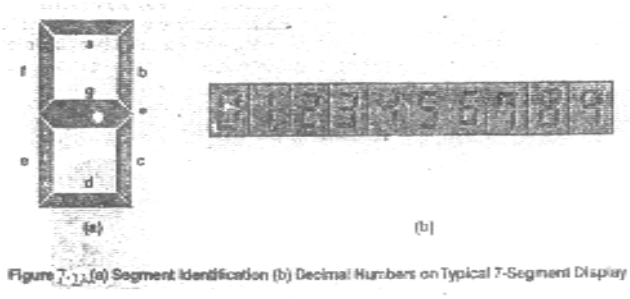
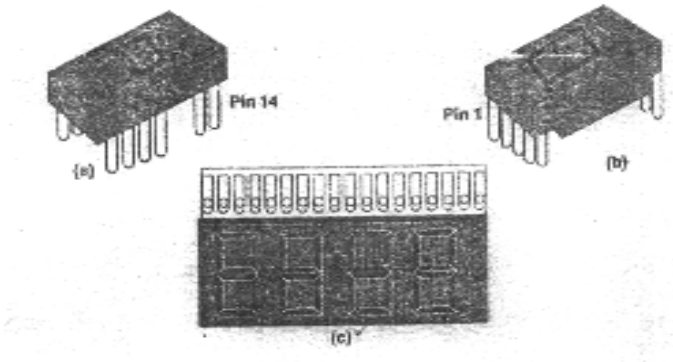


Figure 7.22 (a) Segment Identification (b) Decimal Numbers on Typical 7-Segment Display

Several common 7-segment display packages are shown in Figure 7.23 (a), (b) and (c). The unit represented in Figure 7.23 (c) is a multi-digit LED display widely used in digital clock. The 7-segment display in Figure 7.23 (a) and (b) fits across and cross wise a regular 14-pin DIP IC socket.

The 7-segement display may be constructed with each of the segments being a thin filament that glows. This type of unit is called an incandescent display. The modern vacuum fluorescent display give off a blue green glow at low voltages. The common LED (light emitting diode) displays gives off a characteristic reddish glow when lit.



A basic single LED is illustrated in Figure 7.24 (a) and the cut away of the LED is shown in Figure 7.24 (b). The LED is basically a p-n junction diode under forward biased condition. The LED lights as current flows through it and the emitted light is focused by a plastic lens. LEDs are fabricated from Gallium Arsenide (GaAs) (with doping) and several related materials that come in various colors.

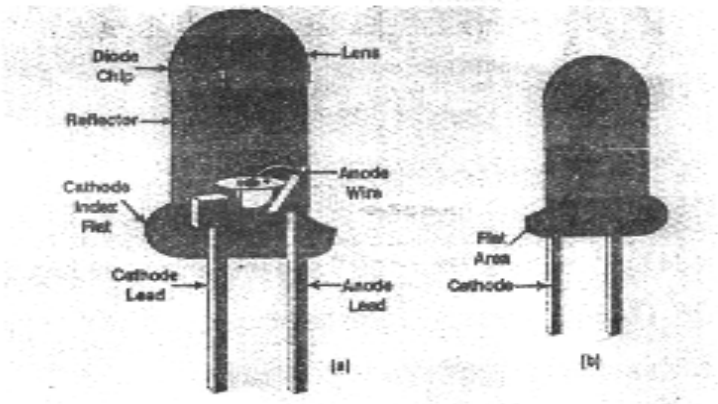


Figure 7.24 (a) Cutaway View of Standard Light-Emitting Diode (b) Identifying the Cathode Lead of the LED

A single LED is shown in Figure 7.25 (a). When the switch SW1 is closed, current flows from the 5 V power supply through the LED, causing it to light. The series resistor limits current to about 20 mA. Without the limiting resistor, the LED would burn out. It is very much sensitive to polarity and accept only about 1.72 to 2.1V across their terminal when bit.

A seven segment LED display is shown in Figure 7.25 (b) in which each segment (a through g) contains an LED, as shown by seven symbols. The display shown has all the anodes tied together and coming out the right side as a single connection called common anode. The inputs on the left go to the various segment of display. The device is referred to as a "common-anode seven-segment LED display".

To understand how segments on the display are activated and lit, consider the circuit as illustrated in Figure 7.25(c). If switch b is closed, current flows from GND through the limiting resistor to the b-segment LED and out of common-anode connection to the power supply. Only segment b will glow. Usually power for the LED segments is provided by an IC called a display driver. It is known as 7-segment decoders/drivers.

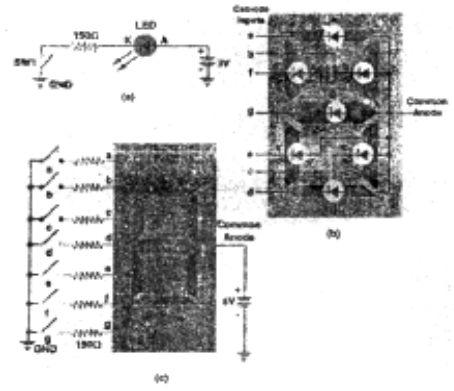


Figure 7.25 (a) Operation of a Simple LED (b) Wiring a Common Anode 7-segment LED Display (c) Driving a 7-Segment LED Display with Switches

The Truth Table of BCD-to-7-segment decoder is shown in Table. Here ABCD is the natural BCD code for numerals 0 through 9 and the corresponding display system is shown in Figure 7.26. Through Karnaugh maps technique, each of the outputs a through g can be simplified as minimum expressions is given by.

$A = \overline{B}\,\overline{D} + BD + CD + A$

$B = \overline{B} + \overline{C}\,\overline{D} + C\,D$

$C = B + \overline{C} + D = \overline{\overline{B}C\overline{D}}$

$D = \overline{B}\,\overline{D} + C\,\overline{D} + \overline{D}\,C + B\overline{C}D$

$E = \overline{B}\,\overline{D} + C\,\overline{D}$

$F = A + \overline{C}\,\overline{D} + B\overline{C} + B\overline{D}$

$G = A + B\overline{C} + \overline{B}C + C\overline{D}$

Decimal Digit Display	Inputs				Outputs						
	A	**B**	**C**	**D**	**a**	**b**	**c**	**d**	**e**	**f**	**g**
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1

Table 7.5 : BCD-to-7-segment Decoder

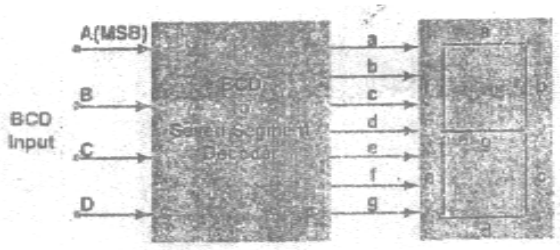


Figure 7.16 Display System

The NAND gate realizations of these Boolean expressions are shown in Figures 7.27 (a) to (g)

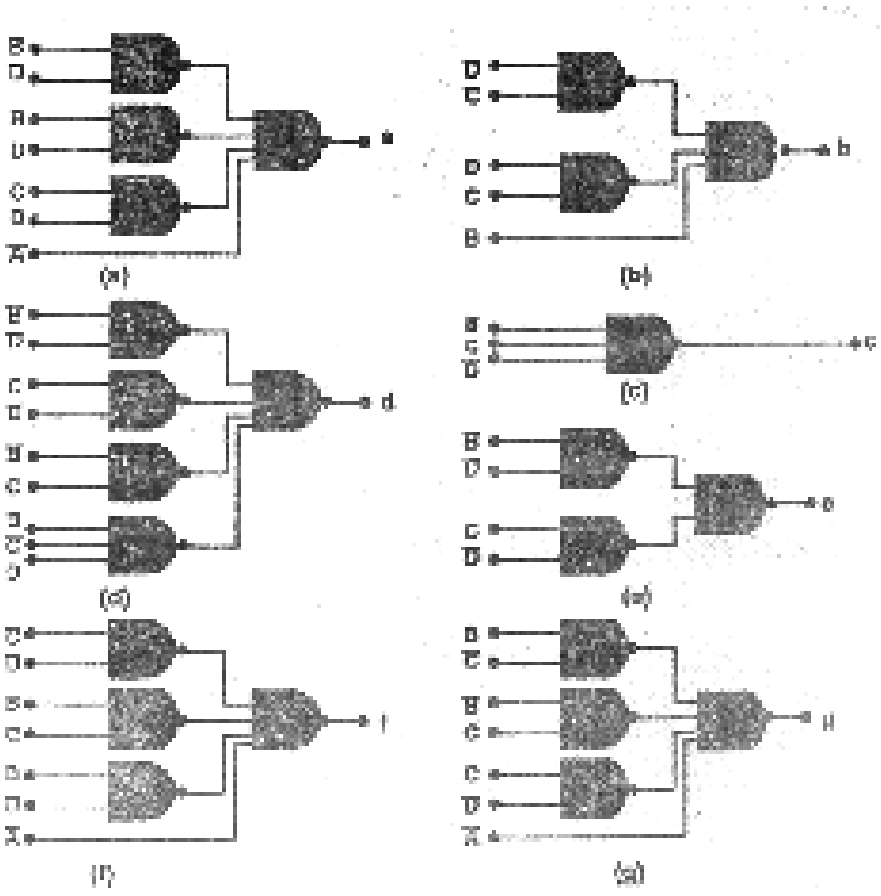


Figure 7.27 NAND Gate Realisations of Boolean Expressions (a) through (g)

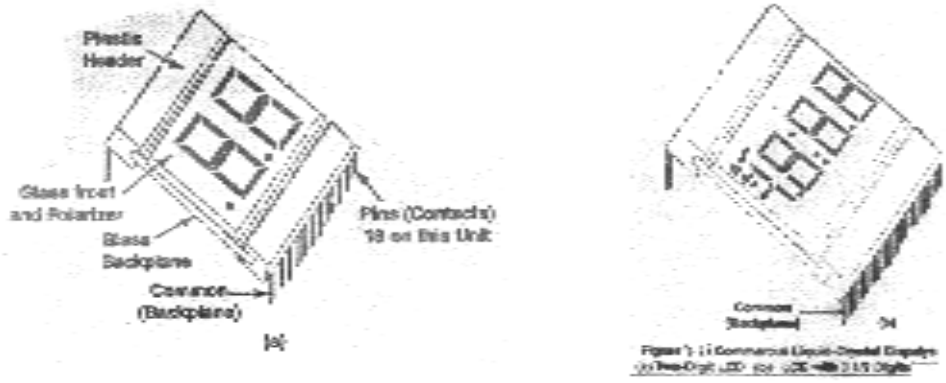
LCD Display

The newer liquid-crystal display (LCD) creates numbers in black or silvery color is one of the widely used display device. It is also suited for more complex displays than just 7-segment decimal. When a segment is energized by a low frequency square wave signal, the LCD signal appears black, while the rest of the surface as well as non-energize segments are nearly invisible. The key to LCD operation is the Liquid crystal, or Nematic fluid.

There are two of LCDs in use today

- Dynamic LCD
- Field Effect LCD

The two types use-different materials for the liquid crystal and work differently. Neither type emits any light nor each must have an external light and each must have an external light source to be seen. Figure 7.28 (a) and (b) illustrates two examples of typical commercial LCD devices. Note that both have pins, which can be soldered into a printed circuit board. A simple two-digit 7-segment LCD uses two glass plates and has 18 and 40 pins. All segments, decimal points and symbol are assigned a pin number. LCD that produce frosty white characters on a dark background are known as "Dynamic Scattering" LCD. The dynamic scattering LCD consumes more power than field effect displays. The field effect LCD is the most popular one.



The 7-segment pattern, which is etched on the front glass plate, is made of a clear electrically conductive material such as Indium Oxide. The back glass is coated with the clear conductor which corresponds to the 7-segments as pictured in Figure 7.29. Only the digit segment will be seen when an electric current is applied to the LCD.

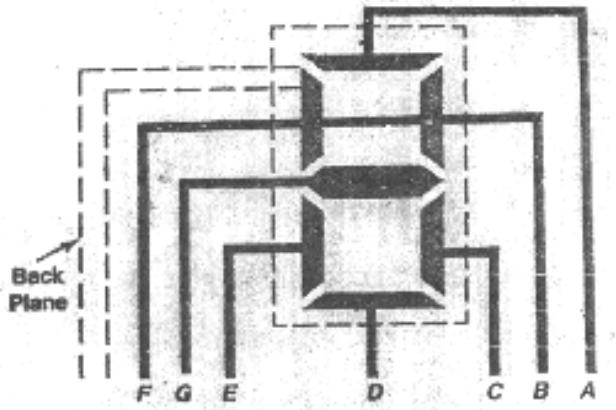


Figure 7.29 Clear LCD Segment Conductors

In actual practice the decoder and EXOR LCD display drivers as shown in Figure 7.30 are usually packaged in a single CMOS IC. The 100-Hg square-wave signal frequency is not critical and may range from 30 to 200 Hz. Liquid-Crystal displays are sensitive to low temperatures. At below-zero temperature the LDC display's turn-off times become very slow. However, the long life-time and extremely low power consumption make them ideal for battery or Solar Cell operation.

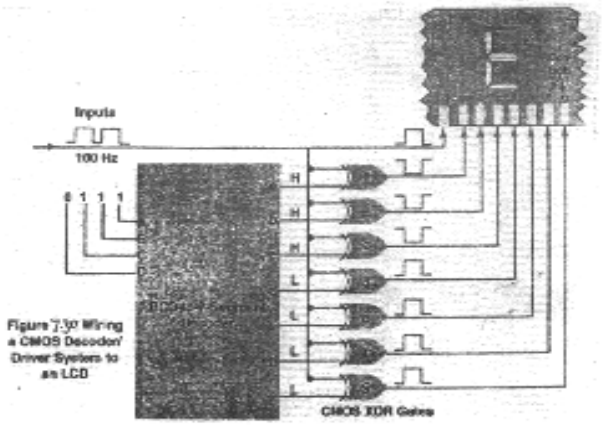


Figure 7.30 Wiring a CMOS Decoder/ Driver System to an LCD

A block diagram of a LCD decoder-driver system is sketched in Figure 7.31.

The input is 8421 BCD and the latch is a temporary memory to hold the BCD data. The output from the decoder is in 7-segment code. The LCD driver consists of EXOR gates as shown in Figure 7.30.

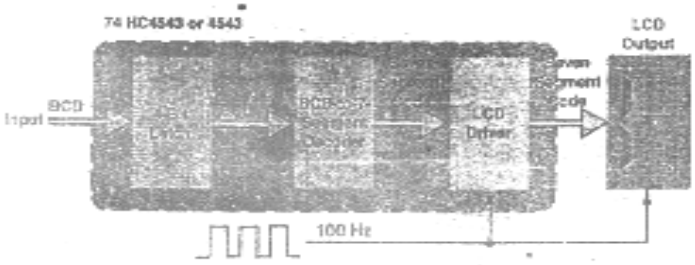


Figure 7.31 Driving a 7-Segment LCD. Block diagram of System used to Decode and Drive LCD

In the wiring diagram of a single LCD driver circuit is sketched in Figure 7.31 a 74HC4543 decoder/driver CMOS IC is being employed. The 8421 BCD input corresponding to the decimal number 3 is 0011, which is decoder into 7-segment code. Only out of phase signals will activate a signal and in-phase signals do not activate LCD segments as clearly seen from the diagram.

The field effect LCD are of much more advantage because of the following reason:

- Runs on a 60Hz AC.
- Requires Low Voltage.
- Requires a low current about –300A.
- Light control device.

Due to these reasons, it is mostly used in clock, calculators, digital multi-meters and in computers.

Arithmetic Logic Unit (ALU) Chip

The heart of a computer called Central Processing Unit (CPU) and all the logical operations is carried out in a chip called arithmetic logic unit. Therefore, ALU may be considered as the heart of any micro-processor. ALU is a Combinational Circuit which is capable of performing arithmetic as well as logical operations. Figure 7.32 shows the black diagram of 47181 ALU.

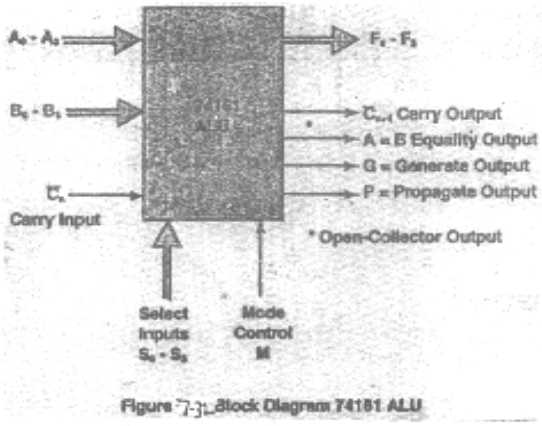


Figure 7-31 Block Diagram 74181 ALU

The main features of ALU chip and functions of various output, input and control lines (IC 74181) are the following:

- A single ALU chip contains near about 100 equivalent gates providing a large number of binary arithmetic operations.
- It perform 16 binary arithmetic functions on two 4 bit input swords. These 4 bit input words are represented A_0-A_3 and B_0-B_3 in the diagram.
- S_0-S_3 represents the select line. The selected operations are performed using these lines.
- The Mode Control pin 'M' can be at LOW or HIGH state. At HIGH (M = 1) state is performs logical operations and at LOW it performs arithmetic operation (M = 0).
- In case of subtraction the carry one C_{n+4} is at active low. For positive results the output is in logic zero (0) and for negative result it is at one (1). The results are expressed in 2's complement form.
- It has two output G and P and is used when more than one ALU chips are used in cascade for making the arithmetic operations faster.
- When two inputs A_0-A_3 and B_0-B_3 are equal then A=B output line is high.
- Carry generate output is G and carry propagate output is P.

The detail function of IC74181 ALU is represented in Table 7.6 below.

0	0000	$F = \overline{A}$	F=A	F=A PLUS 1
1	0001	$F = \overline{A+B}$	F=(A+B)	F=(A+B) PLUS 1
2	0010	$F = \overline{A}+B$	F=A+B	F=(A+B) PLUS 1
3	0011	F=0	F=MINUS 1 (2's COMPL)	F=ZERO
4	0100	$F = \overline{AB}$	F=A PLUS A.$\overline{B}$	F=A PLUS A.$\overline{B}$ PLUS 1
5	0101	$F = \overline{B}$	F=(A+B) PLUS A.$\overline{B}$	F=(A+B) PLUS A.$\overline{B}$ PLUS 1
6	0110	F=A⊕B	F=A MINUS B MINUS 1	F=A MINUS B
7	0111	F=A.$\overline{B}$	F=A.$\overline{B}$ MINUS 1	F=A.$\overline{B}$
8	1000	$F = \overline{A}+B$	F=A PLUS A.B	F=A PLUS A.B PLUS 1
9	1001	$F = \overline{A \oplus B}$	F=A PLUS B	F=A PLUS B PLUS 1
10	1010	F=B	F=(A+B) PLUS A.$\overline{B}$	F=(A+B) PLUS A.$\overline{B}$ PLUS 1
11	1011	F=A.B	F=A.B MINUS 1	F=A.B
12	1100	F=1	F=A PLUS A*	F=A PLUS A PLUS 1
13	1101	F=A+$\overline{B}$	F=(A+B) PLUS A	F=(A+B) PLUS A PLUS 1
14	1110	F=A+B	F=(A+$\overline{B}$) PLUS A	F=(A+B) PLUS A PLUS 1
15	1111	F=A	F=A MINUS 1	F=A

* Each bit is shifted to the next more significant position.

Table 7.6 : Function Table of 74181 ALU Chip

Assignment 1

Design an 8 bit adder/subtractor using 74181s in cascade. Show how it works if (a) A=97 and B=29 (b) A=24 and B=58.

Solution:

For designing an 8-bit adder/subtractor, two 74181 Ics are to be cascaded. The least significant four bits of A and B are applied at the A and B inputs of the least significant 74181and the most significant four bits of A and B are applied at the A and B inputs of the most significant 74181. The carry-out of the least significant 74181is connected to the carry-in of the most significant 74181. The 8-bit output will be available on 'F' outputs. The select lines of both the 74181s are connected together.

Addition is performed with M=0 and S=1001, and subtraction is performed with M=0 and S=0110. Connect 'C_n of least significant 74181 to 1 for addition and 0 for subtraction

(a) A=97=01100001

B=29=00011101

(b) A=24=00011000

B=58=00111010

❖ ❖ ❖

Printed by : Competent Printing Press, Hisar-125001
Ph. : 01662-225100. Mobile : 98960-68720