Basic Electronic Components & Hardware - II

TEM - 2

Centre for Electronics Design & Technology of India

A Scientific Society under Department of Electronics,
Govt. of India, New Delhi

Published by CFS Documentation cell
Division of Centre for Electronics Design and Technology of India
A Scientific Society of Department of Electronics,
Govt. of India,
New Delhi.

First Edition: 1999

CEDTI/CFS/99/4/TEM-2/R1

FOREWORD

The Information Technology and Telecom sectors have suddenly opened up avenues, which require a very large specially trained manpower. These sectors are highly dynamic and need training and re-training of manpower at a rapid rate. The growing gap of requirement of the industry and its fulfilment has created a challenging situation before manpower training institutes of the country. To meet this challenge most effectively, Centre for Electronics Design and Technology of India (CEDTI) has launched its nation-wide franchising scheme.

Centre for Electronics Design and Technology of India (CEDTI) is an Autonomous Scientific Society under the Govt. of India, Department of Electronics with its Headquarters at New Delhi. It operates seven centres located at Aurangabad, Calicut, Gorakhpur, Imphal, Mohali, Jammu and Tezpur. The scheme will be implemented and coordinated by these centres.

The scheme endeavours to promote high quality computer and information technology education in the country at an affordable cost while ensuring uniform standards in order to build a national resource of trained manpower. Low course fees will make this education available to people in relatively small, semi urban and rural areas. State-of-the-art training will be provided keeping in view the existing and emerging needs of the industrial and Govt. sectors. The examinations will be conducted by CEDTI and certificates will also be awarded by CEDTI. The scheme will be operated through all the seven centres of CEDTI.

The CEDTI functions under the overall control and guidance of the Governing Council with Secretary, Department of Electronics as its Chairman. The members of the council are drawn from scientific, government and industrial sectors. The Centres have separate executive committees headed by Director General, CEDTI. The members of these committees are from academic/professional institutes, state governments, industry and department of electronics.

CEDTI is a quality conscious organisation and has taken steps to formally get recognition of the quality and standards in various activities. CEDTI, Mohali was granted the prestigious ISO 9002 certificate in 1997. The other centres have taken steps to obtain the certification as early as possible. This quality consciousness will assist CEDTI in globalizing some of its activities. In keeping with its philosophy of ‘Quality in every Activity’, CEDTI will endeavour to impart state of the art – computer and IT training through its franchising scheme.

The thrust of the Software Courses is to train the students at various levels to carry out the Management Information System functions of a medium sized esTablishment, manufacture Software for domestic and export use, make multimedia presentations for management and effectively produce various manufacturing and architectural designs.

The thrust of the Hardware Courses at Technician and Telecommunication Equipment Maintenance Course levels is to train the students to diagnose the faults and carry out repairs at card level in computers, instruments, EPABX, Fax etc. and other office equipment. At Engineer and Network Engineer levels the thrust is to train them as System Engineers to instal and supervise the Window NT, Netware and Unix Networking Systems and repair Microcontrollers / Microprocessor based electronic applications.

An Advisory Committee comprising eminent and expert personalities from the Information Technology field have been constituted to advise CEDTI on introduction of new courses and revising the syllabus of existing courses to meet the changing IT needs of the trade, industry and service sectors. The ultimate objective is to provide industry-specific quality education in modular form to supplement the formal education.

The study material has been prepared by the CEDTI, document centre. It is based on the vast and rich instructional experience of all the CEDTI centres. Any suggestions on the improvement of the study material will be most welcome.

(R. S. Khandpur)
Director General (CEDTI)

PREFACE

The primary objective of this book is to serve as the text in Digital Electronics for Hardware courses (Especially for TEM level Course) for Electronics engineering students. Our approach is to stress the fundamental concepts of Digital electronics. We hope to convey both the substance and flavor of the subject. The breadth and depth of treatment also makes this volume a valuable adjunct to continuing education of practicing technicians, engineers and computer engineers.

This book has substantial amount of new material that has been added to reflect changes in technology and curricula. This book helps where you need it, offering useful and practical suggestion. The chapter one deals with number systems. It covers Basic numbering system. Chapters two, three, four deal with basics of integrated circuits, different TTL & CMOS logic and semiconductor memories. Chapter five is on eight bit microprocessor with introduction to 16-bit microprocessor such as 80286. Chapter six focuses on Communication Theory. Chapter Seven deals with Modulation.

The review questions will be available at the end of each chapter so, that the student's qualitative knowledge of the text material is increased. The material of this book is collection of articles from various books mentioned in Bibliography.

CEDTI

CONTENTS

Chapter One

NUMBER SYSTEMS

Features:

1.1 INTRODUCTION:

In everyday life, generally a decimal number system is used which is said to be of base 10. The base of number system is that, raised to zero power, is the lowest position value, raised to first power, it is second position value, and so on. In computers, BINARY number system is used which is of base 2. A switch being ON (1) or OFF (0), a voltage being present (1) or zero unit (0), all this can be represented by 2 symbol, 1 and 0 or low or high. Thus, the BINARY system is convenient in computers.

Number Systems

1.2 Decimal number systems:

This number system makes use of 10 symbols i.e. 0,1,2,3,4,5,6,7,8,9 that is the base of this number system is 10.

610 is 6 hundreds, 1 ten, 0 units.
or 623 $= 6 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$
= 6 hundreds + 2 tens + 3 units
a general expression for any number in a system of base R is
$N = d_n R^n + \ldots d_3 R^3 + d_2 R^2 + d_1 R^1 + d_0 R^0$

Where N is number
d_n is digit in that position
R is base or radix

1.3 Binary number system

This number system uses symbols 1 and 0 and hence the base of the system is 2.

N = ... + 8d3 + 4d2 + 2d1 + d0

Where d_3 , d_2 , d_1 ,d_0 are either 0 or 1.

E.g. 1101 is $1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
= 8 + 4 + 0 + 1
= 13

Though binary system is wasteful of space needing four digits to specify a number, that only required one digit in decimal, the system is used largely in computer because of simplicity.

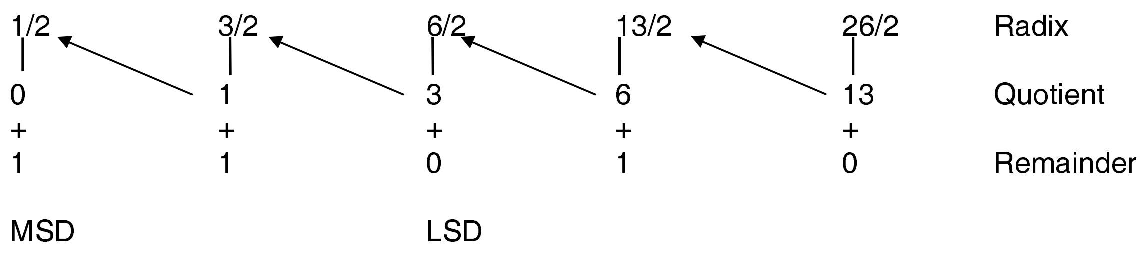
1.4 Conversion of Binary to Decimal

Add the value for each position of number having binary digit of 1

E.g. Binary 10110 can be converted to Decimal as
N $= 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
= 22

1.5 Converting Decimal to Binary

Repeatedly divide by 2 using remainder as conversion value e.g. to convert 26 to binary.



Read the last remainder as MSD (most significant digit) and first remainder as LSD (least significant digit).

1.6 Fractional binary numbers

Any fractional number can be written as

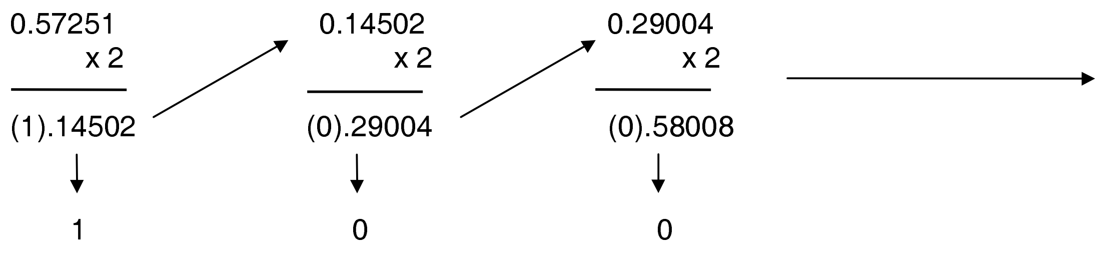
$$N = d_1 \times R^{-1} + d_2 \times R^{-2} + d_3 X\ R^{-3} + ...d_n \times R^{-n}$$

1.7 Converting binary fractional to decimal

$$\begin{aligned} \text{E.g. } (0.101101)_2 &= 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4} + 0 \times 2^{-5} + 1 \times 2^{-6} \\ &= 0.5 + 0.125 + 0.635 + 0.015625 \\ &= (0.703125)_{10} \end{aligned}$$

1.8 Converting decimal fraction to binary

E.g. : $(0.57251)_{10}$



Answer is $(0.10010...)_2$

1.9 Octal number system

Octal is base 8 system. It is popular because it converts easily to binary. Digits are 0 to 7.

1.10 Conversion from Octal to Decimal

$$\begin{aligned} \text{e.g. } (37)_8 &= 3 \times 8^1 + 7 \times 8^0 \\ &= 24 + 7 \\ &= (31)_{10} \end{aligned}$$

1.11 Converting Decimal to Octal

Repeatedly divide for integer part of the number
E.g. $(127)_{10}$ =

$$\begin{tikzcd}
1/8 & 15/8 & 127/8 & \text{Radix} \\
0 \arrow[u, dash] & 1 \arrow[u, dash] \arrow[ul] & 15 \arrow[u, dash] \arrow[ul] & \text{Quotient} \\
+ & + & + & \\
1 & 7 & 7 & \text{Remainder}
\end{tikzcd}$$

$(127)_{10} = (177)_8$

1.12 Hexadecimal number system

It is a base 16 number system having digits from 0 to 9 and letters A, B, C, D, E, and F.

1.13 Converting Hex number to Decimal

$$\begin{aligned} \text{E.g. } (23)_{16} &= 2 \times 16^1 + 3 \times 16^0 \\ &= 32 + 3 \\ &= (35)_{10} \end{aligned}$$

$$\begin{aligned} (1F)_{16} &= 1 \times 16^1 + F \times 16^0 \\ &= 16 + 15 \\ &= (31)_{10} \end{aligned}$$

$$\begin{aligned} (A2)_{16} &= A \times 16^1 + 2 \times 16^0 \\ &= 10 \times 16 + 2 \\ &= (162)_{10} \end{aligned}$$

1.14 Converting Decimal to Hexadecimal

E.g.$(152)_{10}$

```
    9/16          152/16
↓   ↖       ↓
    0              9
    +              +
    9              8          = (98)16
```

E.g. $(249)_{10}$

```
15/16   249/16
  ↖
0       15
+       +
15(F)   9        =      (F9)16
```

1.15 Binary Octal Conversion

$(275)_8$	=	010	111	101
		$(010111101)_2$		
$(101110110)_2$	=	101	110	110
	=	$(566)_8$		

1.16 Binary Hexadecimal Conversion

$(011011110101)_2$	=	0110	1111	0101
	=	6	F	5
	=	$(6F5)_{16}$		
$(1A6)_{16}$	=	0001	1010	0110
	=	$(000110100110)_2$		

1.17 Binary addition

0 + 0 = 0
1 + 0 = 1
1 + 1 = 0   WITH 1 CARRY
E.g.

```
  001101          13
+ 100101     +    37
--------------------
  110010          50
```

1.18 Binary Subtraction

```
0 - 0 = 0
0 - 1 = 1          WITH ONE BORROW
1 - 0 = 1
1 - 1 = 0
```

1.19 Complement Operations

One's complement of 1 = 0
One's complement of 0 = 1

Two's complement = One's complement plus 1
To find 1's and 2's complement of the number
N = 101101

1's complement of N = 010010
2's complement of N = 010011

Subtraction using 1's and 2's complement

```
110010           110010            110010
101101  ──→      010010  1's ──→   010011     2's
───────         ───────           ───────
                1010100           ~~1~~010101 Ignore, carry
                 ↘     1 end around carry
               ────────
                 010101
```

1.20 Binary Multiplication

```
     110101          53
   x  111            x7
───────────────────────────
     110101         371
    110101
   110101
───────────────────────────
  101110011
```

1.21 Binary Division

```
                      1100    ──→    Quotient          12
Divisor ←──     110 ) 1001000 ──→    Dividend      6 ) 72
                     -  110
                      00110
                      -  110
                     0000000 ──→     Remainder
```

1.22 Codes

Coding of information is a means of specifying characters using other symbols. Codes are used for security, so that other will not be able to read the message.

BCD = 8421 CODE

BCD-binary coded decimal is basic code. It uses binary number system to specify decimal numbers 0 to 9. Numbers 0 to 9. Numbers 0 to 9 are written in forms of "0's and "1's. Hence, it is binary code. This code requires four bits to specify a one digit decimal character. E.g. 22 in Decimal is 00100010 in BCD weight of right most position is 2^0 or 1, of second is 2^1, third 2^2, and code is also called an 8-4-2-1 code.

This code is not same as binary numbers.

Ten in binary is 1010 and in BCD are 00010000
$(16)_{16}$ in binary is 10000 and in BCD is 0010110

1.23 Excess-three code

It is modified form of BCD. Each coded character in excess 3 code is 3 larger in BCD. SIX is written as 1001.

Decimal	BCD	Excess-3
25	00100101	01011000
629	011000101001	100101011100

Excess - 3 is not a weighted code.

1.24 ASCII Code

A standard code is accepted by industry. American Standard codes for Information Interchange code is 7-bit code + 8th bit used for parity. It has lowercase, uppercase characters, special characters and command operations.

1.25 ASCII Code Table

Example: A = 41 = 100 00001

SB\ MSB	Hex Binary	0 000	1 001	2 010	3 011	4 100	5 101	6 110	7 111
0	0000	NUL	DLE	SP	0	@	P		p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	"	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101			%	5	E	U	e	u
6	0110			^	6	F	V	f	v
7	0111			&	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	I	y
10	1010	LF	SUB	*	:	J	Z	j	z
11	1011	VT	ESC	+	;	K	[	k	{
12	1100	FF	FS	‘	<	L	\	l	|
13	1101	CR	GS	-	=	M	]	m	}
14	1110	SO	RS	.	>	N	^	n	~
15	1111	SI	US	/	?	O	_	o	DEL

This table may not be remembered.

1.26 Gray Code

It is a non-weighted code, in this case only a single bit (from LSB side) changes between each successive word.

Decimal	Binary	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010

1.27 Gray to Binary Conversion

1. Start with MSB of Gray code word. Binary bit is same as Gray upto and including first 1.
2. Now if gray code bit is 1, change preceding binary digit to obtain present binary digit. If 0, repeat the same digit.

E.g. Convert Gray code Number 10101100111

$$\begin{tikzcd}
\text{Gray} & 1 \arrow[d] & 0 & 1 \arrow[dl] & 0 & 1 \arrow[dl] & 1 \arrow[dl] & 0 & 0 & 1 \arrow[dl] & 1 \arrow[dl] & 1 \arrow[dl] \\
\text{Binary} & 1 & 1 \arrow[r] & 0 & 0 \arrow[r] & 1 \arrow[r] & 0 & 0 & 0 \arrow[r] & 1 \arrow[r] & 0 \arrow[r] & 1
\end{tikzcd}$$

Convert Gray number 1111000101011 in to Binary

$$\begin{tikzcd}
\text{Gray} & 1 & 1 \arrow[dl] & 1 \arrow[dl] & 1 \arrow[dl] & 0 & 0 & 0 & 1 \arrow[dl] & 0 & 1 \arrow[dl] \\
\text{Binary} & 1 \arrow[r] & 0 \arrow[r] & 1 \arrow[r] & 0 & 0 & 0 & 0 \arrow[r] & 1 & 1 \arrow[r] & 0
\end{tikzcd}$$

1.28 Binary to Gray Conversion

1. Starting with MSB, compare each pair of succeeding bits.
2. If they are the same place, put a 0 in gray code word.
3. If they are different place, put a 1 in Gray code. Compare first binary digit to 0 to start.
   E.g. Convert 011010111101 to Gray code.

$$\begin{tikzcd}
\text{Binary} & {} \arrow[r] & 0 \arrow[r] \arrow[d] & 1 \arrow[r] \arrow[d] & 1 \arrow[r] \arrow[d] & 0 \arrow[r] \arrow[d] & 1 \arrow[r] \arrow[d] & 0 \arrow[r] \arrow[d] & 1 \arrow[r] \arrow[d] & 1 \arrow[r] \arrow[d] & 1 \arrow[r] \arrow[d] & 1 \arrow[d] \\
\text{Gray} & & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0
\end{tikzcd}$$

NOTE: Conversion of Gary to Binary and Binary to Gary need not be studied from examination point of view.

1.29 Parity in Codes

Bit Parity: A popular means of detecting an error is the use of parity bits. Parity can be either odd or even; addition of a parity bit will make the total number of 1's in a Code character either an odd number or an even number.

In even parity the added parity bit will make the total number of 1's an even amount.

In odd parity the added parity bit will make the total number of 1's an odd amount.

When a code word is received it is checked for parity (off or even being chosen previously) and is accepted as correct if it passes the test.
Word Parity: With bit parity it is possible to detect a single error. More than one error occurring is not detected. Word parity enables detecting and correcting the single bit error.

Computer does parity check of each word and of each column. Each word has one parity bit and in a block of words there is one parity word, which allows for error correction.

Review Questions

1. Why only Binary numbers system is used in computer (Digital) ?
2. Convert Decimal No.489 to Binary Equivalent.
3. Convert Binary No. 101100111010 to Decimal Equivalent.
4. Explain how parity codes are generated?
5. Convert decimal fraction $(0.5275)_{10}$ to Binary.
6. Find 2's complement of a binary Number 010111010.
7. Convert $(252)_{10}$ Decimal to Hexadecimal Number.
8. Convert Decimal Number $(128)_{10}$ to Octal Number.

NOTES

NOTES

NOTES

NOTES

NOTES

Chapter Two

INTEGRATED CIRCUITS

Features :

2.1 INTRODUCTION:

Many complex digital functions have been realised in a variety of forms and each form is referred to as a logic family. There are two types of semiconductor devices: Bipolar and Unipolar. Accordingly ICs are referred to as Bipolar Logic Family or Unipolar Logic Family.

2.2 Bipolar Logic Family

There are two types of operations in bipolar ICs. 1) Saturated and 2) Non-saturated.

2.3 Saturated Bipolar Logic Families are:

1. Resistor - transistor logic (RTL)
2. Direct coupled transistor logic (DCTL)
3. Integrated - injection logic (I2L)
4. Diode transistor logic (DTL)
5. High threshold logic (HTL)
6. Transistor - transistor logic (TTL)

2.4 Non-saturated bipolar logic families are:

1. Schottky TTL
2. Emitter - coupled logic (ECL)

2.5 Unipolar logic families

MOS devices are Unipolar and only MOSFETs are used in MOS logic circuits. MOS logic families are :

1. PMOS
2. NMOS
3. CMOS

2.6 Characteristics of Digital ICs

Classification based upon number of gates can be given as;

Small-scale integration	gates less than 12
Medium scale integration	gates 12 or more but less than 100
Large scale integration	gates 100 or more but less than 1000
Very large scale integration	gates 1000 or more.

2.7 Characteristics used to compare the performance are :

1. Speed of operation
2. Power dissipation
3. Fan-out
4. Current and voltage parameters
5. Noise immunity
6. Operating temperature
7. Power supply requirements
8. Flexibility's available

RTL, DTL, and DCTL families are no more used for new systems because of their low speed, high power dissipation and low fan-out.

2.8 Integration -Injection logic (I^2L)

I^2L has simplicity, used very small silicon chip area, consumes very little power, requires only four masks and two diffusions and hence is easier and cheaper to fabricate.

I^2L technology is the concept of merging components, one semiconductor region is part of 2 or more devices.

Basic operation of **I^2L** is explained with the help of inverter circuit. If input Vi is at low level, T1 is off so that I B1 = 0, input source acts as a sink for current L1. Therefore, **I^2L** flows through base of T2 driving in saturation when T1 is off, and T2 is on, V BE2 = V CE1~0.8V.

If input is at HIGH level, I B1, will have two components I1 and other due to Vi. T1 saturates. Therefore, V CE1 = V CE sat ~ 0.2 V, which drives T 2 to cut off and T 1 acts as a sink for I2. This shows that logic level at VO is complement to that of Vi. T1 acts as an inverter, logic swing is about 0.6V.

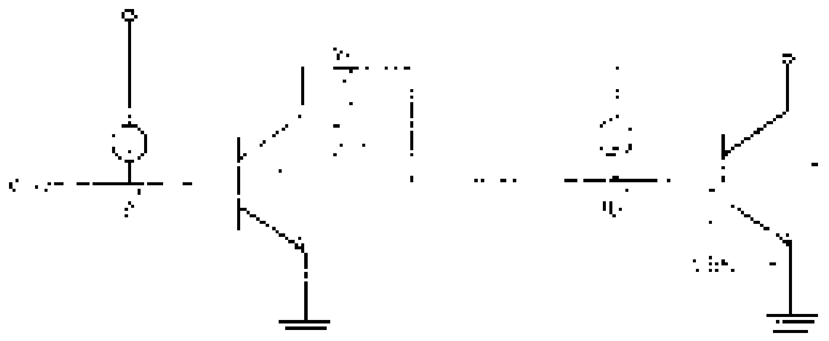


Fig. 2.1 I^2L Technology

Simplified physical structure of a portion of I^2L circuit is shown in fig. 2.2. as :

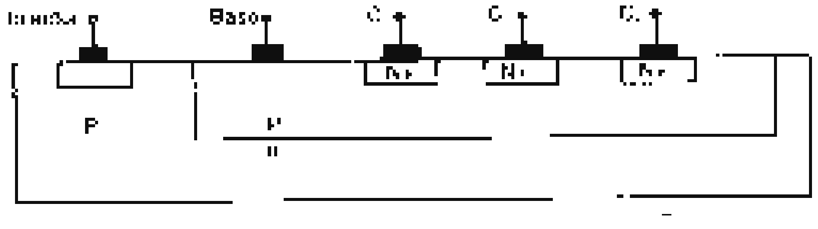


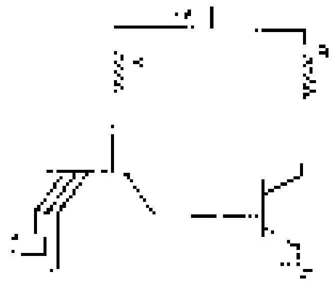
Fig. 2.2. Simplified Physical Structure of I^2L Circuit

Speed of operation depends upon charging current. Propagation delay time is inversely proportional to charging current. Power dissipation is also proportional to charging current. Package density is in range of 120 to 200 gates per square mm. of IC.

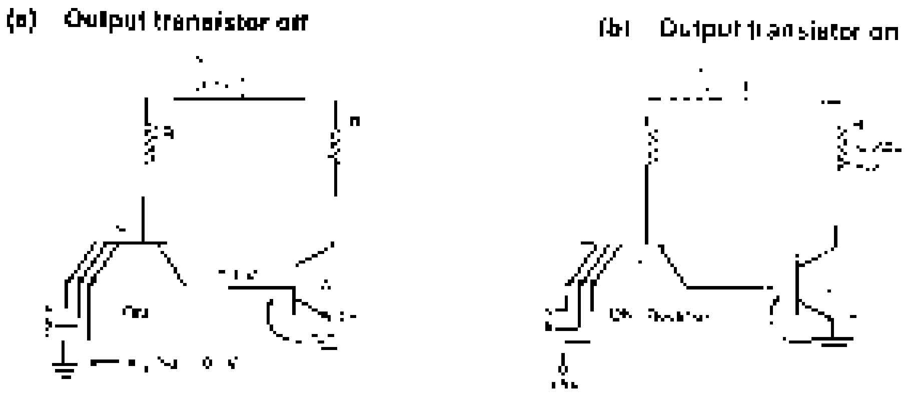
2.9 High - Threshold Logic HTL: Due to pressure of electric motors, on-off control circuits, voltage switches, in an industrial environment, the noise level is quite high. For this purpose, DTL gates are redesigned with higher supply voltage (15V).

2.10 Transistor - transistor logic (TTL)

TTL is most popular general purpose family. It is available in 5 different series. Basic TTL NAND gate is shown in figure. Any input of logical 0 will result in Q1 being driven on Q held off and output =+VCC. Only if all inputs are logic - 1, q is on and output is logical 0.



Operation of basic TTL circuit

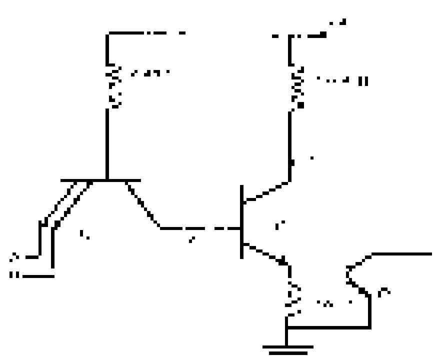


Output transistor off : If one input is OV, Q1 is driven on providing low collector-emitter saturation voltage of Q1 as input to base of Q2. Base of Q2 is held at 0.05 V, Q2 remains off, output going upto +5V.

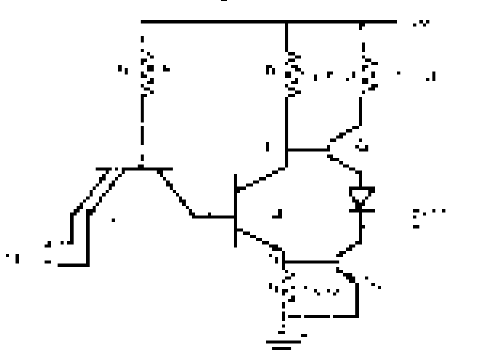
Output transistor on : With all inputs at +5V or left open, q1 operates in reverse mode, driving Q2 on, output then is approximately equal to 0.05V.

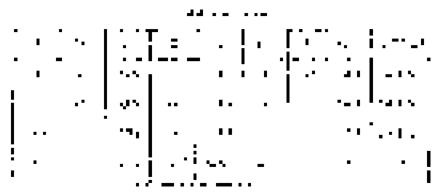
TTL Circuit output connections: Different types of output connections are made available in TTL logic gates. For example the Totem pole outputs will give high switching speed and high current outputs, open collector outputs viz. open, close, and high impedance states. Output connections for the above configuration are shown below :

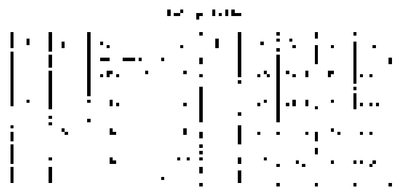
Collector Output.



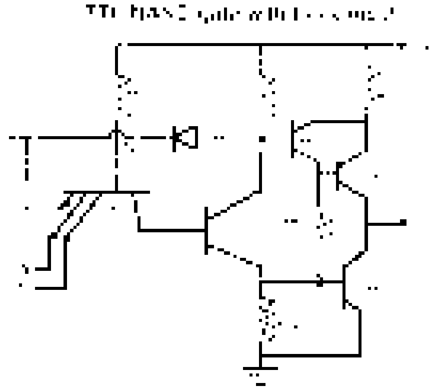
Basic TTL NAND gate with Totem Pole Output

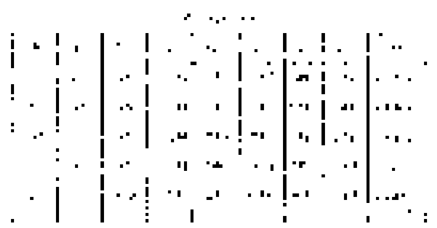






TTL NAND gate with [illegible]





Schottky TTL : Speed limitation of TTL is due to turn off time delays in transistors going from saturation to cut-off. Using Schottky transistors eliminates this. Transistor is prevented from entering saturation and there is saving of turn off time.

Category	**TTL ICs prefix**	**Examples**
Standard TTL	74-	7402, 74193
High Power TTL	74 H-	74H02, 74H193
10 W Power TTL	74 L-	74L02, 74L193
Schottky Power TTL	74 S-	74S02, 74S193
10 W Power Schottky TTL	74 LS-	74LS02, 74LS193

2.11 Emitter - Coupled Logic (ECL) : ECL is fastest of all logic families. Transistors are used in difference amplifier configurations. Therefore, they are not driven in saturation and storage time is eliminated. Emitters of two transistors are connected ECL gate has three parts, middle is difference amplifier which performs logic operation. Emitter followers are for dc level shifting of outputs. Additional transistors are used in parallel to T1 to get required fan-in. Basic difference between ECL and all other families is, in ECL, positive end of supply is connected to ground. This is to minimise the effect of noise in power supply and protection of gate from short circuits.

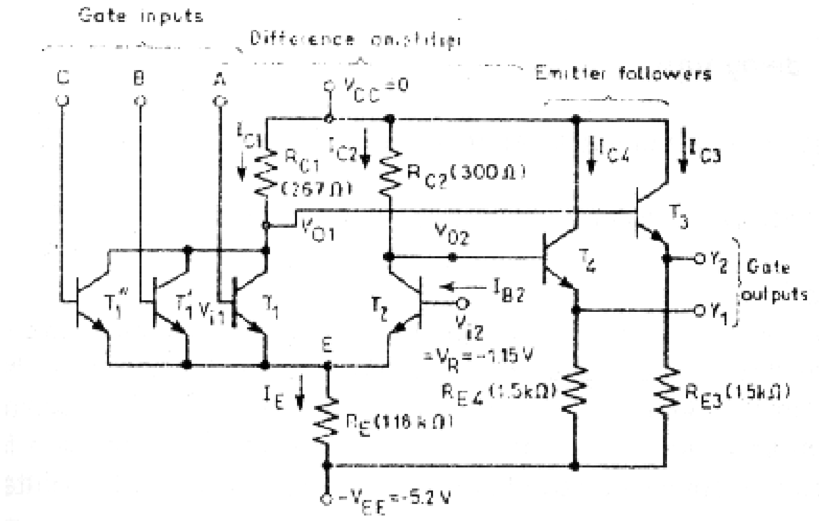


Fig. 2.3. Emitter - coupled Logic (ECL)

2.12 MOS Logic Families : MOS devices occupy very small area and require very small power. Disadvantage of MOS logic is slow speed. The circuits in which only n or p channel devices are used are NMOS and PMOS logic respectively. Enhancement mode p and n channel MOS devices fabricated on same chip are complementary MOSFETs and logic based on those in CMOS Logic.

Basic MOS gate is an inverter as shown in figure. T1 is an enhancement MOSFET acting as a driver. T is an enhancement depletion MOSFET acting as load. Instead of resistor for load, MOSFET itself is used as load.

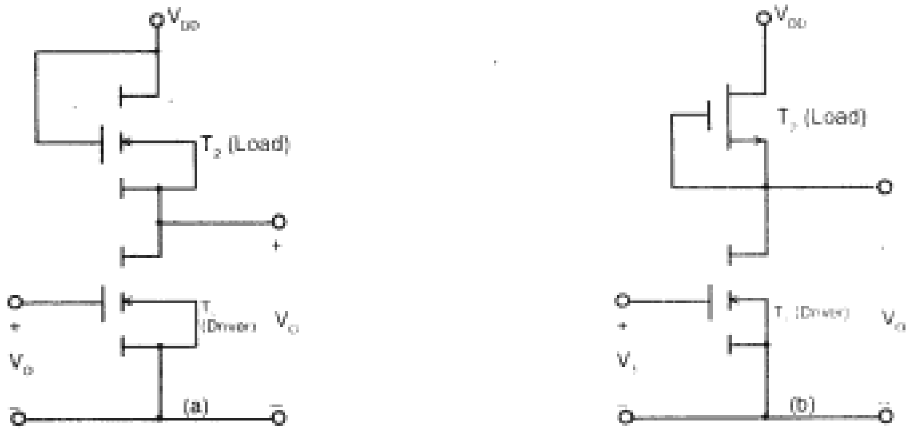


Fig. 2.4. A Mos Inverter with
(a) Enhancement Load (b) Depletion Load

Fan-out: MOS devices have high input impedance, hence fan-out is large.

2.13 Propagation delay time: It is very large because of large capacitance at input and output of MOS devices.

Power Dissipation: Power Consumption in MOS circuits is small.

CMOS LOGIC : A complementary MOSFET is obtained by connecting p and n channel MOSFET in series, with drains tied together and output is taken at common drain. Input is applied at common gate by connecting gates together. In CMOS, package density is reduced because p and n channel enhancement MOS devices are fabricated on same chip. Small power consumption hence ideally suited for battery operated systems.

CMOS INVERTER :

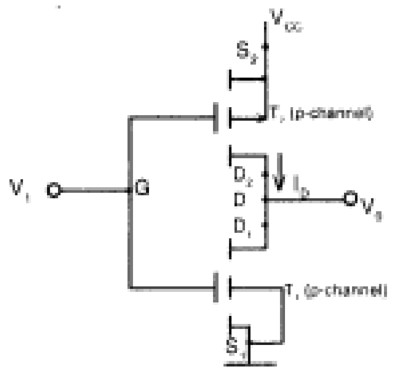
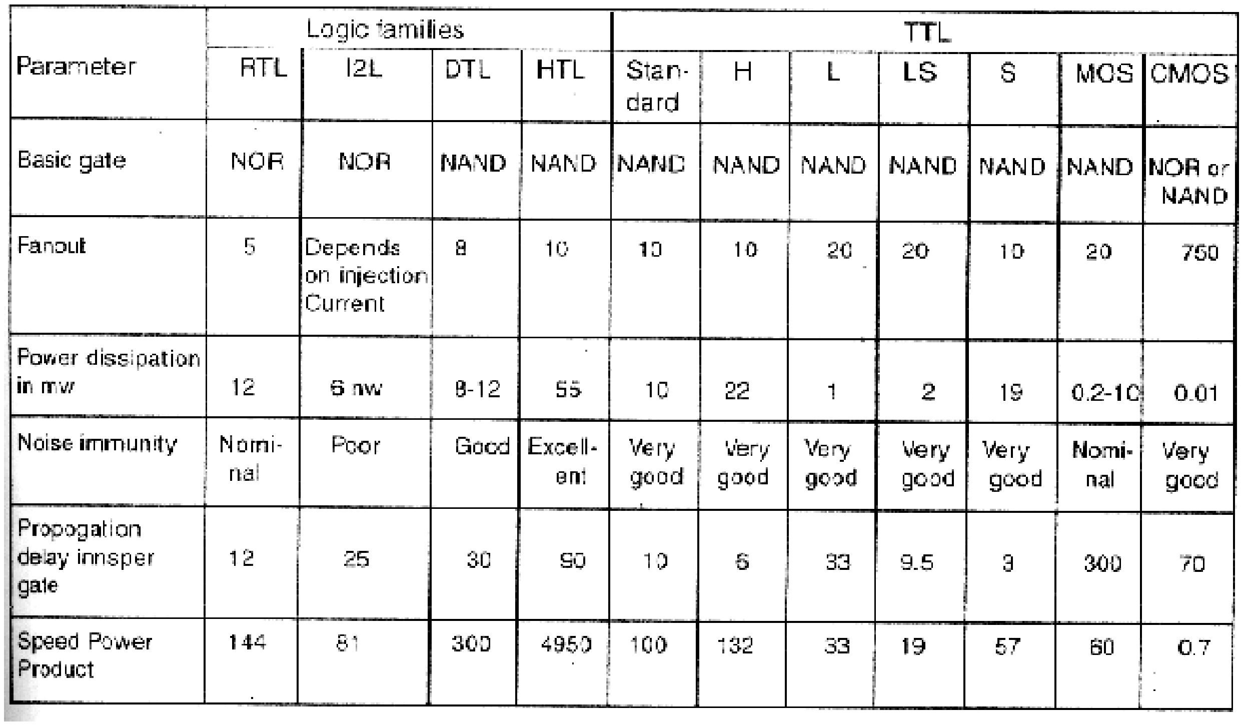


Fig. 2.5. CMOS Inverter

Logic levels are 0V and V_{cc}, when Vi = V_{cc} T_1 turns on and T_2 off. Therefore, V_0 ~ $0V_1$ T_1 turns off and T_2 on, output voltage V_0 .... V_{cc}. I_n in both cases is small. Two popular series 4000 and 54C/74C series are available and it is compatible with 54/74 TTL family.

2.14 Comparison of Digital IC logic families



Parameter	Logic families				TTL						
	RTL	I2L	DTL	HTL	Standard	H	L	LS	S	MOS	CMOS
Basic gate	NOR	NOR	NAND	NAND	NAND	NAND	NAND	NAND	NAND	NAND	NOR or NAND
Fanout	5	Depends on injection Current	8	10	10	10	20	20	10	20	750
Power dissipation in mw	12	6 nw	8-12	55	10	22	1	2	19	0.2-10	0.01
Noise immunity	Nominal	Poor	Good	Excellent	Very good	Very good	Very good	Very good	Very good	Nominal	Very good
Propogation delay innsper gate	12	25	30	90	10	6	33	9.5	3	300	70
Speed Power Product	144	81	300	4950	100	132	33	19	57	60	0.7

2.15 Packaging Techniques:

There are two kinds of packaging techniques used for semiconductor IC assembly: Through Hole, Surface Mount assembly. The packaging of various semiconductor chips are based on following functional aspects :

1. The packaging should meet the ever-increasing number of interconnections or pin counts for the IC.
2. It should protect the chip from moisture, light, which normally affects the IC operation and life.
3. IC should be manufactured so that it can dissipate the heat build up inside the IC due to the functioning of the chip.

Various techniques of packaging are available listed below.

Category : Through Hole : In this category the highest number of pin counts available is 64 (DIP). Beyond this the physical dimensions limits the use of package. A higher count is obtained through PGA which is a two dimensional pattern. This package has some limitations. Finer pitch sizes result in thinner pins which are more sensitive to handling and to insertion into sockets and are generally more difficult to straighten.

SIP : Single in line package

SIP-tab : Single In Line package with metal heat sink
DIP : Dual In Line package
DIP-tab : Dual In Line package with metal heat sink
SIMM : Single IN Line Memory Module
QUIP-tab : Quad IN Line Package with metal heat sink

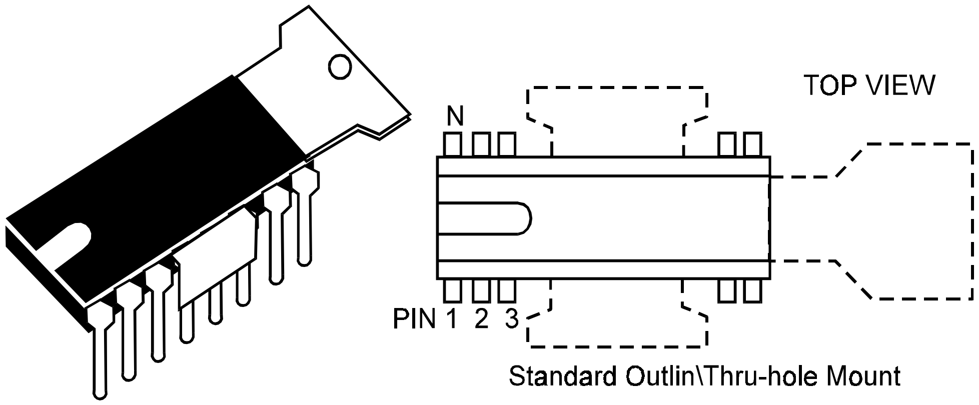


Fig. 2.6 Dual Inline Package with Metal heat sink

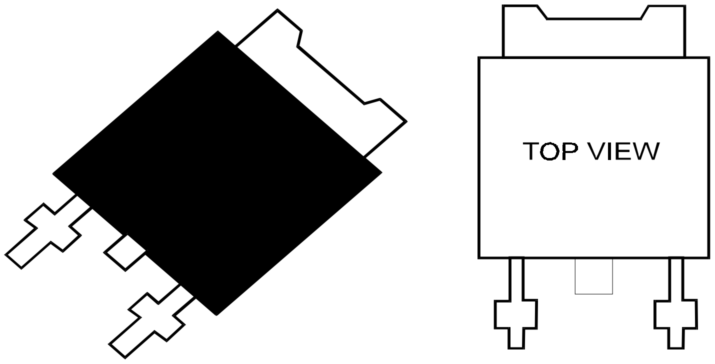


Fig. 2.7 Small Outline (Surface Mount)

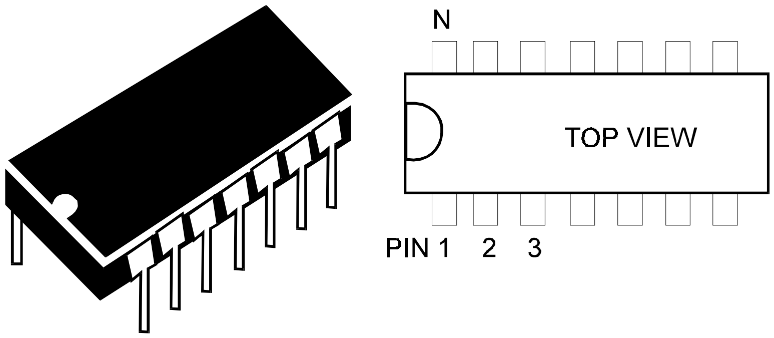


Fig 2. 8 Dual Inline Package

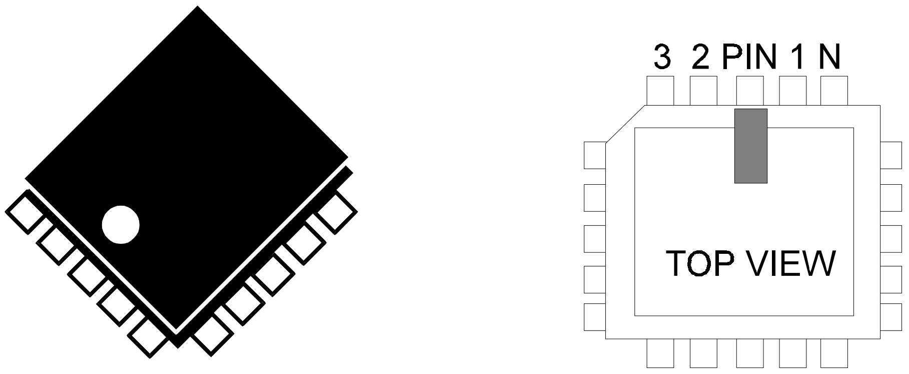
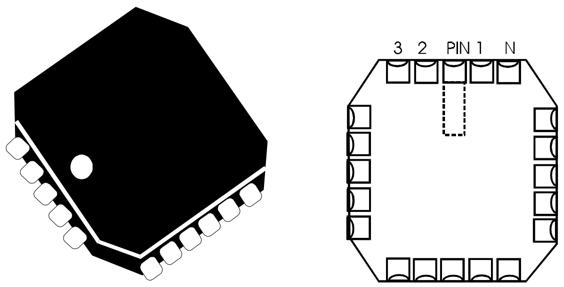


Fig 2.9. Lead Less Chip Carrier



PGA : Pin Grid Array

Surface Mount Category: Surface mount packaging allows better utilisation of the chip area by a much reduced pitch of the leads. Use of SMD results in reduced board area thereby saving in total cost. The trend of packaging is inclined towards surface mount devices because of the higher Input/ Output pin counts. A liquid flux is used for soldering these devices to the board. As the lead pitch is very less hence ordinary machines are not used for soldering. In surface mount technology, different soldering techniques are used, in which finner solder tips are used. The soldering is done at low temperature, which is less than 200 degree centigrade in comparison to standard 300 degree centigrade.

SOP : Small outline package DIP
QFP : Flat package, Quad style
PLCC: Plastic Leaded Chip Carrier
SOJDIP : Small Outline DIP With "J" Leads

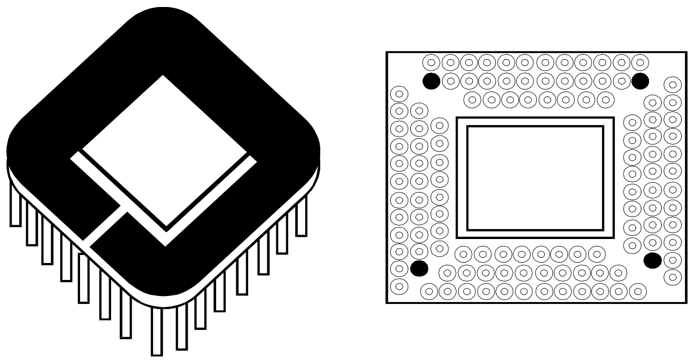


Fig. 2.10. Pin Grid Array

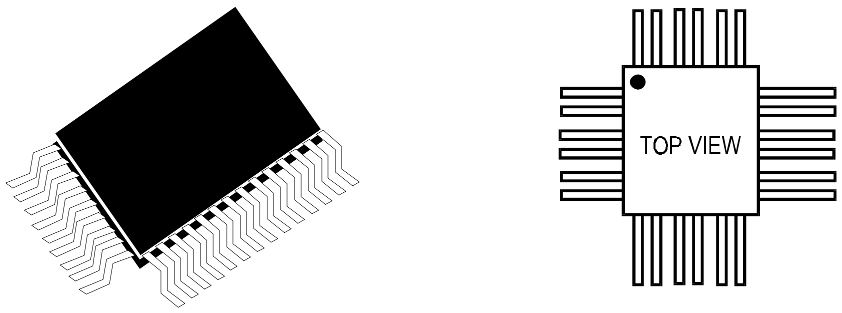


Fig.2.11. Flat Package, Quad Style (Straight-Lead and Gull-Wing)

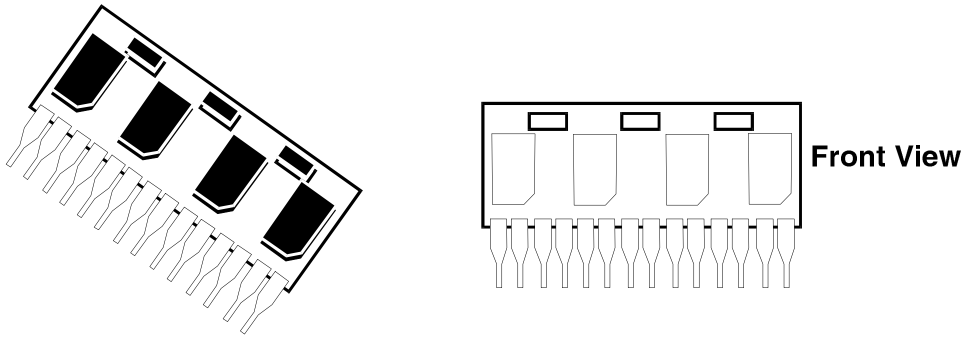


Fig. 2.12. Single Inline Memory Module

Review Questions

1. A standard TTL gate drives five similar gates
   a) When its output is HIGH, how much current can it source ?
   b) When its output is LOW, how much current it sinks?
2. Which TTL sub-family is best for LOW power consumption.
3. List three types of TTL ?
4. What is the fanout of digital IC ?
5. Show the interfacing of CMOS NAND gate directly driving LED so that the indicator light glows (ON) when gate out put is HIGH. use a +10V power supply.
6. List some disadvantages of CMOS ICs over TTL ICs ?
7. When a designer selects a logic family What two very important characteristics must be considered ?

8. Discuss why noise may be less of a problem with TTL even though TTL has higher noise immunity.
9. What is a logic high level & low level voltage TTL IC's ?
10. What is Active pull up ? Explain in detail.

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

Chapter Three

OTHER DIGITAL CIRCUITS

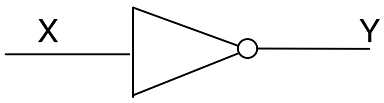
Features :

3.1 Introduction

The digital circuits operate in two states. In binary number system we are using 1,0. Now the voltage level can represent these. The high voltage level is represented by 1 and the low level is represented by 0.Digital circuits perform various logical operations and are commonly termed as logic circuit.

3.2 Logic Gates: Basic logic gates used for the logical operation are inverter gate, AND gate and OR gate.

Inverter: An inverter is a gate, which has only one input and one output. The output is the complementary of the input. It is also known as NOT gate. The logical symbol , equation and the truth table is shown in the fig. 3.1 below.

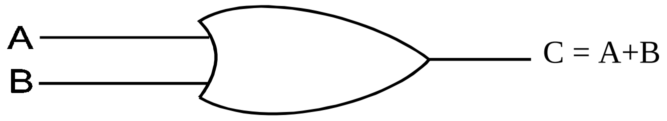


$Y=\overline{X}$

X	Y
0	1
1	0

Fig No. 3.1 NOT GATE

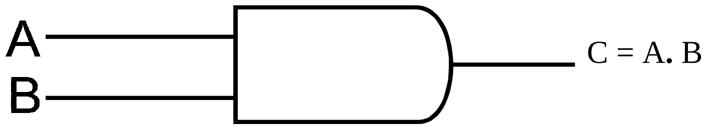
OR Gate: OR Gate is a two or more input gate in which the output is high when one or all the inputs are high. The logic symbol ,equation & the truth table are shown in fig.3.2



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

FIG 3.2 (a) TWO INPUT OR GATE (b) TRUTH TABLE

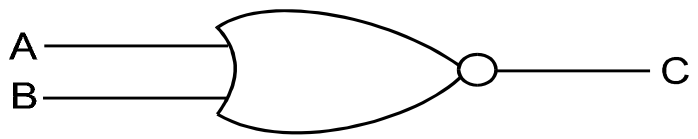
AND Gate : AND Gate is a also a two or more input gate in which the output is high when all the inputs are high .The logic symbol, equation and the truth table are shown in fig. 3.3



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

Fig 3.3 (a) AND GATE (b) TRUTH TABLE

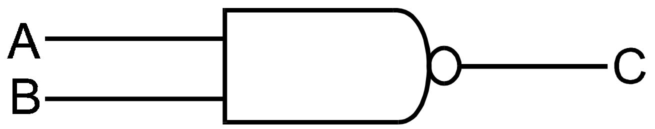
NOR Gate: NOR gate is an inverted OR gate, in which the OR gate is followed by the inverter gate. The output is low when one or all the inputs are high . See fig. 3.4



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

Fig 3.4 (a) NOR GATE (b) TRUTH TABLE

NAND gate : NAND gate is inverted AND gate in which the AND gate is followed by the inverter which is represented by the bubble. The output is low when all the inputs are high. The logic symbol, equation and the truth table is shown in fig. 3.5



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

Fig. 3.5 (a) NAND GATE (b) TRUTH TABLE

The digital logic circuit can be optimised by using the De Morgan's Theorem, which reduces the number of logic gates. It has been observed that the NOR and the NAND gates can replace the basic logic gates OR, AND, NOT .

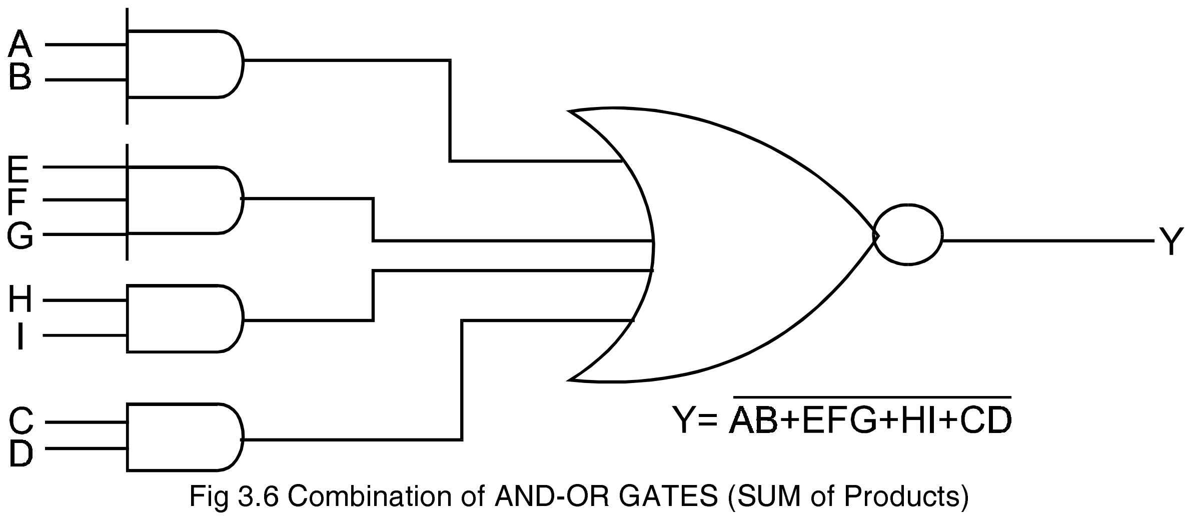


Fig 3.6 Combination of AND-OR GATES (SUM of Products)

In practical circuits a combination of various gates is used. Some of the possible combinations are shown in figure 3.6 & 3.7

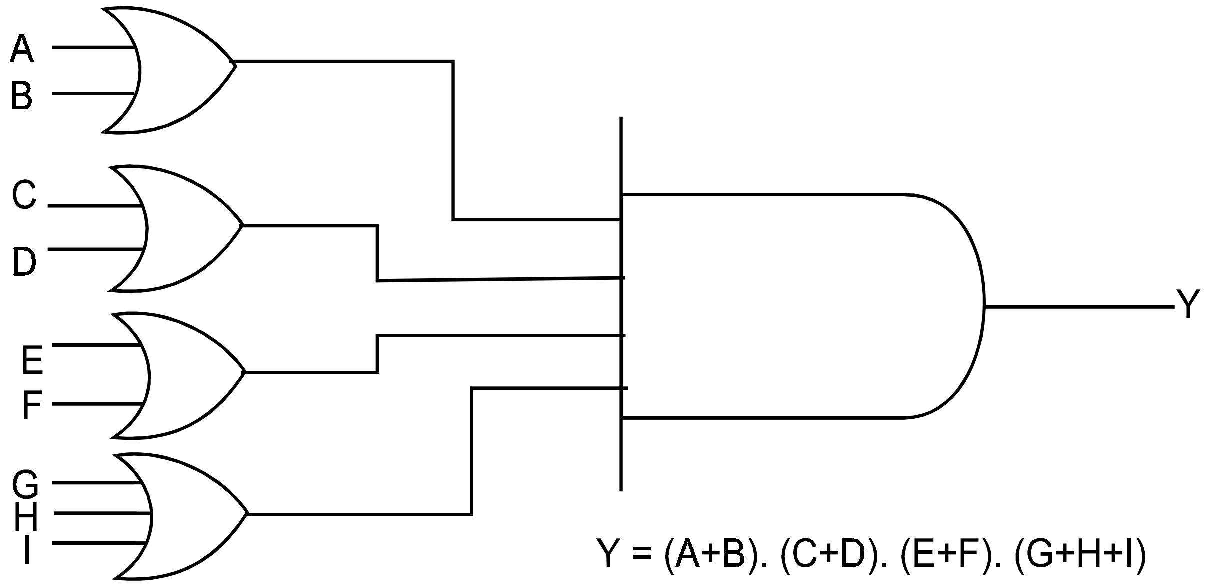
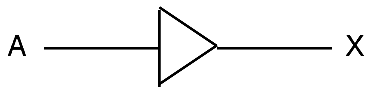
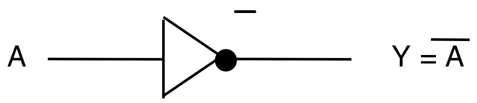


Fig. 3.7 Combination of OR -AND gates (Product of Sums)

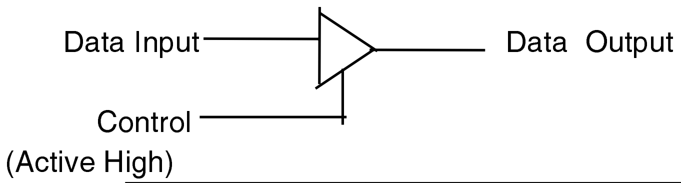
3.3 Unidirectional Buffers: The buffer is basically a NOT gate represented by a symbol. In the fig below A is the input and X is output.



The input is active high; the output X is the active low. If the bubble is put at the output then it indicates active low. The input has no bubble. So it is active high. You can read the function of the device directly from schematic symbol.



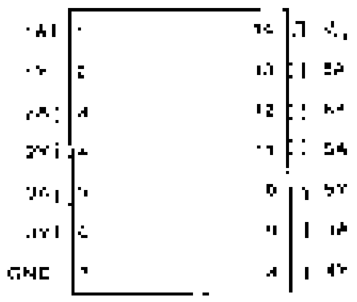
3.4 Tristate Logic : In normal logic operation there are two states of the output, LOW and HIGH. If the output is not in LOW state, it is definitely in other state. Similarly if the output is not in HIGH state, it is definitely in LOW state. In complex digital systems like microcomputers and microprocessors based systems, a number of gate outputs may be required to be connected to a common line which is referred to as bus which in turn may be required to drive a number of gate inputs. The third state is introduced called high impedance state, in addition to LOW and HIGH states. See fig. 3.8 below. The figure shows active high tristate buffer gate.



Control input	Data	Data output
0	0	HIGH-Impedance
0	1	HIGH-Impedance
1	0	0
1	1	1

Fig. 3.8 Active High Tristate Buffer Gate

HEX INVERTER WITH OPEN COLLECTOR OUTPUT 74LS 05



Pin Diagram of 74LS 05

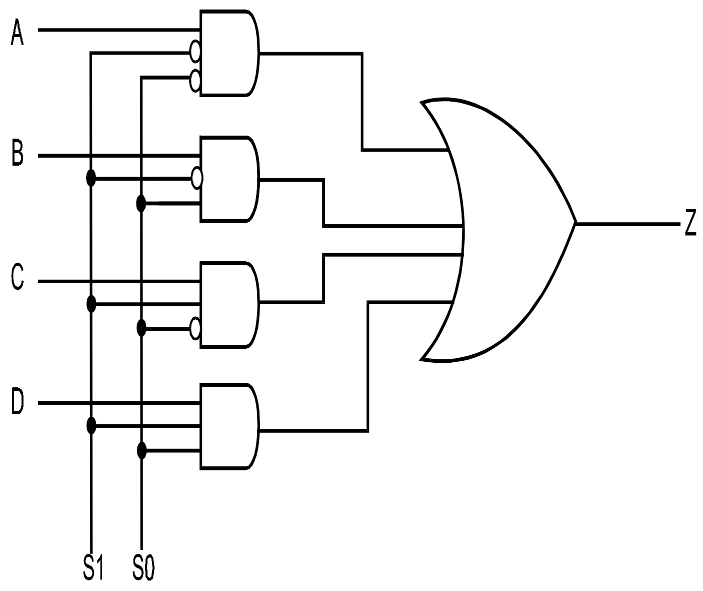
Functional Table

A	Y
Input	Output
H	L
L	H

Description: These devices contain six independent inverters. The open collector outputs require pull up resistors to perform correctly. They may be connected to other open collector outputs to implement active low wired-OR or active high wired AND functions.

3.5 Multiplexer, De-multiplexer and Encoder:

A multiplexer is a circuit with many inputs but with only one output. By applying control signals, we can steer any input to the output. Figure 3.9 illustrates the general idea. The multiplexer has 4 input signals, 2 control signal and one output signal. One of the popular multiplexers is the 16-to-1 multiplexer which has 16 input bits, 4 control bits and 1 output bit. Some times this multiplexer can be used as a design solution for any four-variable truth-table and therefore it is also called a universal logic circuit. A demultiplexer is a logic circuit with one input and many outputs. By applying control signals we can steer the input signal to one of the output lines. Figure 3.10 also shows a block diagram of a demultiplexer with 1 input signal, 2 control signals and 4 output signals. A commonly used example is 1-to-16 demultiplexer. A decoder is similar to a demultiplexer; with one exception-there is no data input. A 1-of-16 decoder is a very common example. In this only 1 out of 16 output lines will be active. An encoder converts an active input signal into a coded output signal. A very common example of an encoder is decimal-to-BCD encoder whose output is binary equivalent of a signal decimal input digit.



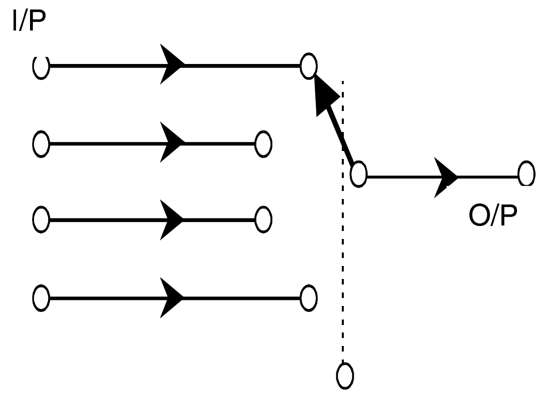


Fig 3.9 (a) 4 line-to-1 line Multiplexer (b) Function of Multiplexer

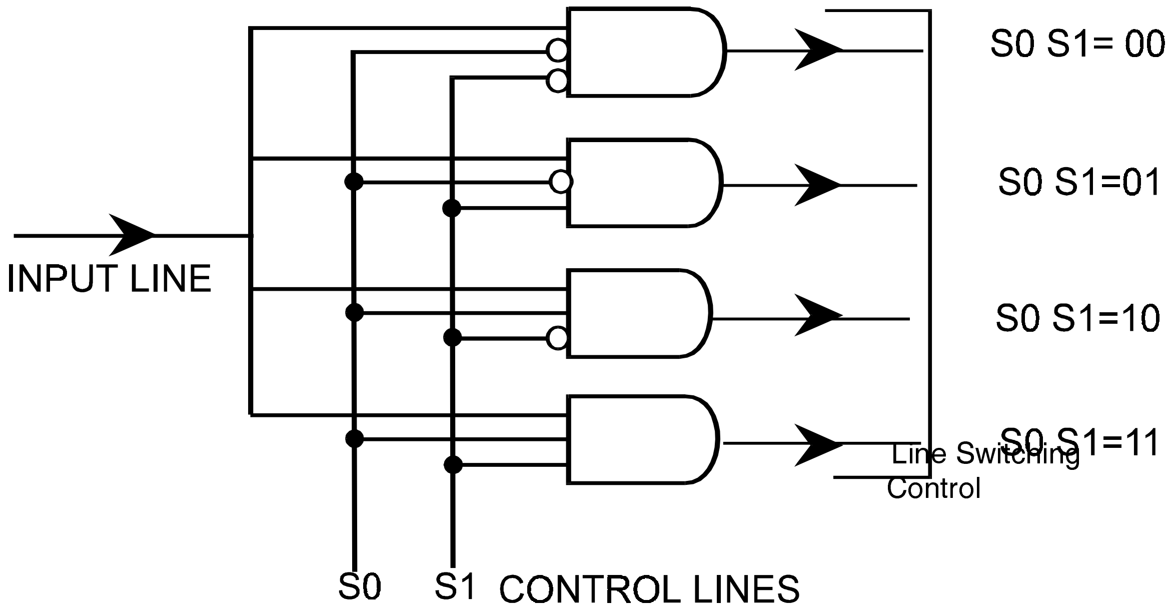
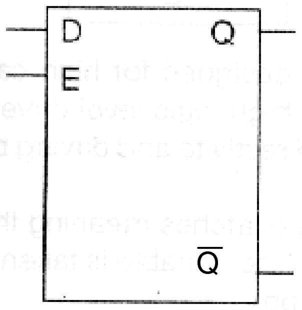


Fig. 3.10 1 line-to-4-line De-Multiplexer

3.6 Latch : A latch is a digital device that stores a one or zero on its output. Fig. 3.11 shows the schematic symbol .



D	E	Q	$\overline{Q}$
X	O	Q_N	$\overline{Q}_N$
0	1	0	1
1	1	1	1

Truth Table

Fig 3.11 Latch

The device functions as follows. If the enable input E is low any data present on D input will have no effect on Q or Q. This is indicated in truth table as X (Don't Care) in the D column. If the enable input is high, a high or low on D input will be passed to Q output. In other words, the Q output will follow the D input as long as enable input is high. The Q output will contain the complement of the logic state on Q, when the enable input is made low again, the state on Q at that time will be latched there. Any changes on D will have no effect on Q until enable input is made high again when the enable input goes low then, the state present on D just before the enable goes low will be stored on the Q output.

This D flip-flop is also called data flip-flop. When the flip-flop receives clock signal, the D input is transferred to the Q output. This flip-flop is commonly used for temporary storage of data and transfer of data from one unit to another unit in a computer. Another use of this flip-flop is as a frequency divider as shown in figure 3.12.

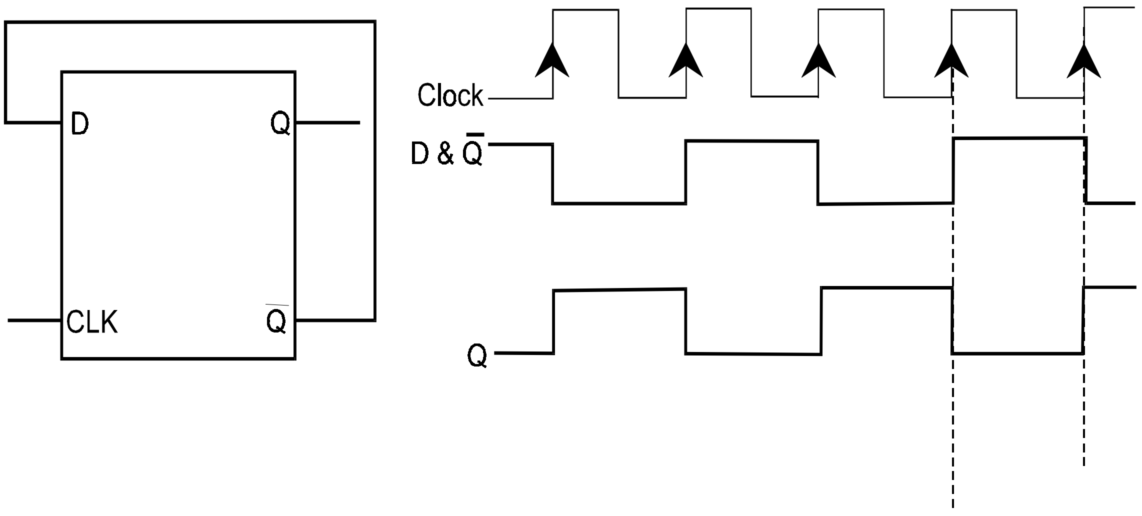


Fig 3.12 D Flip-Flop as frequency divider

Octal D-Type Transparent Latch and edge triggered flip flops (74LS 373)

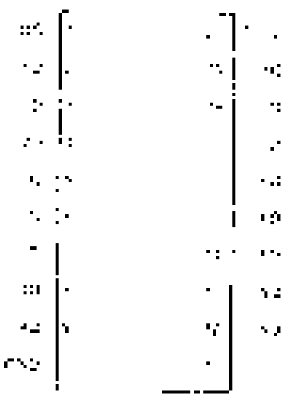


Fig 3.13 Pin Diagram of 74LS 373 (octal D-Type Latch)

These 8-bit register feature three state outputs designed for high capacitive loads. The high impedance third state and increased high-logic level drive provide these registers with the capability of being connected directly to and driving bus lines in a bus-organised system.

Output Enable	Enable Latch	D	Output
L	H	H	H
L	H	L	L
L			Q_0
H			Z

The eight latches of LS 373 are transparent D type latches meaning that while enable (c) is high Q output will follow data (D) inputs. When enable is taken low that output will be latched at the level of data that was set up.

Difference between a Flip-Flop and Latch

Any flip-flop is triggered by a clock-input signal. The data at D input of the flip flop enters a flip flop only when the flip flop is triggered by the clock input signal. Otherwise the data does not enter the flip-flop nor has effect on the flip-flop.

In case of a latch the data enters through the input lines as long as the chip (IC) is selected, or the latch is open (the gate input is high). When the gate input changes from high to low, the latch is closed. When the latch is closed the inputs do not enter the latch.

Thus in a flip-flop, the inputs are ignored before triggering and after triggering. In a latch, the inputs are allowed before closing and ignored after it.

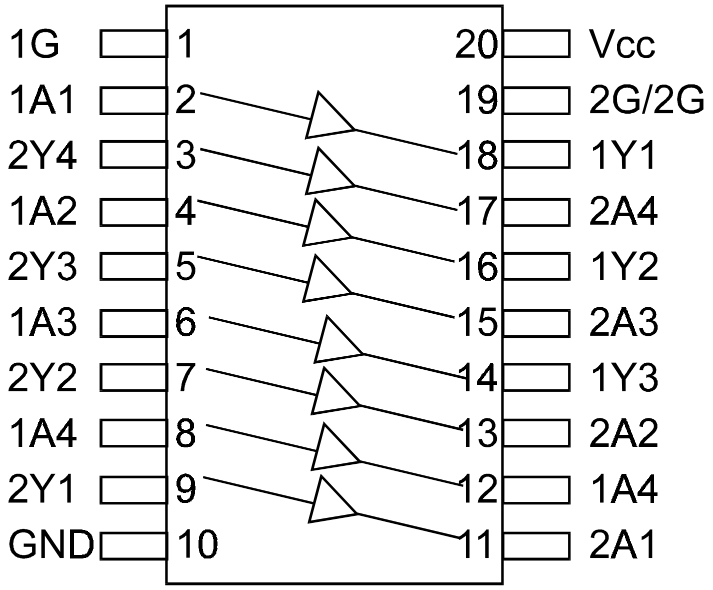


Fig. 3.14Pin Diagram of 74LS244 (Octal Buffer)

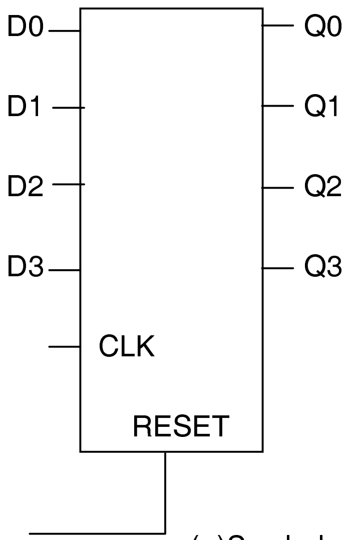
Description of 74LS244 : These octal buffers & line drivers are three state memory address drivers, clock drivers & bus oriented receivers & transmitters. Inverting & Non inverting outputs, symmetrical G (active low output control) inputs, complementary G & $\overline{G}$.

3.7 Registers and Counters :

A register is a group of flip-flops that work together as a unit. The simplest registers do nothing more than storing a binary word; other registers such as shift registers modify the stored word by shifting its bits left or right or by performing other operations. A buffer register is the simplest kind of register; all it does is to store a digital word.

A shift register moves the stored bits left or right. This bit shifting is essential for certain arithmetic and logic operation used in microcomputers there are two ways to shift data into a register (serial and parallel) and similarly two ways to shift the data out of the register. This leads to the construction of four basic register types: serial in-serial out, serial in-parallel out, parallel in-serial out, and parallel in parallel out.

A counter is a register capable of counting the number of clock pulses that have arrived at its clock input. It can also be used for measuring time and therefore period or frequency. In a ripple or serial or synchronous counter flip-flops are connected in cascade i.e. the output of a flop-flop drives the next flip-flop. Therefore it has speed limitation. On the other hand in a parallel or synchronous counter all the flip-flops operate simultaneously therefore they can operate at higher speeds.



CLK	Q0	Q1	Q2	Q3
↑	D0	D1	D2	D3

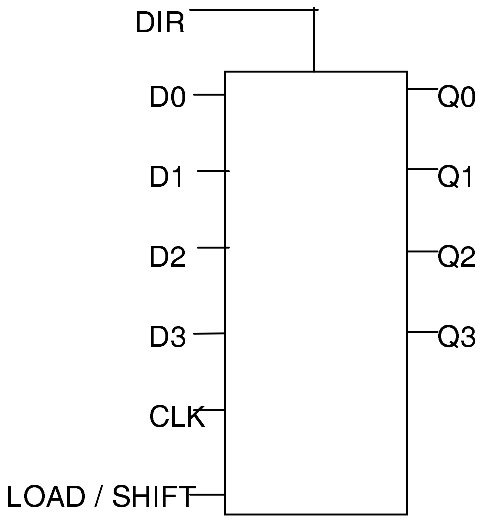
Reset input overrides the clock input
When Reset = 1, the register is cleared

(a)Symbol (b) Truth Table

Fig 3.15 Four Bit Register

SHIFT REGISTER : A shift register contents can be shifted either left or right. There are shift registers having Bi-directional shifting facility with control for left shift and right shift. The shifting of the contents is done by one bit shift to left or right, when the register is triggered by the clock input signal. Many shift registers are available

SISO : Serial In Serial Out
SIPO : Serial In Parallel Out
PIPO : Parallel In Parallel Out
PISO : Parallel In Serial out



If Dir = 0, Left Shift is Enabled ; If Dir = 1, Right Shift is Enabled
When Load/Shift = 1, Shifting is enabled; When Load/Shift = 0, the D0 to D3 data is loaded into the register. The clock shifts the register content by one bit.

Fig 3.16 Four bit Shift Register

Review Questions

1. What is the use of latch?
2. Can latch be used as flip-flop ?
3. What is the use of Buffers ?

NOTES

NOTES

NOTES

NOTES

Chapter Four

SEMICONDUCTOR MEMORIES

Features :

4.1 INTRODUCTION

The function of memories is to store program, data and results. There are two kinds of memories: semiconductor memories and magnetic memories. Semiconductor memories are faster, smaller, lighter, and consume less power. Used as main memory of computer. Memory stores the information on a temporary or a permanent basis for future use. Memory circuit is considered the heart of the microprocessor system as it stores the program as well as the data, which is responsible for the operation of the system.

4.2 MEMORY STORAGE CELL

Memory ICs store programs and data as sequence of binary digits (0s and 1s) where a 0 represents the absence of a signal voltage and a 1 represent the presence of a signal voltage. Each bit represents a voltage level and hence, the voltage must be stored in an electronic circuit. This circuit is known as a storage cell. The contents of any storage cell can be copied to a bus or other device (this process is known as reading). Storage Cells can also be set to new values by copying data from external bus signals (this process is known as writing). Storage cells can be combined / arrays as shown in fig. 4.1. The structure and construction of a storage cell depends on particular memory IC.

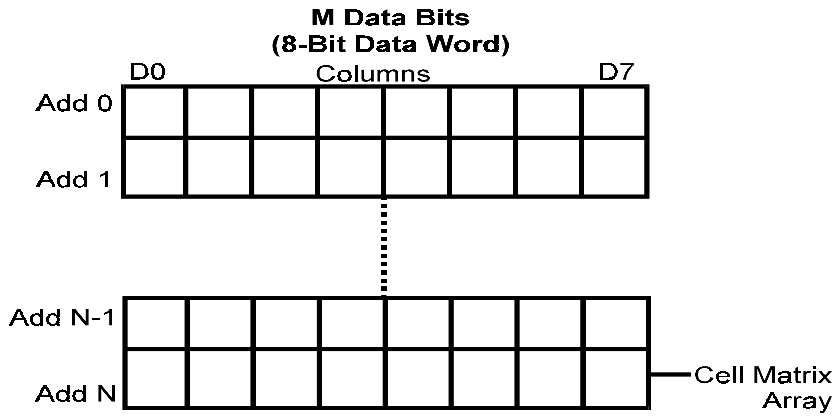


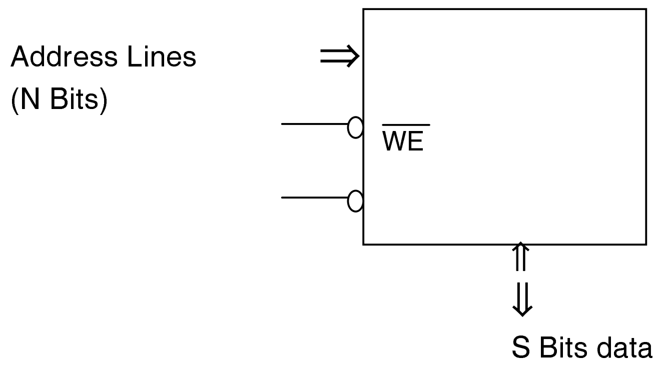
Fig. 4.1. Memory Array

4.3 Memory IC Organisation:

The capacity of a memory chip is generally measured in terms of the number of bits or bytes it contains. As most chips contain more than thousand bits, the memory capacity is expressed in kilobits (Kb) or mega-bits (Mb). Note that the b in these units denotes bits, not bytes.

Besides the capacity, one has to consider the word size of the chip. A word consists of one or more bits. The word size of a chip determines how many bits can be accessed (R/W) by the μP in a single access (R/W) to the chip. A commonly used notation, that indicates both the word size and the capacity of the chip, is N x s. Refer fig. 4.2, here, N is the total number of words (memory locations) and s is the total number of bits per word. The capacity of the memory is obviously N x s bits. Table 4.2 lists the ORGANISATION of some widely used memory chips. A memory chip having S bits per word is also referred to as an S bit wide memory.

The external organisation besides no. of address bits & no. of data bits. Fig. 4.2. shows the external organisation of a memory IC. For read/write $\overline{CS}$ should be low. When $\overline{WE}$ = low, write operation takes place. When $\overline{WE}$=high read, opeation takes place.



No of Address Locations = 2^N
Word Length (Data) = S bits
For Read/Write, $\overline{CS}$ should be low
WE = Low, Write operation takes place, When WE = High read operation takes place

Fig. 4.2. External Organisation of Memory IC

4.4. Memory Classification:

There are two important types of semiconductor memories RAM (Random Access Memory) and ROM (Read Only Memory). There are various types of RAM and ROM.

4.5a RAM:

The read & write memory of a computer is popularly known as RAM. It is also called R/W memory. Information can be read from and written into it during normal operation. It has random access property. It retains stored information as long as power supply is ON; its contents are lost when power supply is switched OFF. So it is volatile in nature. There are two kinds of RAM: Static RAM and Dynamic RAM. In a Static RAM (SRAM) the stored information are retained in it as long as the power supply is ON. But Dynamic RAM (DRAM) loses its contents in a very short time even though the power supply is ON. DRAM stores data in the gate to source capacitor of a transistor. Due to leakage, charge on stray capacitance leaks away after few milliseconds. So it has refreshed periodically usually every 2 Millisecond. DRAM consumes less power and has higher packaging density and cheaper.

4.5.b SRAM:

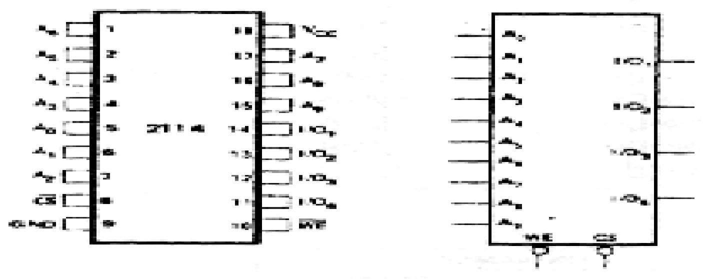
Consumes more power and expensive but does not require refreshing. It is faster than DRAM, CMOS static RAM's are recommended for medium size or memory. DRAM's are used where large size of memory is required.

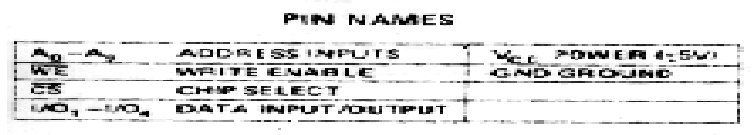
4.5.c DRAM:

Stands for integrated DRAM and its control and refresh circuitry into a separate IC known as DRAM controller. With advance in VLSI technology now it has become possible to integrate DRAM and its controller on a single chip.

STATIC RAM (2114)
1024 X 4

The Intel 2114 is a 4096 bit static Random Access Memory organised as 1024 words by 4 bits.

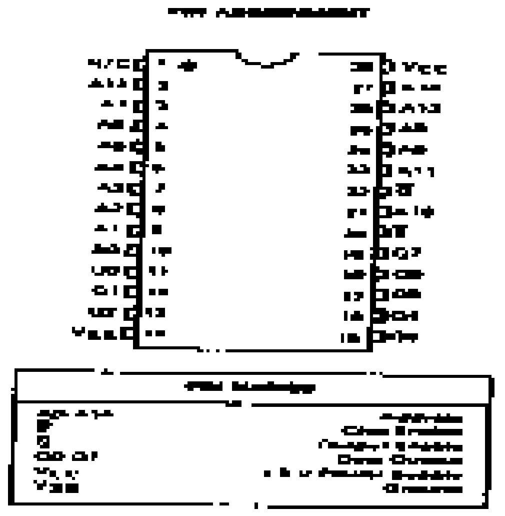




PIN NAMES

A_0–A_9	ADDRESS INPUTS	V_{CC}	POWER (+5V)
$\overline{WE}$	WRITE ENABLE	GND	GROUND
$\overline{CS}$	CHIP SELECT		
I/O_1–I/O_4	DATA INPUT/OUTPUT		

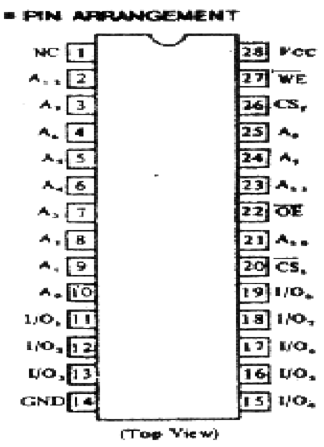
2048 Word x 8 Bit RAM (HM 6116P-2)

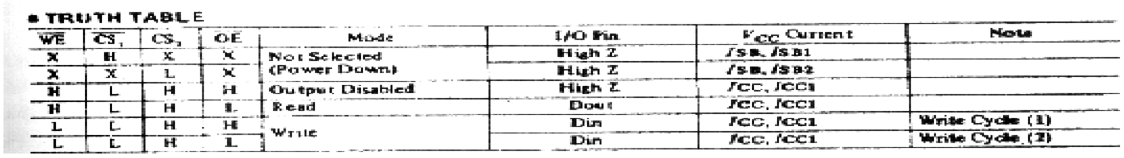


4.6 ROM:

ROM is read only memory and non-volatile. It is used for permanent storage. It has also a random access property. It contains assembler, compiler, monitor, debugging program or any other permanent program. ROM’s are widely used for function tables (sine, cosine, square root, logarithm, exponential etc.) code conversion tables, multiplication and division subroutines etc. The manufacturer decides the contents of ROM. These contents are permanently stored into a ROM. it is not accessible. ROM’s are simple, Cheap and dense.

ROM (MCM 63256) : This is 256K Bit Read Only Memory. It is organised as 32768 bytes of 8 bits. This device has low Power dissipation & wide operating margins. The active low level enables the chip and enables the output along with the memory contents. The chip enable input deselect the output & puts the chip in a power down mode.




• TRUTH TABLE

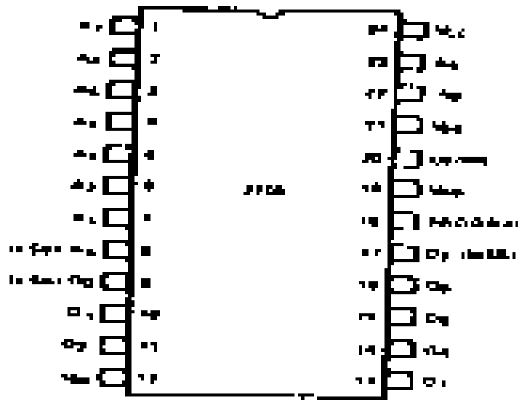
WE	CS1	CS2	OE	Mode	I/O Pin	Vcc Current	Note
X	H	X	X	Not Selected (Power Down)	High Z	ISB, ISB1	
X	X	L	X		High Z	ISB, ISB2	
H	L	H	H	Output Disabled	High Z	ICC, ICC1	
H	L	H	L	Read	Dout	ICC, ICC1	
L	L	H	H	Write	Din	ICC, ICC1	Write Cycle (1)
L	L	H	L		Din	ICC, ICC1	Write Cycle (2)

4.6a PROM : A PROM is programmable ROM. The user decides the contents of PROM. The user can write permanent program in PROM by PROM programmer.

4.6b EPROM : In EPROM is an erasable PROM. Exposing EPROM to high intensity short wave ultraviolet light for 10 to 20 minutes erases the contents. An UV source with wavelengths of 2537 0 A is used for this purpose. EPROM are cheap, reliable and widely available.

UV Erasable PROM (2708) : The Intel 2708 is an 8192 bit ultraviolet light erasable and electrically reprogrammable EPROM data inputs and outputs are TTL compatible both read and programs modes. The A0-A9 are address inputs. O0-O7 are data Output/Inputs.

CS/WC are chip select/write enable input.



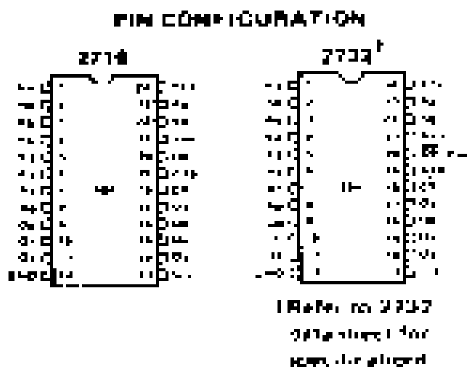
PIN NAMES

A_0 - A_9 = ADDRESS INPUTS

O_0 - O_7 = DATA OUTPUT/INPUTS

CS¯¯ /WE = CHIP SELECT/WRITE ENABLE I/P

UV Erasable PROM (2716) : The Intel 2716 is 16,384 bit ultraviolet erasable programmable Read Only Memory. The 2716 with its single 5 volt power supply & with an access time upto 350 ns.



PIN NAMES

A_0 - A_{10} = ADDRESS INPUTS

O_0 - O_7 = DATA OUTPUT/INPUTS

CE/PGM = CHIP ENABLE/PROGRAMME

$\overline{OE}$ = OUTPUT ENABLE

ERASURE CHARACTERISTICS : The erasure characteristics of the 2716 are such that erasure begins to occur when exposed to light with wavelengths shorter than approximately 4000 Angstroms (A). It should be noted that sunlight and certain types of fluorescent lamps have wavelengths in the 3000-4000A range. Data show that constant exposure to room level fluorescent lighting could erase the typical 2716 in approximately 3 years, while it would take approximately 1 weak to cause erasure when exposed to direct sunlight.

If the 2716 are to be exposed to these types of lighting conditions for extended periods of time, opaque labels should be placed over the 2716 window to prevent unintentional erasure.

The recommended erasure procedure for the 2716 is exposure to short-wave ultraviolet light which has a wavelength of 2537 Angstroms (A). The integrated does (i.e. UV intensity X exposure time) for erasure should be a minimum of 15 W-sec./cm2. The erasure time with this dosage is approximately 15 to 20 minutes using an ultraviolet lamp with a 12000 µW/cm2 power rating. The 2716 should be placed within 1 inch of the lamp tubes during erasure. Some lamps have a filter on their tubes, which should be removed before erasure.

DEVICE OPERATION : The five modes of operation of the 2716 are listed in Table 1. IT should be noted that all inputs for the five modes are at TTL Levels. The power supplies required are a+5V Vcc and a V pp. The V pp power supply must be at 25 V during the three programming modes, and must be at 5V in the other two modes.

Table 1 Mode Selection

Pins / Mode	CE/PGM (18)	OE (20)	V PP (21)	V CC (24)	Outputs (9-11.13-17)
Read	V IL	V IL	+5	+5	DOUT
Standby	V IH	Don't Care	+5	+5	High 2
Program	Pulsed VOL to VIH	V Ih	+25	+5	DIN
Program Verify	V IL	V Il	+25	+5	DOUT
Program Inhibit	V IL	V IH	+25	+2	High Z

READ MODE : The 2716 has two control functions, both of which must be logically satisfied in order to obtain data at the outputs. Chip Enable (CE) is the power control and should be used for device selection. Output Enable (OE) is the output control and should be used to gate data to the output pins, independent of device selection. Assuming that addresses are stable, address access time (t_{ACC}) is equal to the delay from CE to output (t_{CE}). Data is available at the outputs 120 ns (t_{OE}) after the falling edge of OE, assuming that CE has been low and addresses have been stable for at least $t_{ACC\ -t\ OE.}$

STANDBY MODE: The 2716 have a standby mode, which reduces the active power dissipation by 75%, from 525 mW to 132 mW. The 2716 is placed in the standby mode by applying a TTL high signal to the CE input. When in standby mode, the outputs are in a high impedance state, independent of the OE input.

PROGRAMMING : Initially, and after each erasure, all bits of the 2716 are in the "1" state. Data is introduced by selectively programming "0's" into the desired bit locations. Although only "0's" will be programmed, both "1's" and "0's" can be presented in the data word. The only way to change a "0" to a "1" is by ultraviolet light erasure.

The 2716 is in the programming mode when the Vpp power supply is at 25V and OE is at V_{IH}. The data be programmed is applied 8 bits in parallel to the data output pins. The levels required for the address and data inputs are TTL.

When the address and data are stable, a 50 msec, active high, TTL program pulse is applied to the CE/PGM input. A program pulse must be applied at each address location to be programmed. You can program any location at any time either individually, sequentially, or at random. The program pulse has a maximum width of 55 msec. The 2716 must not be programmed with a DC signal applied to the CE/PGM input.

Programming of multiple 2716s in parallel with the same data can be easily accomplished due to the simplicity of the programming requirements. Like inputs of the parallel 2716s may be connected together when they are programmed with the same data. A high level TTL pulse applied to the CE/PGM input programs the paralleled 2716s.

PROGRAM INHIBIT : Programming of multiple 2716s in parallel with different data is also easily accomplished. Except for CE/PGM, all like inputs (including OE) of the parallel 2716s may be common. A TTL level program pulse applied to a 2716a CE/PGM input with V_{PP} at 25V will program that 2716. A low level CE/PGM input inhibits the other 2716 from being programmed.

PROGRAM VERIFY : A verify should be performed on the programmed bits to determine that they were correctly programmed. The verify may be performed with V_{PP} at 25V. Except during programming and program verify, V_{PP} must be at 5V.

4.6c E^2 PROM : E^2PROMs (EEPROMs) are Electrically erasable PROM. They need not be removed from microcomputer board for erasing. The change in contents is made in milliseconds which is much less than the erasing time of an EPROM. A typical time is 10 ms for Motorola 2817. About 10 ms are required to write each byte of data into the device. If required single bit can also be erased. E^2PROM is much easier programmable as compared to EPROM.
Non-volatile flash memory: There are also electrically erasable and reprogrammable non-volatile.

4.7 Main memory & Secondary memory

The memory of computer is of two types:

i) Main Memory

ii) Secondary Memory

i) Main Memory : Main memory consists of semiconductor memories which has very fast access time. And in these types of memory RAM, ROM, EPROM are included. When you turn ON the CPU system, CPU will be guided by program stored in main memory. CPU will take decision accordingly. This memory is not user's memory.

ii) *Secondary memory :* Secondary memory has slow access time. This is user's memory. In these type of memories consists of floppy diskette, Magnetic tape, etc. CPU will read the index point of such memories and proceed further according to command. The driver is required for secondary memory.

TABLE 2 : Common ROM Chips

IC	**Type**	**ORGANISATION**
74187	ROM	256 X 4
8355	PROM	2k x 8
74S288	PROM	32 x 8
MK 36000	ROM	8k x 8
2708	EPROM	1k x 8
2716	EPROM	2k x 8
2732	EPROM	4k x 8

TABLE : Common ROM Chips

IC	Type	ORGANISATION
2764	EPROM	8k x 8
27128	EPROM	16k x 8
27256	EPROM	32k x 8
8755	EPROM	2k x 8
2816	EAROM	2k x 8
82S110	SRAM	1k x 1
6508	SRAM	1k x 1
8155	SRAM	256 x 8
2114	SRAM	1k x 4
2167	SRAM	16k x 1
4816	DRAM	2k x 8
4118	DRAM	1k x 8
4164	DRAM	64k x 1
6664	DRAM	66k x 1
1103	DRAM	1k x 1
2104	DRAM	4k x 1
4116	DRAM	16k x 1
2116	DRAM	16k x 1
4070	DRAM	16k x 1
6616	DRAM	16k x 1
2614	SRAM	4k x 1
2316	SRAM	4k x 1
4016	SRAM	2k x 8
4118	SRAM	1k x 8
5255	SRAM	1K x 4
2142	SRAM	1k x 4
2107 A	DRAM	4k x 1
41256	DRAM	256k x 1

Review Questions

1. What would be structure of binary address for a memory system having a capacity of 1024 bits ?

2. Determine how many address lines are required for memory that has following number of bits.
   a) 1024 b) 4098 c) 256 d) 16384

3. Explain why EPROM is or not a volatile memory.

4. Explain Difference between EPROM & a PROM.
5. Complete the following table.

Memory size	Type of device	No. of rows	No. of columns
4k x 8	2k x 4		
64k x 8	32k x 1		
8k x 16	4k x 4		
1M x 16	64k x 1		

6 How many Boolean function can be implemented by a single 2k x 8 ROM and what are restriction on input variables of these functions.

7. A system uses four 8k bytes ROM Design an address decoder by using a 74LS 138 & other gates necessary, assuming 16 bit address bus.

8. What is the key difference between static & dynamic RAM ? When is dynamic RAM preferable ? What are disadvantages ?

9. Are any hardware changes required to use EEPROM in design ? Do they affect software requirements ? What are the limitations ?

NOTES

NOTES

NOTES

NOTES

NOTES

Chapter Five

INTRODUCTION TO MICROPROCESSORS

Features :

5.1 INTRODUCTION

The history of attempts to make machine that could perform long sequences of calculation the machine designed by Babbage used cardboard. Cards with holes punched in them to introduce both instructions and data into machine. By the 1930's punched cards were in wide use in large business and various types of punched card handling machines were available, In 1937 Harward Aiken, at Havvard, proposed to IBM that a machine could be constructed which would automatically sequence the operations and calculations performed.

By 1965 third generation of computers was introduced (the IBM corporation) present day computers are less easily distinguished from earlier generations. There are some striking and important differences, however. Volume of scale integration increases. To incorporate hundred of thousands of active components in volume of fraction of an inch, leading to what is called large scale integration (LSI) and very large scale integration (VLSI)

Digital Computer : A digital computer is a programmable machine that processes or does calculations in binary data (0,1). Traditionally it is represented by four components. CPU (ALU plus control unit), memory, input and output. Analog Systems process data in continuous variations.

Digital Systems have the following advantages over Analog system

- Storage or transmission of data without deterioration
- Improved resolution
- Operation is completely reliable and trouble free.

5.2. MICROCOMPUTER ORGANISATION

A microcomputer system consists primarily of four components.

1. Microprocessor   2.Memory   3. Input   4. Output

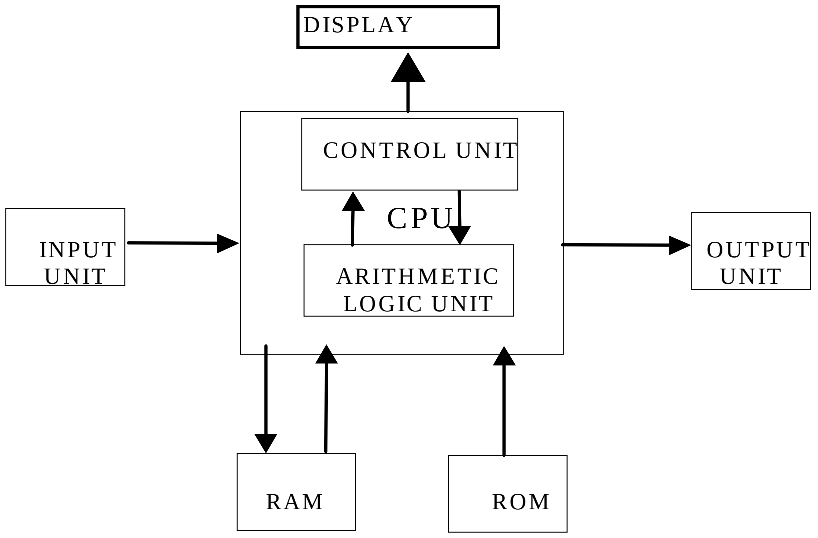


Fig. 5.1. Block Diagram of a Micro Computer

The microprocessor manipulates the data, controls timing of various operations and communicates with such peripherals as memory and I/O. The internal logic design of the microprocessor, called its *architecture*, determines how and what operations are performed by the µP.

Microprocessor is capable of performing computer functions and making decisions to change the sequence of program execution. In simple terms CPU on a chip is called Microprocessor. It consists of the following components:

1) Registers to store data.
2) ALU for arithmetic and logic operations.
3) Decoders to decode instructions.
4) Counters to have sequential data flow
5) Control units for timing the communication process with various devices such as memory, input & output.

Register stores one bit of information. Memory is an array of registers. These arrays are always grouped in powers of two. Input devices allow the entry of data to µP, most widely used is the keyboard. The result can be displayed in several ways such as seven segments LED's, LCD's or printed by printer. The I/O devices are known as *peripherals.*

Memory is of two types:

- *Read only memory* (ROM) &
- *Read write memory*(R/WM) popularly known as Random access memory (RAM).

ROM is used to store programs that need no alteration. The monitor program of a single board microcomputer is generally stored in ROM. This program interprets the information entered through a keyboard and provides its equivalent to the µP.

RW/M is also known as user memory. It is used to store user program & data, in single board microcomputers in which instructions and data are entered through a keyboard.

The ***system bus*** is a communication path between the microprocessor and peripherals; it is a group of wires carrying bits. This is time shared to communicate with various peripherals. It is important to know certain definitions for the proper understanding of the microprocessor.

- *Bit* is a binary digit, 0 or 1.

- *Byte* is defined as the group of 8 bits

- *Nibble* stands for a group of 4 bits.

- *Word or word length* is defined as the number of bits computer recognises and processes at a time.

- *Instruction* is a command in binary that is recognised and executed by the computer in order to accomplish a task. Some instructions are designed with one word, and some require multiple words.

- ***Machine Language*** is a binary medium of a communication with a computer, through a designed set of instructions specific to each computer.

- *Assembly Language* is a medium of communication with a computer in which programs are written in mnemonics (combination of letters). It is specific to a given computer.

- *Low Level Language* is a medium of communication that is machine dependent, which is specific to a given computer. For e.g. Machine and assembly language.

- *High Level Language* is a medium of communication that is independent of a given computer. Programs are written in English like words and they can be executed on a machine using a translator (a compiler or an interpreter)

- *Compiler* is a program that translates English like words of high level language into machine language of the computer. A compiler reads a given program, called a source code, in its entirety, and then translates the program into the machine language, which is called an object code.

- *Interpreter* is a program that translates the English like statements of high level language into machine language of the computer. It translates one statement at a time form a source code to an object code.

- *Assembler* is a computer program that translates an assembly language program from mnemonics to the binary machine code of a computer.

The ALU executes all arithmetic and the logic instructions. For example, the arithmetic addition and logical AND operations will be performed by the ALU. Instructions are made up of one word or several words. The set of instructions designed into the machine makes up its *machine language*, a binary language, composed of 0's & 1's that is specific to each computer. A *program* is a sequence of instructions that operate on data to produce desired results. A microprocessor needs to be programmed before it can perform a useful task. An example of a simple task is comparing two numbers and printing out the greater of two. The program, written to perform a particular task, is stored in the semiconductor memory accessible to the microprocessor. The processor, having been instructed to execute the program, fetches one instruction at a time from the memory and executes it.

5.3. Computer Language & Machine Language:

Instructions are made up of one word or several words. The set of instructions designed into the machine makes up its machine language, a binary language, composed of 0's and 1's that is specific to each computer.

It is difficult to handle a program written in binary; hence use of hexadecimal numbers is made. For example an instruction 0011 1100 which increments the number in the register called the accumulator by one can also be written as 3C in hexadecimal code.

It is still difficult to understand a program written in hexadecimal numbers. Hence a symbolic code is used to represent each instruction called assembly language. Machine language & assembly language are machine specific and are called low level language. The translation from assembly to machine language is done either by hand assembly or by a program called assembler.

Programming language that are intended to be machine independent are called high level languages. These include such languages as FORTRAN, BASIC, PASCAL, COBOL. Instructions written in these languages are known as statements. Program written in high-level language statements is called source code. Source code is converted into machine language compatible with the MP being used in the system. This translation in the machine language is called the object code. Each microprocessor needs its own primary difference between a compiler and an interpreter lies entire program list and then generates the object code, on the other hand the interpreter reads one instruction at a time, produces its object code and executes the instructions before reading the next instruction. M-basic is the common example of an interpreter for the basic language.

Compiler and interpreters require large memory space because each instruction in high level language requires several machine codes to translate it into binary. On the other hand there is one to one correspondence between the assembly language and the machine code. Thus, assembly language is compact and requires less memory space. Hence it is more efficient than the high level language.

The primary advantage of the high-level language program is in trouble shooting. It is much easier to find errors in a program written in a high level language. In certain applications such as traffic control and appliance control, where programs are small and compact, assembly language is suitable. Similarly, in real time applications as converting a high frequency waveform into digital data, program efficiency is critical as events and time should closely match each other without significant delay; therefore assembly language is highly desirable in such applications. On the other hand for applications in which programs are large and memory is not a limitation, high level language may be desirable. The advantage of time saved in a large program in high-level language may out weigh its disadvantages of memory requirements and inefficiency.

5.4. THE ARCHITECTURE OF 8 BIT GENERIC CPU:

A typical central processor unit (CPU) consists of the following interconnected functional units:

- Register
- Arithmetic/Logic Unit (ALU)
- Control Circuitry

Registers are temporary storage units within the CPU. Some registers, such as the program counter and instruction register, have dedicated uses. Other registers, such as the accumulator, are for more general-purpose use.

ACCUMULATOR : The accumulator usually stores one of the operands to be manipulated by the ALU. Typical instruction might direct the ALU to add the contents of some other register to the contents of the accumulator and store the result in the accumulator itself. In general, the accumulator is both a source (operand) and a destination (result) register.

Other a CPU will include a number of additional general-purpose registers that can be used to store operands or intermediate data. The availability of general-purpose registers eliminates the need to "shuffle" intermediate results back and forth between memory and the accumulator, thus improving processing speed and efficiency.

PROGRAM COUNTER (JUMPS, SUBROUTINES AND THE STACK) : The instructions that make up a program are stored in the system's memory. The central processor references the contents of memory, in order to determine what action is appropriate. This means that the processor must know which location contains the next instruction.

Each of the locations in memory is numbered, to distinguish it from all other locations in memory. The number, which identifies a memory location, is called its Address. The processor maintains a counter, which contains the address of the next program instruction. This register is called the **Program Computer**. The processor updates the program counter by adding "1" to the counter each time it fetches and instruction, so that the program counter is always current (pointing to the next instruction).

The programmer therefore stores his instruction in numerically adjacent addresses, so that the lower addresses contain the first instruction to be executed and the higher address contains later instructions. The only time the programmer may violate this sequential rule is when an instruction in one section of memory is a jump instruction to another section of memory.

A jump instruction contains the address of the instruction, which is to follow it. The next instruction may be stored in any memory location, as long as the programmed jumps specifies the correct address. During the execution of a jump instruction, the processor replaces the contents to its program counter with the address embodied in the jump. Thus, the logical continuity of the program in maintained.

A special kind of program jump occurs when the stored program **"Calls"** a subroutine. In this kind of jump, the processor is required "remember" the contents of program counter at the time that the jump occurs. This enables the processor to resume execution of the main program when it is finished with the last instruction of the subroutine.

A Subroutine is a program within a program. Usually it is a general-purpose set of instruction that must be executed repeatedly in the course of a main program. Routines, which calculate the square, the sine, or the logarithm of a program variable, are good examples of functions often written as subroutines. Other examples might be programs designed for inputting or outputting data to a particular peripheral device.

The processor has a special way of handling subroutines, in order to insure an orderly return to the main program. When the processor receives a call instruction, it increments the program counter and stores the counter's contents in a reserved memory area know as the **Stack.** The stack thus saves the address of the instruction to be executed after the subroutine is completed. Then the processor loads the address specified in the call into its program counter. The next instruction fetched will therefore be the first step of the subroutine.

The last instruction in any subroutine is a **Return**. Such an instruction need specify no address. When the processor fetched a Return instruction, it simply replaces the current contents of the Program Counter with the address on the top of the stack. This causes the processor to resume execution of the calling program at the point immediately following the original call instruction.

Subroutines are often **nested;** that is, one subroutine will sometimes call a second subroutine. The second may call a third, and so on. This is perfectly acceptable, as long as the processor has enough capacity to store the necessary return addresses, and the logical provision for doing so. In other words, the depth of the stack itself determines the maximum depth of nesting. If the stack has space for storing three return addresses, then three levels of subroutines may be accommodated.

Processors have different ways of maintaining stacks. Some have facilities for the storage of return addresses built into the processor itself. Other processors use reserved areas of external memory as the stack and simply maintain a **Pointer** register, which contains the address of the most recent stack entry. The external stack allows virtually unlimited subroutine nesting. In addition, if the processor provides instructions that causes the contents of the accumulator and other general purpose registers to be "pushed" onto the stack or "popped" off the stack via the address stored in the stack pointer, multi-level interrupt processing (described later in this chapter) is possible. The status of the processor (i.e., the contents of all the registers) can be saved in the stack when an interrupt is accepted and then restored after the interrupt has been serviced. This ability to save the processor's status at any given time is possible even if an interrupt service routine, itself, is interrupt.

INSTRUCTION REGISTER AND DECODER : Every computer has Word Length that is characteristics of that machine. A computer's word length is usually determined by the size of its internal storage elements and interconnecting paths (referred to Busses); for example, a computer whose register and busses can store and transfer 8 bit and is referred to as an 8-bit parallel processor. An eight-bit parallel processor generally finds, and the memory associated with such a processor is therefore organized to store eight bits in each addressable memory as eight-bit binary numbers, or as numbers that are integral multiples of eight bits: 16 bits, 24 bits, and so on. This characteristic eight-bit field is often referred to as Byte.

Each operation that the processor can perform is identified by a unique byte of data known as an Instruction Code or Operation Code. An eight-bit word used as an instruction code can distinguish between 256 alternative actions, more than adequate for most processors.

The processor fetches an instruction in two distinct operations. First, the processor transmits the address in it Program Counter to the memory. Then the memory returns the addressed byte in a register known as the **Instruction Register,** and uses it to direct activities during the remainder of the instruction execution.

The mechanism by which the processor translates an instruction code into specific processing actions requires more elaboration. The eight bits stored in the instruction register can be decoded and used to selectively activate one or a number of output lines, in this case up to 256 lines. Each line represents a set of activities associated with execution of a particular instruction code. The enable line can be combined with selected timing pulses, to develop electrical signals that can then be used to initiate specific actions. This translation of code into action is performed by the Instruction Decode and by the associated control circuitry.

An eight-bit instruction code is often sufficient to specify a particular processing action. There are times; however, when execution of the instruction requires more information than eight bits can convey.
One example of this is when the instruction references a memory location. The basic instruction code identifies the operation to be performed, but cannot specify the object address

as well. In a case like this a two-or three-byte instruction must be used. Successive instruction bytes are stored in sequentially adjacent memory locations, and the processor performs two or three fetches in succession to obtain the full instruction. The first byte retrieved from memory is placed in the processor then proceeds with the execution phase. Such an instruction is referred to as **Variable Length**.

Address Register (s): A CPU may use a register-pair to hold the address of a memory location that is to be accessed for data. If the address register is **programmable**, (i.e. if there are instructions that allow the programmer to alter the contents of the register) the program can "build" an address in the register prior to executing a **Memory Reference** instruction. Reference Instruction is an instruction that reads data from memory writes data to memory or operates on data stored in memory).

Arithmetic / Logic Unit (ALU): All processors contain an arithmetic/logic unit, which is often referred to simply as the ALU. The ALU, as its name implies, is that portion of the CPU hardware, which performs the arithmetic and logical operations on the binary data.

The ALU must contain an **Adder**, which is capable of combining the contents of two registers in accordance with the logic of binary arithmetic. This provision permits the processor to perform arithmetic manipulations of the data it obtains from memory and from its other inputs.

Using only the basic adder a capable programmer can write routines, which will subtract, multiply and divide, giving the machine complete arithmetic capabilities. In practice, however, most ALUs provide other built-in functions, including hardware subtraction, Boolean logic operations, and shift capabilities.

The ALU contains **Flag Bits**, which specify certain conditions that arise in the course of arithmetic and logical manipulations. Flags typically include **Carry, Zero, Sign,** and **Parity.** It is possible to program jumps which are conditionally dependent on the status of one or more flags. Thus, for example, the program may be designed to jump to a special routine if the carry bit is set following an addition instruction.

Control Circuitry: The control circuitry is the primary functional unit within A CPU. Using clock inputs, the control circuitry maintains the proper sequence of events required for any processing task. After an instruction is fetched and decoded, the control circuitry issues the appropriate signals (to units both internal and external to the CPU) for initiating the proper processing action. Often the control circuitry will be capable of responding to external signals, such as an interrupt or wait request. An **Interrupt** request will cause the control circuitry to temporarily interrupt main program execution, jumps to a special routine to service the interrupting device, then automatically return to the main program. A wait request is often issued by a memory or I/O element that operates slower than the CPU. The control circuitry will make the CPU idle until the memory or I/O ports is ready with the data.

TIMING: The activities of the central processor are cyclical. The processor fetches and instruction, performs the operations required, fetches the next instruction, and so on. This orderly sequence of events requires precise timing, and the CPU therefore requires a free running oscillator clock, which furnishes the reference for all processor actions. The combines fetch and execution of a single instruction is referred to as an Instruction Cycle. The portion of a cycle identified with a clearly defined activity is called a State. And the interval between pulses of the timing oscillator is referred to as a Clock Period. As a general rule, one or more clock periods are necessary for the completion of a state, and there are several states in a cycle.

Instruction Fetch : The first state (s) of any instruction cycle will be dedicated to fetching the next instruction. The CPU issues a read signal and the contents of the program counter are sent to memory, which responds by returning the next instruction word. The first byte of the instruction is placed in the instruction register. If the instruction consists of more than one byte, additional states are required to fetch each byte of the instruction. When the entire instruction is present in the CPU, the program counter is incremented (in preparation for the next instruction) and the instruction is decoded. The operation specified in the instruction will be executed in the remaining states of the instruction cycle. The instruction may call for a memory read or write or an internal CPU operations, such as a register-to-register operation.

Memory Read: An instruction **fetch** is merely a special memory read operation that brings the instruction. It may then call for data to be read from memory into the CPU. The CPU again issues a read signal and sends the proper memory address, memory responds by returning the requested word. The data received is placed in the accumulator or one of the other general purposes registers (not the instruction register).

Memory Write: A memory write operation is similar to a read expect for the direction of data flow. The CPU issues a write signal, sends the proper memory address, then sends the data word to be written into the addressed memory location.

Wait (memory synchronisation) : The activities of the processor are timed by a master clock oscillator. The clock period determines the timing of all processing activity.

The speed of the processing cycle, however, is limited by the memory's Access Time. Once the processor has sent a read address to memory, it cannot proceed until the memory has had time to respond. Most memories are capable of responding much faster than the processing cycle requires. A few, however, cannot supply the addressed byte within the minimum time established by the processor's clock.

Therefore a processor should contain a synchronisation provision, which permits the memory to request a **Wait state**. When the memory receives a read or write enable signal, it places a request signal on the processor's READY line, causing the CPU to idle temporarily. After the memory has had time to respond, it frees the processor's READY line, and the instruction cycle proceeds.

Input / Output : Input and output operations are similar to memory read and write operations with the exception that a peripheral I/O devices are addressed instead of a memory location. The CPU issues the appropriate input or output control signal, sends the proper devices address and either receives the data being input or sends the data to be output.

Data can be input/output in either parallel or serial form. All data within a digital computer is represented in binary coded form. A binary data word consists of a group of bits; each bit is either a one or a zero. **Parallel** I/O consists of transferring all bits in the word at the same time, one bit per line. **Serial I/O** consists of transferring one bit at a time on a single line. Naturally serial I/O is much slower, but it requires considerably less hardware than does parallel I/O.

5.5. Interrupts :

Interrupt provisions are included on many central processors, as a means of improving the processor's efficiency. Consider the case of a computer that is processing a large volume of data, portions of which are to be output to a printer. The CPU can output a byte of data within a single machine cycle but it may take the printer the equivalent of many machine cycles to actually print the character specified by the data byte. The CPU could then remain idle waiting until the printer can accept the next data byte. If an interrupt capability is implemented on the computer, the CPU can output a data byte then return to data processing. When the printer is ready to accept the next data byte, it can request an interrupt. When the CPU acknowledges the interrupt, it suspends main program execution and automatically branches to a routine that will output the next data byte. After the byte is output, the CPU continues with main program execution. Note that this is, in principle, quite similar to subordinate call, except that jump is initiated externally rather than by the program.

More complex interrupt structures are possible, in which several interruption devices share the same processor but have different priority levels. Interrupt processing is an important feature that enables maximum utilisation of a processor's capacity for high system throughput.

Hold : Another important feature that improves the throughput of a processor is the Hold. The hold provision enables Direct Memory Access (DMA) operations.

In ordinary input and output operations, the processor itself supervises the entire data transfer. Information to be placed in memory is transferred from the input device to the processor and then from the processor to the designated memory location. In similar fashion, information that goes from memory to output devices goes by way of the processor.

Some peripheral devices, however, are capable of transferring information to and from memory much faster than the processor itself can accomplish the transfer. If any appreciable quantity of data must be transferred to or from such a device, then having the device accomplish the transfer directly will increase system throughput. The processor must temporarily suspend its operation during such a transfer, to prevent conflicts that would arise if processor and peripheral device attempted to access memory simultaneously. It is for this reason that **hold** provision is included on some processors.

5.6. Memory Addressing

The 8085A is an 8-bit general-purpose microprocessor that is very cost-effective in small systems because of its extraordinarily low hardware overhead requirements. At the same time it is capable of accessing up to 64K bytes of memory and has status lines for controlling large systems.

In the 8085A microprocessor contains the functions of clock generation, in addition to execution of instruction set. The 8085A transfers data on an 8-bit, bi-directional 3-state bus (AD0-AD7) which is time multiplexed so that it can transmit the eight lower-order address bits and data bits on the same pins. An additional eight lines (A8-15) expand the system memory addressing capability to 16 bits, thereby allowing 64K bytes of memory to be accessed directly by the CPU. The 8085A CPU (central processing unit) generates control signals that can be used to select appropriate external devices and functions perform READ and WRITE operations and also to select memory or I/O ports. The 8085A can address up to 256 different I/O locations. These addresses have the same numerical values (00 through FFH) as the first 256 memory addresses; they are distinguished by means of the IO/M output from the CPU.

5.7. DMA (Direct Memory Access) :

Whenever a block of data needs to be transferred to memory from any device, then the DMA IC is normally used. DMA takes the control of transferring the data by putting CPU in a wait state. And this is done as follows :

- DMA issues a hold signal.
- CPU acknowledges the hold signal and goes into a wait state.
- DMA transfers the data by generating address of the memory.

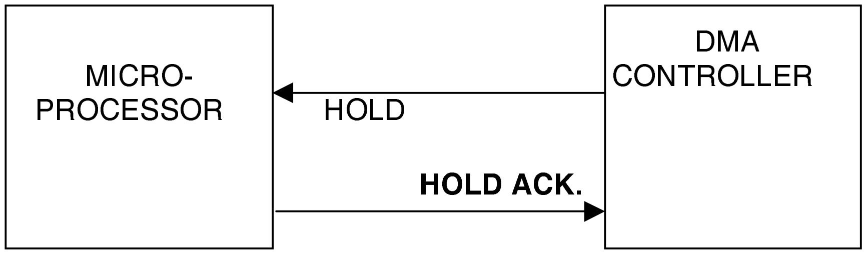


Fig. 5.2. DMA Driven Data Transfer

5.8. 8085 Architecture :

Fig. 5.3 shows internal Architecture of 8085 beyond the programmable registers. It includes ALU. Timing and control unit, Instruction Register, Register array, Interrupt control and serial I/O control.

The arithmetic logic unit performs the computing functions. It includes accumulator, temporary register. The arithmetic and logic circuits and five flags. The temporary register is used to hold data during an arithmetic /logic operation. The result is stored in accumulator and the flags (&flip & flop) are set or reset according to result of operation. Timing and control unit synchronises all microprocessor operations with the clock and generates the control signals for communication between microprocessor and peripherals. The control signals are similar to sync pulses in an oscilloscope. The RD and WR signals are sync pulses indicating the availability of data on the data bus. The instruction register and decoder are part of ALU, when instruction is fetched, it is loaded in instruction register. The decoder decodes the instruction and establishes sequence of events to follow. Two additional registers, called temporary registers W and Z are included in the register array. These registers are used to hold 8 bit data.

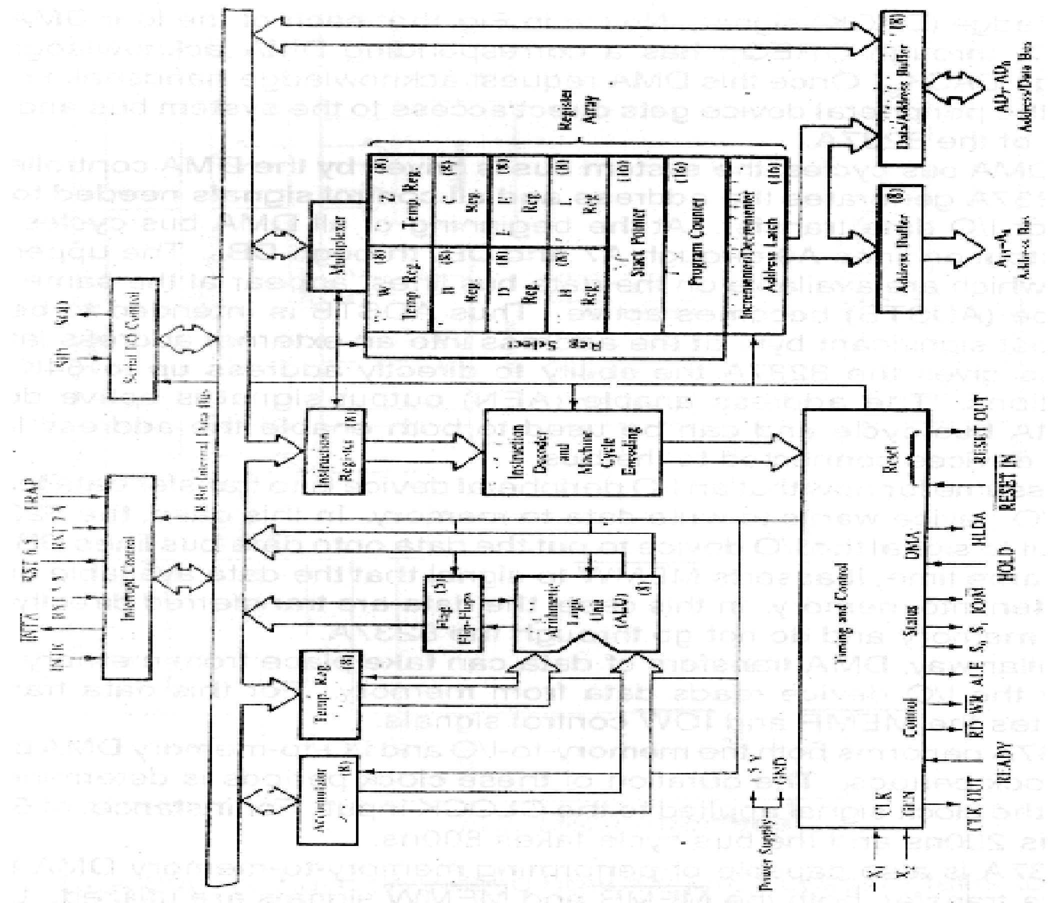


Fig. 5.3. 8085 Microprocessor Architecture

Registers : The 8085A, is provided with internal 8-bit registers and 16-bit registers. The 8085A has eight addressable 8-bit registers. Six of them can be used either as 8-bit registers or as 16-bit register pairs. Register pairs are treated as though they were single, 16-bit registers; the high-order byte of a pair is located in the first register and the low-order byte is located in the second. In addition to the registers, the 8085A contains two more 16-bit registers. The 8085A's CPU registers are distinguished as follows :

- The Accumulator (ACC or A Register) is the focus of all of the accumulator instructions, which include arithmetic, logic, load and store, and I/O instructions. It is an 8-bit register only.

- The program counter (PC) always points to the memory location of the next instruction to be executed. It always contains a 16-bit address.

- General-purpose register BC, DE, and HL may be used as six 8-bit registers or as three 16-bit registers, interchangeably, depending on the instruction being performed. HL functions as a data pointer to reference memory addresses that are either the sources of the destinations in a number of instructions. A smaller number of instructions can use BC or DE for indirect addressing.
- The stack pointer (SP) is a special data pointer that always points to the stack top (next available stack address). It is an indivisible 16-bit register.

- The flag register contains five one-bit flags, each of which records processor status information and may also control processor operation.

Flags : The five flags in the 8085A CPU are shown below :

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z		AC		P		CY

The carry flag (CY) is set and reset by arithmetic operations. Its status is directly tested by a program. For example, the addition of two one-byte numbers can produce an answer that does not fit into one byte.

```
HEXADECIMAL          BINARY

AEH              1 0 1 0 1 1 1 0
+74H             0 1 1 1 0 1 0 0
                 _______________
122H             1 0 0 1 0 0 0 1 0
                 _______________
```

Carry bit sets carry flag to 1

An additional operation that results in an overflow out of the high-order bit of the accumulator sets the carry flag. An addition operation that does not result in an overflow clears the carry flag. The carry flag also acts as a "borrow" flag for subtract operations.

The auxiliary carry flag (AC) indicates overflow out of bit 2 of the accumulator in the same way that the carry flag indicates overflow out of bit 7. This flag is commonly used in BCD (binary coded decimal) arithmetic.

The sign flag is set to the condition of the most significant bit of the accumulator following the execution of arithmetic or logical instructions. These instructions use bit 7 of data to represent the sign of the number contained in the accumulator. This permits the manipulation of numbers in the range from – 128 to + 127.

The zero flag is set if the result generated by certain instructions is zero. The zero flag is cleared if the result is not zero. A result that has a carry but has a zero answer byte in the accumulator will set both the carry flag and the zero flag. For example,

```
HEXADECIMAL          BINARY

A7H              1 0 1 0 0 1 1 1
+59H             0 1 0 1 1 0 0 1
                 _______________
100H             1 0 0 0 0 0 0 0 0
                 _______________
```

Carry bit & zero flag to 1

The Parity flag (P) is set to 1 if the parity (number of 1 bits) of the accumulator is even. If odd, it is cleared.

Stack : The stack pointer maintains the address of the last byte entered into the stack. The stack pointer can be initialized t use any portion of read-write memory as a stack. The stack pointer decrements each time data is pushed onto the stack and is incremented each time data is popped off the stack (i.e., the stack grows downward in terms of memory address, and the stack "top" is the lowest numerical address represented in the stack currently in use). Note that the stack pointer is always incremented or decremented by two bytes since all stack operations apply to register pairs.

Arithmetic-Logic Unit (ALU) : The ALU contains the accumulator and the flag register and some temporary registers that are inaccessible to the programmer.

Arithmetic, logic, and rotate operations are performed by the ALU. The results of these operations can be deposited in the accumulator, or they can be transferred to the internal data bus for use elsewhere.

Instruction Register and Decoder : During an instruction fetch, the first byte of an instruction (containing the opcode) is transferred from the internal bus to the 8-bit instruction register. The contents of the instruction register are, in turn, available to the instruction decoder. The output of the decoder, gates by timing signals controls of registers, ALU, and data and address buffers. The output of the instruction decoder and internal clock generator generates the state and machine cycle timing signals.

Internal Clock Generator: The 8085A CPU incorporates a complete clock generator on its chip, so it requires only the addition of a quartz crystal to establish timing for its operation. (It will accept an external clock input at its XI input. A suitable crystal for the standard 8085A must be parallel resonant at a fundamental of 6.25 MHz or less, twice the desired internal clock frequency. A schmitt trigger is used interchangeably as oscillator or as input conditioner, depending upon whether a crystal or an external source is used. The clock circuitry generates two non-overlapping internal clock signals, t1 and t2 and control the internal timing of the 8085A and are not directly available on the outside of the chip.

Interrupts: The five hardware interrupt inputs provided in the 8085A are maskable (can be enabled or disabled by software instruction), and causes the CPU to fetch in an RST instruction. This vectors branch to any one eight fixed memory locations (Restart addresses). INTR can also be controlled by the 8259 programmable interrupt controller, which generates CALL instructions instead of RSTs, and can thus vector operation of the CPU to a preprogrammed subroutine located anywhere in the system's memory map. The RST 5.5, RST 6.5 and RST 7.5 hardware interrupts are different in function in that they are maskable through the use of the SIM instruction, which enables or disables these interrupts by clearing or setting corresponding mask flags based on data in the accumulator. We may read the status of the interrupt mask previously set by performing a RIM instruction. 'Its execution loads into the accumulator the following information.

- Current interrupt mask status for the RST 5.5, 6.5 and 7.5 hardware status
- Current interrupt enable flag status (except that immediately following TRAP, the IE flags status preceding that interrupt is loaded).
- RST 5.5, 6.5 and 7.5 interrupt pending

RST 5.5, 6.5 and 7.5 are also subject to being enabled or disabled. INTR RST 5.5 and RST 6.5 are level – sensitive, meaning that these inputs may be acknowledged by the processing when they are held at a high level. RST 7.5 is edge sensitive; meaning that an internal flip-flop in the 8085A registers the occurrence of an interrupt the instant a rising edge appears on the RST 7.5 input line. This input need not be held high; the flip-flop will remain set until it is cleared by one of three possible actions:

- The 8085A responds to the interrupt, and sends an internal reset signal to the RST 7.5 flip-flop.

- The 8085A, before responding to the RST 7.5 interrupt, receives a RESET IN signal from an external source; this also activates the internal reset.

- The 8085A executes a SIM instruction, with accumulator bit 4 previously set to 1.

The third type of hardware interrupt is TRAP. This input is not subject to any mask or interrupt enables/disable instruction. The receipt of a positive-going edge on the TRAP input triggers the processor's hardware interrupt sequence, but the pulse must be held high until acknowledged internally.

The sampling of all interrupts occurs on the descending edge of CLK, one cycle before the end of the instruction in which the interrupt input is activated. To be recognised, a valid interrupt must occur at least 160 ns before sampling time in the 8085A. This means that to guarantee being recognised, RST 5.5 and 6.5 and TRAP need to be held on for at least 17 clock states plus 160 ns, assuming that the interrupt might arrive just barely too late to be acknowledged during a particular instruction, and that the following instruction might be an 18-state CALL. This timing assumes that no WAIT or HOLD cycles are used.
Interrupt functions and their priorities are shown in the table that follows.

Name	**Priority**	**Address(1) branched to when interrupt occurs**	**Type trigger**
TRAP	1	24H	Rising and high level until sampled
RST 7.5	2	3CH	Rising edge(latched
RST 6.5	3	34H	High level until sampled
RST 5.5	4	1CH	High level until sampled
INTR	5	(2)	High level until sampled

NOTES :

(1) In the case TRAP and RST 5.5-7.5, the contents of the program counter are pushed onto the stack before the branch occurs.

(2) Depends on the instruction that is provided to the 8085A by the 8259 or other circuitry when the interrupt is acknowledged.

Serial Input and Output: The SID and SOD pins help to minimize chip count in small systems by providing for easy interface to a serial port using software for timing and for coding and decoding of the data. Each time a RIM instruction is executed, the status of the SID pin is read into bit 7 of the accumulator. In similar fashion, SIM is used to latch bit 7 of the accumulator out of the SOD output via an internal flip-flop, providing that bit 6 of the accumulator is set to 1.

SID can also be used as a general-purpose TEST input and SOD can serve as a one-bit control output.

8085 INSTRUCTIONS : The following abbreviations are used in the description of instruction set :

Reg	=	8085 A/8085 Register
Mem	=	Memory location
R	=	Register
Rs	=	Register Source
Rd	=	Register destination
M	=	Memory
Flags	=	S = sign; Z = Zero; AC = Auxiliary Carry; P = Parity; Cy = Carry

8085 MICROPROCESSOR INSTRUCTIONS SET:

ACI = Add Immediate to Accumulator with Carry
The 8 bit data (operand) and the Carry flag are added to the contents of the accumulator, and result stored in accumulator: e.g. ACI 57H

ADC = Add Register to Accumulator with Carry
The contents of the operand are added to the contents of the accumulator and the result stored in the accumulator. E.g. Add M.

ADI = Add Immediate to Accumulator
The 8 bit data (operand) are added to the contents of the accumulator, and the result is placed in the accumulator. E.g. ADI 59H

ANA = Logical AND with Accumulator
The contents of the accumulator are logically added with the contents of the operand; and the result placed in the accumulator. If the operand is a memory location, its address is specified by the contents of HL registers. E.g. ANA D.

ANI = AND immediate with Accumulator
The contents of the accumulator are logically AND with the 8 bit data and result is placed in the accumulator. E.g. ANI 97H

CALL = Unconditional Subroutine Call
The program sequence is transferred to the address specified by the operand. Before the transfer, the address of the next instruction to CALL (the contents of the program counter) is pushed on the stack.

CMA = Complement Accumulator
The contents of the accumulator are complemented E.g. CMA

CMC = Complement Carry
The Carry flag is complemented

CMP = Compare with Accumulator
The contents of the operand (register or memory) are compared with the contents of the accumulator. Both contents are preserved and the comparison is shown by setting the flag as follows :

If (A) < (Reg/Mem) : Carry flag is set and zero flag is reset.

If (A) = (Reg/Mem) : Zero flag is set and carry flag is reset.

If (A) > (Reg/Mem) : Carry and Zero flags are reset
Flags S, P, AC are also modified in addition to Z and CY to reflect the results of operation.

Reg : CMP B

CPI : Compare Immediate with Accumulator
The second byte (8-bit data) is compared with the contents of the accumulator. E.g. CPI 98H

DAD : Add Register Pair to Hand L Registers
The 16 bit contents of the specified register pair are added to the contents of the HL registers and the sum is saved in the HL register. E.g.. DAD H.

DCR : Decrement Source by 1
The contents of the designated register/memory is documented by 1 and the results are stored in the same place. If the operand is a memory location, it is specified by the contents of the HL register pair. E.g.. DCR B

IN : Input Data to Accumulator from a port with 8 bit address. The contents of the input port designated in the operand are read and loaded into the accumulator.

INR : Increment Contents of Register/Memory by 1.
The contents of the designated register/memory are incremented by 1 and the results are stored in the same place. If the operand is a memory location, it is specified by the contents of the HL register pair.

INX : Increment Register Pair by 1
The contents of the specified register pair are incremented by 1. The instruction views the contents of the two registers as a 16-bit number. E.g.. INX H

JMP : Jump Clear Conditionally
The program sequence is transferred to the memory location specified by the 16-bit address. This is a 3-byte instruction, the 2nd byte specified the low order byte and the third byte specifies the high order byte.

LDA : Load Accumulator Direct
The contents of memory location specified by a 16-bit address in the operand, are copied to the Accumulator. The contents of the source are not aligned. This is a 3-byte instruction, the second byte specifies the low order address and the third byte the high order address. E.g.. LDA 2050H

LDAX : Load Accumulator Indirect
The contents of the designated register pair point to memory location. This instruction copies the contents of that memory location into the accumulator. The contents of either the register pair or the memory location are not altered. E.g.. LDAX B

LHLD : Load Hand L Register Direct
The instruction copies the contents of the memory location pointed out by the 16-bit address in register 2 and copies the contents of the next memory location in register H. The contents of source memory locations are not altered. E.g.. LHLD 2050

LX1 : Load Register Pair Immediate
The instruction loads 16 bit data in the register pair designated in the operand. This is a 3-byte instruction, the second byte specifies the low order byte and the third byte specifies the high order byte. E.g.. LX1 B, 2050H.

MOV : Move-Copy from source to Destination.
The instruction copies the contents of the source register into the destination register, the contents of the source register are not altered. One of the operands is a memory location it is specified by the contents of HL Registers.

ORI : Logically OR Immediate
The contents of the accumulator are logically ORed with the 8 bit data in the operand and results are placed in the accumulator.

OUT : Output Data from Accumulator to a Port with 8 bit Address
The contents of the accumulator are copied into the output port specified by the operand.

PCHL : Load Program Counter with HL Contents
The contents of registers H and L are copied into the program counter. The contents of H are placed as a high order byte and of L as a low order byte.

POP : Pop Off Stack to Register Pair
The contents of the memory location pointed out by the stack pointer register are copied to the low order register (such as C, E, L and flags) of the operand. The stack pointer is incremented by 1 and the contents of that memory location are copied to the high order register (B, D, H, A) of the operand. The stack pointer register is again incremented by 1.

PUSH : Push register pair onto stack
The contents of the register pair designated in the operand are copied into the stack in the following sequence. The stack pointer is decremented and the contents of the high order register (B, D, H, A) are copied into that location. The stack pointer register is decremented again and the contents of the low order register (C, E, L, flags) are copied to that location.

RAL : Rotate Accumulator Left through Carry

Each binary bit of the accumulator is rotated left by one position through the Carry flag. Bit D7 is placed in the bit in the Carry flag and the Carry flag is placed in the least significant position Do. E.g.. RAL

RAR : Rotate accumulator Right through Carry
Each binary bit of the accumulator is rotated right by one position through the Carry flag and the bit in the Carry flag is placed in the next significant position D7. E.g.. RAL

RLC : Rotate Accumulator Left
Each bit of the accumulator is rotated left by one position. Bit D7 is placed in the position of D6 as well as in the Carry flag.

RRC : Rotate Accumulator Right
Each binary bit of the accumulator is rotated by one position. Bit D0 is placed in the position of D7 as well as in the Carry flag.

RET : Return from Subroutine the conditionally
The program sequence is transferred from the subroutine to the calling program. The 2 bytes from the top of the stack are copied into the program counter and the program execution begins at the new address. The instruction is equivalent to POP program counter.

SBI : Subtract Immediate with Borrow
The 8 bit date (operand) and the borrow are subtracted from the contents of the accumulator, and the results are placed in the accumulator. E.g.. SB I 25H

SHLD : Store Hand L Registers Direct
The contents of register L are stored in the memory location specified by the 16 bit address in the operand, and the contents of H register are stored in the next memory location by incrementing the operand. The contents of registers HL are not altered. This is a 3-byte instructions, the second byte specifies the low order address and the third byte specifies the high order address.

SPHL : Copy H and L Registers to the stack Pointer
The instruction loads the contents of the H and L registers in the stack pointer register, the contents of the H register provide the high order address, and the contents of the L register provide the low order address. The contents of H and L registers are not altered.

STA : Store Accumulator Direct
The contents of the accumulator are copied to a memory location specified by the operand. This is a 3-byte instruction; the second byte specifies the low order address and the third byte specifies the high order address. E.g.. STA 2050H

STAX : Store Accumulator Induct
The contents of the accumulator are copied into the memory location specified by the contents of the operand (register pair). The contents of the accumulator are not altered

STC : Set Carry
The Carry flag is set to 1

SUB : Subtract Register or Memory from Accumulator
The contents of the register or the memory location specified by the operand are subtracted from the contents of the accumulator, and the results are places in the accumulator. The contents of the source are not altered.

SUI : Subtract Immediate from Accumulator
The 8 bit data (operand) are subtracted from the contents of the accumulator, and the results are placed in the accumulator.

XCHG : Exchange H and L with D and E
The contents of register H are exchanged with the contents of register D, and the contents of register L are exchanged with the contents of register E.

XRA : Exclusive OR with Accumulator
The contents of the operand (register or memory) are exclusive ORed with contents of the accumulator, and the results are placed in the accumulator. The contents of the operand are not altered.

XRI : Exclusive OR Immediate with Accumulator
The 8 bit data (operand) are exclusive ORed with the contents of the accumulator, and the results are placed in the accumulator.

XTHL : Exchange H and L with Top of Stack
The contents of the L register are exchanged with the stack location out by the contents of the stack pointer register. The contents of the H register are exchanged with the next stack location (SP+1), however, the contents of the stack power are not altered.

NOTE : Candidates are not expected to remember the instructions set. This is only for reference example.

STEPS TO BE FOLLOWED TO DEVELOP A PROGRAM : For the solution of each of the subsequent problems (but in more particular for the more complex ones) you should always go through the following steps :-

1. Understand the problem.

2. Find the algorithm (i.e. find a procedure to solve the problem in some cases this is trivial).

3. Draw neat flowchart. For a complex program it will be necessary to draw several flow-charts, starting at the higher level (i.e. with basic blocks) and refining it further till the level is reached where it can easily be converted into a program.

4. Define the memory space for your program the following RAM locations are available for the user 2000 – 20FF (256 Bytes). The top locations 20CB – 20FF are reserved for the monitor. 2800 – 20FF (256 Bytes). Refer to manual for specific address.

- Define the stack pointer (e.g. 20cB).
- Define the CPU registers used for upgrading your temporary data, if the CPU registers are not sufficient. You should clearly distinguish between program memory and Data-Memory, although on the kit both herein to be physically same, namely RAM.
- The program memory is used for storing the program (i.e. the instructions, look-up tables etc.) for which usually a non-volatile memory such as ROM, is utilised. The data memory is used for storing variable or temporary data, for which purpose the read-and-write memory (RAM) is necessary.
- Define symbols and labels, which you will use in your program.

5. Device a test procedure. How will you test whether your program is doing what you want to stop its execution in order to test the CPU registers (i.e. set-break points)? Etc.

6. Write the program (with pencil) on a coding sheet (=Editing). Whenever necessary, a comment should be given to make the program more comprehensive. But the comment should never be trivial give a reason for writing that particular program step

7. Convert the instructions code into the machine code and pnmonics.

8. Load the program. After having loaded the program it inadvisable to check whether all codes and data have been entered properly.

9. Execute and debug the program. The time involved in debugging is inversely proportional to the time you have spent in going through points 1 to 8. Do not forget that your program should always start with defining the Stack-pointer, even if your program does not use the stack. You need it in any case for testing the program in single step.

10. Your program should always be terminated by a HALT – instruction or unless you have program of infinite loop. This will prevent the microprocessor to execute some random data beyond your program, which might alter your instructions.

Example-1 : The accumulator contains data byte 82H and the instructions MOV C, A (4FH) is fetched. The steps in decoding and execution of the instructions will be :

1. The contents of the data bus are placed in the instruction register & decoder.
2. The contents of the accumulator are transferred to the temporary register in the ALU
3. The contents of the temporary register are transferred to register C.

Example-2 : Write a program to add two decimal numbers 9.50 and 500 the final answer should be stored in H&L register

Solution: First correct the decimal number to the hexadecimal numbers

Decimal (500) = (01F4) Hex   Decimal (950) = (03B6) Hex

```
MV1   A, 00   (Initialise the accumulator)
MV1   B, 01
MV1   C, F4
MV1   D, 03
MV1   E, B6
ADD   C
ADD   E
```

```
MOV   L, A
MV1   A,00
ADC   B
ADD   D
MOV   H,A
HALT
```

5.9. Advance Processors :

The iAPX 286/10 (80286) is an advanced, high-performance microprocessor with specially optimised capabilities for multiple user and multi-tasking systems. The 80286 has built-in-memory protection that supports operating systems and task isolation as well as program and data privacy within tasks. The 80286 includes memory management capabilities that map upto 2^{30} bytes (one gigabyte) of virtual address space per task into 2^{24} bytes (16 megabytes) of physical memory.

The iAPX286 is upward compatible with iAPX86 and 88 software. Using iAPX 86 real address mode, the 80286 is object code compatible with existing iAPX86, 88 software. In project virtual address mode, the 80286 is source code compatible with iAPX 86, 88 software and may require upgrading to use virtual addresses supported by the 80286's integrated management and protection mechanism. Both modes operate at full 80286 performance and execute a superset of the iAPX 86 and 88's instructions.

The 80286 provides special operations to support the efficient implementation and execution of operating systems. For example, one instruction can end execution of one task, save its state, switch to a new task, load its state, and start execution of the new task. The 80286 also supports virtual memory systems by providing a segment-not-present exception and restartable instructions.

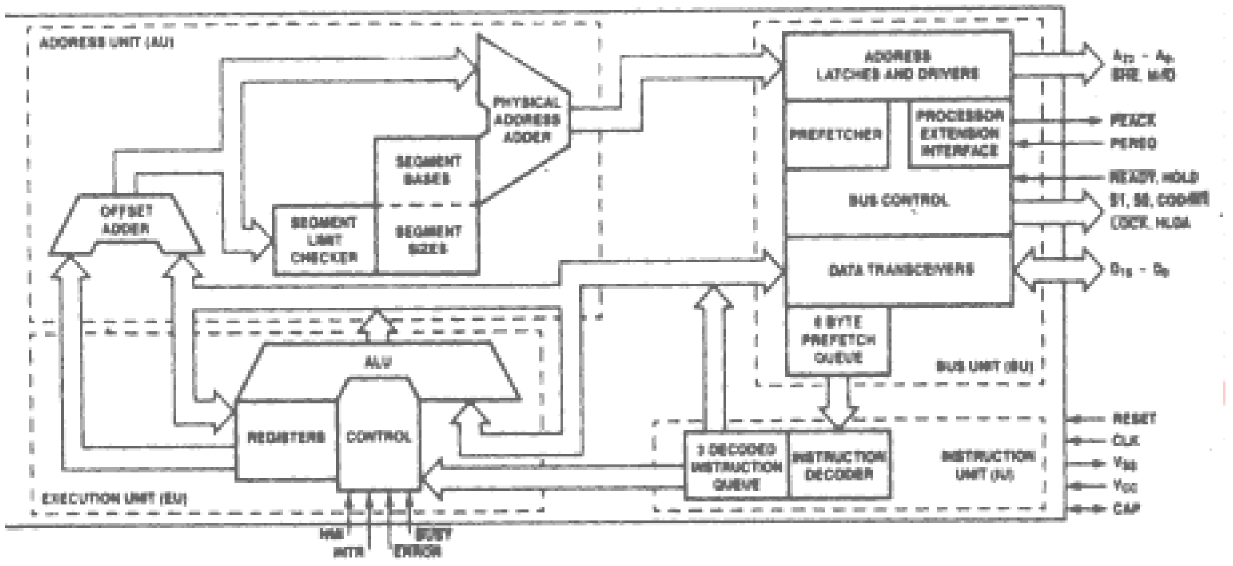


Figure 80286 Internal Block Diagram

Component Pad View - As viewed from underside of component when mounted on the board.

P.C.Board View-As viewed from the component side of the P.C.board

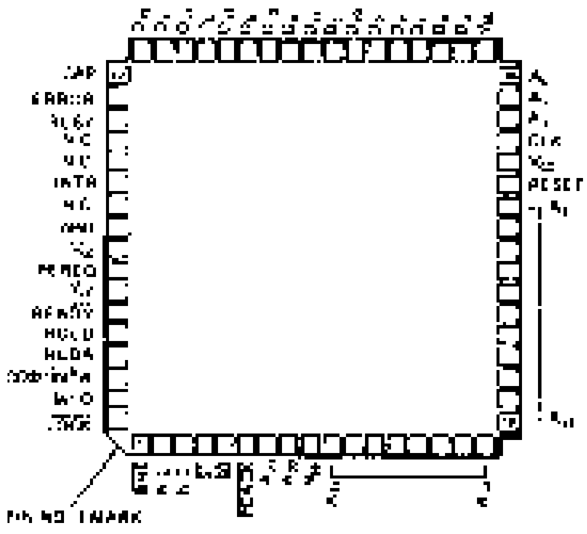


FIGURE 14-10 Pin diagram for 80286 microprocessor.

Figure 80286 Pin Configuration

NOTE : N.C. pads must not be connected.

FUNCTIONAL DESCRIPTION

Introduction :

The 80286 is an advanced, high-performance micro-processor with specially optimised capabilities for multiple user and multi-tasking systems. Depending on the application, the 80286's performance is upto six times faster than the standard 5 MHz8086's, while providing complete upward software compatibility with Intel's iAPX 86, 88, and 186 family of CPU'S.

The 80286 operates in two modes: iAPX86 real address mode. Both modes execute a superset of the iAPX86 and 88-instruction set.

In iAPX86 real address mode program uses real addresses with up to one megabyte of address space. Programs use virtual addresses in protected virtual address mode, also called protected mode. In protected mode, the 80286 CPU automatically maps 1 gigabyte of virtual addresses per task into a 16-megabyte real address space. This mode also provides memory protection to isolate the operating system and ensure privacy of each task programs and data. Both modes provide the same base instruction set registers, and addressing modes.

The following Functional Description describes first, the base 80286 architecture common to both modes, second, iAPX86 real address mode, and third, protected mode.

IAPX 286/10 BASE ARCHITECTURE: The iAPX86, 88, 186 and 286 CPU family all contain the same basic set of registers, instructions, and addressing modes. The 80286 processor is upward compatible with the 8086, 8088, and 80186 CPU's.

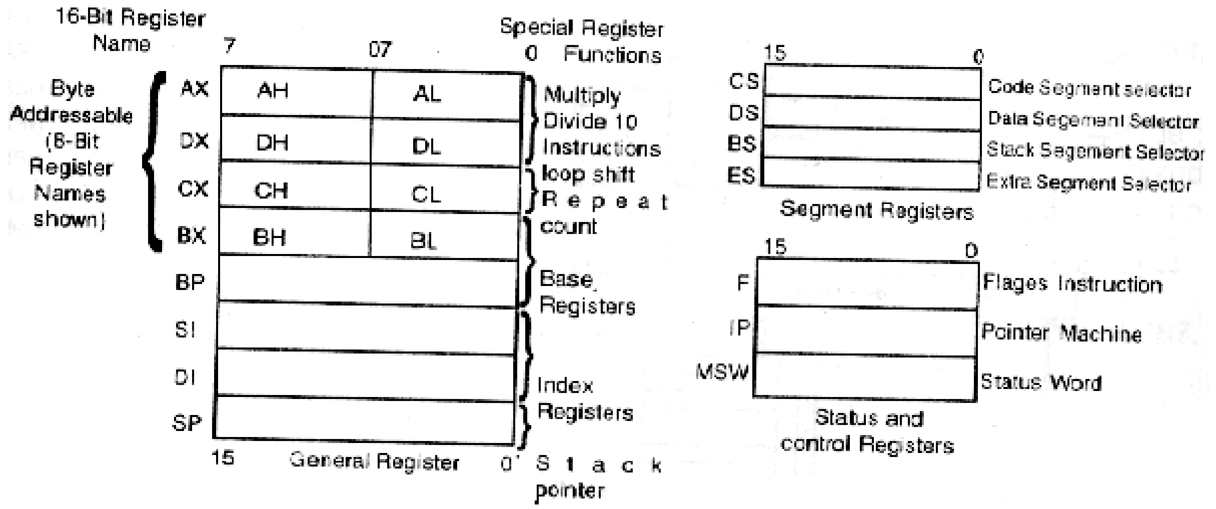
Register Set: The 80286-bus architecture has fifteen registers as shown in figure. These registers are grouped into the following four categories.

General Registers: Eight 16-bit general-purpose registers used to contain arithmetic and logical operands. Four of these (AX, BX, CX, and DX) can be used either in their entirely as 16-bit words or split into pairs of separate 8-bit registers.

Segment Registers: Four 16-bit special purpose registers select, at any given time, the segments of memory that are immediately addressable for code, stack, and data.

Base and Index Registers: Four of the general-purpose registers may also be used to determine offset addresses of operands in memory. These registers may contain base addresses or indexes to particular locations within a segment. The addressing mode determines the specific registers used for operand address calculations.

Status and Control Registers: The 3 16-bit special purpose registers record for control certain aspects of the 80286 processor state including the Instruction Pointer, which contains the offset address of the next sequential instruction to be executed.



80287
80-Bit HMOS

NUMERIC PROCESSOR EXTENSION

The Intel 80287 is a high performance numeric processor extension that extends the iAPX 286/10 architecture with floating point, extended integer and BCD data types. The iAPX 286/20 computing system (80286 with 80287) fully conforms to the proposed IEEE Floating-Point Standard. Using a numeric oriented architecture, the 80287 adds over fifty mnemonics to the iAPX 286/20 instruction set making the iAPX 286/20 a complete solution for high performance numeric processing. The 80287 is implemented in N-channel, depletion load, silicon gate technology (HMOS) and packaged in a 40-pin ceramic package. The iAPX 286/20 is object code compatible with the iAPX 86/20 and iAPX 88/20.

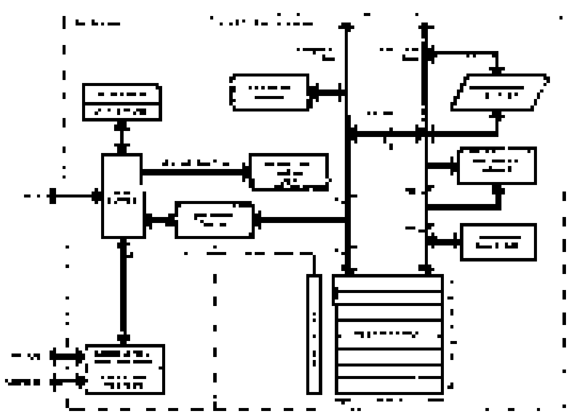
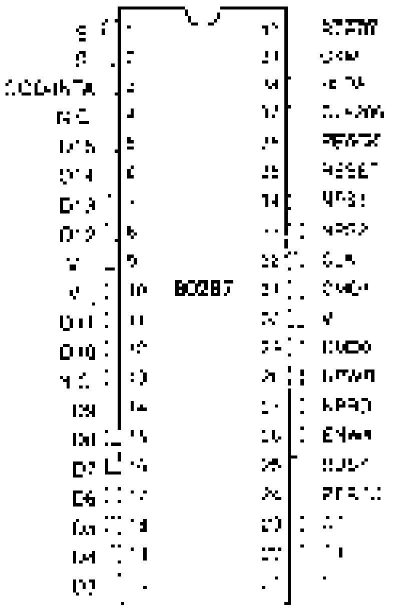


Figure 1.80287 Block Diagram



Note: N.C. Must not be connected

Figure 2.80287 Pin Configuration

FUNCTIONAL DESCRIPTION

The 80287 Numeric Processor Extension (NPX) provides arithmetic instructions for a variety of numeric data types in iAPX 286/20 system. It also executes numerous built-in transcendental functions (e.g. tangent and long functions). The 80287 executes instruction in parallel with a 80286. It effectively extends the register and instruction set of an iAPX 286/10 system for existing iAPX 286 data types and adds several new data type as well. Figure 3 presents the program visible register model of the iAPX 286/20. Essentially, the 80287 can be treated as an additional resource or an extension to the iAPX 286/10 that can be used as a single unified system, the iAPX 286/20.

The 80287 has two operating modes similar to the two modes of the 80286. When reset, 80287 is in the real address mode. It can be placed in the protected virtual address mode by executing the SETPM ESC instruction. The 80287 cannot be switched back to the real address mode except by reset. In the real address mode, the iAPX 286/20 is completely software compatible with iAPX 86/20, 88/20.

Once in protected mode, all references to memory for numeric data or status information, obey the iAPX 286 memory management and protection rules giving a fully protected extension of the 80286 CPU. In the protected mode, iAPX 286/20 numeric software is also completely compatible with iAPX 86/20 and iAPX 86/20.

Review Questions

1. What is use of Assembler & Compiler ?
2. What do you mean by Source program ?
3. How many channels can be used through DMA ?
4. Explain the procedure for DMA request ?
5. What is the use of DMA in PC ?
6. Draw a timing diagram for an 8237 block mode transfer of 4 bytes from an I/O device to memory ?
7. What is the significance of S1, S0 Signals in 80286 ?
8. What are two different modes of 80287 Explain in details ?
9. Explain the utility of a coprocessor in PC-AT ? What signals are involved to interface it with CPU ?

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

Chapter Six

COMMUNICATION THEORY

Features :

6.1 Introduction

In this chapter, we will study the communication system. The list of applications involving the use of communication in one or the other way is almost endless. In the most fundamental sense, communication involves implicitly the transmission of information from one point to another through a succession of processes.

6.2 Communication System:

Figure 6.1 shows the basic functional blocks of a communication system .The overall purpose of this system is to transfer information from one point in space and time, called the source, to another point, the user destination .As a rule, the message produced by a source is not electrical. Hence an input transducer is required for converting the message to a time varying electrical quantity called a message signal. At the destination point, another transducer converts the electrical waveform to the appropriate message.

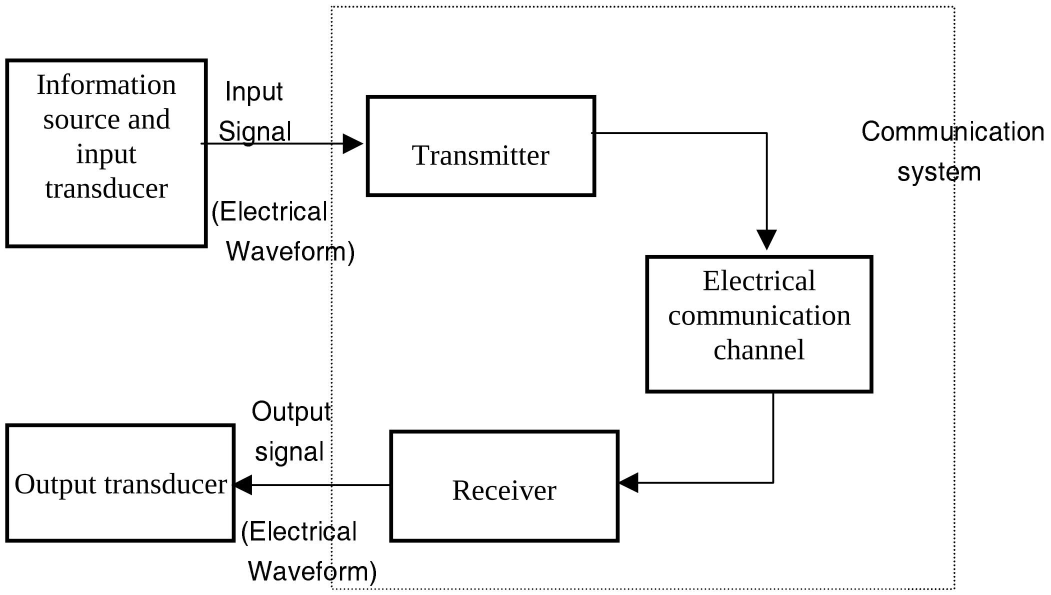


Fig 6.1 An electrical communication system

The information source and the destination point are usually separated in space .The channel provides the electrical connection between the information source and the user. The channel can have many different forms such as a microwave radio link over free space, a pair of wires, or an optical fibre. Regardless of its type, the channel degrades the transmitted signal in a number of ways .The degradation is a result of signal distortion due to imperfect response of the channel and due to undesirable electrical signals (noise) and interference. Noise and signal distortion are two basic problems of electrical communication. The transmitter and the receiver in a communication system are carefully designed to avoid signal distortion and minimise the effects of noise at the receiver so that a faithful reproduction of the message emitted by the source is possible.

The transmitter couples the input message signal to the channel. While it may sometimes be possible to couple the input transducer directly to the channel, it is often necessary to process and modify the input signal or efficient transmission over the channel. Signal processing operations performed by the transmitter include amplification, filtering, and modulation. The most important of these operations is modulation – a process designed to match the properties of the transmitted signal to the channel through the use of a carrier wave.

6.3 Modulation

Modulation is the systematic variation of some attribute of a carrier waveform such as the amplitude, phase, or frequency in accordance with a function of the message signal. Despite the multitude of modulation techniques, it is possible to identify two basic types of modulation: the continuos carrier wave (CW) modulation and the pulse modulation. In continuos wave carrier modulation the carrier wave form is continuos (usually a sinusoidal waveform), and a parameter of the pulse waveform is changed in proportion to the message signal. In both cases the carrier attribute can be changed in continuos or discrete fashion.

Modulation is used in communication systems for matching signal characteristics to channel characteristics, for reducing noise and interference, for simultaneously transmitting several signals over a single channel, and for overcoming some equipment limitations .The success of a communication system depends to a large extent on the modulation.

The main function of the receiver is to extract the input message signal from the degraded version of the transmitted signal coming from the channel .The receiver performs this function through the process of demodulation, the reverse of the transmitter's modulation process. Because of the presence of noise and other signal degradations, the receiver cannot recover the message signal perfectly. In addition to demodulation, the receiver usually provides amplification and filtering.

Based on the type of modulation scheme used and the nature of the output of the information source, we can divide communication systems into three categories:

1. Analogue communication systems designed to transmit analogue information using analogue modulation methods.
2. Digital communication systems designed for transmitting digital information using digital modulation schemes and
3. Hybrid systems that use digital modulation schemes for transmitting sampled and quantized values of an analogue message signal.

Other ways of categorising communication systems include the classification based on the frequency of the carrier and the nature of the communication channel.

6.4 Decibel

In many problems it is found very convenient to compare two powers on a logarithmic scale rather than on a linear scale. The telephone industry proposed a logarithmic unit, named bel after Alexander Graham Bell. The number of bels by which a power P_2 exceeds a power P_1 is defined as

$$\text{Number of bels} = \log_{10} \frac{P_2}{P_1}$$

For practical purposes it has been found that the unit bel is quite large. Another unit, one tenth as large, is more convenient. This smaller unit is called the decibel (abbreviated as dB), and since one decibel is one tenth of a bel, we have

$$\text{Number of Db} = 10 \times \text{Number of bels} = 10 \log_{10} \frac{P_2}{P_1}$$

Note that the unit dB denotes a power ratio. Therefore, the specification of a certain power in dB is meaningless unless a standard reference level is either implied or is stated explicitly. In communication applications, usually 6mW or 1mW is taken as standard reference level. When 1 mW is taken as reference, the unit dB is often referred to as dBm. A negative value of number of dB means that the power P_2 is less than the reference power P_1.

For an amplifier, P1 may represent the input power and P_2 the output power. If V_1 and V_2 are the input and output voltages of the amplifier, then

$$P_1 = \frac{V_1^2}{R_i}$$

$$\text{And} \quad P_2 = \frac{V_2^2}{R_o}$$

Where R_i and R_o are the input and output impedance of the amplifier. Then

$$\text{Number of dB} = 10 \log 10 \frac{V_2^2/R_o}{V_1^2/R_i}$$

In case the input and output impedance of the amplifier are equal i.e $R_i=R_o=R$,then

$$\text{Number of dB} = 10 \log_{10} \frac{V_2^2}{V_1^2} = 10 \log 10 \frac{(V_2)^2}{(V_1)^2}$$

$$= 10 \times 2 \log_{10} \frac{V_2}{V_1} = 20 \log 10 \frac{V_2}{V_1}$$

In general, the input and output impedance are not always equal. But the expression is adopted as a covenient definition of the decibel voltage gain of an amplifier, regardless of the magnitudes of the input and output impedance.

As an example if the voltage gain of an amplifier is 10,it can be denoted on the dB scale as

$$\text{Gain in dB}=20 \log_{10} \frac{V_2}{V_1} = 20\log_{10}10$$

$$=20 \times 1 = 20 \text{ dB}$$

Why dB is used

You may wonder why we use a logarithmic scale to denote voltage or power gains, instead of using the simpler linear scale. It permits us to denote, both very small as well as very large, quantities of linear scale by conveniently small figures. Thus a voltage gain of 0.000001 may be represented as a voltage gain of –120 dB, or a voltage loss of 120 dB. Similarly, a power gain of 456000 is simply 56.59 on the logarithmic scale.

6.5 BANDWIDTH

The bandwidth of a signal provides a measure of the extent of significant spectral content of the signal for positive frequencies. When the signal is strictly band limited, the bandwidth is well defined. There is no universally accepted definition of bandwidth.

The frequency spectrum is shared by many users. Therefore frequency channels must have limited bandwidth so that their significant frequency components are spread over a range of frequencies which is small compared to the carrier frequencies. There are several definitions of bandwidth that are often encountered. A very common definition arises from the design of filters or the measurement of selectivity in a receiver. In this case the bandwidth is described as the difference between the two frequencies at which the power spectrum density is a certain fraction below the centre frequency, when the filter has been excited.

Another bandwidth that is often encountered, especially in receiver design, is the noise bandwidth. This is defined as the bandwidth which, when multiplied by the centre frequency density, would produce the same total power as the output of the filter or receiver.

6.6 The Radio Spectrum

The electromagnetic spectrum is one of our irreplaceable natural resources; once filled with uses, it cannot be expanded. We have touched on ways to better utilize this spectrum in our discussions of efficient use of bandwidth through sharpening the boundaries between frequency-multiplexed channels and the use of time division multiplexing. The reason we need to use these bandwidth conservation techniques is that the portion of the spectrum we can use for broadcasting is, when compared to the entire spectrum, quite small, as shown in Table 6.2. It ranges from very low frequencies of a few kilohertz and wavelengths of several kilometers up to 300 gigahertz (billions of cycles per second) at which point radio microwaves move into the far infrared. (In this context, we use the term radio to encompass all forms of wireless communication, including television, radio broadcasting, and telephone calls sent by microwave radio and satellite.

Radio waves can travel on or near the ground (the ground and direct waves), or they can travel upward some hundreds of kilometers where they are sent or reflected by a band of ionized particles trapped in space- the ionosphere- and bounced back to earth some distance away. These are the skywaves in fig. 6.3. The direct waves may travel in straight lines or line-of-sight between antenna over short distances, usually not more than 100 kilometers, or as ground waves reflected off the ground in order to reach the receiving antenna.

Frequency in Hertz	**Wavelength in Meters**	**Designation**
10^1	10^7	Extremely Low Frequency (ELF)
10^2	10^6	Voice Frequency (VF)
10^3	10^5	Very Low Frequency (VLF)
10^4	10^4	Low Frequency (LF)
10^5	10^3	Medium Frequency (MF)
10^6	10^2	High Frequency (HF)
10^7	10^1	Very High Frequency (VHF)
10^8	$10^0(=1)$	Ultra High Frequency (UHF)
10^9	10^{-1}	Super High Frequency (SHF)
10^{10}	10^{-2}	Extremely High Frequency (EHF)
10^{11}	10^{-3}	Band No. 12
10^{12}	10^{-4}	

Table 6.2

Purists will add that radio waves also can travel through the earth by way of the rock strata.) Higher frequency signals, those in the very high frequency range (VHF) and Ultra High Frequency (UHF) travel between antenna directly, or by line-of-sight.

How this radio spectrum is utilized is governed by the physics of radio wave propagation as just described and by the practical limits of communications engineering design. Let us consider just what is taking place when you sit on the beach with your portable radio listening to the ballgame. The sportscaster in the booth overlooking the

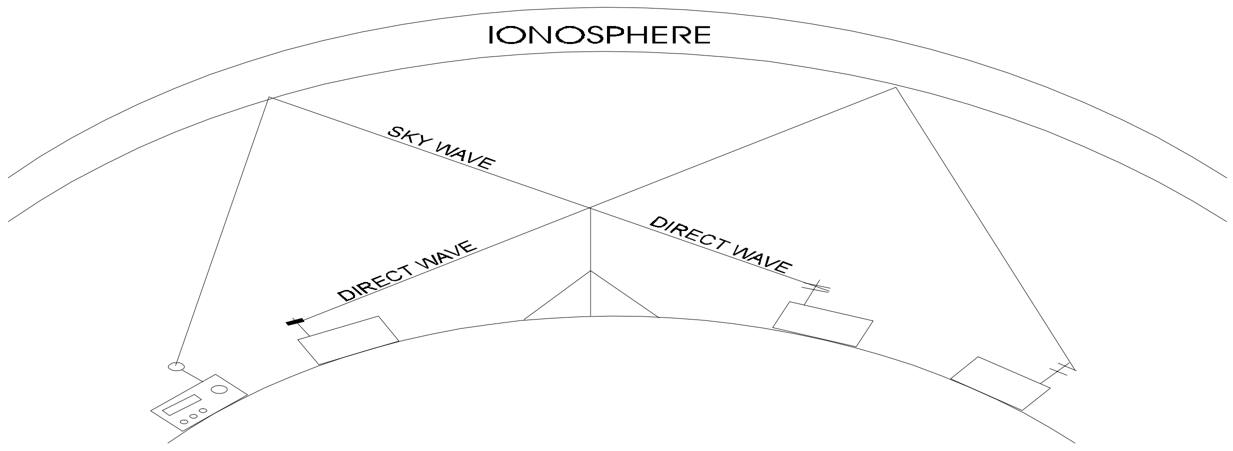


Fig 6.3

field is shouting the play-by-play into the microphone, which transforms his voice into electrical signals, much as the microphone in the telephone handset does. These signals are carried by cable from the range of frequencies of the component sinusoidal signals used to compose the voice signal-in other words, the difference between the highest and lowest frequency used – is called the bandwidth of the signal.

Bandwidth is an extremely important characteristic because the cost of a signal's transmission depends fundamentally on its bandwidth. Generally speaking, higher bandwidth signals cost more to transmit. Because bandwidth is the difference between the cycles per second of the highest frequency and that of the lowest frequency making up the signal, it is also measured in cycles per second. In honor of Heinrich Hertz, who produced the electromagnetic waves about which Clerk Maxwell theorized, we now call a cycle per second a hertz. Today's frequencies are getting higher and higher and bandwidths are getting broader so you will be reading and talking about kilohertz (kHz) or 1000 cycles per second, megahertz (MHz) or 1,000,000 cycles per second, and gigahertz (GHz) or 1,000,000,000 cycles per second.

Here are bandwidths for some common signal transmission systems we have been talking about:

Telegraph	40 hertz
Telephone	4000 hertz
Hi-Fi-Music	20,000 hertz
Color Television	6,000,000 hertz

Different communication channels transmit signals best at different frequencies; they have different bandwidth capacities. Consider the simplest channel of all the air space between two people speaking to one another. The can understand each other if they hear one another at the normal formed when we speak. We can extend the distance between the parties speaking by using loudspeaker systems which can be either mechanical (megaphone) or electrical (an amplification system). But if we want to transmit speech over very long distances a channel or transmission mode better suited for the long-distance transmission of speech or any other signals containing information is required.

6.7 SIGNAL-TO-NOISE RATIO

Communication receivers are required to receive and process a wide range of signal powers, but in most cases it is important that they be capable of receiving distant signals whose power has been attenuated billions of time during transmission. The extent to which such signals can be received usefully is determined by the noise levels received by the antenna and those generated within the receiver. It is also necessary that the receiver produce a level of output power suitable for the application .The ratio of the output power of a device to the input power is known as the gain.

When the gain is sufficiently high, the weakest signal power that may be processed satisfactorily is noise limited. This signal level is referred to as the sensitivity of the system at a particular time and varies depending on the external noise level. It is possible in some systems for the external noise to fall sufficiently so that the system sensitivity is established by the internal noise of the receiver.

To carry out the noise analysis of analog modulation systems, we obviously need a criterion that describes in a meaningful way the noise performance of the system under study. In the case of analog modulation systems, the customary practice is to use the output signal-to-noise ratio as an intuitive measure for describing the fidelity with which the demodulation process in the receiver recovers the original message from the received modulated signal in the presence of noise. Output signal-to-noise ratio is defined as the ratio of the average power of the message signal to the average power of the noise, both measured at the receiver output. Let $(SNR)_o$ denote the output signal-to-noise ratio, expressed as :

$$(SNR)_0 = \frac{\text{Average power of message signal at the receive output}}{\text{Average power of noise at the receiver output}}$$

The output signal-to-noise ratio is unambiguous as long as the recovered message and noise at the demodulator output are additive. This requirement is satisfied exactly in the case of linear receivers using coherent detection, and approximately in the case of nonlinear receivers (e.g. using envelope detection or frequency discrimination) provided that the average input noise power is small compared with the average carrier power.

The calculation of the output signal-to-noise $(SNR)_o$ involves the use of an idealized receiver model, the details of which naturally depend on the channel noise and the type of demodulation used in the receiver. For the present, we wish to point out that knowledge of $(SNR)_o$ by itself may be insufficient, particularly when we have to compare the output signal-to-noise ratios of different analog modulation - demodulation systems. In order to make such a comparison meaningful, we introduce the idea of a base band transmission model, as depicted in fig. 6.4. In this model, two assumptions are made:

1. The transmitted or modulated message signal power is fixed.
2. The baseband low-pass filter passes the message signal, and rejects out-of-band noise.

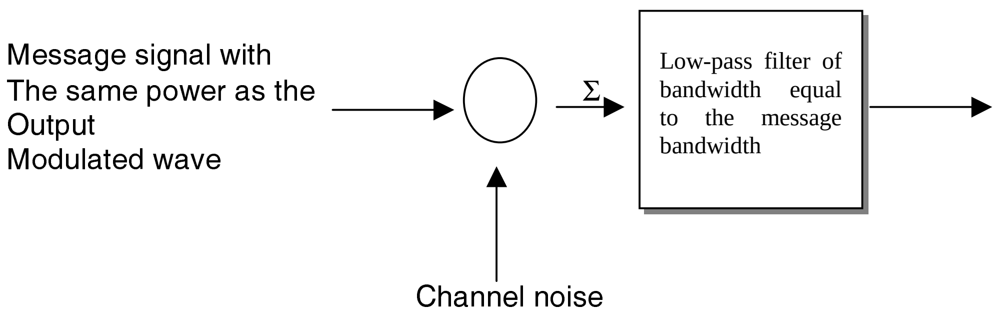


Figure 6.4. The baseband transmission of a message signal for calculating the channel signal-to-noise ratio.

Accordingly, we may define the channel signal-to-noise ratio, referred to the receiver input as

$$(SNR)_0 = \frac{\text{Average power of the modulated message signal}}{\text{Average power of noise measured in the message bandwidth}}$$

This ratio is independent of the type of modulation or demodulation used. The channel signal-to-noise ratio may be viewed as a frame of reference for comparing different modulation systems. Specifically, we may normalize the noise performance of a specific modulation-demodulation system by dividing the output signal-to-noise ratio of the system by the channel signal-to-noise ratio. We may thus define a figure of merit for the system as :

$$\text{Figure of merit} = \frac{(SNR)_0}{(SNR)c}$$

Clearly, the higher the value that the figure of merit has, the better the noise performance of the receiver.

6.8 RF SECTIONS AND CHARACTERISTICS

A radio receiver always has an RF section, which is a tunable circuit connected to the antenna terminals. It is there to select the wanted frequency and reject some of the unwanted frequencies. However, such a receiver need not have a RF amplifier following this tuned circuit. If there is an amplifier, its output is fed to the mixer, at whose input another tunable circuit is present. In many instances, however, the tuned circuit connected to the antenna is the actual input circuit of the mixer; the receiver is then said to have no RF amplifier or, more simply, no RF stage.

Reasons for use and functions of RF Amplifier: The receiver having an RF stage is undoubtedly superior in performance to the receiver without one, all else being equal. One the other hand, there are some instances in which an RF amplifier is uneconomical, i.e., where its inclusion would increase the cost of the receiver significantly while improving performance only marginally. The best example of this kind of receiver is a domestic one used in a high-signal-strength area, such as the metropolitan area of any large city.

The advantages of having an RF amplifier are as follows (reasons 4-7 are either more specialized or less important)

1. Greater gain i.e. better sensitivity
2. Improved image-frequency rejection
3. Improved signal-to-noise ratio
4. Improved rejection of adjacent unwanted signals, i.e. better selectivity
5. Better coupling of the receiver to the antenna (important at VHF and above)
6. Prevention of spurious frequencies from entering the mixer and heterodyning there to produce an interfering frequency equal to the IF from the desired signal.
7. Prevention of reradiation of the local oscillator through the antenna of the receiver (relatively rare)

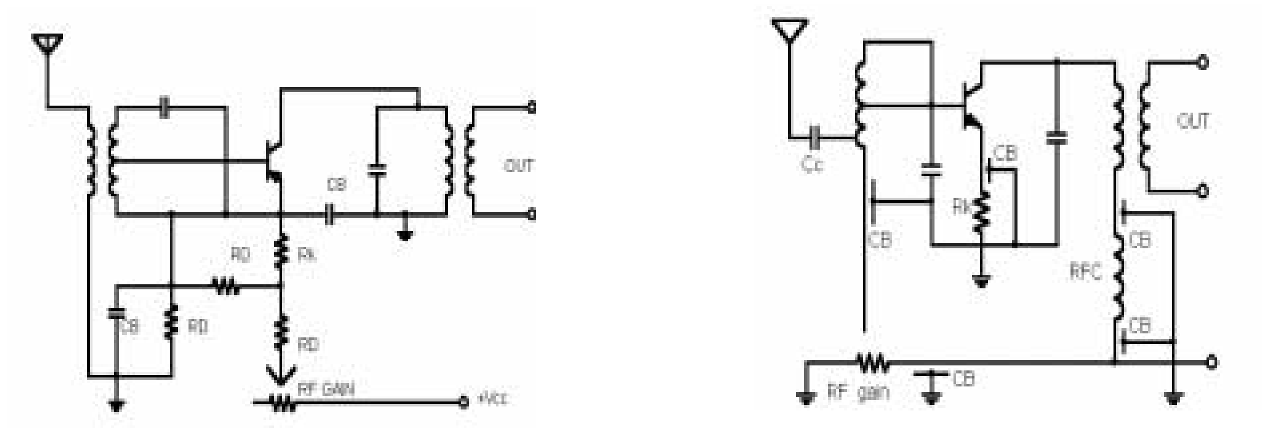


Fig 6.5 Transistor RF amplifiers a) Medium-frequency b) VHF

The single-tuned, transformer-coupled type is the amplifier most commonly employed for RF amplification, as illustrated in Fig. 6.5 a) & b) both diagrams in the figure are seen to have an RF gain control, which is very rare with domestic receivers but quite common in communications receivers. Whereas the medium frequency amplifier of Fig. 6.5 a) is quite straightforward, the VHF amplifier of Fig. 6.5 b) contains a number of refinements. Feed-through capacitors are used as bypass capacitors and, in conjunction with the RF choke, to decouple the output from the V_{CC} . As indicated in Fig. 6.3 b) one of the electrodes of a feed through capacitor is the wire running through it. This is surrounded by the dielectric, and around that is the grounded outer electrode; this arrangement minimizes stray inductance in series with the bypass capacitor. Feed- through capacitors are almost invariably provided for bypassing at VHF and often have a value of 1000 pF. In addition, a single-tuned circuit is used at the input and is coupled to the antenna by means of a trimmer (the latter being manually adjustable for matching to different antennas). Such coupling is used here because of the high frequencies involved. Finally, RF amplifiers in practice have the input and output tuning capacitors ganged to each other and to the one tuning the local oscillator.

Review Questions

1. Define a communication system.
2. What is the unit in which the power of the communication system is defined?
3. What is the bandwidth for the following signal transmission systems

   - Telegraph
   - Telephone
   - Television

NOTES

NOTES

NOTES

NOTES

NOTES

NOTES

Chapter Seven

MODULATION

Features :

7.1 INTRODUCTION

Communications are transmitted by sending time-varying waveforms generated by the source, or by sending waveforms derived from those of the source. In radio communications the varying waveforms derived from the source are transmitted by changing the parameters of a sinusoidal wave at the desired transmission frequency. This process is referred to as modulation, and the sinusoid is referred to as the carrier. In this chapter we will review the types of analogue modulation. Analogue modulation is used for transmitting speech, music, telephone, television, and some telemetering.

7.2 AMPLITUDE MODULATION

In amplitude modulation, the amplitude of a carrier signal is varied by the modulating voltage, whose frequency is invariably lower than that of the carrier. In practices, the carrier may be high-frequency (HF) while the modulation is audio. Formally, AM is defined as a system of modulation in which the amplitude of the carrier is made proportional to the instantaneous amplitude of the modulating voltage.

Let the carrier voltage and the modulating voltage, V_c and V_m, respectively, be represented by

$$V_c = \mathbf{V_c} \sin\omega_c t \qquad (7\text{-}1)$$

$$V_m = V_m \sin\omega_m t \qquad (7\text{-}2)$$

Note that phase angle has been ignored in both expressions since it is unchanged by the amplitude modulation process. However, it will certainly not be possible to ignore phase angle when we deal with frequency and phase modulation.

From the definition of AM, it follows that the (maximum) amplitude V_c of the unmodulated carrier will have to be made proportional to the instantaneous modulating $V_m = V_m \sin\omega_m t$ when the carrier is amplitude-modulated.

FREQUENCY SPECTRUM OF THE AM WAVE

We shall show mathematically that the frequencies present in the AM wave are the carrier frequency and the first pair of sideband frequencies, where a sideband frequency is defined as

$$f_{SB} = f_c \pm n f_m \qquad (7.3)$$

and in the first pair n = 1.

When a carrier is amplitude-modulated, the proportionality constant is made equal to unity, and the instantaneous modulating voltage variations are superimposed onto the carrier amplitude. Thus when there is temporarily no modulation, the amplitude of the carrier is equal to its un-modulated value. When modulation is present, the amplitude of the carrier is varied by its instantaneous value. The situation is illustrated in Fig. 7.1, which shows how the maximum amplitude of the amplitude-modulated voltage is made to vary in accordance with modulating voltage changes. Figure 7.1. also shows that something unusual will occur if V_m is greater than V_c. This, and the fact that the ratio V_m / V_c often occurs, leads to the following definition of the modulation index:

$$m = \frac{V_m}{V_c} \qquad (7.4)$$

The modulation index is a number lying between 0 and 1, and it is very often expressed as a percentage and called the percentage modulation. From fig. 7.1. it is possible to write an equation for the amplitude of the amplitude-modulated voltage. Thus we have :

$$A = V_c + v_m = V_c + V_m \sin \omega_m t = V_c + mV_c \sin \omega_m t$$

$$= V_c (1 + m \sin \omega_m t) \qquad (7.5)$$

The instantaneous voltage of the resulting amplitude-modulated wave is
$v = A \sin\theta = A \sin \omega_c t = V_c (1 + m \sin \omega_m t) \sin \omega_c t$ (7.6)

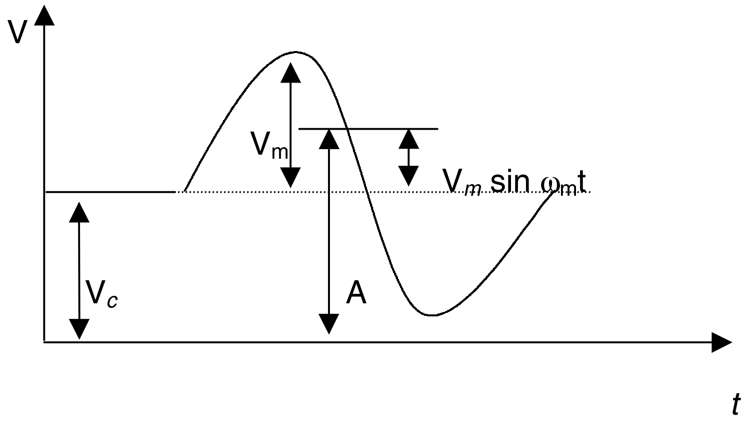


Figure 7.1. Amplitude of AM Wave.

Equation 7.6 may be expanded, by means of the trigonometrical relation
sin x sin y = ½ [cos (x-y)-cos (x+y), to give

$$v = V_c \sin \omega_c t + \frac{mV_c}{2} \cos (\omega_c - \omega_m) t - \frac{mV_c}{2} \cos (\omega_c + \omega_m) t \qquad (7.7)$$

It has thus been shown that the equation of an amplitude-modulated wave contains three terms. The first term is identical to eq. And represents the unmodulated carier. It is thus apparent that the process of amplitude modulation has the effect of adding to the unmodulated wave, rather than changing it. The two additional terms produced are the two sidebands outlined. The frequency of the lower sideband (LSB) is $f_c - f_m$, and the frequency of the upper sideband (USB) is $f_c + f_m$. The very important conclusion to be made at this stage is that the bandwidth required for amplitude modulation is twice the frequency of the modulating signal. In modulation by several sine waves simultaneously, as in the AM broadcasting service, the bandwidth required is twice the highest modulating frequency.

7.3 FREQUENCY MODULATION

Frequency modulation is a system in which the amplitude of the modulated carrier is kept constant,while its frequency is varied by the modulating signal.
The general equation of an unmodulated wave, or carrier, may be written as

$x = A \sin (\omega t + \phi)$

Where x = instantaneous value (of voltage or current)
A = (maximum) amplitude

ω = angular velocity, radians per second (rad/s)
ϕ= phase angle, rad

Note that ωt represents an angle in radians.

If any one of these three parameters is varied in accordance with another signal, normally of a lower frequency, then the second signal is called the modulation, and the first is said to be modulated by the second. Amplitude modulation, already discussed, is achieved when the amplitude A is varied alteration of the phase angle ϕ will yield phase modulation. Finally, if the frequency of the carrier is made to vary, frequency modulated waves are obtained.

For simplicity, it is again assumed that the modulating signal is sinusoidal. This signal has two important parameters which must be represented by the modulation process without distortion; namely, its amplitude and frequency. It is assumed that the phase relations of a complex modulation signal will be preserved. By the definition of frequency modulation, the amount by which the carrier frequency is varied from its unmodulated value, called the deviation, is made proportional to the instantaneous value of the modulating voltage. The rate at which this frequency variation or oscillation takes place is naturally equal to the modulating frequency.

The situation is illustrated in Fig. 7.2 which shows the modulating voltage and the resulting frequency-modulated wave. Fig. 7.2 also shows the frequency variation with time, which is seen to be identical to the variation with time of the modulating voltage. The result of using that modulating voltage to produce AM is also shown for comparision. As an example of FM, all signals having the same amplitude will deviate the carrier frequency by the same amount, say 45 KHz, no matter what their frequencies. Similarly, all signals of the same frequency say 2 KHz, will deviate the4 carrier at the same rate of 2000 times per second, no matter what their individual amplitudes. The amplitude of the frequency-modulated wave remains constant at all times; this is in fact, the greatest single advantage of FM.

Mathematical Representation of FM

From Fig. 7.2, it is seen that the instantaneous frequency f of the frequency-modulated wave is given by :

$$f = f_c\,(1+kV_m \cos\ \omega_m t)$$

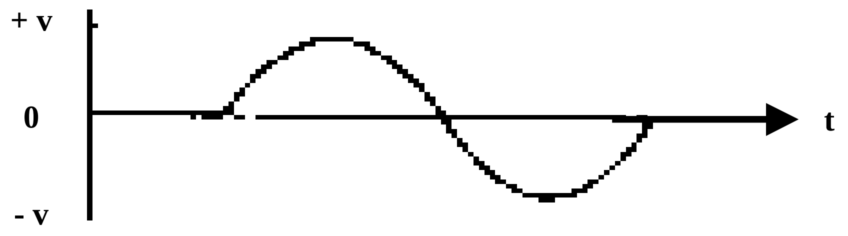
Where f_c = unmodulated (or average) carrier frequency
K = proportionality constant
$V_m \cos \omega_m t$ = instantaneous modulating voltage (cosine being preferred for simplicity in calculations)

The maximum deviation for this particular signal will occur when the cosine term has its maximum value, that is, ± 1. Under these conditions, the instantaneous frequency will be

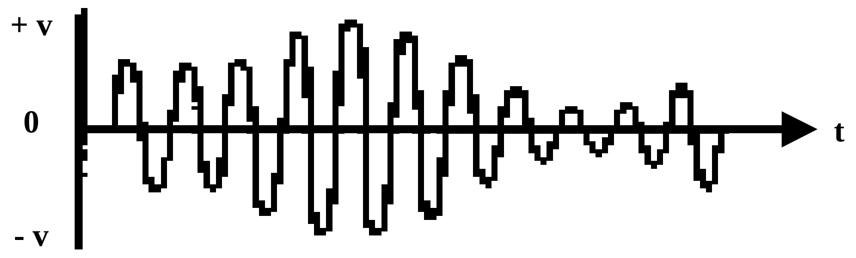
$$f = f\text{c}\ (1\pm kV_m)$$

so that the maximum deviation δ will be given by

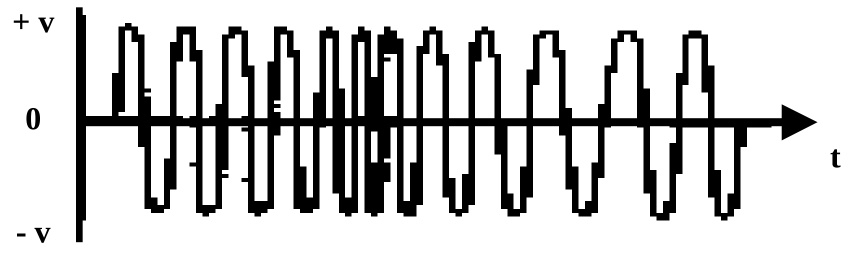
$$\delta = kV_m f_c$$



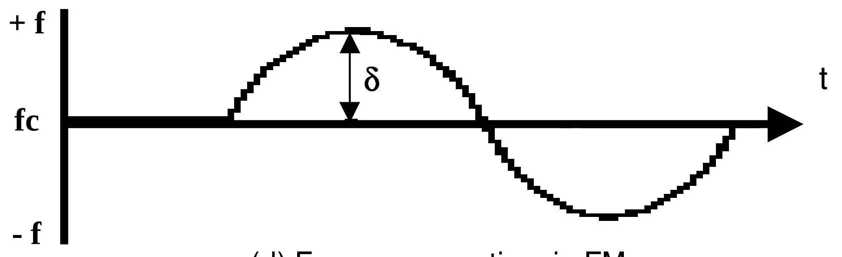
(a) Modulating Signal



(b) Amplitude modulation



(c) Frequency modulation



(d) Frequency vs. time in FM

Figure 7.2. Basic Modulation Waveforms

The instantaneous amplitude of the FM signal will be given by a formula of the form
$v = A \sin [F(\omega_c, \omega_m)] = A \sin \theta$

Where $F(\omega_c, \omega_m)$ is some function, as yet undetermined, of the carrier and modulating frequencies.

Frequency & Amplitude Modulation:

Frequency & amplitude modulation are compared on a different basis from that for FM and PM. These are both practical systems, quite different from each other, and so the performance and characteristics of the two systems will be compared. To begin with, frequency modulation has the following advantages:

1. The amplitude of the frequency-modulated wave is constant. It is thus independent of the modulation depth, whereas in AM modulation depth governs the transmitted power. This means that, in FM transmitters, low-level modulation may be used but all the subsequent amplifiers can be class C and therefore more efficient. Since all these amplifiers will handle constant power, they need not be capable of managing up to four times the average power, as they must in AM. Finally, all the transmitted power in FM is useful, whereas in AM most of it is in the transmitted carrier, which does not indicate any modulation changes.

2. FM receivers can be fitted with amplitude limit's to remove the amplitude variations caused by noise.This makes FM reception a good deal more immune to noise than AM reception.

3. It is possible to reduce noise still further by increasing the deviation . This is a feature which AM does not have, since it is not possible to exceed 100 percent modulation without causing severe distortion.

4. Commercial FM broadcasts being in 1940 decades after their AM counterparts. Consequently, they have a number of advantages due to better planning and other circumstances. The following are the most important ones :

   (a) Standard frequency allocations provide a guard band between commercial FM stations, so that there is less adjacent-channel interference that in AM.
   (b) FM broadcasts operate in the upper VHF and UHF frequency ranges, at which there happens to be less noise than in the MF and HF ranges occupied by AM broadcasts.
   (c) At the FM broadcast frequencies, the space wave is used for propagation, so that the radius of operation is limited to slightly more than line of sight. It is thus possible to operate several independent transmitters on the same frequency with considerably less interference than would be possible with AM.

The advantages are not all one-sided, or there would be no AM transmissions left. The following are some of the disadvantages of FM;

- ❖ A much wider channel is required by FM, up to 10 times as large as that needed by AM. This is the most significant disadvantage of FM.
- ❖ FM transmitting and receiving equipment tends to be more complex, particularly for modulation and demodulation.
- ❖ Since reception is limited to line of sight, the area of reception for FM is much smaller than for AM this may be an advantage for cochannel allocations.

7.4 Pre-emphasis and De-emphasis:

Noise has a greater effect on the higher modulating frequencies than on the lower ones. Thus, if the higher frequencies are artificially boasted at the transmitter and correspondingly cut at the receiver, an improvement in noise immunity could be expected. This boosting of the higher modulating frequencies,

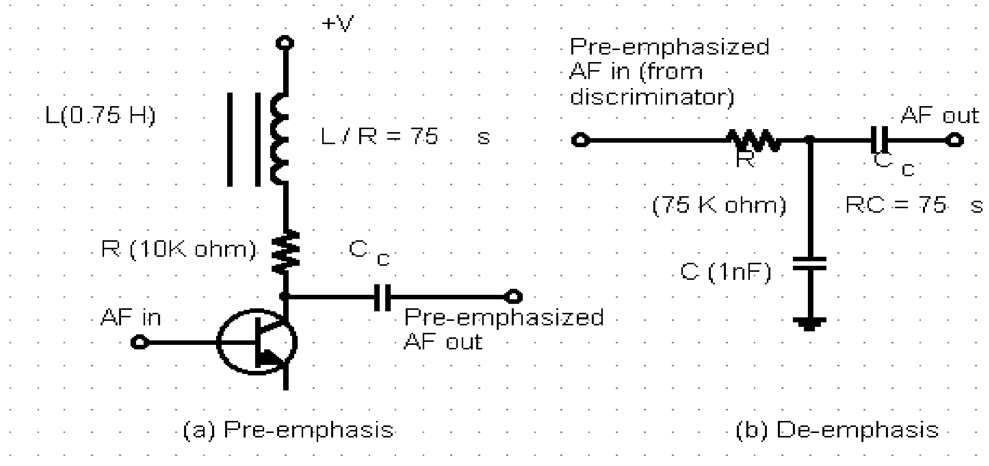


Fig 7.3.

in accordance with a prearranged curve, is termed pre-emphasis, and the compensation at the receiver is called de-emphasis. An example of a circuit used for each function is shown in fig 7.3.

Take two modulating signals having the same initial amplitude, with one of them pre-emphasized to (say) twice this amplitude, whereas the other is unaffected (being at a much lower frequency). The receiver will naturally have to de-emphasize the first signal by a factor of 2, to ensure that both signals have the same amplitude in the output of the receiver. Before demodulation i.e., while susceptible to noise interference, the emphasized signal had twice the deviation it would have had without pre-emphasis and was thus more immune to noise. Alternatively, it is seen that when the signal is de-emphasized, any noise sideband voltages de-emphasized with it and therefore have correspondingly lower amplitude than they would have had without emphasis again, their effect on the output is reduced.

The amount of pre-emphasis in U.S.F□□broadcasting, and in the sound transmissions accompanying television, has been standardized at 75□s, whereas a number of other services, notably European and Australian broadcasting and TV sound transmission use 50 □s. The usage of microseconds for defining emphasis is standard. A 75-□s de-emphasis corresponds to a frequency response curve that is 3 dB down at the frequency whose time constant RC is 75 ms. This frequency is given by □f = 1/2□RC and is therefore 2120 Hz with 50 □s de-emphasis it would be 3180 Hz. Fig. 7.3 shows pre-emphasis and de-emphasis curves for a 75-□s emphasis.

7.5 Phase Modulation :

Phase modulation is a similar system in which the phase of the carrier is varied instead of its frequency.Strictly speaking, there are two types of continuous-wave modulation; amplitude modulation and angle modulation. Angle modulation may be subdivided into two distinct types; frequency modulation and phase modulation (PM). Thus, PM and FM are closely allied, and this is the first reason for considering PM here. The second reason is somewhat more practical; it is possible to obtain frequency modulation from phase modulation by the so-called Armstrong System. It must be stressed, however, that phase modulation as such is not used in practical analog transmission systems.

If the phase f in the equation $v = A \sin(\omega_c t + \phi)$ is varied so that its magnitude is proportional to the instantaneous amplitude of the modulating voltage, the resulting wave is phase-modulated. The expression for a PM wave is

$$v = A \sin(\omega_c t + \phi_m \sin \omega_m t)$$

where is the maximum value of phase change introduced by this particular modulating signal and is proportional to the maximum amplitude of this modulation.

$$v = A \sin(\omega_c t + m_p \sin \omega_m t)$$

Where $m_p = \phi_m$ = modulation index for phase modulation.

To visualize phase modulation, consider a horizontal metronome or pendulum placed on a rotating record turntable. As well as rotating the arm of this metronome is swinging sinusoidally back and forth about its mean point. If the maximum displacement of this swing can be made proportionally to the size of the “push” applied to the metronome, and if the frequency of swing can be made equal to the number of “pushes” per second, then the motion of the arm is exactly the same as that of a phase-modulated vector. Actually, PM seems easier to visualize than FM.

Intersystem Comparisons:

Frequency and Phase Modulation from the purely theoretical point of view, the difference between FM and PM is quite simple-the modulation index is defined differently in each system. However, this is not nearly as obvious as the difference between AM and FM, and it must be developed further. First, however, the similarity will be stressed.

In phase modulation, the phase deviation is proportional to the amplitude of the modulating signal and therefore independent of its frequency. Also, since the phase modulated vector sometimes leads and sometimes lags the reference carrier vector, its instantaneous angular velocity must be continually changing between the limits imposed by ϕ_m ; thus some form of frequency change must be taking place. In frequency modulation, the frequency deviation is proportional to the amplitude of the modulating voltage. Also, if we take a reference vector, rotating with a constant angular velocity which corresponds to the carrier frequency, then the FM vector will have a phase lead or lag with respect to the reference, since the frequency oscillates between $f_c - \delta$ and $f_c + \delta$. Hence FM must be a form of PM. With this close similarity of the two forms of angle modulation established, it now remains to explain the difference.

If we consider FM as a form of phase modulation, we must determine what it is that causes the phase change in FM. Obviously, the larger the frequency deviation, the larger the phase deviation, so that the latter depends at least to a certain extent on the amplitude of the modulation, just as PM. The difference is shown by comparing the definition of PM, which states in part that the modulation index is proportional to the modulating voltage only, with that of the FM which states that the modulation index is also inversely proportional to the modulation frequency. This means that under identical conditions FM and PM are indistinguishable for a single modulating frequency.

Voltage Controlled Oscillators :

Newer receivers control the oscillator band and frequency by electrical rather than mechanical means. Tuning is accomplished by voltage-sensitive capacitors (varactor diodes), and band switching by diodes with low forward conductance. For high-power tuning, inductors having saturable ferrite cores have been used however, these do not appear in receivers. Oscillators that are tuned by varying the input voltage are referred to as VCOs.

7.6 AUTOMATIC FREQUENCY CONTROL :

AFC has been used for many years in some receivers to correct for tuning errors and frequency instabilities. This was of special importance when free-running Los were used. Inaccuracies in the basic tuning of the receiver, and of drifts in some transmitters, could cause the desired signal to fall on the skirts of the IF selectivity, resulting in severe distortion and an increased chance of adjacent channel interference. Provision of a circuit to adjust the tuning so that the received signal falls at (or very near) the center of the IF filter enables the receiver to achieve low demodulation distortion, while maintaining a relatively narrow IF bandwidth for interference rejection.

The need for such circuits has been almost eliminated by the advent of synthesized Los under the control of very accurate and stable quartz crystal standards. However, some case still arise where an AFC may be of value. In some tactical applications it is not possible to afford the power necessary for temperature control of the crystal standard, so that at sufficiently high frequencies the relative drift between the receiver and transmitter may be more than is tolerable in the particular application. Unstable oscillators in some older and less accurate transmitters have not yet been replaced. Current receivers may be required to receive signals from such transmitters for either communications or surveillance. In automobile FM broadcast receivers the cost of synthesis is considerably higher than the cost of AFC, so that progress to high stability has been slower than for communications receivers. Finally, some modulations, notably SSB, require much better frequency accuracy for distortion less demodulation than AM or FM. As such modulations are extended to higher carrier frequencies, AFC can be more economical than still more accurate frequency control. Consequently AFC circuits are still found in receivers, though much less frequently than in the past.

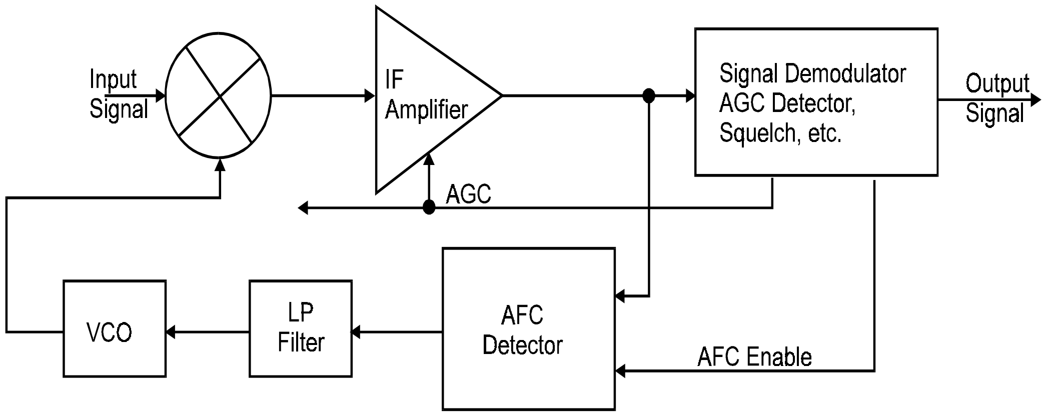


Fig. 7.4 AFC Block Diagram

The basic elements of the AFC loop are a frequency (or phase) detector and a VCO. A typical block diagram is shown in Fig. 7.4 . If the received signal carrier is above or below the nominal frequency, the resulting correction voltage is used to reduce the difference.

7.7 AUTOMATIC GAIN CONTROL (AGC)

AGC is a system by means of which the overall gain of a radio receiver is varied automatically with the changing strength of the received signal to keep the output substantially constant. A dc bias voltage, derived from the detector is applied to a selected number of the RF, IF and mixer stages. The devices used in those stages are ones whose transconductance and hence gain depends on the applied bias voltage or current.

All modern receivers are furnished with AGC, which enables tuning to stations to varying signal strengths without appreciable change in the size of the output signal. Thus AGC “irons out “ input signal amplitude variations; and the gain control does not have to be readjusted every time the receiver is tuned from one station to another, except when the change in signal strengths is enormous. In addition, AGC helps to smooth out the rapid fading which may occur with long distance short-wave reception and prevents the overloading of the last IF amplifier which might otherwise have occurred.

Indirect Synthesizers :

The required frequency range in most synthesizers nowadays is obtained from a variable voltage-controlled oscillator (VCO), whose output is corrected by comparison with that of a reference source. This in-built source is virtually a direct synthesizer. As shown in Fig.7.5. the phase comparator obtains an output from the VCO, compares it with the output of the stable reference source and produces a dc controlling output voltage whenever the VCO output is incorrect. The dc correcting voltage forms the basis of the automatic phase-correction (APC) loop, whose output is applied to a voltage-variable capacitance, which in turn pulls the VCO into line. The APC loop, or phase-locked loop can be quite similar to an automatic frequency correction (AFC) loop. The overall arrangement is known variously as a stabilized master oscillator (SMO), a phase-lock synthesizer or even as a VCO synthesizer with a phase-locked loop.

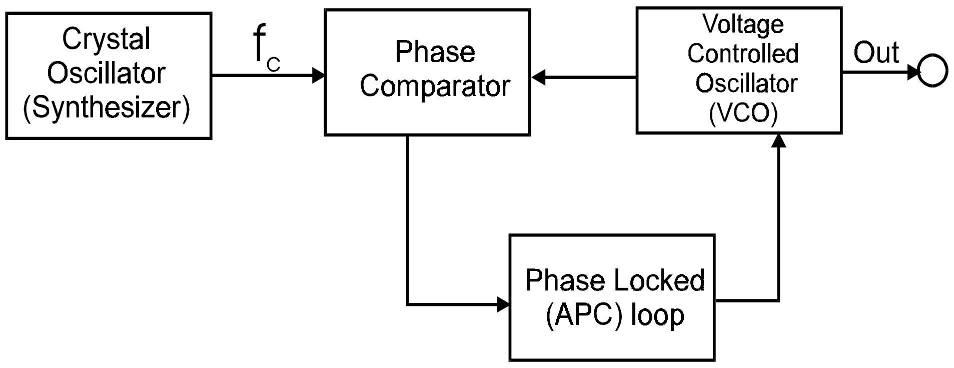

Crystal Oscillator (Synthesizer)
f_c
Phase Comparator
Voltage Controlled Oscillator (VCO)
Out
Phase Locked (APC) loop

Fig. 7.5. Basic Block Diagram of Indirect Synthesizer

7.8 INTERMEDIATE-FREQUENCY AMPLIFIERS :

The IF amplifier is a fixed-frequency amplifier, with the very important function of rejecting adjacent unwanted frequencies. It should thus have a frequency response with steep skirts. When the desire for a flat-topped response is added, the resulting recipe is for a double-tuned or stagger-tuned amplifier. Whereas FET and integrated circuit IF amplifiers generally are (and vacuum-tube ones always were) double-tuned at the input and at the output, bipolar transistor amplifiers often are single-tuned. This departure from a single-stage, double-tuned amplifier is for the sake of extra gain, and hence receiver sensitivity.

Review Questions

1. Explain the Amplitude Modulation.
2. How Pre Emphasis and De Emphasis improve the noise immunity?
3. Compare the Amplitude, Frequency and Phase Modulation.

NOTES

NOTES

NOTES

BIBLIOGRAPHY

1. **Microprocessors and Interfacing : Programming and Hardware**

   - Douglas Hall

   - Publication - MC Graw Hill

2. **CEDT**

   Module - 2

3. **An Introduction to Analog & Digital Communications**

   - Simon Haykin

4. **Digital & Analog Communication Systems**

   - K.Samshanmugan