ACE Engineering Academy

Hyderabad | Delhi | Bhopal |Pune | Bhubaneswar | Bengaluru |Lucknow | Patna | Chennai | Vijayawada | Visakhapatnam | Tirupati | Guntur | Kukatpally (Hyd)

Digital Circuits & Microprocessors

Volume – 1: Study Material with Classroom Practice Questions

GATE

Electronics & Communication Engineering

ACE is the leading institute for coaching in IES, GATE & PSUs

H O: 204, II Floor, Rahman Plaza, Opp. Methodist School, Abids, Hyderabad-500001,

Ph: 040-23234418, 040-23234419, 040-23234420, 040 - 24750437

CONSISTENTLY TOP RANKS IN IES
All INDIA 1ST RANK 32 TIMES IN GATE

Foreword

Digital Circuits & Microprocessors for GATE

The Study material for Digital Circuits & Microprocessors is thoroughly revised this year to meet the requirements of all categories of students. It consists of

Volume - I: Study Material with classroom Practice Questions

- Brief synopsis covering the entire syllabus
- Worked out examples, concept-wise
- Classroom Practice Questions (Covering all essential Concepts)

The students are advised to go through the theory duly understanding the worked out examples to the extent possible and bring the booklet to the class. By doing so, the students will get familiarized with some of the concepts and can really enjoy the lecture in the class. Also, the students can interact with the faculty in a better way.

Volume- II: Student practice booklet - Work Book: It has two levels

Level 1 : Basic Level

Level 2 : Advanced Level

After attending the classes and understanding the typical questions explained by the professors, the student will be in a position to solve these questions in work book. However, for verifying the correctness, Hints / Solutions are also given wherever required. The student is advised to struggle to get the solution for the problems in the work book by self analysis and not to refer the solution first.

Thanks to all Professors who extended their valuable services in the preparation of this Booklet.

It is believed that this volume is also a valuable aid to the students appearing for competitive exams like ESE, DRDO, ISRO, JTO, State service Commissions and other PSUs.

With best wishes to all the Students

Y.V. Gopala Krishna Murthy,

M Tech. MIE,

Managing Director,

ACE Engineering Academy

Contents

S.No.	Name of the Chapter	Page No.
	Digital Circuits	
01	Number Systems	01 – 14
02	Logic Gates & Boolean Algebra	15 – 26
03	K – Maps	27 – 33
04	Combinational Circuits	34 – 44
05	Sequential Circuits	45 – 69
06	Logic Gate Families	70 – 79
07	Semiconductor Memories	80 – 87
08	A/D and D/A Converters	88 – 93
	Microprocessors	
09	Architecture, Pinout of 8085 & Interfacing with 8085	94 – 102
10	Instruction set of 8085 & Programming with 8085	103 - 109

COACHING OFFERED

- IES
- GATE
- PSUs

- ➤ Class Room Coaching
- ➤ Postal Coaching
- ➤ Online Test Series
- ➤ Postal Test Series
- ➤ Interview Guidance

1 Number Systems

Definition:

The number systems are used to quantify the magnitude of something. One way of quantifying the magnitude of something is by proportional values. This is called analog representation. The other way of representation of any quantify is numerical (Digital).

The number systems are called position weighted systems, since the weight of each digit depends on its relative position with in the number.

Base (or) radix of system:

The base (or) radix of number system is defined as the number of different symbols (Digits (or) characters) used in that number system.

Key points:

- If the Base of the number system is 'r', the no. of different symbols used in the system is 'r' (the different symbols are '0 to r-1').
- The largest value of digits in base 'r' system is 'r-1'.

Types of Number of systems:

1. **Binary number system:**

The base value of binary number system '2'. (i.e. r = 2).

(i) The different symbols used in Binary number system are 0 to r-1. (i.e. 0 to 1)
Those are 0,1.

(ii) The maximum digit in binary system
= r-1
= 2-1
= 1

2. **Octal Number system:**

The base value of octal number system is '8'. (i.e. r = 8)

(i) The number of different symbols used in octal system are 0 to r-1. (i.e. 0 to 7)
Those are 0,1,2,3,4,5,6,7.

(ii) The maximum digit in octal number system = r-1
= 8-1
= 7.

3. **Decimal number system:**

The base value of Decimal number system is '10'. (i.e. r = 10)

(i) The number of different symbols used in decimal system are 0 to r -1. (i.e. 0 to 9)
Those are 0,1,2,3,4,5,6,7,8,9.

(ii) The maximum digit in Decimal system
= r-1
= 10-1
= 9.

4. **Hexa decimal number system:**

The base value of Hexadecimal Number system is '16'. (i.e. r = 16)

(i) The number of different symbols used in Hexadecimal number system are 0 to r -1. (i.e. 0 to 15)
Those are 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

(ii) The maximum digit in Hexadecimal system = r-1
=16-1
=15(F).

Key points:

Any positional Number system can be expressed as sum of products of place value and digit value.

The place (or) weights of different digits in mixed number systems is

(i) Decimal Number system:

$\ldots 10^4\ 10^3\ 10^2\ 10^1\ 10^0\ .\ 10^{-1}\ 10^{-2}\ 10^{-3}\ \ldots$

↑ Decimal Point

(ii) Binary number system:

$\ldots 2^4\ 2^3\ 2^2\ 2^1\ 2^0\ .\ 2^{-1}\ 2^{-2}\ 2^{-3}\ \ldots$

(iii) Octal Number system :

$\ldots 8^4\ 8^3\ 8^2\ 8^1 8^0\ .\ 8^{-1}\ 8^{-2}\ 8^{-3}\ \ldots$

Examples

01. $(256.12)_{10} =$

$2\times10^2 + 5\times10^1 + 6\times10^0 + 1\times10^{-1} + 2\times10^{-2}$

02. $(346.71)_8 =$

$3\times8^2 + 4\times8^1 + 6\times8^0 + 7\times8^{-1} + 1\times8^{-2}$

03. $(10111.101)_2 =$

$1\times2^4 + 0\times2^3 + 1\times2^2 + 1\times2^1 + 1\times2^0 +$

$1\times2^{-1} + 0\times2^{-2} + 1\times2^{-3}.$

04. $(A4E2.B3)_{16} =$

$A\times16^3 + 4\times16^2 + E\times16^1 + 2\times16^0 + B\times16^{-1} + 3\times16^{-2}$

$10\times16^3 + 4\times16^2 + 14\times16^1 + 2\times16^0 + 11\times16^{-1} + 3\times16^{-2}.$

Number systems conversion

Decimal to binary conversion:

A) Integer Numbers:

Divide the given decimal number repeatedly by '2' and collect the remainders. This must continue until the integer quotient becomes zero.

Example: $(52)_{10}$

Successive division by '2'

		Remainder
2	52	
2	26	----------0
2	13	----------0
2	6	----------1
2	3	----------0
2	1	----------1
	0	----------1

$(52)_{10} = (110100)_2.$

Key points:

The conversion from decimal Number to any 'base-r' system is similar to the above example except the division is done by 'r' instead of '2'.

B) Fractional Numbers:

First, the given fraction number multiplied by '2' to give an integer and a fraction, the new fraction is multiplied by '2' to give a new integer and a new fraction. This process continued until the fraction becomes zero (or) until the number of digits has sufficient accuracy.

Example: $(0.75)_{10}$

Given fraction	0.75	integer
Multiply 0.75 by 2	1.50	1
Multiply 0.50 by 2	1.00	1

$(0.75)_{10} = (.11)_2$

Mixed decimal:

$(163.875)_{10}$

$(163)_{10} = (10100011)_2$

$(0.875)_{10} = (.111)_2$

$(163.875)_{10} = (10100011.111)_2$

Binary to decimal conversion:

Binary numbers converted to decimal equivalent by the positional weights methods. In this method each binary digit of the number is multiplied by its position weight and the product terms are added to obtain the decimal number.

Example:

Integer: $(10101)_2$

Positional weights $2^4\ 2^3\ 2^2\ 2^1\ 2^0$

$= (1\times2^4)+(0\times2^3)+(1\times2^2)+(0\times2^1)+(1\times2^0)$

Binary number *1 0 1 0 1*

$=16+4+1 = (21)_{10}$.

Fraction: $(.101)_2$

Positional weights $2^{-1}\ 2^{-2}\ 2^{-3}$

$=(1\times2^{-1})+(0\times2^{-2})+(1\times2^{-3})$

$= 0.625$

Mixed: $(11011.101)_2$

$(11011)_2 = (27)_{10}$

$(.101)_2 = (.625)_{10}$

$(11011.101)_2 = (27.625)_{10}$.

Key points:

The conversion from any base 'r' system to decimal system is same as above process except the corresponding base value 'r' is replaced at '2'.

Binary to octal:

The base value of binary is '2' base value of octal is '8' = 2^3. Represent each octal digit with equivalent three digit binary number.

Octal	Binary equivalent of Octal
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Conversion process:

First write the given binary number, start from LSB group three bits, and continue process until MSB reached. For the last grouping if sufficient number of bits not available, pad zeros to left side if it integer and to the right if it fractional and write equivalent octal number for each grouping.

Example:

01. $(1010101010)_2$

Step 1: Write the given binary number

Step 2: Group three bits

$\overleftarrow{001\ \ 010\ \ 101\ \ 010} = (1252)_8$

1 2 5 2

02. $(.10010110)_2$

$\overrightarrow{.100\ \ 101\ \ 100} = (.454)_8$

4 5 4

03. $(1010101010.10010110)_2 = (1252.454)_8$

Key points:

Octal to binary is reverse process. Represent each octal digit with three binary bits and combine them.

Binary to Hexa decimal:

Binary base value =2, Hexadecimal base value = 16. Each hexa digit represented by 4 binary bits.(i.e.$16 = 2^4$)

Hexa decimal Digit	Binary equivalent
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Conversion process:

Start from LSB, group four bits, if sufficient number of bits are not available, Pad zeros to left side if it integer and to right side if it fractional. Write equivalent Hexa decimal Number for each group.

Example:

01. $(1001110101101)_2$

$\overleftarrow{0001\ \ 0011\ \ 1010\ \ 1101} = (13AD)_{16}$

1 3 A D

02. $(.10101101)_2$

$\overrightarrow{.1010\ \ 1101} = (.AD)_{16}$

A D

03. $(1001110101101.10101101)_2 = (13AD.AD)_{16}$

Key points:

- The hexadecimal to binary conversion is reverse process.
- Represent each hexa digit with four binary bits and combine them.

Octal to hexadecimal:

Step 1: Convert octal to binary
Step 2: Convert binary to Hexadecimal

Ex: $(74213)_8$

Step 1: Convert octal to binary

7 4 2 1 3

111 100 010 001 011

$(111100010001011)_2$

Step 2: Convert binary to hexa:
(By grouping four bits from LSB)

$\overleftarrow{0111\ \ 1000\ \ 1000\ \ 1011}$

7 8 8 B

$= (788B)_{16}$

Hexa decimal to octal:

Step 1: Hexadecimal to binary.
Step 2: Binary to octal

Ex: $(AFB2)_{16}$

Step 1: A F B 2

1010 1111 1011 0010

Step 2:

001 010 111 110 110 010 $= (127662)_8$

1 2 7 6 6 2

Examples

01. $\sqrt{(144)_r} = (12)_r$ *find the possible values*

(a) 8 (b) 9 (c) 10 (d) ≥ 5

Sol: $\sqrt{144_r} = 12_r$

$r^2 + 4r + 4 = r^2 + 4r + 4 = 0$

Then any base is applicable which is greater than or equal to 5

02. Convert $(105.15)_{10}$ to binary

Integer 105

2	105	
2	52	——— 1
2	26	——— 0
2	13	——— 0
2	6	——— 1
2	3	——— 0
2	1	——— 1
	0	——— 1

Integer equivalent = $(1101001)_2$

Fraction: 0.15

$0.15 \times 2 = 0.30 - - - - 0$
$0.3 \times 2 = 0.6 - - - - - 0$
$0.6 \times 2 = 1.2 - - - - - 1$
$0.2 \times 2 = 0.4 - - - - - 0$
$0.4 \times 2 = 0.8 - - - - - 0$
$0.8 \times 2 = 1.6 - - - - - 1$
$0.6 \times 2 = 1.2 - - - - - 1$

Fraction equivalent = $(.0010011)_2$

$(105.15)_{10} = (1101001.0010011)_2$.

03. Convert decimal to octal $(378.93)_{10}$

8	378	
8	47	——— 2
8	5	——— 7
	0	——— 5

$(572)_8$

$0.93 \times 8 = 7.44 - - - - 7$
$0.44 \times 8 = 3.52 - - - - 3$
$0.52 \times 8 = 4.16 - - - - 4$
$0.16 \times 8 = 1.28 - - - - 1$

$(.7341)_8$

$(378.93)_{10} = (572.7341)_8$

04. Convert decimal to Hexadecimal $(2598.675)_{10}$

16	2598	
16	162	——— 6
16	10	——— 2
	0	——— 10

$(A26)_{16}$

$0.675 \times 16 = 10.8 - - - 10\,(A)$
$0.8 \times 16 = 12.8 - - - - 12\,(C)$
$0.8 \times 16 = 12.8 - - - - 12\,(C)$
$0.8 \times 16 = 12.8 - - - - 12\,(C)$

$(ACCC)_{16}$

$(2598.675)_{10} = (A26.ACCC)_{16}$

05. Convert $(11011.101)_2$ to decimal.

1 1 0 1 1 1 0 1
2^4 2^3 2^2 2^1 2^0 2^{-1} 2^{-2} 2^{-3}

$(1\times2^4)+(1\times2^3)+(1\times2^1)+(1\times2^0)+(1\times2^{-1})+(0\times2^{-2})+(1\times2^{-3})$

$= 16+8+0+2+1+0.5+0+0.125$

$= (27.625)_{10}$

06. ***Convert $(4057.06)_8$ to decimal.***

4 0 5 7 . 0 6

$8^3\ 8^2\ 8^1\ 8^0\quad 8^{-1}\ 8^{-2}$

$= (4\times8^3)+(0\times8^2)+(5\times8)+$
$(7\times8^0)+(0\times8^{-1})+(6\times8^{-2})$

$= 2048+0+40+7+0+0.09375$

$= (2095.09375)_{10}$

07. ***Convert $(A0F9.0EB)_{16}$ to decimal***

A 0 F 9 . 0 E B

$16^3\ 16^2\ 16^1\ 16^0\ 16^{-1}\ 16^{-2}\ 16^{-3}$

$= (10\times16^3)+(0\times16^2)+(15\times16^1)+(9\times16^0)$
$+(0\times16^{-1})(14\times16^{-2})+(11\times16^{-3})$

$= 40960 + 0 + 240 + 9 + 0 + 0.0546 + 0.0026$

$= (41209.0572)_{10}$.

08. ***Convert $(B9F.AE)_{16}$ to Octal.***

Given Hex number

B 9 F . A E

Convert each hex to binary

1011 1001 1111 1010 1110

Group of three bits

101 110 011 111 . 101 011 100

Convert each three bits group to octal

5 6 3 7 . 5 3 4 $= (5637.534)_8$

1. Binary Addition:

Rules for binary addition:

0+0=0, 0+1=1, 1+0=1, 1+1=10,
i.e. 0 with a carry '1'
1+1+1=11, i.e. 1 with a carry '1'.

Example: Add $(1101.101)_2$, $(111.011)_2$

1101.101
111.011

10101 .000
$= (10101.000)_2$

2. Binary subtraction:

Rules for binary subtraction:

0-0=0, 1-1=0, 1-0=1, 0-1=1, with borrow '1'

Example: Subtract $(111.111)_2$ from $(1010.01)_2$

1010.010
111.111

$(0010.011)_2$.

3. Binary multiplication:

Rules for binary multiplication:

$0\times0=0,\ 0\times1=0,\ 1\times0=0,\ 1\times1=1.$

Example:

01. Multiply $(1101)_2$ by $(110)_2$

1101
× 110
0000
1101
1101
1001110.
$= (1001110)_2$

02. Multiply $(1011.101)_2$ by $(101.01)_2$

1011.101
×101.01
1011101
0000000
1011101
0000000
1011101
111101.00001
$= (111101.00001)_2$

4. Binary division:

The binary division is done by long division method, similar to decimal.

Example: Divide $(101101)_2$ by $(110)_2$

Sol: Divisor 110, can not go in the first three bits of the dividend, i.e. in 101. So, consider the first 4 bits 1011 of dividend. 110 can go in 1011, one time a remainder of 101. Next, 110 can go in 1010, one time with a remainder of 100. Next 110 can go in 1001 one time with a remainder of '11'.Finally 110 can go in 110 with remainder of '0'.

```
110)101101(111.1
    110
   ----
    1010
     110
   ----
     1001
      110
   -----
      110
      110
   -----
      000
```

$= (111.1)_2$

Representation of signed numbers:

The representation of signed numbers is in two ways

- Sign-magnitude form
- Complement form

Two types of complement forms

- 1's complement form
- 2's complement form.

Note:

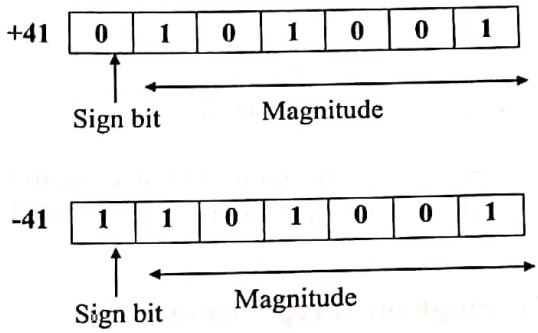
The advantage of performing subtraction by the complement method is reduction in the hardware. Instead of having separate digital circuits for addition, subtraction, only adding circuits are sufficient.

Sign magnitude form:

In sign magnitude form, an additional bit called the sign bit is placed in front of the number. If sign bit is zero number is positive, if sign bit is '1', number is negative.

Zero has two representations in this form.

Example:



Representation using complement methods:

1. 1's complement representation:

1. If the number is positive, the magnitude is represented in its true binary form, and a sign bit '0' placed in front of MSB.
2. If the number is negative, the magnitude is represented in its one's complement form and a sign bit '1' placed in front of MSB.

How to obtain 1's complement:

Invert the bits (1 to 0, 0 to 1) in the true binary representation to obtain 1's complement of number.

Zero has two representations in this form.

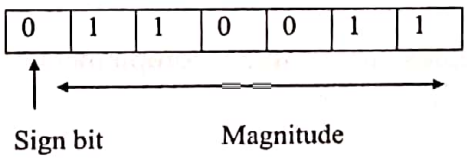
Example:

01. $(10101)_2$

1's complement is $(01010)_2$.

02. Represent +51 in 1's complement form.

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

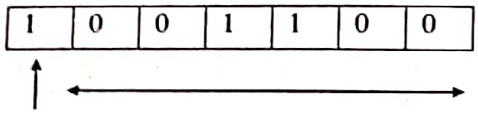


03. Represent -51 in 1's complement form.

$(+51)_{10} = (0110011)_2$

-51 = 1's complement of (+51)

= 1001100



Sign bit =1(Because negative number)
Magnitude in 1's complement form.

2. 2's complement representation:

1. If the given number is positive, the magnitude is represented in its true binary form and sign bit '0' is placed in front of MSB.
2. If the given number is negative, the magnitude is represented in its 2's complement form and a sign bit '1' is placed in front of MSB.

How to obtain 2's complement:

2's complement = 1's complement + 1

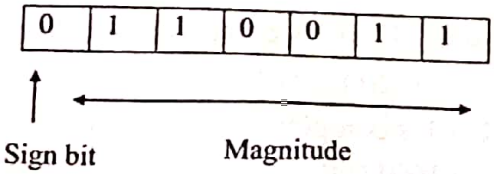
Example:

01. Find 2's complement of $(110110)_2$

	1	1	0	1	1	0
	↓	↓	↓	↓	↓	↓
1's complement	0	0	1	0	0	1
						+1
	0	0	1	0	1	0

$= (001010)_2$

02. Represent +51 in 2's complement form
$(51)_{10} = (110011)_2$

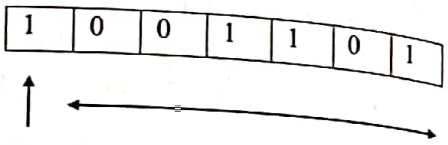


03. Represent -51 in 2's complement form.

$(+51)_{10} = (0110011)_2$

-51 = 2's complement of (+51)

2's complement of $(0110011)_2$ is $(1001101)_2$



Decimal	Sign magnitude	Sign 1's complement	Sign 2's complement
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
-0	1000	1111	----
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	-----	-----	1000

Key points in 2's complement form:

1) There is one unique zero.
2) Two's complement of zero is zero.
3) For an n-bit word which includes sign bit there are $(2^{n-1}-1)$ positive integers, 2^{n-1} negative integers and one '0' for total of 2^n unique states.

Range of numbers represented in 2's complement form is $-(2^{n-1}) \rightarrow +(2^{n-1}-1)$

Example

01. ***Represent - 45 in 8-bit 2's complement form.***

Sol:

+45 in 8-bit form is 00101101

Obtain 1's complement of 00101101 and then add 1.

0	0	1	0	1	1	0	1
↓	↓	↓	↓	↓	↓	↓	↓
1	1	0	1	0	0	1	0
							+1
1	1	0	1	0	0	1	1

2's complement of $(-45)_{10}$ is $(11010011)_2$

02. ***Represent - 73.75 in 12 bit 2's complement form.***

Sol.

+ 73.75 in binary form is	01001001.1100
1's complement	10110110.0011
	+1
2's complement.	10110110.0100

03. ***Which of the following number will have same value in both sign magnitude notation and 2's complement notation?***

(a) 1000 0000 (b) 1100 0000

(c) 1110 0000 (d) 1111 0000

Ans: (b)

Sol:

2's complement	Sign-Magnitude
-128	-0 ----- (a)
-64	-64 ---- (b)
-32	-96 ---- (c)
-16	-112 --- (d)

04. ***Let dollar denote 2's complement operation then for the signed number x, $(x) = x. is satisfied by which of the value other than zero. Let the length of register is 8 bit.***

Sol: $2's\ comp(x) = x$

$8\ bit\ number = 10000000 = (-128)_{10}$

$1's\ complement \rightarrow 01111111$

$+ \quad 1$

$2's\ complement \rightarrow 10000000$

If, $x \neq 0$

$x \neq 128$

$(\$(\$(\$(-x)))) = (\$(\$(+x)))$

$= \$(-x)$

$= +x$

Some problems on sign-magnitude, 1's complement, 2's complement form:

1. Give a signed binary number. Determine the decimal value in each case, if they are in
   (i) Sign magnitude form
   (ii) 2's complement form
   (iii) 1's complement form
   (a) 01101 (b) 010111
   (c) 10111 (d) 1101010

Given number	Sign magnitude form	2's complement from	1's complement form
01101	+13	+13	+13
010111	+23	+23	+23
10111	-7	-9	-8
1101010	-42	-22	-21

2's complement arithmetic:

The 2's complement system is used to represent negative numbers using modulus arithmetic. The word length of a computer is fixed. That means if a 4 bit number is added to another 4 bit the result will be only 4-bits. Carry if any from the fourth bit will overflow. This called the modulus arithmetic.

Example: 1100 + 1111 = 1011

Procedure for 2's complement subtraction:

1. Add the 2's complement of the subtrahend to the minuend.
2. If there is a carry out, ignore it. Look at the sign bit i.e. MSB of the sum term. If MSB is a '0' the result is positive, and is in true binary form.
3. If MSB is '1' the result is negative and is in its 2's complement form and take its 2's complement form. Take its 2's complement to find magnitude in binary.

Example:

01. Subtract 14 from 46 using 8-bit 2'scomplement form.

46-14=?

+14 = 00001110
-14 = 11110010 (2's complement form)

+46 = 00101110
-14 = 11110010 (2's complement of -14)

(1) 00100000 (Ignore the carry)

There is a carry, ignore it, the MSB is zero, so the result is positive and is in normal binary form. The result is 00100000 = +32.

02. Add -75 to +26 using the 8-bit 2's complement arithmetic.

+75 = 01001011
-75 = 10110101 (2's complement)

+26 = 00011010
-75 = 10110101 (2's complement of -75)

11001111 (No carry)

There is no carry, the MSB is '1', so, and the result is negative and is in 2's complement form.

2's complement of 11001111 is 00110001=49

Result is - 49.

03. Add 27.125 to -79.625 using 12 bit 2's complement arithmetic

+27.125 = 00011011.0010
-79.625 = 10110000.0110 (2's complement of -79.625)

11001011.1000 (No carry)

MSB is 1 so negative number, two's complement of 11001011.1000 is 00110100.1000 = – 52.5

One's complement arithmetic:

In 1's complement subtraction, add the 1's complement of the subtrahend to the minuend. If there is a carry out, bring the carry around and add it to the LSB. This is called end around carry. Look at the sign bit (MSB), if this is '0' result is positive, and it is in true binary form. If the MSB is a '1' (whether there is a carry (or) no carry at all) the result is negative and is in 1's complement form. Take its 1's complement to get magnitude in binary.

Examples

01. ***Represent -99 and -77.25 in 8-bit 1's complement form.***

Sol:

a) +99 = 01100011
-99 = 10011100 (1's complement form)

b) +77.25 = 01001101.0100
-77.25 = 10110010.1011
(1's complement form)

02. ***Subtract 14 from 25 using 8-bit 1's complement arithmetic.***

Sol: 25 – 14
25 - 00011001
-14 -11110001 (1's complement form)

(1)00001010
+1 (End around carry)

00001011 result.

The MSB is zero, so, the result is positive and is in pure binary.

The result is = 00001011 = $(+11)_{10}$.

03. ***Add -25 to 14 using 8-bit 1's complement form method.***

Sol: +14 - 00001110
-25 - 11100110 (1's complement form)
11110100 (No carry)

There is no carry. The MSB is a 1. So, the result is negative and is in its 1's complement form. The 1's complement of 11110100 is 00001011.

The result is $(-11)_{10}$.

04. ***Add -25 to -14 using 8-bit 1's complement form method.***

Sol:

-25 = 11100110 (1's complement form)
-14 = 11110001 (1's complement form)

(1) 11010111
+1 (End around carry)
11011000

The MSB is '1'. So, the result is negative and is in its 1's complement form. The 1's complement of 11011000 is 00100111. Therefore the result is -39.

05. ***Add -89.75 to 43.25 using 12-bit 1's complement method.***

Sol:

+43.25 = 00101011.0100
-89.75 = 10100110.0011 (1's complement form)

11010001.0111

There is no carry. The MSB is 1. So, the result is negative and is in its 1's complement form.

The 1's complement of 11010001.0111 is 00101110.1000. Therefore the result is - 46.50.

Classification of Binary codes:

1. Numeric and Alphanumeric codes:

Numeric codes are codes which represent numeric information i.e only numbers as a series of 0's and 1's.

Example: 8421, excess-3, Gray code are numeric codes.

Numeric codes used to represent the decimal digits are called Binary coded decimal (BCD) code.

Example: 8421, 2421, 5421 are BCD codes.

Alpha numeric codes are codes which represent alphanumeric information, i.e. letters of the alphabet and decimal numbers as a sequence of 0's and 1's.

Example: ASCII code, EBCDIC code.

2. Weighted and Non-weighted codes:

The BCD codes may be weighted codes (or) non-weighted codes. The weighted codes are those which obey the position-weighting principle. Each position of the number represents a specific weight.

Example: 8421, 2421 are weighted codes.

Non weighted codes are codes which are not assigned with any weight to each digit position, i.e. each digit position with in the number is not assigned fixed value.

Example: Excesss-3 code, gray code.

3. Self complementing codes:

A code is said to be self-complementing, if the code word of the 9's complement of N, i.e of '9-N' can be obtained from the code word of N by interchanging all the 0's and 1's. Therefore in a self-complementing code, the code for 9 is the complement of the code for 0, the code for 8 is the complement of the code for 1, and so on.

Example: 2421, 5211, 642-3, 84-2-1, Excess-3 self complementing codes.

5421 code is not self complementing code.

Advantage:

Self complementing codes have an advantage that their logical complement is the same as the arithmetic complement. For a code to be self complementing, the sum of all weights must be 9.

4. Cyclic codes:

Cyclic codes are those in which each successive code word differs from the preceding one in only one bit position. They are also called unit distance codes The advantage of unit distance codes have special advantage that they minimize transitional errors or flashing.

Ex: Gray code.

5. BCD code (8421 BCD code (or) Natural BCD code):

The 8421 BCD code is simply referred as BCD code. In this each decimal digit from 0 to 9 is called as 4-bit binary number. It is also called the natural binary code, weighted code, sequential code. The main advantage of this code is its ease of conversion to and from decimal. It is less efficient than binary, in the sense that it requires more bits.

Decimal	Binary	BCD
0	0000	0000
1	0001	0001
2	0010	0010
3	0011	0011
4	0100	0100
5	0101	0101
6	0110	0110
7	0111	0111
8	1000	1000
9	1001	1001
10	1010	00010000
11	1011	00010001
12	1100	00010010
13	1101	00010011
14	1110	00010100
15	1111	00010101.

6. The Excess-3 codes:

It is a non weighted BCD code. These codes derive its name from the fact that each binary code word is the corresponding 8421 code word plus 3(0011). It is a sequential code, self complementing code.

It has invalid states: 0000, 0001, 0010, 1101, 1110, 1111.

Decimal	BCD	Excess-3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100
10	00010000	01000011
11	00010001	01000100
12	00010010	01000101
13	00010011	01000110

7. Gray code (reflective code) :

It is a non-weighted code. These codes are used in instrumentation, data acquisition systems where linear (or) angular displacement is measured, shaft encoders, I/O devices, A/D converters.

Binary to Gray conversions:

Let an n-bit binary number $B_n B_{n-1}$ -------- B_1 and its Gray code equivalent is $G_n G_{n-1}$, G_{n-2},.....G_2, G_1, Where B_n and G_n are MSB bits. The Gray code obtained from binary is $G_n = B_n$, $G_{n-1} = B_n \oplus B_{n-1}$, $G_1 = B_2 \oplus B_1$.

Example: ***Convert $(1001)_2$ to Gray code.***

Binary 1 → ⊕ → 0 → ⊕ → 0 → ⊕ → 1

Gray 1 1 0 1

Gray code - 1101

Gray to Binary:

In an n-bit Gray code $G_n G_{n-1}$---------G_1, and its binary equivalent by B_n , B_{n-1} -----B_1.

$B_n = G_n$, $B_{n-1} = B_n \oplus G_{n-1}$, $B_1 = B_2 \oplus G_1$.

Example: Convert Gray code 1101 to Binary.

Sol: Gray - 1 1 0 1

Binary - 1 0 0 1

Binary - 1001.

8. Error – detecting codes:

When binary data is transmitted and processed, it is susceptible to noise that can alter (or) distort its contents. The 1's may be get changed to 0's and 0's to 1's. Including a parity bit at transmitter side, can detect a single bit error.

Parity:

The simplest technique for detecting errors is that of adding an extra bit is known as parity bit to each word being transmitted. There are two types of parity: odd parity, even parity.

Odd parity:

For odd parity the parity bit is set to a '0' (or) a '1' at the transmitter such that the total no. of 1 bits in the word including the parity bit is an odd number.

Even parity:

The total no. of 1's including parity must be even.

Class Room Practice Questions

01. Given $(135)_x + (144)_x = (323)_x$ What is the value of base x?
(a) 5 (b) 3
(c) 12 (d) 6

02. The number of 1's in 8-bits representation of –127 in 2's complement form is m and that in 1's complement form is n. What is the value of m:n?
(a) 2:1 (b) 1:2
(c) 3:1 (d) 1:3

03. A number in 4- bit two's complement representation is $X_3 X_2 X_1 X_0$. This number when stored using 8 - bits will be
(a) 0000 $X_3 X_2 X_1 X_0$.
(b) 1111 $X_3 X_2 X_1 X_0$.
(c) $X_3 X_3 X_3 X_3 X_3 X_2 X_1 X_0$
(d) $\overline{X}_3 \overline{X}_3 \overline{X}_3 \overline{X}_3 X_3 X_2 X_1 X_0$

04. The 2's complement representation of $(-539)_{10}$ in hexadecimal is
(a) ABE (b) DBC
(c) DE5 (d) 9E7

05. The base (or radix) of the number system such that the following equation holds is_____ $\frac{312}{20} = 13.1$

06. Consider the equation $(123)_5 = (x8)_y$ with x and y as unknown. The number of possible solutions is _____.

07. Let X be the number of distinct 16-bit integers in 2's complement representation. Let Y be the number of distinct 16-bit integers in sign magnitude representation. Then X−Y is_____.

Key for CRPQ

01. (d)	02. (a)	03. (c)	04. (c)	05. 5
06. 3	07. 1			

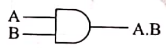
2 Logic Gates & Boolean Algebra

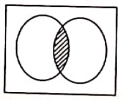
Logic gate:
It is the fundamental building block of digital system. The name logic gate is derived, because the output and input patterns of a gate are assigned logically.
There are three basic gates: AND, OR, NOT.

Logic design:
- The inter connection of gates to perform a variety of logical operations is called logic design.
- The inputs and outputs of logic gates can occur only in two levels. These two levels are termed as High and Low (or) simply 1&0.
- The truth table shows how the logic circuits output responds to various combinations of logic levels of input.

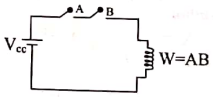
AND Gate:
- It has two or more input but only one output.
- The output of the AND gate is '1' only if all the inputs are 1's. If any one of the input is zero, the output is zero.

Logic symbol





AND is equivalent to A ∩ B



The bulb glows when both A and B are close

The logic symbol of AND operation is " . "

Truth table:

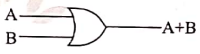
Inputs		Output
A	B	A. B
0	0	0
0	1	0
1	0	0
1	1	1

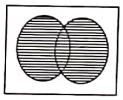
Note: The IC 7408 contains *four* two input AND gates, IC 7411 contain *three* three input AND gates, IC 7421 contains *two* four input AND gates.

OR gate:

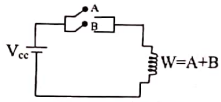
OR gate may have two or more inputs but only one output. The output of the OR gate is '1' if any of the input is '1'. The output is zero if all the inputs are zeros. Symbol for OR gate is '+'.

Logic symbol





OR is equivalent to A ∪ B



The bulb glows when One of the switch among A and B are closed

The logic symbol of OR operation is '+'

Truth table

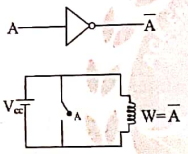
Inputs		Output
A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

Note: IC 7432 contains four two input OR gates.

NOT gate:

A NOT gate is also called inverter, has only one input and only one output. It is a device whose output is always the complement of the input. Symbol for NOT operation is ' ¯ '(bar).

Logic symbol



Truth table

Input	Output
A	X = $\overline{A}$
0	1
1	0

Note: The IC 7404 contains six inverters.

Basic gates:

- AND, OR, NOT are basic gates. Any logic circuit of any complexity can be realized by using only the three basic gates.

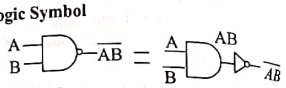
Universal gates:

- The NAND and NOR gates are called universal building gates. Both NAND and NOR gates can perform all the three basic logic functions.

NAND gate:

- NAND = AND output is 'NOT', 'NAND gate is combination of an AND gate and a NOT gate.

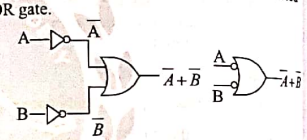
Logic Symbol



Truth table

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

The NAND gate is also equivalent to bubbled OR gate.



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

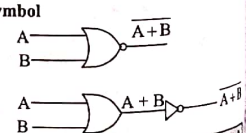
From the above truth table $\overline{AB}=\overline{A}+\overline{B}$

NAND = Bubbled OR

NOR gate:

NOR gate is a combination of OR gate and a NOT gate.

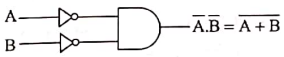
Logic Symbol



Truth table

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

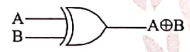
NOR gate is equivalent to bubbled AND



The Ex-OR gate:

The Ex – OR gate produces an output 1 only when the inputs are not equal, it is called an anti – coincidence gate (or) inequality detector. The output of an X – OR gate is the modulo sum of its two inputs. The name Exclusive – OR is derived from the fact that its output is a 1, only when exclusively one of its inputs is a 1 (it excludes the condition when both inputs are 1). The symbol of operation is '⊕'.

Logic symbol



Truth table

A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

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

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

Properties of X – OR gate:

(1) $A \oplus 1 = \overline{A}$

Proof: $A \oplus 1 = A.\overline{1} + \overline{A}.1 = \overline{A}$

(2) $A \oplus 0 = A$

Proof: $A \oplus 0 = A.\overline{0} + \overline{A}.0 = A.1 = A.$

(3) $A \oplus A = 0$

Proof: $A\,\overline{A} + \overline{A}.A = 0 + 0 = 0$

(4) $A \oplus \overline{A} = 1$

Proof:

$A.\overline{\overline{A}} + \overline{A}.\overline{A} = A.A + \overline{A}.\overline{A} = A + \overline{A} = 1$

(5) $AB \oplus AC = A\,(B \oplus C)$

Proof: $AB \oplus AC = AB.\overline{AC} + \overline{AB}.AC$

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

$= AB\overline{A} + AB\overline{C} + \overline{A}AC + \overline{B}AC$

$= 0 + AB\overline{C} + A\overline{B}C$

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

$= A\,(B \oplus C).$

(6) $A \oplus B = C$, then $A \oplus C = B$, $B \oplus C = A$, $A \oplus B \oplus C = 0.$

Proof: $A \oplus B \oplus C = (A \oplus B) \oplus (A \oplus B)$

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

$= 0 + 0 = 0$

Proof: $A \oplus C = A \oplus (A \oplus B)$

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

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

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

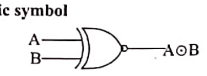
Ex-NOR gate:

Ex – NOR gate is a combination of Ex – OR gate and NOT gate. The output of Ex – NOR gate is '1' only when the two inputs are equal. The output is zero when two inputs are unequal. It is also called a coincidence gate. It can be used as an equality detector.

Symbol for X – NOR operation is ⊙

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

Logic symbol



Truth table

A	B	A⊙B
0	0	1
0	1	0
1	0	0
1	1	1

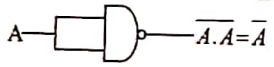
Enable input of a gate:

Which enables the other inputs to impress on the gate output, where as, disable input disable all other inputs to not impress on the gate output. i.e. that itself makes the final decision.

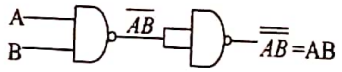
	Enable input	Disable input
NOT	-	-
AND	1	0
OR	0	1
NAND	1	0
NOR	0	1
EX – OR	0 (or) 1	-
EX NOR	0 (or) 1	-

Implementation All gates using NAND gate:

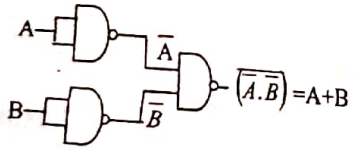
NOT gate:



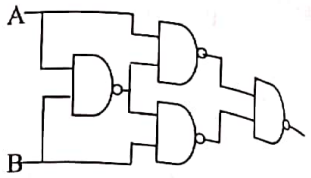
And gate:



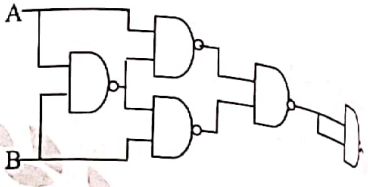
OR gate:



Ex-OR gate:



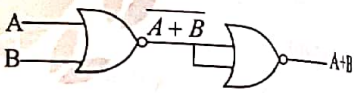
Ex-NOR gate:



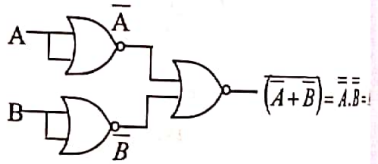
Implementation of All gates using NOR gate:

NOT gate:

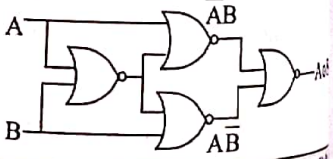
OR gate:



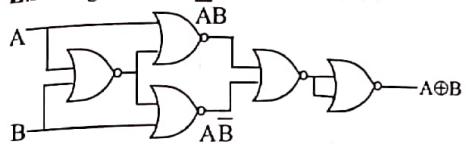
AND gate:



Ex-NOR gate:



Ex-OR gate:



Number of NAND, NOR gates required:

Gate to be realized	Min NAND required	Min NOR required
NOT	1	1
AND	2	3
OR	3	2
NAND	1	4
NOR	4	1
EX-OR	4	5
EX-NOR	5	4

Note: NAND gate most preferable than NOR gate, because it consumes less power.

BOOLEAN ALGEBRA:

Boolean algebra is a system of mathematical logic. It is an algebraic system consisting of the set of elements (0, 1), two binary operators called OR, AND, and one Unary operator NOT. It is the basic mathematical tool in the analysis and synthesis of switching circuits. It is a way to express logic functions algebraically.

Note: Any functional relation in Boolean algebra can be proved by the method of perfect induction. Perfect induction is a method of proof, where by a functional relation is verified for every possible combination of values that the values may assume.

Axioms and Laws of Boolean Algebra:

Axioms (or) postulates of Boolean algebra are a set of logical expressions that we accept without proof and upon which we can build a set of useful theorems.

	AND Operation	OR Operation	NOT Operation
Axiom 1:	0.0 = 0	0 + 0 = 0	$\overline{1}=0$
Axiom 2:	0.1 = 0	0 + 1 = 1	$\overline{0}=1$
Axiom 3:	1.0 = 0	1 + 0 = 1	
Axiom 4:	1.1 = 1	1 + 1 = 1	

Complementation Law:

Law 1: $\overline{0}=1$
2: $\overline{1}=0$
3: if A = 0, then $\overline{A}=1$
4: if A = 1, then $\overline{A}=0$
5: $\overline{\overline{A}}=A$. (Double inversion)

And Law:

Law1: A.0 = 0 (Null law)
2: A.1 = A (Identity law)
3: A.A =A
4: A. $\overline{A}$ = 0

OR Law:

Law1: A+0 =A (Null law)
2: A+1 =1 (Identity law)
3: A+A =A
4: A+ $\overline{A}$ =1

Commutative law:

Law 1: A+B = B+A
A+B+C = B+C+A = C+A+B
= B+A+C
Law 2: AB =BA
A.BC = B.CA = C.AB = B.AC

Violation: Inhibition (/): x/y ('x but not y')
= x $\overline{y}$ ⇒ It is not commutative x/y ≠ y/x

Associative Law:

Law1: (A+B) +C = A+ (B+C)
A+(B+C+D) = (A+B+C) +D
= (A+B) +(C+D)

Law 2: (A.B)C = A.(B.C)
A(BCD) = (ABC)D
A(BCD) = AB.CD

Violation: NAND and NOR gates are not Associative, but commutative.

Distribution Law:

Law1: A(B+C) = AB + AC
Law2: A+BC = (A+B) (A+C)

Redundant literal Rule:

1. $A + \bar{A}B = A + B$.
2. $A(\bar{A} + B) = AB$.

Idempotence law:

Law 1 : A.A = A
Law 2 : A + A = A

Absorption law: A + A.B = A.
A(A + B) = A.

Consensus theorem:
(Included Factor theorem).

Theorem 1: $AB + \bar{A}C + BC = AB + \bar{A}C$.

LHS
$= AB + \bar{A}C + BC = AB + \bar{A}C + BC(A + \bar{A})$
$= AB + \bar{A}C + BCA + \bar{A}BC$
$= AB(1 + C) + \bar{A}C(1 + B)$
$= AB + \bar{A}C$
$= AB + \bar{A}C$
= RHS

This theorem can be extended to any number of variables

$AB + \bar{A}C + BCD = AB + \bar{A}C$.

Proof: from 1, $AB + \bar{A}C + BCD$
$= AB + \bar{A}C + BC + BCD$
$= AB + \bar{A}C + BC(1 + D)$
$= AB + \bar{A}C + BC$
$= AB + \bar{A}C$.

Theorem 2:
$(A + B)(\bar{A} + C)(B + C) = (A + B)(\bar{A} + C)$.

Transposition Theorem:

$= AB + \bar{A}C = (A + C)(\bar{A} + B)$.

$RHS = (A + C)(\bar{A} + B) = A\bar{A} + AB + \bar{A}C + BC$
$= 0 + \bar{A}C + AB + CB = \bar{A}C + AB + BC(A + \bar{A})$
$= AB + ABC + \bar{A}C + \bar{A}BC$
$= AB(1 + C) + \bar{A}C(1 + B)$
$= AB + \bar{A}C$.

Demorgan's Theorem:

Law 1 : $\overline{A + B} = \bar{A}.\bar{B}$.

Similarly $\overline{A + B + C + D + \ldots} = \bar{A}.\bar{B}.\bar{C}.\bar{D}\ldots$

Law 2 : $\overline{AB} = \bar{A} + \bar{B}$.

Similarly $\overline{A.B.C.D\ldots} = \bar{A} + \bar{B} + \bar{C} + \bar{D} + \ldots$

Examples:

1. *Apply demorgans theorem to expression*

$f = \overline{\bar{A}B(CD + \bar{E}F)(\bar{A}\bar{B} + \bar{C}\bar{D})}$

Sol: The given expression is

$f = \overline{\bar{A}B(CD + \bar{E}F)(\bar{A}\bar{B} + \bar{C}\bar{D})}$

$= \overline{\bar{A}\bar{B}} + \overline{(CD + \bar{E}F)} + \overline{(\bar{A}\bar{B} + \bar{C}\bar{D})}$

$= AB + \overline{CD}.\overline{\bar{E}F} + \overline{\bar{A}\bar{B}}.\overline{\bar{C}\bar{D}}$

$= AB + (\bar{C} + \bar{D})(E + \bar{F}) + ABCD$.

2. *Reduce the expression* $f = \overline{\overline{AB} + \bar{A} + AB}$

$f = \overline{\overline{AB}}.\overline{\bar{A}}.\overline{AB}$
$= AB.A.\overline{AB}$
$= AB(\bar{A} + \bar{B}) = A\bar{A}B + AB\bar{B}$
$= 0$.

Duality:

- In positive logic system more positive of two voltage levels represented as '1' and more negative by '0'. In negative logic system more positive of two voltage levels represented as '0' and more negative by '1'.
- The distinction between positive logic & negative logic system is OR gate in the positive logic becomes an AND gate, vice versa.
- The duality theorem applies when we are changing one logic system to another.

  Duality theorem states that

  '0' becomes '1'
  '1' becomes '0'
  'AND' becomes 'OR'
  'OR' becomes 'AND'

- The variables are not complemented in this process.

Given expression

$\bar{0} = 1$
0.1 = 0
0.0 = 0
1.1 = 1

A.0 = 0
A .1 = A
A.A = A
$A.\bar{A} = 0$
A .B = B .A
A .(B + C) = AB + AC

Dual

$\bar{1} = 0$
1 + 0 = 1
1 + 1 = 1
0 + 0 = 0

A + 1 = 1
A + 0 = A
A + A = A
$A + \bar{A} = 1$
A + B = B + A
A + BC = (A + B) (A + C)

The dual of Ex – OR is equal to its complement

$A \oplus B = \bar{A}B + A\bar{B}$ Dual is $(\bar{A} + B)(A + \bar{B})$

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

Dual logic or Inversion logic or (Positive & Negative logic):

Signal status	+ve logic	-ve logic
L	L	H
H	H	L
NOT	NOT	NOT
AND	AND	OR
OR	OR	AND
NAND	NAND	NOR
NOR	NOR	NAND
EX-OR	EX-OR	EX-NOR
EX-NOR	EX-NOR	EX-OR

The one and only NOT gate cannot change its behavior with applied inverted logic.

Example:

AND truth table applied with

+ ve logic			- ve logic		
A	B	AB	A	B	AB
L	L	L	H	H	H
L	H	L	H	L	H
H	L	L	L	H	H
H	H	H	L	L	L

Complement of a function:
A simple procedure to find the complement of a function is to take the dual of the function and complement each literal.

Example:

1. $f = AB + \overline{A}\,\overline{B}$

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

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

Dual of $f = A\overline{B} + \overline{A}B$

2. $\overline{f} = \left(AB + \overline{A}\,\overline{B}\right)$

$= (\overline{AB})(\overline{A}\,\overline{B})$

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

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

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

Complement each literal $f = \overline{A}B + A\overline{B}$

Examples

Show that

(1) $f = A\left[B + \overline{C}\left(\overline{AB + A\overline{C}}\right)\right] = AB$

(2) $f = A + B\left[AC + (B+\overline{C})D\right] = A + BD$

(3) $f = (A+\overline{BC}).(A\overline{B} + ABC) = 0.$

(4) $f = A\overline{B}C + B + B\overline{D} + AB\overline{D} + \overline{A}C = B + C$

01.

Sol: $f = A\left[B + \overline{C}\left(\overline{AB + A\overline{C}}\right)\right]$

$= A\left[B + \overline{C}\left(\overline{AB}.\overline{A\overline{C}}\right)\right]$

$= A\left[B + \overline{C}(\overline{A}+\overline{B})(\overline{A}+C)\right]$

$= A\left[B + \overline{C}\left(\overline{A}\,\overline{A} + \overline{A}C + \overline{A}\,\overline{B} + \overline{B}\overline{C}\right)\right]$

$= A\left[B + \overline{A}\,\overline{C} + \overline{A}C\overline{C} + \overline{A}\,\overline{B}\,\overline{C} + \overline{B}C\overline{C}\right]$

$= A\left[B + \overline{A}\,\overline{C} + 0 + \overline{A}\,\overline{B}\,\overline{C} + 0\right]$

$= AB + A\overline{C}\overline{A} + \overline{A}\overline{B}\overline{C}A$

$= AB + 0 = AB$

02.

Sol: $f = A + B\left[AC + (B+\overline{C})D\right]$

$f = A + B\left[AC + BD + \overline{C}D\right]$

$= A + ABC + BD + B\overline{C}D$

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

$= A + BD$

03.

Sol: $f = \left(\overline{A + B\overline{C}}\right).(A\overline{B} + ABC)$

$f = \left(\overline{A}.\overline{B\overline{C}}\right)(A\overline{B} + ABC)$

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

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

$= 0 + 0$

$= 0.$

04.

Sol: $f = A\overline{B}C + B + B\overline{D} + AB\overline{D} + \overline{A}C$

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

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

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

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

$= \overline{A}C + B + \overline{B}C \qquad \because B + \overline{B}C = B + C$

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

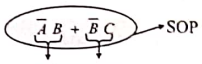
$= B + C(1 + \overline{A})$

$= B + C$

Boolean functions and their representation:
Sum of products form:

Also called Disjunctive normal form (DNF)

Example: $f(A,B,C) = \overline{A}B + \overline{B}C$

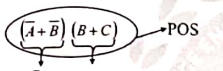


Product terms

Product of sums:

This also called conjunctive normal form (CNF)

Example: $f(A,B,C) = (\overline{A}+\overline{B})(B+C)$



Sum terms

Standard sum of product:

- It is also called canonical SOP form (or) Disjunctive canonical form (DCF). In this, the function is a sum of number of product terms where each product term contains all the variables of the function either in complemented (or) un complemented form.

Minterm:

- A product which contains all the variables of the function either in complemented (or) un complemented form called minterm.
- n - variable function have 2^n minterms. In minterm a variable represented as '1' complemented variable as '0'.

A	B	Minterm
0	0	$\overline{A}\,\overline{B} = m_0$
0	1	$\overline{A}B = m_1$
1	0	$A\overline{B} = m_2$
1	1	$AB = m_3$

Standard POS:

- Also called canonical POS form (or) conjunctive canonical form.
- Each term is a sum of all variables, a variable appeared in complemented (or) un complemented form.

Max term:

- It is a sum term contains all the variables in either complemented (or) un complemented form called max term.
- For a n- variable function, there are 2^n – max terms.
- In max term, a variable represented by '0', a complemented variable represented by '1'

A	B	Max term
0	0	$A + B = M_0$
0	1	$A + \overline{B} = M_1$
1	0	$\overline{A} + B = M_2$
1	1	$\overline{A} + \overline{B} = M_3$

Note:
For an '**n**' variable function,
the product or sum possibilities are 3^n-1
Min or maxterm possibilities are 2^n

Example: Take n = 2 variables (A & B)

Products = $3^2 - 1 = 8$(AB, $\overline{A}B$, $A\overline{B}$, $\overline{A}\,\overline{B}$, A.1, $\overline{A}$.1, B.1, $\overline{B}$.1)

Sums = $3^2 - 1 = 8$($A + B$, $\overline{A} + B$, $A + \overline{B}$, $\overline{A} + \overline{B}$, $A + 0$, $\overline{A} + 0$, $B + 0$, $\overline{B} + 0$)

min terms = $2^2 = 4$(AB, $\overline{A}B$, $A\overline{B}$, $\overline{A}\,\overline{B}$)

max terms = $2^2 = 4$($A + B$, $\overline{A} + B$, $A + \overline{B}$, $\overline{A} + \overline{B}$)

Max terms	A	B	C	Min terms
$A+B+C$	0	0	0	$\bar{A}\bar{B}\bar{C}$
$A+B+\bar{C}$	0	0	1	$\bar{A}\bar{B}C$
$A+\bar{B}+C$	0	1	0	$\bar{A}B\bar{C}$
$A+\bar{B}+\bar{C}$	0	1	1	$\bar{A}BC$
$\bar{A}+B+C$	1	0	0	$A\bar{B}\bar{C}$
$\bar{A}+B+\bar{C}$	1	0	1	$A\bar{B}C$
$\bar{A}+\bar{B}+C$	1	1	0	$AB\bar{C}$
$\bar{A}+\bar{B}+\bar{C}$	1	1	1	ABC

Examples

Conversion between SSOP & SPOS:

01. Expand $\bar{A}+\bar{B}$ ***to SSOP form***

Sol: $f(A,B)=\bar{A}+\bar{B}=\bar{A}(B+\bar{B})+\bar{B}(A+\bar{A})$

$=\bar{A}B+\bar{A}\bar{B}+A\bar{B}+\bar{A}\bar{B}$

$=\bar{A}B+A\bar{B}+\bar{A}\bar{B}$

$=m_0+m_1+m_2=\sum m(0,1,2)$.

02. $f(A,B,C,D)=A+B\bar{C}+AB\bar{D}+ABCD$

Sol: $f(A,B,C,D)=A(B+\bar{B})(C+\bar{C})(D+\bar{D})+B\bar{C}(A+\bar{A})(D+\bar{D})+AB\bar{D}(C+\bar{C})+ABCD$

$=ABCD+AB\bar{C}D+AB\bar{C}\bar{D}+A\bar{B}CD+A\bar{B}C\bar{D}+A\bar{B}\bar{C}D+A\bar{B}\bar{C}\bar{D}+\bar{A}B\bar{C}D+\bar{A}B\bar{C}\bar{D}$

$=\sum m(4,5,8,9,10,11,12,13,14,15)$

03. $f(A,B,C)=\sum m(0,2,4,6,7)$

Sol: $\overline{f(A,B,C)}=\sum m(1,3,5)=m_1+m_3+m_5$

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

$=\overline{m_1+m_3+m_5}=\overline{m_1}.\overline{m_3}.\overline{m_5}$

$=\pi M(1,3,5)$

$\overline{m_j}=M_j$

04. Expand $f=A(\bar{B}+A)B$ ***to maxterms minterms***

Sol: $A=A+B\bar{B}=(A+B)(A+\bar{B})$

$B=B+A\bar{A}=(B+A)(B+\bar{A})$

$f=(A+B)(A+\bar{B})(A+\bar{B})(A+B)(\bar{A}+B)$

$=(A+B)(A+\bar{B})(\bar{A}+B)$

$=\pi M(0,1,2)=\sum m(3)$

- Sum of products form can be implemented by using two level gate network NAND – NAND logic.
- NAND – NAND realization is same as AND – OR.
- Product of sums form can be implemented by using two level gate network NOR – NOR logic.
- NOR – NOR realization is same as OR – AND.
- If the signals are propagating through two stages of gates, then it is called two level gate networks.

Degenerative form:

A two level gate network is said to be degenerative if it degenerates to a single operation.

Example:

AND – AND is equivalent to AND.

The following two level gate networks are degenerative forms.

AND – AND	AND
OR – OR	OR
OR – NOR	NOR
AND – NAND	NAND
NOR – NAND	OR
NAND – NOR	AND

SSOP and SPOS forms from truth table:

(i) SSOP expression from truth table:

A Boolean function may be expressed algebraically from a given truth table by forming a minterm for each combination of the variables which produces 1 in the function, and then taking the OR of all those terms.

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

$F(X,Y)=X\bar{Y}+\bar{X}Y=\sum m(1,2)$

(ii) SPOS expression from truth table:

A Boolean function may be expressed algebraically from a given truth table by forming the max terms for each combination of the variables which produces '0' in the function, and then taking the AND of all those terms.

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

$F=(X+Y)(\bar{X}+\bar{Y})$

$=\pi M(0,3)$

Sum of all the minterms of given Boolean function is equal to '1'.

$F(X,Y,Z)=\sum m(0,1,2,3,4,5,6,7)=1$

Product of all the maxterms of a given Boolean function is equal to '0'

$F(X,Y,Z)=\pi M(0,1,2,3,4,5,6,7)=0$.

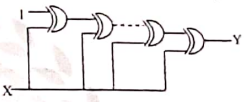
Class Room Practice Questions

01. Two 2's complement numbers having sign bits x and y are added and the sign bit of the result is z. Then, the occurrence of overflow is indicated by the Boolean function.

(a) xyz (b) $\bar{x}\bar{y}\bar{z}$

(c) $\bar{x}\bar{y}z+xy\bar{z}$ (d) xy + yz + zx

02. If the input to the digital circuit (in the figure) consisting of a cascade of 20 XOR-gates is X then the output Y is equal to



(a) 0 (b) 1

(c) $\bar{X}$ (d) X

03. How many min terms (excluding redundant terms) does the minimal switching function

$f(v,w,x,y,z)=x+\bar{y}z$

Originally have?

(a) 16 (b) 20

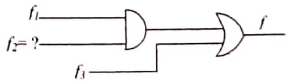
(c) 24 (d) 32

04. Given f_1, f_3, and f in canonical sum of products form (in decimal) for the circuit.

$f_1=\Sigma m(4,5,6,7,8)$

$f_3=\Sigma m(1,15)$

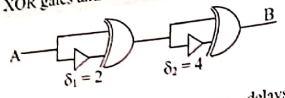
$f=\Sigma m(1,6,8,15)$ Then f_2 is



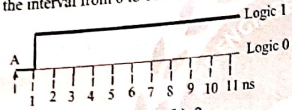
(a) $\Sigma m(4,6)$ (b) $\Sigma m(4,8)$

(c) $\Sigma m(6,8)$ (d) $\Sigma m(4,6,8)$

05. Consider the following circuit composed of XOR gates and non-inverting buffers.



The non-inverting buffers have delays δ_1=2 ns and δ_2= 4 ns as shown in the figure. Both XOR gates and all wires have zero delay. Assume that all gate inputs, outputs and wires are stable at logic level 0 at time0. If the following waveform is applied at input A, how many transition(s) (change of logic levels) occur(s) at B during the interval from 0 to 10 ns?



(a) 1 (b) 2
(c) 3 (d) 4

06. Let, $x_1 \oplus x_2 \oplus x_3 \oplus x_4 = 0$ where x_1, x_2, x_3, x_4 are Boolean variables, and $\oplus$ is the XOR operator. Which one of the following must always be **TRUE**?

(A) $x_1 x_2 x_3 x_4 = 0$

(B) $x_1 x_3 + x_2 = 0$

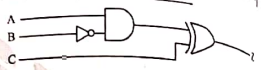
(C) $\bar{x}_1 \oplus \bar{x}_3 = \bar{x}_2 \oplus \bar{x}_4$

(D) $x_1 + x_2 + x_3 + x_4 = 0$

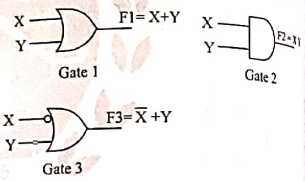
07. A 3 input majority gate is defined by the logic function M(a,b,c) = ab + bc + ca. Which one of the following gates is represented by the function $M(M(a,b,c), M(a,b,\bar{c}), c)$?

(a) 3-Input NAND gate
(b) 3-Input XOR gate
(c) 3-Input NOR gate
(d) 3-Input XNOR gate

08. All the logic gates shown in the figure have a propagation delay of 20 ns. Let A = C = 1 and B = 1 until time t = 0. At t = 0, all the inputs flip (i.e, A = C = 1 and B = 0) and remain in that state. For t > 0, output Z is 1 for a duration (in ns) of ____



09. A universal logic gate can implement any Boolean function by connecting sufficient number of them appropriately. Three gates are shown.



Which one of the following statements is TRUE?

(a) Gate 1 is a universal gate.
(b) Gate 2 is a universal gate.
(c) Gate 3 is a universal gate.
(d) None of the gates shown is a universal gate.

Key for CRPQ

01. (c)	02. (b)	03. (b)	04. (c)	05. (d)
06. (c)	07. (b)	08. 40	09. (c)	

3 K - Maps

Minimization of switching functions using K-map

The Boolean function can be simplified algebraically, but being not a systematic method, we can never be sure that whether the minimal expression obtained is the real minimal (or) not.

Karnaugh map (K-map):

- The K-map method is a systematic method of simplifying the Boolean expressions. The K-map is a chart (or) a graph, composed of an arrangement of adjacent cells, each representing a particular combination of variables in sum (or) product form.
- A K-map is a graphical representation of Boolean expressions, a two variable K-map will have 4 cells (or) squares, 3 Variable K-map will have 8-cells, n-variable K-map will have 2^n cells (or) squares.

Note: Adjacent cells differ by 1 bit

To maintain adjacency property gray code sequence is used in K-maps (Any two adjacent cells will differ by only one bit).

Two variable K-map:

- Each cell represents a term of two literals.
- Grouping two adjacent (pair) squares containing 1's represents, a term of one literal.
- Grouping of four adjacent squares containing 1's represent the function = 1

X \ Y	0	1
0	$\bar{X}\bar{Y}$	$\bar{X}Y$
1	$X\bar{Y}$	XY

Three Variable K-map:

- Each cell represents a term of three literals.
- Grouping two adjacent cells containing 1's (pair) represent a term of two literals.
- Grouping four adjacent cells containing 1's (quad) represent a term of one literal.
- Grouping eight adjacent cells containing 1's represents the function = 1

X \ YZ	00	01	11	10
0	$\bar{x}\bar{y}\bar{z}$	$\bar{x}\bar{y}z$	$\bar{x}yz$	$\bar{x}y\bar{z}$
1	$x\bar{y}\bar{z}$	$x\bar{y}z$	xyz	$xy\bar{z}$

Four variable K-map:

- Each cell (or) square represents one min term, giving a term of four literals.
- Grouping two adjacent squares containing 1's represents a term of three literals.
- Grouping four adjacent squares containing 1's represents a term of two literals.
- Grouping eight adjacent squares containing 1's represents a term of one literal
- Grouping of sixteen adjacent squares containing 1's represent the function = 1

wx \ yz	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

Rules to simplify K-maps:

- At the time of grouping the adjacent cells containing 1's always use maximum possible group
- All the cells containing 1's must be covered at least once in any group.
- At the time of grouping don't cares (X) values can be taken as 1's.
- All don't cares values need not be covered.

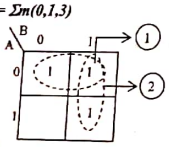
No. of variables	No. of cells containing 1's grouped	No. of variables eliminated	No .of literals present in the resulting term
2	4	2	0
	2	1	1
	1	0	2
3	8	3	0
	4	2	1
	2	1	2
	1	0	3
4	16	4	0
	8	3	1
	4	2	2
	2	1	3
	1	0	4

Examples

By using above rules simplify the given expression using K-map

01. ***F(A,B) = Σm(0,1,3)***

Sol:

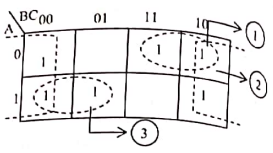


$F = \overline{A} + B$

In the (1) grouping 'A' is constant as a '0' but 'B' varies from 0 to 1, and in the (2) grouping, B is constant as a '1' but 'A' varies from '0' to a '1'. So the reduced expression $= \overline{A} + B$

02. ***Reduce f = Σm(0,2,3,4,5,6)***

Sol:



From (1) grouping resultant expression is $= \overline{C}$

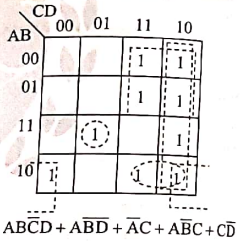
From (2) grouping resultant expression is $= \overline{A}B$

From (3) grouping resultant expression is $= A\overline{B}$

$F = A\overline{B} + \overline{A}B + \overline{C}$

03. ***Reduce the expression using mapping F = Σm(2,3,6,7,8,10,11,13,14)***

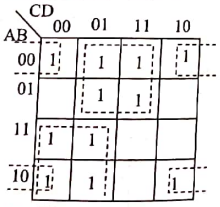
Sol:



$F = AB\overline{C}D + A\overline{B}\overline{D} + \overline{A}C + A\overline{B}C + C\overline{D}$

04. ***F = Σm(0,1,2,3,5,7,8,9,10,12,13)***

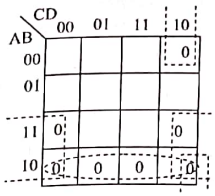
Sol:



$F = \overline{B}\,\overline{D} + A\overline{C} + \overline{A}D$

05. ***F = πM(2,8,9,10,11,12,14)***

Sol:



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

Prime Implicant:

Each square (or) rectangle made up of the bunch of adjacent min terms is called a sub cube. Each of these sub cubes is called a prime implicant (P I).

Essential Prime Implicant:

The prime implicant which contains at least one'1' which can not be covered by any other prime implicant is called Essential prime Implicant (EPI).

Redundant Prime Implicant:

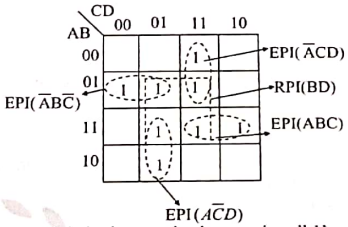
The prime implicant whose each '1' is covered at least by one EPI is called a redundant PI.

Selective Prime Implicant:-

A prime implicant which is neither an EPI nor a RPI is called selective prime Implicant (SPI).

Examples

01. ***The fig shows EPI'S and RPI'S***

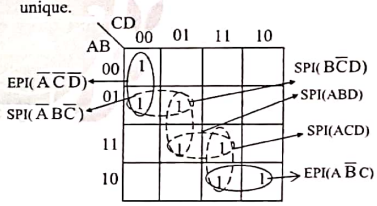


The minimized expression by covering all 1's is $F = \overline{A}CD + ABC + A\overline{C}D + \overline{A}B\overline{C}$

It is unique expression.

02. ***F(A,B,C,D)=∑m(0,4,5,10,11,13,15).***

The SPI's are marked by dotted squares. This shows that minimized expression need not be unique.



- The dotted line shows SPI's, normal line shows EPI's.
- The minimized expression is obtained by including two EPI's and select a set of SPI's to cover the remaining uncovered. Min terms 5,13,15. These can be covered in the following ways

(a) (4,5) and (13,15)– $A\overline{B}\overline{C} + ABD$

(b) (5,13) and (13,15) – $B\overline{C}D + ABD$

(c) (5,13) and (15,11) – $B\overline{C}D + ACD$

$$F(A,B,C,D) = \underbrace{\overline{A}CD}_{EPI} + \underbrace{A\overline{B}C}_{EPI} + \underbrace{\overline{A}B\overline{C}}_{SPI} + \underbrace{ABD}_{SPI}$$

$$= \underbrace{\overline{A}C\overline{D} + A\overline{B}C}_{EPI} + \underbrace{B\overline{C}D}_{SPI} + \underbrace{ABD}_{SPI}$$

$$= \underbrace{\overline{A}C\overline{D} + A\overline{B}C}_{EPI} + \underbrace{B\overline{C}D + ACD}_{SPI}$$

- The above function has three different minimized expressions.

03. ***Differentiate b/w a PI, and a non – PI, and an EPI, and a non EPI.***

Sol: A PI is a square (or) rectangle made up of the bunch of adjacent min terms. It is also Called a sub cube.

A non PI is a min term which does not have any adjacent min terms i.e. which not form a part of a bigger square.

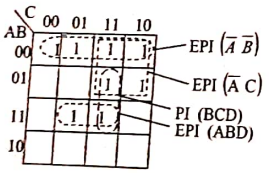
An EPI is a PI, which contains at least one '1' which cannot be covered by any other PI.

A non-EPI is a selective PI. It is neither EPI nor redundant It may (or) may not appear in the minimal expression.

04. ***Reduce the following expression using K-map and identify PI's and EPI.***

$F = \sum m(0,1,2,3,6,7,13,15)$

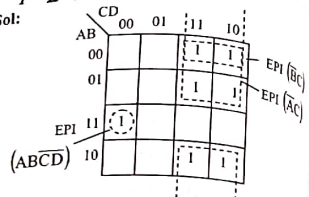
Sol:



Minimal $F = \overline{A}\,\overline{B} + \overline{A}C + ABD$

05. ***Reduce the following using K-map and identify the PI and EPI***
$F = \sum m(2,3,6,7,10,11,12)$

Sol:



Minimal exp $F = B\overline{C} + \overline{A}C + AB\overline{C}\,\overline{D}$

Don't care combinations:

For certain input combinations, the value of the output is unspecified either because of the input combinations are invalid (or) because the precise value of the output is of no consequence. The combinations for which the values of the expression are not specified are called don't care combinations (or) optional combinations, and such expressions, therefore stand incompletely specified. The output is a don't care for these invalid combinations.

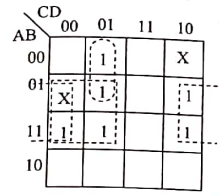
Example:

- In Excess-3 code system, the binary states 0000, 0001, 0010, 1101, 1110, 1111 are unspecified and never occur. These are called don't cares
- In 8421 BCD code, the binary states 1010,1011,1100,1101,1110,1111 are invalid and the corresponding outputs are don't cares. The don't care terms are denoted by d, X, ϕ.
- During the process of design using an sop map, each don't care is treated as a '1' if it helpful in map reduction, otherwise treated as '0' and left above.
- During the process of design using a pos map, each doesn't care is treated as '0' if it is useful in map reduction, otherwise it is treated as a '1' and left above.

Examples

01. $F = \sum m(1,5,6,12,13,14) + \sum d(2,4)$

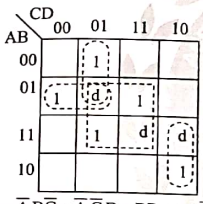
Sol:



Minimal Expression is $F = B\overline{C} + B\overline{D} + \overline{A}\,\overline{C}\,D$

02. $F(A,B,C,D) = \sum m(1,4,7,10,13) + \sum d(5,14,15)$

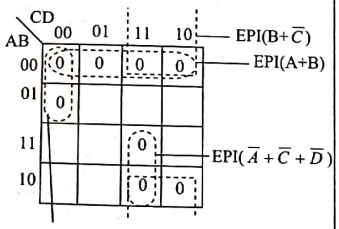
Sol:



$F = \overline{A}B\overline{C} + \overline{A}\,\overline{C}D + BD + AC\overline{D}$

03. $F(A,B,C,D) = \pi M(0,1,2,3,4,10,11,15)$

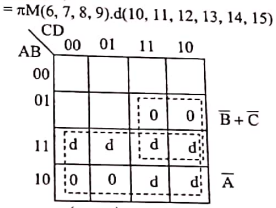
Sol:



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

04. $F(A, B, C, D) = \pi M(6, 7, 8, 9).d(10, 11, 12, 13, 14, 15)$

Sol:



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

05. Make a K-map of the following expression and obtain minimal expression in sop & pos form

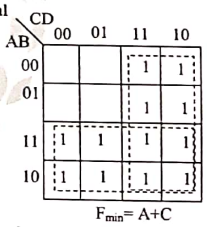
$F = AB + A\overline{C} + C + AD + A\overline{B}C + ABC$

Sol: After converting the above expression in to SSOP form/Canonical Form

$F = \sum m(2,3,6,7,8,9,10,11,12,13,14,15)$

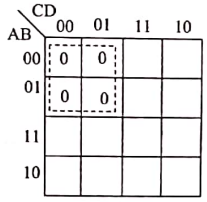
and $F = \pi M(0,1,4,5)$

SOP minimal



$F_{min} = A + C$

POS minimal



$F_{min} = A + C$

Class Room Practice Questions

01. Minimum SOP for f(w, x, y, z) shown in karnaugh – map below is

yz \ wx	00	01	11	10
00	0	1	1	0
01	X	0	0	1
11	X	0	0	1
10	0	1	1	X

(a) $xz + y'z$ (b) $x\,z' + z\,x'$
(c) $x'\,y + z\,x'$ (d) None

02. What is the minimal form of the Karnaugh map shown below? Assume that X denotes a don't care term.

cd \ ab	00	01	11	10
00	1	X	X	1
01	X			1
11				
10	1			X

(a) $\overline{bd}$ (b) $\overline{bd} + \overline{bc}$
(c) $\overline{bd} + a\overline{bc}d$ (d) $\overline{bd} + \overline{bc} + \overline{cd}$

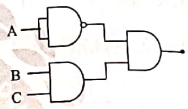
03. Find the SOP and POS expressions for the K- MAP shown below.

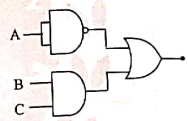
wz → / xy ↓	00	01	11	10
00	0	X	0	0
01	0	X	1	1
11	1	1	1	1
10	0	X	0	0

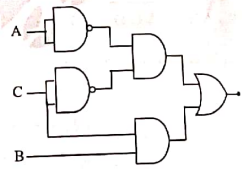
04. For an n-variable Boolean function, the maximum number of prime implicates is
(a) $2^{(n-1)}$ (b) n/2
(c) 2^n (d) $2^{(n+1)}$

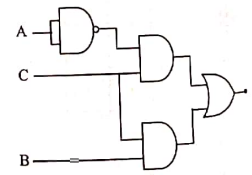
05. Which of the following logic circuits is a realization of the function F whose Karnaugh map is shown in figure.

C \ AB	00	01	11	10
0	1	1		
1		1	1	

(a)


(b)


(c)


(d)


06. The number of min-terms after minimizing the following Boolean expression is _____.
$[D^1 + AB^1 + A^1C + AC^1D + A^1C^1D]^1$

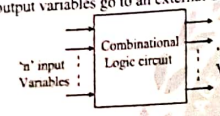
07. The total number of prime implicants of the function f(w, x, y, z) = Σ(0, 2, 4,5, 6, 10) is _____

Key for CRPQ

01. (b)	02. (b)	03. SOP: x y + y w , POS: (y)(x+w)	
04. (a)	05. (c)	06. 1	07. 3

4 Combinational Circuits

- The output of a combinational circuit depends on its present inputs only.
- Combinational circuits perform a specific information processing operation fully specified logically by a set of Boolean functions. The combinational circuit consists of input variables, logic gates, and output variables.
- The block diagram of combinational circuit is shown in fig. The 'n' input binary variables come from an external source and the 'm' output variables go to an external destination.



Design Procedure:

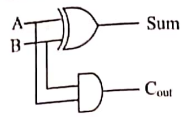
- The problem is stated.
- The no. of available input variables and required o/p variables is determined.
- The input & output variables are assigned letter symbols.
- The truth table that defines the required relationship between inputs and outputs is designed.
- The simplified Boolean function for each output is determined.
- The logic diagram is drawn.

Adders:

Half adder :

- The combinational circuit that performs the addition of two bits is called half adder. (The name is so because, two half adders are required to implement a full adder).
- A Half adder circuit with two binary inputs and two binary output's (sum, carry). It is an arithmetic circuit used to perform the arithmetic operation of addition of two single bit words.

Truth table



Inputs		outputs	
A	B	Sum	C_{out}
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

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

$C_{out} = AB$

Note:

- To realize a half adder circuit, 5 two inpu NAND gates are required.
- To realize a half adder circuit, 5 two inpu NOR gates are required.

Full adder:

A full adder is a combinational circuit used to add 3 bits and outputs sum and carry

Inputs			Outputs	
A	B	C_{in}	Sum	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Sum :

A \ BC_{in}	00	01	11	10
0		1		1
1	1		1	

$Sum = \overline{A}\,\overline{B}C_{in} + \overline{A}B\overline{C_{in}} + A\overline{B}\,\overline{C_{in}} + ABC_{in}$

$Sum = A \oplus B \oplus C_{in}$

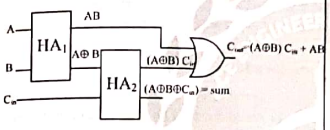
Carry :

A \ BC_{in}	00	01	11	10
0			1	
1		1	1	1

$C_{out} = AB + AC_{in} + BC_{in}$ or

$= (A \oplus B)\,C_{in} + AB$

Implementation of FA using 'HA', 'OR' gate.



Note:

- To implement Full Adder, 9 two input NAND gates are required.
- To implement Full Adder, 9 two input NOR gates are required.

Subtractors:

Half Subtractor:

A Half Subtractor is combinational circuit that subtracts two binary bits and produces the output as difference, borrow.

Inputs		Outputs	
A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$Diff = A \oplus B$

$Borrow = \overline{A}B$

Note:

- To implement a Half subtractor circuit, '5' two input NAND gates are required
- To implement a Half subtractor circuit, '5' two input NOR gates are required.

Full subtractor:

Full subtractor used to subtract three binary digits and produces diff, borrow as outputs.

Inputs			Outputs	
A	B	bi	Diff	Borrow
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$Diff = \overline{A}\,\overline{B}bi + \overline{A}B\overline{bi} + A\overline{B}\,\overline{bi} + ABbi$

$= A \oplus B \oplus bi$

$Borrow = \overline{A}B + \overline{(A \oplus B)}bi$

Note:

- To implement a Full subtractor circuit '9' two input NAND gates are required
- To implement a full subtractor circuit '9' two input NOR gates are required
- To implement a Half adder – The No. of basic (AND, OR, NOT) gates required – 6

  The No. of NAND gates = 5

  The No. of NOR gates = 5.
- To implement a Half subtractor – basic gates = 5

  NAND gate = 5

  NOR gate = 5
- To implement a full adder - NAND gates = 9

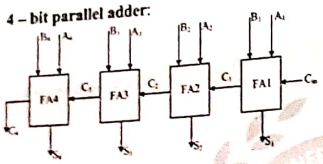
  NOR gates = 9
- To implement a full subtractor -NAND gates = 9

  NOR gates = 9

Binary parallel adder:

It is a digital circuit that adds two binary numbers in parallel form and produces arithmetic sum of those numbers in parallel form. It consists of full adders connected in chain, with the output carry from each full adder connected to the input carry of the next full adder in chain.

4 – bit parallel adder:

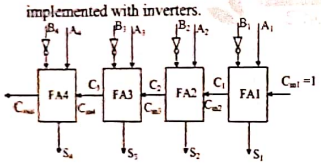


Note:

To implement a n – bit parallel adder requires n – full adders. The output carry from one package must be connected to the input carry of the one with the next higher order bits.

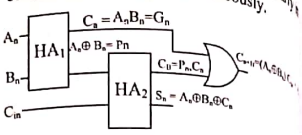
4-bit parallel subtractor using 2's complement method:

The two's complement can be obtained by taking the '1's complement and adding 1 to the LSB. The one's complement can be implemented with inverters.



LOOK – AHEAD - CARRY - ADDER:

In the parallel adder, the speed with which an addition can be performed is governed by the time required for the carries to propagate or ripple through all of the stages of the adder. The look – ahead – carry – adder speeds up the process by eliminating this ripple carry delay. It examines all the input bits simultaneously and also generates the carry bits for all the stages simultaneously.



$S_n = P_n \oplus C_n$, $P_n = A_n \oplus B_n$

Output carry $C_B = P_n \cdot C_n = (A_n \oplus B_n) C_n$, which becomes input carry for (n+1) stage.

$C_{n+1} = G_n + P_n C_n$, where $G_n = A_n \cdot B_n$.

The general expression for n stages designated 0 to n – 1 is

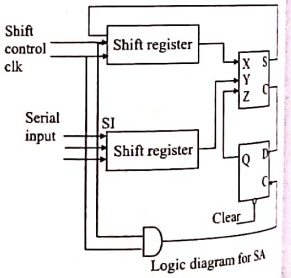
$C_n = G_{n-1} + P_{n-1} \cdot C_{n-1}$

$= G_{n-1} + P_{n-1} G_{n-2} + P_{n-1} P_{n-2}$

$G_{n-3} + - - + P_{n-1} - - - P_0 C_0$

Serial adder:

A serial adder is used to add binary numbers in serial form. The two binary numbers to be added serially and stored in two shift registers A & B. Bits are added one pair at a time through a single full adder. The carry out of the FA is transferred to D – F|F. The output of F/F is used as carry input for the next pair of significant bits. The sum bit from the S output of the FA transferred to third shift register.



Logic diagram for SA

Difference b/w serial Adder and parallel adder:

Serial Adder	Parallel Adder
1. Serial adder uses shift registers	1. Parallel adder uses registers with parallel load
2. Only one FA, and a carry flip flop	2. The no. of full adders is equal to no. of bits in the binary numbers
3. Serial adder is a sequential circuit	3. Excluding registers, it is a combinational circuit

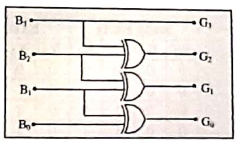
Converters:

A code converter is a logic circuit whose inputs are bit patterns representing numbers (or characters) in one code and whose outputs are corresponding representations in a different code.

Design of a 4 - bit Binary to Gray code Converters:

There are 16 possible combinations of 4 - bit binary input and all of them are valid. Hence no don't care.

4 Bit Binary				4 Bit Gray			
B_4	B_3	B_2	B_1	G_4	G_3	G_2	G_1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0



$G_4 = \Sigma m\ (8, 9, 10, 11, 12, 13, 14, 15) = B_4$

$G_3 = \Sigma m\ (4,5,6,7,8,9,10,11)$

$= \overline{B}_4 B_3 + B_4 \overline{B}_3 = B_4 \oplus B_3$

$G_2 = \Sigma m(2,3,4,5,10,11,12,13)$

$= \overline{B}_3 B_2 + B_3 \overline{B}_2 = B_3 \oplus B_2$

$G_1 = \Sigma m(1,2,5,6,9,10,13,14)$

$= \overline{B}_2 B_1 + B_2 \overline{B}_1 = B_2 \oplus B_1$

Design of 4 bit binary to BCD code converter:

The input is 4 bit binary-16 possible combinations- all are valid-no don't cares. Since the input is 4 bit binary (the max. Value in input represented with two decimal digits) the output has to be on 8 - bit code. But the first three bits are zero.

$A = \Sigma m(10,11,12,13,14,15) = B_4B_3 + B_4B_2$

$B = \Sigma m(8,9) = B_4\overline{B}_3\overline{B}_2$

$C = \Sigma m(4,5,6,7,14,15) = \overline{B}_4 B_3 + B_3 B_2$

$D = \Sigma(2,3,6,7,12,13) = B_4B_1\overline{B}_2 + \overline{B}_4 B_2$

$E = \Sigma m(1,3,5,7,9,11,13,15) = B_1$

Design of 4 bit BCD to Xs-3 code converter:

The combinations of input 1010, 1011, 1100, 1101, 1110, 1111 are invalid BCD they treated as don't cares.

8421 BCD				Excess – 3			
B_4	B_1	B_2	B_1	X_4	X_3	X_2	X_1
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

$X_4 = \Sigma m(5,6,7,8,9)+\Sigma d\ (10,11,12,13,14,15)$

$= B_4 + B_3B_2 + B_3B_1$

$X_3 = \Sigma m(1,2,3,4,9)+\Sigma d\ (10,11,12,13,14,15)$

$= B_3\overline{B}_2\overline{B}_1 + \overline{B}_3B_1 + \overline{B}_3B_2$

$X_2 = \Sigma m(0,3,4,7,8)+\Sigma d(10,11,12,13,14,15)$

$= \overline{B}_2\overline{B}_1 + B_2B_1$

$X_1 = \Sigma m(0,2,4,6,8)+\Sigma d(10,11,12,13,14,15)$

$= \overline{B}_1$

Examples

01. ***Design a sop circuit to detect the decimal Numbers 0,2,4,6,8 in a 4 bit 5211 BCD code input.***

Sol:

Decimal Number	Output (F)	5 A	2 B	1 C	1 D
0	1	0	0	0	0
1	0	0	0	0	1
2	1	0	0	1	1
3	0	0	1	0	1
4	1	0	1	1	1
5	0	1	0	0	0
6	1	1	0	1	0
7	0	1	1	0	0
8	1	1	1	1	0
9	0	1	1	1	1

$F = \Sigma m(0,3,7,10,14)+\Sigma d(2,4,6,9,11,13)$

$F_{min} = \overline{AD} + \overline{AC} + C\overline{D}$

02. ***Design a circuit to detect decimal numbers 0,1,4,6,7 and 8 in a 4-bit XS – 3 code input.***

Sol:

Decimal	4 bit Excess-3 A	B	C	D	output(f)
0	0	0	1	1	1
1	0	1	0	0	1
2	0	1	0	1	0
3	0	1	1	0	0
4	0	1	1	1	1
5	1	0	0	0	0
6	1	0	0	1	1
7	1	0	1	0	1
8	1	0	1	1	1

$F = \Sigma m(3,4,7,9,10,11)+\Sigma d(0,1,2,13,14,15)$

$F_{min} = CD + AD + AC + \overline{A}\,\overline{C}\,\overline{D}$

03. ***Design a logic circuit with 4 inputs A, B, C, D that will produce output '1' only when ever two adjacent input variables are 1's. A and D are also be treated as adjacent.***

Sol:

Input A	B	C	D	Output F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

$F = \Sigma m\ (3,6,7,9,11,12,13,14,15)$

$F_{min} = AB+AD+BC+CD$

Comparators:

A comparator is a logic circuit used to compare the magnitudes of two binary numbers.

Note:

X NOR gate is a basic comparator, because its output is a '1' only if its two input bits are equal.

One bit magnitude comparator:

Let the 1 bit number are $A = A_0$, $B = B_0$

Input A_0	B_0	Output A<B (L)	A = B (E)	A>B (G)
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

$G = A_0\overline{B}_0$

$E = A_0 \odot B_0$

$L = \overline{A}_0B_0$

2- bit magnitude comparator:

Let the two 2-bit Numbers are

$A = A_1\,A_0$,

$B = B_1\,B_0$

A_1	A_0	B_1	B_0	A<B (L)	A=B (E)	A>B (G)
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

$A > B(G) = A_1\overline{B}_1 + (A_1 \odot B_1)A_0\,\overline{B}_0$

$A < B\,(L) = \overline{A}_1B_1 + (A_1 \odot B_1)\,\overline{A}_0B_0$

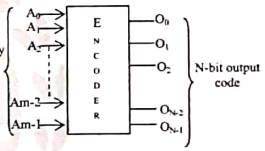
$A = B\,(E) = (A_1 \odot B_1)\,(A_0 \odot B_0)$

Encoder:

An encoder is a device whose inputs are decimal digits and/or alphabetic characters and whose outputs are coded representation of those inputs.

Note:

Encoder has a number of input lines, only one of which activated at a given time, and produces N-bit output – code Depending on which input is activated.



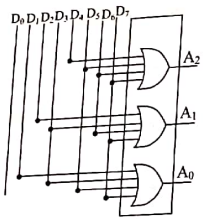
Block diagram of Encoder

Examples

01. ***Octal to Binary Encoder:***

(8- line to 3-line encoder)

Octal digits		Binary A_2	A_1	A_0
D_0	0	0	0	0
D_1	1	0	0	1
D_2	2	0	1	0
D_3	3	0	1	1
D_4	4	1	0	0
D_5	5	1	0	1
D_6	6	1	1	0
D_7	7	1	1	1



D_0 is not used

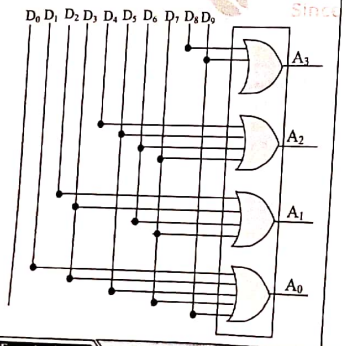
$A_2 = D_4 + D_5 + D_6 + D_7$

$A_1 = D_2 + D_3 + D_6 + D_7$

$A_0 = D_1 + D_3 + D_5 + D_7$

02. *Decimal to BCD encoder:*

Decimal		BCD			
		A_3	A_2	A_1	A_0
D_0	0	0	0	0	0
D_1	1	0	0	0	1
D_2	2	0	0	1	0
D_3	3	0	0	1	1
D_4	4	0	1	0	0
D_5	5	0	1	0	1
D_6	6	0	1	1	0
D_7	7	0	1	1	1
D_8	8	1	0	0	0
D_9	9	1	0	0	1



$A_3 = D_8 + D_9$,

$A_2 = D_4 + D_5 + D_6 + D_7$

$A_1 = D_2 + D_3 + D_6 + D_7$,

$A_0 = D_1 + D_3 + D_5 + D_7 + D_9$

Priority encoder:

- The encoders discussed in previous are operate correctly, provided that one and only one input is HIGH at any give time.
- A priority encoder is a logic ckt that responds to just one input in accordance with some priority system, among all those that may be simultaneously HIGH.

4-bit Priority encoder:

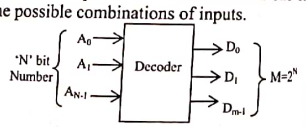
Inputs				Output		
D_0	D_1	D_2	D_3	A	B	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

'V' is a valid bit indicator that is set to '1' when one or more inputs are equal to 1. If all inputs are '0' there is no valid input and 'V' is equal to zero. The other two outputs are not inspected where 'V' equals '0' and are specified as don't care conditions.

In above diagram, D_3 is having highest priority, as D_0 is having least priority

Decoders:

A decoder is a logic ckt that convert an N-bit binary input code to m-output lines such that only one output is activated for each one of the possible combinations of inputs.



Block diagram of Decoder

Among the M outputs only one Output will be high for each input code.

3- Line to 8-line decoder:

Inputs			Outputs							
A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

$D_0 = \bar{A}\bar{B}\bar{C}, D_1 = \bar{A}\bar{B}C$

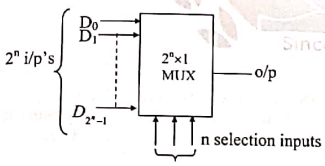
$D_2 = \bar{A}B\bar{C}, D_3 = \bar{A}BC$

$D_4 = A\bar{B}\bar{C}, D_5 = A\bar{B}C$

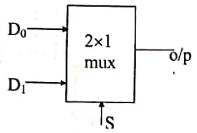
$D_6 = AB\bar{C}, D_7 = ABC$

Multiplexer (Data selector):

A multiplexer also called MUX, or a data selector is a logic ckt that accepts several data inputs and allows only one of them at a time to get through to the output. The routing of the desired data input to the output is controlled by 'SELECT' inputs. Normally there are 2^n input lines and n selection lines and one output.



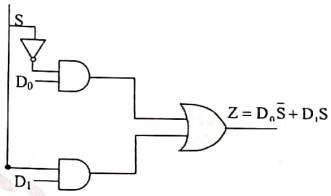
2 - input multiplexer:



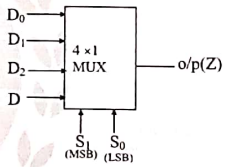
Function table:

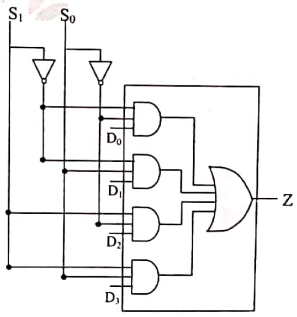
S	Output(Z)
0	D_0
1	D_1

$Z = D_0\bar{S} + D_1S$



4 - input multiplexer (4 ×1 MUX):





Function table:

Select i/p's		Output(Z)
S_1	S_0	Z
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$Z = \overline{S_1}\,\overline{S_0}\,D_0 + \overline{S_1}S_0D_1 + S_1\overline{S_0}\,D_2 + S_1S_2D_3$

Applications of MUX:

- Data selection
- Data routing
- Operation sequencing
- Parallel – to – serial conversion
- Waveform generation
- Logic function generation.

Examples

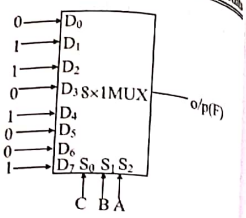
01. ***Implement the logic function $F = A \oplus B \oplus C$ using a multiplexer***

Sol:

There are three input variable we can use a mux with three data select inputs.

Truth table:

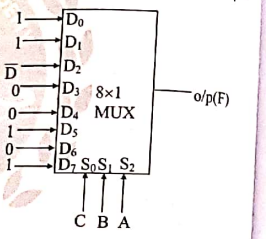
S_2 (A)	S_1 (B)	S_0 (C)	F=A⊕B⊕C
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



02. ***Implement $F = \Sigma m(0,1,2,3,4,10,11,14,15)$ using 8×1 MUX***

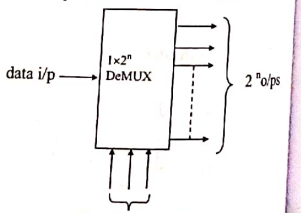
Sol:

	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
$\overline{D}$	(0)	(2)	(4)	6	8	(10)	12	(14)
D	(1)	(3)	5	7	9	(11)	13	(15)
	1	1	$\overline{D}$	0	0	1	0	1



Demultiplexer: (Data Distributor):

It takes single input and distributed it over several outputs. It is a 1 to 2^n device.



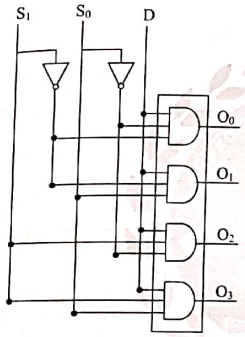
1 line to 4 – line DEMUX:

Truth table:

S_1	S_0	O_3	O_2	O_1	O_0
0	0	0	0	0	D
0	1	0	0	D	0
1	0	0	D	0	0
1	1	D	0	0	0

$O_0 = D\overline{S_0}\,\overline{S_1}$ $O_1 = D\overline{S_1}\,S_0$

$O_2 = DS_1\overline{S_0}$ $O_3 = DS_1S_0$



1 line to 8 – line DEMUX:

Truth table:

Select code			Outputs							
S_2	S_1	S_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	D
0	0	1	0	0	0	0	0	0	D	0
0	1	0	0	0	0	0	0	D	0	0
0	1	1	0	0	0	0	D	0	0	0
1	0	0	0	0	0	D	0	0	0	0
1	0	1	0	0	D	0	0	0	0	0
1	1	0	0	D	0	0	0	0	0	0
1	1	1	D	0	0	0	0	0	0	0

$O_7 = DS_2S_1S_0$

$O_6 = D.\ S_2S_1\overline{S_0}$

$O_5 = DS_2\overline{S_1}S_0$

$O_4 = DS_2\overline{S_1}\,\overline{S_0}$

$O_3 = D\overline{S_2}S_1S_0$

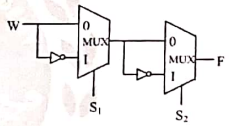
$O_2 = D\overline{S_2}S_1\overline{S_0}$

$O_1 = D\overline{S_2}\,\overline{S_1}S_0$

$O_0 = D\overline{S_2}\,\overline{S_1}\,\overline{S_0}$

Class Room Practice Questions

01. Consider the multiplexer based logic circuit shown in the figure.



Which one of the following Boolean functions is realized by the circuit?

(a) $F = W\overline{S_1S_2}$

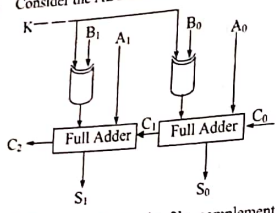
(b) $F = WS_1 + WS_2 + S_1S_2$

(c) $F = \overline{W} + S_1 + S_2$

(d) $F = W \oplus S_1 \oplus S_2$

02. A half adder is implemented with XOR and AND gates. A full adder is implemented with two half adders and one OR gate. The propagation delay of an XOR gate is twice that of and AND/OR gate. The propagation delay of an AND/OR gate is 1.2 microseconds. A 4-bit ripple-carry binary adder is implemented by using four full adders. The total propagation time of this 4-bit binary adder in microseconds is _____.

03. Consider the ALU shown below



If the operands are in 2's complement representation, which of the following operations can be performed by suitably setting the control lines K and C_0 only (+ and – denote addition and subtraction respectively)?

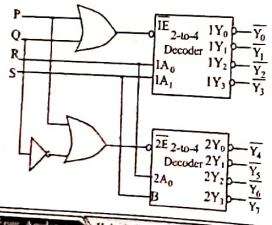
(a) A+B, and A–B, but not A+1

(b) A+B, and A+1, but not A–B

(c) A+B, but not A–B, or A+1

(d) A+B, and A–B, and A+1

04. A 1-to-8 demultiplexer with data input D_{in}, address inputs S_0, S_1, S_2 (with S_0 as the LSB) and $\overline{Y}_0$ *to* $\overline{Y}_7$ as the eight demultiplexed outputs, is to be designed using two 2-to-4 decoders (with enable input $\overline{E}$ and address inputs A_0 and A_1) as shown in the figure. D_{in}, S_0, S_1 and S_2 are to be connected to P,Q,R and S , but not necessarily in this order. The respective input connections to P, Q, R and S terminals should be



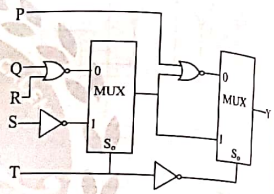
(a) S_2, D_{in}, S_0, S_1

(b) S_1, D_{in}, S_0, S_2

(c) D_{in}, S_0, S_1, S_2

(d) D_{in}, S_2, S_0, S_1

05. For the circuit shown in figure, the delays of NOR gates, multiplexer and inverters are 2ns, 1.5ns and 1ns, respectively. If all the inputs P,Q,R,S and T are applied at the same time instant, the maximum propagation delay (in ns) of the circuit is _____.



06. Consider an eight-bit ripple-carry adder for computing the sum of A and B, where A and B are integers represented in 2's complement form. If the decimal value of A is one, the decimal value of B that leads to the longest latency for the sum to stabilize is _____.

Key for CRPQ

01. (d) 02. 19.2 03. (a) 04. (d) 05.6
06. –1

5 Sequential Circuits

In combinational circuits the present output depends only on present input i.e. any prior input level conditions have no effect on the present outputs.

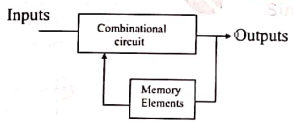
Sequential circuits:-

Sequential circuits are those whose output levels at any instant of time are dependent not only on the levels of present inputs at that time, but also on the state of the circuit, i.e. on the prior input level conditions (past inputs). The past history is provided by feed back from the output back to the input. It means that sequential circuit has memory. Sequential circuits are thus made of combinational circuits and memory elements.

Example:

Counters, shift registers, serial adders, sequence generator, logic function generators.

Block Diagram Of Sequential Circuit:



The information stored in memory element at any given time defines the present state of the sequential circuit. The present state and external inputs determine the output and the next state of the sequential circuit.

Comparison between combinational and Sequential circuits:

Combinational	Sequential
1. The output variables at any instant of time depend only on present input variables.	1. The output variables at any instant of time dependent not only on the present input variables also present state (past input's)
2. Memory unit is not required in Combinational circuits.	2. Memory unit required to store past history of the input variables.
3. These are faster because the delay between the input and output due to propagation delay of gates only	3. Sequential circuits are slower than Combinational circuits.
4. Easy to design	4. Comparatively harder to design

Classification of sequential Circuits:-

These are classified as synchronous and asynchronous sequential circuits, depending on the timing of their signals.

Comparison between synchronous and Asynchronous sequential circuits:

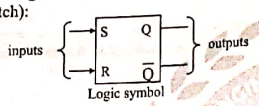
Synchronous sequential circuits	Asynchronous sequential circuits
1. The memory elements are clocked flip-flops.	1. Memory element are either un clocked Flip flops (or) time delay elements.
2. The change in input signals can effect memory elements u p on activation of clock signal.	2. Change in input signals can affect memory elements at any instant of time.
3. Design complexity is more.	3. Design complexity is less

Latches:

It refers to non-clocked flip-flops, because these flip-flops 'latch on' to a '1' (or) a'0' immediately upon receiving the input pulse called SET (or) RESET. They are not dependent on the clock signal for their operation, i.e. a latch is a sequential device that checks all its inputs continuously and changes its outputs accordingly at any time independent of clock signal.

The S-R Latch:

(NOR gate S-R Latch (or) active high S-R Latch):



Logic symbol

Logic diagram

The S-R latch can be constructed by using either two cross coupled NAND gates (or) two cross coupled NOR gates, using NOR gates an Active-High S-R latch can be constructed and using NAND gates an Active-Low S-R latch can be constructed.

The name of the latch, S-R (or) SET-RESET designed from the name of its inputs.

Truth table of S-R Latch:

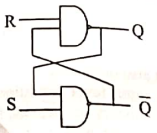
S	R	Q_n	Q_{n+1}	State
0	0	0	0	No Change
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	X	Invalid
1	1	1	X	

Q_n-Present state of FLIP FLOP

Q_{n+1}-next state of FLIP FLOP

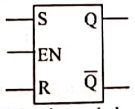
NAND gate S-R Latch (active-low SR Latch):

S	R	Q_n	Q_{n+1}	State
0	0	0	X	Invalid
0	0	1	X	
0	1	0	1	Set
0	1	1	1	
1	0	0	0	Reset
1	0	1	0	
1	1	0	0	No change
1	1	1	1	

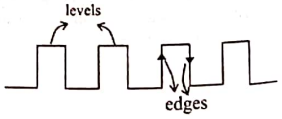


Gated Latches (clocked Flip-Flops):

- In the Latches discussed earlier the Output can change it's state any time the input conditions are changed. So, they are called asynchronous latches.
- A gated S-R Latch requires an ENABLE input. Its S&R inputs control states of the FLIP FLOP. Only when the enable input high. When the enable input is low, the input's becomes ineffective and no change of state can take place. The enable input may be a clock so gated S-R latch is also called a clocked S-R latch (or) Synchronous S-R latch. This type of Flip-Flops are also called level triggered Flip-flops.



Logic symbol



Truth Table:

EN	S	R	Q_n	Q_{n+1}	States
1	0	0	0	0	No change
1	0	0	1	1	
1	0	1	0	0	Reset
1	0	1	1	0	
1	1	0	0	1	Set
1	1	0	1	1	
1	1	1	0	X	Invalid
1	1	1	1	X	
0	X	X	0	0	No change
0	X	X	1	1	

Edge Triggered Flip-flops:

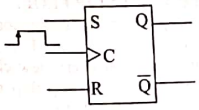
The edge triggered Flip-flops can change its states only when the Clock makes a transition. Clocked flip-flops may be positive edge triggered (or) negative edge triggered. Positive edge triggered flip-flops are those in which 'state transition' takes place only at the positive going (0 to 1 (or) Low to HIGH) edge of the clock pulse and negative edge triggered flip-flops are those in which 'state transitions' takes place only at the negative going (1 to 0(or) HIGH to LOW) edge of the clock signal. Positive edge triggered is indicated by a 'triangle' at the clock terminal of the flip-flop. Negative edge triggering is indicated by a 'triangle' with a bubble at the clock terminal of the flip-flop.

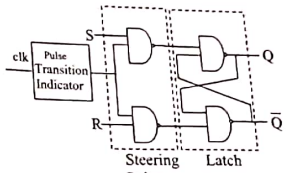
NOTE:

Edge triggering is also called 'dynamic triggering'.

The edge-Triggered S-R flip-flop:

Positive edge triggered

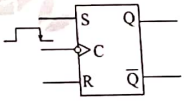




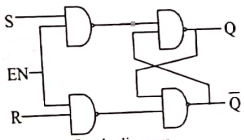
Truth Table

Clock	S	R	Q_n	Q_{n+1}	State
↑	0	0	0	0	No change
↑	0	0	1	1	
↑	0	1	0	0	Reset
↑	0	1	1	0	
↑	1	0	0	1	Set
↑	1	0	1	1	
↑	1	1	0	X	Indeterminate
↑	1	1	1	X	
0	X	X	0	0	No change
0	X	X	1	1	

At the truth table negative edge triggered



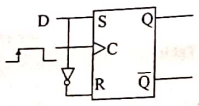
The edge detector generates a positive spike at the positive going (or) negative going edge of the clock pulse



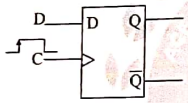
Logic diagram

The edge triggered D-flip-flop:

The edge triggered D-flip-flop has only one input terminal. The D flip-flop may be obtained from an S-R flip-flop by just putting one inverter between the 'S' and 'R'. The operation of D flip-flop is very simple. The output 'Q' will go to the same state that is present on the D input at the positive going transition of the clock pulse.



D flip-flop from S-R flip-flop

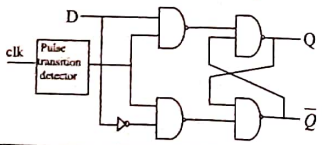


Logic symbol

Truth Table:

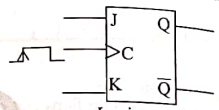
Clock	D	Q_n	Q_{n+1}	State
↑	0	0	0	Reset
↑	0	1	0	
↑	1	0	1	Set
↑	1	1	1	
0	X	0	0	No change
0	X	1	1	

The circuit diagram of edge-triggered D FLIP FLOP:-



The edge-triggered J-K Flip-Flop:

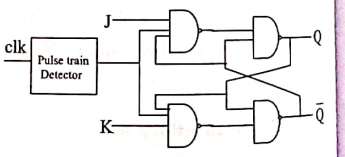
The JK Flip-flop is very versatile and also the most widely used .The J and K designations for the synchronous control inputs have no known significance. The functioning of the J-K FLIP FLOP is identical to that of the S-R FLIP FLOP except that it has no invalid states like that of the S-R Flip- flop. The logic symbol and the truth table of a positive edge-triggered J-K FLIP FLOP are shown.



Logic symbol

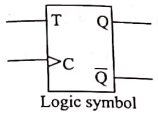
Clock	J	K	Q_n	Q_{n+1}	State
↑	0	0	0	0	No change
↑	0	0	1	1	
↑	0	1	0	0	Reset
↑	0	1	1	0	
↑	1	0	0	1	Set
↑	1	0	1	1	
↑	1	1	0	1	Toggle
↑	1	1	1	0	
0	X	X	0	0	No change
0	X	X	1	1	

Circuit diagram of Edge-Triggered FLIPFLOP:

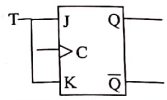


The edge triggered T FLIP FLOP:

The 'T' flip-flop has a single control input, labeled T for toggle. When 'T' is HIGH, the Flip-flop toggles on every new clock pulse. When 'T' is Low the flip-flop remains in what ever states it was before.



Logic symbol



T FLIP FLOP from JK FLIP FLOP

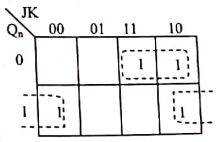
Clock	T	Q_n	Q_{n+1}	State
↑	0	0	0	No change
↑	0	1	1	
↑	1	0	1	Toggle
↑	1	1	0	
0	X	0	0	No change
0	X	1	1	

Characteristic equations:

JK Flip-Flop:-

Present State	Inputs		Next State
Q_n	J	K	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

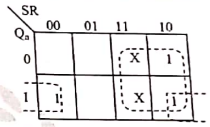
By using K-map



$Q_{n+1} = \overline{Q_n}J + Q_n\overline{K}$

SR flip-flop:

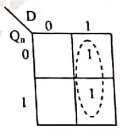
Present State	Inputs		Next State
Q_n	S	R	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
1	0	0	1
1	0	1	0
1	1	0	1



$Q_{n+1} = S + Q_n\overline{R}$

D Flip-flop:

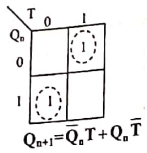
Present states	Input	Next state
Q_n	D	Q_{n+1}
0	0	0
0	1	1
1	0	0
1	1	1



$Q_{n+1}=D$

T flip-flop:

Present state	Input	Next state
Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0



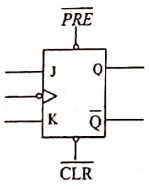
$Q_{n+1} = \bar{Q}_n T + Q_n \bar{T}$

Asynchronous Inputs:-
For the clocked Flip-flops the S-R, D, J-K inputs are called synchronous inputs, because their effect on the flip-flop output is synchronized with the clock input. The asynchronous inputs affect the flip-flop output independently of the synchronous inputs and the clock input. These asynchronous inputs are used to SET (or) RESET the flip-flop at any time regardless of the conditions at other inputs. They are normally labeled as preset (PRE) (or) direct SET (SD) (or) DC SET, and CLEAR (CLR)(or) Direct-RESET (RD) (or) DC CLEAR.

JK FLIP FLOP with active-Low PRESET and CLEAR inputs

Truth Table

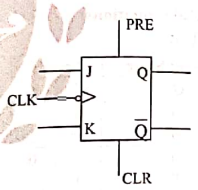
$\overline{PRE}$	$\overline{CLR}$	FF response
0	0	Not used
1	0	Q=0
0	1	Q=1
1	1	Clocked operation

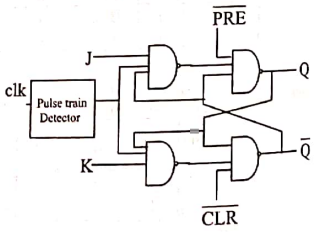


Logic diagram of a basic JK FLIP FLOp with Active High PRESET & CLEAR

PRE	CLR	FF response
0	0	Clocked operation
0	1	Q=0
1	0	Q=1
1	1	Not used

Flip-flop	Characteristic equation
D	$Q_{n+1} = D$
J-K	$Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$
T	$Q_{n+1} = T\bar{Q}_n + \bar{T}Q_n$
S-R	$Q_{n+1} = S + \bar{R}Q_n$

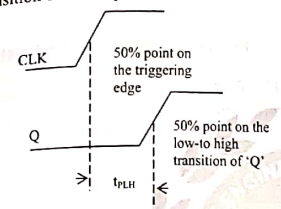




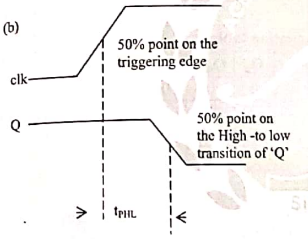
Flip-flop operating characteristics:
Propagation delay time (t_{Pd}):-

- Propagation delay t_{PLH} measured from the triggering of the clock pulse to the Low to HIGH transition of the output.
- Propagation delay t_{PHL} measured from the triggering of the clock pulse to the High-Low transition of the output.

(a)



(b)



Propagation delays t_{PLH} and t_{PHL} with respect to clk

Setup time: The setup time (t_s) is the minimum time for which the control level used to be maintained constant on the input terminals of the flip-flops, prior to the arrival of the triggering edge of the clock pulse, in order to enable the flip-flop to respond reliably.

Hold time (t_h):-The hold time is the minimum time for which the control signal need to be maintained constant at the input terminal of the flip-flop, after the triggering edge of the clock pulse, in order to enable the flip-flop to respond reliably.

Maximum clock frequency:-The maximum clock frequency (f_{max}) is the highest frequency at which a flip-flop can be reliably triggered. If the clock frequency is above this maximum the flip-flop would be unable to respond quickly enough and its operation will be unreliable

Power dissipation:- The power dissipation of a flip-flop is the total power consumption of the device. It is equal to the product of the supply voltage (V_{cc}) and the current (I_{cc}) drawn from the supply by it. $P = V_{cc} . I_{cc}$

If a system having 'N Flipflop's', total Power Dissipation $P_{tot} = N . V_{cc} * I_{cc} = (N.P)$ mW

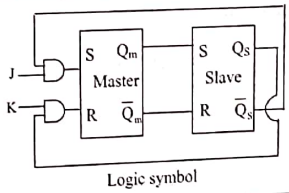
Clock skew and time race:

Clock skew:-The clock signal which is applied simultaneously to the flip-flop in a synchronous system may undergo varying degrees of delay caused by wiring between components, and arrive at the clock inputs of different flip-flops at different times. The delay is called clock skew.

Race around condition:- It is occurred when flip flop output is toggled more than once within one clock cycle completion.

Master slave JK FLIP FLOP:- The race around problem is eliminated in master slave FLIP FLOP, because while the clock drives the master, the inverted clock drives the slave

.



Logic symbol

Flip-flop Excitation Tables:

For the design of sequential clocks we should know the excitation tables of flip-flops. The excitation table of flip-flop can be obtained from its truth table. It indicates the inputs remained to be applied to the flip-flop to take it from the present state to next state.

S-R Flip-flop:

Truth Table:

S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	X

Excitation tables:

PS	NS	Required Inputs	
Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

J-K flip-flop:

Truth Table:

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

Excitation tables:-

PS	NS	Required Inputs	
Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

PS – Present State, NS - Next State

D Flip-flop:

Truth Table:

D	Q_{n+1}
0	0
1	1

Excitation tables:

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

T-Flip-flop:

Truth Table:

T	Q_{n+1}
0	Q_n
1	$\overline{Q}_n$

Excitation tables:

PS	NS	Required input
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Conversion of flip-flops:

To convert one type of flip-flop into another type, a combinational circuit is designed such that if the inputs of the required flip-flop (along with the output's of the actual flip-flop is required) are fed as inputs to the combinational ckt and the output of the combinational ckt is connected to the inputs of the actual flip-flop, then the output of the actual flip-flop is the required flip-flop.

S-R FLIP FLOP to J-K FLIP FLOP:

Here the external inputs to the already available S-R Flip-flop will be J and K.

S and R are the output of the combinational circuit , which are also the actual inputs to the S-R FLIP FLOP we write a truth table with J,K,Q_n,Q_{n+1},S,R.where Q_n is the Present state of the flip-flop and Q_{n+1} is the next state obtained Where the particular J&k inputs are applied .i.e. Q_n denoted the states of the flip-flop before the application of the inputs and Q_{n+1} refers to the states obtained by the flip-flop after the application of inputs .J, K, Q_n have eight combinations for each combination of J, K, Q_n find the corresponding Q_{n+1}.i.e. determine to which next state the J K flip flop will go from the present state if the present inputs J&K are applied. Now complete the table by writing the values of S&R required to get each Q_{n+1} from the corresponding Q_n.

External inputs		Present state	Next state	Flip-flop inputs	
J	K	Q_n	Q_{n+1}	S	R
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	X	0
1	1	0	1	1	0
1	1	1	0	0	1

For 'S'

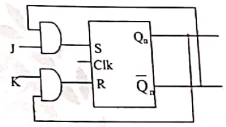
J \ KQ_n	00	01	11	10
0	0	X	0	0
1	1	X	0	1

$S = J\overline{Q_n}$

For 'R'

J \ KQ_n	00	01	11	10
0	X	0	1	X
1			1	0

$R = KQ_n$



Logic diagram

Applications of flip-flops:

- Parallel data storage
- Serial data storage
- Transfer of data
- Serial to parallel conversion
- parallel to serial conversion
- Counting
- Frequency division

Shift register:-

A flip-flop can store one bit of data, a '0' (or) '1' it is referred to as a single bit register. When more bits of data are to be stored, more number of flip-flops are used .A register is a set of flip-flops used to store binary data. Loading may be in serial (or) parallel form. In serial loading data is transferred into the register in serial form i.e. one bit at a time ,Where as in parallel loading the data is transferred into the register in parallel form meaning that all the flip-flops are triggered into their new states at the same time. The output data either in serial/ parallel form.

NOTE:
Shift registers are a type of circuits closely related to counters. They are used basically for the storage and transfer of digital data. The basic difference between a shift register and a counter is that a shift register has no specific sequence of states except in certain very specialized applications. where as a counter has a specified sequence of states.

Applications:
- Registers are used to momentarily store binary information appearing at the output of an encoding matrix.
- A register might be used to accept input data from an alpha-numeric key board and then present data at the input of a micro processor chip.
- Shift register used to momentarily store binary data at the output of decoder.
- Binary complementation, multiplication, division
- Time delay's
- Serial/parallel data conversion
- Ring counter
- Universal asynchronous receiver transmitter (UART)

Universal shift register:-
A register which is able to shift the information from left to right (or) from right to left and which can perform all four operations is called universal shift register.

Examples

01. Left shift operation is nothing but multiplied by'2'.

Q_3	Q_2	Q_1	Q_0	
0	1	0	1	=5
1	0	1	0	=10

Shift left by n-positions is equivalent to multiplication by 2^n

02. If LSB = 0, then right shift operation by one is same as division by 2.

Q_3	Q_2	Q_1	Q_0	
1	0	1	0	= 10
0	1	0	1	= 5

03. If LSB = 1, then right shift operation gives integer division by '2'.

Q_3	Q_2	Q_1	Q_0	
0	1	0	1	=5
0	0	1	0	=2(instead of 2.5)

Ring Counter:
Shift register can be used as ring counter when 'Q_n' output terminal is connected to serial input terminal. An 'n' bit ring counter can have 'n' different output states. It can count n-clock pulses.

Twisted ring counter (Johnson counter):
- It is formed when $\overline{Q_n}$ output terminal is connected to the serial input terminal of shift register.
- An 'n' bit Johnson counter can have maximum of 2n different output states.

Counter:-
A digital counter is set of flip-flops whose states change in response to pulse applied at the input to the counter. The output is the binary equivalent of the total number of pulses that have occurred up to that time. Thus as its name implies, a counter is used to count pulses. A counter can also be used as a frequency divider.

Counter are two types:
- Synchronous
- Asynchronous

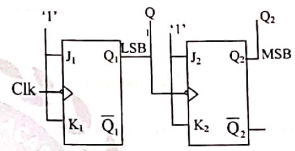
Asynchronous(ripple)	Synchronous
1. The flip-flops are connected in such a way that the output of first FLIP FLOP drives the clock for the second FLIP FLOP, the output of the second drives the third and so on.	1.There is no connection between the output of first FLIP FLOP and clock input of next FF and so on.
2. All the FLIP FLOPS are not clocked simultaneously.	2. All the FLIP FLOPS are clocked simultaneously.
3. Design and implementation very simple even for more number of states.	3. Design becomes difficult if the number of states increased.
4. Low speed	4. High speed

Note:
A counter may be up counter (or) a down counter. An up-counter is a counter which counts in the upwards direction(0,1,2,3,……,N). A down counter is a counter which counts in downwards direction. i.e.N, N-1,……,1,0. Each of these counts is called the state of the counter. The number of states through which the counter passes before returning to the starting states is called modulus of the counter. In other words, the number of input pulses that causes the counter to reset to its initial count is called the modulus of the counter.

- A N-bit counter will have N flip-flops and 2^N states and divides the input frequency by 2^N.Hence it is a divide by 2^N counter.
- A counter may have a shortened modulus. This type of counter does not utilize all the possible states. Some of the states are unutilized .i.e. invalid. The number of flip-flops required to construct a mod-N counter equal the smallest 'n' for which $N \leq 2^n$.

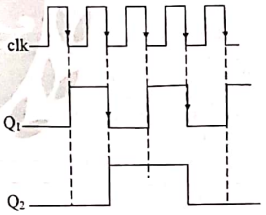
Lock Out:-
In shortened modulus counter, there may occur the problem of lock-out. Some times when the counter is switched on (or) any time during the counting the counter may find itself in some unused states. A counter whose unused states have this feature is said to suffer from the problem of lock out.

Two Bit ripple up counter with negative edge triggered flipflops:

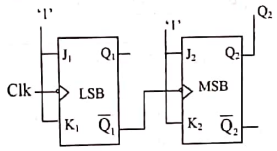


Logic symbol

Timing diagram:

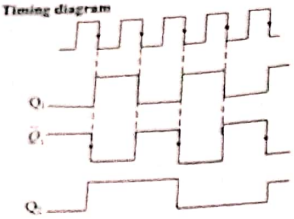


Two bit Ripple down counter using negative edge triggered flip-flop:



Logic symbol

Timing diagram



Examples

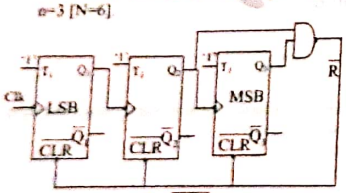
Design of Asynchronous counter:

01. ***Design a mod-6 asynchronous counter using T flip-flops.***

Sol:

A mod-6 counter has six stable states 000,001,010,011,100,101. Where the sixth clock pulse is applied, the counter temporarily goes to 110 states, but immediately resets to 000 because of feed back provided.

It requires 3 flip-flops, because the smallest value of 'n' satisfying the condition $N \le 2^n$ is $n=3$ [N=6].



$R = Q_3Q_2$, $\overline{R} = \overline{Q_3Q_2}$

R= 0 for 000 to101,

R=1 for 110 and R=X for 111.

$\overline{R}$ = 1 for 000 to 101,

$\overline{R}$ = 0 for 110 to 110 and R = X for 111.

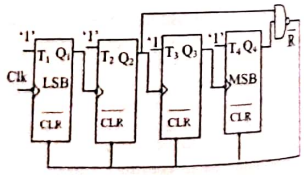
Truth Table:

No. of Clk pulse	States Q_3	Q_2	Q_1	$R=Q_3Q_2$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	0
6	1 ↓	1 ↓	0 ↓	1 ↓
	0	0	0	0
7	0	0	1	0

02. ***Mod-10 asynchronous counter;-R = Q_3Q_2 R=0 for 0000 to 1001, R=1 for 1010, R = 1 for 1011 to 1111***

Truth table:

After pulses	Count			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	0	0	0	0



Design of synchronous counter:

Step1: Number of flip-flops.
The required number of 'n' Flip-flops, the smallest value of 'n' is such that the number of states $N \le 2^n$.

Step2: Choice of flip-flops and excitation table:-
Select the type of flip-flop to be used and write the excitation table. An excitation table is a table that lists the present states (PS), the next states (NS) and the required excitations.

Step3: Minimum expressions for Excitations:-
Using K-maps obtain expression in terms of present states and inputs.

Step4: Logic diagram:-
Draw a logic diagram based on minimal expression.

03. ***Design a synchronous modulo-6 gray code counter using T flip-flops.***

Sol:

Step1: The number of flip-flops:-
The counting sequence for a modulo-6 Gray code counter is 000,001,011, 010,110,111.
It requires n=3 FF's ($N \le 2^n$,N=6),101,100 are invalid ,i.e. don't cares.

PS			NS			Required Excitations		
Q_3	Q_2	Q_1	Q_3	Q_2	Q_1	T_3	T_2	T_1
0	0	0	0	0	1	0	0	1
0	0	1	0	1	1	0	1	0
0	1	1	0	1	0	0	0	1
0	1	0	1	1	0	1	0	0
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1
1	0	0	UN USED STATES			X	X	X
1	0	1				X	X	X

Step-2: Type of Flip-flops & Excitation states

Q_3: MSB Excitation table

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

Step3:

T_3

Q_3 \ Q_2Q_1	00	01	11	10
0				1
1	x	x	1	

$T_3 = Q_3\overline{Q}_1 + \overline{Q}_3Q_2\overline{Q}_1$

T_2

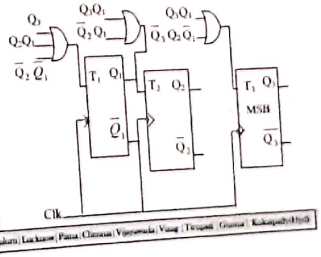
Q_2 \ Q_3Q_1	00	01	11	10
0		1		
1	x	x	1	

$T_2 = Q_3Q_1 + Q_2Q_1$

T_1

Q_1 \ Q_3Q_2	00	01	11	10
0	1		1	
1	x	x	1	1

$T_1 = Q_3 + Q_2Q_1 + \overline{Q}_2\overline{Q}_1$



04. ***Design a synchronous BCD counter using J-K FLIP FLOP's.***

Step1:-Number of flip-flops:-

It is a mod-10 counter (0000 to1001)

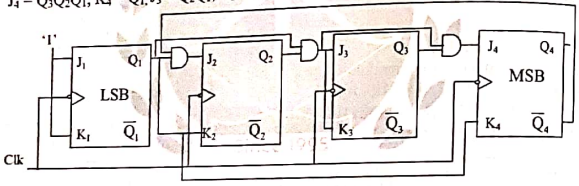
It Requires n=4 FLIP FLOPS ($N \le 2^n, 10 \le 2^n$).1010 to 1111 are don't cares.

PS				NS				Required Excitations							
Q_4	Q_3	Q_2	Q_1	Q_4	Q_3	Q_2	Q_1	J_4	K_4	J_3	K_3	J_2	K_2	J_1	K_1
0	0	0	0	0	0	0	1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	X	0	1	X
0	0	1	0	0	0	1	1	0	X	0	X	X	1	X	1
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	X	1
1	0	0	1	0	0	0	0	X	1	0	X	0	X	X	1

Excitation table

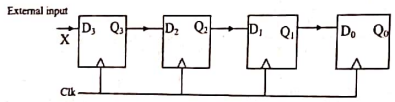
Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

$J_4 = Q_3Q_2Q_1$, $K_4 = Q_1$, $J_3 = Q_2Q_1$, $K_3 = Q_2Q_1$, $J_2 = \overline{Q}_4Q_1$, $K_2 = Q_1$, $J_1 = 1$ $K_1 = 1$

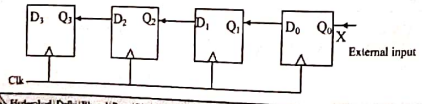


Design of shift register counter:

Right shift register:

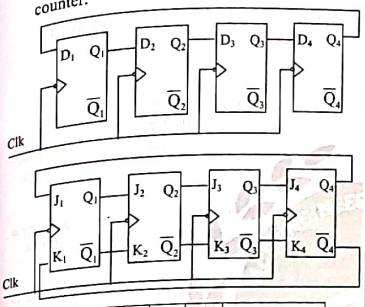


Left shift register:



Ring counter:

In this the flip flops are arranged as in a normal shift register but the output (Q) of the last Flip-flop is connected back to the input of first flip flop such that the array of flip-flops is arranged in a ring, therefore the name ring counter.

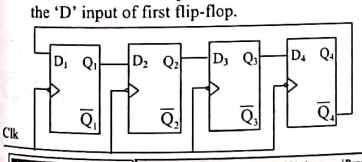


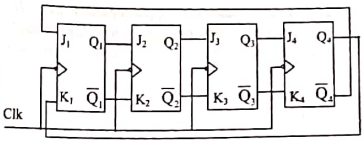
Q_1	Q_2	Q_3	Q_4	After clock pulse
1	0	0	0	0
0	1	0	0	1
0	0	1	0	2
0	0	0	1	3
1	0	0	0	4
0	1	0	0	5
0	0	1	0	6
0	0	0	1	7

Note:- An 'n' bit ring counter can count 'n' states.

Twisted ring counter or Johnson counter:-

This counter is obtained from a serial-in , serial-out shift register by providing feedback from the inverted output of the last flip-flop to the 'D' input of first flip-flop.



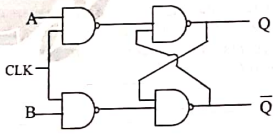


After clock pulse	Q_1	Q_2	Q_3	Q_4
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1
8	0	0	0	0

Note: A 'N' bit Johnson counter can count '2N' states.

Examples

01. ***The circuit given below is a***



(a) J-K Flip-flop
(b) Johnson's counter.
(c) R-S Flipflop
(d) None of above.

Sol: The circuit is S-R Flipflop

Truth table of SR Flipflop

S	R	Q
0	0	No change
0	1	0
1	0	1
1	1	*

02. *A 4 bit modulo-16 ripple counter uses JK flip-flops. If the propagation delay of each FF is 50ns, the maximum clock frequency that can be used is equal to:*

Sol: Propagation delay T_d of each FF = 50ns

4 bit modulo 16 ripple counter represents n= 4

Maximum clock frequency

$$f = \frac{1}{T_d \times n} = \frac{1}{50\times4\times10^{-9}} = 5\text{MHz}$$

03. *An S-R FLIP-FLOP can be converted into a T FLIP-FLOP by connecting _____ to Q and _____ to $\overline{Q}$.*

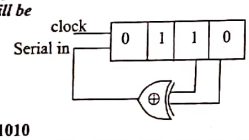
Sol: R, S

TQ_n	Q_{n+1}	S	R
0 0	0	0	×
0 1	1	×	0
1 0	1	1	0
1 1	0	0	1

SR flip flop can be converted into T flipflop by connecting

$S = T\overline{Q}_n$, $R = TQ_n$

04. *The initial contents of the 4-bit serial-in-parallel-out, right-shift, Shift Register shown in figure is 0110. After three clock pulses are applied, the contents of the Shift Register will be*



Ans: 1010

Sol: After three pulses the content of shift Register

$SI = Q_1 \oplus Q_0$

CLK	SI	Q_3	Q_2	Q_1	Q_0
		0	1	1	0
1	1	1	0	1	1
2	0	0	1	0	1
3	1	1	0	1	0

05. *A pulse train with a frequency of 1MHz is counted using a modulo-1024 ripple-counter built with J-K flip flops. For proper operation of the counter, the maximum permissible propagation delay per flip flop stage is _____ n Sec.*

Ans: 100n sec.

Sol: Given f = 1MHz → T = 1000n sec

Mod $1024 = 2^n$

$2^{10} = 2^n$

$\Rightarrow n = 10$

Propagation delay for one flip flop

$$T_d = \frac{T}{n} = \frac{1000\text{n sec}}{10}$$

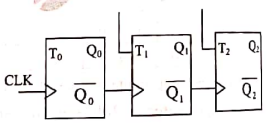
T_d = 100nsec

06. *A switch-tail ring counter is made by using a single D flip-flop. The resulting circuit is a*

Ans: T flip-flop

Sol: In a switch tail ring counter using single D FF, the complementary output '$\overline{Q}$' is connected to D, then it becomes T FF

07. *The given figure shows a ripple counter using positive edge triggered flip-flops. If the present state of the counter is $Q_2Q_1Q_0$=011, then its next state ($Q_2Q_1Q_0$) Will be*



(a) 010 (b) 100

(c) 111 (d) 101

Ans: (b)

Sol: Q_0 changes for every CLK

Q_1 changes if $\overline{Q}_0 \Rightarrow 0 \to 1$

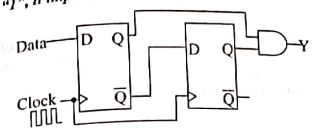
$Q_0 \Rightarrow 1 \to 0$

Q_2 changes if $\overline{Q}_1 \Rightarrow 0 \to 1$

$Q_1 \Rightarrow 1 \to 0$

CLK	Q_2 Q_1 Q_0
	0 0 0
1	0 0 1
	0 1 0
	0 1 1
	1 0 0
	1 0 1
	1 1 0
	1 1 1

08. *When the output Y in the circuit below is "1", it implies that data has*



(a) changed from "0" to "1"

(b) changed from "1" to "0"

(c) changed in either direction

(d) not changed

Ans: (a)

Sol: Y will be '1' if both the inputs of the AND gate are 1. It is possible when data input is changing from 0 to 1.

09. *The Characteristic equation of a level triggered T flip-flop, with T as input and Q as output is* *(JTO)*

Ans: $Q(n+1) = \overline{Q}T + Q\overline{T}$

Sol: from the following excitation table

Clock	Q	T	Q(n+1)
↑	0	0	0
↑	0	1	1
↑	1	0	1
↑	1	1	0

$\therefore Q(n+1) = \overline{Q}T + Q\overline{T}$

10. *If a counter having 10 FF's is initially at 0, what count will it hold after 2060 pulses?*

(a) 000 000 1100 (b) 000 001 1100

(c) 000 001 1000 (d) 000 000 1110

Ans: (a)

Sol: Given counter is having 10 FF's. Therefore it will count from $[0]_{10}$ – $[1023]_{10}$. So total it is having 1024 states

$\therefore$ 2060

-1024

1036

1036

-1024

12

i.e. value of counter after 2060 pulses is same as that after 12 pulses given as 1100.

Convert Decimal no. 12 into binary $(1100)_2$

so after 2060 pulses the count will be $(1100)_2$

11. *A certain JK FF has t_{pd} = 12 ns. The largest MOD counter that can be constructed from such FFs and still operate up to 10 MHz is*

(a) 16 (b) 256 (c) 8 (d) 128

Ans: (d)

Sol: From given information o/p frequency of counter

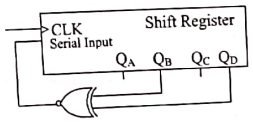
$$f_{out} = \frac{1}{n \times t_{pd}}$$

$$\Rightarrow 10\text{M} = \frac{1}{n\times12n}$$

$$n = \frac{1}{10\text{M}\times12n} = 8.33 \approx 8\text{bits}$$

Given counter can have 8 bits. It can have maximum count nos of 256. So we can construct MOD-256 counter with given information.

12. *A 4-bit serial-in-parallel-out shift register is used with a feedback as shown in Fig. The shifting sequence is $Q_A \to Q_B \to Q_C \to Q_D$.*



If the output is 0000 initially, the output repeats after

(a) 4 clock cycles (b) 6 clock cycles
(c) 15 clock cycles (d) 16 clock cycles

Ans: (b)

Sol: Initially the counter is having 0000

Clock	Q_A	Q_B	Q_C	Q_D	Serial In($Q_B \oplus Q_D$)
X	0	0	0	0	1
1st	1	0	0	0	1
2nd	1	1	0	0	0
3rd	0	1	1	0	0
4th	0	0	1	1	0
5th	0	0	0	1	0
6th	0	0	0	0	1

$\therefore$ After 6 clock cycles the input is repeated

Finite State Machines (FSM):

Sequential circuits are two types
(1) Mealy sequential circuit.
(2) Moore sequential circuit

(1) Mealy sequential circuits:

- In a mealy sequential circuits, the outputs are a function of the present state and the values of the inputs as shown in fig.

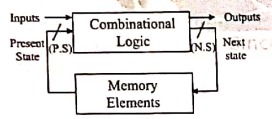


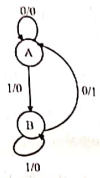
Fig: Mealy sequential circuit

- Accordingly, the outputs may change asynchronously in response to any change in the inputs.
- In order to synchronize, the inputs must be synchronized with clock and the outputs must be sampled immediately before the clock edge.
- Thus the output of the mealy machine is the value that is present immediately before the active edge of the clock

Examples

01. Implement a sequence detector circuit to detect the sequence '10' using Mealy sequential circuit?

Sol: State diagram:



State table:

Present State	Next State		Output (Z)	
	X = 0	X = 1	X = 0	X = 1
A	A	B	0	0
B	A	B	1	0

Mealy Circuit implementation using D-flipflop:

Let us assign A = 0 & B = 1

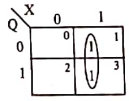
State transition and output table:

P.S (Q)	Input (X)	N.S (Q)	Output (Z)	Flip-flop I/P (D)
A = 0	0	A = 0	0	0
A = 0	1	B = 1	0	1
B = 1	0	A = 0	1	0
B = 1	1	B = 1	0	1

P.S = Present State, N.S = Next State

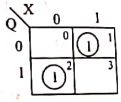
From the table $Z = Q\overline{X}$

K-map for D:



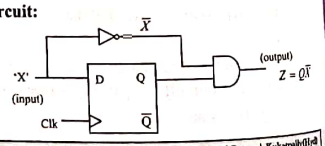
$\Rightarrow D = X$

K-map for Z:



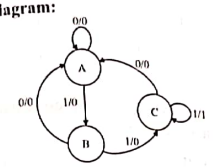
$\Rightarrow Z = Q\overline{X}$

Circuit:



02. Implement a sequence detector circuit to detect the sequence '111' using Mealy sequential circuit?

Sol: State diagram:



State table:

Present State	Next State		Output (Z)	
	X = 0	X = 1	X = 0	X = 1
A	A	B	0	0
B	A	C	0	0
C	A	C	0	1

Mealy circuit implementation using D-flipflop:

Let us assign state A = 00, B = 01 & C = 10.

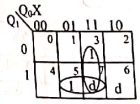
State transition and output table:

P.S Q_1Q_0	Input X	N.S Q_1Q_0	Output Z	Flip-flop I/P D_1 D_0
A = 00	0	A = 00	0	0 0
A = 00	1	B = 01	0	0 1
B = 01	0	A = 00	0	0 0
B = 01	1	C = 10	0	1 0
C = 10	0	A = 00	0	0 0
C = 10	1	C = 10	1	1 0

P.S = Present State, N.S = Next State

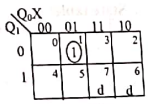
$Q_1Q_0 = 11$ is an unused state which is used as don't care in minimization process.

K-map for D_1:



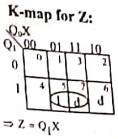
$\Rightarrow D_1 = Q_1X + Q_0X$
$= (Q_1 + Q_0)X$

K-map for D_0:



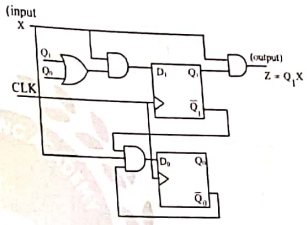
$\Rightarrow D_0 = \overline{Q}_1\overline{Q}_0X$

K-map for Z:



$\Rightarrow Z = Q_1X$

Circuit:



(2) Moore Sequential Circuits:

- In a Moore sequential circuit the outputs depend only on the present state as shown in figure

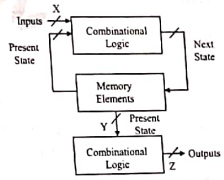


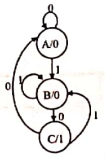
Fig: Moore sequential circuit

- The outputs change synchronously with the state transition triggered by the active clock edge.
- Because they depend only on flip-flop outputs that are synchronized with clock

Examples

01. Implement a sequence detector circuit to detect the sequence '10' using Moore sequential circuit?

Sol: State diagram:



State table:

Present State	Next State		Output	
	X = 0	X = 1	X = 0	X = 1
A	A	B	0	0
B	C	B	0	0
C	A	B	1	1

Moore circuit implementation using D-flipflop:

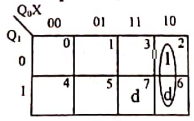
Let us assign A = 00, B = 01 & C = 10

State transistion and output table:

P.S Q_1Q_0	Input X	N.S Q_1Q_0	Output Z	Flip-flop I/P D_1 D_0
A = 00	0	A = 00	0	0 0
A = 00	1	B = 01	0	0 1
B = 01	0	C =10	0	1 0
B = 01	1	B = 01	0	0 1
C = 10	0	A = 00	1	0 0
C = 10	1	C = 01	1	0 1

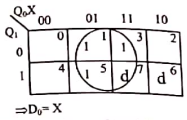
$Q_1Q_0 = 1$ is an unused state which is used as don't care in the minimization process.

K-map for D_1:



$D_1 = Q_0\overline{X}$

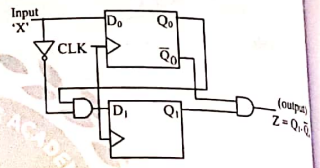
K-map for D_0:



$\Rightarrow D_0 = X$

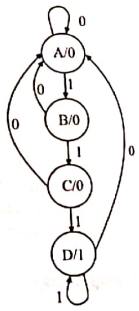
'Z' is function of P.S (present state) only then from table $Z = Q_1.\overline{Q}_0$

Circuit:



02. Implement a sequence detector circuit to detect the sequence '111' using Moore sequential circuit?

Sol: State diagram:



State table:

Present state	Next state		Output (Z)	
	X = 0	X = 1	X = 0	X = 1
A	A	B	0	0
B	A	C	0	0
C	A	D	0	0
D	A	D	1	1

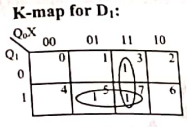
Moore circuit implementation using D-flipflop:

Let us assign A = 00, B = 01, C = 10 & D = 11

State transistion and output table:

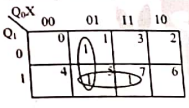
P.S Q_1Q_0	Input X	N.S Q_1Q_0	Output Z	F.F I/P D_1 D_0
A = 0 0	0	A = 0 0	0	0 0
A = 0 0	1	B = 0 1	0	0 1
B = 0 1	0	A = 0 0	0	0 0
B = 0 1	1	C = 1 0	0	1 0
C = 1 0	0	A = 0 0	0	0 0
C = 1 0	1	D = 1 1	0	1 1
D = 1 1	0	A = 0 0	1	0 0
D = 1 1	1	D = 1 1	1	1 1

K-map for D_1:



$\Rightarrow D_1 = Q_1X + Q_0X$
$= (Q_1 + Q_0)X$

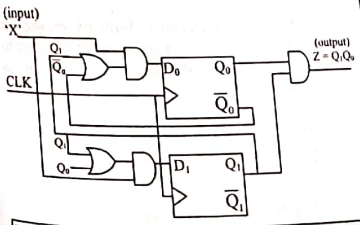
K-map for D_0:



$\Rightarrow D_0 = \overline{Q}_0X + Q_1X$
$= (\overline{Q}_0 + Q_1)X$

'Z' is function of present state only then from the table $Z = Q_1\,Q_0$

Circuit:-

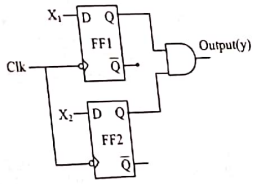


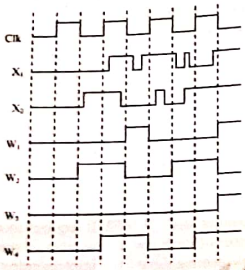
Comparision between Melay and Moore Sequential circuits:

Mealy sequential circuit	Moore sequential circuit
1. Its output is a function of present state and present input	1. Its outputs are a function of present state only
2. Input changes may effect the output of the circuit	2. Input change doesn't effect the output
3. It requires less number of states for implementing same function	3. It requires more number of states for implementing same function
4. Mealy sequential circuits are less safer	4. Moore sequential circuits are more safer.

Class Room Practice Questions

01. In the circuit shown, choose timing the correct timing diagram of the output (y) from the given waveforms W1, W2, W3 and W4.

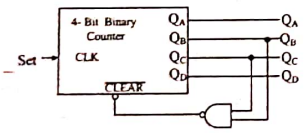




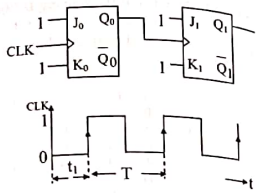
(a) W1 (b) W2
(c) W3 (d) W4

02. We want to design a synchronous counter that counts the sequence 0-1-0-2-0-3 and then repeats. The minimum number of J-K flip-flops required to implement this counter is _____.

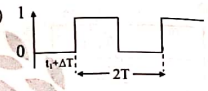
03. A mod-n counter using asynchronous binary up-counter with synchronous clear input is shown in the figure. The value of n is _____



04. For each of the positive edge-triggered J-K flip flop used in the following figure, the propagation delay is ΔT

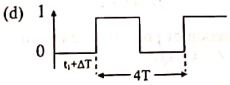


Which of the following waveforms correctly represents the output at Q_1?

(a) 

(b) 

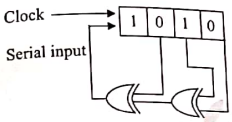
(c) 

(d) 

05. Let $k = 2^n$. A circuit is built by giving the output of an n-bit binary counter as input to an n-to-2^n bit decoder. This circuit is equivalent to a

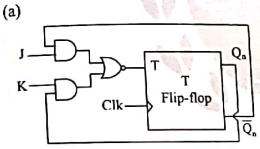
(a) k-bit binary up counter.
(b) k-bit binary down counter.
(c) k-bit ring counter.
(d) k-bit Johnson counter.

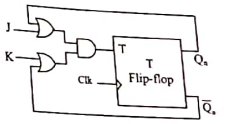
06. The shift register shown in fig. is initially loaded with the bit pattern 1010. Subsequently the shift register is clocked, and with each clock pulse the pattern gets shifted by one bit position to the right. With each shift, the bit at the serial input is pushed to the left most position (MSB). After how may clock pulses will the content of the shift register become 1010 again?

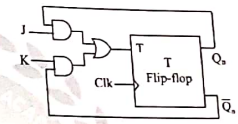


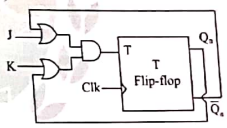
(a) 3 (b) 7
(c) 11 (d) 15

07. A JK flip flop can be implemented by T flip-flops. Identify the correct implementation.

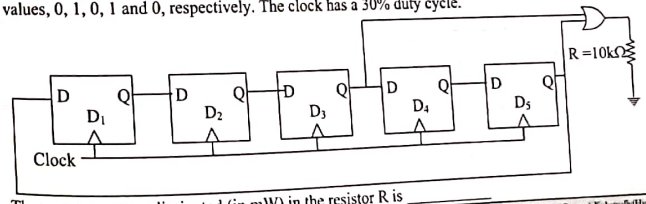
(a) 

(b) 

(c) 

(d) 

08. Assume that all the digital gates in the circuit shown in the figure are ideal, the resistor R = 10 kΩ and the supply voltage is 5V. The D flip-flops D_1, D_2 D_3, D_4 and D_5 are initialized with logic values, 0, 1, 0, 1 and 0, respectively. The clock has a 30% duty cycle.



The average power dissipated (in mW) in the resistor R is _____

09. Find the number of states after reducing the following state table

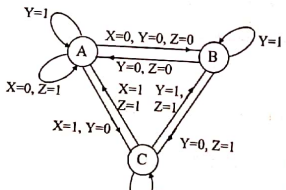
Present State	Next State X = 0	Next State X = 1	Output (Y) X = 0	Output (Y) X = 1
A	A	E	0	0
B	C	A	1	0
C	B	A	1	0
D	A	B	0	1
E	A	C	0	1

(a) 4 (b) 3
(c) 2 (d) 5

10. Determine the function of the following FSM,

* given 'S_0' is the initial state
(a) 2's complement of a binary input
(b) sequence detector of sequence 110
(c) 1's complement of a binary input
(d) none of the above

11. The state transition diagram for a finite state machine with states A, B and C, and binary input X, Y and Z, is shown in the figure.



Which one of the following statements is correct?
(a) Transitions from State A are ambiguously defined
(b) Transition from State B are ambiguously defined
(c) Transitions from State C are ambiguously defined
(d) All of the state transitions are defined unambiguously.

Key for CRPQ

01. (c)	02. 4	03. 7	04. (b)	05. (c)
06. (b)	07. (b)	08. 1.5	09. (b)	10. (c)
			11. (c)	

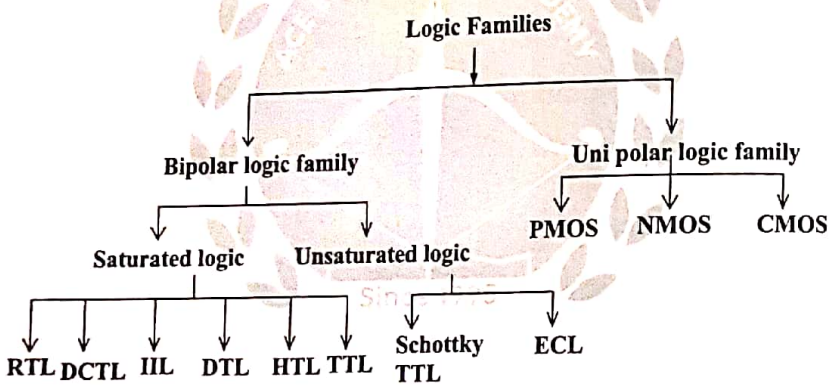
6 Logic Gate Families (Additional topic for ESE)

Digital IC gates are classified not only by their logic operation, but also by specific logic circuit family to which they belong. Each logic family has its own basic electronic circuit upon which more complex digital circuits and functions are developed.

Positive and negative logic system:-

Signal status	Positive logic	Negative logic
HIGH	'1'	'0'
LOW	'0'	'1'

In positive logic system binary '1' equivalent to more positive of the two voltages.
In negative logic system binary'1' equivalent to more negative of the two voltages.



Bipolar saturated logic families:

In this mode, the transistors have to be driven into and out of saturation, so propagation delay is more

1. Resistor Transistor Logic (RTL)
2. Direct Coupled Transistor Logic (DCTL)
3. Integrated Injection Logic (I^2L)
4. Diode Transistor Logic (DTL)
5. High Threshold Logic (HTL)
6. Transistor Transistor Logic (TTL)

Bipolar unsaturated logic families:-

In this, saturation is not occurred hence propagation delay is reduced.

1. Schottky TTL (STTL)
2. Emitter Coupled Logic (ECL)

Uni-polar logic family:

In this, uni-polar transistors are used

1. PMOS
2. NMOS
3. Complementary MOS (CMOS)

Note:

1. HTL is a modified form of DTL and IIL is a modified form of DCTL.
2. The MOS and IIL logic families are used for large scale Integration functions because of high package density.
3. TTL,ECL,CMOS are used for medium scale integration (MSI) (or) small scale integration(SSI).
4. Each logic gate family is identified with a serial number .

Example:

TTL family ICS are available in 74/54 series.
CMOS IC's usually designated with 4000 series.
ECL family with 10,000 series.

Characteristics of logic families:

1. **Fan-out:** The number of standard loads that the output of the gate can drive without disturbing its normal operation.
2. **Fan-in:** The maximum number of inputs that can be applied to the logic gate.
3. **Power dissipation:-**The power consumed per gate.
4. **Propagation delay:** The average transition delay time for the signal to propagate from input to output where the signal changes in value.
5. **Noise margin:** It is defined as the limit of a noise voltage which may be present without impairing the proper operation of the circuit. In digital logic circuit, the binary 0&1 are represented by a pair of voltage levels. Each logic family has a different standard voltage level but in practice it is difficult to get this ideal voltage.

V_{OH}(min):

It is the minimum voltage level at a logic circuit output in the logic '1'state under defined load conditions.

V_{IH}(min):

It is the minimum voltage level required for a logic '1' at input. Any voltage below this level will not be accepted as a 'High' by the logic circuit.

V_{NH}:

It is defined as the difference between the lowest possible high output V_{OH}(min) and minimum voltage V_{IH}(min) required for high output.

$$V_{NH} = V_{OH}(min) - V_{IH}(min)$$

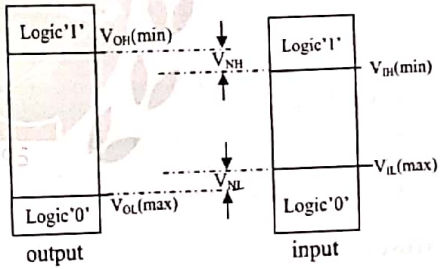
V_{IL}(max):

It is defined as the maximum voltage level required for a logic '0'at input. Any voltage above this level will not be accepted as a 'Low' by the logic circuit.

V_{OL}(max):

It is maximum voltage level at a logic circuit output in the logical'0' state under defined load conditions.

V_{NL}: $V_{NL} = V_{IL}(max) - V_{OL}(max)$



Note: 1.$V_{OH}(min) > V_{IH}(min)$
2. $V_{IL}(max) > V_{OL}(max)$

Figure of merit:-

The product of propagation delay time and power dissipation is known as figure of merit. Normally minimum values are desired.

Logic swing:

The difference between the two output voltages (V_{OH}-V_{OL}) is known as the logic swing of the circuit.

Noise immunity:
The ability to with stand variations in the input levels.

NOTE:

- ECL has ultra-fast-switching speed& low Logic swing.
- The temperature range of 74-series of TTL logic gate family is 0°C to70°C.This series of ICS used for commercial applications.
- The temperature range of 54-series TTL logic gate family is -55°C to125°C. This series of ICS used in military applications.

Breadth:

- The number of various functions available in a logic family is known as breadth of the logic family.

Wired logic:

- Where the output of logic gates is connected together to perform additional logic functions. This is known as wired logic.

Passive pull-up:

- In a bipolar logic circuit a resistance 'R_C' used in the collector circuit of the output transistor is known as passive pull-up.

Active pull up:

- In a bipolar logic circuit, a BJT and diode circuit used in the collector circuit of the output transistor instead of 'R_c' is known as active pull-up. This facility is available in TTL family.

NOTE:
The advantage of active pull up over passive pull-up increases speed of operation and reduces power dissipation.

Open collector output:

- In a bipolar logic circuit, if nothing is connected at the collector of the output transistor and this collector terminal is available to all IC pins known as open collector output.

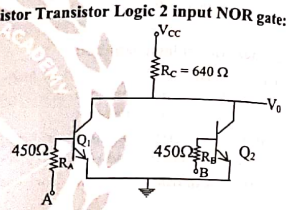
Tri-state logic:

- In the Tri-state logic, in addition to 0 and 1 there is a third state known as high impedance state where the gate is disabled, it is the third state.

Resistor Transistor Logic:

- It uses discrete components like resistors & transistors.

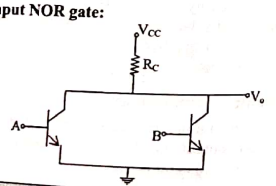
Resistor Transistor Logic 2 input NOR gate:



Note:
An n input NOR gate uses 'n' number of transistors.

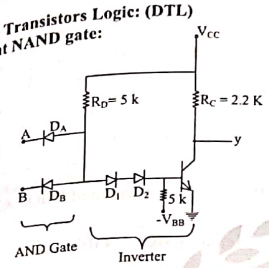
Direct Coupled Transistor logic (DCTL)
In DCTL the inputs are directly connected to bases of transistors. It can be defined as the RTL logic without base resistance.

2-input NOR gate:



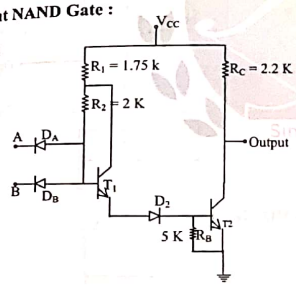
Note:
An 'N" input NOR gate uses 'N' transistors.

Diode Transistors Logic: (DTL)
2 input NAND gate:

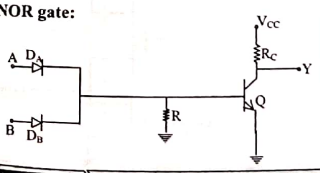


Modified DTL gate:
To increase the fan-out of DTL gate, the base current of T_2 has to be increased. It is increased by replacing D_1 by a transistor., T_1.

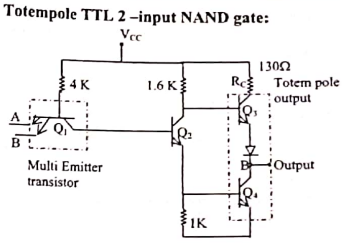
2 Input NAND Gate :



DTL NOR gate:



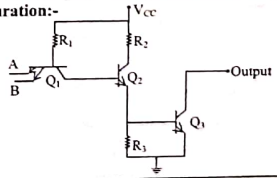
Totempole TTL 2 –input NAND gate:



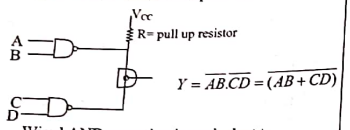
Note:

- In TTL logic gate family, three different types of output configurations are available, they are open collector and totem pole and tri-state
- The advantage of open collector TTL is wired- logic can be performed and loads other than the normal gate can be used.
- If any input of TTL circuit is left floating it will function as it is connected to logic 1 level.
- The supply voltage range of 74-series is $5 \pm 0.25V$,
  54-series is $5 \pm 0.5V$
- Different versions available in TTL logic gate family.
  74/54L -Low power
  74/54H -High power/High speed
  74/54LS -Low power Schottky
  74/54S -Schottky
  74/54AS -Advanced Schottky
  74/54ALS–Advanced low power Schottky

2-Input NAND gate with open collector output configuration:-

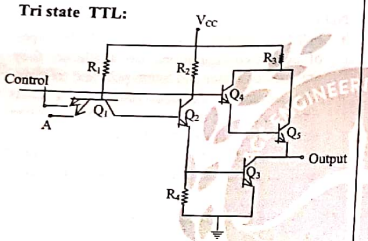


External Pull up resistor should be used in the circuit. Gates with open collector output can be used for wired AND operation



Wired AND operation is equivalent to AND-OR-INVERT.

Tri state TTL:

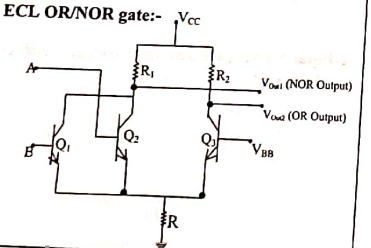


Current Mode Logic (CML)(or)Emitter Coupled Logic:

In ECL, the transistors are not saturated, there by increasing the overall switching speed of the logic gate. In ECL the voltage levels are

'0' = - 1.8 V
'1' = - 0.8 V

ECL OR/NOR gate:-

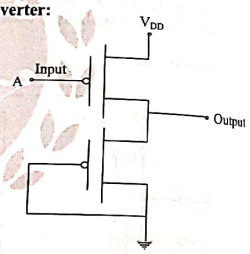


A	B	Q_1	Q_2	Q_3	OUT1	OUT_2
0	0	Off	Off	On	1	0
0	1	Off	On	Off	0	1
1	0	On	Off	Off	0	1
1	1	On	On	Off	0	1

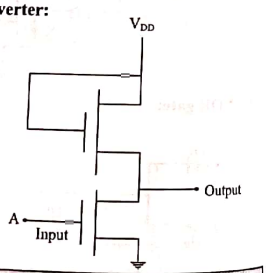
CMOS: It is the combination of PMOS &NMOS transistors.

Input	PMOS	NMOS
0(0v)	On	Off
1(5V)	Off	On

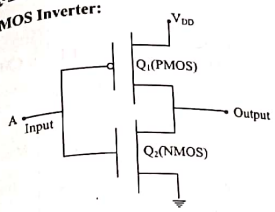
PMOS Inverter:



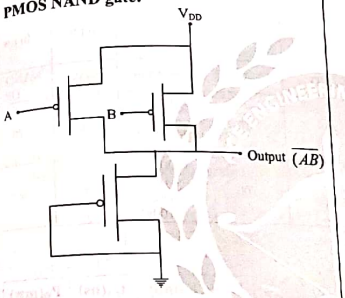
NMOS Inverter:



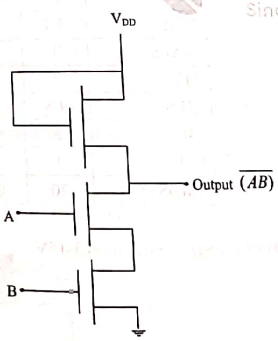
CMOS Inverter:



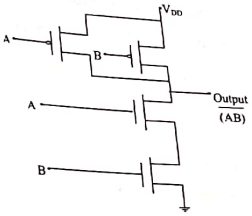
PMOS NAND gate:-



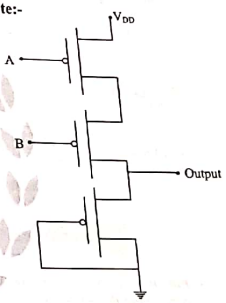
NMOS NAND gate:-



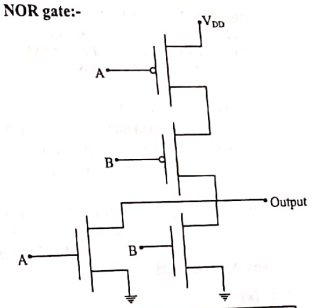
CMOS NAND gate:-



PMOS NOR gate:-



CMOS NOR gate:-



Comparison of logic families:-

Parameter	RTL	DCTL	DTL	standered TTL	ECL	I^2L	CMOS
Components used	Resistors & Transistors	Resistors & Transistors	Resistors, Diodes, Transistors	Resistors, Diodes & Transistors	Resistor & transistors	Transistors	N Channel &P Channel MOSFET
Circuits	Simple	Simplest	Moderate	Complex	Complex	Simple	Moderate
Noise margin	Poor	Poor	High	Medium	Low	poor	High
Fan-out	Low(4)	Low(4)	Low	More(10)	High(25)	8 to 12	50
Power Dissipation (mw) Per gate	30	30	8-12	10	40-55	In μw	In μw
Basic gate	NOR	NOR	NAND	NAND	OR-NOR	NOR	NAND,N OR
Propagation delay(ns)	12	10	30	10	2(ECL 10k) 0.75(ECL 100K)	25-250	70
Speed power product	144	1300	300	100	100(ECL 10k) 40(ECL 100k)	4	0.7

Examples

01. ***Fill in the blanks of the statements below concerning the following Logic Families:***
Standard TTL (74XX), Low power TTL(74LXX) low power Schottky TTL(74LSXX), Schottky TTL(74 SXX), Emitter coupled Logic (ECL), CMOS

(a) Among the TTL Families, _____ family requires considerably less power than the standard TTL(74XX) and also has comparable propagation delay.

(b) Only the _____ family can operate over a wide range of power supply voltages
(GATE-1987)

(b) Ans: CMOS

Sol: (a)

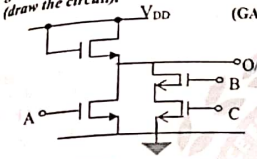
Logic Family	t_{pd}(ns)	P_D(mw)
74	10	10
74AS	1.7	10
74ALS	1	1.2
74S	3	20
74LS	9.5	2
74F	3	5
ECL	1	50
CMOS	70	0.01

(b) Its voltage range is from 3-15V

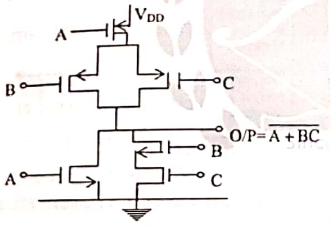
02. ***Among the digital IC-families –ECL, TTL and CMOS:*** **(GATE-1989)**
(a) ECL has the least propagation delay
(b) TTL has the largest fan-out
(c) CMOS has the biggest noise margin
(d) TTL has the lowest power consumption

Ans: (a & c)
Sol: ECL – ultra fast switching speed
TTL–Moderate power dissipation,
CMOS - propagation delay is high

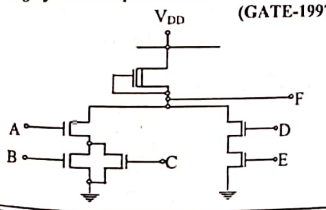
03. ***The CMOS equivalent of the following n MOS gate(figure) is _____ (draw the circuit).*** **(GATE-1991)**



Sol: When 'PFETS' are connected in series then 'NFETS' are connected in parallel and vice-versa



04. ***For the NMOS logic gate shown in figure, the logic function implemented is*** **(GATE-1997)**



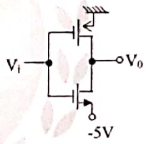
(a) $\overline{ABCDE}$ (b) $(AB+\overline{C})(\overline{D+E})$
(c) $\overline{A.(B+C)+D.E}$ (d) $(\overline{A+B}).C+\overline{D}\,\overline{E}$

Ans: (c)
Sol: Given NMOS Logic gate
In NMOS Logic gate parallel combination = OR
Series combination = AND
$\therefore\ \overline{A(B+C)+DE}$

05. ***The noise margin of a TTL gate is about*** **(GATE-1998)**
(a) 0.2V (b) 0.4V
(c) 0.6V (d) 0.8V

Ans: (b)
Sol: Noise margin of TTL = 0.4V

06. The threshold voltage for each transistor in figure is 2V. For this circuit to work as an inverter, V_i must take the values **(GATE-1998)**



(a) -5V and 0V (b) -5V and 5V
(c) 0V and 5V (d) -3V and 3V

Ans: (a)
Sol: Given $V_T = 2v$
If $V_i = -5v$; $V_0 = 0$.
$V_i = 0v$; $V_0 = -5v$
Hence the circuit acts as an inverter.

07. ***Which of the following technology results in least power dissipation?*** **(ISRO)**
(a) CMOS (b) ECL
(c) TTL (d) NMOS

Ans: (a)
Sol: CMOS is having less power dissipation compared to all logic families.

NOTE:
In CMOS ⇒ Reducing of IC power dissipation is the key to→lower cost (packaging),higher integration, improved reliability.

Advantages of ECL

- Power dissipation remains relatively constant regardless of logic state
- Fan –out is high.
- noise margin is very low
- BJT never go to saturation.

08. Among the following logic families, the one having the lowest power dissipation and highest noise margin is **(JTO)**

(a) Schottky TTL (b) TTL
(c) ECL (d) CMOS

Ans: (d)

Class Room Practice Questions

01. Given that for a logic family, V_{OH} is the minimum output high-level voltage V_{OL} is the maximum output low-level voltage V_{IH} is the minimum acceptable input high- level voltage and $V_{IL}$is the maximum acceptable input low-level voltage. **(GATE - 87)**

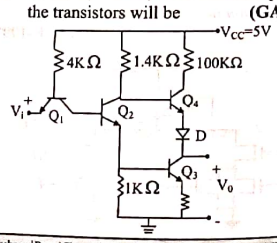
The correct relationship is:
(a) $V_{IH} > V_{OH} > V_{IL} > V_{OL}$
(b) $V_{OH} > V_{IH} > V_{IL} > V_{OL}$
(c) $V_{IH} > V_{OH} > V_{OL} > V_{IL}$
(d) $V_{OH} > V_{IH} > V_{OL} > V_{IL}$

02. The DTL, TTL, ECL and CMOS family GATE of digital ICS are compared in the following 4 columns **(GATE - 03)**

	(P)	(Q)	(R)	(S)
Fan out is minimum	DTL	DTL	TTL	CMOS
Power consumption is Minimum	TTL	CMOS	ECL	DTL
Propagation delay is minimum	CMOS	ECL	TTL	TTL

The correct column is
(a) P (b) Q (c) R (d) S

03. The circuit diagram of a standard TTL NOT gate is shown in the figure.
When V_i = 2.5V, the modes of operation of the transistors will be **(GATE - 07)**



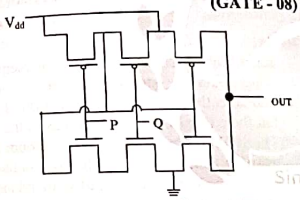
(a) Q_1: reverse active
Q_2: normal active
Q_3: saturation
Q_4: cut-off

(b) Q1: reverse active
Q2: saturation
Q3: saturation
Q4: cut-off

(c) Q1: normal active
Q2: cut-off
Q3: cut-off
Q4: saturation

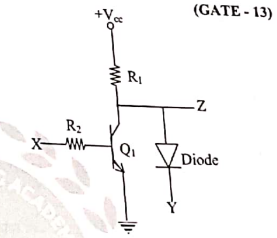
(d) Q1: saturation
Q2: saturation
Q3: saturation
Q4: normal active

04. The logic function implemented by the following circuit at the terminal out is **(GATE - 08)**



(a) P NOR Q (b) P NAND Q
(c) P OR Q (d) P AND Q

05. In the circuit shown below, Q_1 has negligible collector-to-emitter saturation voltage and the diode drops negligible voltage across it under forward bias. If V_{cc} is +5V, X and Y are digital signals with 0 V as logic 0 and V_{cc} as logic 1, then the Boolean expression for Z is **(GATE - 13)**



(a) XY (b) $\overline{X}Y$
(c) $X\overline{Y}$ (d) $\overline{XY}$

Key for CRPQ

01. (b) 02. (b) 03. (b) 04. (d) 05. (b)

7 Semiconductor Memories

Memory:
Memory is the portion of a system for storing binary data in large quantities. Semiconductor memories consist of arrays of storage elements that are generally either latches (or) capacitors.

Units of binary data:
- The smallest unit of binary data is bit.
- 8-bit unit is called a byte .
- The byte can be split into two 4-bit units and each is called nibble
- A complete unit of information is called a word, generally consist of one (or) more bytes.

Cell:
Each storage element in a memory can retain a '1' (or) '0' and is called a cell. Memories are made up of array of cells.
A 64-cell array can be organized in several ways based on units of data.
- 8 × 8 array – 8 byte memory
- 16 × 4 array – 16-nibble memory
- 64 × 1 array – 64 bit memory

Example: A 16K × 8 memory can store 16,384 words of eight bits each. (1K = 1024)

Memory address & capacity:
The location of a unit of data in memory array is called address. The capacity of a memory is the total number of units that can be stored.

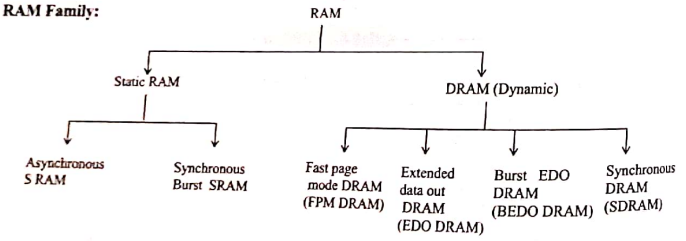
RAM's and ROM's:
The two major categories of semiconductor memories are the RAM and the ROM.

RAM:
Technically it is called Read write memory. Random Access Memory is a type of memory in which all addresses are accessible in an equal amount of time and can be selected on any order for a read or write operation. All RAMS have both read and write capability. RAMS loose stored data when the power is turned off they are volatile memories.

ROM:
Read Only Memory is a type of memory in which data are stored permanently or semi permanently. Data can be read from a ROM, but there is no write operation as in the RAM. The ROM, like the RAM is a random access memory but the term RAM traditionally means a random-access read/write memory.

ROMS store data even if power is turned off they are nonvolatile memories.

RAM Family:



Static RAM (SRAM):

Storage Cell:
All static RAM's are characterized by flip-flop storage cells that are typically implemented with several MOS transistors (MOSFETS). Some smaller RAMS also implemented using BJT's.

Asynchronous SRAM:
In which the operation is not synchronized with a system clock.

Synchronous SRAM:
A synchronous SRAM is synchronized with the system clock.

The burst feature in synchronous SRAMS:
Synchronous SRAMS have an address burst feature, which allow the memory to read or write at up to four locations using a single address.

Burst logic:
The address burst logic in a typical synchronous SRAM consists of a binary counter and EX-OR gates.

Application of SRAM'S:

Cache memory:
It is a relatively small and high speed memory that stores the most recently used instructions or data from the larger but slower main memory.

Dynamic RAM:

Storage cell:
It can store a data bit in a small capacitor rather than a latch. It is having capacitors & MOS transistors.

Advantages: Low Cost

Disadvantages:
The storage capacitor cannot hold its charge over an extended period of time and will lose the stored data bit unless it's charge is refreshed periodically. To refresh it requires additional memory circuit and complicated the operation of DRAM.

Types of DRAMS:

FPM DRAM:
Fast Page Mode DRAM. The page in memory is all of the column address as contained with in one row address.

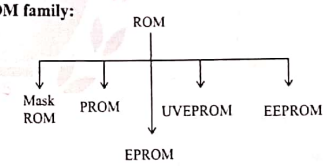
EDO DRAM:
Extended data output DRAM. It is also called hyper page mode DRAM. The difference between FPM DRAM & EDO DRAM is the access time "speed up in EDO DRAM".

BEDO RAM:
It is an EDO DRAM with address burst capabilities.

SDRAM: It is faster DRAM.

ROM's:
A ROM store data permanently (or) semi permanently stored data, which can be read from the memory but either can not be changed at all (or) cannot be changed without specialized equipment.

ROM family:



Masked ROM:
It simply refers ROM. It is permanently programmed during the manufacturing process to provide widely used standard functions. Once the memory is programmed it cannot be changed.

ROM Cells: MOS transistors.

ROMs can use look up tables (LUT's) for code conversions and logic function generation.

PROM:

PROM's are same as mask ROM's, once they have been programmed. The difference is that PROM's come from manufacturer un programmed and are custom programmed in the field to meet the user's needs.

Once a PROM is programmed it cannot be changed.

Cells: MOS transistors with fusible links.

EPROM: (Erasable Programmed ROM):

Unlike an ordinary PROM, EPROM can be reprogrammed if an existing program in the memory array is erased first.

Cell: NMOSFET array with an isolated gate structure.

Two types of EPROMS:
(1) UVEPROM (2) EEPROM

UVEPROM:

Ultra Violet EPROM: Here erase can be done by exposure of the memory array chip to high intensity UV radiation through quartz window on top of the package.

EEPROM:

Electrically Erasable PROM's can be both erased and programmed with electrical pulses.

Two types of EEPROMs: (1) Floating gate MOS (2) Metal Nitride – Oxide silicon (MNOS)

Flash memory:

These are high density read/ write memories that are non volatile.

Storage Cell: Floating – gate MOS transistor.

Comparison of Memories:

Memory Type	Nonvolatile	High density	One transistor Cell	IN-System writability
Flash	Yes	Yes	Yes	Yes
SRAM	No	No	No	Yes
DRAM	No	Yes	Yes	Yes
ROM	Yes	Yes	Yes	No
EPROM	Yes	Yes	Yes	No
EEPROM	Yes	No	No	Yes

Memory Expansion:

Available memory can be expanded to increase word length (number of bits in each address) or the word capacity (number of different addresses) or both Memory expansion is accomplished by adding an appropriate number of memory chips to the address ,data, and control busses.

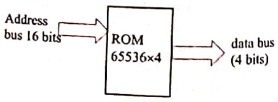
Word-length expansion :

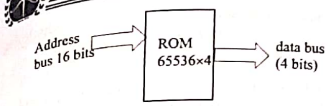
To increase word length in memory the number of bits in the data bus must be increased.

Example:

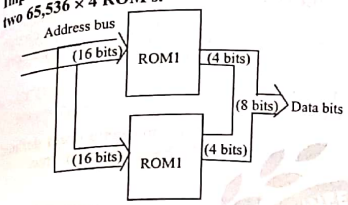
An 8 bit word length can be achieved using two memories, each with 4 – bit words.

Two separate 65,536 × 4 ROM s.

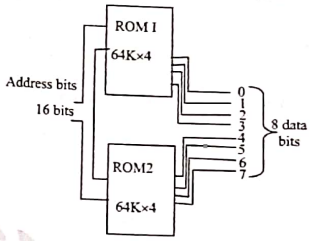




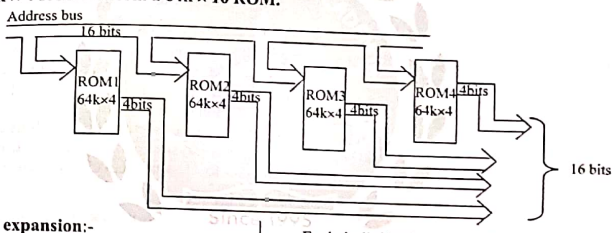
Implementation of one 65,536 × 8 ROM from two 65,536 × 4 ROM s.



Problem: - (1) Expand 65,536 × 4 ROM (64k × 4) to form a 64k × 8 ROM.



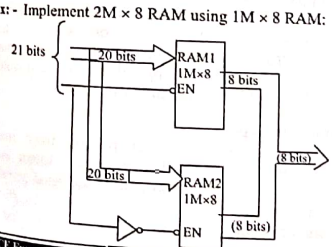
Expand 64k × 4 ROM to form a 64k × 16 ROM.



Word Capacity expansion:-

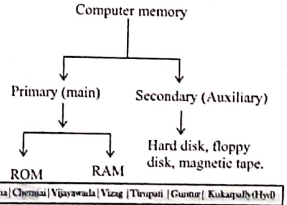
When memories are expanded to increase the number of words, the number of addresses is increased.

Ex: - Implement 2M × 8 RAM using 1M × 8 RAM:



Each individual memory has 20 address bits to select 1,048,576 address. The expanded memory has 2,097,152 addresses requires 21 address bits. The 21st address bit is used to enable appropriate memory chip.

- **Points to remember:**



- Primary memories are semiconductor memories. They are available in the form of IC's with different memory capacities.
- The capacity of memory IC denotes as $2^n \times m$. 2^n – no. of memory locations m – no. of bits stored in each memory location.
  Ex: - $2^{10} \times 8 = 1024 \times 8 = 1k \times 8$.
- The no. of address bits required to identify 2^n memory locations in 'n'.
- To increase the bit capacity (or) length of each memory location, memory ICS connected in parallel.
  Ex: - 1024 × 8 – memory capacity can be obtained by using '4'. ICS of memory capacity 1024 × 2.
- To increase the no. of memory locations, the memory ICS connected such that at any time only one memory IC must be selected.
  Ex: - To get 4k × 8 memory capacity: it is required to use four 1k × 8 ICS and at any time one of the four memory ICS can be selected using chip select decoder.
- The number of memory ICS of capacity 1k × 4 required to construct a 8k × 8 are 16. (16 memory ICS of 1k × 4 capacity).
- **Access time** : - It is the time for a memory to access a memory location for reading or writing.
- **Access rate** : - Reciprocal of access time. Units – words per second.
- **Random access** : - If the access time is independent of position of memory location called Random access. The access time of every memory location is same.
  Ex : RAM.
- **Sequential access**: - A memory in which the locations can be accessed in a sequence.
  Ex: magnetic tape, magnetic bubble.
- **SRAM**: The data stored interms of voltage. Storage cells are conventional flip flops.
- **DRAMs**: The data stored in form of charge on capacitor. Storage cells are – charge storage capacitors with driver transistors. SRAMs are faster than DRAMs.

Magnetic memories :

Access time:
The access time of a magnetic drum defined as the sum of seek time & transfer time.

Seeking time:
It is the time for moment of read/write head to the desired track from current track.

Magnetic hard disks:
Computers use hard disks as the internal mass storage media. Hard disks are rigid "platters" made up of aluminum alloy (on formatted into tracks and sectors. Each track is divided into number sectors, and each track and sector has a physical address that is used by the OS to locate a particular data record.

Latency period:
It is the time taken for the desired sector to spin under the head is positioned over the desired track.

Floppy disks:
It is made up of a flexible polyester material with a magnetic coating on both sides. Floppy disks are formatted into tracks, sectors like hard disks. Floppy disks are removable disks but hard disks are not.

Magnetic tape:
Tape is used for backup data from mass storage devices and is typically slower than disks because data on tape is accessed serially rather than randomly.

Points to remember : -

- Destructive Read out memory (DRO) : If the reading method destroys its contents that memory called DRO. For such memories each read operation must be followed by write operation to restore the contents.
  Ex : Magnetic core.
- **Non destructive Read out** : - (NDRO) if the read operation does not change its contents.
  Ex: Magnetic tapes, disks, RAMS, ROMs.
- CCD is a volatile memory and sequential access type.

Examples

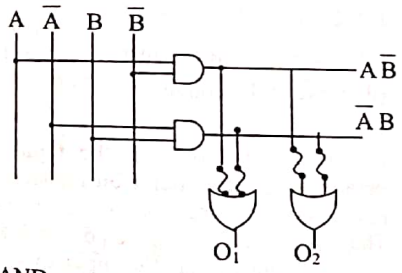
01. Choose the correct statement(s) from the following:

(a) PROM contains a programmable AND array and a fixed OR array.
(b) PLA contains a fixed AND array and a programmable OR array.
(c) PROM contains a fixed AND array and a programmable OR array.
(d) PLA contains a programmable AND array and a programmable OR array.

Ans: (c)

Sol: PROM: Each output can be programmed to be any function of A and B by selectively blowing the appropriate fuses.

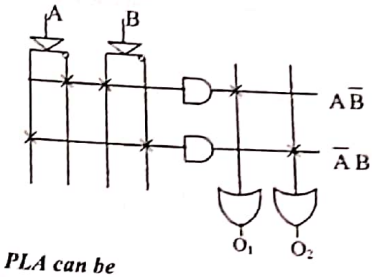
Ex:



∴AND gates are fixed and OR gates are programmed.

PLA: PLA contains programmable AND array and programmable OR arrays

Eg: $\bar{A}B + A\bar{B} = A \oplus B$.



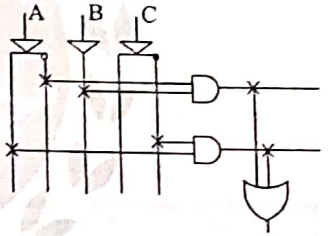
02. A PLA can be

(a) as a microprocessor
(b) as a dynamic memory
(c) to realize a sequential logic
(d) to realize a combinational logic.

Ans: (d)

Sol: PLA's are used to realize combinational circuits

Ex:- $\bar{A}\bar{B} + \bar{C}A = Y$



03. An 8085 microprocessor based system uses a 4k x 8bit RAM whose starting address is AAOO H. The Address of Last byte in this RAM is

(a) 0FFF H (b) 1000 H
(c) B9FF H (d) BA00 H

Ans: (c)

Sol: For 4K × 8 bit RAM, Given starting address = AA00

Memory size = 4K × 8 bit RAM= $4K = 2^{12}$ (i.e No. of address lines are 12)

Last address must be all the bits with '1'.

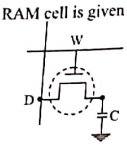
Ending address = starting address + OFFF H
=AA00 H+0FFF H = B9FF H

04. The minimum number of MOS transistors required to make a dynamic RAM cell is

(a) 1 (b) 2 (c) 3 (d) 4

Ans: (a)

Sol: Dynamic RAM cell is given below



05. An 8-bit microcontroller has an external RAM with the memory map from 8000H to 9FFFH. The number of bytes this RAM can store is

(a) 8193 (b) 8192 (c) 8191 (d) 8000

Ans: (b)

Sol: Given memory map is from 8000H to 9FFFH

$\therefore$ 9FFFH

-8000H

1FFFH $(000\ 1111111111111)_2$

$\therefore$ It uses 13 address bits.

i.e $2^{13}=2^3.2^{10} = 8\times1024 = 8192$

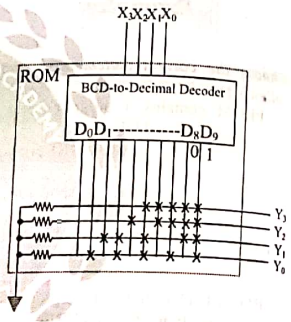
The no. of bytes this RAM can store is 8192

Class Room Practice Questions

01. A ROM is to be used to implement a "squarer", which outputs the square of a 4-bit number. What must be the size of the ROM?
(a) 16 address lines and 16 data lines.
(b) 4 address lines and 8 data lines.
(c) 8 address lines and 8 data lines
(d) 4 address lines and 16 data lines

02. A single ROM is used to design a combinational circuit described by a truth table. What is the number of address lines in the ROM?
(a) Number of input variables in the truth table.
(b) Number of output variables in the truth table.
(c) Number of input plus output variables in the truth-table.
(d) Number of lines in the truth-table.

03. If the input X_3, X_2, X_1, X_0 to the ROM in the figure are 8421 BCD numbers, then the outputs Y_3, Y_2, Y_1, Y_0 are



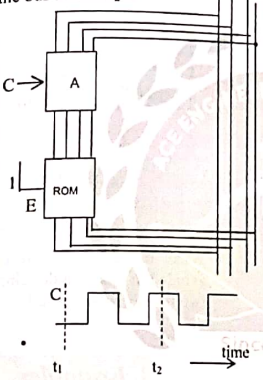
(a) Gray code numbers
(b) 2421 BCD numbers
(c) excess-3 code numbers
(d) none of the above.

04. In the circuit shown in the figure. A is parallel-in, parallel out 4 bit register, which loads at the rising edge of the clock 'C'. The input lines are connected to a 4 bit bus, W. Its output acts as the input to a 16 x 4 ROM whose output is floating when the enable input E is 0. A partial table of the contents of the ROM is as follows.

Address	Data
0	0011
2	1111
4	0100
6	1010
8	1011
10	1000
11	0010
14	1000

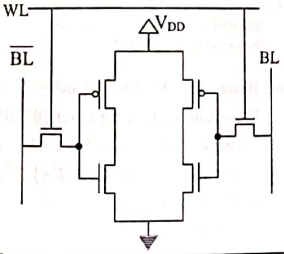
The clock to the register is shown, and the data on the W bus at time t_1 is 0110. The data on the bus at time t_2 is

(a) 1111
(b) 1011
(c) 1000
(d) 0010

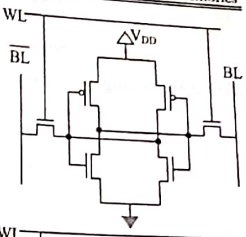


05. If WL is the Word Line and BL the Bit Line, as SRAM cell is shown in

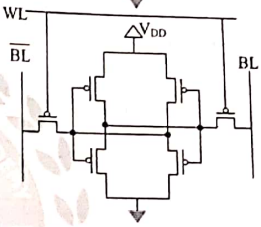
(a)



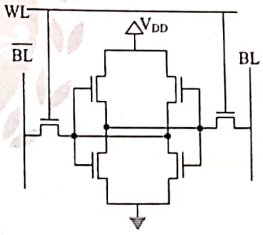
(b)



(c)



(d)



Key for CRPQ

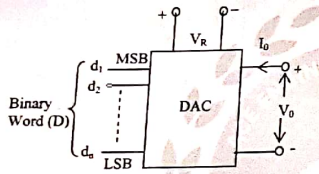
01. (b)	02. (a)	03. (b)
04. (c)	05. (b)	

8 A/D & D/A Converters

Digital to Analog conversion:

These are used to translate the output of a digital system in to an analog form. It is commonly referred to as decoding device. Since it is used to decode the digital signals into proportional voltage or current signals for an entry in to an analog system. D/A converter convert digital information into corresponding analog signals.

The schematic of DAC:



$V_o = K.\ V_R\ (d_1 2^{-1} + d_2 2^{-2} + \text{- - -} + d_n 2^{-n})$.

V_{FS} = full scale output voltage.

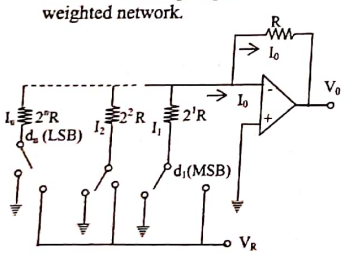
K = scaling factor usually adjusted to unity.

d_1 = MSB with a weight of $V_R/2$

d_n = LSB with a weight of $V_R/2^n$

Weighted resistor DAC:

It uses a summing amplifier with a binary weighted network.



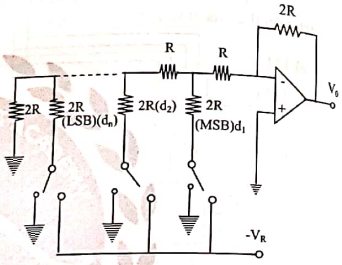
$V_o = -I_o R_f$

$$V_0 = -V_R \frac{R_F}{R}\left[d_1 2^{-1} + d_2 2^{-2} + \ldots\ldots d_n 2^{-n}\right]$$

Disadvantage:-

We require wide range of resistor values hence restricted its use up to 8-bit.

R- 2R ladder:



Wide range of the resistor value is avoided in R-2R ladder circuit. It requires two resistor values R, 2R.

Examples

01. The basic step of a 9-bit DAC is 10.3 mV. If 000000000 represents 0V, what is the output if the input is 101101111.

Sol: Basic step = LSB = 10.3 mV

The output voltage for input ‘101101111’ is

$= 10.3\text{ mV}\ (1\times2^8 + 0\times2^7 + 1\times2^6 + 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 1\times2^1 + 1\times2^0)$.

$= 10.3\text{ mV}(367)$

$= 3.78$ V.

02. Calculate the value of LSB, MSB and full scale output for an 8-bit DAC for V_R=10 V.

Sol: $\text{LSB} = \frac{1}{2^8} = \frac{1}{256}$

For a 10 V range

$\text{LSB} = \frac{10\text{V}}{256} = 39\text{mV}$

$\text{MSB} = \frac{1}{2} \times \text{fullscale} = 5\text{V}$

Full scale o/p = full scale voltage – 1 LSB

$= 10\text{ V} - 0.039\text{V}$

$= 9.96\text{ 1V}$.

Points to remember:

1. R-2R ladder superior than binary weighted.
2. Where the number of bits increased the resistor value used for LSB has to be very high in binary weighted.
3. The advantage of R-2R ladder over binary weighted (i) better linearity (ii) It requires only two different types of resistor R & 2R values
4. A D/A converter is said to be ideally or perfectly linear if it gives equal increments in the analog output voltage for equal increments in the numerical value of digital value.
5. **D/A resolution:-** It is defined as the smallest change in the analog output voltage corresponding to a change of one bit in the digital output.

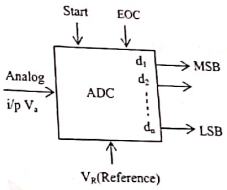
$$\text{Resolution} = \text{LSB} = \frac{V_R}{2^n}$$

1. The resolution of an n-bit DAC with full scale output voltage V_{FS} is given by $\frac{V_{FS}}{2^n - 1}$ volts.
2. The accuracy of D/A converter is a measure of difference between the actual analog output and what the output should be in the ideal case.

Analog to Digital Converter (ADC):

It converts analog voltage into corresponding digital code.

An ADC considered as en coder.



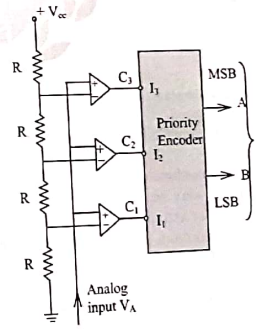
Types of ADCS:

1. Direct type ADCS
2. Integrating type ADCS

Direct type ADCS:

Flash (comparator) type A/D:

It is fastest & most expensive technique.

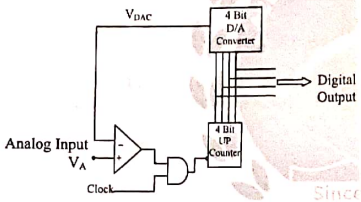


2– bit Flash ADC

Disadvantages:
The number of comparators required almost\
Double for each added bit. The no of comparators required for an 'n' bit flash ADC is $= 2^n$-1.

Counter type A/D:
DAC circuit is used inside ADC.

In the architecture an up counter is used which is reset at the start of every conversion. The counter is incremented for every clock pulse. The output of counter is fed as input to DAC. The output of DAC V_{DAC} is compared with analog input voltage. As long as $V_{DAC} < V_A$ then counter is incremented.
The counter value for which $V_{DAC} \geq V_A$ is the digital output.

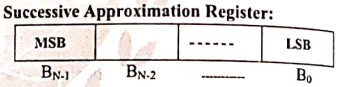


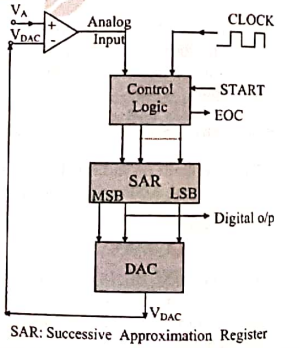
Disadvantages:-
Low speed.
Maximum Conversion time
$= 2^n$ -1 clock periods.
n = no of bits.
If the max analog voltage represented by n-pulses and if the clock period is 'T' sec , the minimum separation between samples is 'nT' sec.

Servo tracking A/D converter:
It is an improved version of counting ADC consists of up/ down counter for controlling the direction.

Successive approximation A/D:
Very efficient to complete n-bit conversion in just n- clock periods.
Special register called successive approximation register is used.
At the starting of conversion process MSB B_{N-1} is set to 1 and remaining bits are reset.
This output of the register is fed to DAC. The output of the DAC V_{DAC} is compared to analog voltage V_A.
If $V_{DAC} > V_A$ then MSB B_{N-1} is reset and the next MSB B_{N-2} is set.
If $V_{DAC} < V_A$ then MSB B_{N-1} is set and the next MSB B_{N-2} is set to 1.
This output of the register is fed to DAC. The output of the DAC V_{DAC} is compared to analog voltage V_A.
If $V_{DAC} > V_A$ then B_{N-2} is reset and the next MSB B_{N-3} is set.
If $V_{DAC} < V_A$ then B_{N-2} is set and the next MSB B_{N-3} is set to 1.
This procedure is followed until LSB is reached.

Successive Approximation Register:





SAR: Successive Approximation Register

Charge balancing ADC:
This principle is first convert the input signal to a frequency using V-F converter. This frequency is measured by a counter and converted to an output code proportional to analog output.

Advantage:
possible to transmit frequency even in noisy environment, isolated form.

Disadvantage:
output depends up on RC product.

Dual- slope ADC:

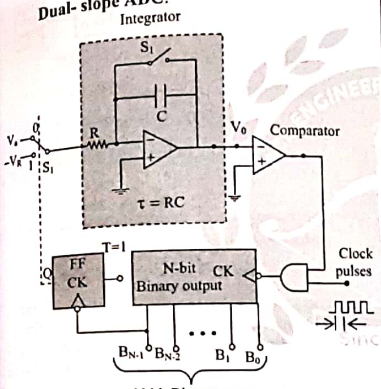
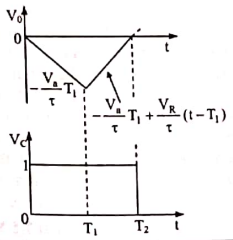


Fig. Dual slope Integrator A/D Converter



$\therefore N = \frac{V_a}{V_R} 2^n$

Disadvantage: long conversion time.

Examples

01. If the analog signal V_a is + 4.129, and V_R = 8V. Find equivalent digital number. It uses 16 bit counter.

Sol: $V_a = V_R \left(\frac{N}{2^n}\right)$

$N = 2^n \left(\frac{V_a}{V_R}\right) = 2^{16}\left(\frac{4.129}{8}\right)$

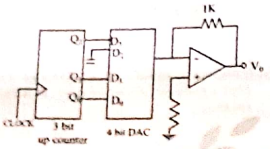
$= 333825.$

$= 1000010000100001$

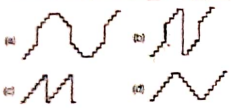
Points to remember:
- Flash type ADC is fastest ADC.
- An n-bit flash type ADC requires 2^n-1 comparators.
- Counter type of ADC uses linear search and successive approximation type of ADC uses binary search.
- An ADC having an analog range of $-\frac{V}{2}$ to $+\frac{V}{2}$ and n-bit digital output has a resolution of $\frac{V}{2^n - 1}$ volts.
- Dual slope ADC is more Accurate.
- Flash type ADC does not require Counter.
- Counter type ADC & successive approximate ADC use DAC.

Class Room Practice Questions

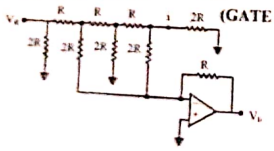
01. A 4-bit D/A converter is connected to a free-running 3-bit up counter, as shown in the following figure which of the following waveforms will be observed at V_0 = ?
(GATE - 06)



In the figure shown above, the ground has been shown by the symbol



Statement for Linked Answer Qs. 02 & 03:
In the Digital-to-Analog converter circuit shown in the figure below,
V_R = 10 V and R = 10kΩ



02. The current i is
(a) 31.25 μA (b) 62.5 μA
(c) 125 μA (d) 250 μA

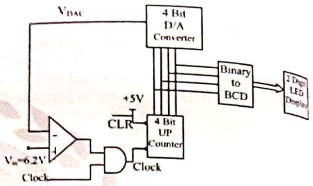
03. The voltage V_0 is
(a) -0.781v (b) -1.562v
(c) -3.125v (d) -6.250v

Statement for Linked Answer Qs. 04 & 05:
In the following circuit, the comparator output is logic "1" if $V_{in} > V_{DAC}$ and is logic "0" otherwise. The D/A conversion is done as per the relation

$$V_{DAC} = \sum_{n=0}^{3} 2^{n-1} b_n \text{ Volts}$$

where b_3(MSB), b_2, b_1 and b_0 (LSB) are the counter outputs.

The counter starts from the clear state.



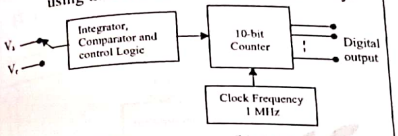
04. The stable reading of the LED display is
(a) 06 (b) 07 (c) 12 (d) 03

05. The magnitude of the error between V_{DAC} and V_{in} at steady state in volts is
(a) 0.2 (b) 0.3 (c) 0.5 (d) 1.0

06. For a dual ADC type 3 ½ digit DVM, the reference voltage is 100 mV and the first integration time is set to 300 ms. For some input voltage, the "de integration" period is 370.2ms. The DVM will indicate.
(a) 123.4 (b) 199.9
(c) 100.0 (d) 1.414

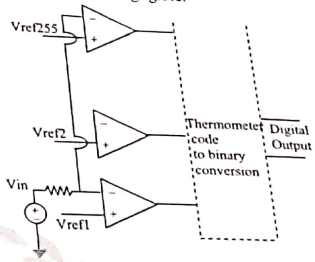
07. For an 8-bit digital-to-analog converter having reference voltage of 8 V, the least significant 4 bits of the input are grounded and the most significant 4 bits are driven by 4 bit data from a binary counter. The maximum obtainable peak-to-peak amplitude of a waveform at the output of the digital-to-analog converter is
(a) 4 V (b) 6 V
(c) 7.2 V (d) 7.5 V

08. The simplified block diagram of a 10-bit A/D converter of dual slope integrator type is shown in the Fig. The 10-bit counter at the output is clocked by a 1 MHz clock. Assuming negligible timing overhead for the control logic, the maximum frequency of the analog signal that can be converted using this A/D converter is approximately.



(a) 2 kHz (b) 1 kHz
(c) 500 Hz (d) 250 Hz

09. In an N bit flash ADC, the analog voltage is fed simultaneously to $2^N - 1$ comparators. The output of the comparators is then encoded to a binary format using digital circuit. Assume that the analog voltage source V_{in} (whose output is being converted to digital format) has a source resistance of 75Ω as shown in the circuit diagram below and the input capacitance of each comparator is 8 pF. The input must settle to an accuracy of ½ LSB even for a full scale input change for proper conversion. Assume that the time taken by the thermometer to binary encoder is negligible.



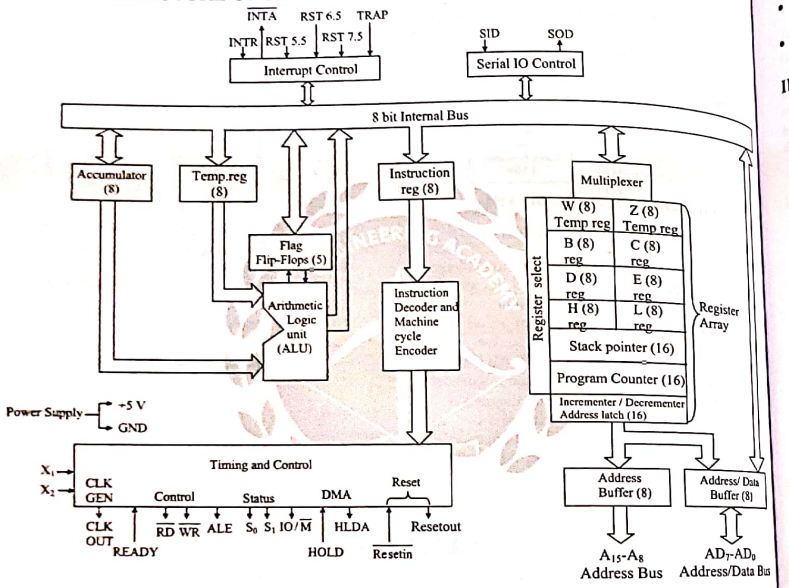
If the flash ADC has 8 bit resolution, which one of the following alternatives is closest to the maximum sampling rate?
(Gate-16)(Set-2)
(a) 1 megasamples per second
(b) 6 megasamples per second
(c) 64 megasamples per second
(d) 256 megasamples per second

Key for CRPQ

01. (b)	02. (b)	03. (c)	04. (d)	05. (b)
06. (a)	07. (d)	08. (d)	09. (b)	

9 Architecture, Pinout of 8085 & Interfacing with 8085

I. ARCHITECTURE OF 8085



❖ **Registers available in 8085**

8 bit Registers	16 bit Registers
Accumulator	Program Status Word
B, C, D, E, H, L	Program counter
Instruction Register	Stack Pointer
Temporary Register	BC pair, DE pair, HL pair
W & Z Registers	
Flag Register	

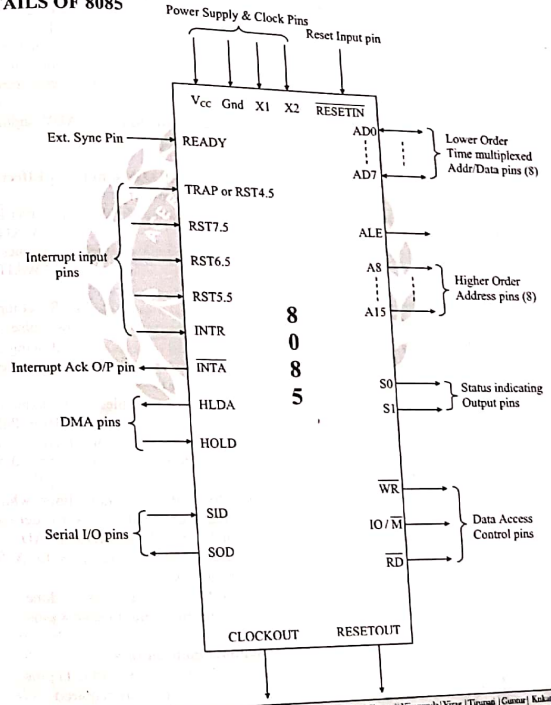
❖ **Flag Register of 8085**

S	Z	X	AC	X	P	X	CY

- Flag register is a 8 bit register which consists of 5 conditional flags namely carry flag (CY), Parity flag (P), Auxilliary carry flag (AC), Zero flag (Z) and Sign flag (S)

- After performing Arithmetic and logical operations, ALU of 8085 updates status of these flags based on result as given below
- CY is set to 1, if there is a carry generated from MSB of result
- P is set to 1, if the total number of ones in the result is even
- AC is set to 1, if there is a carry from D_3 to D_4
- Z is set to 1, if the result contains all zeros
- S is set to 1, if MSB of result is 1

II. PIN DETAILS OF 8085



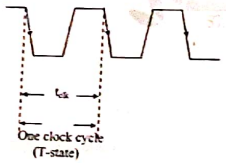
❖ 8085 µP is a 8 bit microprocessor introduced (after 8080 µP) by INTEL corporation in 1977.
- Processing capacity of 8085 is 8 bit.
- 8085 µP can fetch 8 bits of Opcode simultaneously from ROM.
- With Opcode length of 8 bit, the total number of Opcodes available are 2^8 i.e., 256
- Data Handling Capacity of 8085 is 8 bit.
- ALU of 8085 can perform 8 bit operations.
- Length of data Registers (General Purpose Registers) is 8 bit.

❖ 8085 µP is a CISC (Computer Instruction set core) manufactured using NMOS technology and comes in 40 pin DIP.

❖ 8085 µP operates with +5Vdc source connected to V_{cc} input & 0 V applied to GND

❖ 8085 µP has on chip crystal oscillator. As such, the designer has to connect a crystal across X_1 & X_2 of this sequential circuit. the internally generated clock will be distributed throughout architecture of 8085.

❖ All the internal operations will be synchronized with respect to falling edge of this clock.



$$f_{clk} = \frac{f_{crystal}}{2}$$

$$t_{clk} = \frac{1}{f_{clk}}$$

$$= \frac{2}{f_{crystal}}$$

Standard values:

$f_{crystal}$ = 6.14 MHz or 6 MHz

f_{clk} = 3.07 MHz or 3 MHz

Range of f_{clk} is 2 MHz to 5 MHz

- One T-state or one clock cycle is the time taken by 8085 to complete one micro operation
- The clock internally generated by clock generator is also outputted via "CLOCKOUT" output pin which can be utilized by designer to fulfill clock requirements of external devices.

❖ 8085 µP has " READY input pin" to deal with slow devices

READY input pin status	Effect on 8085
0 (lowered by device)	Enter in to "WAIT state"
1 (raised by device)	Comes out of "WAIT state"

❖ 8085 µP has active low Reset input pin. Upon receiving an active low pulse via $\overline{RESETIN}$, 8085 µP performs the following steps
(i) 8085 resets program counter to all zeros (pc) ← 0000H
(ii) 8085 disables all maskable interrupts
(iii) 8085 sends Active High Pulse via Reset out output pin(which can be utilized by designer to reset external devices)

❖ 8085 µP has 8 data lines which are time multiplexed with 8 lower order Address lines and available as AD0 to AD7. The higher order Address lines (A8 to A15) are non-multiplexed.
→ Multiplexing is done by chip manufacturer to save 8 pins.

Without multiplexing:
16 Address pins+8 data pins
i.e., 24 pins are required.

With multiplexing:
- 8 Address/Data pins + 8 Address pins i.e., only 16 pins are required
- Demultiplexing is the technique of separating lower order Address (first placed in T_1) From the multiplexed Bus.
- To support demultiplexing, a control output pin is provided namely ALE(Address Latch Enable)

❖ ALE output made "Active-High (1)" by 8085 when it generates Address in T_1. In subsequent T-states, ALE output is maintained as "Active- Low (0)".

❖ After demultiplexing, 8 bit data bus and 16 bit address bus are available separately
- µP sends 16 bit Address via 16 bit Address Bus for selecting a memory location or IO device (as first microoperation in any external Access Operation)
- The number of unique addresses that can be generated by 8085 is 2^{16} i.e., 65536
- Max accessable size of memory = 2^{16} Bytes = 64 KB
- Address Space = 0 .....0 (16) to 1 .....1 (16) = 0000H to FFFFH
- Length of address Registers(i.e., Pointer register like PC, SP,M etc) is 16 bit.

❖ 8085 µP has two status indicating output pins via which it indicates its status.

S_1	S_0	Status indication
0	0	Halted
0	1	Write
1	0	Read
1	1	Fetch

❖ 8085 µP has three control status (Read & write) or Data Access (Receive & Transmitt) control output pins. When 8085 performs external Access Operation it indicates" type of Access" via these 3 output pins namely IO/$\overline{M}$, $\overline{WR}$ & $\overline{RD}$.
→ For IO access operation, 8085 makes IO/$\overline{M}$ as Active High.
- For memory access operation, 8085 makes IO/$\overline{M}$ as Active low.
- For write operation, 8085 makes $\overline{WR}$ as low and $\overline{RD}$ as High.
- For Read operation, 8085 makes $\overline{RD}$ as low and $\overline{WR}$ as High.

External Access Operation	IO/$\overline{M}$	$\overline{WR}$	$\overline{RD}$
Opcode fetch operation Operand Read operation Memory Read Operation	0	1	0
Memory write operation	0	0	1
IO write operation	1	0	1
IO Read Operation	1	1	0

❖ 8085 µP supports serial communication. Two pins are provided to facilitate serial I/o Access namely SOD (Serial Output Data) pin and SID (Serial Input Data) pin.
- For execution of SIM instruction, the data bit(0/1) in MSB of accumulator is outputted(transmitted out) via SOD output Pin.
- For execution of RIM instruction, the data bit (0/1) at SID input pin is received (inputted)into MSB of Accumulator.
- SIM is used for sending data bit and RIM is used for receiving data bit.

❖ 8085 µP supports Direct Memory Access. DMA is a technique of data transfer between I/O devices and memory without interference of microprocessor. DMA is very much needed for fast transfer of Bulk data between I/O devices and memory. Two pins are provided to facilities DMA namely HOLD I/P pin and HLDA O/P Pin.
- When HOLD input is raised by external device (requesting for DMA), 8085 performs following steps:

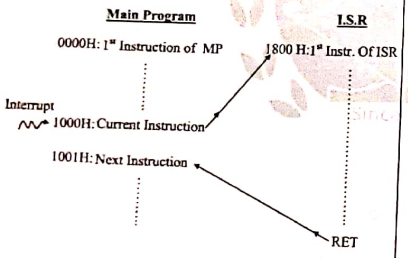
Step 1: 8085 tristates systems Bus
Step 2: 8085 generates active High signal via HLDA output pin and enters into "IDLE state"

- When 8085 is in IDLE state, I/O device accesses memory directly with the help of DMA controller (Ex: 8237 or 8257)
- Once DMA is over, I/O device removes its request by Lowering HOLD i/p pin. When Hold input is lowered, 8085 comes out of IDLE state and regains control over system Bus.

❖ Interrupts of 8085 μP:

- Interrupt is an event that demands attention of microprocessor.
- In general, any microprocessor is said to be in execution of main program. When an interrupt occurs, if μP ignores (i.e., doesn't recognize) then it continues with main program execution.

If μP recognizes occurrence of interrupt, then it performs the following:

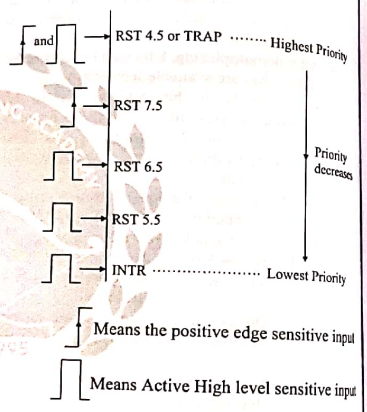


Step 1: 8085 completes execution of currently fetched instruction
Step 2: 8085 saves P.C contents (Next Address or Return Address) to current Top of stock by pushing it.
Step 3: 8085 loads P.C with starting Address of ISR (Vector Address) and starts executing ISR.

Once 8085 completed execution of ISR, it returns to main program by popping top 2 Bytes of stack into P.C (Return Address)

Note:
In above example, 1001H is return Address & 1800H is vector Address.

- 8085 μP has 5 Hardware(external) interrupts which exist in the form of input Pins (Active High)



- 8085 μP has 8 software interrupts which exist in the form of 1 Byte RST(Restart) instructions

RST	0
RST	1
RST	2
RST	3
RST	4
RST	5
RST	6
RST	7

- As such, 8085 has total 13 interrupts categorized as given below:

i) 8085 has 12 maskable interrupts (which can be either enabled or disabled) namely: RST7.5, RST6.5, RST5.5, INTR, RST0 to RST7

ii) 8085 has only 1 Non maskable interrupt (which can not be disabled) namely TRAP or RST4.5. This NMI is to be used only in emergency situations like power failure.

iii) 8085 has 12 vectored interrupts (for which vector address is prefixed & known to μP) namely: RST4.5, RST7.5, RST6.5, RST5.5, RST0 to RST7.

Vector address for any "RST n" can be calculated as $[8n]_{Hex}$

Vector interrupts	Vector Address
RST0	0000H
RST1	0008H
RST2	0010H
RST3	0018H
RST4	0020H
RST4.5	0024H
RST5	0028H
RST5.5	002CH
RST6	0030H
RST6.5	0034H
RST7	0038H
RST7.5	003CH

iv) 8085 has only 1 Non-vectored interrupt (for which, vector Address is not prefixed and left to choice of programmer) namely INTR i.e., Interrupts Request. For servicing such NVI, external hardware (8259 PIC) is needed. Whenever 8085 recognizes interrupt request at INTR input, then it generates Active- low ACK signal via $\overline{INTA}$ o/p. Then 8259 supplies vector Address to μP. With the help of 8259 PIC, number of interrupts can be increased.

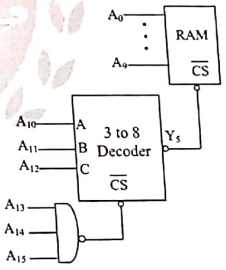
Example1.1:
Three devices A,B and C have to be connected to a 8085 microprocessor. Device A has highest priority and device C has the lowest priority. In this context, which of the following is correct assignment of interrupt inputs?

Sol: The decreasing order of priority of external, hardware interrupts is:

TRAP – High priority
RST 7.5
RST 6.5
RST 5.5
INTR – Low Priority

∴ A uses RST 7.5, B uses RST 6.5
And C uses RST 5.5

Example 1.2:
The range of address of the RAM which is interfaced to a microprocessor as shown in fig is



Sol:

A15	A14	A13	A12	A11	A10	A9	A8	
1	1	1	1	0	1	0	0	=F400H
1	1	1	1	0	1	1	1	=F7FFH

Example1.3

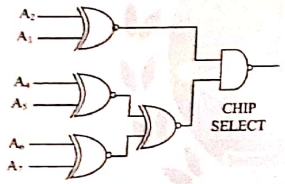
A memory system of size 16KB is required to be designed using memory chips which have 12 address lines and 4 data lines each. Then, the number of such chips required to design the memory system is _____

Sol: The full address space = 16K×8

$$n = \frac{16K \times 8}{2^{12} \times 4} \approx 8$$

Class Room Practice Questions

01. The decoding circuit shown below has been used to generate the active low chip select signal for a microprocessor peripheral. (The address lines are designated as A_0 to A_7 for I/O addresses) **(GATE - 97)**



The peripheral will correspond to IO addresses in the range

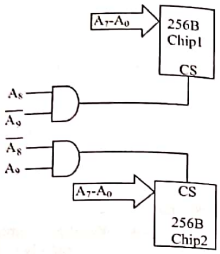
(a) 60 H to 63 H (b) A4 H to A7 H

(c) 50 H to AF H (d) 70 H to 73 H

02. What memory address range is NOT represented by chip 1 and chip2 in the figure.

A_0 to A_{15} in this figure are address lines and CS means chip select.

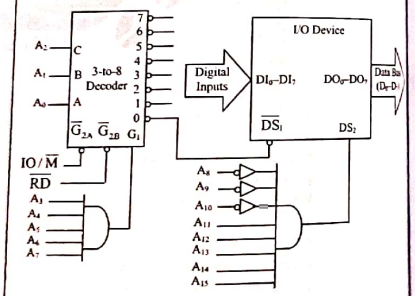
(GATE-EC-05)



(a) 0100H-02FFH (b) 1500H-16FFH

(c) F900H-FAFFH (d) F800H-F9FFH

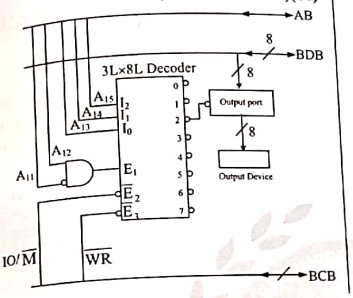
03. For the 8085 microprocessor, the interfacing circuit to input 8-bit digital data (DI_0 – DI_7) from an external device is shown in the figure. The instruction for correct data transfer is **(GATE –EC- 14)(S2)**



(a) MVI A, F8H (b) IN F8H

(c) OUT F8H (d) LDA F8F8H

04. An output device is interfaced with 8 bit microprocessor 8085A. The interfacing circuit is shown in figure **(GATE – EE – 14)(S1)**

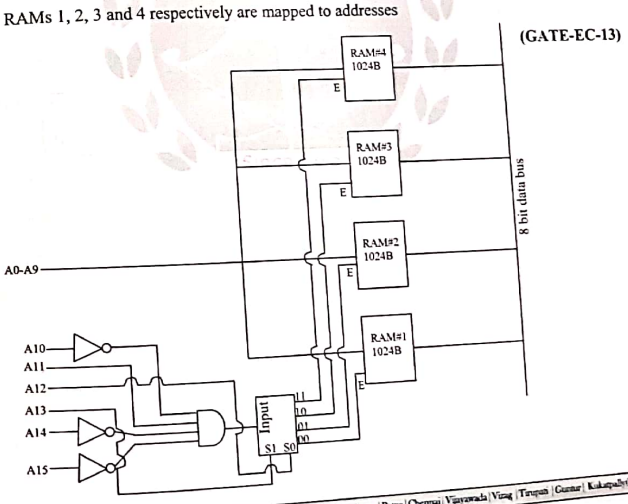


The interfacing circuit makes use of 3 line to 8 line decoder having 3 enable lines E_1 $\overline{E}_2$, $\overline{E}_3$. The address of the device is

(a) 50H (b) 5000 H

(c) A0H (d) A000H

05. There are four chips each of 1024 bytes connected to a 16 bit address bus as shown in the figure below. RAMs 1, 2, 3 and 4 respectively are mapped to addresses **(GATE-EC-13)**



(a) 0C00H-0FFFH, 1C00H-1FFFH, 2C00H-2FFFH, 3C00H-3FFFH
(b) 1800H-1FFFH, 2800H-2FFFH, 3800H-3FFFH, 4800H-4FFFH
(c) 0500H-08FFH, 1500H-18FFH, 3500H-38FFH, 5500H-58FFH
(d) 0800H-0BFFH, 1800H-1BFFH, 2800H-2BFFH, 3800H-3BFFH

06. An 8 Kbyte ROM with an active low Chip Select input ($\overline{CS}$) is to be used in an 8085 microprocessor based system. The ROM should occupy the address range 1000H to 2FFFH. The address lines are designated as A_{15} to A_0, where A_{15} is the most significant address bit. Which one of the following logic expression will generate the correct $\overline{CS}$ signal for this ROM?
(a) $A_{15} + A_{14} + (A_{13}.A_{12} + \overline{A_{13}}.\overline{A_{12}})$
(b) $A_{15}.A_{14}.(A_{13} + A_{12})$
(c) $\overline{A_{15}}.\overline{A_{14}}.(A_{13}.\overline{A_{12}} + \overline{A_{13}}.A_{12})$
(d) $\overline{A_{15}} + \overline{A_{14}} + A_{13}.A_{12}$

07. 3 × 8 decoder with two enable inputs is to be used to address 8 blocks of memory. What will be the size of each memory block when addressed from a sixteen bit bus with two MSB's used to enable the decoder?
(a) 2K (b) 4K (c) 16K (d) 64K

Key for CRPQ

01. (a)	02. (d)	03. (d)	04. (b)	05. (d)	06. (c)	07. (a)

10 Instruction Set of 8085 & Programming with 8085

I. INSTRUCTION SET OF 8085

- All the instructions of 8085 are classified into different groups based on operation namely data transfer, Arithmetic, Logical, Branch control and Machine Control Groups.
- Notation:
  - r – 8 bit register
  - r_s – 8 bit source register
  - r_d – 8 bit destination register
  - r_{16} or r_p – 16 bit register pair or register pointer
  - M – Memory location pointed by HL pair
  - Data8 – 8 bit data
  - Addr8 – 8 bit I/0 Port Address
  - Addr16 – 16 b it memory location Address
  - num16 – 16 bit number
  - TOS - Top of the sack
  - LN - Lower nibble
  - HN - Higher nibble

01. DATA TRANSFER GROUP OF INSTRUCTIONS:

Syntax	Operation
MVI r, data8	(r) ← data 8
LXI r_{16} , num16	(r_{16}) ← num 16
MVI M, data8	((HL)) ← data 8
MOV r_d,r_s	(r_d) ← (r_s)
SPHL	(SP) ← (HL)
PCHL	(PC) ← (HL)
XCHG	(DE) ↔ (HL)
MOV M,r	((HL)) ← (r)
MOV r,M	(r) ← ((HL))
STA Addr 16	(Addr16) ← (A)
LDA Addr 16	(A) ← (Addr16)
STAX r_p	((r_p)) ← (A)
LDAX r_p	(A) ← ((r_p))
SHLD Addr16	(Addr16) ←(L)
	(Addr16+1) ← (H)
LHLD Addr16	(L) ← (Addr16)
	(H) ← (Addr16+1)
OUT Addr8	(Addr8) ← (A)
IN Addr8	(A) ← (Addr8)
PUSH r_{16}	((TOS) ←(r_{16}),(SP)↓↓
POP r_{16}	(r_{16}) ← (TOS), (SP) ↑↑
XTHL	(TOS) ↔ (HL)

02. ARITHMETIC GROUP OF INSTRUCTIONS:

- ALU of 8085 is accumulator based ALU. In most of the Arithmetic operations, Accumulator behaves as one of the source operands and also as the destination operand for result storage

Syntax	Operation
ADD r	(A) ←(A) +(r)
ADC r	(A) ← (A) +(r) +CY
SUB r	(A) ← (A) – (r)
SBB r	(A) ← (A) – (r) –CY

ADD M	(A) ← (A) +((HL))
ADC M	(A) ← (A) +((HL))+CY
SUB M	(A) ← (A) –((HL))
SBB M	(A) ← (A) –((HL))–CY
ADI data8	(A) ← (A)+data8
ACI data 8	(A) ← (A) + data8 + CY
SUI data8	(A) ← (A) – data8
SBI data 8	(A) ← (A) – data8–CY
INR r	(r) ← (r) + 1
INR M	((HL)) ← ((HL))+1
INX r_{16}	(r_{16}) ←(r_{16}) +1
DCR r	(r) ← (r) –1
DCR M	((HL)) ← ((HL)) –1
DCX r_{16}	(r_{16}) ←(r_{16}) –1
DAA	If LN>9 OR AC=1, then add 6 to LN IF HN>9 OR CY=1, then add 6 to HN
DAD r_{16}	(HL) ← (HL)+(r_{16})

ANA M	(A) ← (A) ∧ ((HL))
ORA M	(A) ← (A) ∨ ((HL))
XRA M	(A) ←(A) ⊕ ((HL))
CMP M	(A) – ((HL))
ANI data8	(A) ← (A) ∧ data8
ORI data8	(A) ← (A) ∨ data8
XRI data 8	(A) ←(A) ⊕ data8
CPI data 8	(A) – data8
CMA	Complement (A)
CMC	Complement CY
STC	Set CY
RAL	Rotate (A) to left, including CY
RLC	Rotate (A) to left
RAR	Rotate (A) to Right, including CY
RRC	Rotate (A) to Right

03. LOGICAL GROUP OF INSTRUCTIONS:

❖ ALU of 8085 is accumulator based ALU. In most of the logical operations, Accumulator behaves as one of the source operands and also as the destination operand for result storage.

Syntax	Operation
ANA r	(A) ← (A) ∧ (r)
ORA r	(A) ← (A) ∨ (r)
XRA r	(A) ← (A) ⊕ (r)
CMP r	(A) – (r)

Note:

Effect on Flags for execution of Arithmetical & Logical instructions:

	S	Z	AC	P	CY
Addition, subtraction, comparison, Decimal Adjust instructions	✓	✓	✓	✓	✓
INR & DCR Instructions	✓	✓	✓	✓	x
INX & DCX instructions	x	x	x	x	x
DAD, set & complement, Rotate instructions	x	x	x	x	✓
OR & XOR instructions	✓	✓	0	✓	0
AND Instructions	✓	✓	1	✓	0

04. BRANCH CONTROL GROUP OF INSTRUCTIONS

❖ Branch control instructions (also known as program transfer control instructions) work on program counter & alter its contents. (In general, Program counter consists Next Address.)

❖ There are two types namely unconditional branch control instructions and conditional branch control instructions

4(i) unconditional branch control instructions:

Syntax	Operation
JMP Addr16	(PC) ← Addr16
PCHL	(PC) ← (HL)
CALL Addr16	(TOS) ←(PC), (SP)↓↓ (PC) ←Addr16
RST n	(TOS) ← (PC), (SP) ↓↓ (PC) ← $(8n)_{Hex}$
RET	(PC) ←(TOS), (SP) ↑↑

4(ii) Conditional branch control instructions:

❖ Machine tests a condition first. If test succeeds then only machine branches to target address. otherwise(i.e., if test fails) then machine continues with next address. The testable conditions are 4 flags namely carry, parity, zero & sign.

❖ 8 conditional Jump instructions:

JC	Addr16	Jump if carry (CY = 1?)
JNC	Addr16	Jump if No carry (CY = 0?)
JPE	Addr16	Jump if parity even (P = 1?)
JPO	Addr16	Jump if parity odd (P = 0?)
JZ	Addr16	Jump if zero (Z = 1?)
JNZ	Addr16	Jump if No zero (Z = 0?)
JM	Addr16	Jump if Minus (S = 1?)
JP	Addr16	Jump if Plus (S = 0?)

❖ 8 conditional CALL instructions:

CC	Addr16	Call if carry (CY = 1?)
CNC	Addr16	Call if No carry (CY = 0?)
CPE	Addr16	Call if parity even (P = 1?)
CPO	Addr16	Call if parity odd (P = 0?)
CZ	Addr16	Call if zero (Z = 1?)
CNZ	Addr16	Call if No zero (Z = 0?)
CM	Addr16	Call if Minus (S = 1?)
CP	Addr16	Call if Plus (S = 0?)

❖ 8 conditional RET instructions:

RC	Return if carry (CY = 1?)
RNC	Return if No carry (CY = 0?)
RPE	Return if parity even (P = 1?)
RPO	Return if parity odd (P = 0?)
RZ	Return if zero (Z = 1?)
RNZ	Return if No zero (Z = 0?)
RM	Return if Minus (S = 1?)
RP	Return if Plus (S = 0?)

05. MACHINE CONTROL GROUP OF INSTRUCTIONS:

Syntax	Operation
NOP	No Operation
HLT	μP gets Halted after tristating system Bus
EI	Enables 12 maskable Interrupts, globally
DI	Disables 12 maskable Interrupts, globally
SIM	Set Interrupt Mask
RIM	Read Interrupt Mask

INSTRUCTION LENGTHS

❖ All the instructions of 8085 are classified in to 3 types based on length namely 3 byte instructions, 2 byte instructions & 1 Byte instructions.

❖ 1st byte in any instruction is Opcode

- A 3 byte instruction consists of 16 bit number (which is 16 bit Address). First byte is Opcode, second byte is lower order operand and third byte is higher order operand.
  Ex: JMP 1000H
  CALL 1500H
  LXI H, 4000H
- A 2 byte instruction consists of 8 bit number, which is either 8 bit data or 8 bit port address. First byte is Opcode, second byte is operand.
  Ex: MVI A, 88 H
  ADI 25 H
  OUT F8H
  IN F9H
- A 1 Byte instruction consists neither 8 bit number nor 16 bit number.
  Ex: NOP
  MOV A, B
  ADD D
  DCR C

ADDRESSING MODES OF 8085

- An instruction consists of Opcode and Operand/s. Mode of specifying Address of operands involved in operation is known as Addressing mode.8085 µP's ISA supports following Addressing modes.

(1) Register Addressing mode:
- Operand/s involved is/are register/s.
  Ex: MOV A,B
  ADD B
  DCR C
- Register Addressing mode is also known as Register-direct addressing mode.

(2) Implicit Addressing Mode:
- Operand is not specified explicity in the instruction. µP implicity assumes accumulator as the operand.
  Ex: DAA
  RAL
  CMA
- Implicit addressing mode is also known as Implied addressing mode.

(3) Direct Addressing Mode:
- One of the operands involved in operation is RAM location (or I O Port) whose 16 bit Address(or 8-bit Port address) is directly given in instruction
- Value at address is operand
  Ex: STA 4000H
  LHLD 2884H

(4) Indirect Addressing Mode:
- One of the operands involved in operation is RAM location, whose 16 bit Address is available in given register Pointer.
  Ex: STAX B
  MOV M,C
- Indirect Addressing mode is also known as Register-Indirect Addressing mode

(5) Immediate Addressing Mode:
- Operand is the number given in instruction itself.
  Ex: LXI H, 4000H
  ADI 25H

Example 2.1:
An 8085 assembly Language program is given below.

```
Line 1: MVI A, B5H
     2: MVI B, 0EH
     3: XRI 69H
     4: ADD B
     5: ANI 9BH
     6: CPI 9FH
     7: STA 3010H
     8: HLT
```

The contents of the accumulator just after execution of the ADD instruction in line 4 will be

Ans: EAH

Sol:
```
MVI A, B5H
MVI B, 0EH
XRI 69H ; (A) = 1011 0101
          0110 1001
          ∀
; (A) = 1101 1100 = DCH
```



```
ADD B  : (A) = 1101 1100
         (B) = 0000 1110
         ---------------
       : (A) = 1110 1010 = EAH

ANI 9BH : (A) = 1110 1010
          1001 1011
          ∧
        : (A) = 1000 1010 = 8AH

CPI 9FH ; 9FH=1001 1111
        : 1's compliment of 9FH
        ; is 0110 0000
        ; 2's compliment of 9FH
        ; is 0110 0001 i.e. –9F
        ;   (A) = 1000 1010
        ; –9FH = 0110 0001
                 ---------
                 1110 1011
        ; CY = 1, S = 1, Z = 0,
        ; AC = 0, P = 1
        ; (A) = 1000 1010
        ; (∴ result not stored for CPI)
STA 3010H ; (3010H)←(A) = 8A
HLT       ; Halted
```

The contents of Accumulator just after execution of the ADD instruction in line 4 will be EAH.

Example 2.2:
The following sequence of instructions are executed by an 8085 microprocessor:

```
1000:  LXI SP, 27FF
1003:  CALL 1006
1006:  POP H
```

The contents of the stack pointer SP and the HL register pair on completion or execution of these instructions are.

Ans: HL = 1006, SP = 27FF

Sol:
```
1000H : LXI SP, 27FFH ; (SP) = 27FFH
1003H : CALL 1006H    ; (TOS) ← (PC)
                      ; = 1006H,(SP)↓↓
                      : (PC) ← 1006H
                      ; (SP) = 27FDH
1006H : POP H         ; (HL)←(TOS)
                      ; = 1006H, (SP)↑↑
                      : (SP) = 27FFH
```

Class Room Practice Questions

01. The following instructions have been executed by an 8085 µP **(GATE-EC-97)**

ADDRESS (HEX)	INSTRUCTION
6010:	LXI H, 8A79H
6013:	MOV A, L
6015:	ADD H
6016:	DAA
6017:	MOV H, A
6018:	PCHL

From which address will the next instruction be fetched?
(a) 6019 (b) 0379
(c) 6979 (d) None of the above

Common Data for Questions 02 & 03
Consider an 8085 microprocessor system

02. The following program starts at location 0100H. **(GATE-EC-05)**

```
LXI SP, 00FF
LXI H, 0107
MVI A, 20H
SUB M
```

The contents of accumulator when the program counter reaches 0109H is
(a) 20H (b) 02H
(c) 00H (d) FFH.

03. If in addition following code exists from 0109H onwards ORI 40H,ADD M What will be the result in the accumulator after the last instruction is executed
(a) 40H (b) 20H
(c) 60H (d) 42H

04. In an 8085 processor, the main program calls the subroutine SUB1 given below. When the program returns to the main program after executing SUB1, the value in the accumulator is
(GATE- EI-10)

Address	Opcode	Mnemonics
2000	3E,00	SUB1: MVI A,00H
2002:	CD,05,20	CALL SUB2
2005:	3C	SUB2: INR A
2006:	C9	RET

(a) 00H (b) 01H
(c) 02H (d) 03H

05. An 8085 assembly language program is given as follows. The execution time of each instruction is given against the instruction in terms of T-state.
(GATE- EI-06)

Instruction	T-state
MVI B,0AH	7T
LOOP:MVI C,05H	7T
DCR C	4T
DCR B	4T
JNZ LOOP	10T/7T

The execution time of the program in terms of T-states is
(a) 247 T (b) 250 T
(c) 254 T (d) 257 T

06. A software delay subroutine is written as given below:
(GATE-EE-06)

```
DELAY: MVI H, 255D
       MVI L, 255D
LOOP:  DCR L
       JNZ LOOP
       DCR H
       JNZ LOOP
```

How many times DCR L instruction will be executed?
(a) 255 (b) 510 (c) 65025 (d) 65279

07. An 8085 microprocessor executes "STA 1234H" with starting address location1FFEH (STA copies the contents of the Accumulator to the 16–bit address location). While the instruction is fetched and executed, the sequence of values written at the address pins A15 – A8 is
(GATE-EC-14)(S4)
(a) 1FH, 1FH, 20H, 12H
(b) 1FH, FEH, 1FH, FFH, 12H
(c) 1FH, 1FH, 12H, 12H
(d) 1FH, 1FH, 12H, 20H, 12H

08. In an 8085 microprocessor, which one of the following instructions changes the content of the accumulator?
(GATE - 15)(Set 2)(1M)
(a) MOV B, M (b) PCHL
(c) RNZ (d) SBI BEH

09. Which one of the following 8085 microprocessor programs correctly calculates the product of two 8-bit numbers stored in registers B and C?

(a)
```
MVI A, 00H
JNZ LOOP
CMP C
LOOP: DCR B
HLT
```

(b)
```
MVI A, 00H
CMP C
LOOP: DCR B
JNZ LOOP
HLT
```

(c)
```
MVI A, 00H
LOOP: ADD C
DCR B
JNZ LOOP
HLT
```

(d)
```
MVI A, 00H
ADD C
JNZ LOOP
LOOP: INR B
HLT
```

(GATE - 15)(Set 3)(1M)

10. In a 8085 system, a PUSH operation requires more clock cycles than a POP operation, which one of the following options is the correct reason for this?
(Gate-16)(Set-1)
(a) For POP, the data transceivers remain in the same direction as for instruction fetch (memory to processor), whereas for PUSH their direction has to be reversed
(b) Memory write operations are slower than memory read operations in an 8085 bases system.
(c) The stack pointer needs to be pre-determined before writing registers in a PUSH, whereas a POP operation uses the address already in the stack pointer.
(d) Order of register has to be interchanged for a PUSH operation, whereas POP uses their natural order.

11. In an 8085 microprocessor, the contents of the accumulator and the carry flag are A7 (in hex) and 0, respectively. If the instruction RLC is executed then the contents of the accumulator (in hex) and the carry flag, respectively, will be
(Gate-16)(Set-3)
(a) 4E and 0 (b) 4E and 1
(c) 4F and 0 (d) 4F and 1

Key for CRPQ

01. (c)	02. (c)	03. (c)	04. (c)	05. (c)
06. (d)	07. (a)	08. (d)	09. (c)	10. (c)
11. (d)				

Contribution of ACE in the success of IES toppers