Module 5.

Digital Electronic Fundamentals:.

Difference between analog & digital signals
Number system - Binary Hexadecimal
Conversion - Decimal to Binary
Hexadecimal to decimal and vice-versa

Boolean algebra

Basic and universal gates

Half and Full adder

Multiplexer,

Decoder

SR & JK FlipFlops.

Shift Register, 3-bit Ripple Counter

(refer 10.1 - 10.7 of Text 1)

Basic Communication system

principle of operations of mobile phone

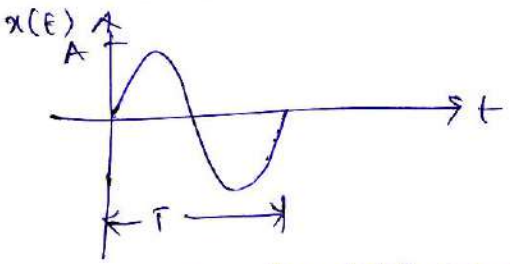
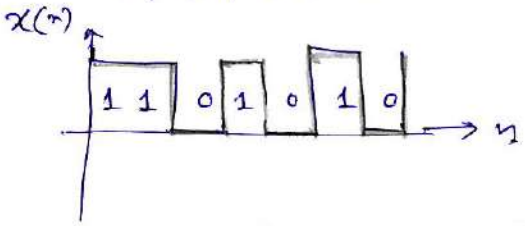
(refer 18.2 and 18.18 of Text 1).

RBT Levels: L1 & L2

Difference between Analog signals & Digital signals

Analog signal : An analog signal is a continuous signal they have infinite number of possible values.
Example sine wave. voice signal. vedio signal.

Digital signal :- These are the discrete & Non continuous signal has finite number of possible values. Example Computer Data. Data storage on memory. It has two possible values binary '1' and '0'.

Analog signal.	Digital signal.
1. An Analog signal is a signal that can have one of infinite number of possible values.	A digital signal is a signal that can have one of finite set of possible values at any time.
2. Analog signal is a continuous signal which represents physical measurements.	Digital signals are non-continuous & discrete in nature.
3. Denoted by sine wave. $x(t) = A\sin(2\pi ft + \phi)$. 	Denoted by square wave. 1 | 0 | 0 | 0 . 
4. uses continuous range of values to represent Information	uses discrete or discontinuous values to represent information.
5. Analog signals are more affected by Noise or distortion during transmission	Digital signals are less immune to Noise or distortion during transmission.
6. They are best suited for transmission of audio & vedio.	They are best suited for transmission of computer Data.

Introduction to Number system.

Number system is the Basis for Counting various items. The representation of Numerals in a specific way is called Number system. We already know the familiar decimal number system with its 10 digits . 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Modern Computers Communicate and operate with binary numbers which use only the digits 0 & 1.

The representation of Decimal number, in binary format takes more digits. For large decimal number, the binary string is large. and ∴ They do not like working with binary numbers. This facts gives rise to three new number system namely. Octal, Hexadecimal and Binary Coded decimal.

Different Number systems

The different Number systems are

- Decimal System
- Hexadecimal Number System
- Binary Number System
- Octal Number System.

Decimal Number System.

Decimal Number can be expressed in terms of units, tens, hundreds, thousands & so on. Decimal Number 5678.9 can be written as $5000 + 600 + 70 + 8 + 0.9 = 5678.9$

The base of the Decimal number system is 10, d, or D. It written as a subscript. The position of a digit with reference to the decimal point determines its value/weight. In above Example the left most digit, which has greatest weight is called most significant digit (MSD). The right most digit, has least weight is called least Significant Bit (LSD).

(1). 5 6 7 8 . 9

In powers of 10 5×10^3 6×10^2 7×10^1 8×10^0 . 9×10^{-1}.

Any number in this decimal system is represented by using the symbols.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Another Eg: $(5689.723)_{10}$

Number	5	6	8	9	.	7	2	3
position.	Digit 3	Digit 2	Digit 1	Digit 0	.	Digit -1	Digit -2	Digit -3.
weightage	10^3	10^2	10^1	10^0		10^{-1}	10^{-2}	10^{-3}

MSD. LSD.

Explain Binary number system.

Binary system with its two digits is a base-two system.
The binary digits (bits) are 0 & 1.
Each symbol that appear in the number is called a Bit.
Each bit has a weightage which depends on the position of the bit in the number.
. The right most bit is called least significant bit. weightage is 2^0. The weightage of bit one left hand side to the Least significant bit is 2^1. Thus as we move to the left the weightage of the bit increases by the multiple of 2.

Eg.

	MSB			LSB
Number	1	0	1	1
position.	bit 3	bit 2	bit 1	bit 0.
weightage.	2^3	2^2	2^1	2^0

$1\times2^3 + 1\times2^1 + 1\times2^0 = 8+2+1 = (11)_{10}$

for fraction.

Number	1	1	0	0	.	1	1
position	bit 3	bit 2	bit 1	bit 0	.	bit -1	bit -2
weightage	2^3	2^2	2^1	2^0	.	2^{-1}	2^{-2}

$1\times2^3 + 1\times2^2 + 1\times2^{-1} + 1\times2^{-2}$
$= 8+4+0.5+0.25 = (12.75)_d = (1100.11)_2$.

Octal number system:-

Any number in this number system is represented by using the symbols.

0, 1, 2, 3, 4, 5, 6, 7.

Consider Example of representing 2356 in octal representation.

- Each symbol in the number is called as Digit.
- Each digit has a weightage which depends on the position of the digit in the number.
- The right most digit is called Least Significant digit.
- The weightage of this digit is 8^0.
- The weightage of the digit on the left hand side of the LSD is 8^1.
- Thus as we move to the left the weightage of the digit increase by the multiple of 8.

Eg.

Number	2	3	4	1
position	Digit 3	Digit 2	Digit 1	Digit 0
weightage	8^3	8^2	8^1	8^0

The value of digit 2 is.

$= 3 \times 8^2 = 3 \times 64 = 192.$

number	1	2	3	.	4	3	1
position.	Digit 2	D-1	D-D	.	Digit -1	-2	-3
weightage	8^2	8^1	8^0	.	8^{-1}	8^{-2}	8^{-3}

The value of Digit 1 is

$= 2 \times 8^1 = 16.$

Digit 2 is

$1 \times 8^2 = 64.$

$= 1\times8^2 + 2\times8^1 + 3\times8^0 + 4\times8^{-1} + 3\times8^{-2} + 1\times8^{-3} = (83.548)_{10}$

Hexadecimal Number System

The hexadecimal number system has a base of 16 having 16 characters. 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

For Example. 32A.1B can be represented in power of 16 as shown below.

3	2	A	.	1	B
Digit 2	Digit 1	Digit 0	.	Digit -1	Digit -2
16^2	16^1	16^0	.	16^{-1}	16^{-2}

$$N = 3\times16^2 + 2\times16^1 + A\times16 + 1\times16^{-1} + B\times16^{-2}.$$

$$= 3\times256 + 2\times16 + 10\times16 + 1\times16^{-1} + 11\times16^{-2}.$$

$$= (960.1054688)_{10}. \quad = (960.1054688)_{10}.$$

Each symbol that appears in the number is called a digit. Each digit has a weightage which depends on the position of the digit in the number.

Number	2	3	5	6
position.	Digit 3	Digit 2	Digit 1	Digit 0.
weightage.	16^3	16^2	16^1	16^0

$$2\times16^3 + 3\times16^2 + 5\times16^1 + 6\times16^0.$$

The value of digit 2 is. $= 2\times16^2$

$= 3\times256$

$= 768.$

$$N = 2\times16^3 + 3\times16^2 + 5\times16^1 + 6\times16^0.$$

N	0	.	1	2	3
P	[illegible]	.	B^{-1}	-2	-3
W	[illegible]	.	16^{-1}	16^{-2}	16^{-3}

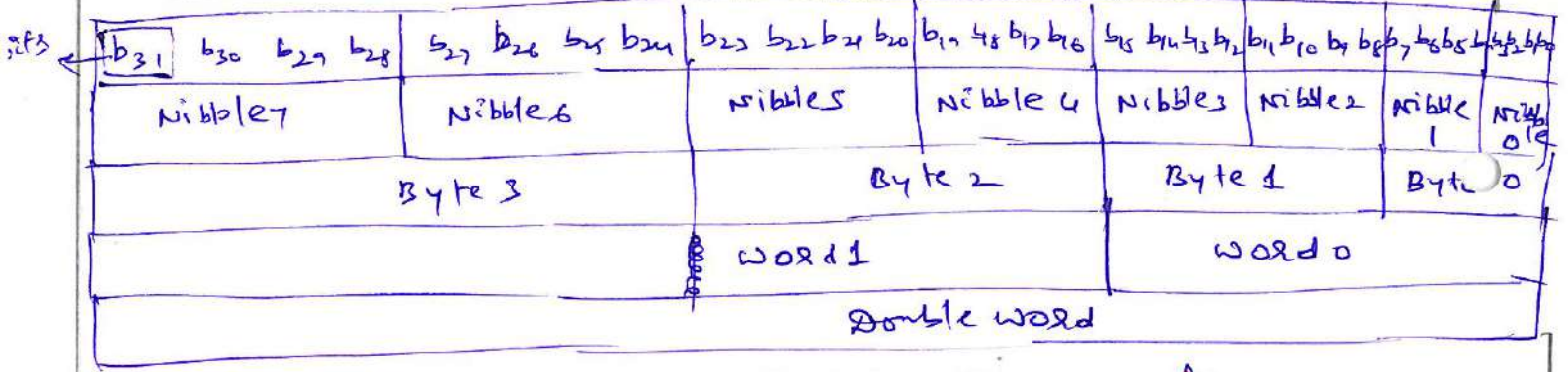
0+

$= (0.07104)_{10}.$

Relationship between decimal binary, Octal & Hexadecimal no.'s

Decimal	Binary	octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F.

What do you mean by bits, bytes, nibbles, word Doubleword.



A Number can be represented by the equation general equation.

$$N = d_n * R^n + d_{n-1} * R^{n-1} + \dots + d_2 * R^2 + d_1 * R^1 + d_0 * R^0$$

N → value of the entire number.

d_n → is the value of the n^{th} digit form the decimal point, and.

R → is the radix or base.

Number of digital $(n+1)$; $(0, 1, 2 \dots n)$

Conversion between Decimal, octal, Hexadecimal and Binary Numbers:-

general :- Conversion from any number system to Decimal.

step 1 : Identify the weightage of each digit in the number system.

step 2 : multiply each digit by the weightage of the digit.

step 3 : add all the products.

Binary to Decimal Conversion

(1) Convert the given binary number 11010 into decimal.

position.	Bit 4.	Bit 3	Bit 2	Bit 1	Bit 0
given Number :	1	1	0	1	0
Weightage :	2^4	2^3	2^2	2^1	2^0

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

$= 16 + 8 + 0 + 2 + 0 = 26$

$\therefore (11010)_2 = (26)_{10}$

(2) $(1101101)_2$ to decimal.

position :	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
given No. :	1	1	0	1	1	0	1
weightage :	2^6	2^5	2^4	2^3	2^2	2^1	2^0

$= 1\times2^6 + 1\times2^5 + 0\times2^4 + 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$

$= 64 + 32 + 0 + 8 + 4 + 0 + 1$

$= (109)_{10}$

Converting Binary Number 1011.110 to Decimal.

position	Bit 3	Bit 2	Bit 1	Bit 0 .	Bit -1	Bit -2	Bit -3
Binary No.	1	0	1	1	1	1	0
Weightage	2^3	2^2	2^1	2^0 .	2^{-1}	2^{-2}	2^{-3}

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

$= (8 + 0 + 2 + 1) + (0.5 + 0.25 + 0)$

$= (11.75)_{10}$

Convert given octal Number 212 into decimal.

position =	Digit 2	Digit 1	Digit 0
given No. =	2	1	2
weightage =	8^2	8^1	8^0

$N = 2\times 8^2 + 1\times 8^1 + 2\times 8^0$.

$= 2\times 64 + 1\times 8 + 2\times 1$

$= 128 + 10 = (138)_{10}$.

Convert $(235.67)_8 = (\ ?\ )_{10}$

position	Digit 2	Digit 1	Digit 0 .	Digit -1	Digit -2
Number	2	3	5 .	6	7
weightage	8^2	8^1	8^0	8^{-1}	8^{-2}

$= 2\times 8^2 + 3\times 8^1 + 5\times 8^0 + 6\times 8^{-1} + 7\times 8^{-2}$

$= 2\times 64 + 24 + 5 + 0.75 + 0.109375$

$= (157.8594)_{10}$

Convert given Hexadecimal No. to Decimal.

(i) $(285.A9)_{16} = (?)_{10}$

position	Digit2	Digit1	Digit0	.	Digit-1	Digit-2
No.	2	8	5	.	A	9
Weightage	16^2	16^1	16^0	.	16^{-1}	16^{-2}

$= 2\times16^2 + 8\times16^1 + 5\times16^0 + (10\times16^{-1}) + (9\times16^{-2})$.

$= 512 + 128 + 5 + 0.625 + 0.03516$

$= (645.66016)_{10}$

(ii). $(235)_{16} = (?)_{10}$.

position	Digit2	Digit1	Digit0.
Number	2	3	5
weigutage	16^2	16^1	16^0

$= 2\times16^2 + 3\times16^1 + 5\times16^0$

$= 2\times256 + 3\times16 + 5\times1$

$= (565)_{10}$.

Case 2:- Conversion of number in decimal Number system to any other Number System.

Integer part Conversion [successive division for Integer part]

- perform repeated division of given decimal value by the base of number system into which the conversion is required.
- After each division, the remainder is taken as one digit / bit in destination number system.
- For the next devision, take the quotient of the previous division as the dividend.

For fractional part Conversion [successive multiplication for fractional part]

perform the repeated multiplication of the fractional part of the decimal Number with the base of the number system. into which the conversion required.

- after each conversion, the integer part is taken as one digit / bit in the destination number system.
- For the next multiplication, take only previous fractional part
- repeat above steps till fractional part of the ~~previous~~ multiplication becomes '0'. (or until desired value we get.

Decimal to Binary:-

① $(25.625)_{10} = (?)_2$

Integer

2 | 25
2 | 12 - 1
2 | 6 - 0
2 | 3 - 0
1 - 1

LSB ↑ MSB

fractional.

0.625 × 2
1 ← 1.25
0.25 × 2
0 ← 0.5 × 2
1 ← 1.0

$= (11001.101)_2$.

② $(264.325)_{10} = (?)_2$.

2 | 264
2 | 132 - 0
2 | 66 - 0
2 | 33 - 0
2 | 16 - 1
2 | 8 - 0
2 | 4 - 0
1 - 0

LSB ↑ MSB

0.325 × 2
0.65 → 0.
× 2
1.3 → 1
0.3 × 2
0.6 → 0.
× 2
1.2 → 1
0.2 × 2
0.4 × 2 - 0
0.8 × 2 - 0
1.6 - 1

$(264.325)_{10} = (10001000.0101001)_2$

Convert the Decimal number 75.62 into octal. Approximate the Result to first four places.

(1) $(75.62)_{10} = (??)_8$.

8 | 75. Remainder.
8 | 9 - 3 ↑ LSD
8 | 1 - 1
0 - 1 MSD

$(113.4753)_8$.

0.62×8
$4.96 \rightarrow 4$
0.96×8
$7.68 \rightarrow 7$
0.68×8
$5.44 \rightarrow 5$
0.44×8
$3.52 \rightarrow 3$

(2). $(384.832)_{10} = (?)_8$.

8 | 384
8 | 48 - 0
6 - 0

$(600.6517)_8$

0.832×8
$6.656 \rightarrow 6$
0.656×8
$5.248 \rightarrow 5$
0.248×8
$1.984 \rightarrow 1$
0.984×8
$7.872 \rightarrow 7$

Solve $(658.825)_{10} = (?)_8$

(2). $(12.125)_{10} = (?)_2$.

Convert 5386.345 decimal into hexadecimal

Conversion of Integer part by successive division method.

16 | 5386
16 | 336 – A
16 | 21 – 0
1 – 5

$(150A.5851\ldots)_{16}$

0.345×16
$5.52 \rightarrow 5.$
0.52×16
$8.32 \rightarrow 8$
0.32×16
$5.12 \rightarrow 5$
0.12×16
$1.92 \rightarrow 1$

Convert $(196.284)_{10} = (?)_{16}$

16 | 196
16 | 12 – 4
D – C

$(C4)_{16}$

$(C4.48B)_{16}$

0.284×16
$4.544 \rightarrow 4$
0.544×16
$8.704 \rightarrow 8$
0.704
$11.264 \rightarrow 11$

Conversion from Binary into octal

Step 1: Arrange group of 3 bits starting from LSB for Integer part & MSB for fractional part, by adding 0's at the end, if required.

Step 2: Write equivalent octal No. for each group of 3-bits.

Decimal	Binary	octal
0	000	0
1	001	1
2	010	2
3	011	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	10

Convert 1010110.0111 to octal equivalent

$(\boxed{0}10\ 101\ 101.011\ 1\boxed{00})_2$ → Adding Zero's to make a group of 3 bits.

Adding 0 to make a group of 3 bits.

$(255.34)_8$

Octal to Binary Conversion :-

Each octal No. is converted individually to its Binary equivalent to get binary conversion.

Convert $(125.62)_8$ to binary.

Step 1 : Write equivalent 3-bit binary no. for each octal digit.

Step 2 : Remove any leading or trailing zeros

$(1 \quad 2 \quad 5 \quad . \quad 6 \quad 2)_8$

$(\boxed{0}0 \quad 010 \quad 101 \quad . \quad 110 \quad 010)_2$

Leading zero. Trailing zero.

$= (1010101.11001)_2$

Conversion from binary into Hexadecimal the following table is used.

Decimal	binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10
17	10001	11

Convert 1101101.101011 into Hexadecimal

$\boxed{0}110 \quad 1101 . \quad 1010 \quad 1110$

$(6 \quad D \quad . \quad AE)_{16}$

Convert binary to Hexadecimal

$(1101101110.100110)_2 = (\quad)_{16}$

$0011 \quad 0110 \quad 1110 . \quad 1001 \quad 1010$

$(3 \quad 6 \quad E \quad . \quad 9 \quad A)_{16}$

HEXADECIMAL to Binary

Convert $(8A9.BA)_{16} = (?)_2$

Step 1 $1000 \quad 1010 \quad 1001 . \quad 1011 \quad 1010$

Step 2 $(100010101001.10111010)_2$

Convert given Hexa to binary.

$(345.AB)_{16} = (?)_2$.

0011 0100 0101 . 1010 1011

$(1101000101.10101011)_2$.

Octal to Hexadecimal ;—

1. Convert octal no. into binary
2. from binary no. convert to Hexadecimal

Convert $(615.25)_8$ to its Hexadecimal equivalent.

Step 1 : (110 001 101 . 010 101.)$_8$

0001 1000 1101 . 0101 0100.

$(1\ 8\ D\ .\ 5\ 4)_{16}$.

Convert Hexadecimal to octal Conversion.

Convert Hexa to binary

Convert binary no. to its octal equivalent.

Convert $(BC66.AF)_{16}$ to octal.

B C 6 6 . A F

Step 1 1011 1100 0110 0110 . 1010 1111

Step 2 001 011 110 001 100 110 . 101 011 110

Step 3 $(1\ 3\ 6\ 1\ 4\ 6\ .\ 5\ 3\ 6)_8$.

Binary representation of decimal Numbers from 0 to 15. is shown below in table. Binary digits in short form we call as bits. These numbers can take only one of the two values.

The primary advantage of using the binary number system as opposed to the 10 discrete line method in Decimal number system is that it minimises the number of lines required to two.

Binary numbers are used in digital systems.

A logic 1 can be represented by a saturated transistor, a light turned ON, a relay energised or a magnet magnetised in a particular direction.

A '0' can be represented as a cut-off transistor, a light turned off, a de-energised relay, or the magnet magnetised in the opposite direction.

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

Converting Decimal to Binary

Decimal to Binary Conversion.

	power of 2	
n	2^n	2^{-n}
0	1	1.0
1	2	0.5
2	4	0.25
3	8	0.125
4	16	0.0625
5	32	0.03125
6	64	0.015625
7	128	0.0078125
8	256	0.00390625
9	512	0.001953125
10	1024	0.0009765625
11	2048	0.00048828125
12	4096	0.000244140625
13	8192	0.0001220703125
14	16348	0.0000 6103515625
15	32768	0.00003051757 8125
16	65536	0.000015258789 0623

Convert $(69)_{10}$ to Binary

From Table 64 is largest number without exceeding 65

$69 > 64$:

$64 = 1000000_2$, $69 - 64 = 5$.

From table 4 is largest number without exceeding 5

$4 = 100_2$ $\quad 5 - 4 = 1$ one is the largest number in the table.

$1 = 1_2$. $\quad 1000000$

100

1

$(69)_{10} = (1000101)_2$

Division method.

$(69)_{10} = (?)_2$

2 | 69

2 | 34 - 1

2 | 17 - 0

2 | 8 - 1

2 | 4 - 0

2 | 2 - 0

1 - 0

$(69)_{10} = (1000101)_2$

Binary Addition and Subtraction.

Similar to Decimal addition.

Binary addition.

The simple addition consists of Four possible elementary operations, namely.

$0+0=0$

$0+1=1$

$1+0=1$

$1+1=10_2$

In above case the first three operation produces a sum whose length is one digit, but when the last operation is performed sum is two digits. Higher significant Bit of this result is called Carry. Lower significant Bit is called Sum.
This operation is known as Half addition.

Full addition:-

The operation which performs addition of three bits {(two significant bits) and one Carry (previous)} is called a Full addition.

Table shows the truth table for Full addition.

A	B	Cin	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

① Add 10111_2 and 10111_2

Carry 1 1 1 1 1
Number 1 1 0 1 1 1 1
Number 2 1 0 1 1 1

$1\ 0\ 0\ 0\ 1\ 1\ 0_2$

② ADD. 1111 and 1111

Carry 1 1 1
Number 1 1 1 1 1
Number 2 1 1 1 1

$1\ 1\ 1\ 1\ 0_2$.

Binary Subtraction :-

① Subtract 1110 from 10000

Borrow
10000
1110.

$0\ 0\ 0\ 1\ 0_2$.

② Subtract 10101 from 101010.

Borrow
101010
10101

$0\ 1\ 0\ 1\ 0\ 1_2$.

Binary multiplication.

① Multiply 1111_2 by 101_2.

1111 * 101

1111
0000
1111

1001011_2.
Result is 8 bit.
$=(0100\ 1011)_2$

② Divide 110110_2 by 110_2.

```
      1001
   ------
110 ) 110110
      110↓↓↓
   ------
      000110
         110.
      ------
           0.
```

These two are called paper methods.

Computer uses. Different multiplication methods.

And different Division method. Such one Long division method is Explained Here

Computer Division method.

It is successive subtraction.

8 bit dividend is to be divided by the devisior in 4-bit representation.

dividend = 00110110 , divisor = 0110.

The Quotient will be formed in the right half of the MQ register and the remainder in the left half.

(1) MQ register = 00110110 D register = 0110.

(2) The divisor is then subtracted from the dividend.
if MSB = 0, then +ve number.
MSB = 1 then negative number.

(3) If the result is positive then the error has occurred for the quotient would be greater than four bits.

MQ : 0011 0110

Subtract D : 0110

→ 1101 0110 Result is negative, the division is valid.

MQ → 0110

Add D : 0110

MQ : 1 0011 0110

Shift MQ left 0110 1100

Subtract D 0110

MQ : 0000 1100 Result is positive

Add 1 to Quotient 0001 1100

MQ :

Shift MQ left 0001 1010

Subtract D 0110

1011 1010 Result is negative

Add D 0110

0001 1010 put a zero Quotient

MQ - 0001 1010

Shift MQ left 0011 0100

Subtract D 0110

1101 0100 Result is negative.

Add D 0110

1 0011 0100

Shift MQ left 0110 1000

Subtract D 0110

0000 1000 Result is positive.

Add 1 to Quotient 1

0000 1001 Final answer.

Remainder Quotient.

If the result is negative then the quotient will be four or less bits that is sufficiently small to be contained in a 4-bit register.

The MQ register is next shifted left one bit & the divisor subtracted from it.

If the result is positive, 1 is added to the LSB (least significant Bit) of the MQ register, which as quotient is accumulated.

If it is negative, the divisor is added to MQ & the MQ shifted left one bit.

This effectively puts a '0' in this bit of quotient.

The process is continued untill the MQ register has been shifted left four bits the number of bits in the D register.

The remainder is then in the left half of the MQ register & the quotient in the right half.

Complement of Binary numbers :-

In Digital computer arithmetic process is done not only with positive Numbers; we also do it with -ve(negative) number.

Processor manages the signed numbers and unsigned Numbers. The 1's & 2's Complement are useful in this case.

What is 1's Complement?

1's Complement of a binary number is just an inversion of individual bit's i.e 1 to 0, & 0 to 1.

1's Complement of $(1010101)_2$ is.

$$=(0101010)_2.$$

2's complemant

The 2's Complemant of binary number is addition of 1 to the 1's Complement of that binary number.

Eg. 2's complement of $(10101011)_2$.

1's Complement 01010100.

ADD 1 to LSB. 1

$(01010101)_2$.

It also holds to Decimal binary number.

Find 2's Complement of 0.1110100.

First 1's complement 0.0001011

ADD 1 to LSB. 1

$(0.0001100)_2$

2's Complement of Floating point Number

Eg ①. (010111.1100) Compute 2's Complement.

Step 1's Complement 101000.0011

ADD. 1 1

$(101000.0100)_2$.

Difference of Two positive Binary Number (m-n).

Consider two binary Numbers m & n. To subtract n from m. we can follow these steps.

1. ADD 2's complement of n to m.
2. Neglect (or Delete) the Carry at the left end, if Carry is present.
3. The result is (m-n).
4. If sum has no carry at the left End, take 2's complement of the sum. Attach a negative sign. The result is (m-n).

Example :-

given m = 11010110 n = 01000101.

Find (m-n) & (n-m).

① 2's Complement of n.

n = 01000101

1'scomp. 10111010.

ADD 1 1

10111011

111

m = 11010110.

10111011

1 10010001

(m-n) = $(10010001)_2$.

(b) n-m

2's complement of m.

m = 11010110.

1's comp. 00101001

Add 1' 1

00101010

n = 01000101

00101010

No carry. 01101111

1's comple. 10010000.

1

10010001

$(n-m) = -10010001$

Binary subtraction using 2's complent method.

$(28)_{10} - (19)_{10}$

Binary number $(28)_{10}$ 011100

$(19)_{10}$ 010011

Let 2's Complement of 19

010011

101100

1

101101

Carry 111

28 011100

19 101101

[1] 001001

Delete

Result = $(001001)_2 = (9)_{10}$.

$(19)_{10} - (28)_{10}$.

19 = 010011

28 = 011100.

2's complant of 28 ⇒ 011100

100011

1

100100.

19 010011

2's 28 100100

110111 → No Carry.

Result is -ve.

Then 2's complant of Result.

→ 001000

001001

The result = $-(001001)_2$.

Subtract using 2's Complement

(i). 4 - 9

4 - 0100

9 - 1001 → 2's complant

0110

1

0111

0100

0111

1011

No carry.

result -ve.

take 2's Complant

0100

1

$(0101)_2$

$-(0101)_2$ $(-5)_{10}$

(ii). 8 - 2

1000 → 8.

0010 → 2

Take 2's complement of 2.

1101.

1

1110

1000

1110

11 0110

neglect

result is +ve.

Result = 0110_2 $(6)_{10}$

Computer Division method.

It uses successive subtraction.

dividend is 01101110.

divisor is 0110.

The Answer is 0000 1001

8 bit Number.

① Add $(28)_{10}$ and $(15)_{10}$ Converting them into binary

$(28)_{10} = 11100$

$(15)_{10} = 1111$

Carry 1 1

11100 — 28

1111 — +15

$(101011)_2$ — 43

2 | 28

2 | 14 — 0

2 | 7 — 0

2 | 3 — 1

1 — 1

Subtraction using 1's Complement Method.

operation M-n using 1' Complement.

1. Take 1's complement of n
2. Result ← M + 1's complement of n.
3. If carry is generated, result is +ve, add in the true form, Add carry to the result to get the final result.
4. If carry is not generated then the result is -ve and in the 1's Complant form.

perform: ① $(28)_{10} - (19)_{10}$ using 1's Complement

$(19)_{10}$ Binary 010011

1's 101100.

$(28)_{10}$ Binary 011100.

```
Carry   1 1 1
28      0 1 1 1 0 0
19  +   1 0 1 1 0 0
    -------------
    [1] 0 0 1 0 0 0
     |              1.
    -------------
    0 0 1 0 0 1   → Result (9)10.
```

② $(19) - (28)_{10}$.

Binary equivalent of 19 = 010011

Binary equivalent of 28 = 011100

Take 1's complement of 011100

100011 → 1's comp 28.

```
Carry    1 1
         0 1 0 0 1 1
         1 0 0 0 1 1
    -------------
         1 1 0 1 1 0   No carry, Negative Result
                       Take 1's complement
         0 0 1 0 0 1
    -------------
```

$-(001001)_2 = (-9)_{10}$ Result.

Boolean Algebra Theorems

In 1854 George Boole invented a systematic treatment of logic for algebraic system.
Boolean algebra is a system of mathematical logic It differs from both ordinary algebra and the binary number system.

Variable :- The symbol which represent an arbitrary elements of an Boolean Algebra is known as variable variables is called biliteral, These can take on two values, binary 1 or 0. It may be any alphabet.

Constant :- The constant value may be '1' or 0.
For Example In Expression $Y = A + 1$ Here A is a variable, 1 is constant.

Complement : A Complement of a variable is represented by a bar over the letter.
Example: Complement of $A = \overline{A}$.
A prime symbol ' is also used instead of bar.

Literal :- Each occurrence of a variable in Boolean function either in a complemented or an uncomplemented form is called a **Literal.**

Boolean function :-
Boolean expressions are constructed by connecting the Boolean constants and variables with the Boolean operations. These Boolean expressions are called Boolean functions.
Example. (1). $f(A, B, C) = (A + \bar{B}) \cdot C$
$f = AB + BC + CA$

Boolean Laws.

AND Law - Multiplication Symbol (·)

$Y = A \text{ AND } B \Rightarrow Y = A \cdot B$.

In AND operation, the output is 1 only if both A and B are 1; otherwise it is zero.

Truth Table

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

OR Law - Addition symbol. (+).

$Y = A \text{ OR } B \Rightarrow Y = A + B$.

In the OR operation, the result is '1' if either A or B or both are 1.

Truth Table.

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

NOT - Complement symbol (Super -, Super')

$Y = \bar{A}$

$Y = A'$

In NOT operation, the output is the result of input

Truth table

A	Y
0	1
1	0

Special Case;

AND operation : $A \cdot 0 = 0$ $A \cdot 1 = A$, $A \cdot A = A$ $A \cdot \bar{A} = 0$

OR operation $A + 0 = A$ $A + 1 = 1$ $A + A = A$ $A + \bar{A} = 1$

NOT operation $\overline{\bar{A}} = A$

$\overline{\overline{\bar{A}}} = \bar{A}$

$\overline{\overline{\overline{\bar{A}}}} = A$.

Identity Law: $1 \cdot A = A$ $\quad 0 + A = A$

Laws of Boolean Algebra.

Three Basic algebra laws.

- Communicative Laws.
- Associative laws.
- Distibutive Law.

Communicative Laws.

Law I : $A + B = B + A$. This states that the order in which the variables are ORed makes no difference in the output.

Proof. Law 1.

$A + B = B + A$.

A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

B	A	B + A
0	0	0
0	1	1
1	0	1
1	1	1

Law 2:

$AB = BA$.

The commutative law of multiplication states that the order in which the variable are ANDed makes no difference in the output.

A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

B	A	BA
0	0	0
0	1	0
1	0	0
1	1	1

Associative Laws.

This law states that in the oRing of several variables the result is the same regardless of the grouping of the variables.

Law 1: $A + (B + C) = (A + B) + C$.

Idempotent law.

Idempotent Law is shown below.

$A + A = A$

$A \cdot A = A$

Redundance law (absorpane law)

$A + AB = A$

$A(A + B) = A$.

Inverse law

$A + \bar{A} = 1$

$A \cdot \bar{A} = 0$.

Null Law

$0 \cdot A = 0$ $\quad 1 + A =$

Law 2 :- $(AB)C = A(BC)$. The associative law of multiplication states that it makes no difference in what order the variable are grouped when ANDing several variables.

Distributive Law :-

Law : $A(B+C) = AB+AC$

The distributive law states that ORing several variables and ANDing the result with a single variable is equivalent to ANDing the result with a single variable with each of the several variables & then ORing the products.

Some identities of Boolean algebra

Name	AND form	OR form.
(1) Identity law	$1 \cdot A = A$	$0 + A = A$
(2) Null Law	$0 \cdot A = 0$	$1 + A = 1$
(3) Idempotent law	$A \cdot A = A$	$A + A = A$
(4) Inverse law	$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$
(5) Commutative law	$A \cdot B = B \cdot A$	$A + B = B + A$
(6) Associative law	$(A \cdot B) \cdot C = A(B \cdot C)$	$(A+B)+C = A+(B+C)$.
(7) Distributive law	$A + BC = (A+B) \cdot (A+C)$	$A(B+C) = A \cdot B + A \cdot C$.
(8) Absorption law	$A \cdot (A+B) = A$	$A + AB = A$
(9) Demorgan's law	$\overline{A \cdot B} = \bar{A} + \bar{B}$	$\overline{A+B} = \bar{A} \cdot \bar{B}$

Proof. Distributive law.

$$\begin{aligned}(A+B)(A+C) &= A \cdot A + AB + AC + BC \\ &= A + AB + AC + BC \\ &= A[1 + (B+C)] + BC \\ &= A + BC \quad \text{as } 1+(B+C) = 1.\end{aligned}$$

Absorption Law

$$\begin{aligned}A(A+B) &= AA + AB \\ &= A + AB \\ A(1+B) &= A.\end{aligned}$$

Demorgan's Theorem :-

$$\overline{A+B} = \overline{A} \cdot \overline{B} \quad \text{and} \quad \overline{A \cdot B} = \overline{A} + \overline{B}$$

It states that any logical binary expressions remains unchanged if we.

1. Change all Variables to their complements.
2. Change all AND operations to ORs.
3. Change all OR operations to ANDs and
4. Take the complemant of the entire expression.

Truth table to prove Demorgan's theorems.

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

Slove :-

$x + \overline{x} \cdot y = x + y$

$1 \cdot x + \overline{x} \cdot y$

$x \cdot (1+y) + \overline{x} y$

$x + xy + \overline{x} y$

$x + y(x + \overline{x})$

$x + y$

prove that.

$x \cdot y + y \cdot z + \overline{y} \cdot z = x \cdot y + z$

Consider LHS.

$x \cdot y + y \cdot z + \overline{y} z \cdot = xy + z(y + \overline{y})$

$= xy + z \cdot 1$

$= xy + z \cdot$ RHS

prove that

$x + y \cdot z = (x+y)(x+z)$

$x + yz$

$1 \cdot x + yz \cdot$

$x \cdot (1+y) + yz$

$x + xy + yz$

$x \cdot 1 + xy + yz$

$x(1+z) + xy + yz$

$= x \cdot 1 + xz + xy + yz$

$= x \cdot x + xy + xz + yz$

$= x(x+y) + z(x+y)$

$= (x+z)(x+y).$

Digital circuits

Digital circuit are divided into two major categories.

- Combinational circuits
- Sequential circuits.

Combinational circuits are circuits where output depends on the present input only.

The sequential circuit produces the output on the basis of both present & previous inputs. Sequential ckts have memory.

Logic gates: are the building blocks of digital circuits.

A logic gate is an electronic device with a single output having one or many inputs. The output is controlled by the input. gates used to create digit ckts & complex IC.

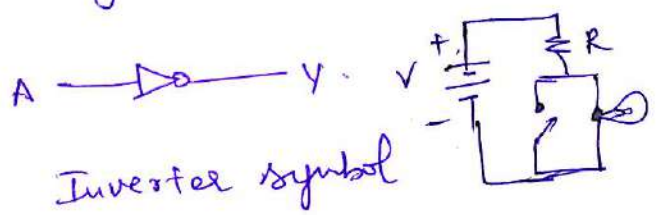
Complex Integrated circuits used to perform several functions

Eg. Microprocessor & Microcontoller.

① Not gate :-

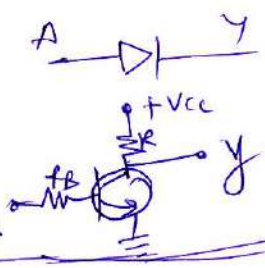
It is also known as an inverter, because it changes the input to its opposite.

A Low voltage input (0) is converted to a high voltage output; a high voltage (1) is converted to low voltage (0).

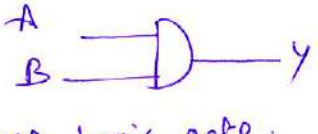


Input A.	o/p. Y
0	1
1	0

Truth table

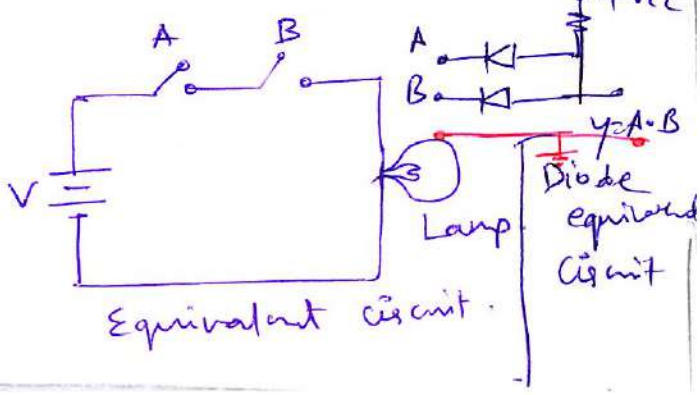


② AND gate :-



Input A	Input B	o/p. Y
0	0	0
0	1	0
1	0	0
1	1	1

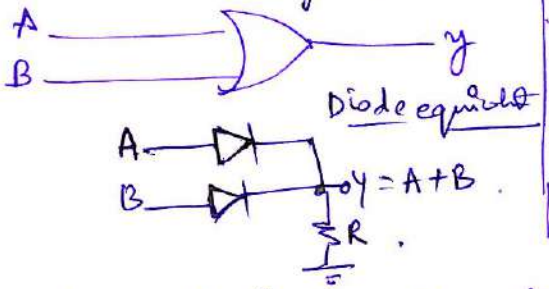
Truth table



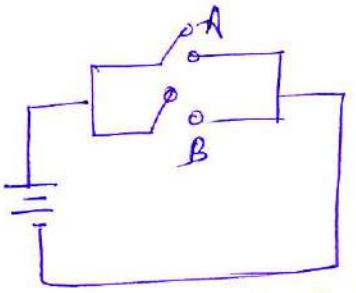
The Output of AND is gate is 1, only when all its Inputs are also '1'. otherwise, its output will be '0'.

OR gate

Symbol



A	B	y
0	0	0
0	1	1
1	0	1
1	1	1

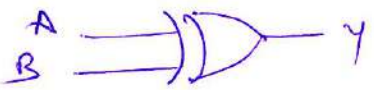


Equivalent circuit.

The output of OR gate will be '0' when all its Inputs are also '0'. otherwise its output will be '1'.

XOR gate:-

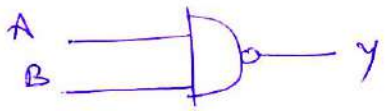
XOR stands for Exclusive OR. An XOR gate compares two Values & if they are different.
So its output will only be at '1' when all its inputs have the same value. otherwise, its output will be '1'.



A	B	y.
0	0	0
0	1	1
1	0	1
1	1	0

NAND gate:-

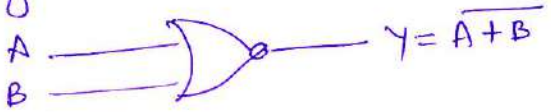
This is an AND gate with the output inverted. The output is high when either of inputs A or B is high. low if neither is high.



A	B	y
0	0	1
0	1	1
1	0	1
1	1	0

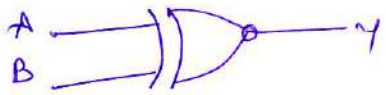
NOR gate:-

The OR gate with the output inverted is called NOR operation. The output is high only when neither A nor B are high.



A	B	y.
0	0	1
0	1	0
1	0	0
1	1	0

XNOR gate:- It is the complement of XOR gate. Its output is 1 when both inputs are equal i.e (0,0) or (1,1). XNOR operation is represented as $Y = A \odot B$.



A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Non-inverter or buffer

The value entered on its input will be found on its output. A typical appln for a buffer is to increase the fan-out of a given logic gate.

Fan out is the maximum number of gates capable of being connected to a given integrated ckt is

A	Y
0	0
1	1

Algebraic simplification.

factorize the following equations.

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

(b) $Y = AB + AC + BD + CD$

(c) $Y = (B + CA)(C + \bar{A}B)$

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

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

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

$= A \cdot 1$

$= A$

(b) $Y = AB + AC + BD + CD$

$= A(B + C) + D(B + C)$

$= (A + D)(B + C)$.

(c) $Y = (B + CA)(C + \bar{A}B)$

$= BC + CCA + \bar{A}BB + \bar{A}ABC$

$= BC + CA + \bar{A}B + 0$; $A\bar{A} = 0$ $B \cdot B = B$ $CC = C$

$= BC + \bar{A}B + AC$.

(d) $Y = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}D + AB\bar{C}\bar{D} + A\bar{B}\bar{C}D$

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

$= \bar{A}\bar{B}\bar{C}\cdot 1 + AB\bar{C}\cdot 1$

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

$= \bar{B}\bar{C}\cdot 1$

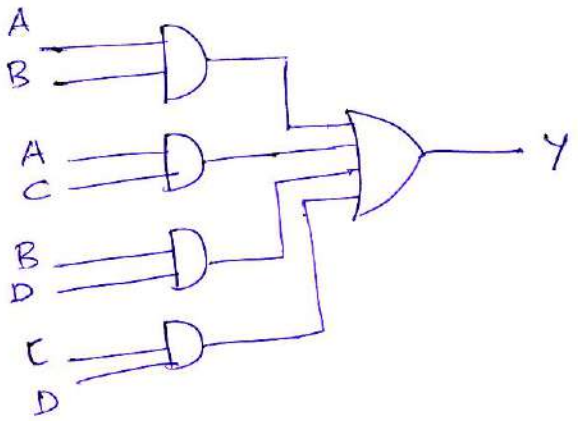
$= \bar{B}\bar{C}$

Compare the AND/OR gate realisation of the Boolean function of Example given below & its reduced form.

$Y = AB + AC + BD + CD$

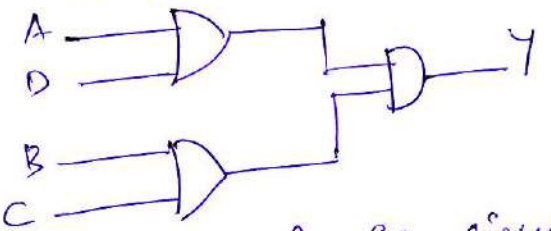
$= A(B+C) + D(B+C)$

$= (A+D)(B+C)$

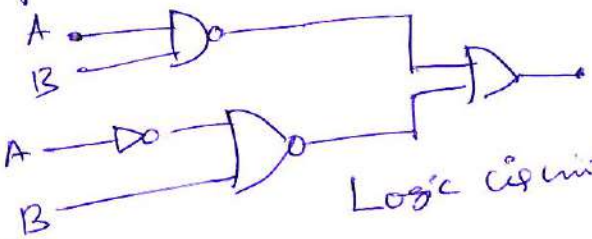


4 AND gates & 1 OR gate necessary implement the above Expression.

The simplified expression consists of only two AND and one OR gate.



Analyze the logic circuit of figure below and show that it can be replaced by a single NAND gate.



Logic circuit analysis.

$f = \overline{AB} + \bar{A} + B$

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

$= \bar{A} + [\bar{B}(1+A)] = \bar{A} + \bar{B}\cdot 1 = \bar{A} + \bar{B} = \overline{A\cdot B}$ = Demorgan's law

Hence the above circuit is simply equivalent to NAND gate.

Sum-of product (SOP).

It is widely used for the canonical form i.e a OR (disjunction) of minterms. Its De morgan dual is a product of Sum (POS).

POS for the canonical form is a conjunction (AND) of maxterms.

minterms are called products because they are the logical AND of a set of variables.

Maxterms are called the sums because they are the logical OR of set of variable.

Example of SOP.

$$F = AB + C\bar{D} + \bar{B}C$$

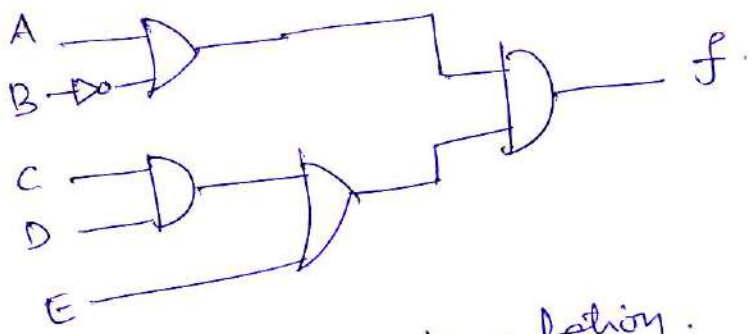
$$Y = A\bar{B} + \bar{A}B$$

POS

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

$$Y = (A+B)(C+D).$$

(1). For the Boolean function $f = (A+\bar{B})(CD+E)$. Obtain the AND-OR Implementation.



AND-OR Implementation.

Simplify and realize using basic gates

$Y = A + \bar{A}B + AB\bar{C}$

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

$= A \cdot 1 + \bar{A}B$

$= A + \bar{A}B$

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

$A + BC = (A + B)(A + C)$

$A + \bar{A} = 1$

$= 1 \cdot (A + B)$

$Y = A + B$

Simplify -

$Y = A + \bar{A}B + ABC + A\bar{C} + AB$

$= A \cdot A + \bar{A}B + ABC + A\bar{C} + 0 + AB$

$= A \cdot A + \bar{A}B + ABC + A\bar{C} + A\bar{A} + AB$

$= A \cdot A + \bar{A}B + A\bar{A} + \bar{A}B + ABC + A\bar{C}$

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

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

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

$= 1 \cdot (A + B) + A[(B + \bar{C})(C + \bar{C})]$

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

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

$= A[1 + B + \bar{C}] + B$

$= A + B$

Simplify

(i) $Y = \bar{A}C + A\bar{C}$

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

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

$Y = \bar{A} + \bar{C}$

(ii) $Y = \overline{AB + \bar{A}B + A}$

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

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

$= 0.$ $\quad | \bar{A} \cdot A = 0.$

(iii) $Y = \overline{(\bar{A}+C)(B+\bar{D})}$

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

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

$= (A \cdot \bar{C}) + \bar{B}D$

$\boxed{Y = A\bar{C} + \bar{B}D}$

Combinational Circuit

These are the circuits whose output is dependent only on the state of its inputs. The output is a pure function of the present input. Does not depends on past values.

Examples of these circuits are

Half adder

Full adder

Multiplexer

Decoder

The following example will be further discussed.

Half adder

It is a simple functional digital circuit built from two logic gates. The half adder adds two one-bit binary numbers (A and B). The output is the sum of the two bits (S) and the carry (C).

The truth table & schematic diagram is shown in figure below

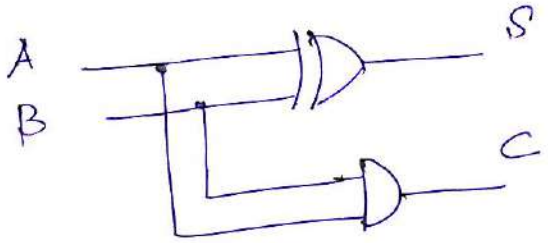
Logic symbol

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

Truth table.

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

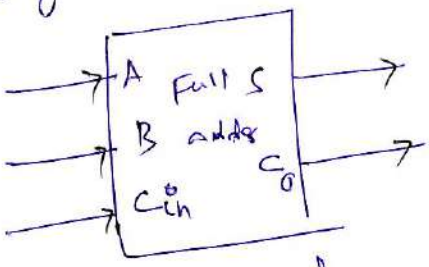
$C = AB$



Full adder

The full adder circuit adds three one-bit binary number (A, B, C), and outputs two one-bit binary numbers, a sum (S) and a Carry (C_o).

The Full adder is simply two half adder joined by an OR gate. The Logic symbol, Truth table & Circuit diagram is shown below.



Logic Symbol

A	B	C_{in}	S	C_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table: Truth Table

From Truth Table

$$S = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$$

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

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

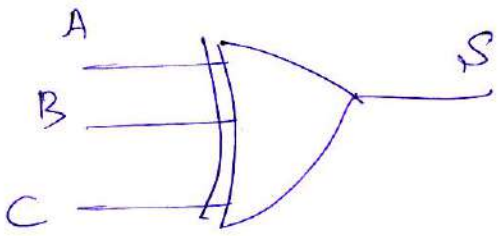
$$= (\overline{A \oplus B})C + (A \oplus B)\bar{C}$$

$$= \text{let } x = A \oplus B$$

$$= \bar{x}C + x\bar{C}$$

$$= x \oplus C$$

$$S = A \oplus B \oplus C$$



$S = A \oplus B \oplus C$

Sum using logic gates is shown in fig

$$\text{Carry} = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

$$= \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC + ABC + ABC$$

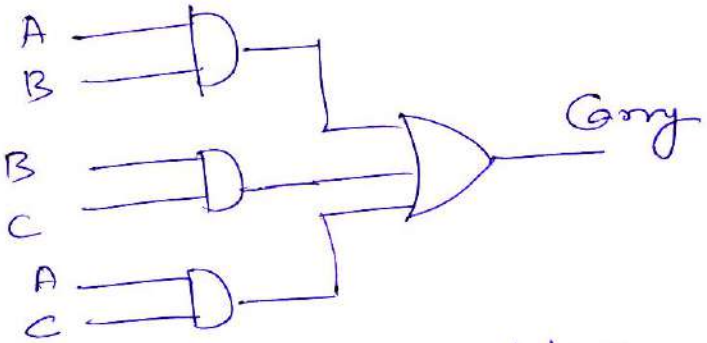
$$= \bar{A}BC + ABC + A\bar{B}C + ABC + AB\bar{C} + ABC$$

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

$$= BC + AC + AB$$

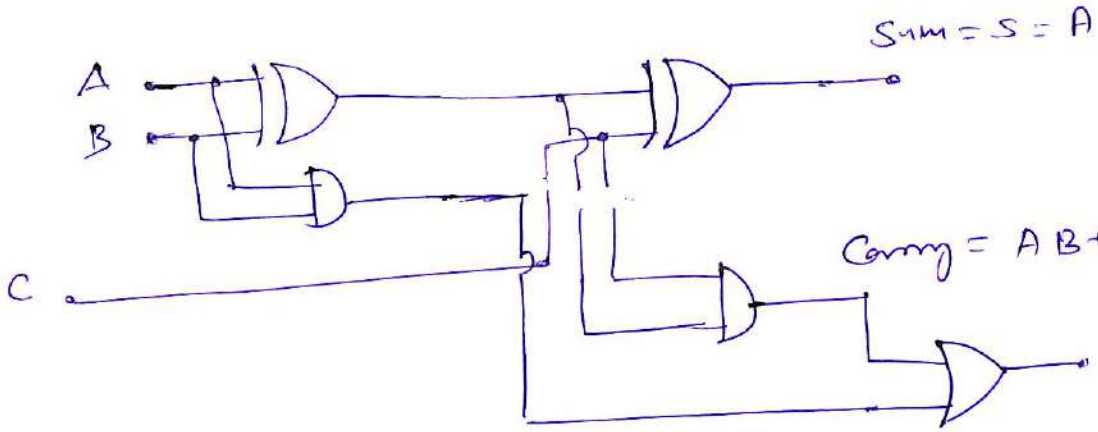
$$\text{Carry} = AB + AC + BC.$$

It can be implemented by three AND gate & one OR gates.



Carry of Full adder

Realizing Full adder using Half adder & OR gates.



$$\text{Carry} = AB + C(A \oplus B)$$

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

$$\text{Carry} = ABC + AB\bar{C} + \bar{A}BC + A\bar{B}C.$$

Ripple - carry - adder

A ripple carry adder is a logic circuit in which the carry-out of each full adder is the carry in of the succeeding next most significant full adder. It is called a ripple carry adder because each carry bit gets rippled into the next stage.

The full adder is a component in a cascade of adders, which adds 8, 16, 32 bit etc binary numbers.

A n-bit adder can be designed by connecting the carry out and carry in lines of n-full adders. Fig shows the 4-bit adder. This adder is called ripple-carry adder. Further, four 4-bit adders can be connected to form a 16-bit adder, etc.

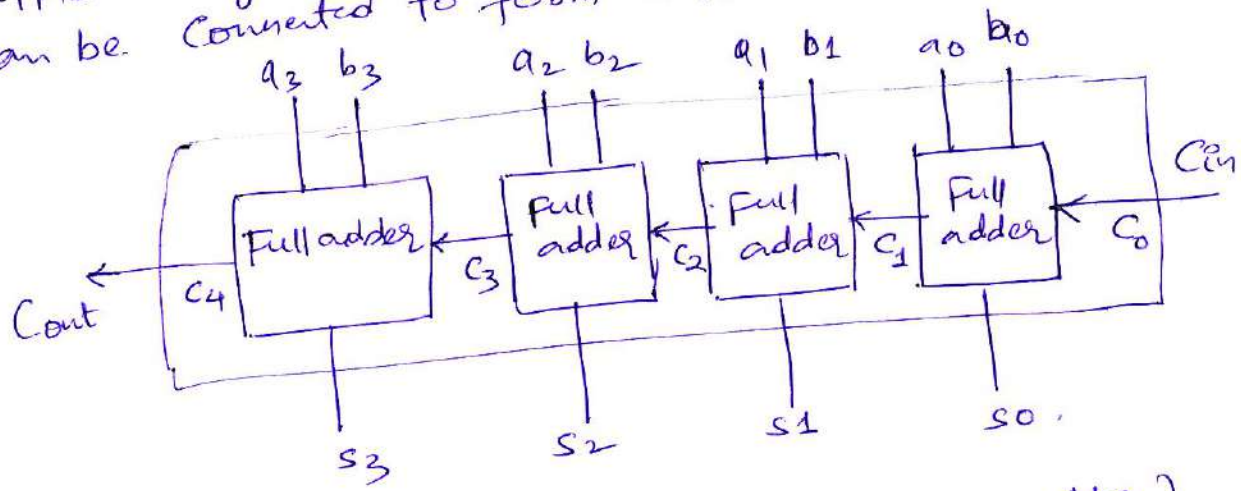


Fig: 4-bit adder (ripple carry adder)

Example ①.

$0 = C_{in}$

$1\ 1\ 1\ 1 = A$

$0\ 0\ 0\ 1 = B$

$1\ 0\ 0\ 0\ 0$

Cout $S_3\ S_2\ S_1\ S_0$

Example ②

$C_{in} = 1$

$A = 1\ 1\ 1\ 1$

$B = 1\ 1\ 1\ 1$

$1\ 1\ 1\ 1\ 1$

Cout $S_3\ S_2\ S_1\ S_0$

Multiplexers

It is a logical circuit that selects one of several analog or digital input signals and forwards the selected input into a single lines.

Multiplex means many to one. Digital multiplexers provide the digital equivalent of an analog selector switch.

A digital multiplexer connects one of '2^n' inputs to a single output line, so that the logical value of the input selected is transferred to the output with the help of select lines.

The objective of a multiplexer is to select one signal from a group of 2^n inputs, to be an output on a single output line.

Block diagram or black box representation of multiplexer (mux) is shown below in figure.

The Data lines $D_0, D_1, D_2, D_3, D_4, D_5, D_6, D_7$ are the data input lines & F is the output line. Lines A, B, and C are called the select lines.

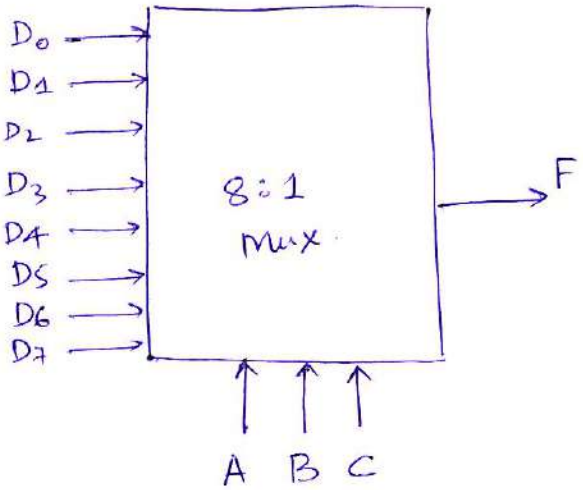
The select lines are interpreted as a three-bit binary number, which is used to choose one of the D-lines to be output on the line F.

The $2^3 = 8$ input lines selected with 3 select lines & one data line is output at one time based on select lines.

Implementation:

mux is designed with a regular pattern of AND and OR gates, as shown in figure.

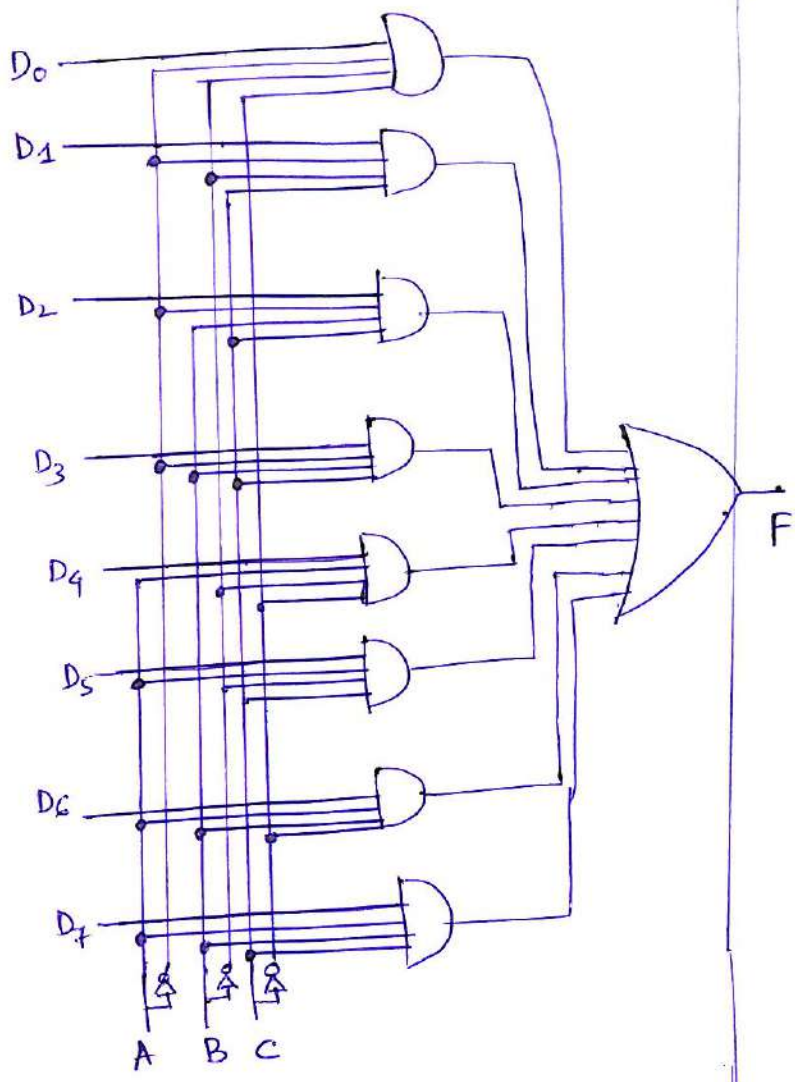
Truth table is shown below.



8-to-1 Multiplexer

select lines			output
A	B	C	F
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	0	0	D_4
1	0	1	D_5
1	1	0	D_6
1	1	1	D_7.

Truth Table.



Implementation of an 8-to-1 Multiplexer.

For Example,
Suppose we want to output $Y = D_5$, then the three inputs to the corresponding AND gate should be 101. Then

if $D_5 = 0$, $F = 0$
$D_5 = 1$, $F = 1$
} $F = D_5$.

The corresponding select inputs are
$5 \rightarrow 101$ or $A\bar{B}C$

Application :-

Choose one of several registers to be used as ALU (Arithmatic Logic unit) input. A 2^n-to-1 multiplexer can be used to implement an arbitrary Boolean function of n variables, by associating each input line of the multiplexer with a row of the truth table for the function.

Decoder :-

A decoder is a multiple-input, multiple output logic circuits which converts coded inputs into coded outputs, where the input and the output codes are different. The input code has lesser bits than the output code. A binary decoder generally has n inputs and 2^n outputs. The binary decoder activates one of the 2^n outputs based on the input applied.

The objective of the decoder is to decode an n-bit binary number, producing a signal on one of 2^n output lines. Figure shows an 3-to-8 decoder.

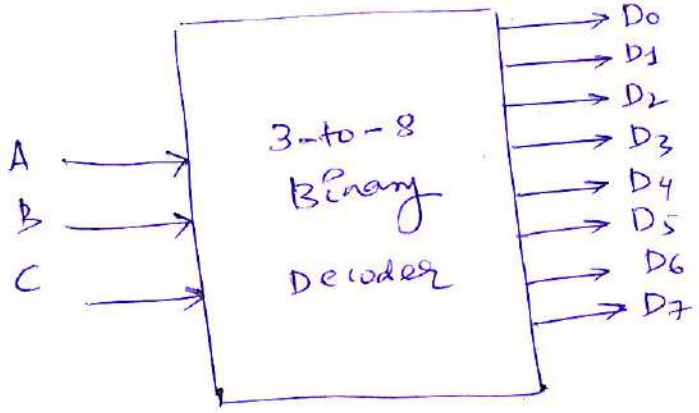


Fig: 3-to-8 decoder

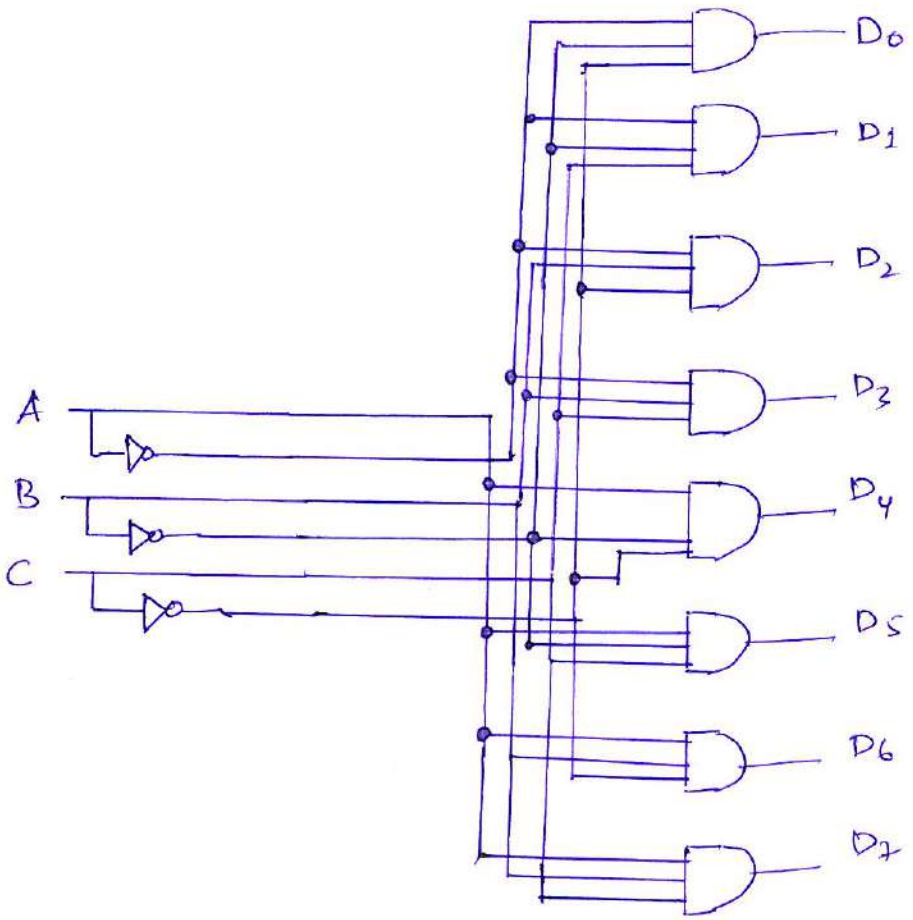
Implementation :-

A decoder is often implementated with an additional input called an "enable" line. When the line is enabled, the circuit is a decoder. When it is disabled, all the outputs are '0'.

The same circuit can be used as a de-multiplexer, which directs a single data input line to one of 2^n output lines, depending on the values of n select lines.

Figure . shows one 3-to-8 decoder implementation.

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



3 to 8 decoder.

SR FlipFlop / SR Latch.

The simplest memory ckt and example of Sequential ckt is SR FlipFlop / SR latch.

A FlipFlop is an electronic ckt which has memory. It is used to store one bit memory. If its output is 1/0, it remains same unless input changes on change in input, its output changes (flips) to 0/1 and then remains constant.

A FlipFlop is a basic element of all sequential system.

Operation :-

A NOR gate produces output 1 only when both inputs are 0, otherwise the output is 1.

Latch is ckt that has two stable states & can be used to store state information.

It holds one bit of data. Latches are example of sequential ckts and are memory elements.

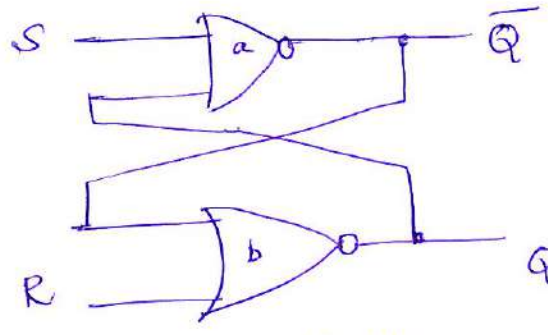
SR latch using NOR gate is explained below.

S means set state output $Q = 1$ and $\overline{Q} = 0$

R means Reset state output $Q = 0$ and $\overline{Q} = 1$.

Previous values are very important here.

Assume SR latch / Flipflop is in **set state**.



SR FlipFlop

Inputs S	Inputs R	present state Q_n	next state Q_{n+1}	
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		Not allowed	

Case ① . Let SR FF be in set state, we make S=0 and R=0. The input to 'a' NOR are (0,1); So $\bar{Q}=0$, & inputs to 'b' NOR (0,0); SO Q=1. Therefore SR FlipFlop remains in set state i.e no State change.

It similarly shows that when the SR FF in reset state we make S=0, R=0 it remains in reset hence this state is called No change & inputs are S=0, R=0.

Case ②.

If output is in set state (Q=1, $\bar{Q}=0$.) we input S=0, R=1, gate b conduct first (1,0) results in Q=0. hence the output changes from set to reset state. But If the output was in reset state (Q=0, $\bar{Q}=1$) and S=0, R=1, gate 'b' conducts first produces the output Q=0. hence the result is for S=0, R=1. Q=0 hence this state is called Reset state

Case ③.

If the output is in Reset state (Q=0, $\bar{Q}=1$) & S=1, R=0, then gate 'a' conducts first & produces output $\bar{Q}=0$ later gate 'b' produces output Q=1. hence For S=1, R=0, the value of Q=1 (set). if the output is in set state, (Q=1, $\bar{Q}=0$) & S=1 and R=0, then also the output Q=1, $\bar{Q}=0$. hence this state output is called as set state.

Case ④

Both S=1 and R=1 are not permitted as this would cause a conflict, & outcome would be uncertain.

The above operations of SR Flipflop are given in table.

Clocked SR Flip Flop.

Within a digital Computer, a clock is used to synchronise changes in the contents of memory elements. A clock is a signal which oscillates between 0 and 1 as shown in figure.

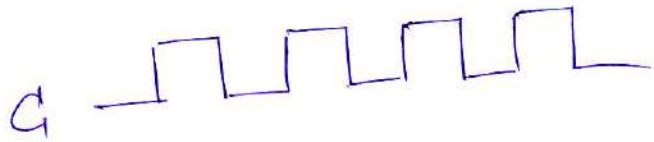


Fig: clock signal.

The clock can be applied to a latch so that change in the latch's value can only occur when the clock is in the '1' (high) state.

The clocked SR latch/FF is shown below.

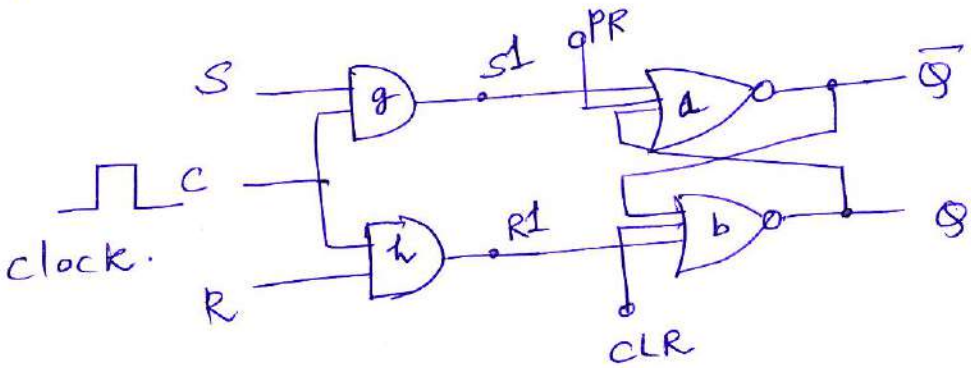


Fig. Clocked SR Flip Flop

When C = 0, the output of both AND gate g and h are zero. So Q cannot change. The latch is non-operational.

When C = 1, output of gate g is 1 then S1 = S & output of gate h is 1. R1 = R, the latch is now operational. The operational procedure is same as discussed in previous section.

The term Level triggered is applied to latches to indicate that its ability to change value depends on the level (low or high) of the clock signal.

The term edge triggered is used with clocked Flip Flop we have positive edge triggering and Negate edge triggering also.

positive edge triggering  The state change from Zero to One ie Low to high.

Negative edge triggering  The output changes when the a clock signal change from high to low.

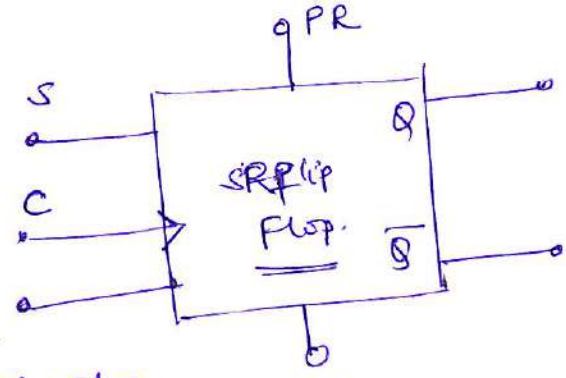
preset and clear:

In figure one additional input is provided to each NOR gate; CLR on Q side NOR and PR on $\bar{Q}$ side NOR. When CLR = 0, PR = 0, the circuit behaves as an SR Flipflop. The block representation is shown below.

In absence of a clock pulse (C=0). the output input to NOR gates are zero. making CLR = 1 & PR = 0. results in Q = 0. $\bar{Q}$ = 1 these by clearing the Flip Flop.

for PR = 1 and CLR = 0, the FF produces Q = 1. $\bar{Q}$ = 0. these by presetting the Flipflop.

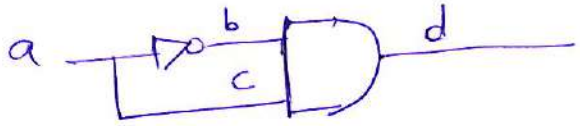
These are asynchronous inputs as this action does not need a clock pulse.



symbol of SR Flip Flop.

FlipFlops.

FlipFlop are used for more precise synchronization. They are edge-triggered. That means they can change with change in the value of the clock signals; that is on its rising or falling edge. This is done by the use of a small circuit, called a pulse generator, as shown in below figure.



pulse generator (edge triggering).

When a is connected to a clock signal, a short pulse will be generated on d, whenever a makes a transition from 0 to 1 and 1 to 0.

There are four commonly used flip-flop.

D-FlipFlop
T-FlipFlop
Rs FlipFlop and
JK FlipFlop.

The behaviour of each of them can be described by a state table. The state table shows one how the values of the FlipFlop changes in response to its inputs.

State table of SR FlipFlop / RS FlipFlop.

Rs (Reset-Set)		
R	S	Qnext
0	0	Q
0	1	1
1	0	0
1	1	undefined

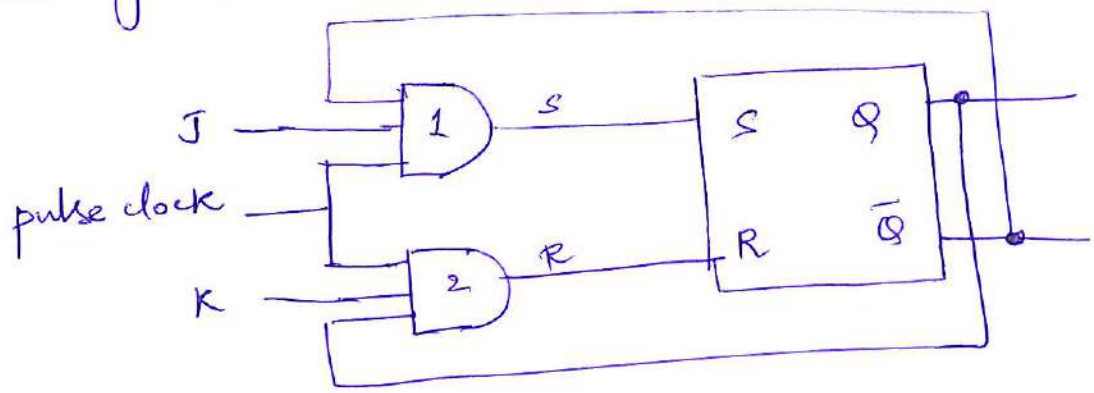
D (Data)	
D	Qnext
0	0
1	1

T (Toggle)	
T	Qnext
0	Q
1	$\overline{Q}$ (Toggle)

JK FlipFlop		
J	K	Qnext
0	0	Q
0	1	0
1	0	1
1	1	$\overline{Q}$

JK FlipFlop.

The undefined state of SR FlipFlop is overcome by converting SR FlipFlop to JK FlipFlop by feeding $\overline{Q}$ to the upper AND & Q to lower AND gate.



JK FlipFlop.

The SR FlipFlop determines the output of JK FlipFlop. The values of S, R are governed by the input J, K and Q.

The outputs for various combinations of inputs (J, K). When a clock pulse is applied is explained.

clock pulse may be positive edge Triggered [PT] or Negative edge triggered).

(i)
(a) when J=0, K=0, Q=0 & $\bar{Q}$=1

Then S=0, R=0 produced by two AND gates, these are inputs to SRFlipflop. The output of SRFF is.

Q_{next}=0, $\bar{Q}_{next}$=1. hence this state is No change.

(b) when J=0, K=0, Q=1. & $\bar{Q}$=0

Here also S=0, R=0 ⇒ Q_{next} = 1, $\bar{Q}_{next}$ = 0, No change

(ii) (a) when J=0, K=1, Q=0 and $\bar{Q}$=1

First AND gate produces S=0. and second AND gate also produces R=0. hence Q_{next}=0, $\bar{Q}_{next}$=1. (No change)

(b) when J=0, K=1 Q=1 and $\bar{Q}$=0.

AND gate 1 produces S=0,
AND gate 2 produces R=1
when S=0, R=1 Then Q_{next} changes to 0 & $\bar{Q}_{next}$ = 1
hence this state is called Reset state.

(iii) (a) when J=1, K=0. Q=0 and $\bar{Q}$=1

First AND gate produces S=1 and AND gate 2 produces R=0, then the output of SR latch Q_{next}=1 (set state) and $\bar{Q}_{next}$=0.

(b) when J=1, K=0, Q=1 and $\bar{Q}$=0

AND gates produces S=0, R=0. hence this state is Nochange because SR latch produces output as Q_{next}=1, & $\bar{Q}_{next}$=0.

all these are shown in table.

Now consider (1,1) = (J,K) inputs which were entries not permitted in an SR-FlipFlop.

(IV)

(a) J=1, K=1, Q=0 ⇒ S=1, R=0, Q_{next} = 1, $\overline{Q}$. Set state

(b) J=1, K=1, Q=1 ⇒ S=0, R=1 Q_{next} = 0, $\overline{Q}$. reset state

This state is called Toggle state because the output toggles ie the state changes from 0 to 1, or 1 to 0.

JK master slave FlipFlop

The toggle state of JK FlipFlop causes the problem. If any clock pulse is too long, the output state will change more than once and the final state of the FlipFlop will be indeterminate. To avoid this problem, a JKFF is constructed with two SR FlipFlops in master slave connection as shown in figure.

In JKFF output is feedback to the input, therefore any changes in the output results in corresponding changes in the input due to this in the positive half of the clock pulse when J & K are both high (1) the output toggles continuously. This condition is known as race around condition.

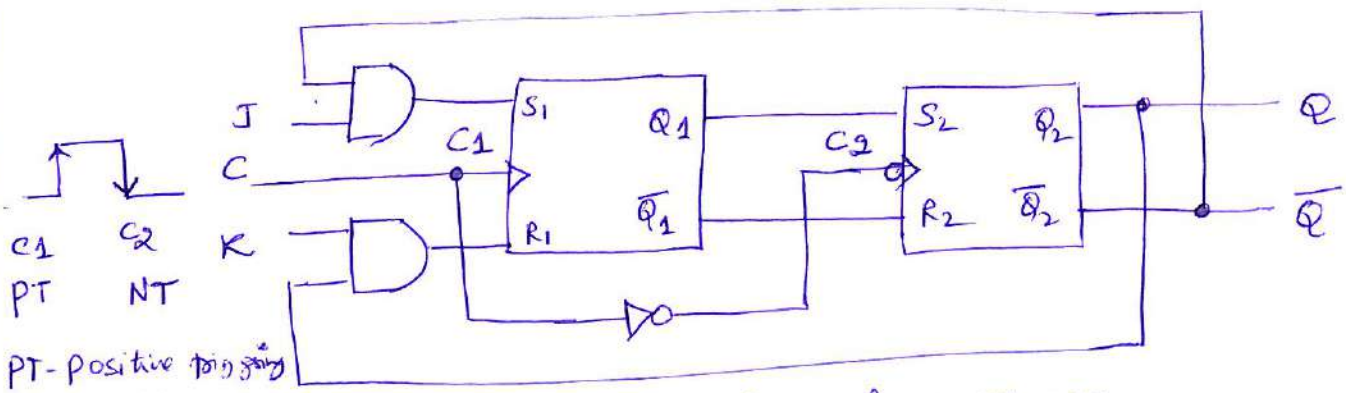


Fig: JK master slave FlipFlop.

Operation of JK masler slave flip flop

In masler slave JKFF : By virtue of masler to slave connection, $S_2 = Q_1$, $R_2 = \overline{Q_1}$.
It means that input to the slave is always 0/1 or 1/0.
∴ As pulse C goes low, $C_2 = 1$ and slave transfers the output of the masler to the system output $Q_1 = Q_2 = Q_1$.

Consider when $J=1, K=0$ and $Q=0$. Then
$S_1 = 1, R_1 = 0$ when clock pulse $C_1 = C = 1$ then
$Q_1 = 1$ (set) $= S_2$. Its connected to slave at negative triggering i.e $C=0, C_2 = 1$ the output $Q_2 = Q = 1$ (set).

(i)
$J=1\ K=0, Q=0 \Rightarrow S_1=1, R_1=0 \xrightarrow{C=C_1=1} Q_1 = 1 = S_2 \xrightarrow{C=0, C_2=1} Q_2 = Q = 1$
$J=1\ K=0, Q=1 \Rightarrow S_1=0, R_1=0 \xrightarrow{C=C_1=1} Q_1 = 1 = S_2 \xrightarrow{C=0, C_2=1} Q_2 = Q = 1$

(ii)
$J=0, K=0$, $Q=0$.
masler & slave Flipflop remains in previous state have this state is called Nochange state

Set

Nochange

(iii)
$J=0, K=1, Q=1 \Rightarrow S_1=0, R_1=1 \xrightarrow{C=C_1=1} Q_1=0 \Rightarrow S_2 \xrightarrow{C=0=C_2=1} Q_2 = Q = 0.$
$J=0, K=1\ Q=0 \Rightarrow S_1=0, R_1=0 \Rightarrow Q_1=0 \Rightarrow S_2 \xrightarrow{C=0=C_2=1} Q_2 = Q = 0.$
This state is called Reset state.

Reset state

(iv). For $J=1, K=1$ toggling should happen at output without any indeterminate solution.

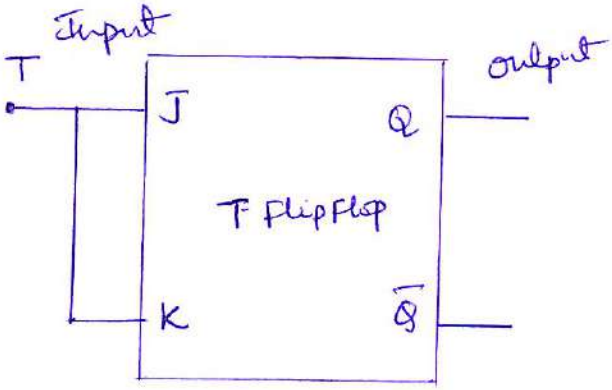
$J=1, K=1, Q=1 \Rightarrow S_1=0, R_1=1 \xrightarrow{C=C_1=1} Q_1=0 \Rightarrow S_2=0, R_2=1 \xrightarrow[C_2=1]{C=0} Q=0=\overline{Q}$
$J=1, K=1, Q=0 \Rightarrow S_1=1, R_1=0 \xrightarrow{C=C_1=1} Q_1=1 \Rightarrow S_2=1, R_2=0 \xrightarrow[C_2=1]{C=0} Q=1=\overline{Q}$

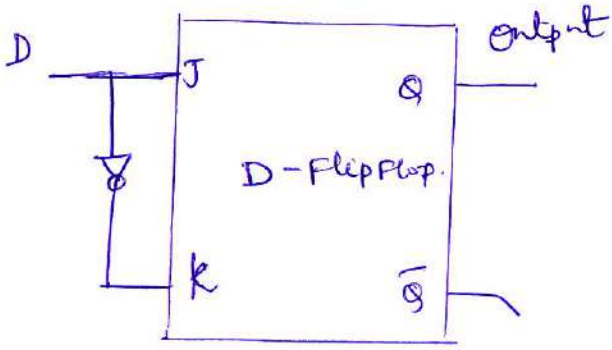
Thus confirms the toggle operation of the the JK masler-slave Flip-flop. The masler FF operates when $C=1$ & slave operates when $C=0$. when slave operates masler is inoperative and unwanted toggling cannot occur. hence JK masler slave avoids race-around condition of JK Flipflop.

The sequential action of slave (after master has become inoperative) and C goes low, the JK master-slave FF is also called as trailing edge triggered Flip Flop.

T-(Toggle) Flip Flop

In this Flip Flop J and K inputs are tied together. when T=0 with the application of the clock pulse the output Q remains in the same state. When T=1 output toggles from 0 to 1 or 1 to 0 state with every clock pulse.





D-(Data) Flip Flop

D Flip Flop also called Data flip-flop has one input D. with the application of each clock pulse the data (0 or 1) is transferred to the output Q. The modified JK FF as D FF is shown above. also Refer Truth Table on previous pages.

Shift register

The shift register is a sequential circuit that can be used for the storage or the transfer of binary data.

shift register is digital circuit constituted by cascade of flipFlops, sharing the same clock, in which the output of each FlipFlop is connected to the data input of the next flip-Flop in the chain, results in shifts of data.

In Figure. shift register consists of two RS FlipFlops when a clock pulse occurs, the value of the left FlipFlop is copied to the right FlipFlop. This is known as a shift register.

FlipFlops are used within a CPU (central processing unit) to implement register. A register is an ordered group of n-FlipFlops.

For Example, in the Intel architecture, EAX is a 32-bit register. It can hold a 32-bit binary register.

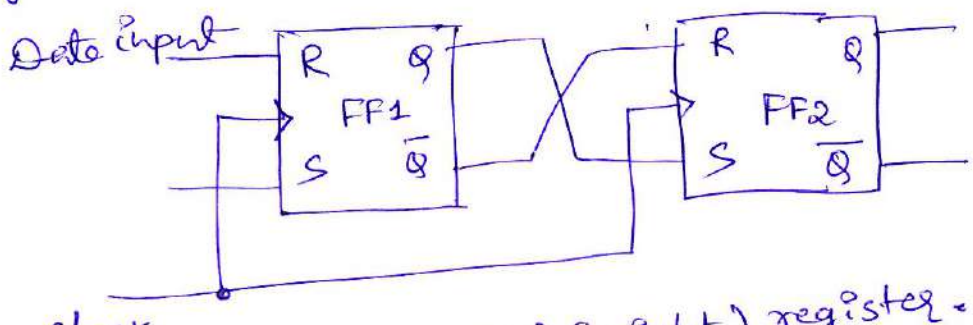


Fig. shift (right) register.

Simple shift register, the date string presented at Data in & is shifted right one stage each time. The bit from FF1 is shifted into FF2 input and upon clock pulse shifted at output of FF2.

The binary information in a register can be moved from stage to stage within the register or into or out of register upon application of clock pulses.

This type of movement or shifting is essential for certain arithmetic and logic operations used in processors. Shift register applications involves storage and transfer of data in a digital system.

Figure shows a four bit shift register employing D as flipflops, which trailing edge triggered indicated by a small circle at the back of the clock triangle.

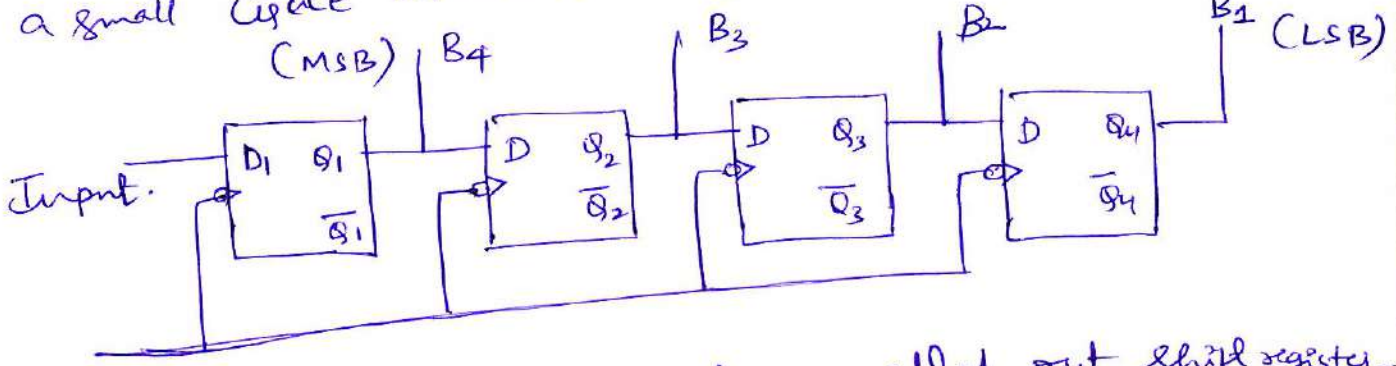


Figure: Serial input - parallel out shift register

Let the input sequence be $A_4 A_3 A_2 A_1$ and initial state of the register be $Q_1 = Q_2 = Q_3 = Q_4 = 0$.

Before applying clock pulse, the four data (D) bits are.

$D_1 = A_1, D_2 = Q_1 = 0$, $D_3 = Q_2 = 0$. and $D_4 = Q_3 = 0$.

After the first pulse arrives, data (D) in all the FlipFlop shifts to Qs and the state of register read from left to right becomes A1,000. After the second pulse, the state of the register becomes $A_2, A_1 00$ and finally at the end of Fourth pulse it is $A_4 A_3, A_2, A_1$. The register state can be read simultaneously at $B_4, B_3 B_2 B_1$. This is called Serial-in parallel-out shift register. After four more clock pulses, the output can be read serially at B1.

The fourth Table is shown below.

Serial I/p Data	shift pulse	Q_1	Q_2	Q_3	Q_4
–	–	x	x	x	x
0	T1	0	x	x	x
1	T2	1	0	x	x
0	T3	0	1	0	x
1	T4	1	0	1	0
0	T5	x	1	0	1
x	T6	x	x	1	0
x	T7	x	x	x	1
x	T8	x	x	x	x
x					

Binary Counter :-

A Counter is a sequential circuit that counts the number of input pulses.

A Counter that counts in terms of binary is called a binary Counter. The count output of n-bit binary Counter is 2^n states.

Therefore, it can count from 0 to $(2^n - 1)$. The number of states of Counter is called as it modulus (m).

For n-bit Counter, $m \leq 2^n$.

Based on the concept of applying inputs to FF to start Counting we have two types of Counter.

- Asynchronous Counter
- Synchronous Counter.

In Asynchronous Counter, the FF are clocked sequentially while in a Synchronous Counter, they are clocked simultaneously.

In Asynchronous Counter, time delay of each FF get added & this Count action is slower than in Synchronous Counter.

Asynchronous Counters (ripple Counters)

T FlipFlops are used in these counters. A 3 bit binary ripple counter is shown in figure below.

The small circles at the back of the clock input stand for the fact that these FF are triggered by trailing edge of the input pulse (1 going 0).

For Toggling, all T inputs are kept high (1).

From left to right, the first T-FlipFlop receives pulses from the counter. The other two receive pulses from the output of the preceding T-FlipFlop.

The pulses kind of ripple through & hence, the name ripple counter. 3-bit ripple counter is presented in table. At 8th pulses, the counter resets to 0.

The Timing diagram of the counter is shown below in figure. It is observed that pulse frequency gets divided by 2 at each stage.

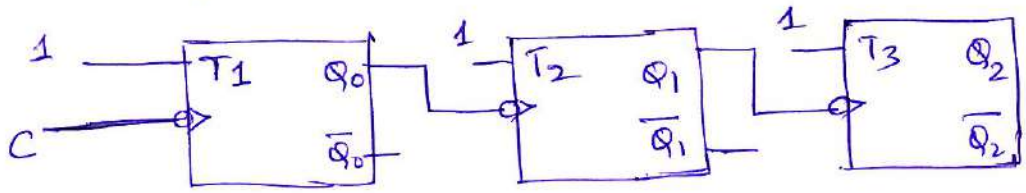


Fig:- 3 bit (modulo 8) ripple counter.

Truth table of 3-bit ripple Counter

Q_2	Q_1	Q_0	Input pulse Count.
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7
0	0	0	8

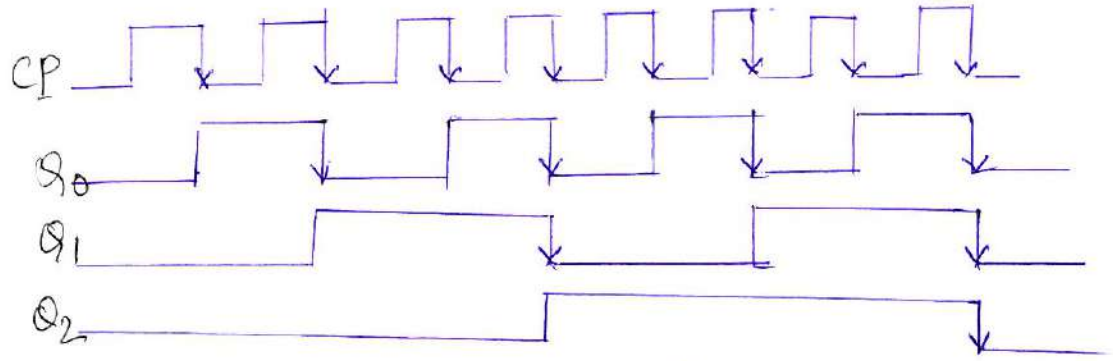
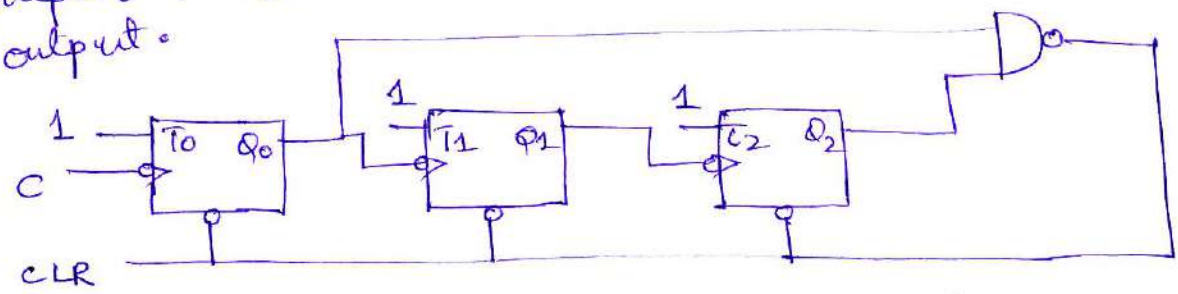


Fig. Timing diagram of 3-bit ripple counter.

Modulo-5 ripple Counter

It is possible to design modulo-5 ripple counter. As $2^3 = 8 > 5$ we need 3 T-FlipFlops. The counter circuit is shown in figure. where the CLR (clear) terminals are also shown. As the pulse count reaches $5 \rightarrow 101$. The counter must reset to zero (000). As outer 1's combination is unique to count 5, these are fed to a NAND gate, which produces 0 output to clear the three FlipFlops & produces 000 as output.



modulo-5 ripple counter.

C	Q_2	Q_1	Q_0
↓	0	0	0
↓	0	0	1
↓	0	1	0
↓	0	1	1
↓	1	0	0
↓	0	0	0

Synchronous Counter

In an asynchronous counter, as the pulse ripples through all the flipflops, the time delay adds up.

In synchronous counter, the clock pulse is applied to all the flip-flop simultaneously and so the time delay is that of one counter. Counters are fast in opelation.

For modulo -16 Synchronous Counter, Four Flip-Flop are needed. Constonted by using T-FlipFlop.

The Truth table for 4-bit Synchronous Counter is shown below.

Truth Table.

Input clock.	Q_3	Q_2	Q_1	Q_0	T_3	T_2	T_1	T_0
0	0	0	0	0	0	0	0	1
1	0	0	0	1	0	0	1	1
2	0	0	1	0	0	0	0	1
3	0	0	1	1	0	1	1	1
4	0	1	0	0	0	0	0	1
5	0	1	0	1	0	0	1	1
6	0	1	1	0	0	0	0	1
7	0	1	1	1	1	1	1	1
8	1	0	0	0	0	0	0	1
9	1	0	0	1	0	0	1	1
10	1	0	1	0	0	0	0	1
11	1	0	1	1	0	1	1	1
12	1	1	0	0	0	0	0	1
13	1	1	0	1	0	0	1	1
14	1	1	1	0	0	0	0	1
15	1	1	1	1	1	1	1	1
0	0	0	0	0				

From Table we find that

Q_0 continuously toggles, therefore $T_0 = 1$

Q_1 toggles when $Q_0 = 1$, therefore $T_1 = Q_0$

Q_2 toggles when $Q_0 = Q_1 = 1$ therefore $T_2 = Q_1 Q_0$

Q_3 toggles when $Q_0 = Q_1 = Q_2 = 1$ therefore $T_3 = Q_2 Q_1 Q_0$

The corresponding $T = 1$ are indicated in the same table. To generate T_2 & T_3, two AND gates are needed. To meet these requirements, the circuit diagram of the synchronous counter is drawn in figure.

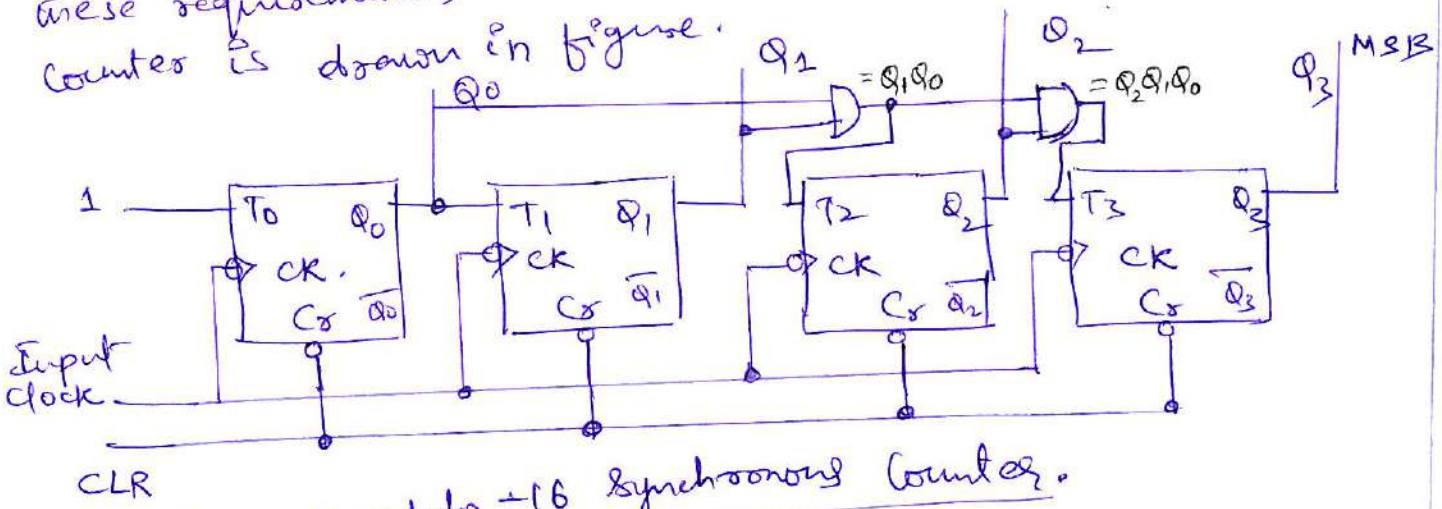


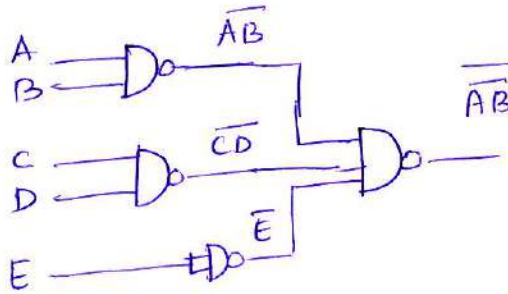
Fig. Modulo -16 Synchronous Counter.

(1) Consider the Boolean expression $Y = AB + CD + E$ Implement it using NAND gates.

$$Y = AB + CD + E$$

$$\overline{Y} = \overline{AB + CD + E}$$

$$= \overline{\overline{AB} \cdot \overline{CD} \cdot \overline{E}}$$



(2) Realise the following logic function using NAND gates only

$$Y = \overline{AB + \overline{A}\,\overline{B} + \overline{A}B}$$

$$\overline{Y} = \overline{AB + \overline{A}\,\overline{B} + \overline{A}B}$$

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

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

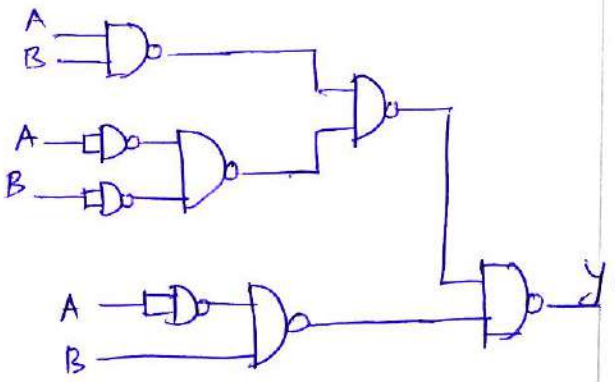
$$= \overline{(AB + \overline{A}\,\overline{B}) \cdot \overline{\overline{A} \cdot B}}$$

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

$$y = \overline{\overline{AB} \cdot \overline{\overline{A}\,\overline{B}}} + \overline{\overline{A}B}$$

$$\overline{y} = \overline{\overline{AB} \cdot \overline{\overline{A}\,\overline{B}} + \overline{A}B}$$

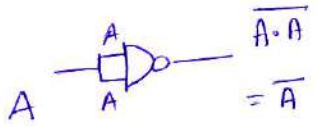
$$y = \overline{\overline{\overline{AB} \cdot \overline{\overline{A}\,\overline{B}} \cdot \overline{\overline{A}B}}}$$



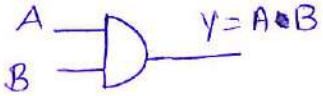
Realize Basic gate using NAND gate

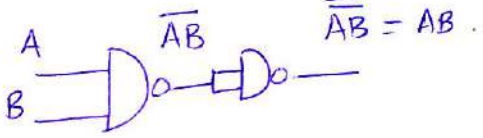
NOT gate

only NAND realization

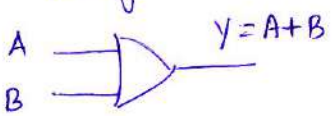


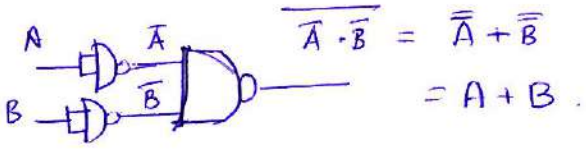
AND gate





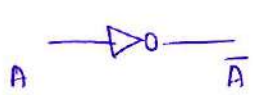
OR gate



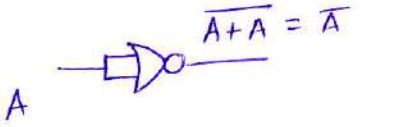


Realize Basic gate using NOR gate

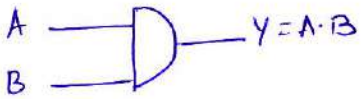
NOT gate

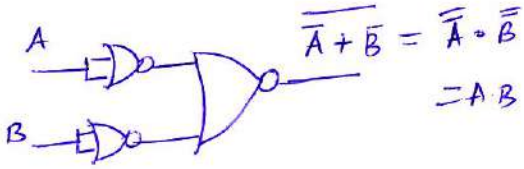


only NOR realization

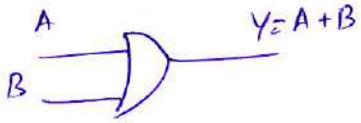


AND gate

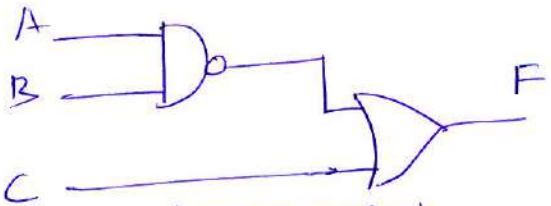




OR gate.



Write the truth table for the given circuit



Inputs A B C	output $\overline{A}B$	output $\overline{A}B + C = F$
0 0 0	1	1
0 0 1	1	1
0 1 0	1	1
0 1 1	1	1
1 0 0	1	1
1 0 1	1	1
1 1 0	0	0
1 1 1	0	1

Simplify the Boolean equation & then draw its logic circuit with appropriate gates.

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

$$= \overset{0}{\cancel{A\overline{A}}} + AB + AC + \overline{A}B + B \cdot B + BC + \overline{A}\,\overline{C} + B\overline{C} + \overset{0}{\cancel{C\overline{C}}}$$

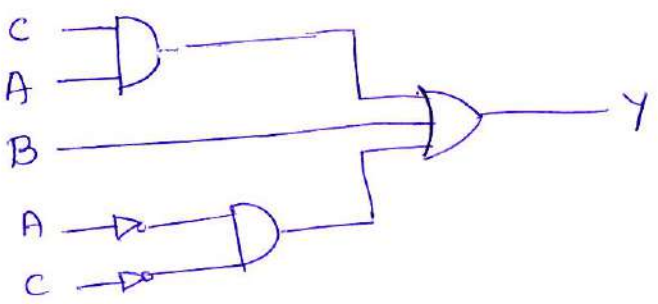
$$= AB + AC + \overline{A}B + B + BC + \overline{A}\,\overline{C} + B\overline{C}$$

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

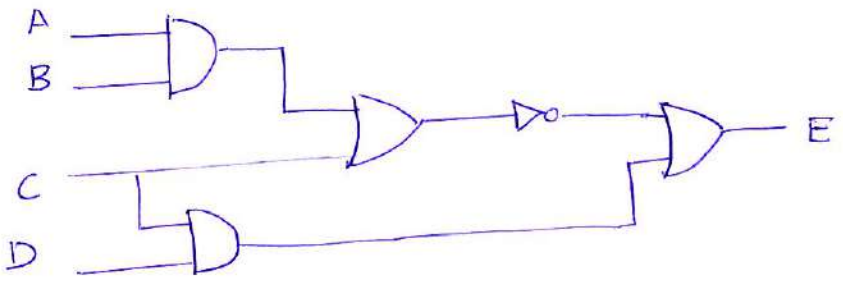
$$= B(1 + A + \overline{A} + C + \overline{C}) + AC + \overline{A}\,\overline{C}$$

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

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

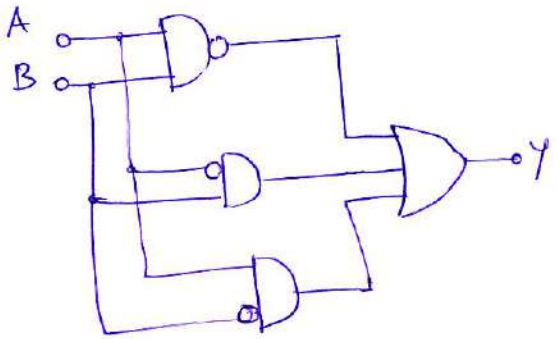


19. Write the Boolean expression for the given circuit in Fig. E = ?



$$E = (\overline{AB+C}) + CD$$

For the Logic circuit of Fig 10.32 write the Boolean expression for y in simple form



$$Y = \overline{AB} + \bar{A}\cdot B + A\bar{B}$$

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

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

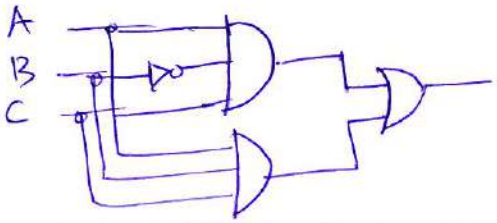
$$= \bar{A} + \bar{B}$$

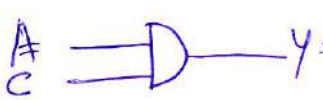
$$\boxed{Y = \overline{A\cdot B}}$$

Draw the Logic circuit for $Y = A\bar{B}C + ABC$ & then Simplify and draw the Simplified circuit

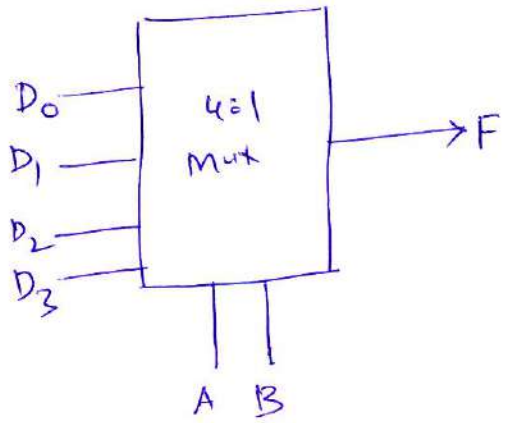
$$Y = A\bar{B}C + ABC$$

$$Y = AC(\bar{B}+B) = AC$$

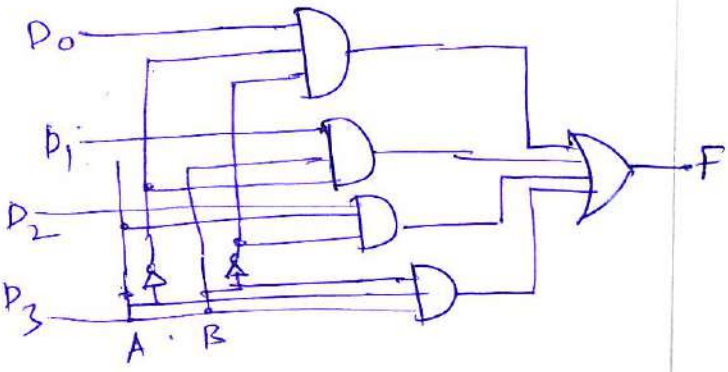




Implement the 4:1 multiplexer using basic gates.



4:1 multiplexer



Implementation of an 4:1 mux

6. Subtract the following into Decimal numbers in 8 bit 2's Complement form.

$+45 - 56$

$(1100001)_2 - (1111000)_2$

$m - n$

$m = (00110001)_2$

$n = (00111000)_2$

2's complement of n $\Rightarrow$ 1's n = 11000111

n = 00111000

$$\begin{array}{r} 11000111 \\ \quad\quad\quad 1 \\ \hline 11001000 \end{array}$$

$$\begin{array}{r} M = 00110001 \\ 2's\ n = 11001000 \\ \hline 11111001 \end{array}$$

$\rightarrow$ No Carry, result is negative.

$$\begin{array}{r} 00000110 \\ \quad\quad\quad 1 \\ \hline -(0000111)_2 \end{array} \quad (-7)_{10}$$

2 | 45
2 | 24 - 1
2 | 12 - 0
2 | 6 - 0
2 | 3 - 0
1 - 1

2 | 56
2 | 28 - 0
2 | 14 - 0
2 | 7 - 0
2 | 3 - 1
1 - 1

perform the indicated operations in binary.

(a) $\Rightarrow (32)_8 + (73)_8$

$32 = 011\ 010$

$\quad\ 111\ 011$

$1010\ 101$

$(125)_8$.

(b) $(175)_8 - (114)_8$

$175 = 001\ 111\ 101$

$114 = 001\ 001\ 100$

000110001

$(061)_8$.

(c) $(7E)_{16} + (AD)_{16}$

$\quad\ 11111$

$(7E)_{16} = 0111\ 1110$

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

100101011

$(12B)_{16}$

(d) $(BC)_{16} - (84)_{16}$

$(BC)_{16} = 1011\ 1100$

$(84)_{16} = 1000\ 0100$

$0011\ 1000$

$(38)_{16}$

Basic Communication System.

communication is application of Electrical Technology. we use satellite television, fax machine cellular phone etc. Communication systems include the generation, storage and transmission of information. The basic elements of Communication system are trasmitter, receiver and Communication channels.

Elements of Communication system

Communication systems are used to transfer information from a generation point to the place where it is needed/ processed.

- The information at the generation point is not in the form that can travel long distance through the channel. we use a device called modulator cum transmitter to modulate data for transmission.
- In the receiving end, the information must undergo the reverse process such as demodulation/decoding.
- The main elements are.

1. Source
2. Input transducer
3. modulator and transmitter
4. Communication channel
5. Demodulator and receiver
6. output transducer.
7. Destination.

The source is mostly analog & sometimes it may be digital. Fig① shows basic elements of a Communication Syslem

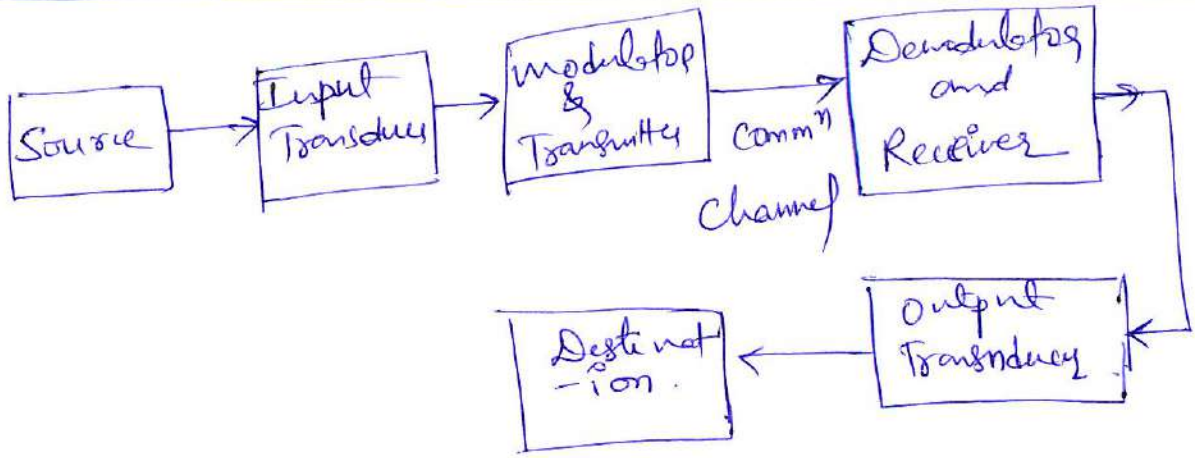


Figure ① : Communication System.

Source is the information we are sending the example of sources are analog audio, video, signals from measuring devices in the field, & Digital data.

Audio signal → range 20Hz to 20kHz.
Video signal → range from dc to 4.2MHz. Binary signal 1's & 0's.
Input transducer converts one form of signal into another. audio signal is converted to electrical signal using Mic or microphone.

Communication channel can be a pair of conductors, optical fiber or just free space.
The signal can be directly sent through a twisted pair telephone cable. But the radio link through free space cannot be used directly for audio signals; Instead an antenna of great height would be required.
A high frequency carrier signal is used to carry the message signal after performing modulation. At the receiver the information can be demodulated. The message is called modulating signal.
The output transducer converts Electrical signal back to audio signal (original message) Example is Speaker.

Principle of operation of Mobile phone

Most Commonly used device in Communication is a mobile phone. It is also called as Cellular phone.

A Cellular/mobile system provides standard telephone operation by full duplex (two way radio at remote locations.

It provides a wireless connection to the public switched telephone network (PSTN) from any user location within the radio range of the system.

The main purpose of the Cellular/mobile system is that it divides the entire geographical area into many small areas called Cells and they will be served by transmitter and receiver as shown in figure (1).

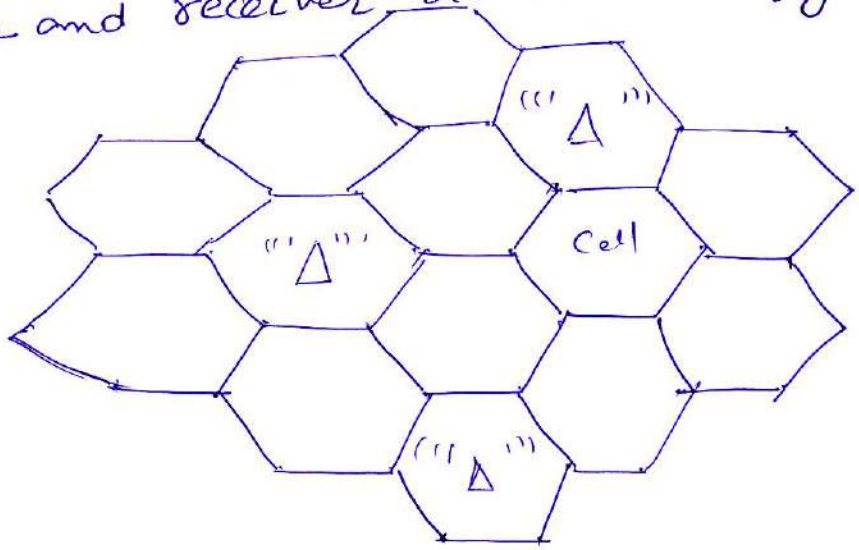


Fig (1). Cell Area in hexagon shape

A single cell covers several square kilometers Contains its own receivers and low power transmitter. The Cell shape is hexagon. If any shape such as triangle Square and Circle Considered results in overlap of areas.

Basic cellular system consists of

- mobile stations
- Base stations
- Mobile switching centre. [MSC]

The mobile switching centre is also known as mobile telephone switching office (MTSO). The MTSO controls the cells and provide the interface between each cell and main telephone office.

Here Each mobile communicates via radio with one of the base stations and may be handed off (switched from one cell to another) to any other base station throughout the duration of the call.

The mobile station commonly consists of

- A Transceiver
- An antenna
- Control unit

The base station consists of several transmitter and receiver which simultaneously handle full duplex communication and generally have towers which support several transmitting and receiving antennas.

The base station serves as a bridge between all mobile users in the cell and connects the simultaneous mobile calls via telephone lines or microwave link to the MSC.

The MSC coordinates the activities of all the base station and connects the entire cellular system to the PSTN. most of the cellular system also provides a service known as roaming.

The cellular system operates in the range 800-900MHz. newer digital cellular system operate at 1.7 - 1.8 GHz band & have greater capacity. Total of 832 channels can be used in the cell.

The block diagram shown in figure ① shows the working of mobile networks through GSM.

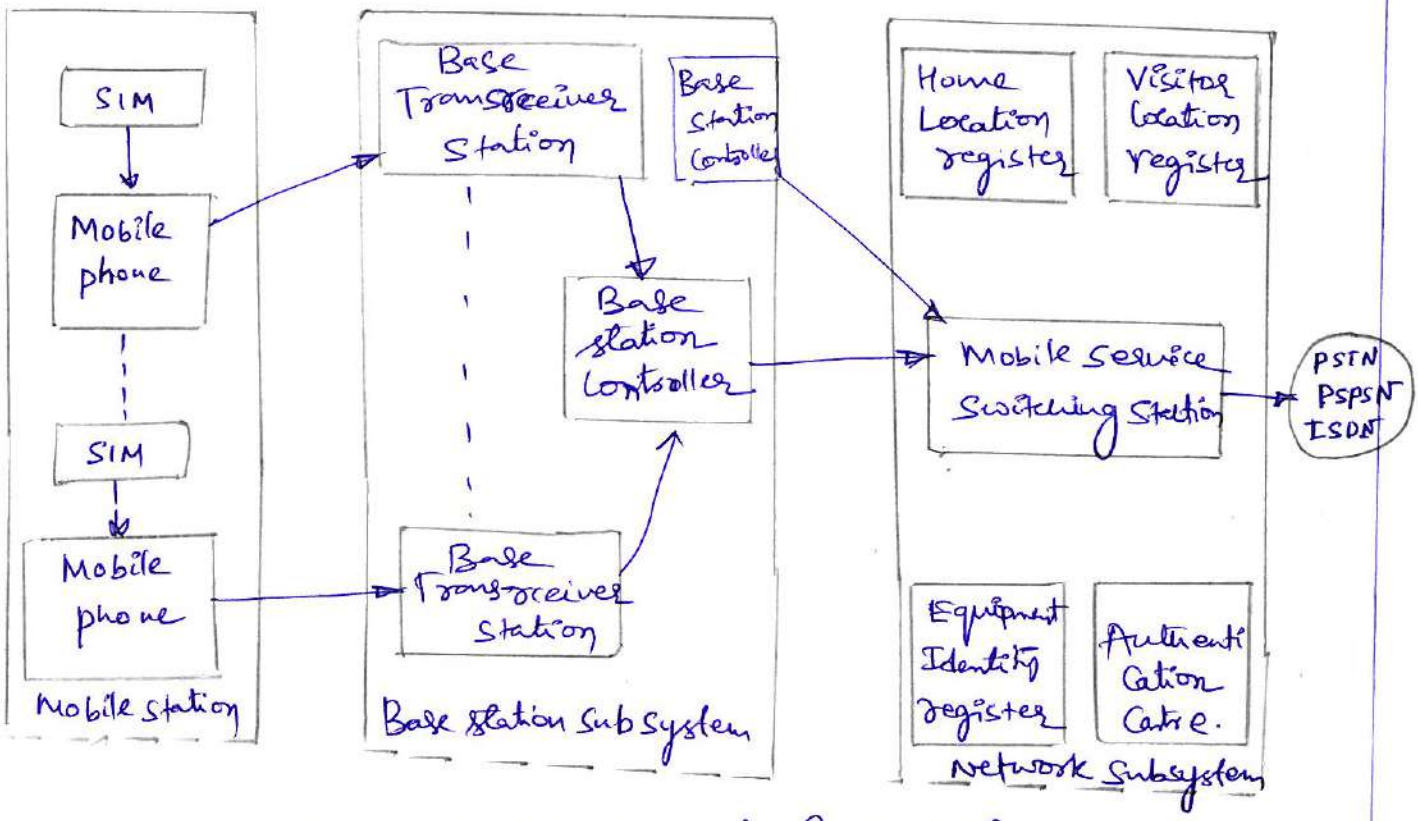


Fig ① Block diagram of GSM System.

The mobile station

It consists of mobile phone (terminal) and a smart card called Subscriber Identity module [SIM]. By inserting SIM card into Mobile phone, the user is able to receive calls at that terminal, make calls from that terminal, and receive other subscribed services. The mobile equipment is uniquely identified by the International Mobile Equipment Identity (IMEI). The SIM card contains the International mobile subscriber Identity (IMSI) used to identify the subscriber to the system, a secret key for authentication, & other information.

Mobile switching centre or Mobile service switching station is a central component of network subsystem. It acts like a normal switching node of the PSTN or ISDN. It provides all the functionality needed to handle a mobile subscriber, such as registeration, authentication, location updating, hand overs, & call routing to a roaming subscribes. The signalling system No. 7 is used for trunking signal in ISDN.

The Home Location Register (HLR) and Vistor Location Register (VLR). together with MSC, provides the call-routing and roaming capabilities of GSM.
HLR contains the administrative information of each subscriber & current location of mobile.

The equipment Identity Register (EIR) is a database that contains a list of valid mobile equipment on the network, where each mobile station is identified by its International mobile equipment Identity (IMEI).

An Authentication Centre is a protected database that stores a copy of the secret key stored in each subscriber's SIM card, which is used for authentication & encryption over the radio channel.

Base station subsystem :-

It composed of two parts, the Base Transceiver Station (BTS) and Base station Controller (BSC).

The BTS houses a radio transceivers that defines a cell and handles the radio-link protocol with the mobile station. In large urban Area more number of BTS will be deployed.

Base station Controller manages the radio resources for one or more BTS's. It handles radio-channel setup, frequency hopping, and hand overs. BSC is the connection between the mobile station and the mobile service switching centre (MSC).

Cellular Telephone unit

The block diagram of a cellular mobile radio unit. The unit consists following blocks.

1. Control unit
2. logic unit
3. Receiver
4. Frequency Synthesizer
5. Transmitter.

Control unit: It is a set of speaker, microphone with touch tone dialing facility and it stores the memory like numbers and dialing features.

Logic unit: - It is the microprocessor controlled master Control unit for cellular radio. It controls the Complete operation of Mobile Telephone switching office (MTSO) and mobile unit.

Receiver: - The cellular receiver consists of RF amplifier FM demodulator and filters. An RF amplifier boots the level of received cell site signal. Received signal is monitored by MTSO. If this is a week signal in the present cell then mobile unit is shifted to other site where the signal is strong.

Frequency Synthesizer: — It is used to generate various signals required for transmitter and receiver. It acts as a signal generator. When a mobile unit initates a call, MTSO identifies the user and assigns a frequency channel which is not used by any other mobile in the cell. MTSO sends a unique code for setting channel Frequencies.

Transmitter:- A low power FM transmitter operating at a frequency range of 825 to 845 MHz. There is a 66630 KHz transmit channel.

The Transmitter produces a deviation of ±12KHz. The modulated output is translated up to final transmitter frequency with the help of mixer, the second input to the transmitter comes from frequency synthesizer. The unique feature of high power translator is that it is controllable by cell site and MTSO. [Mobile Telephone Switching office].

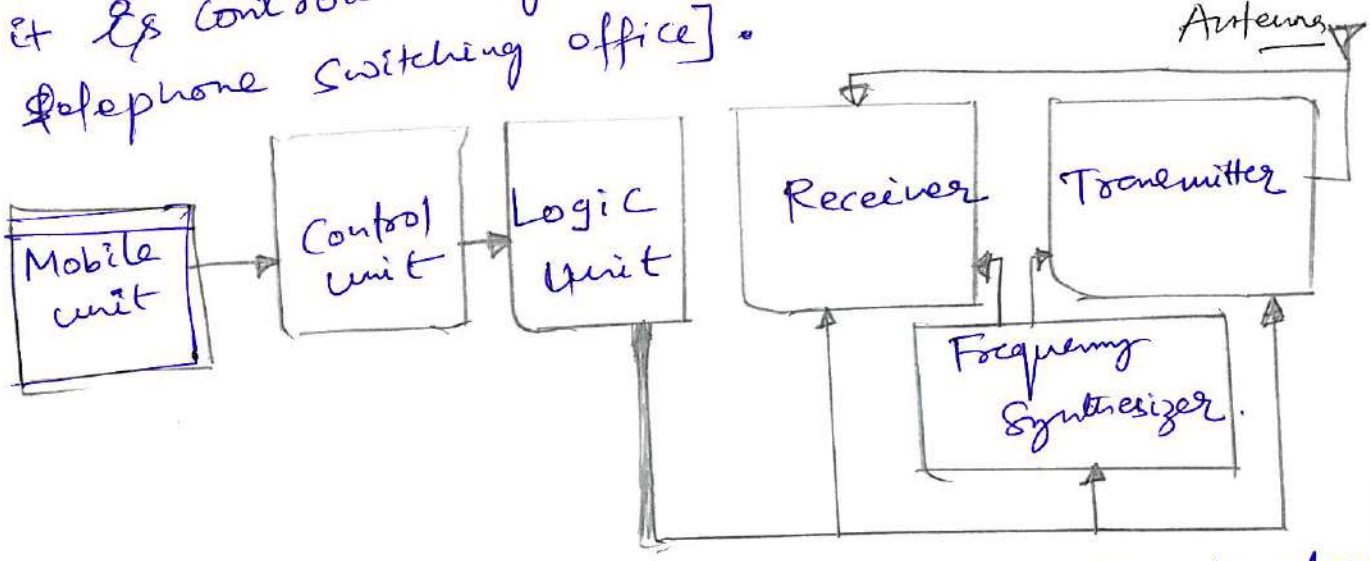


Fig (3): Block diagram of cellular mobile radio network