LECTURE NOTES

ON

DIGITAL AND PULSE CIRCUITS

2018-2019

B.TECH IV Semester(IARE-R16)

Dr.V.Vijay vallabhuni
Mrs.V.Bindusree

ELECTRICAL AND ELECTRONICS ENGINEERING

INSTITUTE OF AERONAUTICAL ENGINEERING
(AUTONOMOUS)
DUNDIGAL, HYDERABAD – 500043

UNIT 1

BOOLEAN ALGEBRA AND SWITCHING FUNCTIONS

- Philosophy of number systems
- Complement representation of negative numbers
- Binary arithmetic
- Binary codes
- Error detecting & error correcting codes Hamming codes _

HISTORY OF THE NUMERAL SYSTEMS:

A numeral system (or system of numeration) is a linguistic system and mathematical notation for representing numbers of a given set by symbols in a consistent manner. For example, It allows the numeral "11" to be interpreted as the binary numeral for *three*, the decimal numeral for *eleven*, or other numbers in different bases. Ideally, a numeral system will:

- Represent a useful set of numbers (e.g. all whole numbers, integers, or real numbers)
- Give every number represented a unique representation (or at least a standard representation) – Reflect the algebraic and arithmetic structure of the numbers.

For example, the usual decimal representation of whole numbers gives every whole number a unique representation as a finite sequence of digits, with the operations of arithmetic (addition, subtraction, multiplication and division) being present as the standard algorithms of arithmetic. However, when decimal representation is used for the rational or real numbers, the representation is no longer unique: many rational numbers have two numerals, a standard one that terminates, such as 2.31, and another that recurs, such as 2.309999999... . Numerals which terminate have no non-zero digits after a given position. For example,numerals like 2.31 and 2.310 are taken to be the same, except in the experimental sciences, where greater precision is denoted by the trailing zero.

The most commonly used system of numerals is known as Hindu-Arabic numerals.Great Indian mathematicians Aryabhatta of Kusumapura (5th Century) developed the place value notation.

Brahmagupta (6th Century) introduced the symbol zero.

BINARY

The ancient Indian writer Pingala developed advanced mathematical concepts for describing prosody, and in doing so presented the first known description of a binary numeral system.A full set of 8 trigrams and 64 hexagrams, analogous to the 3-bit and 6-bit binary numerals, were known to the ancient Chinese in the classic text *I Ching*. An arrangement of the hexagrams of the *I Ching*, ordered according to the values of the corresponding binary numbers (from 0 to 63), and a method for generating thesame, was developed by the Chinese scholar and philosopher Shao Yong in the 11th century

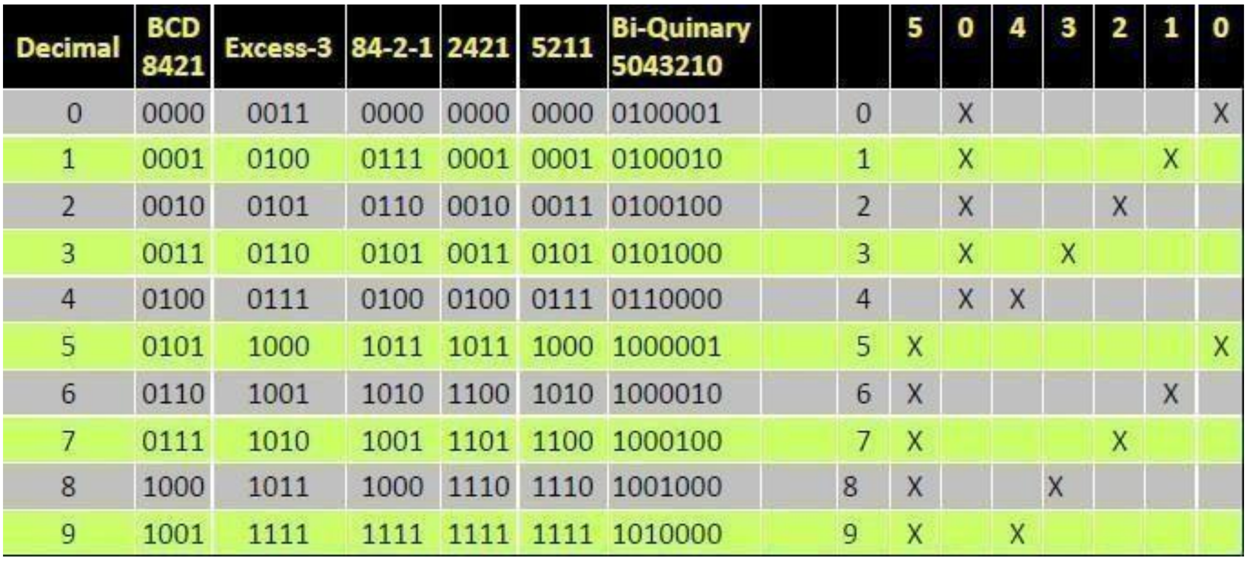
In 1854, British mathematician George Boole published a landmark paper detailing an algebraic system of logic that would become known as Boolean algebra. His logical calculus was to become instrumental in the design of digital electronic circuitry. In 1937, Claude Shannon produced his master's thesis at MIT that implemented Boolean algebra and binary arithmetic using electronic relays and switches for the first time in history. Entitled *A SymbolicAnalysis of Relay and Switching Circuits*, Shannon's thesis essentially founded practical digital circuit design.

Binary codes

Binary codes are codes which are represented in binary system with modification from the original ones.

- Weighted Binary codes
- Non Weighted Codes

Weighted binary codes are those which obey the positional weighting principles, each position of the number represents a specific weight. The binary counting sequence is an example.



Decimal	BCD 8421	Excess-3	84-2-1	2421	5211	Bi-Quinary 5043210			5	0	4	3	2	1	0
0	0000	0011	0000	0000	0000	0100001		0		X					X
1	0001	0100	0111	0001	0001	0100010		1		X				X	
2	0010	0101	0110	0010	0011	0100100		2		X			X		
3	0011	0110	0101	0011	0101	0101000		3		X		X			
4	0100	0111	0100	0100	0111	0110000		4		X	X				
5	0101	1000	1011	1011	1000	1000001		5	X						X
6	0110	1001	1010	1100	1010	1000010		6	X					X	
7	0111	1010	1001	1101	1100	1000100		7	X				X		
8	1000	1011	1000	1110	1110	1001000		8	X			X			
9	1001	1111	1111	1111	1111	1010000		9	X		X				

Reflective Code

A code is said to be reflective when code for 9 is complement for the code for 0, and so is for 8 and 1 codes, 7 and 2, 6 and 3, 5 and 4. Codes 2421, 5211, and excess-3 are reflective, whereas the 8421 code is not.

Sequential Codes

A code is said to be sequential when two subsequent codes, seen as numbers in binary

representation, differ by one. This greatly aids mathematical manipulation of data. The 8421 and Excess-3 codes are sequential, whereas the 2421 and 5211 codes are not.

Non weighted codes

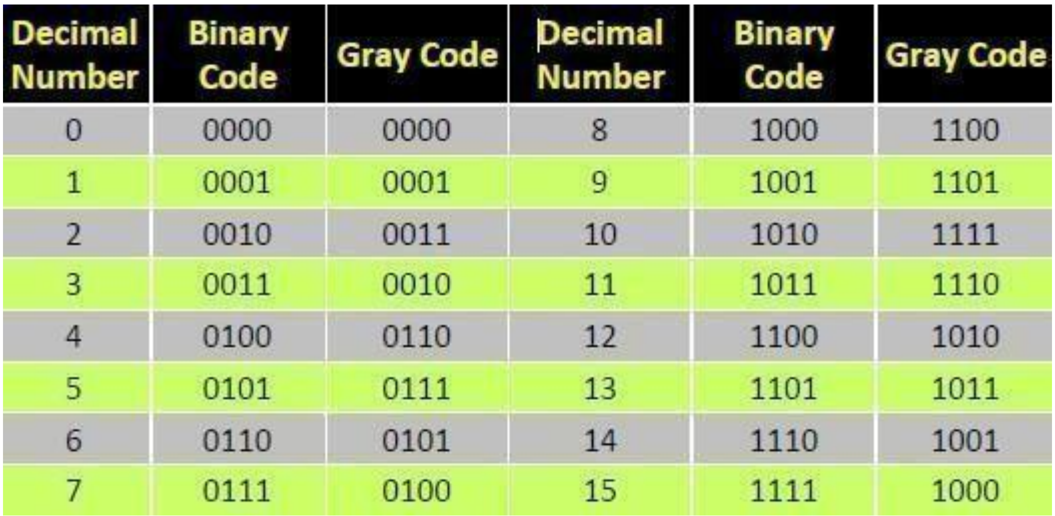
Non weighted codes are codes that are not positionally weighted. That is, each position within the binary number is not assigned a fixed value. Ex: Excess-3 code

Excess-3 Code

Excess-3 is a non weighted code used to express decimal numbers. The code derives its name from the fact that each binary code is the corresponding 8421 code plus 0011(3).

Gray Code

The gray code belongs to a class of codes called minimum change codes, in which only one bit in the code changes when moving from one code to the next. The Gray code is non-weighted code, as the position of bit does not contain any weight. The gray code is a reflective digital code which has the special property that any two subsequent numbers codes differ by only one bit. This is also called a unitdistance code. In digital Gray code has got a special place.



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

Binary to Gray Conversion

- Gray Code MSB is binary code MSB.
- Gray Code MSB-1 is the XOR of binary code MSB and MSB-1.
- MSB-2 bit of gray code is XOR of MSB-1 and MSB-2 bit of binary code.
- MSB-N bit of gray code is XOR of MSB-N-1 and MSB-N bit of binary code.Error detection codes

1)Parity bits

A parity bit is a bit that is added to a group of source bits to ensure that the number of set bits (i.e., bits with value 1) in the outcome is even or odd. It is a very simple scheme that can be used to detect single or any other odd number (i.e., three, five, etc.) of errors in the output. An even number of flipped bits will make the parity bit appear correct even though the data is erroneous.

2)Checksums

A checksum of a message is a modular arithmetic sum of message code words of a fixed word length (e.g.,byte values). The sum may be negated by means of a one's-complement prior to transmission to detect errorsresulting in all-zero messages.Checksum schemes include parity bits, check digits, and longitudinal redundancy checks. Some checksum schemes, such as the Luhn algorithm and the Verhoeff algorithm, are specifically designed to detect errorscommonly introduced by humans in writing down or remembering identification numbers.

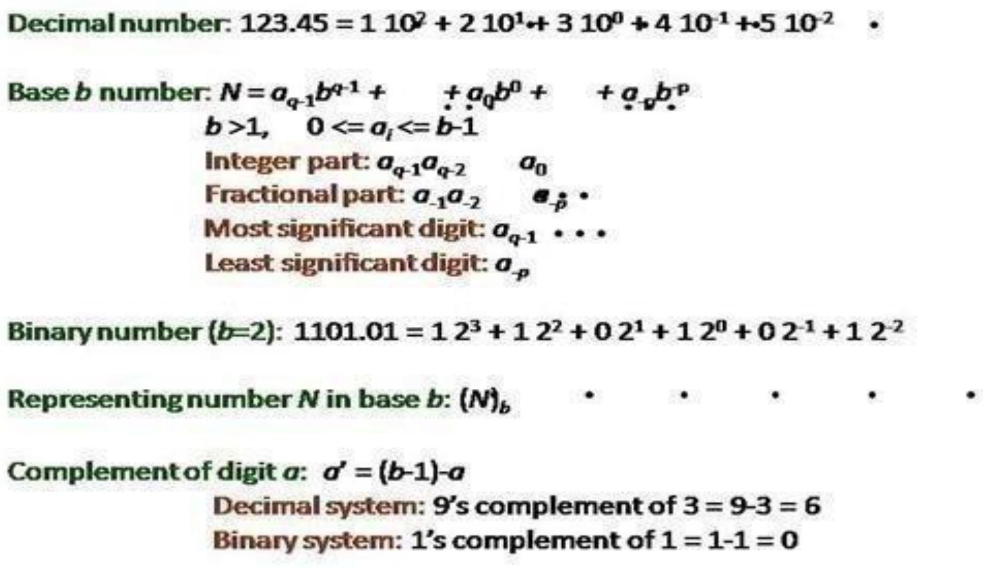
2. Cyclic redundancy checks (CRCs)

A cyclic redundancy check (CRC) is a single-burst-error-detecting cyclic code and non-secure hash function designed to detect accidental changes to digital data in computer networks. It is characterized by specification of a so-called *generator polynomial*, which is used as the divisor in a polynomial long division over a finite field, taking the input data as the dividend, and where the remainder becomes the result.Cyclic codes have favorable properties in that they are well suited for detecting burst errors. CRCs are particularly easy to implement in hardware, and are therefore commonly used in digital networks and storage devices such as hard disk drives.Even parity is a special case of a cyclic redundancy check, where the single-bit CRC is generated by the divisor *x*+1.

NUMBER BASE CONVERSIONS

Any number in one base system can be converted into another base system Types

1) decimal to any base
2) Any base to decimal
3) Any base to Any base



Decimal number: $123.45 = 1\ 10^2 + 2\ 10^1 + 3\ 10^0 + 4\ 10^{-1} + 5\ 10^{-2}$

Base *b* number: $N = a_{q-1}b^{q-1} + \quad + a_0b^0 + \quad + a_{-p}b^{-p}$

$b > 1, \quad 0 <= a_i <= b-1$

Integer part: $a_{q-1}a_{q-2} \quad a_0$

Fractional part: $a_{-1}a_{-2} \quad a_{-p}$

Most significant digit: a_{q-1}

Least significant digit: a_{-p}

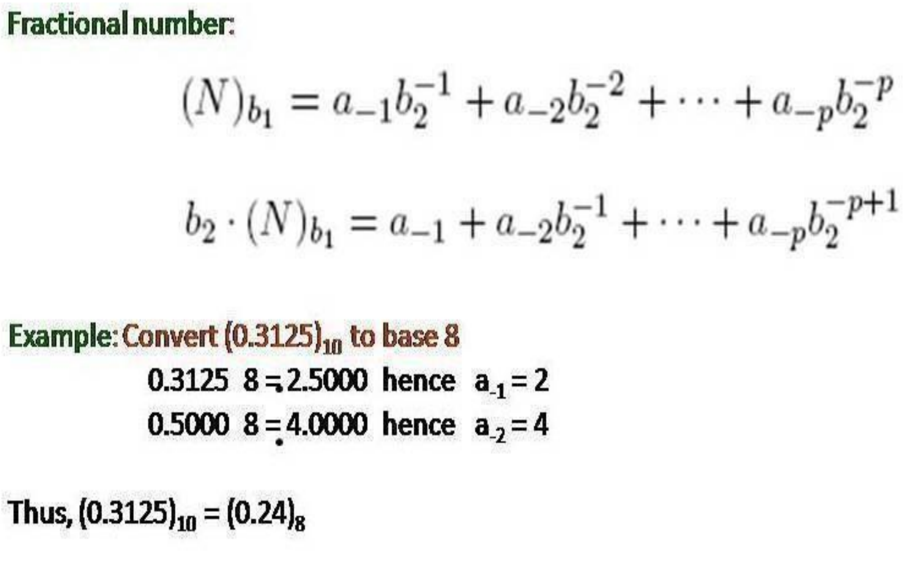
Binary number (b=2): $1101.01 = 1\ 2^3 + 1\ 2^2 + 0\ 2^1 + 1\ 2^0 + 0\ 2^{-1} + 1\ 2^{-2}$

Representing number N in base b: $(N)_b$

Complement of digit a: $a' = (b-1)-a$

Decimal system: 9's complement of 3 = 9-3 = 6

Binary system: 1's complement of 1 = 1-1 = 0



Fractional number:

$$(N)_{b_1} = a_{-1}b_2^{-1} + a_{-2}b_2^{-2} + \cdots + a_{-p}b_2^{-p}$$

$$b_2 \cdot (N)_{b_1} = a_{-1} + a_{-2}b_2^{-1} + \cdots + a_{-p}b_2^{-p+1}$$

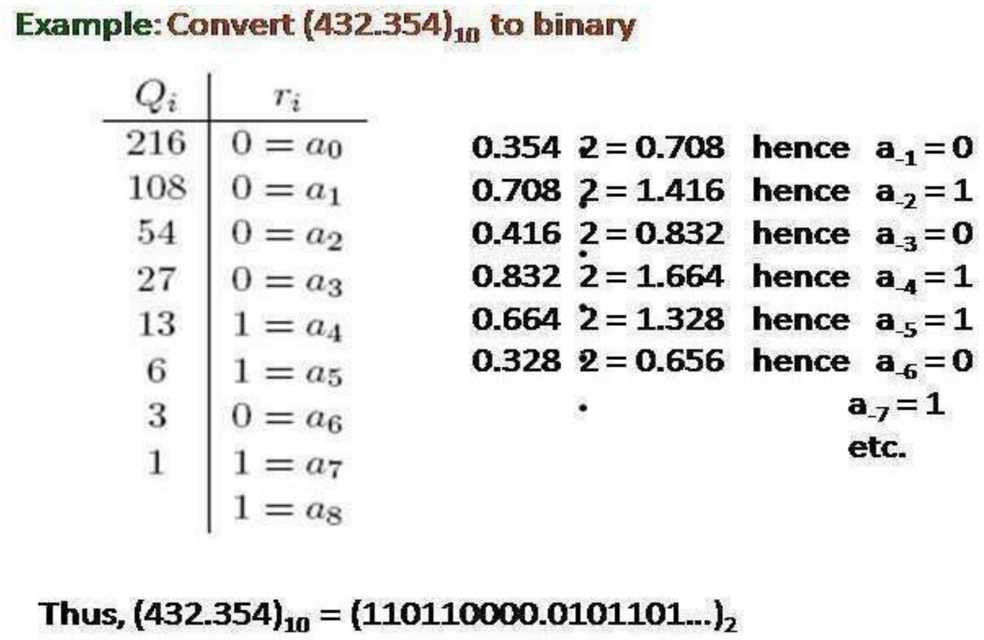
Example: Convert $(0.3125)_{10}$ to base 8

0.3125 8 = 2.5000 hence $a_{-1} = 2$

0.5000 8 = 4.0000 hence $a_{-2} = 4$

Thus, $(0.3125)_{10} = (0.24)_8$

Decimal to Binary



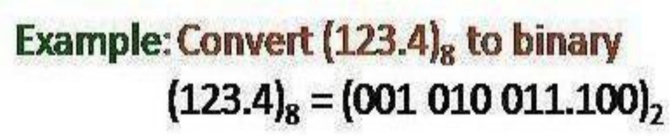
Example: Convert $(432.354)_{10}$ to binary

Q_i	r_i
216	$0 = a_0$
108	$0 = a_1$
54	$0 = a_2$
27	$0 = a_3$
13	$1 = a_4$
6	$1 = a_5$
3	$0 = a_6$
1	$1 = a_7$
	$1 = a_8$

0.354 2 = 0.708 hence $a_{-1} = 0$
0.708 2 = 1.416 hence $a_{-2} = 1$
0.416 2 = 0.832 hence $a_{-3} = 0$
0.832 2 = 1.664 hence $a_{-4} = 1$
0.664 2 = 1.328 hence $a_{-5} = 1$
0.328 2 = 0.656 hence $a_{-6} = 0$
• $a_{-7} = 1$
etc.

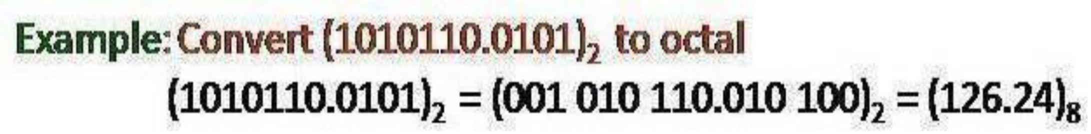
Thus, $(432.354)_{10} = (110110000.0101101\ldots)_2$

Octal To Binary



Example: Convert $(123.4)_8$ to binary

$(123.4)_8 = (001\ 010\ 011.100)_2$



Example: Convert $(1010110.0101)_2$ to octal

$(1010110.0101)_2 = (001\ 010\ 110.010\ 100)_2 = (126.24)_8$

Error Detection and Correction Codes

- No communication channel or storage device is completely error-free
- As the number of bits per area or the transmission rate increases, more errors occur. • Impossible to detect or correct 100% of the errors

Hamming Codes

1. One of the most effective codes for error-recovery
2. Used in situations where random errors are likely tooccur

Error detection and correction increases in proportion to the number of parity bits (error- checking bits) added to the end of the informationbits

code word = information bits + parity bits

Hamming distance: the number of bit positions in which two code words differ.

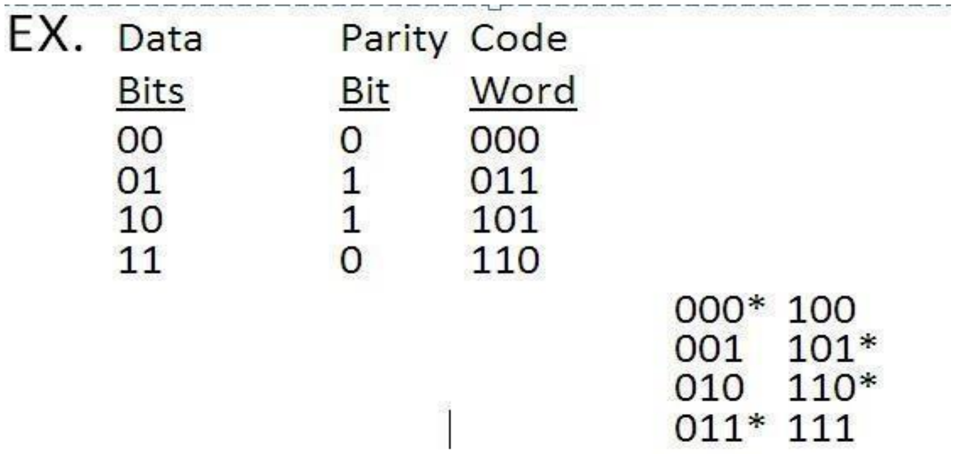
10001001

10110001

* * *

Minimum Hamming distance or D(min) : determines its error detecting and correcting capability.

1.Hamming codes can always detect D(min) – 1 errors, but can only correct half of those errors.



EX. Data Bits	Parity Bit	Code Word
00	0	000
01	1	011
10	1	101
11	0	110

000*	100
001	101*
010	110*
011*	111

1. Single parity bit can only detect error, not correct it

2. Error-correcting codes require more than a single parity bit EX. 0 0 0 0 0

0 1 0 1 1

1 0 1 1 0

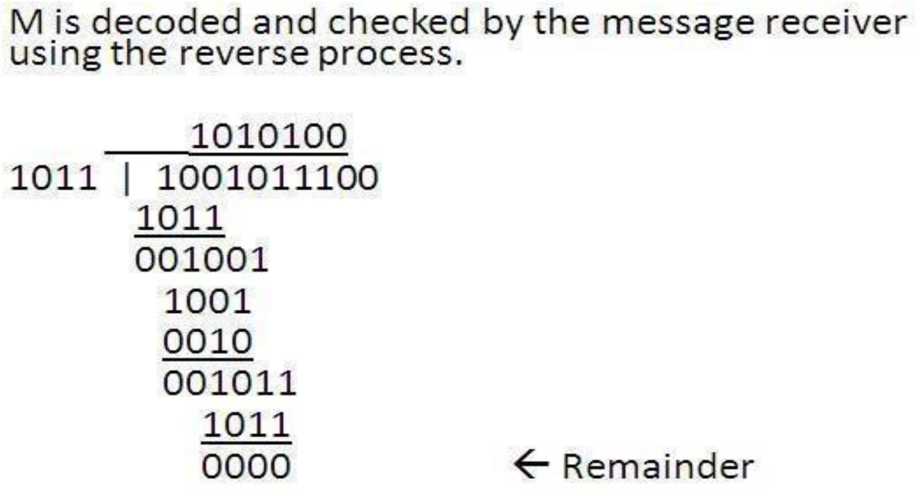
1 1 1 0 1

Minimum Hamming distance = 3

Can detect up to 2 errors and correct 1 error
Cyclic Redundancy Check

1. Let the information byte F = 1001011
2. The sender and receiver agree on an arbitrary binary pattern P. Let P = 1011.
3. Shift F to the left by 1 less than the number of bits in P. Now, F = 1001011000.
4. Let F be the dividend and P be the divisor. Perform –modulo 2 division‖.
5. After performing the division, we ignore the quotient. We got 100 for the remainder, which becomes the actual CRC checksum.

5.Add the remainder to F, giving the message M: 1001011 + 100 = 1001011100 = M



M is decoded and checked by the message receiver using the reverse process.

```
        1010100
1011 | 1001011100
      1011
      001001
       1001
       0010
       001011
         1011
         0000             ← Remainder
```

UNIT-II

MINIMIZATION TECHNIQUES AND DESIGN OF MSI

- Fundamental postulates of Boolean algebra
- Basic theorems and properties
- Switching functions
- Canonical and Standard forms
- Algebraic simplification digital logic gates, properties of XOR gates
- Universal gates
- Multilevel NAND/NOR realizations

Boolean Algebra: Boolean algebra, like any other deductive mathematical system, may be defined with aset of elements, a set of operators, and a number of unproved axioms or postulates. A *set* of elements is anycollection of objects having a common property. If S is a set and *x* and *y* are certain objects, then x Î Sdenotes that *x* is a member of the set S, and *y* ÏS denotes that *y* is not an element of S. A set with adenumerable number of elements is specified by braces: A = {1,2,3,4}, *i.e.* the elements of set A are thenumbers 1, 2, 3, and 4. A *binary operator* defined on a set S of elements is a rule that assigns to each pair ofelements from S a unique element from S._ Example: In *a***b*=*c*, we say that * is a binary operator if it specifies a rule for finding *c* from the pair (*a*,*b*)and also if *a*, *b*, *c* Î S.

CLOSURE: The Boolean system is *closed* with respect to a binary operator if for every pair of Boolean values,it produces a Boolean result. For example, logical AND is closed in the Boolean system because it accepts only Boolean operands and produces only Boolean results.

_ A set *S* is closed with respect to a binary operator if, for every pair of elements of *S*, the binary operator specifies a rule for obtaining a unique element of *S*.

_ For example, the set of natural numbers N = {1, 2, 3, 4, … 9} is closed with respect to the binary operator plus (+) by the rule of arithmetic addition, since for any *a*, *b* Î N we obtain a unique *c* Î N by the operation *a* + *b* = c.

ASSOCIATIVE LAW:

A binary operator * on a set *S* is said to be associative whenever (*x* * *y*) * *z* = *x* * (*y* * *z*) for all *x*, *y*, *z* Î S, forall Boolean values x, y and z.

COMMUTATIVE LAW:A binary operator * on a set *S* is said to be commutative whenever x * *y* = *y* * *x* for all *x*, *y*, *z* є S

IDENTITY ELEMENT:A set *S* is said to have an identity element with respect to a binary operation * on S if there exists an element *e* є S with the property *e* * *x* = *x* * *e* = *x* for every *x* є S

BASIC IDENTITIES OF BOOLEAN ALGEBRA

- Postulate 1(Definition): A Boolean algebra is a closed algebraic system containing a set K of two or more elements and the two operators · and + which refer to logical AND and logical OR •$x + 0 = x$
- $x \cdot 0 = 0$
- $x + 1 = 1$
- $x \cdot 1 = 1$
- $x + x = x$
- $x \cdot x = x$
- $x + x' = x$
- $x \cdot x' = 0$
- $x + y = y + x$
- $xy = yx$
- $x + (y + z) = (x + y) + z$
- $x (yz) = (xy) z$
- $x (y + z) = xy + xz$
- $x + yz = (x + y)(x + z)$
- $(x + y)' = x' y'$
- $(xy)' = x' + y'$
- $(x')' = x$

DeMorgan's Theorem

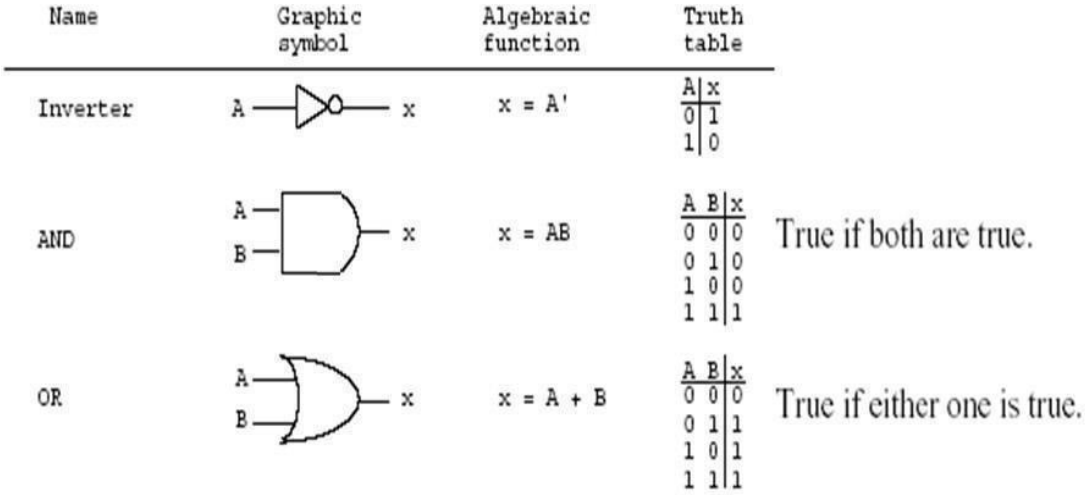
(a) $(a + b)' = a'b'$
(b) $(ab)' = a' + b'$

Generalized DeMorgan's Theorem (a) $(a + b + \ldots z)' = a'b' \ldots z'$ (b) $(a.b \ldots z)' = a' + b' + \ldots z$

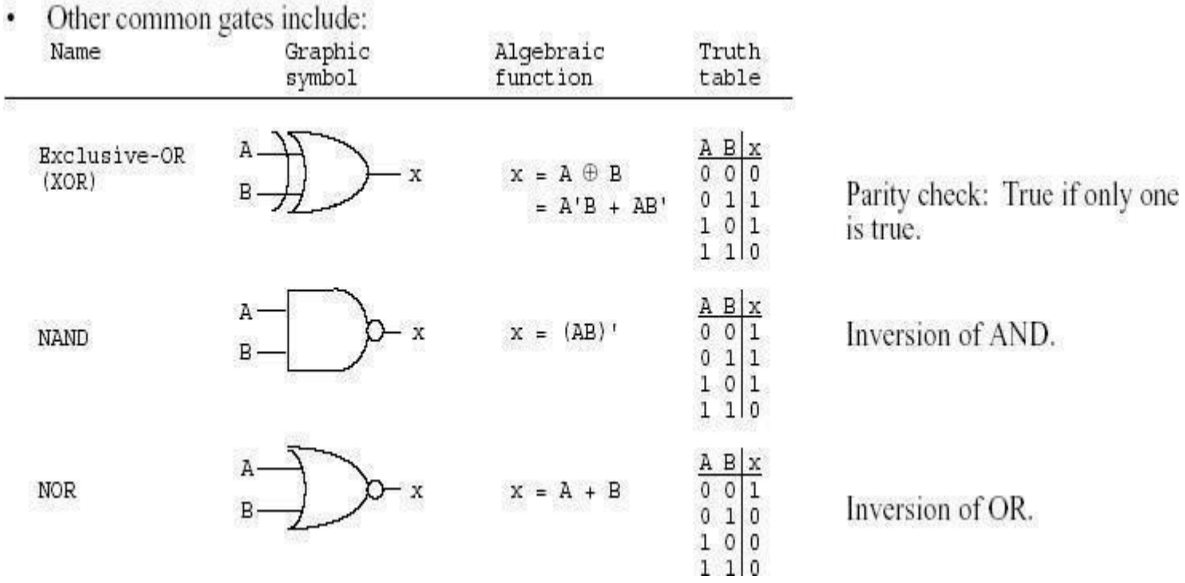
LOGIC GATES

Formal logic: In formal logic, a statement (proposition) is a declarative sentence that is either true(1) or false (0). It is easier to communicate with computers using formal logic.

- Boolean variable: Takes only two values – either true (1) or false (0). They are used as basic units of formal logic.
- Boolean algebra: Deals with binary variables and logic operations operating on those variables.
- Logic diagram: Composed of graphic symbols for logic gates. A simple circuit sketch that represents inputs and outputs of Boolean functions



Name	Graphic symbol	Algebraic function	Truth table	
Inverter	A x	x = A'	A|x 0|1 1|0	
AND	A B x	x = AB	A B|x 0 0|0 0 1|0 1 0|0 1 1|1	True if both are true.
OR	A B x	x = A + B	A B|x 0 0|0 0 1|1 1 0|1 1 1|1	True if either one is true.



- Other common gates include:

Name	Graphic symbol	Algebraic function	Truth table	
Exclusive-OR (XOR)	A B x	x = A ⊕ B = A'B + AB'	A B|x 0 0|0 0 1|1 1 0|1 1 1|0	Parity check: True if only one is true.
NAND	A B x	x = (AB)'	A B|x 0 0|1 0 1|1 1 0|1 1 1|0	Inversion of AND.
NOR	A B x	x = A + B	A B|x 0 0|1 0 1|0 1 0|0 1 1|0	Inversion of OR.

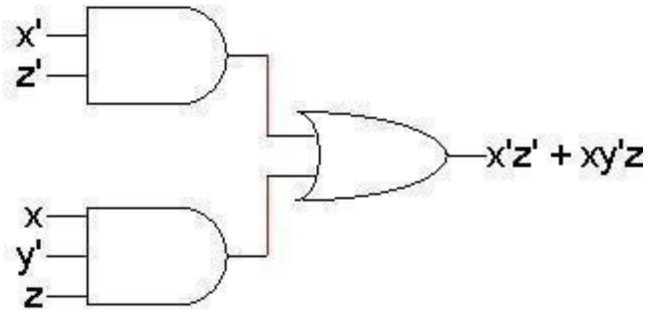
Minimization of switching functions is to obtain logic circuits with least circuit complexity. This goal is very difficult since how a minimal function relates to the implementation technology is important. For example, If we are building a logic circuit that uses discrete logic made of small scale Integration ICs(SSIs) like 7400 series, in which basic building block are constructed and are available for use. The goal of minimization would be to reduce the number of ICs and not the logic gates. For example, If we require two 6 and gates and 5 Or gates,we would require 2 AND ICs(each has 4 AND gates) and one OR IC. (4 gates). On the other hand if the same logic could be implemented with only 10 nand gates, we require only 3 ICs. Similarly when we design logic on Programmable device, we may implement the design with certain number of gates and remaining gates may not be used.

Whatever may be the criteria of minimization we would be guided by the following:

- Boolean algebra helps us simplify expressions and circuits
- Karnaugh Map: A graphical technique for simplifying a Boolean expression into eitherform:
  - minimal sum of products (MSP)

o minimal product of sums (MPS)

- Goal of thesimplification.
- There are a minimal number of product/sum terms oEach term has a minimal number of literals
- Circuit-wise, this leads to a *minimal* two-levelimplementation



A Two-Variable Karnaugh Map

- A two-variable function has four possible minterms. We can re-arrange these minterms into a Karnaugh map

x	y	minterm

X	Y	
0	0	x‘y‘
0	1	x‘y
1	0	xy‘
1	1	xy

→

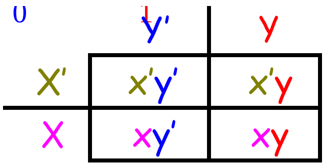
	0	1
0	x‘y‘	x‘y
X 1	xy‘	xy

Now we can easily see which minterms contain
common literals

– Minterms on the left and right sides contain y‘ and y respectively

– Minterms in the top and bottom rows contain x‘ and xrespectively XY

0	x‘y‘	x‘y
1	xy‘	xy



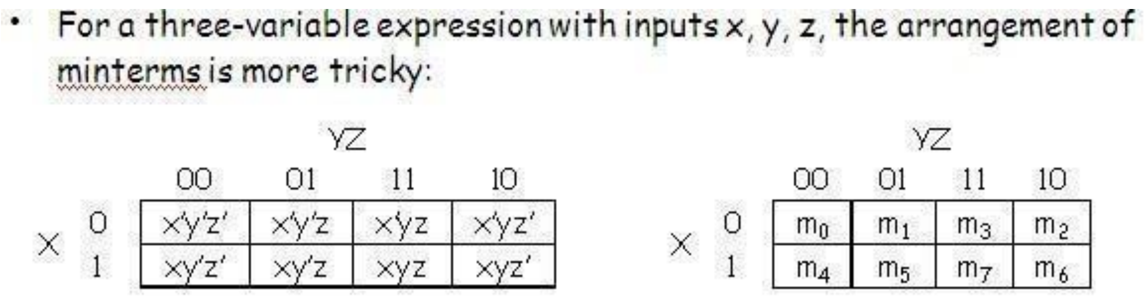
0	y'	y
X'	x'y'	x'y
X	xy'	xy

Simplification

- Imagine a two-variable sum of minterms x‘y‘ + x‘y
- Both of these minterms appear in the top row of a Karnaugh map, which means that they both contain the literal x‘

		Y
	x’y’	x’y
X	xy’	Xy

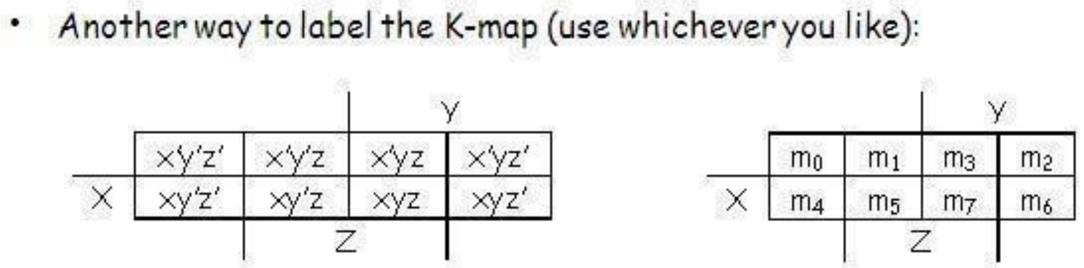
- What happens if you simplify this expression using Boolean algebra?
- x‘y‘ + x‘y= x‘(y‘ + y) [Distributive]



- For a three-variable expression with inputs x, y, z, the arrangement of minterms is more tricky:

		YZ			
		00	01	11	10
X	0	x'y'z'	x'y'z	x'yz	x'yz'
	1	xy'z'	xy'z	xyz	xyz'

		YZ			
		00	01	11	10
X	0	m_0	m_1	m_3	m_2
	1	m_4	m_5	m_7	m_6

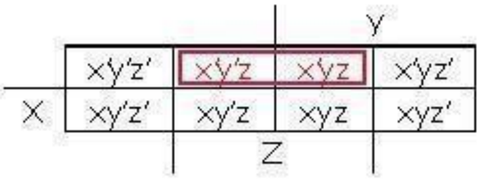


- Another way to label the K-map (use whichever you like):

			Y	
	x'y'z'	x'y'z	x'yz	x'yz'
X	xy'z'	xy'z	xyz	xyz'
		Z		

			Y	
	m_0	m_1	m_3	m_2
X	m_4	m_5	m_7	m_6
		Z		

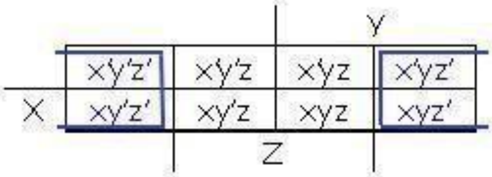
- With this ordering, any group of 2, 4 or 8 adjacent squares on the map contains common literals that can be factored out



	x'y'z'	x'y'z	x'yz	x'yz'
X	xy'z'	xy'z	xyz	xyz'

$$
\begin{aligned}
& x'y'z + x'yz \\
=\ & x'z(y' + y) \\
=\ & x'z \bullet 1 \\
=\ & x'z
\end{aligned}
$$

- "Adjacency" includes wrapping around the left and right sides:

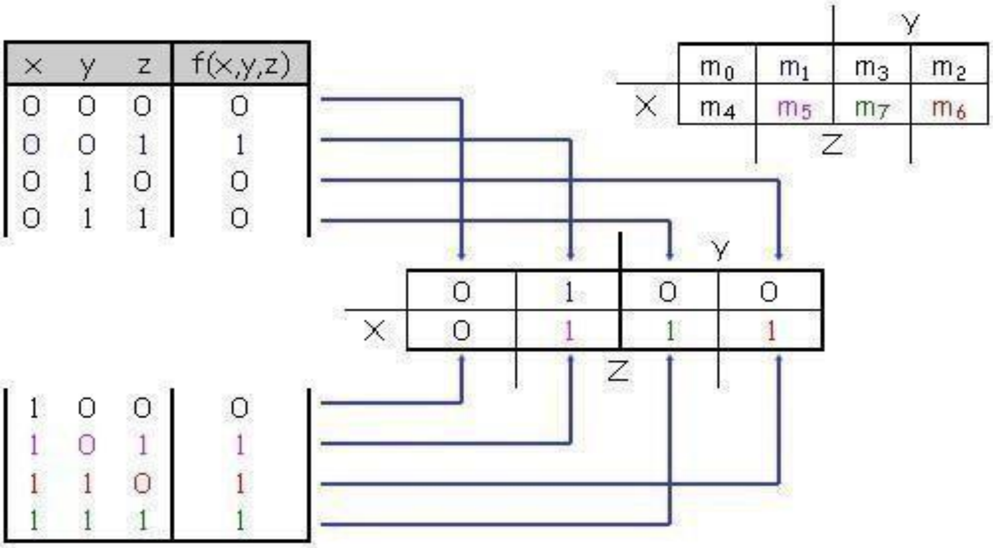


	x'y'z'	x'y'z	x'yz	x'yz'
X	xy'z'	xy'z	xyz	xyz'

$$
\begin{aligned}
& x'y'z' + xy'z' + x'yz' + xyz' \\
=\ & z'(x'y' + xy' + x'y + xy) \\
=\ & z'(y'(x' + x) + y(x' + x)) \\
=\ & z'(y'+y) \\
=\ & z'
\end{aligned}
$$

- We'll use this property of adjacent squares to do our simplifications.

- We can fill in the K-map directly from a truth table
  - The output in row *i* of the table goes into square m_i of the K-map
  - Remember that the rightmost columns of the K-map are "switched"



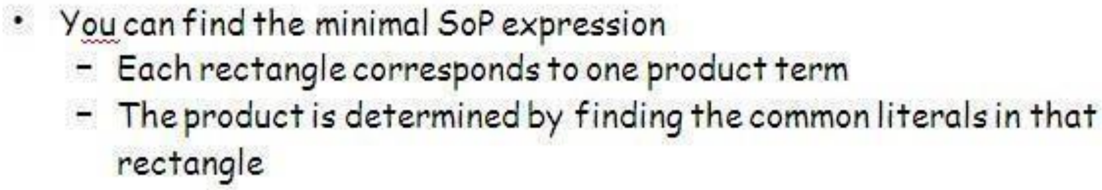
x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

	m_0	m_1	m_3	m_2
X	m_4	m_5	m_7	m_6

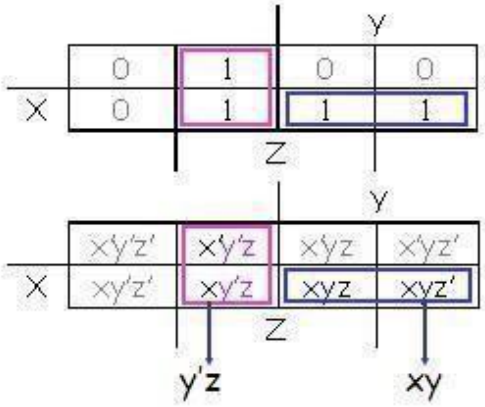
	0	1	0	0
X	0	1	1	1

K-maps From Truth Tables

Reading the MSP from the K-map



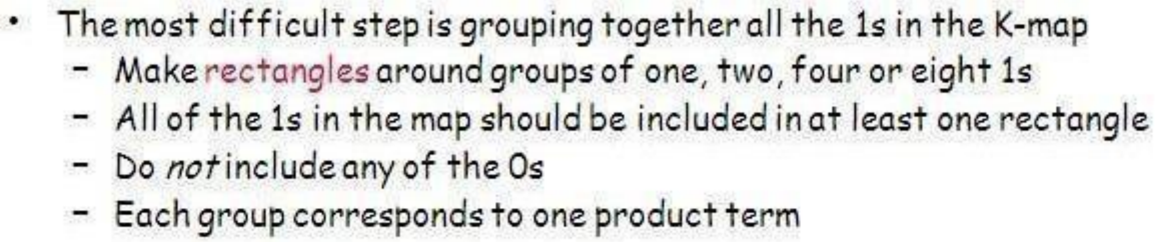
- You can find the minimal SoP expression
  - Each rectangle corresponds to one product term
  - The product is determined by finding the common literals in that rectangle



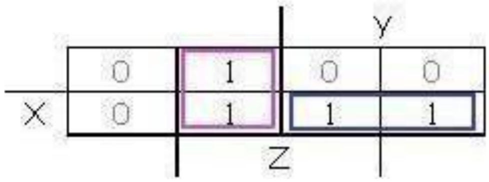
	0	1	0	0
X	0	1	1	1

	x'y'z'	x'y'z	x'yz	x'yz'
X	xy'z'	xy'z	xyz	xyz'

F(x,y,z)= y'z + xy



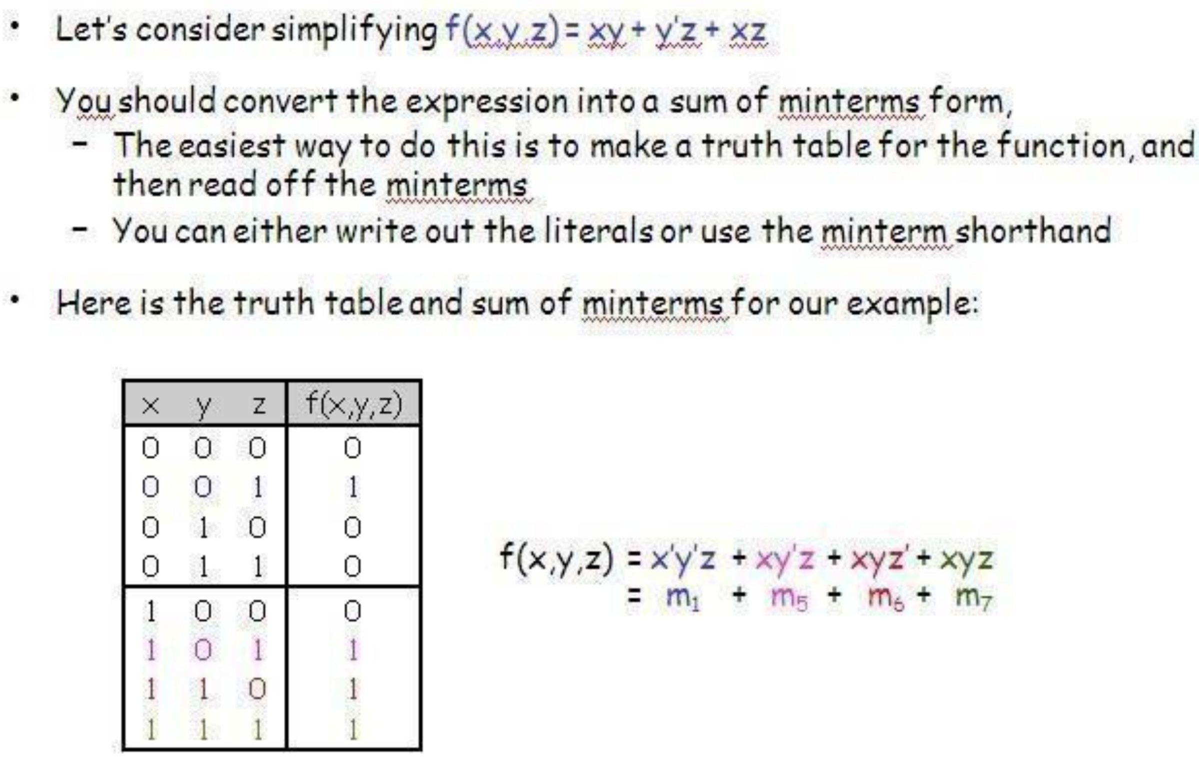
- The most difficult step is grouping together all the 1s in the K-map
  - Make rectangles around groups of one, two, four or eight 1s
  - All of the 1s in the map should be included in at least one rectangle
  - Do *not* include any of the 0s
  - Each group corresponds to one product term



	0	1	0	0
X	0	1	1	1

Simplify $m_0+m_2+m_5+m_8+m_{10}+m_{13}$

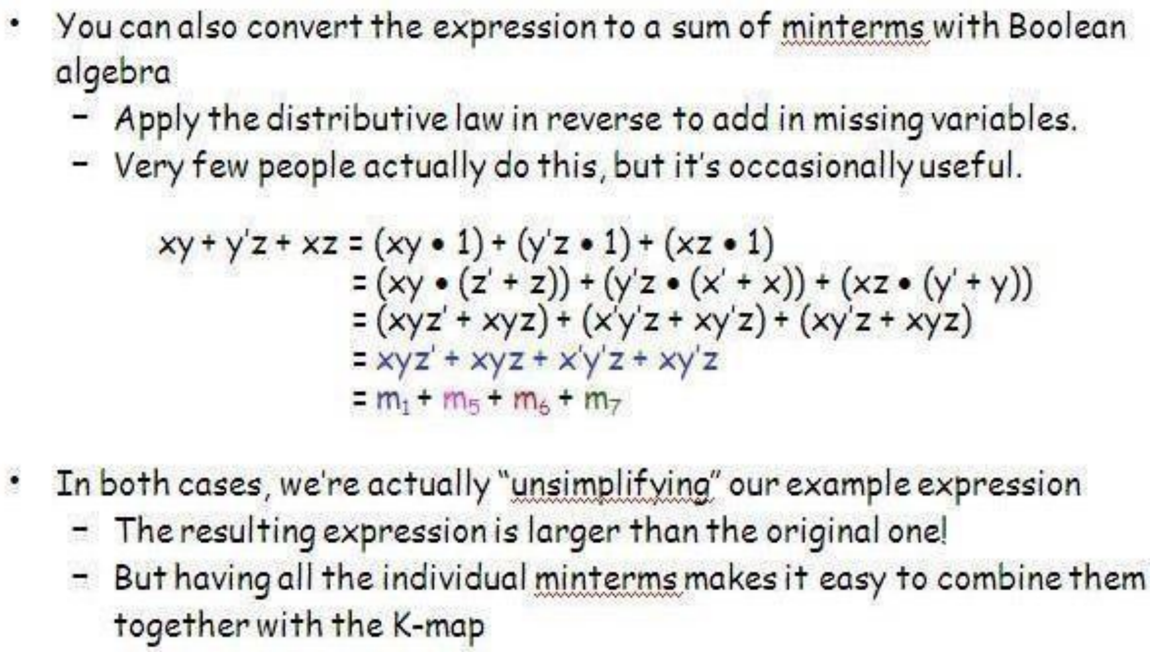
K-map Simplification of SoP Expressions



- Let's consider simplifying $f(x,y,z) = xy + y'z + xz$
- You should convert the expression into a sum of minterms form,
  - The easiest way to do this is to make a truth table for the function, and then read off the minterms
  - You can either write out the literals or use the minterm shorthand
- Here is the truth table and sum of minterms for our example:

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$$\begin{aligned} f(x,y,z) &= x'y'z + xy'z + xyz' + xyz \\ &= m_1 + m_5 + m_6 + m_7 \end{aligned}$$

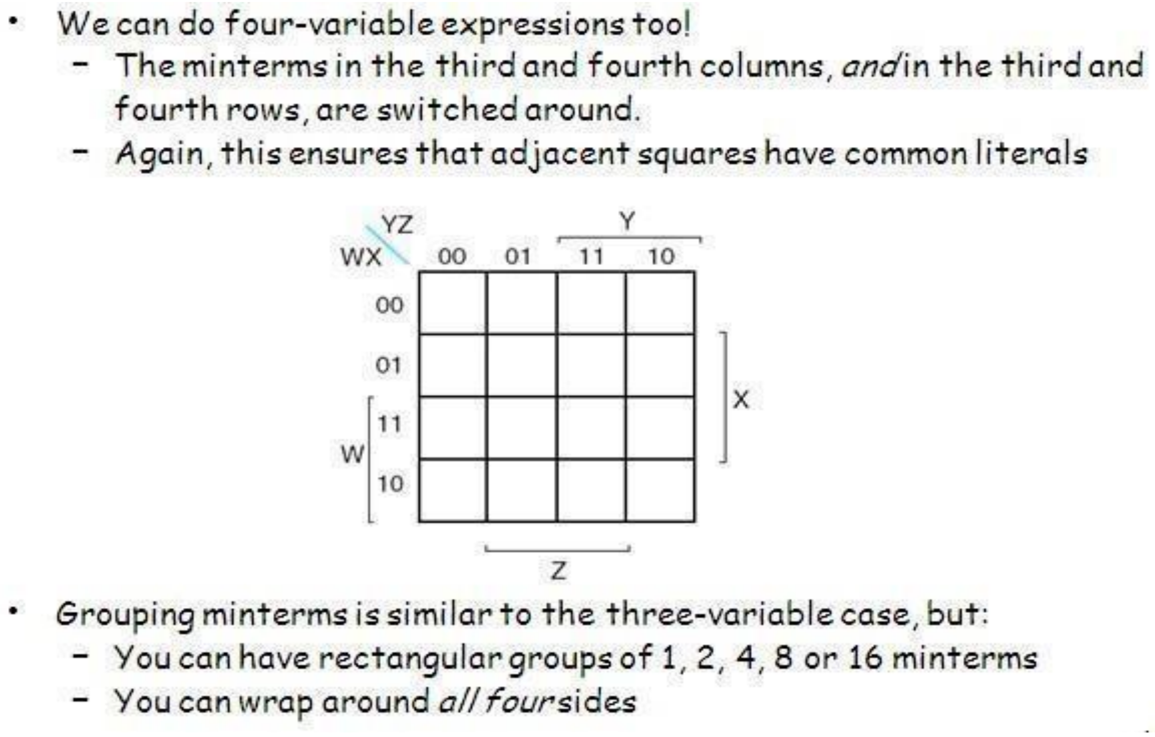


- You can also convert the expression to a sum of minterms with Boolean algebra
  - Apply the distributive law in reverse to add in missing variables.
  - Very few people actually do this, but it's occasionally useful.

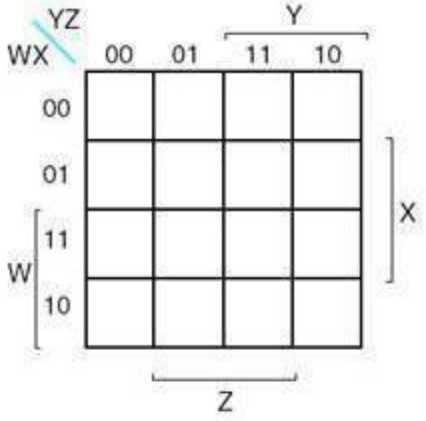
$$\begin{aligned} xy + y'z + xz &= (xy \bullet 1) + (y'z \bullet 1) + (xz \bullet 1) \\ &= (xy \bullet (z' + z)) + (y'z \bullet (x' + x)) + (xz \bullet (y' + y)) \\ &= (xyz' + xyz) + (x'y'z + xy'z) + (xy'z + xyz) \\ &= xyz' + xyz + x'y'z + xy'z \\ &= m_1 + m_5 + m_6 + m_7 \end{aligned}$$

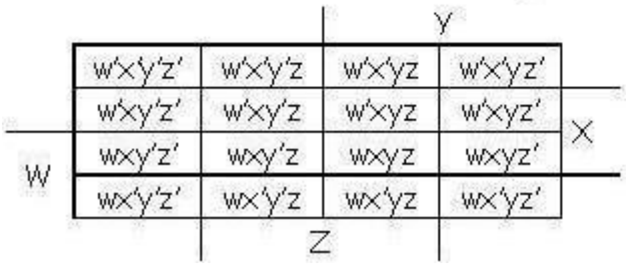
- In both cases, we're actually "unsimplifying" our example expression
  - The resulting expression is larger than the original one!
  - But having all the individual minterms makes it easy to combine them together with the K-map

Simplify $m_0+m_2+m_5+m_8+m_{10}+m_{13}$

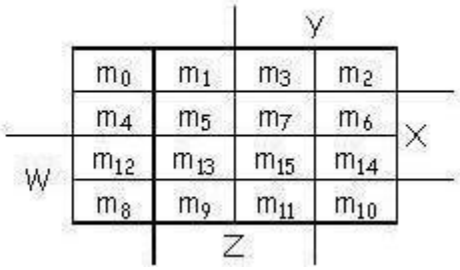


- We can do four-variable expressions too!
  - The minterms in the third and fourth columns, *and* in the third and fourth rows, are switched around.
  - Again, this ensures that adjacent squares have common literals
- Grouping minterms is similar to the three-variable case, but:
  - You can have rectangular groups of 1, 2, 4, 8 or 16 minterms
  - You can wrap around *all four* sides



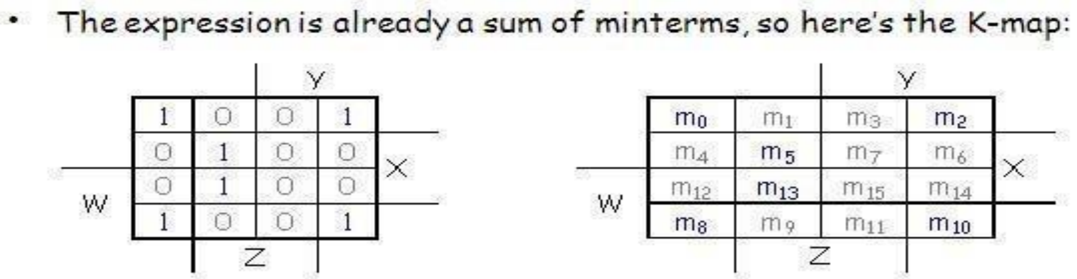


w'x'y'z'	w'x'y'z	w'x'yz	w'x'yz'
w'xy'z'	w'xy'z	w'xyz	w'xyz'
wxy'z'	wxy'z	wxyz	wxyz'
wx'y'z'	wx'y'z	wx'yz	wx'yz'



m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

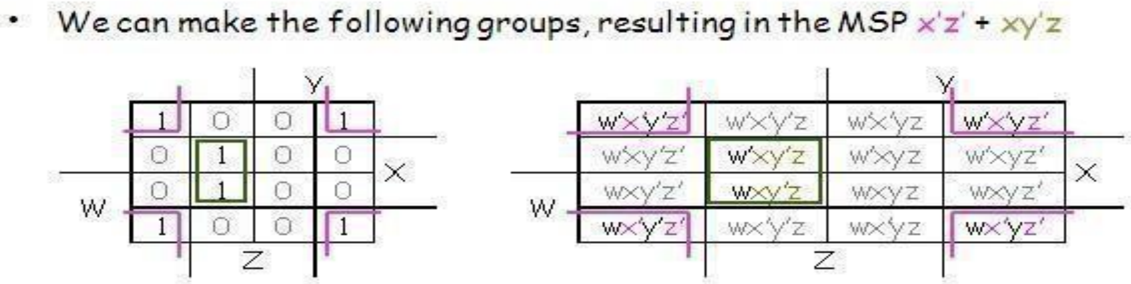
Simplify $m_0+m_2+m_5+m_8+m_{10}+m_{13}$



- The expression is already a sum of minterms, so here's the K-map:

1	0	0	1
0	1	0	0
0	1	0	0
1	0	0	1

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

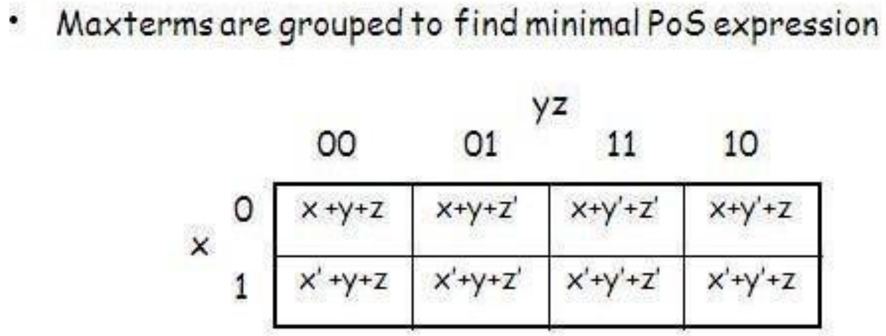


- We can make the following groups, resulting in the MSP x'z' + xy'z

1	0	0	1
0	1	0	0
0	1	0	0
1	0	0	1

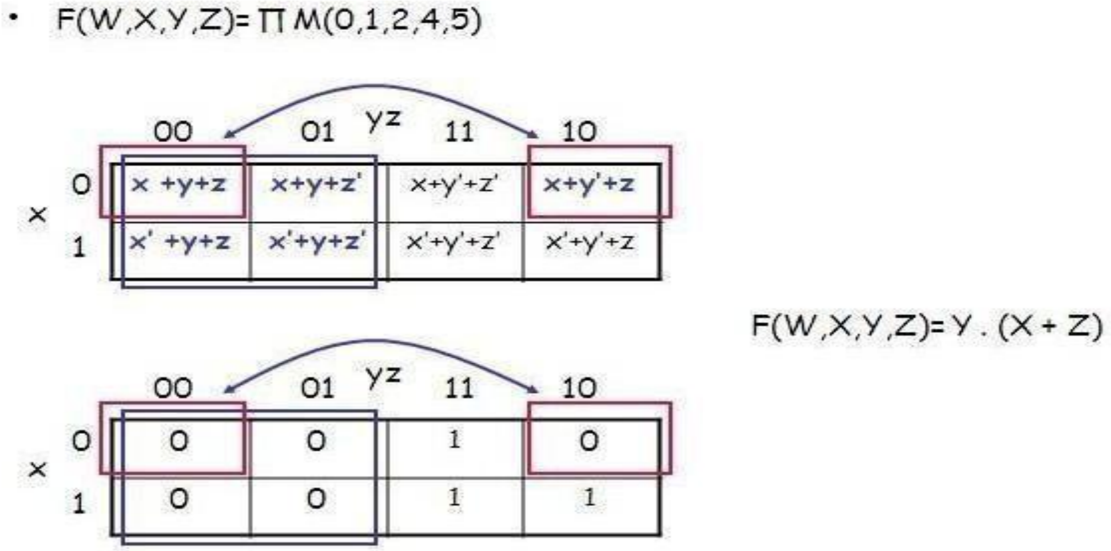
w'x'y'z'	w'x'y'z	w'x'yz	w'x'yz'
w'xy'z'	w'xy'z	w'xyz	w'xyz'
wxy'z'	wxy'z	wxyz	wxyz'
wx'y'z'	wx'y'z	wx'yz	wx'yz'

PoS Optimization



- Maxterms are grouped to find minimal PoS expression

x \ yz	00	01	11	10
0	x +y+z	x+y+z'	x+y'+z'	x+y'+z
1	x' +y+z	x'+y+z'	x'+y'+z'	x'+y'+z



- F(W,X,Y,Z)= ∏ M(0,1,2,4,5)

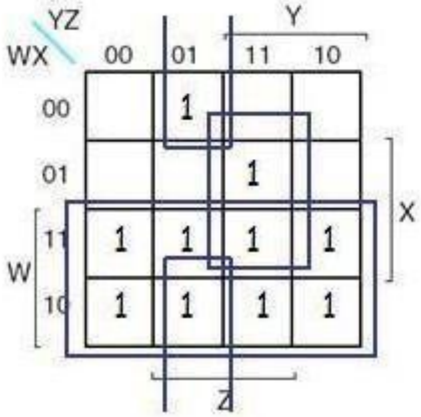
x \ yz	00	01	11	10
0	x +y+z	x+y+z'	x+y'+z'	x+y'+z
1	x' +y+z	x'+y+z'	x'+y'+z'	x'+y'+z

x \ yz	00	01	11	10
0	0	0	1	0
1	0	0	1	1

F(W,X,Y,Z)= Y . (X + Z)

SoP Optimization from PoS

$$F(W,X,Y,Z)=\prod M(0,2,3,4,5,6)$$
$$=\sum m(1,7,8,9,10,11,12,13,14,15)$$

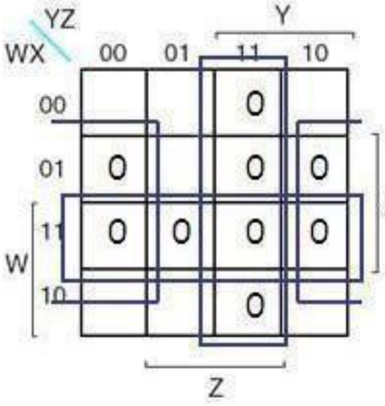


WX \ YZ	00	01	11	10
00		1		
01			1	
11	1	1	1	1
10	1	1	1	1

$$F(W,X,Y,Z)=W+XYZ+X'Y'Z$$

PoS Optimization from SoP

$$F(W,X,Y,Z)=\sum m(0,1,2,5,8,9,10)$$
$$=\prod M(3,4,6,7,11,12,13,14,15)$$



WX \ YZ	00	01	11	10
00			0	
01	0		0	0
11	0	0	0	0
10			0	

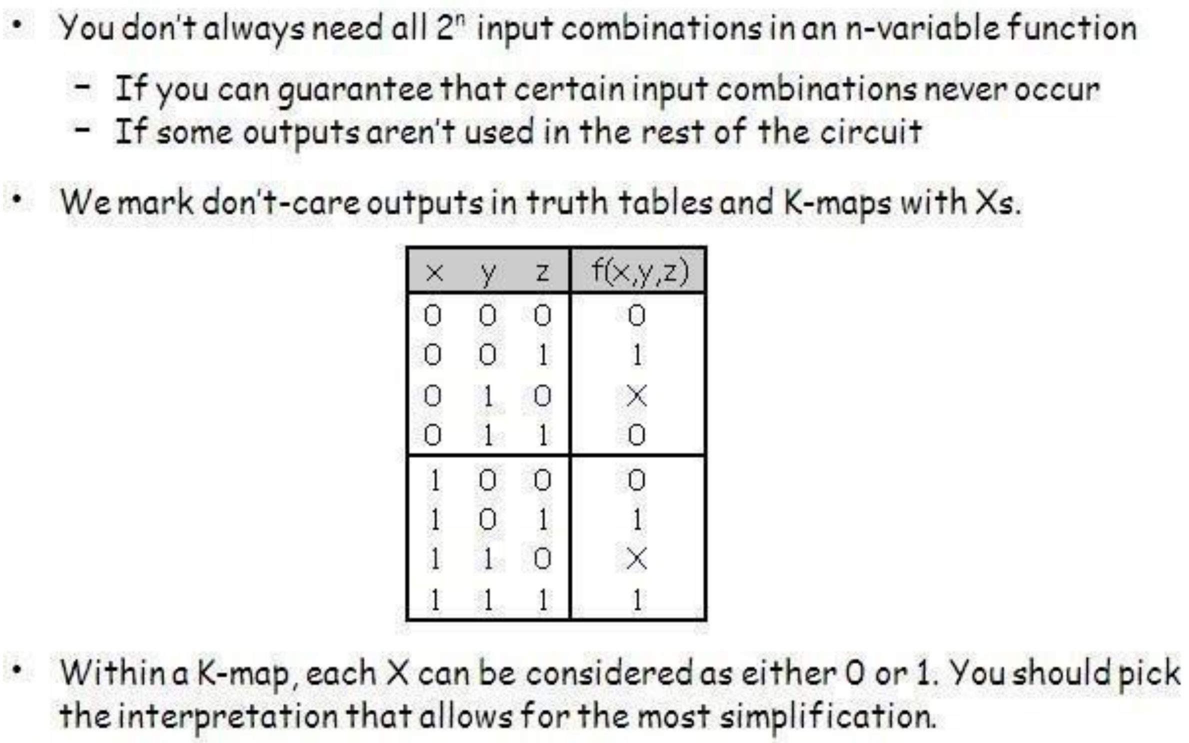
$$F(W,X,Y,Z)=(W'+X')(Y'+Z')(X'+Z)$$

Or,

$$F(W,X,Y,Z)=X'Y'+X'Z'+W'Y'Z$$

Which one is the minimal one?

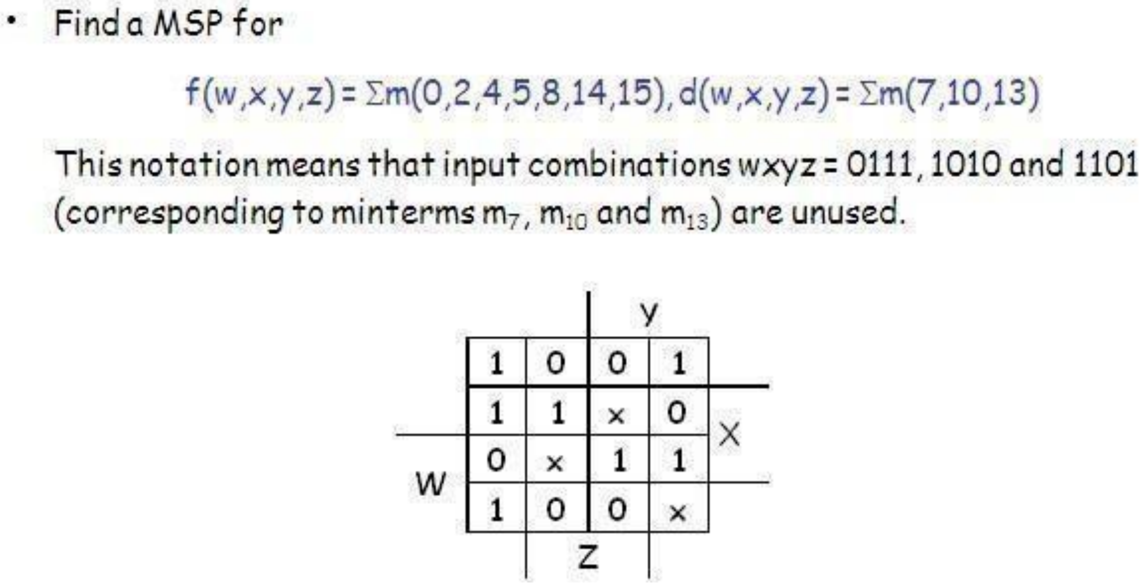
Don't care



- You don't always need all 2^n input combinations in an n-variable function
  - If you can guarantee that certain input combinations never occur
  - If some outputs aren't used in the rest of the circuit
- We mark don't-care outputs in truth tables and K-maps with Xs.

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	X
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	X
1	1	1	1

- Within a K-map, each X can be considered as either 0 or 1. You should pick the interpretation that allows for the most simplification.



- Find a MSP for

$$f(w,x,y,z) = \Sigma m(0,2,4,5,8,14,15), d(w,x,y,z) = \Sigma m(7,10,13)$$

This notation means that input combinations wxyz = 0111, 1010 and 1101 (corresponding to minterms m_7, m_{10} and m_{13}) are unused.

1	0	0	1
1	1	x	0
0	x	1	1
1	0	0	x

K-map Summary

K-maps are an alternative to algebra for simplifying expressions

The result is a MSP/MPS, which leads to a minimal two-level circuit

It‘s easy to handle don‘t-careconditions

K-maps are really only good for manual simplification of small expressions...

Things to keep in mind:

Remember the correct order of minterms/maxterms on the K-map

When grouping, you can wrap around all sides of the K-map, and your groups can overlap

Make as few rectangles as possible, but make each of them as large as possible. This leads to fewer, but simpler, product terms

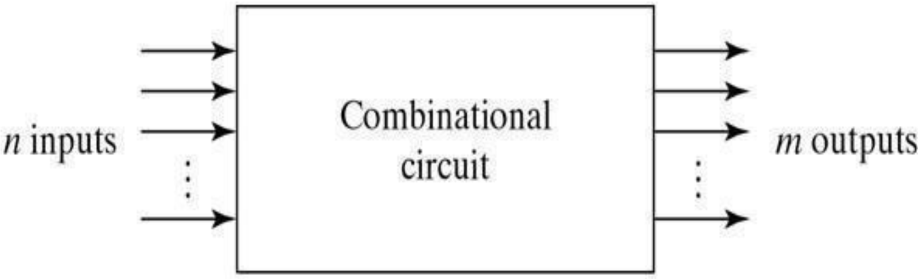
There may be more than one valid solution

UNIT-III

SEQUENTIAL CIRCUITS DESIGN

Combinational Logic

- Logic circuits for digital systems may be combinational or sequential.
- A combinational circuit consists of input variables, logic gates, and output variables.



For n input variables,there are 2^n possible combinations of binary input variables .For each possible input Combination ,there is one and only one possible output combination.A combinational circuit can be described by m Boolean functions one for each output variables.Usually the input s comes from flip-flops and outputs goto flip-flops.

Design Procedure:

1. The problem is stated
2. The number of available input variables and required output variables is determined.
3. 3.The input and output variables are assigned letter symbols.
4.The truth table that defines the required relationship between inputs and outputs is derived.
5.The simplified Boolean function for each output is obtained.
6.The logic diagram is drawn.

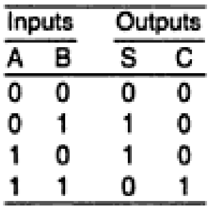
Adders:

Digital computers perform variety of information processing tasks,the one is arithmetic operations.And the most basic arithmetic operation is the addition of two binary digits.i.e, 4 basic possible operations are:

0+0=0,0+1=1,1+0=1,1+1=10

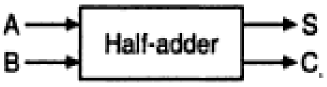
The first three operations produce a sum whose length is one digit, but when augends and addend bits are equal to 1,the binary sum consists of two digits.The higher significant bit of this result is called a carry.A combinational circuit that performs the addition of two bits is called a half- adder. One that performs the addition of 3 bits (two significant bits & previous carry) is called a full adder.& 2 half adder can employ as a full-adder.

The Half Adder: A Half Adder is a combinational circuit with two binary inputs (augends and addend bits and two binary outputs (sum and carry bits.) It adds the two inputs (A and B) and produces the sum
(S) and the carry (C) bits. It is an arithmetic operation of addition of two single bit words.



Inputs		Outputs	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

(a) Truth table



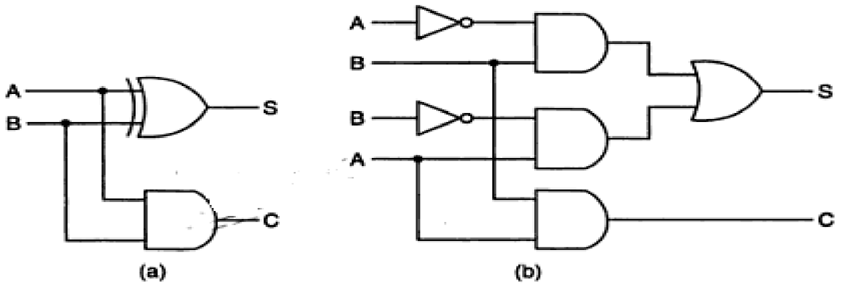
(b) Block diagram

The Sum(S) bit and the carry (C) bit, according to the rules of binary addition, the sum (S) is the X-OR of A and B (It represents the LSB of the sum). Therefore,

S=A+ $A \oplus B$

The carry (C) is the AND of A and B (it is 0 unless both the inputs are 1).Therefore, C=AB A half-adder can

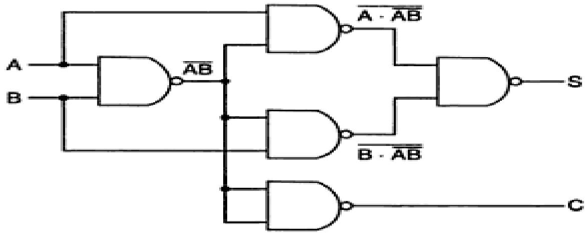
be realized by using one X-OR gate and one AND gate a



Logic diagrams of half-adder

NAND LOGIC:

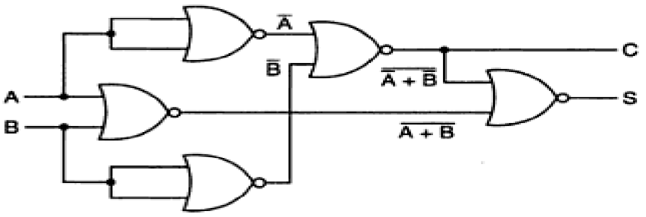
$$
\begin{aligned}
S &= A\bar{B} + \bar{A}B = A\bar{B} + A\bar{A} + \bar{A}B + B\bar{B} \\
&= A(\bar{A} + \bar{B}) + B(\bar{A} + \bar{B}) \\
&= A \cdot \overline{AB} + B \cdot \overline{AB} \\
&= \overline{\overline{A \cdot \overline{AB}} \cdot \overline{B \cdot \overline{AB}}} \\
C &= AB = \overline{\overline{AB}}
\end{aligned}
$$


$\overline{A \cdot \overline{AB}}$
A
B
$\overline{AB}$
S
$\overline{B \cdot \overline{AB}}$
C

Logic diagram of a half-adder using only 2-input NAND gates.

NOR Logic:

$$
\begin{aligned}
S &= A\bar{B} + \bar{A}B = A\bar{B} + A\bar{A} + \bar{A}B + B\bar{B} \\
&= A(\bar{A} + \bar{B}) + B(\bar{A} + \bar{B}) \\
&= (A + B)(\bar{A} + \bar{B}) \\
&= \overline{\overline{A + B} + \overline{\bar{A} + \bar{B}}} \\
C &= AB = \overline{\overline{AB}} = \overline{\bar{A} + \bar{B}}
\end{aligned}
$$



Logic diagram of a half-adder using only 2-input NOR gates.

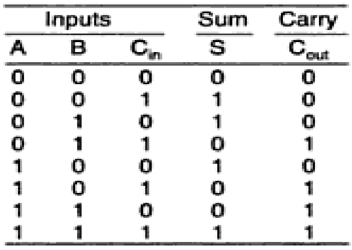
The Full Adder:

A Full-adder is a combinational circuit that adds two bits and a carry and outputs a sum bit and a carry bit. To add two binary numbers, each having two or more bits, the LSBs can be added by using a half-adder.

The carry resulted from the addition of the LSBs is carried over to the next significant column and added to the two bits in that column. So, in the second and higher columns, the two data bits of that column and the carry bit generated from the addition in the previous column need to be added.

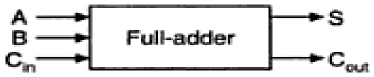
The full-adder adds the bits A and B and the carry from the previous column called the carry-in Cin and outputs the sum bit S and the carry bit called the carry-out C_{out} . The variable S gives the value of the least significant bit of the sum. The variable C_{out} gives the output carry.The

eight rows under the input variables designate all possible combinations of 1s and 0s that these variables may have. The 1s and 0s for the output variables are determined from the arithmetic sum of the input bits. When all the bits are 0s , the output is 0. The S output is equal to 1 when only 1 input is equal to 1 or when all the inputs are equal to 1. The C_{out} has a carry of 1 if two or three inputs are equal to 1.



Inputs			Sum	Carry
A	B	C_{in}	S	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

(a) Truth table



(b) Block diagram

Full-adder.

From the truth table, a circuit that will produce the correct sum and carry bits in response to every possible combination of A,B and C_{in} is described by

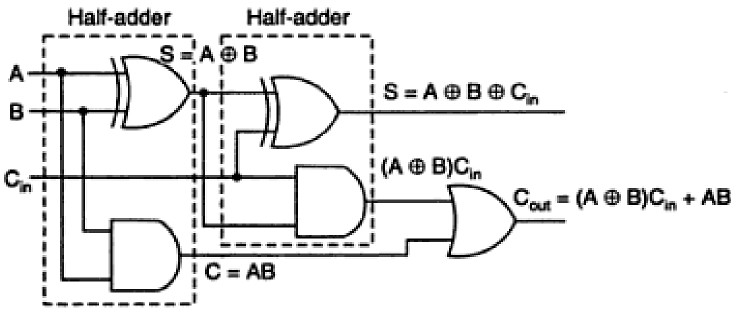
$S \square ABCin \square ABCin \square ABCin \square ABCin \; Cout \square ABCin \square ABCin \square ABCin \square ABCin$

and

$S \square A \square B \square Cin$

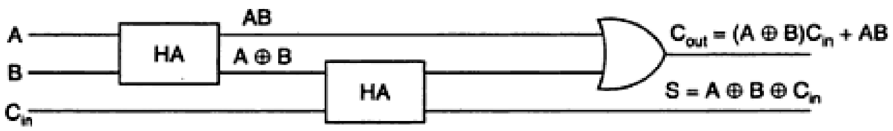
$Cout \square ACin \square BCin \square AB$

The sum term of the full-adder is the X-OR of A,B, and C_{in}, i.e, the sum bit the modulo sum of the data bits in that column and the carry from the previous column. The logic diagram of the full-adder using two X-OR gates and two AND gates (i.e, Two half adders) and one OR gate is



Logic diagram of a full-adder using two half-adders.

The block diagram of a full-adder using two half-adders is :



Block diagram of a full-adder using two half-adders.

Even though a full-adder can be constructed using two half-adders, the disadvantage is that the bits must propagate through several gates in accession, which makes the total propagation delay greater than that of the full-adder circuit using AOI logic.

The Full-adder neither can also be realized using universal logic, i.e., either only NAND gates or only NOR gates as

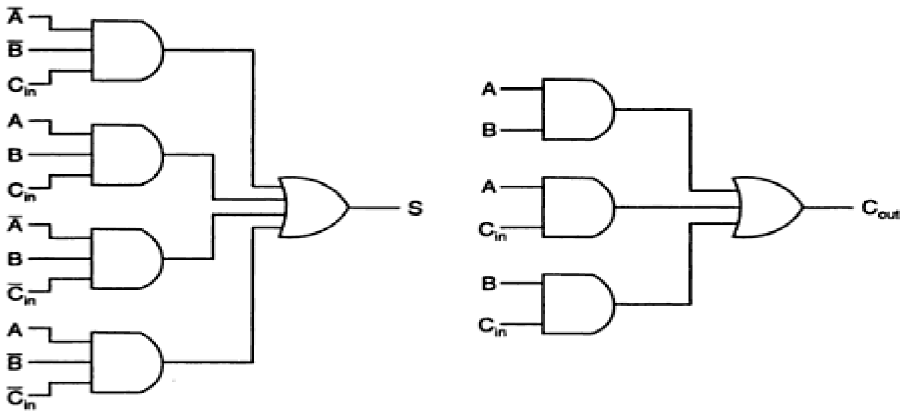
$$A \oplus B = \overline{\overline{A \cdot \overline{AB}} \cdot \overline{B \cdot \overline{AB}}}$$

Then

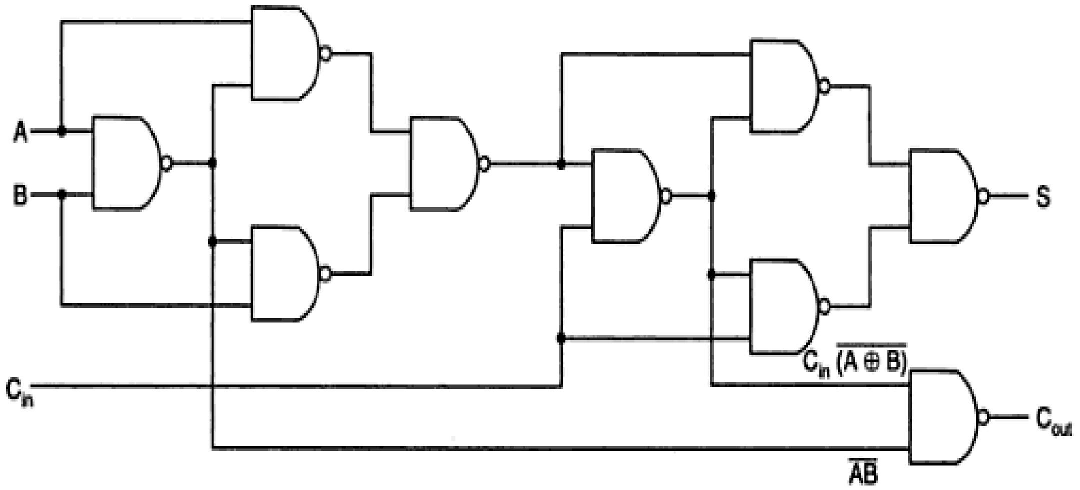
$$S = A \oplus B \oplus C_{in} = \overline{\overline{(A \oplus B) \cdot \overline{(A \oplus B)C_{in}}} \cdot \overline{C_{in} \cdot \overline{(A \oplus B)C_{in}}}}$$

NAND Logic:

$$C_{out} = C_{in}(A \oplus B) + AB = \overline{\overline{C_{in}(A \oplus B)} \cdot \overline{AB}}$$



Sum and carry bits of a full-adder using AOI logic.



Logic diagram of a full-adder using only 2-input NAND gates.

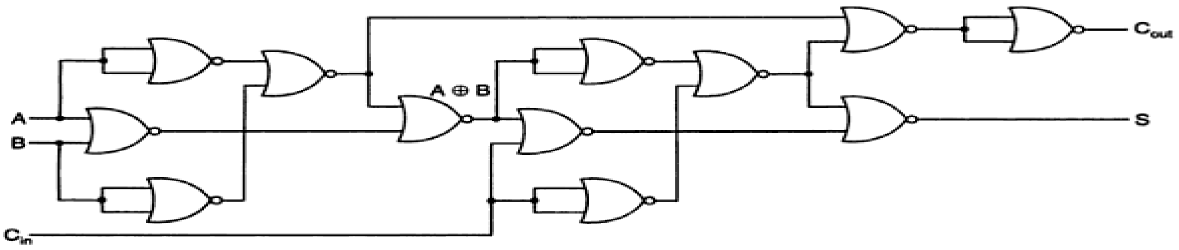
NOR Logic:

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

Then

$$S = A \oplus B \oplus C_{in} = \overline{\overline{(A \oplus B) + C_{in}} + \overline{\overline{(A \oplus B)} + \overline{C}_{in}}}$$

$$C_{out} = AB + C_{in}(A \oplus B) = \overline{\overline{A} + \overline{B}} + \overline{\overline{C}_{in} + \overline{A \oplus B}}$$



Logic diagram of a full-adder using only 2-input NOR gates.

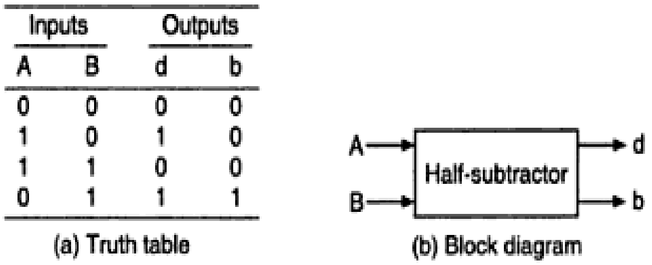
Subtractors:

The subtraction of two binary numbers may be accomplished by taking the complement of the subtrahend and adding it to the minuend. By this, the subtraction operation becomes an addition operation and instead of having a separate circuit for subtraction, the adder itself can be used to perform subtraction. This results in reduction of hardware. In subtraction, each subtrahend bit of the number is subtracted from its corresponding significant minuend bit to form a difference bit. If the minuend bit is smaller than the subtrahend bit, a 1 is borrowed from the next significant position., that has been borrowed must be conveyed to the next higher pair of bits by means of a signal coming out (output) of a given stage and going into (input) the next higher stage.

The Half-Subtractor:

A Half-subtractor is a combinational circuit that subtracts one bit from the other and produces the difference. It also has an output to specify if a 1 has been borrowed. . It is used to subtract the LSB of the subtrahend from the LSB of the minuend when one binary number is subtracted from the other.

A Half-subtractor is a combinational circuit with two inputs A and B and two outputs d and b. d indicates the difference and b is the output signal generated that informs the next stage that a 1 has been borrowed. When a bit B is subtracted from another bit A, a difference bit (d) and a borrow bit (b) result according to the rules given as



Inputs		Outputs	
A	B	d	b
0	0	0	0
1	0	1	0
1	1	0	0
0	1	1	1

(a) Truth table

(b) Block diagram

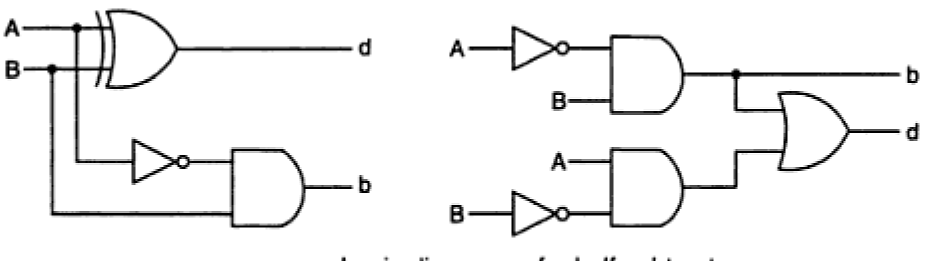
Half-subtractor.

The output borrow b is a 0 as long as A≥B. It is a 1 for A=0 and B=1. The d output is the result of the arithmetic operation2b+A-B.

A circuit that produces the correct difference and borrow bits in response to every possible combination of the two 1-bit numbers is , therefore ,

d=A+ $A \oplus B$ and b=B

That is, the difference bit is obtained by X-OR ing the two inputs, and the borrow bit is obtained by ANDing the complement of the minuend with the subtrahend.Note that logic for this exactly the same as the logic for output S in the half-adder.



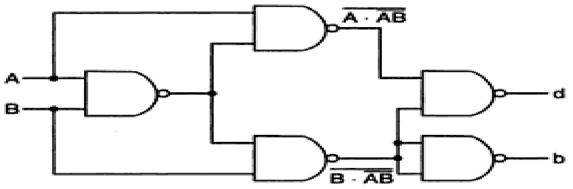
Logic diagrams of a half-subtractor.

A half-substractor can also be realized using universal logic either using only NAND gates or using NOR gates as:

NAND Logic:

$$d = A \oplus B = \overline{\overline{A \cdot \overline{AB}} \cdot \overline{B \cdot \overline{AB}}}$$

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


$\overline{A \cdot \overline{AB}}$
A
B
d
b
$\overline{B \cdot \overline{AB}}$

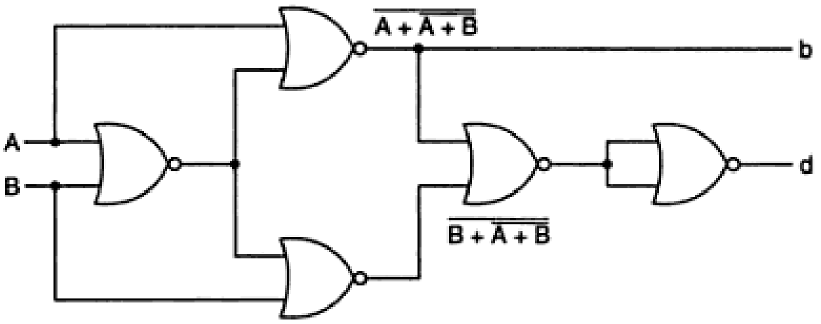
Logic diagram of a half-subtractor using only 2-input NAND gates.

NOR Logic:

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

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

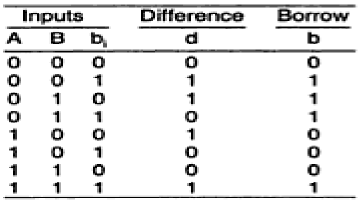
$$d = \overline{A}B = \overline{A}(A + B) = \overline{\overline{\overline{A}(A + B)}} = \overline{A + \overline{(A + B)}}$$



Logic diagram of a half-subtractor using only 2-input NOR gates.

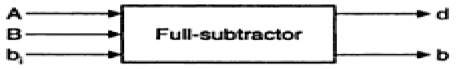
The Full-Subtractor:

The half-subtractor can be only for LSB subtraction. IF there is a borrow during the subtraction of the LSBs, it affects the subtraction in the next higher column; the subtrahend bit is subtracted from the minuend bit, considering the borrow from that column used for the subtraction in the preceding column. Such a subtraction is performed by a full-subtractor. It subtracts one bit (B) from another bit (A) , when already there is a borrow bi from this column for the subtraction in the preceding column, and outputs the difference bit (d) and the borrow bit(b) required from the next d and b. The two outputs present the difference and output borrow. The 1s and 0s for the output variables are determined from the subtraction of A-B-b_i.



Inputs			Difference	Borrow
A	B	b_i	d	b
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

(a) Truth table



(b) Block diagram

Full-subtractor.

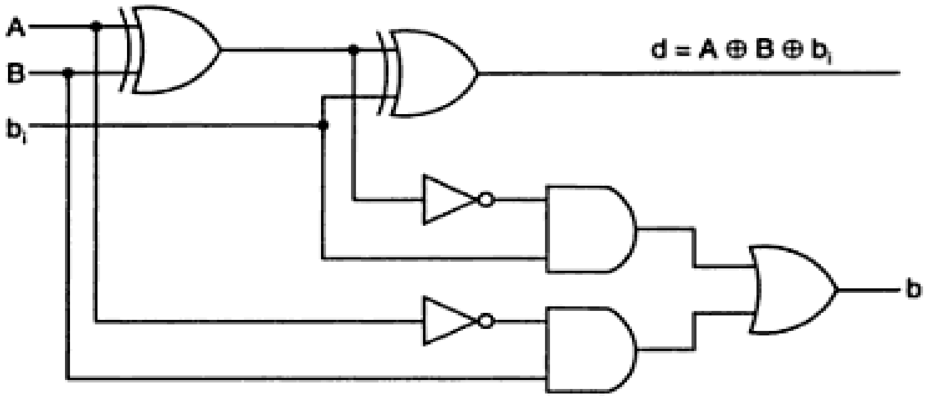
From the truth table, a circuit that will produce the correct difference and borrow bits in response to every possiblecombinations of A,B and b_i is

$$\begin{aligned} d &= \bar{A}\bar{B}b_i + \bar{A}B\,\bar{b}_i + A\bar{B}\,\bar{b}_i + ABb_i \\ &= b_i(AB + \bar{A}\bar{B}) + \bar{b}_i(A\bar{B} + \bar{A}B) \\ &= b_i(\overline{A \oplus B}) + \bar{b}_i(A \oplus B) = A \oplus B \oplus b_i \end{aligned}$$

and

$$\begin{aligned} b &= \bar{A}\bar{B}b_i + \bar{A}B\,\bar{b}_i + \bar{A}Bb_i + ABb_i = \bar{A}B(b_i + \bar{b}_i) + (AB + \bar{A}\bar{B})b_i \\ &= \bar{A}B + (\overline{A \oplus B})b_i \end{aligned}$$

A full-subtractor can be realized using X-OR gates and AOI gates as



Logic diagram of a full-subtractor.

The full subtractor can also be realized using universal logic either using only NAND gates or using NOR gates as:

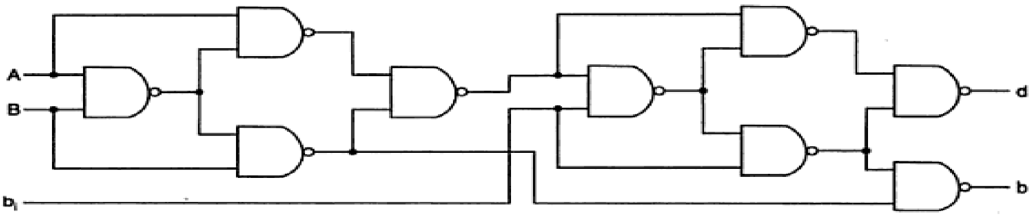
NAND Logic:

$$d = A \oplus B \oplus b_i = \overline{\overline{(A \oplus B) \oplus b_i}} = \overline{\overline{(A \oplus B)\overline{(A \oplus B)b_i}} \cdot \overline{b_i\overline{(A \oplus B)b_i}}}$$

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

$$= \overline{\overline{\bar{A}B} \cdot \overline{b_i(\overline{A \oplus B})}} = \overline{\overline{B(\bar{A} + \bar{B})} \cdot \overline{b_i(\bar{b}_i + \overline{(A \oplus B)})]}}$$

$$= \overline{\overline{B \cdot \overline{AB}} \cdot \overline{b_i[\overline{b_i \cdot (A \oplus B)}]}}$$



Logic diagram of a full-subtractor using only 2-input NAND gates.

NOR Logic:

$$d = A \oplus B \oplus b_i = \overline{\overline{(A \oplus B) \oplus b_i}}$$

$$= \overline{(A \oplus B)b_i + \overline{(A \oplus B)}\bar{b}_i}$$

$$= \overline{[(A \oplus B) + \overline{(A \oplus B)}\bar{b}_i][b_i + \overline{(A \oplus B)}\bar{b}_i]}$$

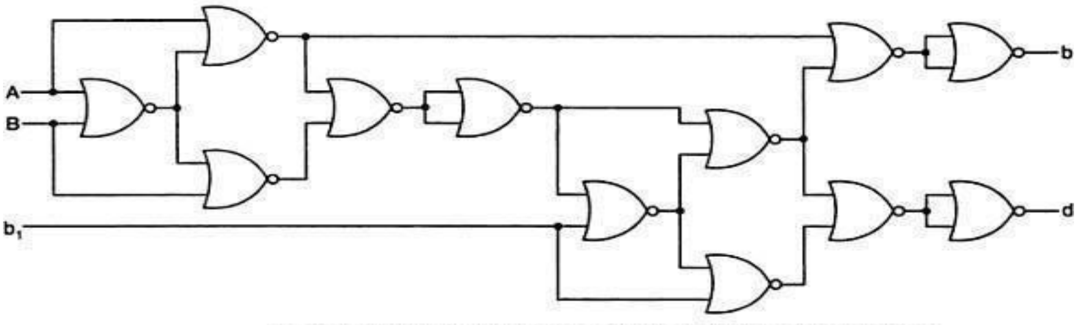
$$= \overline{(A \oplus B) + \overline{(A \oplus B) + b_i}} + \overline{b_i + \overline{(A \oplus B) + b_i}}$$

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

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

$$= \bar{A}(A + B) + (\overline{A \oplus B})[(A \oplus B) + b_i]$$

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

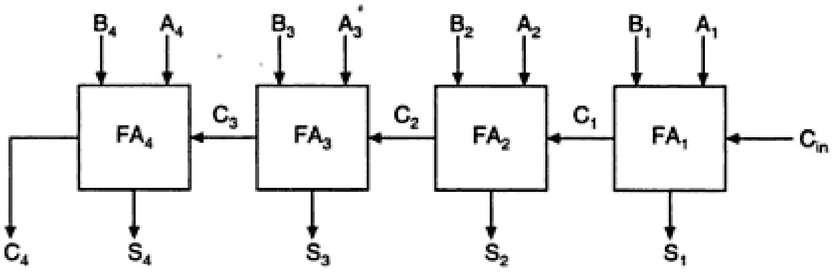


Logic diagram of a full subtractor using only 2-input NOR gates.

Binary Parallel Adder:

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

The interconnection of four full-adder (FA) circuits to provide a 4-bit parallel adder. The augends bits of A and addend bits of B are designated by subscript numbers from right to left, with subscript 1 denoting the lower –order bit. The carries are connected in a chain through the full-adders. The input carry to the adder is C_{in} and the output carry is C_4. The S output generates the required sum bits. When the 4-bit full- adder circuit is enclosed within an IC package, it has four terminals for the augends bits, four terminals for the addend bits, four terminals for the sum bits, and two terminals for the input and output carries. AN n-bit parallel adder requires n-full adders. It can be constructed from 4-bit, 2-bit and 1-bit full adder ICs by cascading several packages. The output carry from one package must be connected to the input carry of the one with the next higher –order bits. The 4-bit full adder is a typical example of an MSI function.



Logic diagram of a 4-bit binary parallel adder.

Ripple carry adder:

In the parallel adder, the carry –out of each stage is connected to the carry-in of the next stage. The sum and carry-out bits of any stage cannot be produced, until sometime after the carry-in of that stage occurs. This is due to the propagation delays in the logic circuitry,

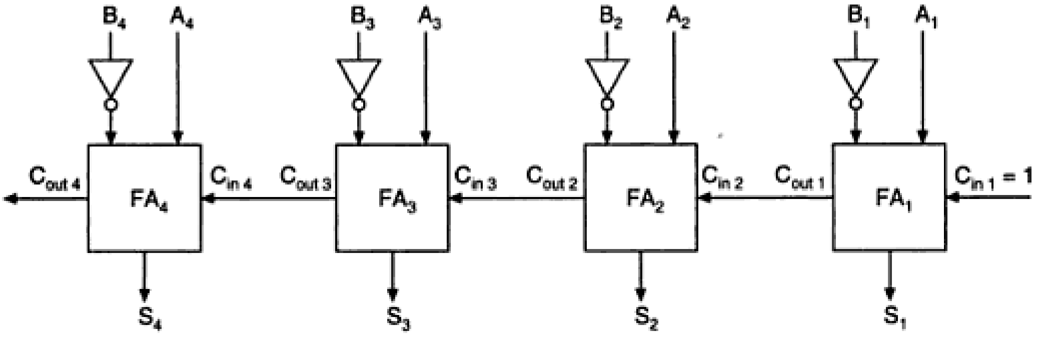
which lead to a time delay in the addition process. The carry propagation delay for each full- adder is the time between the application of the carry-in and the occurrence of the carry-out.

The 4-bit parallel adder, the sum (S_1) and carry-out (C_1) bits given by FA_1 are not valid, until after the propagation delay of FA_1. Similarly, the sum S_2 and carry-out (C_2) bits given by FA_2 are not valid until after the cumulative propagation delay of two full adders (FA_1 and FA_2) , and so on. At each stage ,the sum bit is not valid until after the carry bits in all the preceding stages are valid. Carry bits must propagate or ripple through all stages before the most significant sum bit is valid. Thus, the total sum (the parallel output) is not valid until after the cumulative delay of all the adders.

The parallel adder in which the carry-out of each full-adder is the carry-in to the next most significant adder is called a ripple carry adder.. The greater the number of bits that a ripple carry adder must add, the greater the time required for it to perform a valid addition. If two numbers are added such that no carries occur between stages, then the add time is simply the propagation time through a single full-adder.

4-Bit Parallel Subtractor:

The subtraction of binary numbers can be carried out most conveniently by means of complements , the subtraction A-B can be done by taking the 2_s complement of B and adding it to A . The 2_s complement can be obtained by taking the 1_s complement and adding 1 to the least significant pair of bits. The 1_s complement can be implemented with inverters as

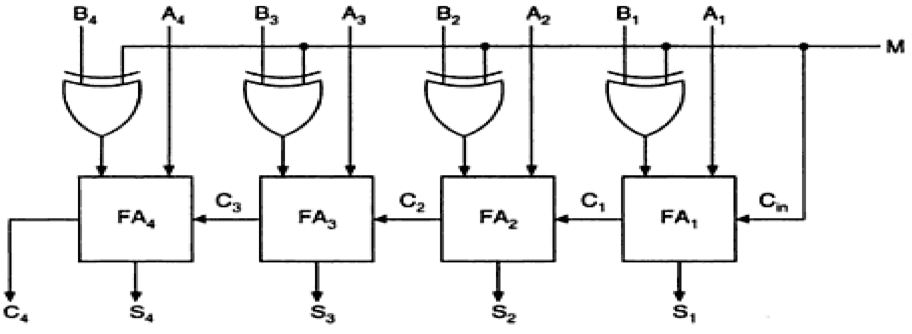


Logic diagram of a 4-bit parallel subtractor.

Binary-Adder Subtractor:

A 4-bit adder-subtractor, the addition and subtraction operations are combined into one circuit with one common binary adder. This is done by including an X-OR gate with each full-adder. The mode input M controls the operation. When M=0, the circuit is an adder, and when M=1, the circuit becomes a subtractor. Each X-OR gate receives input M and one of the inputs of B. When M=0,.The full-adder receives the value of B , the input carry is 0 and the circuit performs A+B. whenand C_1=1. The B inputs are complemented and a 1 is through the input carry. The circuit performs the operation A plus the 2_s complement of B.

$B \oplus 0 = B B \oplus 1 = B'$



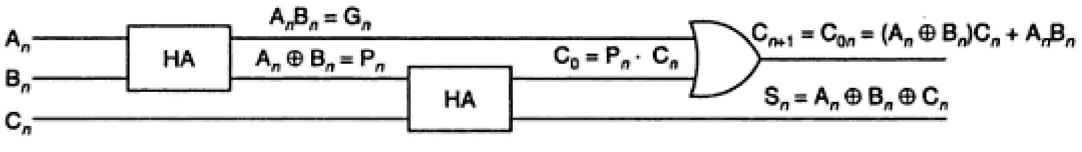
Logic diagram of a 4-bit binary adder-subtractor.

The Look-Ahead –Carry Adder:

In 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-in bits for all the stages simultaneously.

The method of speeding up the addition process is based on the two additional functions of the full-adder, called the carry generate and carry propagate functions.

Consider one full adder stage; say the nth stage of a parallel adder as shown in fig. we know that is made by two half adders and that the half adder contains an X-OR gate to produce the sum and an AND gate to produce the carry. If both the bits A_n and B_n are 1s, a carry has to be generated in this stage regardless of whether the input carry C_{in} is a 0 or a 1. This is called generated carry, expressed as $G_n= A_n.B_n$ which has to appear at the output through the OR gate as shown in fig.


A_n
B_n
C_n
HA
HA
$A_nB_n = G_n$
$A_n \oplus B_n = P_n$
$C_0 = P_n \cdot C_n$
$C_{n+1} = C_{0n} = (A_n \oplus B_n)C_n + A_nB_n$
$S_n = A_n \oplus B_n \oplus C_n$

A full adder (*n*th stage of a parallel adder).

Thereis another possibility of producing a carry out. X-OR gate inside the half-adder

at the input produces an intermediary sum bit- call it P_n –which is expressed as $P_n = A_n \oplus B_n$.
Next P_n and C_n are added using the X-OR gate inside the second half adder to produce the final

sum bit and $S_n = P_n \oplus C_n$ where $P_n = A_n \oplus B_n$ and output carryC_0= $P_n.C_n$=($A_n \oplus B_n$)C_n which becomes carry for the (n+1) thstage.

Consider the case of both P_n and C_n being 1. The input carry C_n has to be propagated to the output only if P_n is 1. If P_n is 0, even if C_n is 1, the and gate in the second half-adder will inhibit C_n . the carry out of the nth stage is 1 when either G_n=1 or $P_n.C_n$ =1 or both G_n and $P_n.C_n$ are equal to 1.

For the final sum and carry outputs of the nth stage, we get the following Boolean expressions.

$$S_n = P_n \oplus C_n \text{ where } P_n = A_n \oplus B_n$$
$$C_{on} = C_{n+1} = G_n + P_nC_n \text{ where } G_n = A_n \cdot B_n$$

Observe the recursive nature of the expression for the output carry at the nth stage which becomes the input carry for the (n+1)st stage .it is possible to express the output carry of a higher significant stage is the carry-out of the previous stage.

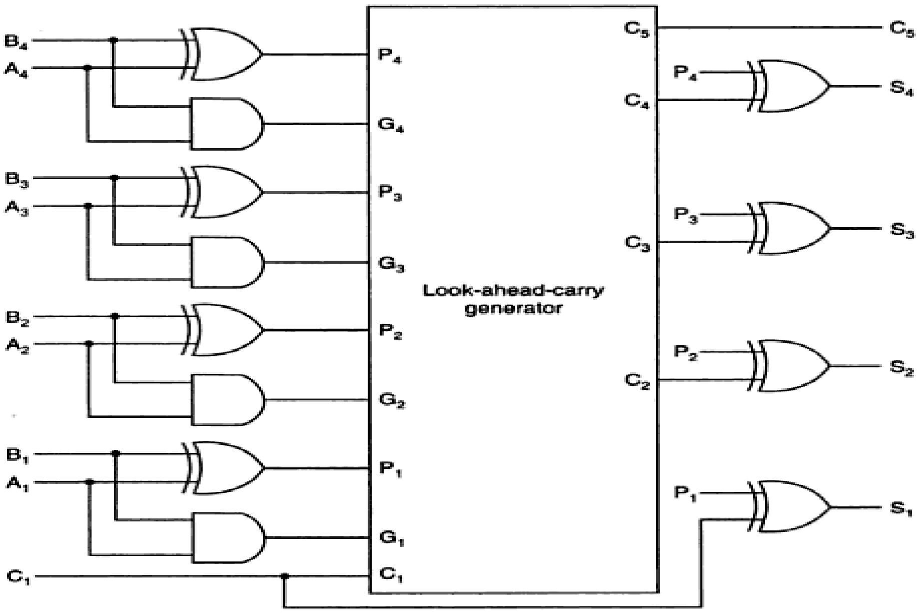
Based on these , the expression for the carry-outs of various full adders are as follows,

$$C_1 = G_0 + P_0 \cdot C_0$$
$$C_2 = G_1 + P_1 \cdot C_1 = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_0$$
$$C_3 = G_2 + P_2 \cdot C_2 = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_0$$
$$C_4 = G_3 + P_3 \cdot C_3 = G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot G_1 + P_3 \cdot P_2 \cdot P_1 \cdot G_0 + P_3 \cdot P_2 \cdot P_1 \cdot P_0 \cdot C_0$$

The general expression for *n* stages designated as 0 through (*n* – 1) would be

$$C_n = G_{n-1} + P_{n-1} \cdot C_{n-1} = G_{n-1} + P_{n-1} \cdot G_{n-2} + P_{n-1} \cdot P_{n-2} \cdot G_{n-3} + \ldots + P_{n-1} \cdot \ldots P_0 \cdot C_0$$

Observe that the final output carry is expressed as a function of the input variables in SOP form. Which is two level AND-OR or equivalent NAND-NAND form. Observe that the full look-ahead scheme requires the use of OR gate with (n+1) inputs and AND gates with number of inputs varying from 2 to (n+1).



Logic diagram of a 4-bit look-ahead-carry adder.

2's complement Addition and Subtraction using Parallel Adders:

Most modern computers use the 2_s complement system to represent negative numbers and to perform subtraction operations of signed numbers can be performed using only the addition operation ,if we use the 2_s complement form to represent negative numbers.

The circuit shown can perform both addition and subtraction in the 2_s complement. This adder/subtractor circuit is controlled by the control signal ADD/SUB_. When the ADD/SUB_ level is HIGH, the circuit performs the addition of the numbers stored in registers A and B. When the ADD/Sub_ level is LOW, the circuit subtract the number in register B from the number in register A. The operation is:

When ADD/SUB_ is a 1:

1. AND gates 1,3,5 and 7 are enabled , allowing B_0,B_1,$B_2$and B_3 to pass to the OR gates 9,10,11,12 . AND gates 2,4,6 and 8 are disabled , blocking B_0_,B_1_,B_2_, and B_3_ from reaching the OR gates 9,10,11 and 12.

2. The two levels B_0 to B_3 pass through the OR gates to the 4-bit parallel adder, to be added to the bits A_0 to A_3. The sum appears at the output S_0 to S_3

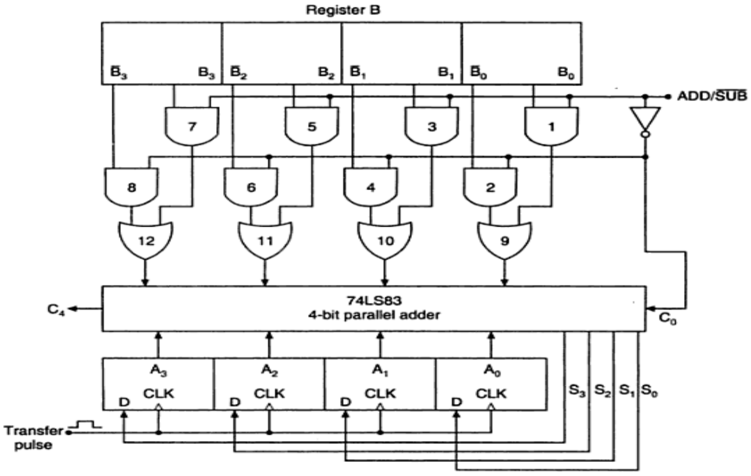
3. Add/SUB_ =1 causes no carry into the adder. When ADD/SUB_ is a0:

4. AND gates 1,3,5 and 7 are disabled , allowing B_0,B_1,$B_2$and B_3 from reaching the OR gates 9,10,11,12 . AND gates 2,4,6 and 8 are enabled , blocking B_0_,B_1_,B_2_, and B_3_ from reaching the OR gates.

1. The two levels B_0_ to B_3_ pass through the OR gates to the 4-bit parallel adder, to be added to the bits A_0 to $A_{3.}$The C_0 is now thus the number in register B is converted to its 2_s complement form.

2. The difference appears at the output S_0 to$S_{3.}$

Adders/Subtractors used for adding and subtracting signed binary numbers. In computers , the output is transferred into the register A (accumulator) so that the result of the addition or subtraction always end up stored in the register A This is accomplished by applying a transfer pulse to the CLK inputs of register A.



Logic diagram of a parallel adder/subtractor using 2's complement system.

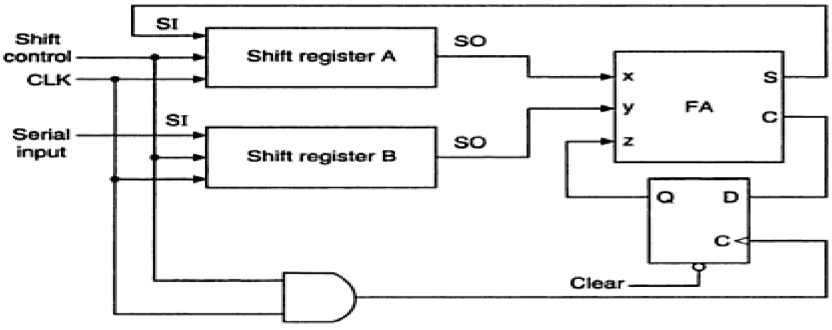
Serial Adder:

A serial adder is used to add binary numbers in serial form. The two binary numbers to be added serially are stored in two shift registers A and B. Bits are added one pair at a time through a single full adder (FA) circuit as shown. The carry out of the full-adder is transferred to a D flip- flop. The output of this flip-flop is then used as the carry input for the next pair of significant bits. The sum bit from the S output of the full-adder could be transferred to a third shift register. By shifting the sum into A while the bits of A are shifted out, it is possible to use one register for storing both augend and the sum bits. The serial input register B can be used to transfer a new binary number while the addend bits are shifted out during the addition.

The operation of the serial adder is:

Initially register A holds the augend, register B holds the addend and the carry flip-flop is cleared to 0. The outputs (SO) of A and B provide a pair of significant bits for the full-adder at x and y. The shift control enables both registers and carry flip-flop , so, at the clock pulse both registers are shifted once to the right, the sum bit from S enters the left most flip-flop of A , and the output carry is transferred into flip-flop Q . The shift control enables the registers for a number of clock pulses equal to the number of bits of the registers. For each succeeding clock pulse a new sum bit is transferred to A, a new carry is transferred to Q, and both registers are shifted once to the right. This process continues until the shift control is disabled. Thus the addition is accomplished by passing each pair of bits together with the previous carry through a single full adder circuit and transferring the sum, one bit at a time, into register A.

Initially, register A and the carry flip-flop are cleared to 0 and then the first number is added from B. While B is shifted through the full adder, a second number is transferred to it through its serial input. The second number is then added to the content of register A while a third number is transferred serially into register B. This can be repeated to form the addition of two, three, or more numbers and accumulate their sum in register A.



Logic diagram of a serial adder.

Difference between Serial and Parallel Adders:

The parallel adder registers with parallel load, whereas the serial adder uses shift registers. The number of full adder circuits in the parallel adder is equal to the number of bits in the binary numbers, whereas the serial adder requires only one full adder circuit and a carry flip- flop. Excluding the registers, the parallel adder is a combinational circuit, whereas the serial adder is a sequential circuit. The sequential circuit in the serial adder consists of a full-adder and a flip-flop that stores the output carry.

BCD Adder:

The BCD addition process:

1. Add the 4-bit BCD code groups for each decimal digit position using ordinary binary addition.

2. For those positions where the sum is 9 or less, the sum is in proper BCD form and no correction isneeded.

3. When the sum of two digits is greater than 9, a correction of 0110 should be added to that sum, to produce the proper BCD result. This will produce a carry to be added to the next decimal position.

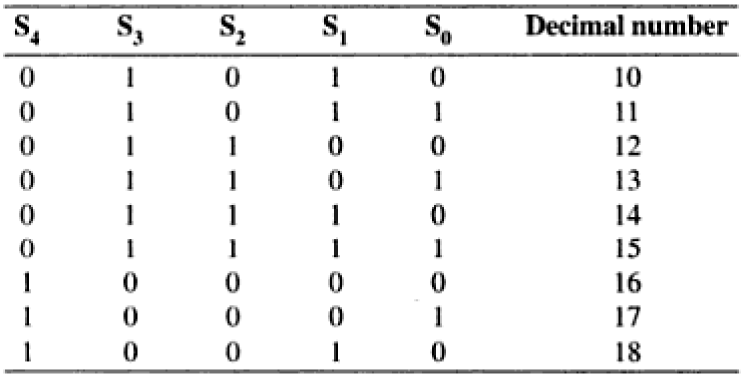
A BCD adder circuit must be able to operate in accordance with the above steps. In other words, the circuit must be able to do the following:

1. Add two 4-bit BCD code groups, using straight binaryaddition.

2. Determine, if the sum of this addition is greater than 1101 (decimal 9); if it is , add 0110 (decimal 6) to this sum and generate a carry to the next decimal position.

The first requirement is easily met by using a 4- bit binary parallel adder such as the 74LS83 IC .For example , if the two BCD code groups $A_3A_2A_1A_0$and $B_3B_2B_1B_0$ are applied to a 4-bit parallel adder, the adder will output $S_4S_3S_2S_1S_0$, where S_4 is actually C_4 , the carry –out of the MSB bits.

The sum outputs $S_4S_3S_2S_1S_0$ can range anywhere from 00000 to 100109when both the BCD code groups are 1001=9). The circuitry for a BCD adder must include the logic needed to detect whenever the sum is greater than 01001, so that the correction can be added in. Those cases , where the sum is greater than 1001 are listed as:



S_4	S_3	S_2	S_1	S_0	Decimal number
0	1	0	1	0	10
0	1	0	1	1	11
0	1	1	0	0	12
0	1	1	0	1	13
0	1	1	1	0	14
0	1	1	1	1	15
1	0	0	0	0	16
1	0	0	0	1	17
1	0	0	1	0	18

Let us define a logic output X that will go HIGH only when the sum is greater than 01001 (i.e, for the cases in table). If examine these cases ,see that X will be HIGH for either of the following conditions:

1. Whenever S_4 =1(sum greater than15)

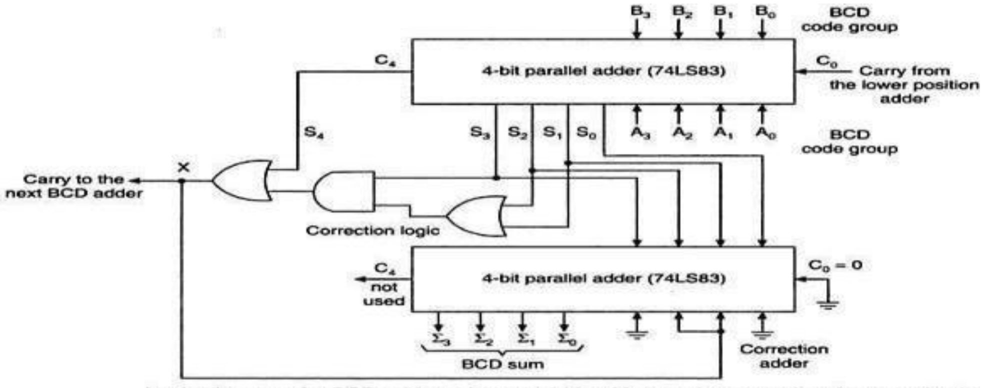
2. Whenever S_3 =1 and either S_2 or S_1 or both are 1 (sum 10 to 15) This condition can be expressed as

$X=S_4+S_3(S_2+S_1)$

Whenever X=1, it is necessary to add the correction factor 0110 to the sum bits, and to generate a carry. The circuit consists of three basic parts. The two BCD code groups $A_3A_2A_1A_0$ and $B_3B_2B_1B_0$ are added together in the upper 4-bit adder, to produce the sum $S_4S_3S_2S_1S_0.$ The logic gates shown implement the expression for X. The lower 4-bit adder will add the correction 0110 to the sum bits, only when X=1, producing the final BCD sum output represented by

$\sum_3\sum_2\sum_1\sum_0.$ The X is also the carry-out that is produced when the sum is greater than 01001. When X=0, there is no carry and no addition of 0110. In such cases, $\sum_3\sum_2\sum_1\sum_0= S_3S_2S_1S_0.$

Two or more BCD adders can be connected in cascade when two or more digit decimal numbers are to be added. The carry-out of the first BCD adder is connected as the carry-in of the second BCD adder, the carry-out of the second BCD adder is connected as the carry-in of the third BCD adder and so on.



Logic diagram of a BCD adder using two 4-bit adders and a correction-detector circuit.

EXCESS-3(XS-3) ADDER:

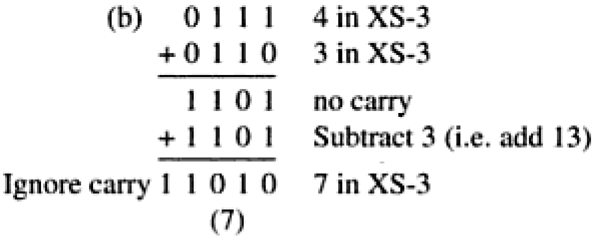
To perform Excess-3 additions,

1. Add two xs-3 code groups
2. If carry=1, add 0011(3) to the sum of those two codegroups

If carry =0, subtract 0011(3) i.e., add 1101 (13 in decimal) to the sum of those two code groups. Ex: Add 9 and 5

```
1100                                   9 in Xs-3
                   +1000               5 in xs-3
                         – –
    1              0100                there is a carry
    +0011          0011                add 3 to each group
    ----------     ----------
    0100           0111                14 in xs-3
    (1)            (4)
```

EX:



```
(b)            0 1 1 1     4 in XS-3
             + 0 1 1 0     3 in XS-3
               1 1 0 1     no carry
             + 1 1 0 1     Subtract 3 (i.e. add 13)
Ignore carry 1 1 0 1 0     7 in XS-3
                 (7)
```

Implementation of xs-3 adder using 4-bit binary adders is shown. The augend (A_3 $A_2A_1A_0$) and addend ($B_3B_2B_1B_0$) in xs-3 are added using the 4-bit parallel adder. If the carry is a 1, then 0011(3) is added to the sum bits $S_3S_2S_1S_0$ of the upper adder in the lower 4-bit parallel adder. If the carry is a 0, then 1101(3) is added to the sum bits (This is equivalent to subtracting 0011(3) from the sum bits. The correct sum in xs-3 is obtained

Excess-3 (XS-3) Subtractor:

To perform Excess-3 subtraction,

1. Complement the subtrahend
2. Add the complemented subtrahend to the minuend.
3. If carry =1, result is positive. Add 3 and end around carry to the result . If carry=0, the result is negative. Subtract 3, i.e, and take the 1_s complement of the result.

```
Ex:                   Perform 9-4
1100                                9 in xs-3
+1000                               Complement of 4 n Xs-3
--------
(1)                    0100         There is a carry
+0011                               Add 0011(3)
------------ 0111
1                                   End around carry
------------
1000                                5 in xs-3
```

The minuend and the 1_s complement of the subtrahend in xs-3 are added in the upper 4- bit parallel adder. If the carry-out from the upper adder is a 0, then 1101 is added to the sum bits of the upper adder in the lower adder and the sum bits of the lower adder are complemented to get the result. If the carry-out from the upper adder is a 1, then 3=0011 is added to the sum bits of the lower adder and the sum bits of the lower adder give the result.

Binary Multipliers:

In binary multiplication by the paper and pencil method, is modified somewhat in digital machines because a binary adder can add only two binary numbers at a time.
In a binary multiplier, instead of adding all the partial products at the end, they are added two at a time and their sum accumulated in a register (the accumulator register). In addition, when the multiplier bit is a 0,0s are not written down and added because it does not affect the final result. Instead, the multiplicand is shifted left by one bit.

The multiplication of 1110 by 1001 using this processis Multiplicand 1110 Multiplier
1001
1110 The LSB of the multiplier is a 1; write down the multiplicand;
shift the multiplicand one position to the left (1 1 1 0 0)
1110The second multiplier bit is a 0; write down the previous result 1110; shift the multiplicand to the left again (1 1 1 00 0)
+1110000The fourth multiplier bit is a 1 write down the new multiplicand add it to the first partial product to obtain the final product.
1111110
This multiplication process can be performed by the serial multiplier circuit , which multiplies two 4-bit numbers to produce an 8-bit product. The circuit consists of following elements
X register: A 4-bit shift register that stores the multiplier --- it will shift right on the falling edge of the clock. Note that 0s are shifted in from the left.
B register: An 8-bit register that stores the multiplicand; it will shift left on the falling edge of the clock. Note that 0s are shifted in from the right.
A register: An 8-bit register, i.e, the accumulator that accumulates the partial products. Adder:An 8-bit parallel adder that produces the sum of A and B registers. The adder outputs S_7 through S_0 are connected to the D inputs of the accumulator so that the sum can be transferred to the accumulator only when a clock pulse gets through the AND gate.
The circuit operation can be described by going through each step in the multiplication of 1110 by 1001. The complete process requires 4 clock cycles.

1. Before the first clock pulse: Prior to the occurrence of the first clock pulse, the register A is loaded with 00000000, the register B with the multiplicand 00001110, and the register X with the multiplier 1001. Assume that each of these registers is loaded using its asynchronous inputs(i.e., PRESET and CLEAR). The output of the adder will be the sum of A and B,i.e., 00001110.
2. First Clock pulse: Since the LSB of the multiplier (X_0) is a 1, the first clock pulse gets through the AND gate and its positive going transition transfers the sum outputs into the accumulator. The subsequent negative going transition causes the X and B registers to shift right and left, respectively. This produces a new sum of A and.
3. Second Clock Pulse: The second bit of the original multiplier is now in X_0 . Since this bit is a 0, the second clock pulse is inhibited from reaching the accumulator. Thus, the sum outputs are not transferred into the accumulator and the number in the accumulator does not change. The negative going transition of the clock pulse will again shift the X and B registers. Again a new sum is produced.
4. Third Clock Pulse: The third bit of the original multiplier is now in X_0;since this bit is a 0, the third clock pulse is inhibited from reaching the accumulator. Thus, the sum outputs are not transferred into the accumulator and the number in the accumulator does not change. The negative going transition of the clock pulse will again shift the X and B registers. Again a new sum is produced.

5. Fourth Clock Pulse: The last bit of the original multiplier is now in X_0 , and since it is a 1, the positive going transition of the fourth pulse transfers the sum into the accumulator. The accumulator now holds the final product. The negative going transition of the clock pulse shifts X and B again. Note that, X is now 0000, since all the multiplier bits have been shifted out.

Code converters:

The availability of a large variety of codes for the same discrete elements of information results in the use of different codes by different digital systems. It is sometimes necessary to use the output of one system as the input to another. A conversion circuit must be inserted between the two systems if each uses different codes for the same information. Thus a code converter is a logic circuit whose inputs are bit patterns representing numbers (or character) in one cod and whose outputs are the corresponding representation in a different code. Code converters are usually multiple output circuits.

To convert from binary code A to binary code B, the input lines must supply the bit combination of elements as specified by code A and the output lines must generate the corresponding bit combination of code B. A combinational circuit performs this transformation by means of logic gates.

For example, a binary –to-gray code converter has four binary input lines B_4, B_3,B_2,B_1 and four gray code output lines G_4,G_3,G_2,G_1. When the input is 0010, for instance, the output should be 0011 and so forth. To design a code converter, we use a code table treating it as a truth table to express each output as a Boolean algebraic function of all the inputs.

In this example, of binary –to-gray code conversion, we can treat the binary to the gray code table as four truth tables to derive expressions for G_4, G3, G2, and G1. Each of these four expressions would, in general, contain all the four input variables B_4, B3,B_2,and B1. Thus,this code converter is actually equivalent to four logic circuits, one for each of the truth tables.

The logic expression derived for the code converter can be simplified using the usual techniques, including _don_t cares_ if present. Even if the input is an unweighted code, the same cell numbering method which we used earlier can be used, but the cell numbers --must correspond to the input combinations as if they were an 8-4-2-1 weighted code. s

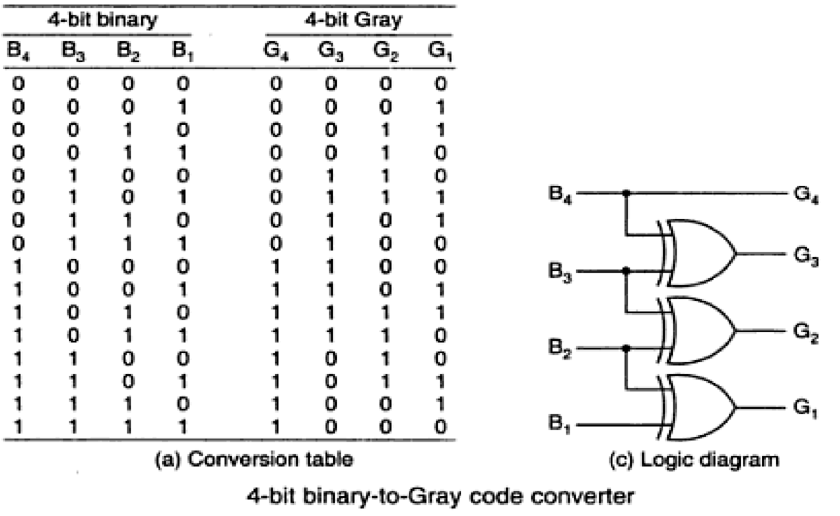
Design of a 4-bit binary to gray code converter:

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

$$G_3 = \Sigma m(4, 5, 6, 7, 8, 9, 10, 11) \qquad G_3 = \bar{B}_4 B_3 + B_4 \bar{B}_3 = B_4 \oplus B_3$$

$$G_2 = \Sigma m(2, 3, 4, 5, 10, 11, 12, 13) \qquad G_2 = \bar{B}_3 B_2 + B_3 \bar{B}_2 = B_3 \oplus B_2$$

$$G_1 = \Sigma m(1, 2, 5, 6, 9, 10, 13, 14) \qquad G_1 = \bar{B}_2 B_1 + B_2 \bar{B}_1 = B_2 \oplus B_1$$

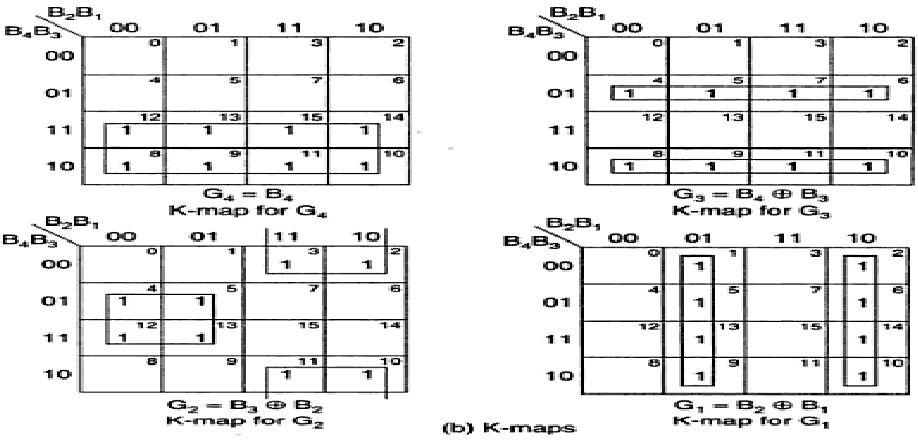


4-bit binary				4-bit Gray			
B_4	B_3	B_2	B_1	G_4	G_3	G_2	G_1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

(a) Conversion table

(c) Logic diagram

4-bit binary-to-Gray code converter



$G_4 = B_4$
K-map for G_4

$G_3 = B_4 \oplus B_3$
K-map for G_3

$G_2 = B_3 \oplus B_2$
K-map for G_2

$G_1 = B_2 \oplus B_1$
K-map for G_1

(b) K-maps

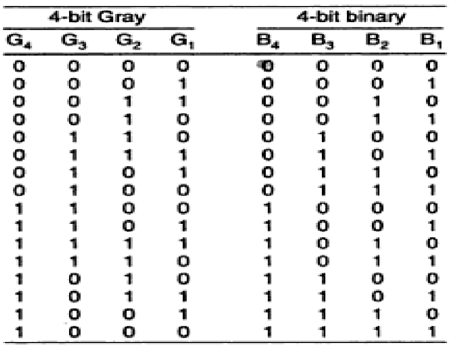
4-bit binary-to-Gray code converter.

Design of a 4-bit gray to Binary code converter:

$$B_4 = \Sigma\, m(12, 13, 15, 14, 10, 11, 9, 8) = \Sigma\, m(8, 9, 10, 11, 12, 13, 14, 15)$$
$$B_3 = \Sigma\, m(6, 7, 5, 4, 10, 11, 9, 8) = \Sigma\, m(4, 5, 6, 7, 8, 9, 10, 11)$$
$$B_2 = \Sigma\, m(3, 2, 5, 4, 15, 14, 9, 8) = \Sigma\, m(2, 3, 4, 5, 8, 9, 14, 15)$$
$$B_1 = \Sigma\, m(1, 2, 7, 4, 13, 14, 11, 8) = \Sigma\, m(1, 2, 4, 7, 8, 11, 13, 14)$$

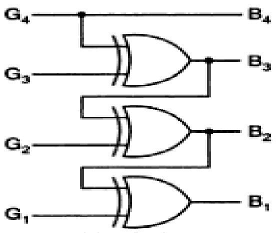
$$B_4 = G_4$$
$$B_3 = \bar{G}_4 G_3 + G_4 \bar{G}_3 = G_4 \oplus G_3$$
$$B_2 = \bar{G}_4 G_3 \bar{G}_2 + \bar{G}_4 \bar{G}_3 G_2 + G_4 \bar{G}_3 \bar{G}_2 + G_4 G_3 G_2$$
$$= \bar{G}_4 (G_3 \oplus G_2) + G_4 (\overline{G_3 \oplus G_2}) = G_4 \oplus G_3 \oplus G_2 = B_3 \oplus G_2$$
$$B_1 = \bar{G}_4 \bar{G}_3 \bar{G}_2 G_1 + \bar{G}_4 \bar{G}_3 G_2 \bar{G}_1 + \bar{G}_4 G_3 G_2 G_1 + \bar{G}_4 G_3 \bar{G}_2 \bar{G}_1 + G_4 G_3 \bar{G}_2 G_1$$
$$+ G_4 G_3 G_2 \bar{G}_1 + G_4 \bar{G}_3 G_2 G_1 + G_4 \bar{G}_3 \bar{G}_2 \bar{G}_1$$

$$= \bar{G}_4 \bar{G}_3 (G_2 \oplus G_1) + G_4 G_3 (G_2 \oplus G_1) + \bar{G}_4 G_3 (\overline{G_2 \oplus G_1}) + G_4 \bar{G}_3 (\overline{G_2 \oplus G_1})$$
$$= (G_2 \oplus G_1)(\overline{G_4 \oplus G_3}) + (\overline{G_2 \oplus G_1})(G_4 \oplus G_3)$$
$$= G_4 \oplus G_3 \oplus G_2 \oplus G_1$$

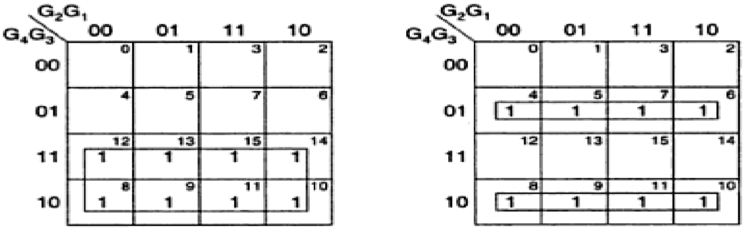


4-bit Gray				4-bit binary			
G_4	G_3	G_2	G_1	B_4	B_3	B_2	B_1
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

(a) Conversion table

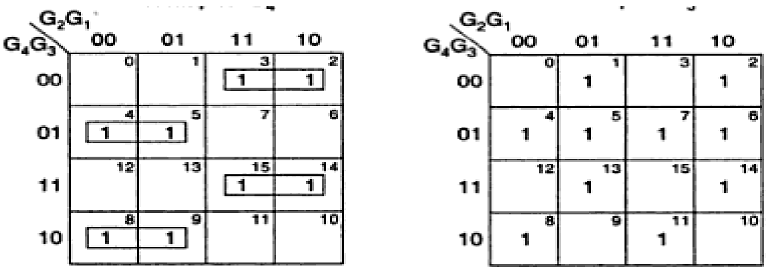


(c) Logic diagram



$B_4 = G_4$
K-map for B_4

$B_3 = G_4 \oplus G_3$
K-map for B_3



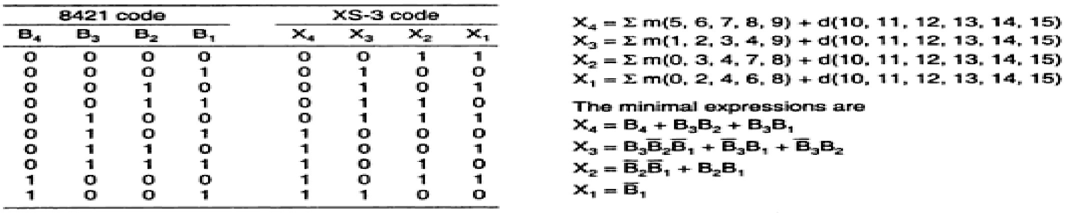
$B_2 = G_4 \oplus G_3 \oplus G_2$
K-map for B_2

(b) K-maps

$B_1 = G_4 \oplus G_3 \oplus G_2 \oplus G_1$
K-map for B_1

4-bit Gray-to-binary code converter.

Design of a 4-bit BCD to XS-3 code converter:



8421 code				XS-3 code			
B_4	B_3	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

$X_4 = \Sigma\, m(5, 6, 7, 8, 9) + d(10, 11, 12, 13, 14, 15)$
$X_3 = \Sigma\, m(1, 2, 3, 4, 9) + d(10, 11, 12, 13, 14, 15)$
$X_2 = \Sigma\, m(0, 3, 4, 7, 8) + d(10, 11, 12, 13, 14, 15)$
$X_1 = \Sigma\, m(0, 2, 4, 6, 8) + d(10, 11, 12, 13, 14, 15)$

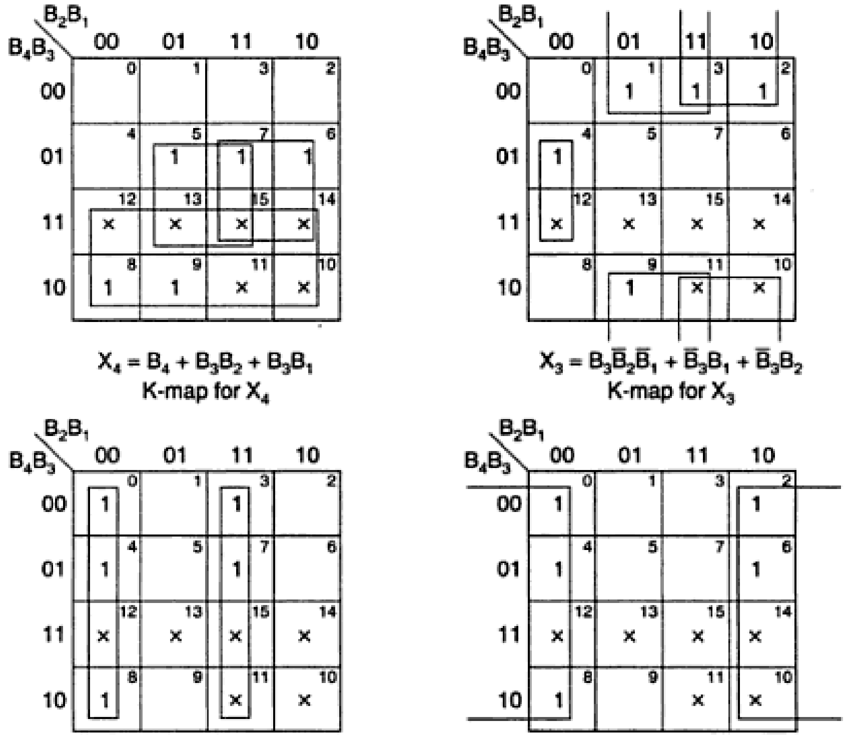
The minimal expressions are

$X_4 = B_4 + B_3B_2 + B_3B_1$
$X_3 = B_3\bar{B}_2\bar{B}_1 + \bar{B}_3B_1 + \bar{B}_3B_2$
$X_2 = \bar{B}_2\bar{B}_1 + B_2B_1$
$X_1 = \bar{B}_1$

(a) Conversion table

(b) Minimal expressions

4-bit BCD-to-XS-3 code converter



$X_4 = B_4 + B_3B_2 + B_3B_1$
K-map for X_4

$X_3 = B_3\bar{B}_2\bar{B}_1 + \bar{B}_3B_1 + \bar{B}_3B_2$
K-map for X_3

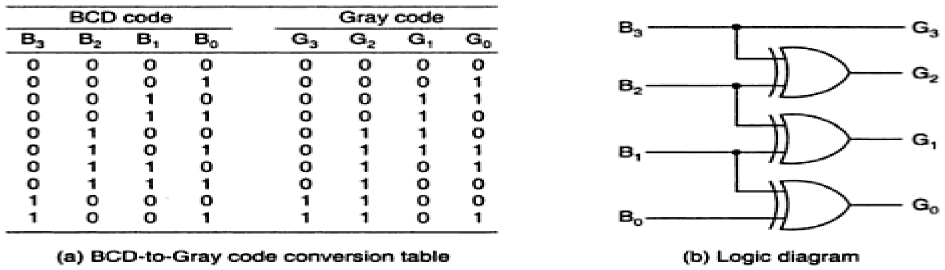
$X_2 = \bar{B}_2\bar{B}_1 + B_2B_1$
K-map for X_2

(c) K-maps

$X_1 = \bar{B}_1$
K-map for X_1

4-bit BCD-to-XS-3 code converter.

Design of aBCD to gray code converter:

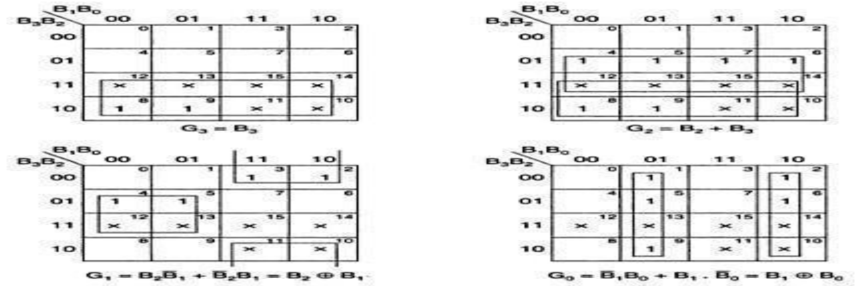


BCD code				Gray code			
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1

(a) BCD-to-Gray code conversion table

(b) Logic diagram

BCD-to-Gray code converter.



$G_3 = B_3$

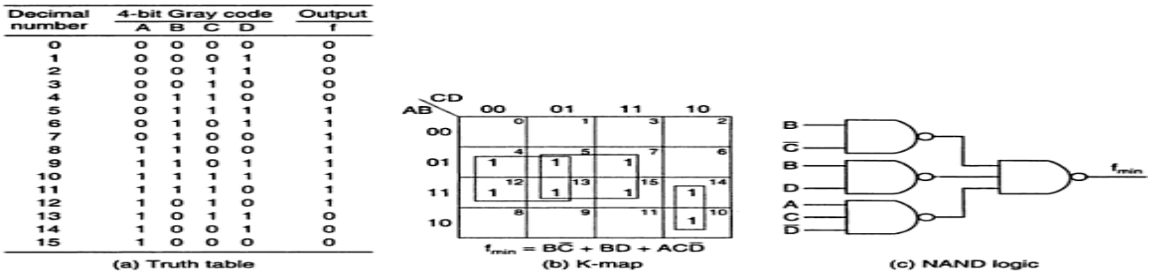
$G_2 = B_2 + B_3$

$G_1 = B_2\bar{B}_1 + \bar{B}_2B_1 = B_2 \oplus B_1$

$G_0 = B_1\bar{B}_0 + \bar{B}_1 . B_0 = B_1 \oplus B_0$

K-maps for a BCD-to-Gray code converter.

DesignofaSOPcircuittoDetecttheDecimalnumbers5through12ina4-bitgraycode Input:



Decimal number	4-bit Gray code				Output
	A	B	C	D	f
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	1	0
3	0	0	1	0	0
4	0	1	1	0	0
5	0	1	1	1	1
6	0	1	0	1	1
7	0	1	0	0	1
8	1	1	0	0	1
9	1	1	0	1	1
10	1	1	1	1	1
11	1	1	1	0	1
12	1	0	1	0	1
13	1	0	1	1	0
14	1	0	0	1	0
15	1	0	0	0	0

(a) Truth table

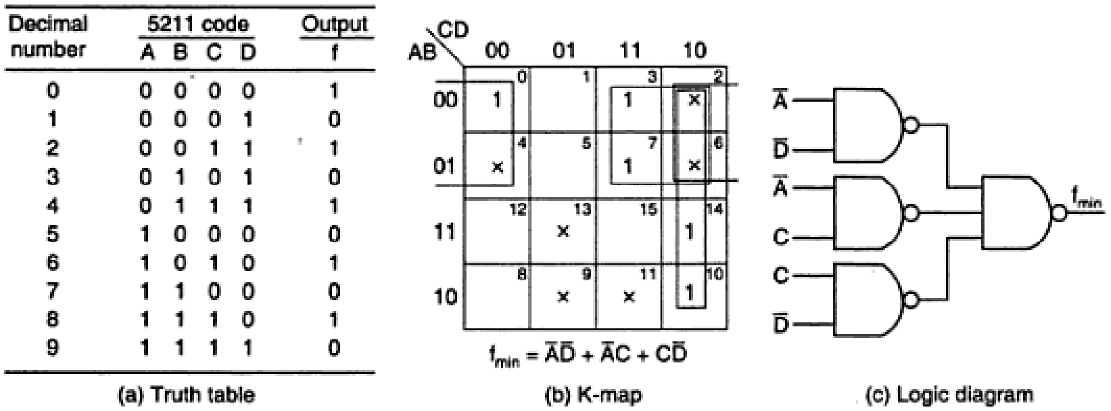
$f_{min} = B\bar{C} + BD + AC\bar{D}$

(b) K-map

(c) NAND logic

Truth table, K-map and logic diagram for the SOP circuit.

Design of a SOP circuit to detect the decimal numbers 0,2,4,6,8 in a 4-bit 5211 BCD code input:



Decimal number	5211 code				Output
	A	B	C	D	f
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	1	1
3	0	1	0	1	0
4	0	1	1	1	1
5	1	0	0	0	0
6	1	0	1	0	1
7	1	1	0	0	0
8	1	1	1	0	1
9	1	1	1	1	0

(a) Truth table

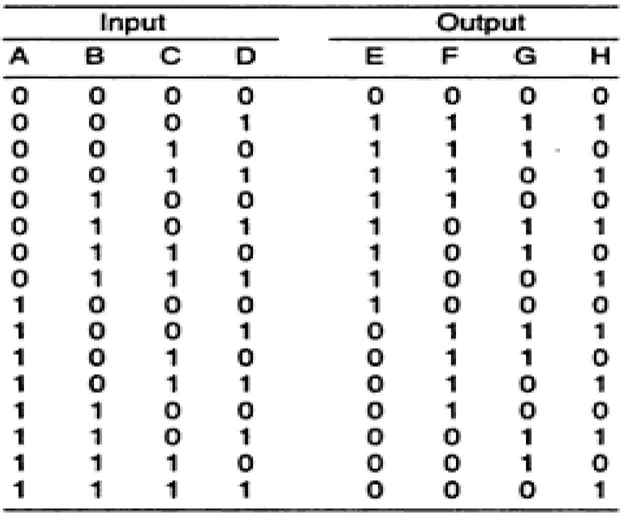
$f_{min} = \bar{A}\bar{D} + \bar{A}C + C\bar{D}$

(b) K-map

(c) Logic diagram

Truth table, K-map and logic diagram for the SOP circuit.

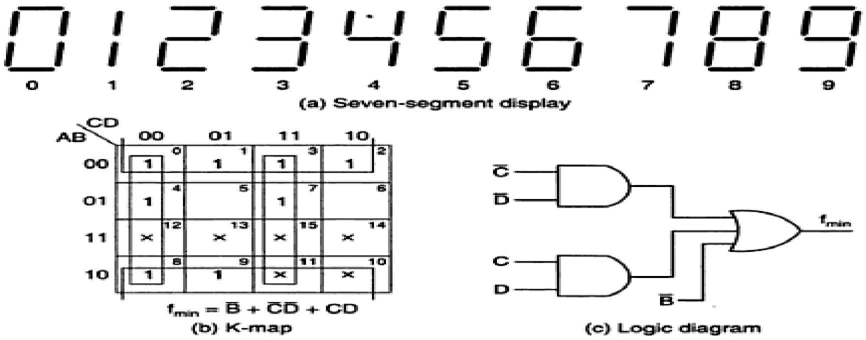
DesignofaCombinationalcircuittoproducethe*2’s*complementofa4-bitbinarynumber:



Input				Output			
A	B	C	D	E	F	G	H
0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	1
0	0	1	0	1	1	1	0
0	0	1	1	1	1	0	1
0	1	0	0	1	1	0	0
0	1	0	1	1	0	1	1
0	1	1	0	1	0	1	0
0	1	1	1	1	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	0	1	1	1
1	0	1	0	0	1	1	0
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	0
1	1	0	1	0	0	1	1
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

(a) Conversion table

Conversion table and K-maps for the circuit



(a) Seven-segment display

AB \ CD	00	01	11	10
00	1	1	1	1
01	1		1	
11	×	×	×	×
10	1	1	×	×

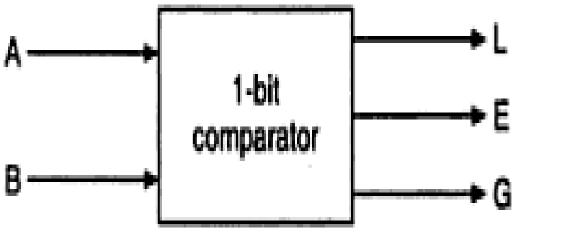
$f_{min} = \bar{B} + \bar{C}\bar{D} + CD$

(b) K-map

(c) Logic diagram

Comparators:

$$\text{EQUALITY} = (A_3 \odot B_3)(A_2 \odot B_2)(A_1 \odot B_1)(A_0 \odot B_0)$$



Block diagram of a 1-bit comparator.

The logic for a 1-bit magnitude comparator: Let the 1-bit numbers be $A = A_0$ and $B = B_0$.
If $A_0 = 1$ and $B_0 = 0$, then $A > B$.
Therefore,

$$A > B: G = A_0\overline{B}_0$$

If $A_0 = 0$ and $B_0 = 1$, then $A < B$.
Therefore,

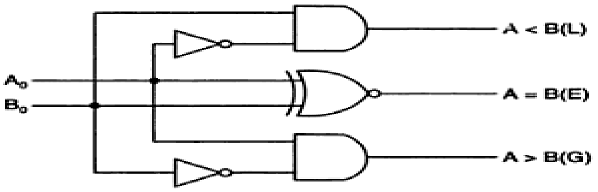
$$A < B: L = \overline{A}_0B_0$$

If A_0 and B_0 coincide, i.e. $A_0 = B_0 = 0$ or if $A_0 = B_0 = 1$, then $A = B$.
Therefore,

$$A = B : E = A_0 \odot B_0$$

A_0	B_0	L	E	G
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

(a) Truth table



(b) Logic diagram

1-bit comparator.

1. Magnitude Comparator:

1- bit Magnitude Comparator:

The logic for a 2-bit magnitude comparator: Let the two 2-bit numbers be $A = A_1 A_0$ and $B = B_1 B_0$.
1. If $A_1 = 1$ and $B_1 = 0$, then $A > B$ or
2. If A_1 and B_1 coincide and $A_0 = 1$ and $B_0 = 0$, then $A > B$. So the logic expression for $A > B$ is

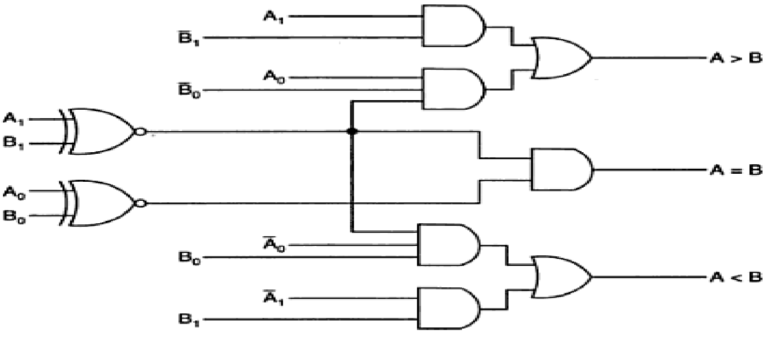
$$A > B : G = A_1\overline{B}_1 + (A_1 \odot B_1)A_0\overline{B}_0$$

1. If $A_1 = 0$ and $B_1 = 1$, then $A < B$ or
2. If A_1 and B_1 coincide and $A_0 = 0$ and $B_0 = 1$, then $A < B$. So the expression for $A < B$ is

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

If A_1 and B_1 coincide and if A_0 and B_0 coincide then $A = B$. So the expression for $A = B$ is

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



Logic diagram of a 2-bit magnitude comparator.

4- Bit Magnitude Comparator:

The logic for a 4-bit magnitude comparator: Let the two 4-bit numbers be $A = A_3A_2A_1A_0$ and $B = B_3B_2B_1B_0$.

1. If $A_3 = 1$ and $B_3 = 0$, then $A > B$. Or
2. If A_3 and B_3 coincide, and if $A_2 = 1$ and $B_2 = 0$, then $A > B$. Or
3. If A_3 and B_3 coincide, and if A_2 and B_2 coincide, and if $A_1 = 1$ and $B_1 = 0$, then $A > B$. Or
4. If A_3 and B_3 coincide, and if A_2 and B_2 coincide, and if A_1 and B_1 coincide, and if $A_0 = 1$ and $B_0 = 0$, then $A > B$.

From these statements, we see that the logic expression for $A > B$ can be written as

$$(A > B) = A_3\overline{B}_3 + (A_3 \odot B_3)A_2\overline{B}_2 + (A_3 \odot B_3)(A_2 \odot B_2)A_1\overline{B}_1 + (A_3 \odot B_3)(A_2 \odot B_2)(A_1 \odot B_1)A_0\overline{B}_0$$

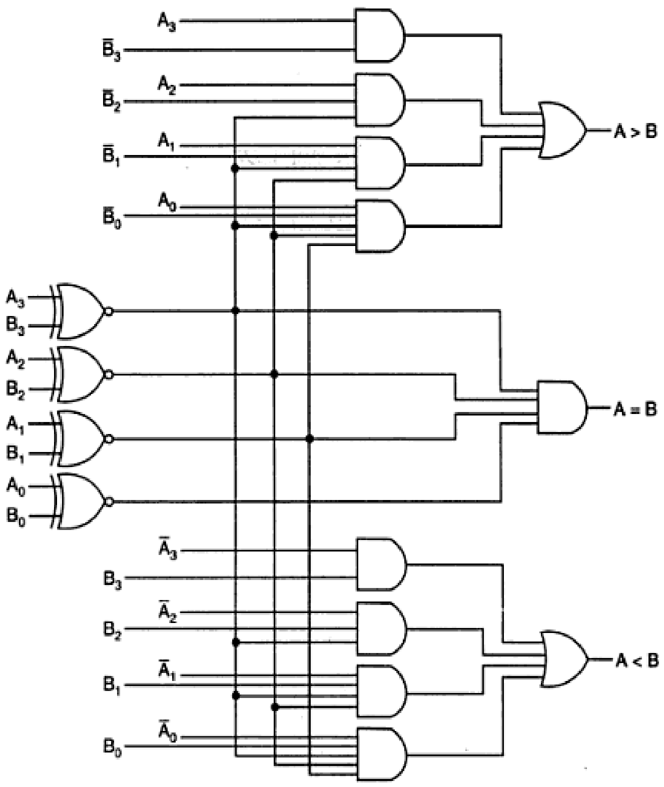
Similarly, the logic expression for $A < B$ can be written as

$$A < B = \overline{A}_3B_3 + (A_3 \odot B_3)\overline{A}_2B_2 + (A_3 \odot B_3)(A_2 \odot B_2)\overline{A}_1B_1 + (A_3 \odot B_3)(A_2 \odot B_2)(A_1 \odot B_1)\overline{A}_0B_0$$

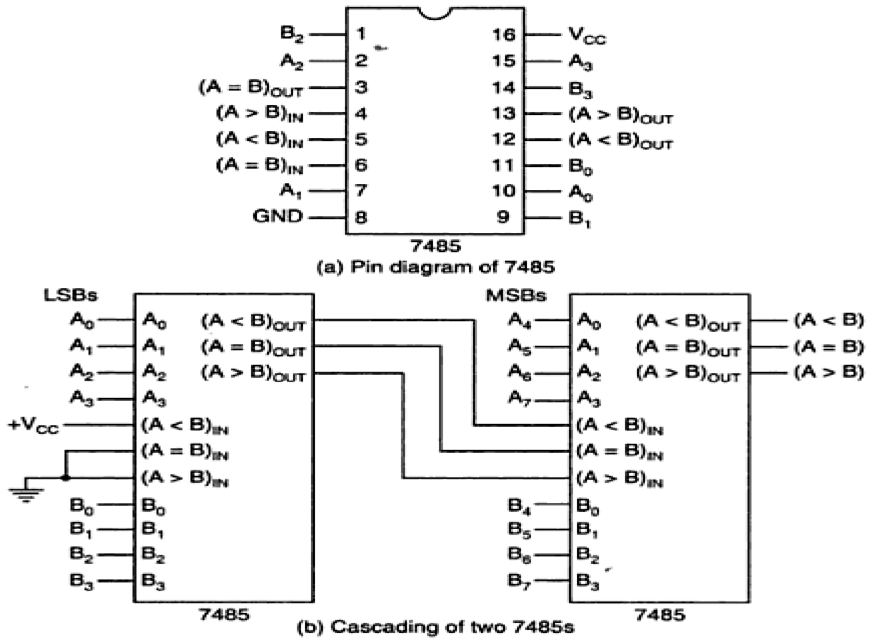
If A_3 and B_3 coincide and if A_2 and B_2 coincide and if A_1 and B_1 coincide and if A_0 and B_0 coincide, then $A = B$.

So the expression for $A = B$ can be written as

$$(A = B) = (A_3 \odot B_3)(A_2 \odot B_2)(A_1 \odot B_1)(A_0 \odot B_0)$$



IC Comparator:

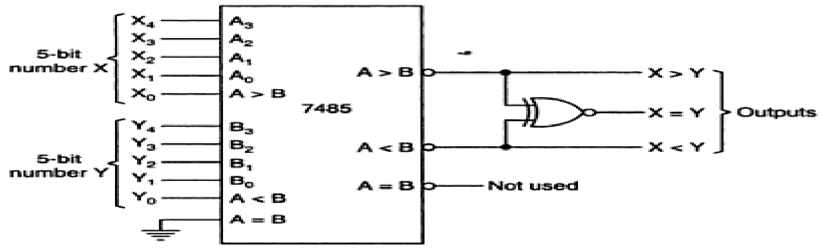


(a) Pin diagram of 7485

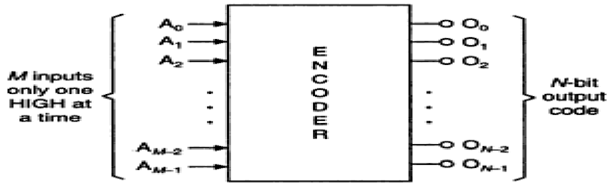
(b) Cascading of two 7485s

Pin diagram and cascading of 7485 4-bit comparators.

ENCODERS:



Use of 7485 as a 5-bit comparator.

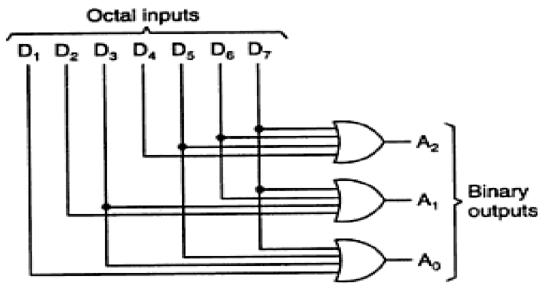


Block diagram of encoder.

Octal to Binary 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

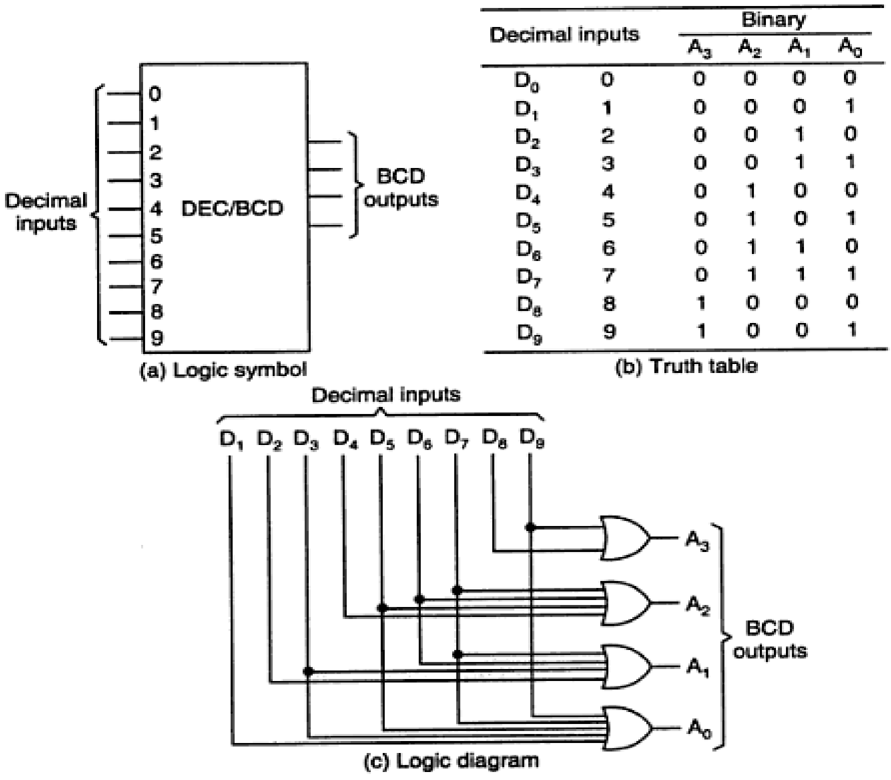
(a) Truth table



(b) Logic diagram

Octal-to-binary encoder.

Decimal to BCD Encoder:



(a) Logic symbol

Decimal inputs		Binary			
		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

(b) Truth table

(c) Logic diagram

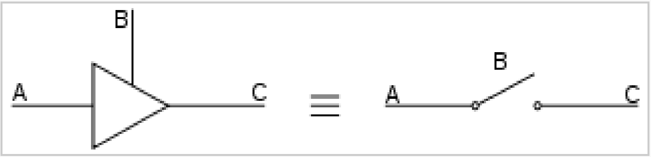
Decimal-to-BCD encoder.

Tristate bus system:

In digital electronics three-state, tri-state, or 3-statelogic allows an output port to assume a high impedance state in addition to the 0 and 1 logic levels, effectively removing the output from the circuit.

This allows multiple circuits to share the same output line or lines (such as a bus which cannot listen to more than one device at a time).

Three-state outputs are implemented in many registers, bus drivers, and flip-flops in the 7400 and 4000 series as well as in other types, but also internally in many integrated circuits. Other typical uses are internal and external buses in microprocessors, computer memory, and peripherals. Many devices are controlled by an active-low input called OE (Output Enable) which dictates whether the outputs should be held in a high-impedance state or drive their respective loads (to either 0- or 1-level).



A tristate buffer can be thought of as a switch. If *B* is on, the switch is closed. If B is off, the switch is open.

INPUT		OUTPUT
A	B	C
0	1	0
1		1
X	0	Z (high impedance)

Unit-IV

Feedback Amplifiers

A practical amplifier has a gain of nearly one million *i.e.* its output is one million times the input. Consequently, even a casual disturbance at the input will appear in the amplified form l in its output. The noise in the output of an amplifier is undesirable and must be kept to as small a level as possible. The noise level in amplifiers can be reduced considerably by the use of *negative feedback i.e.* by injecting a fraction of output in phase opposition to the input signal. The object of this chapter is to consider the effects and methods of providing negative feedback in transistor amplifiers.

Ideally an amplifier should reproduce the input signal, with change in magnitude and with or without change in phase. But some of the short comings of the amplifier circuit are

I. Change in the value of the gain due to variation in supplying voltage, temperature or due to components.

2. Distortion in wave-form due to non linearities in the operating characters of the Amplifying device.

3. The amplifier may introduce noise (undesired signals)

The above drawbacks can be minimizing if we introduce feedback

CLASSIFICATION OF AMPLIFIERS

Amplifiers can be classified broadly as,

I. Voltage amplifiers.

2. Current amplifiers.

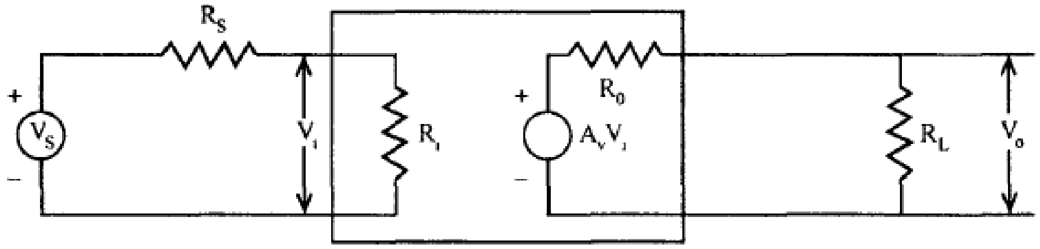
3. Trans conductance amplifiers.

4. Tran sresistance amplifiers.

This classification is with respect to the input and output impedances relative to the load and source impedances.

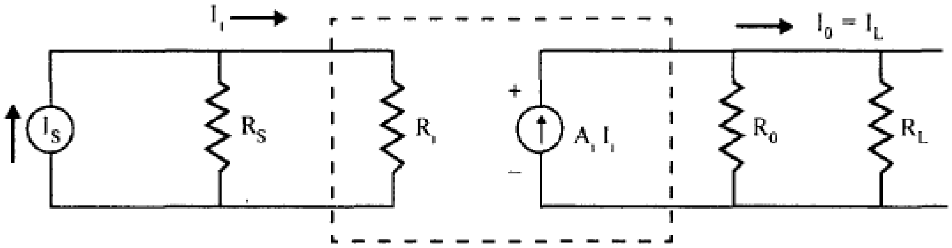
VOLTAGE AMPLIFIER

This circuit is a 2-port network and it represents an amplifier (see in Fig 7.1). Suppose R,» Rs, drop across Rs is very small.



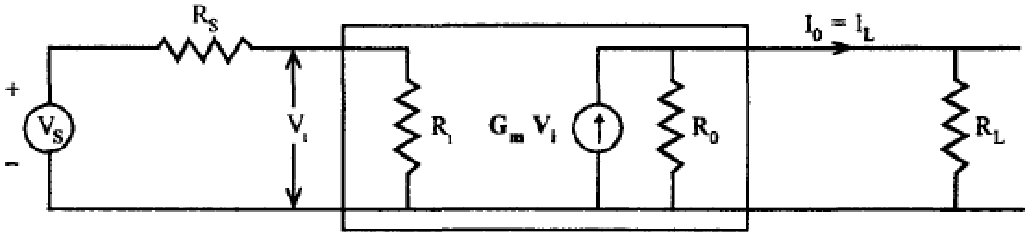
CURRENT AMPLIFIER

An ideal current amplifier is one which gives output current proportional to input current and the proportionality factor is independent ofRs and RL.



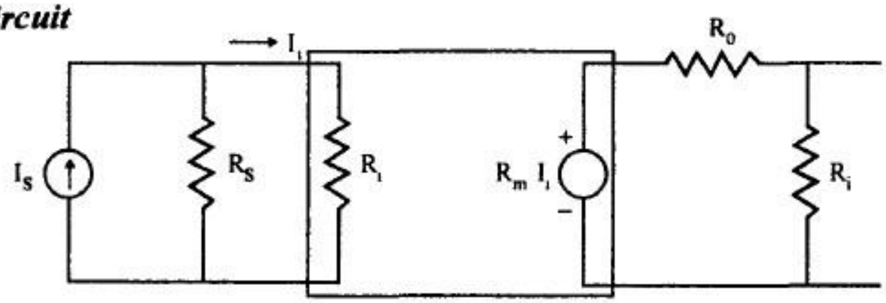
TRANSCONDUCTANCE AMPLIFIER

Ideal Transconductance amplifier supplies output current which is proportional to input voitage independently ofthe magnitude ofRs and RL.



TRANS RESISTANCE AMPLIFIER
It gives output voltage Vo proportional to Is, independent of Rs a. RL. For *ideal amplifiers*
Rj =0, Ro=O

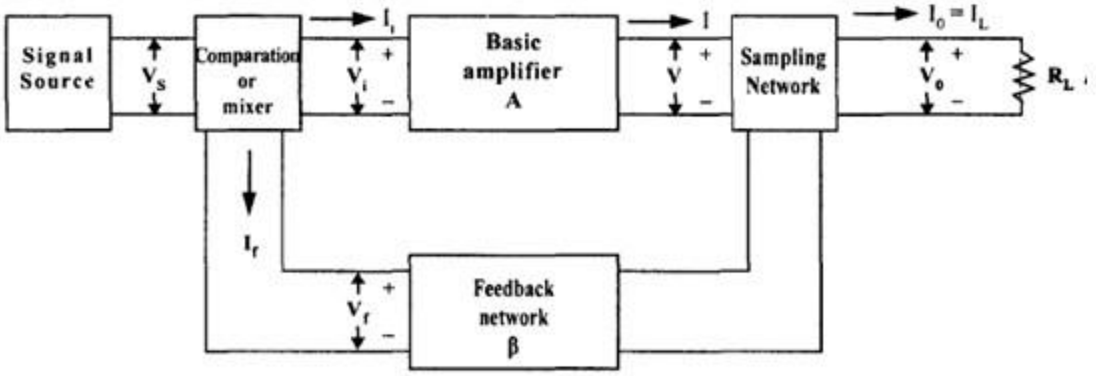
Equivalent circuit



Concepts of feedback

The process of injecting a fraction of output energy of some device back to the input is known as **feedback**. The principle of feedback is probably as old as the invention of first machine but it is only some 50 years ago that feedback has come into use in connection with electronic circuits. It has been found very useful in reducing noise in amplifiers and making amplifier operation stable. Depending upon whether the feedback energy aids or opposes the input signal, there are two basic types of feedback in amplifiers *viz positive feedback* and *negative feedback*.

GENERALIZED BLOCK SCHEMATIC



Signal Source

It can be a voltage source V s or a current source Is

FEEDBACK NETWORK

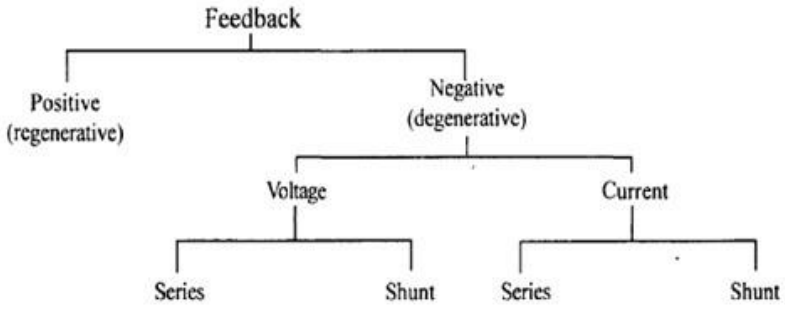
It is a passive two port network. It may contain resistors, capacitors or inductors. But usually a resistance is used as the feedback element. Here the output current is sampled and feedback. The feedback network is connected in series with the output. This is called as *Current Sampling or* Loop Sampling.

A voltage feedback is distinguished in this way from current feedback. For voltage feedback, the feedback element (resistor) will be in parallel with the output. For current feedback the element will be in series.

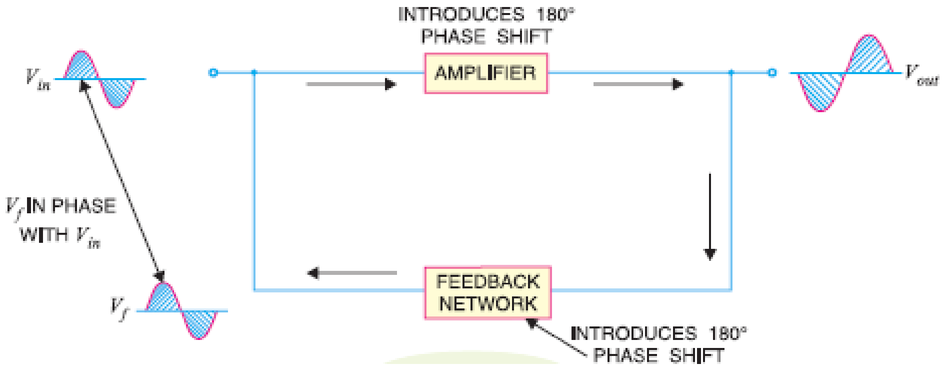
COMPARATOR OR MIXER NETWORK

This is usually a differential amplifier. It has two inputs and gives a single output which is the difference of the two inputs.

basic types of feedback in amplifiers

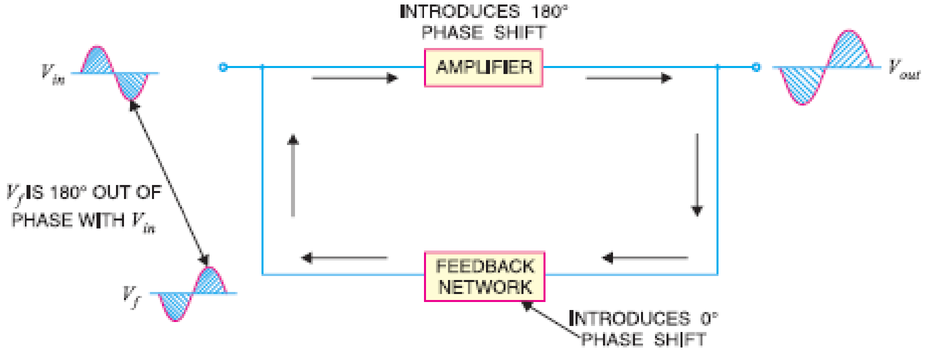


(i) Positive feedback. When the feedback energy (voltage or current) is in phase with the input signal and thus aids it, it is called *positive feedback*. This is illustrated in Fig.. Both amplifier and feedback network introduce a phase shift of 180°. The result is a 360° phase shift around the loop, causing the *feedback voltage Vf* to be in phase with the input signal *Vin*.


INTRODUCES 180° PHASE SHIFT
AMPLIFIER
V_{in}
V_{out}
V_f IN PHASE WITH V_{in}
V_f
FEEDBACK NETWORK
INTRODUCES 180° PHASE SHIFT

The positive feedback increases the gain of the amplifier. However, it has the disadvantages of increased distortion and instability. Therefore, positive feedback is seldom employed in amplifiers. One important use of positive feedback is in oscillators. As we shall see in the next chapter, if positive feedback is sufficiently large, it leads to oscillations. As a matter of fact, an oscillator is a device that converts d.c. power into a.c. power of any desired frequency.

(ii) Negative feedback. When the feedback energy (voltage or current) is out of phase with the input signal and thus opposes it, it is called *negative feedback*. This is illustrated in Fig.. As you can see, the amplifier introduces a phase shift of 180° into the circuit while the feedback network is so designed that it introduces no phase shift (*i.e.*, 0° phase shift). The result is that the *feedback voltage Vf* is 180° out of phase with the input signal *Vin*.


INTRODUCES 180° PHASE SHIFT
AMPLIFIER
V_{in}
V_{out}
V_f IS 180° OUT OF PHASE WITH V_{in}
V_f
FEEDBACK NETWORK
INTRODUCES 0° PHASE SHIFT

General characteristics of negative feedback amplifiers

Negative feedback reduces the gain of the amplifier. However, the advantages of negative feedback are: reduction in distortion, stability in gain, increased bandwidth and improved input and output impedances. It is due to these advantages that negative feedback is frequently employed in amplifiers.

Advantages of Negative Voltage Feedback

The following are the advantages of negative voltage feedback in amplifiers :

(i) Gain stability. An important advantage of negative voltage feedback is that the resultant gain of the amplifier can be made independent of transistor parameters or the supply voltage variations.

$$A_{vf} = \frac{A_v}{1 + A_v\, m_v}$$

For negative voltage feedback in an amplifier to be effective, the designer deliberately makes the product *Av mv* much greater than unity. Therefore, in the above relation, 1 can be neglected as compared to *Av mv* and the expression becomes :

$$A_{vf} = \frac{A_v}{A_v\, m_v} = \frac{1}{m_v}$$

It may be seen that the gain now depends only upon feedback fraction *mv i.e.*, on the characteristics of feedback circuit. As feedback circuit is usually a voltage divider (a resistive network), therefore, it is unaffected by changes in temperature, variations in transistor parameters and frequency. Hence, the gain of the amplifier is extremely stable.

(ii) Reduces non-linear distortion. A large signal stage has non-linear distortion because its voltage gain changes at various points in the cycle. The negative voltage feedback reduces the nonlinear distortion in large signal amplifiers. It can be proved mathematically that :

$$D_{vf} = \frac{D}{1 + A_v\, m_v}$$

where *D* = distortion in amplifier without feedback
Dvf = distortion in amplifier with negative feedback

It is clear that by applying negative voltage feedback to an amplifier, distortion is reduced by a factor 1 + *Av mv*.

(iii) Improves frequency response. As feedback is usually obtained through a resistive network, therefore, voltage gain of the amplifier is *independent of signal frequency. The result is that voltage gain of the amplifier will be substantially constant over a wide range of signal frequency. The negative voltage feedback, therefore, improves the frequency response of the amplifier.

(iv) Increases circuit stability. The output of an ordinary amplifier is easily changed due to variations in ambient temperature, frequency and signal amplitude. This changes the gain of the amplifier, resulting in distortion. However, by applying negative voltage feedback, voltage gain of the amplifier is stabilized or accurately fixed in value. This can be easily explained. Suppose the output of a negative voltage feedback amplifier has increased because of temperature change or due to some other reason. This means more negative feedback since feedback is being given from the output. This tends to oppose the increase in amplification and maintains it stable. The same is true should the output voltage decrease. Consequently, the circuit stability is considerably increased.

(v) Increases input impedance and decreases output impedance. The negative voltage feedback increases the input impedance and decreases the output impedance of amplifier. Such a change is profitable in practice as the amplifier can then serve the purpose of impedance matching.

(a) Input impedance. The increase in input impedance with negative voltage feedback can be explained by referring to Fig.. Suppose the input impedance of the amplifier is *Zin* without feedback and *Z 'in* with negative feedback. Let us further assume that input current is *i*1.

$$Z'_{in} = Z_{in}(1 + A_v\, m_v)$$

(b) Output impedance. Following similar line, we can show that output impedance with negative voltage feedback is given by :

$$Z'_{out} = \frac{Z_{out}}{1 + A_v\, m_v}$$

It is clear that by applying negative feedback, the output impedance of the amplifier is decreased by a factor 1 + *Av mv*. This is an added benefit of using negative voltage feedback. With lower value of output impedance, the amplifier is much better suited to drive low impedance load

Effect of feedback on amplifier characteristics

CLASSIFACTION OF FEEDBACK AMPLIFIERS

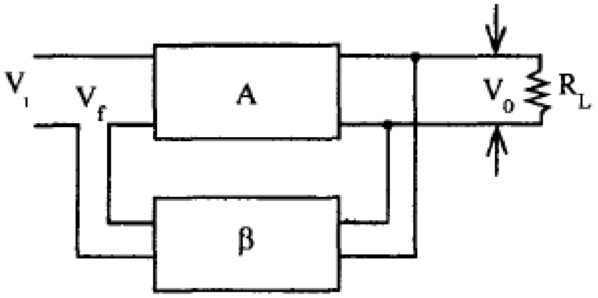
There are four types of feedback,

1. Voltage series feedback.
2. Voltage shunt feedback.
3. Current shunt feedback.
4. Current series feedback.

If the feedback signal is taken across RL, it is a Vo or so it is *Voltage feedback.*
If the feedback signal is taken in series with the output terminals, feedback signal is proportional to I0 , So it is *current feedback.*
If the feedback signal is in series with the input, it is *seriesfeedback.*
If the feedback signal is in shunt with the input, it is *shuntfeedback.*

Voltage Series Configuration

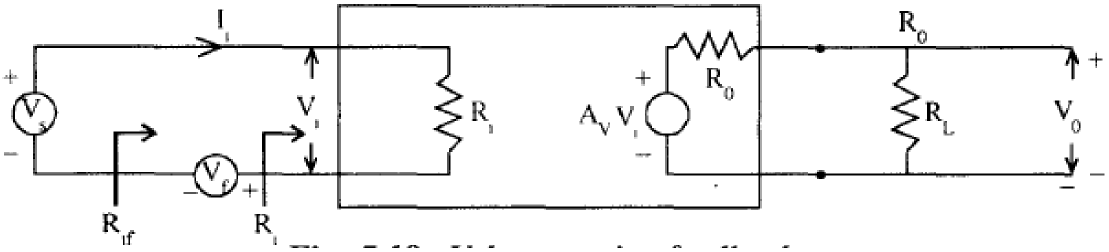
Feedback signal is taken across RL.proportional to Vo. So it is voltage feedback. Vf is coming in series with VI So it is Voltage series feedback.



Voltage series feedback.

EXPRESSION FOR INPUT RESISTANCE R_I WITH VOLTAGE SERIES FEEDBACK

In this circuit Av represents the open circuit voltage gain taking Rs into account



Voltage series feedback

$$V_i = V_S - V_f$$
$$V_s = V_i + V_f$$
$$R_0' = R_{if} = V_S/I_i.$$
$$V_f = I_i\, R_i + V_f = I_i\, R_i + \beta V_0$$
$$V_0 = \frac{A_v V_i R_L}{R_o + R_L} = (A_V\, I_i\, R_i)$$
$$\frac{A_v.R_L}{R_o + R_L} = A_V,\ V_i = I_i.\ R_i.$$

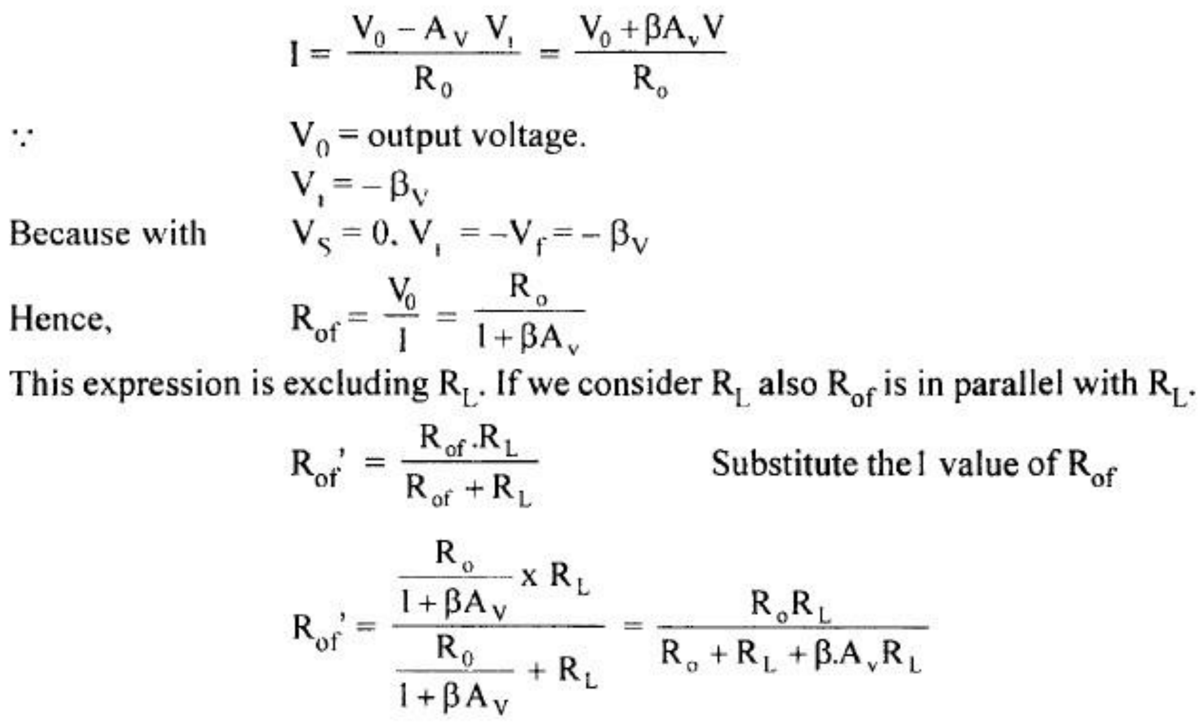
$$A_V = \frac{V_o}{V_i} = \frac{A_v R_L}{R_o + R_L}$$
$$V_S = I_i\, R_i + \beta\ .\ V_0$$
$$\boxed{\frac{V_s}{I_i} = R_i + \frac{\beta.V_o}{I_i} = R_i\left(1 + \frac{\beta.V_o}{I_i \times R_i}\right)}$$
$$R_{if} = R_i\left(1 + \beta.\frac{V_o}{V_i}\right) = R_i(1 + \beta.A_V)$$
$$A_V = \text{voltage gain, without feedback.}$$

EXPRESSION FOR OUTPUT RESISTANCE R_o WITH VOLTAGE SERIES FEEDBACK

R_O is determined by impressing voltage 'V' at the output terminals or messing 'I', with input R_{of} terminals.-shorted.

Disconnect R_{oL} To find R_{of}' remove external signal (set Vs = 0, or Is = 0)
Let $R_L = \infty$
Impress a voltage V across the output terminals and calculate the current I delivered by V.

Then, $R_{Of} = V/I$.



$$I = \frac{V_0 - A_V\ V_i}{R_0} = \frac{V_0 + \beta A_v V}{R_o}$$

∵ V_0 = output voltage.

$V_i = -\beta_V$

Because with $V_S = 0.\ V_i = -V_f = -\beta_V$

Hence, $R_{of} = \frac{V_0}{I} = \frac{R_o}{1+\beta A_v}$

This expression is excluding R_L. If we consider R_L also R_{of} is in parallel with R_L.

$$R_{of}' = \frac{R_{of}.R_L}{R_{of}+R_L}$$ Substitute the1 value of R_{of}

$$R_{of}' = \frac{\frac{R_o}{1+\beta A_V} x\ R_L}{\frac{R_0}{1+\beta A_V} + R_L} = \frac{R_o R_L}{R_o + R_L + \beta.A_v R_L}$$

UNIT – V

SINGLE STAGE AMPLIFIERS AND MULTISTAGE AMPLIFIERS

INTRODUCTION

An audio amplifier amplifies a wide band of frequencies equally well and does not permit the selection of a particular desired frequency while rejecting all other frequencies. However, sometimes it is desired that an amplifier should be selective *i.e.* it should select a desired frequency or narrow band of frequencies for amplification. For instance, radio and television transmission are carried on a specific radio frequency assigned to the broadcasting station. The radio receiver is required to pick up and amplify the radio frequency desired while discriminating all others. To achieve this, the simple resistive load is replaced by a parallel tuned circuit whose impedance strongly depends upon frequency. Such a tuned circuit becomes very selective and amplifies very strongly signals of resonant frequency and narrow band on either side. Therefore, the use of tuned circuits in conjunction with a transistor makes possible the selection and efficient amplification of a particular desired radio frequency. Such an amplifier is called a *tuned amplifier*. In this chapter, we shall focus our attention on transistor tuned amplifiers and their increasing applications in high frequency electronic circuits.

Advantages of Tuned Amplifiers

In high frequency applications, it is generally required to amplify a single frequency, rejecting all other frequencies present. For such purposes, tuned amplifiers are used. These amplifiers use tuned parallel circuit as the collector load and offer the following advantages :

(i) Small power loss. A tuned parallel circuit employs reactive components *L* and *C*. Consequently, the power loss in such a circuit is quite low. On the other hand, if a resistive load is used in the collector circuit, there will be considerable loss of power. Therefore, tuned amplifiers are highly efficient.

(ii) High selectivity. A tuned circuit has the property of selectivity *i.e.* it can select the desired frequency for amplification out of a large number of frequencies simultaneously impressed upon it. For instance, if a mixture of frequencies including *fr* is fed to the input of a tuned amplifier, then maximum amplification occurs for *fr*. For all other frequencies, the tuned circuit offers very low impedance and hence these are amplified to a little extent and may be thought as rejected by the circuit. On the other hand, if we use resistive load in the collector, all the frequencies will be amplified equally well *i.e.* the circuit will not have the ability to select the desired frequency.

(iii) Smaller collector supply voltage. Because of little resistance in the parallel tuned circuit, it requires small collector supply voltage *VCC*. On the other hand, if a high load resistance is used in the collector for amplifying even one frequency, it would mean large voltage drop across it due to zero signal collector current. Consequently, a higher collector supply will be needed.

Why not Tuned Circuits for Low Frequency Amplification?

The tuned amplifiers are used to select and amplify a specific high frequency or narrow band of frequencies. The reader may be inclined to think as to why tuned circuits are not used to amplify low frequencies. This is due to the following reasons :

(i) *Low frequencies are never single.* A tuned amplifier selects and amplifies a single frequency. However, the low frequencies found in practice are the audio frequencies which are a mixture of frequencies from 20 Hz to 20 kHz and are not single. It is desired that all these frequencies

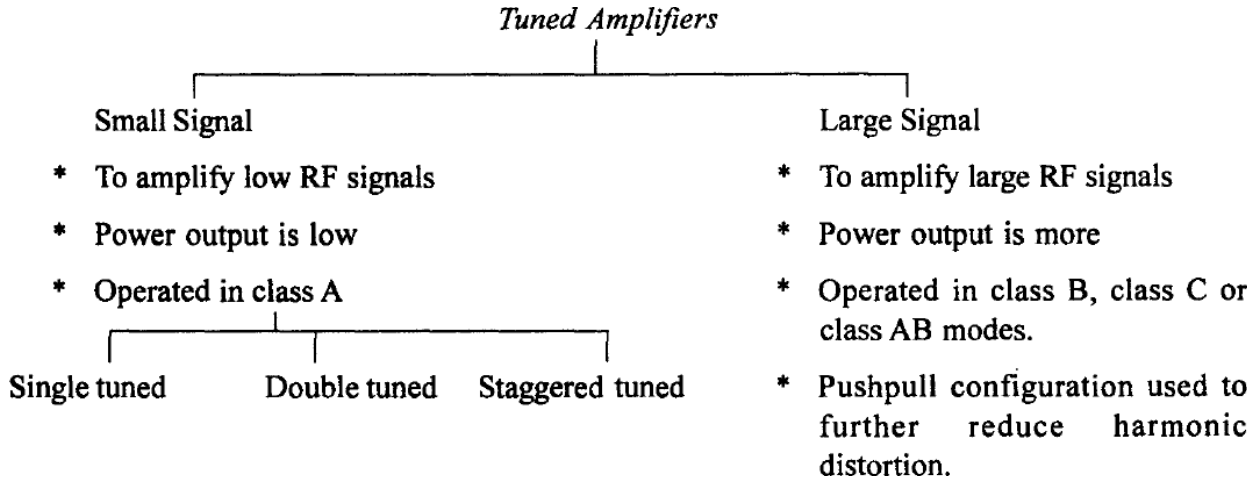
should be equally amplified for proper reproduction of the signal. Consequently, tuned amplifiers cannot be used for the purpose.

(ii) *High values of L and C.* The resonant frequency of a parallel tuned circuit is given by;

$fr = 1/2\pi LC$

For low frequency amplification, we require large values of *L* and *C*. This will make the tuned Circuit bulky and expensive. It is worthwhile to mention here that *R-C* and transformer coupled Amplifiers, which are comparatively cheap, can be conveniently used for low frequency applications.

Classification



Tuned Amplifiers

Small Signal

- To amplify low RF signals
- Power output is low
- Operated in class A

Single tuned Double tuned Staggered tuned

Large Signal

- To amplify large RF signals
- Power output is more
- Operated in class B, class C or class AB modes.
- Pushpull configuration used to further reduce harmonic distortion.

Tuned Amplifiers

Amplifiers which amplify a specific frequency or narrow band of frequencies are called tuned amplifiers. Tuned amplifiers are mostly used for the amplification of high or radio frequencies.

It is because radio frequencies are generally single and the tuned circuit permits their selection and efficient amplification. However, such amplifiers are not suitable for the amplification of audio frequencies as they are mixture of frequencies from 20 Hz to 20 kHz and not single.

Tuned amplifiers are widely used in radio and television circuits where they are called upon to handle radio frequencies. Fig. 5.1 shows the circuit of a simple transistor tuned amplifier. Here, instead of load resistor, we have a parallel tuned circuit in the collector. The impedance

of this tuned circuit strongly depends upon frequency. It offers a very high impedance at *resonant frequency* and very small impedance at all other frequencies. If the signal has the same frequency as the resonant frequency of *LC* circuit, large amplification will result due to high impedance of *LC* circuit at this frequency. When signals of many frequencies are present at the input of tuned amplifier, it will select and strongly amplify the signals of resonant frequency while

*rejecting all others. Therefore, such amplifiers are very useful in radio receivers to select the signal from one particular broadcasting station when signals of many other frequencies are present at the receiving aerial.

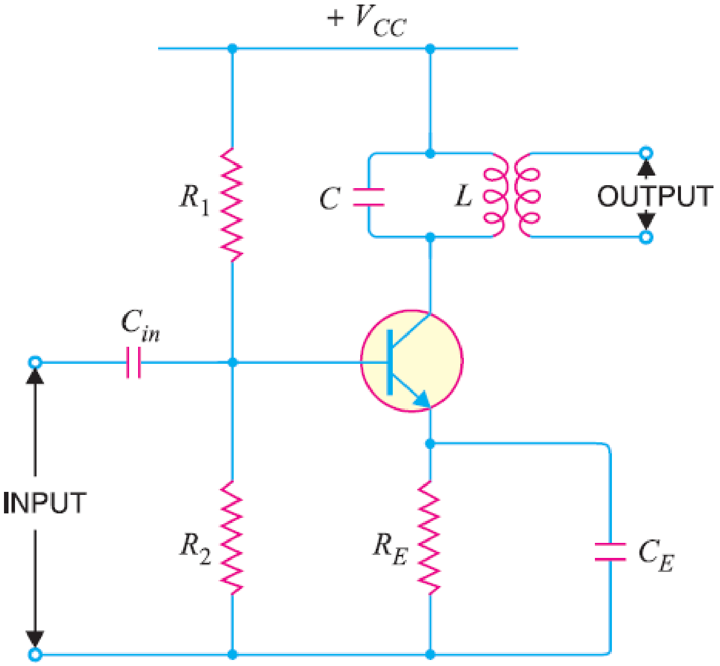


Fig.5.1

Distinction between Tuned Amplifiers and other Amplifiers

We have seen that amplifiers (*e.g.*, voltage amplifier, power amplifier *etc.*) provide the constant gain over a limited band of frequencies *i.e.*, from lower cut-off frequency *f*1 to upper cut-off frequency *f*2. Now bandwidth of the amplifier, *BW* = *f*2 − *f*1. The reader may wonder, then, what distinguishes atuned amplifier from other amplifiers? The difference is that tuned amplifiers are designed to have specific, usually narrow bandwidth. This point is illustrated in in Fig.5.2. Note that *BWS* is the bandwidth of standard frequency response while *BWT* is the bandwidth of the tuned amplifier. In many applications, the narrower the bandwidth of a tuned amplifier, the better it is.

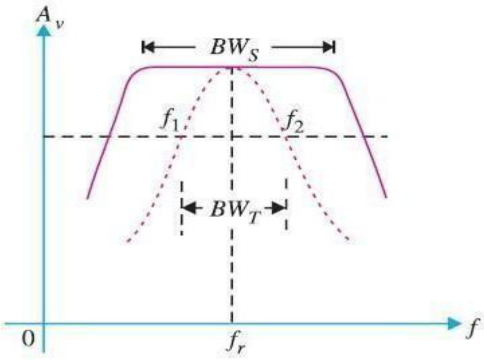


Fig.5.2

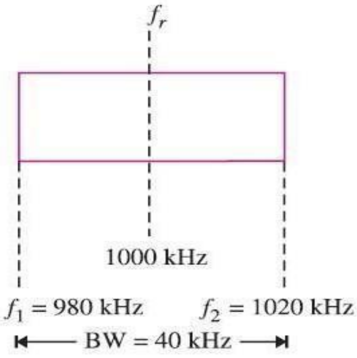


Fig.5.3

Illustration. Consider a tuned amplifier that is designed to amplify only those frequencies that are within ± 20 kHz of the central frequency of 1000 kHz (*i.e.*, *fr* = 1000 kHz). Here [See Fig. 5.3], *f*1 = 980 kHz, *fr* = 1000 kHz, *f*2 = 1020 kHz, *BW* = 40 kHz

This means that so long as the input signal is within the range of 980 – 1020 kHz, it will be amplified. If the frequency of input signal goes out of this range, amplification will be drastically reduced.

Analysis of Parallel Tuned Circuit

A parallel tuned circuit consists of a capacitor *C* and inductor *L* in parallel as shown in Fig. 15.4 (*i*).In practice, some resistance *R* is always present with the coil. If an alternating voltage is applied across this parallel circuit, the frequency of oscillations will be that of the applied voltage. However, if the frequency of applied voltage is equal to the natural or resonant frequency of *LC* circuit, then *electrical resonance* will occur. Under such conditions, the impedance of the tuned circuit becomes maximum and the line current is minimum. The circuit then draws just enough energy from a.c. supply necessary to overcome the losses in the resistance *R*.

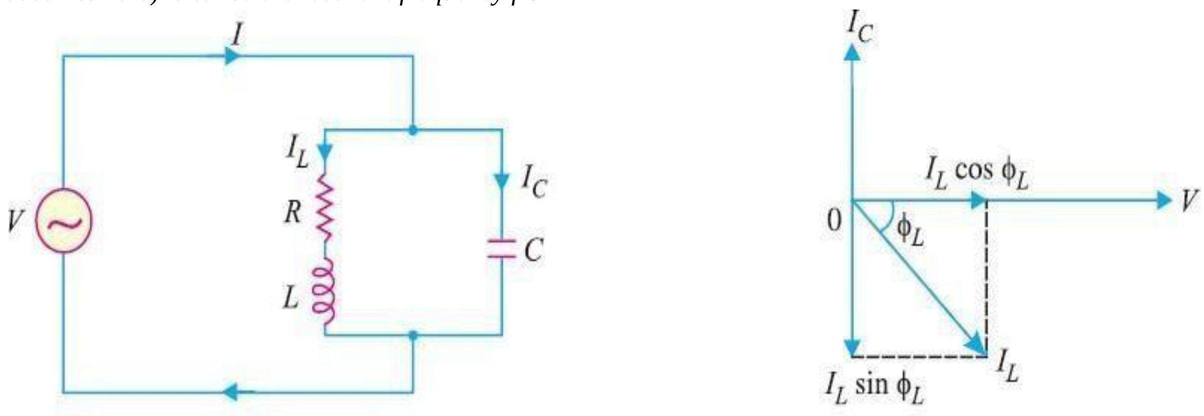
Parallel resonance. A parallel circuit containing reactive elements (*L* and *C*) is *resonant when the circuit power factor is unity *i.e*. applied voltage and the supply current are in phase. The phasor diagram of the parallel circuit is shown in Fig. 15. 4 (*ii*). The coil current *IL* has two rectangular components *viz* active component *IL* cos □*L* and reactive component *IL* sin w*L*. This parallel circuit will resonate when the circuit power factor is unity. This is possible only when the net reactive component of the circuit current is zero *i.e*.

$$I_C - I_L \sin\phi_L = 0$$

$$I_C = I_L \sin\phi_L$$

Resonance in parallel circuit can be obtained by changing the supply frequency. At some frequency *fr* (called resonant frequency), *IC* = *IL* sin w*L* and resonance occurs.

Resonant frequency. The frequency at which parallel resonance occurs (*i.e.* reactive component of circuit current becomes zero) is called the *resonant frequency fr*.



(*i*) (*ii*)

Fig. 5.4

At parallel resonance, we have, $I_C = I_L \sin \phi_L$

Now $I_L = V/Z_L$; $\sin \phi_L = X_L/Z_L$ and $I_C = V/X_C$

$$\therefore \quad \frac{V}{X_C} = \frac{V}{Z_L} \times \frac{X_L}{Z_L}$$

$$\text{or} \quad X_L X_C = Z_L^2$$

$$\text{or} \quad \frac{\omega L}{\omega C} = Z_L^2 = R^2 + X_L^2 \qquad ...(i)$$

$$\text{or} \quad \frac{L}{C} = R^2 + (2\pi f_r L)^2$$

$$\text{or} \quad (2\pi f_r L)^2 = \frac{L}{C} - R^2$$

$$\text{or} \quad 2\pi f_r L = \sqrt{\frac{L}{C} - R^2}$$

$$\text{or} \quad f_r = \frac{1}{2\pi L}\sqrt{\frac{L}{C} - R^2}$$

$$\therefore \quad \text{Resonant frequency, } f_r = \frac{1}{2\pi}\sqrt{\frac{1}{LC} - \frac{R^2}{L^2}} \qquad ...(ii)$$

If coil resistance R is small (as is generally the case), then,

$$f_r = \frac{1}{2\pi\sqrt{LC}} \qquad ...(iii)$$

The resonant frequency will be in Hz if *R*, *L* and *C* are in ohms, henry and farad respectively.

Note. If in the problem, the value of *R* is given, then eq. (*ii*) should be used to find *fr*. However, if *R* is not given, then eq. (*iii*) may be used to find *fr*.

15.4 Characteristics of Parallel Resonant Circuit

It is now desirable to discuss some important characteristics of parallel resonant circuit.

(i) Impedance of tuned circuit. The impedance offered by the parallel *LC* circuit is given by the supply voltage divided by the line current *i.e.*, *V/I*. Since at resonance, line current is minimum, therefore, impedance is maximum at resonant frequency. This fact is shown by the impedance- frequency curve of Fig 5.5. It is clear from impedance-frequency curve that impedance rises to a steep peak at resonant frequency *fr*. However, the impedance of the circuit decreases rapidly when the frequency is changed above or below the resonant frequency. This characteristic of parallel tuned circuit provides it the selective properties *i.e.* to select the resonant frequency and reject all others.

Line current, $I = I_L \cos \phi_L$

$$\text{or} \quad \frac{V}{Z_r} = \frac{V}{Z_L} \times \frac{R}{Z_L}$$

$$\text{or} \quad \frac{1}{Z_r} = \frac{R}{Z_L^2}$$

$$\text{or} \quad \frac{1}{Z_r} = \frac{R}{L/C} = \frac{CR}{L}$$

$$\left[\text{Q } Z_L^2 = \frac{L}{C} \text{ from eq. } (i)\right]$$

$$\therefore \quad \text{Circuit impedance, } Z_r = \frac{L}{CR}$$

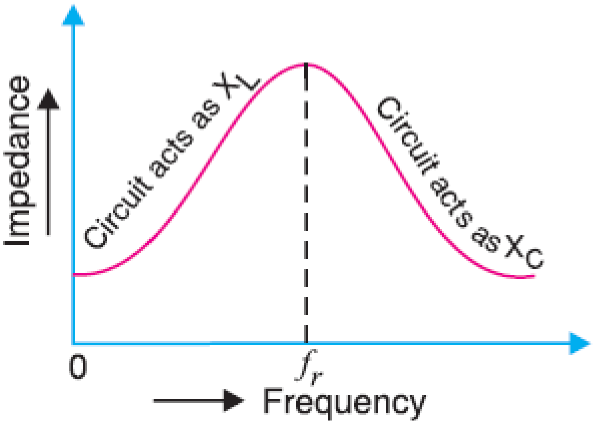


Fig. 5.5

Thus at parallel resonance, the circuit impedance is equal to **L*/*CR*. It may be noted that *Zr* will be in ohms if *R*, *L* and *C* are measured in ohms, henry and farad respectively.

(ii) Circuit Current. At parallel resonance, the circuit or line current *I* is given by the applied voltage divided by the circuit impedance *Zr i.e.*,

$$\text{Line current, } I = \frac{V}{Z_r} \qquad \text{where } Z_r = \frac{L}{CR}$$

Because *Zr* is very high, the line current *I* will be very small.

Quality factor Q. It is desired that resonance curve of a parallel tuned circuit should be as sharp as possible in order to provide selectivity. The sharp resonance curve means that impedance falls rapidly as the frequency is varied from the resonant frequency. The smaller the resistance of coil, the more sharp is the resonance curve. This is due to the fact that a small resistance consumes less power and draws a relatively small line current. The ratio of inductive reactance and resistance of the coil at resonance, therefore, becomes a measure of the quality of the tuned circuit. This is called *quality factor* and may be defined as under :

The ratio of inductive reactance of the coil at resonance to its resistance is known as **quality factor Q *i.e.*,

$$Q = \frac{X_L}{R} = \frac{2\pi f_r L}{R}$$

The quality factor *Q* of a parallel tuned circuit is very important because the sharpness of resonance curve and hence selectivity of the circuit depends upon it. The higher the value of *Q*, the more selective is the tuned circuit. Fig. 5.6 shows the effect of resistance *R* of the coil on the sharpness of the resonance curve. It is clear that when the resistance is small, the resonance curve is very sharp. However, if the coil has large resistance, the resonance curve is less sharp. It may be emphasised that where high selectivity is desired, the value of *Q* should be very large.

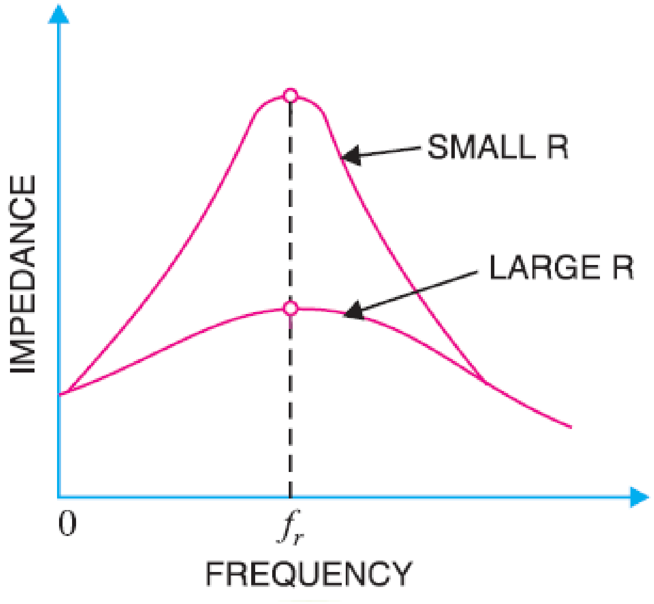


Fig. 5.6

Two things are worth noting. First, *Zr* (= *L*/*CR*) is a pure resistance because there is no frequency term present. Secondly, the value of *Zr* is very high because the ratio *L*/*C* is very large at parallel resonance.

** Strictly speaking, the *Q* of a tank circuit is defined as the ratio of the energy stored in the circuit to the energy lost in the circuit *i.e.*,

$$Q = \frac{\text{Energy stored}}{\text{Energy lost}} = \frac{\text{Reactive Power}}{\text{Resistive Power}} = \frac{I_L^2\, X_L}{I_L^2\, R} \quad \text{or} \quad Q = \frac{X_L}{R}$$

Example 5.1. A parallel resonant circuit has a capacitor of 250pF in one branch and inductance of 1.25mH plus a resistance of 10ohm in the parallel branch. Find (i) resonant frequency (ii) impedance of the circuit at resonance (iii) Q-factor of the circuit.

Solution.

$R = 10\Omega\,;\, L = 1.25 \times 10^{-3}\text{H};\; C = 250 \times 10^{-12}\text{ F}$

Fig. 15.6

(*i*) Resonant frequency of the circuit is

$$f_r = \frac{1}{2\pi}\sqrt{\frac{1}{LC} - \frac{R^2}{L^2}}$$

$$= \frac{1}{2\pi}\sqrt{\frac{10^{12}}{1.25 \times 10^{-3} \times 250} - \frac{10^2}{(1.25 \times 10^{-3})^2}} \text{ Hz}$$

$$= 284.7 \times 10^3 \text{ Hz} = \mathbf{284.7\ kHz}$$

(*ii*) Impedance of the circuit at resonance is

$$Z_r = \frac{L}{C\,R} = \frac{1.25 \times 10^{-3}}{250 \times 10^{-12} \times 10} = 500 \times 10^3\ \Omega$$

$$= \mathbf{500\ k\Omega}$$

(*iii*) Quality factor of the circuit is

$$Q = \frac{2\pi f_r L}{R} = \frac{2\pi\,(284.7 \times 10^3) \times 1.25 \times 10^{-3}}{10} = \mathbf{223.6}$$

Example 5.2. A parallel resonant circuit has a capacitor of 100 pF in one branch and inductance of 100 µH plus a resistance of 10 ohm in parallel branch. If the supply voltage is 10 V, calculate
(i) resonant frequency (ii) impedance of the circuit and line current at resonance.
Solution.

$R = 10\,\Omega$, $L = 100 \times 10^{-6}\,\text{H}$; $C = 100 \times 10^{-12}\,\text{F}$

(*i*) Resonant frequency of the circuit is

$$f_r = \frac{1}{2\pi}\sqrt{\frac{1}{LC} - \frac{R^2}{L^2}}$$

$$= \frac{1}{2\pi}\sqrt{\frac{10^{12}}{100 \times 10^{-6} \times 100} - \frac{10^2}{(100 \times 10^{-6})^2}}\ \text{Hz}$$

$$= 1592.28 \times 10^3\ \text{Hz} = \mathbf{1592.28\ kHz}$$

(*ii*) Impedance of the circuit at resonance is

$$Z_r = \frac{L}{C\,R} = \frac{L}{C} \times \frac{1}{R} = \frac{100 \times 10^{-6}}{100 \times 10^{-12}} \times \frac{1}{R}$$

$$= 10^6 \times \frac{1}{R} = 10^6 \times \frac{1}{10} = 10^5\ \Omega = \mathbf{0.1\ M\Omega}$$

Note that the circuit impedance *Zr* is very high at resonance. It is because the ratio *L*/*C* is very large at resonance.Line current at resonance is

$$I = \frac{V}{Z_r} = \frac{10\ \text{V}}{10^5\ \Omega} = \mathbf{100\ \mu A}$$

Example 5.3. The *dynamic impedance of a parallel resonant circuit is 500 k . The circuit consists of a 250 pF capacitor in parallel with a coil of resistance 10ohm. Calculate (i) the coil inductance (ii) the resonant frequency and (iii) Q-factor of the circuit.
Solution.

(*i*) Dynamic impedance, $Z_r = \dfrac{L}{CR}$

$\therefore$ Inductance of coil, $L = Z_r\,CR = (500 \times 10^3) \times (250 \times 10^{-12}) \times 10$

$= 1.25 \times 10^{-3}\ \text{H} = \mathbf{1.25\ mH}$

(*ii*) Resonant frequency, $f_r = \dfrac{1}{2\pi}\sqrt{\dfrac{1}{LC} - \dfrac{R^2}{L^2}}$

$$= \frac{1}{2\pi}\sqrt{\frac{10^{12}}{1.25 \times 10^{-3} \times 250} - \frac{10^2}{(1.25 \times 10^{-3})^2}}$$

$$= 284.7 \times 10^3\ \text{Hz} = \mathbf{284.7\ kHz}$$

(*iii*) Q-factor of the circuit $= \dfrac{2\pi f_r L}{R} = \dfrac{2\pi \times (284.7 \times 10^3) \times (1.25 \times 10^{-3})}{10} = \mathbf{223.6}$

Frequency Response of Tuned Amplifier
The voltage gain of an amplifier depends upon , input impedance and effective collector load. In a tuned amplifier, tuned circuit is used in the collector. Therefore, voltage gain of such an amplifier is given by :

$$\text{Voltage gain} = \frac{\beta Z_C}{Z_{in}}$$

where *ZC* = effective collector load

Zin = input impedance of the amplifier

The value of *ZC* and hence gain strongly depends upon frequency in the tuned amplifier. As *ZC* is maximum at resonant frequency, therefore, voltage gain will be maximum at this frequency. The value of *ZC* and gain decrease as the frequency is varied above and below the resonant frequency. Fig. 5.7 shows the frequency response of a tuned amplifier. It is clear that voltage gain is maximum at resonant frequency and falls off as the frequency is varied in either direction from resonance.

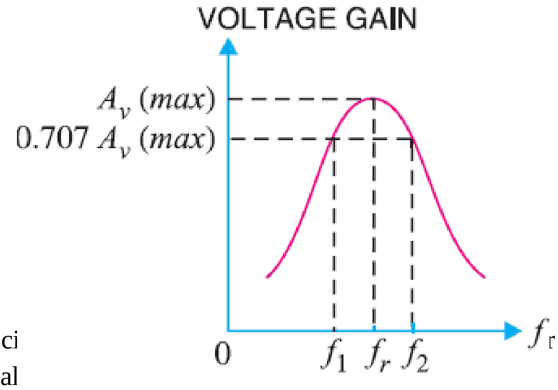


Fig.5.7

Bandwidth. The range of frequenci r falls to 70.7 % of the maximum gain is cal 15.7, the bandwidth of tuned amplifier is $f_1 - f_2$. The amplifier will amplify nicely any signal in this frequency range. The bandwidth of tuned amplifier depends upon the value of *Q* of *LC* circuit *i.e.* upon the sharpness of the frequency response. The greater the value of *Q* of tuned circuit, the lesser is the bandwidth of the amplifier and *vice-versa*. In practice, the value of *Q* of *LC* circuit is made such so as to permit the amplification of desired narrow band of high frequencies. The practical importance of bandwidth of tuned amplifiers is found in communication system. In radio and TV transmission, a very high frequency wave, called *carrier wave* is used to carry the audio or picture signal. In radio transmission, the audio signal has a frequency range of 10 kHz. If the carrier wave frequency is 710 kHz, then the resultant radio wave has a frequency range *between (710 –5) kHz and (710 +5) kHz. Consequently, the tuned amplifier must have a bandwidth of 705 kHz to 715 kHz (*i.e.* 10 kHz). The *Q* of the tuned circuit should be such that bandwidth of the amplifier lies in this range.

Relation between Q and Bandwidth

The quality factor *Q* of a tuned amplifier is equal to the ratio of resonant frequency (*fr*) to bandwidth (*BW*) *i.e.*,

$$Q = \frac{f_r}{BW}$$

The *Q* of an amplifier is determined by the circuit component values. It may be noted here that *Q* of a tuned amplifier is generally greater than 10. When this condition is met, the resonant frequency at parallel resonance is approximately given by:

$$f_r = \frac{1}{2\pi\sqrt{LC}}$$

Example 5.4. The Q of a tuned amplifier is 60. If the resonant frequency for the amplifier is 1200 kHz, find (i) bandwidth and (ii) cut-off frequencies.
Solution.

(*i*) $$BW = \frac{f_r}{Q} = \frac{1200 \text{ kHz}}{60} = \mathbf{20 \text{ kHz}}$$

(*ii*) $$\text{Lower cut-off frequency, } f_1 = 1200 - 10 = \mathbf{1190 \text{ kHz}}$$

$$\text{Upper cut-off frequency, } f_2 = 1200 + 10 = \mathbf{1210 \text{ kHz}}$$

Example 5.5. A tuned amplifier has maximum voltage gain at a frequency of 2 MHz and the bandwidth is 50 kHz. Find the Q factor.

Solution. The maximum voltage gain occurs at the resonant frequency. Therefore, fr = 2 MHz = 2 × 10^6 Hz and BW = 50 kHz = 50 × 10^3 Hz.
Now

$$BW = \frac{f_r}{Q}$$

$$Q = \frac{f_r}{BW} = \frac{2 \times 10^6}{50 \times 10^3} = \mathbf{40}$$

Example 5.6. Draw the frequency response of an ideal tuned amplifier and discuss its characteristics.

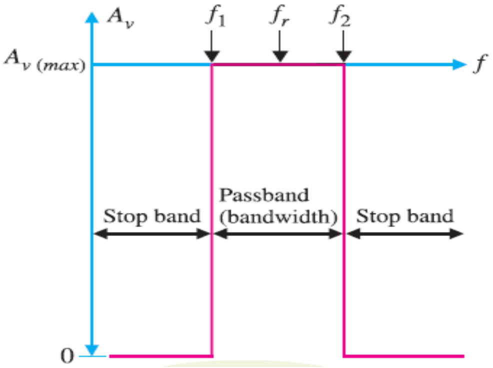


Fig.5.8

Solution. Fig. 5.8 shows the frequency response of an ideal tuned amplifier. The ideal tuned amplifier has zero gain for all frequencies from 0 Hz up to the lower cut-off frequency *f1*. At this point, the gain *instantly* jumps to the maximum value [*Av* (*max*)]. The gain stays at the maximum value until *f*2 is reached. At this time, the gain *instantly* drops back to zero. Thus all the frequencies within the bandwidth (*f*1 to *f*2) of the amplifier would be *passed* by the circuit while all others would be effectively stopped. This is where the terms *pass band* and *stop band* come from. The pass band is the range of frequencies that is passed (amplified) by a tuned amplifier.

On the other hand, the stop band is the range of frequencies that is outside the amplifier‘s pass band. In practice, the ideal characteristics of the tuned amplifier cannot be achieved. In a practical frequency response (refer back to Fig. 5.7), the gain falls gradually from maximum value as the frequency goes outside the *f*1 or *f*2 limits. However, the closer the frequency response of a tuned amplifier to that of the ideal, the better.

Single Tuned Amplifier

A single tuned amplifier consists of a transistor amplifier containing a parallel tuned circuit as the collector load. The values of capacitance and inductance of the tuned circuit are so selected that its resonant frequency is equal to the frequency to be amplified. The output from a single tuned amplifier can be obtained either (*a*) by a coupling capacitor *CC* or (*b*) by a secondary coil

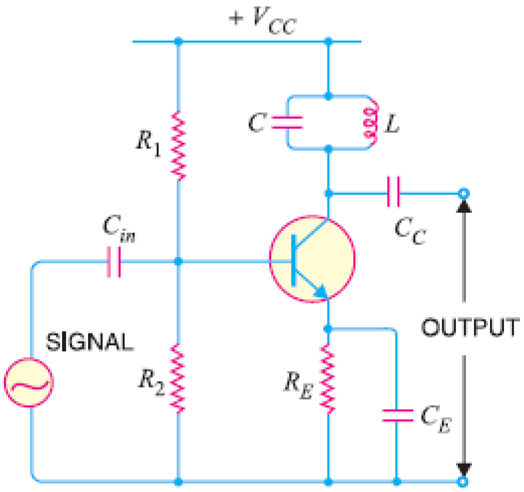


Fig.5.9 Capacitive coupled single tuned amplifier

Operation. The high frequency signal to be amplified is given to the input of the amplifier. The resonant frequency of parallel tuned circuit is made equal to the frequency of the signal by changing the value of *C*. Under such conditions, the tuned circuit will offer very high impedance to the signal frequency. Hence a large output appears across the tuned circuit. In case the input signal is complex containing many frequencies, only that frequency which corresponds to the resonant frequency of the tuned circuit will be amplified. All other frequencies will be rejected by the tuned circuit. In this way, a tuned amplifier selects and amplifies the desired frequency.

. The fundamental difference between *AF* and tuned (*RF*) amplifiers is the bandwidth they are expected to amplify. The *AF* amplifiers amplify a major portion of *AF* specturm (20 Hz to 20 kHz) equally well throughout. The tuned amplifiers amplify a relatively narrow portion of *RF* spectrum, rejecting all other frequencies L, C tuned circuit is not connected between collector and ground because, the transistor will be short circuited at some frequency other than resonant frequency. The output of the tuned circuit is coupled to the next stage or output device, through capacitor Cb. So this circuit is called single tuned capacitbe coupled amplifier. RI, R2, RE, CE are biasing resistors and capacitors. The tuned circuit formed by Inductance (L) and capacitor

(C) resonates at the frequency of operation. Transistor hybrid equivalent circuit must be used since the transistor is operated at high frequencies. Tuned circuits are high frequency circuits. Rj = input resistance of the next stage.

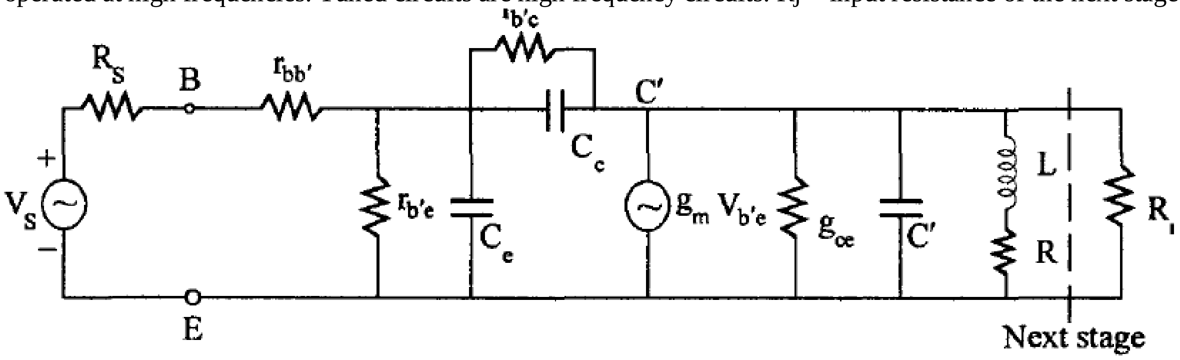


Fig. 5.10 Equivalent circuit

Modified equivalent circuit using Miller's Theorem.

According to Miller's theorem, the feedback capacitance Cc is Cc (1 - A) on the input side and on the output side.

$C_c\left(\frac{A-1}{A}\right)$ But where as resistance is $\frac{r_{b'c}}{(1-A)}$ on the input side $\frac{r_{b'c}}{\left(\frac{A-1}{A}\right)}$ on the output side.

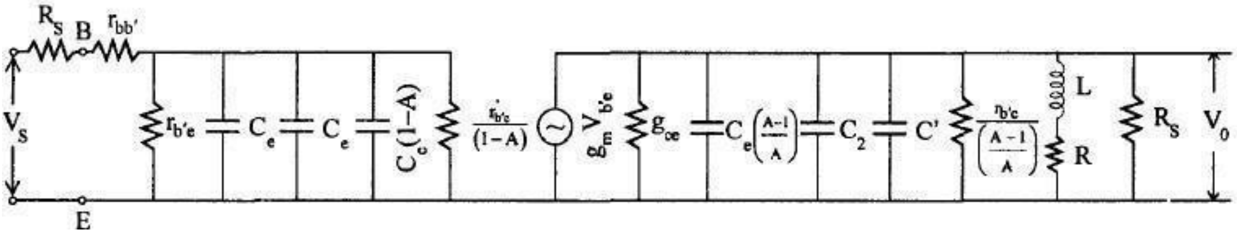


Fig. 5.11 Equivalent circuit (applying Miller's Theorem)

The equivalent circuit after simplification, neglecting $\frac{r_{b'c}}{\left(\frac{A-1}{A}\right)}$ is shown in Fig. 5.12.

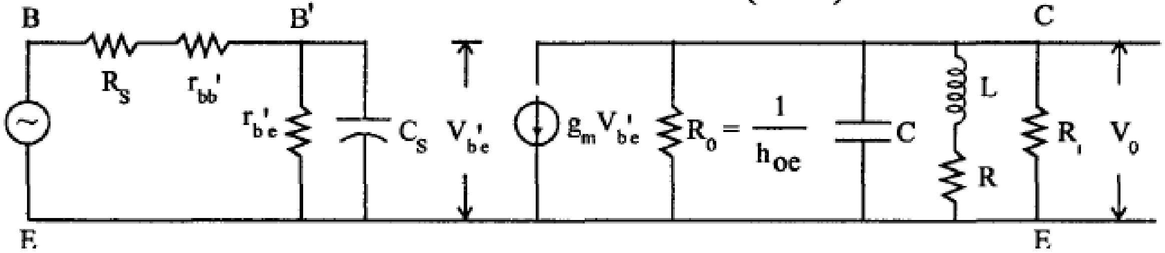


Fig. 5.12 Simplified equivalent circuit

$$Y_i = \frac{1}{R + j\omega L} = \frac{R - j\omega L}{R^2 + \omega^2 L^2} = \frac{R}{R^2 + \omega^2 L^2} - j\frac{\omega L}{R^2 + \omega^2 L^2}$$

Input admittance as seen by II stage.

Instead of L and R being in series, they are being represented as equivalent shunt element~ Rp and Lp for parallel

$$= \frac{1}{R_P} + \frac{1}{j\omega L_P}$$

$$R_P = \frac{R^2 + \omega^2 L^2}{R}$$

$$L_P = \frac{R^2 + \omega^2 L^2}{\omega^2 L}$$

Inductor is represented by Rp in series with inductance Lp. Q at

resonance

$$Q_0 = \frac{\omega_0 L}{R}$$

$$\omega L >> R$$

$$R_P = \frac{\omega^2 L^2}{R}$$

$$L_P = L$$

.: Resistance of the inductor R is small, neglecting R^2 compared to $^2 L^2$. Therefore □

output circuit is simplified to,

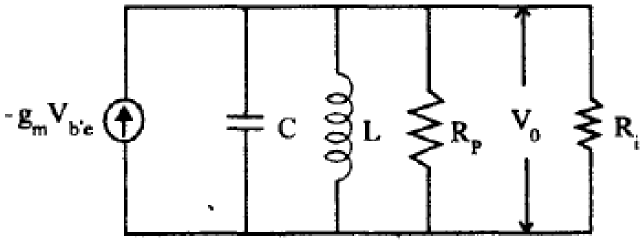


Fig. 5.13 Simplified circuit

$$\frac{1}{R_t} = \frac{1}{R} + \frac{1}{R_P} + \frac{1}{R_i}$$

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

R_i is the input Fesistance of the next stage

$$Q_e = \frac{\text{Suspectance of L or capacitance of C}}{\text{Conductance of shunt resistance } R_t}$$

R_t = resistance of tuned circuit

$$Q_e = \omega_0 CR_t$$
$$= \frac{R_t}{\omega_0 L} \frac{(1/\omega_0 L)}{1/R_t} = \left(\frac{\omega_0 C}{1/R_t}\right)$$

□ □ □□ be the resonant angular frequency in rad/sec.

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

Output voltage Vo = -gm Vb'e·Z (-gm Vb'e is the current source). where Z is the impedance of C, Land Rt in parallel.

Admittance

$$Y = \frac{1}{Z} = \frac{1}{R_t} + \frac{1}{j\omega L} + j\omega C$$

$$Y = \frac{1}{R_t}\left[1+\frac{R_t}{j\omega L}+j\omega C R_t\right]$$

$$= \frac{1}{R_t}\left[1+j\,\frac{\omega_0\,\omega C\,R_t}{\omega_0}+\frac{R_t\,\omega_0}{j\omega_0\,\omega L}\right] \text{(Multiplying and dividing by } \omega_0\text{)}$$

$$Y = \frac{1+jQ_e\left[\frac{\omega}{\omega_0}-\frac{\omega_0}{\omega}\right]}{R_t} \qquad (\because \quad Q_e = \frac{R_t}{w_0 L} = \omega_0\,CR_t)$$

$$Q_e = \omega_0\,C\,R_t$$

$$= \frac{R_t}{\omega_0 L} \qquad \because \quad \omega_0 L = \frac{1}{\omega_0 C}$$

$$Z = \frac{R_t}{1+jQ_e\left[\frac{\omega}{\omega_0}-\frac{\omega_0}{\omega}\right]}$$

Qe is defined as

$$\frac{\text{Susceptance of L or C}}{\text{Conductance of shunt resistance } R_t}$$

$$\delta = \frac{\omega-\omega_0}{\omega_0} = \frac{\omega}{\omega_0} - 1$$

$$\frac{\omega}{\omega_0} = 1+\delta$$

Rewriting the expression for Z, as

$$Z = \frac{R_t}{1+jQ_e\left[(1+\delta)-\frac{1}{(1+\delta)}\right]}$$

$$= \frac{X+\delta^2+2\delta-X}{1+\delta} = \frac{2\delta\left(1+\frac{\delta}{2}\right)}{1+\delta}$$

$$Z = \frac{R_t}{1+j2Q_e\,\delta\left[\frac{1+\delta/2}{1+\delta}\right]}$$

If the frequency □ is close to resonant frequency □$_{0,}$□ «Therefore Simplified expression for Z is

$$Z = \frac{R_t}{1 + j2Q_e\delta}$$

$$R_p = Q_0^2 R = Q_0 = \sqrt{\frac{L}{C}} = \omega_0 L Q_0$$

$$V_{b'e} = V_i \cdot \frac{r_{b'e}}{r_{bb'} + r_{b'e}}$$

$$V_0 = -g_m V_{be} \cdot Z$$

$$V_0 = -g_m V_i \cdot \frac{{}^{\iota}b'e}{r_{b'e} + r_{b'e}} \; Z$$

$$A = \frac{V_0}{V_i} = -g_m \cdot \frac{r_{b'e}}{r_{b'e} + r_{bb'}} \cdot Z$$

$$A = -g_m \cdot \frac{r_{b'e}}{r_{b'e} + r_{bb'}} \cdot \frac{R_t}{1 + j2\delta Q_e}$$

voltage gain at resonance. Since at resonance w0 = 0

$$A_{reso} = \frac{-g_m \cdot r_{b'e}}{r_{b'e} + r_{bb'}} \cdot R_t$$

$$\frac{A}{A_{reso}} = \frac{1}{1 + j2\delta Q_e} \cdot$$

Magnitude

$$\left|\frac{A}{A_{reso}}\right| = \frac{1}{\sqrt{1 + (2\delta Q_e)^2}}$$

Phase angle

$$\left|\frac{A}{A_{reso}}\right| = -\tan^{-1}(2\,\delta\,Q_e)$$

At a frequency □ $_1$ below the resonant frequency □ has the value,

$$= -\frac{1}{2Q_e};$$

$$\frac{A}{A_{reso}} = \frac{1}{\sqrt{2}} = 0.707$$

□ $_1$ is the lower 3db frequency. Similarly □ $_2$, the upper 3db frequency is

$$\delta = +\frac{1}{2Q_e};\; \frac{A}{A_{reso}} = \frac{1}{\sqrt{2}} = 0.707$$

The 3 db band width □ = (□ 2 □ $_1$l)

$$= \frac{[(\omega_2 - \omega_0) + (\omega_0 - \omega_1)] \cdot \omega_0}{\omega_0}$$

$$= [\delta + \delta]\,\omega_0 \qquad = 2\,\delta\,\omega_0$$

$$\delta = \frac{1}{2Q_e}$$

$$\Delta \omega = \frac{\omega_0}{Q_e} = \frac{\omega_0}{R_t \omega_0 C} = \frac{1}{R_t C} \text{ rad/sec.}$$

Tapped Single Tuned Capacitance Coupled Amplifier:

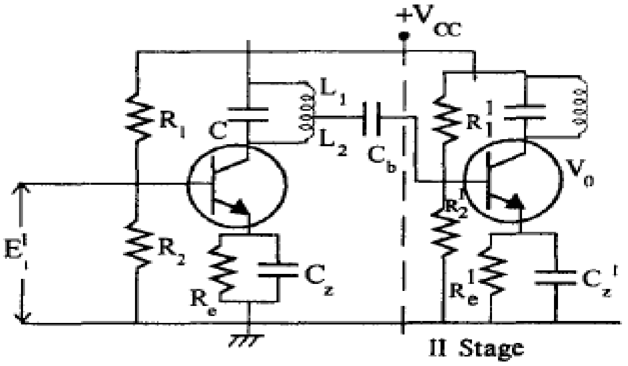
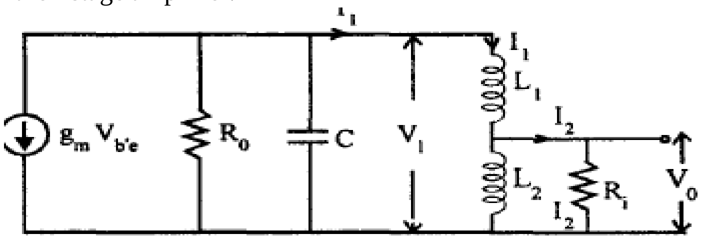


Fig. 5.7 Tapped single tuned capacitive coupled amplifier circuit

5.3.1 Equivalent Circuit on the Output Side of the I Stage

RI is the input resistance of the II stage.

Ro is the output resistance of the I stage amplifier.



The input IZI of the common emitter amplifier circuits will be less. So the output impedance of the circuit being coupled to one common emitter amplifier, should also have low IZI for impedance matching and to get maximum power transfer. So in order to reduce the impedance of the LC resonant circuit, to match the low IZI of the common emitter circuit, tapping is made in the LC tuned circuit. Tapped single tuned circuits are used in such applications. 5.3.2 Expression for 'Inductance' for Maximum Power Transfer Let the tapping point divide the impedance into two parts LI and L2. Let LI = nL so that L2 = (1 - n) Writing Kirchoff's Voltage Law (KVL)

Where M is the mutual inductance between LI and L2. Solving equations 1 and 2,

$$V_1 = j\omega L \,.\, I_1 - j\omega\,(L_2 + M)\, I_2 \qquad \text{.....(1)}$$

$$0 = -j\omega\,(L_1 + M)\, I_1 + (R_i + j\omega L_2)\, I_2 \qquad \text{.....(2)}$$

$$I_1 = \frac{V_1\left(R_i + j\omega L_2\right)}{j\omega L\left(R_i + j\omega L_2\right) + \omega^2\left(L_2 + M\right)^2}$$

Hence the IZI offered by the coil along with input resistance ~i of the next stage is

$$Z_1 = \frac{V_1}{I_1} = \frac{j\omega L\left(R_i + j\omega L_2\right) + \omega^2\left(L_2 + M\right)^2}{\left(R_i + j\omega L_2\right)}$$

$$= j\omega L + \frac{\omega^2\left(L_2 + M\right)^2}{R_i + j\omega L_2}$$

But wL2 much less than Ri.
As Ri, the input resistance of transistor circuit II stage is KQ and much greater than roL2

$$Z_1 = j\omega L + \frac{\omega^2\left(L_2 + M\right)^2}{R_i}$$

$$M = K\sqrt{L_1 L_2} \qquad \text{M = Mutual Inductance}$$

Where K is the coefficient of coupling. Since Ll = nL, L2 = (l-n)L

$$= K\sqrt{nL(1-n)L} \;=\; KL\sqrt{\left(n - n^2\right)}$$

Putting K = I, we get

$$M \simeq L\sqrt{n - n^2}$$

$$Z_1 \simeq j\omega L \pm \frac{\omega^2\left[(1-n)L + L\sqrt{n-n^2}\right]^2}{R_i}$$

The resistance effectively reflected in series with the coil due to the resistance R, is given by,

$$R_{is} \simeq \frac{\omega^2 L^2\left[(1-n) + \sqrt{n-n^2}\right]^2}{R_i}$$

This is the resistance component; $: series, i : input
This resistance R is in series with the coil L may be equated to a resistance Rip in shunt with the coil where Rip is given by, ~

$$R_{ip} = (\omega L)^2 / R_{is}$$

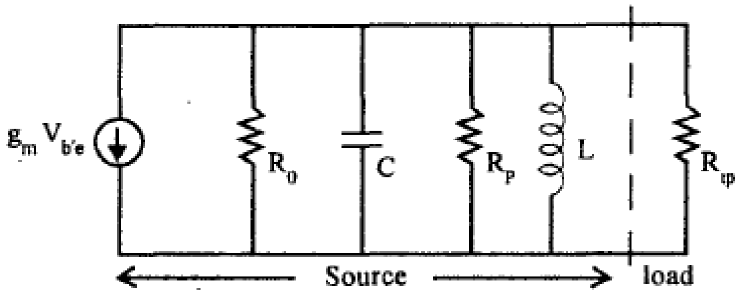


Fig. 5.9 Equivalent circuit

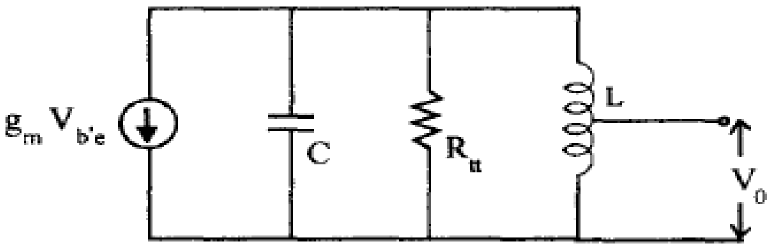


Fig. 5.10 Equivalent circuit after simplification

$$\frac{1}{R_{tt}} = \frac{1}{R_0} + \frac{1}{R_p} + \frac{1}{R_{ip}}$$

$$Q_e = \frac{R_{tt}}{\omega_0 L}$$

Rtt : tuned tapped circuit.

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

Under the conditions of maximum power transfer theorem, the total resistance appearing in shunt with the coil is = Rop

Since it is a resonant circuit, at resonance, the IZI in purely resistive. For maximum power transfer IZI = R/2.

$$Q_e = \frac{R_{op}/2}{\omega_0 L}; \quad R_{tt} = R_{op}/2$$

$$R_{op} = 2\, Q_e \cdot \omega_0\, L$$

$$R_{op} = \frac{R_0 R_p}{R_0 + R_p}$$

$$2\, Q_e\, \omega_0\, L = \frac{R_0\, \omega_0\, Q_0\, L}{R_0 + \omega_0\, Q_0\, L}$$

$$L = \frac{R_0 (Q_0 - 2Q_e)}{2\omega_0 Q_0 Q_e}$$

$$L = \frac{R_0}{\omega_0}\left[\frac{1}{2Q_e} - \frac{1}{Q_0}\right]$$

This is the value of L for maximum power transfer.

Expression for voltage gain and Bandwidth are determined in the same way as done for a single tuned circuit. In this circuit we have,

1. R_{tt} instead of ~ (as in single tuned) tapped tuned circuit.

2. Output voltage equals (l - n) times the voltage developed across the complete coil. IZI at any frequency close to Wo is given by,

$$Z = \frac{R_{tt}}{1 + j\,2\delta\,Q_e}$$

$$V_0 = \frac{-g_m V_i\, r_{b'e}}{r_{b'e} + r_{bb'}} \cdot Z\,(1-n)$$

$$A = \frac{V_0}{V_i} = -g_m\,(1-n)\,\frac{r_{b'e}}{r_{b'e} + r_{bb'}} \cdot Z$$

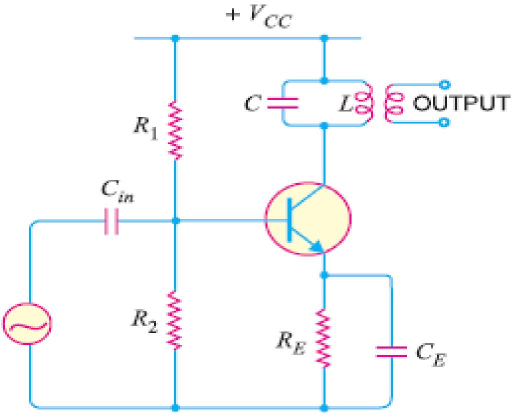
$$= -g_m\,(1-n) \cdot \frac{r_{b'e}}{r_{b'e} + r_{bb'}} \cdot \frac{R_{tt}}{1 + j\,2\delta Q_e}$$

At resonance, voltage gain is

$$A_{reso} = -\,g_m\,(1-n) \cdot \frac{r_{b'e}}{r_{b'e} + r_{bb'}} \cdot R_{tt}$$

$$\frac{A}{A_{reso}} = \frac{1}{1 - j\,2\delta Q_e}$$

Inductive coupled single tuned amplifier



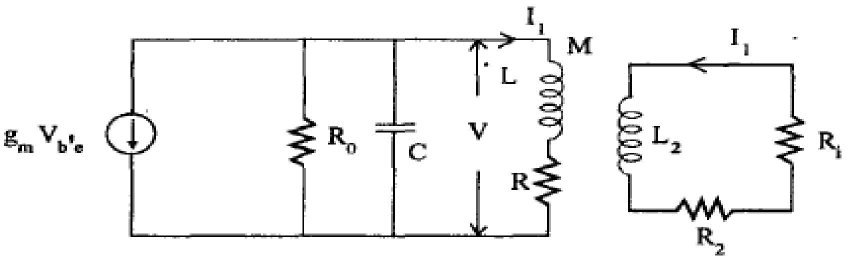


Fig. 5.11 Inductive coupled amplifier circuit (a) and its equivalent (b)
in this circuit, the voltage developed across the tuned circuit is inductively coupled to the next stage. Coil L, of the tuned circuit, and the inductor coupling the voltage to the II stage, L2 form' a transformer with mutual coupling M. This type of circuit is also used, where the input IZI of the II stage is smaller or different from the tuned circuit. SO IZI matching is done by the transformer depending on its tum ratio. In such requirements, this type of circuit is used.

The resistors RI, R2 and R; and R2 are the biasing resistors. The parallel tuned circuit,
Land C resonates at the frequency of operation. Fig. (b) shows output equivalent circuit. Input equivalent circuit will be the same as that of the capacitive coupled circuit. In the output equivalent circuit, C is the total capacitance, including the stray capacitance, Miller equivalent capacitance C (A-1)/A. L2 and R2 are the inductance and resistance of the secondary winding.

5.4.1 Expression for Ll for Maximum Power Transformer Writing KVL to the primary and secondary windings,
V = I_1 Z_{11} + I_2 Z_{12} o = I_1 Z_{21} + I_2 Z_{22}

where Z_{11} = R + jwL Z_{12} = Z_{21} = jwM
Z_{22} = R2 + Rj + jwL2

$$I_1 = \frac{V.Z_{22}}{Z_{11}Z_{22} - Z_{12}^2}$$

The impedance seen looking into the primary is,

$$Z_{in} = \frac{V}{I_1} \frac{Z_{11}Z_{22} - Z_{12}^2}{Z_{22}}$$

$$= Z_{11} - \frac{Z_{12}^2}{Z_{22}}$$

Substituting the values of Zll' Z22 and ZI2 in equation (7) we get,

$$Z_{in} = (R + j\omega L) + \frac{\omega^2 M^2}{R_2 + R_i + j\omega L_2}$$

Rj generally much greater than R2 and wL2.

$$Z_{in} \simeq (R + j\omega L) + \frac{\omega^2 M^2}{R_i}$$

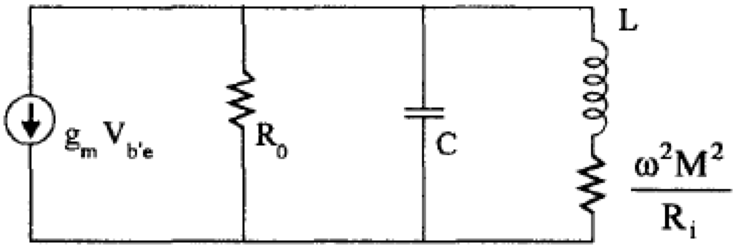


Fig. 5.12 Equivalent circuit

$$\frac{\omega^2 M^2}{R_i}$$

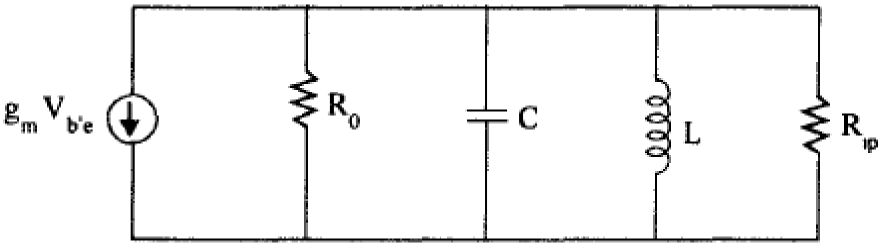
is the impedance of the secondary side reflected to the primary. If M is reasonably large, then I

$$R << \frac{\omega^2 M^2}{R_i}$$

$$Z_{in} \simeq j\omega L + \frac{\omega^2 M^2}{R_i}$$

The equivalent circuit may be written as,
Inductance L with series resistance may be represented as L in shunt with Rio as R. shown below, where

$$R_{ip} = \frac{(\omega L)^2}{\left(\frac{\omega^2 M^2}{R_i}\right)} = \left(\frac{L}{M}\right)^2 . R_i$$



$$R_0 = \left(\frac{L}{M}\right)^2 . R_i$$

Fig. 5.13 Simplified circuit

$$R_{ip} = R_0$$

gives the value of M for maximum power transfer.

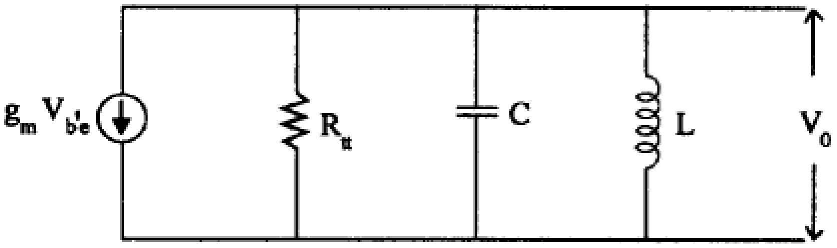


Fig. 5.14 Equivalent circuit
L and L2 are the primary and secondary windings of inductances.

$$M = K\sqrt{LL_2}$$

$$R_0 = \left(\frac{L}{K^2 L_2}\right) R_i$$

Therefore from equation 14, for a given value of Ro and coefficient of coupling K and ~, we can determine L2 for maximum transformer of power.

Shunt resistance Ro and Rip may be combined to yield the total shunt resistance Ru.

$$\frac{1}{R_{tt}} = \frac{1}{R_0} + \frac{1}{R_{ip}}$$

Rtt = Resistance of tapped tuned circuit Effective Q of the entire circuit is,

$$Q_e = \frac{R_{tt}}{\omega_0 L}$$

where Wo is the resonaqt frequency of Land C.

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

Under conditions of maximum transfer of power, total resistance appearing in shunt with the coil equals *Ro/2.* Since it is respnant circuit, at resonance, IZI = resistance only.

maximum power, R = Rl2.

$$Q_e = \frac{R_0/2}{\omega_0 L}$$

$$R_0 = 2\,Q_e\,\omega_0\,L$$

$$I_2 = \frac{V.Z_{21}}{Z_{21}.Z_{12} - Z_{11}\,Z_{22}}$$

$$V_0 = -I_2\,.\,R_i$$

$$= V\,.\,\frac{R_i.Z_{21}}{Z_{11}.Z_{22} - Z_{12}^2}$$

IZI of the output circuit at any frequency 'w' close to _w0' is given by, Impedance of output circuit is

$$Z = \frac{R_{tt}}{1 + j2\delta Q_e}$$

$$V_0 = -g_m\,.\,\frac{V_0\,r_{b'e}}{r_{b'e} + r_{bb'}}\,.\,Z\,.\,\frac{R_i\,Z_{21}}{Z_{11}\,Z_{22} - Z_{12}^2}$$

$$= -g_m\,V_i\,.\,\frac{r_{b'e}}{r_{b'e} + r_{bb'}}\,.\,\frac{R_i.Z_{21}}{Z_{11}\,Z_{22} - Z_{12}^2}\,.\,\frac{R_{tt}}{1 + j2\delta Q_e}$$

Voltage gain A at any frequency wo is,

$$A = \frac{V_0}{V_i} = -g_m\,\frac{r_{b'e}}{r_{b'e} + r_{bb'}}\,.\,\frac{R_i\,Z_{21}}{Z_{11}\,Z_{22} - Z_{12}^2}\,.\,\frac{R_{tt}}{1 + j2\delta Q_e}$$

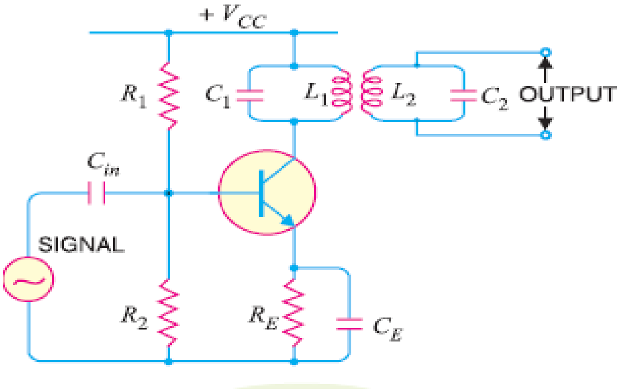
Voltage at resonance,

$$A_{reso} = -g_m \cdot \frac{r_{b'e}}{r_{b'e} + r_{bb'}} \cdot \frac{R_i Z_{22}}{Z_{11} Z_{22} - Z_{12}^2}$$

$$\boxed{\frac{A}{A_{reso}} = \frac{1}{1 + j2\delta Q_e}}$$

Double Tuned Amplifier

Fig. 15.13 shows the circuit of a double tuned amplifier. It consists of a transistor amplifier containing two tuned circuits ; one (*L*1*C*1) in the collector and the other (*L*2*C*2) in the output as



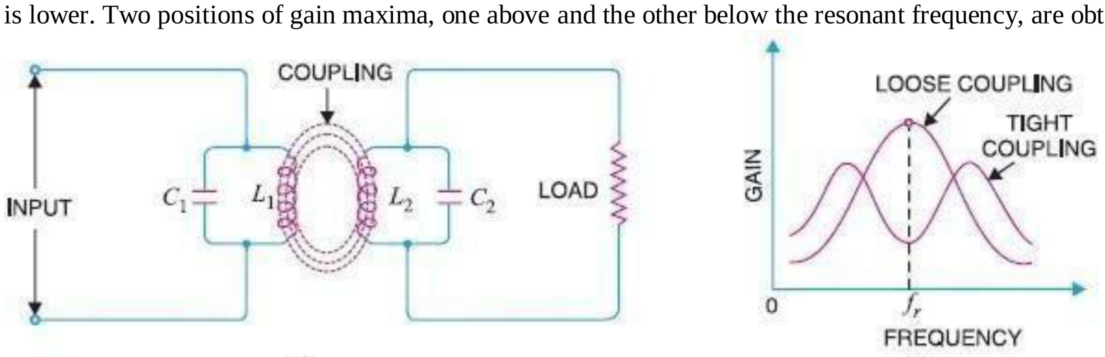
shown. The high frequency signal to be amplified is applied to the input terminals of the amplifier. The resonant frequency of tuned circuit *L*1*C*1 is made equal to the signal frequency. Under such conditions, the the signal frequency. Consequently, large output appears across the tuned circuit *L*1*C*1. The output from this tuned circuit is transferred to the second tuned circuit *L*2*C*2 through mutual induction. Double tuned circuits are extensively used for coupling the various circuits of radio and television receivers.

Frequency response. The frequency response of a double tuned circuit depends upon the degree of coupling *i.e.* upon the amount of mutual inductance between the two tuned circuits. When coil *L*2 is coupled to coil *L*1 [See Fig. 15.14 (*i*)], a portion of load resistance is coupled into the primary tank circuit *L*1*C*1 and affects the primary circuit in exactly the same manner as though a resistor had been added in series with the primary coil *L*1. When the coils are spaced apart, all the primary coil *L*1 flux will not link the secondary coil *L*2. The coils are said to have *loose coupling*. Under such conditions, the resistance reflected from the load (*i.e.* secondary circuit) is small. The resonance curve will be sharp and the circuit *Q* is high as shown in Fig. 15.14 (*ii*).

When the primary and secondary coils are very close together, they are said to have *tight coupling*.

Under such conditions, the reflected resistance will be large and the circuit *Q*

is lower. Two positions of gain maxima, one above and the other below the resonant frequency, are obtained.



(*i*) (*ii*)

Bandwidth of Double–Tuned Circuit

If you refer to the frequency response of double-tuned circuit shown in Fig. 15.14 (*ii*), it is clear that bandwidth increases with the degree of coupling. Obviously, the determining factor in a doubletuned circuit is not *Q* but the coupling. For a given frequency, the tighter the coupling, the greater is the bandwidth.

BWdt = k fr

The subscript *dt* is used to indicate double-tuned circuit. Here *k* is coefficient of coupling.

Example 15.8. It is desired to obtain a bandwidth of 200 kHz at an operating frequency of 10 MHz using a double tuned circuit. What value of co-efficient of coupling should be used ?

Solution.

BWdt = k fr

$$\therefore \quad \text{Co-efficient of coupling, } k = \frac{BW_{dt}}{f_r} = \frac{200 \text{ kHz}}{10 \times 10^3 \text{ kHz}} = 0.02$$

Stagger Tuning

Tuned amplifiers have large gain, since at resonance, Z is maximum. So Av is maximum. To get this large Av over a wide range of frequencies, stagger tuned amplifiers are employed. This is done by taking two single tuned circuits of a certain Bandwidth, and displacing or staggering their resonance peaks by an amount equal to their Bandwidth. The resultant staggered pair will have a Bandwidth,

√*2* times as great as that of each of individual pairs.