Digital Logic

UNIT 1

Chapter 1

Number Systems

LEARNING OBJECTIVES

- *Digital circuits*
- *Number system with different base*
- *Conversion of number systems*
- *Complements*
- *Subtraction with complement*
- *Numeric codes*
- *Weighted and non-weighted codes*
- *Error detection and correction code*
- *Sequential, reflective and cyclic codes*
- *Self complementing code*

Digital Circuits

Computers work with binary numbers, which use only the digits '0' and '1'. Since all the digital components are based on binary operations, it is convenient to use binary numbers when analyzing or designing digital circuits.

Number Systems with Different Base

Decimal number system

Decimal numbers are usual numbers which we use in our day-to-day life. The base of the decimal number system is 10. There are ten numbers 0 to 9.

The value of the nth digit of the number from the right side = nth digit × $(\text{base})^{n-1}$

Example 1: $(99)_{10} \rightarrow 9 \times 10^1 + 9 \times 10^0$
$= 90 + 9 = 99$

Example 2: $(332)_{10} \rightarrow 3 \times 10^2 + 3 \times 10^1 + 2 \times 10^0$
$= 300 + 30 + 2$

Example 3: $(1024)_{10} \rightarrow 1 \times 10^3 + 0 \times 10^2 + 2 \times 10^1 + \times 10^0$
$= 1000 + 0 + 20 + 4 = 1024$

Binary number system

In binary number system, there are only two digits '0' and '1'. Since there are only two numbers, its base is 2.

Example 4: $(1101)_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
$= 8 + 4 + 1 = (13)_{10}$

Octal number system

Octal number system has eight numbers 0 to 7. The base of the number system is 8. The number $(8)_{10}$ is represented by $(10)_8$.

Example 5: $(658)_8 = 6 \times 8^2 + 5 \times 8^1 + 8 \times 8^0$
$= 384 + 40 + 8 = (432)_{10}$

Hexadecimal number system

In hexadecimal number system, there are 16 numbers 0 to 9, and digits from 10 to 15 are represented by A to F, respectively. The base of hexadecimal number system is 16.

Example 6: $(1A5C)_{16} = 1 \times 16^3 + A \times 16^2 + 5 \times 16^1 + C \times 16^0$
$= 1 \times 4096 + 10 \times 256 + 5 \times 16 + 12 \times 1$
$= 4096 + 2560 + 80 + 12 = (6748)_{10}$.

Table 1 *Different number systems*

Decimal	Binary	Octal	Hexadecimal
0	000	0	0
1	001	1	1
2	010	2	2
3	011	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

(*Continued*)

Table 1 (*Continued*)

Decimal	Binary	Octal	Hexadecimal
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

1. For a number system with base n, the number of different symbols in the number system will be n. Example: octal number system will have total of 8 numbers from 0 to 7.
2. The number 'n' in the number system with base 'n' is represented as $(10)_n$.
3. The equivalent of number $(a_3a_2a_1a_0 \cdot a_{-1}a_{-2})_n$ in decimal is $a_3 \times n^3 + a_2 \times n^2 + a_1 \times n^1 + a_0 \times n^0 + a_{-1} \times n^{-1} + a_{-2} \times n^{-2}$.

Conversion of Number Systems

The conversion of decimal to any other number system involves successive division by the radix until the dividend reaches 0. At each division, the remainder gives a digit of converted number; and the last one is most significant digit, the remainder of the first division is least significant digit.

The conversion of other number system to decimal involves multiplying each digit of number system with the weight of the position (in the power of radix) and sum the products calculated, the total is the equivalent value in decimal.

Decimal to binary conversion

Example 7: $(66)_{10}$

2	66	
2	33	0
2	16	1
2	8	0
2	4	0
	2	0
	1	0

Reading remainders from bottom to top

$= (1000010)_2$

Example 8: $(928)_{10}$

2	928	
2	464	0
2	232	0
2	116	0
2	58	0
2	29	0
2	14	1
2	7	0
2	3	1
	1	1

$= (1110100000)_2$

Example 9: $(105.75)_{10}$

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

$(105)_{10} = (1101001)_2$

$(0.75)_{10}$

Multiply 0.75 by 2 = 1.50

Multiply 0.50 by 2 = 1.00

Reading integers from top to bottom $0.75 = (0.11)_2$

$\therefore (105.75)_{10} = (1101001.11)_2$

Binary to decimal conversion

Example 10: $(10100011)_2$

$= 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

$= 128 + 0 + 32 + 0 + 0 + 0 + 2 + 1$

$= (163)_{10}$

Example 11: $(11010011.101)_2$

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

$= 128 + 64 + 0 + 16 + 0 + 0 + 2 + 1 + 0.5 + 0 + 0.125$

$= (211.625)_{10}$

Decimal to octal conversion

Example 12: $(16)_{10}$

8	16	0
	2	

Remainder from bottom to top $= (20)_8$

Example 13: $(347.93)_{10}$

$(.93)_{10}$

$0.93 \times 8 = 7.44$

$0.44 \times 8 = 3.52$

$0.52 \times 8 = 4.16$

$0.16 \times 8 = 1.28$

……..

Read the integers of octal point from top to bottom.

$\therefore (0.93)_{10} = (0.7341)_8$

$(347)_{10}$

8	347	3
8	43	3
	5	

$\therefore (347)_{10} = (533)_8$

Ans: $(533.7341)_8$

Octal to decimal conversion

Example 14: $(33)_8$

$3 \times 8^1 + 3 \times 8^0 = 24 + 3$

$(27)_{10}$

Example 15: $(1023.06)_8$

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

$= 512 + 0 + 16 + 3 + 0 + 0.0937 = (2095.0937)_{10}$

Octal to binary conversion

To convert octal to binary, replace each octal digit with their equivalent 3-bit binary representation.

Example 16: $(7777)_8$

Convert each octal digit to binary

$= \frac{7}{111}\ \frac{7}{111}\ \frac{7}{111}\ \frac{7}{111}$

$= (111\ \ 111\ \ 111\ \ 111)_2$

Example 17: $(567.62)_8$

5	6	7	.	6	2
101	110	111	.	110	010

$= (101110111.110010)_2$

Binary to octal conversion

To convert a binary number to an octal number, starting from the binary point, make groups of 3-bits each on either side of the binary point, and replace each 3-bit binary group by the equivalent octal digit.

Example 18: $(010011101)_2$

$\frac{010}{2}\ \frac{011}{3}\ \frac{101}{5} = (235)_8$

Example 19: $(100101110111.1011)_2$

$\frac{010}{2}\ \frac{010}{2}\ \frac{111}{7}\ \frac{011}{3} \cdot \frac{101}{5}\ \frac{100}{4} = (2273.54)_8$

Decimal to hexadecimal conversion

Example 20: $(527)_{10}$

$16 \mid 527$

$16 \mid 32 \quad 15$

$2 \quad 0$

Decimal		Hexa
2	→	2
0	→	0
15	→	F

$= (20F)_{16}$

Example 21: $(18.675)_{10}$

$(18)_{10}$

$16 \mid 18$

$1 \quad 2$

Decimal		Hexa
1	–	1
2	–	2

$(18)_{10} = (12)_{16}$

$(0.675)_{10}$

$0.675 \times 16 \mid 10.8$

$0.800 \times 16 \mid 12.8$

$0.800 \times 16 \mid 12.8$

$0.800 \times 16 \mid 12.8$

Decimal	Hexa
10	A
12	C
12	C
12	C

$= (0.ACCC)_{16}$

$\therefore$ Hexadecimal equivalent is

$= (12.AC\ CC)_{16}$

Hexadecimal to decimal conversion

Example 22: $(A3F)_{16}$

Decimal		Hexa
A	–	10
3	–	3
F	–	15

$\rightarrow 10 \times 16^2 + 3 \times 16^1 + 15 \times 16^0$

$\rightarrow 2560 + 48 + 15 \quad \rightarrow (2623)_{10}$

Example 23: $(1F63.0EB)_{16}$

1	1
F	15
6	6
3	3
0	0
E	14
B	11

$\rightarrow 1 \times 16^3 + 15 \times 16^2 + 6 \times 16^1 + 3 \times 16^0 \times (0 \times 16^{-1}) + (14 \times 16^{-2}) + (11 \times 16^{-3})$

$\rightarrow 4096 + 3840 + 96 + 3 + 0 + 0.0546 + 0.0026$

$\rightarrow (8035.0572)_{10}$

Hexadecimal to binary number system

To represent hexadecimal in binary, represent each HEX number with its 4-bit binary equivalent.

Example 24: $(34F)_{16}$

Hexa	Decimal	Binary
3	3	0011
4	4	0100
F	15	1111

$= (001101001111)_2$

Example 25: $(AFBC \cdot BED)_{16}$

Hexa	Decimal	Binary
A	10	1010
F	15	1111
B	11	1011
C	12	1100
B	11	1011
E	14	1110
D	13	1101

$= (1010111110111100.101111101101)_2$

Binary to hexadecimal number system

To convert binary number to a hexadecimal number, starting from the binary point, make groups of 4-bits each on either side of the binary point and replace each 4-bit group by the equivalent hexadecimal digit.

Example 26: $(11001001)_2$

$\rightarrow \frac{1100}{12}\ \frac{1001}{9}$

$\rightarrow (C9)_{16}$

Example 27: $(1011011011.01111)_2$

$$\frac{0010}{2}\ \frac{1101}{D}\ \frac{1011}{B} \cdot \frac{0111}{7}\ \frac{1000}{8} = (2\,DB.78)_{16}$$

Hexadecimal to octal number system

The simplest way to convert hexadecimal to octal is, first convert the given hexadecimal number to binary and the Binary number to Octal.

Example 28: $(C3AF)_{16}$

$\rightarrow 001100001110101111$

$\rightarrow (141657)_8$

Example 29: $(C6.AE)_{16}$

$\rightarrow 0011000110.10101110$

$\rightarrow (306.534)_8$

Octal to hexadecimal number system

The simplest way to convert octal to hexadecimal is first convert the given octal number to binary and then the binary number to hexadecimal.

Example 30: $(775)_8$

$\rightarrow (000111111101)_2$

$\rightarrow (1FD)_{16}$

Example 31: $(34.7)_8$

$\rightarrow (00011100.1110)_2$

$\rightarrow (1C.E)_{16}$

Complements

Complements are used in digital computers to simplify the subtraction operation and for logical manipulation.

There are two types of complements for each base - r-system.

1. Radix complement (or) r's complement: the r's complement of an m digit number N in base r is $r^m - N$ for $N \neq 0$.

   For example, $N = 0$, r's complement is 0.
2. Diminished radix complement: (or) $(r-1)$'s complement: Given a number N in base r having m digits, then $(r-1)$'s complement is $(r^m - 1) - N$.

   For example, decimal number system will have 10's complement and 9's complement.

   Similarly, binary number system will have 2's complement and 1's complement.

Example 32: 10's complement of $(2657)_{10}$ is $(10)^4 - 2657$

10000
<u>2657</u>
7343

Example 33: 9's complement of $(2657)_{10}$ is $(10^4 - 1) - 2657$

10000
<u>– 1</u>
9999
<u>2657</u>
7342

- r's complement can be obtained by adding 1 to $(r-1)$'s complement.

$$r^m - N = \{(r^m - 1) - N\} + 1$$

Example 34: 2's complement of $(101101)_2$ is

$= (2)^6 - 101101$

$(2^6)_{10} = (100000)_2$

2's complement is 100000

<u>–101101</u>
010011

Example 35: 1's complement of $(101101)_2$ is

$2^6 - 1 = 1000000$

<u>– 1</u>
111111
<u>101101</u>

1's complement –010010

The one's complement of a binary number is formed by changing 1's to 0's and 0's to 1's, The 2's complement can be formed by leaving all least significant 0's and the first 1 unchanged, and replacing 1's with zeros and zeros with 1's in all other bits.

If the number M contains radix point, the point should be removed temporarily in order to form r's/ $(r-1)$'s complement.

The radix point is then restored to the complemented number in the same relative position.

Example 36: What is 1's complement of $(1001.011)_2$?

$\rightarrow$ Consider without radix point 1001011

Take 1's complement 0110100

Place radix point again $(0110.100)_2$

Example 37: What is 2's complement of $(1001.011)_2$?

Consider without radix point 1001011

Take 2's complement 0110101

Place radix point again $(0110.101)_2$

Complement of a complement is equal to the original number $r^m - (r^m - M) = M$

Subtraction with Complements

Subtraction of two n digit unsigned numbers $A - B$ in base r can be done as follows by r's complement method.

Add A to the r's complement of B. Mathematically $A + (r^n - B) = A - B + r^n$

If $A \geq B$ the sum will produce an end carry r^n; which can be discarded. (Discarding carry is equivalent to subtracting r^n from result). What is left is the result $A - B$?

$A = 1100 \rightarrow$ 1100
$B = 1010$ (2's complement) + 0110
Sum: 10010
discard carry ($-r^n$) – 10000
$A - B$: 0010

If $A < B$, the sum does not produce an end carry and result is $r^n - (B - A)$. Then take r's complement of the sum, and place a negative sign in front.

If $A = 1010$
$B = 1100$
$A - B$ can be done as
$A \rightarrow$ 1010
$B \rightarrow$ 2's complaint + 0100
Sum: 1110

Here, no carry generated, so result is a negative number.
2's complement of result $\rightarrow 0010 = 2$
result = –2

Subtraction of unsigned numbers by using $(r - 1)$'s complement can be done in similar way. However, $(r - 1)$'s complement is one less than the r's complement. Because of this, the sum produced is one less than the correct difference when an end carry occurs. So end carry will be added to the sum. Removing the end carry and adding 1 to the sum is referred to as an end-around-carry.

Consider $A = 1100$, $B = 1010$
For $A - B$
$A \rightarrow$ 1100
$B \rightarrow$ (1's complement) + 0101
Sum: 10001
End around carry + 1
$A - B = 0010$

For $B - A$
$B \rightarrow$ 1010
$A \rightarrow$ (1's complement) + 0011
Sum: 1101

There is no end carry, for there result is
$-(B - A) = -$(1's complement of 1101)
$= -0010 = -2$

Signed Binary Numbers

Positive integers can be represented as unsigned numbers; but to represent negative integer, we need a notation for negative values in binary.

It is customary to represent the sign with a bit placed in the left most position of the number. The convention is to make the sign bit 0 for positive and 1 for negative. This representation of signed numbers is referred to as sign-magnitude convention

S Magnitude

+24 is 0 11000
sign magnitude
–24 is 1 11000
sign magnitude

Other notation for representation of signed numbers is signed complement system. This is convenient to use in a computer for arithmetic operations. In this system, a negative number is indicated by its complement (i.e., complement of corresponding positive number) whereas the sign-magnitude system negates a number by changing its sign bit, the signed-complement system negates a number by taking its complement. Positive numbers use same notation in sign-magnitude as well as sign-complement systems.

The signed-complement system can be used either as the 1's complement or the 2's complement.

But 2's complement is the most common.

+24 in 1's/2's complement representation is 011000
–24 in 1's complement representation 100111
–24 in 2's complement representation 101000

Table 2 *Signed binary numbers – (4-bits)*

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

The ranges of signed binary numbers with n-bits

Signed-magnitude: $-2^{n-1} + 1$ to $+2^{n-1} - 1$

1's complement representation: $-2^{n-1} + 1$ to $+2^{n-1} - 1$
2's complement representation: -2^{n-1} to $+2^{n-1} - 1$

Signed 2's complement representation can be directly used for arithmetic operations. The carryout of the sign bit position is discarded.

In order to obtain a correct answer, we must ensure that the result has a sufficient number of bits to accommodate the sum/product.

Example 38: $X = 00110$, $Y = 11100$ are represented in 5-bit signed 2's complement system

Then their sum $X + Y$ in 6-bit signed 2's complemented representation is?

Solution: $X = 00110$
$Y = 11100$
are 5-bit numbers
But result needs to be in 6-bit format.
Operands X and Y also should be in 6-bit format

$X =$ 000110
$Y =$ 111100
$X + Y = (1)\ 000010$

The carry out of sign bit position is discarded result is 000010.

Example 39: $(36x\ 70)_{10}$ is 10's complement of $(yzyz0)_{10}$ Then values of x, y, z are?
(A) 4, 5, 2 (B) 4, 6, 3
(C) 3, 6, 3 (D) 3, 5, 4

Solution: (C)
$(36x70)_{10}$ is 10's complement of $(yzyz0)_{10}$.
10's complement of $(yzyz0)_{10}$ is

$$10^5 - yzyz0 = 36 \times 70$$

So $36x70 + yzyz0 = 100000$

$$\begin{array}{r} 36x70 \\ +yzyz0 \\ \hline 100000 \end{array}$$

so $7 + z = 10$,
$1 + x + y = 10 \quad z = 3$
$1 + 6 + z = 10 \quad y = 6$
$1 + 3 + y = 10$,
$\rightarrow x = 3$

Example 40: The 10's complement of $(843)_{11}$ is?
(A) $(157)_{11}$ (B) $(267)_{11}$
(C) $(156)_{11}$ (D) $(268)_{11}$

Solution: (B)
Given $(843)_{11}$ is base 11 number system and the number in the number system range from 0 to 9 & A ($A = 10$)
10's complement for $(843)_{11}$ means $(r-1)$'s complement.
So $(r^n - 1) - N = [(11)^n - 1] - N$
$(11)^n - 1 \Rightarrow 1000$

$$\begin{array}{r} -\ \ 1 \\ \hline AAA \\ -843 \\ \hline 267 \end{array}$$

10's complement is $(267)_{11}$

Example 41: Consider the signed binary number to be 10111011. What is the decimal equivalent of this number if it is in Sign-Magnitude form, or 1's complement form, or 2's complement form?

Solution: Given binary number = 10111011. As sign bit is 1, it is a negative number. If it is in sign-magnitude format, then MSB is sign bit, and remaining bits represent the magnitude,
$(0111011)_2 = 32 + 16 + 8 + 2 + 1 = 59$. So if the given number is in sign-magnitude format, then the number is –59.

If it is in 1's complement/2's complement form, then the magnitude of negative number can be obtained by taking 1's complement/2's complement for the number, respectively.
$10111011 \Rightarrow$ 1's complement $\Rightarrow 01000100 = (68)_{10}$.
In 1's complement format, the number is –68.
$10111011 \Rightarrow$ 2's complement $\Rightarrow 01000101 = (69)_{10}$.
In 2's complement format, the number is –69.

Example: Find $(-9.625)_{10}$ in signed 2's complement representation.
Signed binary fraction can be represented in the same way of signed integer.

$$\begin{array}{r|l} 2 & 9 \\ 2 & 4 - 1 \\ 2 & 2 - 0 \\ & 1 - 0 \end{array}$$

$0.625 \times 2 = 1.25$
$0.25 \times 2 = 0.5$
$0.5 \times 2 = 1.0$
$= 0.101$
$+(9.625) = 01001.101$
$-9.625 = 10110.011$ (by taking 2's complement)

Binary Multipliers

Multiplication of binary number is done in the same way as multiplication of decimal.
The multiplicand (m) is multiplied by each bit of the multiplier (N), starting from the LSB.
Let
$M = B_3\ B_2\ B_1\ B_0$
$N = A_3\ A_2\ A_1\ A_0$
If $M \times N = P$

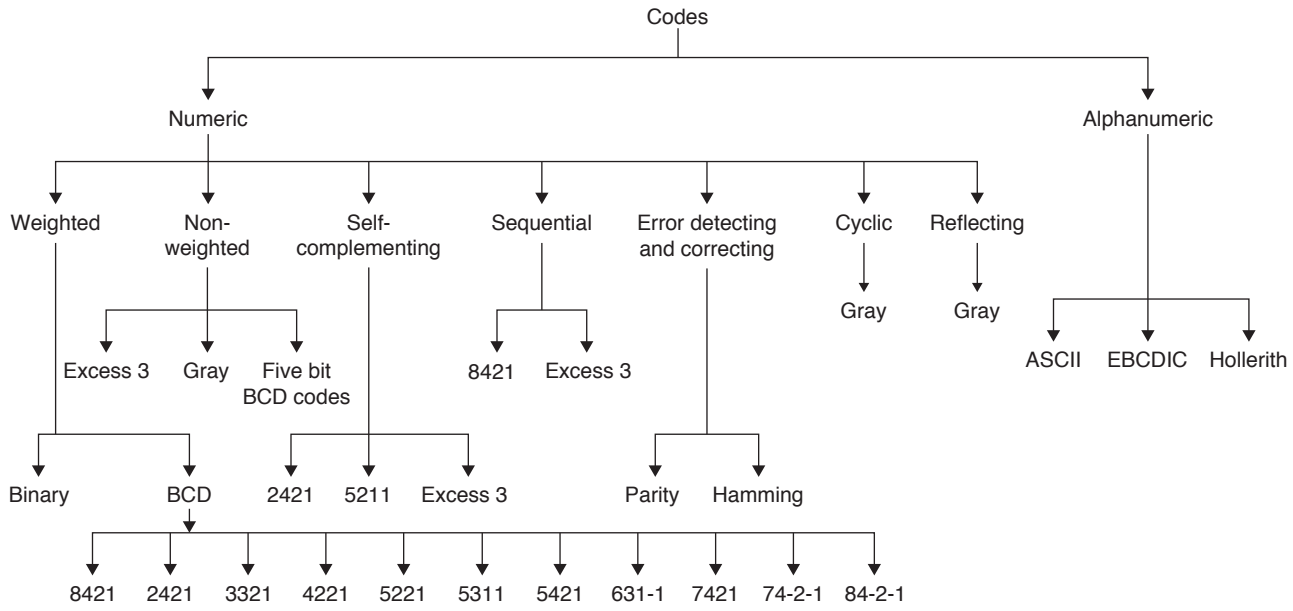
			A_0B_3	A_0B_2	A_0B_1	A_0B_0	
		A_1B_3	A_1B_2	A_1B_1	A_1B_0		
	A_2B_3	A_2B_2	A_2B_1	A_2B_0			
A_3B_3	A_3B_2	A_3B_1	A_3B_0				
P_7P_6	P_5	P_4	P_3	P_2	P_1	P_0	$=P$

Example: Let $M = 1\ 0\ 1\ 1$
$N = 1\ 1\ 0\ 0$
$M \times N = P$

$$\begin{array}{r} 1\ 0\ 1\ 1 \\ 1\ 1\ 0\ 0 \quad \times \\ \hline 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0 \\ 1\ 0\ 1\ 1 \\ 1\ 0\ 1\ 1 \\ 1\ 1\ 1 \\ \hline 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0 = P \end{array}$$

Binary Codes

Binary codes can be classified as numeric codes and alphanumeric codes. The figure below shows the classification of codes.



Numeric Codes

Numeric Codes are the codes which represent numericals in binary, i.e., only numbers as a series of 0s and 1s.

Weighted and non-weighted codes

- The weighted codes are those which obey the position-weighting principle. Each position of a number represents a specific weight.
  Example: BCD, Binary, 84-2-1, 2421,
- Non-weighted codes are codes which are not assigned fixed values.
  Example: Excess-3, Gray code
  2421, 5211, 84 – 2 – 1 are examples of weighted codes, in which weight is assigned to each position in the number.
  $(27)_{10}$ in 2421 code → 0010 1101
  $(45)_{10}$ in 5211 code → 0111 1000
  $(36)_{10}$ in 84 – 2 – 1 code → 0101 1010
  Any digit in decimal will be represented by the weights represented by the code.

Error-detecting and correcting codes

Codes which allow only error detection are error-detecting codes.

Example: Parity

Codes which allow error detection as well as correction are called error correcting codes.

Example: Hamming codes

Sequential codes

A sequential code is one in which each succeeding code word is one binary number greater than the preceding code word.

Example: XS–3, BCD

Cyclic codes (unit distance codes)

Cyclic codes are those in which each successive code word differs from the preceding one in only one bit position.

Example: Gray code

Reflective codes

Binary code in which the n least significant bits for code words 2^n through $2^{n+1} - 1$ are the mirror images of than for 0 through $2^n - 1$ is called reflective codes.

Example: Gray Code

Self-complementing codes

A code is said to be self-complementing, if the code word of the 9's complement of number 'N', i.e., of "9-N" can be obtained from the code word of 'N' by interchanging all the zeros and ones, i.e., by taking 1's complement. In other words, logical complement of number code is equivalent to representation of its arithmetic complement.

Example: 84-2-1, 2421, XS -3.

All weighted BCD codes are self-complementing codes.

Binary-coded decimal (BCD)

In BCD, each decimal digit 0 to 9 is coded by a 4-bit binary number. BCD codes are convenient to convert to/or from decimal number system.

Decimal	BCD Digit
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Example 42: $(628)_{10} = (0110\ 0010\ 1000)_{BCD}$

BCD addition

- BCD addition is performed by individually adding the corresponding digits of the decimal number expressed in 4-bit binary groups starting from the LSB.
- If there is no carry and the sum term is not an illegal code, no correction is needed.
- If there is a carry out of one group to the next group or if the sum term is an illegal code, the $(6)_{10}$ is added to the sum term of that group, and the resulting carry is added to the next group.

Example 43: 44 + 12

0100	0100	(44 in BCD)
0001	0010	(12 in BCD)
0101	0110	(56 in BCD)

Example 44: 76.9+ 56.6

0111	0110 . 1001		
0101	0110 . 0110		
1100	1100 . 1111		(all are illegal codes)
0110	0110 . 0110		
0010	0010 . 0101		
+1	+1	+1	(propagate carry)
0001	0011	0011 . 0101	
1	3	3 . 5	

BCD subtraction BCD subtraction is performed by subtracting the digits of each 4-bit group of the subtrahend from the corresponding 4-bit group of the minuend in binary starting from the LSB.

Example 45:

42	0100	0010	(42 in BCD)
−12	−0001	0010	(12 IN BCD)
30	0011	0000	(No borrow, so this is the correct difference)

Example 46:

247.7	0010	0100	0111 .	0111	(Borrow are present, subtract 0110)
−156.9	0001	0101	0110 .	1001	
90.8	0000	0111	·0000 .	1110	
		−01001	−0110		
		1001	000 ·	1000	Corrected difference (90.8)

Excess-3 (XS-3) code

Excess-3 code is a non-weighted BCD code, where each digit binary code word is the corresponding 8421 code word plus 0011.

Find the XS-3 code of

Example 47: $(3)_{10} \rightarrow (0011)_{BCD} = (0110)_{xS3}$

Example 48: $(16)_{10} \rightarrow (0001\ 0110)_{BCD}$
$\rightarrow (0100\ 1001)_{xS3}$

Gray code

Each gray code number differs from the preceding number by a single bit.

Decimal	Gray Code
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111

Binary to gray conversion

Step I: Shift the binary number one position to the right, LSB of the shifted number is discarded.

Step II: Exclusive or the bits of the binary number with those of the binary number shifted.

Example 49: Convert $(1001)_2$ to gray code

Binary $\rightarrow$ 1010
Shifted Binary $\rightarrow$ 101 $\oplus$
Gray $\rightarrow$ 1111

Gray to binary conversion

(a) Take the MSB of the binary number is same as MSB of gray code number.
(b) X-OR the MSB of the binary to the next significant bit of the gray code.
(c) X-OR the 2nd bit of binary to the 3rd bit of Gray code to get 3rd bit binary and so on.
(d) Continue this till all the gray bits are exhausted.

Example 50: Convert, gray code 1010 to binary

Gray	1	0	1	0
1010	⇓	⊘ ‖	⊘ ‖	⊘ ‖
1100	1	1	0	0
$= (1100)_2$				

Exercises

Practice Problems I

Directions for questions 1 to 15: Select the correct alternative from the given choices.

1. Assuming all the numbers are in 2's complement representation, which of the following is divisible by 11110110?
(A) 11101010 (B) 11100010
(C) 11111010 (D) 11100111

2. If $(84)_x$ (in base x number system) is equal to $(64)_y$ (in base y number system), then possible values of x and y are
(A) 12, 9 (B) 6, 8
(C) 9, 12 (D) 12, 18

3. Let A = 1111 1011 and B = 0000 1011 be two 8-bit signed 2's complement numbers. Their product in 2's complement representation is

(A) 11001001 (B) 10011100
(C) 11010101 (D) 10101101

4. Let r denotes number system's radix. The only value(s) of r that satisfy the equation $\sqrt[3]{(1331)_r} = (11)_r$ is/are
(A) 10 (B) 11
(C) 10 and 11 (D) any $r > 3$

5. X is 16-bit signed number. The 2's complement representation of X is $(F76A)_{16}$. The 2's complement representation of $8 \times X$ is
(A) $(1460)_{16}$ (B) $(D643)_{16}$
(C) $(4460)_{16}$ (D) $(BB50)_{16}$

6. The HEX number $(CD.EF)_{16}$ in octal number system is
(A) $(315.736)_8$ (B) $(513.637)_8$
(C) $(135.673)_8$ (D) $(531.367)_8$

7. 8-bit 2's complement representation a decimal number is 10000000. The number in decimal is
(A) +256 (B) 0
(C) −128 (D) −256

8. The range of signed decimal numbers that can be represented by 7-bit 1's complement representation is
(A) −64 to + 63 (B) −63 to + 63
(C) −127 to + 128 (D) −128 to +127

9. Decimal 54 in hexadecimal and BCD number system is respectively
(A) 63, 10000111 (B) 36,01010100
(C) 66, 01010100 (D) 36, 00110110

10. A new binary-coded hextary (BCH) number system is proposed in which every digit of a base −6 number system is represented by its corresponding 3-bit binary code. For example, the base −6 number 35 will be represented by its BCH code 011101.

In this numbering system, the BCH code 001001101011 corresponds to the following number in base −6 system.
(A) 4651 (B) 4562
(C) 1153 (D) 1353

11. The signed 2's complement representation of $(-589)_{10}$ in Hexadecimal number system is
(A) $(F24D)_{16}$ (B) $(FDB3)_{16}$
(C) $(F42D)_{16}$ (D) $(F3BD)_{16}$

12. The base of the number system for which the following operation is to be correct $\frac{66}{5} = 13$
(A) 6 (B) 7
(C) 8 (D) 9

13. The solution to the quadratic equation $x^2 - 11x + 13 = 0$ (in number system with radix r) are $x = 2$ and $x = 4$. Then base of the number system is (r) =
(A) 7 (B) 6
(C) 5 (D) 4

14. The 16's complement of BADA is
(A) 4525 (B) 4526
(C) ADAB (D) 2141

15. $(11A1B)_8 = (12CD)_{16}$, in the above expression A and B represent positive digits in octal number system and C and D have their original meaning in Hexadecimal, the values of A and B are?
(A) 2, 5 (B) 2, 3
(C) 3, 2 (D) 3, 5

Practice Problems 2

Directions for questions 1 to 20: Select the correct alternative from the given choices.

1. The hexadecimal representation of $(567)_8$ is
(A) 1AF (B) D77
(C) 177 (D) 133

2. $(2326)_8$ is equivalent to
(A) $(14D6)_{16}$ (B) $(103112)_4$
(C) $(1283)_{10}$ (D) $(09AC)_{16}$

3. $(0.46)_8$ equivalent in decimal is?
(A) 0.59375 (B) 0.3534
(C) 0.57395 (D) 0.3435

4. The 15's complement of $(CAFA)_{16}$ is
(A) $(2051)_{16}$ (B) $(2050)_{16}$
(C) $(3506)_{16}$ (D) $(3505)_{16}$

5. 53 in 2's complement from is?
(A) 1001011 (B) 001010
(C) 0110101 (D) 001011

6. Signed 2's complement representation of $(-15)_{10}$ is
(A) 11111 (B) 10001
(C) 01111 (D) 10000

7. $(0.25)_{10}$ in binary number system is?
(A) (0.01) (B) (0.11)
(C) 0.001 (D) 0.101

8. The equivalent of $(25)_6$ in number system with base 7 is?
(A) 22 (B) 23
(C) 24 (D) 26

9. The operation 35 + 26 = 63 is true in number system with radix
(A) 7 (B) 8
(C) 9 (D) 11

10. The hexadecimal equivalent of largest binary number with 14-bits is?
(A) 2FFF (B) 3FFFF
(C) FFFF (D) 1FFFF

11. If x is radix of number system, $(211)_x = (152)_8$, then x is
(A) 6 (B) 7
(C) 9 (D) 5

12. The value of r for which $\sqrt{(224)_r} = (13)_r$ is valid expression, in number system with radix r is?
(A) 5 (B) 6
(C) 7 (D) 8

13. Which of the representation in binary arithmetic has a unique zero?
(A) sign-magnitude (B) 1's compliment
(C) 2's complement (D) All of these

14. For the binary number 101101111 the equivalent hexadecimal number is
(A) 14E (B) 9E
(C) B78 (D) 16F

15. Subtract 1001 from 1110
(A) 0010 (B) 0101
(C) 1011 (D) 1010

16. Which of the following is a positively weighted code?
(A) 8421 (B) 84-2-1
(C) EXS-3 (D) 74-2-1

17. Match the items correctly

Column 1	Column 2
(P) 8421	(1) Cyclic code
(Q) 2421	(2) self-complementing
(R) 5212	(3) sequential code
(S) Gray code	(4) non-sequential code

(A) P–2, Q–4, R–3, S–1
(B) P–1, Q–4, R–3, S–2
(C) P–3, Q–2, R–4, S–1
(D) P–2, Q–4, R–1, S–2

18. Perform the subtraction in XS-3 code 57.6 – 27.8
(A) 0101 1100.1011 (B) 0010 1001.1100
(C) 00011101.1100 (D) 1010 1110.1011

19. The 2's complement representation of –17 is
(A) 101110 (B) 111110
(C) 101111 (D) 110001

20. The decimal 398 is represented in 2421 code by
(A) 110000001000 (B) 001110011000
(C) 001111111110 (D) 010110110010

Previous Years' Questions

1. $(1217)_8$ is equivalent to **[2009]**
(A) $(1217)_{16}$ (B) $(028F)_{16}$
(C) $(2297)_{10}$ (D) $(0B17)_{16}$

2. P is a 16-bit signed integer. The 2's complement representation of P is $(F87B)_{16}$. The 2's complement representation of $8*P$ is **[2010]**
(A) $(C3D8)_{16}$ (B) $(187B)_{16}$
(C) $(F878)_{16}$ (D) $(987B)_{16}$

3. The smallest integer that can be represented by an 8-bit number in 2's complement form is **[2013]**
(A) –256 (B) –128
(C) –127 (D) 0

4. The base (or radix) of the number system such that the following equation holds is ——— $\frac{312}{20} = 13.1$ **[2014]**

5. Consider the equation $(123)_5 = (x8)_y$ with x and y as unknown. The number of possible solutions is ———. **[2014]**

6. Consider the equation $(43)_x = (y3)_8$ where x and y are unknown. The number of possible solutions is _____ **[2015]**

7. Suppose X_i for $i = 1, 2, 3$ are independent and identically distributed random variables whose probability mass functions are $Pr[X_i = 0] = Pr[X_i = 1] = ½$ for $i = 1, 2, 3$. Define another random variable $Y = X_1 X_2 \oplus X_3$, where $\oplus$ denotes XOR. Then

$Pr[Y = 0 | X_3 = 0] =$ _____ **[2015]**

8. The 16-bit 2's complement representation of an integer is 1111 1111 1111 0101; its decimal representation is ___ . **[2016]**

9. Consider an eight - bit ripple - carry adder for computing the sum of A and B, where A and B are integers represented in 2's complement form. If the decimal value of A is one, the decimal value of B that leads to the longest latency for the sum to stabilize is _____ . **[2016]**

10. Let $x_1 \oplus x_2 \oplus x_3 \oplus x_4 = 0$ where x_1, x_2, x_3, x_4 are Boolean variables, and $\oplus$ is the XOR operator. Which one of the following must always be **TRUE**? **[2016]**
(A) $x_1 x_2 x_3 x_4 = 0$
(B) $x_1 x_3 + x_2 = 0$
(C) $\bar{x}_1 \oplus \bar{x}_3 = \bar{x}_3 \oplus \bar{x}_4$
(D) $x_1 + x_2 + x_3 + x_4 = 0$

11. Consider a quadratic equation $x^2 - 13x + 36 = 0$ with coefficients in a base b. The solutions of this equation in the same base b are $x = 5$ and $x = 6$. Then $b =$ _____. **[2017]**

Answer Keys

Exercises

Practice Problems 1

1. B **2.** C **3.** A **4.** D **5.** D **6.** A **7.** C **8.** B **9.** B **10.** C
11. B **12.** D **13.** C **14.** B **15.** D

Practice Problems 2

1. C **2.** B **3.** A **4.** D **5.** D **6.** B **7.** A **8.** B **9.** B **10.** B
11. B **12.** A **13.** C **14.** D **15.** B **16.** A **17.** C **18.** A **19.** C **20.** C

Previous Years' Questions

1. B **2.** A **3.** B **4.** 5 **5.** 3 **6.** 5 **7.** 0.75 **8.** –11 **9.** –1.0 **10.** C
11. 8.0 to 8.0

Chapter 2

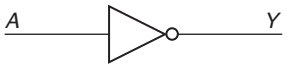
Boolean Algebra and Minimization of Functions

LEARNING OBJECTIVES

- *Logic gates*
- *Boolean algebra*
- *AXIOMS and Laws of Boolean algebra*
- *Properties of Boolean algebra*
- *Conversion from Min term to Max term*
- *Minimization of Boolean function*
- *K-map method*
- *Prime implicant*
- *Implementation of function by using NAND-NOR Gates*
- *EX-OR, EX-NOR GATE*

Logic Gates

1. **Inverter or NOT gate (7404 IC):** The inverter performs a basic logic operation called inversion or complementation. The purpose of the inverter is to change one logic level to the opposite level. In terms of digital circuits, it converts 1 to 0 and 0 to 1.



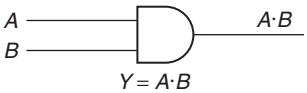
Symbol

$Y = \overline{A}$

Table 1 *Truth Table*

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

2. **AND gate (logical multiplier 7408 IC):** The AND gate performs logical multiplication more commonly know as AND function. The AND gate is composed of 2 or more inputs and a single output



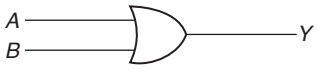
$Y = A \cdot B$

Figure 1 2 input AND gate

Table 2 *Truth Table*

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

3. **OR gate (logical adder 7432 IC):** The OR gate performs logical, addition commonly known as OR function.



Symbol

$Y = A + B$

Figure 2 2 input OR gate

Table 3 *Truth Table*

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

4. **NAND gate (7400 IC):** The NAND gate's function is basically AND + NOT function.

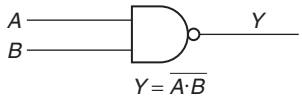


Figure 3 2 input NAND gate

Table 4 *Truth Table*

Input			Output
A	**B**	**A·B**	$\overline{A} + \overline{B}$ **(Y)**
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

5. **NOR gate (7402 IC):** The NOR gate is basically OR + NOT function.

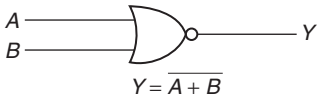


Figure 4 2 input NOR gate

Table 5 *Truth Table*

Input			Output
A	**B**	**A + B**	$\overline{A} + \overline{B}$ **(Y)**
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

6. **Exclusive OR gate X-OR (7486 IC):** X-OR is a gate in which unequal inputs create a high logic level output and if both inputs are equal, the output will be low. Other name for EX-OR gate is unequivalent gate.
2 input X-OR Gate

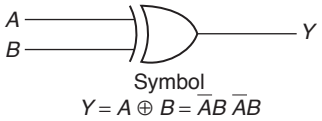


Figure 5 2 input X-OR Gate

Table 6 *Truth Table*

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

7. **Exclusive NOR gate (X-NOR):** X-NOR is a gate in which equal inputs create a high logic level output; and if both inputs are unequal, then the output will be low. Other name for X-NOR gate is equivalent gate.

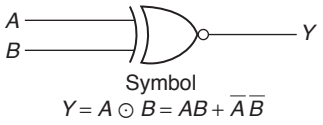


Figure 6 2 input X-NOR Gate

Table 7 *Truth Table*

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

X-NOR Gate is complement of X-OR Gate.

Boolean Algebra

Boolean algebra is a system of mathematical logic. It is an algebraic system consisting of the set of elements (0, 1), two binary operators OR and AND and one unary operator NOT. The Boolean algebra is governed by certain well-developed rules and laws.

AXIOMS and Laws of Boolean Algebra

1. **AXIOMS**
   - **(a) AND operation**
     - (1) $0 \cdot 0 = 0$
     - (2) $0 \cdot 1 = 0$
     - (3) $1 \cdot 0 = 0$
     - (4) $1 \cdot 1 = 1$
   - **(b) OR operation**
     - (5) $0 + 0 = 0$
     - (6) $0 + 1 = 1$
     - (7) $1 + 0 = 1$
     - (8) $1 + 1 = 1$
   - **(c) NOT operation**
     - (9) $\overline{1} = 0$
     - (10) $\overline{0} = 1$
2. **Laws**
   - **(a) Complementation law**
     - (1) $\overline{0} = 1$
     - (2) $\overline{1} = 0$
     - (3) If $A = 0$, then $\overline{A} = 1$
     - (4) If $A = 1$, then $\overline{A} = 0$
     - (5) $\overline{\overline{A}} = A$
   - **(b) AND laws**
     - (1) $A \cdot 0 = 0$ (NULL Law)
     - (4) $A \cdot 1 = A$ (Identity Law)
     - (3) $A \cdot A = A$
     - (4) $A \cdot \overline{A} = 0$

(c) OR laws

(1) $A + 0 = A$ (NULL Law)

(2) $A + 1 = 1$ (Identity Law)

(3) $A + A = A$

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

(d) Commutative laws

(1) $A + B = B + A$

(2) $A \cdot B = B \cdot A$

(e) Associative laws

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

(2) $(A \cdot B)C = A(B \cdot C)$

(f) Distributive laws

(1) $A(B + C) = AB + AC$

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

(g) Redundant literal rule (RLR)

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

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

(h) Idempotence laws

(1) $A \cdot A = A$

(2) $A + A = A$

(i) Absorption laws

(1) $A + A \cdot B = A$

(2) $A(A + B) = A$

3. Theorems

(a) Consensus theorem

Theorem 1:

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

Proof:

$$\begin{aligned}\text{LHS} &= AB + \overline{A}C + BC \\ &= AB + \overline{A}C + BC(A + \overline{A}) \\ &= AB + \overline{A}C + BCA + BC\overline{A} \\ &= AB(1 + C) + \overline{A}C(1 + B) \\ &= AB(1) + \overline{A}C(1) \\ &= AB + \overline{A}C \\ &= \text{RHS.}\end{aligned}$$

Theorem 2:

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

Proof:

$$\begin{aligned}\text{LHS} &= (A + B)(\overline{A} + C)(B + C) \\ &= (A\overline{A} + AC + B\overline{A} + BC)(B + C) \\ &= (AC + BC + \overline{A}B)(B + C) \\ &= ABC + BC + \overline{A}B + AC + BC + \overline{A}BC \\ &= AC + BC + \overline{A}B\end{aligned}$$

$$\begin{aligned}\text{RHS} &= (A + B)(\overline{A} + C) \\ &= A\overline{A} + AC + BC + \overline{A}B \\ &= AC + BC + \overline{A}B \\ &= \text{LHS.}\end{aligned}$$

(b) Transposition theorem

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

Proof:

$$\begin{aligned}\text{RHS} &= (A + C)(\overline{A} + B) \\ &= A\overline{A} + C\overline{A} + AB + CB \\ &= 0 + \overline{A}C + AB + BC \\ &= \overline{A}C + AB + BC(A + \overline{A}) \\ &= AB + ABC + \overline{A}C + \overline{A}BC \\ &= AB + \overline{A}C \\ &= \text{LHS}\end{aligned}$$

(c) De Morgan's theorem

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

This law states that the complement of a sum of variable is equal to the product of their individual complements.

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

This law states that the complement of the product of variables is equal to the sum of their individual complements.

Example 1: Simplify the Boolean function $Y = A(A + \overline{B})$

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

Solution:

$$\begin{aligned}Y &= A + A\overline{B} \\ &= A(1 + \overline{B}) \\ &= A\end{aligned}$$

Example 2: Simplify the Boolean function $Y = A + \overline{A}B$

Solution:

$$\begin{aligned}Y &= A \cdot (B + 1) + \overline{A} \cdot B \\ &= A \cdot B + A + \overline{A}B \\ &= B(A + \overline{A}) + A \\ &= A + B\end{aligned}$$

Example 3: Simplify the Boolean function

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

Solution:

$$\begin{aligned}Y &= A \cdot A + A \cdot B + B \cdot \overline{A} + B \cdot B \\ &= A + B(A + \overline{A}) + B \\ &= A + B \cdot 1 + B \\ &= A + B + B \\ &= A + B\end{aligned}$$

Example 4: Simplify the Boolean function

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

Solution:

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

Example 5: Simplify the Boolean function

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

Solution:

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

Example 6: Simplify the Boolean function

$$Y = AB + C\bar{B} + CA + ABD$$

Solution: $Y = AB(1+D) + C\bar{B} + CA$

$= AB + C\bar{B} + CA$

$= AB + C\bar{B}$

Properties of Boolean Algebra

With n variables, maximum possible distinct functions $= 2^{2^n}$.

Duality consider the distributive law

1. $x\,(y+z) = xy + xz$
2. $x + yz = (x+y)\,(x+z)$

Second one can be obtained from the first law if the binary operators and the identity elements are interchanged. This important property of Boolean algebra is called the duality principle.

The dual of an algebraic expression can be written by interchanging OR and AND operators, 1s by 0, and 0s by 1s.

Example 7: $x + x' = 1 \xleftrightarrow{\text{Dual}} x \cdot x' = 0$

Solution: $xy + xy' = x \xleftrightarrow{\text{Dual}} (x+y)\,(x+y') = x$

$$x + x'y = x + y \xleftrightarrow{\text{Dual}} x(x' + y) = xy$$

Example 8: The dual of $F = xy + xz + yz$ is?

Solution: Dual of $F = (x+y)\,(x+z)\,(y+z)$

$= (x + xz + xy + yz)\,(y+z) = xy + yz + xz$

So dual of $xy + xz + yz$ is same as the function itself; For N variables maximum possible self-dual functions $= 2^{2^{n-1}} = 2^{(2^n/2)}$

Example 9: Which of the following statement/*s* is/are true

S_1: The dual of NAND function is NOR

S_2: The dual of X-OR function is X-NOR

(A) S_1 and S_2 are true

(B) S_1 is true

(C) S_2 is true

(D) None of these

Solution: (A)

NAND $= (xy)' = x' + y'$

Dual of NAND $= (x+y)' = x'y'$

X-OR $= xy' + x'y$

Dual of X-OR $= (x+y')\,(x'+y) = xy + x'y' =$ X-NOR

Both S_1 and S_2 are true

Operator precedence The operator precedence for evaluating Boolean expression is

1. Parentheses
2. NOT
3. AND
4. OR

So the expression inside the parentheses must be evaluated before all the operations. The next operation to be performed is the complement and then follows AND and finally the OR.

Complement of function The complement of a function F is F' is obtained from an interchange of 0s for 1s and 1s for 0s in the value of F. The complements of a function may be derived algebraically through De Morgan's theorems.

$$(x_1 \,.\, x_2 \,.\, x_3 \ldots x_n)' = x_1' + x_2' + x_3' + \cdots + x_n'$$
$$(x_1 + x_2 + x_3 + \cdots + x_n)' = x'_1 \,.\, x_2' \,.\, x'_3 \,.\, x'_4 \ldots x_n'$$

Example 10: The complement of function $F = a(b'c + bc')$ is?

Solution: $(F)' = [a(b'c + bc')]'$

$= a' + (b'c + bc')'$

$= a' + (b'c)' \cdot (bc')'$

$= a' + (b + c)'\,(b' + c)$

$F' = a' + bc + b'c'$

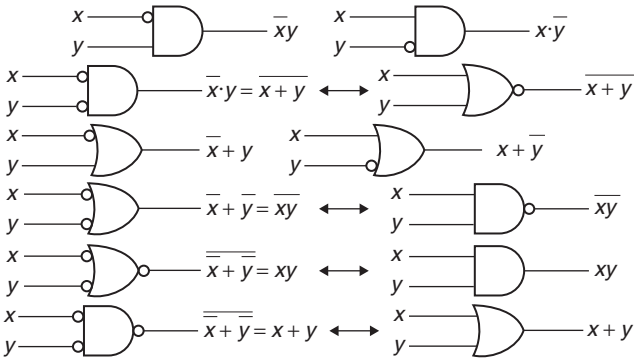


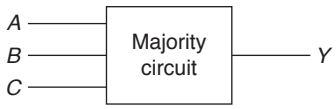
Figure 7 Gates with inverted inputs

Boolean Functions, Min Terms and Max Terms

The starting point for designing most logic circuits is the truth table, which can be derived from the statement of problem. The truth table is then converted into a Boolean expression and finally create the assembly of logic gates accordingly.

Let us consider the example of majority circuit. This circuit takes three inputs (A, B, C) and have one output (Y) which will give the majority of the inputs, i.e., if A, B, C are having more number of zeros. $Y = 0$ else if A, B, C are having more number of 1s, $Y = 1$.

So from the statements we can derive the truth table as follows:



As we are using three Boolean variables A, B, C, total number of combinations in truth table are $2^3 = 8$.

Similarly for n variables, the truth table will have total of 2^n combinations, for a Boolean function.

Sl. no.	Input A	B	C	Output Y	
1	0	0	0	0 ⟶	$Y = 0$, If inputs are having more zeros.
2	0	0	1	0	
3	0	1	0	0	
4	0	1	1	1 ⟶	$Y = 1$, If inputs are having more 1's
5	1	0	0	0	
6	1	0	1	1	
7	1	1	0	1	
8	1	1	1	1	

For some combinations, output $Y = 1$, and for others $Y = 0$. The input combinations for which output $Y = 1$ are called as min terms.

Similarly the input combinations for which output $Y = 0$ are called as max terms.

Min terms are expressed as product terms, Similarly, max terms are expressed as sum terms.

The output $Y = 1$, only in rows 4, 6, 7, 8.

So the min terms combinations are 011, 101, 110, 111 in Boolean Algebra, 1 input will be written as A, B, C and 0 input will be written as $\overline{A}, \overline{B}, \overline{C}$ in complement form, we express these min terms as product terms, $\overline{A}BC, A\overline{B}C, AB\overline{C}, ABC$.

To express Y as Boolean expression, we can write it as sum of the min terms.

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

We know that AND operation is a product while OR is sum. So the above equation is a sum of the products (SOP), (or) min terms expression.

The other way of expressing Y is $Y = \Sigma m\ (3, 5, 6, 7)$.

$$Y = m_3 + m_5 + m_6 + m_7.$$

The min term numbers are the decimal equivalent of input binary combinations.

Similar to SOP we can have product of sums (POS) Boolean expression.

The output $Y = 0$ for the input combinations 000, 001, 010, 100. For max terms 1 input will be indicated as $\overline{A}, \overline{B}, \overline{C}$ in complement form, 0 input will be indicated as A, B, C and max terms are expressed as sum terms.

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

Any function can be expressed as product of max terms.

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

The above equation is a product of sum expression (POS) or max terms expression.

In other way $Y = \pi M\ (0, 1, 2, 4)$

$$= M_0 \cdot M_1 \cdot M_2 \cdot M_4$$

The max term numbers are decimal equivalents of corresponding input binary combinations.

Min Term and Max Term

All the Boolean expressions can be expressed in a standard sum of product (SOP) form or in a standard product of sum (POS) form.

- A standard SOP form is one in which a number of product terms, each contains all the variables of the function either in complement or non-complement form are summed together.
- A standard POS form is one in which a number of sum terms, each one of which contain all the variable of he function either in complemented or non-complemented form are multiplied together.
- Each of the product term in standard SOP form is called a min term.
- Each of the sum term in the standard POS form is called a max term.

Conversion from min terms to max terms representation

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

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

$$= (\overline{A}BC)'(A\overline{B}C)'(AB\overline{C})'(ABC)'$$

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

$$= [\pi(3, 5, 6, 7)]^1 = \pi(0, 1, 2, 4)$$

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

$$\text{or } Y = \Sigma(3, 5, 6, 7) = \pi(0, 1, 2, 4)$$

Conversion from normal SOP/POS form to canonical SOP/POS

Let us consider $f(A, B, C) = A + BC + \overline{A}C$

The above function is in normal (minimized) SOP from, to convert this function to standard SOP(or) canonical SOP form, include missing variable in each and every term, to make it complete. First term A, Missing literals are B, C. Consider $A\ X\ X$, so possible combinations are $\overline{A}BC, A\overline{B}C, AB\overline{C}, ABC$ or we can write

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

Second term BC-missing literal is A. Consider $XBC \Rightarrow$ So possible combinations are ABC, $\overline{A}BC$ or we can write

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

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

Third term $\overline{A}C$ = missing literal is B. Consider $\overline{A}XC \rightarrow$ so possible combinations are $\overline{A}BC, \overline{A}, \overline{B}C$ or we can write

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

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

Now, $f(A,B,C) = ABC + AB\overline{C} + A\overline{B}C + \overline{A}BC + \overline{A}\overline{B}C$, after removing the redundant terms.

Now consider

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

To convert this expression to canonical form or standard POS form we can write

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

Here the C variables is absent from first term and B from second term. So add $C\cdot\overline{C} = (0)$ to first, and $B\cdot\overline{B}$ to second, and using distributive law arrive at the result.

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

Minimization of Boolean Functions

Simplification Procedure

- Obtain truth table, and write output in canonical form or standard form
- Generate K-map!
- Determine Prime implicants.
- Find minimal set of prime implicants.

Karnaugh Map (K-map) Method

Karnaugh map method is a systematic method of simplifying the Boolean expression. K-map is a chart or a graph composed of an arrangement of adjacent cell, each representing a particular combination of variable in sum or product form. (i.e., min term or max term).

Two-variable K-map

x	y	F
0	0	m_0
0	1	m_1
1	0	m_2
1	1	m_3

m_0	m_1
m_2	m_3

x \ y	0	1
0	$x'y'$	$x'y$
1	xy'	xy

Three-variable K-map

A three-variable map will have eight min terms (for three variables $2^3 = 8$) represented by 8 squares

x	y	z	F
0	0	0	m_0
0	0	1	m_1
0	1	0	m_2
0	1	1	m_3
1	0	0	m_4
1	0	1	m_5
1	1	0	m_6
1	1	1	m_7

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

x \ yz	00	01	11	10
0	$x'y'z$	$x'y'z$	$x'yz$	$x'yz'$
1	$xy'z'$	$xy'z$	xyz	xyz'

3-variable K-map

Four-variable K-maps

The K-map for four variables is shown here, 16 min terms are assigned to 16 squares.

wx \ yz	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

The map is considered to lie on a surface with the top and bottom edges as well as the right and left edge touching each other to form adjacent squares.

- One square → a min term of four literals
- Two adjacent square → a term of three literals
- Four adjacent square → a term of two literals
- Eight adjacent square → a term of one literal
- Sixteen adjacent square → The constant one

Don't-care Combinations

It can often occur that for certain input combinations, the value of the output is unspecified either because the input combination are invalid or because the precise value of the output is of no consequence. The combination for which the values of the expression are not specified are called don't-care combinations. During the process of design using an SOP, K-map, each don't-care is treated as a 1 if it is helpful in Map Reduction, otherwise it is treated as a 0 and left alone. During the process of design using a POS K-map, each Don't-care is treated as a 0 if it is useful in Map Reduction, otherwise it is treated as a 1 and left alone.

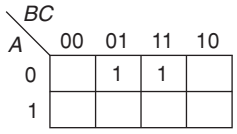
Example 11: Find the Minimal expression
$\Sigma m\,(1, 5, 6, 12, 13, 14) + d\,(2, 4)$

Solution:

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

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

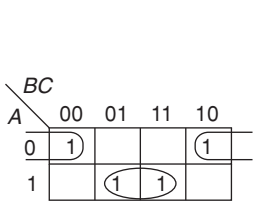
Pairs, Quads and Octets



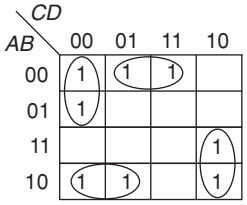
The map contains a pair of 1s those are horizontally adjacent. Those cells represent $\overline{A}\,\overline{B}C, \overline{A}BC$.

For these two min terms, there is change in the form of variable B. By combining these two cells we can form a pair, which is equal to $\overline{A}\,\overline{B}C + \overline{A}BC = \overline{A}C(\overline{B}+B) = \overline{A}C$.

If more than one pair exists on K-map, OR the simplified products to get the Boolean function.



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

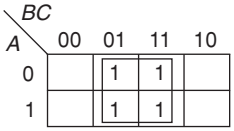


$F = \overline{A}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}D + A\overline{B}\,\overline{C} + AC\overline{D}$

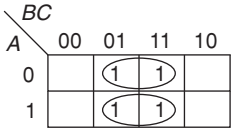
So Pair eliminates one variable by minimization.

Quad

Quad is a group of four 1s those are horizontally or vertically adjacent.



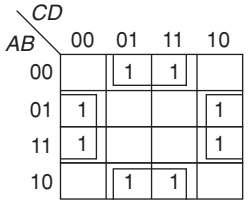
$F = C$



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

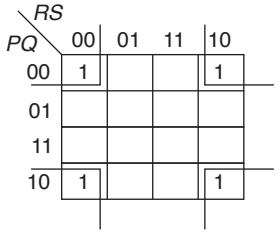
By considering two pairs also it will be simplified to C.

Quad eliminates two variables from the function



$F = B\overline{D} + \overline{B}D$

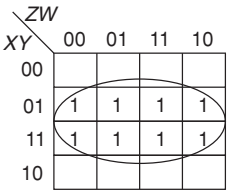
Corner min terms can from a Quad



$F = \overline{Q}\,\overline{S}$

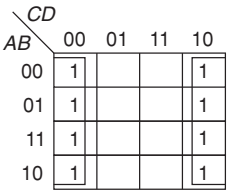
Octet

The group of eight cells will form one octet.



$F = Y$

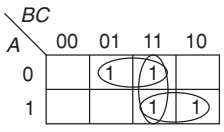
Other variable X, Z, W changes their form in octet. Octet can eliminate three variables and their complements.



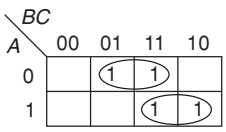
$F = \overline{D}$

Other variable A, B, C are vanished.

Eliminating Redundant Groups

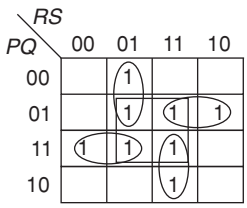


$AB + \overline{A}C + BC$

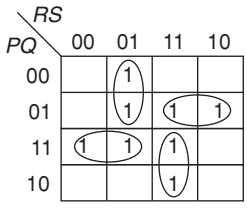


$AB + \overline{A}C$

Here BC is redundant pair, which covers already covered min terms of $AB, \overline{A}C$.



This K-map gives fourpairs and one quad.

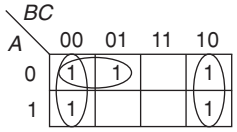


But only four pairs are enough to cover all the min times, Quad is not necessary.

$\overline{P}\,\overline{R}S + \overline{P}QR + PQ\overline{R} + PRS$ is minimized function.

Prime Implicant

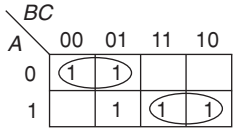
The group of adjacent min terms is called a Prime Implicant, i.e., all possible pairs, quads, octets, etc.



Prime implicants are $\overline{B}\,\overline{C}$, $B\overline{C}$, $\overline{C}$, $\overline{A}\,\overline{B}$. Minimized function is $\overline{C} + \overline{A}\,\overline{B}$

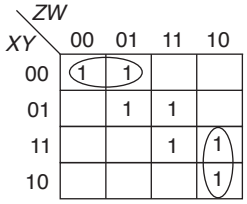
Essential Prime Implicant

The prime implicant which contains at least one min term which cannot be covered by any other prime implicant is called Essential prime implicant.



Min term 0, 6 can be grouped with only one pair each.

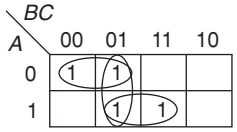
The total possible prime implicants are $\overline{A}\,\overline{B}$, $\overline{B}C$, AC, AB but min term 0, 6 can be covered with $\overline{A}\,\overline{B}, AB$. So we call them as essential prime implicants. Min term 5 can be paired with any of 1 or 7 min term.



The essential prime Implicants are $XZ\overline{W}, \overline{X}\,\overline{Y}\,\overline{Z}$

Redundant Prime Implicant

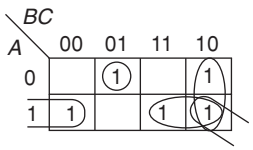
The prime implicant whose min terms are already covered by at least one min term is called redundant prime implicants.



Here prime implicants are $\overline{A}\,\overline{B}, AC, \overline{B}C$. But $\overline{B}C$ is already covered by other min terms So $\overline{B}C$ is redundant prime implicant.

Example 12: Find the minimal expression for $\Sigma m(1, 2, 4, 6, 7)$ and implement it using Basic gates.

Solution: K-map is



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

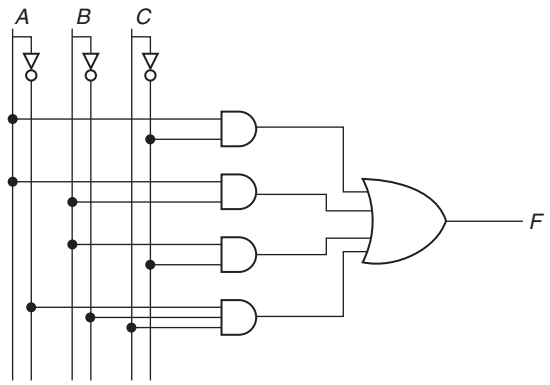
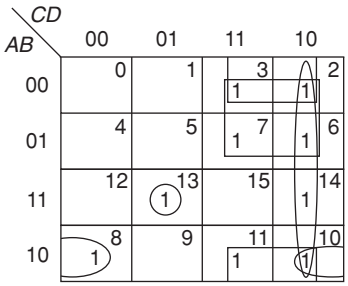


Figure 8 Logic diagram

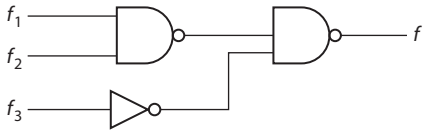
Example 13: Find the minimal expression for Σm (2, 3, 6, 7, 8, 10, 11, 13, 14)

Solution: K-map is:



$$\therefore \quad F(A,B,C,D) = AB\overline{C}D + A\overline{B}\overline{D} + \overline{A}C + \overline{B}C + C\overline{D}$$

Example 14: Three Boolean functions are defined as below $f_1 = \Sigma m(0, 1, 3, 5, 6)$, $f_2 = \Sigma m(4, 6, 7)$ $f_3 = \Sigma m(1, 4, 5, 7)$, then find f.



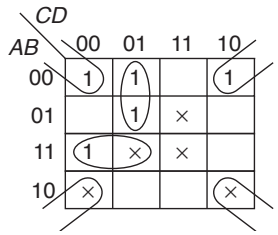
Solution: When two Boolean functions are ANDed, the resultant will contain the common min terms of both of the functions (like, intersection of min terms). If two Boolean functions are ORed, then resultant is the combination of all the min terms of the functions (like union of min terms)

Here $f = \overline{\overline{f_1 f_2} \cdot \overline{f_3}} = f_1 f_2 + f_3$

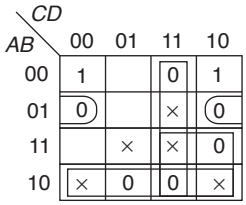
Here $f_1 \cdot f_2$ = Common min terms in f_1 and $f_2 = \Sigma m(6)$
$f_1 \cdot f_2 + f_3$ = Combination of min terms of $f_1 \cdot f_2$ and f_3
$= \Sigma(1, 4, 5, 6, 7)$

Example 15: What is the literal count for the minimized SOP, and minimized POS form for the following function?
$f(A, B, C, D) = \Sigma m(0, 1, 2, 5, 12) + \phi d(7, 8, 10, 13, 15)$

Solution: $f(A, B, C, D) = \Sigma m(0, 1, 2, 5, 12) + \phi(7, 8, 10, 13, 15)$



f = 1 quad + 2 pairs
Literal count = $1 \times 2 + 2 \times 3 = 8$
$f(A, B, C, D) = \pi M(3, 4, 6, 9, 11, 14) + \phi(7, 8, 10, 13, 15)$



f will consists of 3 quads + 1pair
$= 3 \times 2 + 1 \times 3 = 6 + 3 = 9$

Implementation of Function by Using NAND–NOR Gates

NAND or NOR gates are called as universal gates, because any function can be implemented by using only NAND gates or only using NOR gates.

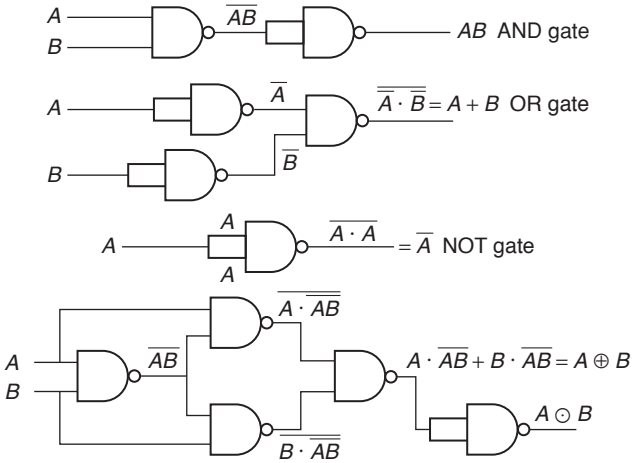


Figure 9 Implement of basics gates by using NAND gates

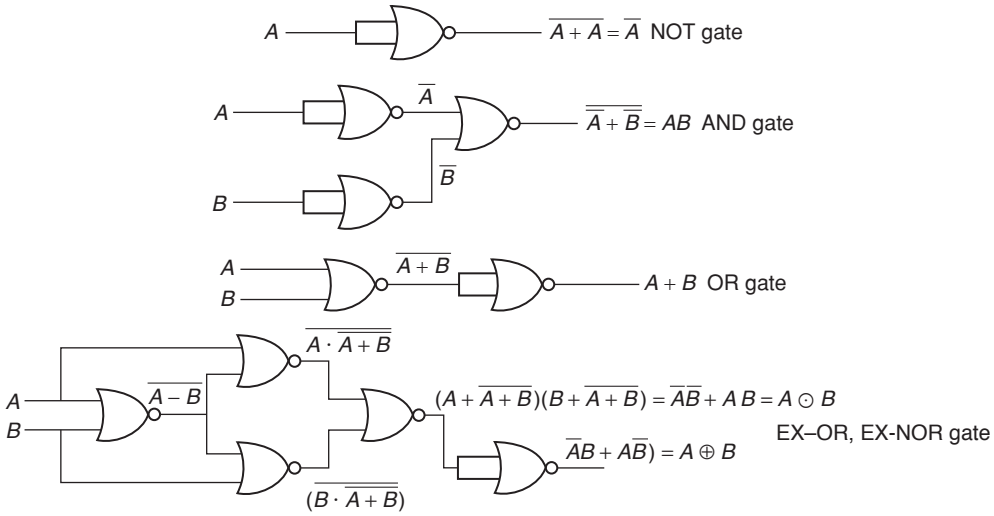
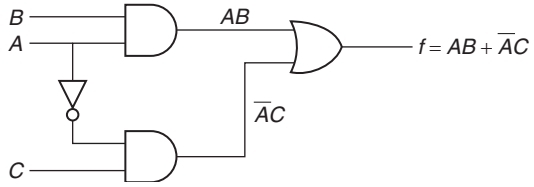


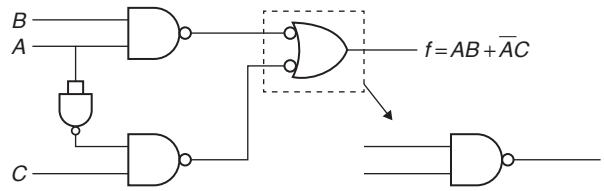
Figure 10 Implementation of basic gates by using NOR gates

Any function which is in the SOP form can be implemented by using AND-OR gates, which is also equivalent to NAND–NAND gates.

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



By considering bubble at AND gate output and OR gate input, and by changing NOT gates to NAND gates the circuit becomes as,



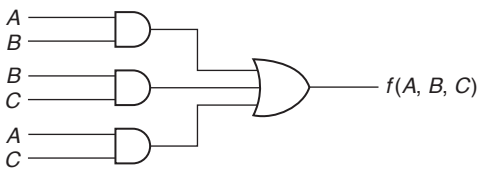
Now the circuit is in completely in NAND – NAND form

So the functions expressed in SOP form, can be implemented by using AND – OR, (or) NAND–NAND gates.

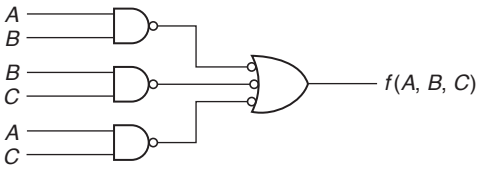
Any function in POS from, can be implemented by using OR–AND gates, which is similar to NOR–NOR gate.

Example 16: How many number of NAND gates are required to implement $f(A, B, C) = AB + BC + AC$
(A) 3 (B) 4 (C) 5 (D) 6

Solution:



By considering bubbles at output of AND gate and input of OR gate.

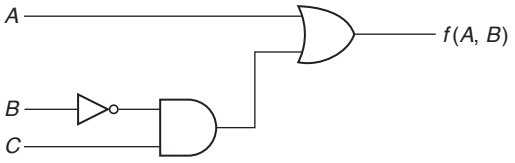


So four NAND gates are required.

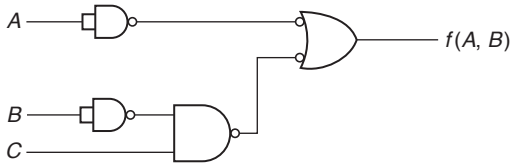
Example 17: Number of NAND gates required for implementation of $f(A,B) = A + \overline{B}C$ is

(A) 3 (B) 4 (C) 5 (D) 6

Solution:



To convert the all gates into NAND gates, place bubble output of AND, and inputs of OR gates. Now, the circuit can be drawn as



Four NAND gates are required.

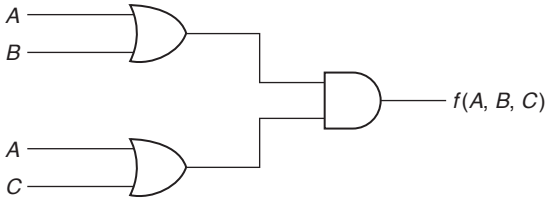
Example 18: $f = A + BC$, the number of NOR gates required to implement f, are?

(A) 3 (B) 4 (C) 5 (D) 2

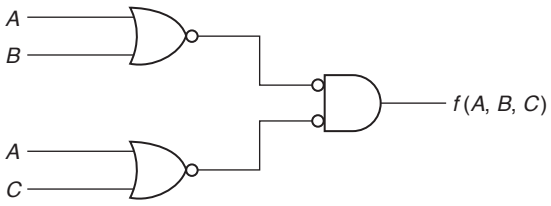
Solution: $A + BC$ is in SOP form.

To implement this function by using NOR gates, we can write $f(A, B, C) = A + BC = (A + B)(A + C)$

Which is in the form of POS?



By including bubbles at output of OR gate, and input of AND gate, the circuit becomes



Now the circuit consists of all NOR gates. Three NOR Gates are required.

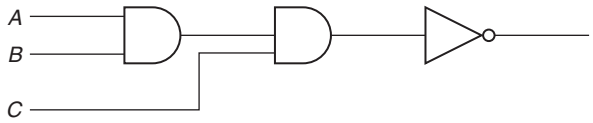
Example 19: How many number of two-input NAND–NOR gates are required to implement three-input NAND–NOR gates respectively?

(A) 2, 2 (B) 2, 3

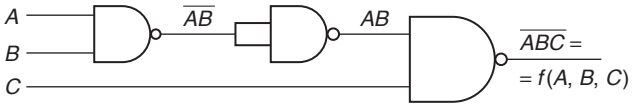
(C) 3, 2 (D) 3, 3

Solution: $f(A,B,C) = \overline{ABC} = \overline{AB} + \overline{C}$

(1) Implement above function by using two-inputs gates

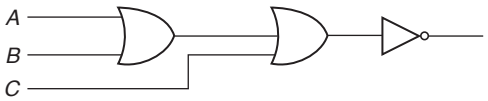


Now convert each gate to NAND gate

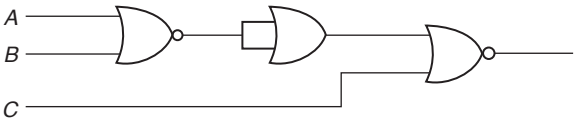


Three two-input NAND gates are required.

(2) $G(A,B,C) = \overline{A + B + C}$ Implement it by using two-input gates



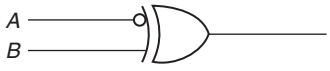
Now convert each gate to NOR gate



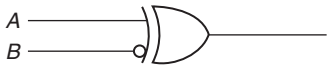
Three two-input, NOR gates are required.

EX-OR, EX-NOR Gates

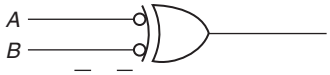
Inverted inputs for EX OR, EX-NOR gates



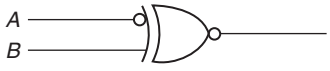
$$\overline{A} \oplus B = \overline{\overline{A}B} + \overline{A}\,\overline{B} = AB + \overline{A}\,\overline{B} = A \odot B$$



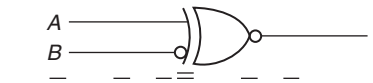
$$A \oplus \overline{B} = \overline{A\overline{B}} + \overline{A}\,\overline{B} = AB + \overline{A}\,\overline{B} = A \odot B$$



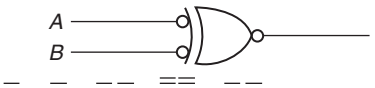
$$\overline{A} \oplus \overline{B} = \overline{A}\,\overline{\overline{B}} + \overline{\overline{A}}\,\overline{B} = \overline{A}B + A\overline{B} = A \oplus B$$



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



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

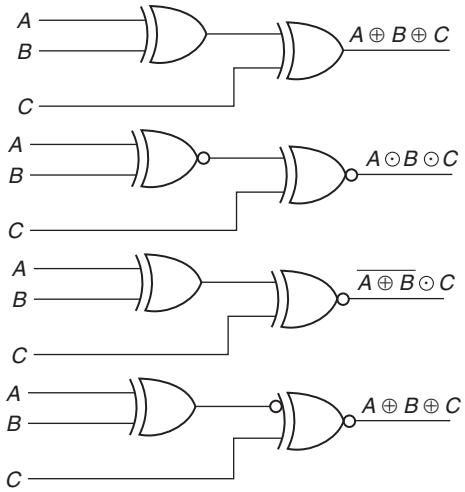


$$\overline{A} \odot \overline{B} = \overline{A}\,\overline{B} + \overline{\overline{A}}\,\overline{\overline{B}} = \overline{A}\,\overline{B} + AB = A \odot B$$

From the above discussions we can conclude that inverted input EXOR gate is EX-NOR gate.

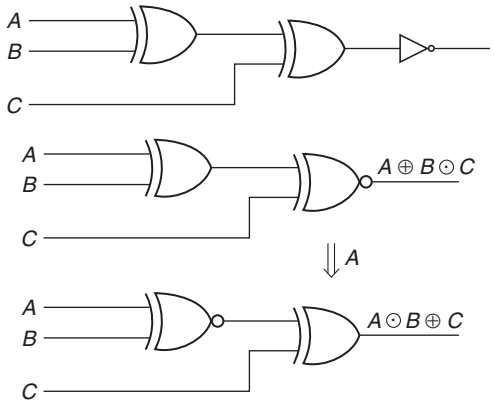
Similarly, inverted input EX-NOR gate is EX-OR gate. If both inputs are inverted the EX-OR / EX-NOR will remain as it is.

Consider a three-inputs X-OR gates by using two-input XOR gates.



So we can conclude that $A \oplus B \oplus C = A \odot B \odot C$

$$\overline{A \oplus B \oplus C} = \overline{A \odot B \odot C}$$



$$\overline{A \oplus B \oplus C} = \overline{A \odot B \odot C} = A \oplus B \odot C$$

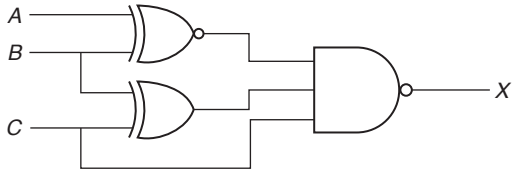
$$= A \odot B \oplus C$$

$$A \oplus B \oplus C \oplus D = A \oplus B \odot C \odot D = A \odot B \odot C \oplus D = A \odot B \oplus C \odot D$$

$$A \odot B \odot C \odot D = \overline{A \oplus B \oplus C \oplus D}$$

$$A \odot B \odot C \odot D = A \oplus B \oplus C \odot D = A \oplus B \odot C \oplus D$$

Example 20: For the logic circuit shown in figure, the required input condition (A, B, C) to make the output $X = 0$ is?



(A) 1, 1, 1 (B) 1, 0, 1
(C) 0, 1, 1 (D) 0, 0, 1

Solution: (D)

To get output $X = 0$, all inputs to the NAND gate should be 1, so $C = 1$.

When $C = 1$, the output of X-OR gate $B \oplus C = 1$ only when $B = 0$.

If $B = 0$ the output of X-NOR gate $A \odot B = 1$.

Only when $A = 0$

So $X = 1$, only when $(A, B, C) = (0, 0, 1)$.

Example 21: The minimized expression of

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

Solution: $(A + \overline{B})(A\overline{B} + AC)(\overline{A}\,\overline{C} + \overline{B})$

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

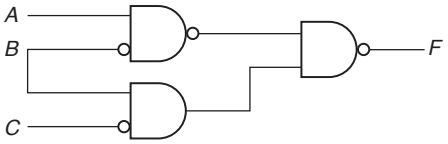
$$= (A + \overline{B})(A\overline{B} + A\overline{B}C) = (A + \overline{B})A\overline{B}(1 + C)$$

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

Example 22: The Boolean function f is independent of

(A) a (B) b
(C) c (D) None of these

Solution: (A)

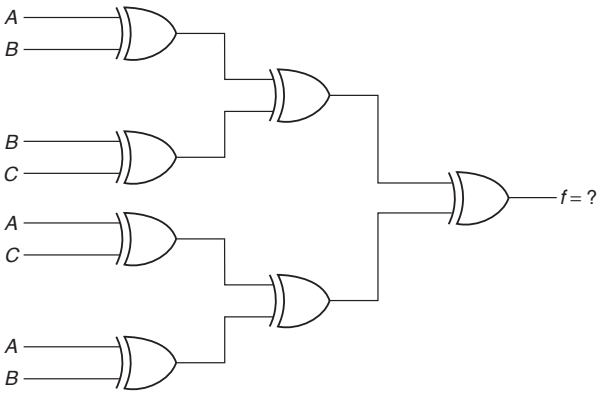


$$F = \overline{\overline{a\overline{b}} \cdot \overline{bc}}$$

$$= ab + \overline{\overline{bc}} = ab + b + \overline{c}$$

$= b + \overline{c}$ is independent of 'a'.

Example 23:



Solution: $f = \{A \oplus B \oplus B \oplus C\} \oplus \{A \oplus C \oplus B \oplus A\}$

$= \{A \oplus 0 \oplus C\} \oplus \{0 \oplus C \oplus B\}$

$= A \oplus C \oplus C \oplus B = A \oplus 0 \oplus B = A \oplus B$

Solved Examples

Example 1: Simplify the Boolean function, $x\,y + x'z + y\,z$

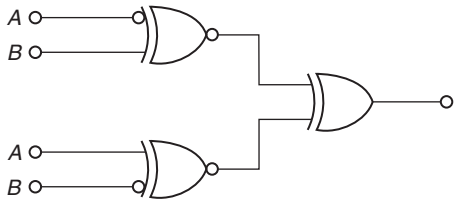
Solution: $x\,y + x'\,z + y\,z$

By using consensus property

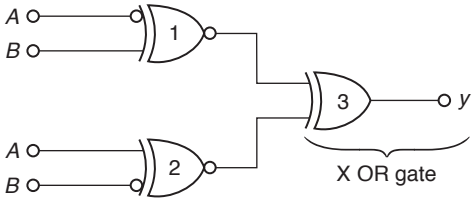
$xy + x'z + yz = xy + x'z$

$Y = xy + x'z$

Example 2: The output of the given circuit is equal to



Solution: $\overline{A} \odot B = \overline{A}B + A\overline{B}$



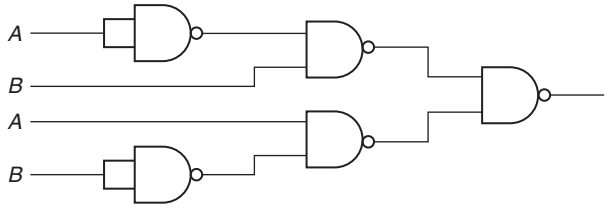
$$A \odot \overline{B} = \overline{A}B + A\overline{B}$$

So the output of above circuit is '0'. As two inputs are same at third gate.

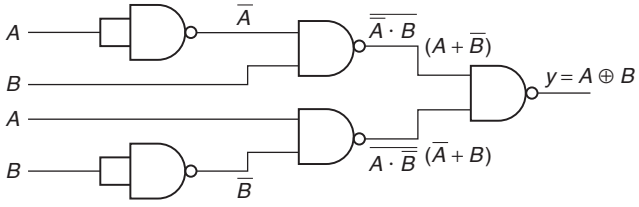
Output of XOR gate with two equal inputs is zero.

$$\therefore y = 0$$

Example 3: The circuit shown in the figure is functionally equivalent to



Solution:



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

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

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

Example 4: Simplify the Boolean function $A \oplus \overline{A}\,B \oplus \overline{A}$

Solution: $A \oplus \overline{A}\,B \oplus \overline{A}$

Associativity

$= 1 \oplus \overline{A}B = \overline{\overline{A}\,B}$

$= A + \overline{B}$ ($\because$ De Morgan's)

Example 5:

CD \ AB	00	01	11	10
00	0	0	1	1
01	0	×	×	1
11	×	×	1	×
10	1	0	1	1

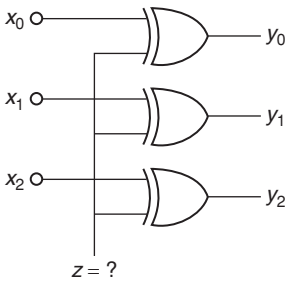
The minimized expression for the given K–map is

Solution:

CD \ AB	00	01	11	10
00	0	0	1	1
01	0	×	×	1
11	×	×	1	×
10	1	0	1	1

$= A + \overline{B}C$

Example 6: In the figure shown, y_2, y_1, y_0 will be 1s complement of $x_2\,x_1\,x_0$ if $z = ?$

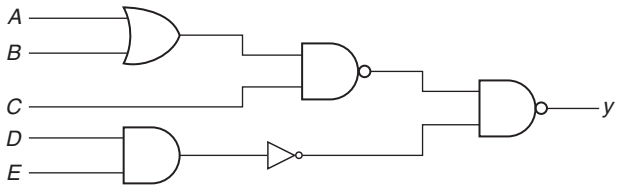


Solution: We are using X-OR gate

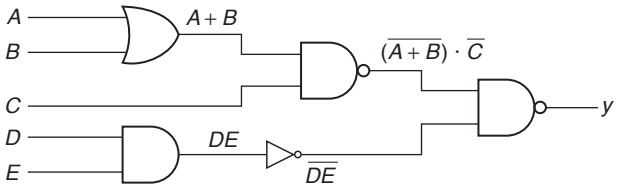
$\therefore$ XOR out-put is complement of input only when other input is high.

$\therefore Z = 1$

Example 7: The output y of the circuit shown is the figure is



Solution:



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

$$= (A+B)C + DE(\overline{x\cdot y} = \bar{x} + \bar{y})$$

Example 8: Simplify the following function

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

Solution: $\overline{\overline{A(\overline{AB})}\cdot\overline{B(\overline{A\cdot B})}}$

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

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

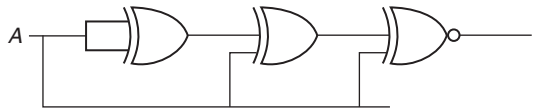
$$= \bar{A}\cdot\bar{A} + \bar{A}\bar{B} + \bar{B}\bar{A} + \bar{B}\cdot\bar{B} = \bar{A} + \bar{B} = \overline{AB}$$

EXERCISES

Practice Problems I

Directions for questions 1 to 25: Select the correct alternative from the given choices.

1. The output of the following circuit is

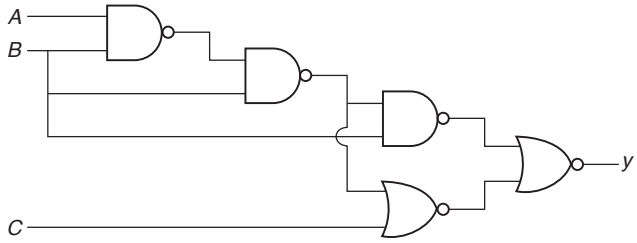


(A) 0 (B) 1
(C) A (D) A'

2. The circuit which will work as OR gate in positive level will work as ___ gate in negative level logic
(A) NOR gate
(B) NAND gate
(C) Both NAND and NOR gate
(D) AND gate

3. Four logical expressions are given below:
(a) $\bar{A}\cdot\bar{B}\cdot\bar{C}\cdot\bar{D}\,\bar{E}\cdot\bar{F}\cdot\bar{G}\cdot\bar{H}$
(b) $\overline{AB}\cdot\overline{CD}\cdot\overline{EF}\cdot\overline{GH}$
(c) $\bar{A}+\bar{B}+\bar{C}+\bar{D}+\bar{E}+\bar{F}+\bar{G}+\bar{H}$
(d) $(\bar{A}+\bar{B})(\bar{C}+\bar{D})(\bar{E}+\bar{F})(\bar{G}+\bar{H})$
Two of these expression are equal. They are
(A) c and d (B) b and d
(C) a and b (D) a and c

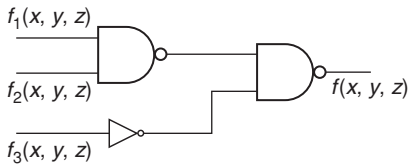
4. For the logic circuit shown in figure, the simplified Boolean expression for the out put y is



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

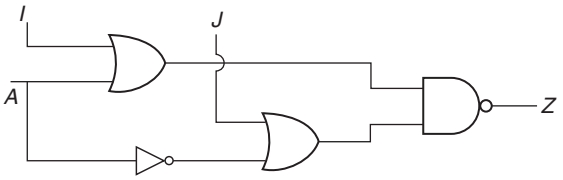
5. In a digital system, there are three inputs A, B and C. The output should be high when at least two inputs are high. The minimized Boolean expression for the out put is
(A) $AB + BC + AC$
(B) $ABC + AB\bar{C} + \bar{A}BC + A\bar{B}C$
(C) $AB\bar{C} + A\bar{B}C + \bar{A}BC$
(D) $A\bar{B} + B\bar{C} + \bar{A}C$

6. Consider the following logic circuit whose inputs are functions f_1, f_2, f_3 and output is f.



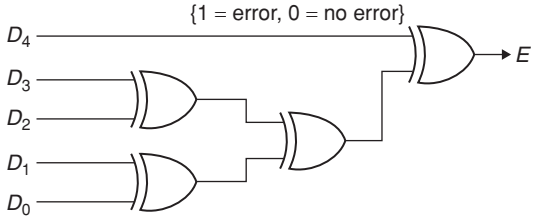
Given that $f_1(x, y, z) = \bullet(0, 1, 3, 5)$ $f_2(x, y, z) = \bullet(6, 7)$ and $f(x, y, z) = \bullet(1, 4, 5)$, then f_3 is
(A) $\bullet(1, 4, 5)$ (B) $\bullet(6, 7)$
(C) $\bullet(0, 1, 3, 5)$ (D) None of these

7. The circuit shown above is to be used to implement the function $z = f(A, B) = \bar{A} + B$ what values are to be selected for I and J?



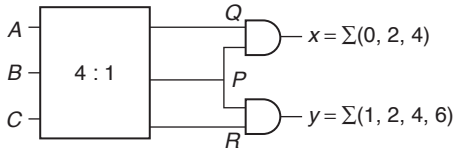
(A) $I = 0, J = B$ (B) $I = 1, J = B$
(C) $I = B, J = 1$ (D) $I = B, J = 0$

8. Parity checker output from the below figure, if input is 11111 ($D_4\,D_3\,D_2\,D_1\,D_0$) and 10000 ($D_4\,D_3\,D_2\,D_1\,D_0$).



(A) error, error
(B) error, no error
(C) no error, error
(D) no error , no error

9. For the given combinational network with three inputs A, B and C, three intermediate outputs P, Q and R, and two final outputs $X = P \cdot Q = \sum(0, 2, 4)$ and $Y = P \cdot R = \sum(1, 2, 4, 6)$ as shown below. Find the smallest function P(containing minimum number of min terms that can produce the output x and y)



(A) $\sum(2, 4)$
(B) $\sum(0, 1, 2, 4, 6)$
(C) $\sum(3, 5, 7)$
(D) $\sum(1, 2, 6)$

10. The standard form of expression $AB + ACD + \overline{A}C$ is

(A) $AB\overline{C}\,\overline{D} + ABC\overline{D} + AB\overline{C}D + ABCD + A\overline{B}CD + \overline{A}BCD + \overline{A}BC\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}\,\overline{B}C\overline{D}$

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

(C) $AB\overline{C} + ABC + ABCD + \overline{A}CB + \overline{A}C\overline{B}\,\overline{D}$

(D) $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + ABCD + \overline{A}BC + AB\overline{D} + ABC$

11. Factorize $\overline{A}\,\overline{B}\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\overline{C}D + A\overline{B}\overline{C}\overline{D} + A\overline{B}\overline{C}D$

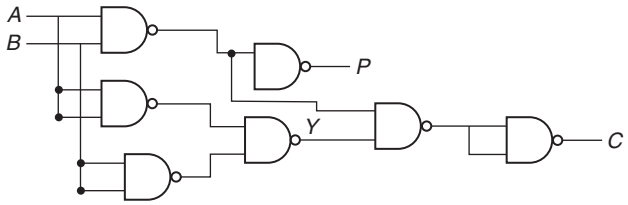
(A) $B + C$ (B) $AB + CD$
(C) $\overline{B}\overline{C}$ (D) AC

12. The K-map of a function is as shown. Find the function.

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

(A) wx (B) $\overline{z}$
(C) $\overline{w(z + \overline{z}) + \overline{z}w}$ (D) $\overline{wx + \overline{z}}$

13. The Boolean expression for P is



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

14. The Boolean expression for the truth table is

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

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

15. Simplify (d represents don't-care)

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

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

16. Simplify $\overline{(AB + \overline{C})(\overline{A + B + C})}$

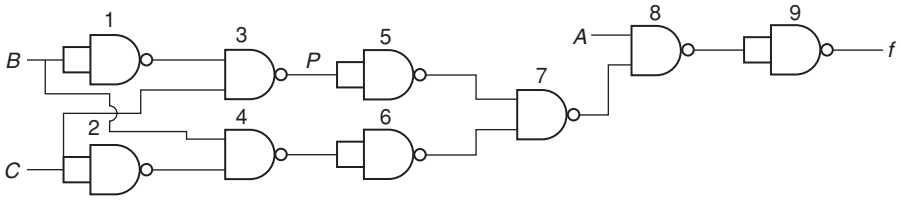
(A) $(\overline{A} + \overline{B} + \overline{C}) \cdot (A + B + C)$

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

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

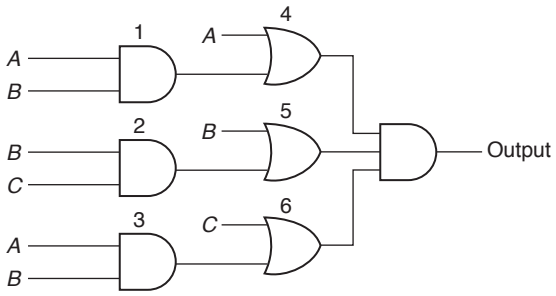
(D) None of these

17. The point P in the figure is stuck at 1. The output f will be



(A) $\overline{ABC}$ (B) $\overline{A}$ (C) $AB\overline{C}$ (D) A

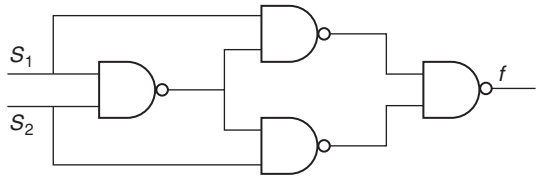
18. Find the function represented by the figure



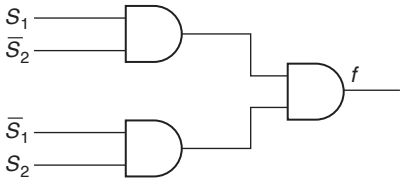
(A) $A + B + C$ (B) AB
(C) $AB + C$ (D) $B + C$

19. A staircase light is controlled by two switches, one is at the top of the stairs and other at the bottom of stairs. Realization of this function using NAND logic results in which of the following circuits? (Assume S_1 and S_2 are the switches)

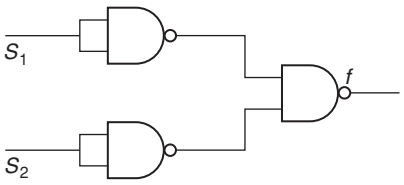
(A)



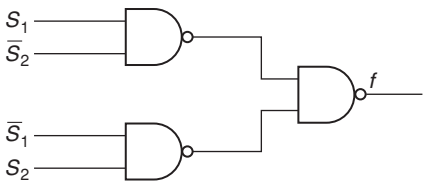
(B)



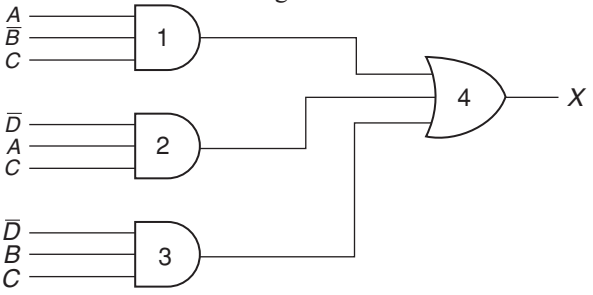
(C)



(D)



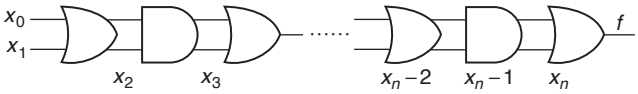
20. For the given figure simplify the expression and find which is the redundant gate?



(A) $ABC + DBC$, 4 (B) $A\bar{B}C + \bar{D}AC$, 3
(C) $\bar{D}AC + \bar{D}BC$, 1 (D) $A\bar{B}C + \bar{D}BC$, 2

21. The function $f = A \oplus B \oplus C \oplus D$ is represented as
(A) $f(A, B, C, D) = \sum(2, 6, 10, 11, 12, 13, 14)$
(B) $f(A, B, C, D) = \sum(3, 5, 7, 10, 11, 12, 13, 14)$
(C) $f(A, B, C, D) = \sum(1, 2, 6, 8, 10, 12, 13, 14)$
(D) $f(A, B, C, D) = \sum(1, 2, 4, 7, 8, 11, 13, 14)$

22. Find the function represented

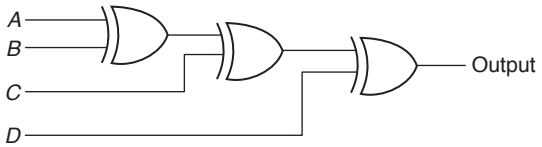


(A) $(x_0 + x_1)(x_2 + x_3)(x_4 + x_5) \ldots (x_{n-1} + x_n)$
(B) $x_0 + x_1 + x_2 + x_3 + \cdots + x_n$
(C) $x_0x_2x_4 \ldots x_n + x_1x_2 \ldots x_n + x_{n-1}x_n$
(D) $x_0x_1 + x_2x_3 + \cdots + x_{n-1}x_n$

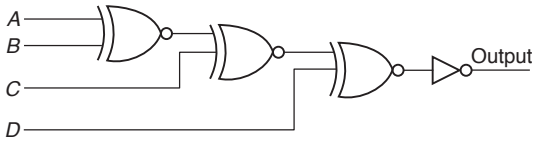
23. The minimum number of NAND gates required to implement $A \oplus B \oplus C$ is
(A) 8 (B) 10
(C) 9 (D) 6

24. Which of the following circuit will generate an odd parity for *a* 4-bit input? (Assume *ABCD* as input)

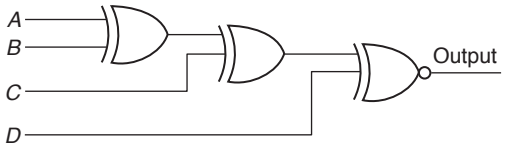
(A)



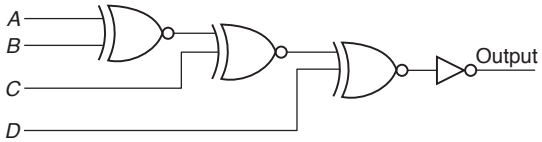
(B)



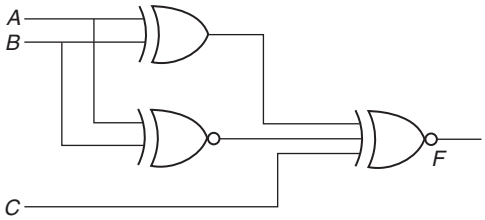
(C)



(D)



25. For the output *F* to be 1 in the circuit, the input combination should be



(A) $A = 1, B = 1, C = 0$ (B) $A = 1, B = 0, C = 0$
(C) $A = 0, B = 1, C = 0$ (D) $A = 0, B = 0, C = 1$

Practice Problems 2

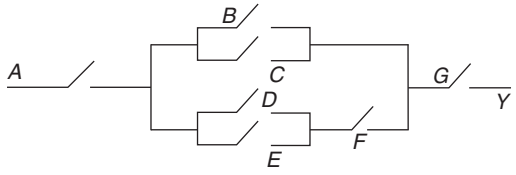
Directions for questions 1 to 25: Select the correct alternative from the given choices.

1. An OR Gate has six inputs. How many input words are there in its truth table?
(A) 6 (B) 36
(C) 32 (D) 64

2. Sum of product form can be implemented by using
(A) AND–OR
(B) NAND–NAND
(C) NOR–NOR
(D) Both A and B

3. Which one of the following is equivalent to the Boolean expression?
$Y = AB + BC + CA$
(A) $\overline{AB + BC + CA}$
(B) $(\overline{A} + \overline{B})(\overline{B} + \overline{C})(\overline{A} + \overline{C})$
(C) $\overline{(A + B)(B + C)(A + C)}$
(D) $\overline{(\overline{A} + \overline{B})(\overline{B} + \overline{C})(\overline{C} + \overline{A})}$

4. What Boolean function does the following circuit represents?



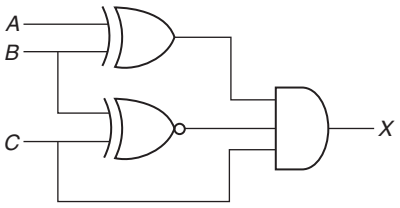
(A) $A\,[F + (B + C)\cdot(D + E)]\,G$
(B) $A + BC + DEF + G$
(C) $A\,[(B + C) + F\,(D + E)]\,G$
(D) $ABG + ABC + F(D + E)$

5. The minimum number of two input NOR gates are required to implement the simplified value of the following equation
$f(w, x, y, z) = \Sigma m(0, 1, 2, 3, 8, 9, 10, 11)$
(A) One (B) Two
(C) Three (D) Four

6. The out put of a logic gate is '1' when all inputs are at logic '0'. Then the gate is either
(1) NAND or X-OR gate
(2) NOR or X-OR gate
(3) NOR or X-NOR gate
(4) NAND or X-NOR gate
(A) 1 and 2 (B) 2 and 3
(C) 3 and 4 (D) 4 and 1

7. If the functions w, x, y, and z are as follows.
$w = R + \overline{P}Q + \overline{R}S$
$x = PQ\overline{R}\,\overline{S} + \overline{P}\,\overline{Q}RS + P\overline{Q}\,\overline{R}\,\overline{S}$
$y = RS + \overline{PR + P\overline{Q} + \overline{P}\cdot\overline{Q}}$
$z = R + S + \overline{PQ + \overline{P}\cdot\overline{Q}\cdot\overline{R} + P\cdot\overline{Q}\cdot\overline{S}}$
(A) $w = z,\ x = y$ (B) $w = z,\ x = \overline{z}$
(C) $w = y$ (D) $w = y = z$

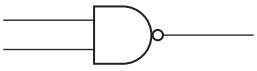
8. For the logic circuit shown in the figure, the required in put condition (A, B, C) to make the output (x) = 1 is

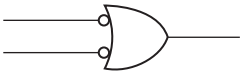


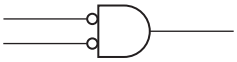
(A) 0, 0, 1 (B) 1, 0, 1
(C) 1, 1, 1 (D) 0, 1, 1

9. Which of the following is a basic gate?
(A) AND (B) X-OR
(C) X-NOR (D) NAND

10. Which of the following represent the NAND gate?

(a) 

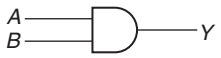
(b) 

(c) 

(A) a only (B) a, b, c
(C) b, a (D) a, c

11. The universal gates are
(A) NAND and NOR (B) AND, OR, NOT
(C) X-OR and X-NOR (D) All of these

12. In the circuit the value of input A goes from 0 to 1 and part of B goes from 1 to 0. Which of the following represent output under a static hazard condition?



(a) 

(b) 

(c) 

(d) 

(A) Output a (B) Output b
(C) Output c (D) Output d

13. The consensus theorem states that
(A) $A + \overline{A}B = A + B$
(B) $A + AB = A$
(C) $AB + \overline{A}C + BC = AB + \overline{A}C$
(D) $(A + B)\cdot(A + \overline{B}) = A$

14. The dual form of expression
$AB + \overline{A}C + BC = AB + \overline{A}C$ is

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

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

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

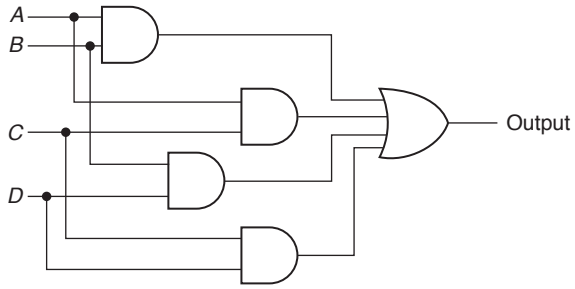
(D) $\bar{A}\bar{B}+A\bar{C}+\bar{B}\bar{C}=\bar{A}\bar{B}+A\bar{C}$

15. The max term corresponding to decimal 12 is

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

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

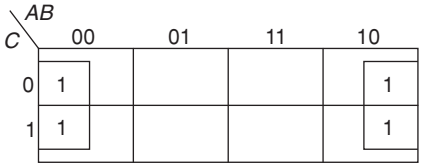
16. The given circuit is equivalent to



(A) $(A+C)(B+D)$ (B) $AC+BD$

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

17. Minimized expression for Karnaugh map is



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

(C) $\bar{B}$ (D) $\bar{B}+C$

18. An XOR gate will act as _____ when one of its input is one and as _____ when one of its input is zero.

(A) buffer, buffer (B) buffer, inverter

(C) inverter, buffer (D) inverter, inverter

19. The minimum number of two input NAND gates required to implement $A \odot B$ if only A and B are available

(A) 6 (B) 3

(C) 5 (D) 4

20. Negative logic in a logic circuit is one in which

(A) logic 0 and 1 are represented by GND and positive voltage respectively.

(B) logic 0 and 1 are represented by negative and positive voltage.

(C) logic 0 voltage level is lower than logic 1 voltage level.

(D) logic 0 voltage level is higher than logic 1 voltage level.

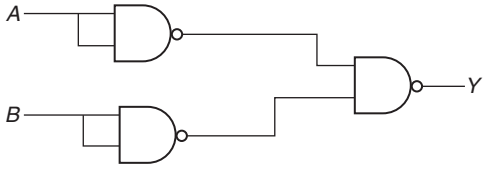
21. If the input to a gate is eight in number, then its truth table contains _____ input words.

(A) 128 (B) 8

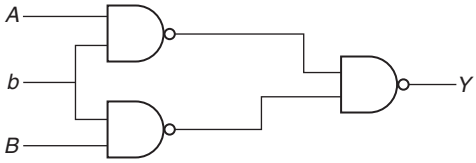
(C) 64 (D) 256

22. The X-OR gate implementation using NAND gate is

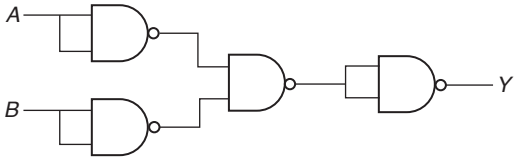
(A)



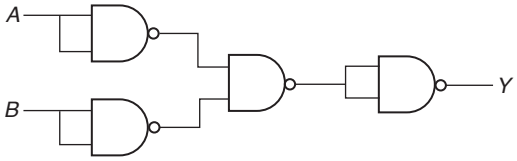
(B)



(C)



(D)



23. The equivalent of AND–OR logic circuit is

(A) NAND–NOR (B) NOR–AND

(C) NAND–NAND (D) NAND–OR

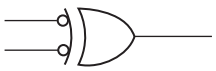
24. The X-OR is equivalent to

(A)

(B)

(C)

(D)



25. Simplify $A\bar{B}C+B+B\bar{D}+AB\bar{D}+\bar{A}C$

(A) B (B) $B+C$

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

Previous Years' Questions

1. Let $f(w,x,y,z) = \sum(0, 4, 5, 7, 8, 9, 13, 15)$. Which of the following expressions are NOT equivalent to f? **[2007]**

(P) $x'y'z' + w'xy' + wy'z + xz$
(Q) $w'y'z' + wx'y' + xz$
(R) $w'y'z' + wx'y' + xyz + xy'z$
(S) $x'y'z' + wx'y' + w'y$

(A) P only (B) Q and S
(C) R and S (D) S only

2. Define the connective $*$ for the Boolean variables X and Y as: $X * Y = XY + X'Y'$. Let $Z = X * Y$. Consider the following expressions P, Q and R. **[2007]**

$P : X = Y * Z$ $Q : Y = X * Z$
$R: X * Y * Z = 1$
Which of the following is TRUE?

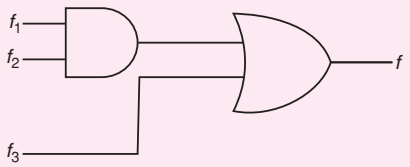
(A) Only P and Q are valid.
(B) Only Q and R are valid.
(C) Only P and R are valid.
(D) All P, Q, R are valid.

3. In the Karnaugh map shown below, × denotes a don't-care term. What is the minimal form of the function represented by the Karnaugh map? **[2008]**

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

(A) $\bar{b}\cdot\bar{d} + \bar{a}\cdot\bar{d}$
(B) $\bar{a}\cdot\bar{b} + \bar{b}\cdot\bar{d} + \bar{a}\cdot b\cdot\bar{d}$
(C) $\bar{b}\cdot\bar{d} + \bar{a}\cdot b\cdot\bar{d}$
(D) $\bar{a}\cdot\bar{b} + \bar{b}\cdot\bar{d} + \bar{a}\cdot\bar{d}$

4. Given f_1, f_3 and f in canonical sum of products form (in decimal) for the circuit **[2008]**



$f_1 = \Sigma m(4, 5, 6, 7, 8)$
$f_3 = \Sigma m(1, 6, 15)$
$f = \Sigma m(1, 6, 8, 15)$
then f_2 is

(A) $\Sigma m(4, 6)$ (B) $\Sigma m(4, 8)$
(C) $\Sigma m(6, 8)$ (D) $\Sigma m(4, 6, 8)$

5. If P, Q, R are Boolean variables, then $(P+\bar{Q})(P\cdot\bar{Q} + P\cdot R)(\bar{P}\cdot\bar{R} + \bar{Q})$ simplifies to **[2008]**

(A) $P\cdot\bar{Q}$ (B) $P\cdot\bar{R}$
(C) $P\cdot\bar{Q} + R$ (D) $P\cdot\bar{R} + Q$

6. What is the minimum number of gates required to implement the Boolean function $(AB + C)$, if we have to use only two-input NOR gates? **[2009]**

(A) 2 (B) 3
(C) 4 (D) 5

7. The binary operation □ is defined as follows **[2009]**

P	Q	P□Q
T	T	T
T	F	T
F	T	F
F	F	T

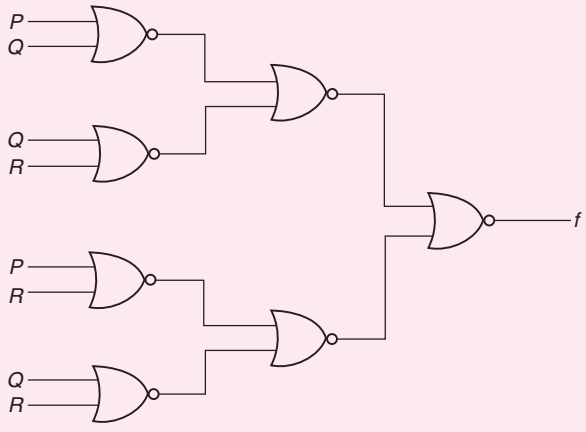
Which one of the following is equivalent to P□Q?

(A) $\neg Q\ \neg P$ (B) P□$\neg Q$
(C) $\neg P$□Q (D) $\neg P$□$\neg Q$

8. The min term expansion of $f(P, Q, Q) = PQ + Q\bar{R} + P\bar{R}$ is **[2010]**

(A) $m_2 + m_4 + m_6 + m_7$
(B) $m_0 + m_1 + m_3 + m_5$
(C) $m_0 + m_1 + m_6 + m_7$
(D) $m_2 + m_3 + m_4 + m_5$

9. What is the Boolean expression for the output f of the combinational logic circuit of NOR gates given below? **[2010]**



(A) $\overline{Q + R}$ (B) $\overline{P + Q}$
(C) $\overline{P + R}$ (D) $\overline{P + Q + R}$

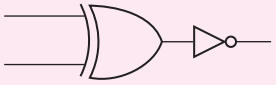
10. The simplified SOP (Sum of Product) form of the Boolean expression. **[2011]**

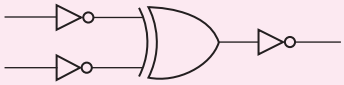
$(P+\overline{Q}+\overline{R})\,(P+\overline{Q}+R)\,(P+Q+\overline{R})$ is

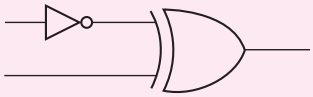
(A) $(PQ+\overline{R})$ (B) $(P+\overline{Q}\,\overline{R})$

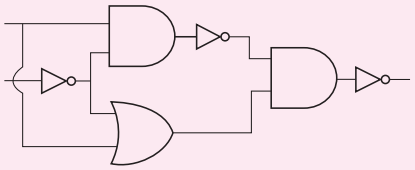
(C) $(\overline{P}Q+R)$ (D) $(PQ+R)$

11. Which one of the following circuits is NOT equivalent to a two-input X-NOR (exclusive NOR) gate? **[2011]**

(A) 

(B) 

(C) 

(D) 

12. The truth table **[2012]**

X	Y	F(X, Y)
0	0	0
0	1	0
1	0	1
1	1	1

represents the Boolean function

(A) X (B) $X+Y$

(C) $X \oplus Y$ (D) Y

13. What is the minimal form of the Karnaugh map shown below? Assume that × denotes a don't-care term. **[2012]**

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

(A) $\overline{b}\,\overline{d}$ (B) $\overline{b}\,\overline{d}+\overline{b}\,\overline{c}$

(C) $\overline{b}\,\overline{d}+a\overline{b}\overline{c}d$ (D) $\overline{b}\,\overline{d}+\overline{b}\,\overline{c}+\overline{c}\,\overline{d}$

14. Which one of the following expressions does not represent exclusive NOR of x and y? **[2013]**

(A) $xy+x'y'$ (B) $x \oplus y'$

(C) $x' \oplus y$ (D) $x' \oplus y'$

15. Consider the following Boolean expression for F:

$F(P,Q,R,S)=PQ+\overline{P}QR+\overline{P}Q\overline{R}S$

The minimal sum of products form of F is **[2014]**

(A) $PQ+QR+QS$ (B) $P+Q+R+S$

(C) $\overline{P}+\overline{Q}+\overline{R}+\overline{S}$ (D) $\overline{P}R+\overline{P}\,\overline{R}S+P$

16. The dual of a Boolean function $f(x_1, x_2, \ldots x_n, +, ., ')$, written as F^D, is the same expression as that of F with + and . swapped. F is said to be self-dual if $F = F^D$. The number of self-dual functions with n Boolean variables is **[2014]**

(A) 2^n (B) 2^{n-1}

(C) 2^{2^n} (D) $2^{2^{n-1}}$

17. Consider the following min term expression for F:

$F(P, Q, R, S) = \Sigma m(0, 2, 5, 7, 8, 10, 13, 15)$

The min terms 2, 7, 8 and 13 are 'don't-care terms. The minimal sum of products form for F is **[2014]**

(A) $Q\overline{S}+\overline{Q}S$

(B) $\overline{Q}\,\overline{S}+QS$

(C) $\overline{Q}\,\overline{R}\,\overline{S}+\overline{Q}\,R\,\overline{S}+Q\,\overline{R}\,S+QRS$

(D) $\overline{P}\,\overline{Q}\,\overline{S}+\overline{P}\,QS+PQS+P\overline{Q}\,\overline{S}$

18. The binary operator ≠ is defined by the following truth table

p	q	p ≠ q
0	0	0
0	1	1
1	0	1
1	1	0

Which one of the following is true about the binary operator ≠? **[2015]**

(A) Both commutative and associative

(B) Commutative but not associative

(C) Not commutative but associative

(D) Neither commutative nor associative

19. Consider the operations **[2015]**

$f(X, Y, Z) = X^1\,YZ + XY^1 + Y^1Z^1$ and

$g(X, Y, Z) = X^1\,YZ + X^1\,YZ^1 + XY.$

Which one of the following is correct?

(A) Both $\{f\}$ and $\{g\}$ are functionally complete

(B) Only $\{f\}$ is functionally complete

(C) Only $\{g\}$ is functionally complete

(D) Neither $\{f\}$ nor $\{g\}$ is functionally complete

20. The number of min-terms after minimizing the following Boolean expression is _____ **[2015]**

$[D^1 + AB^1 + A^1C + AC^1\,D + A^1C^1D]^1$

21. Let # be a binary operator defined as **[2015]**

$X \# Y = X^1 + Y^1$ where X and Y are Boolean variables.

Consider the following two statements.

(S1) $(P \# Q) \# R = P \# (Q \# R)$

(S2) $Q \# R = R \# Q$

Which of the following is/are true for the Boolean variables P, Q and R?

(A) Only S_1 is true

(B) Only S_2 is true

(C) Both S_1 and S_2 are true
(D) Neither S_1 nor S_2 are true

22. Given the function $F = P^1 + QR$, where F is a function in three Boolean variables P, Q and R and $P^1 = !P$, consider the following statements. **[2015]**

(S_1) $F = \Sigma(4, 5, 6)$

(S_2) $F = \Sigma(0, 1, 2, 3, 7)$

(S_3) $F = \pi(4, 5, 6)$

(S_4) $F = \pi(0, 1, 2, 3, 7)$

Which of the following is true?

(A) (S_1) – False, (S_2) – True, (S_3) – True, (S_4) – False
(B) (S_1) – True, (S_2) – False, (S_3) – False, (S_4) – True
(C) (S_1) – False, (S_2) – False, (S_3) – True, (S_4) – True
(D) (S_1) – True, (S_2) – True, (S_3) – False, (S_4) – False

23. The total number of prime implicants of the function $f(w, x, y, z) = \Sigma(0, 2, 4, 5, 6, 10)$ is _____ **[2015]**

24. Consider the Boolean operator # with the following properties: **[2016]**

$x \# 0 = x$, $x \# 1 = \bar{x}$, $x \# x = 0$ and

$x \# \bar{x} = 1$. Then $x \# y$ is equivalent to

(A) $x\bar{y} + \bar{x}y$ (B) $x\bar{y} + \bar{x}\bar{y}$
(C) $\bar{x}y + xy$ (D) $xy + \bar{x}\bar{y}$

25. Consider the Karnaugh map given below, where X represents *"don't care"* and blank represents 0.

dc \ ba	00	01	11	10
00		X	X	
01	1			X
11	1			1
10		X	X	

Assume for all inputs (a, b, c, d), the respective complements $(\bar{a}, \bar{b}, \bar{c}, \bar{d})$ are also available. The above logic is implemented using 2-input NOR gates only. The minimum number of gates required is _____. **[2017]**

26. If w, x, y, z are Boolean variables, then which one of the following is INCORRECT? **[2017]**

(A) $wx + w(x + y) + x(x + y) = x + wy$
(B) $\overline{w\bar{x}(y + \bar{z})} + \bar{w}x = \bar{w} + x + \bar{y}z$
(C) $\left(w\bar{x}\left(y + x\bar{z}\right) + \bar{w}\bar{x}\right)y = x\bar{y}$
(D) $(w + y)(wxy + wyz) = wxy + wyz$

27. Given $f(w, x, y, z) = \sum m(0,1,2,3,7,8,10) + \sum d(5,6,11,15)$, where d represents the *don't-care* condition in Karnaugh maps. Which of the following is a minimum product-of-sums (POS) form of $f(w, x, y, z)$? **[2017]**

(A) $f = (\bar{w} + \bar{z})(\bar{x} + z)$
(B) $f = (\bar{w} + z)(x + z)$
(C) $f = (w + z)(\bar{x} + z)$
(D) $f = (w + \bar{z})(\bar{x} + z)$

28. Let $\oplus$ and $\odot$ denote the Exclusive OR and Exclusive NOR operations, respectively. Which one of the following is NOT CORRECT? **[2018]**

(A) $\overline{P \oplus Q} = P \odot Q$
(B) $\bar{P} \oplus Q = P \odot Q$
(C) $\bar{P} \oplus \bar{Q} = P \oplus Q$
(D) $(P \oplus \bar{P}) \oplus Q = (P \odot \bar{P}) \odot \bar{Q}$

29. Consider the minterm list form of a Boolean function F given below.

$$F(P, Q, R, S) = \sum m(0, 2, 5, 7, 9, 11) + d(3, 8, 10, 12, 14)$$

Here, m denotes a minterm and d denotes a don't care term. The number of essential prime implicants of the function F is _____. **[2018]**

Answer Keys

Exercises

Practice Problems 1

1. B	**2.** D	**3.** B	**4.** C	**5.** A	**6.** A	**7.** B	**8.** A	**9.** B	**10.** A
11. C	**12.** D	**13.** A	**14.** A	**15.** A	**16.** A	**17.** D	**18.** B	**19.** A	**20.** D
21. D	**22.** C	**23.** A	**24.** C	**25.** D					

Practice Problems 2

1. D	**2.** D	**3.** D	**4.** C	**5.** A	**6.** C	**7.** B	**8.** D	**9.** A	**10.** C
11. A	**12.** D	**13.** A	**14.** A	**15.** A	**16.** C	**17.** C	**18.** C	**19.** C	**20.** D
21. D	**22.** C	**23.** C	**24.** B	**25.** B					

Previous Years' Questions

1. D	**2.** D	**3.** A	**4.** C	**5.** A	**6.** B	**7.** B	**8.** A	**9.** A	**10.** B
11. D	**12.** A	**13.** B	**14.** D	**15.** A	**16.** D	**17.** B	**18.** A	**19.** B	**20.** 1
21. B	**22.** A	**23.** 3	**24.** A	**25.** 1	**26.** C	**27.** A	**28.** D	**29.** 3	

Chapter 3

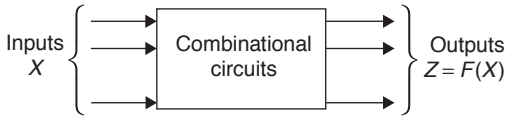
Combinational Circuits

LEARNING OBJECTIVES

- *Combinational logic design*
- *Arithmetic circuit*
- *Half adder*
- *Full adder*
- *Half subtractor*
- *Full subtractor*
- *n-bit comparator*
- *Parity bit generator and parity bit checker*
- *Code converter*
- *Decoder*
- *Designing high order decoders from lower order decoder*
- *Combinational logic implementation*
- *Encoders*
- *Multiplexer*
- *Demultiplexer*

Introduction

Combinational logic is a type of logic circuit whose output is a function of the present input only.



Combinational Logic Design

The design of combinational circuit starts from the problem, statement and ends with a gate level circuit diagram.

The design procedure involves the following steps:

1. Determining the number of input variables and output variables required, from the specifications.
2. Assigning the letter symbols for input and output.
3. Deriving the truth table that defines the required relationship between input and output.
4. Obtaining the simplified Boolean function for each output by using K-map or algebraic relations.
5. Drawing the logic diagram for simplified expressions.

We will discuss combinational circuits under the following categories:

- Arithmetic circuits
- Code converters
- Data processing circuits

Arithmetic Circuits

Arithmetic circuits are the circuits that perform arithmetic operation. The most basic arithmetic operation is addition.

Half Adder

Addition is an arithmetic operation, and here to implement addition in digital circuits we have to implement by logical gates. So the addition of binary numbers will be represented by the logical expressions. Half adder is an arithmetic circuit which performs the addition of two binary bits, and the result is viewed in two output—sum and carry.

The sum 'S' is the X-OR of 'A' and 'B' where A and B are inputs.

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

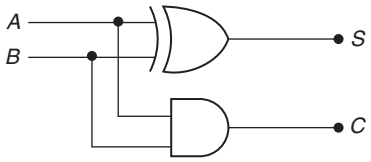
The carry 'C' is the AND of A and B.

$$\therefore C = AB$$

Table 1 *Truth Table*

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

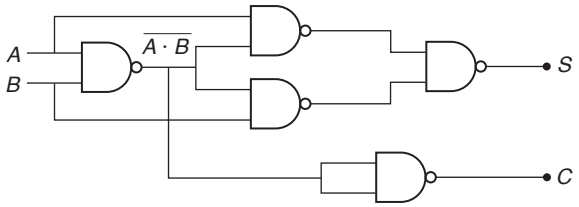
So, half adder can be realized by using one X-OR gate and one AND gate.



Half adder can also be realized by universal logic such as only NAND gate or only NOR gate as given below.

NAND logic

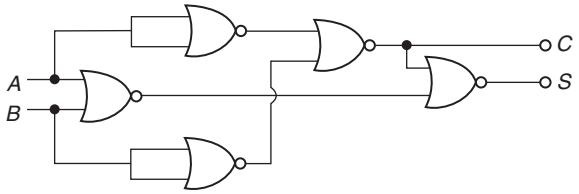
$$S = A\bar{B} + \bar{A}B$$
$$= A\bar{B} + A\bar{A} + \bar{A}B + B\bar{B}$$
$$= A(\bar{A} + \bar{B}) + B(\bar{A} + \bar{B})$$
$$= \overline{\overline{A\bar{A}B} \cdot \overline{B\bar{A}B}}$$
$$C = AB = \overline{\overline{A \cdot B}}$$



Half adder using NAND logic

NOR logic

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



Half adder using NOR logic

Full Adder

Full adder is an arithmetic circuit that performs addition of two bits with carry input. The result of full adder is given by two outputs—sum and carry. The full adder circuit is used in parallel adder circuit as well as in serial adder circuit.

For full adder, if total number of 1's is odd at input lines, the sum output is equal to logic 1, and if total number of 1's at input lines are more than or equal to 2, then the carry output is logic 1.

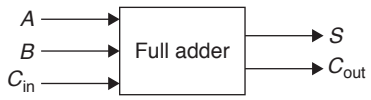


Figure 1 Block diagram

Table 2 *Truth Table*

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

$$S = \bar{A}\bar{B}C_{in} + \bar{A}B\overline{C_{in}} + A\bar{B}\,\overline{C_{in}} + ABC_{in}$$
$$= A \oplus B \oplus C_{in}$$
$$C_{out} = \bar{A}BC_{in} + A\bar{B}C_{in} + AB\overline{C_{in}} + ABC_{in}$$
$$= AB + (A \oplus B)\,C_{in}$$
$$= AB + A\,C_{in} + B\,C_{in}$$

Full adder can also be realized using universal logic gates, i.e., either only NAND gates or only NOR gates as explained below.

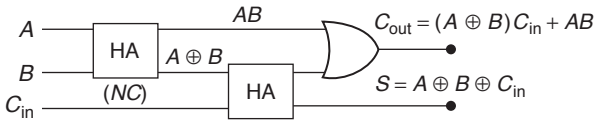


Figure 2 Block diagram of full adder by using Half adder

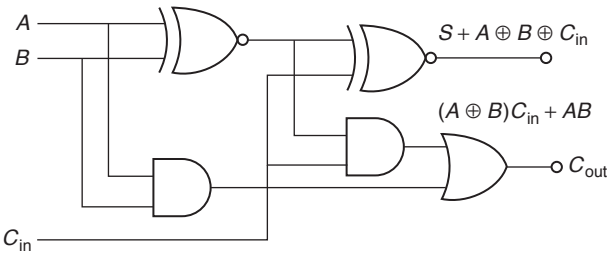


Figure 3 Logic diagram of full adder

NAND logic

$$A \oplus B = \overline{\overline{A\bar{A}B}\;\;\overline{B\bar{A}B}}$$

So $A \oplus B \oplus C_{in}$

Let $A \oplus B = x$ then $s = \overline{\overline{X \cdot \overline{XC_{in}}}\;\;\overline{C_{in}\overline{X \cdot C_{in}}}}$

$$= X \oplus C_{in}$$

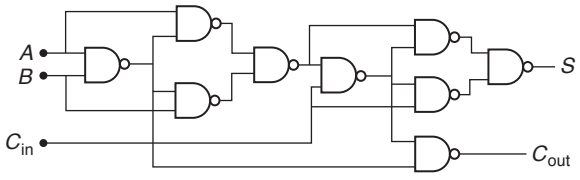


Figure 4 Logic diagram of a full adder using only 2-input NAND gates

NOR logic

Full adder outputs

Sum = $a \oplus b \oplus c$, carry = $ab + bc + ac$ are self dual functions

[$\because$ A function is called as self dual if its dual is same as the function itself $f^D = f$]

For self dual functions, the number of NAND gates are same as number of NOR gates.

By taking the dual for above NAND gate implementation, all gates will become NOR gates, and the output is dual of the sum and carry, but they are self dual ($f^D = f$).

So, output remain same, and only 9 NOR gates are required for full adder, structure similar to NAND gate circuit.

Half Subtractor

Half subtractor is an arithmetic circuit which performs subtraction of one bit (subtrahend) from other bit (minuend), and the result gives difference and borrow each of one bit. The borrow output is logic 1 only if there is any subtraction of 1 from 0.

When a bit 'B' is subtracted from another bit 'A', a difference bit (d) and a borrow bit (b) result according to the rule given below.

Table 3 *Truth Table*

A	B	d	b
0	0	0	0
1	0	1	0
1	1	0	0
0	1	1	1

$$d = A\bar{B} + B\bar{A}$$
$$= A \oplus B$$
$$b = \bar{A}B$$

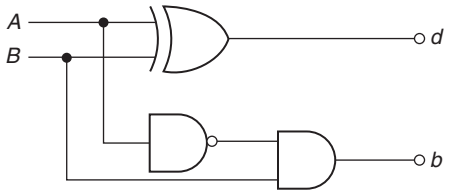


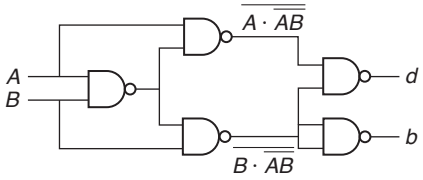
Figure 5 Logic diagram of a half subtractor

A half subtractor can also realized using universal logic either using only NAND gates or only NOR gates as explained below.

NAND logic

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

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



NOR logic

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

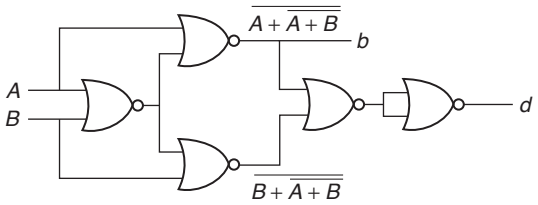


Figure 6 Logic diagram of half subtractor using NOR gate

Full Subtractor

Full subtractor is an arithmetic circuit similar to half subtractor but it performs subtraction with borrow, it involves subtraction of three bits—minuend, subtrahend and borrow-in, and two outputs—difference and borrow. The subtraction of 1 from 0 results in borrow to become logic 1. The presence of odd number of 1's at input lines make difference as logic 1.

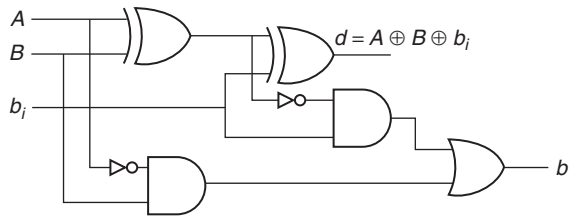
Table 4 *Truth Table*

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

$$d = \bar{A}\bar{B}b_i + \bar{A}B\bar{b_i} + A\bar{B}\bar{b_i} + ABb_i$$
$$= b_i(AB + \bar{A}\bar{B}) + \bar{b_i}(A\bar{B} + \bar{A}B)$$
$$= b_i(\overline{A \oplus B}) + \bar{b_i}(A \oplus B)$$
$$= A \oplus B \oplus b_i$$

and

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



NAND logic

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

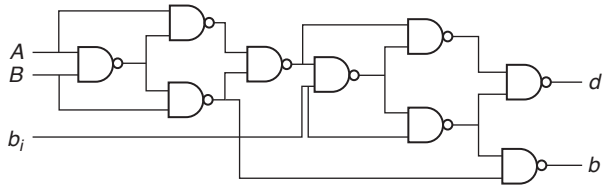


Figure 7 Logic diagram of a full subtractor using NAND logic

NOR logic

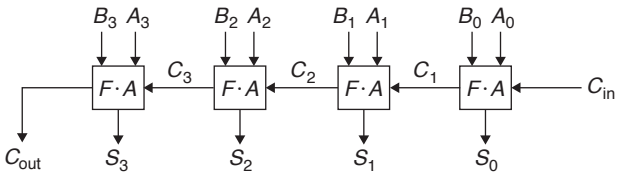
Output of full subtractor is also self dual in nature. So, same circuit, with all NAND gates, replaced by NOR gates gives the NOR gate full subtractor. 9 NOR gates required.

Example 1: How many NAND gates are required for implementation of full adder and full subtractor respectively?
(A) 11, 10 (B) 11, 11 (C) 9, 9 (D) 9, 10

Solution: (C)
From the circuit diagrams in the previous discussion, full adder requires 9 NAND gates and full subtractor requires 9 NAND gates.

Binary Adder

A binary adder is a digital circuit that produces the arithmetic sum of two binary numbers.



Four bit parallel adder the output carry from each full adder is connected to the input carry of next full adder.

The bits are added with full adders, starting from the LSB position to form the sum bit and carry bit.

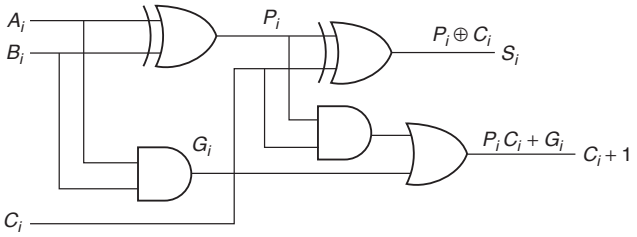
The longest propagation delay time in parallel adder is the time it takes the carry to propagate through the full adders.

For n-bit parallel adders consider t_{pds} is the propagation delay for sum of each full adder and t_{pdc} is the propagation delay of carry.

The total time required to add all n-bits at the nth full adder is

$$T_S = t_{pds} + (n-1)t_{pdc}$$

So propagation delay increases with number of bits. To overcome this difficulty we use look ahead carry adder, which is the fastest carry adder.



Consider the full adder circuit for ith stage, in parallel adder, with two binary variables A_i, B_i, input carry C_i are:

Carry propagate (P_i) and carry generate (G_i)

$$P_i = A_i \oplus B_i$$
$$G_i = A_i \cdot B_i$$

The output sum and carry can be expressed as

$$S_i = P_i \oplus C_i$$
$$C_{i+1} = P_i C_i + G_i$$

Now, the Boolean functions for each stage can be calculated as substitute $i = 0$

C_0 is input carry

$$C_1 = G_0 + P_0\,C_0$$

Substitute $i = 1, 2 \ldots$

$$C_2 = G_1 + P_1C_1 = G_1 + P_1\,(G_0 + P_0C_0)$$
$$= G_1 + P_1\,G_0 + P_1\,P_0\,C_0$$
$$C_3 = G_2 + P_2\,C_2 = G_2 + P_2\,(G_1 + P_1\,G_0 + P_1\,P_0\,C_0)$$
$$= G_2 + P_2\,G_1 + P_2\,P_1\,G_0 + P_2\,P_1\,P_0\,C_0$$

Since the Boolean function for each output carry is expressed in SOP form, each function can be implemented with AND–OR form or two level NAND gates.

From the above equations we can conclude that this circuit can perform addition in less time as C_3 does not have to wait for C_2 and C_1 to propagate. C_3, C_2, C_1 can have equal time delays.

The gain in speed of operation is achieved at the expense of additional complexity (hardware).

n-bit Comparator

The comparison of two numbers is an operation that determines whether one number is greater than, less than, or equal to the other number.

A magnitude comparator is a combinational circuit that compares two input numbers A and B, and specifies the output with three variables, $A > B$, $A = B$, $A < B$:

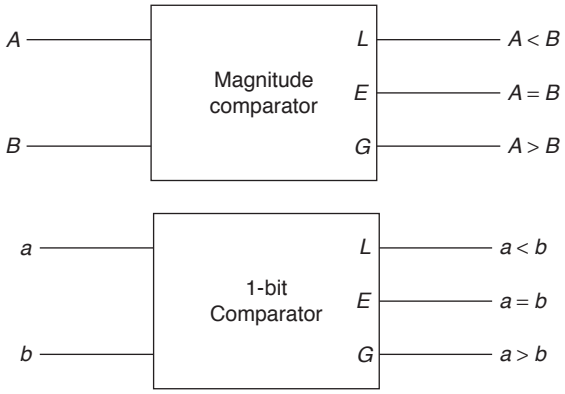


Figure 8 1-bit comparator will have only 1 bit input *a*, *b*.

a	b	a < b	a = b	a > b
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

By considering minterms for each output.

$$(a < b) = a'b$$
$$(a = b) = a'b' + ab = a \odot b$$
$$(a > b) = ab'$$

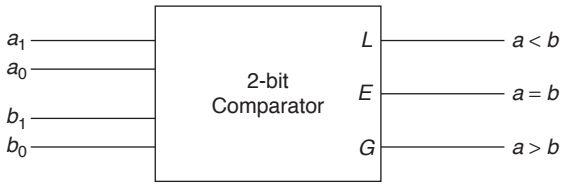
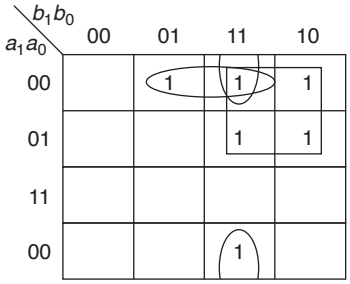


Figure 9 2-bit comparator will have 2-bit inputs a_1 a_0 and b_1 b_0.

a_1	a_0	b_1	b_0	L a < b	E a = b	G a > b
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

$$(a < b) = \Sigma(1, 2, 3, 6, 7, 11)$$
$$(a > b) = \Sigma(4, 8, 9, 12, 13, 14)$$
$$(a = b) = \Sigma(0, 5, 10, 15)$$



$$a < b = a_1'a_0'b_0 + a_0'b_1b_0 + a_1'b_1$$
$$L = \overline{a_1}b_1 + (a_1 \odot b_1)\overline{a_0}b_0$$

Similarly, $a > b = a_0b_1'b_0' + a_1a_0b_0' + a_1b_1'$

$$G = a_1\overline{b_1} + (a_1 \odot b_1)a_0\overline{b_0}$$

$a = b$ is possible when $a_1 = b_1$, $a_0 = b_0$

So $(a = b) = (a_1 \odot b_1)(a_0 \odot b_0)$

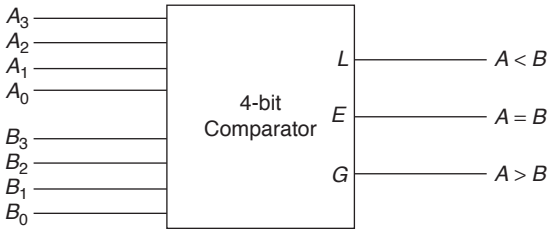


Figure 10 4-bit comparator will compare 2 input numbers each of 4-bits A_3 A_2 A_1 A_0 and B_3 B_2 B_1 B_0 ($A = B$) output will be 1 when each bit of input *A* is equal to corresponding bit in input *B*.

So we can write $(A = B) = (A_3 \odot B_3)\ (A_2 \odot B_2)\ (A_1 \odot B_1)\ (A_0 \odot B_0)$.

To determine whether A is greater or less than B, we inspect the relative magnitudes of pairs of significant bits, starting from MSB. If the two bits of a pair are equal, we compare the next lower significant pair of bits. The comparison continues until a pair of unequal bits is reached.

for $A < B$, $A = 0$, $B = 1$

for $A > B$, $A = 1$, $B = 0$

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

$$A > B = A_3B_3' + (A_3 \odot B_3)\ A_2B_2' + (A_3 \odot B_3)\ (A_2 \odot B_2) \times A_1B_1' + (A_3 \odot B_3)(A_2 \odot B_2)\ (A_1 \odot B_1)\ A_0B_0'$$

4-bit comparator will have total 8 inputs and $2^8 = 256$ input combinations in truth table.

For 16 combinations $(A = B) = 1$, and for 120 combinations $A < B = 1$.

For remaining 120 combination $A > B = 1$

Parity Bit Generator and Parity Bit Checker

When digital information is transmitted, it may not be received correctly by the receiver. To detect one bit error at receiver we can use parity checker.

For detection of error an extra bit, known as parity bit, is attached to each code word to make the number of 1's in the code even (in case of even parity) or odd (in case of odd parity).

For n-bit data, we use n-bit parity generator at the transmitter end. With 1 parity bit and n-bit data, total $n + 1$ bit will be transmitted. At the receiving end $n + 1$ parity checker circuit will be used to check correctness of the data.

For even parity transmission, parity bit will be made 1 or 0 based on the data, so that total $n + 1$ bits will have even number of 1's. For example, if we want to transmit data 1011 by even parity transmission, then we will use parity bit as 1, so data will have even number of 1's, i.e., data transmitted will be 11011. At the receiving end this data will be received and checked for even number of ones.

To transmit data $B_3B_2B_1B_0$ using even parity, we will transmit sequence $P\,B_3B_2B_1B_0$, where $P = B_3 \oplus B_2 \oplus B_1 \oplus B_0$. (Equation for parity generator)

At the receiving end we will check data received $PB_3B_2B_1B_0$ for error, $E = P \oplus B_3 \oplus B_2 \oplus B_1 \oplus B_0$ (equation for parity checker). If $E = 0$ (no error), or if $E = 1$ (1 bit error).

We use EX-OR gates for even parity generator/checker as EX-OR of bits gives output 1 if there are odd number of 1's else EX-OR output is 0.

Odd parity generator/checker is complement of even parity generator/checker. Odd parity circuits check for presence of odd number of 1's in data.

Code Converters

There are many situations where it is desired to convert from one code to another within a system. For example, the information from output of an analog to digital converter is often in gray code, before it can be processed in arithmetic unit, conversion to binary is required.

Let us consider simple example of 3-bit binary to gray code converter. This will have input lines supplied by binary codes and output lines must generate corresponding bit combination in gray code. The combination circuit code converter performs this transformation by means of logic gates.

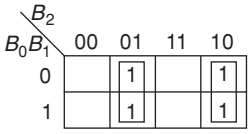
The output logic expression derived for code converter can be simplified by using the usual techniques including 'don't-care' if any present. For example, BCD code uses only codes from 0000 to 1001 and remaining combinations are treated as don't-care combinations. Similarly, EXS-3 uses only combinations from 0011 to 1100 and remaining combinations are treated as don't-care.

The relationship between the two codes is shown in the following truth table:

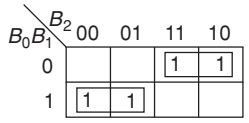
Decimal	B_2	B_1	B_0	G_2	G_1	G_0
0	0	0	0	0	0	0
1	0	0	1	0	0	1
2	0	1	0	0	1	1
3	0	1	1	0	1	0
4	1	0	0	1	1	0
5	1	0	1	1	1	1
6	1	1	0	1	0	1
7	1	1	1	1	0	0

For conversion we have to find out minimized functions of

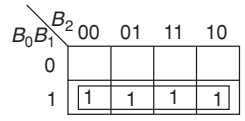
$$G_2(B_2, B_1, B_0) = \sum m(4, 5, 6, 7)$$
$$G_1(B_2, B_1, B_0) = \sum m(2, 3, 4, 5)$$
$$G_0(B_2, B_1, B_0) = \sum m(1, 2, 5, 6)$$



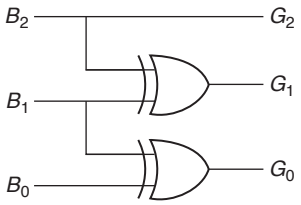
$$G_0(B_2, B_1, B_0) = B'_1B_0 + B_1B'_0 = B_1 \oplus B_0$$



$$G_1(B_2, B_1, B_0) = B'_1B_2 + B_1B'_2 = B_2 \oplus B_1$$



$$G_2(B_2, B_1, B_0) = B_2$$



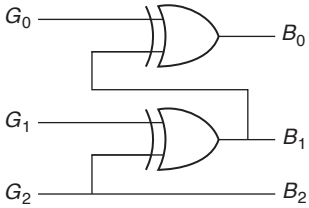
In similar fashion we can derive n-bit binary to gray code conversion as

$$G_n = B_n$$
$$G_{n-1} = B_{n-1} \oplus B_n$$
$$G_{i-1} = B_{i-1} \oplus B_i$$

Thus conversion can be implemented by $n - 1$ X-OR gates for n-bits.

For reverse conversion of gray to binary, by following similar standard principle of conversion, we will get

$$B_0 = G_0 \oplus G_1 \oplus G_2,\ B_1 = G_1 \oplus G_2,\ B_2 = G_2$$



In general for n-bit gray to binary code conversion

$$B_i = G_n \oplus G_{n-1} \oplus G_{n-2} \ldots \oplus G_{i-1} \oplus G_i$$

$B_n = G_n$ (MSB is same in gray and binary). It also requires $n-1$ X-OR gates for n-bits.

Example 2: Design 84-2-1 to XS-3 code converter.

Solution: Both 84-2-1 and *XS*-3 are *BCD* codes, each needs 4-bits to represent. The following table gives the relation between these codes. 84-2-1 is a weighted code, i.e., each position will have weight as specified. *XS*-3 is non-weighted code; the binary code is 3 more than the digit in decimal.

Decimal	84-2-1 $B_3B_2B_1B_0$	*XS*-3 $X_3X_2X_1X_0$
0	0000	0011
1	0111	0100
2	0110	0101
3	0101	0110
4	0100	0111
5	1011	1000
6	1010	1001
7	1001	1010
8	1000	1011
9	1111	1100

We will consider minterm don't-care combinations as 1, 2, 3, 12, 13, 14. For these combinations 84-2-1 code will not exist and the remaining minterms can be found from truth table.

$$X_0(B_3, B_2, B_1, B_0) = \sum m(0, 4, 6, 8, 10) + \sum \Phi(1, 2, 3, 12, 13, 14) = \overline{B_0}$$

$$X_1(B_3, B_2, B_1, B_0) = \sum m(0, 4, 5, 8, 9, 15) + \sum \Phi(1, 2, 3, 12, 13, 14) = \overline{B_1}$$

$$X_2(B_3, B_2, B_1, B_0) = \sum m(4, 5, 6, 7, 15) + \sum \Phi(1, 2, 3, 12, 13, 14) = B_2$$

$$X_3(B_3, B_2, B_1, B_0) = \sum m(8, 9, 10, 11, 15) + \sum \Phi(1, 2, 3, 12, 13, 14) = B_3$$

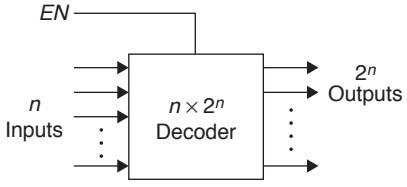
Decoder

A binary code of n-bits is capable of representing up to 2^n elements of distinct elements of coded information.

The three inputs are decoded into eight outputs, each representing one of the minterms of the three input variables.

A decoder is a combinational circuit that converts binary information from n input lines to a maximum 2^n unique output lines.

A binary decoder will have n inputs and 2^n outputs.



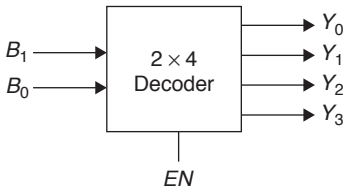


Figure 11 2 × 4 decoder

Table 5 *Truth Table*

EN	B_1	B_0	Y_3	Y_2	Y_1	Y_0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

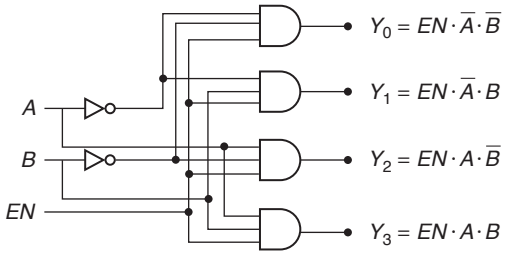


Figure 12 2 × 4 decoder

Decoder outputs are implemented by AND gates, but realization of AND gates at circuit level is done by the NAND gates (universal gates). So, the decoders available in IC form are implemented with NAND gates, i.e., the outputs are in complemented form and outputs are maxterms of the inputs rather than minterms of inputs as in AND gate decoders.

Furthermore, decoders include one or more enable inputs to control the circuit operation. Enable can be either active low/high input.

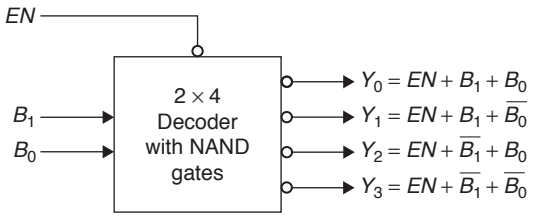


Figure 13 Active low 2 × 4 decoder

Table 6 *Truth Table*

EN	B_1	B_0	Y_3	Y_2	Y_1	Y_0
1	X	X	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

The block diagram shown here is 2 × 4 decoder with active low output and active low enable input.

The logic diagram is similar to the previous 2 × 4 decoder, except, all AND gates are replaced by NAND gates and *EN* will have inverter, *EN* is connected to all NAND inputs, as $\overline{EN}$ is active low input for this circuit.

The decoder is enabled when EN is equal to 0.

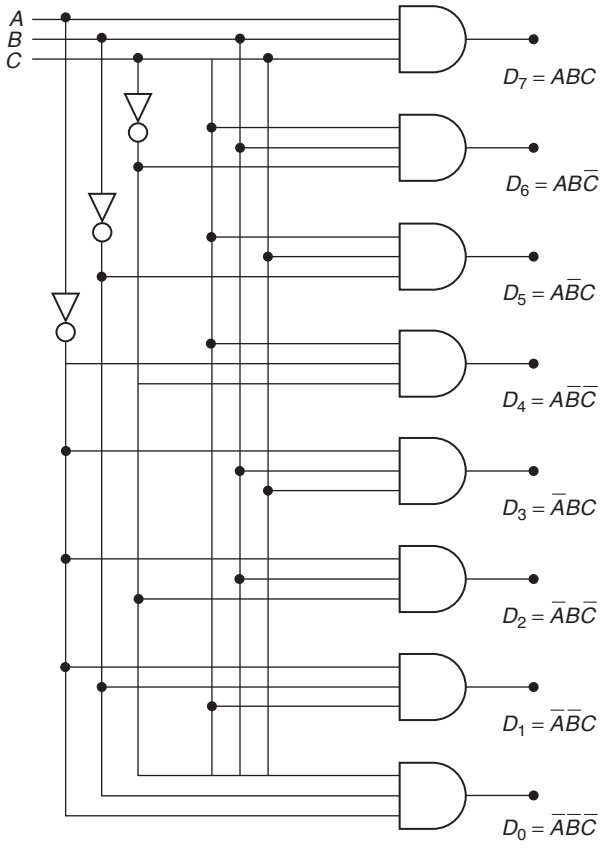
As shown in the truth table, only one output can be equal to 0 at any given time, all other outputs are equal to 1. The output whose value is equal to 0 represents the minterm selected by inputs, enable.

Consider a 3–8 line decoder

Table 7 *Truth Table*

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

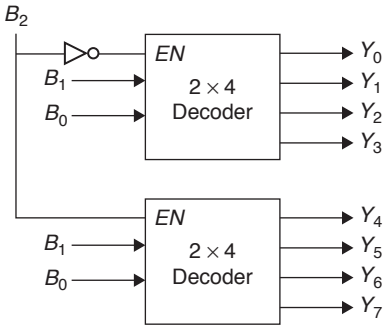
A 3–8 decoder has 3 input lines and 8 output lines, based on the combination of inputs applied for the 3 inputs, one of the 8 output lines will be made logic 1 as shown in the truth table. So, each output will have only one minterm.



Designing High Order Decoders from Lower Order Decoders

Decoder with enable input can be connected together to form larger decoder circuit.

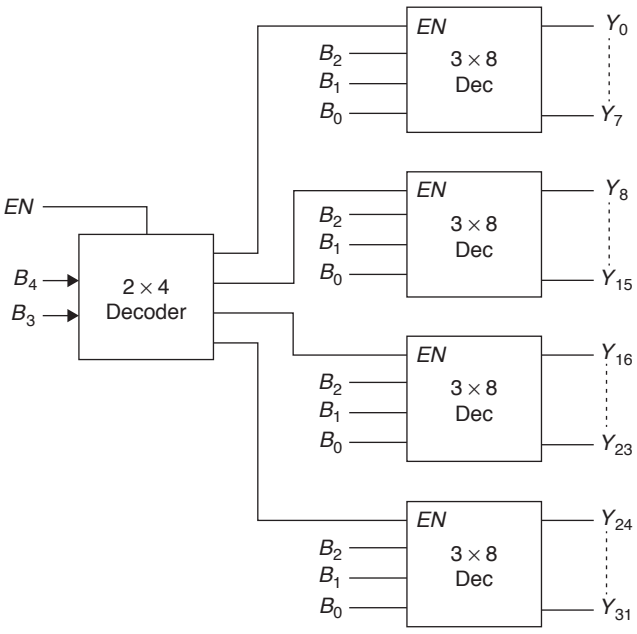
The following configuration shows 3 × 8 decoder with 2 × 4 decoders.



When $B_2 = 0$, top decoder is enabled and other is disabled, for 000–011 inputs, outputs are Y_0–Y_3, respectively, and other outputs are 0.

For $B_2 = 1$, the enable conditions are reversed.

The bottom decoder outputs generates minterms 100–111, while the outputs of top decoder are all 0's. 5 × 32 decoder with 3 × 8 decoders, 2 × 4 decoders



5 × 32 decoder will have 5 inputs $B_4 B_3 B_2 B_1 B_0$. 3 × 8 decoder will have 8 outputs, so 5 × 32 requires four 3 × 8 decoders, and we need one of the 2 × 4 decoders to select one 3 × 8 decoders and the connections are as shown in the circuit above.

Combinational Logic Implementation

An $n \times 2^n$ decoder provides 2^n minterms of n input variables. Since any Boolean function can be expressed in sum-of-minterms form, a decoder that generates the minterms of

the function, together with an external OR gate that forms their logical sum, provides a hardware implementation of the function.

Similarly, any function with n inputs and m outputs can be implemented with $n \times 2^n$ decoders and m OR gates.

Example 3: Implement full adder circuit by using 2×4 decoder.

Sum = Σ (1, 2, 4, 7), Carry = Σ (3, 5, 6, 7)

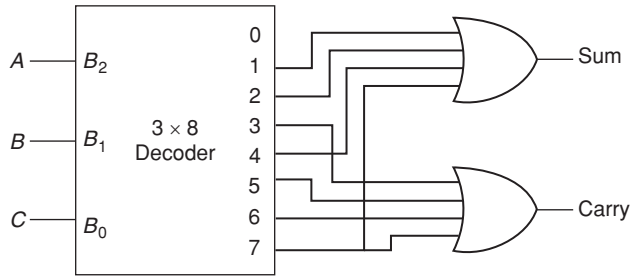


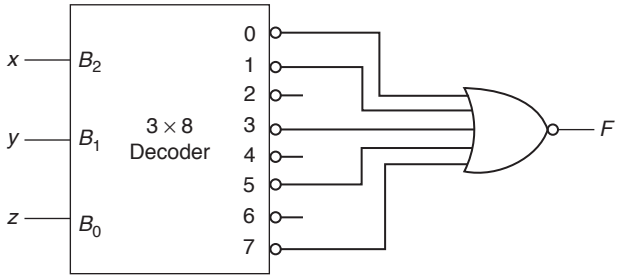
Figure 14 Implementation of full adder circuit with decoder

The 3×8 decoder generates the 8 minterms for A, B, and C. The OR gate for output sum forms the logical sum of minterms 1, 2, 4 and 7. The OR gate for output carry forms the logical sum of minterms 3, 5, 6 and 7.

Example 4: The minimized SOP form of output $F(x, y, z)$ is

(A) $x'y + z'$ (B) $x'y' + z'$

(C) $x'y' + z'$ (D) $x' + y'z$



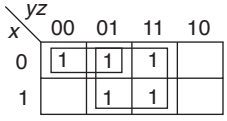
Solution: (C)

The outputs of decoder are in active low state. So, we can express outputs as $\overline{Y_7}, \overline{Y_6} \cdots \overline{Y_0}$

Outputs 0, 1, 3, 5, 7 are connected to NAND gate to form function $F(x, y, z)$

So
$$F = \overline{\overline{Y_0} \cdot \overline{Y_1} \cdot \overline{Y_3} \cdot \overline{Y_5} \cdot \overline{Y_7}}$$
$$= Y_0 + Y_1 + Y_3 + Y_5 + Y_7$$
$$= \Sigma(0, 1, 3, 5, 7)$$

By using K-maps

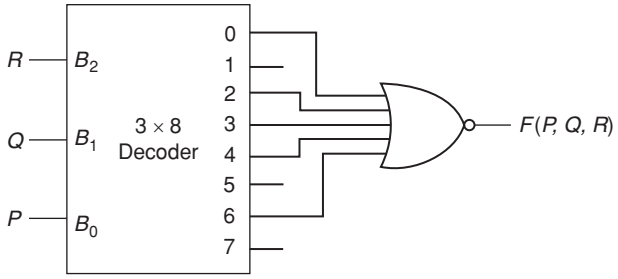


$F = z + x'y'$

Example 5. The minimal *POS* form of output function $f(P, Q, R)$ is

(A) $P\overline{Q} + PR$ (B) $P + \overline{Q}R$

(C) $P(\overline{Q} + R)$ (D) $Q(\overline{P} + R)$



Solution: (C)

The outputs of decoder are in normal form. 0, 2, 3, 4, 6 outputs are connected to NOR gate to form $F(P, Q, R)$

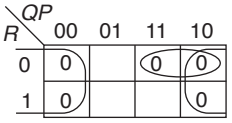
So
$$F = \overline{Y_0 + Y_2 + Y_3 + Y_4 + Y_6}$$
$$= \overline{Y_0} \cdot \overline{Y_2} \cdot \overline{Y_3} \cdot \overline{Y_4} \cdot \overline{Y_6}$$

$Y_0, Y_1, \ldots, Y_7$ indicate minterms, whereas $\overline{Y_0}, \overline{Y_1}, \cdots, \overline{Y_7}$ are maxterms.

So $F = \pi(0, 2, 3, 4, 6)$

Here, from the decoder circuit MSB is R, LSB is P.

By using K-map



$$F(P, Q, R) = P(R + \overline{Q})$$

Encoders

It is a digital circuit that performs the inverse operation of a decoder.

An encoder has 2^n (or fewer) input lines and n output lines.

It is also known as an octal to binary converter.

Consider an 8–3 line encoder:

Table 8 *Truth Table*

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

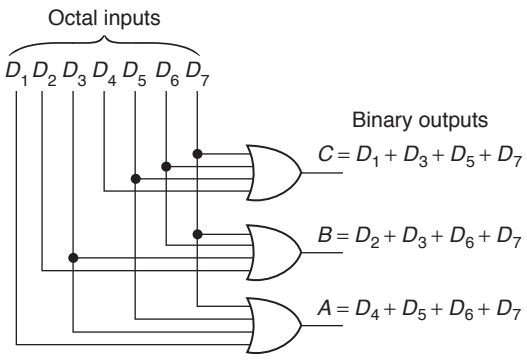


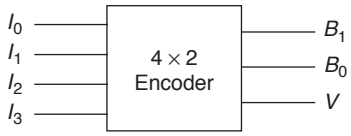
Figure 15 Logic diagram

Priority Encoder

A priority encoder is an encoder circuit that includes the priority function.

When two or more inputs are present, the input with higher priority will be considered.

Consider the 4 × 2 priority encoder.



I_3	I_2	I_1	I_0	B_1	B_0	V
1	X	X	X	1	1	1
0	1	X	X	1	0	1
0	0	1	X	0	1	1
0	0	0	1	0	0	1
0	0	0	0	X	X	0

I_3–I_0 are inputs and B_1 B_0 are binary output bits, valid (V) output is set to 1, when at least one input is present at input ($I_3 - I_0$).

When there is no input present, ($I_3 - I_0 = 0000$) then $V = 0$, for this combination the output B_1B_0 will not be considered.

The higher the subscript number, the higher the priority of the input. Input I_3 has the highest priority, I_2 has the next priority level. Input I_0 has lowest priority level. The Boolean expressions for output B_1 B_0 are

$$B_1 = I_3 + \overline{I}_3 I_2$$
$$= I_3 + I_2$$
$$B_0 = I_3 + \overline{I}_3 \overline{I}_2 I_1$$
$$= I_3 + \overline{I}_2 I_1$$
$$V = I_3 + I_2 + I_1 + I_0$$

Multiplexer

A multiplexer (MUX) is a device that allows digital information from several sources to be converted on to a single line for transmission over that line to a common destination.

The MUX has several data input lines and a single output line. It also has data select inputs that permits digital data on any one of the inputs to be switched to the output line.

Depending upon the binary code applied at the selection inputs, one (out of 2^n) input will be gated to single output. It is one of the most widely used standard logic circuits in digital design. The applications of multiplexer include data selection, data routing, operation sequencing, parallel to serial conversion, and logic function generation.

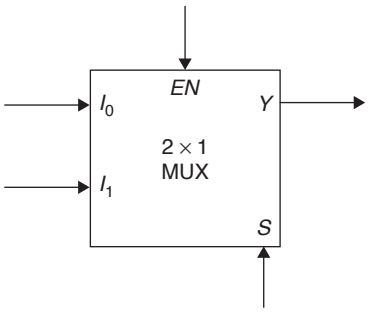
2^n inputs will be controlled by n selection lines and multiplexer will have 1 output, we denote it as $2^n \times 1$ multiplexer (data selector).

In other words, a multiplexer selects 1 out of n input data sources and transmits the selected data to a single output channel, this is called as multiplexing.

Basic 2 × 1 Multiplexer

The figure shows 2 × 1 multiplexer block diagram; it will have 2 inputs—I_0 and I_1, one selection line S, and one output Y. The function table is as shown here.

EN	S	Y
0	x	0
1	0	I_0
1	1	I_1

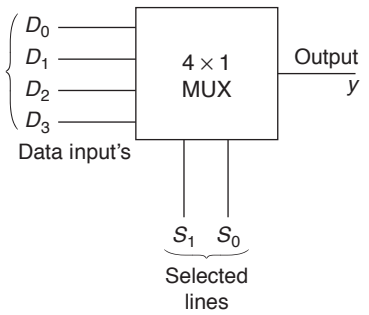


The output equation of 2 × 1 multiplexer is $Y = EN(I_0\overline{S} + I_1 S)$.

When enable is 1, the multiplexer will work in normal mode, else the multiplexer will be disabled.

Sometimes enable input will be active low enable $\overline{EN}$, then $Y = \overline{EN}(I_0\overline{S} + I_1 S)$.

The 4 × 1 Multiplexer



If a binary zero $S_1 = 0$ and $S_0 = 0$ as applied to the data select line the data input D_0 appear on the data output line and so on.

S_1	S_0	y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$$y = \overline{S_1}\ \overline{S_0} D_0 + \overline{S_1} S_0 D_1 + S_1 \overline{S_0} D_2 + S_1 S_0 D_3$$

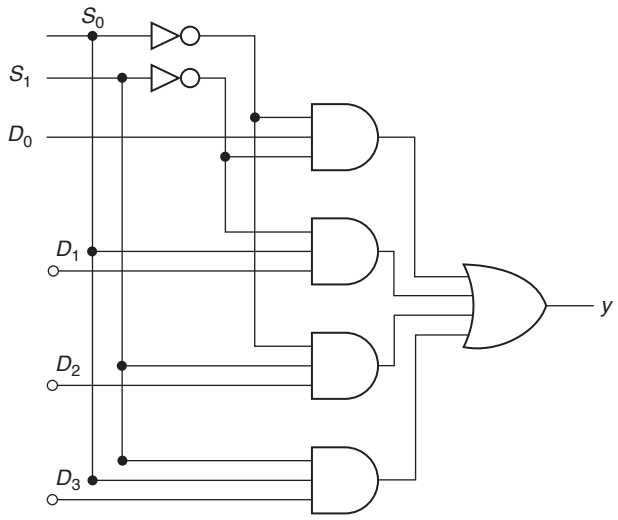


Figure 16 Logic diagram

For 8 × 1 multiplexer with 8 inputs from I_0–I_7 based on selection inputs $S_2 S_1 S_0$, the equation for output

$$Y = I_0 \overline{S_2} \overline{S_1} \overline{S_0} + I_1 \overline{S_2} \overline{S_1} S_0 + I_2 \overline{S_2} S_1 \overline{S_0} + I_3 \overline{S_2} S_1 S_0 + I_4 S_2 \overline{S_1} \overline{S_0} + I_5 S_2 \overline{S_1} S_0 + I_6 S_2 S_1 \overline{S_0} + I_7 S_2 S_1 S_0$$

From multiplexer equation, we can observe, each input is associated with its minterm (in terms of selection inputs).

Basic Gates by Using MUX

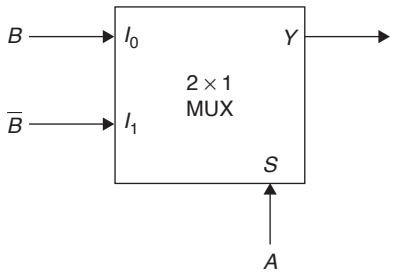


Figure 17 X-OR gate by using 2 × 1 MUX

$Y = \overline{A}B + A\overline{B}$ = X-OR gate, we can interchange inputs A and B also,

By interchanging inputs I_0 and I_1, $Y = \overline{A}\,\overline{B} + AB$, X-NOR gate.

Similarly, we can build all basic gates by using 2 × 1 multiplexer.

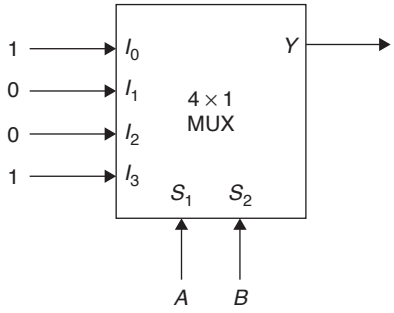
Example 6: If $I_0 = 1$, $I_1 = 0$, $S = A$, then Y is

Solution: $Y = (I_0 \overline{S} + I_1 S) = \overline{A}$. It Implements NOT gate.

Example 7: What should be the connections to implement NAND gate by using 2 × 1 MUX?

Solution: $Y = \overline{AB} = \overline{A} + \overline{B} = \overline{A} + A\overline{B} = 1 \cdot \overline{A} + \overline{B} \cdot A$

By considering $I_0 = 1$, $I_1 = \overline{B}$, $S = A$, we can implement NAND gate, or by interchanging A and B also we can get the same answer.



For the above 4 × 1 multiplexer $Y = \overline{AB} + AB$ = X-NOR gate, similarly to implement 2 input gates by using 4 × 1 multiplexer, the inputs I_0, I_1, I_2, I_3 should be same as the terms in the truth table of that gate.

Logic Function Implementation by Using Multiplexer

Let us consider a full subtractor circuit (borrow) to be implemented by using multiplexer.

Full subtractor borrow (B) is a function of 3 inputs X, Y, Z. The truth table is

X	Y	Z	B	4 × 1 MUX	2 × 1 MUX
0	0	0	0	$B = Z$	$B = Y + Z$
0	0	1	1		
0	1	0	1	$B = 1$	
0	1	1	1		
1	0	0	0	$B = 0$	$B = YZ$
1	0	1	0		
1	1	0	0	$B = Z$	
1	1	1	1		

To implement borrow by using 8 × 1 multiplexer, connect the three variables X, Y, Z directly to selection lines of the multiplexer, and connect the corresponding values of B to inputs, i.e., for $I_0 = 0$, $I_1 = 1$, $I_2 = 1$, etc. as per above truth table.

To implement borrow by using 4 × 1 multiplexer, connect any two variables to selection lines (in this case X, Y) and write output (B) in terms of other variable, for $XY = 00$, output B is same as Z, so connect $I_0 = Z$, similarly 1, 0, Z for remaining inputs.

To implement the function by using 2 × 1 multiplexer, connect 1 variable as selection line (in this case consider X) and write output (B) in terms of other variables, for $X = 0$,

output B is varies as $B = Y + Z$, so connect $I_0 = Y + Z$. For $X = 1$, output B varies as $B = YZ$, connect $I_1 = YZ$.

N–variable function can be implemented by using $2^{N-1} \times 1$ multiplexer without any extra hardware.

Implementation of Higher Order Multiplexer by Using Lower Order Multiplexers

By using lower order multiplexers, we can implement higher order multiplexers, for example by using 4×1 multiplexer, we can implement 8×1 MUX or 16×1 MUX or other higher order multiplexers.

Let us consider implementation of 16×1 MUX by using 4×1 MUX. 16×1 MUX will have inputs I_0–I_{15} and selection lines S_0–S_3, whereas 4×1 MUX will have only 4 input lines, and 2 selection lines, so we require four 4×1 MUX to consider all inputs I_0–I_{15}, and again to select one of the four outputs of these four multiplexers one more 4×1 multiplexer is needed (for which we will connect higher order selection lines S_2 and S_3). So, total of 5, 4×1 multiplexers are required to implement 16×1 MUX.

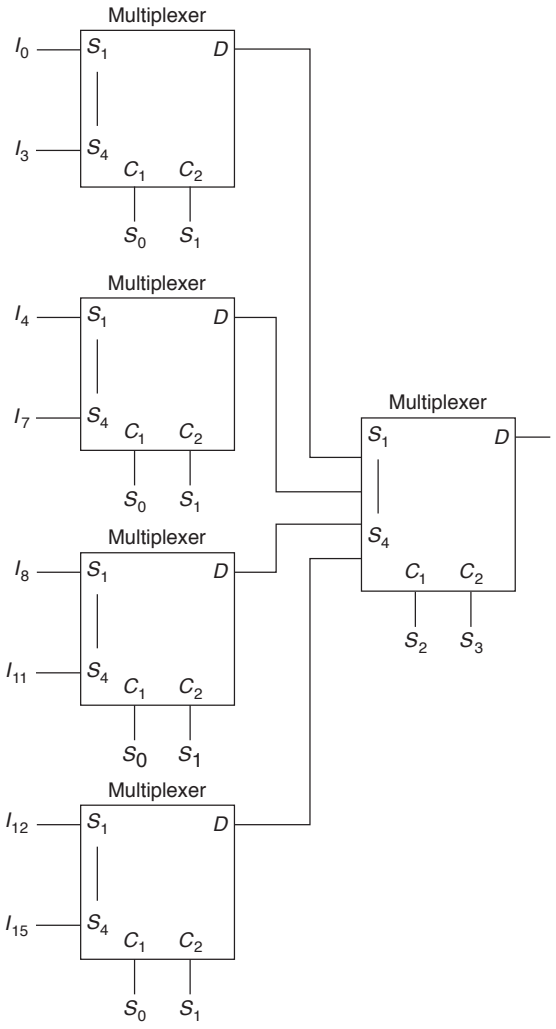


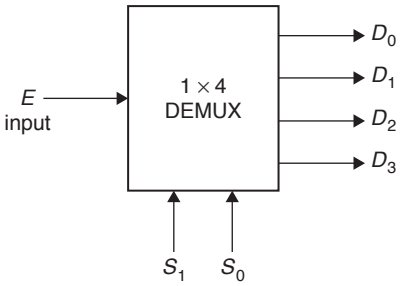
Figure 18 Realization of 16 x 1 multiplexer by using 4 x 1 multiplexers

In a similar fashion, to design 4×1 MUX, we require 3, 2×1 multiplexers, and to design 8×1 multiplexer, we require 7, 2×1 multiplexers.

Demultiplexer

The demultiplexer [DeMUX] basically serves opposite of the multiplexing function. It takes data from one line and distributes them to a given number of output lines.

The other name for demultiplexer is data distributor, as it receives information on a single line and distributes it to a possible 2^n output lines, where n is the number of selection lines, and value of n selects the line.



S_1	S_0	D_3	D_2	D_1	D_0
0	0	0	0	0	E
0	1	0	0	E	0
1	0	0	E	0	0
1	1	E	0	0	0

When $S_1S_0 = 10$; D_2 will be same as input E, and other outputs will be maintained at zero (0).

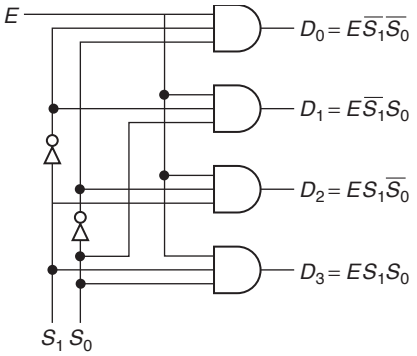
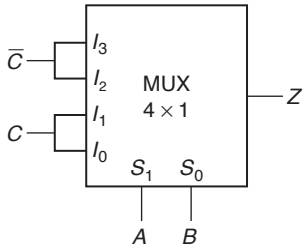


Figure 17 Logic diagram

Solved Examples

Example 1: The multiplexer shown in the figure is a 4 : 1 multiplexer. The output z is

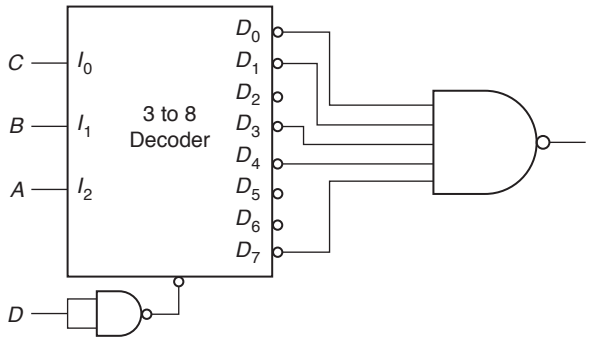


Solution:

A_1	B_0	Z
0	0	C
0	1	C
1	0	$\overline{C}$
1	1	$\overline{C}$

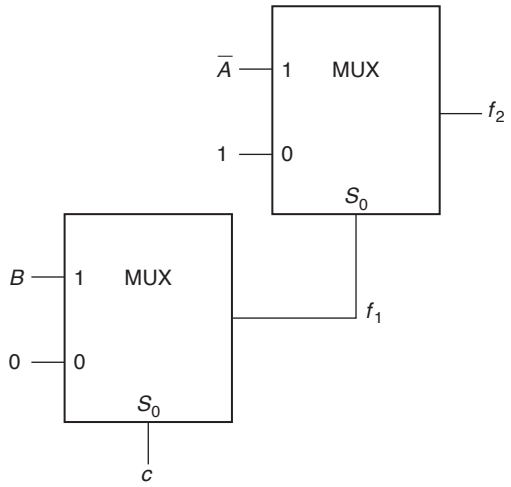
$$\therefore \quad Z = \overline{A}\,\overline{B}C + \overline{A}BC + AB\overline{C} + A\overline{B}\,\overline{C}$$
$$= \overline{B}(\overline{A}C + A\overline{C}) + B(\overline{A}C + A\overline{C})$$
$$= (\overline{A}C + A\overline{C})(B + \overline{B})(x + \overline{x} = 1)$$
$$\therefore \quad = \overline{A}C + A\overline{C} = A \oplus C$$

Example 2: The logic circuit shown in figure implements



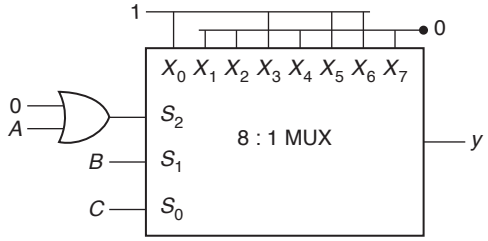
Solution: $z = D(\overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}BC + A\overline{B}\,\overline{C} + ABC)$
$= D(\overline{A}\,\overline{B}(C + \overline{C}) + BC(\overline{A} + A) + A\overline{B}\,\overline{C})$
$\times D(\overline{B}\,\overline{A} + \overline{B}\,\overline{C} + BC)$
$= D(B \,\Theta\, C + \overline{A}\,\overline{B})$

Example 3. The network shown in figure implements



Solution: $f_1 = \overline{C}0 + CB = CB, f_1 = CB$
$F_2 = \overline{f_1} + f_1\overline{A} = \overline{A} \cdot CB + \overline{CB}$
$= \overline{A} + \overline{CB}$
$= \overline{A} + \overline{C} + \overline{B} = \overline{ABC}$
$\therefore$ NAND Gate

Example 4: In the TTL circuit in figure, S_2–S_0 are select lines and x_7–x_0 are input lines. S_0 and X_0 are LSBs. The output Y is

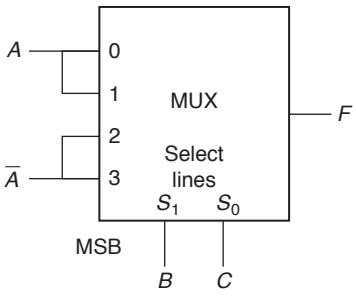


Solution: $S_2 = A, S_1 = B, S_0 = C$

$S_2(A)$	$S_1(B)$	$S_0(C)$	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$$Y = \overline{A}\,\overline{B}\,\overline{C} + AB\overline{C} + A\overline{B}C + \overline{A}BC$$
$$= \overline{C}(\overline{A}\,\overline{B} + AB) + C(A\overline{B} + \overline{A}B)$$
$$Y = \overline{C}(\overline{A} \oplus B) + C(A \oplus B) = A \oplus B \odot C$$

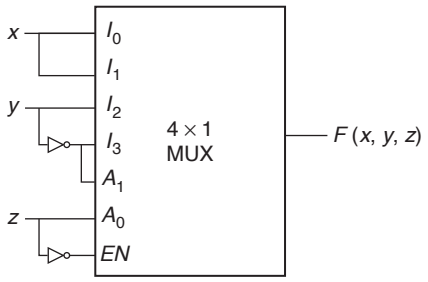
Example 5: The logic realized by the adjoining circuit is



Solution: $F = \overline{B}\,\overline{C}A + \overline{B}CA + B\overline{C}\,\overline{A} + BC\overline{A}$
$\times \overline{C}(\overline{B}A + B\overline{A}) + C(\overline{B}A + B\overline{A})$
$\times A\overline{B} + \overline{A}B(C + \overline{C}) = A \oplus B$

Example 6: Consider the following multiplexer, where I_0, I_1, I_2, I_3 are four date input lines selected by two address line combinations A_1A_0 = 00, 01, 10, 11, respectively and f

is the output of the multiplexer. *EN* is the enable input, the function $f(x, y, z)$ implemented by the below circuit is



Solution: $A_1 = \bar{y} \cdot A_0 = z, EN = \bar{z}$

A_1	A_0	S	I
0	0	$(y\bar{z})$	x
0	1	$(y\ z)$	x
1	0	$(\bar{y}\ \bar{z})$	y
1	1	$(\bar{y} z)$	$\bar{y}$

$$f(x, y, z) = \text{S.I} = (xy + 0 + \bar{y}z) \cdot EN$$
$$= xy \cdot \bar{z}$$

EXERCISES

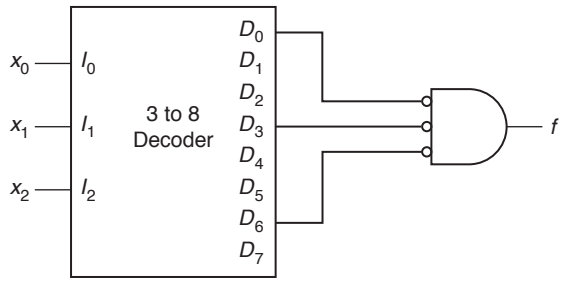
Practice Problems I

Directions for questions 1 to 21: Select the correct alternative from the given choices.

1. The binary number 110011 is to be converted to gray code. The number of gates and type required are
(A) 6, AND (B) 6, X-NOR
(C) 6, X-OR (D) 5, X-OR

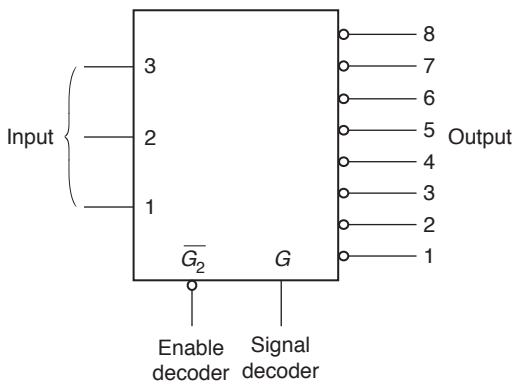
2. The number of 4-to 16-line decoder required to make an 8- to 256-line decoder is
(A) 16 (B) 17
(C) 32 (D) 64

3. $f(x_2, x_1, x_0) = ?$



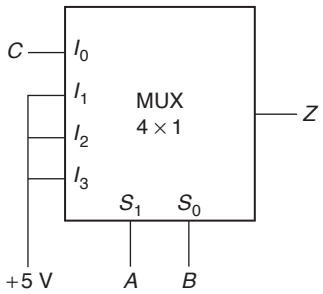
(A) $\pi(1, 2, 4, 5, 7)$ (B) $\Sigma(1, 2, 4, 5, 7)$
(C) $\Sigma(0, 3, 6)$ (D) $\pi(0, 2, 3, 6)$

4. A 3-to-8 decoder is shown below



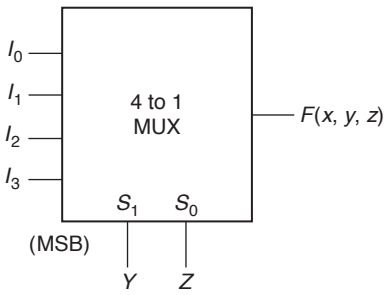
All the output lines of the chip will be high except pin 8, when all the inputs 1, 2, and 3
(A) are high; and G, G_2 are low
(B) are high; and G is low G_2 is high
(C) are high; and G, G_2 are high
(D) are high; and G is high G_2 is low

5. The MUX shown in figure is 4 × 1 multiplexer the output z is



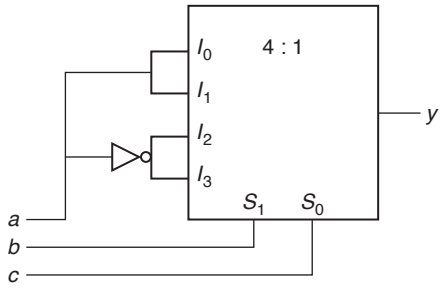
(A) $A\ B\ C$
(B) $A \oplus B \oplus C$
(C) $A\ \Theta\ B\ \Theta\ C$
(D) $A + B + C$

6. If a 4 to 1 MUX (shown below) realizes a three variable function $f(x, y, z) = xy + x\bar{z}$ then which of the following is correct?



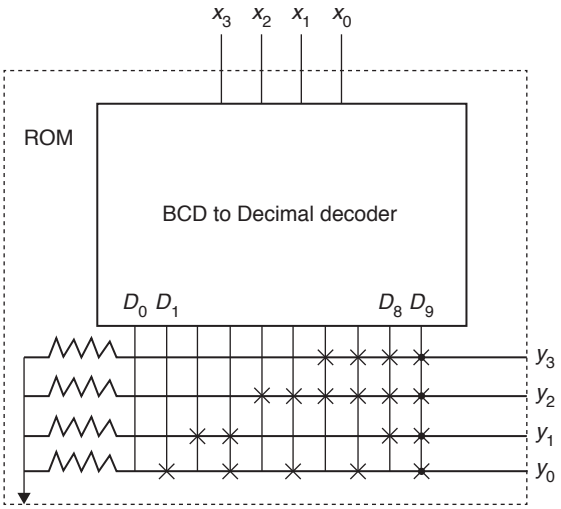
(A) $I_0 = X, I_1 = 0, I_2 = X, I_3 = X$
(B) $I_0 = 0, I_1 = 1, I_2 = Y_1, I_3 = X$
(C) $I_0 = X, I_1 = 1, I_2 = 0, I_3 = X$
(D) $I_0 = X, I_1 = 0, I_2 = X, I_3 = Z$

7. The circuit shown in the figure is same as



(A) two input NAND gate with a and c inputs
(B) two input NOR gate with a and c inputs
(C) two input X-OR gates with a and b inputs
(D) two input X-NOR gate with b and c inputs

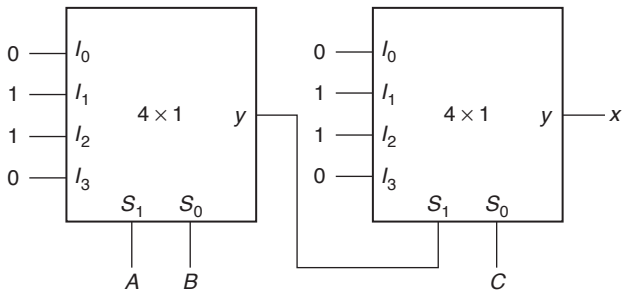
8. If the input x_3, x_2, x_1, x_0 to the ROM in the figure are 8421 BCD numbers, then the outputs y_3, y_2, y_1, y_0 are



(A) gray code numbers (B) 2421 *BCD*
(C) Excess – 3 code numbers (D) 84–2–1

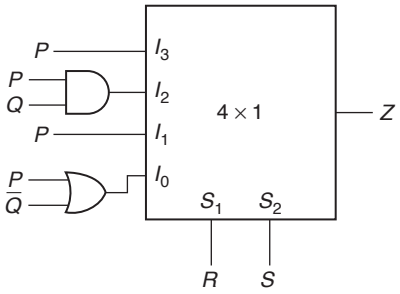
9. A 4-bit parallel full adder without input carry requires
(A) 8 HA, 4 OR gates (B) 8 HA, 3 OR gates
(C) 7 HA, 4 OR gates (D) 7 HA, 3 OR gates

10. In the circuit find X.



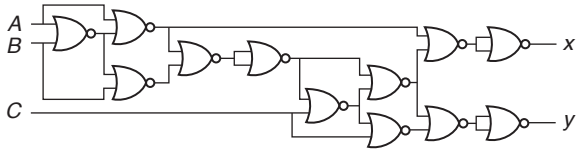
(A) $A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}C + ABC$
(B) $\bar{A}BC + A\bar{B}C + AB\bar{C} + \bar{A}\bar{B}\bar{C}$
(C) $AB + BC + AC$
(D) $\bar{A}\bar{B} + \bar{B}\bar{C} + \bar{A}\bar{C}$

11. Find the function implemented.



(A) $PQ + PS + \bar{Q}\bar{R}\bar{S}$
(B) $P\bar{Q} + PQ\bar{R} + \bar{P}\bar{Q}\bar{S}$
(C) $P\bar{Q}\bar{R} + \bar{P}QR + PQRS + \bar{Q}\bar{R}\bar{S}$
(D) $PQ\bar{R} + PQR\bar{S} + P\bar{Q}\bar{R}S + \bar{Q}\bar{R}\bar{S}$

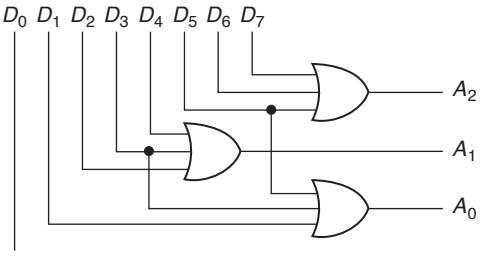
12. Which function is represented by the given circuit?



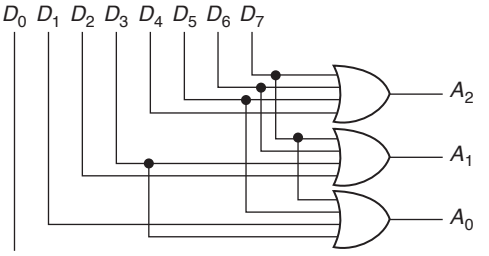
(A) Full adder (B) Full subtractor
(C) Comparator (D) Parity generator

13. Which of the following represents octal to binary encoder?

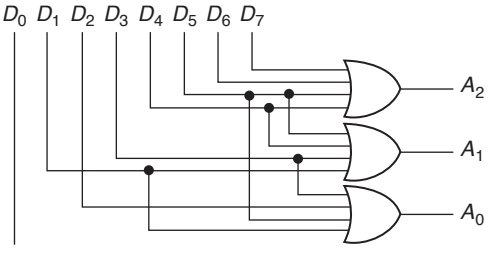
(A)



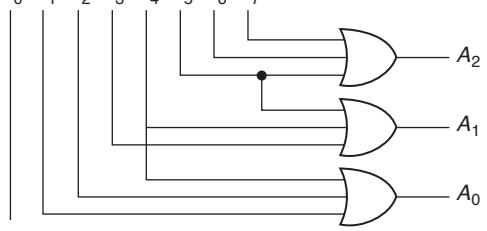
(B)



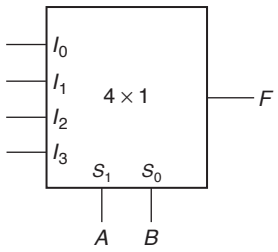
(C)



(D)



14. For a MUX to function as a full adder what should be the input provided to the I_0, I_1, I_2, I_3 if the A and B are the select lines?



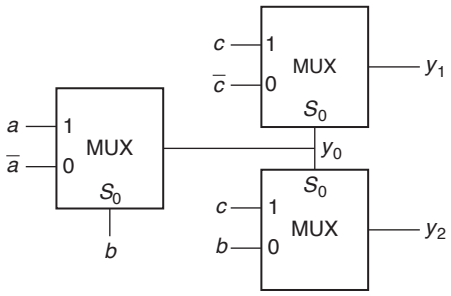
(A) $I_0 = I_1 = C_{in};\ I_2 = I_3 = \overline{C_{in}}$

(B) $I_0 = I_1 = \overline{C_{in}};\ I_2 = I_3 = C_{in}$

(C) $I_0 = I_3 = C_{in};\ I_1 = I_2 = \overline{C_{in}}$

(D) $I_0 = I_3 = \overline{C_{in}};\ I_1 = I_2 = C_{in}$

15. The given circuit act as



(A) Full adder (B) Half adder

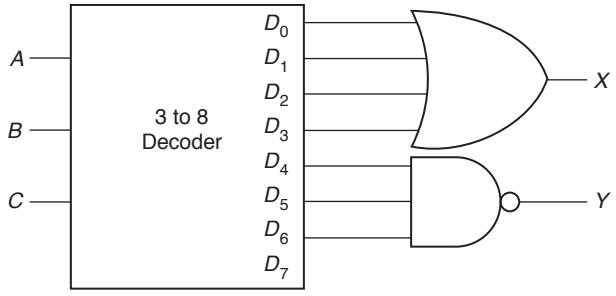
(C) Full subtractor (D) Half subtractor

16. For a 4 × 16 decoder circuit, the outputs of decoder ($y_0, y_1, y_4 \cdot y_5 \cdot y_{10} \cdot y_{11} \cdot y_{14} \cdot y_{15}$) are connected to 8 input NOR gate, the expression of NOR gate output is

(A) $A \oplus D$ (B) $A \odot D$

(C) $A \odot C$ (D) $A \oplus C$

17. The function implemented by decoder is



(A) $X = A'BC' + B'C', y = A + B$

(B) $X = A'C' + B'C', y = 1$

(C) $X = \overline{A},\ y = 0$

(D) $X = \overline{A},\ y = 1$

18. A relay is to operate with conditions that it should be on when the input combinations are 0000, 0010, 0101, and 0111. The states 1000, 1001, 1010 don't occur. For rest of the status, relay should be off. The minimized Boolean expression notifying the relationship is

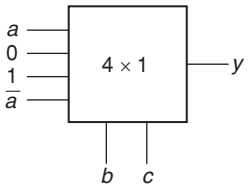
(A) $BC + ACD$

(B) $\overline{B}\overline{D} + \overline{A}BD$

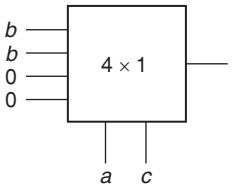
(C) $BD + AC$

(D) $AB + CD$

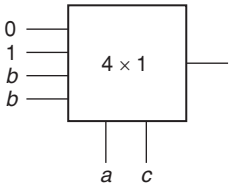
19. If a function has been implemented using MUX as shown, implement the same function with a and c as the select lines



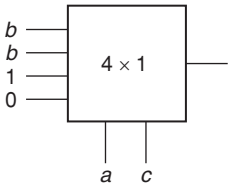
(A)



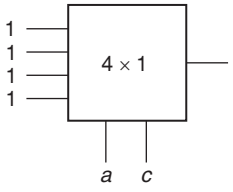
(B)



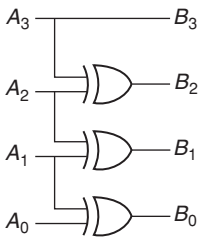
(C)



(D)



20. The circuit is used to convert one code to another. Identify it.



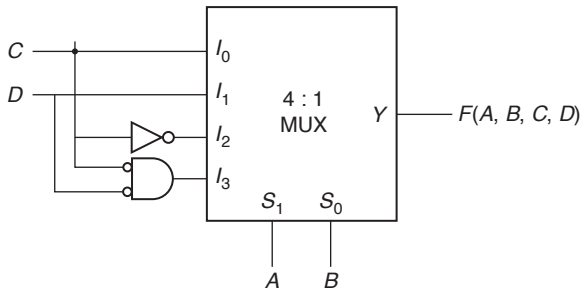
(A) Binary to gray

(B) Gray to binary

(C) Gray to XS–3

(D) Gray to 8421

21. The Boolean function realised by logic circuit is



(A) $F = \Sigma m(0, 1, 3, 5, 9, 10, 14)$

(B) $F = \Sigma m(2, 3, 5, 7, 8, 12, 13)$

(C) $F = \Sigma m(1, 2, 4, 5, 11, 14, 15)$

(D) $F = \Sigma m(2, 3, 5, 7, 8, 9, 12)$

Practice Problems 2

Directions for questions 1 to 21: Select the correct alternative from the given choices.

1. For a binary half subtractor having two input A and B, the correct set of logical expression for the outputs D = (A minus B) and X (borrow) are

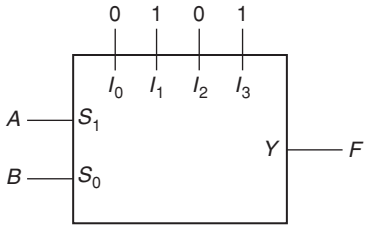
(A) $D = AB + \overline{AB}, X = \overline{A}B$

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

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

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

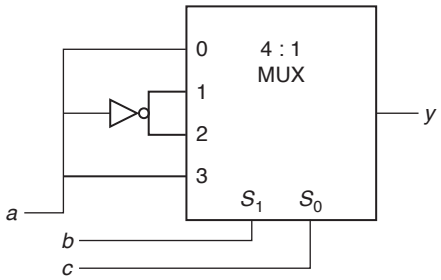
2. The function 'F' implemented by the multiplexer chip shown in the figure is



(A) A (B) B

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

3 The following multiplexer circuit is equal to



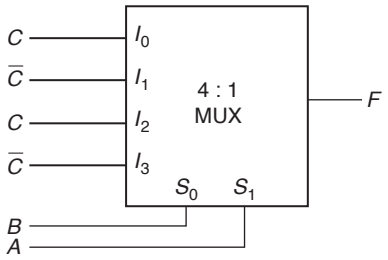
(A) implementation of sum equation of full adder

(B) implementation of carry equation of full adder

(C) implementation of borrow equation of full substractor

(D) all of the above

4 The output 'F' of the multiplexer circuit shown in the figure will be



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

(C) $A \oplus B$ (D) $B \oplus C$

5. Full subtractor can be implemented by using

(A) 3-to-8 line decoder only

(B) 3-to-8 line decoder and one OR gate

(C) 3-to-8 line decoder and two OR gates

(D) None

6. What are the difference and borrow equations for the above circuit?

(A) $D = x \Theta y \Theta z, B = x'y + yz + zx'$

(B) $D = X \oplus y \oplus z, B = xy + yz + zx$

(C) $D = x \oplus y \oplus z, B = x'y + yz + zx'$

(D) A and C both

7. Combinational circuits are one in which output depends _____, whereas sequential circuit's output depends _____

(A) only on present input, only on past input

(B) only on present input, only on past and future input

(C) only on present input, only on present input and past output

(D) on present input, on past and present output

8. The sum output of the half adder is given by (assume A and B as inputs)

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

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

9. MUX implements which of the following logic?

(A) NAND–XOR (B) AND–OR

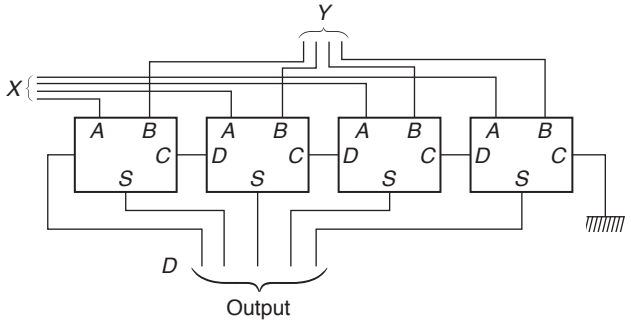
(C) OR–AND (D) XOR–NOT

10. A DeMUX can be used as a

(A) Comparator (B) Encoder

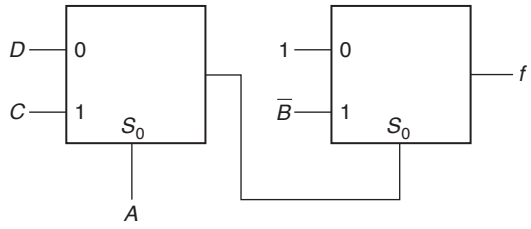
(C) Decoder (D) Adder

11. If we have inputs as A, B and C and output as S and D. We are given that $S = A \oplus B \oplus C$. $D = BC + \overline{A}B + \overline{A}C$. Which of the circuit is represented by it?



(A) 4-bit adder giving $X + Y$
(B) 4-bit subtractor giving $X - Y$
(C) 4-bit subtractor giving $Y - X$
(D) 4-bit adder giving $X + Y + S$

12. The Boolean function f implemented in the figure using two input multiplexers is



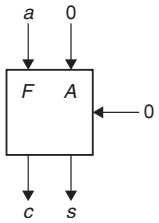
(A) $A\overline{C} + \overline{A}\overline{D} + \overline{D}C + \overline{A}\overline{B}D + A\overline{B}C$
(B) $\overline{A} + A\overline{C} + \overline{A}\overline{D} + \overline{D}\overline{C}$
(C) $\overline{B} + A\overline{C} + \overline{A}\overline{D} + \overline{D}\overline{C}$
(D) $AC + AD +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

13. The carry generate and carry propagate function of the look ahead carry adder is
(A) $CG = A + B$, $CP = A \oplus B$
(B) $CG = A \oplus B$, $CP = A + B$
(C) $CG = AB$, $CP = A \oplus B$
(D) $CG = AB$, $CP = A + B$

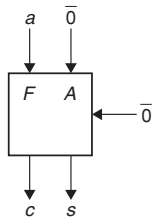
14. If we have a comparator and if E represents the condition for equality i.e., $(A_n \oplus B_n)$, if A_n and B_n are to be compared then the expression $A_3\overline{B}_3 + E_3A_2\overline{B}_2 + E_3E_2A_1\overline{B}_1 + E_3E_2E_1A.\overline{B}$. represents which of the condition for a 4-bit number?
(A) $A > B$ (B) $B > A$
(C) $A = B$ (D) None of these

15. When full adder is used to function as a 1-bit incrementor, which of the circuit configurations must be used?

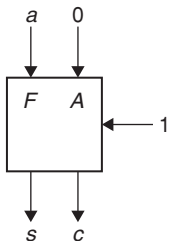
(A)



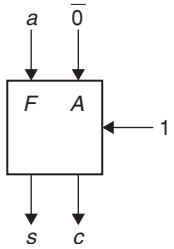
(B)



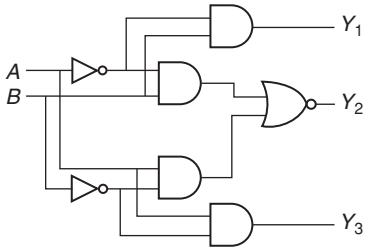
(C)



(D)



16. Identify the circuit.



(A) Half adder
(B) Full adder
(C) 1-bit magnitude comparator
(D) Parity generator

17. In order to implement n variable function (without any extra hardware) the minimum order of MUX is
(A) $2^n \times 1$ (B) $2^n \times 1$
(C) $(2^n - 1) \times 1$ (D) $(2^{n-1}) \times 1$

18. A full adder circuit can be changed to full subtractor by adding a
(A) NOR gate (B) NAND gate
(C) Inverter (D) AND gate

19. The half adder when implemented in terms of NAND logic is expressed as
(A) $A \oplus B$ (B) $\overline{A \cdot \overline{AB} \cdot B \cdot \overline{AB}}$
(C) $\overline{A \cdot \overline{AB}} \cdot \overline{B \cdot \overline{AB}}$ (D) $\overline{A \cdot \overline{ABB} \cdot \overline{AB}}$

20. For a DeMUX to act as a decoder, what is the required condition?
(A) Input should be left unconnected and select lines behave as a input to decoder
(B) Input should be always 0 and select line behave as inputs to decoder
(C) Both are same
(D) Input should become enable and select lines behave as inputs to decoder

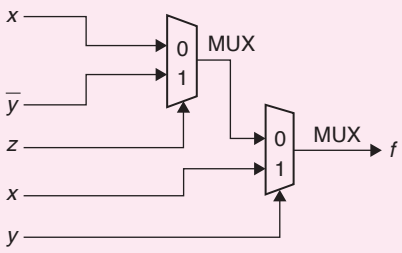
21. For a full subtractor, which of the combination will give the difference?
(A) $\overline{(A \oplus B)\overline{(A \oplus B)b_i}} \cdot \overline{b_i\overline{(A \oplus B)b_i}}$
(B) $\overline{B \cdot \overline{AB} \cdot \overline{b_i(A \oplus B)}}$
(C) $\overline{\overline{A} + \overline{B}} + \overline{b_i} + \overline{\overline{A \oplus B}}$
(D) None of these

Previous Years' Questions

1. A 4-bit carry look ahead adder, which adds two 4-bit numbers, is designed using AND, OR, NOT, NAND, NOR gates only. Assuming that all the inputs are available in both complemented and uncomplemented forms and the delay of each gate is one time unit, what is the overall propagation delay of the adder? Assume that the carry network has been implemented using two-level AND–OR logic. **[2004]**

(A) 4 time units (B) 6 time units
(C) 10 time units (D) 12 time units

2.



Consider the circuit above. Which one of the following options correctly represents $f(x, y, z)$? **[2006]**

(A) $x\bar{z} + xy + \bar{y}z$ (B) $x\bar{z} + xy + \overline{\bar{y}z}$
(C) $xz + xy + \overline{\bar{y}z}$ (D) $xz + x\bar{y} + \bar{y}z$

3. Given two 3-bit numbers $a_2a_1a_0$ and $b_2b_1b_0$ and c, the carry in, the function that represents the carry generate function when these two numbers are added is **[2006]**

(A) $a_2b_2 + a_2a_1b_1 + a_2a_1a_0b_0 + a_2a_0b_1b_0 + a_1b_2b_1 + a_1a_0b_2b_0 + a_0b_2b_1b_0$

(B) $a_2b_2 + a_2b_1b_0 + a_2a_1b_1b_0 + a_1a_0b_2b_1 + a_1a_0b_2 + a_1a_0b_2b_0 + a_2a_0b_1b_0$

(C) $a_2 + b_2 + (a_2 \oplus b_2)\,(a_1 + b_1 + (a_1 \oplus b_1)(a_0 + b_0))$

(D) $a_2b_2 + \overline{a_2}a_1b_1 + \overline{a_2a_1}a_0b_0 + \overline{a_2}a_0\overline{b_1}b_0 + a_1\overline{b_2}b_1 + \overline{a_1}a_0\overline{b_2}b_0 + a_0\overline{b_2b_1}b_0$

4. We consider the addition of two 2's complement numbers $b_{n-1}b_{n-2} \ldots b_0$ and $a_{n-1}a_{n-2} \ldots a_0$. A binary adder for adding unsigned binary numbers is used to add the two numbers. The sum is denoted by $c_{n-1}c_{n-2} \ldots c_0$ and the carry-out by c_{out}. Which one of the following options correctly identifies the overflow condition? **[2006]**

(A) $c_{out}\overline{(a_{n-1} \oplus b_{n-1})}$
(B) $a_{n-1}b_{n-1}\overline{c_{n-1}} + \overline{a_{n-1}}\,\overline{b_{n-1}}c_{n-1}$
(C) $c_{out} \oplus c_{n-1}$
(D) $a_{n-1} \oplus b_{n-1} \oplus c_{n-1}$

5. Consider numbers represented in 4-bit gray code. Let $h_3h_2h_1h_0$ be the gray code representation of a number n and let $g_3g_2g_1g_0$ be the gray code of $(n + 1)$ modulo 16 value of the number. Which one of the following functions is correct? **[2006]**

(A) $g_0(h_3h_2h_1h_0) = \Sigma(1, 2, 3, 6, 10, 13, 14, 15)$
(B) $g_1(h_3h_2h_1h_0) = \Sigma(4, 9, 10, 11, 12, 13, 14, 15)$
(C) $g_2(h_3h_2h_1h_0) = \Sigma(2, 4, 5, 6, 7, 12, 13, 15)$
(D) $g_3(h_3h_2h_1h_0) = \Sigma(0, 1, 6, 7, 10, 11, 12, 13)$

6. How many 3-to-8 line decoders with an enable input are needed to construct a 6-to-64 line decoder without using any other logic gates? **[2007]**

(A) 7 (B) 8
(C) 9 (D) 10

7. Suppose only one multiplexer and one inverter are allowed to be used to implement any Boolean function of n variables. What is the minimum size of the multiplexer needed? **[2007]**

(A) 2^n line to 1 line (B) 2^{n+1} line to 1 line
(C) 2^{n-1} line to 1 line (D) 2^{n-2} line to 1 line

8. In a look-ahead carry generator, the carry generate function G_i and the carry propagate function P_i for inputs A_i and B_i are given by:

$P_i = A_i \oplus B_i$ and $G_i = A_iB_i$

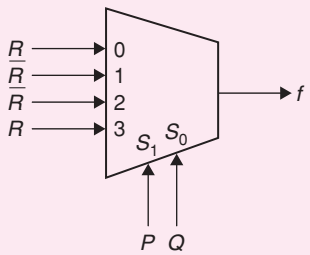
The expressions for the sum bit S_i and the carry bit C_{i+1} of the look-ahead carry adder are given by:

$S_i = P_i \oplus C_i$ and $C_{i+1} = G_i + P_iC_i$, where C_0 is the input carry.

Consider a two-level logic implementation of the look-ahead carry generator. Assume that all P_i and G_i are available for the carry generator circuit and that the AND and OR gates can have any number of inputs. The number of AND gates and OR gates needed to implement the look-ahead carry generator for a 4-bit adder with S_3, S_2, S_1, S_0 and C_4 as its outputs are respectively: **[2007]**

(A) 6, 3 (B) 10, 4
(C) 6, 4 (D) 10, 5

9. The Boolean expression for the output f of the multiplexer shown below is



(A) $\overline{P \oplus Q \oplus R}$
(B) $P \oplus Q \oplus R$
(C) $P + Q + R$
(D) $\overline{P + Q + R}$

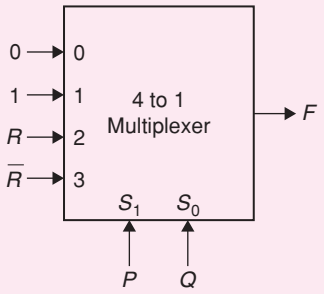
10. The amount of ROM needed to implement a 4-bit multiplier is **[2012]**
(A) 64 bits
(B) 128 bits
(C) 1K bits
(D) 2K bits

11. In the following truth table, $V = 1$ if and only if the input is valid.

Inputs				Outputs		
D_0	D_1	D_2	D_3	X_0	X_1	V
0	0	0	0	x	x	0
1	0	0	0	0	0	1
x	1	0	0	0	1	1
x	x	1	0	1	0	1
x	x	x	1	1	1	1

What function does the truth table represent? **[2013]**
(A) Priority encoder
(B) Decoder
(C) Multiplexer
(D) Demultiplexer

12. Consider the 4-to-1 multiplexer with two select lines S_1 and S_0 given below.



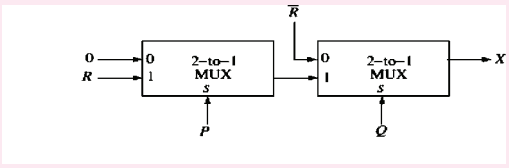
The minimal sum-of-products form of the Boolean expression for the output F of the multiplexer is **[2014]**
(A) $\bar{P}Q + Q\bar{R} + P\bar{Q}R$
(B) $\bar{P}Q + \bar{P}Q\bar{R} + PQ\bar{R} + P\bar{Q}R$
(C) $\bar{P}QR + \bar{P}Q\bar{R} + Q\bar{R} + P\bar{Q}R$
(D) $PQ\bar{R}\ \bar{P}QR + \bar{P}Q\bar{R} + Q\bar{R} + P\bar{Q}R$

13. Consider the following combinational function block involving four Boolean variables x, y, a, b, where x, a, b are inputs and y is the output. **[2014]**
$f(x, y, a, b)$
{
if (x is 1) $y = a$;
else $y = b$;
}
Which one of the following digital logic blocks is the most suitable for implementing this function?
(A) Full adder
(B) Priority encoder
(C) Multiplexer
(D) Flip–flop

14. Let $\oplus$ denote the Exclusive OR (X-OR) operation. Let '1' and '0' denote the binary constants. Consider the following Boolean expression for F over two variables P and Q:
$F(P, Q) = ((1 \oplus P) \oplus (P \oplus Q)) \oplus ((P \oplus Q) \oplus (Q \oplus 0))$ **[2014]**
The equivalent expression for F is
(A) $P + Q$
(B) $\overline{P + Q}$
(C) $P \oplus Q$
(D) $\overline{P \oplus Q}$

15. A half adder is implemented with XOR and AND gates. A full adder is implemented with two half adders and one OR gate. The propagation delay of an XOR gate is twice that of an AND/OR gate. The propagation delay of an AND/OR gate is 1.2 microseconds. A 4-bit ripple-carry binary adder is implemented by using four full adders. The total propagation time of this 4-bit binary adder in microseconds is _____ **[2015]**

16. Consider the two cascaded 2-to-1 multiplexers as shown in the figure.



minimal sum of products form of the output x is
The minimal sum of products form of the output X is **[2016]**
(A) $\bar{P}\ \bar{Q} + PQR$
(B) $\bar{P}\ Q + QR$
(C) PQ + $\bar{P}\ \bar{Q}\ R$
(D) $\bar{P}\ \bar{Q} + PQR$

17. When two 8-bit numbers $A_7 \ldots A_0$ and $B_7 \ldots B_0$ in 2's complement representation (with A_0 and B_0 as the least significant bits) are added using a **ripple-carry adder**, the sum bits obtained are $S_7 \ldots S_0$ and the carry bits are $C_7 \ldots C_0$. An overflow is said to have occurred if **[2017]**
(A) the carry bit C_7 is 1
(B) all the carry bits ($C_7, \ldots, C_0$) are 1
(C) $(A_7 \cdot B_7 \cdot \overline{S_7} + \overline{A_7} \cdot \overline{B_7} \cdot S_7)$ is 1
(D) $(A_0 \cdot B_0 \cdot \overline{S_0} + \overline{A_0} \cdot \overline{B_0} \cdot S_0)$ is 1

Answer Keys

Exercises

Practice Problems 1

1. D	**2.** B	**3.** B	**4.** D	**5.** D	**6.** A	**7.** C	**8.** B	**9.** D	**10.** A
11. A	**12.** B	**13.** B	**14.** C	**15.** C	**16.** D	**17.** D	**18.** B	**19.** C	**20.** A
21. D									

Practice Problems 2

1. C	**2.** B	**3.** A	**4.** D	**5.** C	**6.** C	**7.** C	**8.** B	**9.** B	**10.** C
11. B	**12.** C	**13.** C	**14.** A	**15.** C	**16.** C	**17.** B	**18.** C	**19.** C	**20.** D
21. A									

Previous Years' Questions

1. B	**2.** A	**3.** A	**4.** B	**5.** C	**6.** C	**7.** C	**8.** B	**9.** B	**10.** D
11. A	**12.** A	**13.** C	**14.** D	**15.** 19.2	**16.** D	**17.** C			

Chapter 4

Sequential Circuits

LEARNING OBJECTIVES

- *Sequential circuit*
- *Basic storage elements*
- *Latches (SR Latch, D Latch, JK Latch)*
- *Flip-flops (JK flip-flop, T flip-flop, D flip-flop)*
- *Counters*
- *Asynchronous counter design*
- *Synchronous counter design*
- *Registers*
- *Various types of registers*
- *Application of shift register*

Sequential Circuits

In sequential circuits the output depends on the input as well as on the previous history of output, i.e., they contain memory elements.

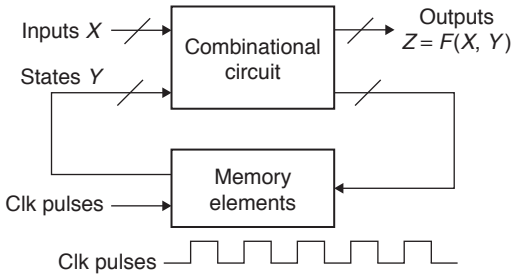


Figure 1 Block diagram of sequential circuit

Table 1 *Comparison between combinational and sequential circuits*

Combinational Circuits	Sequential Circuits
1. Output at any time depends on the combine set of input applied to it simultaneously at that instant of time	Output depends on the present input as well as on the previous history of output
2. Contains no memory element	Contains at least one memory element
3. Easy to design due to absence of memory	Difficult to design
4. Totally described by the set of output values	Its performance is totally described by the set of subsequent values as well as set of output values
5. Faster in speed because all inputs are primary inputs and applied simultaneously	Slower in speed because secondary inputs are also needed which are applied after delay
6. It need more hardware for realization	Less hardware required
7. Expensive in cost	Cheap in cost

Sequential circuits are of two types:

1. Clocked or synchronous
2. Unclocked or asynchronous

In synchronous sequential circuits the logic circuits action is allowed to occur in synchronization with the input clock pulse from a system clock.

In asynchronous sequential circuits the logic sequential action is allowed to occur at any time.

Basic Storage Elements

Latches and flip-flops

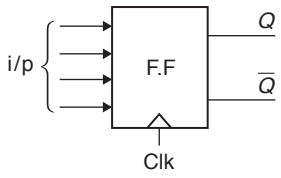
A storage element in digital circuit can maintain a binary state indefinitely until directed by an input signal to switch states. Storage elements that operate with signal levels (i.e., level triggering of signal inputs) are referred to as latches. Those controlled by a clock transition (i.e., edge triggering) are flip-flops.

Latches and filp-flops are related because latches are basic circuits from which all flip-flops are constructed. Latches are useful for storing binary information and for the design of asynchronous sequential circuits. But latches are not practical for use in synchronous sequential circuits, so we use flip-flops.

Flip-flops

They are also known as bistable multivibrators. This is a basic memory element to store 1-bit of information 0 or 1 and is used in storage circuits, counters, shift register, and many other computer applications. It has two stable states: 1 and 0. The high state is called set state and zero as reset.

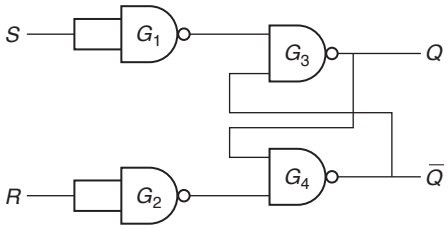
It has two outputs one being the complement of the other usually designated by Q and $\overline{Q}$.



There are different types of flip-flops S-R flip-flop, D-flip-flop, T-flip-flop, J-K flip–flops, etc.

Latches

(i) S-R Latch: The simplest latch is called S-R latch. S-R means Set-Reset. It has two outputs Q and $\overline{Q}$ and two inputs S and R, which represent set or reset signal.



Above figure shows two cross coupled gates G_3 and G_4 and inverters G_1 and G_2. Here output of G_3 is connected to the input of G_4 and output of G_4 is applied to the input of G_3. $S = 1$, $R = 0$ output of $G_1 = 0$ and $G_2 = 1$. Since one of the input of G_3 is 0, so its output will be certainly 1 and consequently both input of G_4 will be 1 and the output $\overline{Q} = 0$.

For $S = 1$, $R = 0$, $Q = 1$, $\overline{Q} = 0$. $S = 0$, $R = 1$ the output will be $Q = 0$ and $\overline{Q} = 1$. The first of the input condition $S = 1$ and $R = 0$ makes $Q = 1$ which referred as the set state and the second condition $S = 0$ and $R = 1$ makes $Q = 0$ which is referred as reset state.

For $S = 0$ and $R = 0$ output of both G_1 and G_2 will be one and hence there will be no change in Q and $\overline{Q}$.

For $S = R = 1$, both the outputs Q and $\overline{Q}$ will try to become one, which produces invalid results and should not be used for the above latch.

Input		Output		
S	R	Q	$\overline{Q}$	State
1	0	1	0	Set
0	1	0	1	Reset
0	0	0	0	No change
1	1	?	?	Invalid

(ii) SR latch by using NAND/NOR gates: The SR latch is a circuit with two cross-coupled NOR gates or two cross-coupled NAND gates. Two inputs labelled S for set and R for reset. Latch will have two outputs:

- Q: output state in normal form and
- Q': output state in complemented form.

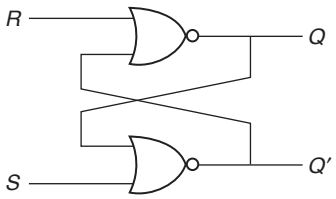


Figure 2 Logic diagram for SR latch

S	R	Q_{n+1}	Q'_{n+1}	
0	0	Q_n	Q'_n	(no change)
0	1	0	1	(Reset)
1	0	1	0	(set)
1	1	0	0	(invalid)

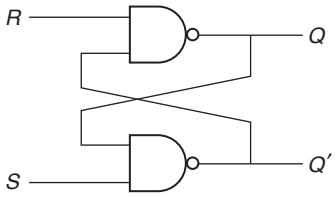
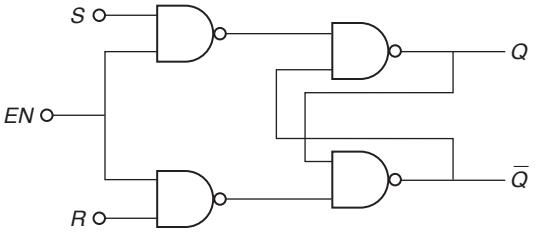


Figure 3 $\overline{S}\,\overline{R}$ Latch

S	R	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	1	1	(Invalid)
0	1	1	0	(Set)
1	0	0	1	(Reset)
1	1	Q_n	Q'_n	(No change)

$\overline{S}\,\overline{R}$ latch is active low SR latch

(iii) SR latch with control input: The working of gated SR latch is exactly the same as SR latch when the EN pulse is present. When the EN pulse is not present (EN pulse = 0) the gates G_1 and G_2 are inhibited and will not respond to the input.

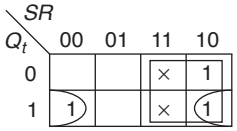


Characteristic table of SR latch shows the operation of latch in tabular form. Q_t stands as the binary state of the latch before the application of latch pulse and referred to as the present state. The S and R columns give the possible values of the inputs and Q_{t+1} is the state of the latch after the application of a single pulse, referred to as next stage. EN input is not included in the characteristic table.

Characteristic table for SR latch is given below:

Q_t	S	R	Q_{t+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	X
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X

Characteristic equation of the latch is derived from the K-map.



$$\therefore Q_{t+1} = S + \bar{R}Q_t$$

This equation specifies the value of the state as a function of the present state and the inputs.

(iv) Preset and clear inputs: For the latch/flip-flop, when the power is switched ON, the state of the circuit is uncertain. It can be either $Q = 0$ (reset) or $Q = 1$ (set) state.

In many applications it is desired to set or reset the circuit, so that initial state of the circuit will be known. This is accomplished by using the asynchronous, inputs referred to as preset (Pr) and clear (Clr), inputs.

These inputs can be applied any time, and are not synchronized with EN input/Clr input.

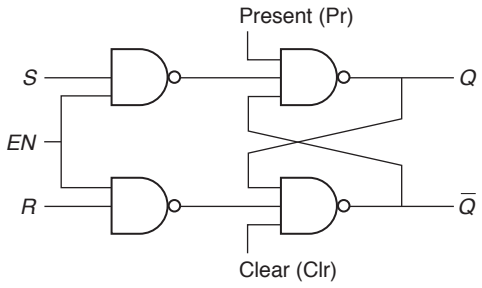


Figure 4 SR latch with Pr and Clr inputs

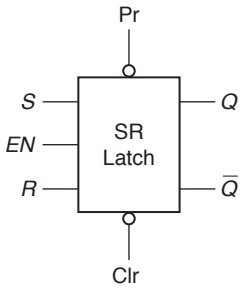
If Pr = Clr = 1, the circuits operates as of S–R latch explained previously.

If Pr = 0, Clr = 1, the output Q will become 1, which in turn changes $\bar{Q} = 0$.

If Pr = 1, Clr = 0 the output $\bar{Q}$ will become 1, which in turn changes $Q = 0$.

If Pr = Clr = 0, both Q and $\bar{Q}$ will become 1, which is invalid case, so Pr = Clr = 0 condition must not be used.

Pr	Clr	Q_{n+1}
1	1	Q – No change
0	1	1 – Set
1	0	0 – Reset
0	0	X – Invalid



(v) D latch (Transparent latch): One way to eliminate the invalid condition of SR latch (when $S = R = 1$) is to ensure that inputs S and R are never equal to 1 at the same time.

By connecting a NOT gate between S and R inputs. i.e, complement of S will be given to R, we can form D latch as shown in block diagram.

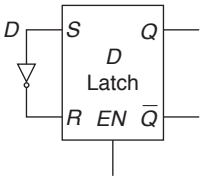


Figure 5 Block diagram for D latch

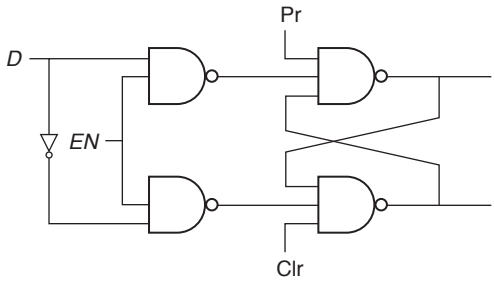


Figure 6 Logic diagram for D latch

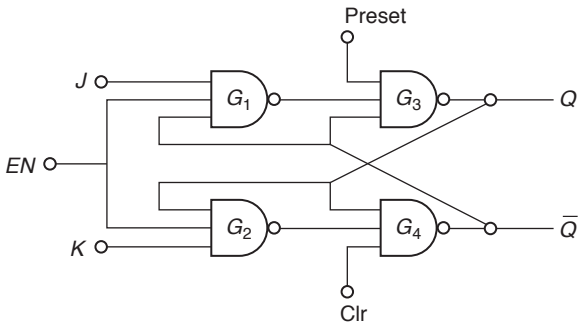
EN	D	Q_{n+1}
0	X	Q_n – No change (Disabled)
1	0	0 – Reset state
1	1	1 – Set state

When EN = 0, the circuit will be disabled and input D will not have only effect on output, and output will be same as previous state.

When EN = 1, $D = 0$, i.e., $S = 0$, $R = 1$ which makes output $Q = 0$ and $\bar{Q} = 1$ (Reset state).

When EN = 1, $D = 1$, i.e., $S = 1$, $R = 0$ which makes output $Q = 1$, and $\bar{Q} = 0$ (Set state).

(vi) JK latch: The function of JK latch is identical to that of SR latch except that it has no invalid state as that of SR latch where $S = R = 1$. In this case the state of the output is changed as complement of previous state.



EN	J	K	Q_{t+}	
1	0	0	Q_t	No change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	$\bar{Q}_t$	Toggle
0	x	x	Q_t	No change

JK latch by using SR latch: The uncertainty of SR flip-flop (when $S = 1$, $R = 1$) can be eliminated by converting it into JK latch.

The data inputs J and K, which are ANDed with $\bar{Q}$ and Q respectively, to obtain S and R inputs.

$$S = J \cdot \bar{Q}, \quad R = K \cdot Q$$

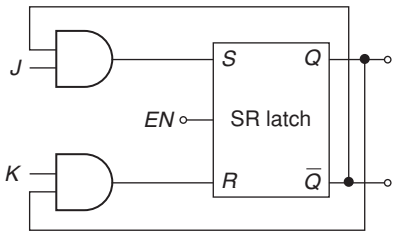
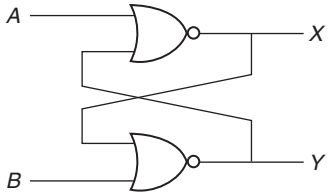


Figure 7 JK latch by using SR latch

J	K	S	R	Q_{n+1}	$\bar{Q}_{n+1}$
0	0	0	0	Q_n	$\bar{Q}_n$-No change
0	1	0	Q_n	0	1-Reset
1	0	$\bar{Q}_n$	0	1	0-Set
1	1	$\bar{Q}_n$	Q_n	$\bar{Q}_n$	Q_n-Toggle

Example 1: The following binary values were applied to A and B inputs of NOR gate latch shown in the figure, in the sequence indicated below. $A = 1$, $B = 0$; $A = 1$, $B = 1$; $A = 0$, $B = 0$. The corresponding stable X, Y outputs will be



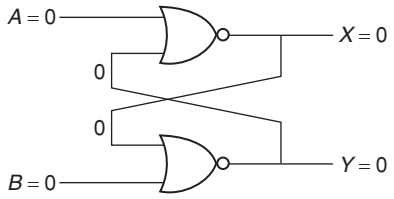
(A) 10, 01, 10 or 01
(B) 11, 00, 10
(C) 01, 00, 10 or 01
(D) 10, 11, 10 or 01

Solution: (C)
Given circuit is RS latch with NOR gates.
By comparing with RS latch $A = R$, $B = S$, and $X = Q$, $Y = \bar{Q}$, so from truth table of RS latch

S/B	R/A	Q/X	$\bar{Q}$/Y	
0	1	0	1	Reset (Invalid)
1	1	0	0	
0	0	1	0	(Same as previous state)
		0	1	

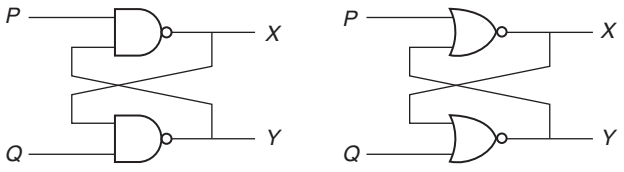
After invalid case $S = 1$, $R = 1$, i.e., $A = B = 1$,
The output $Q = 0$, $\bar{Q} = 0$, i.e., $X = Y = 0$
By applying $A = 0$, $B = 0$
The output X becomes $\overline{(0+0)} = 1$ and which in turn changes

$$Y = \overline{(B + X)} = \overline{(0+1)} = 0$$



(or) the output Y becomes $\overline{(0+0)} = 1$ and which in turn changes $X = \overline{(Y + A)} = \overline{(0+1)} = 0$. So, output (X, Y) cannot be predicted after the invalid condition. So, $X = 0$, $Y = 1$ or $X = 1$, $Y = 0$

Example 2: Refer to the NAND and NOR latches shown in the figure the inputs (P, Q) for both the latches are first made (1, 0) and then after a few seconds, made (0, 0). The corresponding stable outputs (X, Y) are



(A) NAND: first (0, 1) then (0, 1); NOR: first (1, 0) then (1, 0)
(B) NAND: first (0, 1) then (1, 1); NOR: First (0, 1) then (0, 1)
(C) NAND: first (1, 0) then (0, 0); NOR: first (1, 0) then (1, 0)
(D) NAND: first (1, 0), then (1, 0); NOR: first (1, 0) then (1, 1)

Solution: (B)
From the truth table of SR latch and $\bar{S}$ $\bar{R}$ latch SR latch with NOR gates:
For $(P, Q) = (1, 0) = (R, S)$ output $(X, Y) = (Q, \bar{Q}) = (0, 1)$

Then (P, Q) are made (0, 0), i.e., (R, S) = (0, 0), which results in no change at output. So, $(X, Y) = (Q, \bar{Q}) = (0,1)$

SR latch with NAND gates:

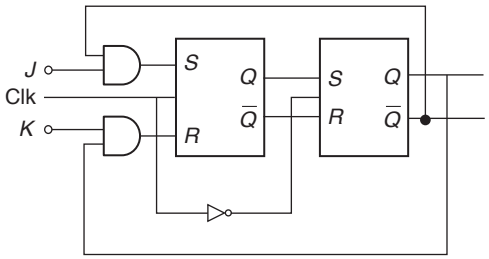
For $(P, Q) = (1, 0) = (S, R)$ output $(X, Y) = (Q, \bar{Q}) = (0,1)$. Then (P, Q) are made (0, 0), i.e., $(S, R) = (0, 0)$ which is invalid conditions for $\bar{S}\,\bar{R}$ latch. So, $(X, Y) = (Q, \bar{Q}) = (1,1)$

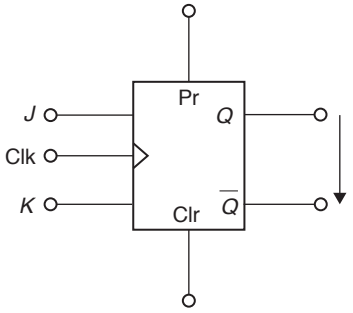
(vii) Race around condition: The difficulties of both the inputs ($S = R = 1$) being not allowed in an SR latch is eliminated in JK latch by using the feedback connection from the output to the input of the gate G_1 and G_2. In a normal JK latch if $J = K = 1$ and $Q = 0$ and enable signal is applied without RC differentiator, after a time interval Δt (the propagation delay through two NAND gate in series) the output will change to $Q = 1$. Now we have $J = K = 1$ and $Q = 1$ and after another time interval of Δt the output will change back to $Q = 0$. Hence for the duration of (t_p) of the enable signal the output will oscillates back and forth between 0 and 1. At the end of the enable signal the values of Q is uncertain. This situation is referred to as race around condition.

The race around condition can be avoided if enable time period $t_p < \Delta t$ but it may be difficult to satisfy this condition, because of very small propagation delays in ICs. To solve this problem the enable signals are converted to narrows spike using RC differentiator circuit having a short time constant. Its output will be high during the high transmission time of the enable. Another method to avoid this problem is master-slave JK flip-flop.

FLIP-FLOPS

(i) Master-slave JK flip-flop: This is a cascade of 2 SR latches with feedback from the output of the second SR latch to the inputs of the first as shown in the figure below.





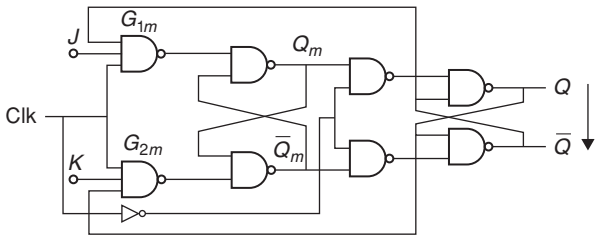


Figure 8 Logic diagram of JK flip-flop

Positive clock pulse is applied to the first latch and the clock pulse will be inverted before its arrival at the second latch. When Clk = 1, the first latch is enabled and the outputs Q_m and $\bar{Q}_m$ responds to the inputs J and K, according to the truth table of JK latch. At this time the 2nd latch is inhibited because its clock is low $(\overline{\text{Clk}} = 0)$. When the clock goes low (Clk = 0), the first latch is inhibited and the second is enabled. Therefore, the outputs Q and $\bar{Q}$ follow the outputs Q_m and $\bar{Q}_m$, respectively. Since the second latch simply follows the first one, it is referred to as slave and the first one as the master. Hence this configuration is known as master-slave JK flip-flop. In this circuit, the input to the gate G_{1m} and G_{2m} do not change, during the clock pulse levels.

The race around condition does not exist.

Table 2 *State/characteristic Table*

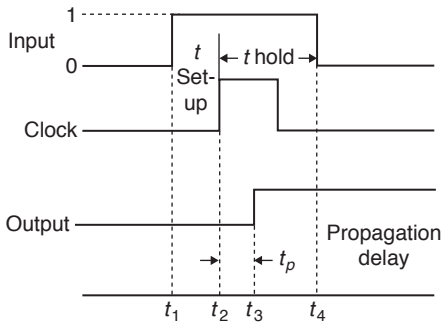
Clk	J	K	Q_t	Q_{t+1}	
↓	0	0	0	0	Q_1
↓	0	0	1	1	
↓	1	0	0	1	1
↓	1	0	1	1	
↓	0	1	0	0	0
↓	0	1	1	0	
↓	1	1	0	1	$\bar{Q}_t$
↓	1	1	1	0	

Q \ JK	00	01	11	10
0			1	1
1	1			1

$$Q_{t+1} = J\bar{Q}_t + Q_t\bar{K}$$

(ii) Flip-flop switching time: In designing circuits with flip-flop the following parameters are important:

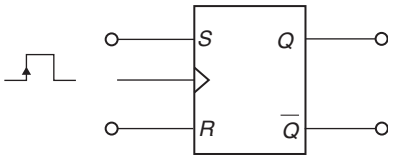
1. **Set-up time:** The minimum amount of time required for the data input to be present before the clock arrived.
2. **Hold time:** The minimum amount of time that the data input to be present after the clock trigger arrived.
3. **Propagation delay:** The amount of time it takes for the output to change states after an input trigger.



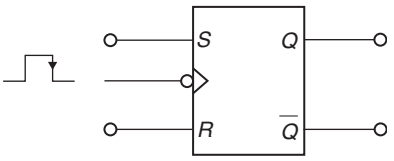
For example, t setup = 50 m sec and t hold = 5 m sec, the data bit has to be the input at least 50 m sec before the clock bit arrives and hold at least 5 m sec after the clock edge.

(iii) Triggering of flip-flop: The flip-flop can be triggered to set or reset either at one of the edges of the clock pulse. There are three types of triggering as described below:

1. **Positive edge triggering flip-flop:** These set or reset at the positive (rising or leading) edge of the clock pulse depending upon the state of i/p signal and o/p remain steady for 1 clock period. Positive edge triggering is indicated by an arrow head at the clock terminal of the flip-flop.



2. **Negative edge triggered flip-flop:** There are flip-flops those in which state transmissions take place only at the negative edge (falling or trailing) of the clock signal. Negative edge triggering is indicated by arrow head with bubble at the clock terminal.



3. **Level triggering:** Level triggering means the specified action occurs based on the steady state value of the input. That is, when a certain level is reached (0 or 1) the output will change states level triggering will be used in latches.

(iv) *D* flip-flop: It receives the designation from its ability to hold data into its internal storage. An SR/JK flip-flop has two inputs. It requires two inputs S/J and R/K to store 1 bit. This is a serious disadvantage in many application to overcome the difficulty D flip-flop has been developed which has only one input line. A D flip-flop can be realized using a SR/JK as show in the figure below.

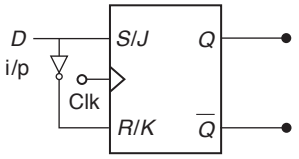


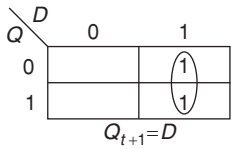
Table 3 *Truth Table*

clk	D	Q_{t+1}
↑	X	Q_t
↑	0	0
↑	1	1

There is no raising problem with D flip-flop. High or 1 state will set the flip-flop and a low or 0 state will reset the flip-flop. The presence of inverter at the input ensure that S/J and R/K inputs will always be in the opposite state.

Table 4 *Characteristic Table of D Flip-flop*

Q_t	D	Q_{t+1}
0	0	0
0	1	1
1	0	0
1	1	1



From the characteristic table of D.flip-flop, the next state of the flip-flop is independent of the present state since $Q_{t+1} = D$, whether $Q_t = 0$ or 1.

(v) *T* flip-flop: In a JK flip-flop $J = K = 1$ and the resulting flip-flop is referred to as a T flip-flop.

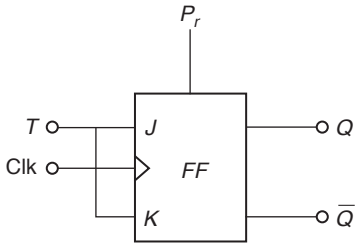


Table 5 *Truth Table*

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

If $T = 1$, it acts as a toggle switch for every Clk pulse with high input, the Q changes to its opposite state.

Table 6 *Characteristic Table*

Q_t	T	Q_{t+1}
0	0	0
0	1	1
1	0	1
1	1	0
1	0	1

Q \ T	0	1
0		1
1	1	

$$Q_{t+1} = T\overline{Q}_t + Q_t\overline{T}$$

(vi) Excitation table of flip-flops: The truth table of flip-flop is also referred to as the characteristic table, which specifies the operational characteristic of flip-flop.

Sometimes we come across situations in which present state and the next state of the circuit are known and we have to find the input conditions that must prevail to cause the desired transition of the state.

Consider initially JK flip-flop output $Q_n = 1, \overline{Q}_n = 0$, after clock pulse it changed to $Q_{n+1} = 0, \overline{Q}_{n+1} = 1$,

The input conditions, which made this transition, can be

Toggle – for $J = 1, K = 1, Q_{n+1} = \overline{Q}_n$

or

Reset – for $J = 0, K = 1, Q_{n+1} = 0, \overline{Q}_{n+1} = 1$

From the above conditions we can conclude that for transition $Q_n = 1$ to $Q_{n+1} = 0$ occurs when $J = 0$ (or) 1 (don't care) and $K = 1$.

Similarly, input conditions can be found out for all possible situations.

Table 7 *Excitation table of flip-flop.*

Present State	Next State	SR Flip-flop		JK Flip-flop		T Flip-flop	D Flip-flop
Q_n	Q_{n+1}	S	R	J	K	T	D
0	0	0	×	0	×	0	0
0	1	1	0	1	×	1	1
1	0	0	1	×	1	1	0
1	1	×	0	×	0	0	1

These excitation tables are useful in the design of synchronous circuits.

(vii) State diagrams of flip-flops: State diagram is a directed graph with nodes connected with directed arcs. State of the circuit is represented by the node, the directed arcs represent the state transitions, from present state (node) to next state (node) at the occurrence of clock pulse.

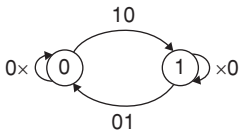


Figure 9 State diagram of SR flip-flop

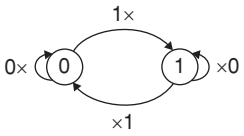


Figure 10 State diagram of JK flip-flop

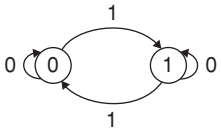


Figure 11 State diagram of *T* flip-flop

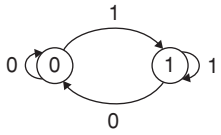


Figure 12 State diagram of *D* flip-flop

(viii) Conversion of one flip-flop to other flip-flop

Conversion of *T* flip-flop to JK flip-flop

1. Write the characteristic table of required flip-flop (here JK).
2. Write the excitation table of available or given Flip-flop (here *T*).
3. Solve for inputs of given flip-flop in terms of required flip-flop inputs and output.

Table 8 *JK flip-flop characteristic and T flip-flop excitation table*

JK Flip-flop Characteristic Table		T Flip-flop Excitation Table		
J	K	Q_n	Q_{n+1}	T
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	1

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

$$T = J\overline{Q}_n + KQ_n$$

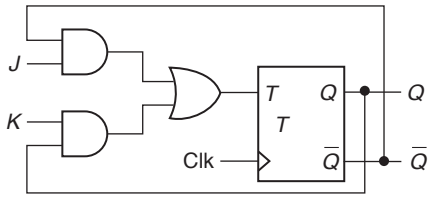


Figure 13 *D* Flip-flop by using other flip-flops

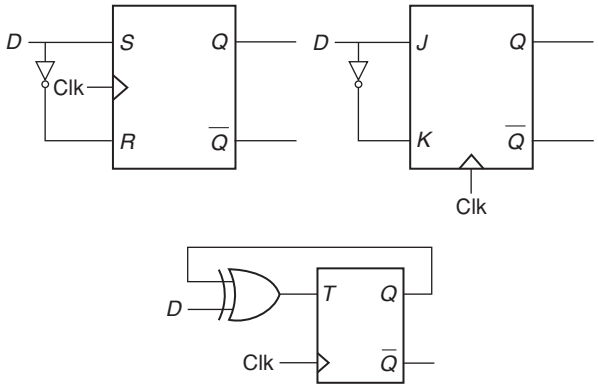
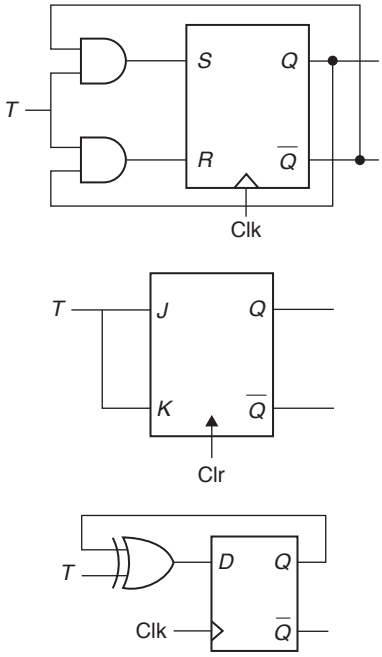
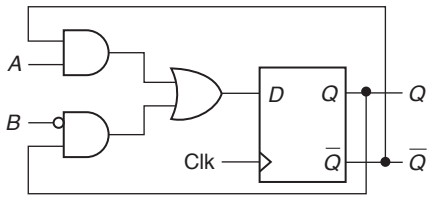


Figure 14 *T* flip-flop by using other flip-flops



Example 3: A sequential circuit using *D* flip-flop and logic gates is shown in the figure, where *A* and *B* are inputs and *Q* is output.



The circuit is

(A) SR flip-flop with inputs $A = S, B = R$

(B) SR flip-flop with inputs $A = \overline{R}, B = S$

(C) JK flip-flop with inputs $A = J, B = K$

(D) JK flip-flop with inputs $A = K, B = \overline{J}$

Solution: (C)

The characteristic equation of *D* flip-flop is

$$Q_{n+1} = D$$

Here input $D = A\overline{Q}_n + \overline{B}Q_n$

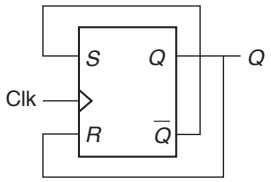
So, output $Q_{n+1} = A\overline{Q}_n + \overline{B}Q_n$

By comparing this equation with characteristic equation of JK

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

If $A = J, B = K$, then this circuit works like JK flip-flop.

Example 4: The input Clk frequency for the flip-flop given is 10 kHz, then the frequency of *Q* will be



(A) 10 kHz (B) 5 kHz

(C) 20 kHz (D) 2.5 kHz

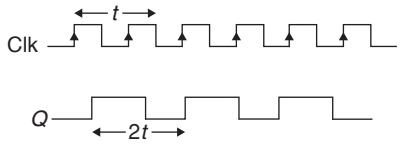
Solution: (B)

Form circuit we can say $S = \overline{Q}_n, R = Q_n$.

If initially $(Q_n, \overline{Q}_n) = (0, 1)$, then inputs $(S, R) = (1, 0)$, by applying clk pulse $(Q_{n+1}\overline{Q}_{n+1})$ becomes $(1, 0)\ldots$

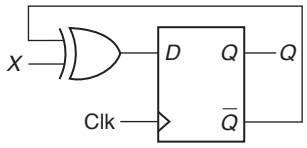
Clk	Q_n	$\overline{Q}_n$	S	R	Q_{n+1}	$\overline{Q}_{n+1}$
1	0	1	1	0	1	0
2	1	0	0	1	0	1
3	0	1	1	0	1	0
4	1	0	0	1	0	1

The output Q_{n+1} toggles for every clock pulse.



So frequency of $Q = \frac{1}{2t} = \frac{f}{2} = \frac{10}{2} = 5$ kHz

Examples 5: For the D flip-flop shown, if initially Q_n is set then what is the output state Q_{n+1} for $X = 0$, and for $X = 1$?



(A) 0, 0 (B) 0, 1
(C) 1, 0 (D) 1, 1

Solution: (B)
The characteristic equation of D is $Q_{n+1} = D$
Here $D = X \oplus \overline{Q_n}$
So $Q_{n+1} = X \oplus \overline{Q_n}$
We have $Q_n = 1$ (Q_n is set) for $X = 0$

$$Q_{n+1} = 0 \oplus 0 = 0$$

We have $Q_n = 1$ (Q_n is set), for $X = 1$

$$Q_{n+1} = 1 \oplus 0 = 1$$

Applications of flip-flops:

1. Data storage: A group of flip-flops connected in series/parallel is called a register, to store a data of N-bits, N-flip-flops are required. Data can be stored in parallel or serial order. Similarly, serial to parallel conversion and parallel to serial conversion can be done by using registers.
2. Counting: A number of flip-flops can be connected in a particular fashion to count the pulses applied (Clk) electronically. One flip-flop can count 2 Clk pulses, two flip-flops can count up to $2^2 = 4$ pulses, similarly n flip-flops can count up to 2^n pulses. Flip-flops may be used to count up/down.
3. Frequency division: Flip-flops may be used to divide input signal frequency by any number. A single flip-flop may be used to divide the input frequency by 2. Similarly n flip-flops may be used to divide the input frequency by 2^n. Output of a MOD-n counter (i.e., which counts n states) will divide input frequency by n.

Counters

Digital counters consist of a number of flip-flops. Their function is to count the number of clock pulses arriving at its clock input.

(i) Counter classification: Counters are classified according to their operational characteristic. Some of these characteristics include:

1. Counter triggering techniques
2. Frequency division characteristic
3. Counter modulus
4. Asynchronous or synchronous

In a synchronous counter all flip-flops are clocked simultaneously. In asynchronous counter the flip-flops are not clocked simultaneously. Each flip-flop is triggered by the previous flip-flop.

(ii) Asynchronous counters (ripple counters): Asynchronous counters do not have a common clock that controls all the flip-flop stages. The control clock is input to the first stage. The clock for each stage subsequent is obtained from the flip-flop of the prior stages. Let us analyze the 3-bit counter and its corresponding wave form diagram shown below.

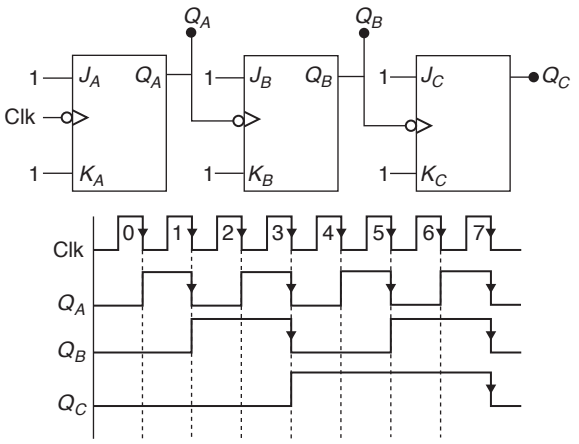


Figure 15 Timing diagrams

- The counter has three flip-flops and three output bits, therefore it is a three stage counter.
- The input clock does not trigger the three flip-flops, therefore it is an asynchronous counter.
- The J and K inputs are tied together as kept high. So they are considered to be toggle flip-flops.
- The flip-flops are negative edge triggered.
- The wave form analysis reveals that Q_A is the LSB and that its frequency is $\frac{1}{2}$ the input clock frequency. Further more, Q_c is the MSB and its frequency is $\frac{1}{8}$ the input clock frequency.
- The count sequence is 000, 001, 010, 011, 100, 101, 110, 111 where the LSB is Q_A. Thus it is MOD-8 binary up counter.
- Asynchronous counters are also known as ripple counters because the effect of the input clock ripples through the counter until it reaches the final stage.

Asynchronous Counter Design

Step I: Write the counting sequence.
Step II: Tabulate the values of reset signals. R for various state of counter.
Step III: Obtain the minimal expression for R and $\overline{R}$ using K-map or any other method.
Step IV: Provide a feedback such that R or $\overline{R}$ resets all the flip-flops after the desired count.

Table 9 *Identification of up/down Counters*

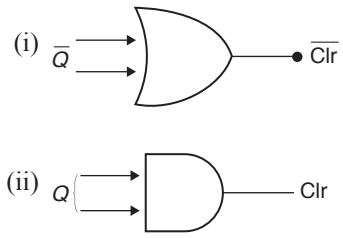
Clock Triggering	Q	Type
+ve edge	$\overline{Q}$	Up
+ve edge	Q	Down
–ve edge	$\overline{Q}$	Down
–ve edge	Q	Up

Clock is negative triggering pulse and Q is connected to next level clock, it is acting like a up counter.

Table 10 *Identification of GATE to Clear the Flip-flops*

Input to the Gate	Output of the Gate	Type of Gate
$\overline{Q}$	$\overline{Clr}$	OR
$\overline{Q}$	Clr	NOR
Q	$\overline{Clr}$	NAND
Q	Clr	AND

Example:

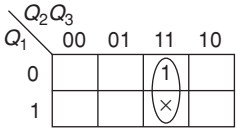


Example 6: Design and Implement a MOD-6 asynchronous counter using T flip-flops.

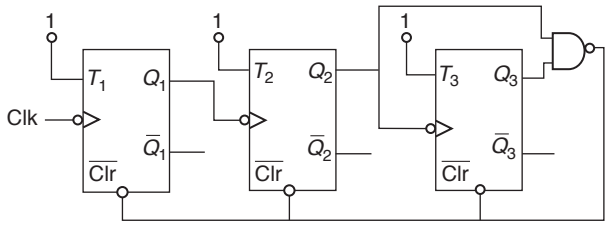
Solution: Counting sequence is 00, 001, 010, 011, 100, 101

After Pulses	States Q_3, Q_2, Q_1	Reset R
0	000	0
1	001	0
2	010	0
3	011	0
4	100	0
5	101	0
6	110	1
7	↓↓↓	
	000	0
	111	X

From the Truth table $R = Q_3 Q_2$
For active Low $\overline{R}$ is used.
$\therefore$ $R = 0$ for 000 to 101
$R = 1$ for 110
$R = X$ for 111
$\therefore$ K-map is



$\therefore$ $R = Q_2 Q_3$
Logic diagram is



Asynchronous Decode Counter

A ripple counter is an asynchronous sequential circuit, clock is applying only for LSB side. Decade ripple counter it counts from 0 to 9 for up counter.

MOD-10 counter it counts starting from 0000 to 1001. If the NAND gate output is logic '0' at that instant the counter reset to initial state.

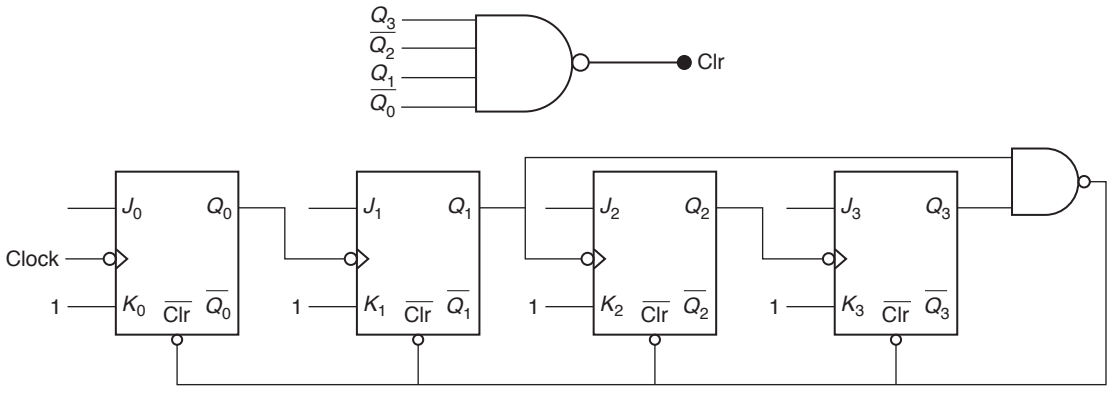


Figure 16 MOD-10 or decade counter

To design a MOD-N counter minimum number of flip-flops required is

$N \le 2^n$

where $N \rightarrow$ MOD

$n \rightarrow$ No. of flip-flops

Example:

MOD-5 counter

$5 \le 2^n$

$\therefore n = 3$

Operating Clock Frequency

(i) Synchronous counter:

$$f_{Clk} \le \frac{1}{t_{pd}}$$

(ii) Asynchronous counter:

$$f_{Clk} \le \frac{1}{n t_{pd}}$$

Output frequency of the MOD-N counter is

$$\Rightarrow f_o = \frac{f_{Clk}}{N}.$$

(iii) Synchronous counter: When counter is clocked such that each flip-flop in the counter is triggered at the same time, the counter is called as synchronous counter.

- Synchronous counters have the advantage of high speed and less severe decoding problems.
- Disadvantage is having more circuiting than that of asynchronous counter.

Synchronous Series Carry Counters

For normal ring counters to count N sequence total N flip-flops are required.

Unused states in ring counter = $2^N - N$.

Unused states in Johnson ring counter = $2^N - 2N$.

Asynchronous counters are slower than the synchronous counters. By using synchronous series carry adders we can design MOD-N counter with n Flip-flops-only.

For non-binary counters $N \le 2^n$

3-bit series carry up counter

It counts from initial state 000 to 111.

$\therefore$ MOD = 2^n = 8 states

$\therefore$ MOD-8

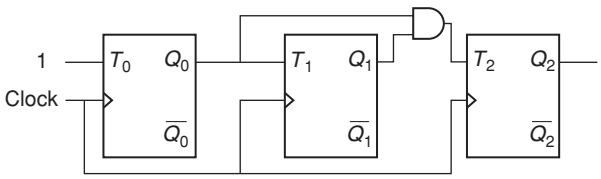


Figure 17 3-bit series carry counter

$$f_{Clk} \le \frac{1}{t_{pd} + (n-2)t_{pd\ AND}}$$

where

$t_{pd} \rightarrow$ Propagation delay of each flip-flop.

$t_{pd\ AND} \rightarrow$ Propagation delay of AND gate.

$n \rightarrow$ Number of flip-flops.

In this, Q_o toggles for every clock pulse.

Q_1 toggles when Q_o is 1.

Q_2 toggles when o/p of AND gate is logic 1.

Note: To design a synchronous series carry down counter. Connect $\overline{Q_o}$ to the next flip-flop input.

Design of Synchronous Counter

Step I: Determine the required number of flip-flop.

Step II: Draw the state diagram showing all possible states.

Step III: Select the type of flip-flop to be used and write the excitation table.

Step IV: Obtain the minimal expressions for the excitations of the FFs using the K-maps.

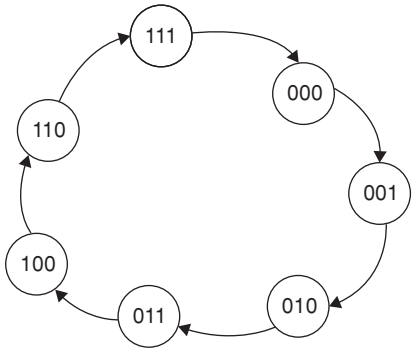
Step V: Draw a logic diagram based on the minimal expression. Let us employ these techniques to design a MOD-8 counter to count in the following.

Example 7: Sequence: 0, 1, 2, 3, 4, 5, 6, and 7. Design a synchronous counter by using JK flip-flops.

Solution:

Step I: Determine the required number of flip-flops. The sequence shows a 3-bit up counter that requires 3 flip-flops.

Step II: Draw the state diagram.

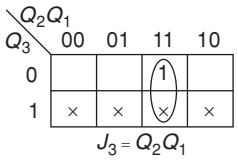


Step III: Select the type of flip-flop to be used and write the excitation table.

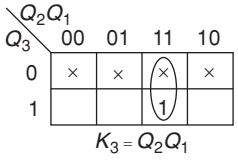
JK flip-flop is selected and excitation table of a 3-bit up counter is

PS	*NS*	Required Excitation
$Q_3\ Q_2\ Q_1$	$Q_3\ Q_2\ Q_1$	$J_3\ K_3\ J_2\ K_2\ J_1\ K_1$
0 0 0	0 0 1	0 x 0 x 1 x
0 0 1	0 1 0	0 x 1 x x 1
0 1 0	0 1 1	0 x x 0 1 x
0 1 1	1 0 0	1 x x 1 x 1
1 0 0	1 0 1	x 0 0 x 1 x
1 0 1	1 1 0	x 0 1 x x 1
1 1 0	1 1 1	x 0 x 0 1 x
1 1 1	0 0 0	x 1 x 1 x 1

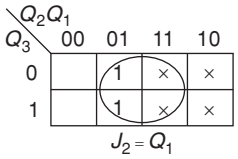
Step IV: Obtain the minimal expression using K-map.



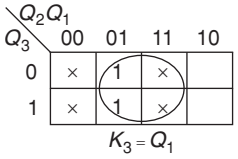
$J_3 = Q_2Q_1$



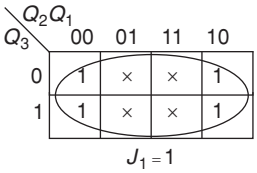
$K_3 = Q_2Q_1$



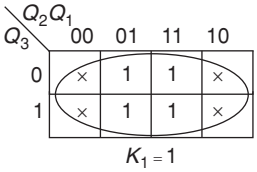
$J_2 = Q_1$



$K_3 = Q_1$



$J_1 = 1$



$K_1 = 1$

Step V: Draw the logic diagram based on the minimal expression.

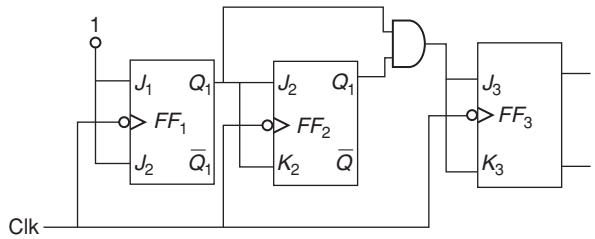


Table 11 *Comparison between asynchronous counter and synchronous counter*

Asynchronous Counter	Synchronous Counter
1. In this type of counter, flip-flops are connected in such a way that output of first flip-flop drives the clock for the next flip-flop	In this type there is no connection between output of first flip-flop and clock input of the next flip-flop
2. All the flip-flops are not clocked simultaneously	All the flip-flops are clocked simultaneously
3. Logic circuit is very simple even for more number of states	Design involves complex logic circuits as number of state increases
4. Main draw back of these counters is their low speed as the clock is propagated through number of flip-flops before it reaches last flip-flop	As clock is simultaneously given to all flip-flops, there is no problem of propagation delay. Hence they are preferred when number of flip-flops increases in the given design.

The main drawback of ripple counters is their high delays, if propagation delay of each flip-flop is assumed as x, then to get output of the first flip-flop it takes x, i.e., after x seconds the second flip-flop will get its clock pulse from previous stage, and output of second flip-flop will be out after another x seconds, similarly the final output of last flip-flop will be after nx seconds, where n is the number of flip-flops. So the propagation delay of ripple counter is nx, which is directly proportionate to the number of flip-flops.

The maximum frequency of operation of ripple counter is inverse of delay, $f_{max} = \frac{1}{nx}$

Maximum operating frequency is the highest frequency at which a sequential circuit can be reliably triggered. If the clock frequency is above this maximum frequency the flip-flops in the circuit cannot respond quickly and the operation will be unreliable.

In case of synchronous counters (synchronous circuits) as clock is applied simultaneously to all the flip-flops, the output of all the flip-flops change by x seconds (delay of one flip-flop) and this delay is independent of number of flip-flops used in circuit.

The maximum frequency of operation of synchronous counter is inverse of delay $f_{max} = \frac{1}{x}$

Example 8: The maximum operation frequency of a MOD-64 ripple counter is 33.33 kHz, the same flip-flops are used to design a MOD-32 synchronous counter, and then the maximum operating frequency of the new counter is

(A) 400 kHz (B) 200 kHz
(C) 40 kHz (D) 500 kHz

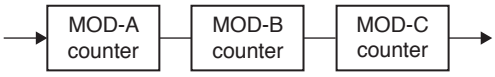
Solution: For ripple counter $f_{max} = \frac{1}{nx}$, given is a MOD-64 ripple counter, i.e., 2^6 states, so $n = 6$ flip-flops are required.

$$x = \frac{1}{33.33K \times 6} = 5\mu S$$

For synchronous counter

$$f_{max} = \frac{1}{x} = \frac{1}{5\ \mu S} = 0.2\ \text{MHz} = 200\ \text{kHz}$$

When multiple counters are connected in cascade, then the total number of states of the new counter is $A \times B \times C$, i.e., it will work as MOD-$A \times B \times C$ counter.



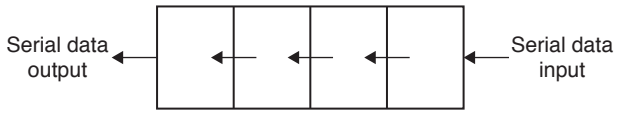
For example, decade counter counts from 0 to 9, 10 states – If two such decade counters are connected in cascade, then the total counting states will be 10 × 10 = 100, it will work as MOD-100 counter, which counts from 00 to 99.

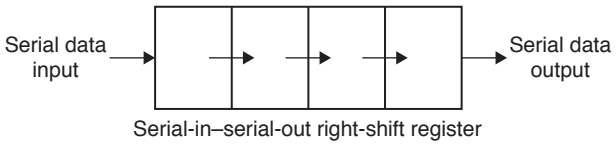
Registers

A number of flip-flops connected together such that data may be shifted into and shifted out of them is called a shift register. There are four basic types of shift register:

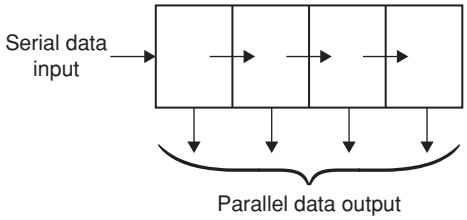
1. Serial-in–serial-out
2. Serial-in–parallel-out
3. Parallel-in–serial-out
4. Parallel-in–serial-out

(i) Serial-in–serial-out:

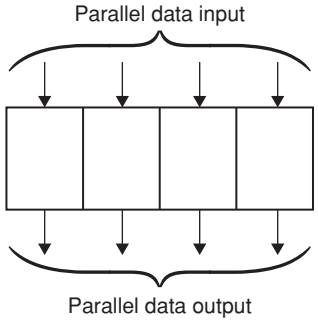




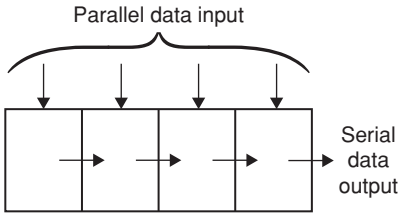
(ii) Serial-in–parallel-out:



(iii) Parallel-in–parallel-out:



(iv) Parallel-in–serial-out:



Serial input and serial output register: This type of shift register accepts data serially, i.e., one bit at a time and also outputs data serially. The logic diagram of 4-bit serial input, serial output, shift-right, shift register is shown in the following figure. With four D flip-flops the register can store up to four bits of data.

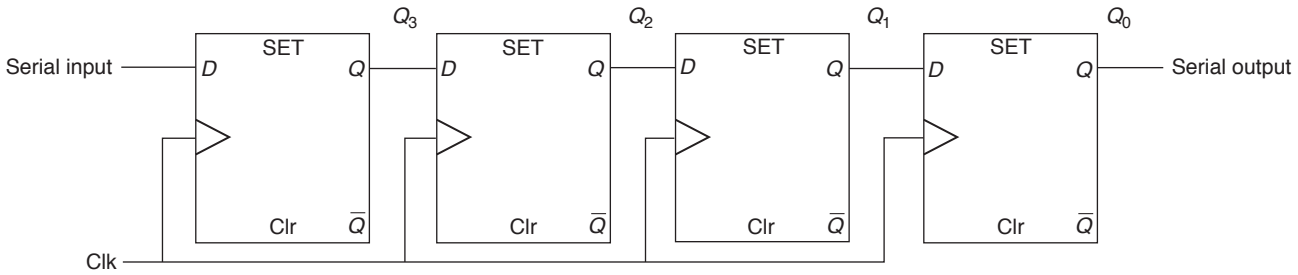


Figure 18 Serial input and serial output register

If initially, all flip-flops are reset, then by applying serial input 1101, the flip-flop states will change as shown in below table.

Clk	S.I	Q_3	Q_2	Q_1	Q_0
0	1	0	0	0	0
1	0	1	0	0	0
2	1	0	1	0	0
3	1	1	0	1	0
4		1	1	0	1

The first data bit 1 will appear at serial output after 4 clock pulses.

Application of Shift Registers

1. Delay line: Serial input and serial output shift register can be used to introduce delay in digital signals.

   $$\text{Delay} = \text{no.of flip-flops} \times \frac{1}{\text{Clk frequency}} = \text{No. of flip-flops} \times \text{time period of clock pulse}$$

2. Serial to parallel, parallel to serial converter: SIPO, PISO registers used for data conversion.
3. Sequence generator: A circuit, which generates a prescribed sequence of bits, with clock pulses is called as sequence generator
   The minimum number of flip-flops 'n' required to generate a sequence of length 'S' bits is given by $S \le 2^n - 1$

Shift register counters

One of the applications of the shift register is that they can be arranged to work as ring counters. Ring counters are constructed by modifying the serial-in, serial-out, shift registers. There are two types of ring counters—basic ring counter and twisted ring counter (Johnson counter). The basic ring counter is obtained from SISO shift register by connecting serial output to serial input.

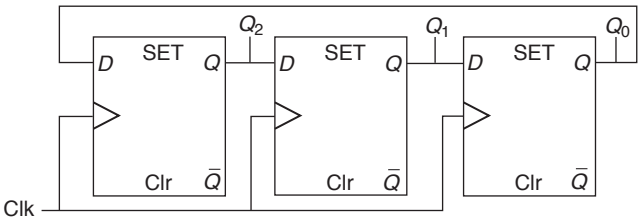
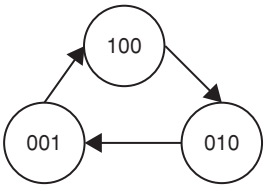


Figure 19 Ring counter

In most instances, only a single 1 or single 0 is in the register and is made to circulate around the register as long as the clock pulses are applied. Consider initially first flip-flop is set, and others are reset. After 3 clock pulses, again we will get initial state of 100. So this is a MOD-3 counter.

Clk	Q_2	Q_1	Q_0
0	1	0	0
1	0	1	0
2	0	0	1
3	1	0	0
4	0	1	0



A ring counter with N flip-flops can count up to N states, i.e., MOD-N counter, whereas, N-bit asynchronous counter can count up to 2^N states. So, ring counter is uneconomical compared to a ripple counter, but has the advantage of requiring no decoder. Since it is entirely synchronous operation and requires no gates for flip-flop inputs, it has further advantage of being very fast.

Twisted ring counter (Johnson counter): This counter is obtained from a SISO shift register by connecting the complement of serial output to serial input as shown in below figure.

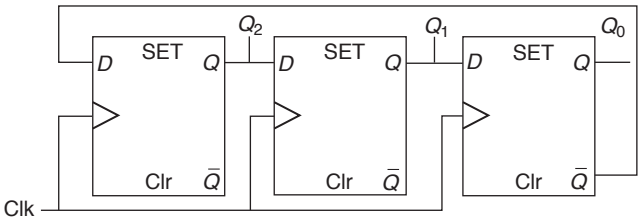
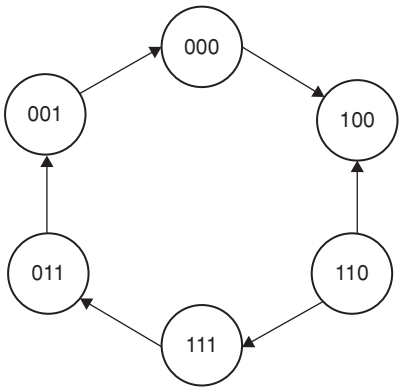


Figure 20 Twisted Ring Counter

Let initially all the FFs be reset, after each clock pulse the complement of last bit will appear as at MSB, and other bits shift right side by 1-bit. After 6 clock pulses the register will come to initial state 000. Similarly, the 3-bit Johnson counter will oscillate between the states 101, 010.

Clk	Q_2	Q_1	Q_0
0	0	0	0
1	1	0	0
2	1	1	0
3	1	1	1
4	0	1	1
5	0	0	1
6	0	0	0



An n-bit Johnson counter can have $2n$ unique states and can count up to $2n$ pulses, so it is a MOD-$2n$ counter. It is more economical than basic ring counter but less economical than ripple counter.

Solved Examples

Example 1: Assume that 4-bit counter is holding the count 0101. What will be the count after 27 clock pulses?

Solution: Total clock pulses: 27 = 16 + 11
0101 + 1011 = 0000

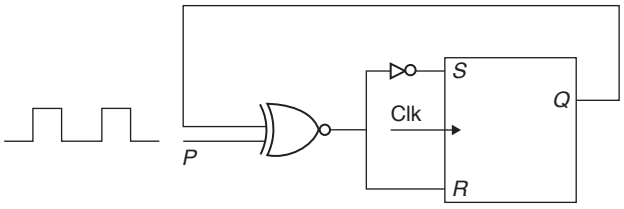
Example 2: A MOD-2 counter followed by MOD-5 counter is

Solution: A decade counter, counts 10 states (5 × 2).

Example 3: A 4-bit binary ripple counter uses flip-flops with propagation delay time of 25 msec each. The maximum possible time required for change of state will be

Solution: The maximum time = 4 × 25 ms = 100 ms

Example 4: Consider the circuit, the next state Q^+ is



Solution

P	Q	S	R	Q^+
0	0	0	1	0
0	1	1	0	1
1	0	1	0	1
1	1	0	1	0

So, $Q^+ = P \oplus Q$

Example 5: A certain JK FF has t_{pd} = 12 n sec what is the largest MOD counter, that can be constructed from these FF and still operate up to 10 MHz?

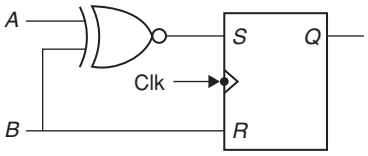
Solution: $N \le \dfrac{1}{f_{\max} \cdot t_{pd}}$

$f_{\max}$ = 10 MHz $N \le 8$
t_{pd} = 12 ns

$$N \le \frac{1}{10 \times 10^6 \times 12 \times 10^{-9}}$$

MOD counter is = $2^N = 2^8 = 256$

Example 6: An AB flip-flop is constructed from an SR flip-flop as shown below. The expression for next state Q^+ is

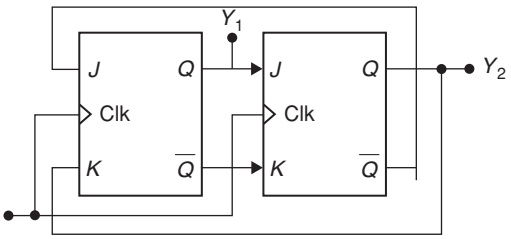


Solution:

A	B	Q	S	R	Q^+
0	0	0	1	0	1
0	0	1	1	0	1
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	0	0	0
1	0	1	0	0	1
1	1	0	1	1	×
1	1	1	1	1	×

$\therefore Q^+ = \overline{A}\overline{B} + AQ = \overline{A}\overline{B} + \overline{B}Q$

Example 7: In the circuit shown below, the output y_1 and y_2 for the given initial condition $y_1 = y_2 = 1$ and after four input pulses will be



Solution:
After 1st pulse $y_1 = 0, y_2 = 1$
After 2nd pulse $y_1 = 0, y_2 = 0$
After 3rd pulse $y_1 = 1, y_2 = 0$
After 4th pulse $y_1 = 1, y_2 = 1$

Example 8: A ripple counter is to operate at a frequency of 10 MHz. If the propagation delay time of each flip-flop in the counter is 10 ns and the storbing time is 50 ns, how many maximum stages can the counter have?

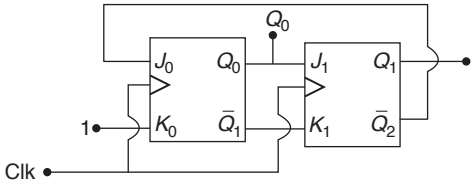
Solution: $nt_{pd} + t_s \le \frac{1}{f}$

where, n = number of stages
t_{pd} = propagation delay time
t_s = strobing time
f = frequency of operation = $10 \times 10^{-9} n + 50 \times 10^{-9}$

$$\le \frac{1}{10 \times 10^6}$$

(or) $10n + 50 \le 100$
(or) $10n \le 50$
For max stages $n = \frac{50}{10} = 5$

Example 9: In the circuit assuming initially $Q_0 = Q_1 = 0$. Then the states of Q_0 and Q_1 immediately after the 33rd pulse are

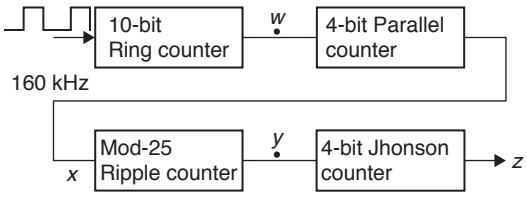


Solution:

J_0	K_0	J_1	K_1	Q_0	Q_1	Count
1	1	0	1	0	0	Initial
1	1	1	0	1	0	1st pulse
0	1	0	1	0	1	2nd
1	1	0	1	0	0	3rd
1	1	1	0	1	0	4th
0	1	0	1	0	1	5th pulse

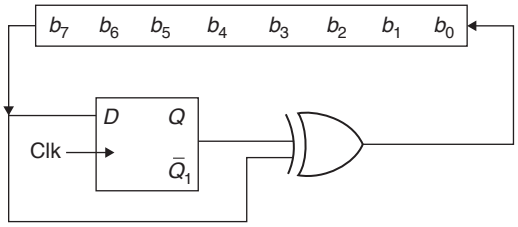
After 4th pulse, output is same as after 1st one, so, sequence gets repeated. So output after 33rd pulse would be same as after 3rd pulse. i.e., (00).

Example 10: The frequency of the pulse at z in the network shown in figure is



Solution: 10-bit ring counter is a MOD-10. So, it divides the 160 kHz input by 10. Therefore, w = 16 kHz. The 4-bit parallel counter is a MOD-16. Thus, the frequency at x = 1 kHz. The MOD-25 ripple counter produces a frequency at y = 40 Hz (1 kHz/25 = 40 Hz). The 4-bit Johnson counter is a MOD-8. The frequency at Z = 5Hz.

Example 11: The 8-bit shift left shift register, and D flip-flop shown in the figure is synchronized with the same clock. The D flip-flop is initially cleared. The circuit acts as

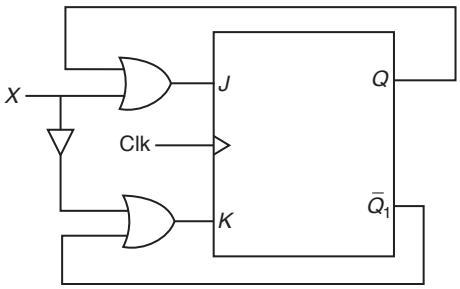


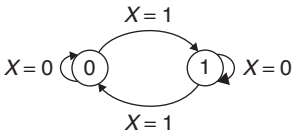
Solution: The output of XOR gate is $Z = b_{i+1} \oplus b_i$ and this output shift the register to left. Initially, $Z = 0$
After 2nd clock $Z = b_7 \oplus 0 = b_7$
After 2nd clock $Z = b_7 \oplus b_6$
3rd clock $Z = b_6 \oplus b_5$
4th clock $Z = b_5 \oplus b_4$
It is a binary to gray code converter.

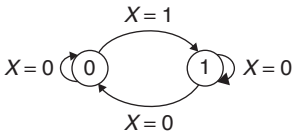
Example 12: A 4-bit MOD-16 ripple counter uses JK flip-flops. If the propagation delay of each flip-flop is 50 ns sec, the maximum clock frequency that can be used is equal to

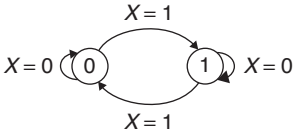
Solution: Max = clock frequency $= \frac{1}{4 \times 50 \times 10^{-9}} = 5$ MHz

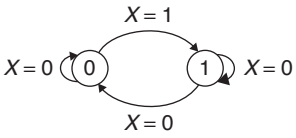
Example 13: What is the state diagram for the sequential circuit shown?



(A)


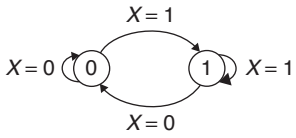
(B)


(C)


(D)


Solution: (D)
State diagram of a sequential circuit will have states (output) of all the flip-flops.

Present state	Next state	Q_{n+1}
Q_n	For $x = 0$	For $x = 1$
0	0	1
1	0	1



EXERCISES

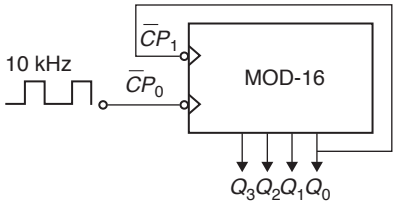
Practice Problems I

Directions for questions 1 to 22: Select the correct alternative from the given choices.

1. How many flip-flops are needed for MOD-16 ring counter and MOD-16 Johnson counter?
(A) 16, 16 (B) 16, 8
(C) 4, 3 (D) 4, 4

2. A 2-bit synchronous counter uses flip-flops with propagation delay time of 25 n sec, each. The maximum possible time required for change of state will be
(A) 25 n sec (B) 50 n sec
(C) 75 n sec D) 100 n sec

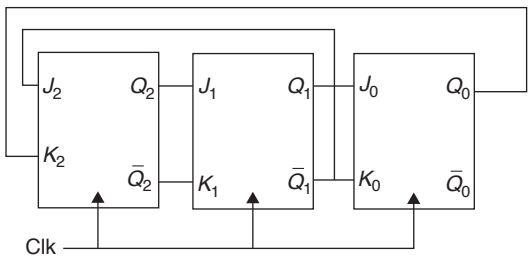
3. For given MOD-16 counter with a 10 kHz clock input determine the frequency at Q_3



(A) 625 Hz (B) 10 kHz
(C) 2.5 kHz (D) 0 Hz

4. A 4-bit ripple counter and a 4-bit synchronous counter are made using flip-flops having a propagation delay of 10 n sec each. If the worst case delay in the ripple counter and the synchronous counter be R and S, respectively, then
(A) $R = 10$ ns, $S = 40$ ns (B) $R = 40$ ns, $S = 10$ ns
(C) $R = 10$ ns, $S = 30$ ns (D) $R = 30$ ns, $S = 10$ ns

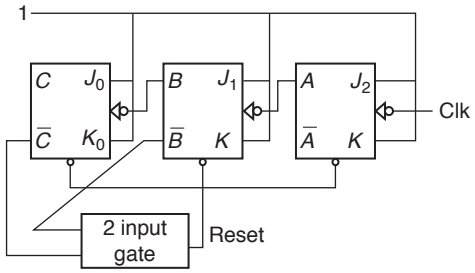
5. The counter shown in the figure has initially $Q_2Q_1Q_0 = 000$. The status of $Q_2Q_1Q_0$ after the first pulse is



(A) 001 (B) 010
(C) 100 (D) 101

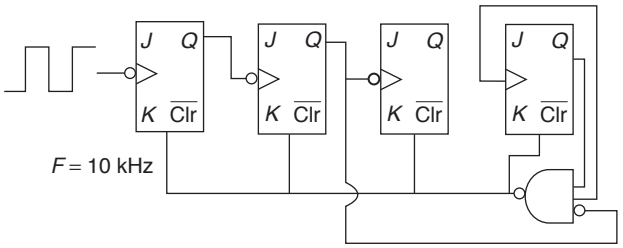
6. 12 MHz clock frequency is applied to a cascaded counter of MOD-3 counter, MOD-4 counter and MOD-5 counter. The lowest output frequency is
(A) 200 kHz (B) 1 MHz
(C) 3 MHz (D) 4 MHz

7. In the modulo-6 ripple counter shown in the figure below, the output of the 2-input gate is used to clear the JK flip-flops. The 2-input gate is



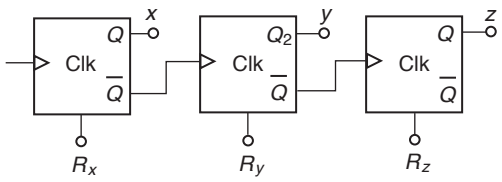
(A) a NAND gate (B) a NOR gate
(C) an OR gate (D) an AND gate

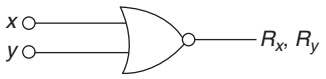
8. In figure, J and K inputs of all the 4 flip-flops are made high, the frequency of the signal at output y is

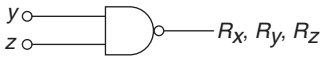


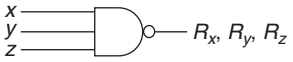
(A) 0.833 kHz (B) 1.0 kHz
(C) 0.91 kHz (D) 0.77 kHz

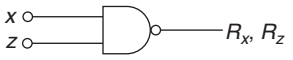
9. In a number system a counter has to recycle to 0 at the sixth count. Which of the connections indicated below will realize this resetting? (a logic '0' at the R inputs resets the counters)



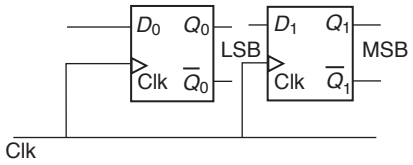
(A) 

(B) 

(C) 

(D) 

10. Two D flip-flops, as shown below, are to be connected as a synchronous counter that goes through the following Q_1Q_0 sequence $00 \rightarrow 01 \rightarrow 11 \rightarrow 10 \rightarrow 00$. The inputs D_0 and D_1 respectively should be connected as



(A) $\overline{Q}_0$ and Q_1 (B) $\overline{Q}_0$ and Q_1

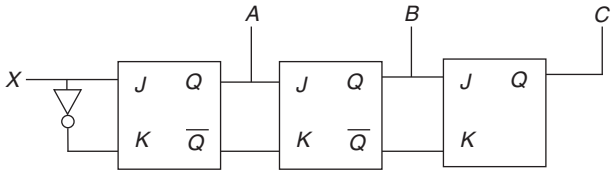
(C) $\overline{Q}_1\overline{Q}_0$ and $\overline{Q}_1Q_0$ (D) $\bar{Q}_1\bar{Q}_0$ and $\bar{Q}_1Q_0$

11. N flip-flops can be used to divide the input clock frequency by

(A) N (B) $2N$

(C) 2^N (D) 2^{N-1}

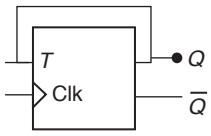
12. For a shift register as shown, $x = 1011$, with initially FF cleared, ABC will have value of after 3 clock pulses



(A) 101 (B) 011

(C) 001 (D) 111

13. If a FF is connected as shown what will be the output? (initially $Q = 0$)



(A) 11111 (B) 0000

(C) 1010 (D) 0101

14. The excitation table for a FF whose output conditions are if $AB = 00$, no change of state occurs
$AB = 01$, FF becomes 1 with next clock pulse
$AB = 10$, FF becomes 0 with next clock pulse
$AB = 11$, FF changes its state

(A)

Q_n	Q_{n+1}	A	B
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

(B)

Q_n	Q_{n+1}	A	B
0	0	1	x
0	1	0	x
1	0	x	0
1	1	x	1

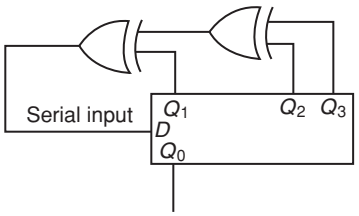
(C)

Q_n	Q_{n+1}	A	B
0	0	x	0
0	1	x	1
1	0	x	x
1	1	0	x

(D)

Q_n	Q_{n+1}	A	B
0	0	x	0
0	1	1	x
1	0	x	1
1	1	0	x

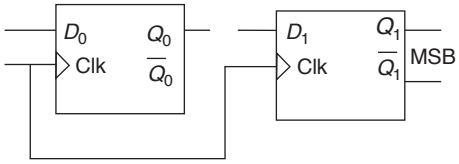
15. A shift register that shift the bits 1 position to the right at each clock pulse is initialized to 1100 (Q_0, Q_1, Q_2, Q_3). The outputs are combined using an XOR gate circuit and fed to the D input. After which clock pulse, will the initial pattern reappear at the output?



(A) 6th (B) 5th

(C) 4th (D) 7th

16. If we need to design a synchronous counter that goes through the states $00 \rightarrow 01 \rightarrow 11 \rightarrow 10 \rightarrow 00$ using D FF, what should be the input to the FF?



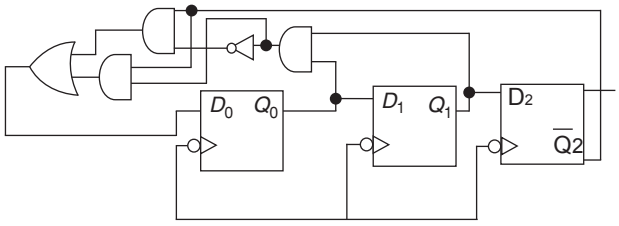
(A) $D_0 = Q_0, D_1 = \bar{Q}_1$

(B) $D_0 = \bar{Q}_1, D_1 = Q_0$

(C) $D_0 = \bar{Q}_1 \cdot Q_0, D_1 = \bar{Q}_1\bar{Q}_0$

(D) $D_0 = \bar{Q}_0, D_1 = Q_1$

17. Find the counter state sequence (Assume Q_0 as MSB).

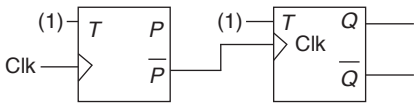


(A) 4, 6, 7, 3, 1, 0, 4 (B) 4, 6, 5, 3, 1, 0, 4

(C) 4, 5, 6, 7, 0, 4, 5 (D) 4, 6, 7, 1, 0, 4

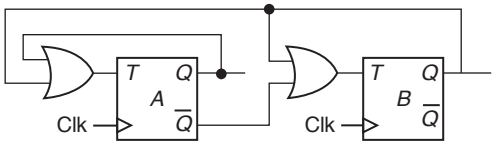
18. If the propagation delay of each FF is 50 ns, and for the AND gate to be 20 ns. What will be the f_{max} for MOD-32 ripple and synchronous counters?
(A) 14.3 MHz, 4 MHz (B) 14.3 MHz, 5 MHz
(C) 5 MHz, 14.3 MHz (D) 3.7 MHz, 14.3 MHz

19. For a given counter identify its behavior



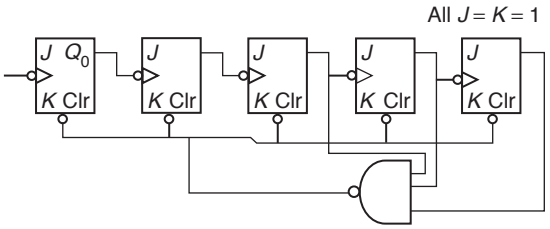
The output is taken from PQ.
(A) MOD-4 up counter
(B) MOD-2 down counter
(C) MOD-4 down counter
(D) MOD-2 up counter

20. A circuit using T FF is given. Identify the circuit.



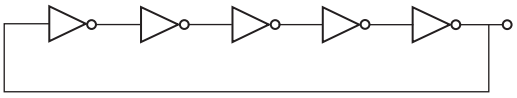
(A) MOD-2 counter
(B) MOD-4 counter
(C) MOD-3 counter
(D) MOD-2 generate 00, 10, 00

21. The MOD number of asynchronous counter shown



(A) 24 (B) 48
(C) 29 (D) 28

22. For the oscillator, find the fundamental frequency if propagation delay of each inverter is 1000 psec.



(A) 100 MHz (B) 10 MHz
(C) 1 GHz (D) 10 GHz

Practice Problems 2

Directions for questions 1 to 30: Select the correct alternative from the given choices.

1. Match List 1 (operation) with List 2 (associated device) and select the correct answer using the codes given below:

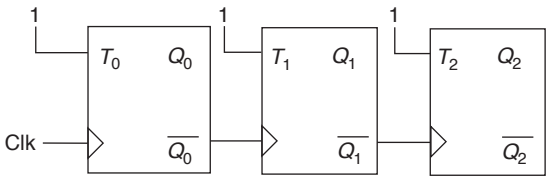
List 1	List 2
(a) Frequency Ddivision	(1) ROM
(b) Decoding	(2) Multiplexer
(c) Data selection	(3) Demultiplexer
(d) Code conversion	(4) Counter

(A) a–3, b–4, c–2, d–1
(B) a–3, b–4, c–1, d–2
(C) a–4, b–3, c–1, d–2
(D) a–4, b–3, c–2, d–1

2. A MOD-5 synchronous counter is designed by using JK flip-flop, the number of counts skipped by it will be
(A) 2 (B) 3
(C) 5 (D) 0

3. A counter starts off in the 0000 state, then clock pulses are applied. Some time later the clock pulses are removed and the counter flip-flops read 0011. How many clock pulses have occurred?
(A) 3 (B) 35
(C) 51 (D) Any of these

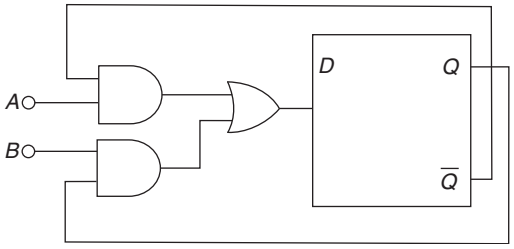
4. Figure below shown as ripple counter using positive edge triggered flip-flops. If the present state of the counters is $Q_2\,Q_1\,Q_0 = 011$, then its next state ($Q_2Q_1Q_0$) will be



(A) 010 (B) 100
(C) 111 (D) 101

5. A synchronous sequential circuit is designed to detect a bit sequence 0101 (overlapping sequence include). Every time, this sequence is detected, the circuit produces output of 1. What is the minimum number of states the circuit must have?
(A) 4 (B) 5
(C) 6 (D) 7

6. What is represented by digital circuit given below?



(A) An SR flip-flop with $A = S$ and $B = R$
(B) A JK flip-flop with $A = K$ and $B = J$
(C) A JK flip-flop with $A = J$ and $B = \overline{K}$
(D) An SR flip-flop with $A = R$ and $B = S$

7. In a ripple counter, the state whose output has a frequency equal to $\frac{1}{8}$th that of clock signal applied to the first stage, also has an output periodically equal to $\frac{1}{8}$th that of the output signal obtained form the last stage. The counter is
(A) MOD-8 (B) MOD-6
(C) MOD-64 (D) MOD-16

8. A flip-flop is popularly known as
(A) Astable multivibrator
(B) Bistable multivibrator
(C) Monostable multivibrator
(D) None of these

9. Which of the following represents the truth table for JK flip-flop?

(A)

J	K	Output
0	0	Q_0
0	1	0
1	0	1
1	1	$\overline{Q}_0$

(B)

J	K	Output
0	0	$\overline{Q}_0$
0	1	0
1	0	1
1	1	Q_0

(C)

J	K	Output
0	0	Q_0
0	1	0
1	0	1
1	1	Invalid

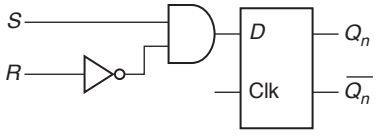
(D)

J	K	Output
0	0	1
0	1	0
1	0	1
1	1	0

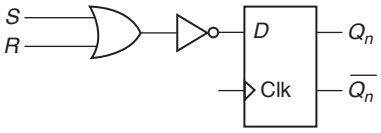
10. One disadvantage of master-slave FF is
(A) setup time becomes longer.
(B) it requires input to be held constant before clock transition.
(C) unpredictable output even if input held constant.
(D) hold time becomes longer.

11. Which of the following converts *D* FF to SR FF?

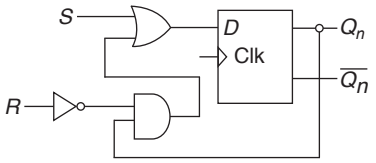
(A)



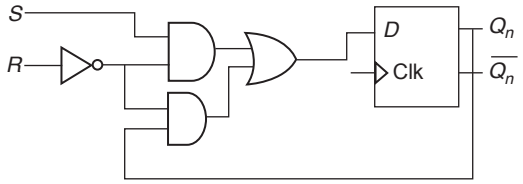
(B)



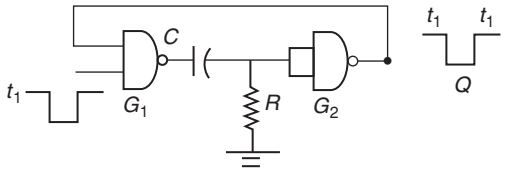
(C)



(D)



12. Which of the circuit is being represented by the figure?



(A) NAND gate
(B) Monostable multivibrator
(C) Astable multivibrator
(D) Schmitt trigger

13. Hold time is
(A) time for which output is held constant.
(B) time for which clock is to be held constant on applying input.
(C) time for which input should be maintained constant after the triggering edge of clock pulse.
(D) time for which input should be maintained constant prior to the arrival of triggering edge of clock pulse.

14. Shift registers are made up of
(A) MOS inverters (B) FF
(C) Latches (D) None of these

15. Data from a satellite is received in serial form. If the data is coming at 8 MHz rate, how long will it take to serially load a word in 40-bit shift register?
(A) 1.6 μs (B) 5 μs
(C) 6.4 μs (D) 12.8 μs

16. A JK FF can be converted into *T* FF by connecting
(A) $\overline{Q}$ to 0
(B) 0 to $\overline{Q}$
(C) 0 to Q
(D) by connecting both *J* and *K* inputs to *T*

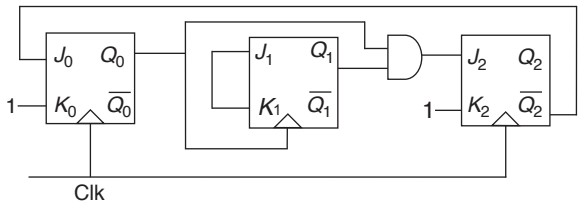
17. The flip-flop that is not affected by race around condition
(A) T FF (B) JK FF
(C) SR FF (D) None of these

18. The characteristic equation of JK FF is
(A) $J'Q(t) + KQ'(t)$ (B) $J'Q(t) + KQ(t)$
(C) $JQ'(t) + K'Q(t)$ (D) None of these

19. For a D.FF input, the s $\overline{Q}$ is connected. What would be the output sequence?
(A) 0000 (B) 1111
(C) 010101 (D) 101010

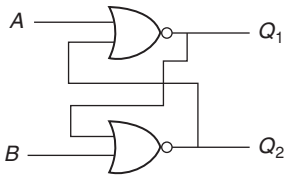
20. In order to implement a MOD-6 synchronous counter we have 3 FF and a combination of 2 input gate(s). Identify the combination circuit.
(A) One AND gate
(B) One OR gate
(C) One AND and one OR gate
(D) Two AND gates

21. Given a MOD-5 counter. The valid states for the counter are (0, 1, 2, 3, 4). The propagation delay of each FF is T_F and that of AND gate is t_A. The maximum rate at which counter will operate satisfactorily



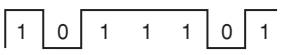
(A) $\frac{1}{t_F + t_A}$ (B) $\frac{1}{3t_F}$
(C) $\frac{1}{2t_F + t_A}$ (D) $\frac{1}{3t_F - t_A}$

22. For a NOR latch as shown up A and B are made first (0, 1) and after a few seconds it is made (1, 1). The corresponding output (Q_1, Q_2) are



(A) first (1, 0) then (0, 0)
(B) first (1, 0) then (1, 0)
(C) first (1, 0) then (1, 1)
(D) first (1, 0) then (0, 1)

23. In order to design a pulse generator to generate the wave form using a shift register, what is the number of FF required?

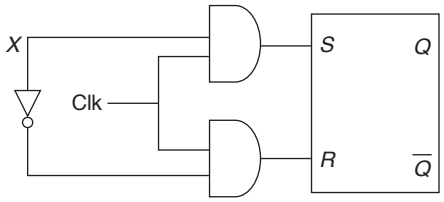


(A) 3 (B) 4
(C) 5 (D) 6

24. For what minimum value of propagation delay in each FF will a 10-bit ripple counter skip a count when it is clocked at 5 MHz?
(A) 10 ns (B) 20 ns
(C) 25 ns (D) 15 ns

25. A divide by 50 counter can be realized by using
(A) 5 no. of MOD-10 counter
(B) 10 no. of MOD-5 counter
(C) One MOD-5 counter followed by one MOD-10 counter
(D) 10 no. of MOD-10 counter

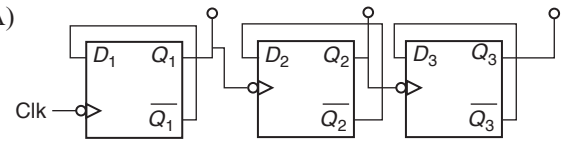
26. The following latch is



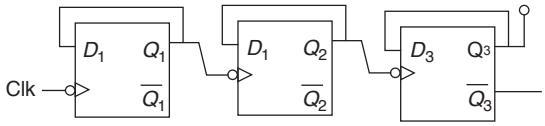
(A) D latch (B) T latch
(C) JK latch (D) RS latch

27. Which of the following represent a 3-bit ripple counter using D FF?

(A)

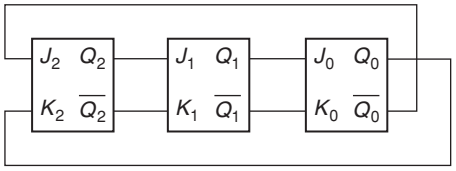


(B)



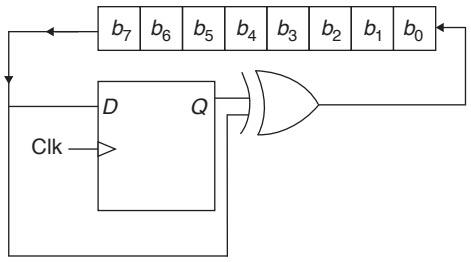
(C) Both (A) and (B)
(D) None of these

28. For the Johnson counter with initial Q_2, Q_1, Q_0 as 101, the frequency of the output is (Q_2, Q_1, Q_0)



(A) $\frac{f_c}{8}$ (B) $\frac{f_c}{6}$
(C) $\frac{f_c}{2}$ (D) $\frac{f_c}{4}$

29. For the given circuit the contents of register ($b_7 - b_0$) are 10010101, what will be the register contents after 8 clock pulses?



(A) 10010101 (B) 01101010
(C) 11011111 (D) 01101011

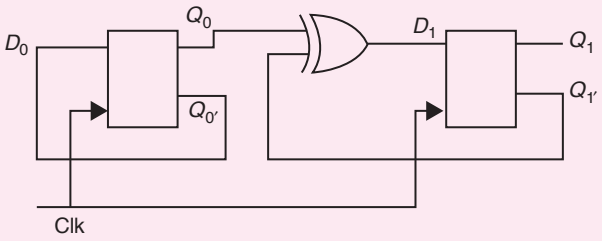
30. A latch is to be build with A and B as input. From the table find the expression for the next state Q^+

A	B	Q	Q^+
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

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

Previous Years' Questions

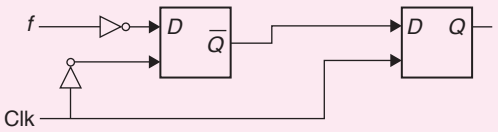
1. Consider the following circuit.

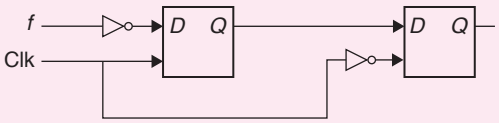


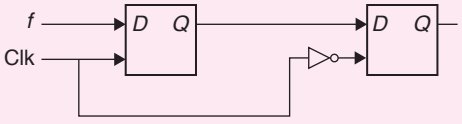
The flip-flops are positive edge triggered D FFs. Each state is designated as a 2-bit string Q_0Q_1. Let the initial state be 00. The state transition sequence is **[2005]**

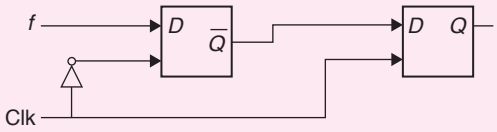
(A) 00 → 11 → 01
(B) 00 → 11
(C) 00 → 11 → 01 → 11
(D) 00 → 11 → 01 → 10

2. You are given a free running clock with a duty cycle of 50% and a digital waveform f which changes only at the negative edge of the clock. Which one of the following circuits (using clocked D flip-flops) will delay the phase of f by 180°? **[2006]**

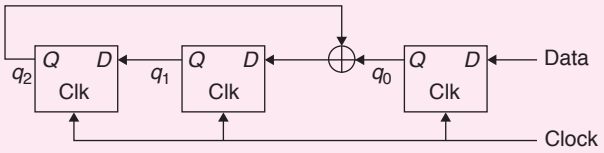
(A)


(B)


(C)


(D)


3. Consider the circuit in the diagram. The ⊕ operator represents Ex-OR. The D flip-flops are initialized to zeros (cleared). **[2006]**



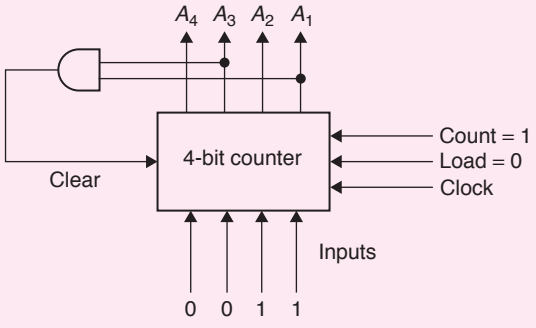
The following data: 100110000 is supplied to the data terminal in 9 clock cycles. After that the values of $q_2q_1q_0$ are

(A) 000 (B) 001
(C) 010 (D) 101

4. The control signal functions of a 4-bit binary counter are given below (where X is 'don't care'):

Clear	Clock	Load	Count	Function
1	X	X	X	Clear to 0
0	X	0	0	No change
0	↑	1	X	Load input
0	↑	0	1	Count next

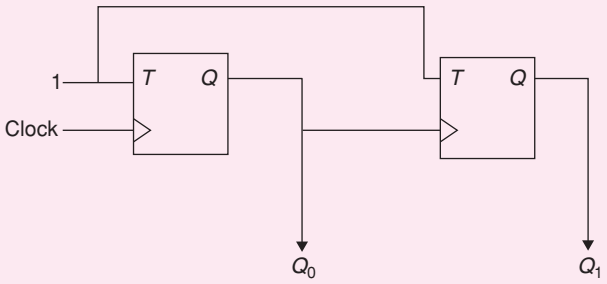
The counter is connected as follows:



Assume that the counter and gate delays are negligible. If the counter starts at 0, then it cycles through the following sequence: **[2007]**

(A) 0, 3, 4 (B) 0, 3, 4, 5
(C) 0, 1, 2, 3, 4 (D) 0, 1, 2, 3, 4, 5

5. In the sequential circuit shown below, if the initial value of the output Q_1Q_0 is 00, what are the next four values of Q_1Q_0? **[2010]**

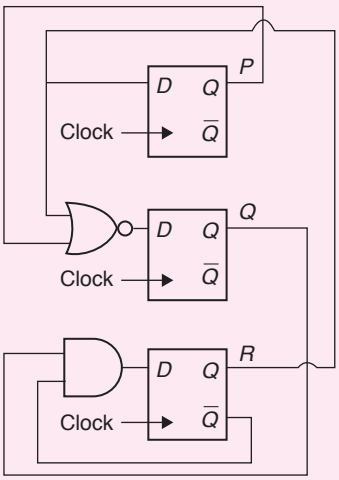


(A) 11, 10, 01, 00 (B) 10, 11, 01, 00
(C) 10, 00, 01, 11 (D) 11, 10, 00, 01

6. The minimum number of D flip-flops needed to design a MOD-258 counter is **[2011]**

(A) 9 (B) 8
(C) 512 (D) 258

Common Data for Questions 7 and 8: Consider the following circuit involving three D-type flip-flops used in a certain type of counter configuration.



7. If all the flip-flops were reset to 0 at power on, what is the total number of distinct outputs (states) represented by PQR generated by the counter? **[2011]**

(A) 3 (B) 4
(C) 5 (D) 6

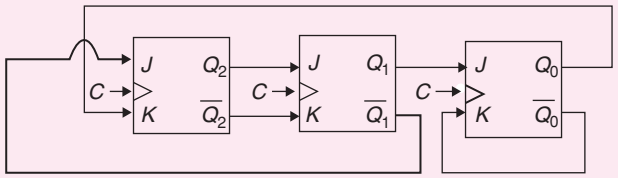
8. If at some instance prior to the occurrence of the clock edge, P, Q, and R have a value 0, 1, and 0, respectively, what shall be the value of PQR after the clock edge? **[2011]**

(A) 000 (B) 001
(C) 010 (D) 011

9. Let $K = 2^n$. A circuit is built by giving the output of an n-bit binary counter as input to an n-to-2^n-bit decoder. This circuit is equivalent to a **[2014)**

(A) K-bit binary up counter
(B) K-bit binary down counter
(C) K-bit ring counter
(D) K-bit Johnson counter

10.



The above synchronous sequential circuit built using JK flip-flops is initialized with $Q_2Q_1Q_0 = 000$. The state sequence for this circuit for the next 3 clock cycles is **[2014]**

(A) 001, 010, 011 (B) 111, 110, 101
(C) 100, 110,111 (D) 100, 011, 001

11. Consider a 4-bit Johnson counter with an initial value of 0000. The counting sequence of this counter is **[2015]**

(A) 0, 1, 3, 7, 15, 14, 12, 8, 0
(B) 0, 1, 3, 5, 7, 9, 11, 13, 15, 0
(C) 0, 2, 4, 6, 8, 10, 12, 14, 0
(D) 0, 8, 12, 14, 15, 7, 3, 1, 0

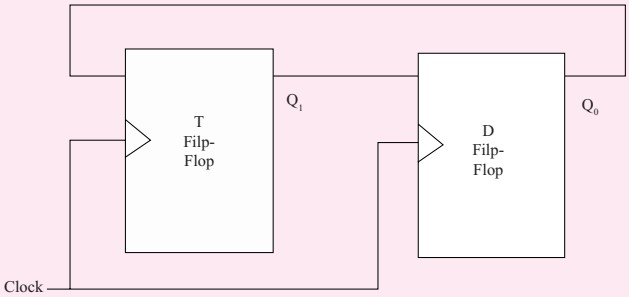
12. A positive edge-triggered D–flip-flop is connected to a positive edge-triggered JK flip-flop as follows. The Q output of the D flip-flop is connected to both the J and K inputs of the JK flip-flop, while the Q output of the JK flip-flop is connected to the input of the D flip-flop. Initially, the output of the D flip-flop is set to logic one and the output of the JK flip-flop is cleared. Which one of the following is the bit sequence (including the initial state) generated at the Q output of the JK flip-flop when the flip-flops are connected to a free-running common clock? Assume that $J = K = 1$ is the toggle mode and $J = K = 0$ is the state-holding mode of the JK flip-flop. Both the flip-flops have non-zero propagation delays. **[2015]**

(A) 0110110… (B) 0100100…
(C) 011101110… (D) 011001100…

13. The minimum number of JK flip-flops required to construct a synchronous counter with the count sequence (0, 0, 1, 1, 2, 2, 3, 3, 0, 0, …) is _____ **[2015]**

14. We want to design a synchronous counter that counts the sequence 0-1-0-2-0-3 and then repeats. The minimum number of *J-K* flip-flops required to implement this counteris _____ . **[2016]**

15. Consider a combination of T and D flip-flops connected as shown below. The output of the D flip-flop is connected to the input of the T flip-flop and the output of the T flip-flop is connected to the input of the D flip-flop.



Initially, both Q_0 and Q_1 are set to 1 (before the 1st clock cycle). The outputs **[2017]**

(A) $Q_1 Q_0$ after the 3rd cycle are 11 and after the 4th cycle are 00 respectively

(B) $Q_1 Q_0$ after the 3rd cycle are 11 and after the 4th cycle are 01 respectively

(C) $Q_1 Q_0$ after the 3rd cycle are 00 and after the 4th cycle are 11 respectively

(D) $Q_1 Q_0$ after the 3rd cycle are 01 and after the 4th cycle are 01 respectively

16. The next state table of a 2-bit saturating up-counter is given below.

Q_1	Q_0	Q_1^+	Q_0^+
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	1

The counter is built as a synchronous sequential circuit using T flip-flops. The expressions for T_1 and $T_0$are **[2017]**

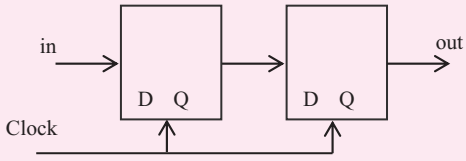
(A) $T_1 = Q_1 Q_0$, $\quad T_0 = \overline{Q}_1 \overline{Q}_0$

(B) $T_1 = \overline{Q}_1 Q_0$, $\quad T_0 = \overline{Q}_1 + \overline{Q}_0$

(C) $T_1 = Q_1 + Q_0$, $\quad T_0 = \overline{Q}_1 + \overline{Q}_0$

(D) $T_1 = \overline{Q}_1 Q_0$, $\quad T_0 = Q_1 + Q_0$

17. Consider the sequential circuit shown in the figure, where both flip-flops used are positive edge-triggered D flip-flops.



The number of states in the state transition diagram of this circuit that have a transition back to the same state on some value of "in" is _____. **[2018]**

Answer Keys

Exercises

Practice Problems 1

1. B	**2.** A	**3.** A	**4.** B	**5.** C	**6.** A	**7.** C	**8.** A	**9.** B	**10.** A
11. C	**12.** B	**13.** B	**14.** C	**15.** D	**16.** B	**17.** A	**18.** D	**19.** A	**20.** C
21. D	**22.** A								

Practice Problems 2

1. D	**2.** B	**3.** D	**4.** B	**5.** A	**6.** C	**7.** C	**8.** B	**9.** A	**10.** B
11. C	**12.** B	**13.** C	**14.** B	**15.** B	**16.** D	**17.** C	**18.** C	**19.** C	**20.** D
21. C	**22.** A	**23.** D	**24.** B	**25.** C	**26.** A	**27.** A	**28.** C	**29.** C	**30.** A

Previous Years' Questions

1. D	**2.** C	**3.** C	**4.** C	**5.** A	**6.** A	**7.** B	**8.** D	**9.** C	**10.** C
11. D	**12.** A	**13.** $3(8 = 2^3)$		**14.** 3	**15.** B	**16.** B	**17.** 2		

TEST

DIGITAL LOGIC

Time: 60 min.

Directions for questions 1 to 30: Select the correct alternative from the given choices.

1. What is the range of signed decimal numbers that can be represented by 4-bit 1's complement notation?
(A) –7 to + 7 (B) –16 to +16
(C) –7 to +8 (D) –15 to +16

2. Which of the following signed representation have a unique representation of 0?
(A) Sign-magnitude (B) 1's complement
(C) 0's complement (D) 2's complement

3. Find the odd one out among the following
(A) EBCDIC (B) GRAY
(C) Hamming (D) ASCII

4. Gray code for number 8 is
(A) 1100 (B) 1111
(C) 1000 (D) 1101

5. Find the equivalent logical expression for $z = x + \bar{x}y$
(A) $z = xy$ (B) $Z = \bar{x}y$
(C) $Z = \bar{x} + y$ (D) $Z = x + y$

6. The number of distinct Boolean expression of 3 variables is
(A) 256 (B) 16
(C) 1024 (D) 65536

7. The Boolean expression for the truth table shown is

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

(A) $Y(X + Z)(\bar{X} + \bar{Z})$ (B) $Y(X + \bar{Z})(\bar{x} + Z)$
(C) $\bar{Y}(X + \bar{Z})(\bar{x} + Z)$ (D) $\bar{Y}(X + Z)(\bar{X} + \bar{Z})$

8. The number of essential prime implicants for the Boolean functions shown in the given K-map.

XY \ WZ	00	01	11	10
00	1	1	0	1
01	1	0	0	1
11	1	0	0	0
10	1	0	0	1

(A) 4 (B) 5
(C) 6 (D) 8

9. The number of product terms in the minimized SOP from is

1	0	0	1
0	D	0	0
0	0	D	1
1	0	0	1

(A) 2 (B) 4
(C) 5 (D) 3

10. The minimum number of 2 input NAND gates needed to implement $Z = XY + VW$ is
(A) 2 (B) 3
(C) 4 (D) 5

11. The operation $\bar{a} \oplus \bar{b}$ represents
(A) $ab + \bar{a}\,\bar{b}$ (B) $\bar{a}b + \overline{ab}$
(C) $a\bar{b} + \bar{a}b$ (D) $a - \bar{b}$

12. Find the dual of $X + [Y + XZ] + U$
(A) $X + [Y(X + Z)] + U$ (B) $X(Y + XZ)U$
(C) $X + [Y(X + Z)]U$ (D) $X[Y(X + Z)]U$

13. The simplified form of given function $AB + BC + A\bar{C}$ is equal to
(A) $AB + A\bar{C}$ (B) $A\bar{C} + BC$
(C) $\bar{A}C + BC$ (D) $A\bar{B} + A\bar{C}$

14. Simplify the following

WX \ YZ				
	1	1	0	1
	1	1	0	1
	0	0	1	1
	0	0	0	0

(A) $\bar{W}\bar{Y} + \bar{W}\bar{Z} + WXY$
(B) $\bar{W}\bar{X} + \bar{W}\bar{Z} + WXY$
(C) $WY + WYZ + WXY + XY\bar{Z}$
(D) $\bar{W}\bar{X} + \bar{Y}\bar{Z} + \bar{W}\bar{Z}$

15. Simplify the following
$F = ABCD + A\bar{B}CD + \bar{A}C\bar{B}D + \bar{A}BCD$
(A) CD (B) BC
(C) AB (D) $\bar{C} + \bar{D}$

16. Find the equivalent Boolean expression for $AC + B\bar{C}$
(A) $\bar{A}C + B\bar{C} + AC$
(B) $ABC + A\bar{B}C + AB\bar{C} + \bar{A}B\bar{C}$
(C) $ABC + A\bar{B}C + AB\bar{C} + \bar{A}\bar{B}\bar{C}$
(D) $\bar{A}C + B\bar{C} + \bar{A}\bar{C}$

17. Simplify the following expression

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

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

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

18. If $A = 1$ in the logic equation $[A + C\{\bar{B} + (\bar{C} + A\bar{B})\}]$ $[\bar{A} + \bar{C}(A + B)] = 1$, then

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

(C) $C = 1$ (D) $C = 0$

19. Which is the odd function with 3 Boolean variables in it

(A) $\Sigma(0, 3, 5, 6)$ (B) $\Sigma(0, 2, 4, 6)$

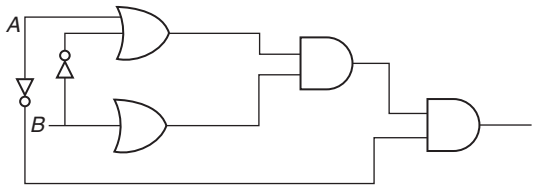
(C) $\Sigma(1, 2, 4, 7)$ (D) $\Sigma(1, 3, 5, 7)$

20. Which of the following expressions is/are incorrect?

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

(C) $\overline{\overline{a}\;\overline{b}} = \bar{a} + \bar{b}$ (D) $\overline{\overline{a+b}} = \bar{a}\,\bar{b}$

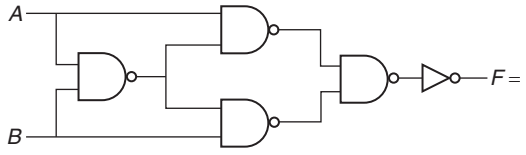
21. The simplified form of logic circuit is



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

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

22. The circuit shown in figure is equivalent to —— gate.



(A) X-OR gate (B) EX-NOR gate

(C) Half adder (D) Half subtractor

23. The truth table of the circuit shown in figure

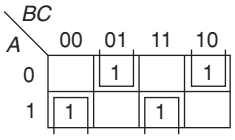
A	B	C	Z
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

The Boolean expression for Z

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

(C) $\overline{\overline{(\bar{A}+B)}\,\overline{(B+\bar{C})}}$ (D) All of the above

24. A combinational circuit has input A, B and C and its K-map is as shown in figure. The output of the circuit is given by



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

(C) $\bar{A}\,\bar{B}\bar{C}$ (D) $A \oplus B \oplus C$

25. Which of the following two 2-input gates will realize the Boolean expression $X(P, Q, R) = \pi(0, 5)$

(A) AND and OR (B) NAND and OR

(C) AND and X-OR (D) OR and X-OR

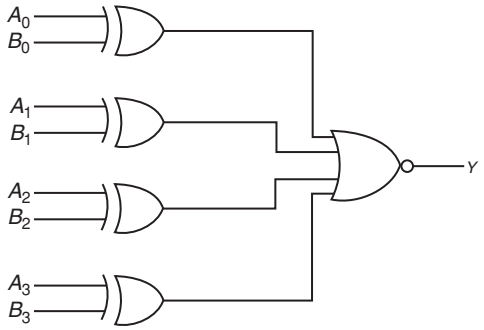
26. Simplify the given function

$f(x, y, z) = \Sigma m(0, 2, 3, 4, 5, 7)$

(A) $\bar{x}y + \bar{y}\bar{z} + xz$ (B) $\bar{x}\,\bar{z} + x\bar{y} + yz$

(C) Both (A) and (B) (D) $\bar{x}\,\bar{z} + \bar{x}y + x\bar{y} + xz$

27. Figure below shows a digital circuit, which compares two numbers $A_0\ A_1\ A_2\ A_3$, $B_0\ B_1\ B_2\ B_3$. Choose the pair of correct input number to get output $Y = 0$.



(A) 1100, 1100 (B) 0110, 0110

(C) 1011, 0010 (D) 1011, 1011

28. How many 3 to 8 line decoders with an enable input are required to build 6 of 34 decoder?

(A) 6 (B) 2

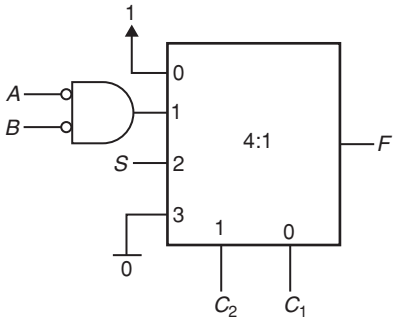
(C) 9 (D) 4

29. It is required to construct a 2^n to 1 multiplexer by using 2-to-1 multiplexer only. How many of 2-to-1 multiplexer are needed?

(A) n (B) 2^{2n}

(C) 2^{n-1} (D) $2^n - 1$

30. Consider the following circuit



Which one of the following give the function implemented by the MUX based digital circuit?

(A) $F = C_2 \cdot \overline{C_1}S + \overline{C_2}C_B(\overline{A} + \overline{A})$

(B) $F = \overline{C_2} \cdot \overline{C_1} + C_2C_1 + C_2\overline{C_1}S + \overline{C_2}C_1\,\overline{AB}$

(C) $F = \overline{AB} + S$

(D) $F = \overline{C_2} \cdot \overline{C_1} + C_2 \cdot \overline{C_1}S + \overline{C_2}C_1\overline{A} \cdot \overline{B}$

Answer Keys

1. A	**2.** D	**3.** C	**4.** A	**5.** D	**6.** A	**7.** A	**8.** A	**9.** A	**10.** B
11. C	**12.** D	**13.** B	**14.** A	**15.** A	**16.** B	**17.** A	**18.** D	**19.** C	**20.** D
21. D	**22.** B	**23.** B	**24.** D	**25.** D	**26.** C	**27.** C	**28.** C	**29.** D	**30.** D