1. LOGIC FUNCTIONS

1. BASIC PROPERITIES OF SWITCHING ALGEBRA

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

$(xy)\cdot z = x(yz)$

Idempotency: $x \cdot x = x$ | $x+1 = 1$

$x+x = x$ | $x \cdot 0 = 0$

$x+0 = x$

$x \cdot 1 = x$

commutativity: $x+y = y+x$

$xy = y \cdot x$

Complementation: $x+\bar{x} = 1$

$x \cdot \bar{x} = 0$

Distributive: $x(y+z) = xy+xz$

$\boxed{x+yz = (x+y)(x+z)}$

$+$ = OR

$\cdot$ = AND

$-$ = NOT

2. SWITCHING EXPRESSIONS AND SIMPLIFICATIONS

Switching expression is a finite no. of combinations of switching variables and constants $\{0,1\}$ by means of switching operations $(+, \cdot, \text{NOT})$

Ex: $x+\bar{x}y^2+x\bar{z}$, $a+bc+\bar{b}d$

Properities for simplifying SE

Absorption: $x+xy = x$

$= x \cdot 1 + xy$

$= x(1+y)$

$= x(1)$

$= x$ //.

$x+x'y = x+y$

LHS $= (x+x')(x+y)$

$= 1\,(x+y)$

$= x+y$ = RHS.

$x(x'+y) = xy$

$x \cdot x' + xy$

$= 0 + xy$

$= xy$ = RHS.

** Consensus theorem: $xy+\bar{x}z+yz = xy+\bar{x}z$

$= xy+\bar{x}z+yz(1)$

$= xy+\bar{x}z+yz(x+\bar{x})$

$= xy+\bar{x}z+yzx+yz\bar{x}$

$= xy(1+z)+xz(1+y)$

$= xy+\bar{x}z$

If the value of an Expr does not depend on any term then it is called Redundant Expr

Minimise $x'y'z+yz+xz$

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

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

$= z(x'+y+x)$ $= z(1+y)$

$= z$.

4. DEMORGANS LAW AND SIMPLIFICATION

① $(\overline{xy}) = \bar{x} + \bar{y}$ ② $(\overline{x+y}) = \bar{x}\,\bar{y}$

$f(a,b,c---z,0,1,\cdot,+)$

$\bar{f} = (\bar{a},\bar{b},\bar{c}---\bar{z},1,0,+,\cdot)$

f_d = duality = $(a,b,c,---z,1,0,+,\cdot)$

$f = \bar{x} + \bar{y}$

$\bar{f} = x \cdot y$

$f_d = \bar{x} \cdot \bar{y}$

Simplify $(x+y)[x'(y'+z')]' + x'y' + x'z'$

$= (x+y)\left[x + (yz)\right] + x'y' + x'z'$

$= x + xyz + yx + yz + x'y' + x'z'$

$= x(1+yz) + yx + yz + x'y' + x'z'$

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

$= x(1) + yz + x'y' + x'z'$

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

$= x + y' + yz + x'z'$

$= x + z' + y' + yz$

$= x + z' + y' + z = x + y' + 1 = x + y' + 1$

$= 1$ //

5. CANONICAL SUM OF PRODUCTS

① → A product term which contains each of 'n' variables as factors either in complemented or uncomplemented form is called a minterm.

② → A minterm given the value '1' for exactly one combination of the variables

③ → The sum of all minterms of 'f' for which 'f' assumes '1' is called canonical sum of products or disjunctive Normal form.

F(a,b,c) then the no. of min terms = 8

F(a,b,c---n) then the no. of min terms = 2^n //

i) $f(a,b,c) = \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}c + \bar{a}b\bar{c} + \bar{a}bc + a\bar{b}\bar{c} + a\bar{b}c + ab\bar{c} + abc$ ③

= ab is not a minterm (it should contain all the variables a,b,c in complemented or

ii) 2nd point: $\bar{a}\bar{b}\bar{c}$

$\Rightarrow 000 \Rightarrow \bar{a}\bar{b}\bar{c} = 1$

$\bar{a}\bar{b}c$

$001 \Rightarrow \bar{a}\bar{b}c = 1$

for the values of 001 the minterm $\bar{a}\bar{b}c$ gives '1'

iii).

a	b	c	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

⇓

canonical sum of products

Sum of products ⇒ need not contain all the variables

$f(a,b,c) \Rightarrow a + bc + ac$

Canonical sum of products ⇒ should contain all the variables

$= abc + \bar{a}bc$

$f = \underbrace{\bar{a}\bar{b}c + \bar{a}b\bar{c} + a\bar{b}\bar{c} + ab\bar{c}}_{\text{canonical sum of products.}} = \Sigma(1,2,4,6)$ → compact Representation.

6. CANONICAL PRODUCT OF SUMS

→ A sum term which contains each of 'n' variables as factors either in complemented or uncomplemented form is called a maxterm.

→ A Maxterm gives the value '0' for exactly one combination of the variables

→ The product of all maxterms of 'f' for which 'f' assumes '0' is called canonical product of sums or Conjunctive Normal form.

	a	b	c	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

$f = [(a+b+c)(a+\bar{b}+\bar{c})(\bar{a}+b+\bar{c})(\bar{a}+\bar{b}+\bar{c})]$

1 = complemented form

0 = uncomplemented form.

7. EXAMPLES ON CANONICAL FORMS

	a	b	c	f1	f2	f3
1)	0	0	0	0	1	1
2)	0	0	1	0	0	1
3)	0	1	0	1	1	0
4)	0	1	1	1	0	0
5)	1	0	0	0	1	0
6)	1	0	1	1	0	0
7)	1	1	0	1	1	1
8)	1	1	1	1	0	1

f_1 = sum of products = minterms = $\Sigma(3,4,6,7,8)$

product of sum = $\pi(1,2,5)$

f_2 = sum of products = $\Sigma(1,3,5,7)$

product of sum = $\pi(2,4,6,8)$

$f_3 = \Sigma(1,2,7,8) = \pi(3,4,5,6)$

1> $f(x,y,z) = x'y + z' + xyz$ convert into canonical form?

$f(x,y,z) = x'y + z' + xyz$ (sum of products) but not (canonical sum of products)

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

$= x'yz + x'yz' + z'(xy + xy' + x'y + x'y') + xyz$

$= x'yz + x'yz' + z'(xy) + xy'z' + x'yz' + x'y'z' + xyz$

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

$= \Sigma(0,2,3,4,6,7) = \pi(1,5)$

8. FUNCTIONAL PROPERTIES

→ The canonical SOP or POS form a switching function is unique

→ Two switching functions $f_1(x_1 \cdots x_n)$ and $f_2(x_1 \cdots x_n)$ are said to be logically Equivalent iff both functions have some value for each and every combination of $(x_1 x_2 \cdots x_n)$

→ Two switching functions are equivalent if their canonical POS or SOP are identical.

9. NO. OF FUNCTIONS

n- Bodean variables

How many boolean variables

1	2	3	4	---	n	f
2	× 2	× 2	× 2	--	× 2	

combinations of

⇒ 2^n ↑ boolean variables are present

⇒ Now, each combination has 2 choices (maybe present / may be absent)

∴ No. of Bodean functions possible are = $2^{(2^n)}$

⇒ n- ternary variables

⇒ no.of combinations of n-ternary variables = 3^n $\left\{\begin{matrix}1,2,3 & \cdots & n\\ \downarrow & \downarrow & \downarrow\\ 3 & 3 & 3\end{matrix}\right\}$

⇒ **No.of functions = $(2)^{(3^n)}$**

⇒ 'n' K-ary variables, How many m-ary functions possible?

1	2	3	- - -	n	f
k	k	k	- - -	K	

⇒ k^n combinations ∴ **No.of functions = $m^{(k^n)}$**

10. COUNTING NO.OF FUNCTIONS AND NEUTRAL FUNCTIONS

1) How many boolean functions are possible with 3 variables such that there are exactly 3 minterms?

a	b	c	f
0	0	0	
0	0	1	
0	1	0	(3 1's)
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

⇒ Now, we should assign '3' ones to the 8 combinations of the boolean variables.

$8C_3$ ways

2) Atmost 3 minterms = **$8C_0 + 8C_1 + 8C_2 + 8C_3$**

3) Atleast 3 minterms

⇒ **$8C_3 + 8C_4 + 8C_5 + \cdots + 8C_8$**

3) Assume that there are K variables then 2^K combinations of boolean variables are possible, Now if they want to assign 'm' minterms exactly then the no.of ways = $(2^K)C_m$

4) How many Neutral functions are possible with two boolean variables?

A Neutral func is a func in which no.of minterms = No.of maxterms.

(No.of 1's = No.of 0's)

A	B	f=A	f=$\bar{A}$	f=$\bar{B}$	f=B	f=$\odot$ (EX-NOR)	f=A⊕B. (EX-OR)
0	0	0	1	1	0	1	0
0	1	0	1	0	1	0	1
1	0	1	0	1	0	0	1
1	1	1	0	0	1	1	0

Neutral functions = $4C_2$.

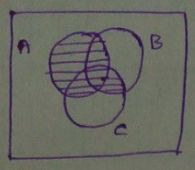
'n' variables = 2^n combinations

$= (2^n)C_{(2^n/2)}$

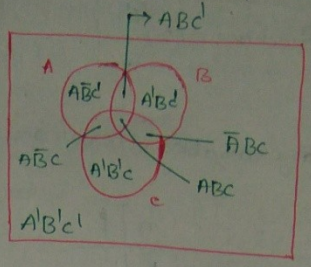
$= (2^n)C_{(2^{n-1})}$

11. VENN DIAGRAM REPRESENTATION

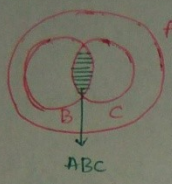
Find the Boolean function that the following venn diagram represents.

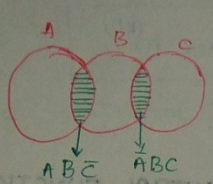


a) $A + A'B'c$
b) $A + BC$
c) ~~$A + A'BC$~~
d) $AB + C$



$F = [A + \bar{A}BC] \rightarrow$ Ans.



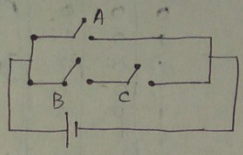


12. CONTACT REPRESENTATION

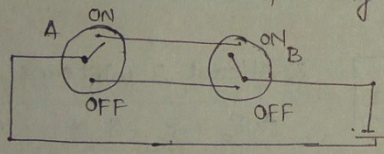
→ Every Boolean function can be represented with the help of serial and parallel contact

→ Serial contacts are performing "AND" operation

→ parallel contacts are performing "OR" operation.

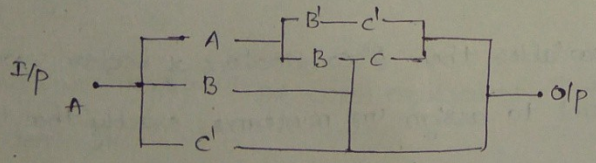


Which function does this following circuit represent.



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

I> Identify the boolean expression given by the following circuit



① Find the valid forward path

② perform OR among them.

Forward path:
Any path starting from i/p and ending at o/p without forming cycle

Validity: No path should contain a variable in both true and complemented form.

In above example are: $AAB'c + AABC + AABA + ABC + ABA + \underbrace{Ac'c}$ → Invalid forward paths.

13. NESTED FUNCTION

In the following simultaneous Boolean Expressions what are the values of w, x, y, z.

i) $x+y+z=1$

ii) $xy+wz'=0$

iii) $xw'+yz'=1$

N	x	y	z	
a)	0	0	0	1 X
b)	1	1	0	1 X
c)	0	1	0	1 ✓
d)	1	0	0	0

② $f(A,B) = A'+B$ then find $f(f(x+y,y),z))$

Now, $f(x+y,y) = (x+y)'+y$

$= x'\cdot y'+y$

$= (x'+y)(y+y') = x'+y$ //.

Now $f(x'+y,z)$

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

$= xy'+z$

14. NAND GATE AND PROPERITIES

NAND = NOT AND. $= (\overline{A \cdot A})$

⇒ NAND gate does not obey Identity

⇒ $\boxed{A \uparrow A \neq A}$

A	B	$A \uparrow B$
0	0	1
0	1	1
1	0	1
1	1	0

⇒ $\overline{A \cdot B} = \overline{B \cdot A}$

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

⇒ $\boxed{A \uparrow (B \uparrow C) \neq (A \uparrow B) \uparrow C}$

↓ NAND is not Associative

15. NOR GATE AND PROPERITIES

NOR = NOT OR

A	B	$(A \downarrow B)$
0	0	1
0	1	0
1	0	0
1	1	0

⇒ $A \downarrow B = (\overline{A+B})$

⇒ $(A \downarrow A) \neq (A) = (\overline{A})$

⇒ $(A \downarrow B) = (B \downarrow A)$

⇒ NOR is not Associative

$A \downarrow (B \downarrow C) = (A \downarrow B) \downarrow C$

16. EX-OR GATE AND PROPERITIES

A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

$A \oplus B = \overline{A}B + A\overline{B}$ (modulo 2 sum)

$1 \oplus 1 \oplus 1 = 3 \bmod 2 = 1$ ✓

$1 \oplus 0 \oplus 1 = 2 \bmod 2 = 0$ ✓

⇒ $A \oplus A \neq A$ $(1 \oplus 1 = 0 \neq 1)$

⇒ $A \oplus B = B \oplus A$

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

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

17. EXNOR GATE AND PROPERITIES

EXNOR = Exclusive -NOR. = Negation of EX OR

A	B	A⊙B
0	0	1
0	1	0
1	0	0
1	1	1

$\Rightarrow$ $\boxed{A \odot B = AB + \bar{A}\bar{B}}$

$\Rightarrow A \odot A \neq A$

$\Rightarrow A \odot B = B \odot A$

$\Rightarrow A \odot (B \odot C) = (A \odot B) \odot C$

$\Rightarrow A \odot B = \bar{A}\bar{B} + AB$

$\Rightarrow \overline{A \odot B} = A' \odot B = A \odot B'$

$\Rightarrow \overline{A \odot B} = A \oplus B = A' \odot B = A \odot B'$

$\Rightarrow \overline{A \oplus B} = A \odot B = A' \oplus B = A \oplus B'$

18. PROPERITIES OF EX-OR AND EX-NOR

Now we have checked the Ex-OR and Ex-NOR both are compliments to each other

XOR

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

X-NOR

A	B	A⊙B
0	0	1
0	1	0
1	0	0
1	1	1

$\oplus = 1$, for odd no of 1's

$\odot = 1$ for even no of 0's.

Now, for three variables both XOR and X-NOR are showing same? why?

A	B	C	A⊕B⊕C	A⊙B⊙C
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	0
1	0	0	1	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Now, Let $\boxed{n = \text{no of inputs.}}$

If 'n' is even:

If 1's are odd $\Rightarrow$ no of 0's are odd

for such combinations $\oplus = 1$ $\odot = 0$

If n is even:

If no of 1s even $\Rightarrow$ no of 0's are even

$\oplus = 0$, $\odot = 1$

$\therefore$ If n = even both XOR, XNOR are compliment

If n = odd

$\Rightarrow$ If 1's odd $\Rightarrow$ 0's is even

$\oplus = 1$, $\odot = 1$

$\Rightarrow$ If 1's are even $\Rightarrow$ 0's is odd

$\oplus = 0$, $\odot = 0$ //

$\therefore$ If n = odd both XOR, XNOR are same

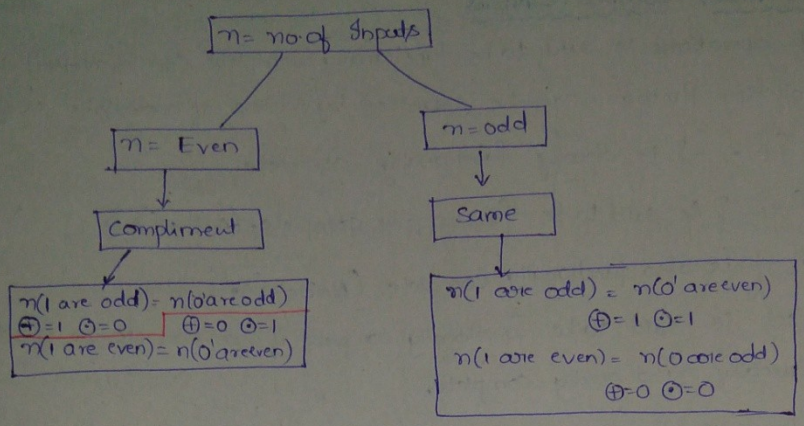
$\therefore$ From the above proofs

i) $A \oplus B = \overline{A \odot B}$

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

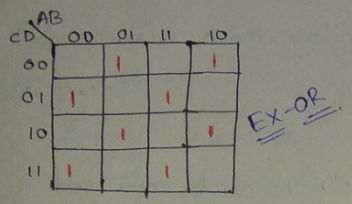
iii) $A \oplus B \oplus C \oplus D = A \odot B \odot C \odot D$

iv) $A \oplus B \oplus C \oplus D \oplus E = A \odot B \odot C \odot D \odot E$



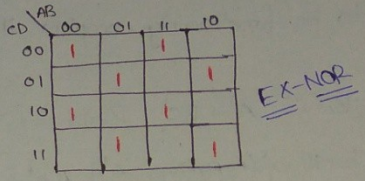
Now, drawing Karnaugh maps:

i) $A \oplus B \oplus C \oplus D$



{ ones are filled at positions where no. of 0's are odd ∴ it is EX-OR we cannot minimise them further }

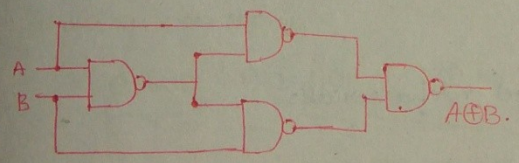
ii) $A \odot B \odot C \odot D$



{ ones are filled at positions where no. of 0's are even ∴ it is EX NOR. we cannot minimise them further. }

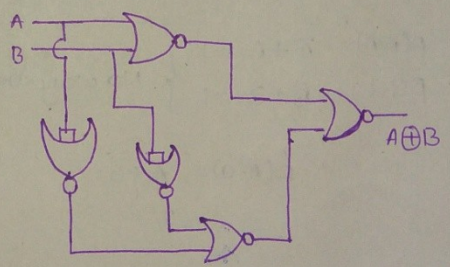
19. MINIMUM NUMBER OF GATES REQUIRED XOR AND XNOR

Ex-OR (using NAND)



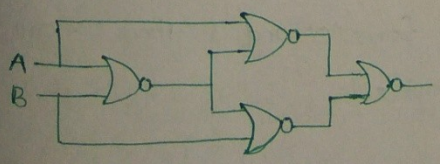
Min no. of NAND gates Reqd for build XOR gate is = 4

Ex-OR (using NOR)



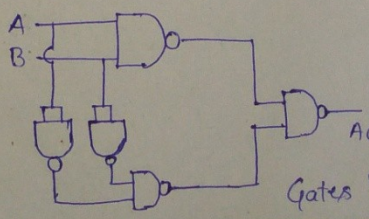
Min. no. of NOR gates Reqd to build XOR gate = 5//.

Ex-NOR (using NOR)



Min. no. of NOR gates required to construct XNOR is 4

Ex-NOR (USING NAND)



Gates Reqd = 5//.

20. FUNCTIONALLY COMPLETENESS

⇒ A set of operations is said to be functionally complete (or) universal if and only if every switching function can be expressed by means of operations in it.

→ The set $\{+, \cdot, -\}$ is clearly functionally complete

→ The set $\{+, -\}$ is said to be functionally complete. $(A \cdot B = \overline{(\bar{A} + \bar{B})})$

→ The set $\{\cdot, -\}$ is also functionally complete $(A + B = \overline{(\bar{A} \cdot \bar{B})})$

→ NOTE : A set is said to be functionally complete if we can derive a set which is already functionally complete.

21. EXAMPLE 1 ON FUNCTIONAL COMPLETENESS

1> $F(A,B,C) = A' + BC'$ is this functionally complete?

For it to be functionally complete it should definitely derive $-$ (NOT) and $(+$ or $\cdot)$

∴ $(-, \cdot)$ $(-, +)$

⇒ $f(A,B,C) = A' + BC'$

⇒ Now check $f(A,A,A) = A' + A'A' = A'$ ∴ compliment can be realised.

⇒ $f(\underbrace{f(A,A,A)}_{A'}, B, \underbrace{f(B,B,B)}_{B'}) \Rightarrow f(A', B, B') = (A')' + B(B')'$

$\boxed{f(A', B, B') = A + B}$

22. EXAMPLE 2 ON FUNCTIONAL COMPLETENESS

1> $F(A,B) = (\bar{A} + B)$ is this functionally complete?

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

$F(A,A) = \bar{A} + A = 1$
$F(B,B) = \bar{B} + B = 1$ } No way we can get rid of a variable

$F(A,0) = \bar{A} + 0$
$\therefore = \bar{A}$

Now, $F(f(A,0), B)$
$= f(\bar{A}, B) = A + B.$ } using '0' we can make it complete so when we take support from '0' and '1' then we call it partially complete.

if and only if

which

23. EXAMPLE 3,4,5,6 ON FUNCTIONAL COMPLETENESS

1> $F(A,B) = \bar{A}B$

$f(A,A) = \bar{A}\cdot A = 0$.

$f(B,B) = \bar{B}B = 0$.

$f(A,1) = \bar{A}$ (complementation Acheived).

$f(f(A,1),B) = (\bar{A})B = AB$.

} partially functionally complete.

2> $f(A,B,C) = AB+BC+CA$

$f(A,A,A) = A+A+A$

$f(A,A,A) = A$

} complementation cannot be acheived (Not functionally complete)

⇒ If a function doesnot contain compliment then it cannot be functionally compliment.

3> $f(x,y) = \bar{x}y + x\bar{y}$

$f(x,x) = \bar{x}x + x\bar{x}$

$= 0+0 = 0$.

$f(y,y) = \bar{y}y + y\bar{y}$

$= 0+0 = 0$.

$f(x,1) = \bar{x}1 + x(0)$

$\boxed{f(x,1) = \bar{x}}$ I'm getting compliment with '1'

⇒ partially functionally complete

$f(x',y) = xy + \bar{x}\bar{y}$

$f(x,\bar{y}) = \bar{x}\bar{y} + xy$

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

$\{\cdot,-\}$ $\{+,-\}$

cannot be acheived

∴ Not functionally complete.

4> Any Boolean function can be defined with which of the following operations

a) $\oplus$, NOT b) $\oplus$, 1, OR c) $\oplus$, 1, NOT d) $\odot$, 1, NOT

a) $\oplus$ doesnot lead to $\{+,\cdot\}$ check above example

b) partially complete → Functionally complete.

27. SELF DUAL FUNCTIONS

1> $f(A,B,C) = AB+BC+CA$

2> $f_d(A,B,C) = (A+B)(B+C)(C+A)$

$= AB+BC+CA$

2> $f(A,B,C) = AB(C+C') + (A+A')BC + C(B+B')A$

$= ABC + ABC' + A'BC + AB'C$

* Boolean function is self dual if

1> It is Neutral (No of minterms = No of maxterms)

2> The function doesnot contain two mutually exclusive terms.

$(ABC) \rightarrow (A'B'C')$

$(AB'C') \Rightarrow (A'BC)$ } mutually Exclusive

28. NO OF SELF DUAL FUNCTIONS

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

(0,7) (1,6) (2,5) (3,4) ⇒ ME pairs

$F = 2 \times 2 \times 2 \times 2 = 2^4 = 16$

(0 1 2 3) = self dual function.

∴ No of self dual functions = $2^{(2^{n-1})}$

∴ n variables → 2^n terms

↓ divide into pairs (having two in each set)

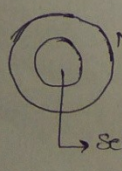
$2^n/2 = 2^{n-1}$ pairs

↓

Each pair has two choices (include 1st ele or 2nd ele)

∴ $2^{(2^{n-1})}$

∴ No of self dual functions = $\left[2^{(2^{n-1})}\right]$



29. SELF DUAL FUNCTIONS ARE CLOSED UNDER COMPLEMENTATION

Which of the following functions are closed under complementation?

1) $f(A,B,C) = \Sigma(0,2,3)$ = No of minterms ≠ No of maxterms ⇒ Not self dual

2) $f(A,B,C) = \Sigma(0,1,6,7) = (0,7)(1,6)(2,5)$ ⇒ Neutral but not self dual (contains both ele of mutual exclusive pair)

3) $f(A,B,C) = \Sigma(0,1,2,4) = (0,7)(1,6)(2,5)(3,4) \Rightarrow$ No of minterms = No of maxterms & ME pairs. are unique (SD) ✓

4) $f(A,B,C) = \Sigma(3,5,6,7) = (3,4)(5,2)(6,1)(0,7) \Rightarrow$ ME(✓) N(min) = N(max) = SD ✓

⇒ self dual is closed under complementation (compliment of selfdual function is selfdual)

30. INTRODUCTION TO ELECTRONIC GATES

1) Electronic gates generally receive voltages as inputs and produce voltages as outputs.

2) The precise values of these voltages are not significant towards determination of logical operation of gates.

3) The significant point is that voltages are restricted to two ranges of values high and low.

4) Thus two valued variables may be used to represent these voltages

5) If we associate constant 1 with high voltage and '0' with low voltage, it is called positive logic system.

6) If we associate the constant 1 with low voltage and '0' with high voltage then it is called Negative logic system.

2. MINIMISATION

1. INTRODUCTION TO MINIMISATION OF BOOLEAN EXPRESSIONS

→ A switching function can usually be represented by using a no. of expressions

Ex: $f(x,y,z) = xy + yz + zx$

$= xyz + xyz' + xyz' + xyz$

→ while simplifying a switching function $f(x_1, x_2 \cdots x_n)$ our aim is to find an expression $g(x_1, x_2 \cdots x_n)$ which is equivalent to 'f', which minimises some cost criteria

Criteria to determine minimal cost:

1> Minimum no. of appearences of literals

2> Min no. of literals in sop or pos expression

3> Minimum no. of terms in Sop expression, provided there is no other such expression with the same no. of terms and fewer literals.

2. IRREDUNDANT OR IRREDUCABLE EXPRESSIONS

$F(x,y,z) = x'yz' + x'y'z' + xy'z' + x'yz + xyz + xyz'$

$= x'z'(y+y') + xy'z' + yz(x+x') + xy'z$

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

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

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

$F(x,y,z) = x'z' + y'z' + yz + xz$

$x'z'(y+y') + yz(x+x') + xy'(z+z')$

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

⇒ Minimal is not always unique

→ An Sop expression from which no term or literal can be deleted without altering its logical value is called an irredundant or ireducable expression.

→ Note: An irredundant expression is not necessarily be minimal, nor the minimal expression always unique

3. K-MAP INTRODUCTION

→ The Algebraic procedure of combining various terms and applying to them the rules becomes very tidious as the no. of various variables increases.

→ The map method provides a systematic method for combining the terms and derive minimal expression.

→ A K-map is a modified form of the truth table in which the arrangement of combinations is particularly convinient for minimising

→ Every nr variable map consists of 2^n "cells", srepre senting all possible combinations of variables.

x \ y	0	1
0	0	1
1	2	3

2v- K-map.

z \ xy	00	01	11	10
0	0	2	6	4
1	1	3	7	5

3v- Kmap

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

4v-K-map

4. K-MAP SIMPLIFICATION

1> A collection of 2^m cells, each adjacent to 'm' cells of collection is called a subcube, and the subcube is said to cover these cells.

2> Each subcube can be expressed by a product containing n-m literals where 'n' = no.of variables on which function depends.

3> Any cell may be included in as many subcubes as desired

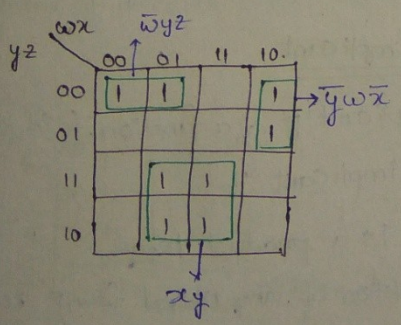
4> A function 'f' can be expressed as sum of those product terms which corresponds to subcubes necessary to cover all its '1' cells.

5> The no.of product terms in the expression for 'f' is equal to the no.of subcubes while the no.of literals in each term in determined by size of corresponding subcubes

6> Therfore to obtain a minimal expression we must cover all '1' cells with smallest no.of subcube is as long as possible

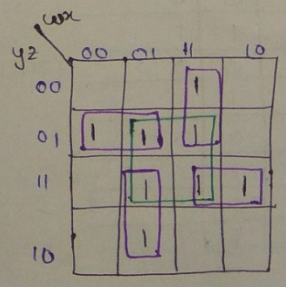
5. EXAMPLES ON K-MAP

Minimise $f(w,x,y,z) = \Sigma(0,4,6,7,8,9,15)$



$\therefore F = \bar{w}yz + \bar{y}w\bar{x} + xy$

(2) $F(w,x,y,z) = \Sigma(1,5,6,7,11,12,13,15)$



$F = \bar{w}\bar{y}z + wx\bar{y} + \bar{w}xy + wyz$

~~+xyz~~

Not Required.

6. COVERING FUNCTIONS

If the minterms present in 'g' are also present in 'f' then 'f' is said to cover 'g'.

Implicants:

A switching function $f(x_1, x_2 \cdots x_n)$ is said to be cover of $g(x_1, x_2 \cdots x_n)$ denoted by "f is superset of g" if 'f' assumes true value whenever 'g' does.

Note: If 'f' covers 'g' and 'g' covers 'f' then both 'f' and 'g' are equivalent.

⇒ If 'g' has x min terms and 'g' is a function of 'n' variables, then no. of covering functions of $g = 2^{(2^n - x)}$

Ex: $f(w,x,y,z) = wx + yz$ how many functions are covering f'

⇒ NO. of min terms possible = $2^4 = 16$ minterms

⇒ 7 minterms are choosen Remaining = 9 minterms

⇒ NO. of covering functions = $2^9 - 2^9 = 512$

wx — — + — — yz

0 0		0 0		
0 1		0 1	= NO. of minterms = 7	
1 0		1 0		
[1 1]		[1 1]	NO. of remaining terms = 16 − 7 = 9.	

7. IMPLICANTS AND PRIME IMPLICANTS

↦ If 'f' covers 'g' then 'g' is said to imply 'f' This denoted by "g → f"

Ex: $f(a,b,c) = ab + c$

a	b	c	f = ab + c
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Both ab, c are implicants of 'f'.

prime Implicant

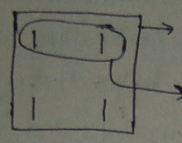
⇒ An implicant 'p' of a function 'f' is said to be prime implicant if

i) 'p' is a product term

ii) Deletion of any literal from 'p' results in a new product which is not covered by 'f' (i.e. A subcube cannot be part of any other subcube)

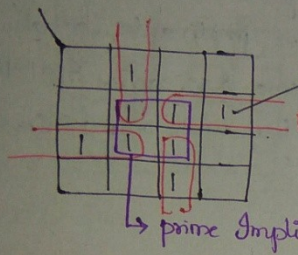
8. ESSENTIAL PRIME IMPLICANTS

A prime implicant 'p' of a function 'f' is said to be an essential prime implicant if it covers atleast one minterm of 'f' which is not covered by any other prime implicants.



→ Implicant (prime implicant)

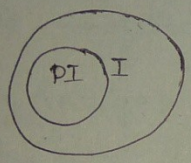
→ Implicant (Not prime implicant because it is completely covered by another implicant.



→ This one cannot be covered by any other any other Implicant

Essential prime Implicants (If cannot be covered by any other Implicant).

→ prime Implicant



I = Implicants

PI = prime Implicant

9. PROCEDURE FOR OBTAINING MINIMAL SOP

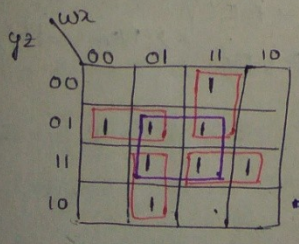
1) Determine all essential prime implicants and include them in minimal sop.

2) Remove from list of prime implicants all those which are covered by essential prime implicants.

3) If the set determined in step1 covers all the minterms of 'f', then it is unique minimal expression, otherwise select the additional prime implicants so that the function 'f' is covered completely and the total number and size of prime implicants added are minimal.

Ex

$$F = \Sigma(1,5,6,7,11,12,13,15)$$



$f = \bar{w}\bar{y}z + wx\bar{y} + wyz + \bar{w}xy$ (Unique minimal expression)

⇒ Now, to check whether the above ~~minimal~~ expression is minimal or not then we go for Tabulation method.

	1	5	6	7	11	12	13	15
* $\bar{w}\bar{y}z$	1	1						
* $wx\bar{y}$						1	1	
* wyz					1			1
* $\bar{w}xy$			1	1				
xz		1		1			1	1

⇒ Now, $\bar{w}\bar{y}z$ covers the boxes 1,5 in K-map. so put 1,1 in 1,5 column and $\bar{w}\bar{y}z$ row, and mark 1,5 (meanwhile these are covered)

⇒ similarly do for all minterms implicants (Rows).

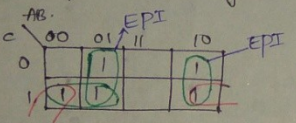
⇒ Now check 1 minterm (1 column) (column numbered 1.) it is covered only with $\bar{w}yz$ so it is a prime implicant that is essential so put * on $\bar{w}\bar{y}z$ and it also covers 5

⇒ Similarly the min term '6' is covered by $\bar{w}xy$ so $\bar{w}xy$ is the essential prime implicant

⇒ Similarly analyze each column and mark the essential prime implicants and also marks the minterms that are also covered by the prime implicants, if all the minterms are covered then check *'s on the Rows, and they will form Minimal expression.

10. MINIMAL SOP EXAMPLE

Which of the following prime implicants are essential?



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

a) $B'c$, $A'B$.

b) $A'c$, $A'B$.

c) $A'B$, AB' ✓

d) $A'B$, AB', $B'c$

Now, if they have asked how many minimal expressions are there then how to find is.

	1	2	3	4	5
* $A'B$		(1)	1		
* AB'				(1)	1
$A'c$	(1)		1		
$B'c$	(1)				1

Equally probable candidates.

$\therefore F = \bar{A}B + A\bar{B} + \bar{A}c$

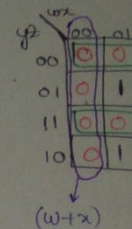
$F = \bar{A}B + A\bar{B} + B\bar{c}$

} 2 Minimal Expressions

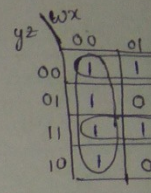
11. MINIMAL POS

Continued in Nextpage

① $F(w,x,y,z) =$



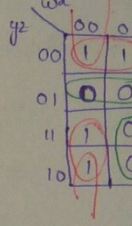
②



12. EXAMPLES

Find the mo

$f(w,x$



13. INTRODUCT

→ A function is
of variables.

→ Combinations
Care combi

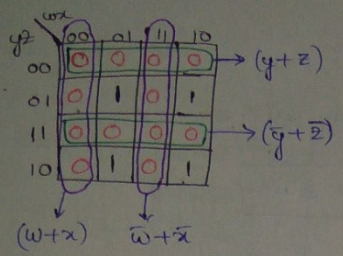
→ The value o

→ since each c
specified fu
of 2^K di

→ our task is t

→ we could as
a unuto in

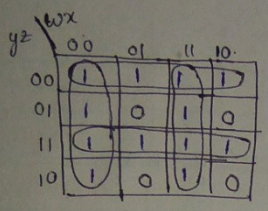
① $F(w,x,y,z) = \Sigma(5,6,9,10)$



$\therefore F = (y+z)(w+x)(\bar{y}+\bar{z})(\bar{w}+\bar{x}) = pos.$

$= \bar{w}x\bar{y}z + w\bar{x}\bar{y}z + \bar{w}\bar{x}y\bar{z} + w\bar{x}y\bar{z} = sop$

②



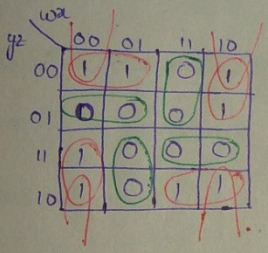
$Sop = \bar{y}\bar{z} + \bar{w}\bar{x} + wx + yz$

$pos = (w+\bar{x}+y+\bar{z})(\bar{w}+x+y+\bar{z})(w+\bar{x}+\bar{y}+z)(\bar{w}+x+\bar{y}+z)$

12. EXAMPLES ON MINIMAL pos.

Find the no. of literals in minimum pos and sop for

$f(w,x,y,z) = \pi(1,5,6,7,11,12,13,15)$



$pos = (\bar{w}+\bar{x}+y)*(w+y+\bar{z})(\bar{w}+\bar{y}+\bar{z})(w+\bar{x}+\bar{y})$

$Sop = w\bar{x}\bar{y} + wy\bar{z} + \bar{w}\bar{x}y + \bar{w}\bar{y}\bar{z}$

13. INTRODUCTION TO DONT CARES

→ A function is said to be completely specified if it is given '0' or '1' for every combination of variables. There exist some function which are not completely specified.

→ Combinations for which the value of a function is not specified are called dont care combinations.

→ The value of a function for such combination is denoted by 'ϕ' or 'd'.

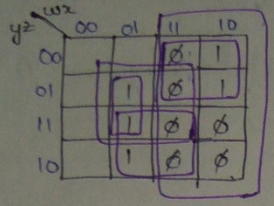
→ Since each dont care combination represents 2 values {0,1} an incompletely specified function containing k-dont care combinations corresponds to a class of 2^k distinct functions

→ our task is to choose a function having minimal representation out of these 2^k functions.

→ we could assign a '0' or '1' to a '0' or '1' to a dont care combination in such a way to increase or decrease size of a subcube.

14. EXAMPLES ON DONT CARES -1

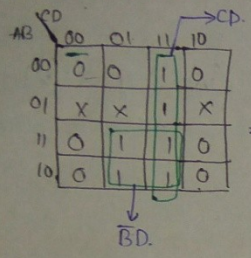
$W = \Sigma(5,6,7,8) + \phi(10,11,12,13,14,15)$



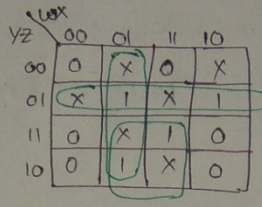
Now, Sop = $wx + wy + xy + z\bar{w}x$

Sop = $w + xy + z\bar{w}x$

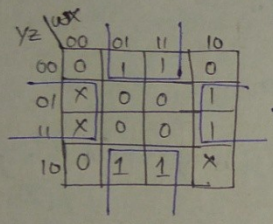
15. EXAMPLES ON DONT CARE SET-2



$= CD + \bar{B}D$

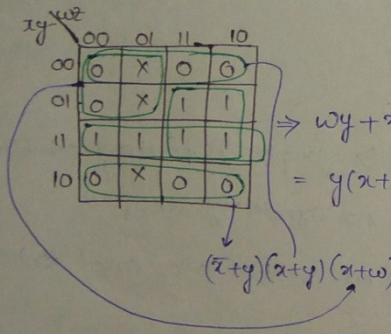


$= \bar{y}z + xy$



$x\bar{z} + \bar{x}z$

16. EXAMPLES ON DONT CARE SET 3



$\Rightarrow wy + xy$

$= y(x+w)$

$(\bar{x}+y)(x+y)(x+w)$

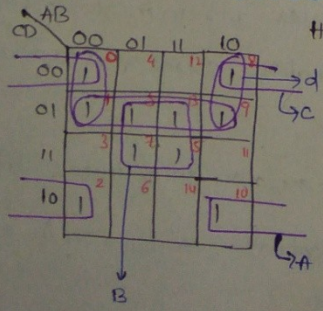
(A) $(w+x)y$

(B) $xy + yw$

(C) $(w+x)(\bar{w}+y)(\bar{x}+y)$

(D) None

17. FINDING MINIMAL EXPRESSIONS

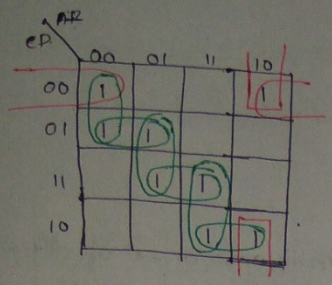


How many minimal expressions are possible?

	0	1	2	5	7	8	9	10	13	15
*A	1		1			1		1		
*B				1	1				1	1
C		(1)		1			(1)		1	
D	1	(1)				1	(1)			

∴ No. of minimal expressions = 2×2=4

18. BRANCHING TECHNIQUE FOR MINIMISING CYCLIC FUNCTIONS



	0	1	5	7	8	10	14	15
A = $w x' y'$	X	X						
B = $w y' z'$			X					
C = $w' x z$			X	X				
D = $x y z$				X				X
E = $w x y$							X	X
F = $w y z'$						X	X	
G = $w x z'$					X	X		
H = $x' y' z'$	X				X			

⇒ Now, if i choose 'A' and include it in minimal expression then A (included in ME)

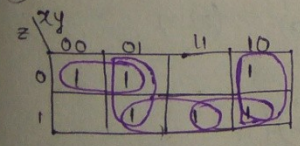
⇒ Now, 0,1 minterms are covered by 'A' so delete 'A' row and 0,1 columns

⇒ Now, we can observe B → C (B is covered by C ⇒ B is irredundant) and G → H

⇒ (G is covering by H ⇒ H is irredundant) so delete, B, H rows and continue the procedure and mark the essential prime implicants.

⇒ ∴ The minimal ans will be A + C + G + E

(2)



	0	2	3	4	5	7
A = $x z'$	X	X				
B = $x' y$		X	X			
C = $y z$			X			X
D = $x z$					X	X
E = $x y'$				X	X	
F = $y' z'$	X			X		

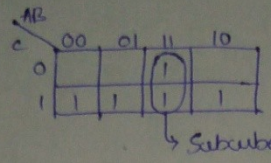
⇒ Let me choose A + C + E

B → C) Because F → E ⇒ F is not needed.

19. IMPLICANT AND PRIME IMPLICANT DIFFERENCE

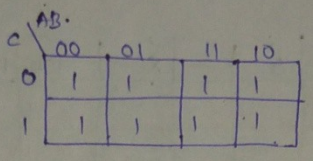
Which of the following can be a prime implicant of a function f(a,b,c)

a) ab+c
b) bc+a
c) ac+b
d) ab.



② which of the following cannot represent prime implicants onset of f(A,B,C)

a) {ab, bc, ca}
b) {bca, ba, bc}
bac', ba
c) {a', b, ac}
d) {a'b, ab}.



{ab, bc, ca}
↓
If ab is PI then anything containing 'ab' should not be present

20. CONVERTING A FUNCTION INTO SELF DUAL

What minterms have to be added to make the following funtion a self dual

f(A,B,C,D) = A'BC + (AC+B)D.

= A'BC + ACD + BD. ⇒ Now the Mutually exclusive terms for this are A'B'C' + A'C'D' + B'D' these minterm should not be added.

f(A,B,C,D) = A'BC(D+D') + ACD(B+B') + BD(A+A')*(C+C')

= A'BCD + A'BCD' + ABCD + AB'CD + [ABD + A'BD](C+C')

= A'BCD + A'BCD' + ABCD + AB'CD + ABCD + A'BCD + ABC'D + A'BC'D

= A'BCD + [redacted] + ABCD + AB'CD + [redacted] + A'BC'D + ABC'D

(A'BCD')

= 7 6 15 11 5 13

(7,8) (6,9) (0,15) (4,11) (5,10) (2,13)

(0,15) (1,14) (2,13) (4,11)
(5,10) (6,9) (7,8) (3,12)

The left out pairs are: (1,14) (3,12)

↓ ① minterm from this. → 1 minterm from this
= 2 choices 2 choices

= 4 minterms can be added.

21. COMB

How many

$f_1(a,b,c)$ =

$f_2(a,b,c)$

Ans

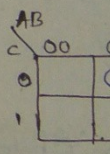
1).

	A	f_1
000 =	0	1
001 =	1	0
010	2	1
011 =	3	∅
100 =	4	1
101 =	5	∅
110	6	0
111 =	7	∅

22. PRIM

What are

prime



A function

23. NUMBE

In a k-

covering a

by 3- no

21. COMBINING FUNCTIONS HAVING DONT CARE

How many functions does $f_1 \cdot f_2$ and $f_1 + f_2$ Represent?

$f_1(a,b,c) = \Sigma(0,2,4) + \phi(3,5,7) = \Sigma(0,2,4) + \Sigma_\phi(3,5,7)$.

$f_2(a,b,c) = \Sigma(2,3) + \Sigma_\phi(1,6,7)$

Ans

1).

A	f_1	f_2	$f_1 \cdot f_2$	$f_1 + f_2$
000 = 0	1	0	0	1
001 = 1	0	ϕ	0	ϕ
010 2	1	1	1	1
011 = 3	ϕ	1	ϕ	1
100 = 4	1	0	0	1
101 = 5	ϕ	0	0	ϕ
110 6	0	ϕ	0	ϕ
111 = 7	ϕ	ϕ	ϕ	ϕ

4 Dontcares ⇒ $f_1 + f_2 = 2^4$ functions

2 Dont cares(ϕ, ϕ)

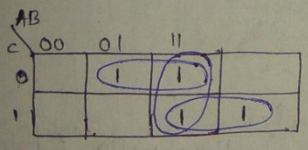
⇒ can take 2 values (0,1) can take two values (0,1)

$= 2 \times 2$

$= 4$ ∴ $f_1 \cdot f_2 = 4$ functions

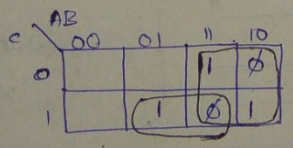
22. PRIME IMPLICANTS AND DONT CARES

What are the no. of prime Implicants, Essential prime Implicants and Redundant prime Implicants for the function $f(A,B,C) = \Sigma(2,5,6,7)$.



No of Implicants = 3.
Essential = 2
Redundant = 1

A function $f(A,B,C) = \Sigma(3,5,6)$ is minimised to $(A + BC)$ then what are the dont cares?



$= \Sigma_\phi(4,7)$ are the Dontcares.

23. NUMBER OF MINIMAL EXPRESSIONS

In a K-map it was found out that all the essential prime Implicants are covering all terms except 2 min terms. Those 2 minterms are in turn covered by 3- non-essential prime implicants each. what is the no. of minimal sop expressions?

The soln is EPI + ___ + ___
3 choices, 3 choices

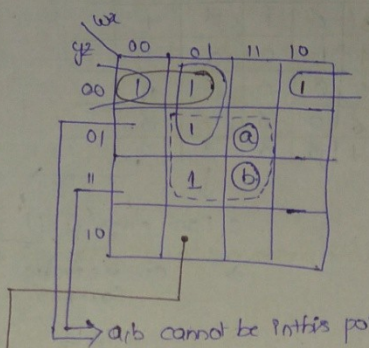
∴ 9 minimal expressions are possible

24. BEAUTIFUL QUESTION ON PRIME IMPLICANT CHART

→ Sh a prime implicant chart representing boolean expression $f(w,x,y,z)$ columns represent minterms and rows represent PI's, Identify P,Q,R,S and a,b?

		0	4	5	7	8	a	b.
$\bar{w}\bar{y}\bar{z}$ =	P	✓	✓					
$\bar{x}\bar{y}\bar{z}$ = *	Q	✓				✓		
$\bar{w}x\bar{y}$ =	R		✓	✓				
xz = *	S			✓	✓		✓	✓

∴ a=13 b=15



a/b cannot be in this positions because, if they were present 'P' would not become a prime implicant

a/b cannot be present here because, if they are present then the should cover the minterm ④ but if we check 'S' it does not cover the minterm 4.

25. VARIABLE ENTRANT MAP

$f(A,B,C) = \Sigma(1,3,5,7)$

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

So for the given values of A,B the function 'f' behaves exactly as 'c'

A	B	f
0	0	c
0	1	c
1	0	c
1	1	c

} = variable Entrant map.

A=0, B=0 ⇒ f=c
A=0, B=1 ⇒ f=c'
A=1 ↑ B=0 ⇒ f=c'

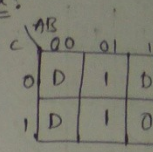
similarly for $f = \Sigma(1,2,5,6)$ = behaves as C and C'
A=1, B=1, f=c

$f = \Sigma(1,2,3,5,6)$ = behaves as (c, 1, c, c')
A=0, B=0 ; A=0 B=1 ; A=1 B=0 ; A=1 B=1

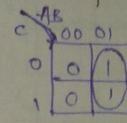
26. MINIMIS

1> set all the
2> (a) Make one
minterms (
(b) Multiply
3> Repeat the
4> Sop of VE

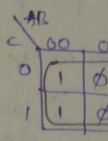
Ex:



Step-1: Repla



Step-2: Repl
minterms (c



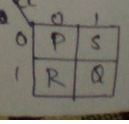
Step-2(b): Nou
and 1's s

Step-3: A'B

f = A'B

27. EXAMPLE

f(A,B,C) = Σ



26. MINIMISATION USING VEM

1> set all the variables in the cell as '0' and obtain sop expression

2> (a) Make one variable in the cell as '1' and obtain sop by making earlier minterms (1's) as dont cares

(b) Multiply the above 'sop' with the concerned variable

3> Repeat the step'2' uhtill all the variables in the cell are covered

4> Sop of VEM is obtained by OR ing the previous sop expressions.

Ex:

C \ AB	00	01	11	10
0	D	1	D'	D'
1	D	1	0	∅

→ If a K-map is used then it contains 4×4 = 16 entries

→ It is 4-variable map and hence the function behaves as one of the variable for a given set of other variables Here 'f' behaves as D/D' for given set of inputs ABCD

Step-1: Replace the variables with 0's and obtain sop expression.

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

sop = A'B.

Step-2: Replace any one of the variables D or D' with 1's and replace the minterms (cells having ones) with Dontcares (∅); (D) replaced.

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

sop = $\overline{A}1$.

Step-2(b): Now you have replaced 'D' in step 2(a) now replace D' with 1's and and 1's should be made as Dontcares and remaining cells as '0's

(D') replaced

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

= AC'

Step-3: A'B + D'A + AC'D'

$$F = A'B + A'D + AC'D'$$

27. EXAMPLE ON VEM

f(A,B,C) = Σ(3,5,6,7) is realised by following VEM, then find P,Q,R,S.

B \ A	0	1
0	P	S
1	R	Q

a) P=0, R=S=C, Q=1

b) P=Q=0, R=C, S=1

c) P=0, Q=R=C, S=1

Given $f(A,B,C) = \Sigma(3,5,6,7)$

Now,

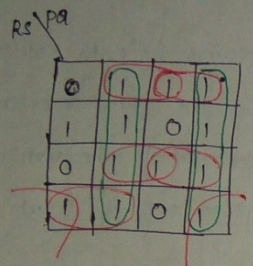
	A	B	C	f
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

Now,

B \ A	0	1
0	0	C
1	C	1

P = 0
R = C
S = C
Q = 1

28. PROBLEM ON K-MAP



10 - prime implicants.

12 - Minterms.

How many Minimal functions are possible?

	1	2	4	5	6	7	8	9	10	11	12	15
$\bar{P}\bar{Q}$			✓	✓	✓	✓						
$P\bar{Q}$							✓	✓	✓	✓		
$Q\bar{R}\bar{S}$			✓								✓	
$P\bar{R}\bar{S}$							✓				✓	
$\bar{P}RS$	✓			✓								
$\bar{Q}RS$	✓							✓				
QRS					✓							✓
PRS										✓		✓
$Q\bar{R}\bar{S}$		✓							✓			
$\bar{P}R\bar{S}$		✓		✓								

Now, if in include $\Rightarrow \bar{P}\bar{Q}$ then (4,5,6,7) will be covered

if $\Rightarrow P\bar{Q}$ included then (8,9,10,11) will be covered.

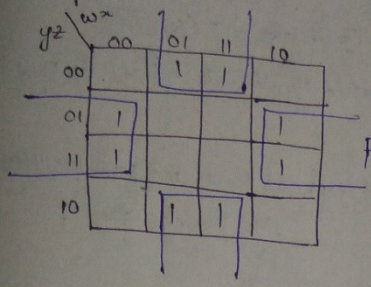
Now,

- $Q\bar{R}\bar{S}$, $P\bar{R}\bar{S}$ → 12 is covered by these → (2) ways (either $Q\bar{R}\bar{S}$ / $P\bar{R}\bar{S}$ can be used to generate '2').
- $\bar{P}RS$, $Q\bar{R}\bar{S}$ → 1 can be covered by these → (2) ways. These are
- QRS, PRS → 15 can be covered by these → 2 ways
- $\bar{Q}RS$, $\bar{P}R\bar{S}$ → 2 ways. → 'Σ' can be covered by these

$\Rightarrow 2^4 = \underline{16}$ functions are possible

29. FINDING FREE VARIABLES

How many variables are free in the expression denoted by the following K-map?

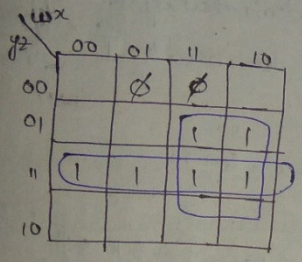


$F \Rightarrow \bar{x}z + x\bar{z}$

$F = \bar{x}z + x\bar{z}$ sop ∴ (w, y) are not Required.

pos = $(x+\bar{z})(\bar{x}+z)$.

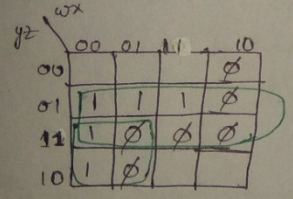
30. RELATIONSHIP IN BETWEEN MINIMAL POS, SOP IN CASE OF DONT CARES--1



$F = (yz + wz)$

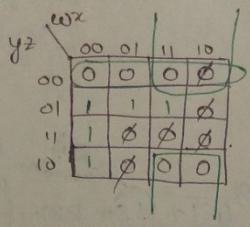
31. RELATIONSHIP IN BETWEEN MINIMAL POS SOP IN CASE OF DONT CARES --2

Let $t = (w,x,y,z) = \Sigma(1,2,3,5,13) + \Sigma_{\phi}(6,7,8,9,11,15)$ and let 'f'(w, x, y, z) be minimal sop and g(w, x, y, z) be the minimal pos. Are 'f' and 'g' identical.



$F = z + yw$

$F = \Sigma(1,2,3,5,6,7,9,11,13,15)$



$g = \Sigma(1,2,3,5,13,19)$

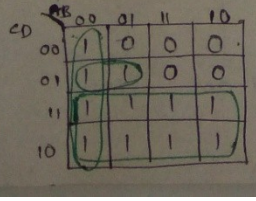
(Here 2^6 functions are possible (6 Dont care and each dontcare has 2 choices ∴ 2^6 functions are possible).

∴ 'F' and 'G' are not identical always.//

32. COMPARING INDEPENDENT VARIABLES IN MINIMAL SOP AND POS.

Minimal expression represented by K-map is free form.

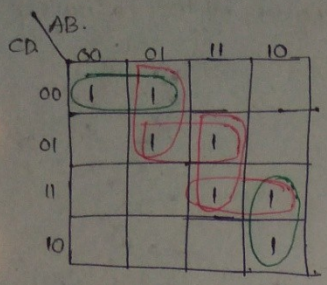
a) 1-variable b) 2-variables c) 3 variables d) Dependent on all.



$C + A'B' + A'C'D$.

∴ Dependent on all.

33. NO·OF IRREDUNDANT AND MINIMAL EXPRESSION



1) = No·of prime Implicants = 6 (Both Red + Green)

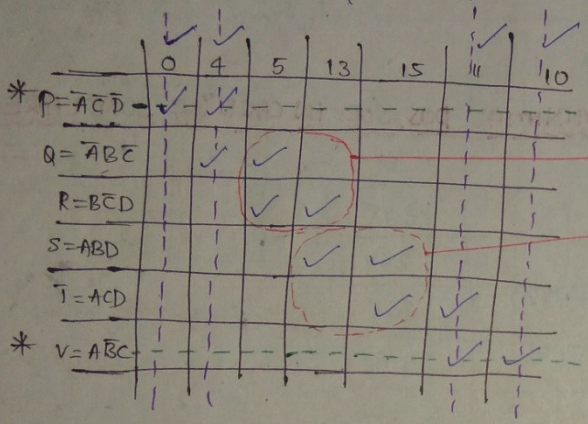
2) = No·of essential prime Implicants = 2 (Green)

3) = No·of Redundant prime Implicants = (6-2) = 4 (Red)

4) = Minal sop?

5) How many minimal expressions are possible that are irredundant?

Now, To answer 4,5 construct prime Implicant chart



Q is dominated by (R)

'T' is dominated by (S)

(Q → R) waste = (T → S)

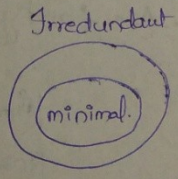
P + V + R + S

∴ Minimal pos = P+V+R+S.

5). No·of Minimal sop's possible are, 5 → 2 ways (Q+T)

13 → 2 ways (R+S)

15 → 2 ways (S+T)



∴ (Q+T)(R+S)(S+T)

= QRS + QRT + QSS + QST + RRS + RRT + RSS + RST

(RS is coveredin RS)

= QRS + QRT + QS + QST + RT + RS + RST

= QS + RT + RS.

∴ P+V+Q+S
P+V+R+T
P+V+R+S.

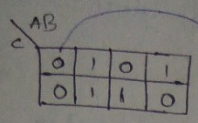
No·of mE = No·of Irredundant mE = 3.
(In this case)

⇒ Don't cares are not included in prime implicant chart.

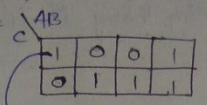
37. FUNCTIONS INVOLVING FUNCTIONS EXAMPLE 2

Consider a new boolean operation '$' defined as $A \$ B = A' + B$, then find $f_1 \$ f_2$

$f_1 =$



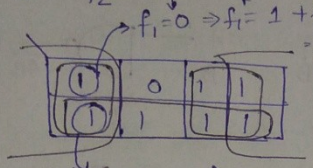
$f_2 =$



Now, $f_1 \$ f_2 = f_1' + f_2$

$f_1 = 0 \Rightarrow f_1' = 1 + f_2$

$= 1 + 1 = 1.$



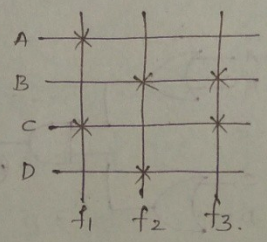
$f_1 = 0 \Rightarrow f_1' = 1 + 0$

$f_1 \$ f_2 = 1.$ //

(Q → R)
waste
(T → S.)

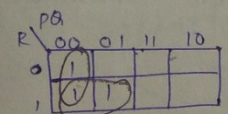
38. FUNCTIONS INVOLVING FUNCTIONS - EXAMPLE-3

Consider three- three variable functions $f_1(P,Q,R) = \Sigma(0,1,3)$, $f_2(P,Q,R) = \Sigma(3,5,7)$ $f_3(P,Q,R) = \Sigma(1,3,7)$. These functions are sharing 4 prime implicants A,B,C,D (all of them are pairs) as shown.



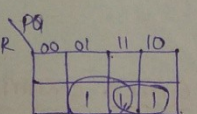
find A,B,C,D?

Now, $f_1 =$



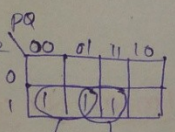
$f_1 = P'Q' + R\bar{P}$

$f_2 :$

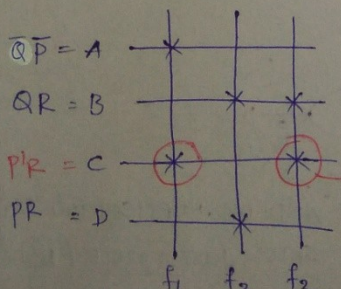


$f_2 : QR + PR$

$f_3 :$



$f_3 : P'R + QR$

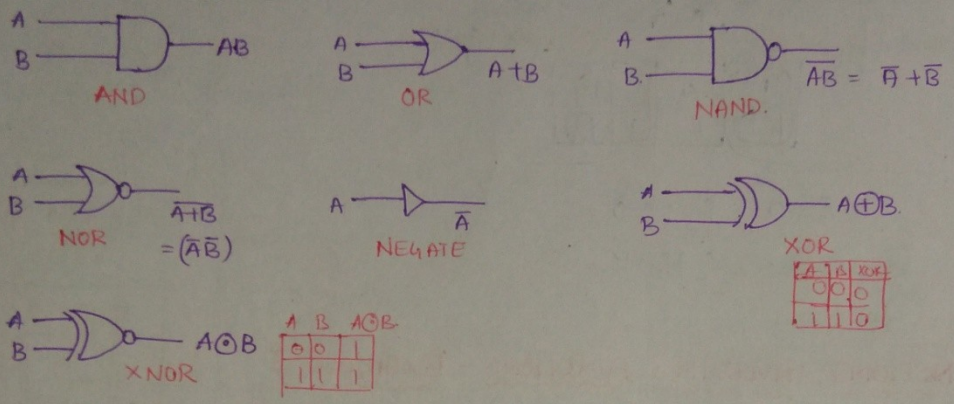


3. DESIGN AND SYNTHESIS OF COMBINATIONAL CIRCUITS

1. INTRODUCTION TO LOGIC DESIGN

→ The main application of switching theory is in the design of digital circuits. (called logic design)

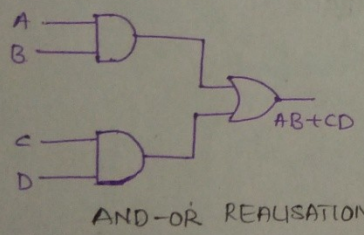
→ The circuits are designed using the basic elements called gates.



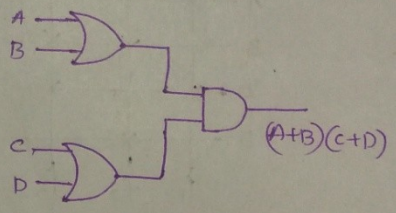
2. AND-OR OR-AND REALISATION

SOP: $f(A,B,C,D) = AB+CD$.

POS: $f(A,B,C,D) = (A+B)(C+D)$



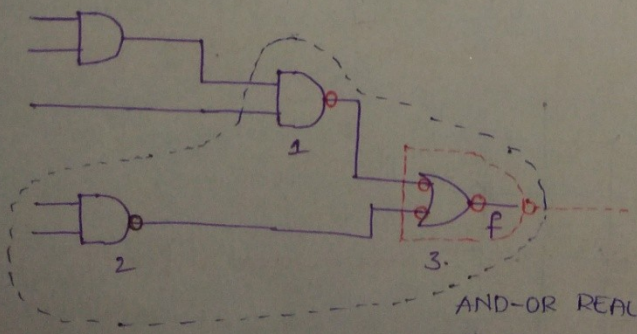
AND-OR REALISATION



OR-AND REALISATION

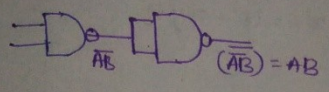
3. MINIMUM NO OF NAND GATES EXAMPLE

Identify min no. of two i/p NAND gates required to represent the following



AND-OR REALISATION. TO MAKE IT NAND GATE JUST PLACE BUBBLE AS SHOWN

Now, $AB = \overline{(\overline{AB})}$

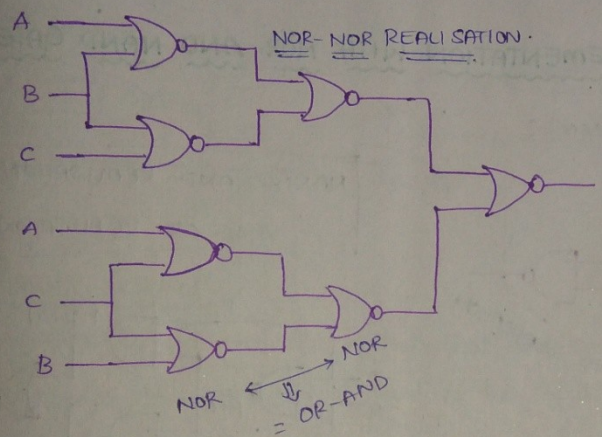


= 2 NAND GATES.

∴ Total no of NAND GATES Required = 5.

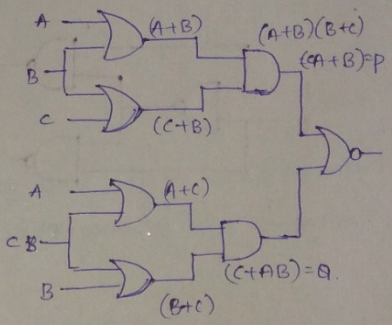
4. NOR- NOR EXAMPLE

What does the following function Represent?



NOR- NOR REALISATION = OR- AND REALISATION

⇓



⇒ Now. output will be $\overline{P+Q}$

$= \overline{P}\,\overline{Q}$

∴ $\overline{P}\overline{Q} = (\overline{B+AC})(\overline{C+AB})$

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

$\overline{P}\overline{Q} = \overline{B}\,\overline{A} + \overline{B}\,\overline{C} + \overline{C}\,\overline{A} + \overline{C}\,\overline{B}$

$\overline{P}\overline{Q} = \overline{B}[\overline{A}+\overline{C}]\,\overline{C}(\overline{A}+\overline{B}) = (\overline{A}\overline{C}+\overline{C})(\overline{A}\overline{B}+\overline{B})$

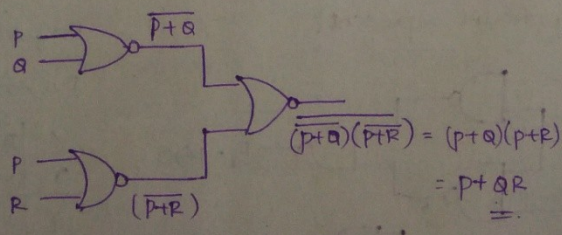
$\boxed{\overline{P}\overline{Q} = \overline{C}+\overline{B}.}$

5. MINIMUM NO-OF NOR GATES EXAMPLE

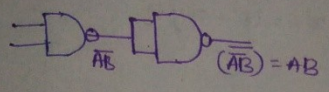
Find the min no of 2 i/p NOR gates required to represent $F(P,Q,R) = P+QR$

$F(P,Q,R) = (P+Q)(P+R)$

$= \overline{(\overline{P+Q})(\overline{P+R})}$



Now, $AB = \overline{(\overline{AB})}$

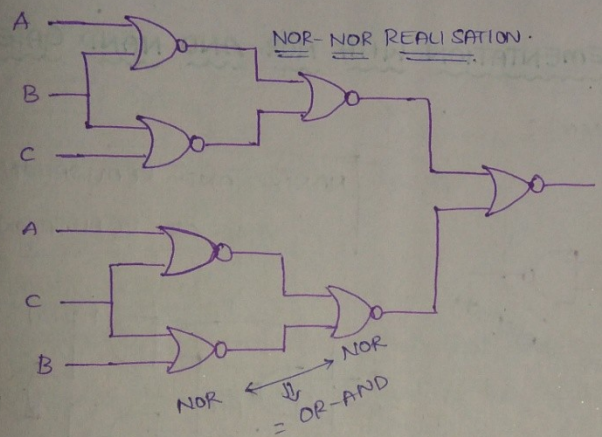


= 2 NAND GATES.

∴ Total no of NAND GATES Required = 5.

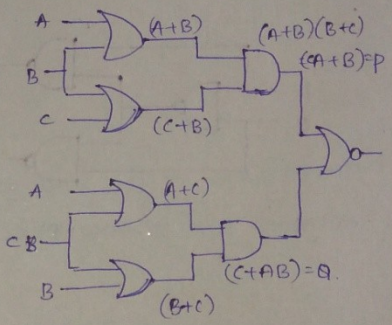
4. NOR- NOR EXAMPLE

What does the following function Represent?



NOR- NOR REALISATION = OR- AND REALISATION

⇓



⇒ Now. output will be $\overline{P+Q}$

$= \overline{P}\,\overline{Q}$

∴ $\overline{P}\overline{Q} = (\overline{B+AC})(\overline{C+AB})$

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

$\overline{P}\overline{Q} = \overline{B}\,\overline{A} + \overline{B}\,\overline{C} + \overline{C}\,\overline{A} + \overline{C}\,\overline{B}$

$\overline{P}\overline{Q} = \overline{B}[\overline{A}+\overline{C}]\,\overline{C}(\overline{A}+\overline{B}) = (\overline{A}\overline{C}+\overline{C})(\overline{A}\overline{B}+\overline{B})$

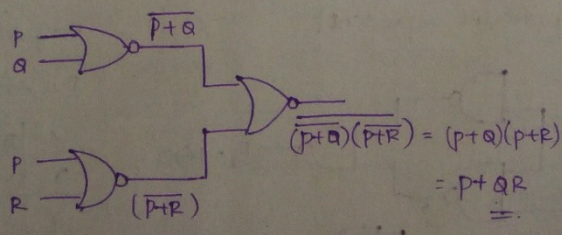
$\boxed{\overline{P}\overline{Q} = \overline{C}+\overline{B}.}$

5. MINIMUM NO-OF NOR GATES EXAMPLE

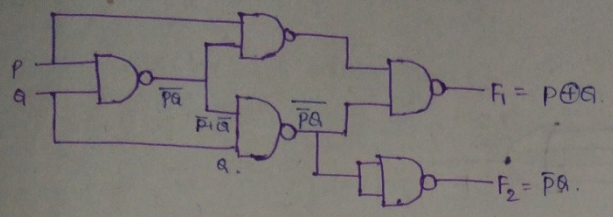
Find the min no of 2 i/p NOR gates required to represent $F(P,Q,R) = P+QR$

$F(P,Q,R) = (P+Q)(P+R)$

$= \overline{(\overline{P+Q})(\overline{P+R})}$



9. HALF SUBTRACTOR

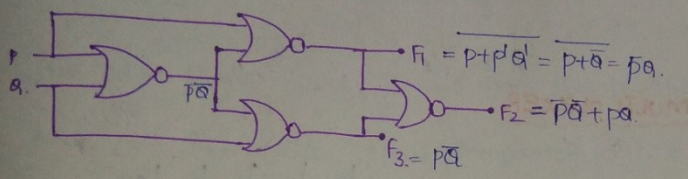


Difference

Borrow.

P	Q	F_1	F_2
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

10. COMPARATOR



$P=Q$ $P<Q$. $P>Q$.

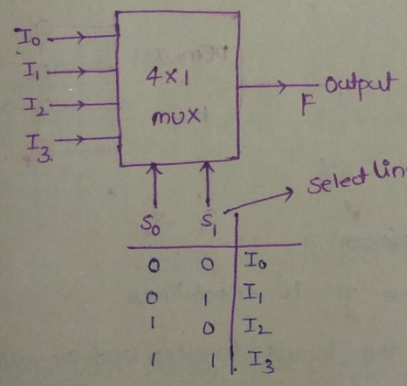
P	Q	F_2	F_1	F_3
0	0	1	0	0
0	1	0	1	0
1	0	0	0	1
1	1	1	0	0

11. MULTIPLEXER

→ A MUX is an electronic switch that can connect one out 'n' inputs to output

→ It cannot change the logical level of the input, it only provides the connection between input and output.

→ It is functionally complete, i.e all boolean func can be realized using only multiplexer without any other gates.



S_0	S_1	
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

$F = \bar{S}_0\bar{S}_1 I_0 + S_0\bar{S}_1 I_1 + S_1\bar{S}_0 I_2 + S_1 S_0 I_3$

If S_0, S_1 are 0,0 then o/p will be I_0

S_0, S_1 are (0,1) then o/p will be I_1

S_0, S_1 are (1,0) then o/p will be I_2

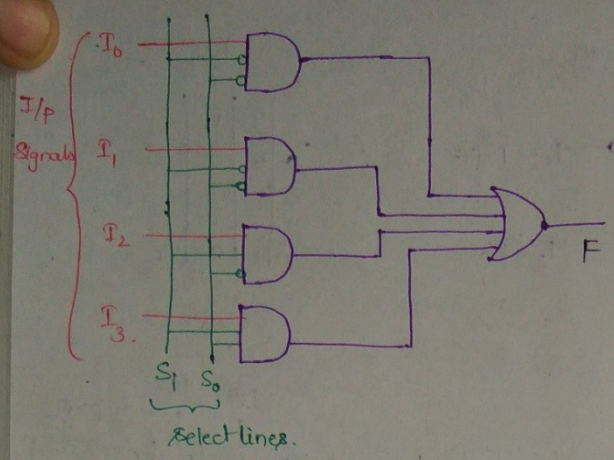
S_0, S_1 are (1,1) the o/p will be = I_3

⇒ The select lines are used to pick one among the inputs and show it as output

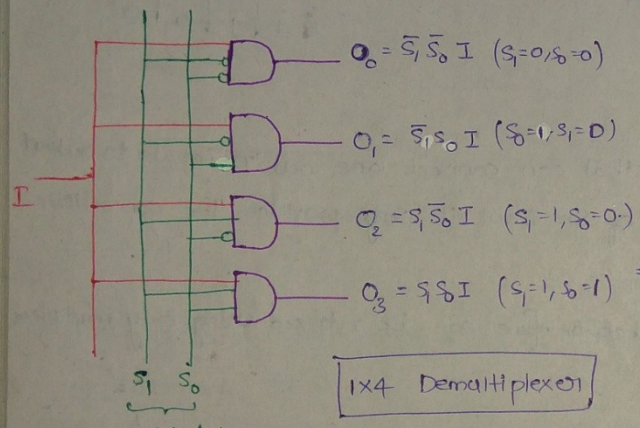
⇒ If we have $2^n \times 1$ MUX then we have 'n' select lines. and 'n' bits are uniquely required to represent each input signal

$\therefore F = \bar{S}_0\bar{S}_1 I_0 + S_0\bar{S}_1 I_1 + S_1\bar{S}_0 I_2 + S_1 S_0 I_3$.

AND-OR REALISATION.



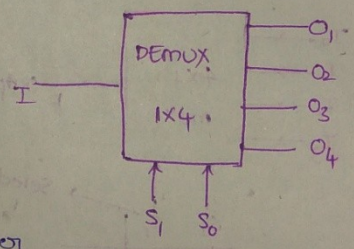
22. INTRODUCTION TO DE MULTIPLEXER



1×4 Demultiplexer

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

⇒ The AND GATE works if all the other inputs (other than that you are giving) are ones)



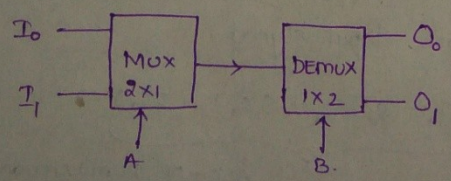
→ It performs opposite operation of Multiplexer

→ It has one input and 2^n outputs where 'n' is select lines

→ It is derived by Mux by joining all the inputs together and removing OR GATE

→ Mainly used in construction of switches

→ DEmux is used at Receiving end and mux is used at transmiting end



A	B	
0	0	$(I_0 - O_0)$
1	1	$(I_1 - O_1)$
0	1	$(I_0 - O_1)$
1	0	$(I_1 - O_0)$

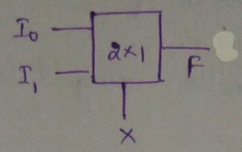
} Straight connection

} Cross connections

12. PROVING MUX IS FUNCTIONALLY COMPLETE

To say functionally complete we should derive (+, ·, −) or (+, −) (·, −)

Now, let us consider 2×1 Multiplexer.



output func: $\boxed{F = \bar{X} I_0 + X I_1}$

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

when X=0, I_0 selected

when X=1 I_1 selected

Now, the select line 'X' can be '0' or 1 then if X=0. then I_0 will be selected

Now, $\bar{X} = \bar{X} \cdot 1 + X \cdot 0$. Now comparing with F

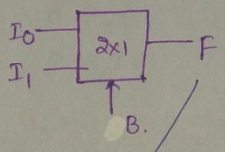
when $I_0 = 1$, $I_1 = 0$. then it generates NOT (Behaves in compliment manner)

⇒ Now,

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

$F = AB = B(A) + \bar{B}(0)$

$F = \bar{B} I_0 + B I_1$



$I_0 = 0$, $I_1 = A$ then AND can be realised

⇒ Now,

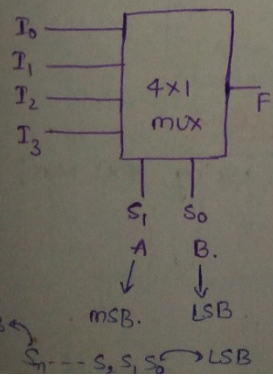
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

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

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

$F = \bar{B} I_0 + B I_1$ ∴ OR can be Realised.

13. IMPLEMENTING FUNCTIONS WITH MUX EXAMPLE 1



$F = \bar{A}\bar{B} I_0 + \bar{A} B I_1 + A \bar{B} I_2 + AB I_3.$

Now, I want to implement function 'g' say

A	B	g
0	0	1
0	1	0
1	0	0
1	1	1

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

$g(A,B) = \bar{A}\bar{B}(1) + \bar{A}B(0) + A\bar{B}(0) + AB(1)$

↓ I_0 ↓ I_1 ↓ I_2 ↓ I_3.

∴ MUX can implement 'g'

Now, using 4×1 multiplexer i can implement a function of 2 variables

⇒ $(2^2 \times 1)$ MUX → 2 variable fun can be Implemented

⇒ $(2^3 \times 1)$ MUX → 3 variable fun can be Implemented

⇒ $(2^n \times 1)$ MUX → A function of n-variable can be Implimented

Now, can I implement a 3 variable function using (4×1) MUX ?? Yes it is possible but we might have to use more gates (in some cases)

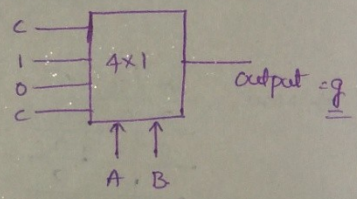
A	B	C	g
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

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

$g = \bar{A}\bar{B}(C) + \bar{A}B(C+\bar{C}) + AB(0) + AB(C)$

Now, (4×1) MUX equation $F = \bar{A}\bar{B}I_0 + \bar{A}BI_1 + A\bar{B}I_2 + ABI_3$

comparing f/g $I_0 = C$, $I_1 = C + \bar{C} = 1$, $I_2 = 0$, $I_3 = C$



∴ Now, just by using $(2^n \times 1)$ MUX we may implement a function having more than 'n' variables. but i may require some more gates.

⇒ Now, we cannot implement a function of 4 variables using (4×1) MUX because out of the 4 variables 2 variables must be given to select lines and the remaing two variables are not independent they are function of two variables and we require more gates.

The function that is formed with 4 variables is

$$F = \underbrace{\bar{A}\bar{B}}_{I_0}(CD) + \underbrace{\bar{A}B}_{I_1}(\bar{C}\bar{D}) + \underbrace{A\bar{B}}_{I_2}(C\bar{D}) + \underbrace{AB}_{I_3}(C\bar{D})$$

Select lines — function requires AND GATE

14. IMPLEMENTING FUNCTIONS WITH MUX - EXAMPLE-2

Implement the following function $f(A,B,C) = \Sigma(1,2,4,6,7)$ using 4×1 MUX?

ented

it is possible
ve might have
e more gates
me cases)

$_2 + AB I_3$.

output = g

more than

MUX because
el the
variables

MUX?

$f(A,B,C) = \Sigma(1,2,4,6,7)$ ⇒ for 4×1 MUX the characteristic Equation is

$G_1 = \bar{A}\bar{B}I_0 + \bar{A}BI_1 + AB\bar{I}_2 + ABI_3$.

	A	B	C	f
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

$\Rightarrow f = A'B'C + A'BC' + AB'C' + ABC' + ABC$

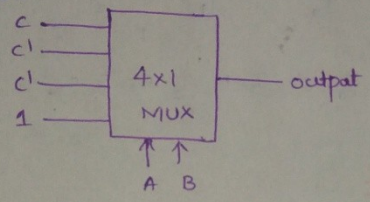
$= A'B'(C) + A'B(C') + AB'(C') + AB(C+C')$

$f = A'B'(C) + A'B(C') + AB'(C') + AB$.

comparing with G_1 $I_0 = C$ $I_2 = C'$

$I_1 = C'$ $I_3 = 1$

The MUX will be



A=0, B=0 ⇒ o/p = C

A=0, B=1 ⇒ o/p = C'

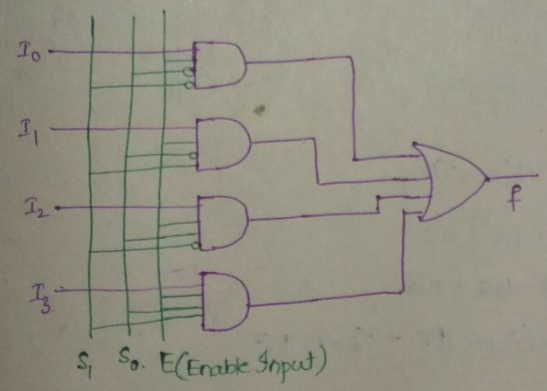
A=1, B=0 ⇒ o/p = C'

A=1, B=1 ⇒ o/p = 1

15. IMPLEMENTING WITH MUX - EXAMPLE - 2

$f(A,B,C) = \Sigma$

15. MULTIPLEXER WITH ENABLE INPUT

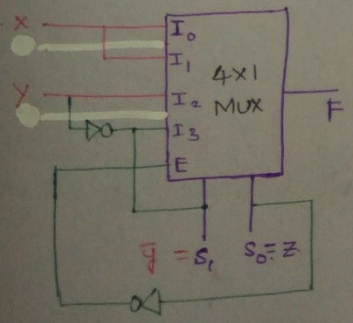


$f = E(I_0\bar{S}_1\bar{S}_0 + I_1S_1\bar{S}_0 + I_2\bar{S}_1S_0 + I_3S_1S_0)$

⇒ If the value of E=1 then mux works and gives o/p 1

⇒ If E=0 ⇒ MUX stops and gives '0'

Mimise the function represented by following MUX?



The characteristic Equation of 4×1 MUX is

$F = E(I_0\bar{S}_0\bar{S}_1 + I_1\bar{S}_1S_0 + I_2S_1\bar{S}_0 + I_3S_1S_0)$

$= \bar{z}(x\bar{z}\bar{y} + x\bar{y}z + y\bar{y}\bar{z} + \bar{y}\bar{y}z)$

$= \bar{z}(x\bar{z}\bar{y} + xyz + 0 + \bar{y}z)$

$= \bar{z}(xy + \bar{y}z)$ ∴ $F = xy\bar{z}$

$= xy\bar{z} + \bar{y}z\bar{z} = xy\bar{z}$ //

16. RELATION BETWEEN SELECT LINES AND INPUTS OF A MUX

$F(A,B,C) = \Sigma(2,3,5,6,7)$ Implement using 4x1 MUX such that,

a) $S_1 = B$, $S_0 = C$

b) $S_1 = C$ $S_0 = B$

Sol: $F = \Sigma(2,3,5,6,7) = \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$

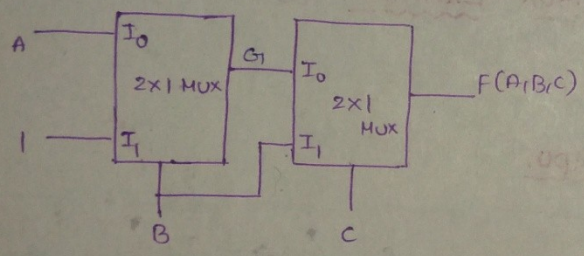
Now, the select lines are B,C $\Rightarrow$ $(0)\,\bar{B}\bar{C} + (A)\,\bar{B}C + (1)\,B\bar{C} + (A)\,BC$

$\downarrow I_0$ $\downarrow I_1$ $\downarrow I_2$ $\downarrow I_3$

Now, if $S_1 = C$ and $S_0 = B$ then $I_0 = 0$, $I_1 = 1$, $I_2 = A$, $I_3 = A$

17. CASCADING MUX EXAMPLE -1

What is the function in canonical sop?



$G_1 = \bar{B}A + B(1)$

$\boxed{G_1 = B + \bar{B}A}$

$F = \bar{C}I_0 + CI_1$

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

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

$F = A\bar{B}\bar{C} + BC + B\bar{C}$

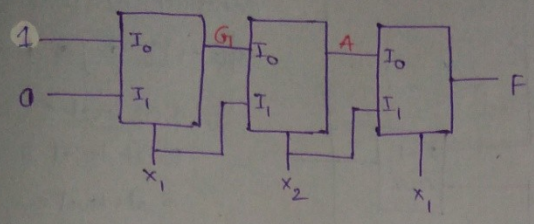
$= AB\bar{C} + (A+A')BC + (A+A')B\bar{C}$

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

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

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

18. CASCADING MUX EXAMPLE-2



$G_1 = \bar{X}_1(1) + X_1(0) = \bar{X}_1$

$A = \bar{X}_2(I_0) + X_2(I_1)$

$= \bar{X}_2(G_1) + X_2(X_1)$

$= \bar{X}_2 \bar{X}_1 + \bar{X}_2 X_1$

$A = (X_1 \odot X_2)$

$F = \bar{X}_1 A + X_1(I_1)$

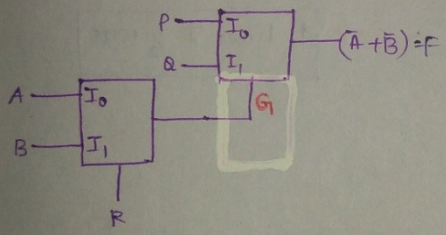
$= \bar{X}_1(X_1 \odot X_2) + X_1(X_2)$

$= \bar{X}_1(X_2 X_1 + \bar{X}_2 \bar{X}_1) + X_1 X_2$

$= \bar{X}_2 \bar{X}_1 + X_1 X_2$

$\boxed{F = X_1 \odot X_2}$

19. CASCADING MULTIPLEXERS - EXAMPLE-3



$G_1 = \bar{R}(I_0) + R(I_1) = \bar{R}(A) + R(B)$

$F = \bar{G}_1(P) + G_1(Q)$

$\bar{A}+\bar{B} = \overline{[\bar{R}(A)+R(B)]}P + [\bar{R}(A)+R(B)]Q$

$\bar{A}+\bar{B} = [(R+\bar{A})(\bar{R}+\bar{B})]P + [\bar{R}A + RB]Q.$

$\bar{A}+\bar{B} = [R\bar{R} + R\bar{B} + \bar{A}\bar{R} + \bar{A}\bar{B}]P + [\bar{R}AQ + RBQ]$

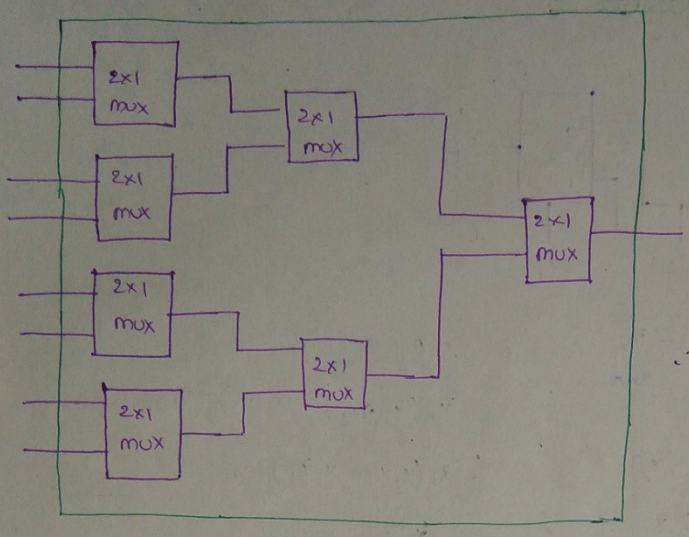
$\bar{A}+\bar{B} = R\bar{B}P + \bar{A}\bar{R}P + \bar{A}\bar{B}P + \bar{R}AQ + RBQ.$

= Now, do opvm.

=.

20. EXPANSION OF MUX'S

How can you construct a 8×1 MUX using 2×1 MUX?



8×1 Mux

Now, it is clear that to construct 8×1 mux we need 7 [(2×1) MUX] and at 3 levels

∴ 4 at level 1
2 at level 2
1 at level 3.

∴ Totally 7 Mux are needed
3 levels are formed

Now, Initially I have 8 lines to cover and now my (2×1) Mux will cover two lines

⇒ 1 Mux will cover 2 lines

⇒ 2 lines → 1 mux

⇒ 1 line → ½ mux

∴ 8 lines are covered by $8/2$ mux = (4)

Now, 4 lines are covered by $4/2$ mux = (2) mux

Now 2 lines are covered by = (1) mux

7 mux

i.e 8 lines → $8/2$ mux → 4

4 lines → $4/2$ mux → 2 mux

2 lines → $2/2$ mux → 1 mux.

(2) Ex: Construct 32×1 MUX with 4×1 MUX

Initially 32 lines and my small 4×1 mux covers 4 lines ⇒ 1 mux → 4 lines

1 line → ¼ mux

32 lines → $32/4$ mux = 8

8 lines → $8/4$ mux = 2

2 lines → $2/4$ mux = $\lceil 1/2 \rceil = 1$

∴ 11 mux are needed to cover 32×1 MUX

11 MUX

So, when you are trying to manfacture a (M×1) MUX using (N×1) MUX then how many levels are Required and how many MUX are required.

Given,

(m×1)
m lines

∴ one

⇒ M li

No

⇒ Now,

Ex : No

Devices

(8×1)
↓
m

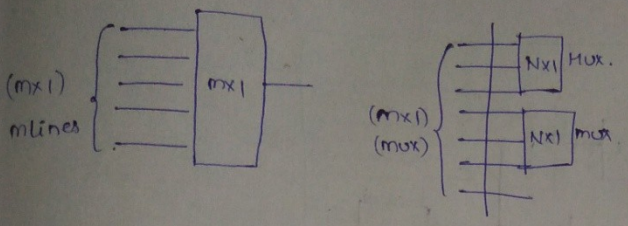
Ex: (32×

↓
m

⇒ levels

that to
x we need
d at 3 levels
2
.
e needed
ned
two lines
2 lines
ux
x
4 lines
1/4 mux
1 mux
then

Given, (M×1) MUX should be covered by (N×1) MUX



∴ one (N×1) MUX is going to cover N lines ⇒ 1 Device covers N lines

⇒ 1 line → 1/N Devices

⇒ M lines will be covered by M/N Devices.

Now, M lines → (M/N) devices

M/N lines → (M/N²) devices

(M/N²) lines → M/N³ devices

⋮

① we continue this procedure until $\frac{M}{N^K} \leq 1$

$\Rightarrow N^K \geq m \Rightarrow K \geq \lceil \log_N m \rceil$

No. of levels

⇒ Now, the No of devices used are $M/N + M/N^2 + M/N^3 + \cdots + M/N^K$

$$\text{Devices} = \sum_{K=1}^{\lceil \log_N m \rceil} \left(M/N^K\right)$$

Ex: Now, I want to implement 8×1 mux using (2×1) mux, how many levels and Devices are needed?

(8×1) → (2×1) mux ⇒ No. of levels = $\lceil \log_N m \rceil = \lceil \log_2 8 \rceil = \lceil 3 \rceil = 3$ levels

m N

⇒ No. of devices = $\sum_{K=1}^{3} \left(8/2^K\right) = 8/2 + 8/4 + 8/8$

$= 4 + 2 + 1 = 7$ mux.

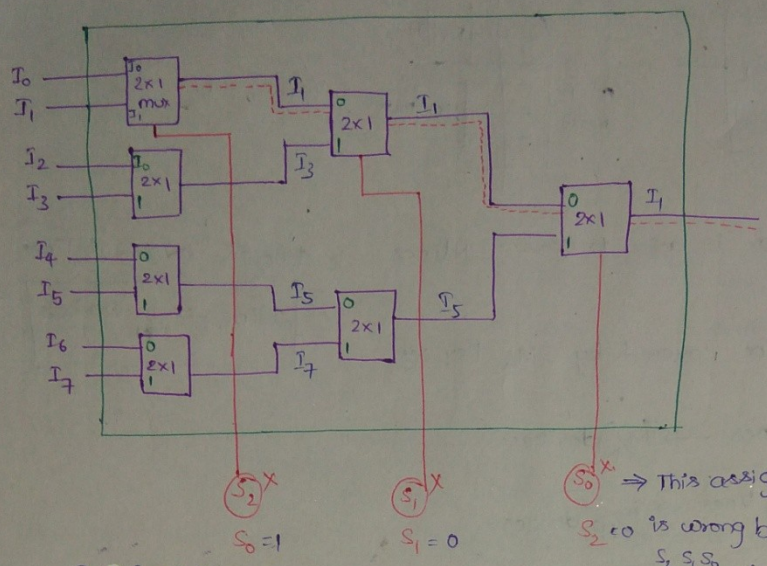
Ex: (32×1) MUX → (4×1) mux.

m N

⇒ levels = $\lceil \log_N m \rceil = \lceil \log_4 32 \rceil = \lceil 5/2 \rceil = \lceil 2.5 \rceil = 3$ //

Devices = $32/4^1 + 32/4^2 + 32/4^3 = 32/4 + 32/16 + 32/64 = 8 + 2 + \underset{1}{(1/2)} = 11$ mux //.

ASSIGNING SELECT LINES WHILE EXPANDING THE MUX. 21



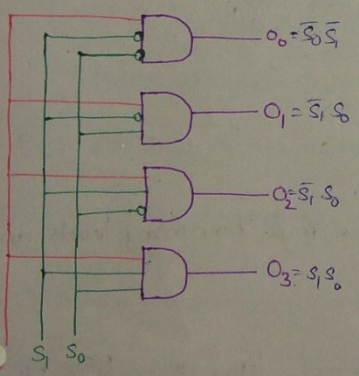
$S_0 = 1$ $S_1 = 0$ $S_2 = 0$

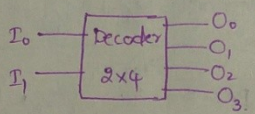
$S_0 = 1, S_1 = 0, S_2 = 0$

The Inputs that gets selected at level one are (I_1, I_3, I_5, I_7)

⇒ This assignment of select lines is wrong because if we give $S_2 S_1 S_0$ 001 then the o/p should be I_1, but if $S_2 = 0$ then it will select 'I_0' not I_1

23. INTRODUCTION TO DECODER





MUX = $2^n \times 1$

DEMUX = 1×2^n

DECODER = $n \times 2^n$

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

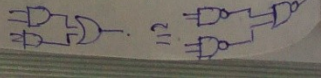
$O_0 = \bar{S}_1 \bar{S}_0$
$O_1 = \bar{S}_1 S_0$
$O_2 = S_1 \bar{S}_0$
$O_3 = S_1 S_0$

→ A demux can be converted to a decoder by always setting I = 1 and making select lines as inputs.

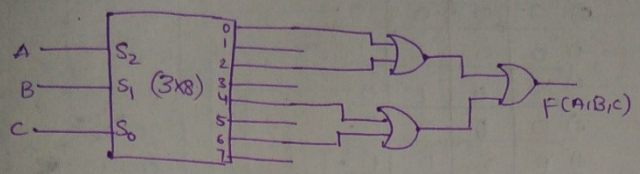
→ Since a decoder provides all the minterms, we could implement, we could implement a func in canonical sop using "OR" Gate.

→ If all the 'AND' gates are replaced with NAND gates the decoder becomes active low.

→ Implementing func with decoder is AND-OR REALISATION / NAND-NAND REALISATION.



24. IMPLEMENTING FUNCTIONS WITH DECODER EXAMPLE-1

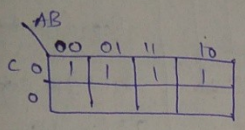


F is free from

a) 1 variable c) 3 variable

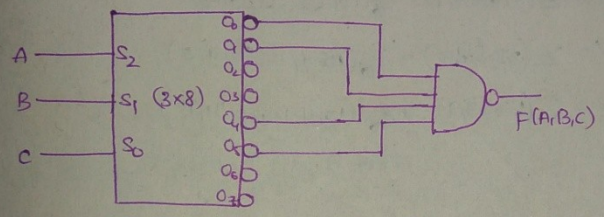
b) 2 variables d) None

$F = \Sigma(0,2,4,6)$



$\therefore \boxed{F = c'} \Rightarrow$ Independent of 2 variables.

25. IMPLEMENTING FUNCTIONS WITH DECODER EXAMPLE-2

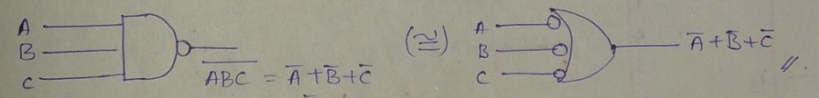


F is free from.

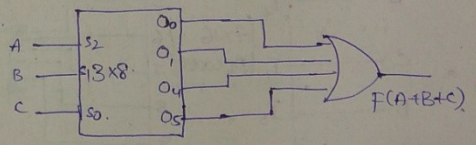
a) 1 variable b) 2 variables

3) 3 variables d) None.

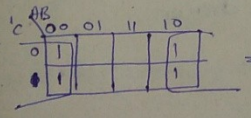
Now,



Now, the above diagram can be modified as.



$\Sigma = (0,1,4,5)$



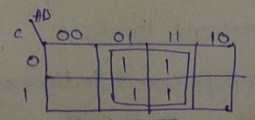
$\therefore \boxed{F = \bar{B}}$ Independent of two variables.

26. DECODER FOR FUNCTION IMPLEMENTATION - Ex-3

Same Question above but the NAND gate is replaced by AND Gate.

⇒ Now, when a minterm is bubbled (complemented) then it becomes maxterm.

$O_0 = \overline{(\bar{a}\bar{b}\bar{c})} = (a+b+c) \quad \therefore \Pi(0,1,4,5) = \Sigma(2,3,6,7)$



$\boxed{F = B}$ = Independent of '2' variables

27. CONVERTING ONE CODE TO ANOTHER USING DECODER

	A (8)	B (4)	C (2)	D (1)
0 =	0	0	0	0
1 =	0	0	0	1
2 =	0	0	1	0
3 =	0	0	1	1
4 =	0	1	0	0
5 =	0	1	0	1
6 =	0	1	1	0
7 =	0	1	1	1
8 =	1	0	0	0
9 =	1	0	0	1

	W (2)	X (4)	Y (2)	Z (1)
0)	0	0	0	0
1)	0	0	0	1
2)	0	0	1	0
3)	0	0	1	1
4)	0	1	0	0
5)	1	0	1	1
6)	1	1	0	0
7)	1	1	0	1
8)	1	1	1	0
9)	1	1	1	1

compliments of each other (self complimented Representation)

In 2421 code the 9's compliment of 5 = 9-5 = 4

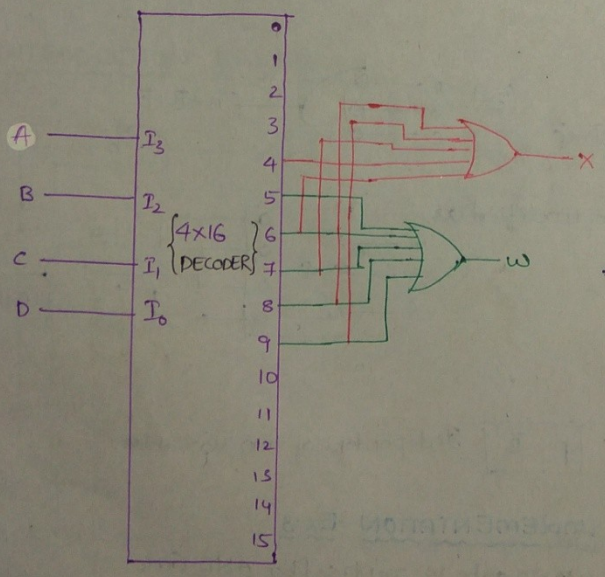
So take 4 & compliment it

$W = \Sigma(5,6,7,8) + \phi(10,11,12,13,14,15)$

$X = \Sigma(4,6,7,8,9) + \Sigma_\phi(10,11,12,13,14,15)$

$Y = \Sigma(2,3,5,8,9) + \Sigma_\phi(10,11,12,13,14,15)$

$Z = \Sigma(1,3,5,7,9) + \Sigma_\phi(10,11,12,13,14,15)$



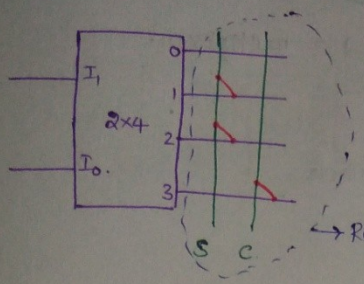
→ Similarly Y, Z can also be drawn.

28. Rom IMPLEMENTATION USING DECODER

→ ROM can be used to realise combinational functions by storing appropriate values at appropriate locations

→ Every ROM is expressed in terms of ROM Matrix and decoder

→ Rom matrix contain set of links and connections. The lines entering the matrix and leaving it are called the intersection of rows and columns are called links.



	A	B	Sum(S)	Carry(C)
0)	0	0	0	0
1)	0	1	1	0
2)	1	0	1	0
3)	1	1	0	1

29. IMPLEMENTING FUNCTIONS USING ONLY DECODER

→ To implement a function of 'n' variables we need $(n \times 2^n)$ decoder

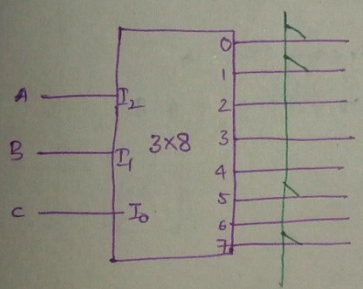
∴ The no. of lines comming/entering into rom matrix is 2^n

The no. of connections to implement m-func = $(2^n + m)$ connections

" " " links (x^n points) to implement m-func = $(2^n * m)$ links

30. IMPLEMENTING FUNCTIONS USING DECODER + MUX EXAMPLE 1

Now, let us consider 3×8 decoder

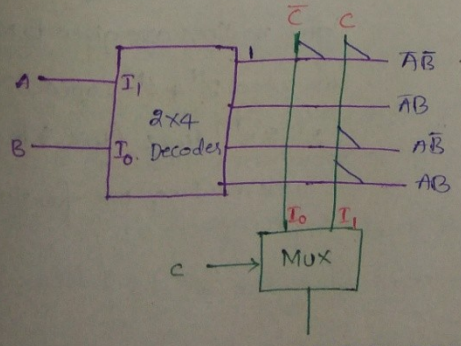


(8×1) links

(8+1) connections.

$f(A,B,C) = \Sigma(0,1,5,7) = \bar{A}\bar{B}\bar{C} + \bar{A}BC + A\bar{B}\times C + ABC$

Now, I need to implement the same function using 2×4 decoder. then we need a multiplexer also.

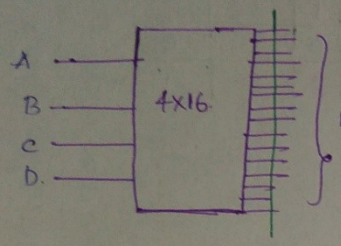


$f(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}C + ABC$

for all this inputs output should be one! and whenwe give inputs other than these it should give

31. IMPLEMENTING FUNCTIONS USING DECODER + MUX- EXAMPLE-2

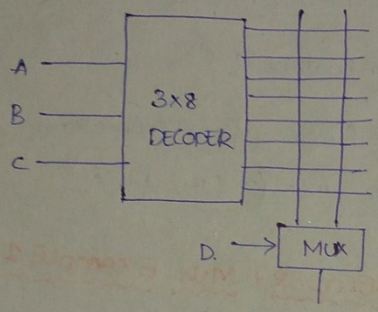
Implement 4x16 Decoder using 3x8 decoder and a multiplexer?



(16+1) connections

(16*1) links.

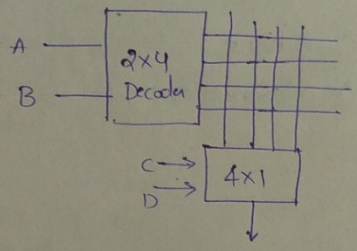
Now, 3x8 Decoder



links = 8x2 = 16

connections = 8+2 = 10

Now, 2x4 Decoder.



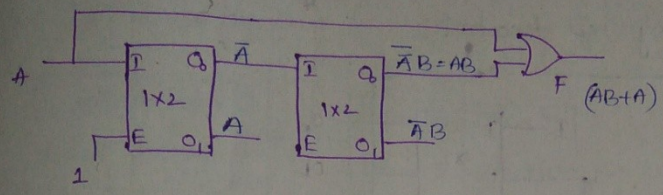
links = 4x4 = 16

connections = 4+4 = 8.

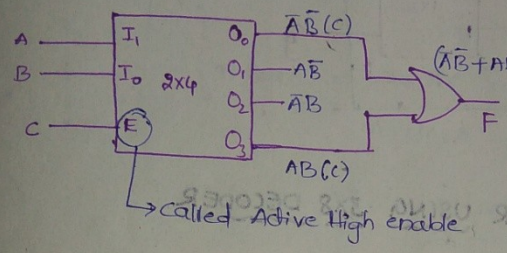
Decoder(n)	MUX	connections	Links
10×2^{10}	0	1024+1	1024
5	5	$2^5+2^5=64$	$2^5*2^5=1024$
4	6	$2^4+2^6=80$	$2^4*2^6=1024$
3	7	$2^3+2^7=136$	$2^3 \times 2^7=1024$

Now, if i want to implement a function of (n+m) variables where n lines are given to decoder and m lines are given to MUX then we need $2^N + 2^M$ connections and $2^N * 2^m$ links. (This is true for implementing one function

32. DECODER WITH ENABLE INPUT

① 

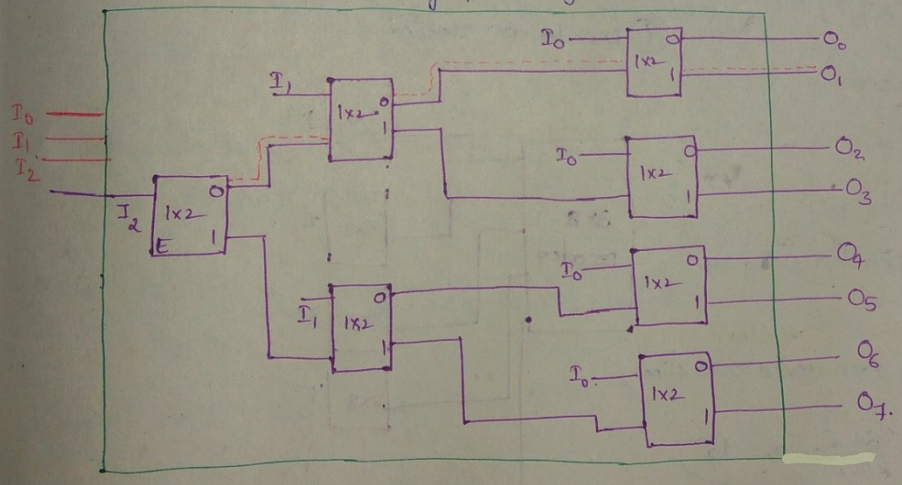
What will be the function?

② 

33. CONSTRUCTING 3X8 DECODER USING 1X2 DECODER

Construct 3X8 Decoder using 1X2 Decoder.

For Decoders start building from Right side (←→).



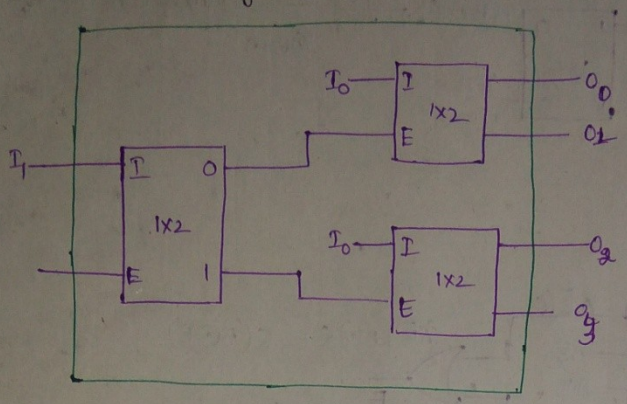
⇒ we have to cover 8 lines $(O_0\ O_1 \dots O_7)$ and each 1X2 Decoder covers 2 lines so at level 1 we will have 4 decoders Now, join all the Enables of 4 Decoders and 4 Enables need two decoders at level 2 and one decoder at level 1.

⇒ Now let me assign Input I_0 at level 1 for all decoders and I_1 at level 2 and I_2 at level 3 (I'm not sure about it just assigned randomly to check correct/not)

⇒ Now, If is say $I_2\ I_1\ I_0 = (0,0,1)$ ⇒ $I_2 = 0$ ⇒ 0 is selected and passed
⇒ $I_1 = 0$ → 0 is selected and passed
⇒ $I_0 = 1$ ⇒ 1 is selected and O_1 is enabled
then O_1 should be enabled

34. CONSTRUCTING 4X2 DECODER USING 1X2 DECODER

Construct 2×4 decoder using 1×2 decoder?



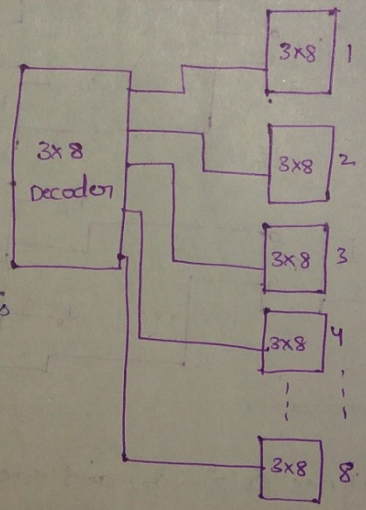
35. CONSTRUCTING 6X64 DECODER USING 3X8 DECODER

64 lines → are present, Each device covers 8 lines

⇒ 64 lines → 64/8 devices = 8 Decoders

8 lines → 8/8 devices = 1 decoder

(9) Decoders are needed.



Now, 6×64 decoder using 2×4 decoders

64 lines and each device covers 4 lines

= 64 lines — $\frac{64}{4}$ Devices = 16

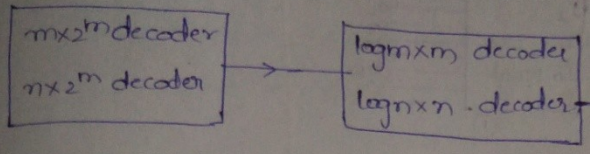
16 lines — 16/4 devices = 4

4 lines — $\frac{4}{4}$ D = 1 device

21 decoders are needed (16 at level 1
4 at level 2
1 at level 3)

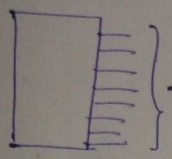
36. EXPANSION OF DECODER IN GENERAL

Now I want to construct $(m \times 2^m)$ decoder by using $(n \times 2^n)$ decoder.



$\Rightarrow$ 1 line means/covers $1/n$ devices

$\Rightarrow$ **1 line = $1/n$ devices**



$\Rightarrow$ Now the no. of decoders required to cover m lines = (m/n) devices are needed.

$\Rightarrow$

m lines $\longrightarrow (m/n)$ devices

(m/n) lines $\longrightarrow (m/n^2)$ devices

(m/n^2) lines $\longrightarrow (m/n^3)$ devices

$\vdots$

$m/n^k \Rightarrow m/n^k \leq 1$

$\Rightarrow m \leq n^k$

$\Rightarrow n^k \geq m \Rightarrow k \geq \lceil \log_n m \rceil$

No. of levels = $\lceil \log_n m \rceil$

No. of devices needed = $\sum_{k=1}^{\log_n m} (m/n^k)$

Ex: 6×64 decoder | $m = 64$

4×16 decoder | $n = 16$

$$\text{levels} = \frac{\log 64}{\log 16} = \lceil 6/4 \rceil = \lceil 3/2 \rceil = 2 \text{ levels are needed}$$

$$\text{No. of devices} = 64/16 + 64/16^2$$

$$= 6 + \frac{64}{16 \times 16}$$

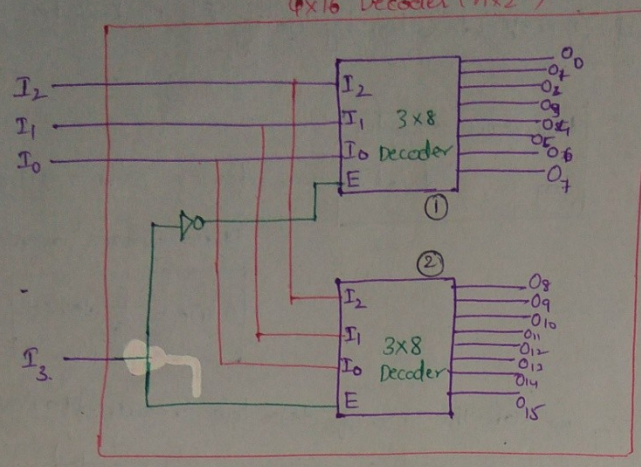
$$= 6 + \lceil 1/4 \rceil = 6 + 1 = 7 \text{ decoders are needed}$$

38. EXPANSION OF DECODERS IN ANOTHER WAY

Construct 4×16 Decoder using two (3×8) decoders.

We assume that there are no restrictions (only so and so decoders are available like wise).

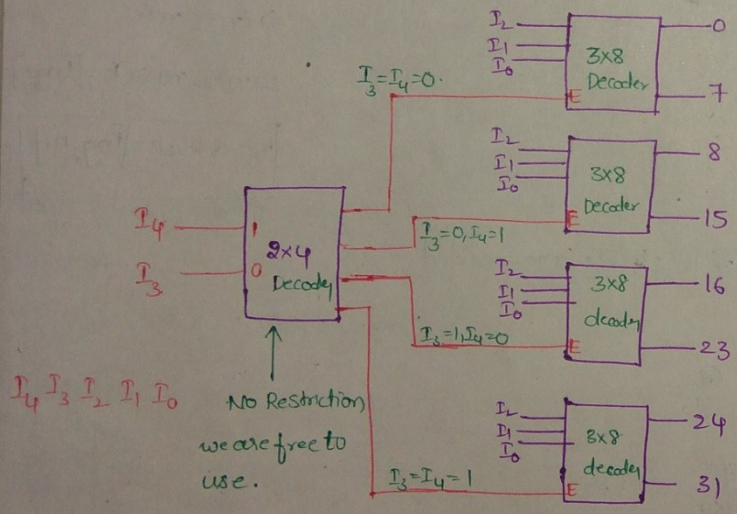
4x16 Decoder ($n \times 2^n$)



Now, if $I_3 I_2 I_1 I_0 = 0000$ then the 2nd decoder will be disables because E=0 (In fact the decoder ② represent the output in which the most significant bit is 1 (I_3 is the MSB) and Decoder 1 represents the MSB (I_3)=0.).

$I_3 I_2 I_1 I_0 = 0000 \Rightarrow O_0$ will be output

Ex:2 construct a (5x32) decoder using 4(3x8) decoders.

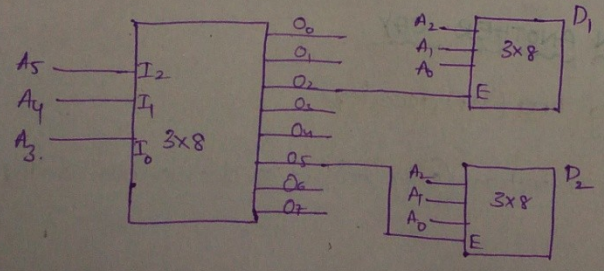


= 32 lines

= each device-8 line

= 32/8 devices = 4D

4.1. FINDING THE ADDRESS RANGES OF DEVICES



⇒ Now, the given Address lines are $A_5\ A_4\ A_3\ A_2\ A_1\ A_0$

Now, for D_1 to be enables the O_2 should be enabled ⇒ $A_5\ A_4\ A_3$ should be = (010)

$\therefore$ $A_5\ A_4\ A_3\ A_2\ A_1\ A_0$

$0\ 1\ 0 \begin{Bmatrix} 0\ 0\ 0 \\ 0\ 0\ 1 \\ \vdots \\ 1\ 1\ 1 \end{Bmatrix}$

similarly for D_2, O_5 should be enabled

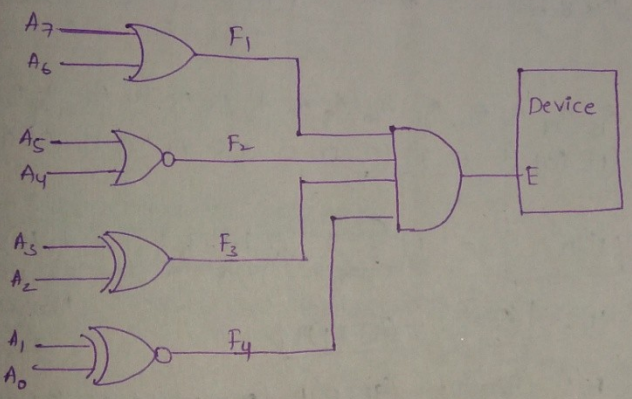
⇒ $A_5 A_4 A_3 = 101$

$A_5\ A_4\ A_3\ A_2\ A_1\ A_0$

$1\ 0\ 1 \begin{Bmatrix} 0\ 0\ 0 \\ 0\ 0\ 1 \\ \vdots \\ 1\ 1\ 1 \end{Bmatrix}$

42. EXAMPLE ON ENABLING A DEVICE

Consider a device that is enabled using 8 address bits as shown below. for how many address, the device is enabled



The device is Enabled when the output of AND GATE is '1', the o/p of AND GATE is one iff All of its inputs should definitely be '1'.

$\therefore$ $F_1 = 1$, $F_2 = 1$, $F_3 = 1$, $F_4 = 1$

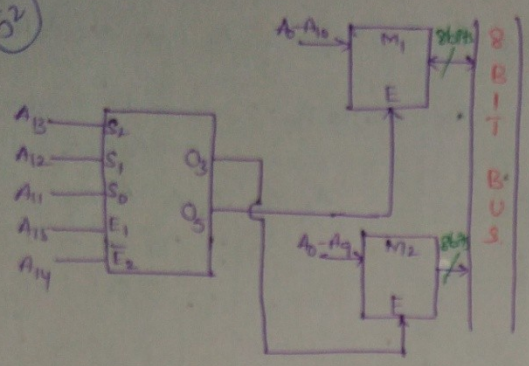
F_1	F_2	F_3	F_4
3 Combination	1 combination	2 combinations	2 combinations
makes $F_1 = 1$ $\begin{Bmatrix} 0,1 \\ 1,0, \\ 11 \end{Bmatrix}$	$F_2 = \{1\} = \{0,0\}$ (A B)	$\begin{Bmatrix} 0,1 \\ 1,0 \end{Bmatrix}$	$\begin{Bmatrix} 1,1 \\ 0,0 \end{Bmatrix}$

$\therefore$ Total Addresses = 3 × 1 × 2 × 2 = 12 Addresses.

43. FINDING THE ADDRESS RANGES OF MEMORY DEVICES

Consider the following interface of two memory devices with 3×8 decoder. Compute the Nature and Address range of each memory device

(52)



Clearly M_1 is Bidirectional Device
M_2 is unidirectional Device
→ M_1 = Ram, M_2 = Rom

M_1: No of address bits = 11 ⇒ 2^{11} address are possible
⇒ 2^{11} words are stored
⇒ size of each word = 8 bits
⇒ size of m_1 = $2^{11} \times 8 = 2^{14}$ bits
$= 2^{11}$ B

M_2: No of Address bits = $A_9 A_8 A_7 \dots A_0 = 10$
⇒ 2^{10} Address are possible ⇒ 2^{10} words can be uniquely identified
⇒ size of each word = $2^{10} \times 8$ bits = 2^{13} bits = 2^{10} Bytes.

M_1 = (A_0 – A_{15}) are present

To enable m_1 ⇒ O_5 should be enabled ⇒ $S_2 S_1 S_0$ ≠ (101) & the decoder should also be enabled so that $S_2 S_1 S_0$ = (101) will be enabled.

$A_{15}\ A_{14}\ A_{13}\ A_{12}\ A_{11}\ A_{10}\ A_9\ A_8\ A_7\ A_6\ A_5\ A_4\ A_3\ A_2\ A_1\ A_0$
(1) (0) (1 0 1) (0 0 0 0 0 0 0 0 0 0 0)

(1)(0) (1 0 1) (1 1 1 1 1 1 1 1 1 1 1)

M_1: (A80d)H – (AFFF)H

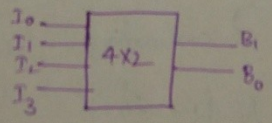
M_2: (980d)H – (9FFF)H (enumerate as you did above)

Now combine 4,4 together

44. INTRODUCTION TO ENCODERS

Ex: 4x2 Encoder (Non-priority)

I_0	I_1	I_2	I_3	B_1	B_0
0	0	0	1	1	1
0	0	1	0	1	0
0	1	0	0	0	1
1	0	0	0	0	0



MUX : $2^n \times 1$
DEMUX : 1×2^n
DECODER : $n \times 2^n$
ENCODER : $2^n \times n$

$B_1(I_3, I_2, I_1, I_0) = I_0' I_1' I_2 I_3 + I_0' I_1' I_2 I_3'$

$B_1 = I_0' I_1' (I_2 \oplus I_3)$

$B_2(I_3, I_2, I_1, I_0) = I_0' I_1' I_2' I_3 + I_0' I_1 I_2' I_3'$

$= I_0' I_2' (I_1 I_3 + I_3' I_1)$

$B_2 = I_0' I_2' (I_1 \oplus I_3)$

→ They convert one code to another
→ They perform lossless compression
→ They are of two types — Non priority (Doesn't support simultaneous i/p activation)
priority (supports simultaneous i/p activation and used for interrupt servicing)
→ Due to static priorities, the lower priority i/p is exposed to starvation

45. PRIORITY ENCODER

priority of Inputs: $I_3 > I_2 > I_1 > I_0$.

Highest suffix i/p is given highest priority.

I_0	I_1	I_2	I_3	B_1	B_0
∅	∅	∅	1	1	1
∅	∅	1	0	1	0
∅	1	0	0	0	1
1	0	0	0	0	0

$B_1 = I_3 + I_2 \bar{I}_3$ $\quad$ $B_0 = I_3 + I_1 \bar{I}_2 \bar{I}_3$

$\boxed{B_1 = I_2 + I_3}$ $\quad$ $\boxed{B_0 = I_3 + \bar{I}_2 I_1}$

46. INTRODUCTION TO HAZARDS

→ Hazards < Temporary Hazards
permanent Hazards — mainly occur because of open circuiting / short circuiting.

47. HAZARDS IN DIGITAL CIRCUIT

→ The mal function of a digital circuit is called Hazard. They can be Temporary and permanent. The temporary hazards are due to uneven delays of inputs.
→ The permanent hazards are resulted due to open circuit or short circuit of the connecting leads (Terminals).
→ The hazards can be stuck at 0 (s-a-0) stuck at 1 (s-a-1)
→ The single fault analysis is made using "path sensitization" technique
This technique provides test vector.

Procedure for finding Test vector
→ Inactive all the paths except the tested one, For AND and NAND apply the logic 1 for inactivation, For, OR, NOR apply logic 0 for inactivation
→ Apply the opposite logic value at tested place
→ The input combinations satisfying the above requirements form test vector.

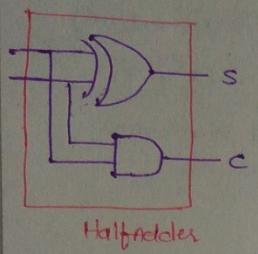
49. HALF ADDER

▷ Half Adder – (Two Input, Two output) circuit →

x	y	sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

sum = $S = \bar{x}y + x\bar{y} = x \oplus y$

carry = xy.

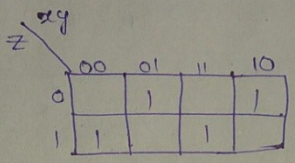


Halfadder

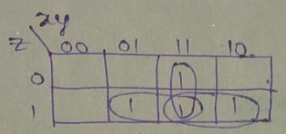
50. FULL ADDER

used for adding 3 bits.

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

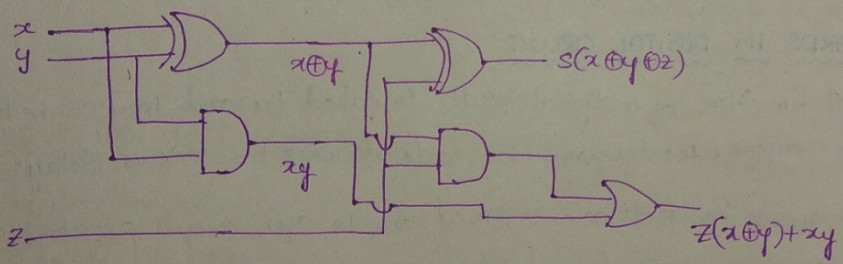


$S = x \oplus y \oplus z$



$C = xy + yz + zx$

$C = z(x \oplus y) + xy$



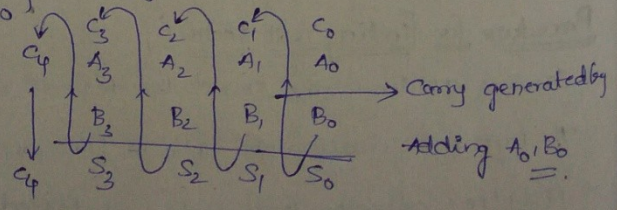
51. RIPPLE CARRY ADDER

The ripple carry adder is used to add two Binary numbers.

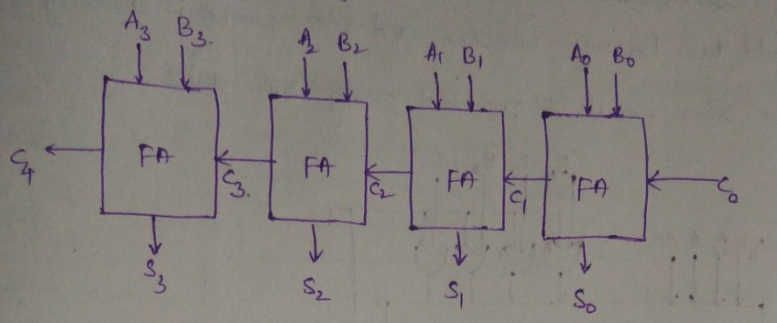
1st Binary number = $A_3\ A_2\ A_1\ A_0$

2nd Binary number = $B_3\ B_2\ B_1\ B_0$

} Now, say carry 'C_0' is provided, then



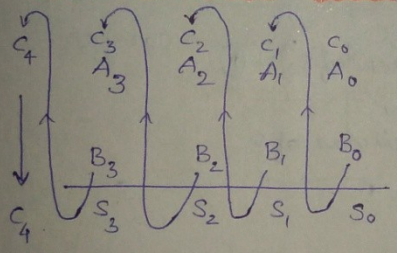
Now we can implement this using Full Adder.



Ripple carry Adder.

⇒ The disadvantage of the Ripple carry adder is C_1 is generated after C_0 and C_2 is generated only after C_1 ∴ The FullAdders cannot be executed in parallel and it takes more time to compute. So some Automated circuits are designed and placed at each full Adder. and it is called carry look ahead adder.

52. CARRY LOOK AHEAD ADDER



$$C_1 = C_0(A_0 \oplus B_0) + A_0B_0$$
$$C_2 = C_1(A_1 \oplus B_1) + A_1B_1$$
$$C_3 = C_2(A_2 \oplus B_2) + A_2B_2$$
$$C_4 = C_3(A_3 \oplus B_3) + A_3B_3.$$

Now, Apply Back substitution method.

Now, let us assume $G_{i_0} = A_iB_i$, $P_i = A_i \oplus B_i$ Now, $C_1 = C_0P_0 + G_0$

Generating func ↓ propagating func

$C_2 = C_1P_1 + G_1$

$C_3 = C_2P_2 + G_2$

$C_4 = C_3P_3 + G_3$

Now, $C_1 = C_0P_0 + G_{10}$

$C_2 = (C_0P_0 + G_{10})P_1 + G_{11}$

$C_2 = (C_0P_0P_1 + G_{10}P_1 + G_{11})$

$C_3 = (C_0P_0P_1 + G_{10}P_1 + G_{11})P_2 + G_{12}$

$C_3 = C_0P_0P_1P_2 + G_{10}P_1P_2 + G_{11}P_2 + G_{12}$

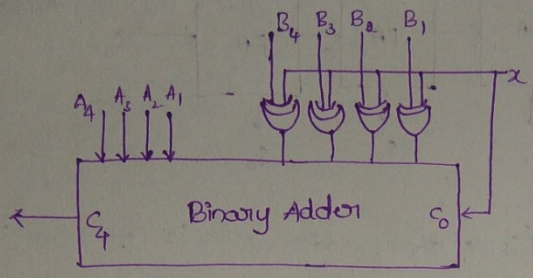
$C_4 = [C_0P_0P_1P_2 + G_{10}P_1P_2 + G_{11}P_2 + G_2]P_3 + G_{13}$

$C_4 = C_0P_0P_1P_2P_3 + G_{10}P_1P_2P_3 + G_{11}P_2P_3 + G_2P_3 + G_{13}$

=

56. BINARY ADDER - SUBSTRACTOR

Now, I need to add two Binary numbers and one number is directly provided as Input and the other Input is provided using XOR gates.



Now $B \oplus 0 = B$
$B \oplus 1 = \overline{B}$
} if $x=0$ then the o/p of XOR GATES will be same $B_4 B_3 B_2 B_1$ and it behaves Binary adder.

if $x=1$ then the o/p of XOR GATES will be complimented
$(\overline{B_4}, \overline{B_3}, \overline{B_2}, \overline{B_1})$ = 1 st compliment

If $x=1$, $A + (1\text{s compliment of } B) + 1$ → Carry's.

$= A +$ 2's compliment of B

$= A - B$

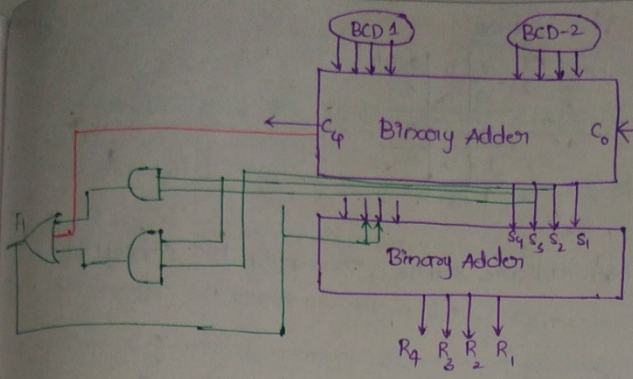
∴ The Adder behaves as Adder when $x=0$
" " " " Substractor " $x=1$.

The above circuit not only functions as Adder and substractor

A	B	x	Function.
BCD	0011	0	BCD → Ex-3 Converter.
Ex-3	0011	1	Ex3 → BCD Converter
1001	BCD	1	(9-B) = 9's compliment of given no B in (BCD)
1010	BCD	1	(10-B) = 10's compliment of given no. B in (BCB).

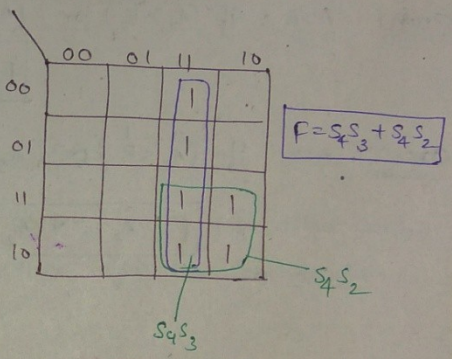
57. BCD ADDER

The output is Invalid if o/p is >9, and we do a small correction (we add 6 to the result) so we need two Adders.



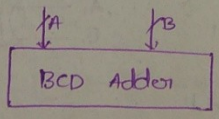
⇒ Correction is Required when $S_4S_3S_2S_1 > 9$ (or) if $C_4 = 1$, Now, lets form a func where the sum of $S_4S_3S_2S_1 > 9$, now the combinations for which the sum is 9 are.

S_4	S_3	S_2	S_1	F
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



⇒ Now, if the sum have to be greater than 10 then the either S_4S_2 should be 1 or S_4S_3 should be 1, irrespective of other combinations if S_4S_2 or S_4S_3 are 1 then (sum is >9) and when $C_4 = 1$ the sum is >9.

58. INVALID COMBINATIONS OF BCD ADDER



A, B are each of 4 bits so 16 combinations are are possible for A and 16 for B, out of which the sum (0-9) (10 nos) are valid.

Invalid combinations = Total combinations − valid combinations

$= 16 \times 16 - (10 \times 10)$

$= 156$

59. 2 BIT COMPARATOR

⇒ Now, say we have to compare unsigned numbers. $[(00 < 10)][(10 > 01)]$ $(11 = 11)$

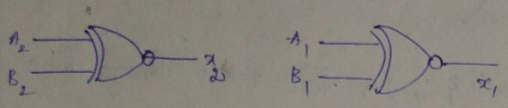
⇒ Exclusive-NOR is going to act as a Bit comparator. (for 2 bits)

A		B		A=B	A<B	A>B
A_2	A_1	B_2	B_1			
↓	↓	↓	↓			
MSB	LSB	MSB	LSB			

XNOR

a	b	a⊙b
0	0	1
0	1	0
1	0	0
1	1	1

Now, the XNOR produces 1 iff both a,b are same, now apply the concept to comparators.



This diagrams represent $A=B$ of.

case(1) $A=B$: if $x_1=1$ and $x_2=1$ i.e $x_1 \cdot x_2 = 1$

case(2) $A>B$: if $(A_2 > B_2)$ or $(A_2 = B_2$ and $A_1 > B_1)$ then $A>B$

$$F = A_2\bar{B}_2 + x_2 A_1 \bar{B}_1$$ if $f=1$ then $A>B$

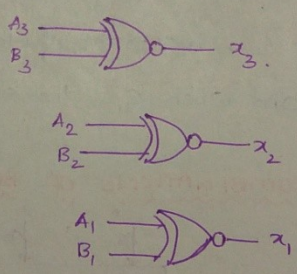
case(3): $A<B$: if $(A_2 < B_1)$ or $(A_2 = B_2$ and $A_1 < B_1)$

$$F = \bar{A}_2 B_1 + x_2 \bar{A}_1 B_1$$ = if $A<B$.

60. 3,4 BIT COMPARATORS

$A = A_3 A_2 A_1$

$B = B_3 B_2 B_1$



case(1): $A=B$ $\quad x_3 \cdot x_2 \cdot x_1 = 1$

case(2): $(A>B)$: $A_3\bar{B}_3 + x_3 \cdot A_2\bar{B}_2 + x_3x_2A_1\bar{B}_1 = 1$

case(3): $(A<B)$: $\bar{A}_3 B_3 + x_3\bar{A}_2 B_2 + x_3x_2\bar{A}_1 B_1 = 1$

(2)

$A = A_4\ A_3\ A_2\ A_1$

$B = B_4\ B_3\ B_2\ B_1$

$\downarrow\ \downarrow\ \downarrow\ \downarrow$

$x_4\ x_3\ x_2\ x_1$

(i) $A=B$: $x_4x_3x_2x_1 = 1$

(ii) $A>B$: $A_4\bar{B}_4 + x_4A_3\bar{B}_3 + x_4x_3A_2\bar{B}_2 + x_4x_3x_2A_1\bar{B}_1$

(iii) $A<B$: $\bar{A}_4B_4 + x_4\bar{A}_3B_3 + x_4x_3\bar{A}_2B_2 + x_4x_3x_2\bar{A}_1B_1$

If there are 2 n-bit numbers, then the no of combinations in which $A=B$ are 2^n

$$A>B = \frac{2^{2n}-2^n}{2}$$

$$A<B = \frac{2^{2n}-2^n}{2}$$

1- INTRO

→ The

on pr

→ The pre

new

presen

Inputs

2. LATCH

→ A flipp

→ A memo

stored

be step

→ The basi

→ It has

called

3. SR - FLIP

S

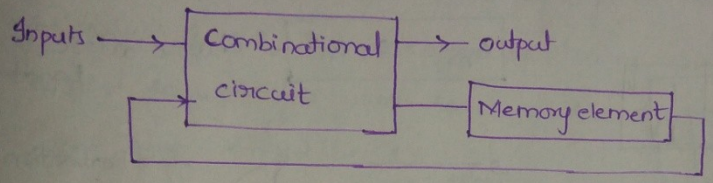
R

4. SEQUENTIAL CIRCUITS

1. INTRODUCTION TO SEQUENTIAL CIRCUITS.

→ The external o/p of a sequential circuit depends on external i/p's and on present contents of memory element

→ The present contents of the memory elements are called present state and the new contents of memory elements are obtained by taking external Inputs and present state This is called Next state.



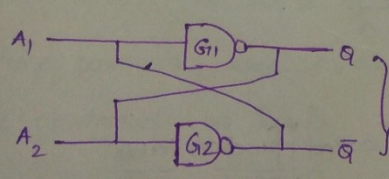
2. LATCH AND FLIP FLOP

→ A Flipflop is going to store 1 bit of information

→ A memory element is some medium in which one bit of information can be stored or retained until necessary, and there after its contents can be replaced by New values.

→ The basic binary or digital memory circuit is known as flipflop.

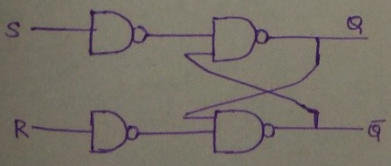
→ It has 2 stable states which are known as '1' or '0'. so it is called called bistable multi-vibrator.



LATCH (OR) LOCKING.

(Guessing)

3. SR - FLIPFLOP



⇒ Let the present state be represented by Q_n and the next state be represented by Q_{n+1}

$$Q_{n+1} = F(S, R, Q_n)$$

↓

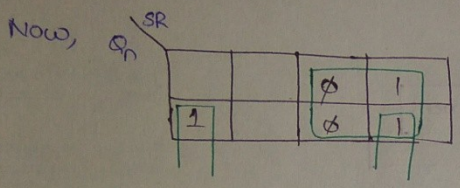
The o/p is not just a func of the present Inputs but also previous outputs.

S	R	Q_n	Q_{n+1}	
0	0	0	0	latch
0	0	1	1	latch
0	1	0	0	Reset
0	1	1	0	Reset
1	0	0	1	set
1	0	1	1	set
1	1	0	ϕ	invalid. (o/p is irrespective of the values of Q and $\bar{Q}$)
1	1	1	ϕ	invalid. (o/p is irrespective of the values of Q and $\bar{Q}$)

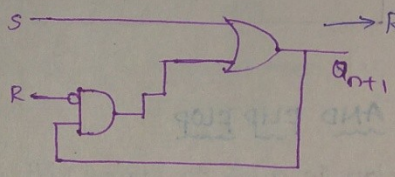
Charectaristic table

⇒ We can use the SR-flipflop in only 3 modes ⇒ latch, set, Reset modes.

Now,



$$Q_{n+1} = S + Q_n\bar{R}$$ = charectaristic Equation



→ Realisation of SR-FF.

→ Excitation table represents the present output and what is the output that you are expecting in the next state and what are the combinations that you should provide to get the o/p exact of next state that you have assumed.

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

→ R can be '0' or '1' so {'ϕ'}

→ minimised charectaristic table is called "function table"

S	R	Q_{n+1}
0	0	Q_n
1	0	1
0	1	0
1	1	ϕ

4. CLOCK

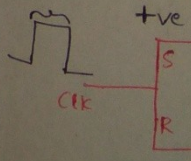
→ Now, w
some
then th
we sho
not le

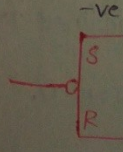
→ clock va

→ whever
the valu
enabled
clock

(p

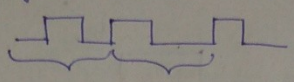
⇒ Edge trig
only on
the o/p



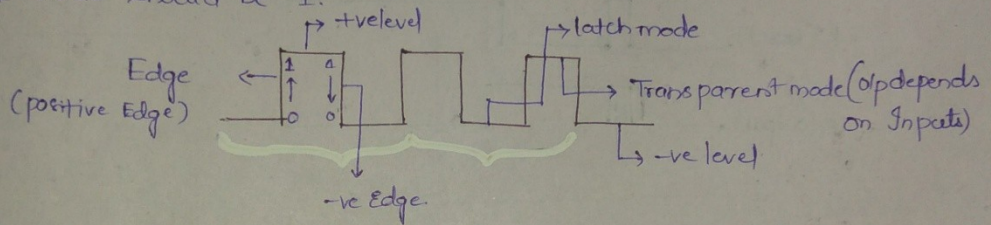


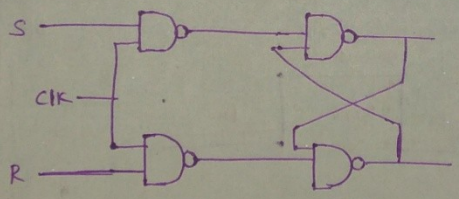
4. CLOCKED FLIPFLOPS

→ Now, what happens in general with SR-flipflop/flipflop is, because of some external disturbances the i/p symboll will suddenly fluctuate and then the output will change. and so what we are supposed to do is we should make the Inputs work only during some time and we should not let it change during the other time.

→ clock varies b/w 0 and 1 

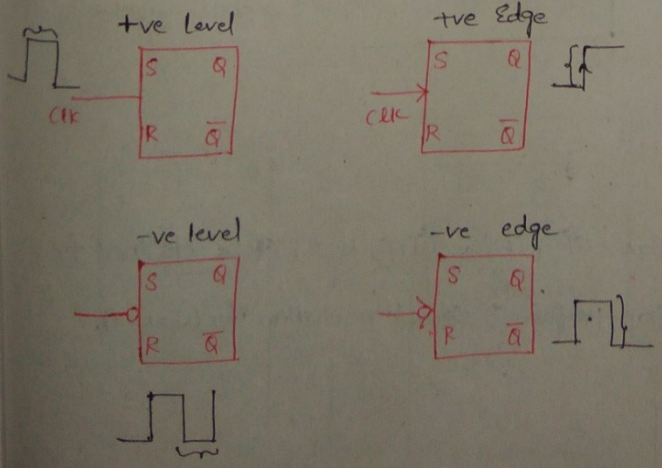
→ whever clock is '0' then the o/p of NAND gates will be 1 and 1 irrespective of the values of S,R it is as good as having S,R as 0,0 ⇒ latch mode is enabled. so, when you want to change the o/p depending on S,R then the clock value should be '1'.





⇒ positive level triggered flipflop.

⇒ Edge triggered means the output of the flipflop depends on the Input only on some particular edge so, when we trigger that particular edge then the o/p of FF depends on Input.



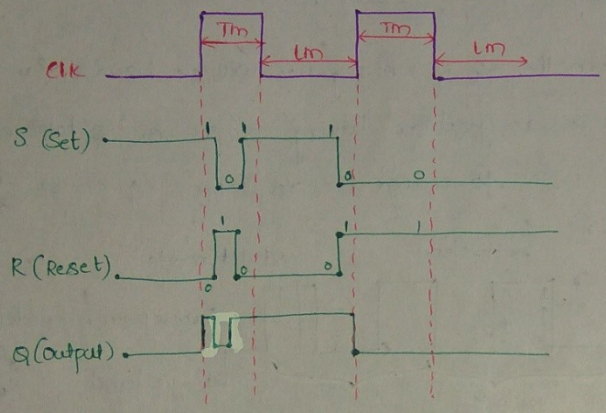
5. POSITIVE LEVEL TRIGGERD

Now, lets see how the positive level triggered FF react to the inputs.

⇒ In SR-FF 'S' means set and 'R' means Reset. The reason is whenever 'S' is '1' the o/p is always going to be one, and whenever R=1 the output is always going to be Zero. and when both S&R=0, then o/p will be in latch mode

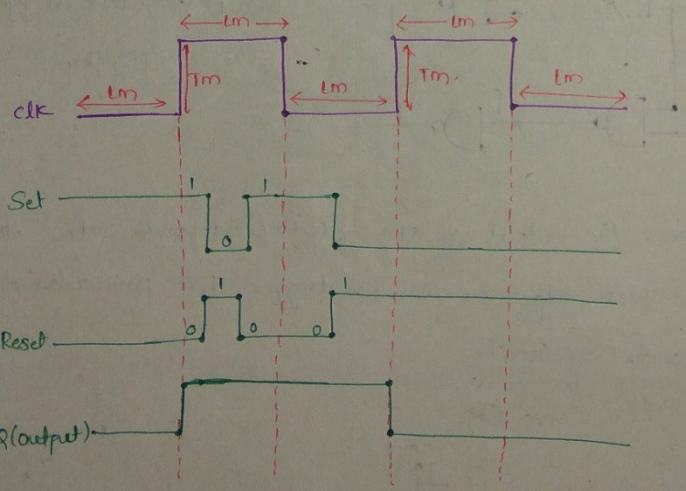
Tm= Transparent mode

Lm= Latch mode



⇒ Assume the o/p is present at '0' level.

6. EDGE TRIGGERED



Tm= Transparent mode

7. JK- FLIP FLOP

⇒ The J-K - flipflop is same as SR flipflop (J=S, K=R) It is defined for J=K=1 also and the flip-flop perfoorms complementation for (J=K=1).

J	K	Q_n
0	0	Q_n
1	0	1
0	1	0
1	1	$\overline{Q_n}$

Chare ctaristic

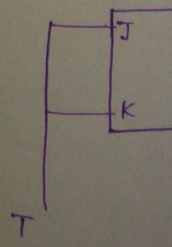
J	K	Q_n	
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Excitation tabl

Q	Q_{n+1}
0	→ 0
0	→ 1
1	→ 0
1	→ 1

8. T- FLIP FLOP

T- Flipflop = To

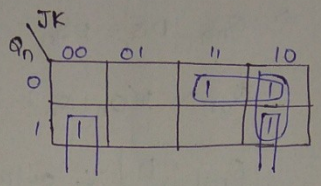


J	K	Q_{n+1}
0	0	Q_n
1	0	1
0	1	0
1	1	$\overline{Q_n}$

function table for J-K- flipflop.

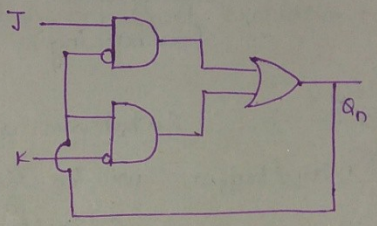
Characteristic Table:

J	K	Q_n	Q_{n+1}	
0	0	0	0	same as SR-FF
0	0	1	1	
0	1	0	0	
0	1	1	0	
1	0	0	1	
1	0	1	1	
1	1	0	1	Complementation.
1	1	1	0	



$F \equiv J\overline{Q_n} + \overline{K}Q_n$

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

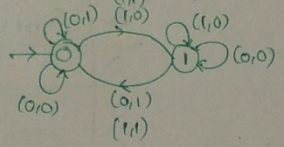


Excitation table

Q :	Q_{n+1}	J	K
0	→ 0	0	∅
0	→ 1	1	∅
1	→ 0	∅	1
1	→ 1	∅	0

State diagram

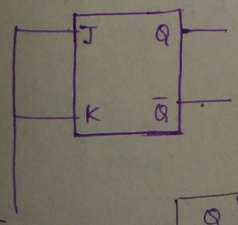
⇒ The 2 states of flipflop are {0,1}



look at function table and draw.

8. T- FLIP FLOP

T-flipflop = Toggle flipflop (This flip flop is either latching or complementing so it is called Toggling)



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

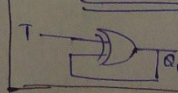
function table

T	Q_n	Q_{n+1}
0	0	$Q_n = 0$
0	1	$Q_n = 1$
1	0	$\overline{Q_n} = 1$
1	1	$\overline{Q_n} = 0$.

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

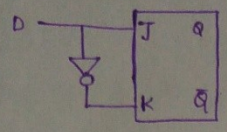
Exitation table

Characteristic table : $Q_{n+1} = T\overline{Q_n} + \overline{T}Q_n$



$\Leftarrow$ $Q_{n+1} = T \odot Q_n$

9. D-FLIP FLOP (Delay - Flipflop)



D	Q_{n+1}
0	0
1	1

= function table

Characteristic table

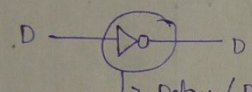
D	Q_n	Q_{n+1}
0	0	0
0	1	0
1	0	1
1	1	1

⇒

⇒ $Q_{n+1} \; D\bar{Q}_n + DQ_n$

$Q_{n+1} = D(Q_n + \bar{Q}_n)$

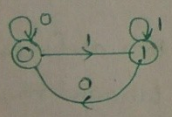
$\boxed{Q_{n+1} = D}$ ∴ output is directly depending on Input and there is no need of memory element at all (No feed backing).

⇒ 

↳ Delay / Buffer

(what are giving at the Input the same Input will be displayed as output after some delay so it is called delay flipflop

State diagram



Excitation table

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

10. EXAMPLE ON FLIP FLOP 1

The characteristic Equation of a flipflop is $Q_n = \bar{x}_1\bar{Q} + \bar{x}_2 Q$. Define the behaviour of this.

Function table

x_1	x_2	Q_n	
0	0	1	→ set state
0	1	$\bar{Q}$	→ Toggle
1	0	Q	→ latch mode
1	1	0	→ comptir Reset state

$Q_n = \bar{x}_1\bar{Q} + \bar{x}_2 Q$.

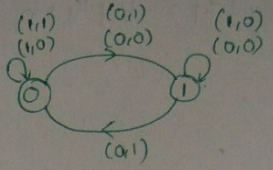
characteristic table

x_1	x_2	Q	Q_n
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Excitation table

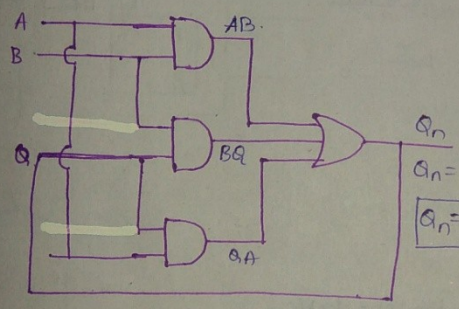
Q	Q_n	x_1	x_2
0	0	1	ϕ
0	1	0	ϕ
1	0	ϕ	1
1	1	ϕ	0

state diagram



11. EXAMPLE ON FLIPFLOP-2

consider the following realization. construct excitation table?



$Q_n = AB + BQ + QA$

$\boxed{Q_n = AB + Q(B+A)}$

Function table

A	B	Q_n	
0	0	0	→ Reset
0	1	Q	→ latch
1	0	Q	→ latch
1	1	1	→ seting

Excitation table

Q	Q_n	A	B
0	0	0	ϕ
0	1	1	1
1	0	0	0
1	1	1	ϕ
0	0	1	0
1	1	0	1

12. INTRODUCTION TO FLIP FLOP INTER CONVERSION

Conversion of given FLIPFLOP to another FLIPFLOP.

1) Get the chaorectoristic table of target FF
2) Replace the next state using excitation of the given FF
3) Obtain the expressions for the i/p of given FF and realise them.

Ex1: Convertion of J-K flipflop to T- flipflop

1) Target FF = T-flipflop ⇒ The chaorecteristic table =

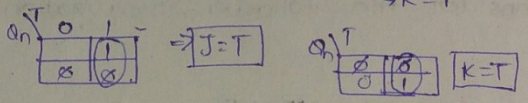
T	Q	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

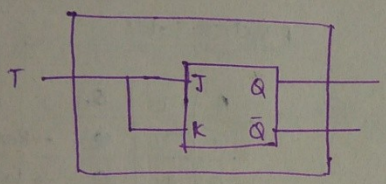
2) Replace the next state with excitation

T	Q	Q_{n+1}	J	K
0	0 → 0		0	∅
0	1 → 1		∅	0
1	0 → 1		1	∅
1	1 → 0		∅	1

Now, write the values of JK where the output changes to (0→0) (1→1) (0→1) (one → zero)

⇒ Now, chaorecteristic eqn for ⇒ J = T, K = T





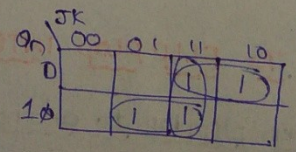
Ex2: convertion of T-FF into J-K FF

1) Target FF = J-k-flip flop ⇒ The chaorecteristic table =

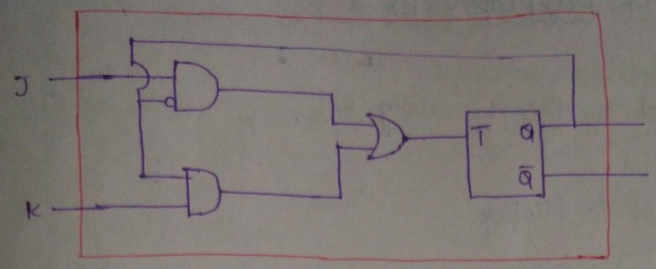
J	K	Q_n	Q_{n+1}	T
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	1

write the values of T for which output change from ($Q_n \rightarrow Q_{n+1}$) looking at chaorecteristic table of T.

chaorecteristic equation for T = Q_n



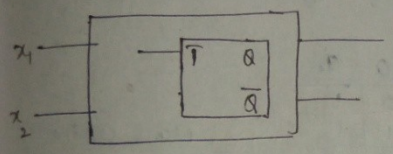
$T = J\bar{Q}_n + KQ_n$



13. INTER CONVERSION OF FLIPFLOPS - EXAMPLE-1

A new FF x_1x_2 has characteristic expression $Q_n = \bar{x}_1\bar{Q} + \bar{x}_2 Q$. Realise it using T- flipflop.

⇒ Realise it using T-flipflop means the given flipflop is 'T' and using T-flipflop you want to recognise new flipflop.



∴ charectaristic tables of x_1x_2 should be written first and then excitation tables of 'T' should be written.

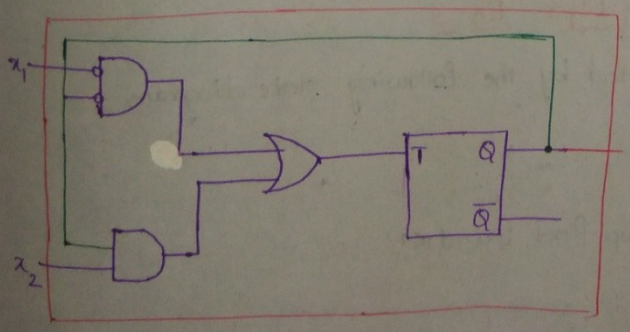
Function table

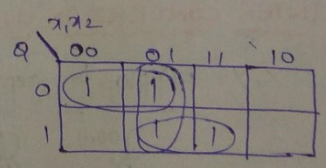
x_1	x_2	Q_n	
0	0	1	— set
0	1	$\bar{Q}$	— Toggle
1	0	Q	— latch
1	1	0	— Reset

Charectaristic table of x_1x_2Q.

Excitation table of 'T'.

x_1	x_2	Q	Q_n		T
0	0	0	1	set	1
0	0	1	One (1)	set	0
0	1	0	1	Toggle	1
0	1	1	0	Toggle	1
1	0	0	0	latch	0
1	0	1	1	latch	0
1	1	0	0	Reset	0
1	1	1	0	Reset	1





$T = \bar{x}_1\bar{Q} + x_2 Q$

14. INTER CONVERSION OF FLIPFLOPS - - Ex-2

$x_1 x_2 \rightarrow T$

$Q_n = \bar{x}_1 \bar{Q} + \bar{x}_2 Q$. } construct 'T' - flipflop using $x_1 x_2$.

The characteristic table of T- flipflop.

T	Q	Q_n	x_1	x_2
0	0	0	1	ϕ
0	1	1	ϕ	0
1	0	1	0	ϕ
1	1	0	ϕ	1

Excitation table for $x_1 x_2$

$\Rightarrow x_1 = \bar{T}$ $x_2 = T$

Now, inorder to get excitation table of $x_1 x_2$ / Q we should get function table of $x_1 x_2$

→ characteristic table of $x_1 x_2$ → Excitation table of $x_1 x_2$.

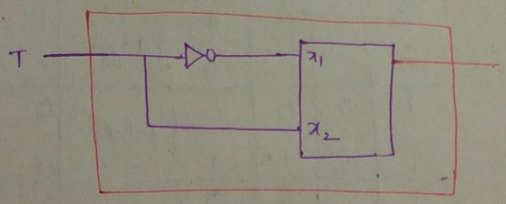
x_1	x_2	Q_n
0	0	1
0	1	$\bar{Q}$
1	0	Q
1	1	0

This are written by eqn $Q_n = \bar{x}_1 \bar{Q} + \bar{x}_2 Q$

→

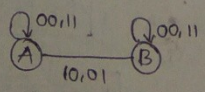
x_1	x_2	Q	Q_n
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

→ Now, write excitation table beside the characteristic table of $x_1 x_2$.

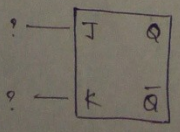


15. INTER CONVERSION OF FLIPFLOPS - Ex-3

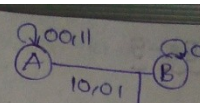
An, X,Y flipflop is represented by the following state diagram.



To realise it using J,K flipflop find J and K?

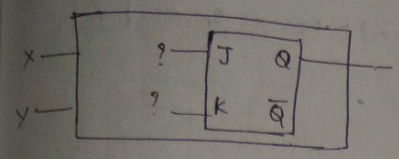


Given state diagram



↳ undirected Edge which means $\{A \xrightarrow{10} B\}$ $\{B \xrightarrow{10} A\}$

$\{A \rightleftarrows (B)\}$ $\{B \rightleftarrows A\}$

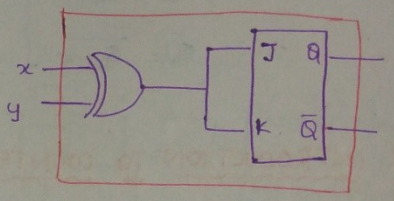


⇒ Now, find the chaorectaristic Equation of x,y and excitation table of J,K

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

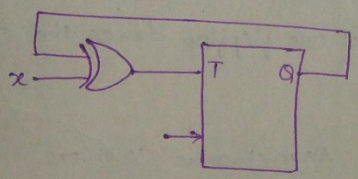
$J = \bar{x}y + x\bar{y} = x \oplus y$

$K = \bar{x}y + x\bar{y} = x \oplus y$



16. INTER CONVERSION OF FLIPFLOPS - Ex-1

What is the behaviour of following one-input flipflop 'X'?



a) D-FF b) T-FF

c) Inverted DFF d) Invested T-FF

⇒ This problem is nothing but implementation of X-flipflop using T-flipflop.

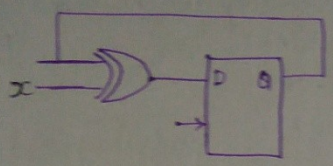
chaoreclaristic table of X.

x	Q_n	Q_{n+1}	$T = x \oplus Q_n$
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

⇒ The output Q_{n+1} is a combination of x, Q, T so first find $T = x \oplus Q_n$ and then find Q_{n+1}

$\therefore Q_{n+1} = x$. ∴ what ever you give as Input is displayed as output after some delay so it acts as Delayed flipflop.

17. INTER CONVERSION OF FLIPFLOPS - Ex-5



a) D-FF b) T-FF

c) Inverted D-FF d) Inverted D-FF

x	Q_n	Q_{n+1}	$D = Q_n \oplus x$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

when $x=0$, $Q_n = Q_{n+1}$

$x=1$, $Q_n = \overline{Q_{n+1}}$ } T flipflop.

18. INTRODUCTION TO COUNTERS

→ The counters are used to provide accurate timing and control signals.

→ These are of 2 types → Synchronous
→ Asynchronous

→ In synchronous counters all the flipflops respond to the same clock instances

→ In Asynchronous counters, the output of one flipflop drives the clock of another flipflop

→ Synchronous counters are faster than Asynchronous counters.

→ Due to simplicity of design, Asynchronous counters are used in IC fabrication

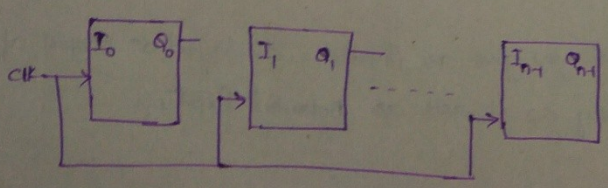
→ The simplified version of synchronous counter is called shift counters.

→ The basic element in shift counter is D-flipflop.

→ The Ring counter and Jhonson counter are further simplified version of shift counter.

19. ASYNCHRONUS AND SYNCHRONUS COUNTERS

Synchronus counter



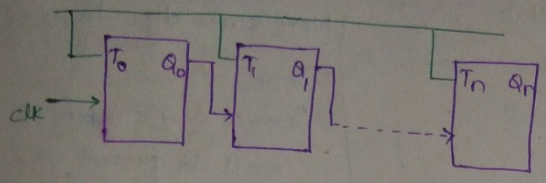
⇒ propagation delay in Synchronous counter is

$T_{pdsyn} = T_{FF} + T_{combinational}$

$T_{clk} \geq T_{pdsyn}$

⇒ T_{clk} means the amount of time we should wait before sending the next input signal.

Asynchronous counter:



⇒ propogation delay in Asynchronous Counter

$T_{pdasyn} = N * T_{FF} + T_{combinational}$

$T_{clk} \geq T_{pdasyn}$ T_{FF} = PD of each flipflop

20. SHIFT COUNTERS

Synchronous counter

⇒ $I_i = f(Q_0, Q_1 \cdots Q_{N-1})$ $0 \leq i \leq (N-1)$

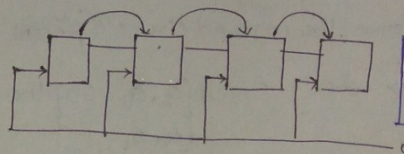
⇒ Its design is more complex, so simplifications are done as follows

1) $I_i = Q_{i-1}$ $\boxed{1 \leq i \leq N-1}$

$I_1 = Q_0$

$I_2 = Q_1$

$I_3 = Q_2$



Each FF receives the output of previous FF as input, then what about 1st FF?

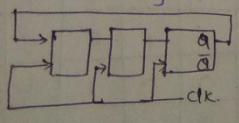
⇒ Here data is shifted from one flipflop to another FF so this is called shift counter (D-FF's are used)

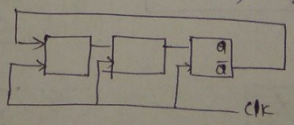
2) $I_0 = f(Q_{N-1})$

$I_0 = f(Q_{N-1})$

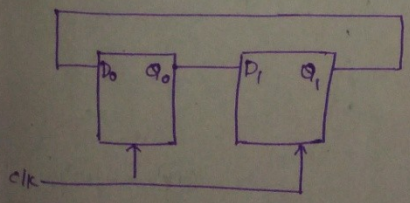
$I_0 = (Q_{N-1})$ (Ring counter)

$I_0 = (\bar{Q}_N - 1)$ (Johnson counter).





21. Mod-2 RING COUNTERS



⇒ Generally there are 2 types of Questions are asked i) They will give you the counter and ask you what it is counting

ii) They will tell you what you should count and ask you to design the counter.

⇒ First we should understand what are the states in which the FF (D-FF's) can be) if we observe there are 2FF and one FF is going to give Q_0 state and another FF is going to give Q_1 state (Here assume Q_0 = LSB, Q_1 = MSB) (12)

D_1	D_0	Present state Q_1	Present state Q_0	Next state Q_{1N}	Next state Q_{0N}
0	0	0	0	0	0
1	0	0	1	1	1
0	1	1	0	0	0
1	1	1	1	1	1

(2) (1) (3)

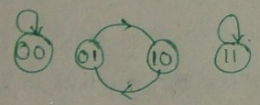
Next state depend on 2 things
1) Input
2) In case any combinational circuit is present what is the i/p to the combinational circuit.

Here Q_{0N} depends on D_0 and D_0 depends on Q_1 (look at diagram)

$\therefore \boxed{Q_{0N} = Q_1}$ $(Q_{0N} = D_0 = Q_1)$ $\Rightarrow (Q_{0N} = Q_1)$

⇒ Now, Q_{1N} is nothing but Input given at D_1 ⇒ $\boxed{Q_{1N} = D_1}$

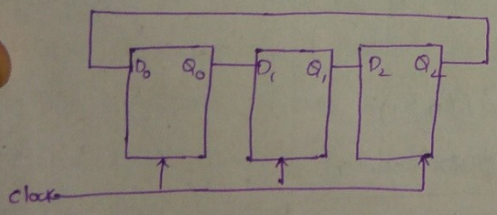
⇒ Now observe the marked circles (present state, Next state) and find how they behave, ⇒ when Input $\{Q_0 Q_1 = 00\}$ $\{Q_0 Q_1 = 11\}$ then they remain in same state



⇒ mod 2 counter (only 2 states are in counting).

⇒ using Ring counter we get "mod N" counters.

22. MOD 3 RING COUNTERS



$Q_{0N} = D_0 = Q_2 \Rightarrow \boxed{Q_{0N} = Q_2}$

$Q_{1N} = D_1 = Q_0 \Rightarrow Q_{1N} = Q_0$

$Q_{2N} = Q_{1N} \Rightarrow Q_{2N} = Q_0$

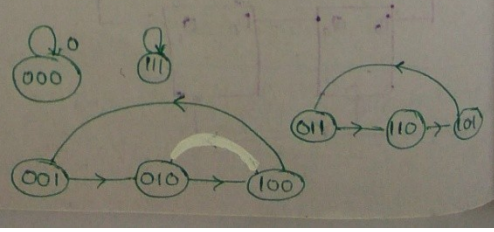
$\boxed{D_0 = Q_2}$ $\boxed{D_1 = Q_0}$ $\boxed{D_2 = Q_1}$

$\Rightarrow \boxed{Q_{0N} = D_0}$ $\boxed{Q_{1N} = D_1}$ $\boxed{Q_{2N} = D_2}$

LSB ↑ MSB ↑

Now, Q_0, Q_1, Q_2 are the states.

present Q_2	Q_1	Q_0	Next Q_{2N}	Q_{1N}	Q_{0N}	D_0	D_1	D_2
0	0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	1	0
0	1	0	1	0	0	0	0	1
0	1	1	1	1	0	0	1	1
1	0	0	0	0	1	1	0	0
1	0	1	0	1	1	1	1	0
1	1	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	1



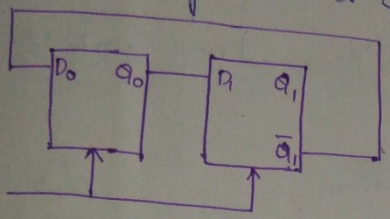
→ If a Ring c
→ In above
23. MOD 4 J
⇒ Twisted
D_0 Q_0
00
⇒
24. MOD 5 J
D_0 Q_0
CLK
$D_2 = Q_1$
$D_1 = Q_0$
$D_0 = \bar{Q}_2$

→ If a Ring counter has 'n' D-flipflops then it can perform "mod n" calculation

→ In above Ring counter it has 3-ff so it can perform mod3 operation.

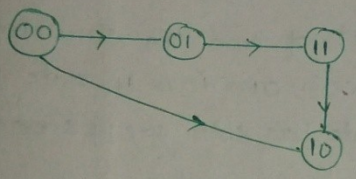
23. MOD 4 JOHNSON COUNTER

⇒ Twisted Ring counter or Johnson counter.



There are two states Q_0, Q_1 so.

Q_1	Q_0	Q_{1N}	Q_{0N}	D_0	D_1
0	0	0	1	1	0
0	1	1	1	1	1
1	0	0	0	0	0
1	1	1	0	0	1



= mod 4 counter

$Q_{1N} = D_1$

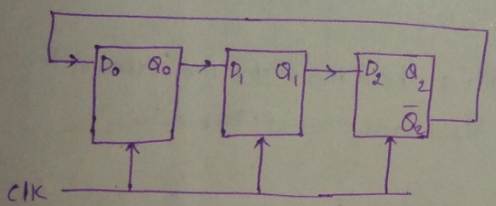
$Q_{0N} = D_0$

$D_0 = \bar{Q}_1$

$D_1 = Q_0$

⇒ [illegible]

24. MOD 6 JOHNSON COUNTER



Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}	D_2	D_1	D_0
0	0	0	0	0	1	0	0	1
0	0	1	0	1	1	0	1	1
0	1	0	1	0	1	1	0	1
0	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0
1	0	1	0	1	0	0	1	0
1	1	0	1	0	0	1	0	0
1	1	1	1	1	0	1	1	0

$D_2 = Q_1$

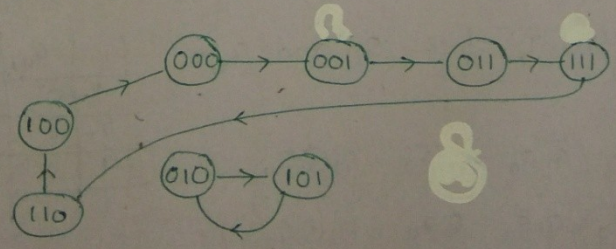
$D_1 = Q_0$

$D_0 = \bar{Q}_2$

$Q_{2N} = D_2$

$Q_{1N} = D_1$

$Q_{0N} = D_0$



∴ check the loop, it is mod6 counter.

25. MOD-4 GRAY COUNTER USING T-FF

Now, letus say how to design a counter with specifications.

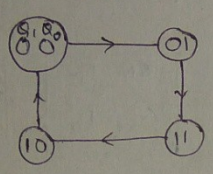
Designing a counter

1> Get the state table from state diagram

2> Identify flipflops to be used and replace the concerned next state using excitation table of associated flip flop.

3> Get the expressions for inputs and realise them.

Design a synchronus counter for the following using T-flipflops.



⇒ Here two flips are enough, at any instant of time the circuit will be in 00 (or) 01 (or) 10 (or) 11 state so each state has 2 bits so 2 FF's are enough

⇒ Now, these 2FF should be connected in such a way that when a clock signal is applied it should make any of the above transitions (00→01, 01→11, 11→10, 10→00).

⇒ Now, we need to find the Input combinations that lead to the above transitions (This is also called Excitation table), Now, first draw state table.

present state		Next state		Excitation (Inputs) → for 2 flipflops, two Inputs.	
Q_1	Q_0	Q_{1N}	Q_{0N}	T_1	T_0
0	0	0	1	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	1	1	0	1	0

(written by seeing at diagram)

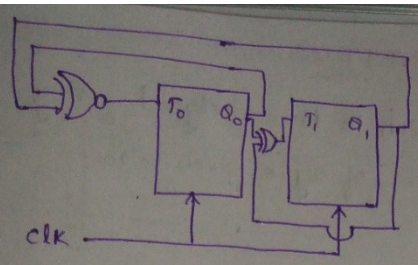
Now, to find T_1 check Q_1 and Q_{1N}

Q_1	Q_{1N}	T_1	
0	0	0	→ latch
0	1	1	
1	0	0	
1	1	1	

{ If o/p is complimented then T=1, o/p is same then T=0 {look at func table of T-flipflop}.

$$T_1 = \bar{Q}_1 Q_0 + Q_1 \bar{Q}_0 = Q_1 \oplus Q_0$$

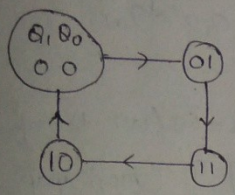
$$T_0 = \bar{Q}_1 \bar{Q}_0 + Q_1 Q_0 = Q_1 \odot Q_0$$



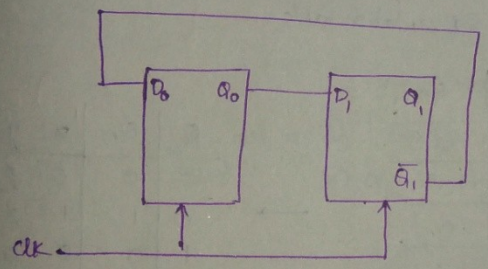
⇒ Mod 4 Gray-counter

26. MOD-4 -GRAY COUNTER USING D-FF

⇒ 2-flipflops are needed.



present state		Next state		Excitation (Inputs)	
Q_1	Q_0	Q_{1N}	Q_{0N}	D_1	D_0
0	0	0	1	0	1
0	1	1	1	1	1
1	0	0	0	0	0
1	1	1	0	1	0



$D_1 = \overline{Q_0} Q_1 + Q_1 Q_0 = Q_1$

$D_0 = \overline{Q_1}\,\overline{Q_0} + \overline{Q_1} Q_0 = \overline{Q_1}(Q_0 + \overline{Q_0})$

$= \overline{Q_1}$.

27. MOD 4 GRAY COUNTER USING 1D AND 1T FLIPFLOP

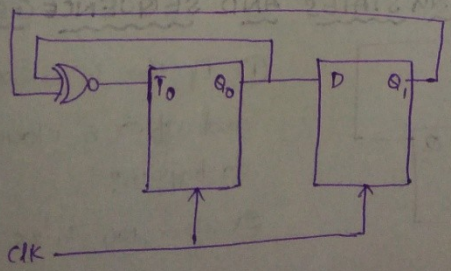
Same Question above

Q_1	Q_0	Q_{1N}	Q_{0N}	D_1	T_0
0	0	0	1	0	1
0	1	1	1	1	0
1	1	1	0	1	1
1	0	0	0	0	0

$D_1 = Q_0$

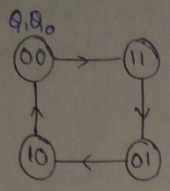
$T_0 = \overline{Q_1}\,\overline{Q_0} + Q_1 Q_0$

$\boxed{T_0 = Q_1 \odot Q_0}$



28. COUNTER USING TWO DIFFERENT FF'S

Consider the following state diagram which is to be designed using T- Flipflops for msB and xy for LSB. The behaviour of xy is given below:



x	y	Q_n
0	0	0
0	1	Q
1	0	$\bar{Q}$
1	1	1

↳ used for msB so we consider Q_1 and Q_{1N}

Q_1	Q_0	Q_{1N}	Q_{0N}	T_1	x	y
0	0	1	1	1	1	ø
1	1	0	1	1	ø	1
0	1	1	0	1	ø	0
1	0	0	0	1	0	ø

⇒ Now $x = \bar{Q}_1$ (use K-map)

$y = Q_1$ (use K-map)

Given functable → characteristic table → Excitation table

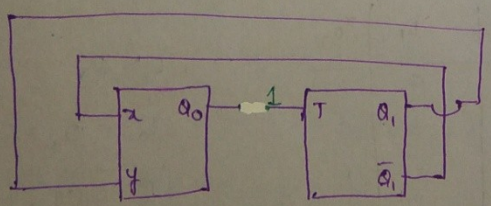
x	y	Q_n
0	0	0
0	1	Q
1	0	$\bar{Q}$
1	1	1

→

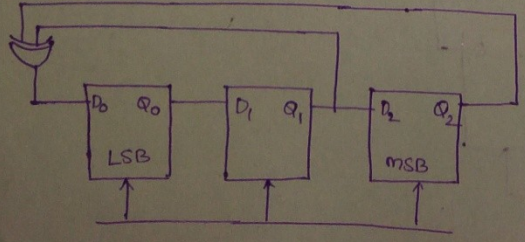
x	y	Q	Q_n
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

→

Q	Q_n	x	y
0	0	0	ø
0	1	1	ø
1	0	ø	0
1	1	ø	1



29. MODEL ON ANALYSIS COUNTING STATES AND SEQUENCE GENERATIONS



① If the initial state is 80 & so and after 5 clocks what is going to happen?

② After 100 clocks what will happen?

This all depends on what initial state is.

flipflops

rap)

ap)

y
0
0
0
1

RATIONS

is so & so

hat is going

t will happen?

is.

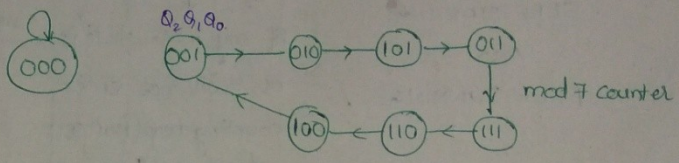
First lets analyse the state diagram.

Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	1	0

$Q_{0N} = D_0 = Q_1 \oplus Q_2$

$Q_{1N} = D_1 = Q_0$

$Q_{2N} = D_2 = Q_1$



① How many states are there is mod counter = 7.

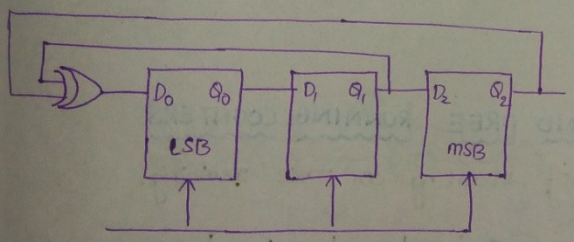
② Given Initial state = 011 then what happens after 4 clock cycles ⇒ state after 4 clk cycles = 001

③ Initial state = 001 ⇒ state after 10 clock cycles = 011

④ I tap the output at Q_2 ⇒ what is the output = 0010111 ← write Q_2 values in all states

⑤ o/p is tapped at Q_0 ⇒ (1011100) (1st bit of all states) → count Sequence.

30. DERIVING THE CLOCK FREQUENCY



$T_{FF} = 15ns.$, $T_{comb} = 5ns.$

which of the following clock frequency ensures proper counting?

a) 40MHz b) 60MHz c) 90mHz d) 300mhz

$T_{clk} \geq T_{FF} + T_{comb}$

$T_{clk} \geq 15ns + 5ns$

$T_{clk} \geq 20ns$

$\Rightarrow f_{clk} \leq \frac{1}{20 \times 10^{-9}}$ (Time and freq are inversely proportional)

$f_{clk} \leq 50mHz$

flipflops

rap)

ap)

y
0
0
0
1

RATIONS

is so & so

hat is going

t will happen?

is.

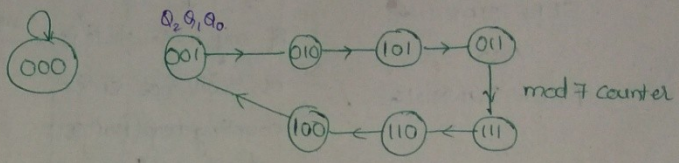
First lets analyse the state diagram.

Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	1
0	1	1	1	1	1
1	0	0	0	0	1
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	1	0

$Q_{0N} = D_0 = Q_1 \oplus Q_2$

$Q_{1N} = D_1 = Q_0$

$Q_{2N} = D_2 = Q_1$



① How many states are there is mod counter = 7.

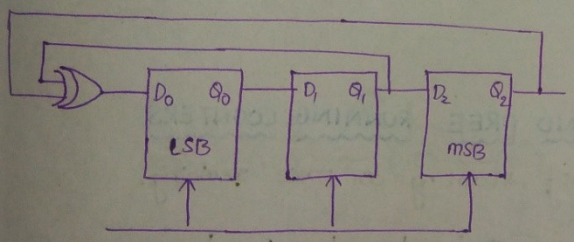
② Given Initial state = 011 then what happens after 4 clock cycles ⇒ state after 4 clk cycles = 001

③ Initial state = 001 ⇒ state after 10 clock cycles = 011

④ I tap the output at Q_2 ⇒ what is the output = 0010111 ← write Q_2 values in all states

⑤ o/p is tapped at Q_0 ⇒ (1011100) (1st bit of all states) → count Sequence.

30. DERIVING THE CLOCK FREQUENCY



$T_{FF} = 15ns.$, $T_{comb} = 5ns.$

which of the following clock frequency ensures proper counting?

a) 40MHz b) 60MHz c) 90mHz d) 300mhz

$T_{clk} \geq T_{FF} + T_{comb}$

$T_{clk} \geq 15ns + 5ns$

$T_{clk} \geq 20ns$

$\Rightarrow f_{clk} \leq \frac{1}{20 \times 10^{-9}}$ (Time and freq are inversely proportional)

$f_{clk} \leq 50mHz$

unting loop

tes in the

STARTING)

quare not
entering counting
loop.

state you
no. of clk
enter the

running".

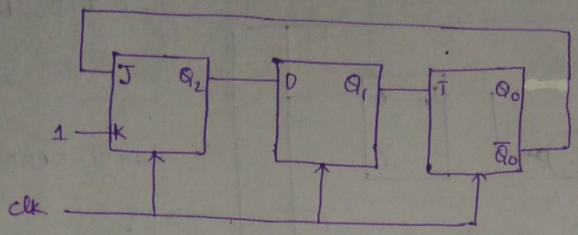
S.

g.

od3 counter

33. COUNTER USING 3 DIFFERENT FLIPFLOPS

Consider the following counter, Initially $Q_2Q_1Q_0 = 000$



Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}	J_0	D_1	T_2
0	0	0	1	0	0	1	0	0
0	0	1	0	0	1	0	0	0
0	1	0	1	0	1	1	0	1
0	1	1	0	0	0	0	0	1
1	0	0	0	1	0	1	1	0
1	0	1	0	1	1	0	1	0
1	1	0	0	1	1	1	1	1
1	1	1	0	1	0	0	1	1

$D_1 = Q_2$

$T_2 = Q_1$

$J_0 = Q_0$.

$Q_{2N} = ?? = JK$ ⟶

J	K	Q_{2N}
0	1	0
1	1	$\overline{Q_N}$

$= \overline{Q_0}$

⇒

$\overline{Q_0}$	K	Q_{2N}
0	1	0
1	1	$\overline{Q_N}$

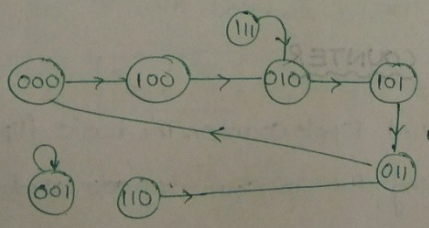
⇓

Q_0	K	Q_{2N}
1	1	0
0	1	$\overline{Q_N}$

$Q_{1N} = D_1 = Q_2$.

$Q_{0N} = Q_0 (T=0) \Rightarrow Q_0 (Q_1 = 0)$

$\overline{Q_0} (T=1) \Rightarrow Q_0 (Q_1 = 1)$



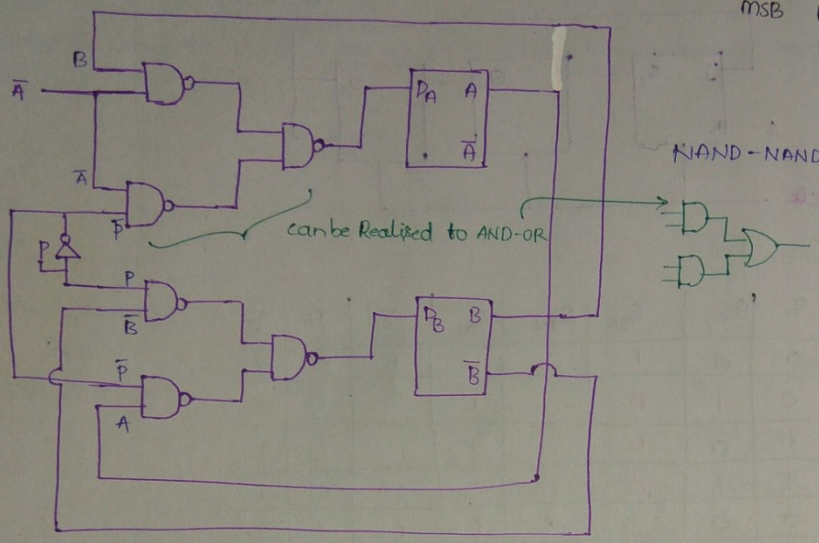
① what will be the states after 4 clocks? a) 000 b) 010 c) 011 d) 101

② Modulus of the Counter? a) 4 b) 5 c) 6 d) 7

34. EXAMPLES ON COMBINATIONAL CIRCUITS AND FLIPFLOPS

Consider the following counter, If P=0, the counting sequence of AB is

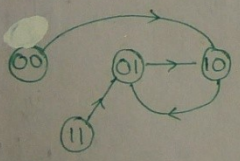
MSB LSB.



NAND - NAND = AND-OR

can be Realised to AND-OR

A	B.	A_n	B_n
0	0	1	0
0	1	1	0
1	0	0	1
1	1	0	1



{mod 2 counter}

$A_n = D_A$ } If we observe the CC,
$B_n = D_B$ } they can be Realised to AND-OR Realisation

$D_A = A\bar{B} + \bar{A}\bar{P} \Rightarrow D_A = A\bar{B} + \bar{A}\bar{P}$ $\{P=0 \Rightarrow \bar{P}(1+A) = \bar{P}\}$

$D_B = P\bar{B} + \bar{P}A \Rightarrow D_B = A$ //

$\Rightarrow$ $\boxed{B_n = A}$ $\boxed{A_n = \bar{A}B}$

35. INTRODUCTION TO ASYNCHRONOUS COUNTER

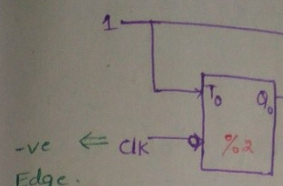
The asynchronous counters are also called Ripple counter. The basic flipflop is T and basic counting is binary up counting. The up counters are used for implementing incrementation. Due to simpler design, Asynchronous counters are preferred in IC counter fabrication.

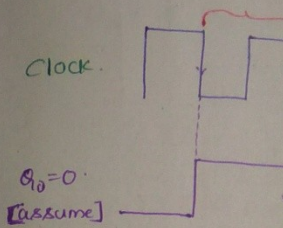
$\Rightarrow$ IC 7490 is a decade counter i.e (%10) counter.

$\Rightarrow$ IC 7492 is a hexadecimal counter i.e (%16) counter.

$\Rightarrow$ The o/p of one flipflop is going to be clock to the next FF.

36. MOD 8 UP COUN





$Q_1 = 0$
(assume)

$Q_2 = 0$

$\Rightarrow$ 3! % 2 counters form

Now, consider 'Q_0' it

it T-FF so 'T' will

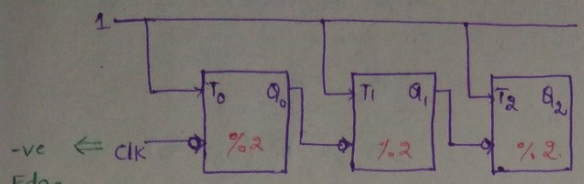
$\therefore$ $\boxed{Q_{0N} =}$

$\Rightarrow$ Now, Q_1 is changed

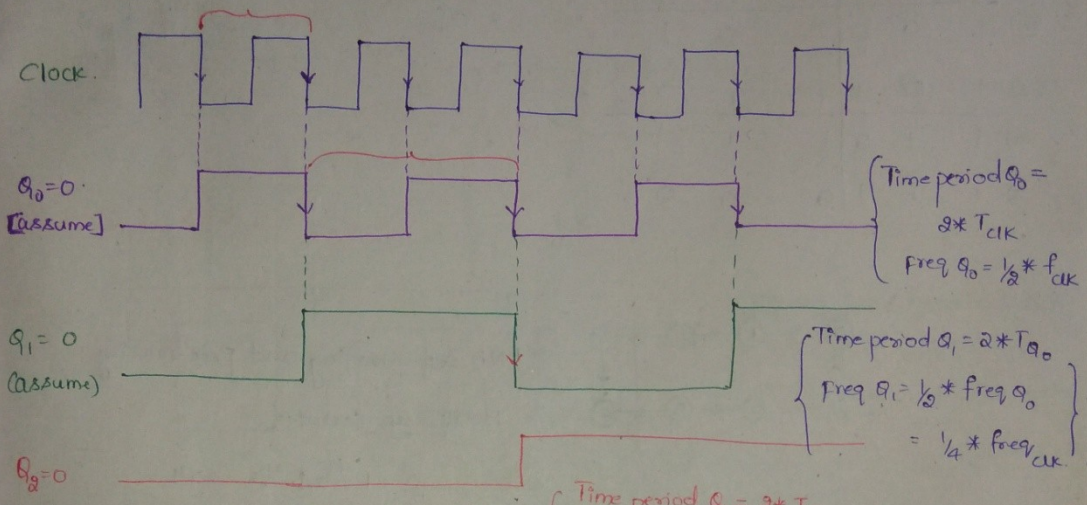
$\boxed{Q_{1N} = \bar{Q}_1}$ (

$\Rightarrow$ Now, $Q_{2N} = \bar{Q}_2$ (

36. MOD 8 UP COUNTER



Assume $Q_0 = Q_1 = Q_2 = 0$. (Initially)



⇒ 3 / %2 counters form %8 counters.

Now, consider 'Q_0' it is having Input 1 and it it –ve edge triggered and it it T-FF so 'T' will become 1 and it will Toggle ∴ The new o/p Q_{0N} will be $\overline{Q_0}$ for every clock

∴ $\boxed{Q_{0N} = \overline{Q_0}}$ for every clock

⇒ Now, Q_1 is changed / Q_1 depends on how Q_{0N} is providing the clock signal.

$\boxed{Q_{1N} = \overline{Q_1}}$ (when there is –ve edge from Q_0 $(1 \to 0)$).
⇒ Q_0 will toggle.

⇒ Now, $Q_{2N} = \overline{Q_2}$ $(Q_1 : 1 \to 0)$.

∴
$Q_{0N} = \overline{Q_0}$
$Q_{1N} = \overline{Q_1}$ $(Q_0 : 1 \to 0)$
$Q_{2N} = \overline{Q_2}$ $(Q_1 : 1 \to 0)$

→ whenever a Que like above is given construct the equation and proceed.

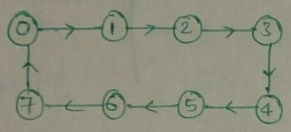
Present state			Next state		
Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

Above equations,

$Q_{0N} = \overline{Q_0}$

$Q_{1N} = \overline{Q_1}$ $(Q_0 : 1 \to 0)$

$Q_{2N} = \overline{Q_2}$ $(Q_1 : 1 \to 0)$

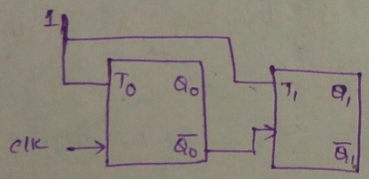


⇒ Both self starting and free running

⇒ Mod 8 up counter

$0 \to 1 \xrightarrow{+1} 2 \xrightarrow{+1} 3 \xrightarrow{+1} 4 \cdots$

37. MOD 4 UP COUNTER



1) What type of counting it is performing and what is the type of the counter?

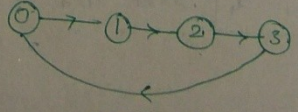
$Q_{0N} = \overline{Q_0}$ (for every clock)

$Q_{1N} = \overline{Q_1}$ $(\overline{Q}\ : 0 \to 1)$

$\cong \downarrow$

$(Q_0 : 1 \to 0)$

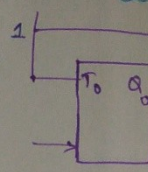
Q_1	Q_0	Q_{1N}	Q_{0N}
0	0	0	1
0	1	1	0
1	0	1	1
1	1	0	0



mod 4 upcounter

=

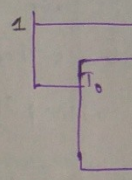
38. MOD 4 DO



$Q_{0N} = \overline{Q_0}$

$Q_{1N} = \overline{Q_1}$ (

39. MOD 8 RA



$Q_{0N} = \overline{Q_0}$

$Q_{1N} = \overline{Q_1}$ $(\overline{Q_0}$:

$Q_{2N} = \overline{Q_2}$ $(Q_1$:

positive e

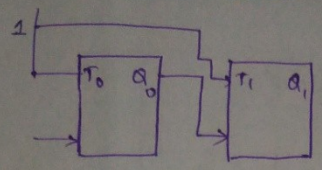
Symbol (→

40. APPLICATION

1) shift Regist

2) counters →

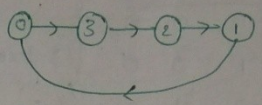
38. MOD 4 DOWN COUNTER



$Q_{0N} = \overline{Q_0}$

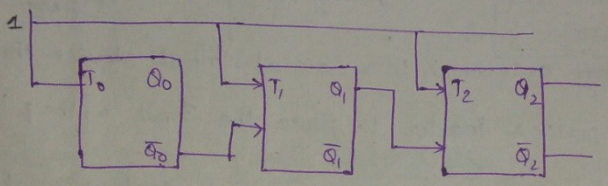
$Q_{1N} = \overline{Q_1}$ $(Q_0 : 0 \to 1)$.

Q_1	Q_0	Q_{1N}	Q_{0N}
0	0	1	1
0	1	0	0
1	0	0	1
1	1	1	0



mod4 down counter

39. MOD 8 RANDOM COUNTER



If the initial state $Q_2Q_1Q_0$ = (101) (5), what will be the state after 4 clock cycles?

= 001 = 1

$Q_{0N} = \overline{Q_0}$

$Q_{1N} = \overline{Q_1}$ $(\overline{Q_0} : 0 \to 1) \Rightarrow (Q_0 : 1 \to 0)$

$Q_{2N} = \overline{Q_2}$ $(Q_1 : 0 \to 1)$

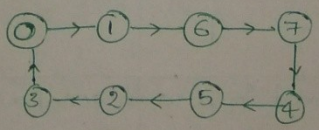
↓

In diagram it is positive edge so $(0 \to 1)$

↓

Symbol ($\to$) = +ve edge

Q_2	Q_1	Q_0	Q_{2N}	Q_{1N}	Q_{0N}
0	0	0	0	0	1
0	0	1	1	1	0
0	1	0	0	1	1
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	0	1	0
1	1	0	1	1	1
1	1	1	1	0	0



Self starting and Free Running.

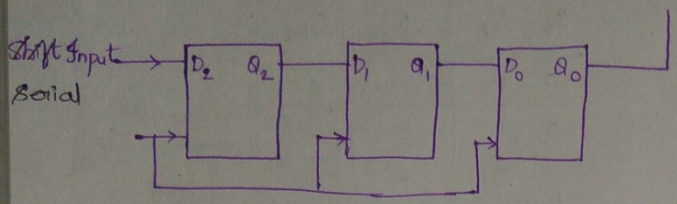
40. APPLICATIONS OF FLIP FLOPS

1) Shift Register → used as "sequential memory". Ex: Accumulator in Micro processors.

2) Counters → (1) used to count no. of pulses.

(2) used as frequency divider

41. 3-BIT SHIFT REGISTERS (It requires 3 D-flipFlops).

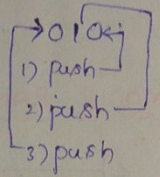


101

clk	SI/P	Q_2	Q_1	Q_0
0	–	0	0	0
1	1	1	0	0
2	0	0	1	0
3.	1	1	0	1

3 clock pulses.
Entire for pushing Input on the shift Register.

Say I want to push 101 onto shift Register
1> push 1
2> push 0
3> push 1



⇒ we have n-bit shift Registers which is a serial Input device then we require 'n' clock pulses inorder to place the Input on the flipflop

⇒ In case of shift Registers 'n' clk pulses are reqd to place 'n'-bits onto the shiftRegister

clk	Q_2	Q_1	Q_0	o
0	1	0	1	1
1	0	1	0	0
2	0	0	1	1

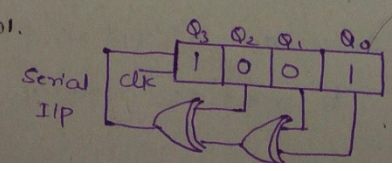
∴ To retrieve the o/p we need 2 clock pulses (~~1st one is counted in the pushing I/p on Register~~).

∴ Totally 3 clk + 2 clk pulses are needed for 3-bit serial Input and serial output.

∴ In general, for 'n' bit serial Input and serial output we need "2n-1" clock pulses (n for serial I/p, (n-1) for serial o/p).

42. EXAMPLE 1 ON SHIFT RIGHT REGISTER

In the following Right shift Register, determine the no. of clocks required to bring it to the initial state of 1001.



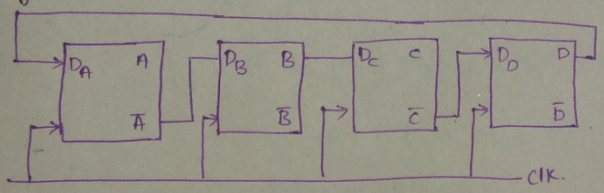
Clk.	SIp	States Q_3 Q_2 Q_1 Q_0	
0	–	1 0 0 1	→ $SIp = (Q_2 \oplus Q_1 \oplus Q_0)$ at previous clock
1	1	1 1 0 0	$= (0 \oplus 0 \oplus 1) = 1$
2	1	1 1 1 0	→ $SIp = (Q_2 \oplus Q_1 \oplus Q_0) = (1 \oplus 0 \oplus 0) = 1$
3	0	0 1 1 1	→ $SIp = 1 \oplus 1 \oplus 0 = 0$
4	1	1 0 1 1	→ $SIp = 1 \oplus 1 \oplus 1 = 1$
5	0	0 1 0 1	→ $SIp = 0 \oplus 1 \oplus 1 = 0$
6	0	0 0 1 0	→ $SIp = 1 \oplus 0 \oplus 1 = 0$
7	1	1 0 0 1	→ $SIp = 0 \oplus 1 \oplus 0 = 1$

∴ 6 | 7 clocks are needed.

43. EXAMPLE 2 ON SHIFT REGISTER

Initial value of ABCD = 0000, then what are the sequence of no's the following circuit is counting.

→ LSB
→ MSB.



Inputs:

$D_A = D$.

$D_B = \bar{A}$

$D_C = B$.

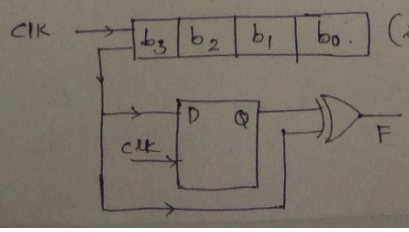
$D_D = \bar{C}$

Clk	A	B	C	D	
0	0	0	0	0	0
1	0	1	0	1	5
2	1	1	1	1	15
3	1	0	1	0	10
4	0	0	0	0	0
5	0	1	0	1	5

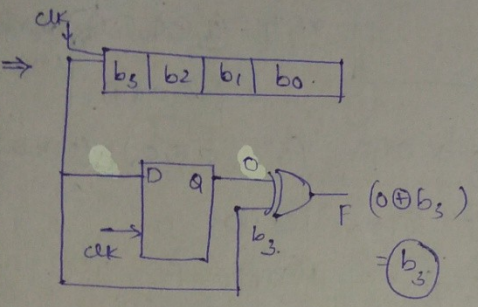
∴ 0, 5, 15, 10, 0, 5

44. BINARY TO GRAY CONVERTOR

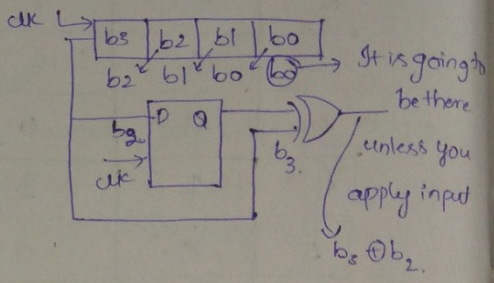
Determine the function of the following circuit



Initially assume the output of D-flipflop is '0'. Now, b_3 is given to the output of D-FF ∴ The output of XOR Gate will be $0 \oplus b_3 = b_3$



After applying 1 clock:



continuing this procedure, b_3, $b_3 \oplus b_2$, $b_2 \oplus b_1$, $b_1 \oplus b_0$ ...

⇒ Graycode conversion