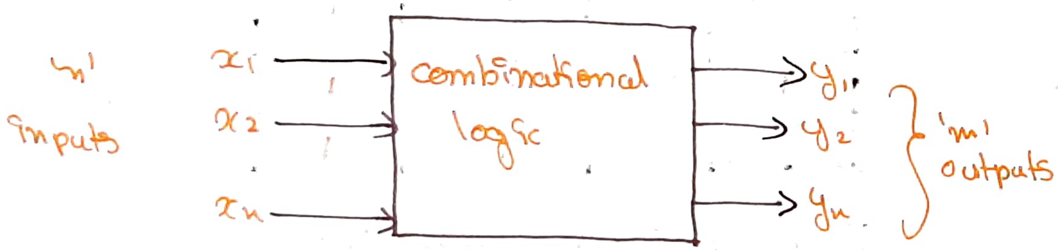
Combinational logic circuit:->

-> output ~~depen~~ dependent on combinational of input variables

-> it doesn't use any memory.

-> it has n-inputs & m-outputs.



Analysis of combinational logic

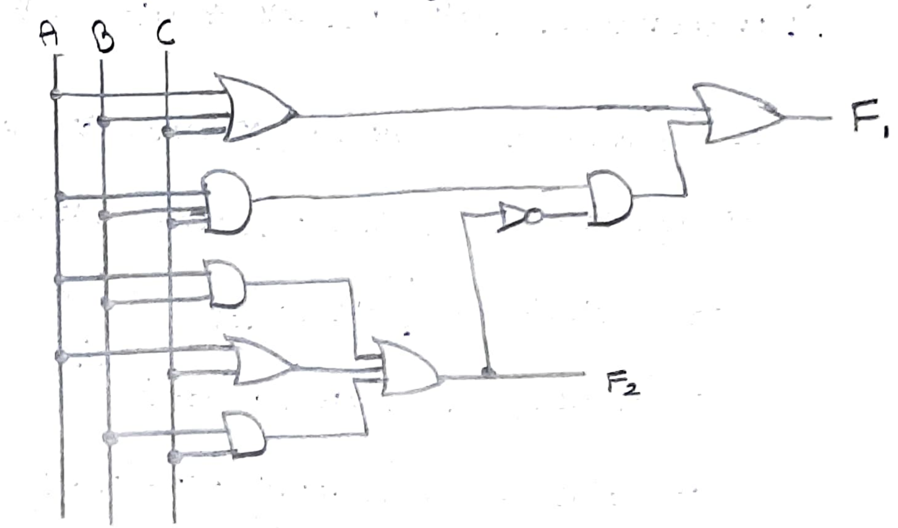
-> logic diagram (given in question)

-> boolean ~~a~~ expression

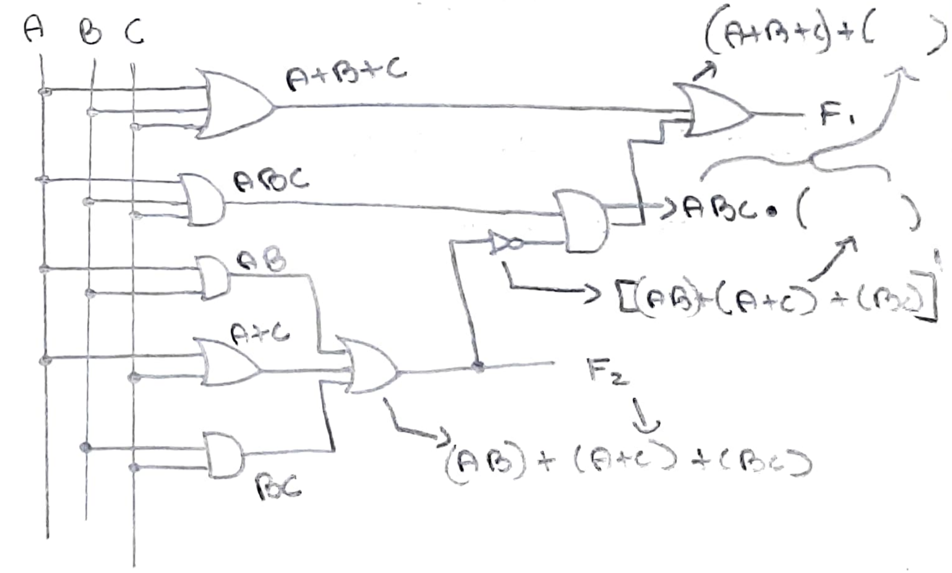
-> truth table

-> values of sop & pos.

analyze the logic diagram.



ans:->



$$F_1 = A+B+C + \left(ABC \cdot [(AB)+(A+C)+(BC)]\right)'$$

$$= (A+B+C) + \left(ABC \cdot [B(A+C) + (A+C)]\right)'$$

$$= (A+B+C) + \left(ABC \cdot [(A+C)(B+1)]\right)'$$

$$\Rightarrow (A+B+C) + [ABC \cdot (A+C)']$$

$$= (A+B+C) + [ABC \cdot (A'C')]$$

$$= (A+B+C) + \underbrace{[\underbrace{(A \cdot A')}_{0} \cdot B \cdot \underbrace{(C \cdot C')}_{0}]}_{0}$$

$$= A+B+C \;//.$$

$$F_2 = AB + (A+C) + (BC)$$

$$= A+C \;//$$

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

$F_1(A,B,C) = \Sigma(1,2,3,4,5,6,7)$

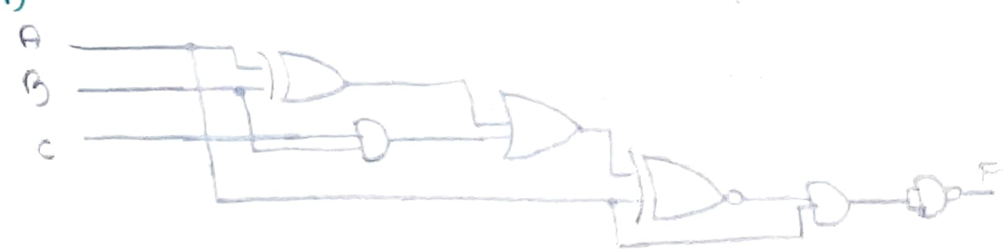
$F_1(A,B,C) = \Pi(0)$

$F_2(A,B,C) = \Sigma(1,3,4,5,6,7)$

$F_2(A,B,C) = \Pi(0,2)$

Sums to solve

1)



Solution available on github & telegram channel.

Design of combinational circuit

→ truth table (for question)

→ do Kmap for each output

→ draw logic diagram.

Design a combinational circuit with three inputs x, y & z & three outputs A, B & C. when the binary input is 0, 1, 2, 3, the binary output is two greater than the input. when the binary input is 4, 5, 6, 7 the binary output is three less than the input.

	Input			Output		
	X	Y	Z	A	B	C
0	0	0	0	0	1	0
1	0	0	1	0	1	1
2	0	1	0	1	0	0
3	0	1	1	1	0	1
4	1	0	0	0	0	1
5	1	0	1	0	1	0
6	1	1	0	0	1	1
7	1	1	1	1	0	0

Kmap for A

X \ YZ	y'z' 00	y'z 01	yz 11	yz' 10
X' 0			1	1
X 1			1	

$A = YZ + X'Y$

Kmap for B

X \ YZ	y'z'	y'z	yz	yz'
X'	1	1		
X		1		1

$B = x'y' + y'z + xyz'$

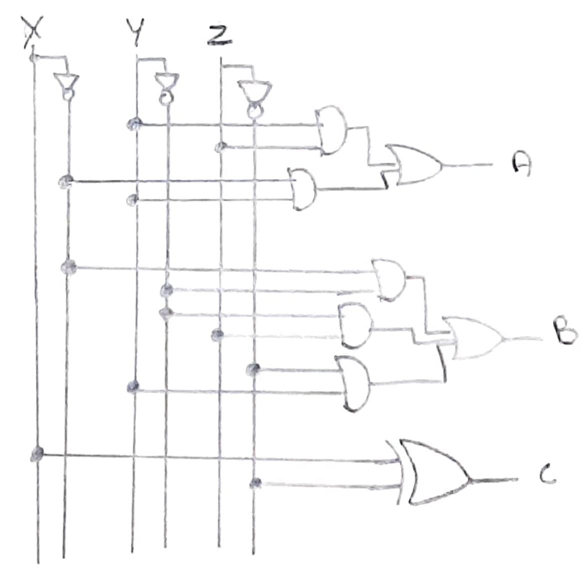
Kmap for C

X \ YZ	y'z'	y'z	yz	yz'
X'		1	1	
X	1			1

$C = x'z + xz' = x \oplus z$

[$A'B + AB' = A \oplus B$

$A'B' + AB = (A \oplus B)$[illegible]



Code Conversion :→

converting one set of code to another.

Convert 4bit binary to gray code.

4bit binary	Gray code
A B C D	P Q R S
0 0 0 0	0 0 0 0
0 0 0 1	0 0 0 1
0 0 1 0	0 0 1 1
0 0 1 1	0 0 1 0
0 1 0 0	0 1 1 0
0 1 0 1	0 1 1 1
0 1 1 0	0 1 0 1
0 1 1 1	0 1 0 0
1 0 0 0	1 1 0 0
1 0 0 1	1 1 0 1
1 0 1 0	1 1 1 1
1 0 1 1	1 1 1 0
1 1 0 0	1 0 1 0
1 1 0 1	1 0 1 1
1 1 1 0	1 0 0 1
1 1 1 1	1 0 0 0

Kmap for P

AB \ CD	C'D' 00	C'D 01	CD 11	CD' 10
A'B' 00				
A'B 01				
AB 11	1	1	1	1
AB' 10	1	1	1	1

$P = A$

Kmap for Q

AB \ CD	C'D'	C'D	CD	CD'
A'B'				
A'B	1	1	1	1
AB				
AB'	1	1	1	1

$Q = A'B + AB'$

$= A \oplus B$

Kmap for R

AB \ CD	C'D'	C'D	CD	CD'
A'B'			1	1
A'B	1	1		
AB	1	1		
AB'			1	1

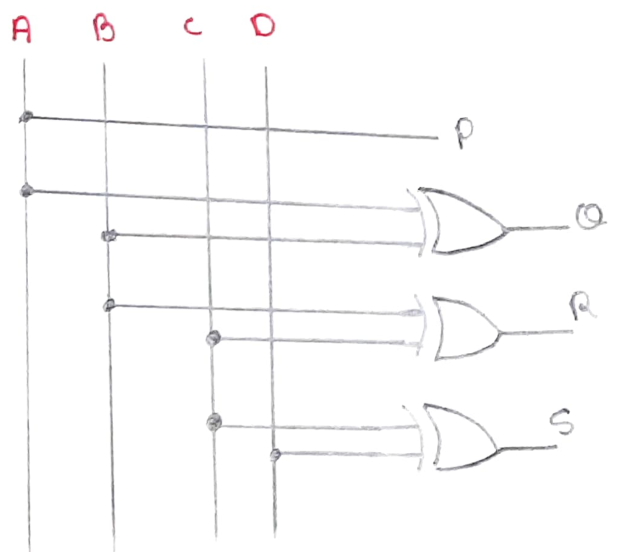
$R = BC' + B'C$

$= B \oplus C$

Kmap for S

AB \ CD	C'D'	C'D	CD	CD'
A'B'		1		1
A'B		1		1
AB		1		1
AB'		1		1

$S = C'D + CD' = C \oplus D.$



Convert Excess-3 code to BCD

Excess-3 A B C D	BCD P Q R S
0 0 1 1	0 0 0 0
0 1 0 0	0 0 0 1
0 1 0 1	0 0 1 0
0 1 1 0	0 0 1 1
0 1 1 1	0 1 0 0
1 0 0 0	0 1 0 1
1 0 0 1	0 1 1 0
1 0 1 0	0 1 1 1
1 0 1 1	1 0 0 0
1 1 0 0	1 0 0 1

Don't care input $= 2^4 = 16$ combinations

$\Rightarrow$ 10 (used)

$\Rightarrow 16 - 10 = 6$ (unused)

0000 1101
0001 1110
0010 1111

Kmap for P

AB \ CD	C'D'	C'D	CD	CD'
A'B'	X	X		X
A'B				
AB	1	X	X	X
AB'			1	

$P = AB + ACD$

Kmap for Q

AB \ CD	C'D'	C'D	CD	CD'
A'B'	X	X		X
A'B			1	
AB		X	X	X
AB'	1	1		1

$Q = BCD + B'D' + B'C'$

Kmap for R

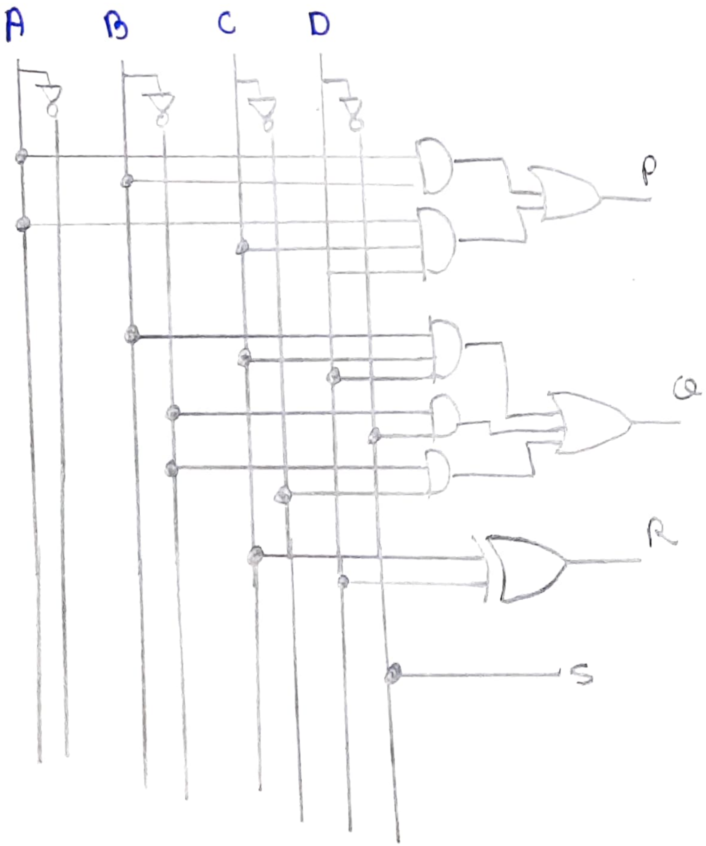
AB \ CD	00	01	11	10
00	X	X		X
01		1		1
11		X	X	X
10		1		1

$R = C'D + CD' = C \oplus D$

Kmap for S

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

$S = D'$



Sums to solve

2. gray to 4 bit binary
3. BCD to excess
4. Design a combinational circuit, its output is 1 when the ~~binary~~ input is even

Solution are available on our github & telegram channel.

Half adder

used to add 2 bits.

eg:->
$0 + 1 = 1 \rightarrow$ sum

$1 + 1 = 10$ (1 → carry, 0 → sum)

X	Y	Sum	carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Kmap for sum

X \ Y	Y'	Y
X'		1
X	1	

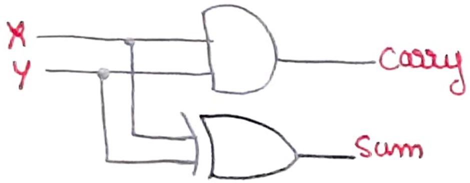
$Sum = X'Y + XY'$
$= X \oplus Y$

Kmap for carry

X \ Y	Y'	Y
X'		
X		1

$Carry = XY$

logic diagram.



full adder

used for 3bit addition

eg:→ 0 +1 +1 = 10 (carry, sum)

1, 1 } 10; + 1 → 10+1 = 11 (carry, sum)

X	Y	Z	sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

kmap for sum

X \ YZ	00	01	11	10
0		1		1
1	1		1	

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

$$\Rightarrow y'(x'z + xz') + y(xz + x'z')$$

$$\Rightarrow \underbrace{y'}_{A'}\underbrace{(x \oplus z)}_{B} + \underbrace{y}_{A}\underbrace{(x \oplus z)'}_{B'}$$

$$\Rightarrow A'B + AB' = A \oplus B$$

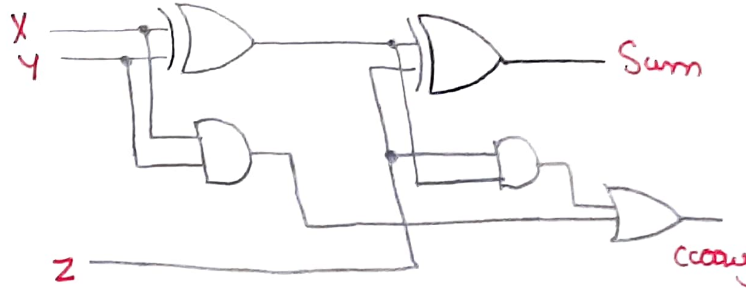
$$\Rightarrow Y \oplus X \oplus Z$$

$$\Rightarrow X \oplus Y \oplus Z$$

kmap for carry.

X \ YZ	00	01	11	10
0			1	
1		1	1	1

$$carry = XY + ZX + YZ$$



NOTE :→ YOU CAN ALSO TAKE X, Y, Z AS A, B, Cin (It is just a variable to express)

half subtractor :→

used for 2bit subtraction.

X	Y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

eg:→ 1 − 1 = 0 (difference)

10 (2) − 1 (1) = 1 (difference), borrow(1)

kmap for B

$x \backslash y$	y'	y
x'		$\boxed{1}$
x		

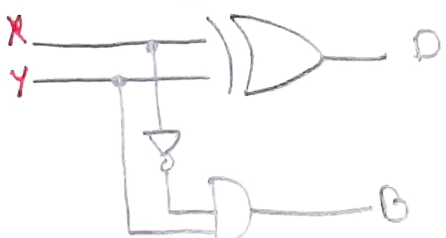
$B = x'.y$

kmap for D

$x \backslash y$	y'	y
x'		$\boxed{1}$
x	$\boxed{1}$	

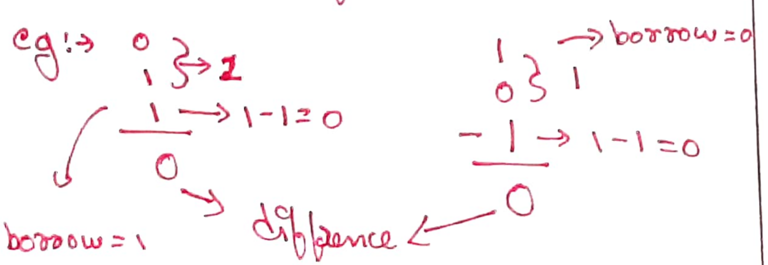
$D = xy' + x'y$
$= x \oplus y$

logic diagram.



full subtractor

used for 3bit subtraction.



X	Y	Z	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

kmap for D

$x \backslash yz$	00	01	11	10
0		$\boxed{1}$		$\boxed{1}$
1	$\boxed{1}$		$\boxed{1}$	

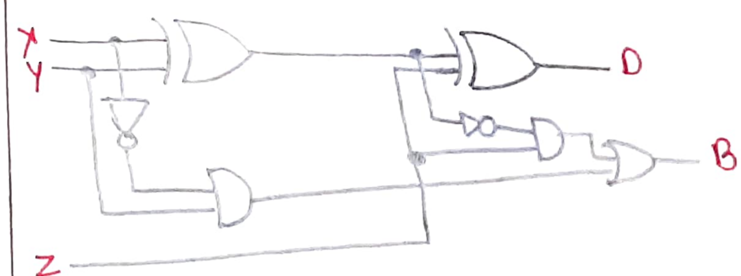
kmap for B

$x \backslash yz$	00	01	11	10
0		$\boxed{1}$	$\boxed{1}$	$\boxed{1}$
1			$\boxed{1}$	

~~D = x'y'Bin' + x'y Bin'~~

$D = x'y'z + x'yz' + xy'z'$
$+ xyz$
$= x \oplus y \oplus z.$

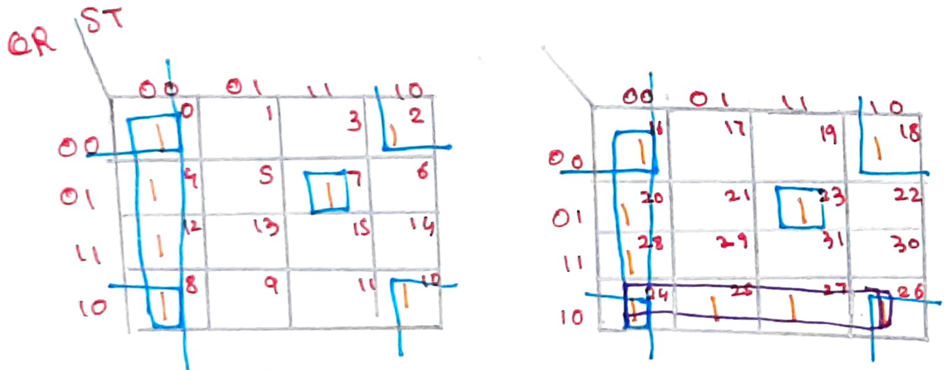
$B = x'z + x'y + yz$
$= x'z(y+y') + x'y(z+z')$
$+ yz(x'+x)$
$= x'yz + x'y'z + x'yz + x'yz'$
$+ x'yz + xyz$
$= x'yz + x'y'z + x'yz' + xyz$
$= x'y(z+z') + z(x'y' + xy)$
$B = x'y + z(x \oplus y)'$



5 Variable Kmap

we will use 2 4 variable kmap here

Solve $F(PQRST) = \Sigma(0, 2, 4, 7, 8, 10, 12, 16, 18, 20, 23, 24, 25, 26, 27, 28)$



P=0 P=1

→ not overlapping

→ overlapping

$F = S'T' + R'T' + Q'RST + PQR'$

Sums to solve

5. $F(A,B,C,D,E) = \Sigma(0, 2, 8, 10, 16, 18, 24, 26, 31, 30)$

6. $F(A,B,C,D,E) = \Sigma(30, 17, 16, 24, 25, 28)$

Solution on github on telegram channel.

BCD adder :→

→ addition of two BCD number

→ if the number goes beyond 9 then add 6 to it (0110)

Binary sum					BCD sum → carry				
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	0	0	1	0
0	0	0	1	1	0	0	0	1	1
0	0	1	0	0	0	0	1	0	0
0	0	1	0	1	0	0	1	0	1
0	0	1	1	0	0	0	1	1	0
0	0	1	1	1	0	0	1	1	1
0	1	0	0	0	0	1	0	[illegible]	0
0	1	0	0	1	0	1	0	[illegible]	1
0	1	0	1	0	1	0	0	0	0
0	1	0	1	1	1	0	0	0	1
0	1	1	0	0	1	0	0	[illegible]	0
0	1	1	0	1	1	0	0	[illegible]	1
0	1	1	1	0	1	0	1	0	0
0	1	1	1	1	1	0	1	0	1
1	0	0	0	0	1	0	1	[illegible]	0
1	0	0	0	1	1	0	1	[illegible]	1
1	0	0	1	0	1	1	0	0	0
1	0	0	1	1	1	1	0	0	1

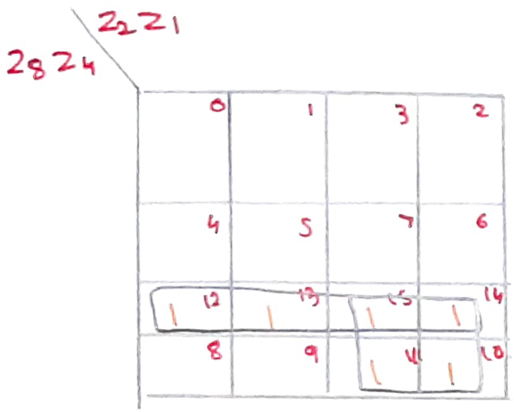
→ 9+9+1 = 19 (0 to 19)

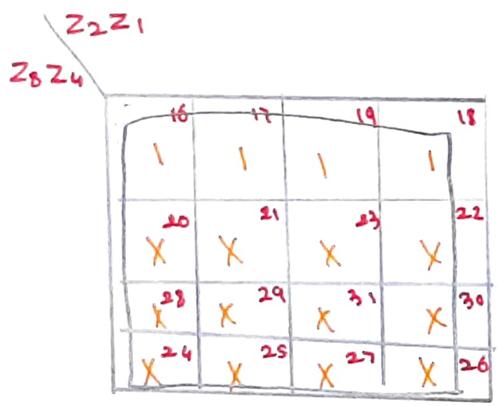
add 6 to all numbers more than 9

eg:→ (5)

$\begin{array}{r} 01111 \\ +\ 0110 \\ \hline 10101 \end{array}$

Kmap for BCD (carry)

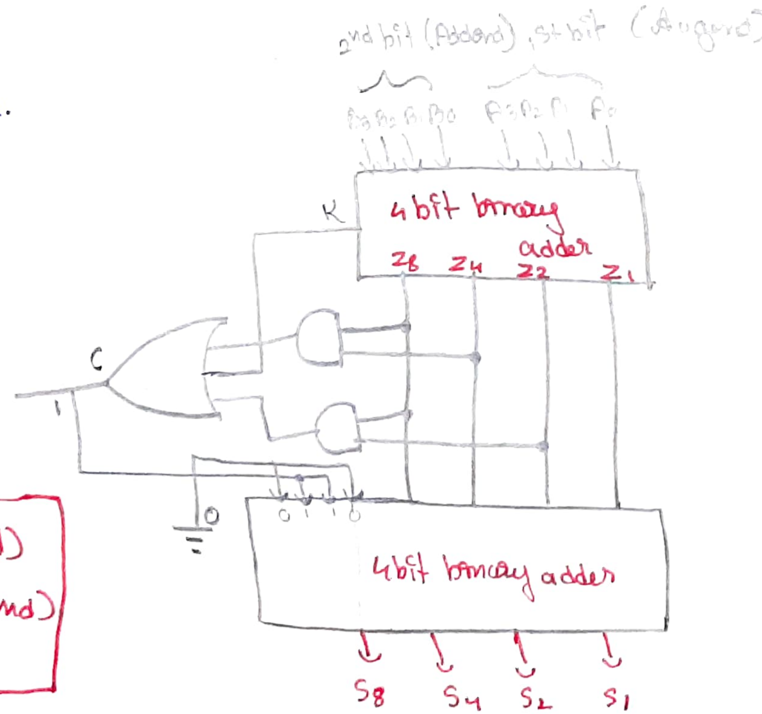




$K=0$ $K=1$

$$C = Z_8Z_4 + Z_8Z_2 + K$$

block diagram.



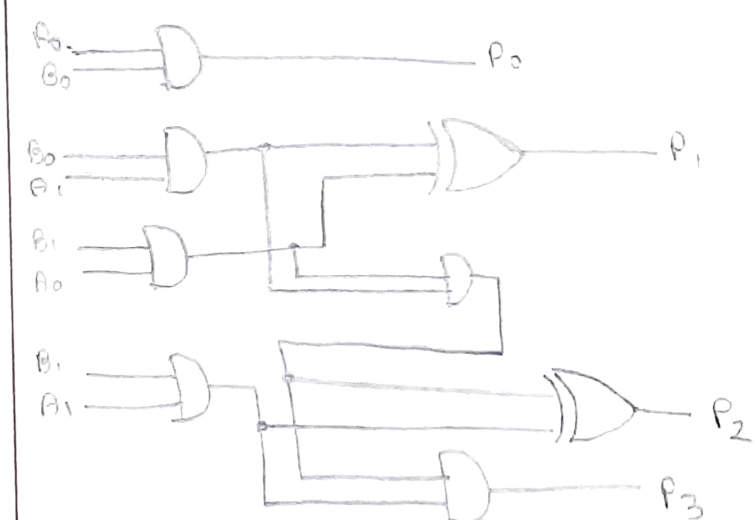
$$\begin{array}{r} 20 \text{ (added)} \\ +\ 40 \text{ (augend)} \\ \hline 60 \end{array}$$

2x2 Binary Multiplier

- → used to multiply 2 bits
- → uses combinational circuit

$$\begin{array}{cccc} & & A_1 & A_0 \\ & & B_1 & B_0 \\ \hline & & A_1B_0 & A_0B_0 \\ & A_1B_1 & A_0B_1 & X \\ \hline P_3 & P_2 & P_1 & P_0 \end{array}$$

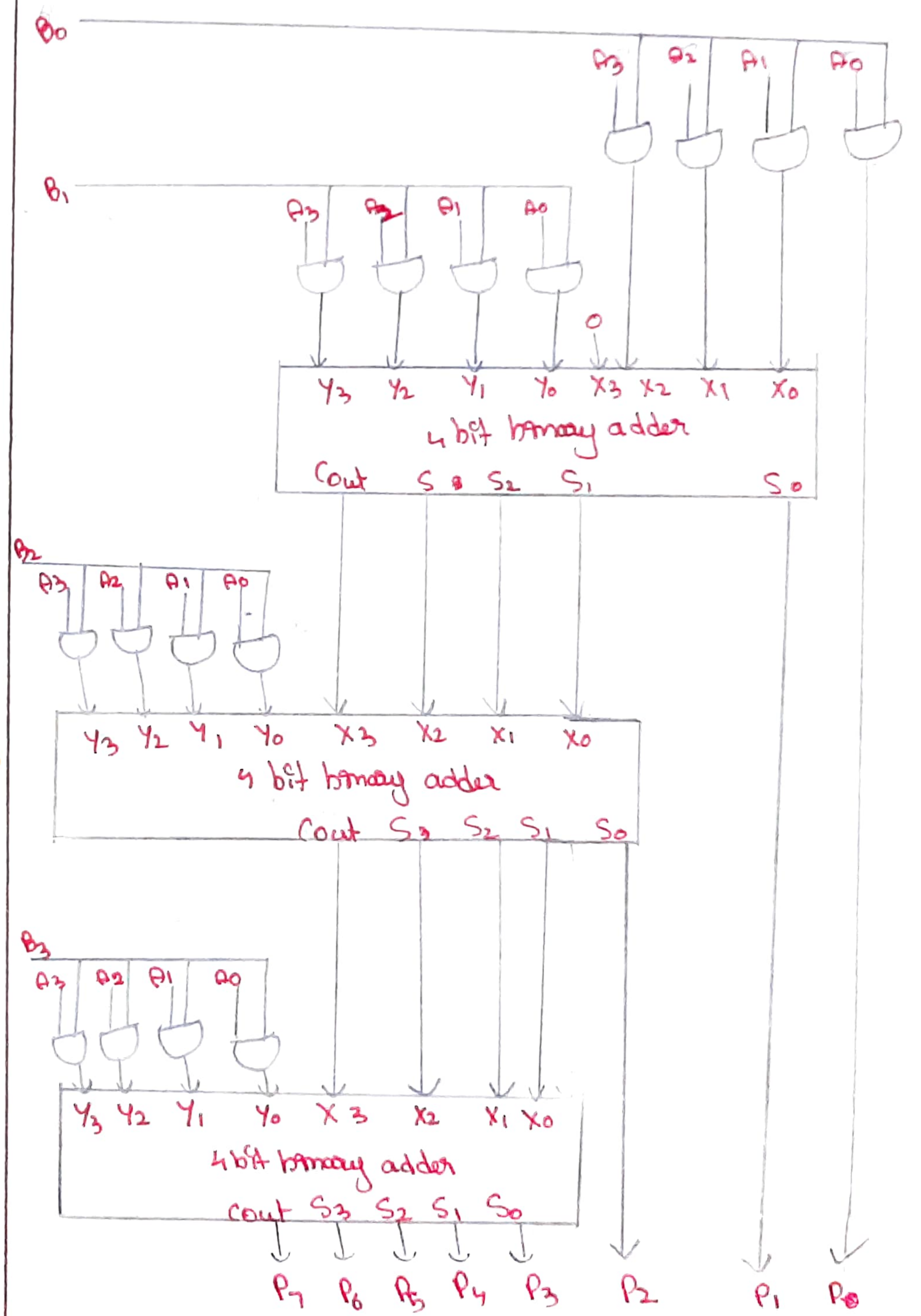
- P_3 → carry of P_2
- P_2 → A_1B_1 + carry of P_1
- P_1 → $A_1B_0 + A_0B_1$
- P_0 → A_0B_0



4x4 Binary multiplier

				A3	A2	A1	A0
				B3	B2	B1	B0
				A3B0	A2B0	A1B0	A0B0
			A3B1	A2B1	A1B1	A0B1	X
		A3B2	A2B2	A1B2	A0B2	X	X
	A3B3	A2B3	A1B3	A0B3	X	X	X
P7	P6	P5	P4	P3	P2	P1	P0
carry of P6	A3B3 + carry of P5	A3B2 + A2B3 + carry of P4	A3B1 + A2B2 + A1B3 + carry of P3	A3B0 + A2B1 + A1B2 + A0B3 + carry of P2	A2B0 + A1B1 + A0B2 + carry of P1	A1B0 + A0B1	A0B0

~~Here the maximum addition is b/w 2 no. So there will be 2 4bit adders.~~



Magnitude comparator:

→ used to compare two binary numbers

→ implemented using combinational circuit.

$(A = B) \Rightarrow AB + A'B' = X$ $\left[\begin{array}{l} A = A_3A_2A_1A_0 \\ B = B_3B_2B_1B_0 \end{array}\right]$

$$\boxed{X_i = A_iB_i + A_i'B_i'}$$

$X_1 = A_1B_1 + A_1'B_1'$

$X_2 = A_2B_2 + A_2'B_2'$

$X_3 = A_3B_3 + A_3'B_3'$

$X_0 = A_0B_0 + A_0'B_0'$

$$\boxed{A = B \Rightarrow X_3X_2X_1X_0}$$

$(A>B) \Rightarrow A_3B_3' + X_3A_2B_2' + X_3X_2A_1B_1' + X_3X_2X_1A_0B_0'$

$(A<B) \Rightarrow A_3'B_3 + X_3A_2'B_2 + X_3X_2A_1'B_1 + X_3X_2X_1A_0'B_0$

compare :→ $A = 1011$, $B = 1100$

	A_3	A_2	A_1	A_0
A	1	0	1	1

	B_3	B_2	B_1	B_0
B	1	1	0	0

$X_0 = (A_0 \oplus B_0)'$

$\Rightarrow (1 \oplus 0)' = (1)' = 0$

$X_1 = (1 \oplus 0)' = (1)' = 0$

$X_2 = (0 \oplus 1)' = 0$

$X_3 = (1 \oplus 1)' = (0)' = 1$

$A = B \Rightarrow X_3X_2X_1X_0 = 1\cdot0\cdot0\cdot0 = 0$ //.

$A > B = 1\cdot(1)' + 1\cdot0\cdot(1)' + 1\cdot0\cdot1\cdot(0)' + 1\cdot0\cdot0\cdot0\cdot1\cdot(0)'$

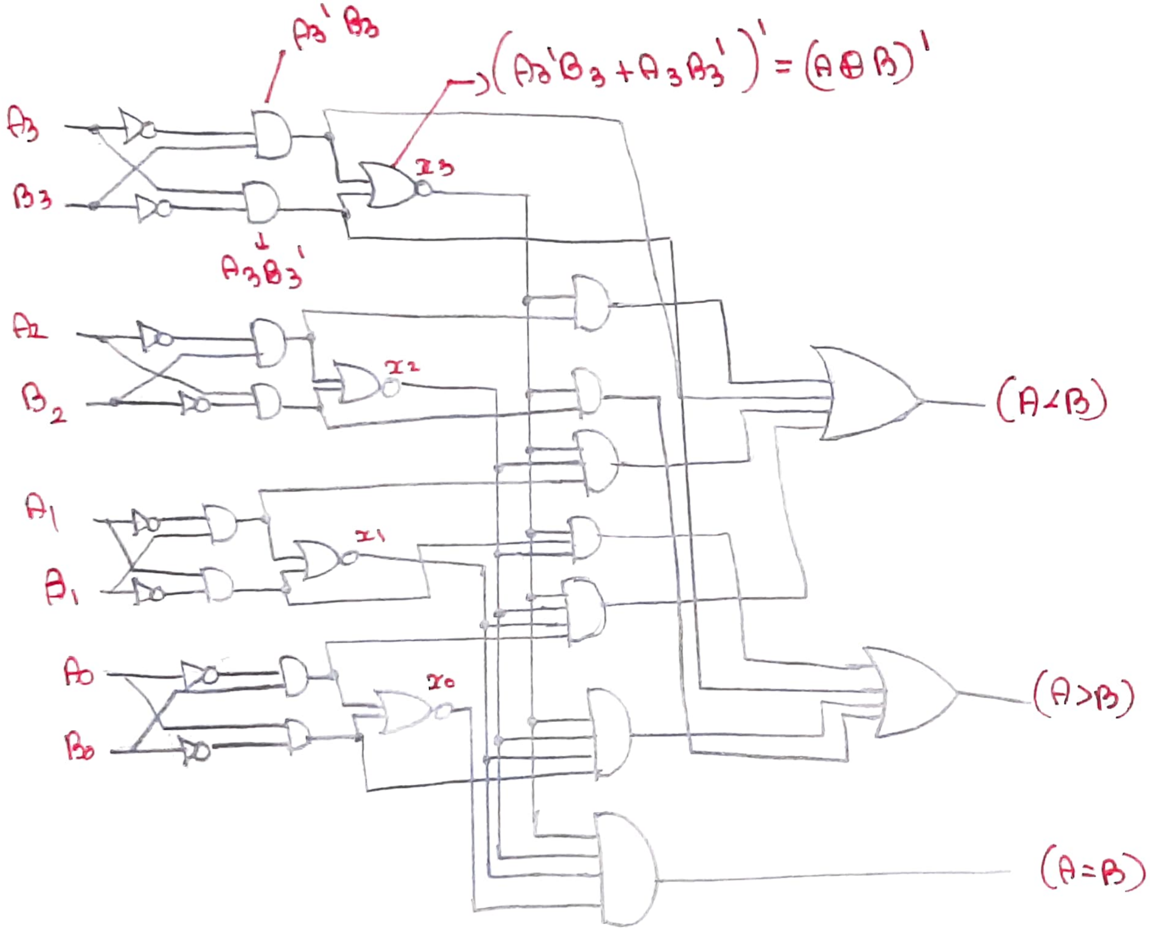
$= 1\cdot0 + 1\cdot0\cdot0 + 1\cdot0\cdot1\cdot1 + 1\cdot0\cdot0\cdot0\cdot1\cdot1$

$= 0$ //

$A < B = (1)'\cdot1 + 1\cdot(0)'\cdot1 + 1\cdot0\cdot(1)'\cdot0 + 1\cdot0\cdot0\cdot(1)'\cdot0$

$= 0\cdot1 + 1\cdot1\cdot1 + 1\cdot0\cdot0\cdot0 + 1\cdot0\cdot0\cdot0\cdot0$

$= 1$ //



Decoder:->

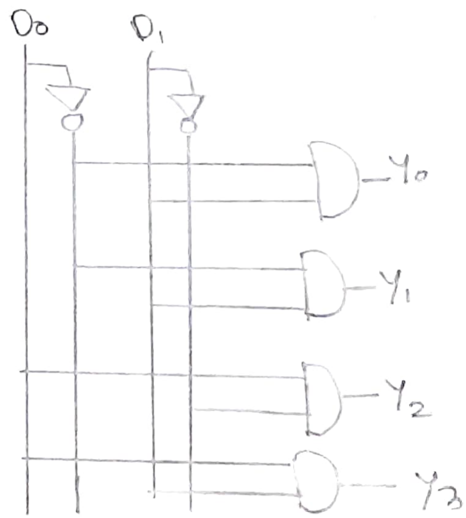
-> generate minterm

-> $n \times 2^n$ -> output

n -> input

truth table of 2x4 decoder.

D_0	D_1	Y_0	Y_1	Y_2	Y_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



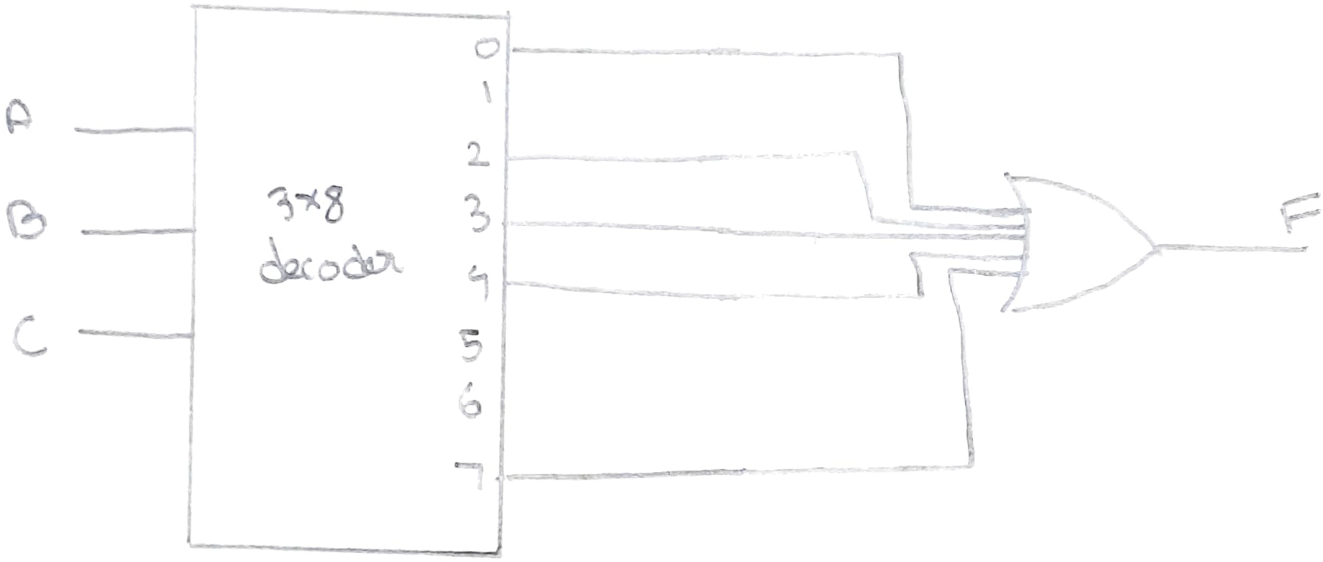
Implementation of a boolean function using decoder

$F(A, B, C) = \Sigma(0, 2, 3, 4, 7)$

$F(A, B, C) = \Sigma(0, 2, 3, 4, 7)$

3 inputs

decoder $= n \times 2^n = 3 \times 2^3 = 3 \times 8$



Implement the boolean function $F = A + BC$ using decoder.

$F = A + BC$

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

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

$\Rightarrow \underset{\substack{111 \\ 7}}{ABC} + \underset{\substack{101 \\ 5}}{AB'C} + \underset{\substack{110 \\ 6}}{ABC'} + \underset{\substack{100 \\ 4}}{AB'C'} + \underset{\substack{011 \\ 3}}{A'BC}$

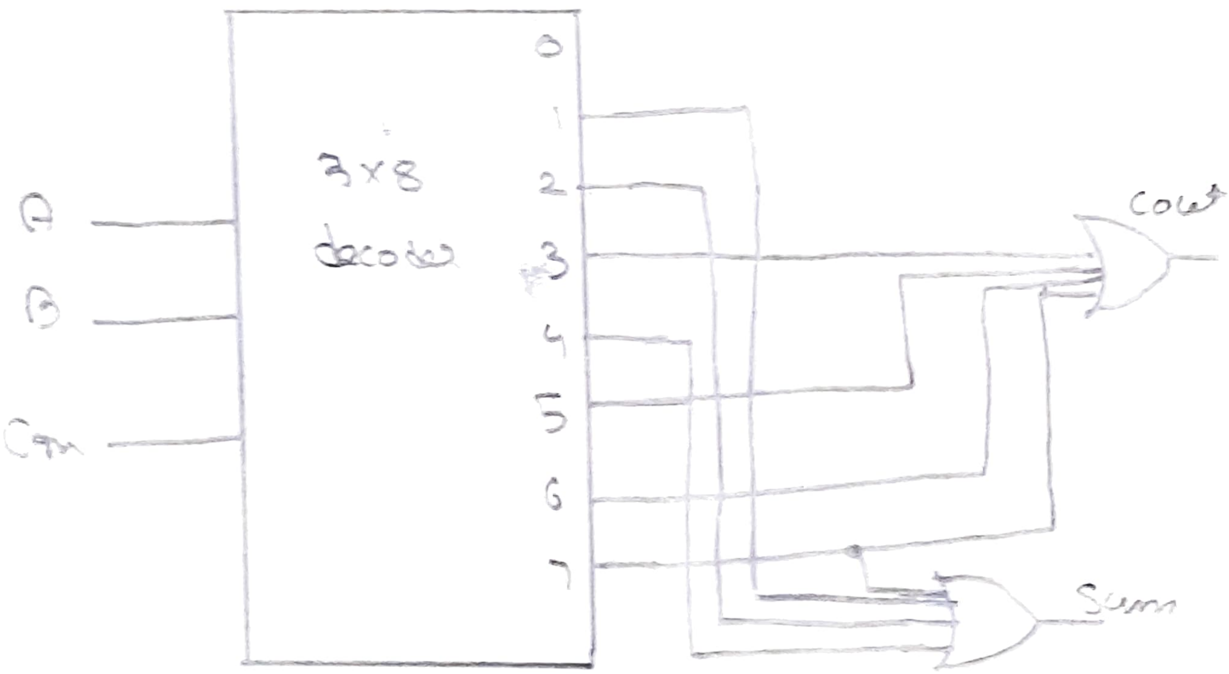

A
B
C
3x8
decoder
0
1
2
3
4
5
6
7
F

Implement a full adder using decoder.

A	B	Cin	Cout	Sum
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

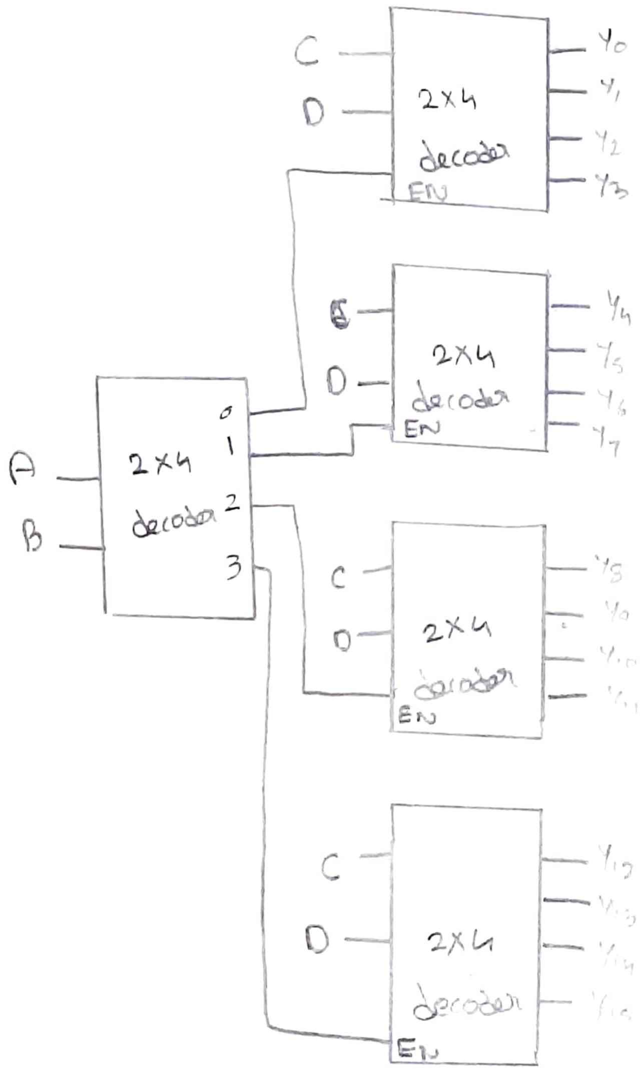
$Cout = \Sigma(3,5,6,7)$

$Sum = \Sigma(1,2,4,7)$



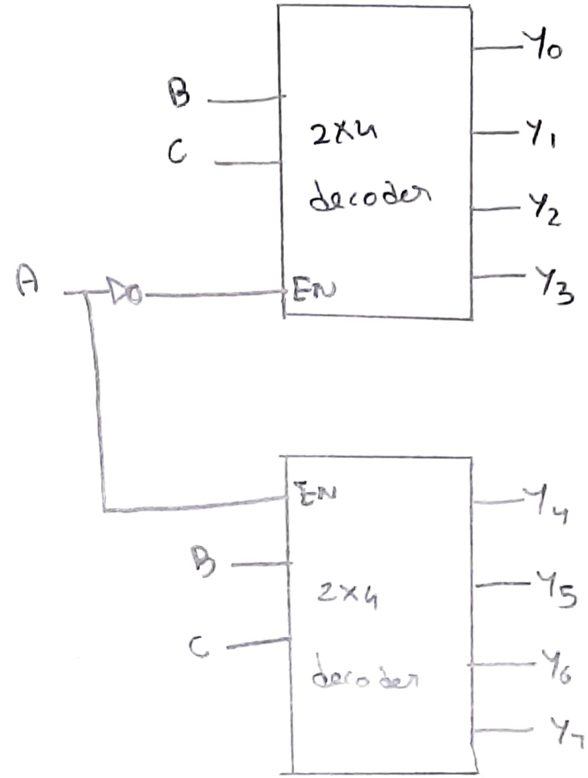
implement higher order to lower order decoder using

1) 4X16 using 2X4



EN → enable → additional input
if EN=1 output is 1
if EN=0 output is 0

2) 3x8 using 2x4



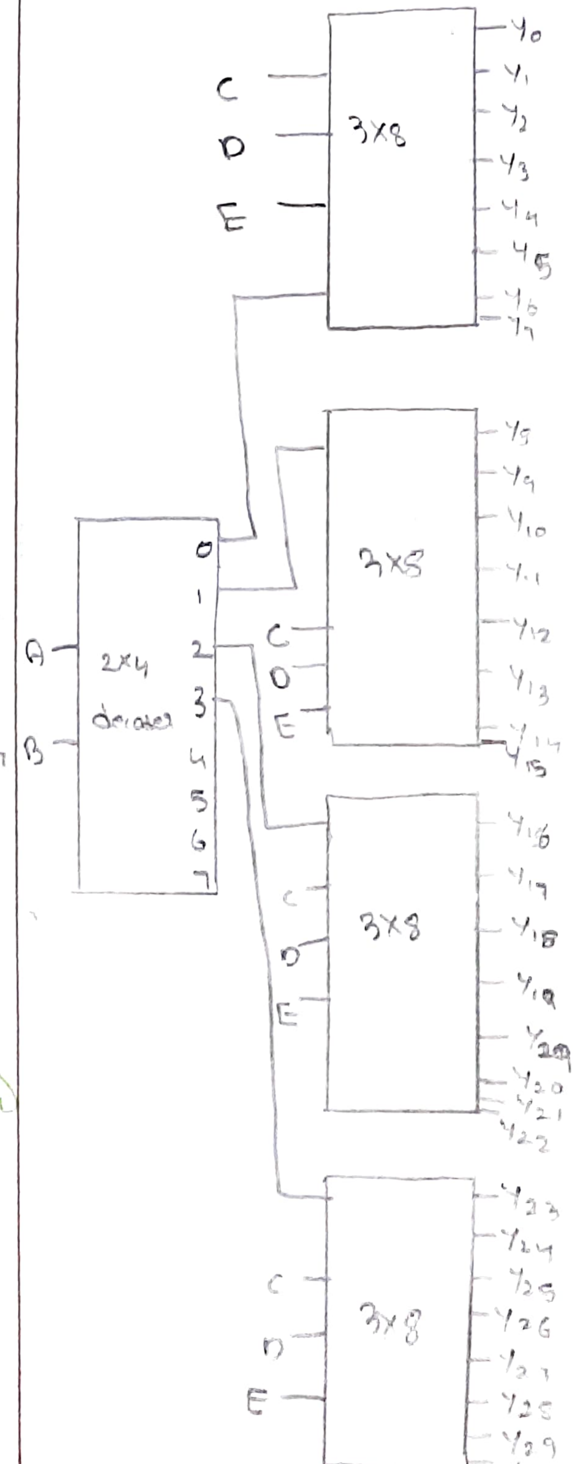
Sums to solve

* implement 1. $F(A,B,C,D) = \Sigma(0,1,2,14)$

2. $F = A+BD+C$

3. 3X8 using 1X2

3) 5X32 using 3X8



Multiplexer :→

it is combinational circuit that selects one of many input & direct its to the output.

$\Rightarrow 2^n \times 1$

input ↙ ↘ output

implement the boolean function $F = \Sigma(0,1,3,7,8,12,14,15)$ using MUX

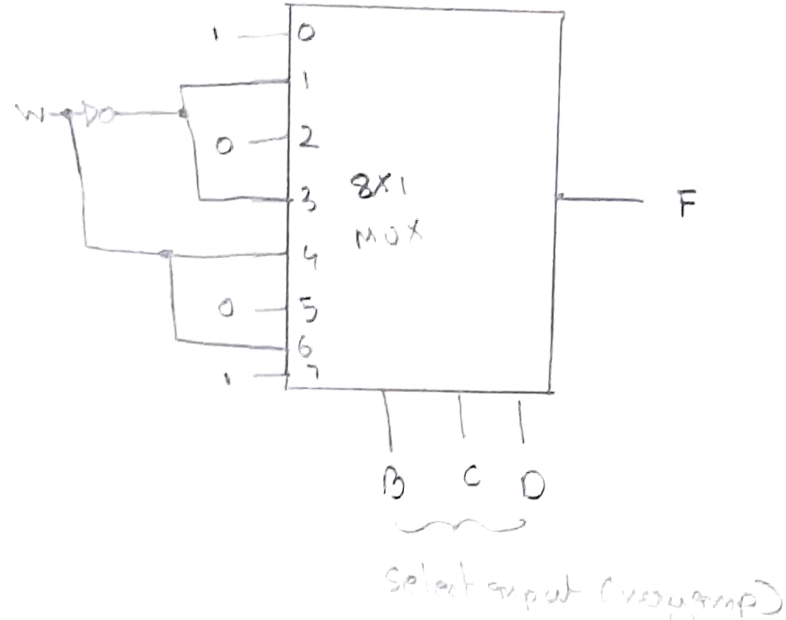
let inputs be A, B, C, D

data input $= A$

select input $= B, C, D$

$\Rightarrow 2^n \times 1 = 2^3 \times 1 = 8 \times$ MUX

		I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
W'	0	0	1	2	3	4	5	6	7
W	1	8	9	10	11	12	13	14	15
		1	W'	0	W'	W	0	W	1



implement $F = A'BC + AD' + BCD$ using MUX

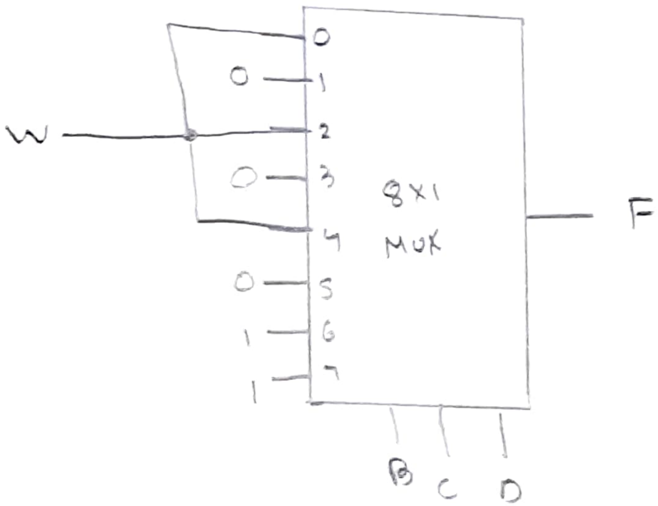
$= A'BC(D+D') + AD'(B+B')(C+C') + BCD(A+A')$

$= A'BCD + A'BCD' + ABCD' + AB'CD' + AB'C'D' + ABC'D' + ABCD + A'BCD$

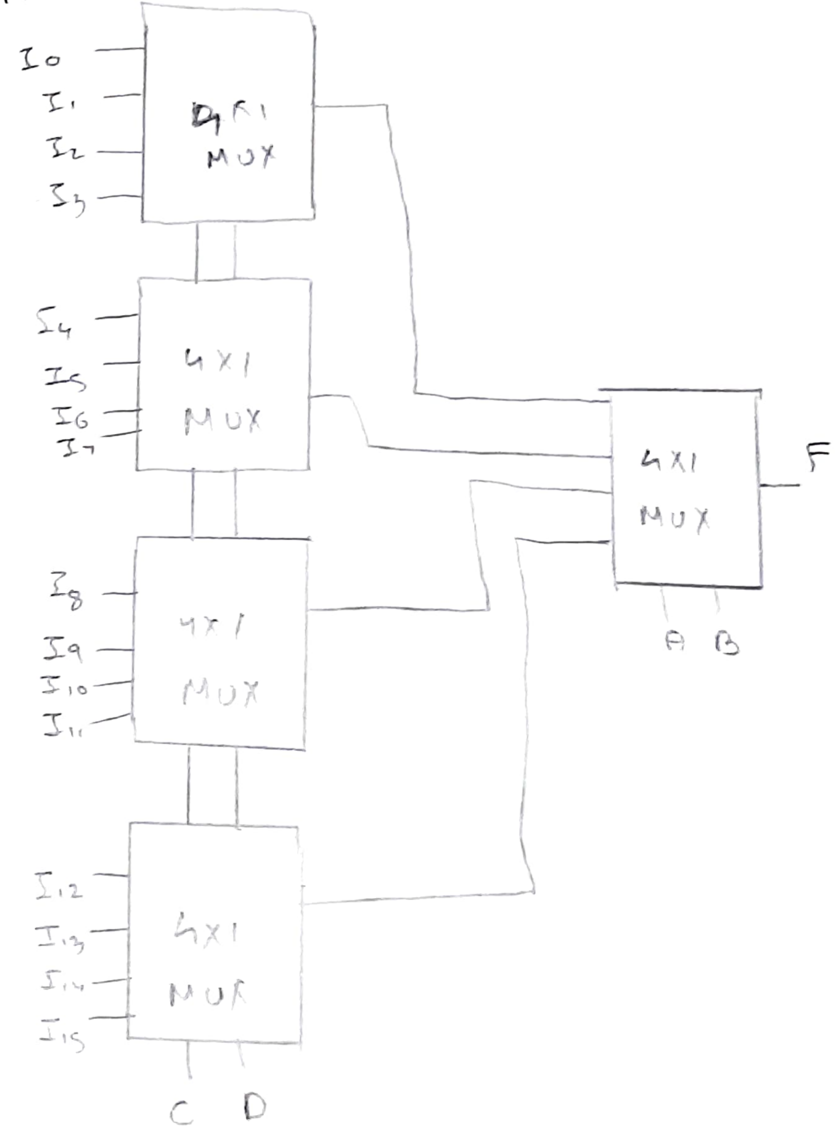
$= A'BCD + A'BCD' + ABCD' + AB'CD' + AB'C'D' + ABC'D' + ABCD$

$= \Sigma(6,7,8,10,12,14,15)$

		I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
W'	0	0	1	2	3	4	5	6	7
W	1	8	9	10	11	12	13	14	15
		W	0	W	0	W	0	1	1



16x1 using 4x1



Sums to solve 1. $F(A,B,C,D)=\Sigma(0,1,2,14,15)$

2. $F = A+B+C+AB+CD$

3. 32X1 using 8X1

Encoder:->

-> it is combinational circuit

-> reverse of decoder

=> $2^n \times n$ -> output

input

truth table of 4x2 encoder

D_0	D_1	D_2	D_3	Y_0	Y_1
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

drawback:->

-> if more than one input is high then O/P not defined

-> when all input are 0 then O/P ~~not defined~~ .will be 00

Priority encoder:->

-> uses priority function

-> if D_1, D_2, D_3 are high D_3 will have highest priority.

4x2 priority encoder:->

D_0	D_1	D_2	D_3	Y_0	Y_1	V
0	0	0	0	X	X	0
1	0	0	0	0	0	1
X	1	0	0	0	1	1
X	X	1	0	1	0	1
X	X	X	1	1	1	1

kmap for Y_0

D_0D_1 \ D_2D_3	00	01	11	10
00	X	1	1	1
01		1	1	1
11		1	1	1
10		1	1	1

$Y_0 = D_3 + D_2$

kmap for Y_1

D_0D_1 \ D_2D_3	00	01	11	10
00	X	1	1	
01	1	1	1	
11	1	1	1	
10		1	1	

$Y_1 = D_3 + D_1 D_2'$

Kmap for V

D_0D_1 \ D_2D_3	00	01	11	10
00		1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$V = D_0 + D_1 + D_2 + D_3$$