Digital Ckts

Combinational	Sequential
# Present o/p ⇒ present i/p	Present o/p → Present i/p, previous o/ps.
# No feed back	Feed back
# No Memory	Memory
# Ex.→ H.A, H.S, MUX, Decoder	Ex.→ F/F, Latches, Register, Counter.

1. COMBINATIONAL CKT

Procedure →

1. Identify no. of i/ps & o/ps
2. Construct truth table.
3. Write logical exp. in SOP (or) POS
4. Minimize logic exp.
5. Implement logic ckt.

✱ Arithmetic Ckts

addition →

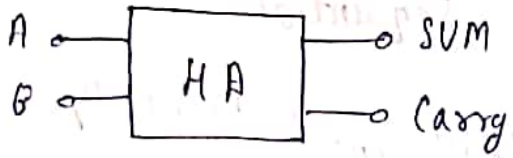
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 1 0
↑ ↑
Carry Sum

Scanned by CamScanner

1. Half Adder ⇒

☆ H.A Question are –

① Logical Exp. Sum → $A \oplus B$
② Logical Exp. Carry. → AB
③ No. of NAND gate – 5
④ No. of NOR
⑤ No. of MUX
⑥ No. of Decoder

① 

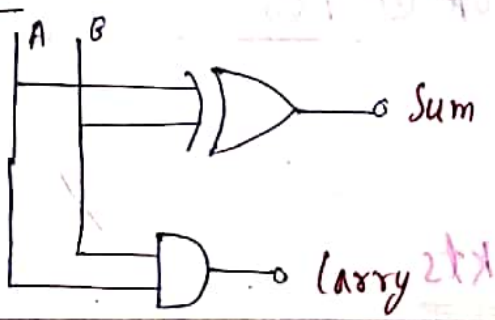
② Truth table ⇒

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

③ Logical Expression →

$$Sum = \bar{A}B + A\bar{B}$$
$$Carry = AB$$

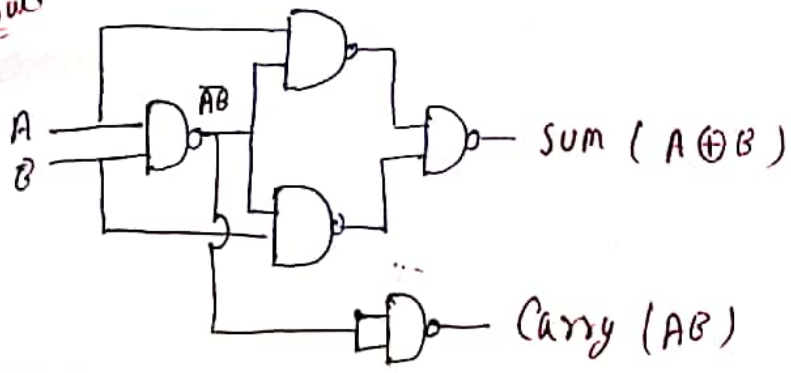
⑤ Logic Ckts ⇒



IES-Conven.

Q. Implement HA using NAND gate ?

Sol.



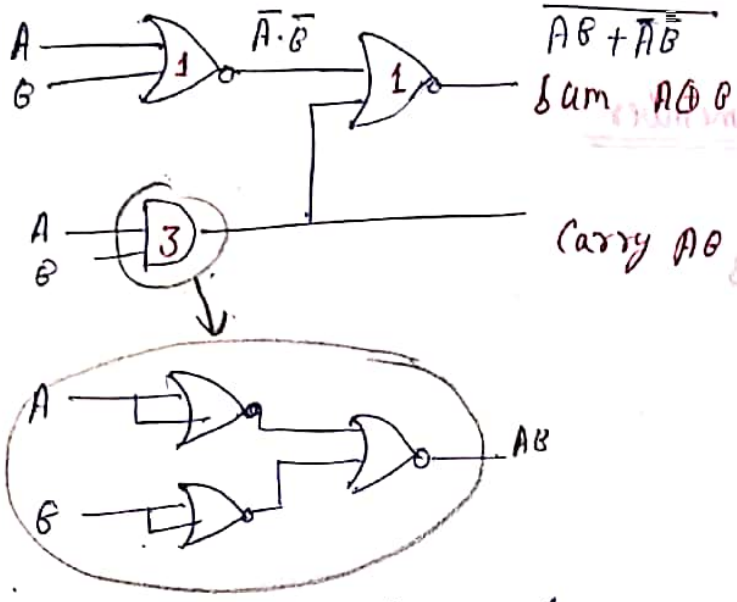
Ans. So Min no of NAND gate is 5

Scanned by CamScanner

IES

Q. Impliment HA using NOR =>

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



So Min no. of NOR gate - 5 Ans.

IES

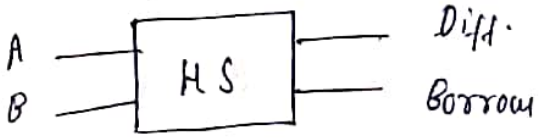
Q. Min no. of AND & NOR gate required For HA

Sol. 2-NOR , 1-AND

---

2. Half Substractor (HS)

(i)



(ii) Truth table =>

	A	B	Diff A-B	Borrow
	0	0	0	0
①	0	1	1	1
	1	0	1	0
	1	1	0	0

Scanned by CamScanner

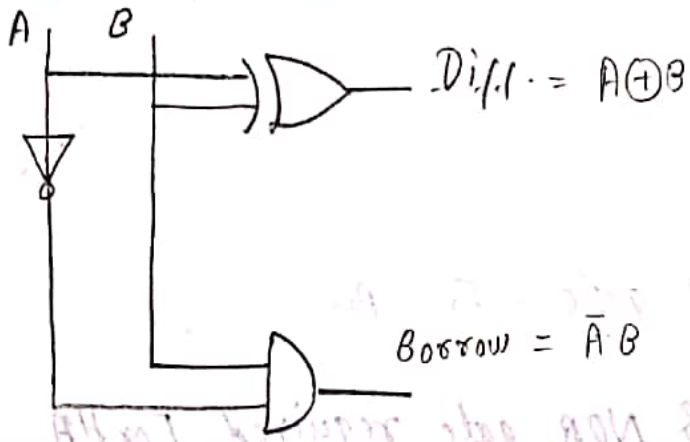
(3.) Logical Expression. ⇒

$Diff. = A \oplus B$

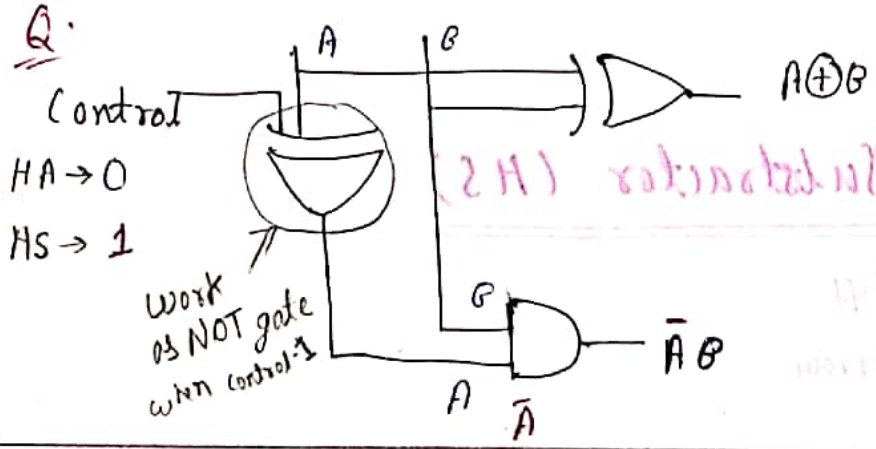
$Borrow = \bar{A} B$

* If A − B then $\bar{A} B$
If B − A then $A \bar{B}$
} Remember

(5.) Implement Logic ckt ⇒



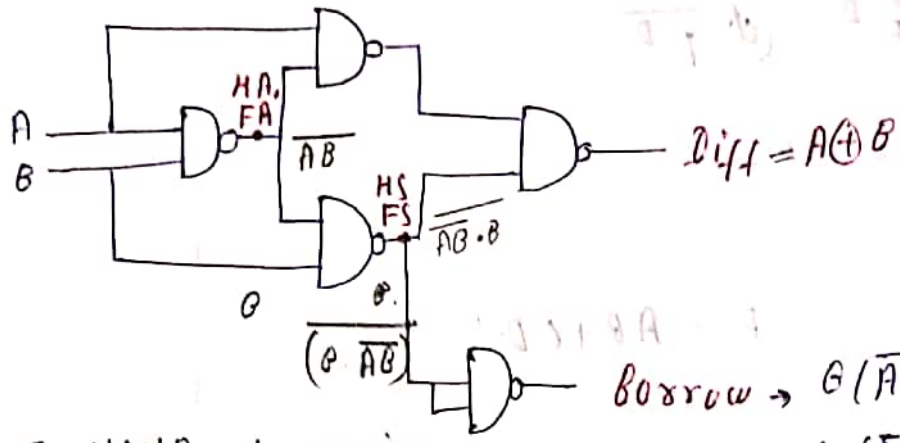
Q.



# Control = 0 ⟶ HA
Control = 1 ⟶ HS

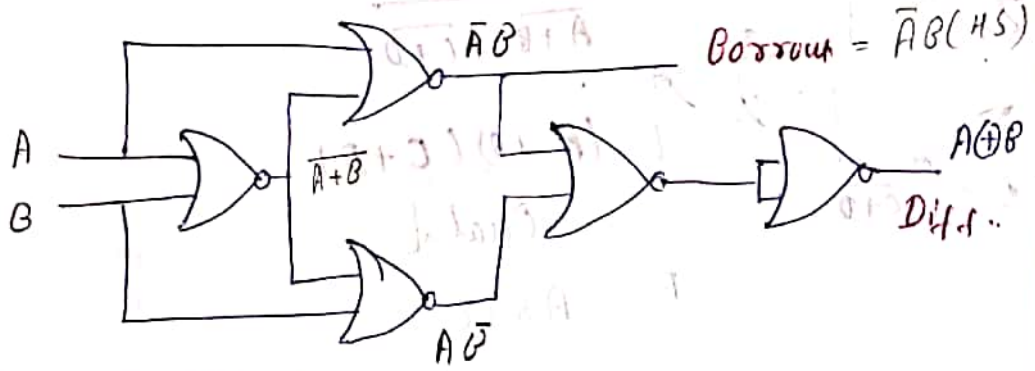
Scanned by CamScanner

✱ Implementation of HS Using NAND gate. →



✱ 5-NAND gate require for HS.

✱ Implementation of HS Using NOR gate-



So 5-NOR gate req.

~~A~~ ✱ AND ⟷ (Dual) OR

EL ✱ NAND ⟷ (Dual) NOR

✱ EXOR ⟷ (Dual) EXNOR

✓ $\bar{A}B + A\bar{B}$ ⟷ (Dual) $(\underline{\bar{A}}+B)(\underline{A}+\bar{B}) = AB + \bar{A}\bar{B}$

✱ NOT ⟷ (Dual) NOT

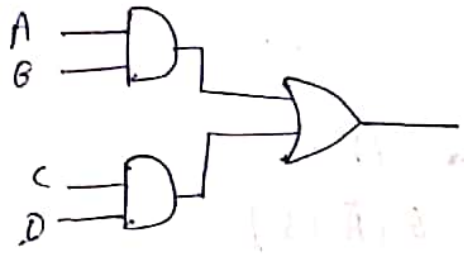
✓ $\bar{A}$ ⟷ (Dual) $\bar{A}$

Scanned by CamScanner

Q. IES-05 A two level AND-OR n/w, o/p is F. If ckt is replaced with NOR gate, then o/p is

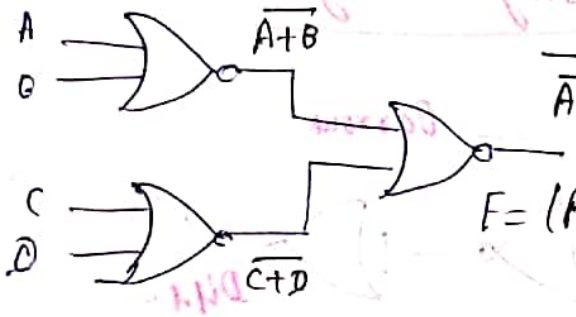
(a) F (b) $\bar{F}$ (c) F^D (d) $\overline{F^D}$

Sol 2-level AND-OR



$F = AB + CD$

By NOR →



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

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

Dual ↓

$F = AB + CD$

Ans. F^D

Full Adder

① Identify no. of i/p & o/p

② Truth Table →

A B C	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

Scanned by CamScanner

Logical exp. for sum ⇒

$$SUM = A \oplus B \oplus C$$

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

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

$$= AB + BC + AC \rightarrow (1)$$

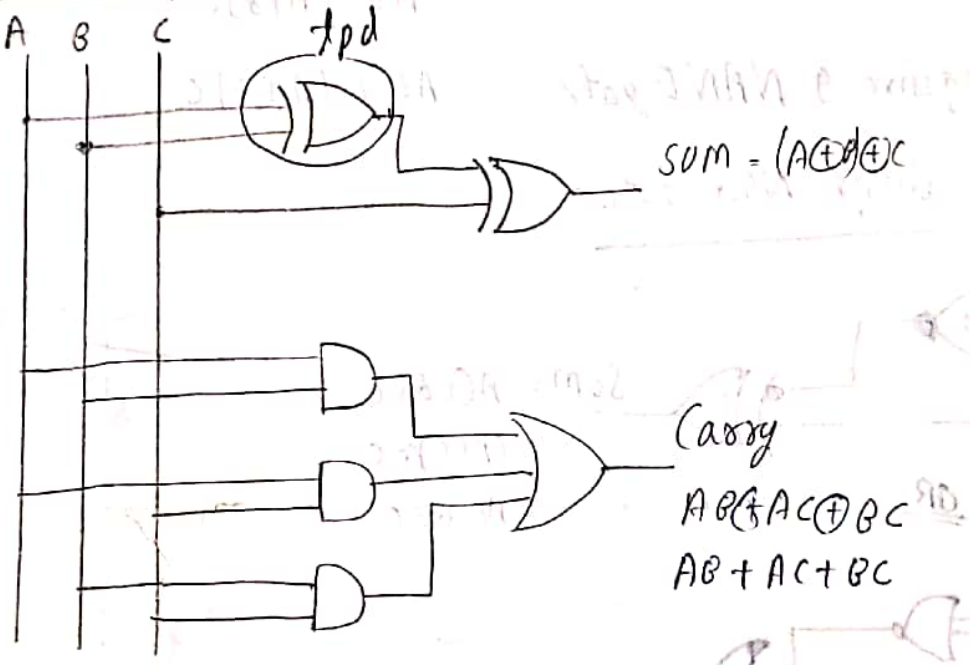
$$= \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

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

$$= AB + (A \oplus B)C \rightarrow (2)$$

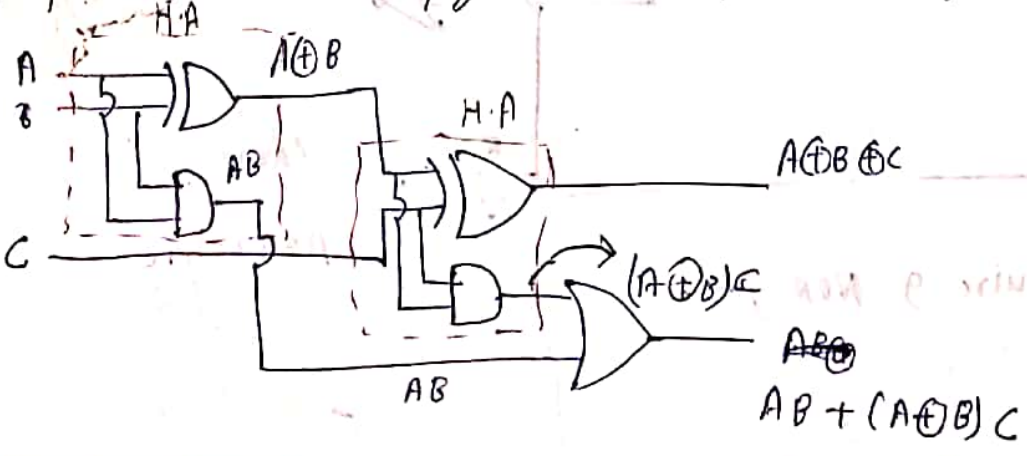
$$Carry = \Sigma m(3, 5, 6, 7)$$

Implementation ⇒



* In Full-adder if all logic gates have same propagation delay than to provide sum or carry output minimum 2 t_{pd} delay is required

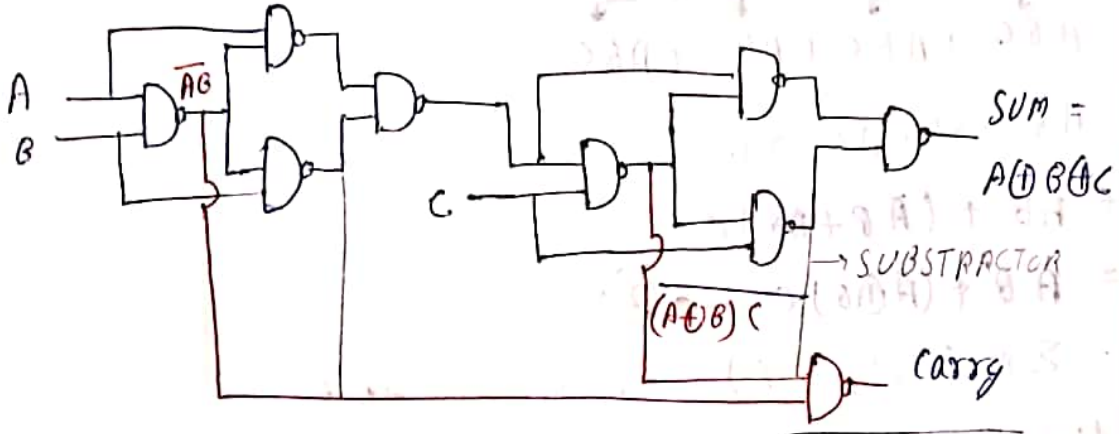
* Ckt for Minimum no. of gate used ⇒ 2-HA + 1 OR gate ⇒



Scanned by CamScanner

FA ⟶ 2 HA and 1 OR gate

Full adder using NAND gate →

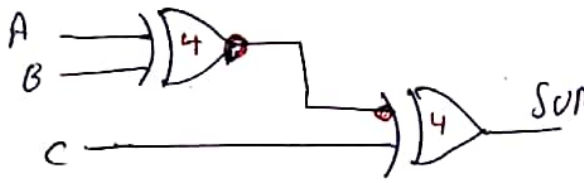


$\overline{AB}$ $\overline{(A+B)C}$

$AB + (A \oplus B)C$

Full adder require 9 NAND gate

Full adder using NOR gate →

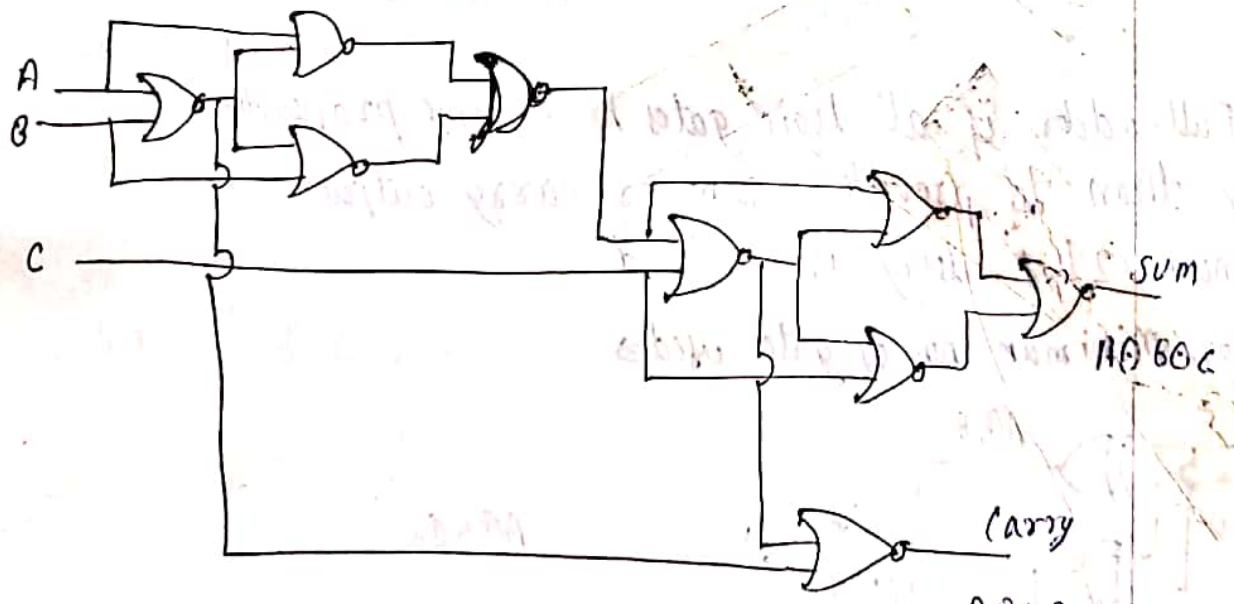


SUM = A⊙B⊙C

(A⊙B)⊙C

A⊙B⊙C

OR



$AB + BC + AC$

So FA require 9 NOR gate

Note →

Scanned by CamScanner

	NAND	NOR
HA	5	5
HS	5	5
FA	9	9
FS	9	9

FA

① SUM — $A \oplus B \oplus C$
$\Sigma m(1,2,4,7)$

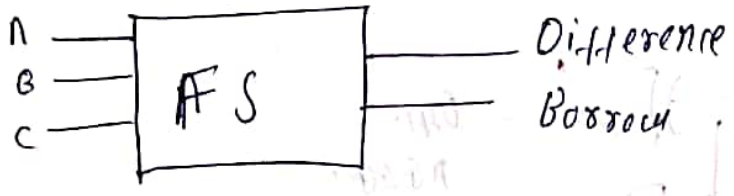
② Carry – $AB + AC + BC$
$AB + (A \oplus B)C$
$\Sigma m(3,5,6,7)$

③ No. of HA & OR → (2, 1)

④ NAND → 9

⑤ NOR → 9

Full Substractor



A B C	Diff (A − B) − C	Borrow
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

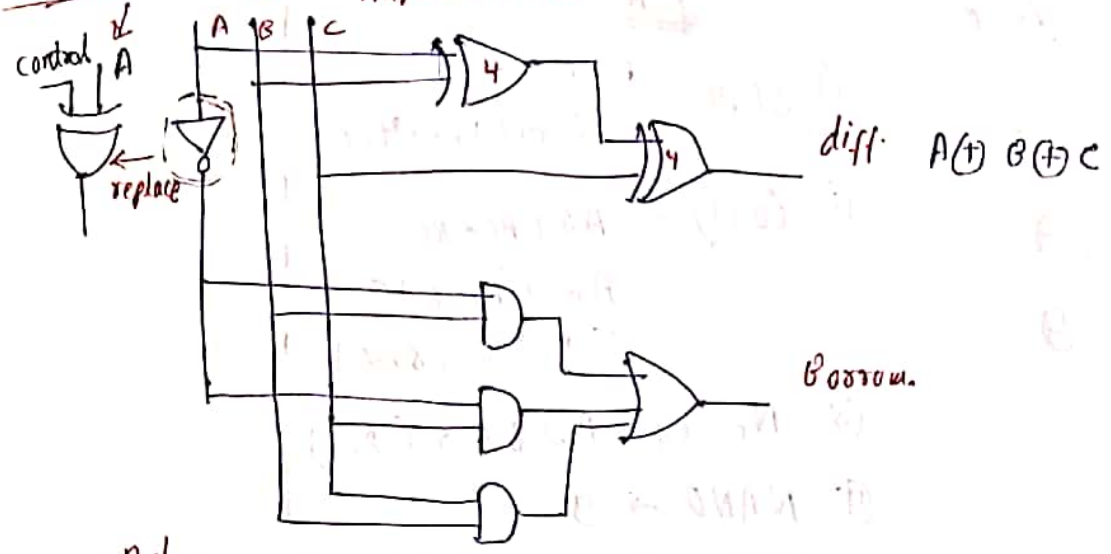
Difference → $= A \oplus B \oplus C$
$= \Sigma m(1,2,4,7)$

Borrow $= \Sigma m(1,2,3,7)$
$= \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}BC + ABC$
$= \bar{A}B + \bar{A}C + BC \rightarrow (1)$

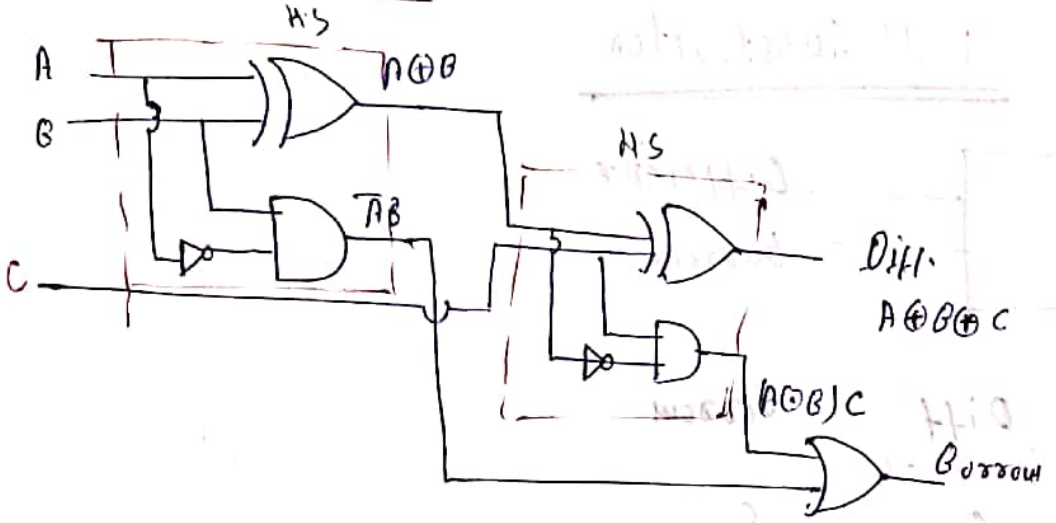
Borrow $= \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}BC + ABC$
$= \bar{A}B + (\bar{A}\bar{B} + AB)C$
$= \bar{A}B + (A \odot B)C$

Scanned by CamScanner

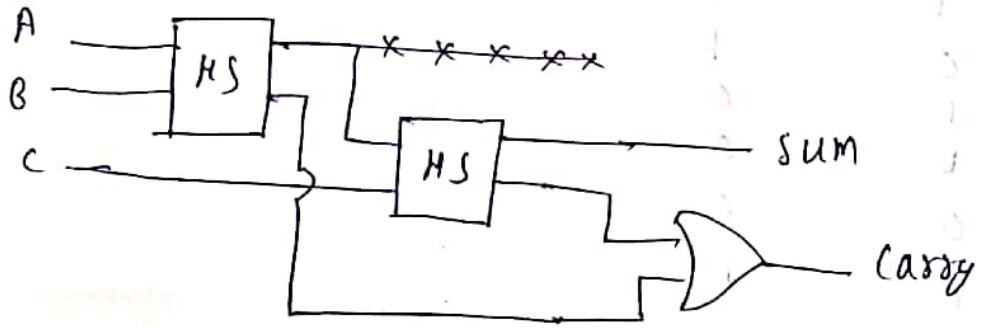
GATE-CS 1st implementation



2nd Implementation →



FS → 2 HS and 1 OR gate



FS

① Diff → $A \oplus B \oplus C$
$\Sigma m(1,2,4,7)$

② Borrow $\bar{A}B + \bar{A}C + BC$
$\bar{A}B + (A \odot B)C$
$\Sigma m(1,2,3,7)$

③ No. of HS & OR → (2, 1)

④ NAND → 9, ⑤ NOR → 9

Scanned by CamScanner

Note ⇒

In order to Add group of bits, following adder ckt is used ÷

(1) Serial adder
(2) Parallel adder
(3) Look ahead carry adder

(1) Serial adder ⇒

In serial adder to provide n-bit addition we require only 1 full-adder.

It is a slowest adder, whose it require n-clock pulse for n-bit addition.

(2) Parallel adder ⇒

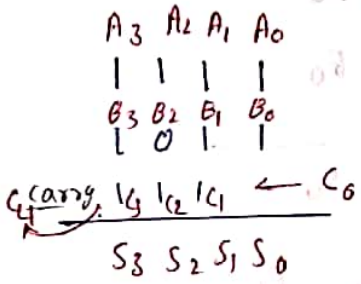
4-bit → 3 FA and 1 HA

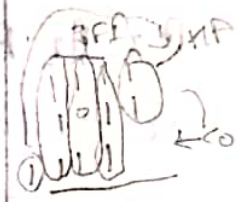
or

4 FA

or

7 HA & 3 OR gates.



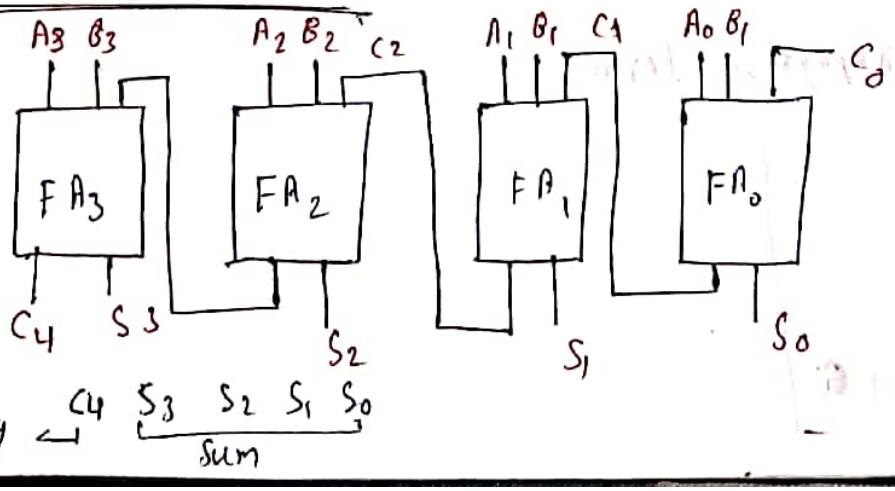


In parallel adder to add each bit seprate Full adder or Half adder are used.

In this to provide n-bit addition, it require ÷

* (n-1) FA and one HA
* n FA
* (2n-1) HA and (n-1) OR gates.

4-bit Full adder ⇒



Scanned by CamScanner

In parallel adder each full-adder will provide $2t_{pd}$ delay ~~and~~ by carry.

In n-bit parallel adder to provide final result, it require $\boxed{2nt_{pd}}$ delay

In parallel adder carry propogation delay will be present from i/p to o/p hence ~~it~~ parallel adder is also known as ripple carry adder.

4-bit Look ahead carry adder

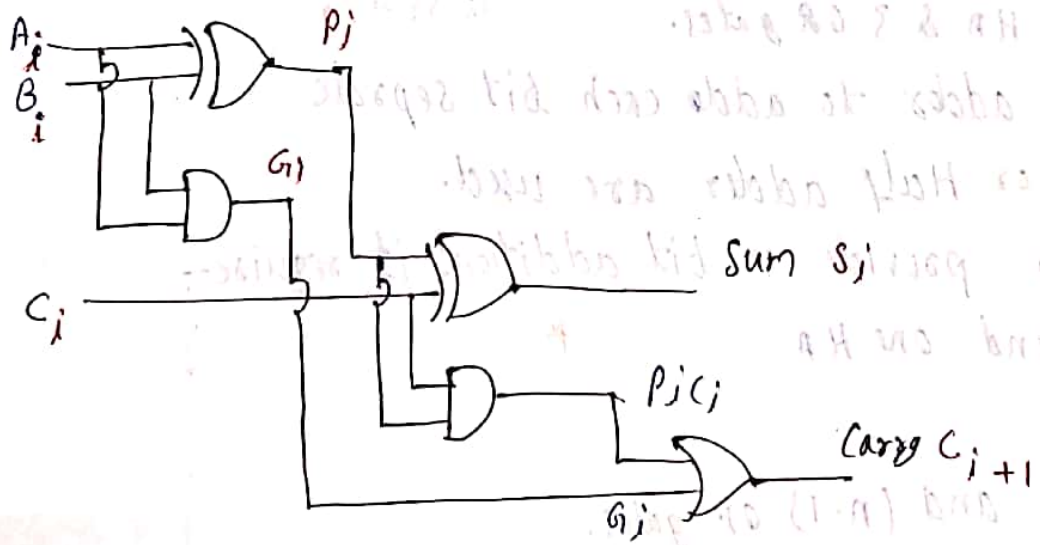
$A_3\ A_2\ A_1\ A_0$

$B_3\ B_2\ B_1\ B_0$

$C_4\ C_3\ C_2\ C_1 \leftarrow C_0$

$S_3\ S_2\ S_1\ S_0$

Ex.



$G_i \rightarrow$ carry generating term (Immediately)

$P_i \rightarrow$ carry propogation term (more time)

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

Scanned by CamScanner

$P_0 = A_0 \oplus B_0$	$G_0 = A_0 \cdot B_0$	$S_0 = P_0 \oplus C_0$
$P_1 = A_1 \oplus B_1$	$G_1 = A_1 \cdot B_1$	$S_1 = P_1 \oplus C_1$
$P_2 = A_2 \oplus B_2$	$G_2 = A_2 \cdot B_2$	$S_2 = P_2 \oplus C_2$
$P_3 = A_3 \oplus B_3$	$G_3 = A_3 \cdot B_3$	$S_3 = P_3 \oplus C_3$

* Look ahead carry network (n/w) ⇒

$C_0 \rightarrow$ input carry.

$C_1 = P_0 C_0 + G_0$

$C_2 = P_1 C_1 + G_1$

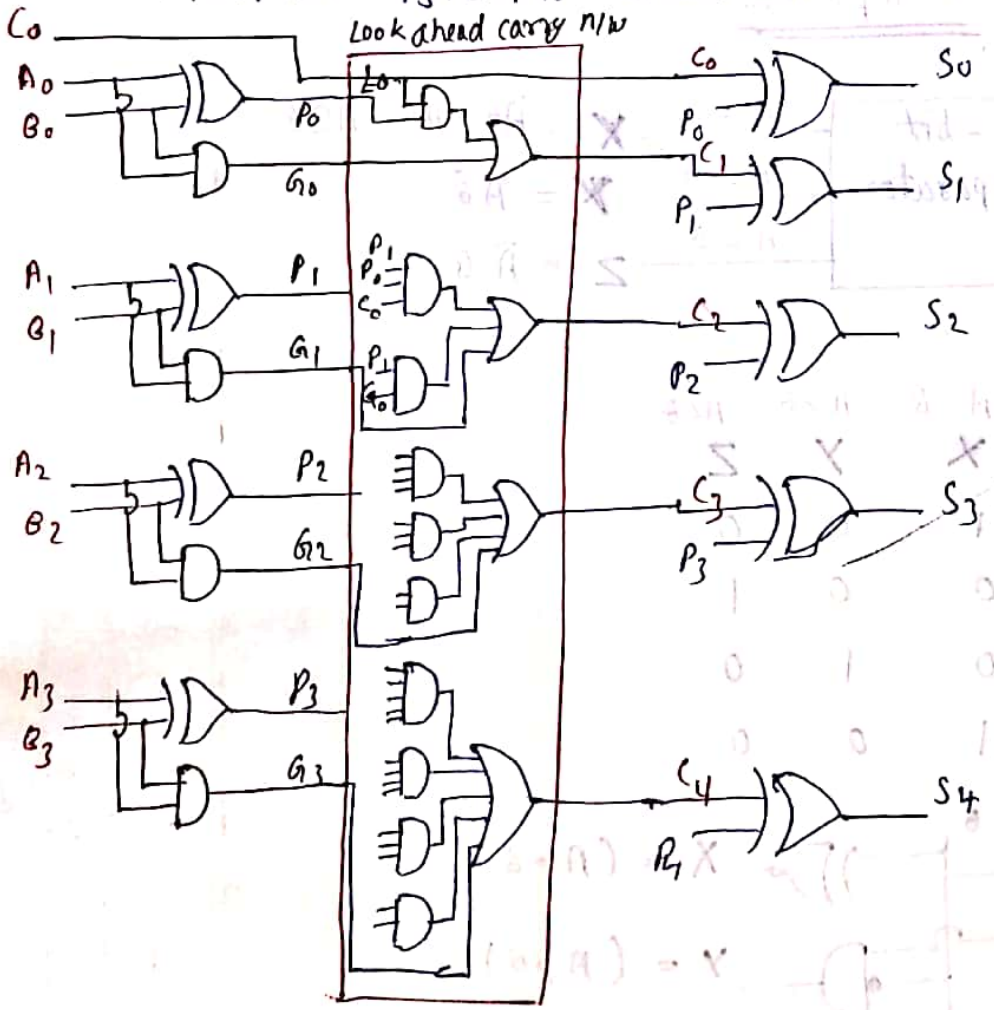
$= P_1 P_0 C_1 + P_1 G_0 + G_1$

$C_3 = P_2 C_2 + G_2$

$= P_2 P_1 P_0 C_0 + P_2 P_1 G_0 + P_2 G_1 + G_2$

$C_4 = P_3 C_3 + G_3$

$= P_3 P_2 P_1 P_0 C_0 + P_3 P_2 P_1 G_0 + P_3 P_2 G_1 + P_3 G_2 + G_3$



Scanned by CamScanner

Look ahead carry is fastest adder

In this carry is generated in parallel using two level AND-OR look ahead carry adder.

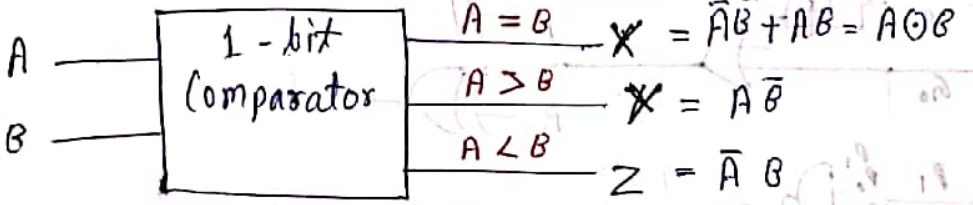
In this if all logic gate have same propogation delay than to provide carry output it require minimum 3 tpd delay and to provide sum o/p 4 tpd delay.

In look ahead carry n/w no. of AND gate used is $\frac{n(n+1)}{2}$

Similary no. of OR gate → n

Comparator

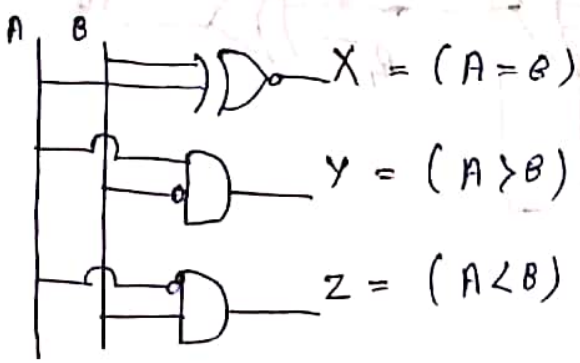
* 1-bit Comparator →

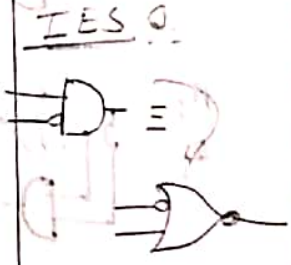


Truth table →

A	B	A=B X	A>B Y	A<B Z
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

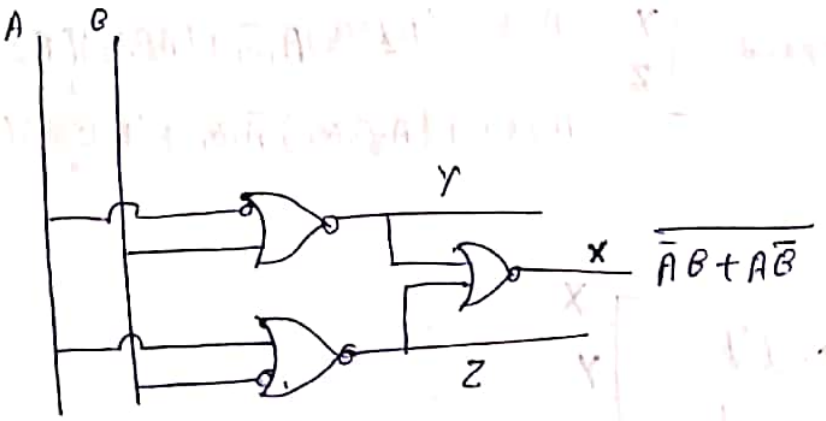
Implementation →



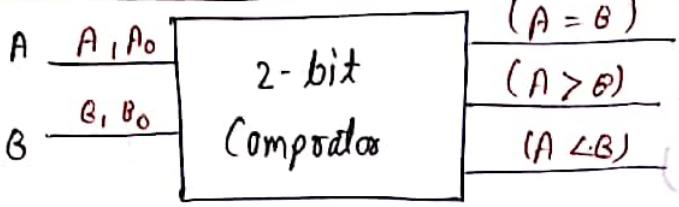


Scanned by CamScanner

How to make min. no. of gate.



2-bit Comparator ⇒



$X \Rightarrow (A_1 \odot B_1) \cdot (A_0 \odot B_0)$

$Y \Rightarrow (A_1 > B_1) + (A_1 = B_1)(A_0 > B_0)$

$Z \Rightarrow A_1 \bar{B}_1 + (A_1 \odot B_1) \cdot A_0 \bar{B}_0$

$\Rightarrow \bar{A}_1 B_1 + (A_1 \odot B_1) \cdot \bar{A}_0 B_0$

Truth table ⇒

A_1	A_0	B_1	B_0	A=B X	A>B Y	A<B Z
0	0	0	0	1	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	0	1	0
0	1	0	1	1	0	0
0	1	1	0			
0	1	1	1			1
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

Equal

A_1 A_0

B_1 B_0

Ex. 3 7

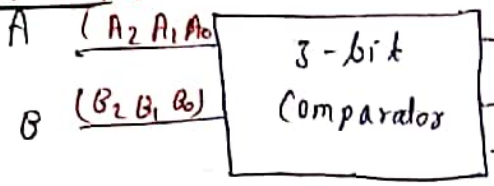
3 7

GATE- 12

CS, ELE, EC, EI

Scanned by CamScanner

3-bit comp ⇒

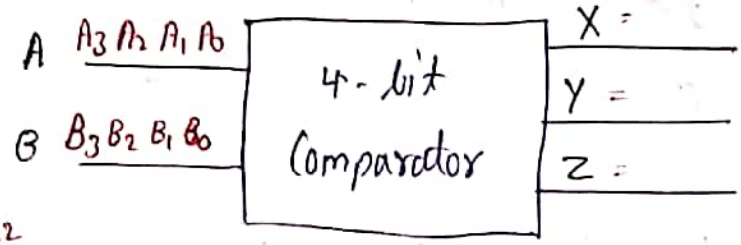


$X = (A_2 \odot B_2) \cdot (A_1 \odot B_1)(A_0 \odot B_0)$

$Y = A_2\overline{B}_2 + (A_2 \odot B_2)A_1\overline{B}_1 + (A_2 \odot B_2)(A_1 \odot B_1)A_0\overline{B}_0$

$Z = \overline{A}_2 B_2 + (A_2 \odot B_2)\overline{A}_1 B_1 + (A_2 \odot B_2)(A_1 \odot B_1)\overline{A}_0 B_0$

4-bit Comp. ⇒



IES-12

Q. Simplified eq.n of

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

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

$A + \overline{B}C$ Ans.

IES-12

Q. Which of the eq.n is correct

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

Gray Code

Unweighted Code.

In Gray code successive number will differ by only one bit.

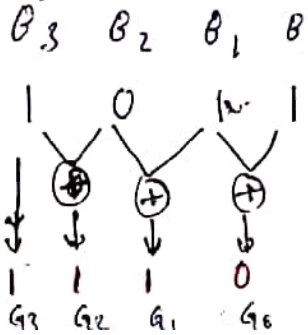
UDC (Unic distance code)

called Minimum error code

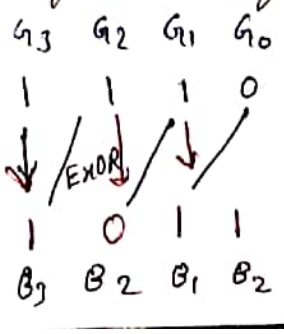
Also called → cyclic code
→ Reflective code

Binary to Gray

#



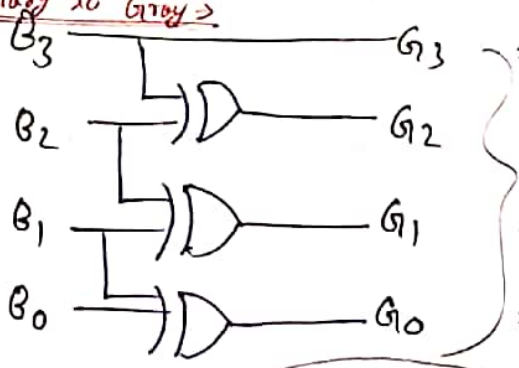
Gray to binary



Scanned by CamScanner

Q. Binary to Gray →

IES-02
20 Mark
2 Mins (EM)



$G_3 = B_3$

Logic ckt $= B_3 \oplus B_2$

$= B_2 \oplus B_1$

$= B_1 \oplus B_0$

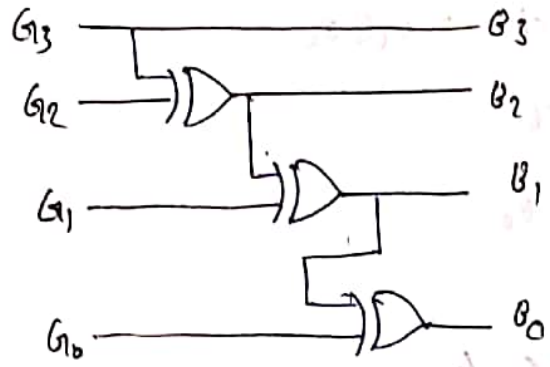
Truth table →

B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
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

① Find i/p o/p
② Truth table
③ Logical Exp.
④ Implement

Scanned by CamScanner

Gray to Binary code →



K-map

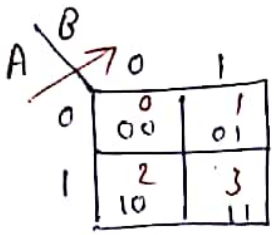
K-map → 0, 1, X

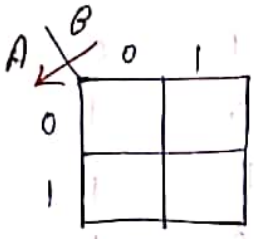
SOP → 2, 3, 4, (5)

POS → 2, 3, 4

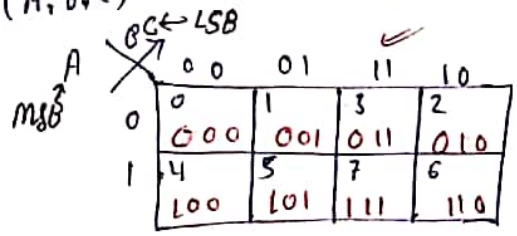
f(A, B)

MSB LSB

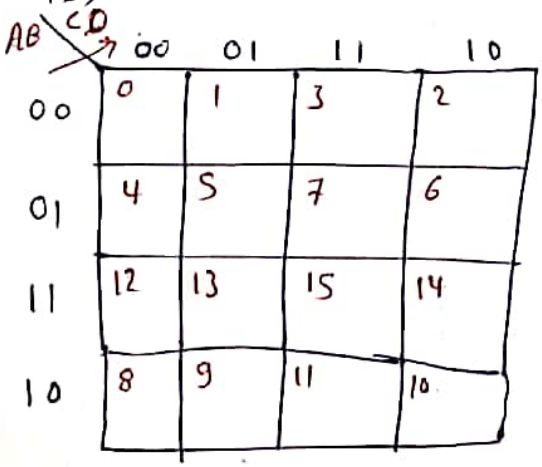




f(A, B, C)

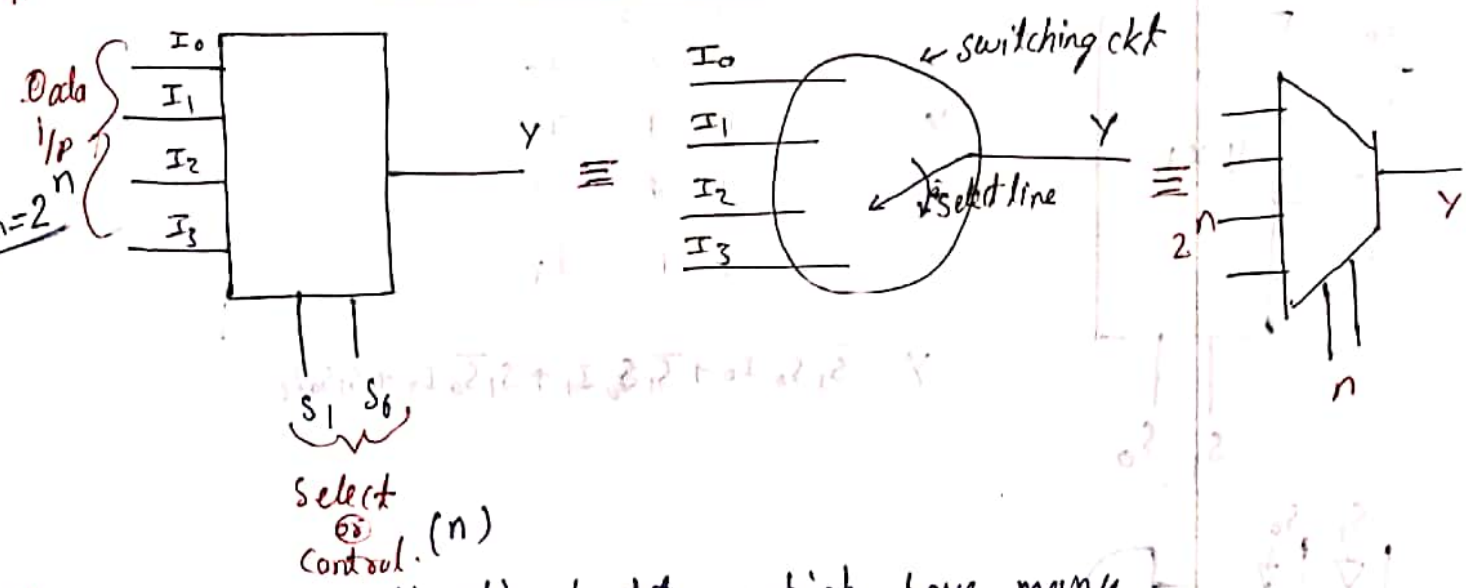


f(A, B, C, D)



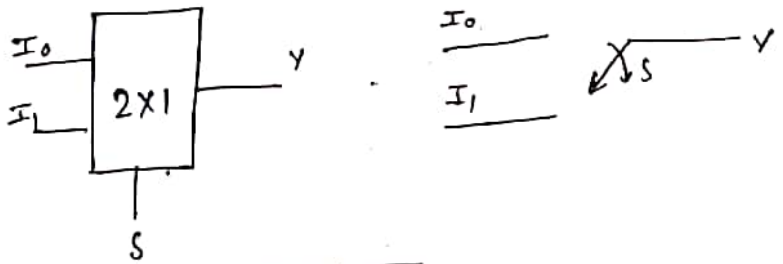
Scanned by CamScanner

MUX (Multiplexer)



MUX is a combinational ckt, which have many data i/p & single o/p.

Depending on select or control, one of the data i/p is transfer to the o/p. Hence it is known as ① Many to one ckt
② Data selector
③ Universal logic ckt
④ Parallel to serial data converter.

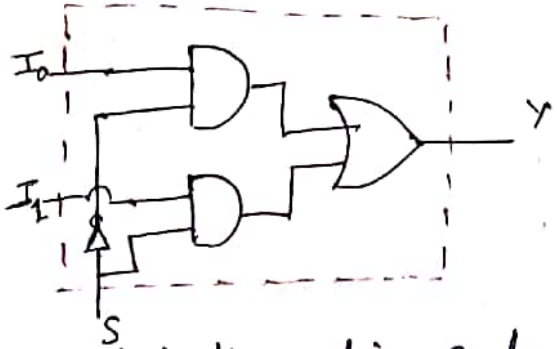
* 

Truth table ⇒

S	Y
0	I_0
1	I_1

$Y = \bar{S} \cdot I_0 + S I_1$

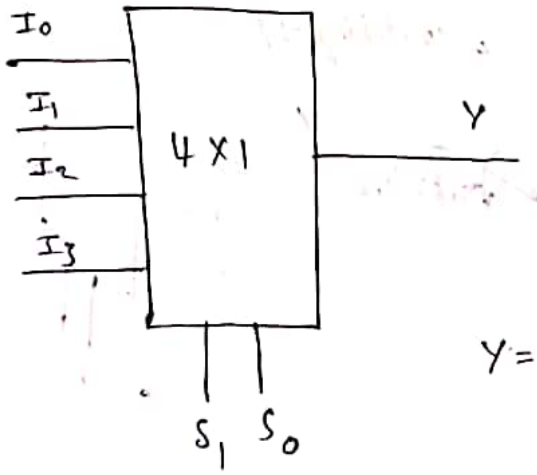
Implement ⇒



MUX basically contain 2-level AND-OR network.

Scanned by CamScanner

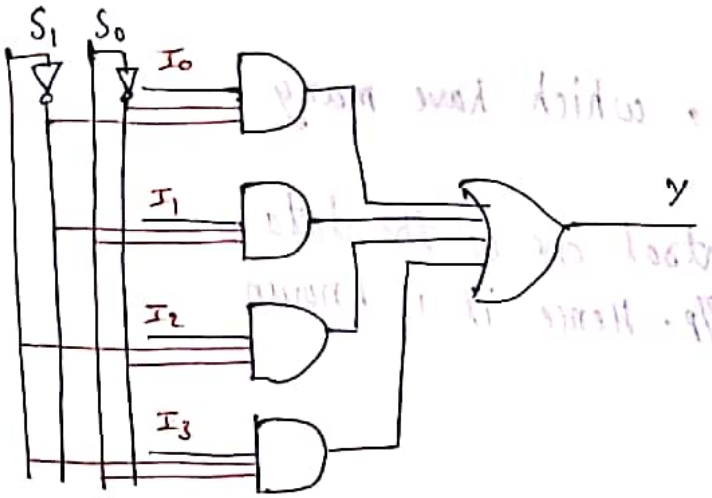
* 4-Level MUX ⇒



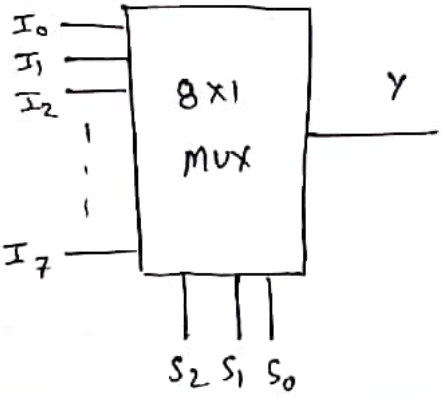
Truth table ⇒

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

$$Y = \bar{S_1}\bar{S_0}I_0 + \bar{S_1}S_0I_1 + S_1\bar{S_0}I_2 + S_1S_0I_3$$



* 8-Level MUX ⇒

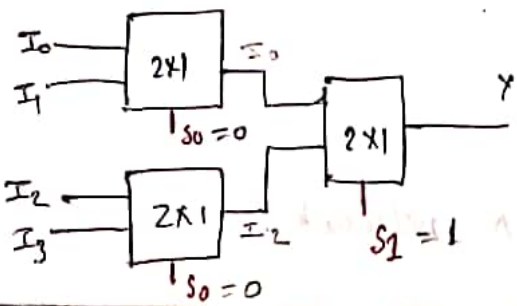


* Implementation of Higher order MUX using Lower order MUX ⇒

GATE-4

4 x 1 MUX ⟶ 2 x 1

S_1 S_0 S



$Y = I_2$

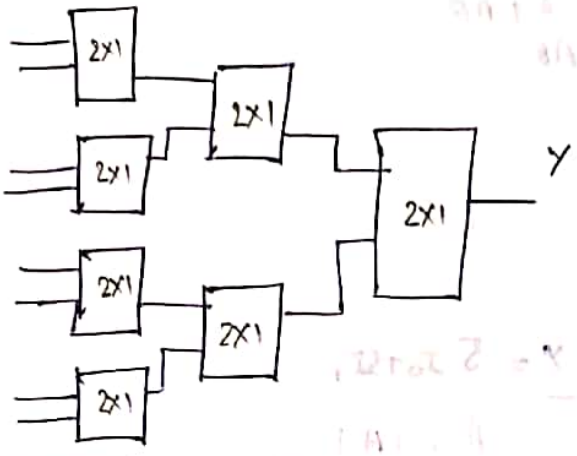
I_1 S_1 S_0

1 0 → I_2

Scanned by CamScanner

Q. To implement 8×1 MUX $\xrightarrow[4+2+1]{}$ 2×1

Ans. 7 2×1 MUX is require



Q. $16 \times 1 \xrightarrow{\text{MUX}} 15 \rightarrow 2 \times 1$

Ans. 15

$64 \times 1 \xrightarrow{63} 2 \times 1$

$256 \times 1 \xrightarrow{255} 2 \times 1$

\# $\boxed{2^n \times 1 \xrightarrow{2^n - 1} 2 \times 1}$

\# $16 \times 1 \xrightarrow[4+1]{5} 4 \times 1$

$64 \times 1 \xrightarrow[16+4+1]{21} 4 \times 1$

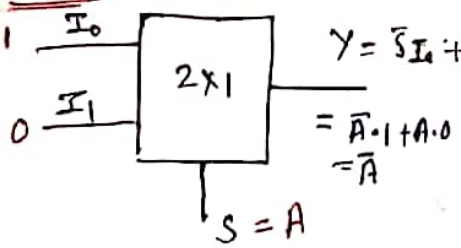
$64 \times 1 \xrightarrow[8+1]{9} 8 \times 1$

$256 \times 1 \xrightarrow[16+1]{17} 16 \times 1$

---

② MUX as a Universal logic ckt ⇒

① NOT ⇒



$Y = \bar{S} I_0 + S I_1$

$= \bar{A} \cdot 1 + A \cdot 0$

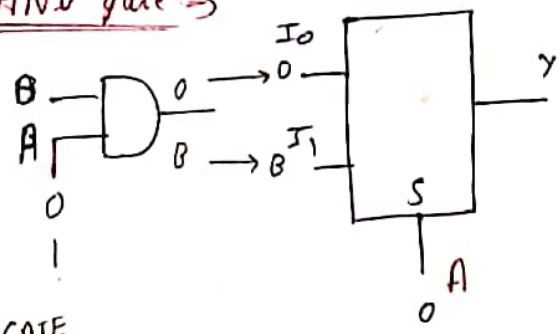
$= \bar{A}$

A	Y
0	1
1	0

Scanned by CamScanner

GATE

* AND gate →



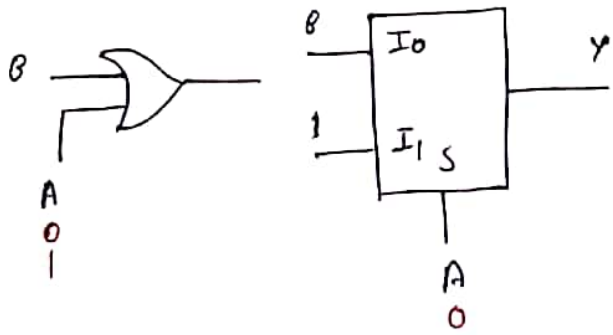
$Y = \bar{S} I_0 + S I_1$

$= \bar{A} \cdot 0 + AB$

$= AB$

GATE

* Implement OR gate →

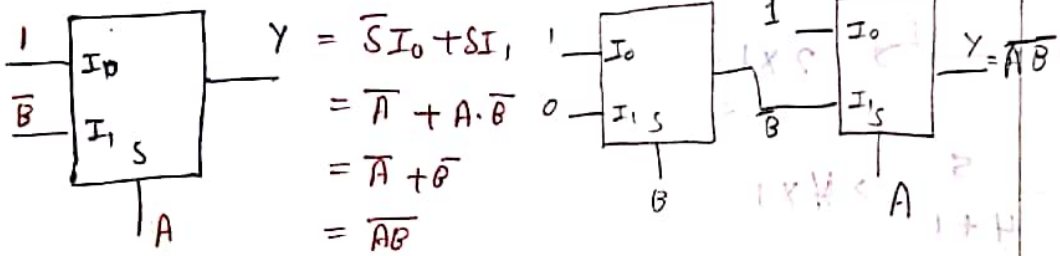


$Y = \bar{S} I_0 + S I_1$

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

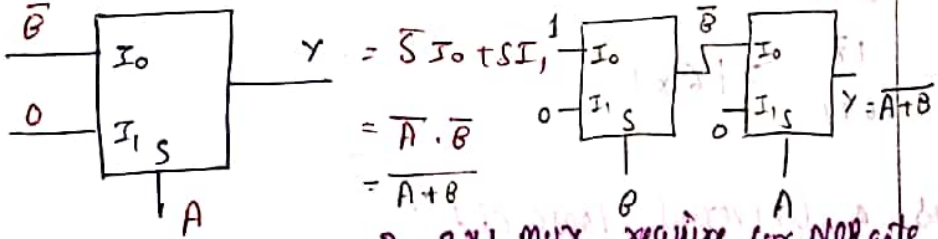
$= A + B$

* Implement of NAND gate →



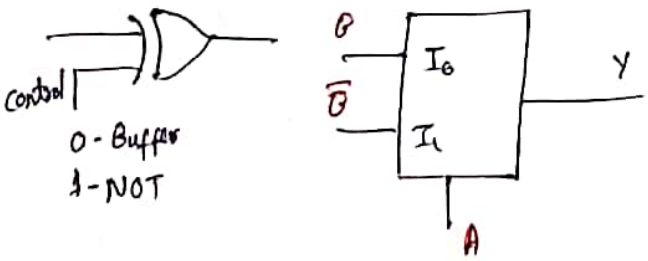
* To implement NAND gate require 2 MUX.

* Implement of NOR gate →



2 - 2x1 MUX require for NOR gate

* Implement of EX-OR gate →



$Y = \bar{S} I_0 + S I_1$

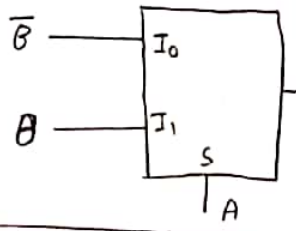
$= \bar{A} B + A \cdot \bar{B}$

$= A \oplus B$

2 - 2x1 MUX require for EX-OR gate

Scanned by CamScanner

Implement of Ex-NOR gate → XNOR = $\bar{A}\bar{B} + AB$



$Y = I_0\bar{S} + I_1 S$
$= \bar{B}\bar{A} + BA$
$= AB + \bar{A}\bar{B}$
$= XNOR$

So 2 - 2×1 MUX Require

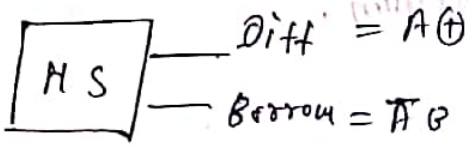
GATE-09:

Q. To implement EXOR, AND gate Min. no: of 2×1 MUX require respectively.

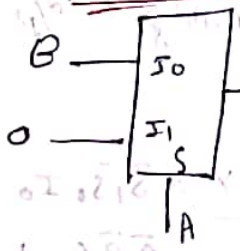
(a) 1, 1 (b) 1, 2 (c) 2, 1 (d) 2, 2

---

* HA $\xrightarrow{3}$ 2×1
* FS $\xrightarrow{3}$ 2×1



For $\bar{A}B$ ↴



$Y = \bar{S} I_0 + S I_1$
$= \bar{A} \cdot B + A \cdot 0$
$= \bar{A} \cdot B$

Ans $A \oplus B \rightarrow 2$ MUX
& $\bar{A}B \rightarrow 1$ MUX
So Total 3 MUX require.

Q. 2 i/p AND gate ⇒

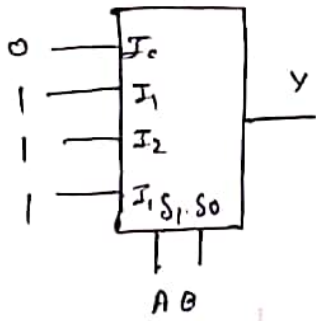
$Y = AB$

A	B
0	0
0	1
1	0
1	1

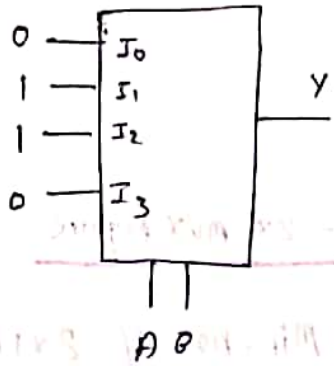
So 1 MUX require for 2 i/p AND gate.

Scanned by CamScanner

* OR



* EX-OR



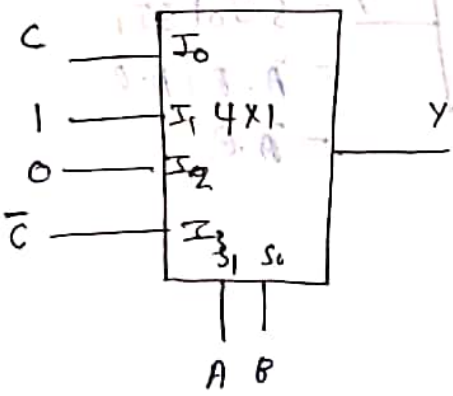
* S_0 For HA $\xrightarrow{2}$ 4x1 MUX

* HS $\xrightarrow{2}$ 4x1 MUX

Model-III

* Determine Minimized logical Expression ⇒

Q For give 4x1 mux Minimized o/p expression.



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

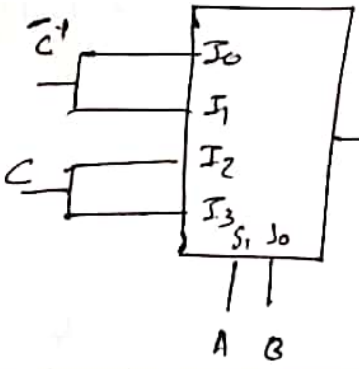
$= \bar{A}\bar{B}C + \bar{A}B \cdot 1 + A\bar{B} \cdot 0 + AB\bar{C}$

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

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

$= \bar{A}C + B\bar{C}$ Ans.

Q: Mini. o/p expression for 4x1 MUX



$Y =$

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

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

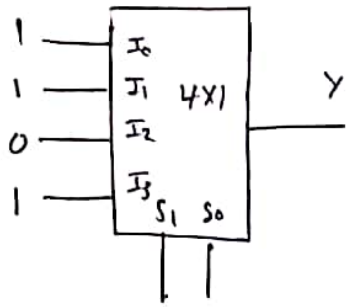
$Y = A \odot C$ Ans

Scanned by CamScanner

Model-4

* Implementation of given logical exp. ⇒

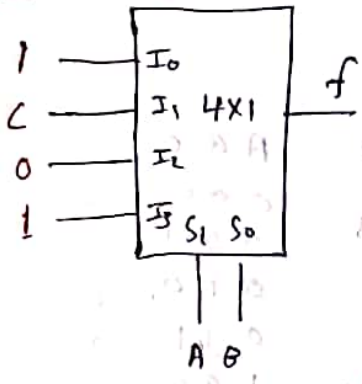
$f(A,B) = \Sigma m(0,1,3) \longrightarrow 4 \times 1$



MSB LSB

Q. Implement $f(A,B,C) = \Sigma m(0,1,3,6,7)$ using 4 x 1 MUX with A,B as select line.

Here (3 variable)



only 4x1 MUX

A	B	C	
0	0	0	0 → $\bar{C}$
1	0	0	1 → C
2	0	1	0 → $\bar{C}$
3	0	1	1 → C
4	1	0	0 → $\bar{C}$
5	1	0	1 → C
6	1	1	0 → $\bar{C}$
7	1	1	1 → C

* Implementation Table →

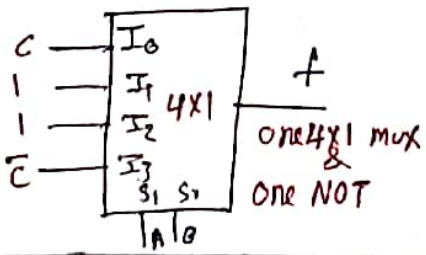
	I_0	I_1	I_2	I_3
$\bar{C}$	⓪	2	4	⑥
C	①	③	5	⑦

1 C 0 1

Q. Implement $f(A,B,C) = \Sigma m(1,2,3,4,5,6)$ using 4 x 1 MUX with:

(i) A B as select line
(ii) A C as select line
(iii) B C as select line

Sol



one 4x1 MUX & one NOT

	I_0	I_1	I_2	I_3
$\bar{C}$	0	②	④	⑥
C	①	③	⑤	7
	C	1	1	$\bar{C}$

Scanned by CamScanner

One 4×1 MUX → any 2 variable
                → Some of 3 variable

One 4×1 MUX & One NOT → any 2 variable
                        → any 3 variable

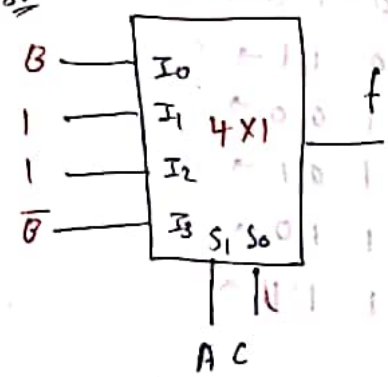
Q. GATE-2003

one 8×1 MUX → any 3 variable
                → Some of 4 variable

one 8×1 MUX & one NOT → any 3 variable
                         → any 4 variable

PSO

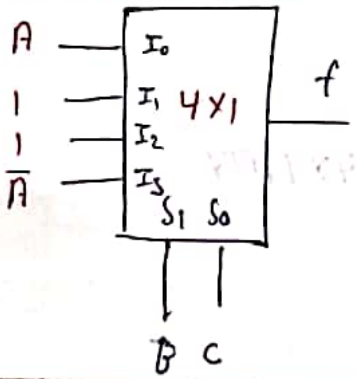
(ii) Sol. AC as select line ⇒



	I_0	I_1	I_2	I_3
$\bar{B}$	0	1	4	5
B	2	3	6	7
	B	1	1	$\bar{B}$

A	B	C	
0	0	0	$\bar{B}$
0	0	1	$\bar{B}$
0	1	0	B
0	1	1	B
1	0	0	$\bar{B}$
1	0	1	$\bar{B}$
1	1	0	B
1	1	1	B

(iii) BC as select line-

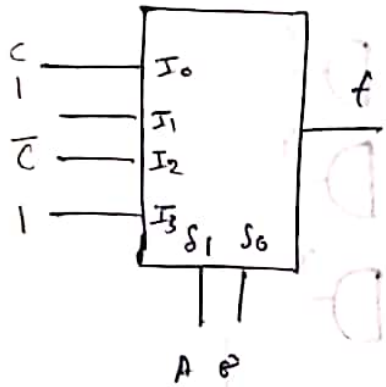


	I_0	I_1	I_2	I_3
$\bar{A}$	0	1	2	3
A	4	5	6	7
	A	1	1	$\bar{A}$

Scanned by CamScanner

Implementation of given logical exp. by K-map ⇒

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



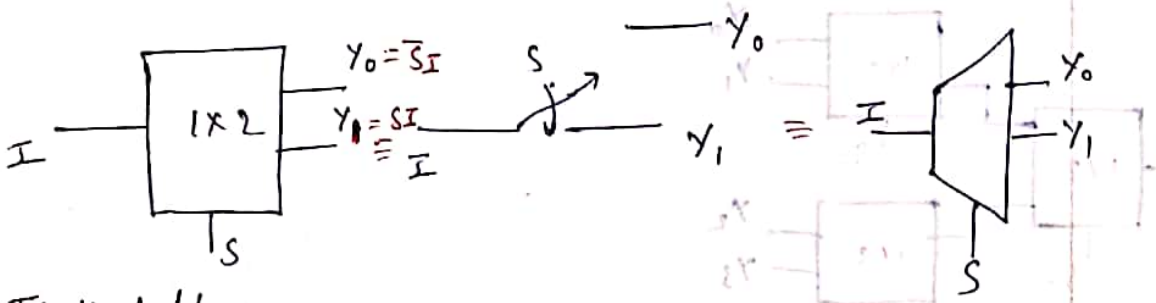
AB \ C	C̄	C	
Ā B̄ (00)	0	①	→ C → I_0
Ā B (01)	②	③	→ 1 → I_1
A B (11)	⑥	⑦	→ C̄ → I_3
A B̄ (10)	④	⑤	→ 1 → I_2

DE MUX (Demultiplexer)

DEMUX is a combinational ckt which has 1 i/p and many o/p.

Also called
- 1 to Many ckt
- Data distributer

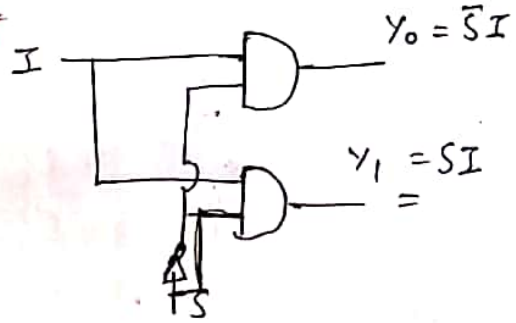
#



Truth table ⇒

S	Y_1	Y_0
0	0	I
1	I	0

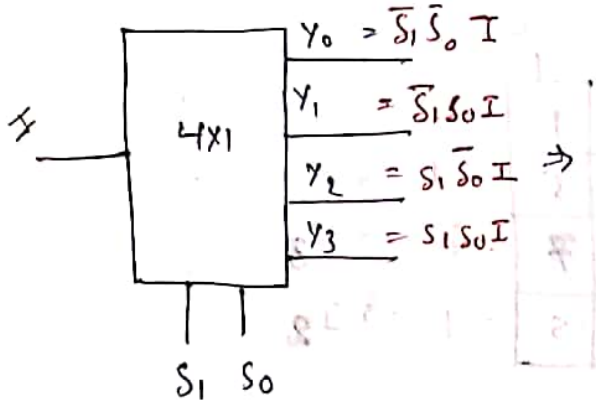
#



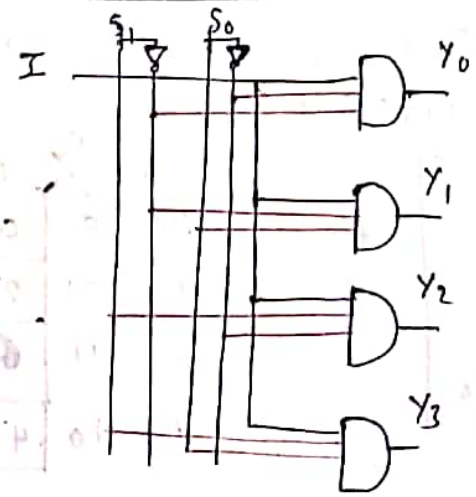
DEMUX basically contain AND gate.

Scanned by CamScanner

4 o/p DEMUX ⇒



⇒

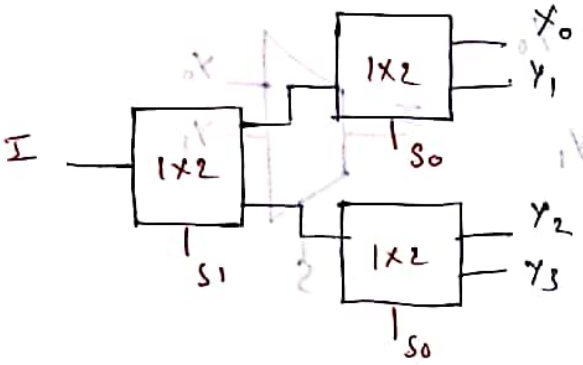


Model ⇒

* Higher order DEMUX using Lower ⇒

DEMUX 1x4 $\xrightarrow{3 = 1+2}$ 1x2

& MUX 4x1 $\xrightarrow[2+1]{3}$ 2x1



1x8 $\xrightarrow{7 = 1+2+4}$ 1x2

1x16 $\xrightarrow{5}$ 1x4

1x64 $\xrightarrow{21 = 1+4+16}$ 1x4

1x64 $\xrightarrow{9}$ 1x8

1x256 $\xrightarrow{17}$ 1x16

Scanned by CamScanner

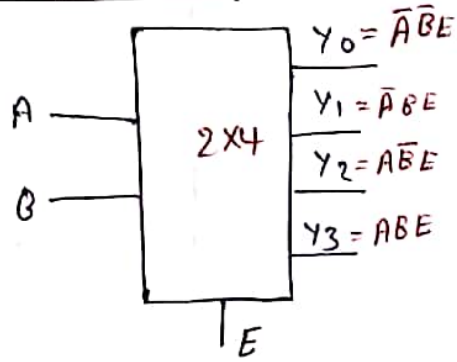
Decoder / DEMUX

(same as)

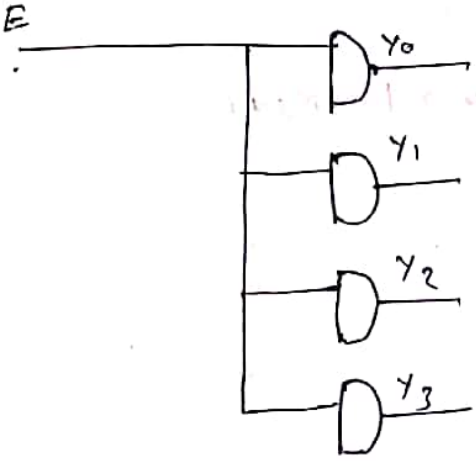
Decoder is a combination ckt which have many i/p and many o/p

Decoder is used to convert binary to other code such as * Binary to octal. (3×8)
* BCD to Decimal (4×10)
* Binary to Hexadecimal (4×16)

2 i/p Decoder ⇒



E	A	B	Y_3	Y_2	Y_1	Y_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1 ✓
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



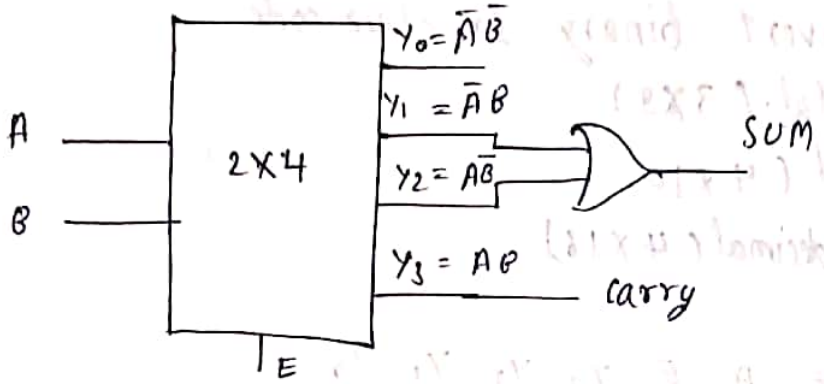
Decoder & DEMUX internal ckt is same.

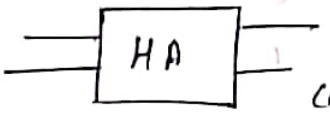
#
2×4 Decoder = 1×4 DEMUX
3×8 Decoder = 1×8 DEMUX
4×16 Decoder = 1×16 DEMUX

Scanned by CamScanner

Q: Implement HA using 2×4 Decoder ckt.

Ans. HA can be implemented using 1 - 2×4 Decoder & 1 - OR gate.



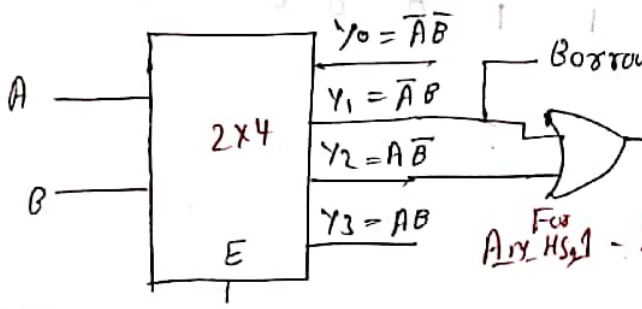


Sum = $\bar{A}B + A\bar{B}$

Carry = AB

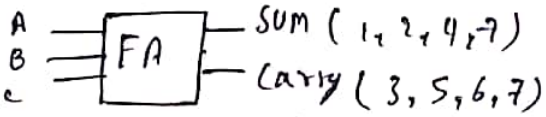
Ans. HA, 1 - 2×4 Decoder & 1 OR gate

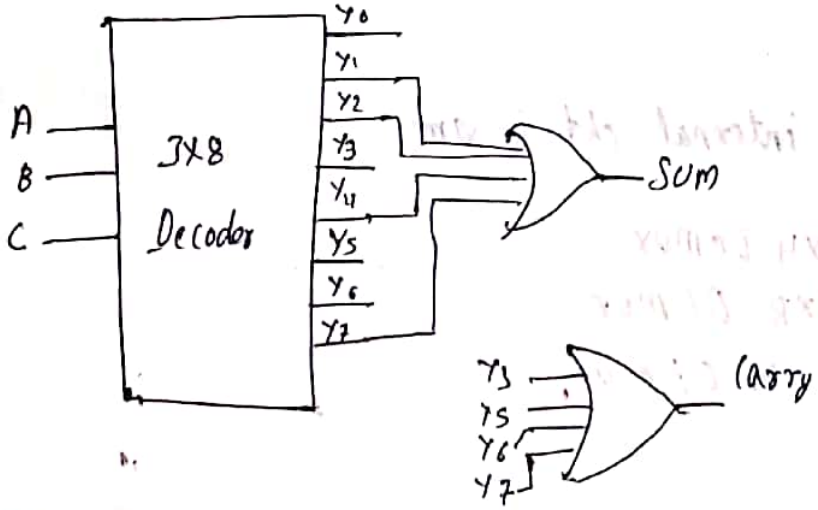
Q: HS using 2×4 Decoder.



Ans. For HS, 1 - 2×4 Decoder & 1 - OR gate

Q: FA using 3×8 Decoder





FA, 1-3×8 Decode & 2-OR gate.

Scanned by CamScanner

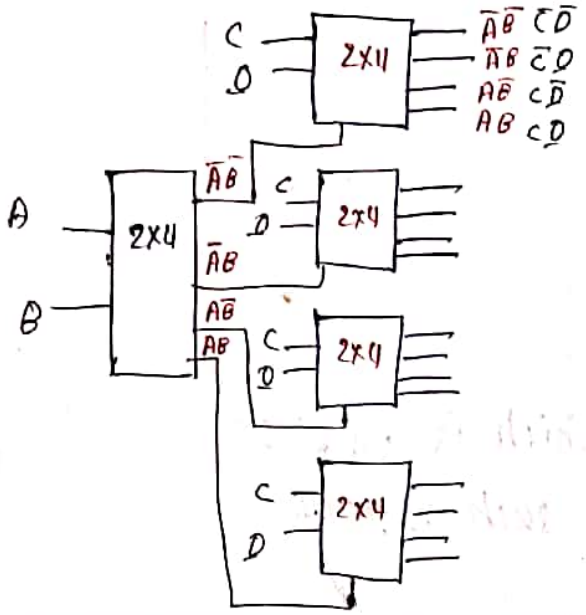
Q. To implement 4x16 Decoder, how many 2x4 Decoder require.

4x16 ⟶ 2x4

So 1x16 DEMUX $\xrightarrow{5}$ 1x4

16x1 MUX $\xrightarrow{5}$ 4x1

∴ 4x16 $\xrightarrow{5}$ 2x4 Decoder.



4x16 $\xrightarrow{5}$ 2x4

6x64 $\xrightarrow{21}$ 2x4

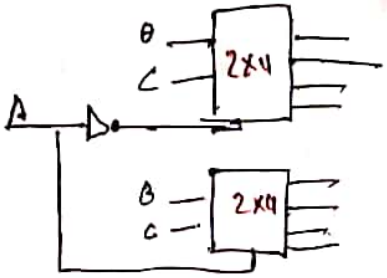
1x64 $\xrightarrow{1+4+16}$ 1x4

6x64 $\xrightarrow{9}$ 3x8

8x256 $\xrightarrow{17}$ 4x16

1x256 ⟶ 1x16

Q. To implement 3x8 How many 2x4 & NOT gate require



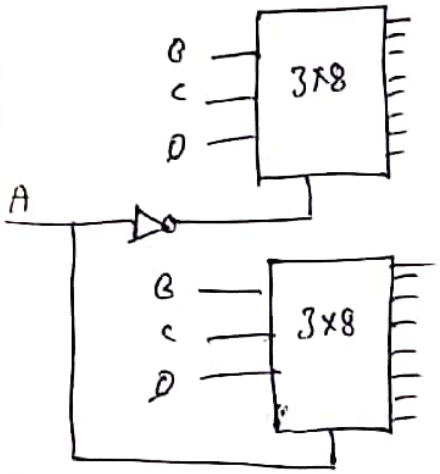
So 2 - 2x4 Decoder
& 1 - NOT gate

Scanned by CamScanner

23/06/12

Q. 4 x 16 Decoder ⟵ 2 ⟶ 3 x 8 Decoder & one NOT gate.

So 2 - (3x8) Decoder
1 - NOT gate

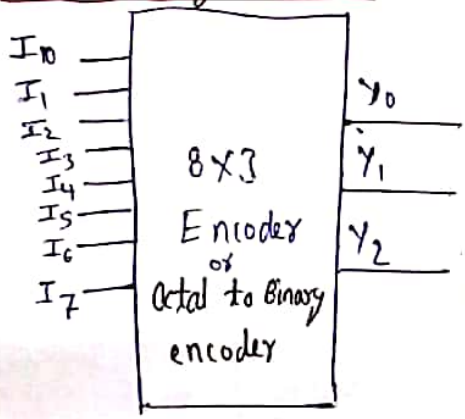


Encoder

Encoder is a combination ckt which is used to convert other code to binary such as octal to binary. (8x3)

* Decimal to BCD (10x4)
* Hex- to Binary (16x4)

Octal to Binary (8x3) ⇒



I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	Y_2	Y_1	Y_0	
0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	1	0	0	0	1	1
0	0	0	0	0	1	0	0	0	1	0	2
0	0	0	0	1	0	0	0	0	1	1	3
0	0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	0	0	0	0	1	0	1	5
0	1	0	0	0	0	0	0	1	1	0	6
1	0	0	0	0	0	0	0	1	1	1	7

$Y_0 = I_1 + I_3 + I_5 + I_7$

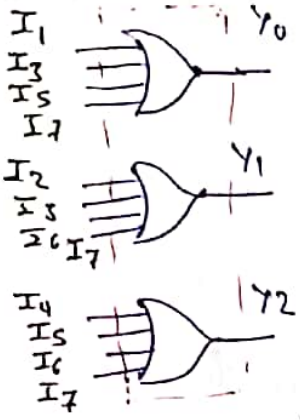
$Y_1 = I_2 + I_3 + I_6 + I_7$

$Y_2 = I_4 + I_5 + I_6 + I_7$

Scanned by CamScanner

Imp

MUX → AND - OR
Decoder / DEMUX → AND
Encoder → OR



Note ⇒

In Priority Encoder Any no. of i/p can be logic-1 but binary o/p is available corresponding to highest priority no.

Parity Generator ⇒

even odd

	A	B	C	Y (even)	Y (odd)
0	0	0	0	0	1
1	0	0	1	1	0
2	0	1	0	1	0
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	1
7	1	1	1	1	0

Even parity = i/p bit + o/p (Parity bit) = should be even
Odd parity = i/p bit + o/p (Parity bit) = shold be odd
Include

For Even Parity Generator ⇒

$Y = \sum m(1, 2, 4, 7)$

A \ BC	00	01	11	10
0		1		1
1	1		1	

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

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

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

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

$= A \oplus B \oplus C$

Scanned by CamScanner