DATE : ___/___/____
PAGE : _____

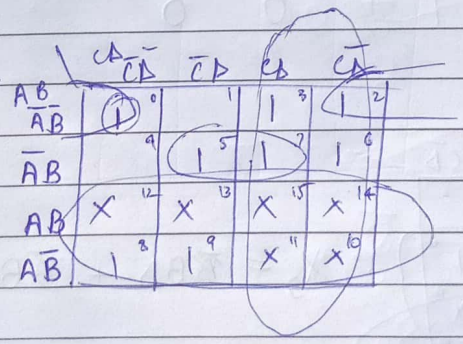
Digital logic Design

* Code Converter

BCD to 7 segment.

A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1

for a

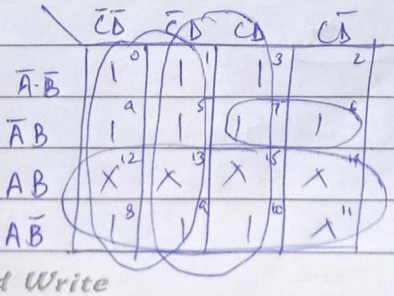


$a = C + A + BD + \bar{B}\cdot\bar{D}$

similarly for b

$b = \bar{B} + \bar{C}\cdot\bar{D} + C\cdot D.$

for c



$c = \bar{C} + D + A + \bar{A}BC$

Good Write

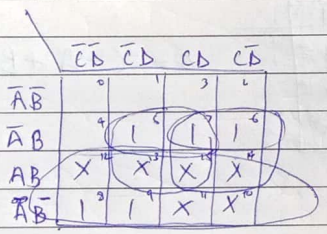
Similarly
d =

$$e = C\bar{D} + \bar{B}\bar{D}$$

BCD to excess 3.

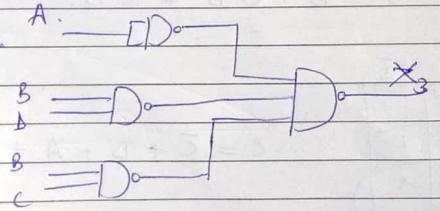
A	B	C	D	X_3	X_2	X_1	X_0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

for X_3



$$X_3 = \cancel{B}D + \cancel{B}C + \cancel{B}A.$$

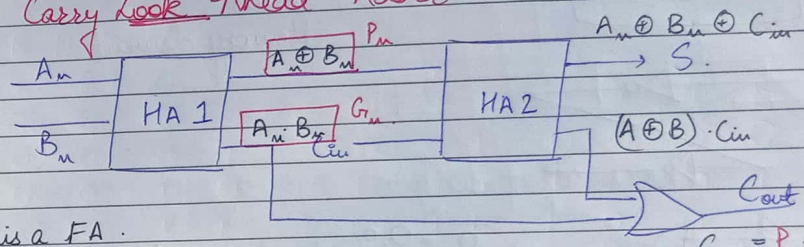
for X_2



Good Write

NOTE for NAND implementation use SOP
NOR implementation use POS.

* Carry Look Ahead Adder



This a FA.

$$C_{m+1} = P_m \cdot C_m + G_m$$

- In Ripple adder, there was a lot of delay, hence we need this.

There are 2 important things:-
1. Carry Propagate (P_m)
2. Carry Generate (G_m).

Now

$A_3\ A_2\ A_1\ A_0$
$-\ B_3\ B_2\ B_1\ A_0$ $\quad C_0 = C_{in}$

$$C_1 = P_0 C_0 + G_0$$

$$\Rightarrow C_2 = P_1 C_1 + G_1 \Rightarrow C_2 = G_1 + P_1(P_0 C_0 + G_0)$$

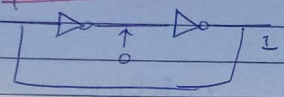
$$\Rightarrow C_3 = C_2 \cdot P_2 + G_1 \Rightarrow C_3 = G_2 + P_2(P_1(P_0 C_0 + G_0))$$

Hence we can find carry using just inputs.
⇒ saves time.

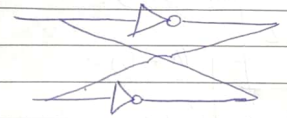
$$\Rightarrow C_m = G_{m-1} + P_{m-1} \cdot G_{m-2} + P_{m-1} \cdot P_{m-2} \cdot G_{m-3} + \cdots$$
$$\cdots + P_{m-1} \cdot P_{m-2} \cdots P_2 G_0$$

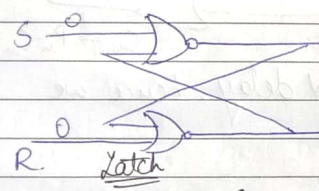
Good Write

* Sequential Circuit
* Latches



1 gets stored in the circuit
Memory element





$\bar{Q}_n = 0,0$

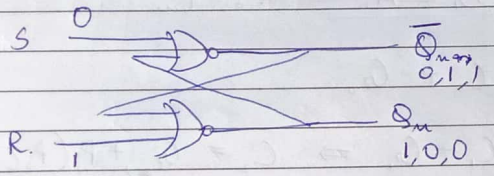
$\bar{Q}_n = 1,1$

S & R are control inputs

Q_{n-1}	Q_n	Q_{n+1}
previous state	current state	next state

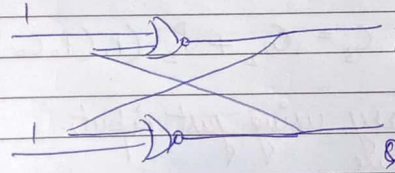
S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	Q_n
1	1	Invalid

assuming $Q_n = 1$



$\bar{Q}_n = 0,1,1$

$Q_n = 1,0,0$



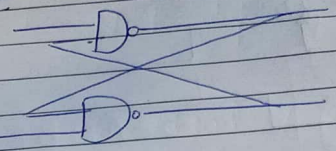
$\bar{Q}_n = 0,0,0$

$Q_n = 1,0,0$

$Q_n = \bar{Q}_n = 0$
Invalid.

Good Write

Case

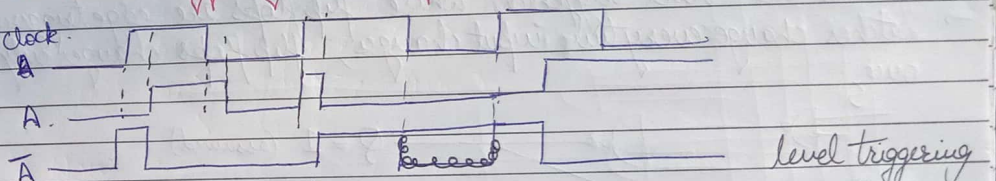


$Q_n (= 0, 0)$ 0,0
$= 0$

$\bar{Q}_n (= 1, 1)$ 1,1
$= 1$

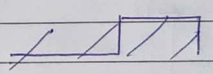
S	R	Q_{n+1}
0	0	Invalid.
0	1	1 set
1	0	0 (take Q_n = 0 & 1 & check) Reset.
1	1	Q_n no change

* Level Triggering & Edge triggering.

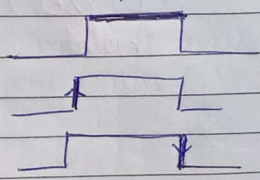


level triggering

- clock is like enable.



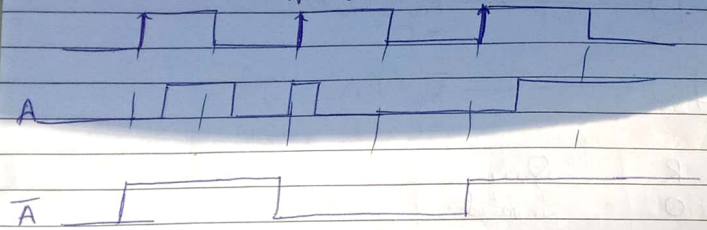
Level Triggering
Whenever clock is high, input is reflected in output accordingly. (output)
When clock is negative, its previous state is retain



Level triggering
+ve edge "
-ve edge "

Good Write

⇒ Positive Edge triggering



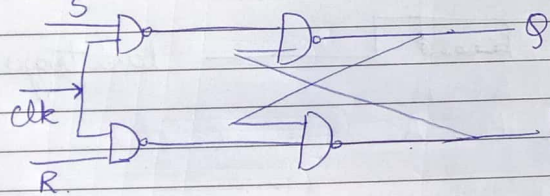
Check input only at +ve edge.

Similarly in -ve edge, check at -ve edge.

* Flip Flops

→ Latches are level triggered while flip flops are edge triggered

→ Latches change every time input changes, flip flops changes only once.



clk	S	R	Q_{n+1}
0	x	x	x
1	0	0	Q_n
1	0	1	0 (Reset).
1	1	0	1 (set)
1	1	1	Invalid.

Good Write

* Programmable Logic Device

- made of AND & OR gates.

	AND	OR
PROM (Prog. Read only Memory)	F (Fixed)	P (Prog.)
PAL (Prog. Array Logic)	P	F
PLA (Prog. As Logic Array)	P	P.

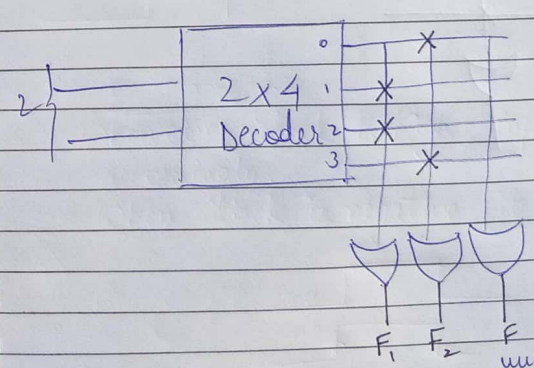
PROM. (Prog. Read only Memory)

AND → fixed OR array = prog.

- Memory is usually of form. $2^k \times n$ (k → input, n → OR gate.)

eg 4 x 3 PROM

input k - 2 3 OR gate.



given

$F_1 = A \oplus B = m_1 + m_2$

$F_2 = A \odot B$

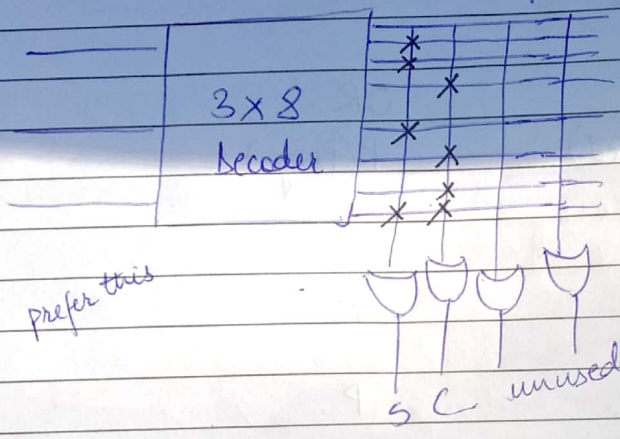
$= m_0 + m_3$

x ⇒ Fuse.

Good Write

DATE : ___/___/____
PAGE : _____

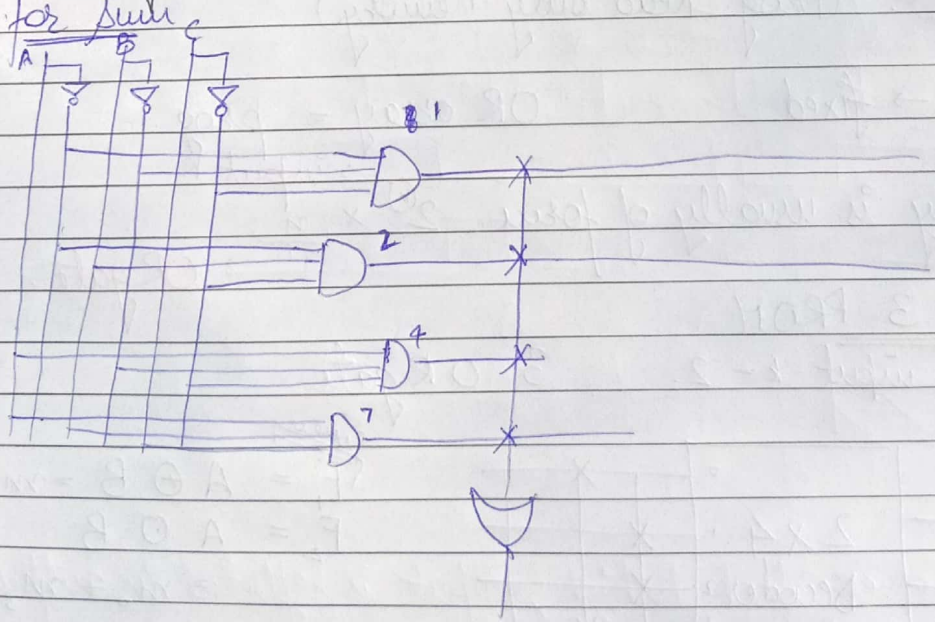
Q.) Design FA by 8x4 PROM.
3 input 4 OR gate.



$S = A \oplus B \oplus C$
$= m\bar{A}\bar{B}C + \bar{A}B\bar{C}$
$+ A\bar{B}\bar{C} + ABC$
$= 1, 2, 4, 7.$

Carry = 3, 5, 6, 7.

Fuse Diagram
for sum



Good Write