FLIP FLOPS

→ computers use OR, AND, XOR and other gates for decision making, binary addition and subtraction. multiplication and division are performed through repeated addition and subtraction.

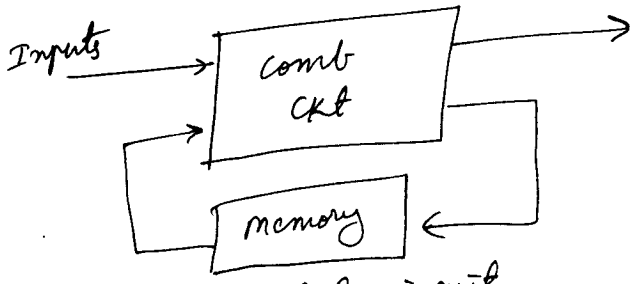
→ In addition, a computer needs memory or storage elements for storing a binary digit.

→ These storage devices are also digital devices with two stable states. By the application of trigger pulse to one of the inputs, the states can be changed from one to the another.

A storage device is heart of a binary counting system.

→ the flip flops are widely used as switches, latches, counters, registers and memory cells in computers.

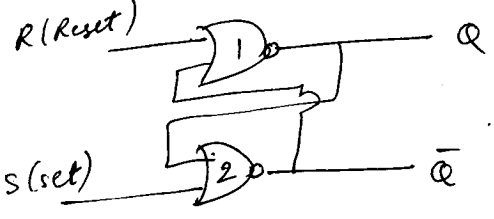
→ Since the information is locked or latched in the FF, the FF are also called as latch.



output depends on present inputs & present state of the system

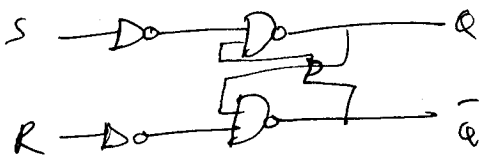
Fig: sequential circuit.

1-Bit memory element (Latch):-

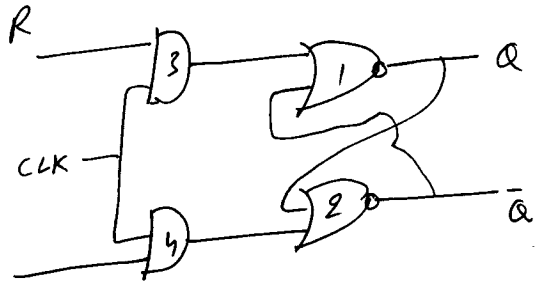


S R
1 1 → impractical state

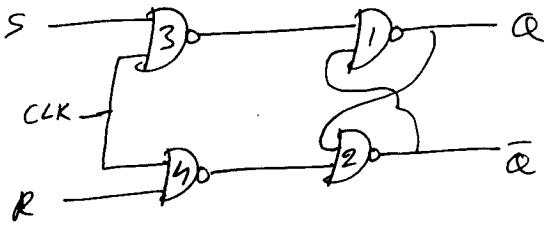
S	R	Q
0	1	0
1	0	1
0	0	No change
1	1	—

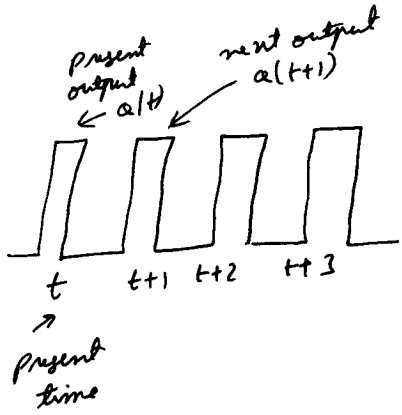


Clocked S-R flip flop :-



|← steering gates →| ← Latch ⟶|





Truth table

S	R	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	(Ambiguous state)

Characteristic table

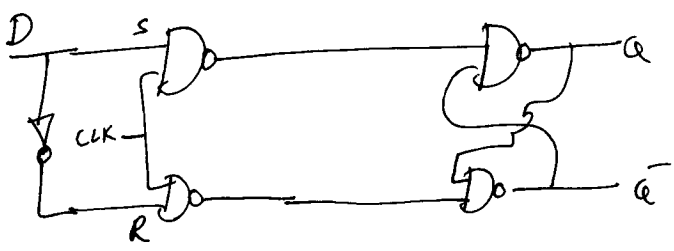
S	R	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	X
1	1	1	X

Characteristic equation.

S \ RQ(t)	00	01	11	10
0	0	1	0	0
1	1	1	X	X

$$Q(t+1) = S + \bar{R}Q$$

Clocked D - flip flop :-



Truth table

D	Q(t+1)
0	0
1	1

Characteristic table

D	Q(t)	Q(t+1)
0	0	0
0	1	0
1	0	1
1	1	1

$Q(t+1) = D\bar{Q} + DQ$

$\Rightarrow Q(t+1) = D$

Clocked JK flip flop :-

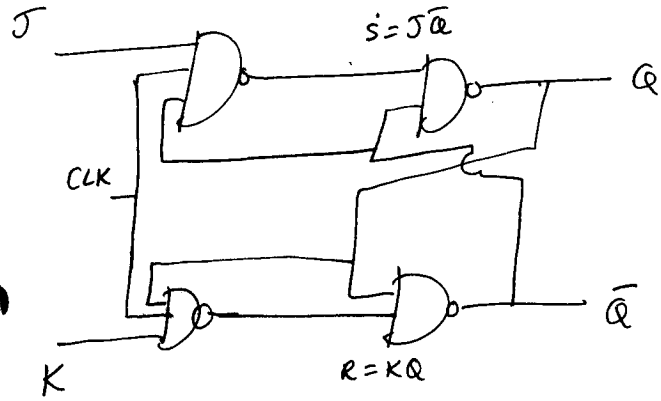
→ JK flip flop is the most widely used type of flip flop.

→ J and K designations for inputs have no known significance except that they are adjacent letters in the alphabet.

→ The functioning of J-K flip flop is identical to that of R-S flip flop in RESET, SET, and No change conditions of operation.

The difference is that the J-K flip flop has no invalid state as does the R-S flip flop.

→ JK flip flop is widely used in digital devices such as counters, registers, arithmetic logic units, and other digital systems

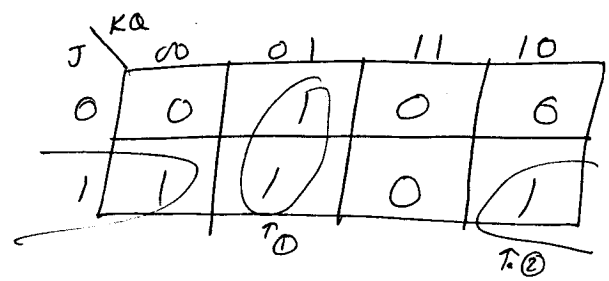


Truth table

J	K	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	Q̄(t)

Characteristic table

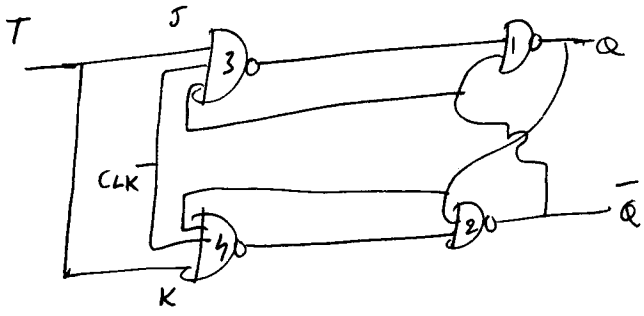
J	K	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0 → 1	
1	1	1 → 0	



$$Q(t+1) = J\bar{Q} + \bar{K}Q.$$

Clocked T-flip flop

(Toggle)



Truth table

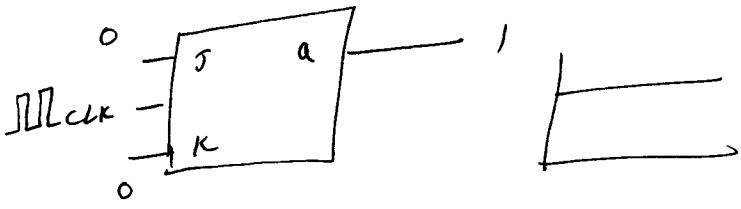
T	Q(t+1)
0	Q(t)
1	$\bar{Q}$(t)

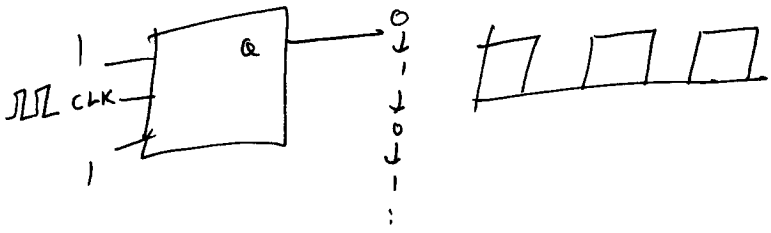
Characteristic table

T	Q(t)	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

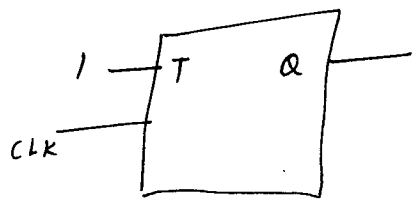
$Q(t+1) = \bar{T}\,Q(t) + T\,\bar{Q} =$

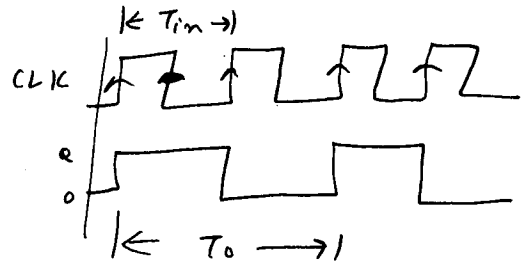
$= T \oplus Q.$





1) Determine the output frequency of the following flip flop if the clock frequency is 1 KHz.





$T_0 = 2T_{in} \Rightarrow f_0 = \frac{1}{2} T_{in}.$

Sol

$f_0 = 500\ Hz$

$= 0.5\ KHz.$

→ Toggle flip flop acts as a frequency divider.

Types of Triggering

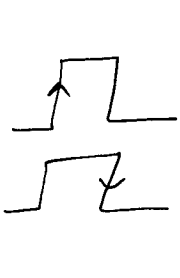
(1) Level triggering

← can be used with D.c i/p.

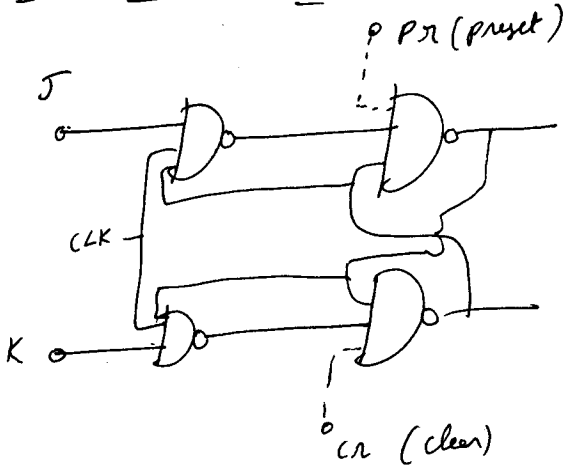
(b) Edge triggering

(i) +ve edge triggered

(ii) -ve edge triggered

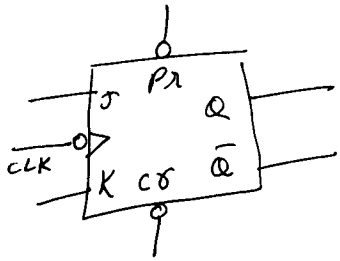


Asynchronous (or) Direct inputs



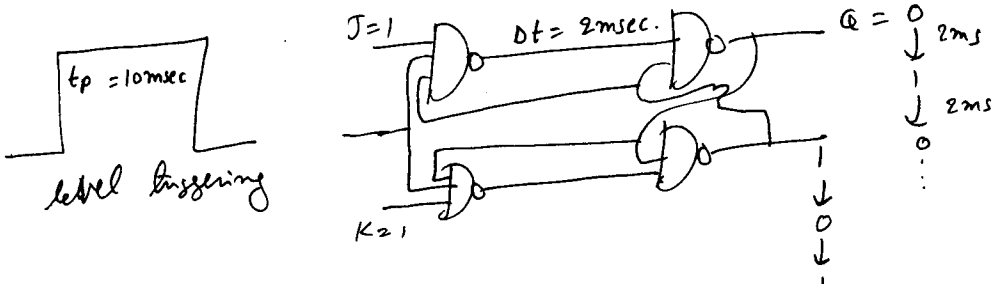
CLK	PR	CR	Q
0	0	1	1
0	1	0	0
⊓	1	1	o/p depends on J & K i/ps

→ when the clock pulse is applied PR & CR must be high in order to get proper functioning.



Race around condition

J=1, K=1 (in JK flip flop)



when $tp >> \Delta t \Rightarrow$ Race around condition

where the o/p toggles many times.

Note: The Race around condition does not occur in edge triggered flip flop

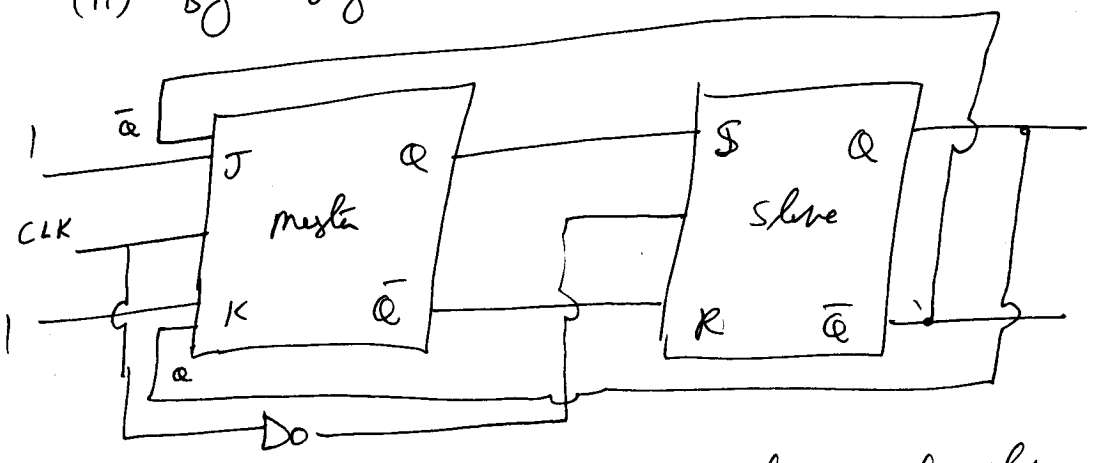
It occurs only in level triggered flip flops.

→ To avoid race around condition $\boxed{t_p \leq \Delta t}$

→ Race around condition in J-K flip flop can be avoided by making $\boxed{t_p \leq \Delta t}$, this is achieved

(i) By introducing the delay elements in the feed back path of J-K flip flop.

(ii) By using master-slave J-K flip flop.



CLK = 1 ⇒ master is active and slave is inactive.

Boolean Algebra

AND law, OR law.

(1) Commutative law (2) Associative law (3) Distribution law (4) Absorption law.

(5) Consensus law (6) Transposition law (7) Demorgans law (8) Duality property

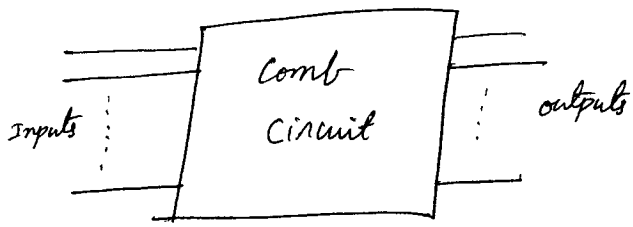
→ min term, max term. , → SOP, POS form.

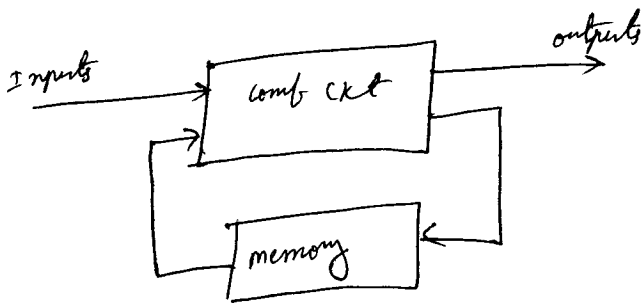
→ Degenerative & non Degenerative form.

→ K-map.

→ Combinational CKTs.

Combinational circuits :-

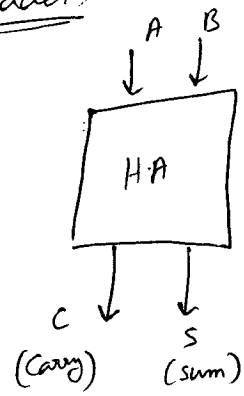




sequencial CKt.- the output depends on present input & present state (previous outputs) of the system.

Arithmatic combinational circuits :-

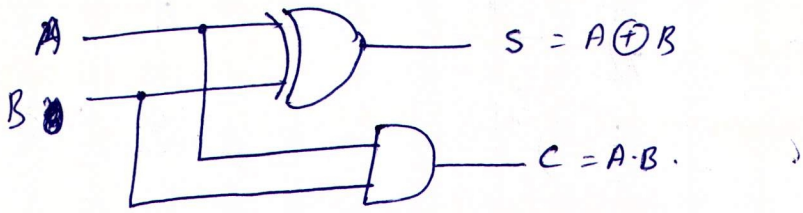
(1) Half adder :



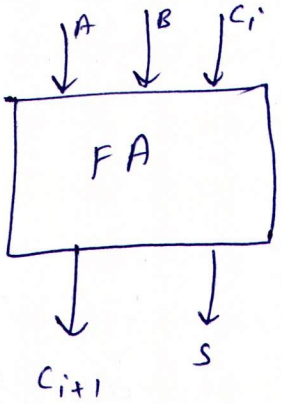
Truth table

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

$S = \bar{A}B + A\bar{B} = A \oplus B$; $C = A \cdot B$.



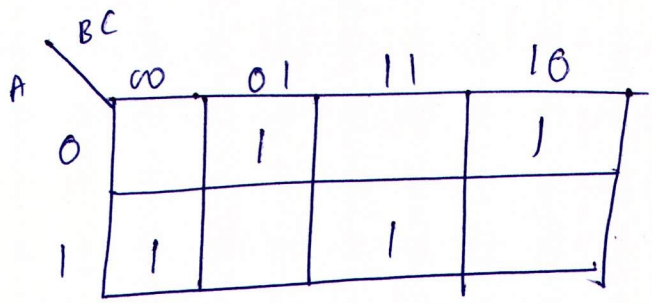
(2) Full adder:-

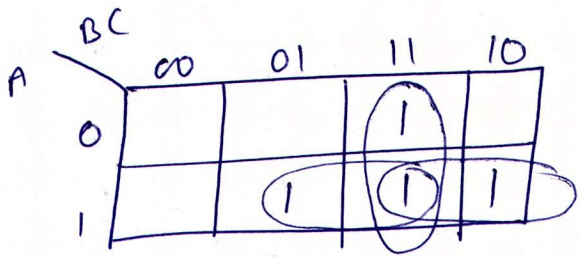


A	B	C_i	S	C_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$S = A \oplus B \oplus C_i$$

$$C_{i+1} = C_i (A \oplus B) + AB.$$





$$S = \bar{A}\bar{B}C_i + A\bar{B}\bar{C}_i + ABC_i + \bar{A}B\bar{C}_i$$

$$S = \cancel{A \oplus (B \oplus C)}\ (A \oplus B) \oplus C$$

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

$$= \bar{A}B\bar{C} + A\bar{B}\bar{C} + \left[\overline{(\bar{A}B)} \cdot \overline{(A\bar{B})}\right]C$$

$$= \bar{A}B\bar{C} + A\bar{B}\bar{C} + \left[(A + \bar{B}) \cdot (\bar{A} + B)\right]C$$

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

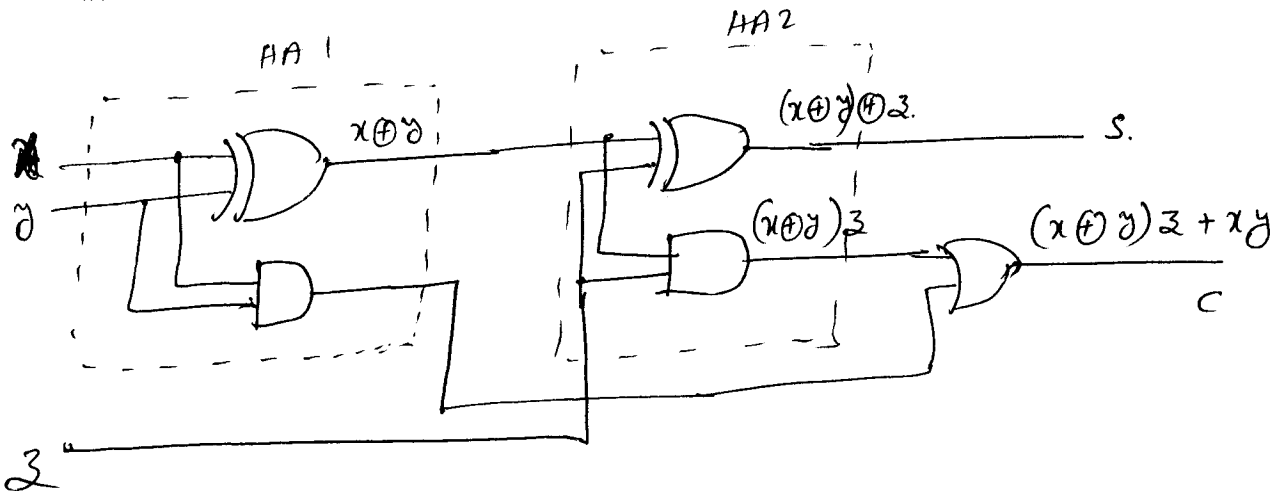
$$C_{i+1} = \bar{A}BC_i + A\bar{B}C_i + AB$$

(or)

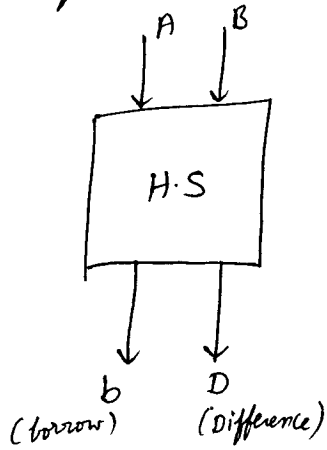
$$= \bar{A}BC_i + A\bar{B}C_i + AB\bar{C}_i + ABC_i$$

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

$$= (A \oplus B)C_i + AB.$$



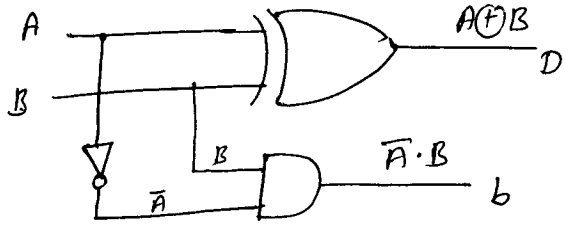
(3) Half subtractor:-



A	B	D	b
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$$D = A \oplus B$$

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



(4) Full subtractor:

A	B	b_i	D	b_{i+1}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$$D = A \oplus B \oplus b_i$$

$$b_{i+1} = \bar{A}\bar{B}b_i + \bar{A}B\bar{b_i} + \bar{A}Bb_i + AB b_i$$

$$= \bar{A}b_i + \bar{A}B + Bb_i$$

ex:- Implement the following simultaneous equations using only half adders.

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

$$E = ABC$$

$$F = A \oplus B \oplus C$$

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

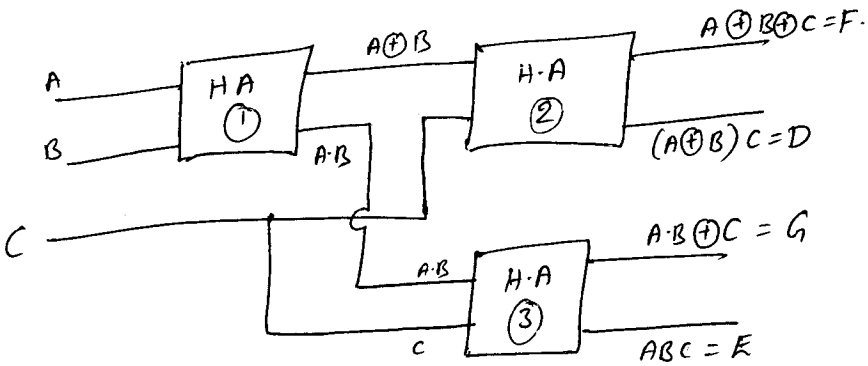
Sol:-

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

$$E = A \cdot B \cdot C$$

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

$$G = AB\bar{C} + \overline{AB} \cdot C = AB \oplus C$$



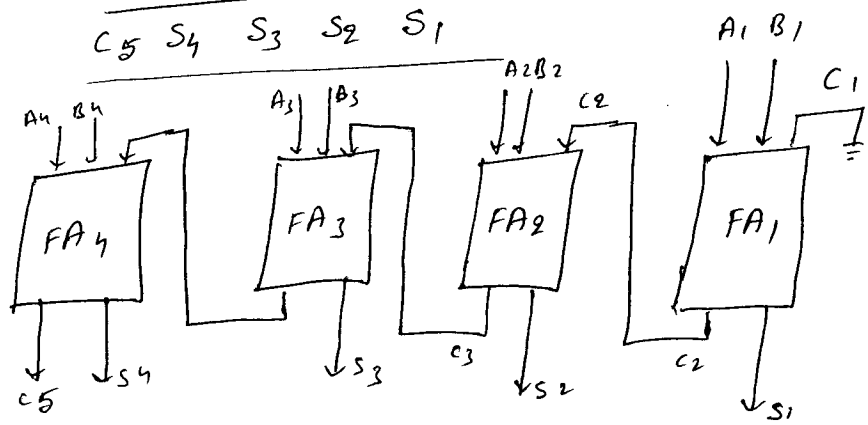
4- Bit parallel Binary Adder:-

$A = A_4\ A_3\ A_2\ A_1$

$B = B_4\ B_3\ B_2\ B_1$

$C_4\ C_3\ C_2$

$C_5\ S_4\ S_3\ S_2\ S_1$



dis:- 1) the speed of operation is less due to propagation delays adds.

adv:- easy of implementation.