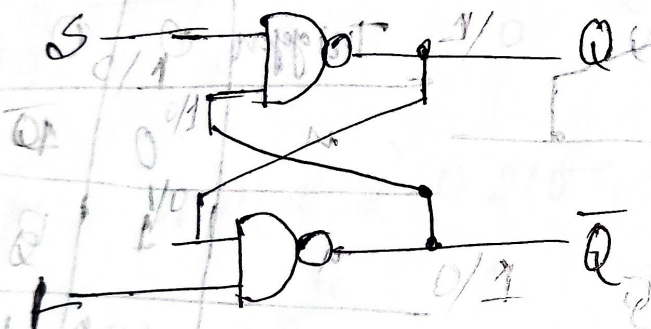


chapter: 05

1. SR latch using NAND Gate :-

		AND	NAND
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

S	R	Q(n+1)
0	0	invalid
0	1	1
1	0	0
1	1	Hold



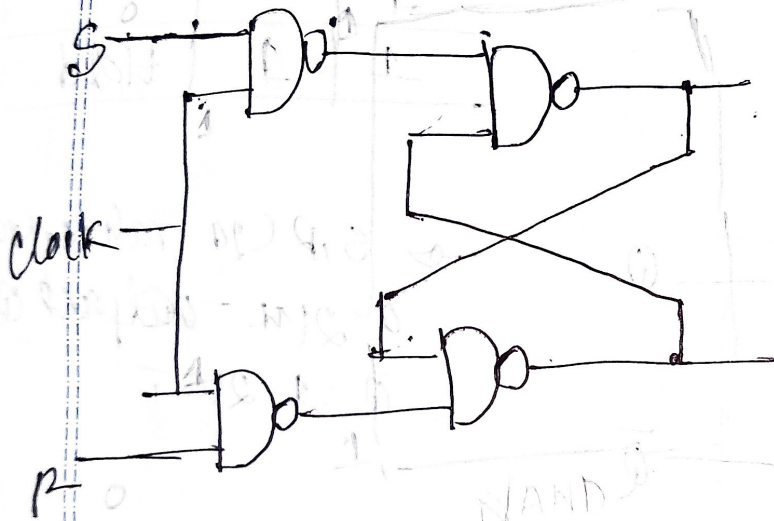
S, R (0) values (00)
 0, 2 (1) - output Q or
 Q = 1, 2 (0) 1

$$1 + Q = 1 + \bar{Q} = 0 + Q$$

$$1 + \bar{Q} = 1 + Q = Q$$

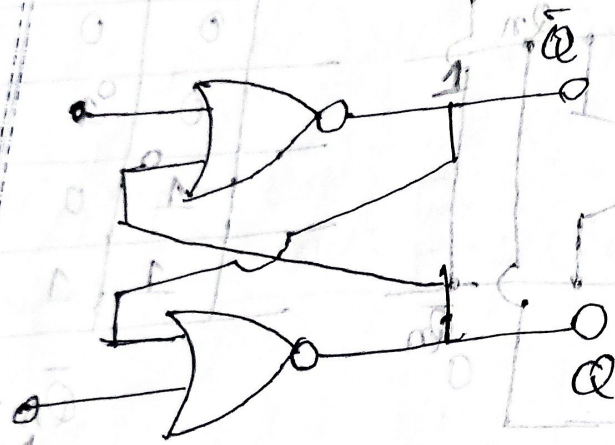
71 SR-Flip-flop using NAND gates

clock triggerere mein clock is 1 apply
 and (not)



clock	S	R	Q(m+1)
Not trigger	X	X	Q _n
Trigger	0	0	Hold
"	0	1	0 reset
"	1	0	1 - Set
"	1	1	Invalid

Ex. SR latch using NOR gate:-

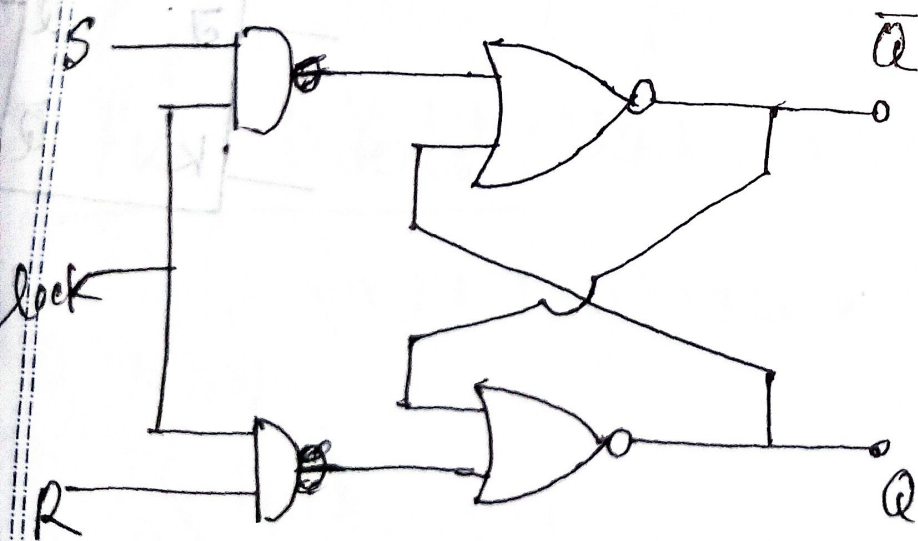


Set	Reset	$Q(n+1)$
S	R	
0	0	Hold
0	1	0
1	0	1
1	1	Invalid

→ 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

AND gate

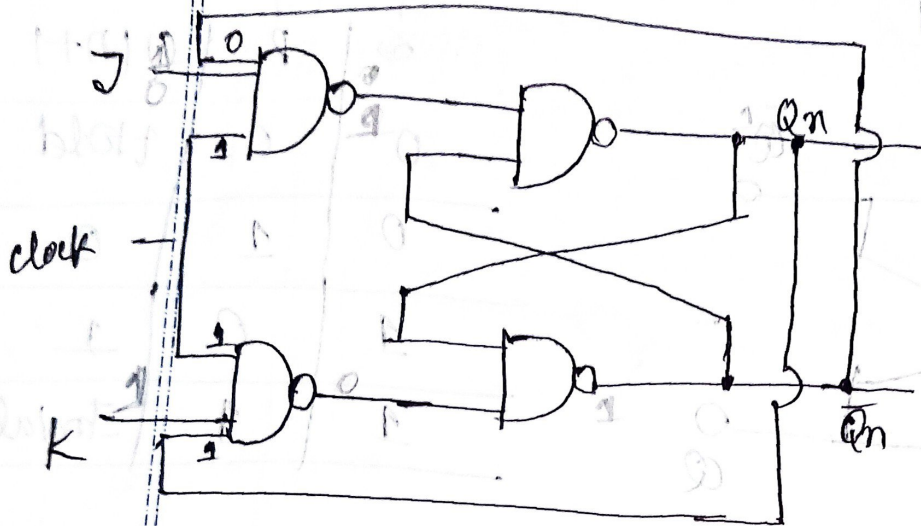
OR	NOR
0 0	0 1
0 1	1 0
1 0	1 0
1 1	1 0



C	S	R	$Q(n+1)$
0	0	0	Q_n
1	0	0	Hold
1	0	1	0
1	1	0	1
1	1	1	Invalid

Introduction to JK flip-flop:-

Date: / /



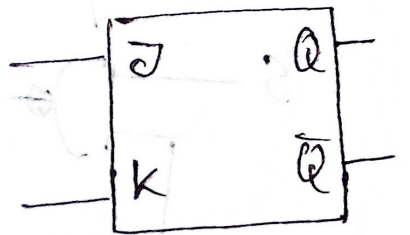
J	K	Q _{n+1}
0	0	Hold
0	1	Reset
1	0	Set
1	1	Toggle

$\bar{Q}_n \rightarrow$ Toggle

case I, $Q_n = 1$, $Q_{n+1} = 0$

case II, $Q_n = 0$, $Q_{n+1} = 1$

→ NAND. (When input 0 2nd output 1.



Q1 Half adder: -

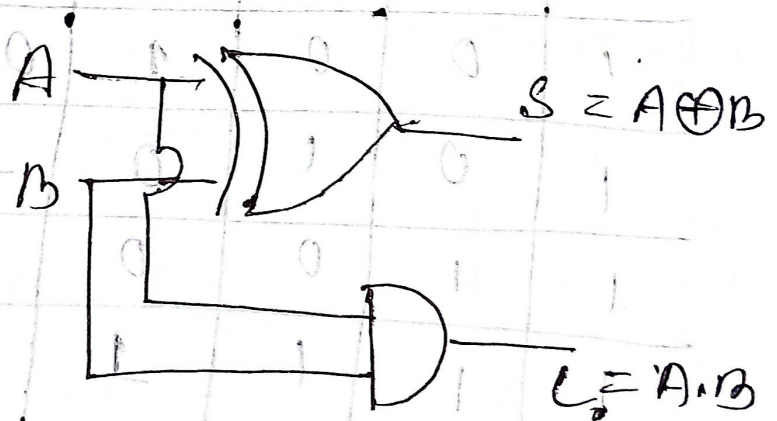
- Single bit numbers
- It doesn't take carry from previous sum.



A	B	S	Co
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$S = A \oplus B$$

$$Co = A \cdot B$$

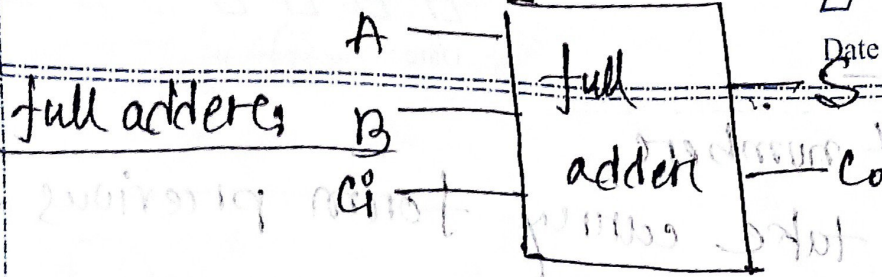


Q2 full adder:

$$\begin{aligned}
 & (m_0 A + m_1 \bar{A}) + (m_2 \bar{A} + m_3 A) = m_0 + m_3 \\
 & (m_0 + m_2) \bar{A} + (m_1 + m_3) A = \\
 & (m_0 + m_2) \bar{A} + (m_1 + m_3) A = \\
 & (m_0 \oplus m_2) \bar{A} + (m_1 \oplus m_3) A =
 \end{aligned}$$

Sat / Sun / Mon / Tue / Wed / Thu / Fri
☐ ☐ ☐ ☐ ☐ ☐ ☐

Date :



A	B	C_{in}	S	C_o
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

$$C \oplus A = 2$$

$$B \oplus A = 0$$

$$Sum = \bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}_{in} + A\bar{B}\bar{C}_{in} + AB\bar{C}_{in}$$

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

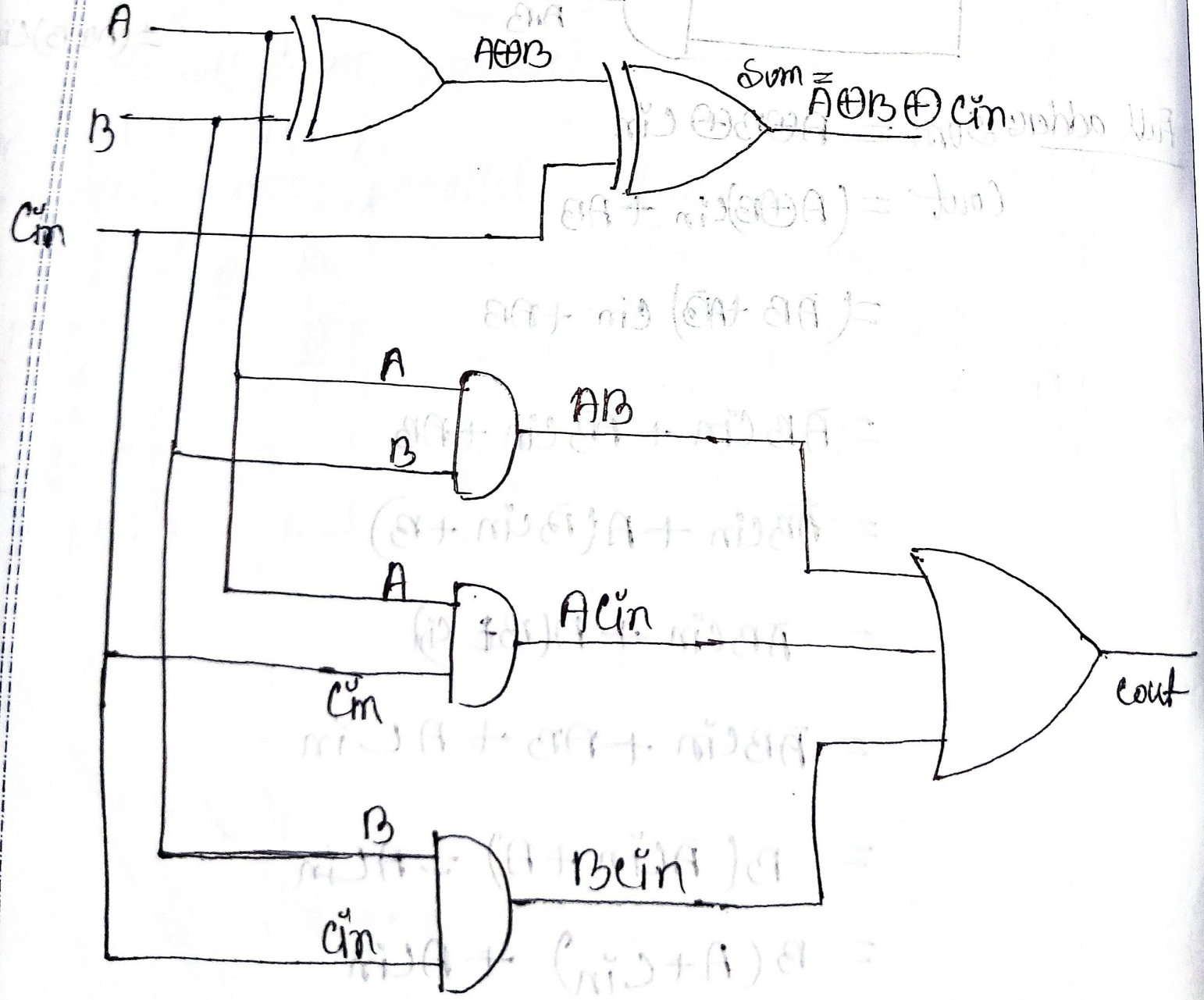
$$= \bar{B}C_{in} +$$

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

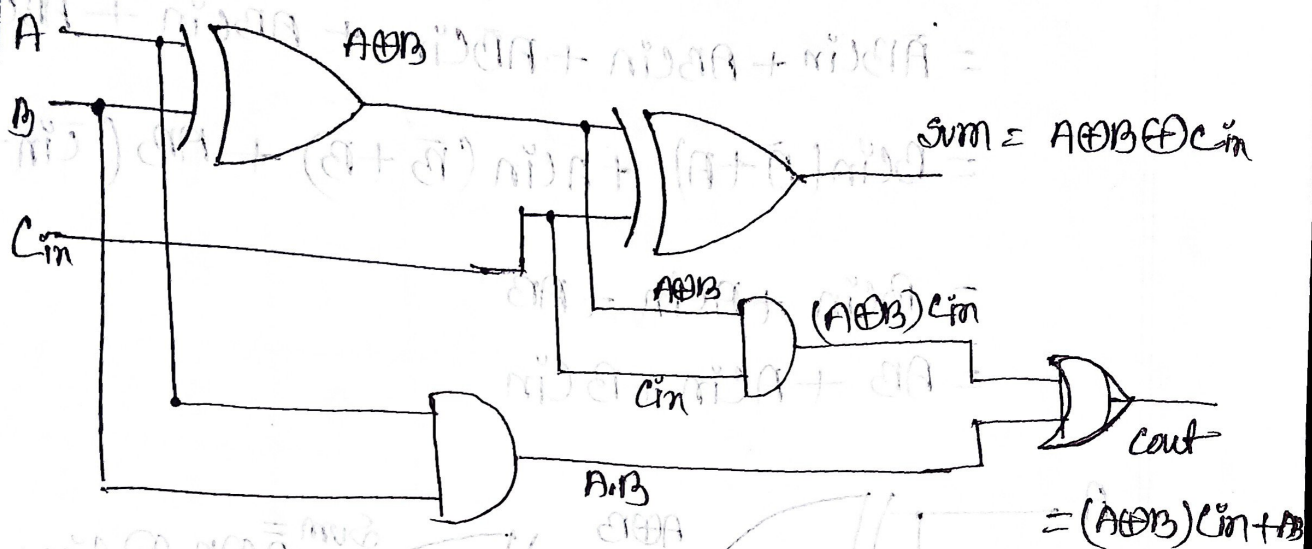
$$= \bar{A}(B \oplus C_{in}) + A(\overline{B \oplus C_{in}})$$

$$= A \oplus B \oplus C_{in}$$

$$\begin{aligned}
 \text{cout} &= \bar{A}B\bar{c}_{in} + A\bar{B}\bar{c}_{in} + AB\bar{c}_{in} + A\bar{B}c_{in} + A\bar{B}c_{in} + ABc_{in} + ABc_{in} + ABc_{in} \\
 &= \bar{A}B\bar{c}_{in} + AB\bar{c}_{in} + A\bar{B}c_{in} + ABc_{in} + A\bar{B}c_{in} + ABc_{in} + ABc_{in} + ABc_{in} \\
 &= B\bar{c}_{in}(\bar{A} + A) + A\bar{c}_{in}(\bar{B} + B) + AB(\bar{c}_{in} + c_{in}) \\
 &= B\bar{c}_{in} + A\bar{c}_{in} + AB \\
 &= AB + A\bar{c}_{in} + B\bar{c}_{in}
 \end{aligned}$$



Q1 Full adder using two half adders:



Full adder: $Sum = A \oplus B \oplus C_{in}$

$$Cout = (A \oplus B)C_{in} + AB$$

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

$$= \bar{A}BC_{in} + A\bar{B}C_{in} + AB$$

$$= \bar{A}BC_{in} + A(\bar{B}C_{in} + B)$$

$$= \bar{A}BC_{in} + A(B + C_{in})$$

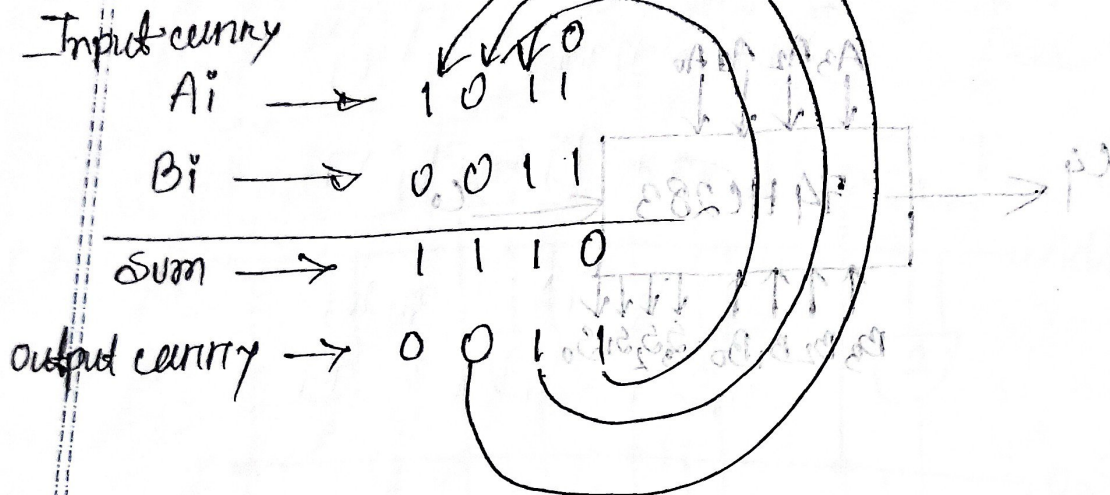
$$= \bar{A}BC_{in} + AB + AC_{in}$$

$$= B(\bar{A}C_{in} + A) + AC_{in}$$

$$= B(A + C_{in}) + AC_{in}$$

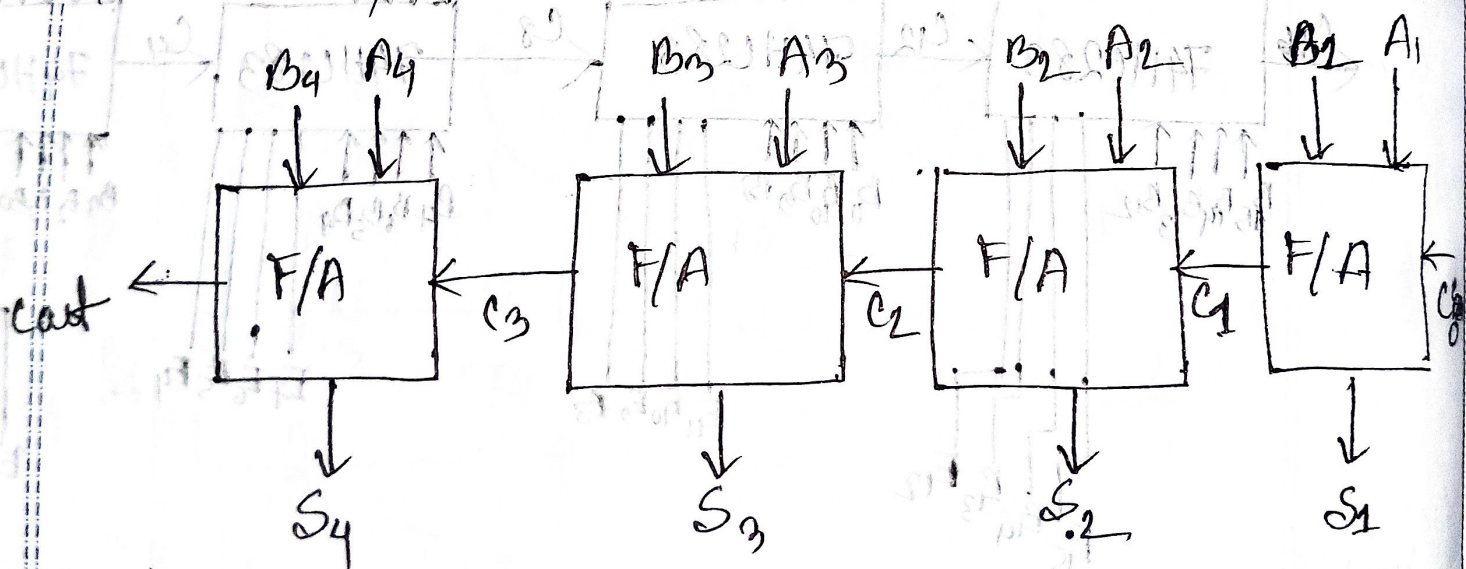
$$= AB + BC_{in} + AC_{in}$$

4 bit parallel adder:



$\rightarrow$ n bit binary parallel adder $\rightarrow$ n bit full adder

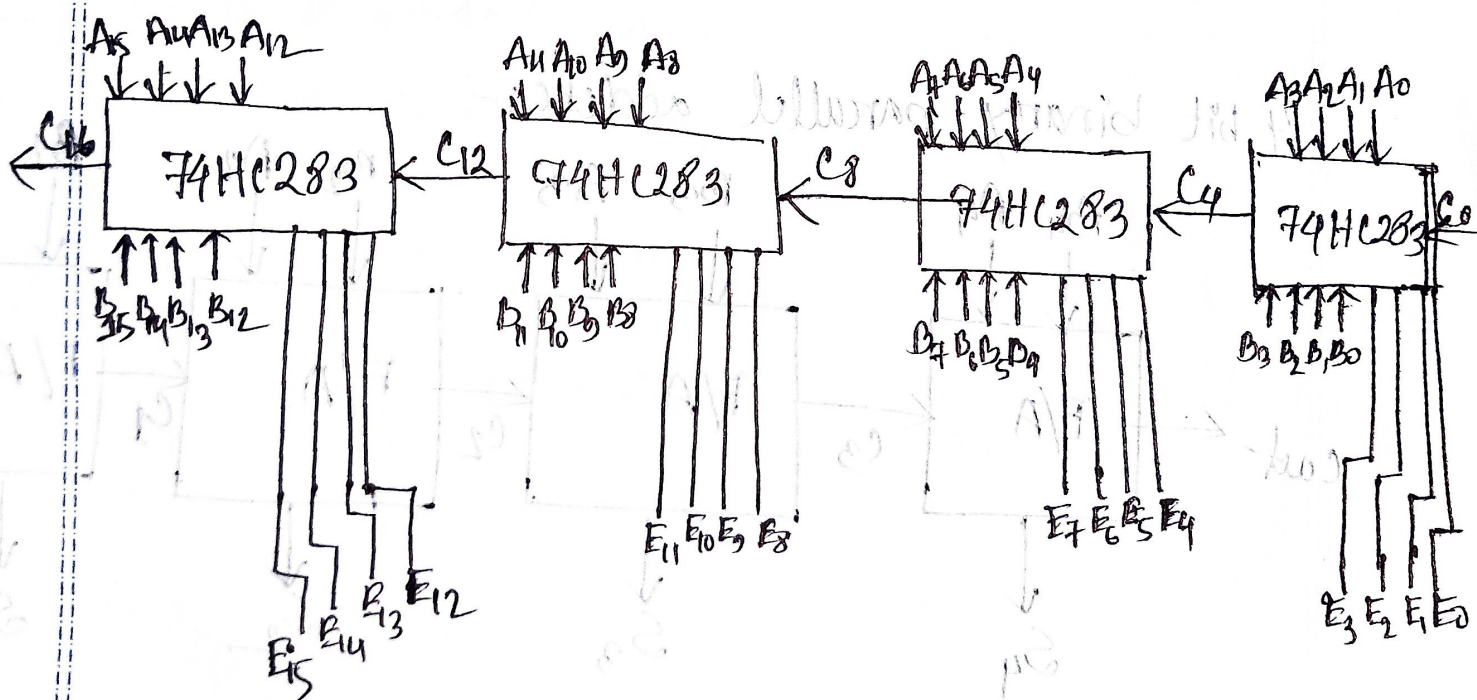
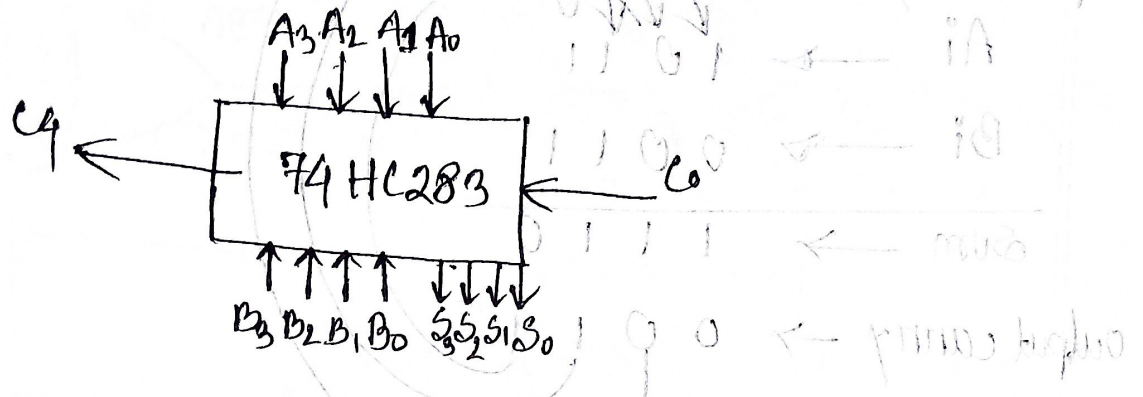
4 bit binary parallel adder:-



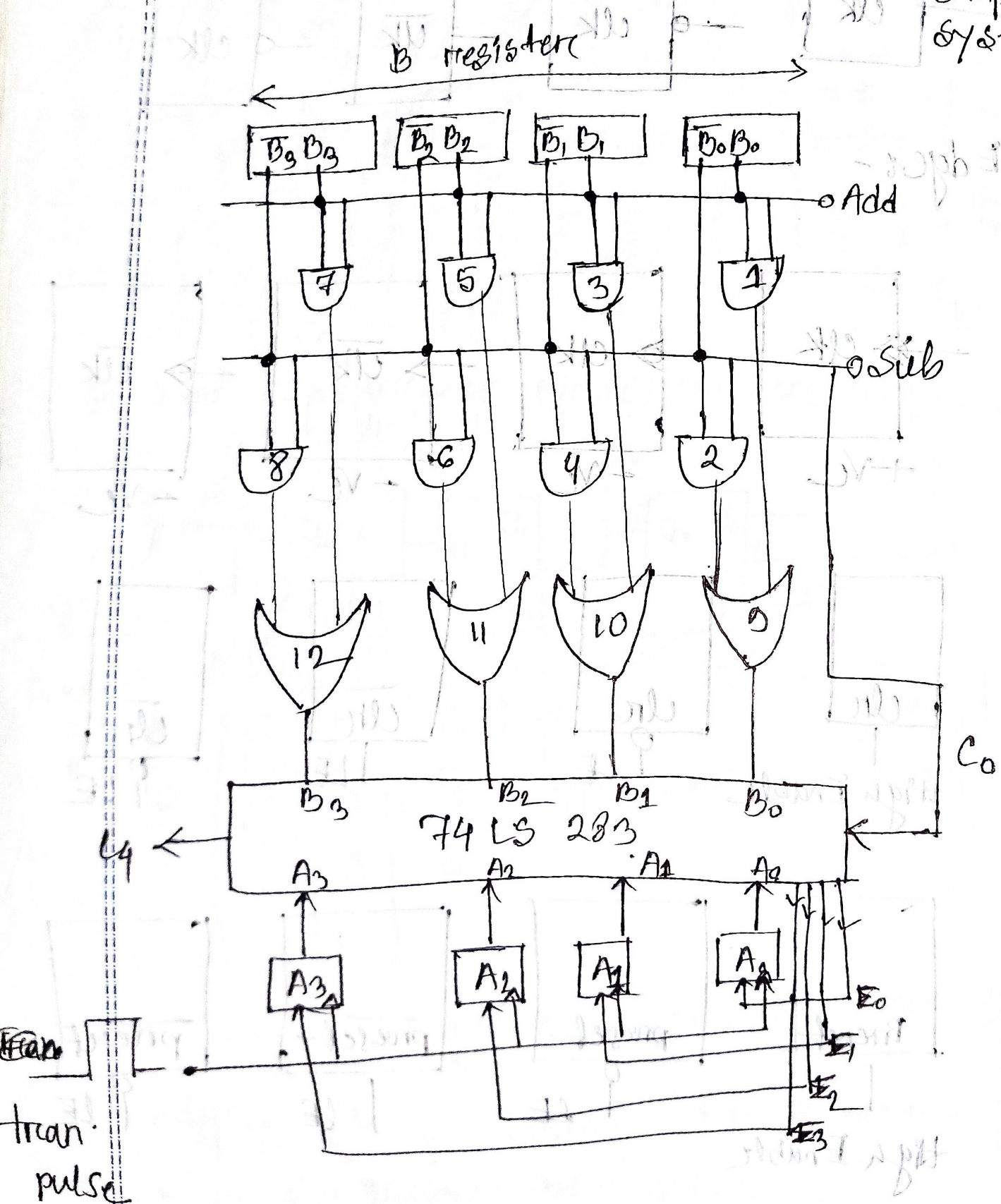
Here,

~~A~~ Augend bits from A register
 & Addend bits from B register
 also sum appears at S_4, S_3, S_2, S_1 outputs.

44 74HC283 parallel adder:



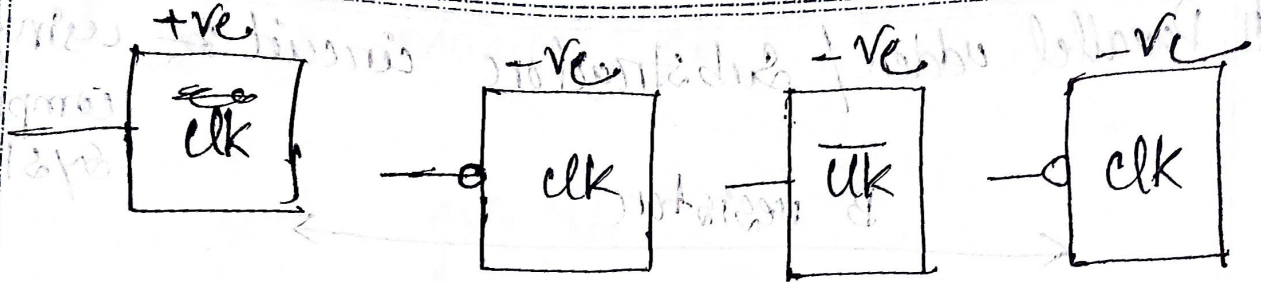
Q1 Parallel adder / Subtractor circuit using 2's complement system:-



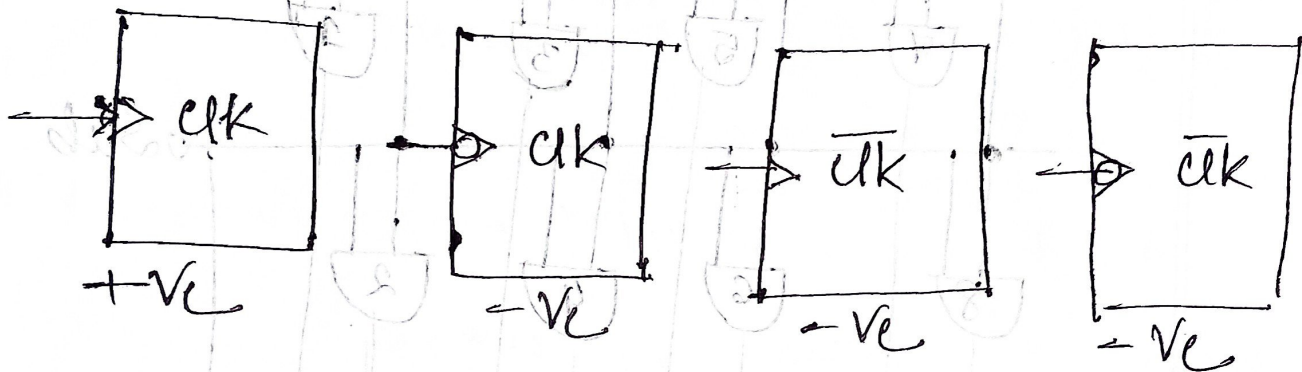
chapter 07

Sat / Sun / Mon / Tue / Wed / Thu / Fri
☐ ☐ ☐ ☐ ☐ ☐ ☐
 Date:/...../.....

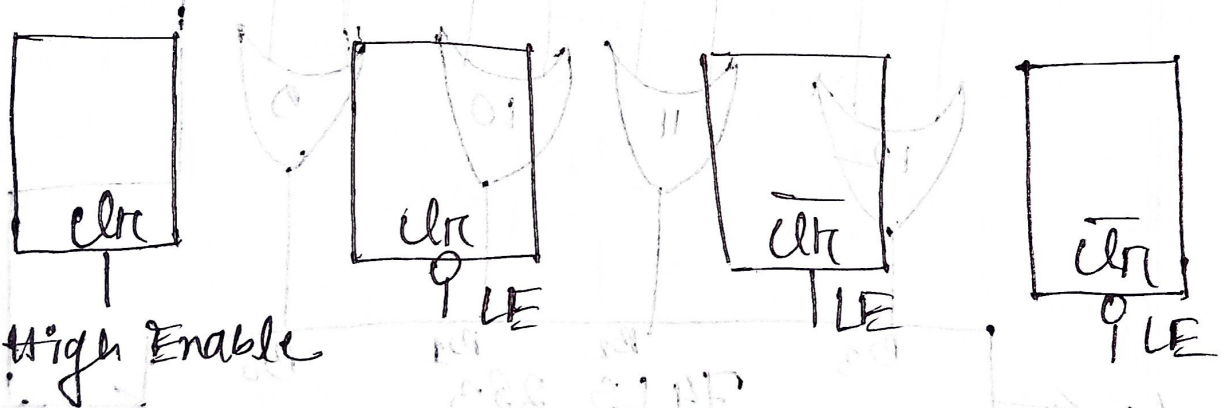
level:



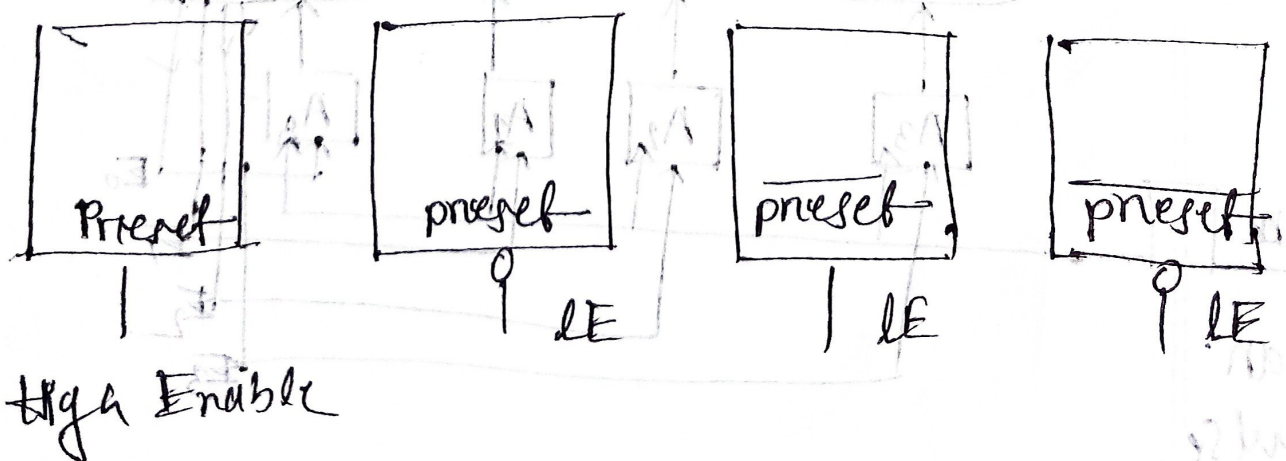
Edge:



Clear:

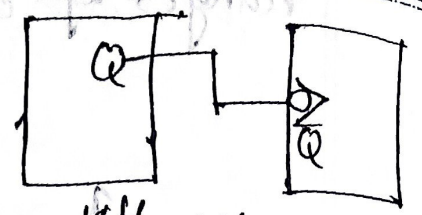


Preset:

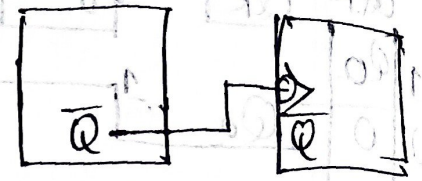


	$+V_c$	$-V_c$
	Q	$\bar{Q}$
$+V_c \rightarrow$	Down	up
$-V_c \rightarrow$	up	down

Same - Down
 diff - up

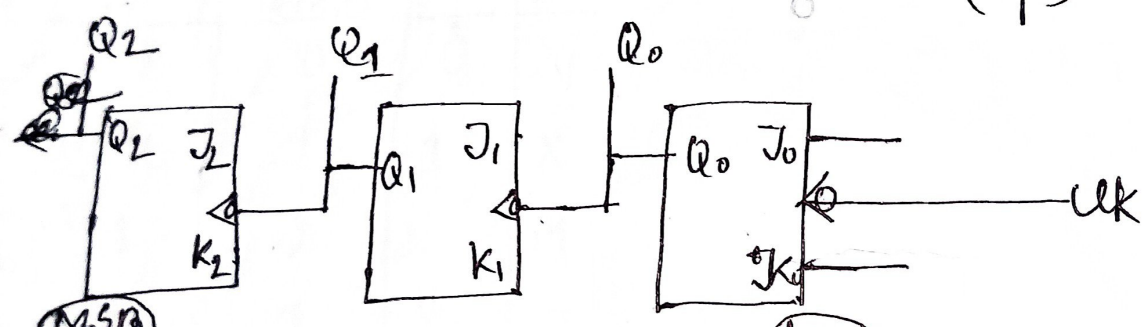


diff - up



Same - Down

3 bit asynchronous (Ripple) counter: (up)



Toggle:

Present

$Q(t)$	J	K	$Q(t+1)$ - next
0	1	1	1
1	1	1	0

opposite of $Q(t)$

Toggle 2 (MSB)
 - or 1 flip-flop
 each time

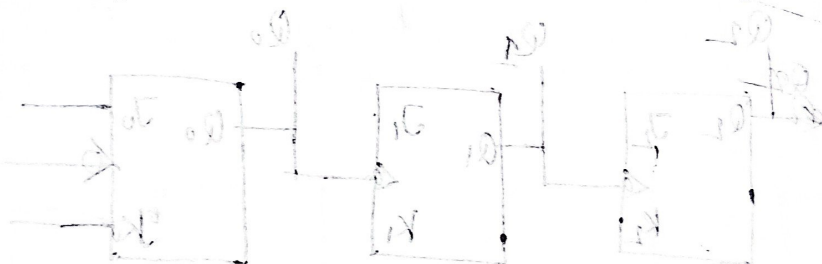
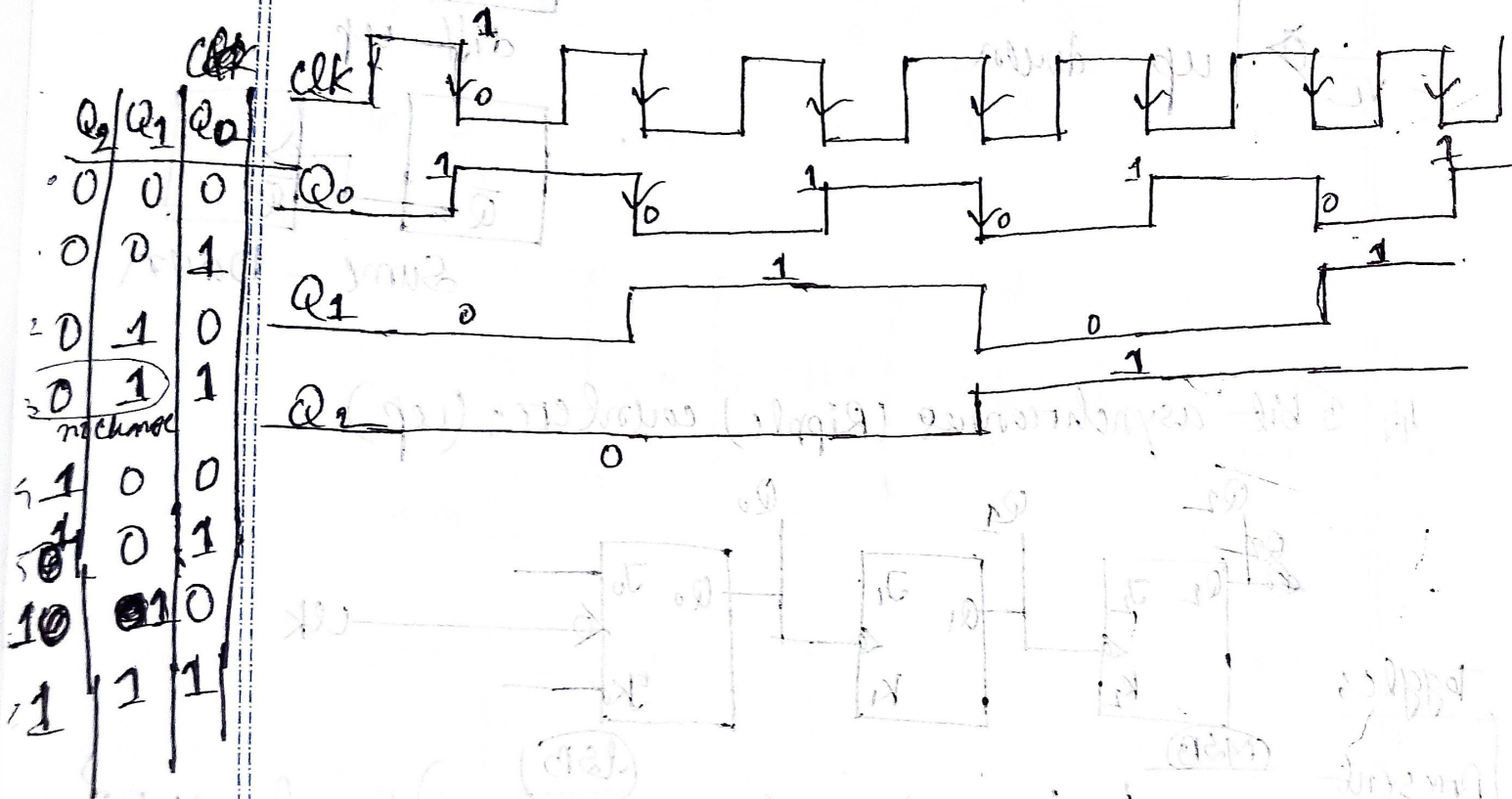
negative edge, each time
 edge - 1
 falling edge & value complement
~~falling edge~~ 2(2) 2(1) 0

number of flip-flop, $n = 3$

number of states = 2^3
 $N = 8$

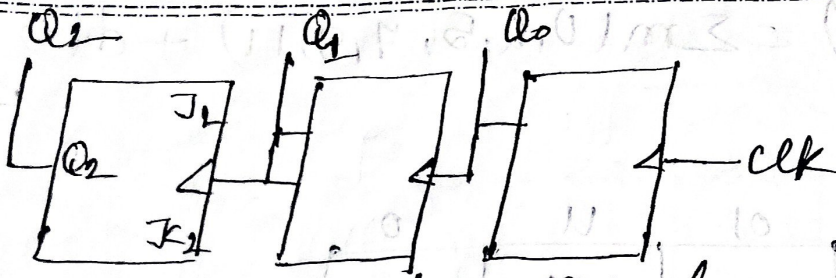
Sat / Sun / Mon / Tue / Wed / Thu / Fri
☐ ☐ ☐ ☐ ☐ ☐ ☐
 Date: / /

range of counting 0 to $(N-1) = 0$ to $(8-1)$
 $= 0$ to 7



Q2	Q1	Q0	Count
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

downs :



2 bit up counter synchronous :-

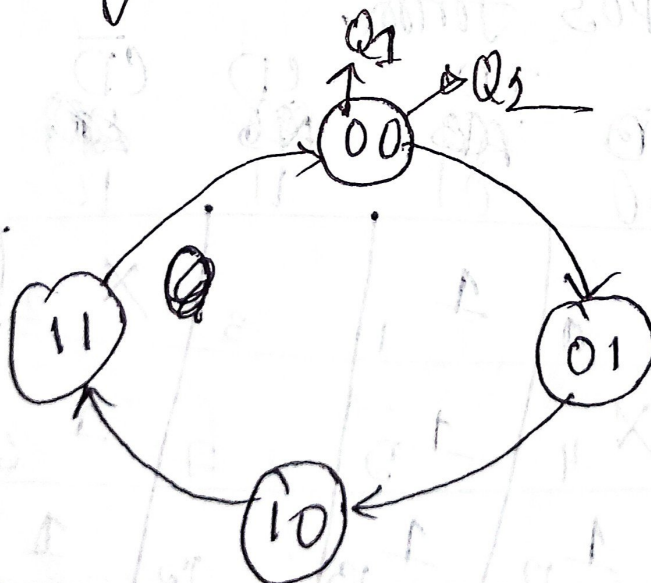
① Decides the number of flip flops :-

②

③ Excitation table of JK FF :-

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

④ State diagram :-



K-Map: -

$$F(P, Q, R, S) = \sum m(0, 2, 5, 7, 9, 11) + d(3, 8, 10, 12, 14)$$

PQ \ RS	00	01	11	10
00	1		X	1
01		1	1	
11	X			X
10	X	1	1	X

W.H. 1 2100
 113510 21
 0 2200
 155 210 21

Fixed 1100
 0 101 110
 0 1110 210
 0 common

min (SOP)

$$\sum m(1, 5, 6, 12, 13, 14) + d(2, 4)$$

in SOP and POS form.

$$F = BD + \overline{A}CD$$

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

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

$$F = BC + \overline{A}CD + BD$$

MAX Term (POS)

Sat / Sun / Mon / Tue / Wed / Thu / Fri
☐ ☐ ☐ ☐ ☐ ☐ ☐
 Date :/...../.....

POS 4

$\Pi M(0, 3, 7, 8, 9, 10, 11, 15) + d(2, 4)$

		$\overline{C}\overline{D}$ 00	$\overline{C}D$ 01	$C\overline{D}$ 11	CD 10
$\overline{A}B$ 00	0	1	2	3	4
$\overline{A}B$ 01	4	5	7	6	
$\overline{A}B$ 11	12	13	15	14	
$\overline{A}B$ 10	8	9	11	10	

Ignore 4
 If don't find any pair

$$F = (\overline{A} + B)(\overline{C} + D)(B + D)$$

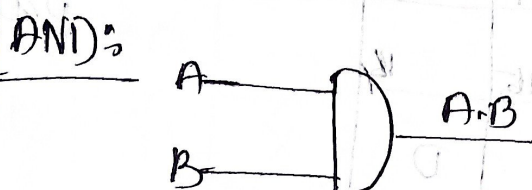
$$F = CD + A\overline{B} + B\overline{D}$$

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



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

get output 1
when ~~all~~
only one
input is 1



A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

get output
1 only
when all
inputs are 1

BOOLEAN THEOREMS

1. $X \cdot 0 = 0$

2. $X \cdot 1 = X$

3. $X \cdot X = X$

4. $X \cdot \bar{X} = 0$

5. $X + 0 = X$

6. $X + 1 = 1$

7. $X + X = X$

8. $X + \bar{X} = 1$

9. $X + Y = Y + X$

10. $X \cdot Y = Y \cdot X$

11. $X + (Y + Z) = (X + Y) + Z = X + Y + Z$

12. $X(YZ) = (XY)Z = XYZ$

13. $X(Y + Z) = XY + XZ$

$(W + X)(Y + Z) = WY + XY + WZ + XZ$

14. $X + \bar{X}Y = X + Y$

15. $X + \bar{X}Y = \bar{X} + Y$

16. $\bar{X} + XY = \bar{X} + Y$

DeMorgan's

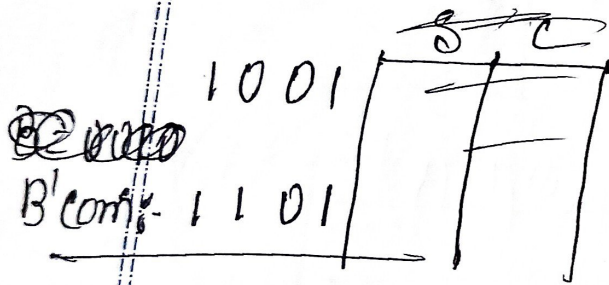
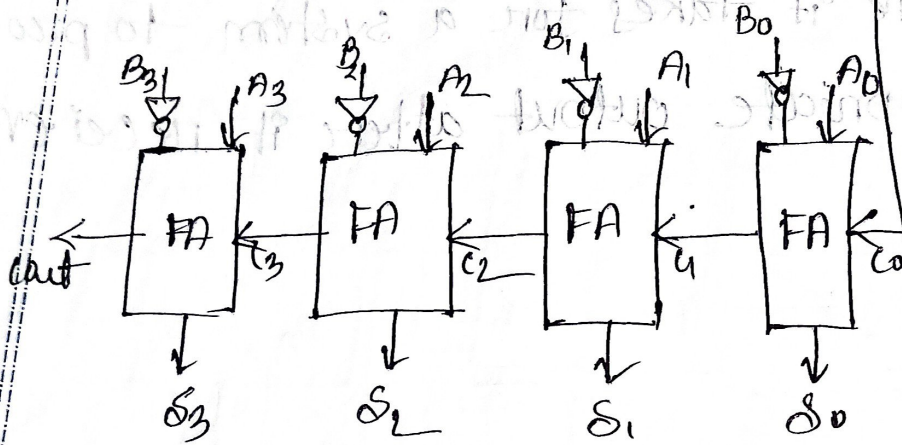
16. $\overline{X + Y} = \bar{X} \cdot \bar{Y}$

17. $\overline{X \cdot Y} = \bar{X} + \bar{Y}$

Chapter: 05

Sat / Sun / Mon / Tue / Wed / Thu / Fri
 Date :/...../.....

4) parallel subtractor:



		cin	s	carb
1	1	0	1	1
0	0	1	1	0
0	1	0	1	0
1	1	0	0	1

Binary subtraction

A	B	BD	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

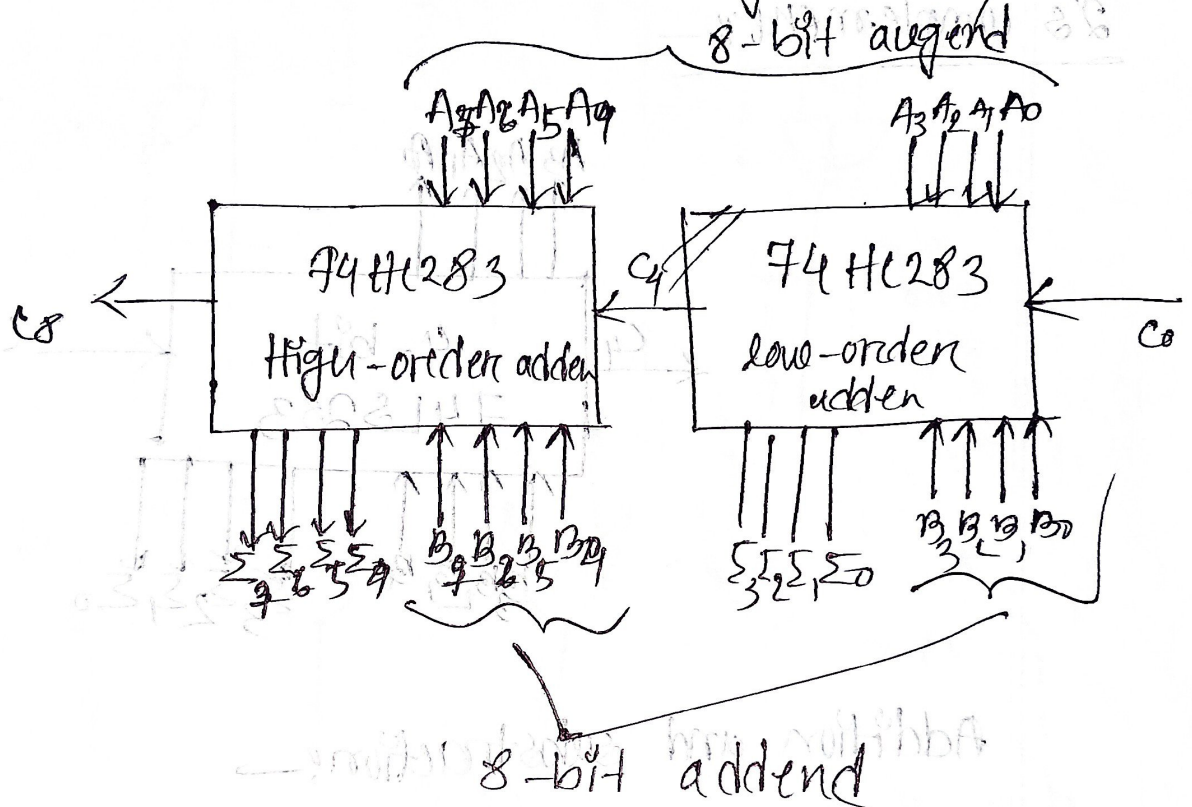
$$A = 1001$$

$$B = 0010$$

complement
 and - ACB
 answer 1011

1001
 0010
 1011

1. Integrated circuit parallel adders: (cascading parallel)
 → Two more more IC can be connected together to accomplish the addition of larger binary numbers.



Here,

- 74HC283 adders connected to eight-bit numbers $A_7A_6A_5A_4A_3A_2A_1A_0$ & $B_7B_6B_5B_4B_3B_2B_1B_0$
- adder on the right adds lower order bits of the numbers, & left adds higher-order bits pulls the C_4 carry out of the lower order.

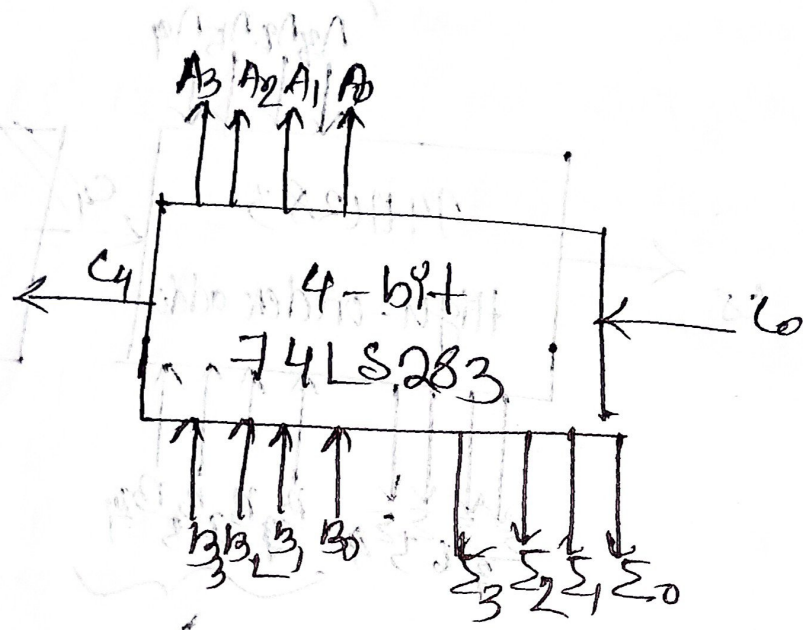
- c_0 is the carry out of ^{the} MSB position

Sat / Sun / Mon / Tue / Wed / Thu / Fri
□ □ □ □ □ □ □

→ the sum bits are $\Sigma_7 \Sigma_6 \Sigma_5 \Sigma_4 \Sigma_3 \Sigma_2 \Sigma_1 \Sigma_0$

→ the look-ahead carry feature of the 74HC283 speeds up the operation.

2's complement



Addition and subtraction

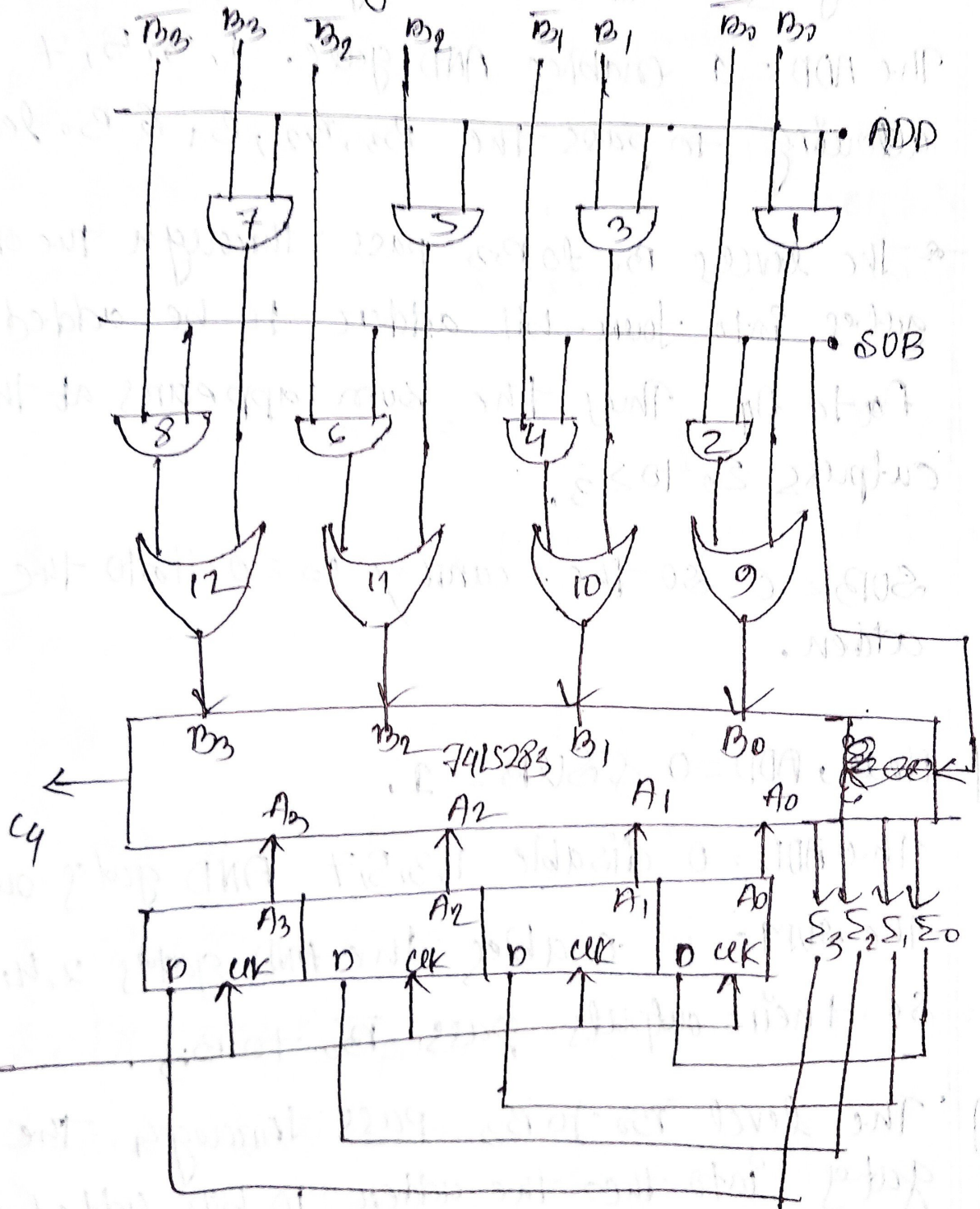
→ This adder ~~adder~~ / subtractor circuit is controlled by the two control signals ADD and SUB.

→ when the ADD level is HIGH, the circuit performs addition of the numbers stored in the A & B registers.

Sat / Sun / Mon / Tue / Wed / Thu / Fri
☐ ☐ ☐ ☐ ☐ ☐ ☐

Date :/...../.....

- when the SUB level is HIGH the circuit subtracts the B from A register numbers.



1/ Let, $ADD = 1$ & $SUB = 0$ disable

AND gates 2, 4, 6, 8 holding their output 0.

The $ADD = 1$ enables AND gates 1, 3, 5, 7 allowing to pass the B_3, B_2, B_1 & B_0 level.

2/ The levels B_0 to B_3 pass through the OR gates into four-bit adder to be added A_4 to A_4 . Thus the sum appears at the outputs Σ_0 to Σ_3 .

3/ $SUB = 0$ so the carry $C_0 = 0$ into the adder.

4/ Now, $ADD = 0$ & $SUB = 1$.

The $ADD = 0$ disable 1, 3, 5, 7 AND gates outputs. The $SUB = 1$ enables the AND gates 2, 4, 6, 8 so their outputs pass $\bar{B}_0$ to $\bar{B}_3$.

5/ The level $\bar{B}_0$ to $\bar{B}_3$ pass through the OR gates into the adder to bits added A_0 to A_3 . Now, the carry C_0 is 1. Thus, the B-register number convert into 2's complement

6] The difference appears at the outputs Σ_0 to Σ_3 .

Sat / Sun / Mon / Tue / Wed / Thu / Fri
☐ ☐ ☐ ☐ ☐ ☐ ☐

Date :/...../.....

7] A Parallel adder is a digital circuit that has adds multiple binary numbers simultaneously. While parallel adders are commonly used in digital systems, they do have certain limitations. Here are some limitations given below:

1. Propagation delay due to ripple carry.
2. Increased circuit complexity for higher bit widths.
3. High power consumption.
4. Significant hardware cost.
5. Slower operation for large-bit adders.
6. Difficult in parallel processing.
7. Scalability issues.
8. Thermal management.
9. Signal integrity.

counters

Sat / Sun / Mon / Tue / Wed / Thu / Fri

☐ ☐ ☐ ☐ ☐ ☐ ☐

Date : / /

1. 3-bit ripple counter:

Here, given number of bits, $n = 3$

number of FF, $n = 3$

number of states $2^n = 2^3 = 8$

Range of counting 0 to $(n-1) = 0$ to $(8-1) = 0$ to 7

Number of clock pulse count by the counter, is 8;

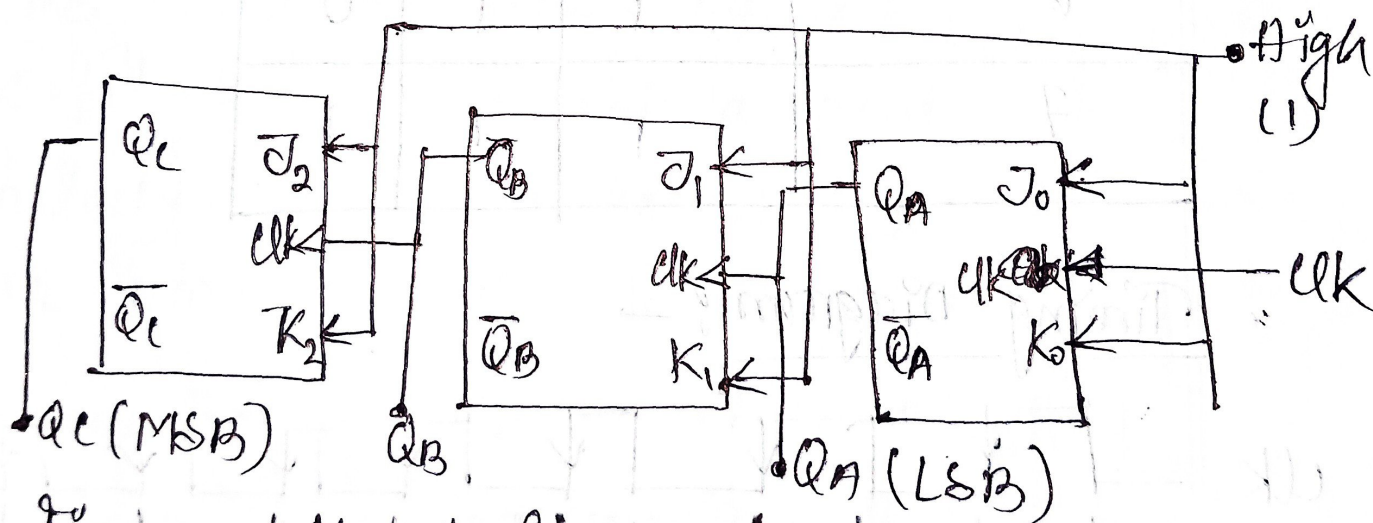
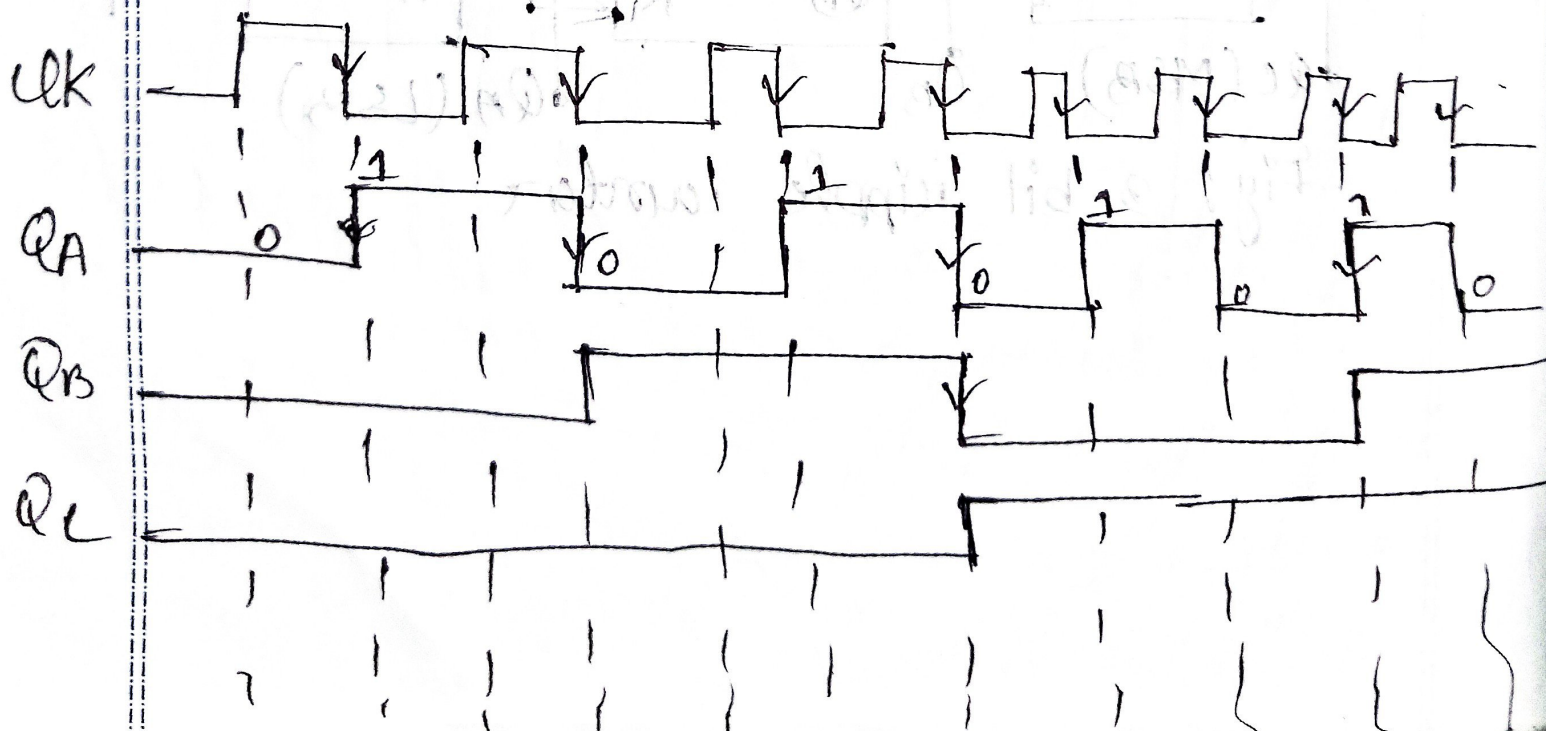


Fig: 3-bit ripple counter

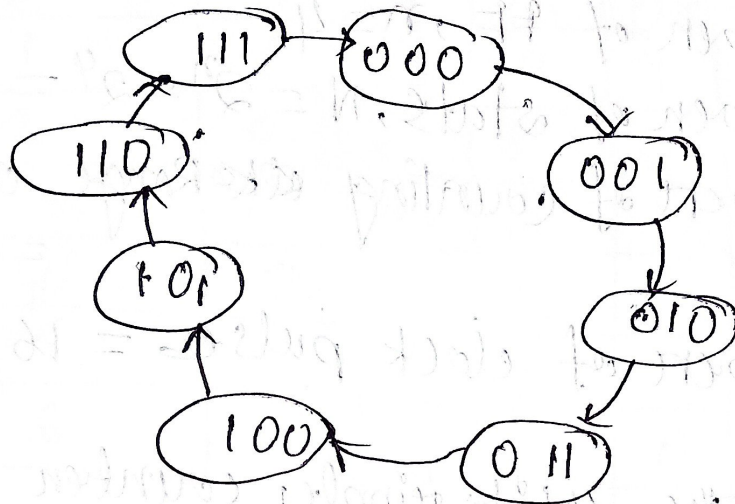
Truth table :-

count state	Q_A	Q_B	Q_C
0	0	0	0
1	1	0	0
2	0	1	0
3	1	1	0
4	0	0	1
5	1	0	1
6	0	1	1
7	1	1	1

Timing Diagram :-



State Diagram:-



working principle:- The 3 bit ripple counter is used in the circuit has eight different states each one of which represent a count value. Similarly, a counter having n flip-flop can have a minimum of 2 to the power n states. The counter works by the counting in binary from 000 to 111; incrementing by 1 for each clk pulse. When the last significant bit (LSB) changes from 1 to 0, it triggers. The next ripple effect to changes its state, creating a ripple effect through the counter.

Q 4 bits -

there, number of bits, = 4

number of FF, $n = 4$

number of state, $N = 2^n = 2^4 = 16$

number of counting ~~etc~~ Range 0 to $(N-1)$
 = 0 to 15

number of clock pulse = 16

Now, draw the 4 bit ripple counter diagram is given below,

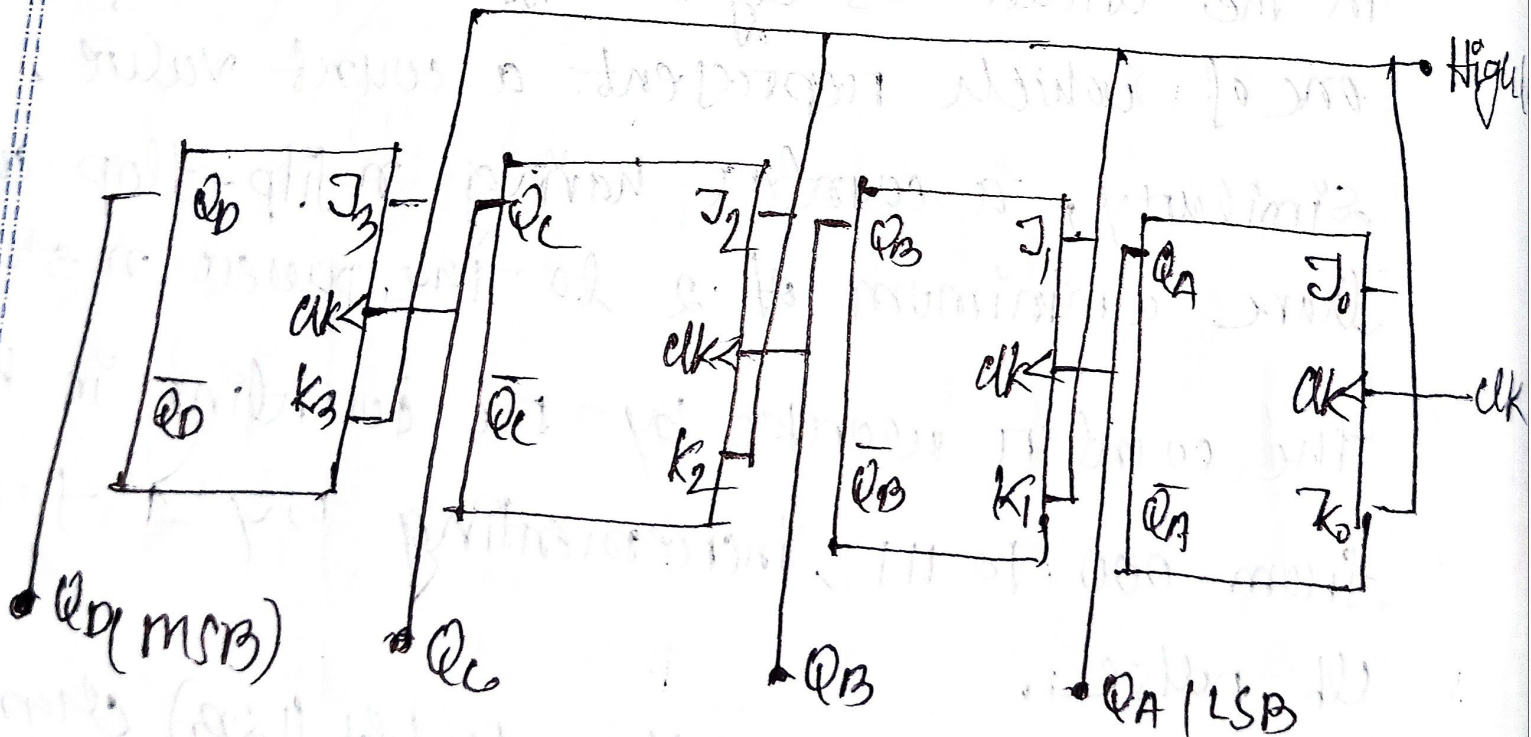


Fig: 4 bit ripple (asynchronous) counter

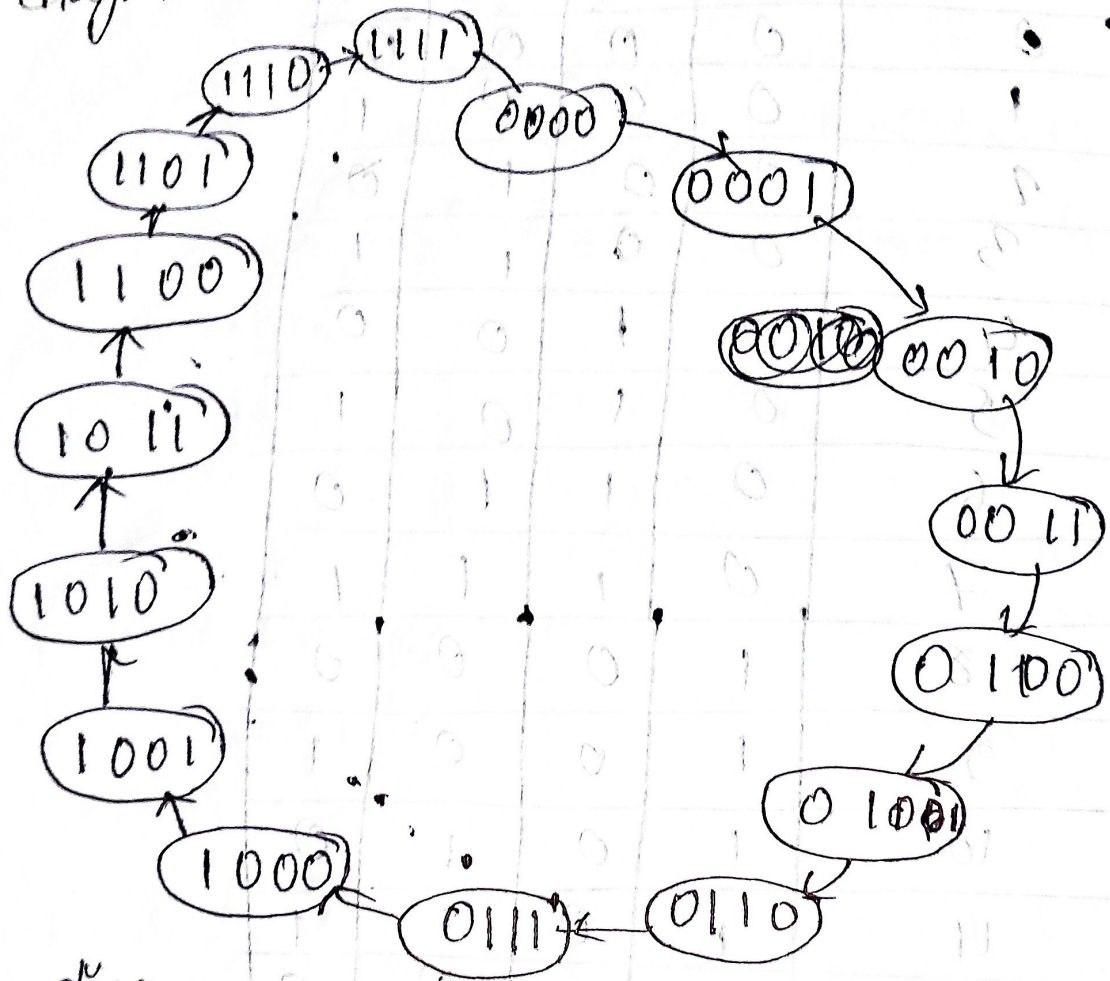
Sat / Sun / Mon / Tue / Wed / Thu / Fri
☐ ☐ ☐ ☐ ☐ ☐ ☐

Date : / /

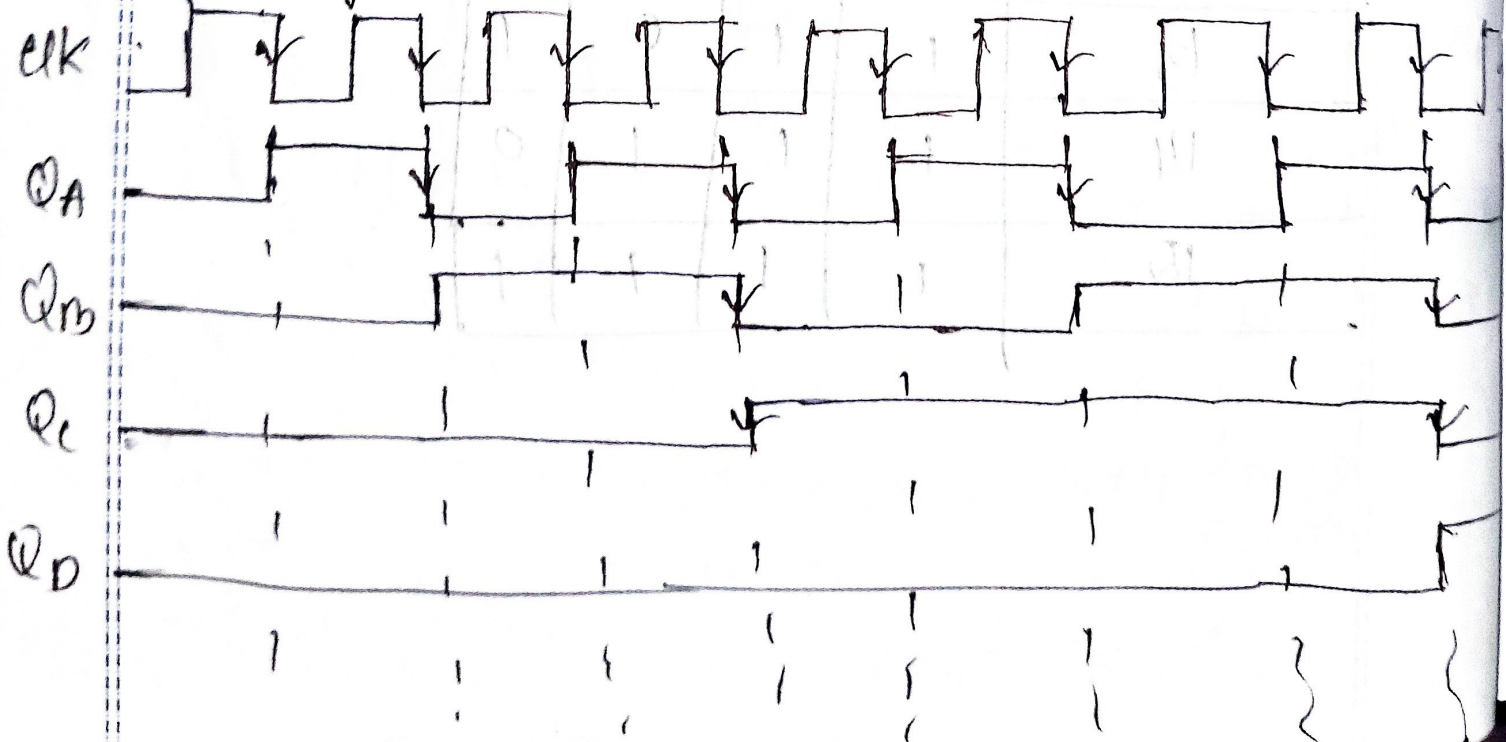
truth table :-

count state	Q ₀	Q ₁	Q ₂	Q ₃
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
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

State diagrams



Timing diagram



~~Synchronous MOD-16 counter~~

MOD-14 (Decade) counter
FF

Sat / Sun / Mon / Tue / Wed / Thu / Fri

☐ ☐ ☐ ☐ ☐ ☐ ☐

Date :/...../.....

resets: $Q_D \cdot Q_C \cdot Q_B \cdot \overline{Q_A}$

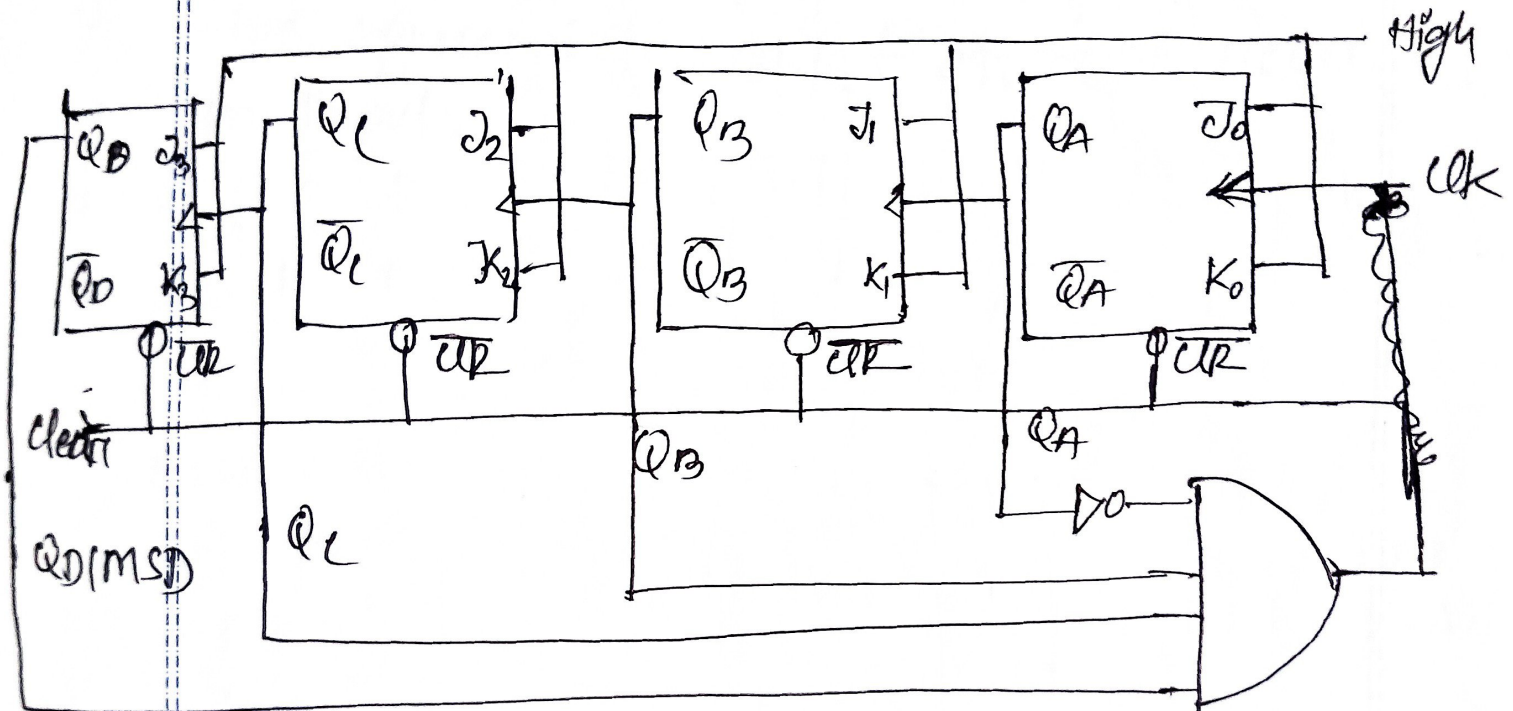


Fig: MOD-14 (Decade) counter.