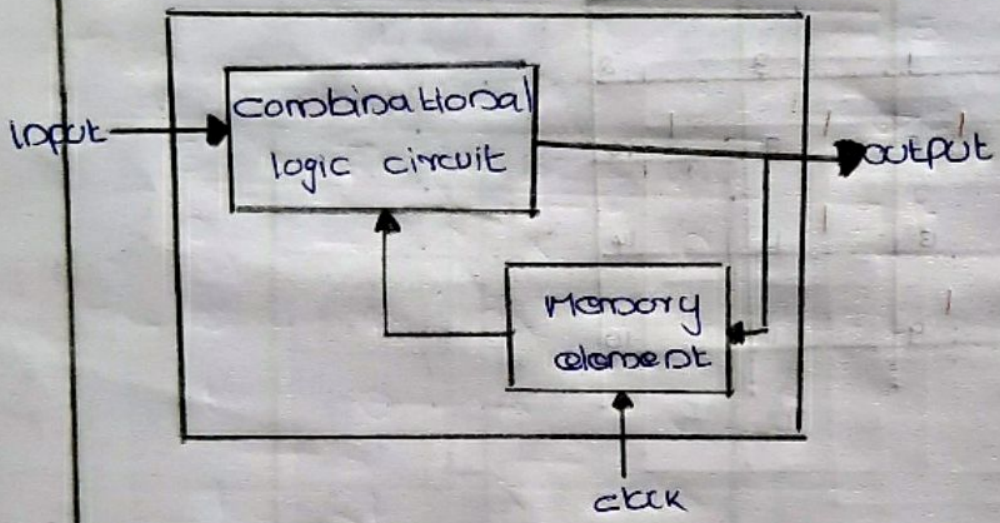
Sequential Circuits

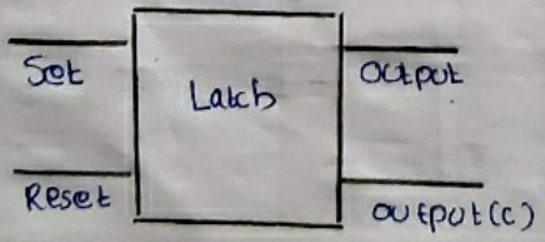
A sequential circuit is a logical circuit, where the output depends on the present value of the input signal as well as the sequence of past inputs. while a combinational circuit is a function of present input only.

* A sequential circuit is a combination of combinational circuit and a storage element.
* The sequential circuits use current input variables and previous input variables which are stored and provides the data to the circuit on the next clock cycle.



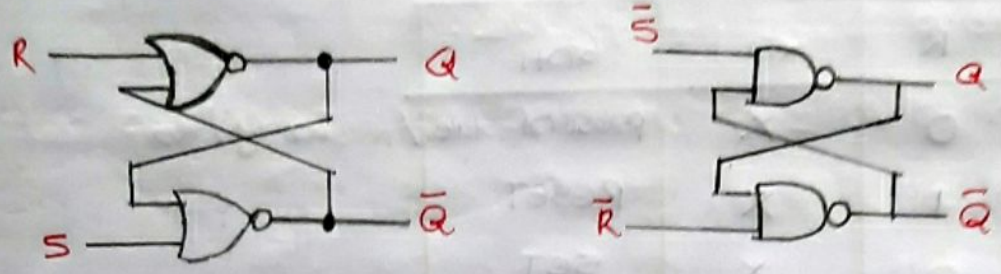
Latch:

Latch is an electronic device, which changes its output immediately based on the applied input. It is used to store either 1 or 0 at any specified time. It consists of two inputs namely "SET" and RESET and two outputs, which are complement to each other.



The S-R (SET-RESET) Latch using NOR gate

A latch is a type of bistable logic device or multivibrator. An active HIGH input S-R latch is formed with two cross-coupled NOR gates. An active-Low input $\bar{S}$-$\bar{R}$ latch is formed with a cross-coupled NAND gates. Notice that the output of each gate is connected to an input of the opposite gate.



(a) active-HIGH input S-R latch (b) active-Low input S-R latch

Truth table (a)

S	R	Q_n	Q_{n+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	1·0
1	1	1	1·0

Case 1: S=0 R=0

$Q_{n+1} = \overline{(R + \bar{Q}_n)} = \overline{(0 + \bar{Q}_n)} = \bar{0} \cdot \bar{\bar{Q}}_n = 1 \cdot \bar{\bar{Q}}_n = Q_n$

$\bar{Q}_{n+1} = \overline{(S + Q_n)} = \overline{(0 + Q_n)} = 1 \cdot \bar{Q}_n = \bar{Q}_n$

Case 2: S=0 R=1

$Q_{n+1} = \overline{(1 + \bar{Q}_n)} = 0 \cdot Q_n = 0$

Case 3. $S=0, R=0$

$Q_{n+1} = \overline{(0+\bar{Q}_n)} = 1 \cdot Q_n = Q_n$

$\bar{Q}_{n+1} = \overline{(1+Q_n)} = 0 \cdot \bar{Q}_n = 0$ //

case 4: $S=1, R=1$

$Q_{n+1} = \overline{(1+\bar{Q}_n)} = 0 \cdot Q_n = 0$ //

$\bar{Q}_{n+1} = \overline{(1+Q_n)} = 0 \cdot \bar{Q}_n = 0$ //

S	R	Q_n	Q_{n+1}
0	0	X	Present state / memory / Q_0
0	1	X	RESET
1	0	X	SET
1	1	X	ID

Truth table (b)

$\bar{S}$	$\bar{R}$	Q_n	Q_{n+1}
0	0	X	ID
0	1	X	1
1	0	X	0
1	1	X	PS

Case 1: $\bar{S}=0, \bar{R}=0$

$Q_{n+1} = \overline{(\bar{S} \cdot \bar{Q}_n)} = \overline{(0 \cdot \bar{Q}_n)} = 1 + \bar{Q}_n = \bar{Q}_n$ 1

$\bar{Q}_{n+1} = \overline{(\bar{R}+Q_n)} = \overline{(0 \cdot Q_n)} = 1 + \bar{Q}_n = 1$

Case 2: $\bar{S}=0, \bar{R}=1$

$Q_{n+1} = \overline{(0 \cdot \bar{Q}_n)} = 1 + Q_n = 1$

case 3: $\bar{S}=1, \bar{R}=0$

$Q_{n+1} = \overline{(1 \cdot \bar{Q}_n)} = 0 + \bar{Q}_n = \bar{Q}_n$

$\bar{Q}_{n+1} = \overline{(Q \cdot Q_n)} = 0 + \bar{Q}_n = \bar{Q}_n$ 1 //

Case 4: $\bar{S}=1, \bar{R}=1$

$Q_{n+1} = (1 \cdot \overline{\bar{Q}_n}) = 0 + Q_n = Q_n$

$\bar{Q}_{n+1} = (1 \cdot \overline{Q_n}) = 0 + \bar{Q}_n = \bar{Q}_n$

Edge Triggered Flip-Flops

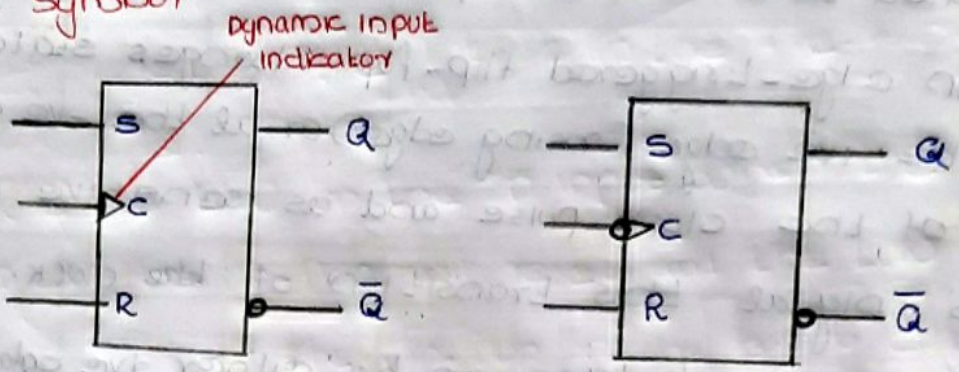
* Flip-flop is a circuit that maintains a state until directed by input to change the state. A basic flip-flop can be constructed using four-NAND or four-NOR gates.
* The output changes state only at a specified point on the triggering input called the clock (CLK), which is designated as a control input, C
* An edge-triggered flip-fop changes state either at the +ve edge (rising edge) or at the -ve edge (falling edge of the clock pulse and is sensitive to its inputs only at this transition of the clock.
* Notice that each type can be either +ve edge-triggered (no bubble at C input) or -ve edge-triggered (bubble at C input).
* The key to identifying an edge-triggered flip-flop by its logic symbol is the small triangle inside the block at the clock(C) input. This triangle is called the dynamic input indicator.

S-R flip-flop

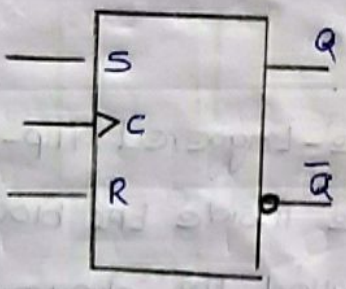
The S & R inputs of the S-R flip-flop are called synchronous inputs bcz data on these inputs are transferred to the flip flop's output only on the triggerring edge of the clock pulse.

* when S is HIGH and R is LOW, the Q output goes HIGH on the triggering edge of the clock pulse, and the flip-flop is SET.
* When S is LOW and R is HIGH, the Q output goes LOW on the triggering edge of the clock pulse, and the flip-flop is RESET.
* when both S and R are LOW, the output doesnot change from its prior state.
* when An invalied condition exists when both S and R are HIGH.

Logic symbol



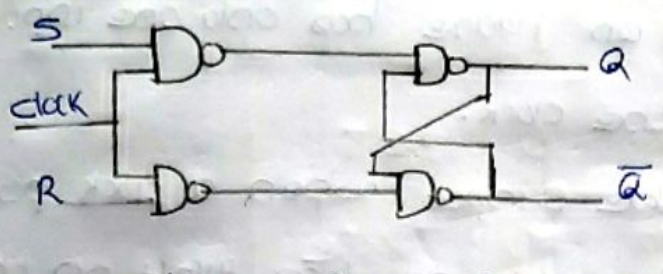
The basic operation of a +ve edge-triggering flip-flop



Inputs			Outputs		
S	R	CLK	Q	$\bar{Q}$	comments
0	0	X	Q_0	$\bar{Q}_0$	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	?	?	Invalied

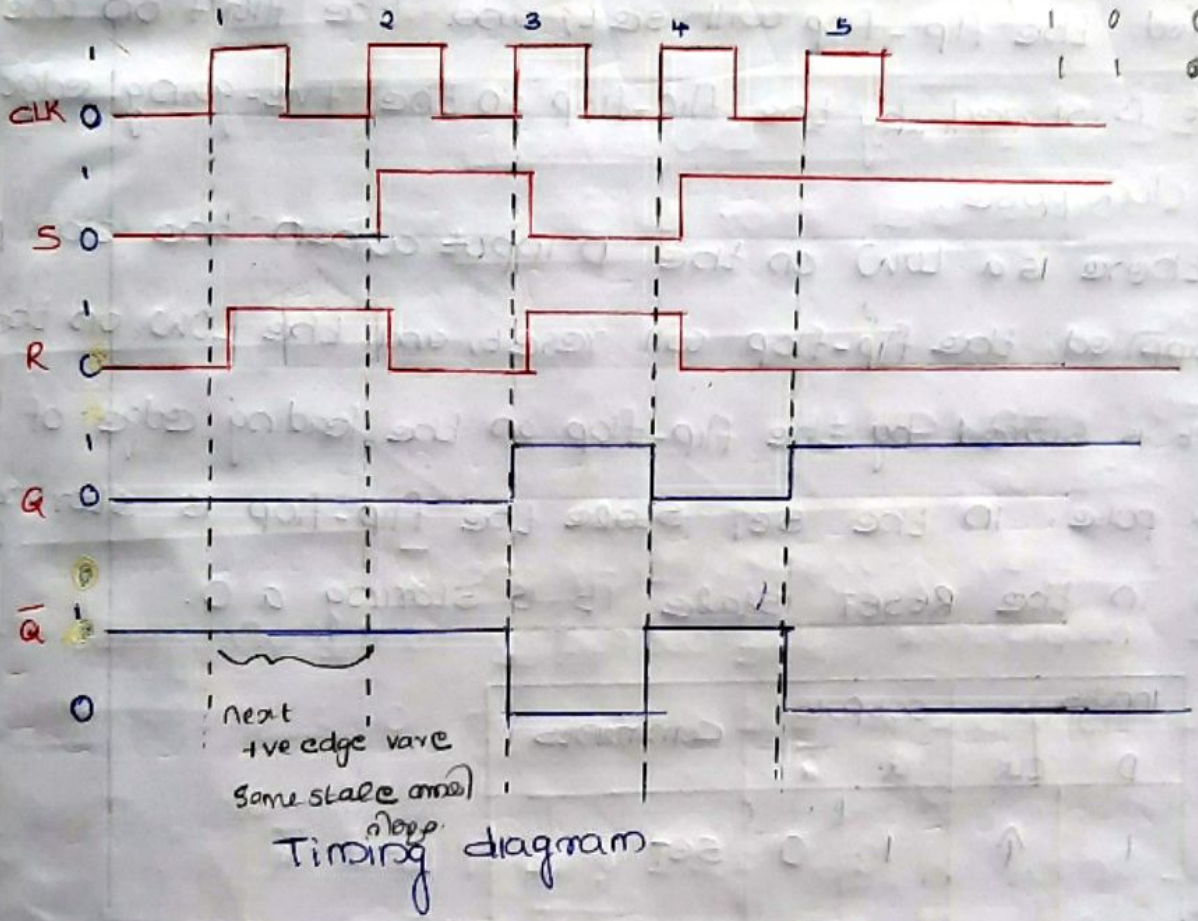
The operation and truth table for a -ve edge-triggered S-R flip-flop are the same as those for a +ve edge-triggered device except that the falling edge of the clock pulse is the triggering edge.

2 Determine the Q & $\bar{Q}$ output waveforms of the flip-flops for the S-R and CLK inputs. Assume that the +ve edge-triggering flip-flop is initially RESET



Active low S-R

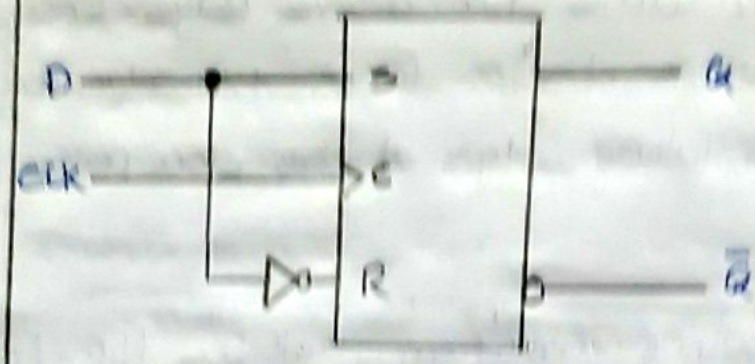
$\bar{S}$	$\bar{R}$	Q	$\bar{Q}$
0	0	?	?
0	1	1	0
1	0	0	1
1	1	Q_0	$\bar{Q}_0$



Timing diagram

The edge-triggered D flip-flop

The D flip-flop is useful when a single data bit (1 or 0) is to be stored. The addition of an inverter to an S-R flip-flop creates a basic D flip-flop.



A +ve edge triggered D flip-flop formed with an S-R flip flop and an inverter.

Notice that the flip-flop on figure has only one input, the D input, in addition to the clock.

If there is a high on the D input when a clock pulse is applied, the flip-flop will set, and the HIGH on the D input is stored by the flip-flop on the +ve-going edge of the clock pulse.

If there is a LOW on the D input when the clock pulse is applied, the flip-flop will reset, and the LOW on the D input is stored by the flip-flop on the leading edge of the clock pulse. In the SET state the flip-flop is storing a 1, and in the RESET state it is storing a 0.

inputs		outputs		comments
D	CLK	Q	$\overline{Q}$	
1	↑	1	0	SET
0	↑	0	1	RESET

using S-R flip-flop implementation using NAND gate,

case 1: D = 1, S = 0, R = 0

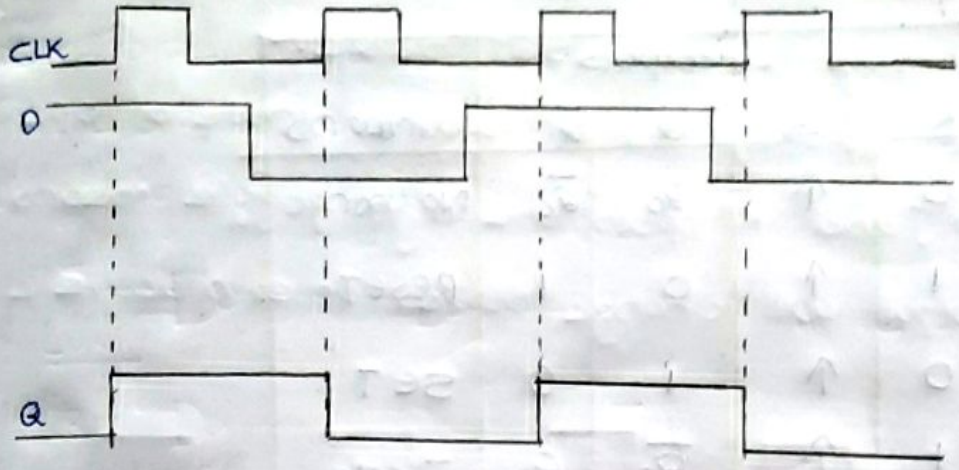
$\overline{S} = 0$, R = 1

$Q_{n+1} = (\overline{0 \cdot \overline{Q_n}}) = 1 + Q_n = 1$ //

Case 2: $D=0$, $\bar{S}=1$, $\bar{R}=0$

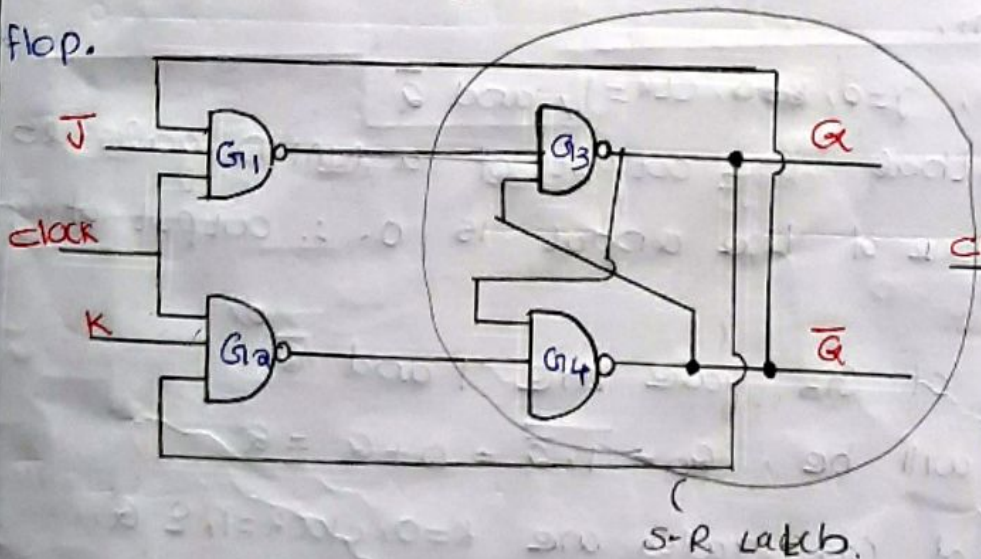
$Q_{n+1} = (1 \cdot \overline{Q_n}) = 0 + \bar{Q}_n = \bar{\bar{Q}}_n = Q_n$ //

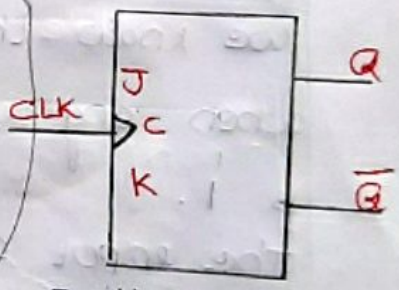
$\overline{Q_{n+1}} = (0 \cdot Q_n) = 1 + \bar{Q}_n = 1$ //



The Edge-Triggered J-K Flip-flop

J-K flip-flop is widely used type of flip-flop. The functioning of the J-K flip-flop is identical to that of the S-R flip-flop in the SET, RESET, and no change conditions of operation. The difference is that the J-K flip-flop has no invalid state as does the S-R flip-flop.





Positive edge triggered J-K Flip-Flop

* It differs from the S-R edge-triggered flip-flop in that the Q output is connected back to the input of gate G_2, and the $\bar{Q}$ output is connected back to the input of gate G_1. The two control inputs are labeled J and K in honor of Jack Kilby, who invented the Integrated

circuit.

A J-K flip-flop can also be of the -ve-edge-triggering type, in which case the clock input is inverted.

Truth table.

Inputs			outputs		
J	K	CLK	Q	$\bar{Q}$	comments
0	0	↑	Q_0	$\bar{Q}_0$	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	$\bar{Q}_0$	Q_0	Toggle

↑ - clock transition LOW to HIGH

Q_0 - output level prior to clock transition

Case 1

J=0, K=0, CLK=1,

In gate G_1,

Inputs are, J=0, ~~K=0~~, CLK=1, and $\bar{Q}$

We know that the output of a NAND gate is 1 when any one of the input is 0. ∴ output of G_1 gate is 1.

The input of G_3 gate are 1 and $\bar{Q}$

∴ output will be, $Q_0 = \overline{1 \cdot \bar{Q}} = 0 + Q = Q$

The input of G_2 gates are K=0, clock=1, & Q

The output of G_2 gate is 1.

The input of G_4 gate are 1 & Q, The output of G_4 gate is,

$\bar{Q}_0 = (\overline{1 \cdot Q}) = 0 + \bar{Q} = \bar{Q}$

∴ If you apply a LOW to both the J and K inputs, the

flipflop will stay in its present state when a clock pulse occurs.

case 2. J = 0, K = 1

Inputs of G_1 gate : J = 0, CLK = 1 and $\bar{Q}$.

output = 1

Inputs of G_2 gate : K = 1, CLK = 1 & $\bar{Q}$

output = $\overline{(1 \cdot 1 \cdot Q)} = 0 + 0 + \bar{Q} = \bar{Q}$

G_3 and G_4 gate constitute a S-R latch using NAND gate. The output of SR latch is,

$Q = \overline{1 \cdot \bar{Q}} = 0 + Q_0 = Q_0$.

$\bar{Q} = \overline{Q_0 * \bar{Q}_0} = \bar{Q}_0 + Q_0 = 1$

This will cause the latch portion of the flip-flop to change to the RESET state.

case 3. J = 1, K = 0

Inputs of G_1 gate : J = 1, CLK = 1 & $\bar{Q}$

output = $\overline{(1 \cdot 1 \cdot \bar{Q}_0)} = 0 + 0 + Q_0 = Q_0$

Inputs of G_2 gate : K = 0, CLK = 1 & Q

output = 1

The output of S-R latch,

$Q = \overline{(Q_0 * \bar{Q}_0)} = \bar{Q}_0 + Q_0 = 1$

This will cause the latch portion of the flip-flop to change to the SET state.

case 4: J = 1, K = 1

Inputs of G_1 gate : J = 1, CLK = 1 & $\bar{Q}$

output = Q_0

Inputs of G_2 gate : K = 1, CLK = 1 & Q

output = $\overline{(1 \cdot 1 \cdot Q_0)} = 0 + 0 + \bar{Q}_0 = \bar{Q}_0$

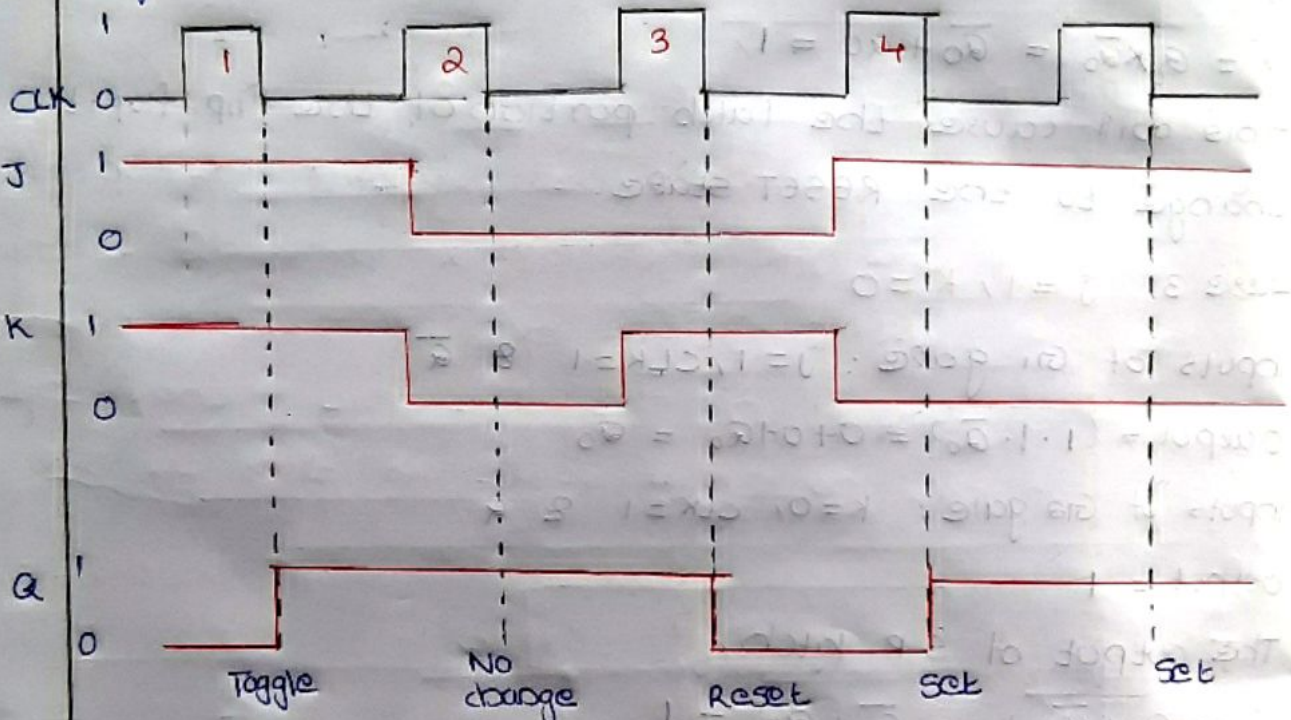
The output of the latch portion of the flip-flop, ~~$Q = \overline{(Q_0 * \bar{Q}_0)} = \bar{Q}_0 + Q_0 = 1$~~ depends upon the value

of Q_0 & $\bar{Q}_0$.

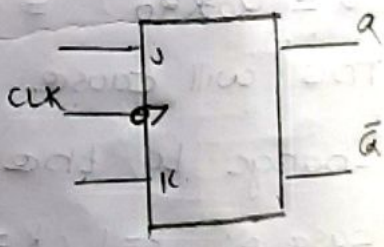
If $Q_0 = 0$ then $\bar{Q}_0 = 1$, then the clatch portion of the flip-flop change to the RESET stake.

If $Q_0 = 1$ then $\bar{Q}_0 = 0$, then the latch portion of the flip-flop change to the SET stake.

∴ The J-K flip-flop has a toggle mode of operation when both J and K inputs are high. Toggle means that the Q output will change states on each active clock edge.



Timing diagram

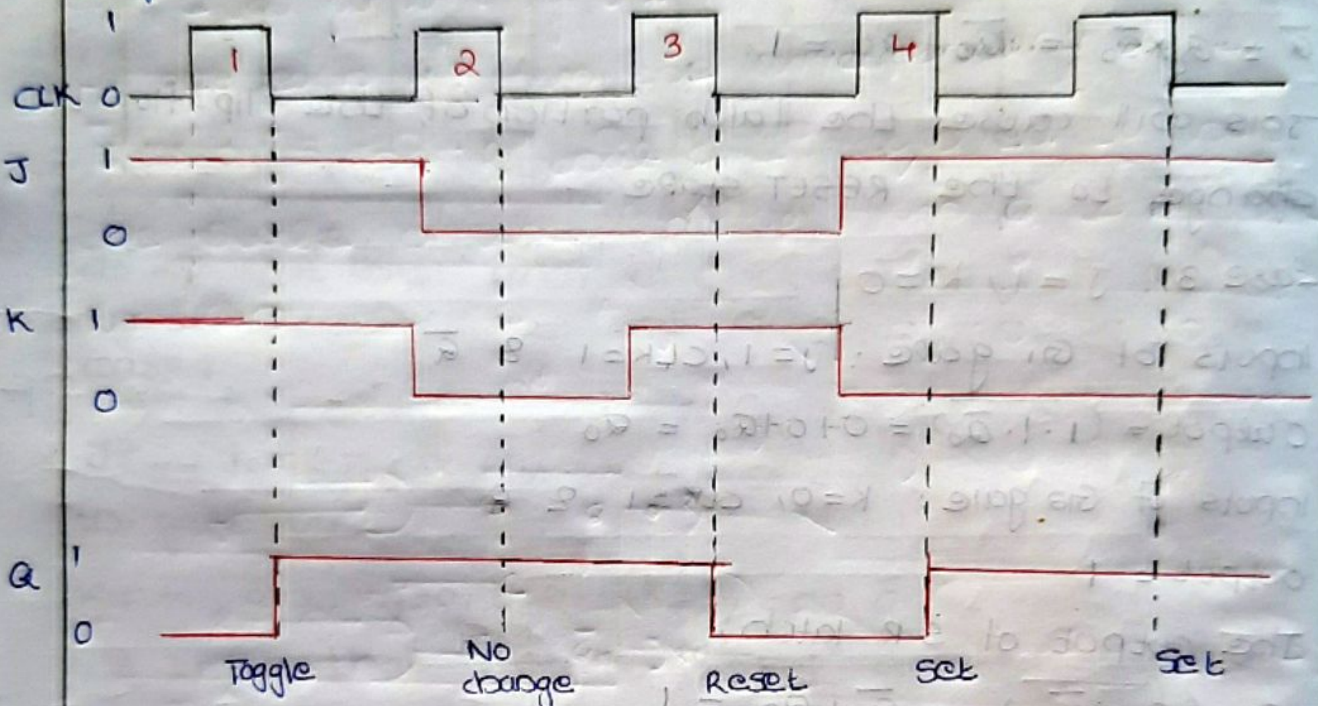


of Q_0 & $\bar{Q}_0$.

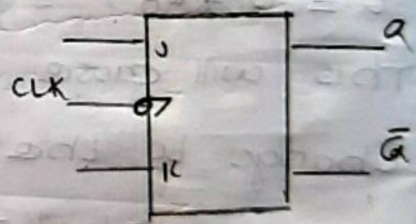
If $Q_0 = 0$ then $\bar{Q}_0 = 1$, then the latch portion of the flip-flop change to the RESET state.

If $Q_0 = 1$ then $\bar{Q}_0 = 0$, then the latch portion of the flip-flop change to the SET state.

∴ The J-K flip-flop has a toggle mode of operation when both J and K inputs are high. Toggle means that the Q output will change states on each active clock edge.



Timing diagram

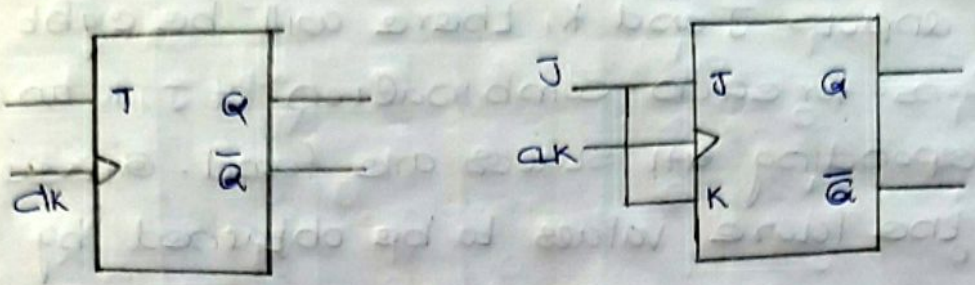


T flip-flop

T → means Toggle

The toggle flip-flop can be used as a basic digital element for storing one bit of information, as and Toggle flip-flops have a single input and one or two complementary outputs of Q and $\bar{Q}$ which change state on the positive edge (rising edge) or -ve edge (falling edge) of an input clock signal or pulse.

T flip-flop can be easily constructed by connecting together the J and K inputs of a basic JK flip-flop where the J input behaves like a set (S) command, and the K input behaves like a Reset (R) command



Truth table

T	clk	Q	Q+1
0	↑	0	0
0	↑	1	1
1	↑	0	1
1	↑	1	0

Flip-flop conversion

steps:

1. Consider truth table for destination flip-flop and extend it as Excitation table of source flip-flop
2. Draw K-map for source flip-flop variables, it will provide expression for conversion.
3. Draw the circuit according to K-map expressions.

Conversions

SR → D JK → T T → D

SR → JK JK → D D → T

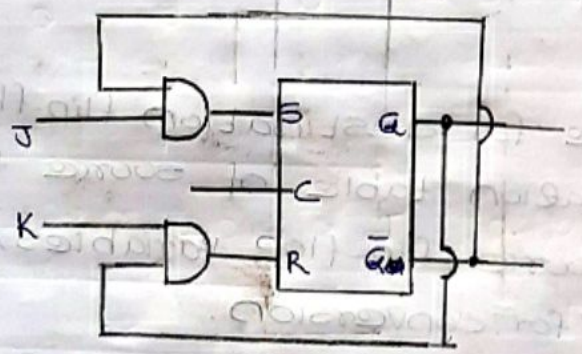
SR → T

SR flip-flop to JK flip-flop

* J and K will be given as external inputs to S and R. As shown in the logic diagram below, S and R will be the outputs of the combinational circuit.

* The truth tables for the flip-flop conversion are given below. The present state is represented by Q and Q_{n+1} as the next state to be obtained when the J and K inputs are applied.

* For two inputs J and K, there will be eight possible combinations. for each combination of J, K and Q, the corresponding Q_{n+1} states are found. Q_{n+1} simply suggests the future values to be obtained by the JK flipflop after the value of Q. The table is then completed by writing the values of S and R required to get each Q_{n+1} from the corresponding Q. ie, the values of S and R that are required to change the state of the flip flop from Q to Q_{n+1} are written.



Truth table of JK

J	K	Q	Q_{n+1}
0	0	x	Q
0	1	x	0
1	0	x	1
1	1	x	$\bar{Q}$

Excitation Table of SR flipflop

Q	Q_{n+1}	S	R
0	0	0	x
0	1	1	0
1	0	0	1
1	1	x	0

The excitation table of any flipflop is drawn using its truth table.

Truth table of SR

S	R	Q	Q+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	ID
1	1	1	ID

$Q=0, Q_{+1}=0$

$S=0, R=0$

$S=0, R=1$

ie, $S=0, R=X$

$Q=0, Q_{+1}=1$ $\quad$ $Q=1, Q_{+1}=0$

$S=1, R=0$ $\quad$ $S=0, R=1$

$Q=1, Q_{+1}=1$

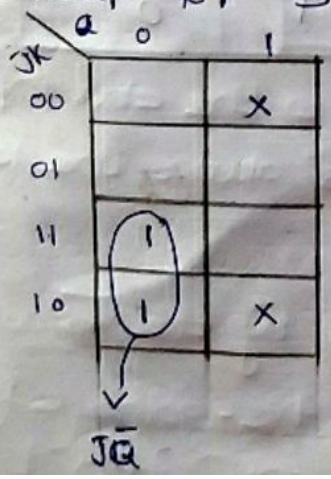
$S=0, R=0$

$S=1, R=0$

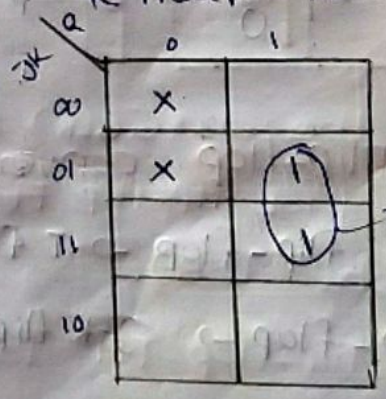
Extended truth table of JK flip flop

J	K	Q	Q+1	S	R
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	X	0
1	1	0	1	1	0
1	1	1	0	0	1

K map for S



K map for R



Excitation table of Jk flip flop

Truth table of Jk flip-flop

J	K	Q	Q+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

Excitation Table.

Q	Q+1	J	K.
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Excitation table of D flip flop.

D	Q	Q+1
0	0	0
0	1	0
1	0	1
1	1	1

Excitation table.

Q	Q+1	D
0	0	0
0	1	1
1	0	0
1	1	1

Excitation table of T flip-flop

T	Q	Q+1
0	0	0
0	1	1
1	0	1
1	1	0

Excitation table

Q	Q+1	T
0	0	0
0	1	1
1	0	1
1	1	0

JK to T Flip-flop conversion.

Destination flip-flop → T flip flop.

Source flip-flop → J-K flip-flop.

T	Q	Q+1	
0	0	0	
0	1	1	
1	0	1	
1	1	0	

Q	Q+1	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

extended truth table of T flipflop

T	Q	Q+1	J	K
0	0	0	0	X
0	1	1	X	0
1	0	1	1	X
1	1	0	X	1

K map for J

	Q'	Q
T'		X
T	1	X

$J = T$

K map for K

	Q'	Q
T'	X	
T	X	1

$K = T$

circuit diagram.

Applications of flip-flops

* Counters
* Frequency Dividers
* Shift Registers.
* Storage Registers.
* Boonce elimination switch
* Data storage
* Data transfer

* Latch
* Registers
* Memory

Counters.

Counter is a sequential circuit. A digital circuit which is used for a counting pulses is known counter. counter is the widest application of flip-flops. It is a group of flip-flops with a clock signal applied.

counters are of two types.

- Asynchronous or ripple counters.
- synchronous counters.

A 2-Bit Asynchronous Binary counter

Asynchronous counter:

Asynchronous counters are those whose output is free from the clock signal. Because the flipflops in asynchronous counters are supplied with different clock signals, there may be delay in producing output.

Notice that the clock(CLK) is applied to the clock input (C) of only the first flip-flop, FF0, which is always the least significant bit (LSB).

The second flip-flop, FF1, is triggered by the $\overline{Q_0}$ output of FF0, FF0 changes state at the +ve-going edge of each clock pulse, but FF1 changes only when triggered

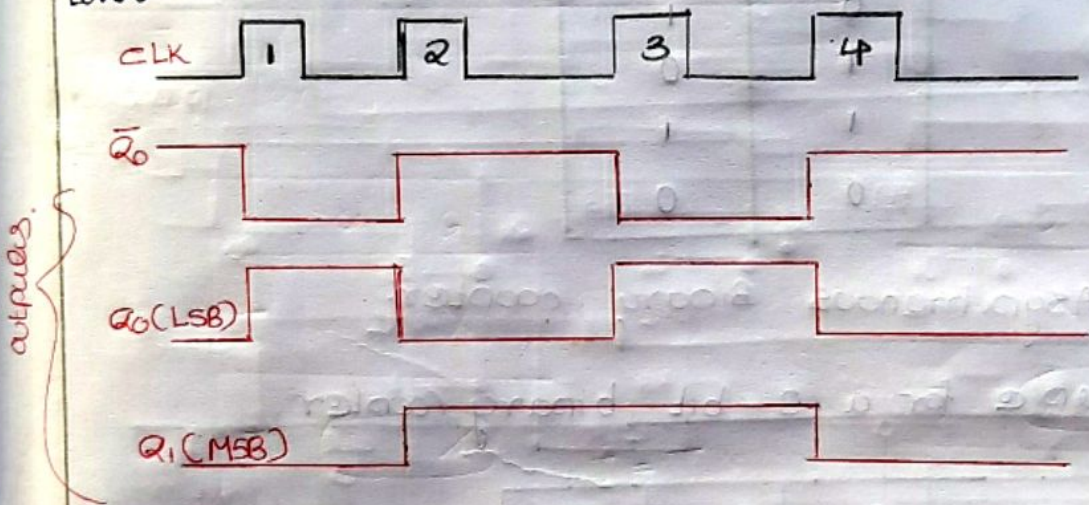
* when we use -ve edge triggered, the clock of FF1 is Q_0 (not $\overline{Q_0}$)

by a +ve-going transition of the $\bar{Q}_0$ output of FF0.

* Because of the inherent propagation delay time through a flip-flop, a transition of the input clock pulse (CLK) and a transition of the $\bar{Q}_0$ output of FF0 can never occur at exactly the same time. ∴ the two flip-flops are never simultaneously triggered, so the counter operation is asynchronous.

The Timing Diagram:

The following figure illustrates the changes in the state of the flip-flop outputs in response to the clock pulses. Both flip-flops are connected for toggle operation (J=1, K=1) and are assumed to be initially RESET (Q LOW).



* The +ve-going edge of CLK1 (clock pulse 1) causes the Q_0 output of FF0 to go HIGH. At the same time $\bar{Q}_0$ output goes low, but it has no effect on FF1 because a +ve-going transition must occur to trigger the flip-flop. After the leading edge of CLK1, $Q_0=1$ and $Q_1=0$.

* The +ve-going edge of CLK2 causes Q_0 to go LOW. output $\bar{Q}_0$ goes HIGH and triggers FF1, causing Q_1 to go HIGH. After the leading edge of CLK2, $Q_0=0$ and $Q_1=1$.

The +ve-going edge of CLK3 causes Q_0 to go HIGH again output Q_0 goes low and has no effect on FF1. Thus after the leading edge of CLK3, $Q_0 = 1$ and $Q_1 = 1$.

The +ve-going edge of CLK4 causes Q_0 to go LOW, while $\bar{Q}_0$ goes HIGH and triggers FF1, causing Q_1 to go LOW.

After the leading edge of CLK4, $Q_0 = 0$ and $Q_1 = 0$.

The counter has now recycled to its original state (both flip-flops are RESET)

Binary state sequence for the counter

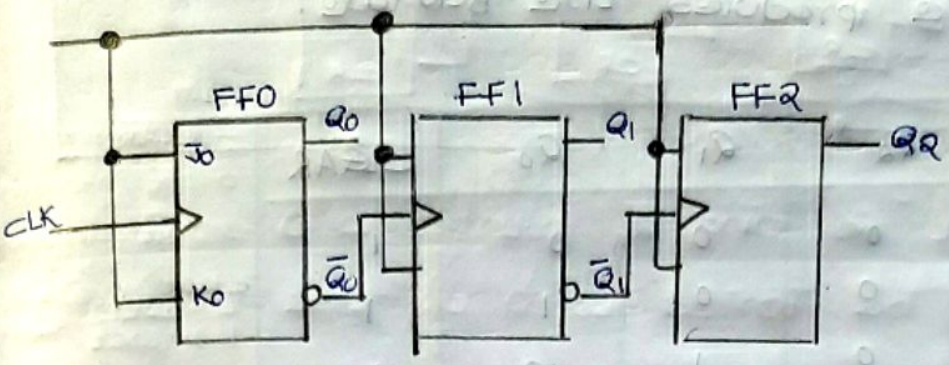
clock pulse	Q_1	Q_0
initially	0	0
1	0	1
2	1	0
3	1	1
4	0	0

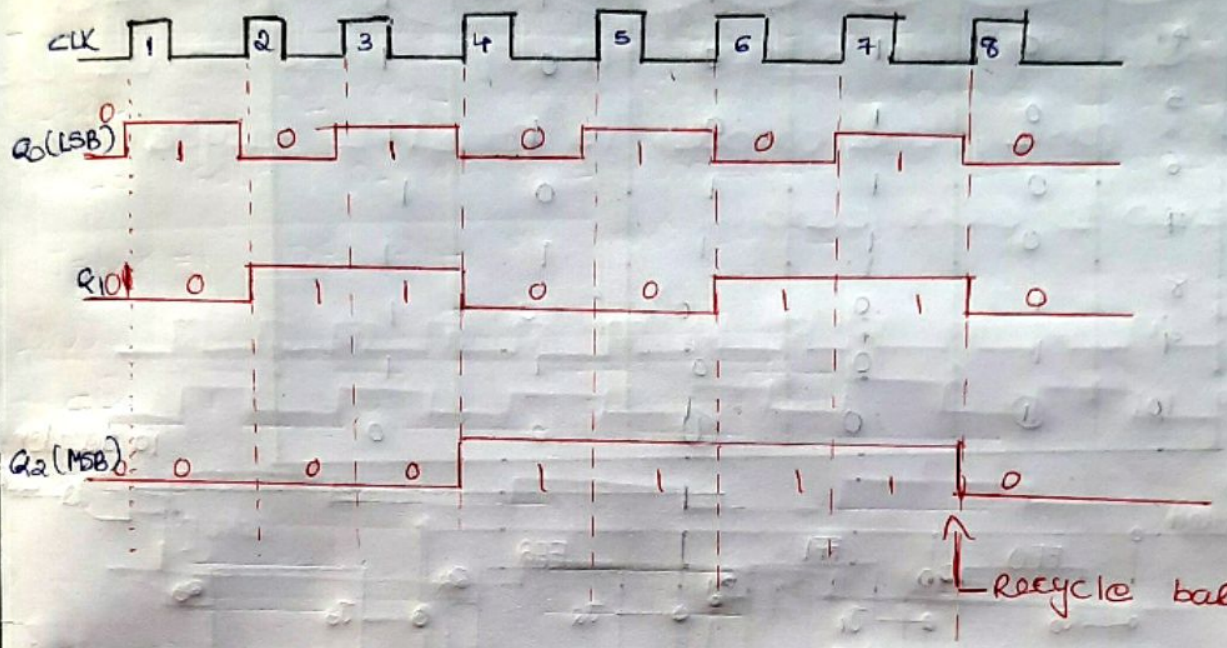
A 3-Bit Asynchronous Binary counter.

state sequence for a 3-bit binary counter

clock pulse	Q_2	Q_1	Q_0
Initially	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8 (recycles)	0	0	0

Three-bit asynchronous binary counter and its timing diagram for one cycle.





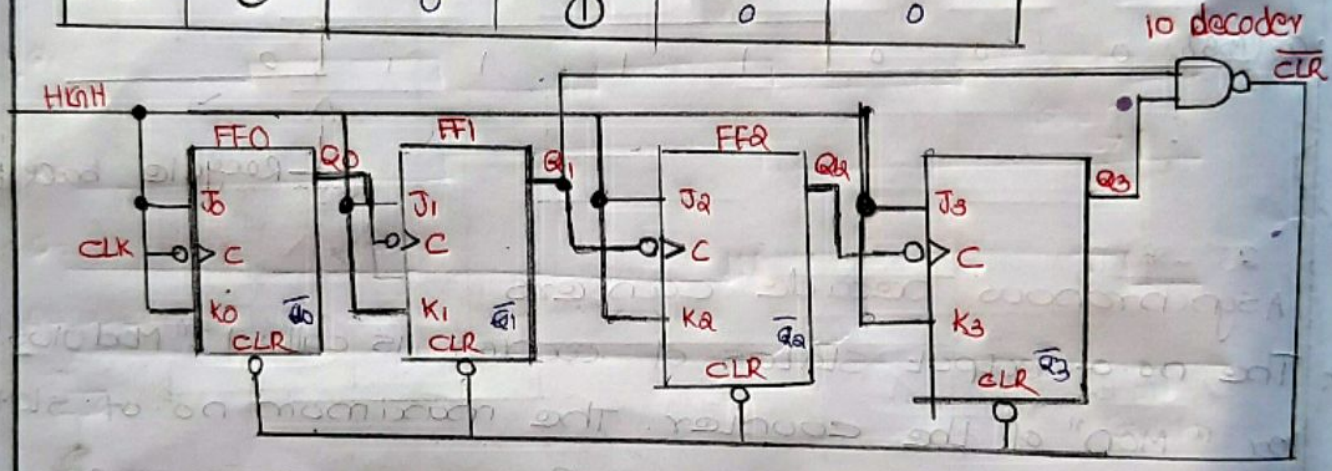
Asynchronous decade counters

* The no. of output states of counter is called "Modulus" or "MOD" of the counter. The maximum no. of states that a counter can have is 2^n, where n represents the no. of flip-flops used in counter.
* Counters can be designed to have a no. of states in their sequence that is less than the maximum of 2^n. This type of sequence is called a truncated sequence.
* One common modulus for counters with truncated sequences is ten (called MOD10). Counters with ten states in their sequence are called decade counters.

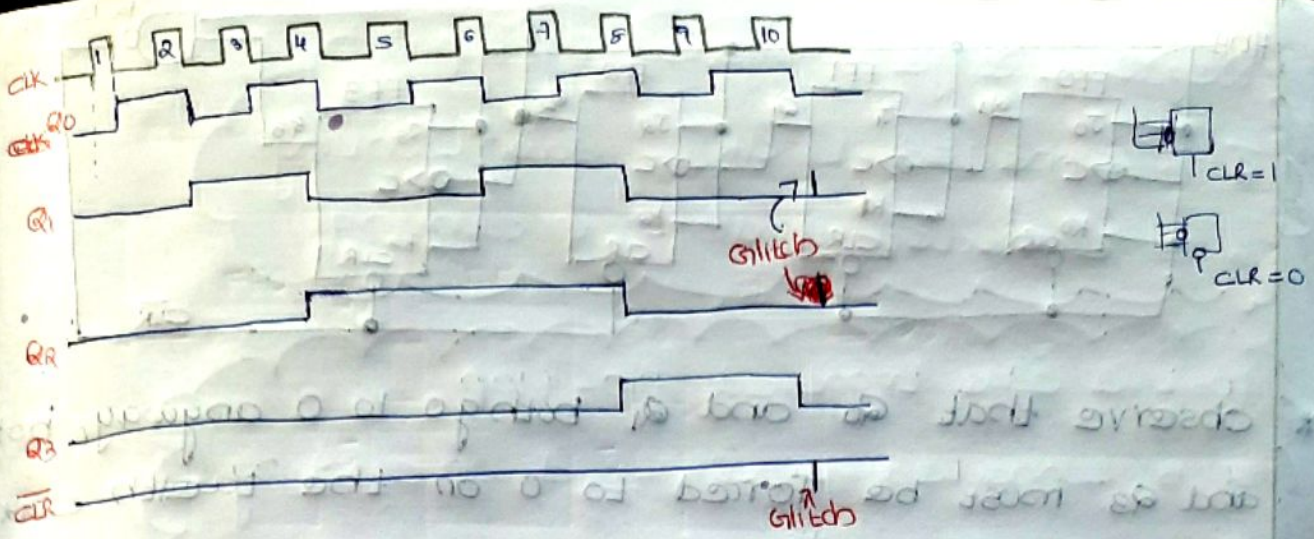
A decade counter with a count sequence of zero (0000) through nine (1001) is a BCD decade counter bcoz its ten state sequence produces the BCD code.

I/P	Q_3 (MSB)	Q_2	Q_1	Q_0 (LSB)	CLEAR
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1
10	①	0	①	0	0

clear
-ve ~~asynchronous~~
o asynchronous and
reset asynchronous
clear (+ve) =>
1 -> reset



Notice that only Q_1 and Q_3 are connected to the NAND gate inputs. This arrangement is an example of partial decoding, in which the two unique states ($Q_1 = 1$ and $Q_3 = 1$) are sufficient to decode the count of ten because none of the other states (zero through nine) have both Q_1 and Q_3 HIGH at the same time. When the counter goes into count ten (1010), the decoding gate's output goes LOW and asynchronously resets all the flip-flops.

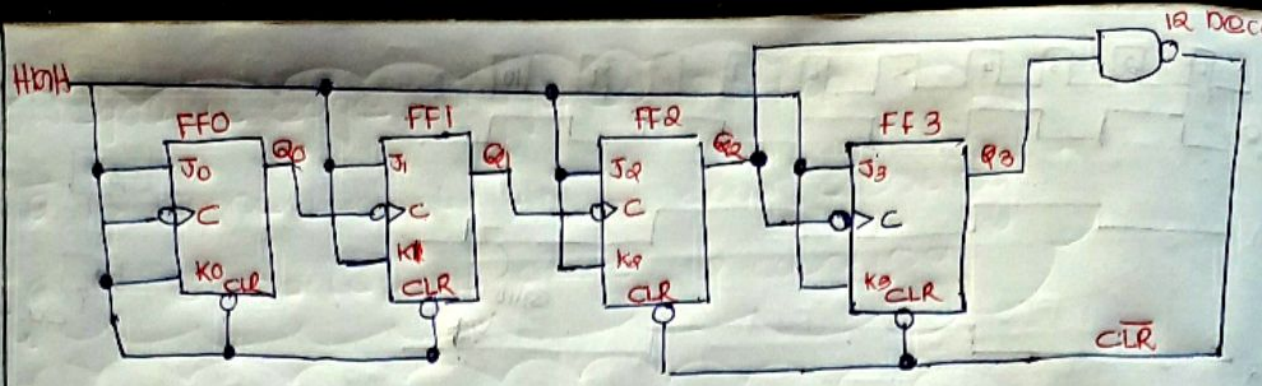


Notice that there is a glitch on the Q_1 waveform. The reason for this glitch is that Q_1 must first go HIGH before the count of ten can be decoded.

2 Show how an asynchronous can be implemented having a modulus of twelve with a straight binary sequence from 0000 through 1011.

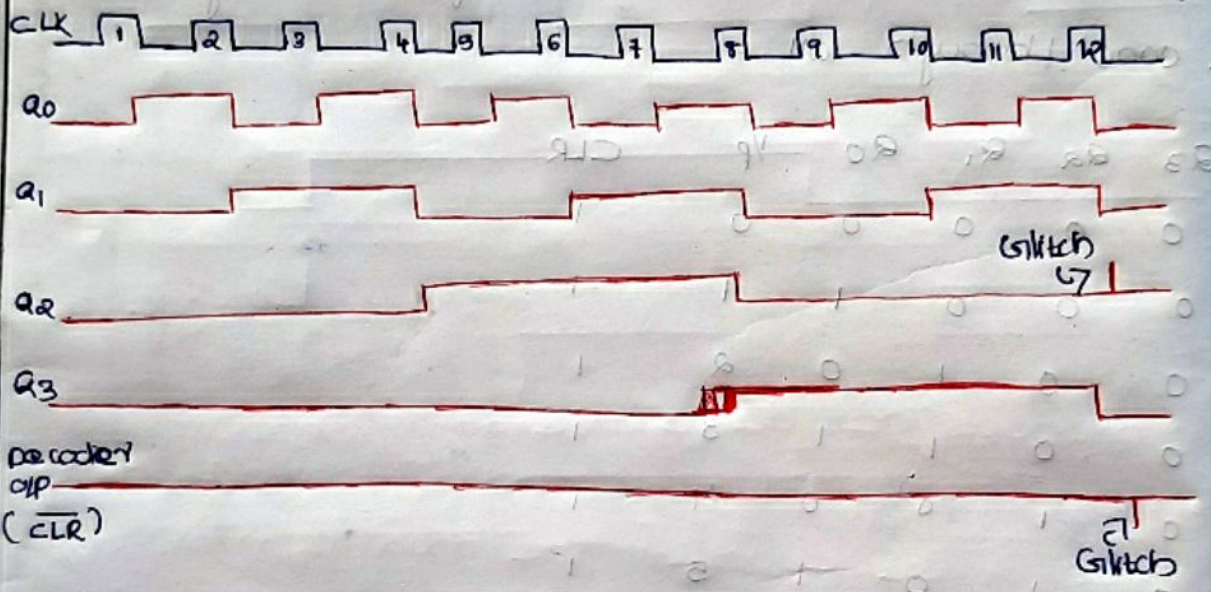
A.

Q_3	Q_2	Q_1	Q_0	I/p	CLR
0	0	0	0	0	1
0	0	0	1	1	1
0	0	1	0	2	1
0	0	1	1	3	1
0	1	0	0	4	1
0	1	0	1	5	1
0	1	1	0	6	1
0	1	1	1	7	1
1	0	0	0	8	1
1	0	0	1	9	1
1	0	1	0	10	1
1	0	1	1	11	1
1	1	0	0	12	0



* observe that Q_0 and Q_1 both go to 0 anyway, bot... and Q_3 must be forced to 0 on the twelfth clock p...

* The NAND gate partially decodes count twelve (1100), ... resets flip-flop 2 and flip-flop 3. Thus on the twelfth clock pulse, the counter is forced to recycle from count eleven to count zero.



UP-DOWN counter

A counter which can count values either in the forward direction or reverse direction is called an up-down counter or reversible counter.

3 bit up/down ripple counter

We have designed the up counters and down counters separately. But in practice both these modes are combined. A mod control input (M) is used to select either up or down

mode. A combinational circuit is required b/w each pair of flip-flops.

M = 0 → up counting

M = 1 → down counting

M	Q	$\bar{Q}$	Y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Q

$\bar{Q}$

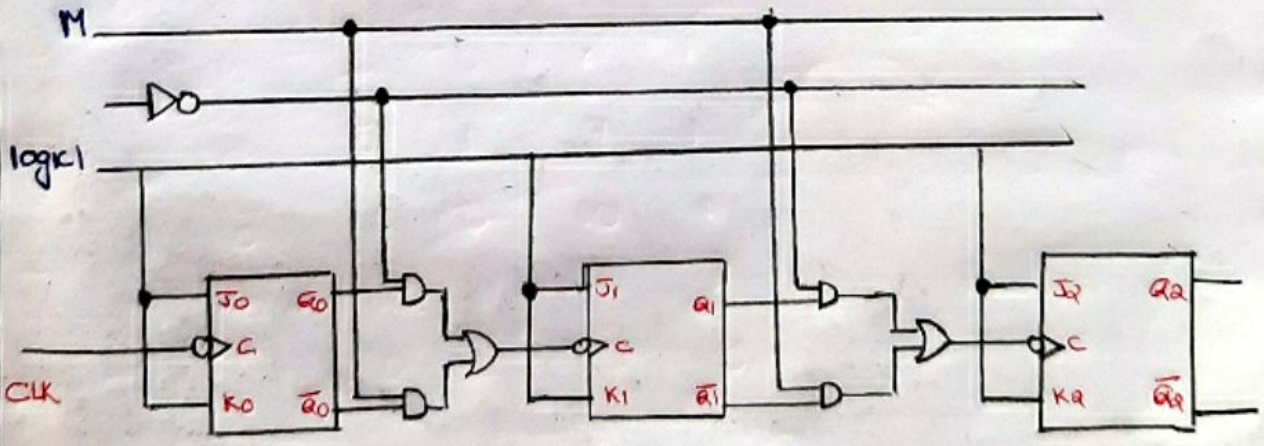
K map for Y

$Q\bar{Q}$ \ M	0	1
00	0	0
01	0	1
11	1	1
10	1	0

→ $M\bar{Q}$

$\bar{M}Q$

$M\bar{Q} + \bar{M}Q$ (clock for second and third flip-flop).



Negative edge triggered flip flop

$\bar{Q}$ is connected to clk - down counting

Q is connected to clk - up counting

positive edge triggered flip-flop

Q is connected to clk - down counting

$\bar{Q}$ is connected to clk - up counting

Negative edge triggered flip flop

$\bar{Q}$ is connected to clk - down counting

Q is connected to clk - up counting

positive edge triggered flip-flop

Q is connected to clk - down counting

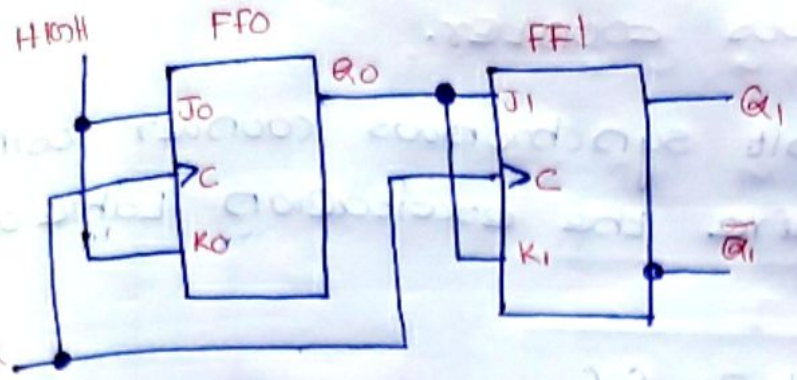
$\bar{Q}$ is connected to clk - up counting

Synchronous counter

* Synchronous counters are so called because the clock input of all the individual flip-flops within the counter are all clocked together at the same time by the same clock signal.
* However, with the synchronous counter, the external clock signal is connected to the clock input of every individual f-f within the counter so that all of the flip-flops are clocked together simultaneously at the same time giving a fixed time relationship. In other words, changes in the output occur in "synchronisation" with the clock signal.
* No propogation delay.

A 2-Bit synchronous counter

The operation of this synchronous counter is as follows,



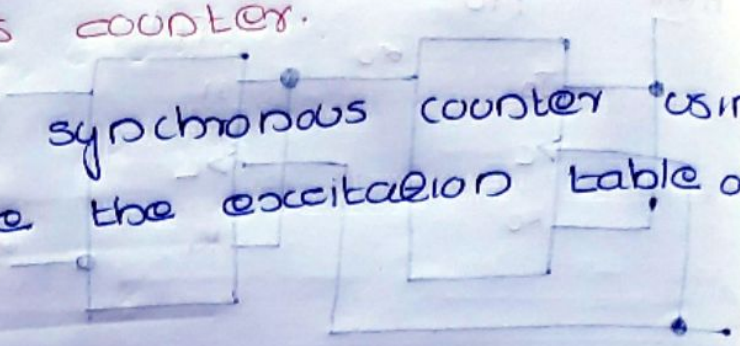
* First assume that the counter is initially in the binary 0 state, ie, both F-f are RESET. when the +ve edge of the 1st F-F is applied. FF-0 will toggle and Q0 will therefore go HIGH. Inputs J_1 and K_1 are both low bcoz Q_0, to which they are connected, so FF1 remains on the present state ie, $Q_1 = 0$

* After CLK1, $Q_0 = 1$ and $Q_1 = 0$. when the leading edge of CLK2 occurs, FF0 will toggle and Q_0 will go Low since FF1 has a HIGH ($Q_0 = 1$) on its J_1 and K_1 inputs at the triggering edge of this clock pulse, the F-F toggles and Q_1 goes HIGH and so on.

CLK	Q_1	Q_0
initially	0	0
1	0	1
2	1	0
3	1	1
4	0	0

3 bit syn

3 bit synchronous counter.



Prob To design 3 bit synchronous counter using J-K ff, we require the excitation table of J-K f-f.

excitation table of JK f-f

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

Truth table of 3 bit synchronous counter

Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	X	0	X	1	X
0	0	1	0	X	1	X	X	1
0	1	0	0	X	X	0	1	X
0	1	1	1	X	X	1	X	1
1	0	0	X	0	0	X	1	X
1	0	1	X	0	1	X	X	1
1	1	0	X	0	X	0	1	X
1	1	1	X	1	X	1	X	1
0	0	0						

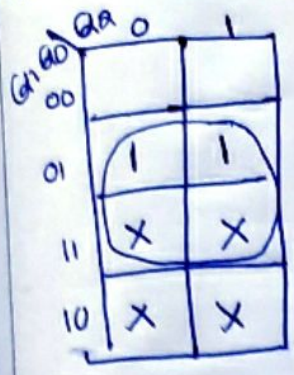
here MSB → Q_2 and LSB → Q_0

we have to draw the k map for J_0, K_0, J_1, K_1, J_2 and K_2

J_0 and K_0 contains all the element is either 1 or X

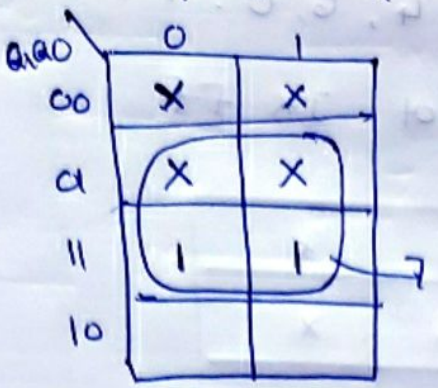
$\therefore$ $J_0 = K_0 = 1$

K map for J_1



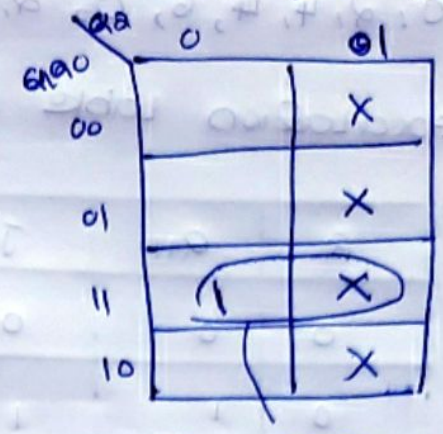
$J_1 = Q_0$

K map for K_1



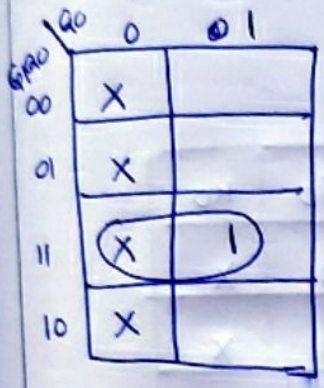
$K_1 = Q_0$

K map for J_2.



$J_2 = Q_1 Q_0$

K map for K_2.

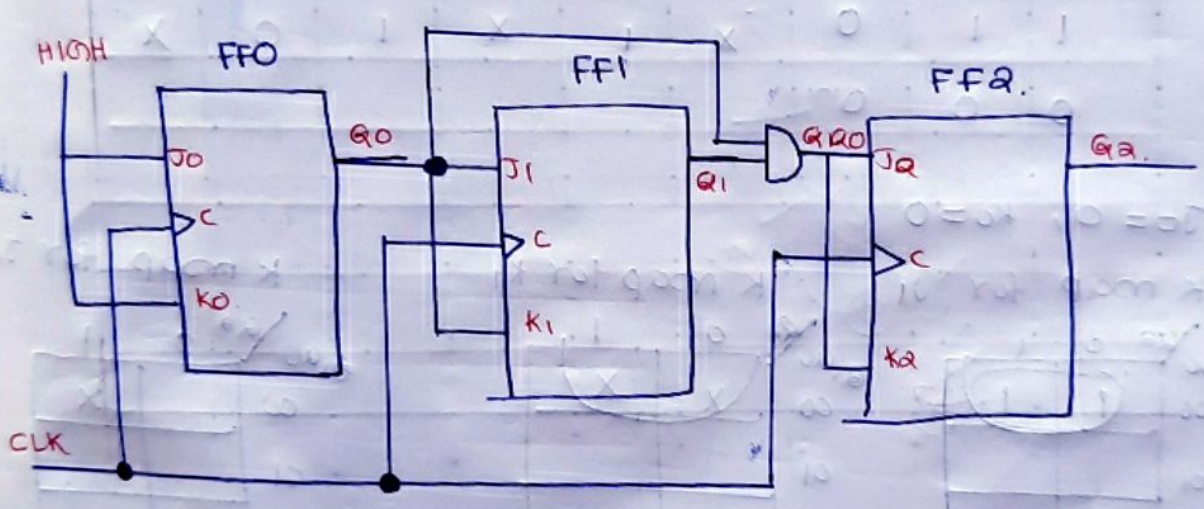


$K_2 = Q_1 Q_0$

$\therefore$ Also $J_0 = K_0 = 1$

$J_1 = K_1 = Q_0$

$J_2 = K_2 = Q_1 Q_0$.



In the circuit diagram J_0, K_0 inputs of first FF is set to HIGH and the J_1/K_1 input of 2nd FF is Q_0 and the J_2 K_2 inputs of 3rd Flip flop (FF2) is $Q_1 Q_0$.

2. 0, 2, 4, 6, 8, 10, 12, 14, 16, 18 ...

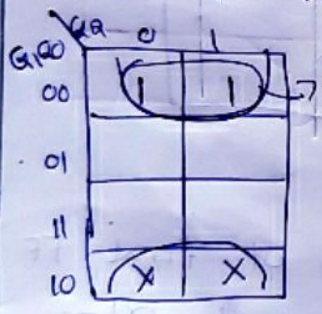
A Excitation table of JK FF

Q	Q+1	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	X	1	X	0	X
0	1	0	1	X	X	1	0	X
1	0	0	X	0	1	X	0	X
1	1	0	X	1	X	1	0	X
0	0	0						

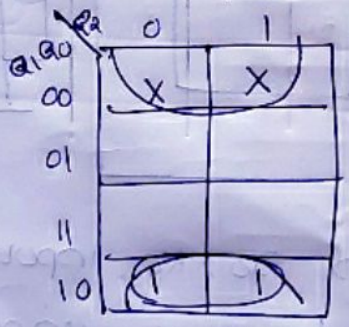
$J_0 = 0$, $K_0 = 0$

K map for J_1



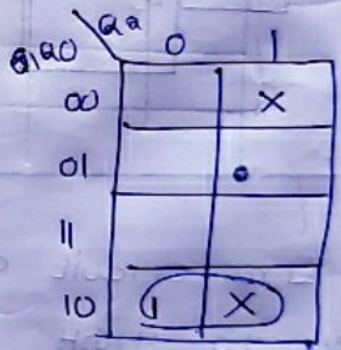
$J_1 = \bar{Q_0}$

K map for K_1



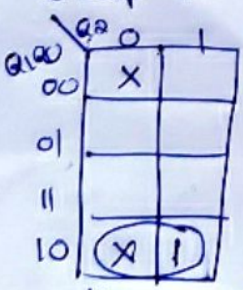
$K_1 = \bar{Q_0}$

K map for J_2



$J_2 = Q_1\bar{Q_0}$

kmap for K_2

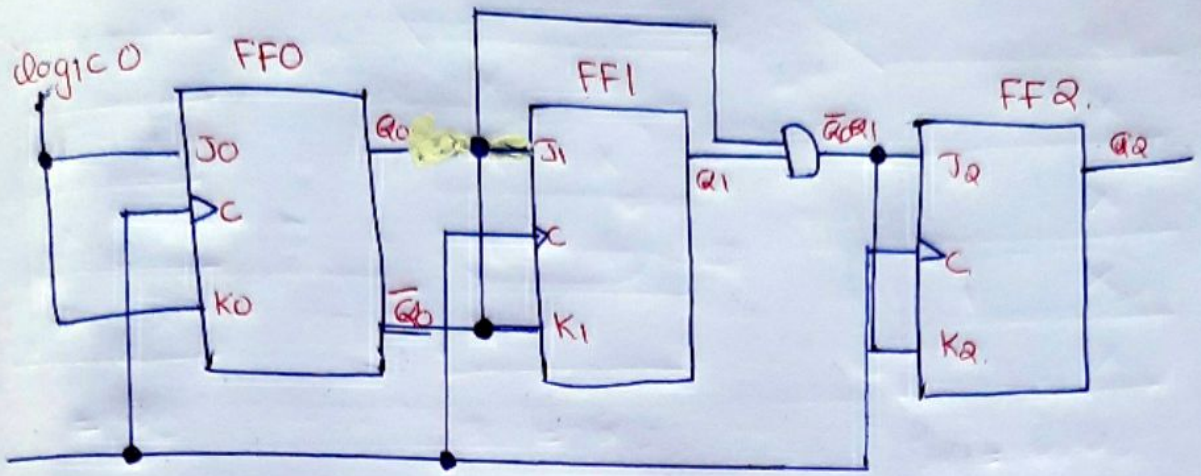


$K_2 = Q_1\bar{Q_0}$

$J_1 = K_1 = \bar{Q_0}$

$J_2 = K_2 = Q_1\bar{Q_0}$

$J_0 = K_0 = 0$


logic 0
FF0
FF1
FF2.
J0
C
K0
Q0
$\bar{Q}_0$
J1
C
K1
Q1
$\bar{Q}_0 Q_1$
JQ
C
KQ.
QQ

K map draw considering inputs of binary
only 1 no cell is mark zero.

0, 2, 4, 6 series synchronous counter

Truth table.

Here we also consider the invalid cases

Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	X	1	X	0	X
0	1	0	1	X	X	1	0	X
1	0	0	X	0	1	X	0	X
1	1	0	X	1	X	1	0	X
0	0	1	X	X	X	X	X	X
0	1	1	X	X	X	X	X	X
1	0	1	X	X	X	X	X	X
1	1	1	X	X	X	X	X	X

(Row labels in margin: 0, 2, 4, 6, 1, 3, 5, 7; last four rows marked "10 states")

$J_0 = 0$ (in k map no groups)

All the values of K_0 is X ∴

$K_0 = 1$

K map for J_1

Q_1Q_0 \ Q_2	0	1
00	1 (0)	1 (1)
01	X (2)	X (3)
11	X (6)	X (7)
10	X (4)	X (5)

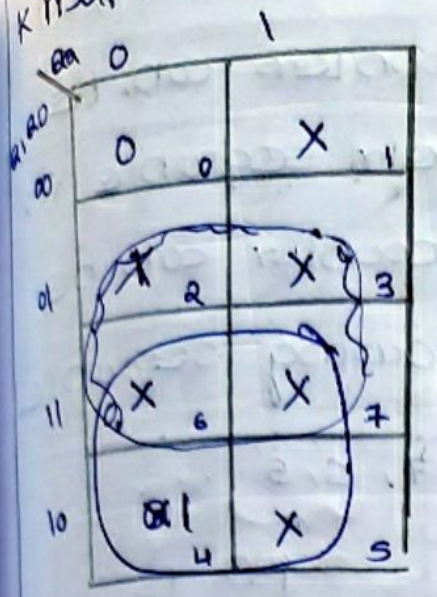
$J_1 = 1$

K map for K_1

Q_1Q_0 \ Q_2	0	1
00	X (0)	X (1)
01	1 (2)	X (3)
10	1 (6)	X (7)
11	X (4)	X (5)

$K_1 = 1$

K map for J_2.



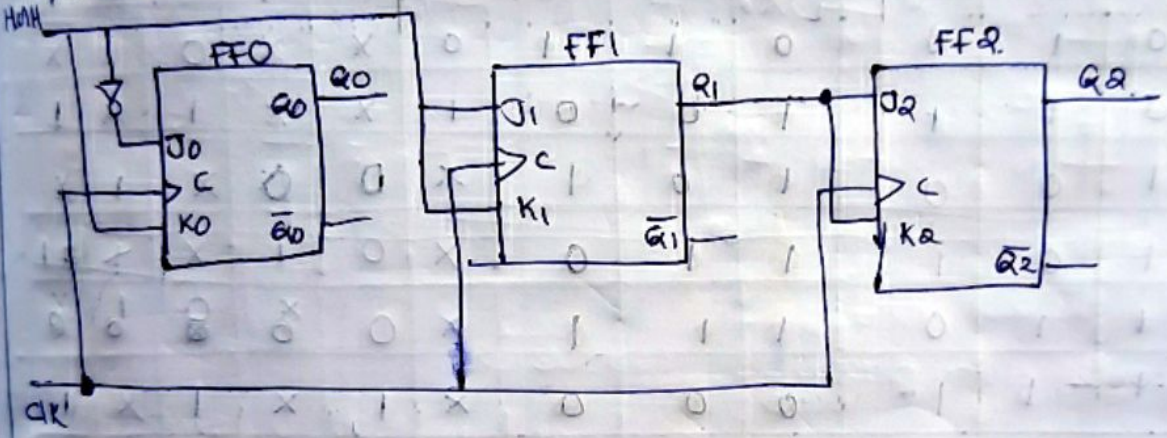
$J_2 = Q_0 Q_1 = K_2$.

$J_0 = 0,\ K_0 = 1$

$J_1 = K_1 = 1$

$J_2 = K_2 = Q_1$

Circuit



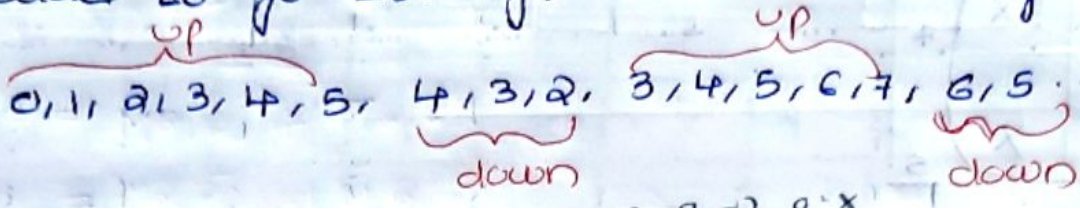
3-bit up/down synchronous counter

An up/down counter is one that is capable of progressing in either direction through a certain sequence. An up/down counter is sometimes called a bidirectional counter.

A 3-bit binary up counter that advances upward through its sequence (0, 1, 2, 3, 4, 5, 6, 7) and then can be reversed so that it goes through the sequence..

in the opposite direction (7, 6, 5, 4, 3, 2, 1, 0)

* in general, most up/down counters can be reversed at any point in their sequence. for instance, the 3-bit binary counter can be made to go through the following sequence.



Truth table

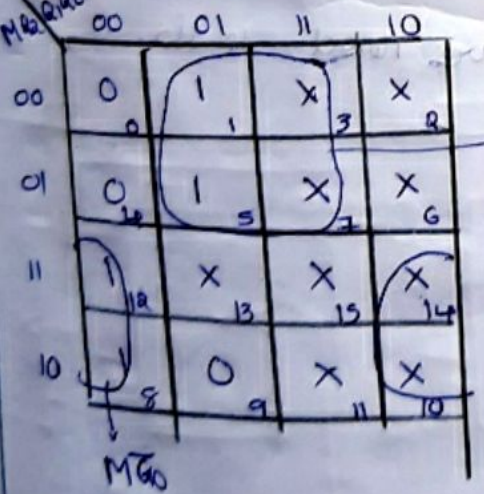
M	Q_2	Q_1	Q_0	Q_2^{+1}	Q_1^{+1}	Q_0^{+1}	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	0	1	0	X	0	X	1	X
0	0	0	1	0	1	0	0	X	1	X	X	1
0	0	1	0	0	1	1	0	X	X	0	1	X
0	0	1	1	1	0	0	1	X	X	1	0X	1
0	1	0	0	1	0	1	X	0	0	X	1	X
0	1	0	1	1	1	0	X	0	1	X	X	1
0	1	1	0	1	1	1	X	0	X	0	1	X
0	1	1	1	0	0	0	X	1	X	1	X	1
1	0	0	0	1	1	1	1	X	1	X	1	X
1	0	0	1	0	0	0	0	X	0	X	X	1
1	0	1	0	0	0	1	0	X	X	1	1	X
1	0	1	1	0	1	0	0	X	X	0	X	1
1	1	0	0	0	1	1	X	1	1	X	1	X
1	1	0	1	1	0	0	X	0	0	X	X	1
1	1	1	0	1	0	1	X	0	X	1	1	X
1	1	1	1	1	1	0	X	0	X	0	X	1

M=0 → upcounting

M=1 → down counting

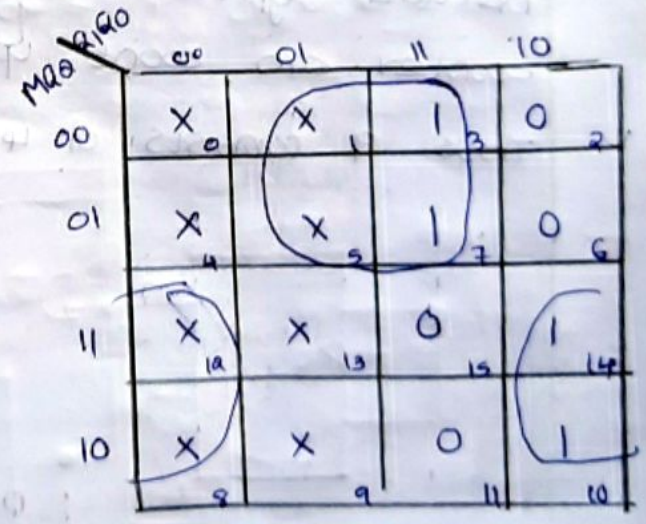
$J_0 = K_0 = 1$

k map for J_1



$J_1 = M\bar{Q}_0 + \bar{M}Q_0$

k map for K_1



$K_1 = M\bar{Q}_0 + \bar{M}Q_0$

K map for J_2.

MQ_1 \ Q_0	00	01	11	10
00	0 (0)	0 (1)	1 (3)	0 (2)
01	X (4)	X (5)	X (7)	X (6)
11	X (12)	X (13)	X (15)	X (14)
10	1 (8)	X (9)	0 (11)	0 (10)

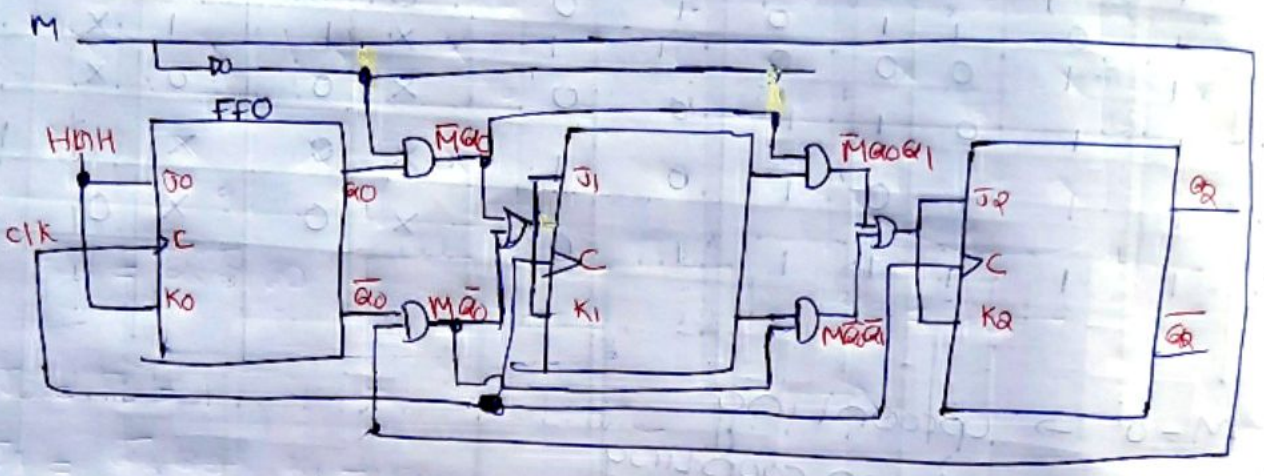
$$\bar{J}_2 = M\bar{Q}_1\bar{Q}_0 + \bar{M}Q_1Q_0$$

$$K_2 = M\bar{Q}_1\bar{Q}_0 + \bar{M}Q_1Q_0$$

$$J_0 = K_0 = 1$$

$$J_1 = K_1 = M\bar{Q}_0 + \bar{M}Q_0$$

$$J_2 = K_2 = M\bar{Q}_1\bar{Q}_0 + \bar{M}Q_1Q_0$$



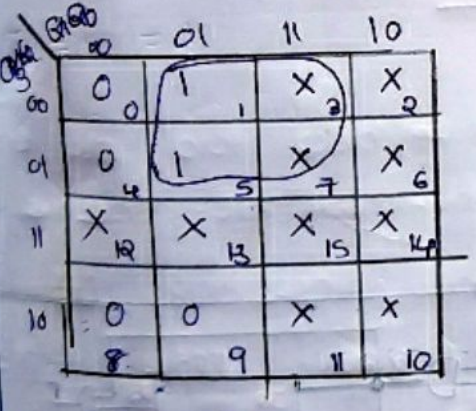
Decade synchronous counters.

Decade synchronous counter or Mod 10 synchronous counter can count upto 10 states (ie, from 0 to 9)

Thus 9 consists of 4 bits, we need 4 ffs.

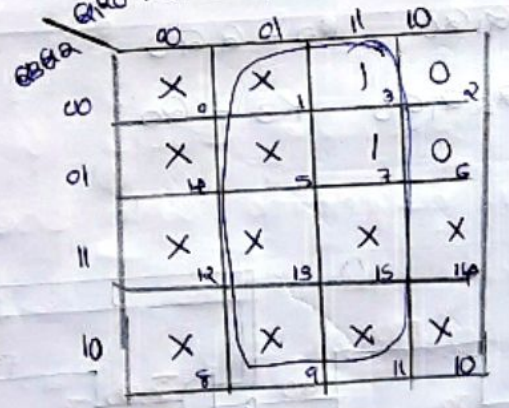
Q_3	Q_2	Q_1	Q_0	Q_3^+	Q_2^+	Q_1^+	Q_0^+	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	0	0	0 1	0	X	0	X	0	X	1	X
0	0	0	1	0	0	1	0	0	X	0	X	1	X	X	1
0	0	1	0	0	0	1	1	0	X	0	X	X	0	1	X
0	0	1	1	0	1	0	0	0	X	1	X	X	1	X	1
0	1	0	0	0	1	0	0 1	0	X	X	0	0	X	1	X
0	1	0	1	0	1	1	0	0	X	X	0	1	X	X	1
0	1	1	0	0	1	1	1	0	X	X	0	X	0	1	X
0	1	1	1	1	0	0	0	1	X	X	1	X	1	X	1
1	0	0	0	1	0	0	1	X	0	0	X	0	X	1	X
1	0	0	1	0	0	0	0	X	1	0	X	0	X	X	1
1	0	1	0	X	X	X	X	X	X	X	X	X	X	X	X
1	0	1	1	X	X	X	X	X	X	X	X	X	X	X	X
1	1	0	0	X	X	X	X	X	X	X	X	X	X	X	X
1	1	0	1	X	X	X	X	X	X	X	X	X	X	X	X
1	1	1	0	X	X	X	X	X	X	X	X	X	X	X	X
1	1	1	1	X	X	X	X	X	X	X	X	X	X	X	X

Kmap for J_1



$\overline{J_1} = \overline{Q_3} Q_0$

Q_2Q_0 kmap for K_1



$K_1 = Q_0$

K - map for J_2.

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0 (0)	0 (1)	1 (3)	0 (2)
01	X (4)	X (5)	X (7)	X (6)
11	X (12)	X (13)	X (15)	X (14)
10	0 (8)	0 (9)	X (11)	X (10)

Q_1Q_0

K - map for J_1.

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	X (0)	X (1)	X (3)	X (2)
01	0 (4)	0 (5)	1 (7)	0 (6)
11	X (12)	X (13)	X (15)	X (14)
10	X (8)	X (9)	X (11)	X (10)

Q_1Q_0

K - map for J_3.

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	0 (0)	0 (1)	0 (3)	0 (2)
01	0 (4)	0 (5)	1 (7)	0 (6)
11	X (12)	X (13)	X (15)	X (14)
10	X (8)	X (9)	X (11)	X (10)

$J_3 = Q_2Q_1Q_0$

K map for K_3

Q_3Q_2 \ Q_1Q_0	00	01	11	10
00	X (0)	X (1)	X (3)	X (2)
01	X (4)	X (5)	X (7)	X (6)
11	X (12)	X (13)	X (15)	X (14)
10	0 (8)	1 (9)	X (11)	X (10)

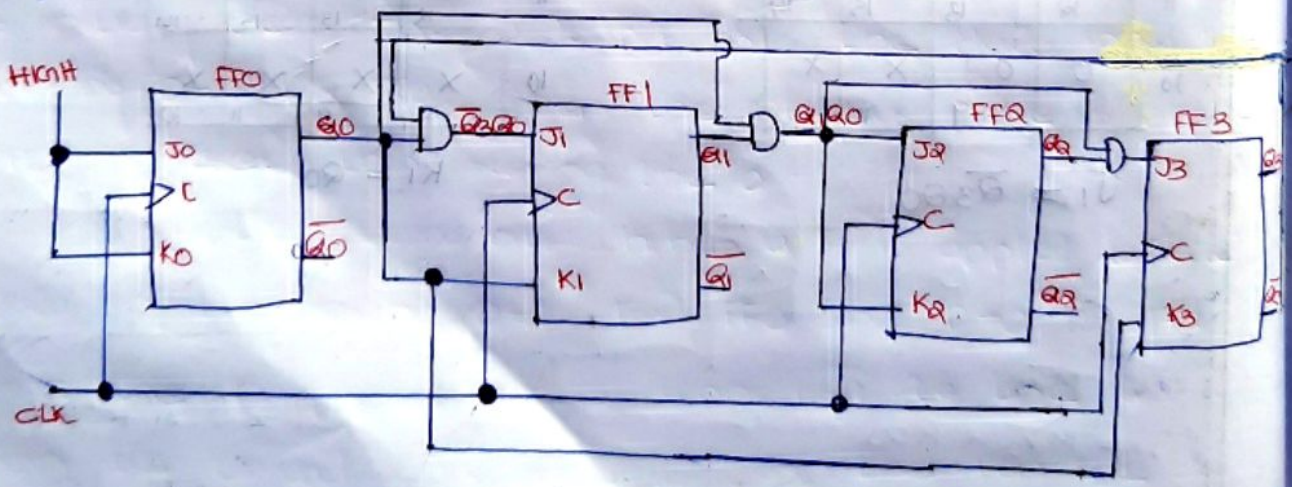
$K_3 = Q_0$.

$J_0 = K_0 = 1$

$J_1 = \overline{Q_3}\,Q_0 \quad K_1 = Q_0$

$J_2 = Q_1Q_0 \quad K_2 = Q_1Q_0$

$J_3 = Q_2Q_1Q_0 \quad K_3 = Q_0$.

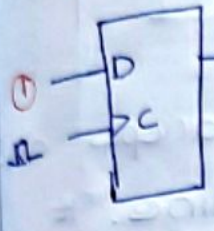


Register

A register is a digital circuit with two basic functions, data storage and data movement.

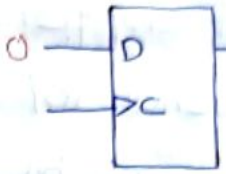
* the concept of storing a 1 or a 0 in a D ff.

1 is stored



when a 1 is on D, Q becomes a 1 at the triggering edge of clk or remains a 1 if already in the SET state.

0 is stored



when a 0 is on D, Q becomes a 0 at the triggering edge of CLK or remains a 0 if already in the RESET state.

Shift Register

* shif registers consist of arrangements of ffs and are amp on applications involving the storage and transfer of data in a digital system.
* If the register is capable of shifting bits either towards right hand side or towards LHS is known as shift register. An 'N' bit shift register contains 'N' ffs.
* shift registers are basically of 4 types. There are.

1. serial in serial out shift register.
2. serial in parallel out " "
3. parallel in serial out " "
4. Parallel in parallel out " "

* The storage capacity of a register is the total no of bits (1s and 0s) of digital data it can retain.
* Each stage (ff) in a shift register represents one bit of

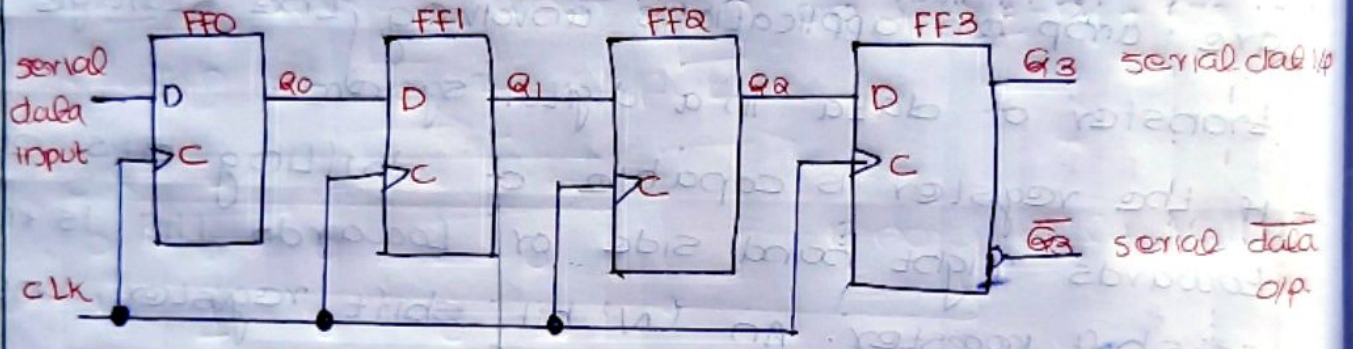
storage capacity. therefore, the no. of stages on a register determines its storage capacity.

* The shift capability of a register permits the movement of data from stage to stage within the register or into or out of the register upon application of clock pulses.

serial in/ serial out shift registers.

The serial in/ serial out shift register accepts data serially, i.e. one bit at a time on a single line. It produces the stored information on its output also in serial form.

Figure shows a 4-bit device implemented with D ffs. with 4 ffs, this register can store up to 4 bits of data.



* Figure illustrates entry of the 4 bits 1010 into the register, beginning with the right-most bit. The register is initially clear. The 0 is put onto the data input line, making D=0 for FF0, when the 1st clock pulse is applied, FF0 is reset, thus storing the 0

* Next the second bit, which is a 1, is applied to the data input, making D=1 for FF0 and D=0 for FF1 bcoz the D input of FF1 is connected to the Q0 output. when the second clock pulse occurs, the 1 on the data input is shifted into FF0, causing FF0 to set, and the

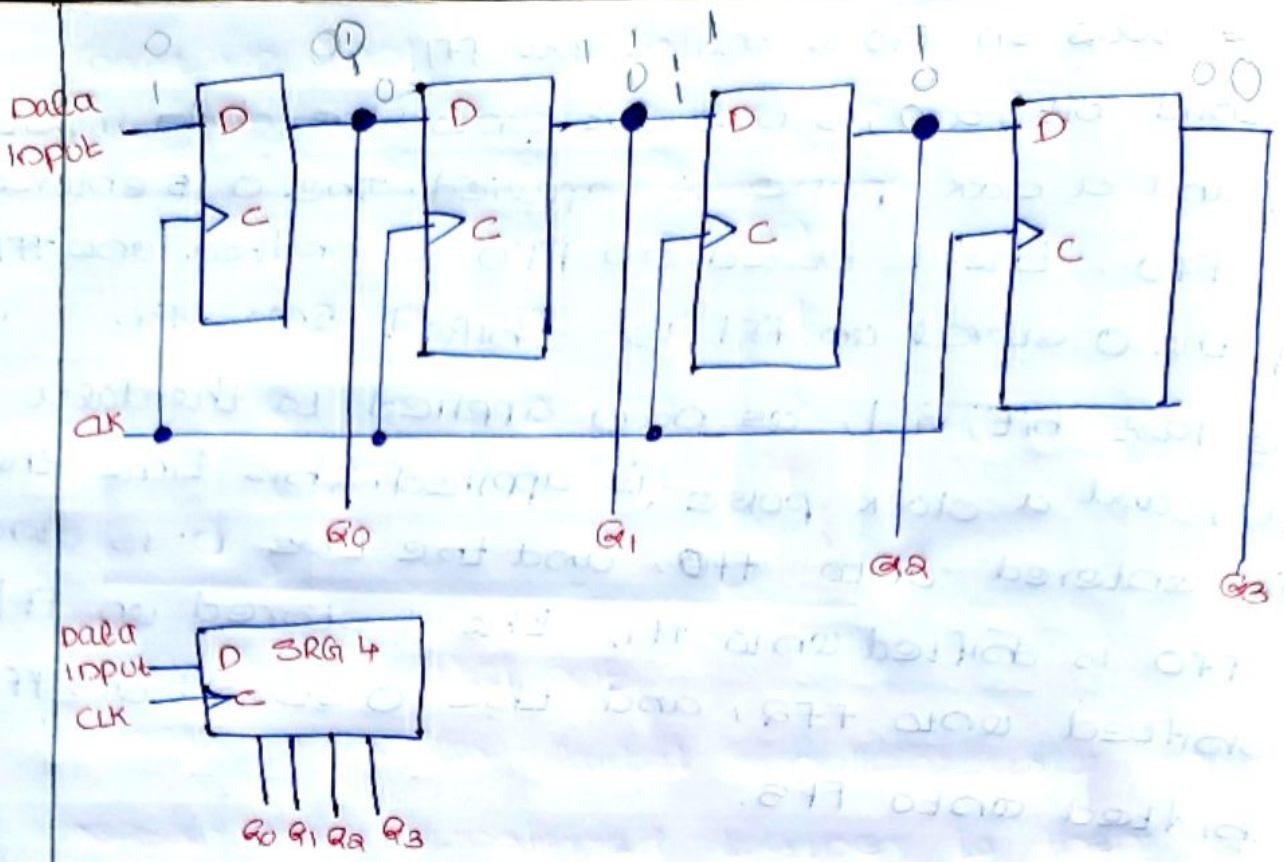
0 that was on FF0 is shifted into FF1
the third bit, a 0, is now put onto the data-input line, and a clock pulse is applied. The 0 is entered into FF0, the 1 stored on FF0 is shifted onto FF1, and the 0 stored on FF1 is shifted onto FF2.
The last bit, a 1, is now applied to the data input, and a clock pulse is applied. This time the 1 is entered onto FF0, and the 0 is stored on FF0 is shifted onto FF1, the 1 stored on FF1 is shifted onto FF2, and the 0 stored on FF2 is shifted onto FF3.

FF
Truth table.

CLK	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	0	0	0	0
2	1	0	0	0
3	0	1	0	0
4	1	0	1	0
5	0	1	0	1
6	0	0	1	0
7	0	0	0	1
8	0	0	0	0

After clk 4, the 4-bit number is completely stored on register
→ 1st data bit
→ 2nd " "
3rd " "
4th " "

Serial in parallel out (SIPO)
The data is stored onto the register serially while it is retrieved from it in parallel-fashion.



Once the data are stored, each bit appears on its respective output line, and all bits are available simultaneously, rather than on a bit-by-bit basis as with the serial output.

clk	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	0	1	1	0
4	1	0	1	1

Binary data 1011

1 → 1 0 0 0

10 → 1 1 0 0

0 → 0 1 1 0

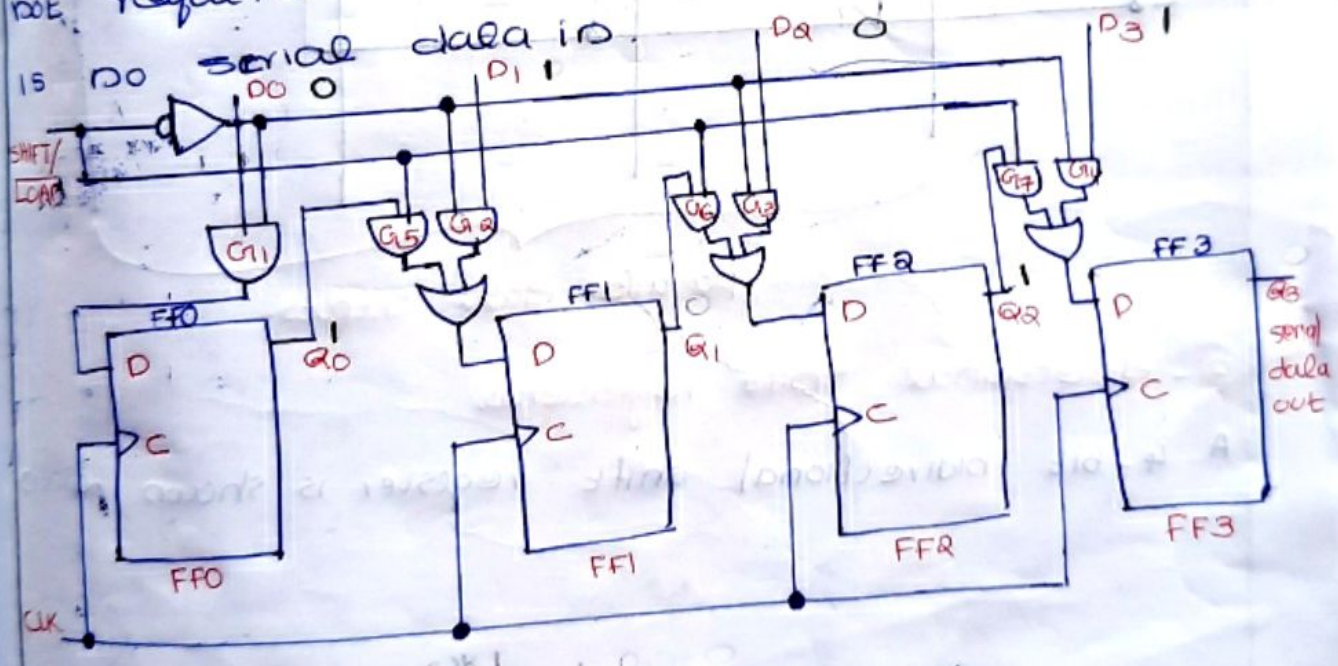
1 → 1 0 1 1

Parallel in serial out shift registers

* In parallel in serial out shift registers, the data is clocked onto the register in parallel format while it is retrieved from it serially.
* Notice that there are four data-input lines, D_0, D_1, D_2 and D_3, and a SHIFT/$\overline{\text{LOAD}}$ input, which allows

four bits of data to load in parallel onto the register when SHIFT/$\overline{LOAD}$ is LOW, gates G_1 through G_4 are enabled, allowing each data bit to be applied to the D input of its respective ff. when a clock pulse is applied, the ffs with D=1 will set and those with D=0 will reset, thereby storing all 4 bits simultaneously.

when SHIFT/$\overline{LOAD}$ is HIGH, gates G_1 through G_4 are disabled and gates G_5 through G_7 are enabled, allowing the data bits to shift right from one stage to the next. The OR gates allow either the normal shifting operation or the parallel data-entry operation, depending on which AND gates are enabled, by the level on the SHIFT/$\overline{LOAD}$ input. Notice that FF0 has a single AND to disable the parallel input D_0. It does not require an AND/OR arrangement bcoz there is no serial data in.



load = 0
shift = 1

CLK 1
load = 0
$Q_0 = 0$ $Q = 1$, $Q_2 = 0$, $Q_3 = 1$

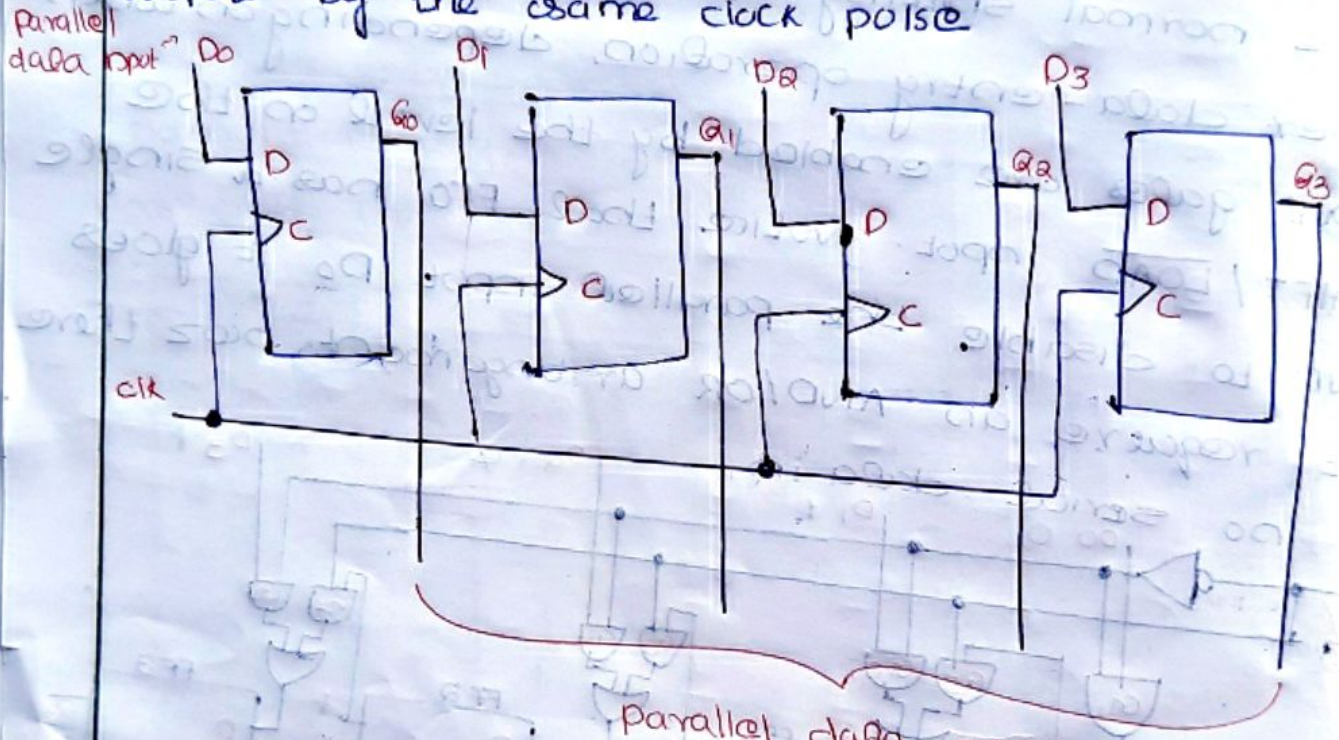
CLK 2

Data i/p: 0101

load	clk	Q_0	Q_1	Q_2	Q_3	
1	0	0	0	0	0	
0	1	0	1	0	1	shifting
	2	0	0	1	0	
	3	0	0	0	1	
	4	0	0	0	0	

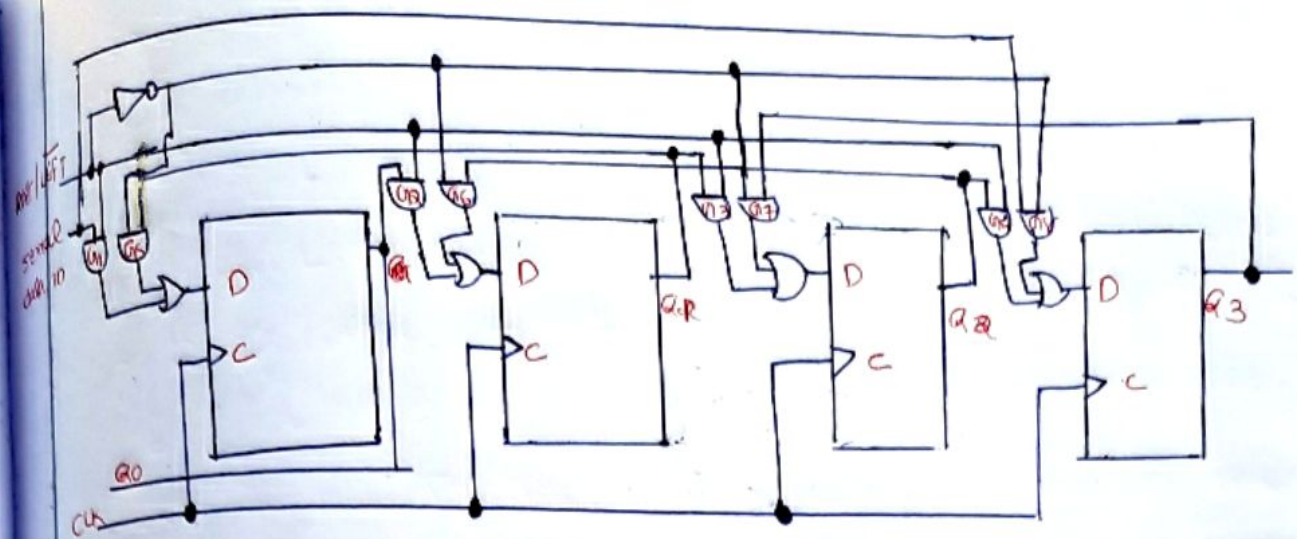
Parallel in parallel out

Parallel data is loaded simultaneously into the register, and transferred together to their respective outputs by the same clock pulse



Bi-directional shift registers

A 4-bit bidirectional shift register is shown below.



A HIGH on the RIGHT / $\overline{\text{LEFT}}$ control input allows data bits inside the register to be shifted to the right, and a Low enables data bits inside the register to be shifted to the left.

when the RIGHT / $\overline{\text{LEFT}}$ control input is HIGH, gates G1 through G4 are enabled, and the state of the Q output of each ff is passed through to the D input of the following ff.

when a clock pulse occurs, the data bits are shifted one place to the right. when the RIGHT / $\overline{\text{LEFT}}$ control input is Low, gates G5 through G8 are enabled, and the Q output of each ff is passed through to the D input of the preceding ff. when a clock pulse occurs, the data bits are then shifted one place to the left.