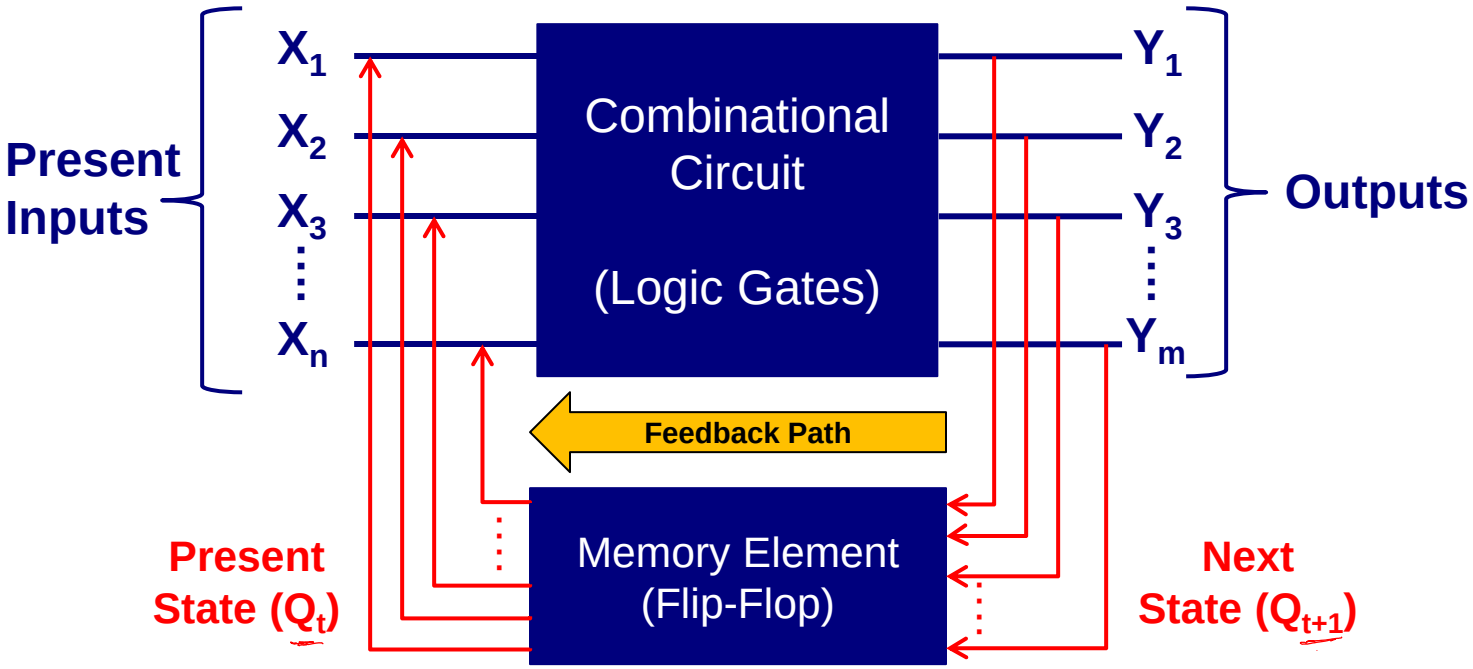
Sequential Circuits (Part-I)

By Nilesh Patidar and Shiraz Husain

Sequential Circuit



Does contain memory element/feedback path

By Nilesh Patidar

Combinational v/s Sequential

Combinational Circuits

- Outputs depend only on present inputs.
- Feedback path is not present.
- Memory elements are not required.
- Clock signal is not required.
- Easy to design.

Sequential Circuits

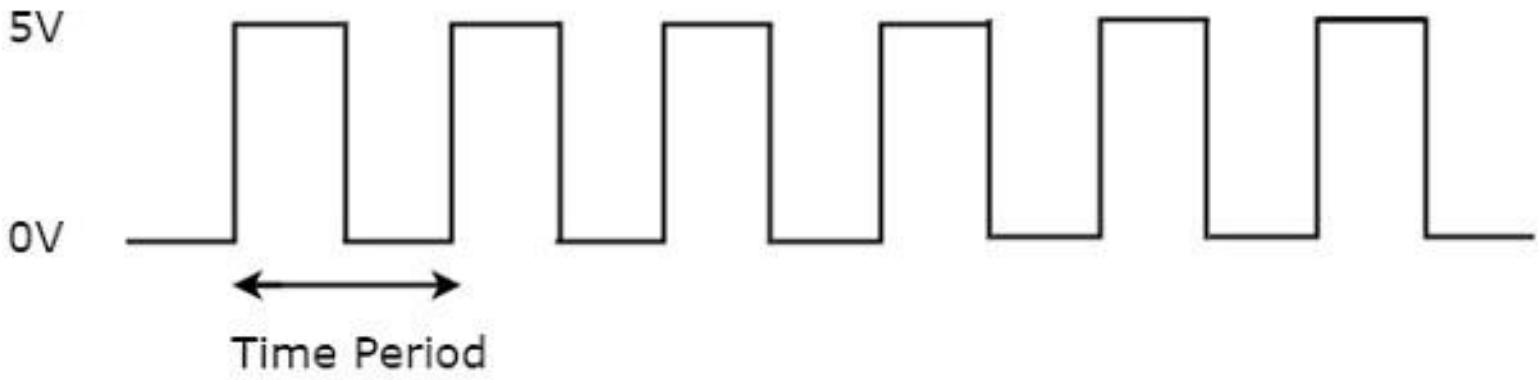
- Outputs depend on both present inputs and present state.
- Feedback path is present.
- Memory elements are required.
- Clock signal is required.
- Difficult to design.

Types of Sequential Circuits

- Following are the two types of sequential circuits −
  - Asynchronous sequential circuits
  - Synchronous sequential circuits
- Asynchronous sequential circuits
  - If the output of a sequential circuit depend on active transition of input, then it is called as **Asynchronous sequential circuit**.
- Synchronous sequential circuits
  - If the output of a sequential circuit depend on active transition of clock, then it is called as **synchronous sequential circuit**.

Clock signal

- Clock signal is a periodic signal and its ON time and OFF time need not be the same. We can represent the clock signal as a **square wave.**



Types Triggering

- Following are the two possible types of triggering in Sequential Circuits-
  - Level triggering
    - Positive Level
    - Negative Level
  - Edge triggering
    - Positive Edge
    - Negative Edge

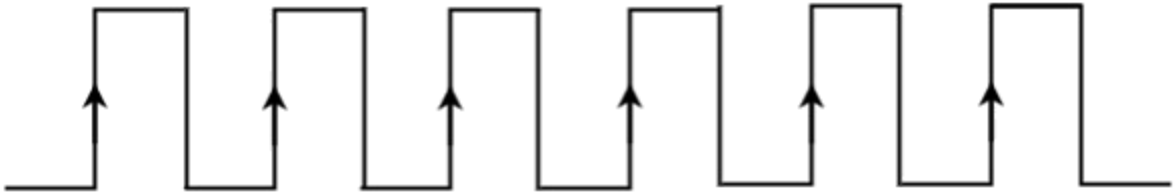
Level triggering

- If the sequential circuit is operated with the clock signal when it is in **Logic High**, then that type of triggering is known as **Positive level triggering**.

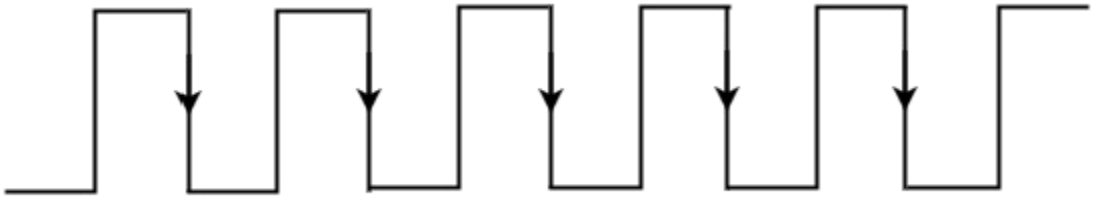
- If the sequential circuit is operated with the clock signal when it is in **Logic Low**, then that type of triggering is known as **Negative level triggering**.

Edge triggering

- If the sequential circuit is operated with the clock signal that is transitioning from Logic Low to High, then that type of triggering is known as **Positive edge triggering**.



- If the sequential circuit is operated with the clock signal that is transitioning from Logic High to Low, then that type of triggering is known as **Negative edge triggering**.

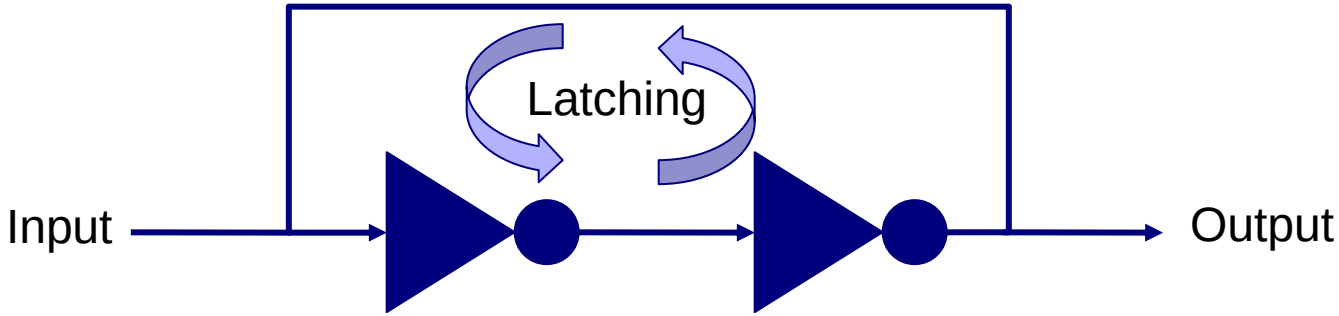


Memory Elements (Latch or Flip-flop)

- There are two types of memory elements based on the type of triggering that is suitable to operate it.
  - Latches
  - Flip-flops
- Latches operate with enable signal, which is **level sensitive** *(level triggering).*
- Flip-flops operate with clock signal, which is **edge sensitive** *(edge triggering).*

Basic Latch

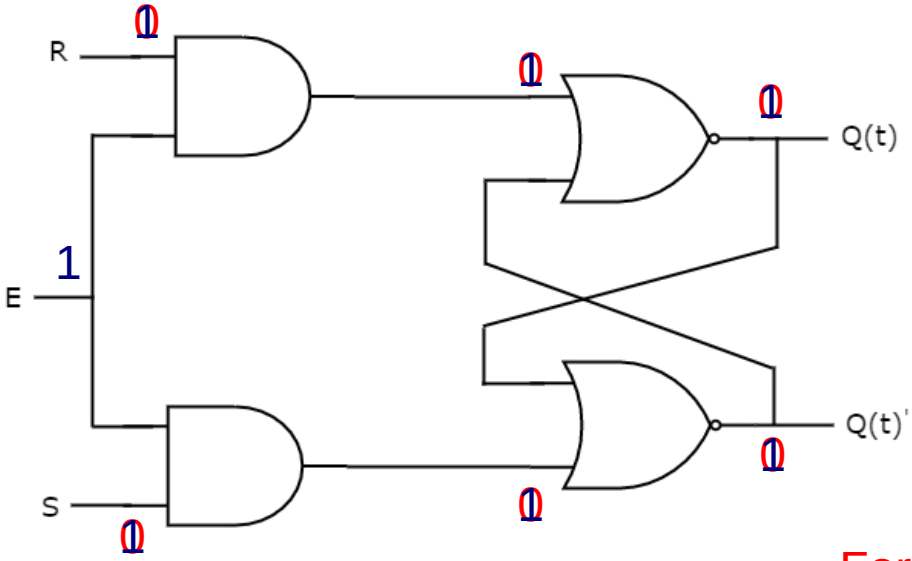
- A latch is a device used to hold the digital information.
- The basic latch can be designed using 2 inverters.



- Latch means store the information.

SR Latch (NOR)

- SR Latch is also called as **Set Reset Latch**. This latch affects the outputs as long as the enable, E is maintained at ‘1’



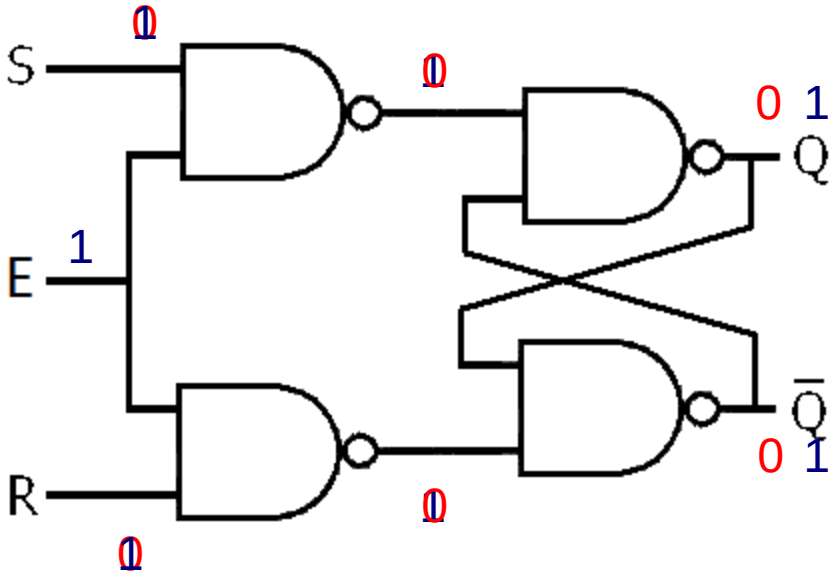
State Table

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	-

Forbidden State

By Nilesh Patidar

SR Latch (NAND)



State Table

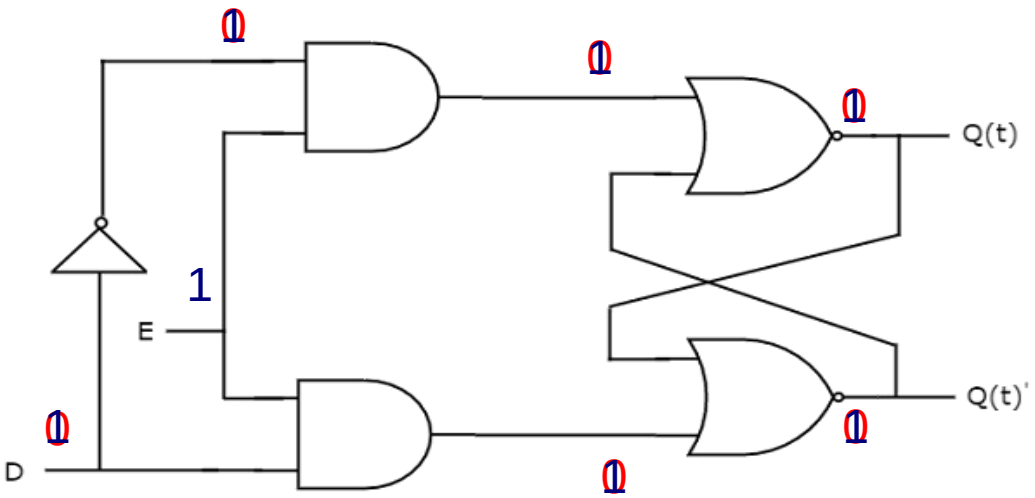
S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	-

Forbidden State

By Nilesh Patidar

D Latch

- There is one drawback of SR Latch. That is the next state value can't be predicted when both the inputs S & R are one. So, we can overcome this difficulty by D Latch. It is also called as **Data Latch.**



State Table

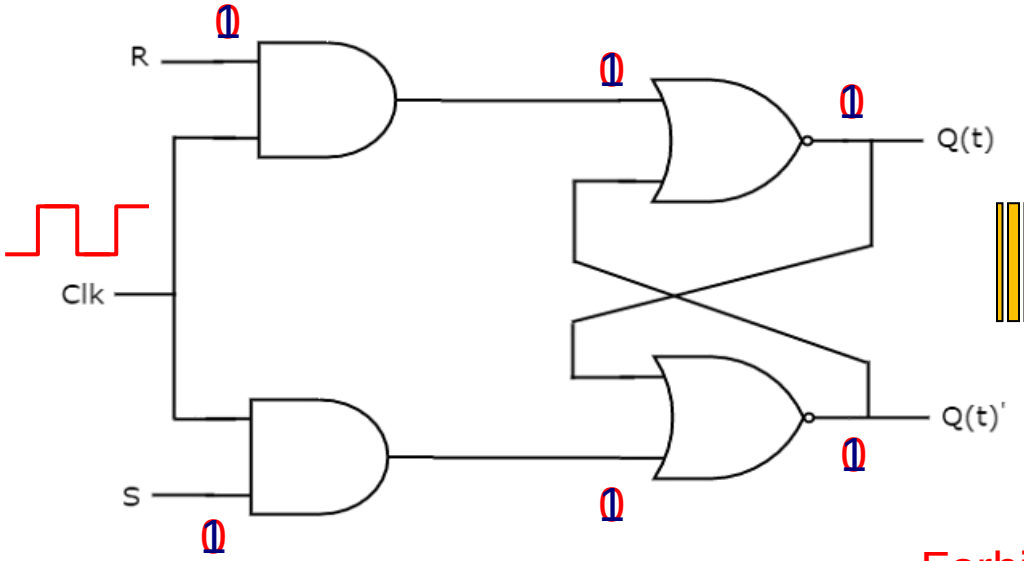
D	Q_{t+1}
0	0
1	1

Flip-Flops

- by **cascade two latches** in such a way that the first latch is enabled for every positive clock pulse and second latch is enabled for every negative clock pulse. So that the combination of these two latches become a flip-flop.
- There are four types of flip-flops.
  - SR Flip-Flop
  - D Flip-Flop
  - JK Flip-Flop
  - T Flip-Flop

SR Flip-flop

- SR flip-flop operates with only positive clock transitions or negative clock transitions.



State Table

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	-

Forbidden State

By Nilesh Patidar

SR Flip-flop

Characteristic Table

Present Input		Present State	Next State	Status
S	R	Q_t	Q_{t+1}	
0	0	0	0	No Change
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	X	Invalid
1	1	1	X	

Excitation Table

Present State	Next State	Present Input	
Q_t	Q_{t+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

By Nilesh Patidar

SR Flip-flop

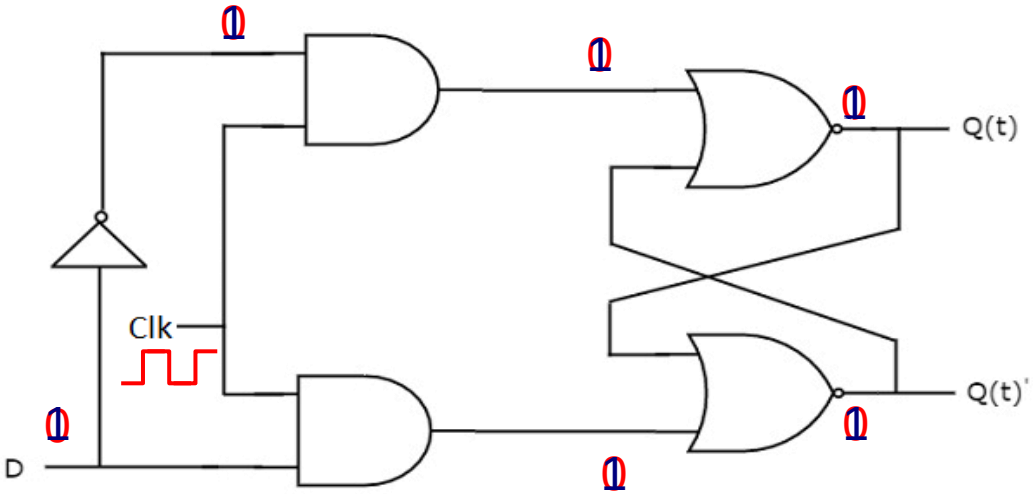
K-map for Q_{t+1}

Q_{t+1}

S \ RQ_t	$R'Q_t'$ (00)	$R'Q_t$ (01)	RQ_t (11)	RQ_t' (10)
S' (0)	0 (0)	1 (1)	0 (3)	0 (2)
S (1)	1 (4)	1 (5)	X (7)	X (6)

$$Q_{t+1} = S + R'Q_t$$

D Flip-flop



State Table

D	Q_{t+1}
0	0
1	1

D Flip-flop

Characteristic Table

Present Input	Present State	Next State	Status
D	Q_t	Q_{t+1}	
0	0	0	Reset
0	1	0	
1	0	1	Set
1	1	1	

Excitation Table

Present State	Next State	Present Input
Q_t	Q_{t+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

By Nilesh Patidar

D Flip-flop

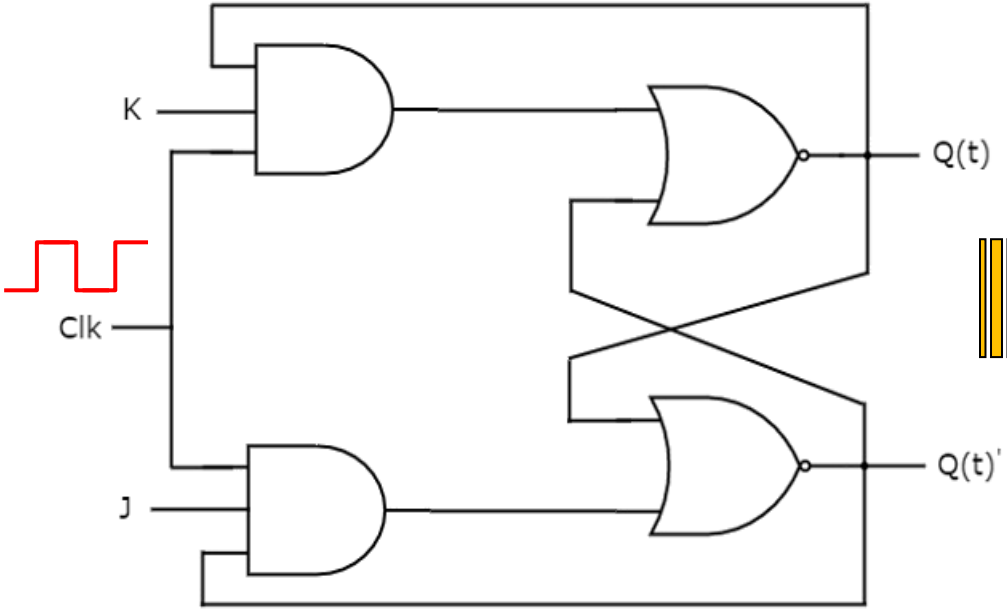
K-map for Q_{t+1}

Q_{t+1} \ Q_t, D	Q_t' (0)	Q_t (1)
D' (0)	0 (0)	0 (1)
D (1)	1 (2)	1 (3)

$Q_{t+1} = D$

JK Flip-flop

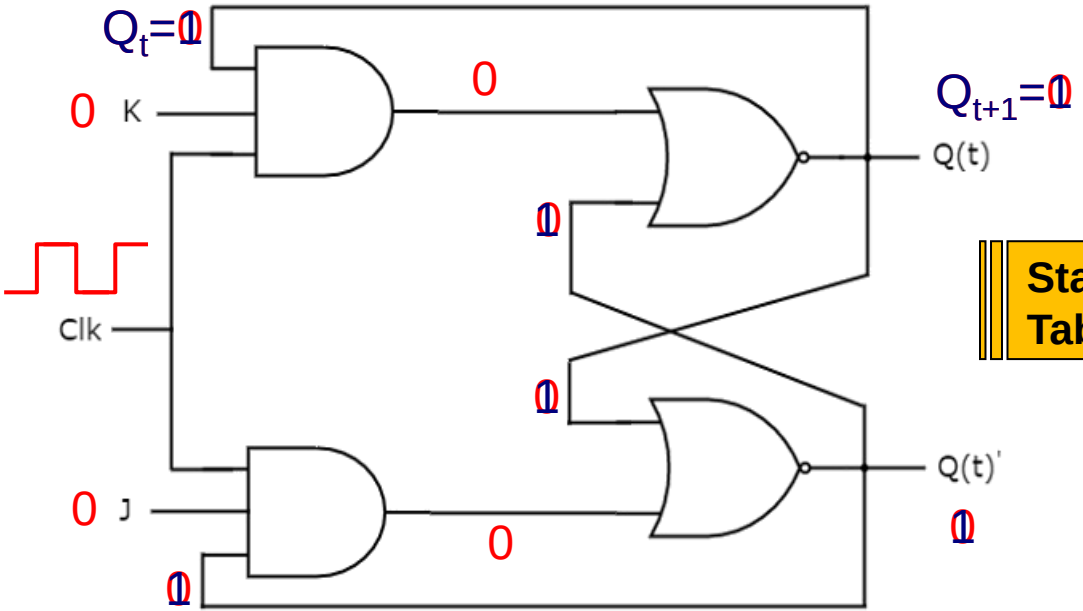
- JK flip-flop is the modified version of SR flip-flop..



State Table

J	K	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	Q_t'

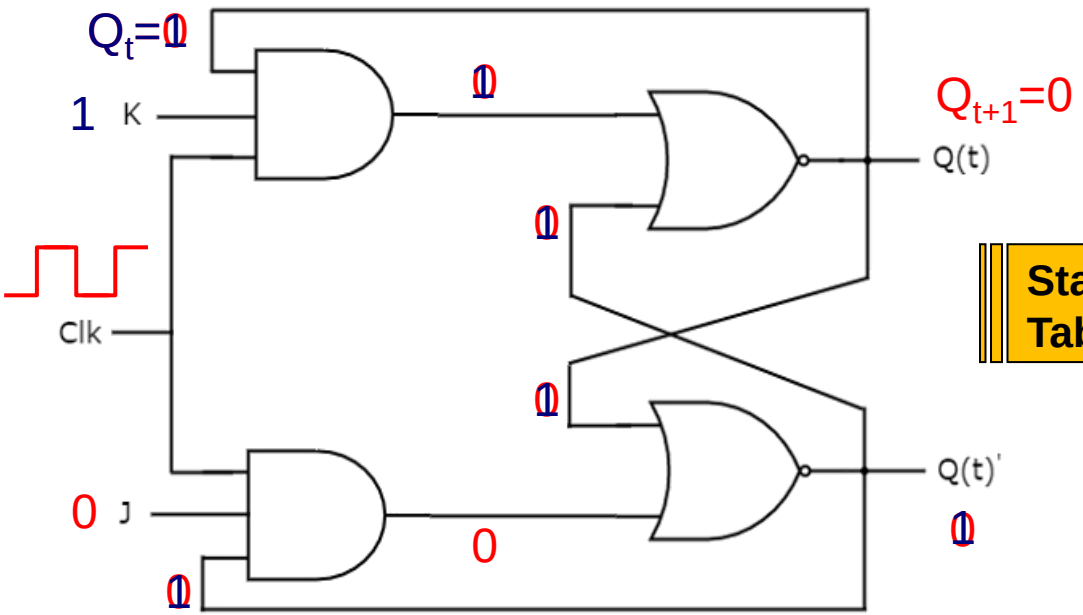
JK Flip-flop



State Table

J	K	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	Q_t'

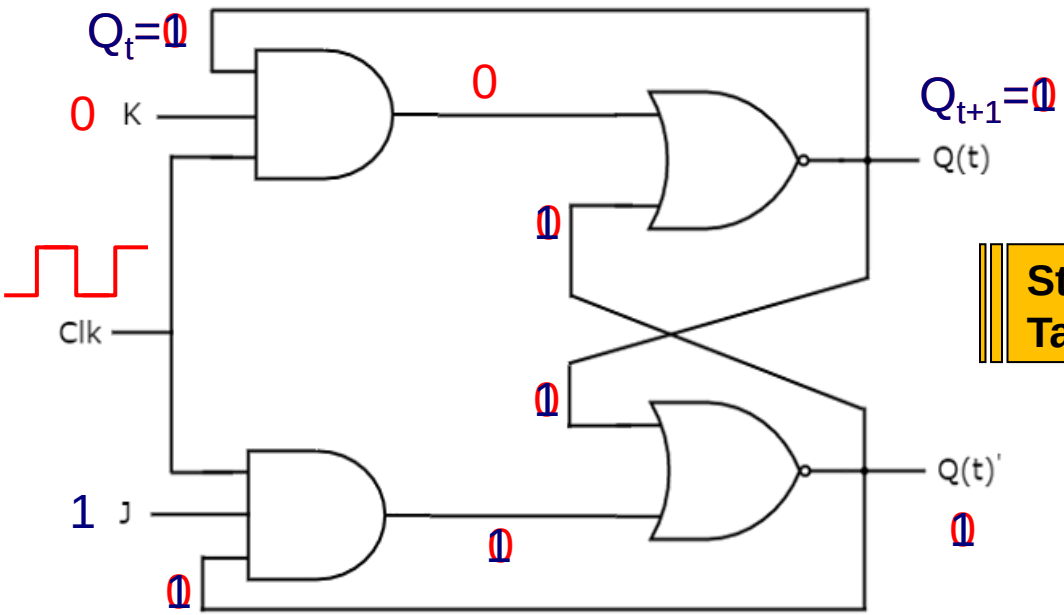
JK Flip-flop



State Table

J	K	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	Q_t'

JK Flip-flop

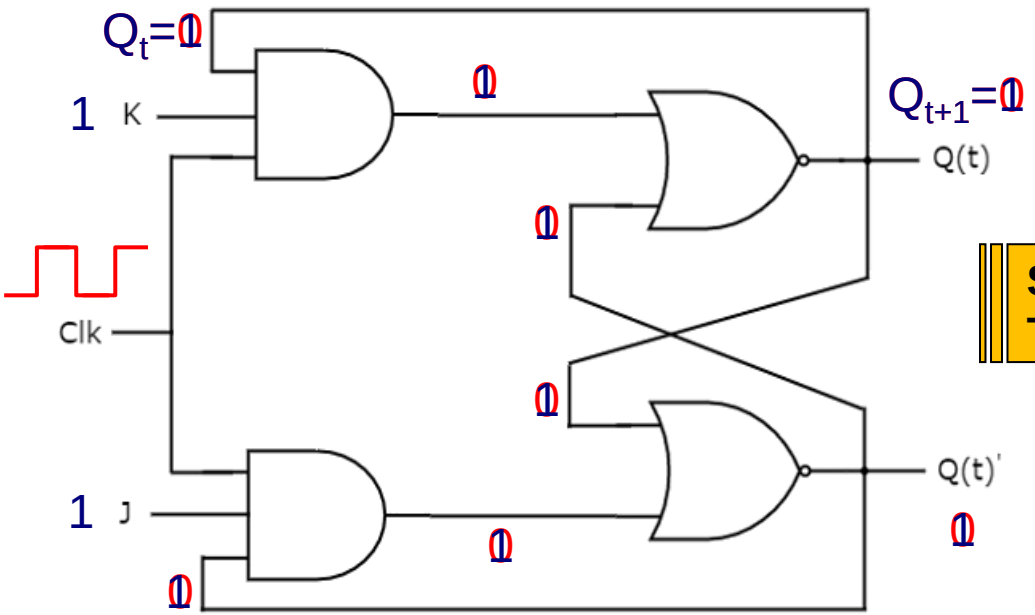


State Table

J	K	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	Q_t'

By Nilesh Patidar

JK Flip-flop



State Table

J	K	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	Q_t'

By Nilesh Patidar

JK Flip-flop

Characteristic Table

Present Input		Present State	Next State	Status
J	K	Q_t	Q_{t+1}	
0	0	0	0	No Change
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	1	Toggle
1	1	1	0	

Excitation Table

Present State	Next State	Present Input	
Q_t	Q_{t+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

By Nilesh Patidar

JK Flip-flop

K-map for Q_{t+1}

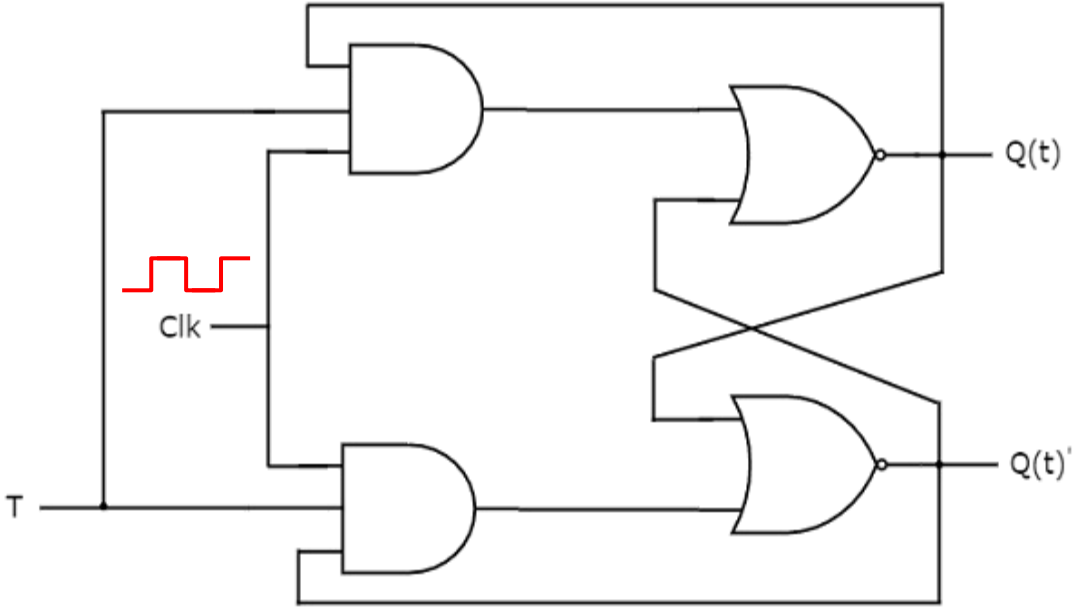
Q_{t+1}

J \ KQ_t	$K'Q_t'$ (00)	$K'Q_t$ (01)	KQ_t (11)	KQ_t' (10)
J' (0)	0 (0)	1 (1)	0 (3)	0 (2)
J (1)	1 (4)	1 (5)	0 (7)	1 (6)

$$Q_{t+1} = JQ_t' + K'Q_t$$

T Flip-flop

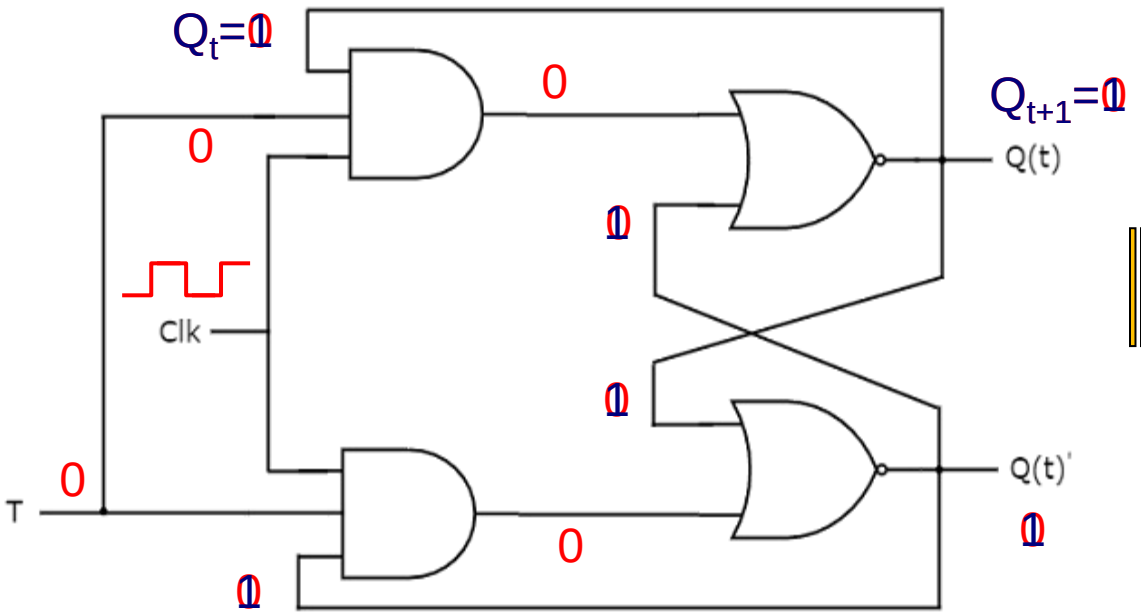
- We can implement T Flip-flop by joining the inputs JK flip-flop.



State Table

T	Q_{t+1}
0	Q_t
1	Q_t'

T Flip-flop

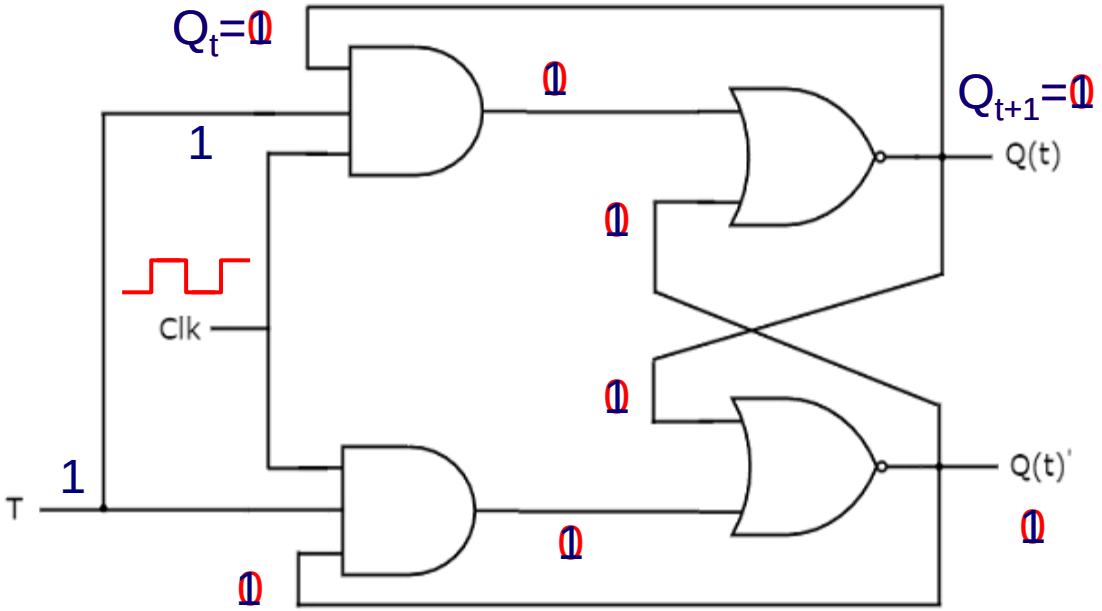


State Table

T	Q_{t+1}
0	Q_t
1	Q_t'

By Nilesh Patidar

T Flip-flop



State Table

T	Q_{t+1}
0	Q_t
1	Q_t'

By Nilesh Patidar

T Flip-flop

Characteristic Table

Present Input	Present State	Next State	Status
T	Q_t	Q_{t+1}	
0	0	0	No Change
0	1	1	
1	0	1	Toggle
1	1	0	

Excitation Table

Present State	Next State	Present Input
Q_t	Q_{t+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

By Nilesh Patidar

T Flip-flop

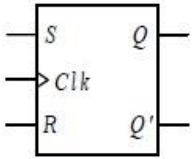
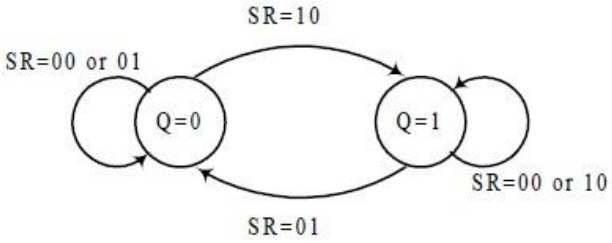
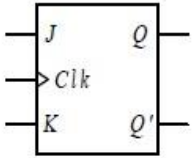
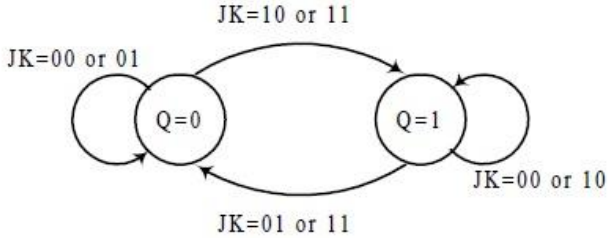
K-map for Q_{t+1}

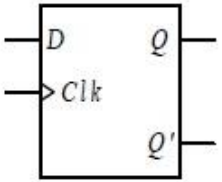
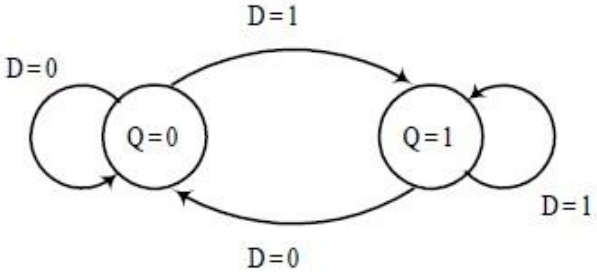
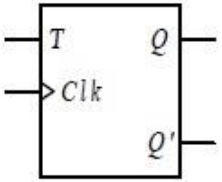
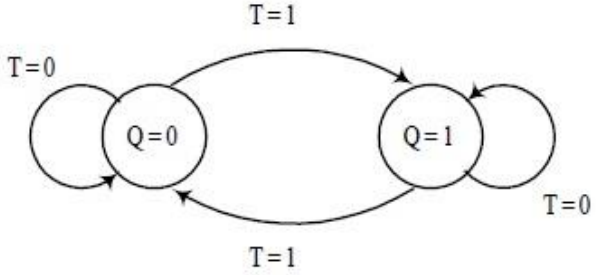
Q_{t+1} (T \ Q_t)	Q_t' (0)	Q_t (1)
T' (0)	0 (0)	1 (1)
T (1)	1 (2)	0 (3)

$$Q_{t+1} = TQ_t' + T'Q_t$$

$$Q_{t+1} = T \oplus Q_t$$

By Nilesh Patidar

Name / Symbol	Characteristic (Truth) Table	State Diagram / Characteristic Equations	Excitation Table
SR  	S R Q Q_{next} 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 ×	 $Q_{next} = S + R'Q$ $SR = 0$	Q Q_{next} S R 0 0 0 × 0 1 1 0 1 0 0 1 1 1 × 0
JK  	J K Q Q_{next} 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	 $Q_{next} = J'K'Q + JK' + JKQ'$ $= J'K'Q + JK'Q + JK'Q' + JKQ'$ $= K'Q(J'+J) + JQ'(K'+K)$ $= K'Q + JQ'$	Q Q_{next} J K 0 0 0 × 0 1 1 × 1 0 × 1 1 1 × 0

D  		 $Q_{next} = D$	
T  		 $Q_{next} = TQ' + T'Q = T \oplus Q$	

D	Q	Q_{next}
0	×	0
1	×	1

Q	Q_{next}	D
0	0	0
0	1	1
1	0	0
1	1	1

T	Q	Q_{next}
0	0	0
0	1	1
1	0	1
1	1	0

Q	Q_{next}	T
0	0	0
0	1	1
1	0	1
1	1	0

Conversion of Flip-Flops

- The **excitation table** for all flip-flops is shown below.

Present State	Next State	SR-FF		D-FF	JK-FF		T-FF
Q_t	Q_{t+1}	S	R	D	J	K	T
0	0	0	X	0	0	X	0
0	1	1	0	1	1	X	1
1	0	0	1	0	X	1	1
1	1	X	0	1	X	0	0

Conversion of Flip-Flops

- We can convert one flip-flop into the remaining three flip-flops by including some additional logic. So, there will be total of twelve **flip-flop conversions**.
- Follow these **steps** for converting one flip-flop to the other.
  - Step1: Consider the **characteristic table** of desired flip-flop.
  - Step2: Fill the excitation values (*inputs)* of given flip-flop for each combination of present state and next state.
  - Step3: Get the **simplified expressions** for each excitation input. If necessary, use K-maps for simplifying.
  - Step4: Draw the **circuit diagram** of desired flip-flop according to the simplified expressions using given flip-flop and necessary logic gates.

Conversion of SR-FF to D-FF

- Step1: Consider the **characteristic table** of desired flip-flop ***(D-FF)***.
- Step2: Fill the excitation values (*inputs)* of given flip-flop ***(SR-FF)*** for each combination of present state and next state.

Present Input	Present State	Next State	Excitation Inputs	
D	**Q_t**	**Q_{t+1}**	**S**	**R**
0	0	0	0	X
0	1	0	0	1
1	0	1	1	0
1	1	1	X	0

Present State	Next State	Excitation Inputs	
Q_t	**Q_{t+1}**	**S**	**R**
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

By Nilesh Patidar

Conversion of SR-FF to D-FF

- Step3: Get the **simplified expressions** for each excitation input. If necessary, use K-maps for simplifying.

Present Input	Present State	Excitation Inputs	
D	**Q_t**	**S**	**R**
0	0	0	X
0	1	0	1
1	0	1	0
1	1	X	0

K-map for S

S \ Q_t / D	Q_t' (0)	Q_t (1)
D' (0)	0 (0)	0 (1)
D (1)	1 (2)	X (3)

S = D

Conversion of SR-FF to D-FF

- Step3: Get the **simplified expressions** for each excitation input. If necessary, use K-maps for simplifying.

Present Input	Present State	Excitation Inputs	
D	**Q_t**	**S**	**R**
0	0	0	X
0	1	0	1
1	0	1	0
1	1	X	0

K-map for R

R: D \ Q_t	Q_t' (0)	Q_t (1)
D' (0)	X (0)	1 (1)
D (1)	0 (2)	0 (3)

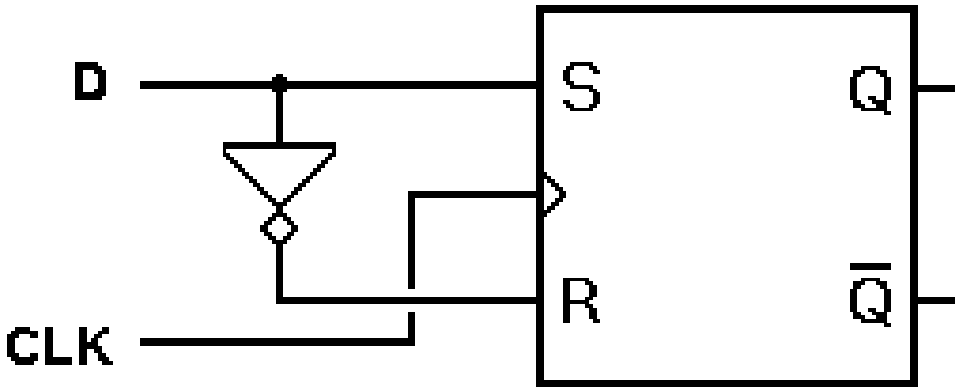
R = D'

Conversion of SR-FF to D-FF

- Step4: Draw the **circuit diagram** of desired flip-flop according to the simplified expressions using given flip-flop and necessary logic gates.

S = D

R = D’



Conversion of D-FF to T-FF

- Step1: Consider the **characteristic table** of desired flip-flop ***(T-FF)***.
- Step2: Fill the excitation values (*inputs)* of given flip-flop ***(D-FF)*** for each combination of present state and next state.

Present Input	Present State	Next State	Excitation Inputs
T	**Q_t**	**Q_{t+1}**	**D**
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

Present State	Next State	Excitation Inputs
Q_t	**Q_{t+1}**	**D**
0	0	0
0	1	1
1	0	0
1	1	1

Conversion of D-FF to T-FF

- Step3: Get the **simplified expressions** for each excitation input. If necessary, use K-maps for simplifying.

Present Input	Present State	Excitation Inputs
T	Q_t	D
0	0	0
0	1	1
1	0	1
1	1	0

K-map for D

D: T \ Q_t	Q_t' (0)	Q_t (1)
T' (0)	0 (0)	1 (1)
T (1)	1 (2)	0 (3)

$$D = TQ_t' + T'Q_t$$

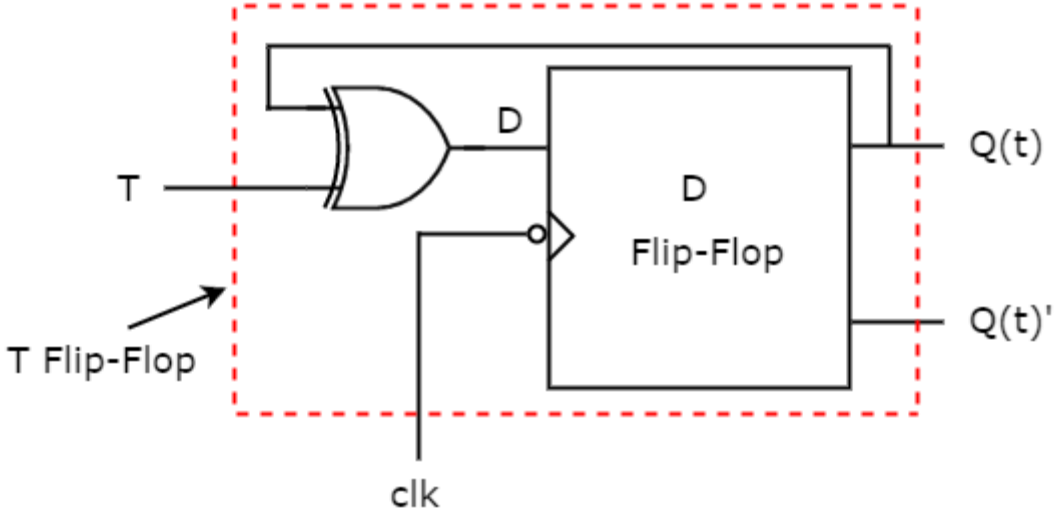
$$D = T \oplus Q_t$$

Conversion of D-FF to T-FF

- Step4: Draw the **circuit diagram** of desired flip-flop according to the simplified expressions using given flip-flop and necessary logic gates.

$$D = TQ_t' + T'Q_t$$

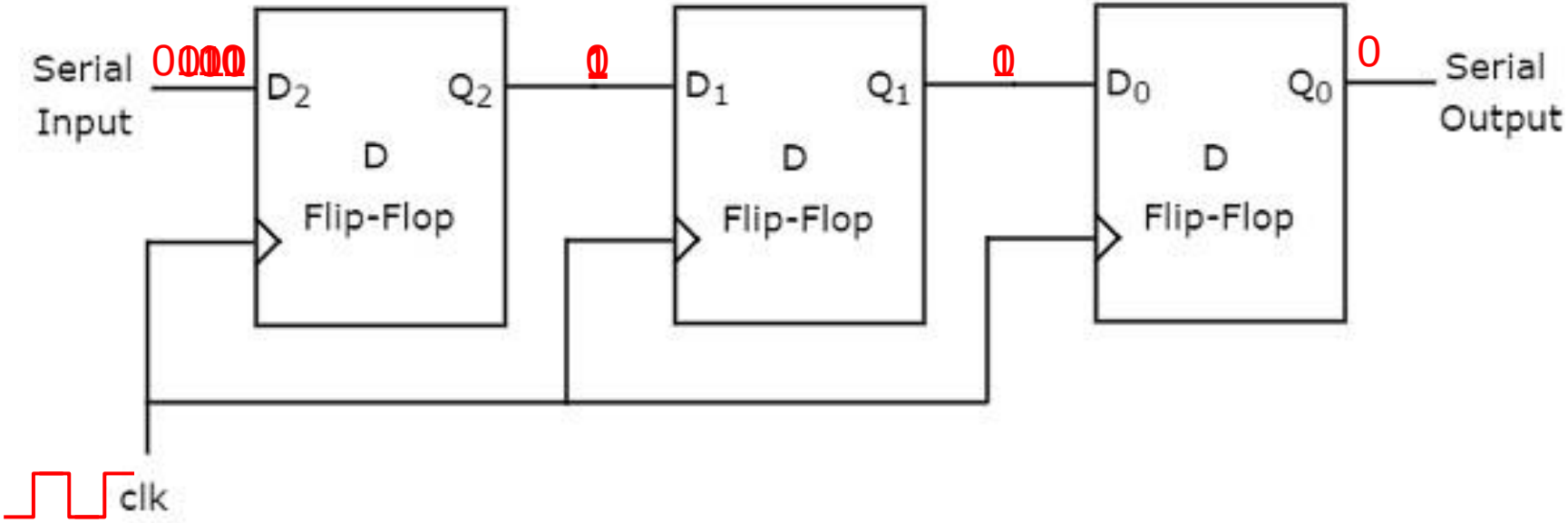
$$D = T \oplus Q_t$$



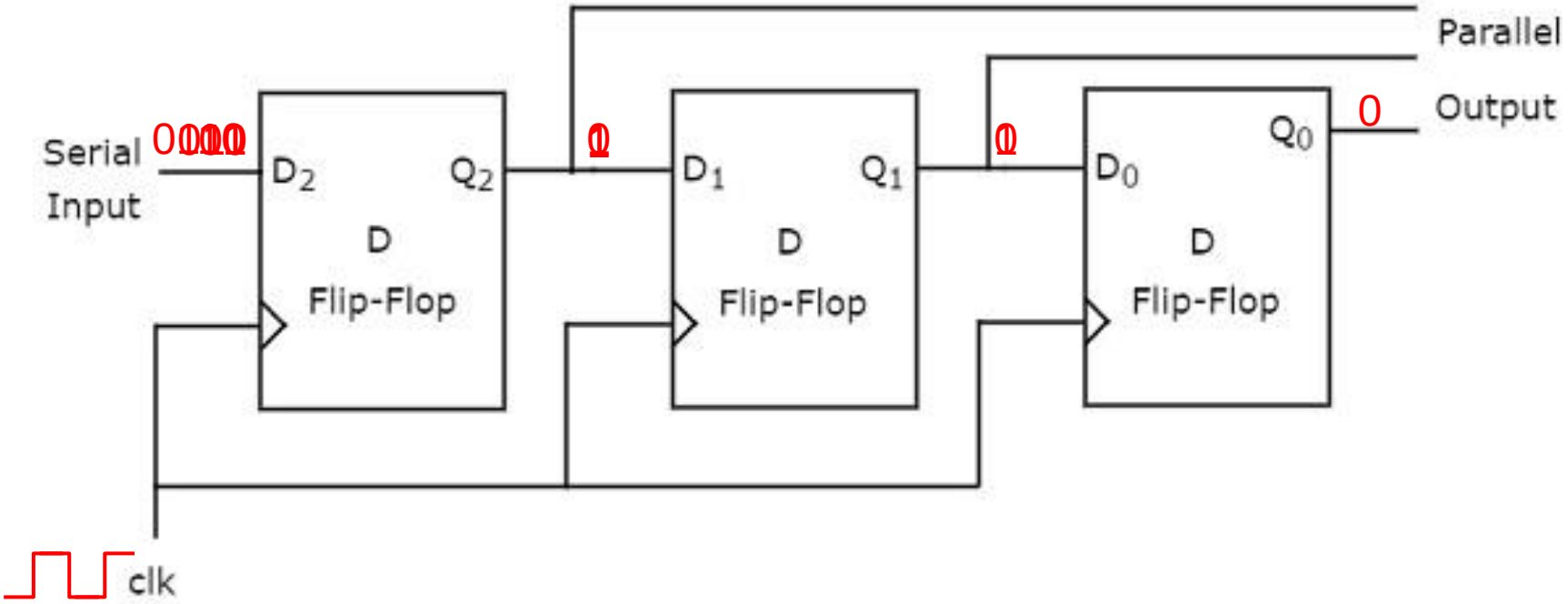
Shift Register

- One flip-flop can store one-bit of information.
- In order to store multiple bits of information, we require multiple flip-flops.
- The group of flip-flops, which are used to hold store the binary data is known as register.
- If the register is capable of shifting bits either towards right hand side or towards left hand side is known as shift register.
- There are the four types of shift registers-
  - Serial In − Serial Out shift register
  - Serial In − Parallel Out shift register
  - Parallel In − Serial Out shift register
  - Parallel In − Parallel Out shift register

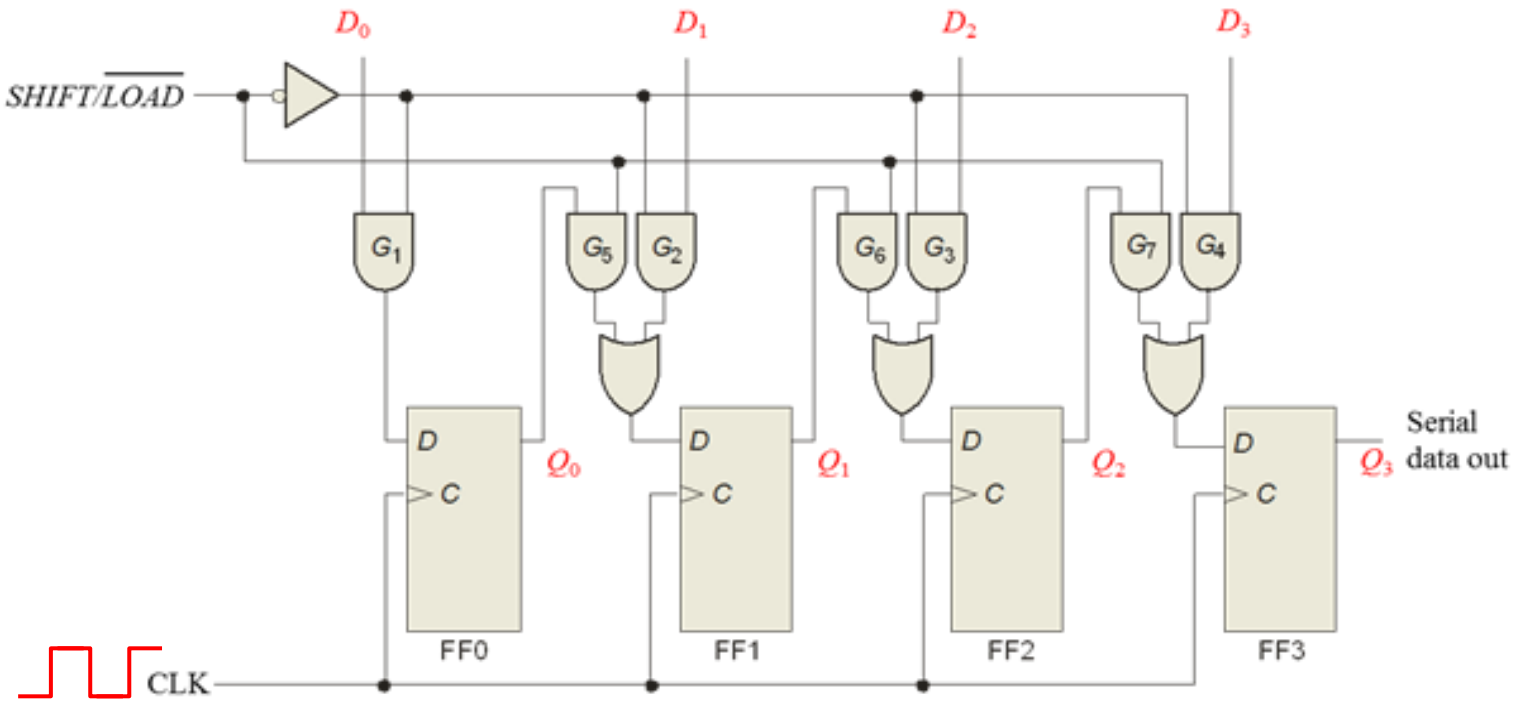
Serial In – Serial Out (SISO)



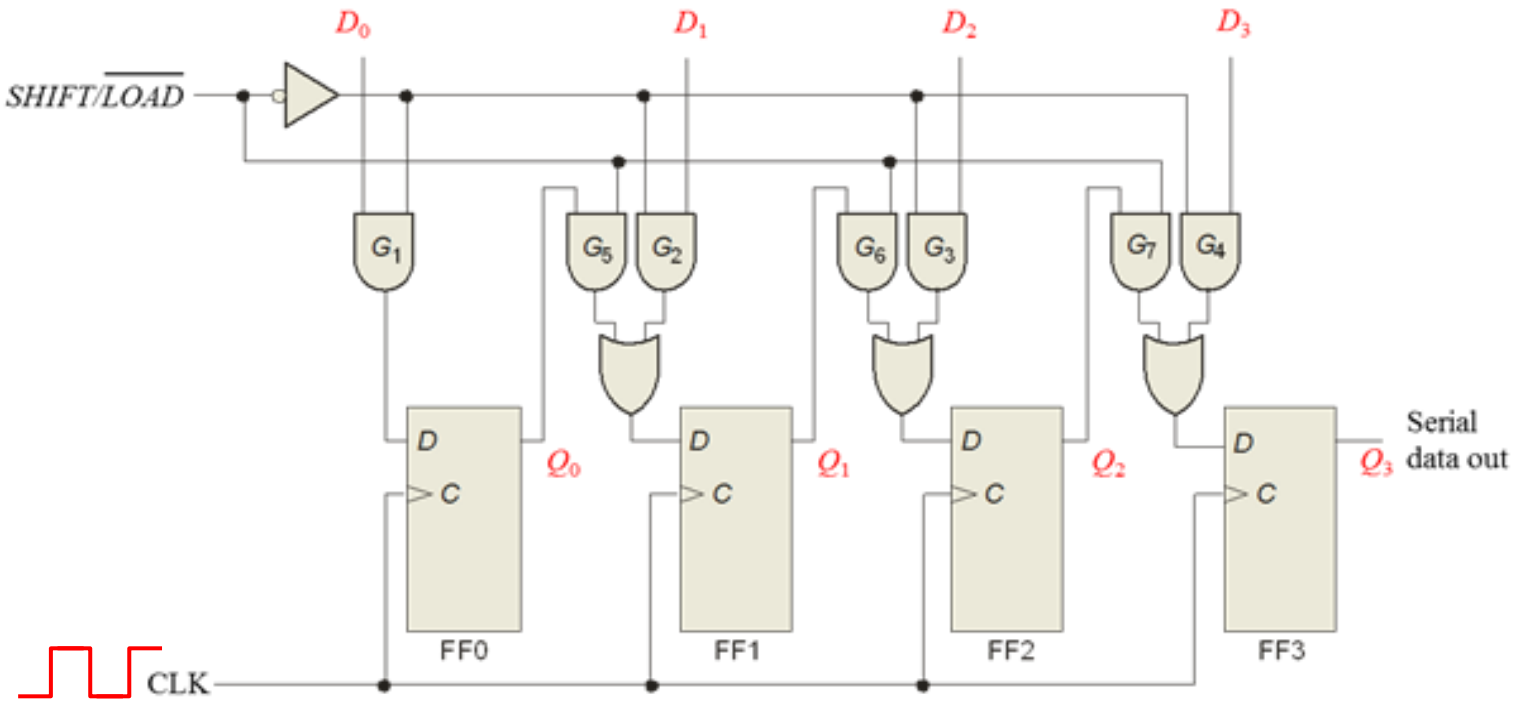
Serial In – Parallel Out (SIPO)



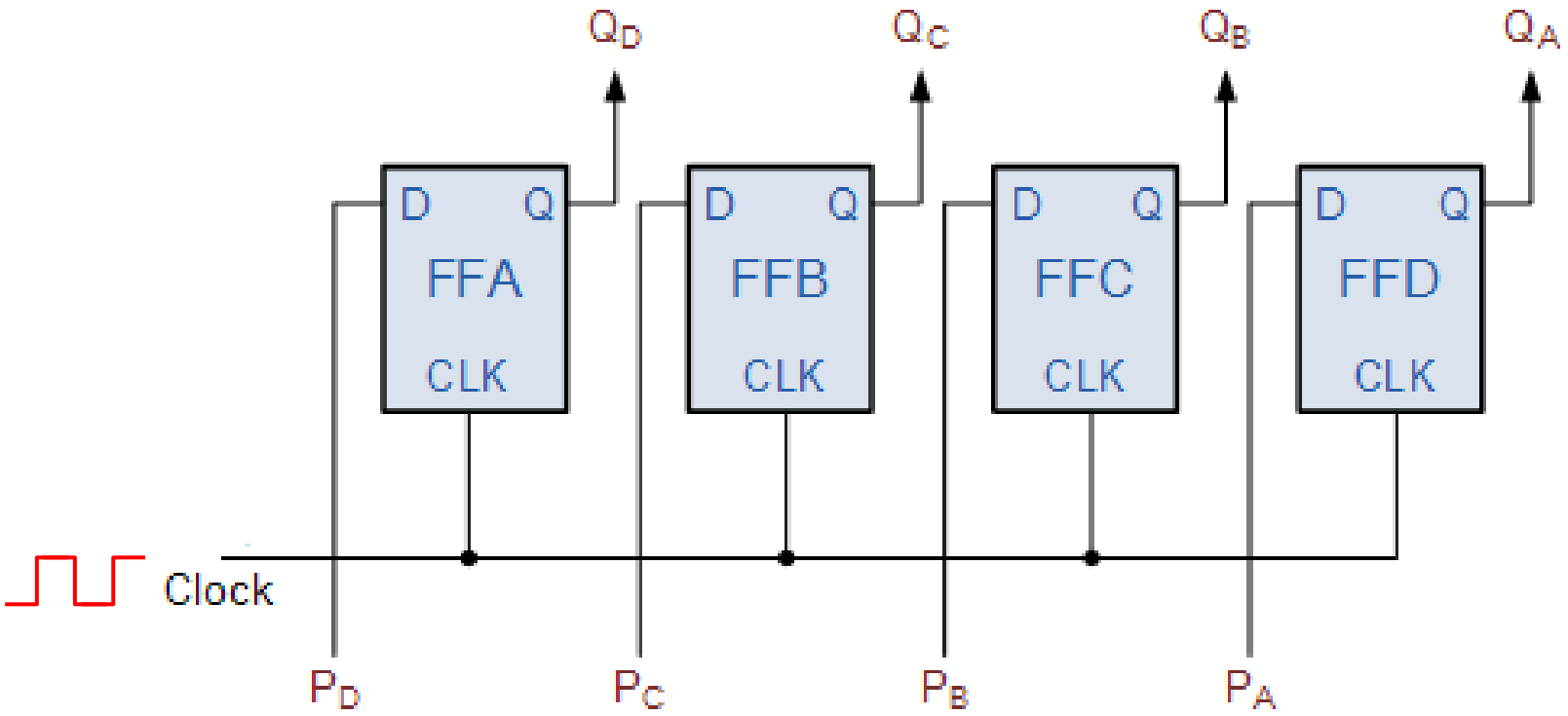
Parallel In – Serial Out (PISO)



Parallel In – Serial Out (PISO)

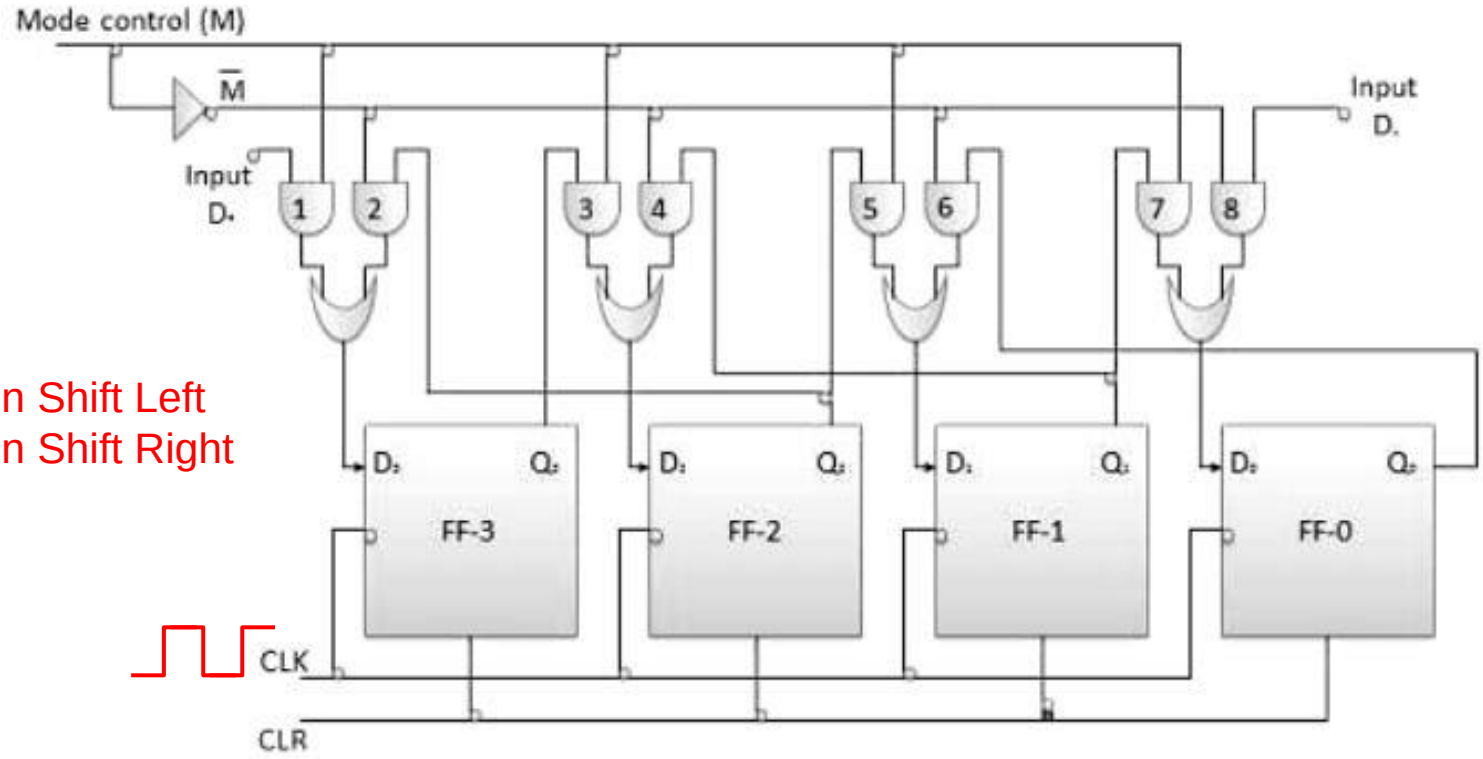


Parallel In – Parallel Out (PIPO)



By Nilesh Patidar

Bidirectional Shift Register



M=0 then Shift Left
M=1 then Shift Right

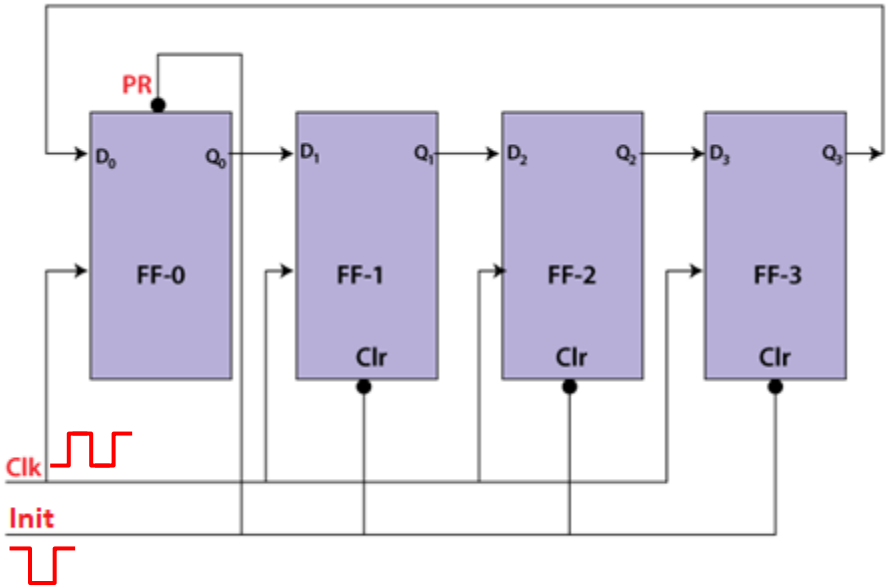
Applications of Shift Register

- Shift register is used as Parallel to serial converter, which converts the parallel data into serial data. It is utilized at the transmitter section after Analog to Digital Converter ADC block.
- Shift register is used as Serial to parallel converter, which converts the serial data into parallel data. It is utilized at the receiver section before Digital to Analog Converter DAC block.
- Shift register along with some additional gates generate the sequence of zeros and ones. Hence, it is used as sequence generator.
- Shift registers are also used as counters.

Ring Counter

- No. of states in Ring counter = No. of flip-flop used

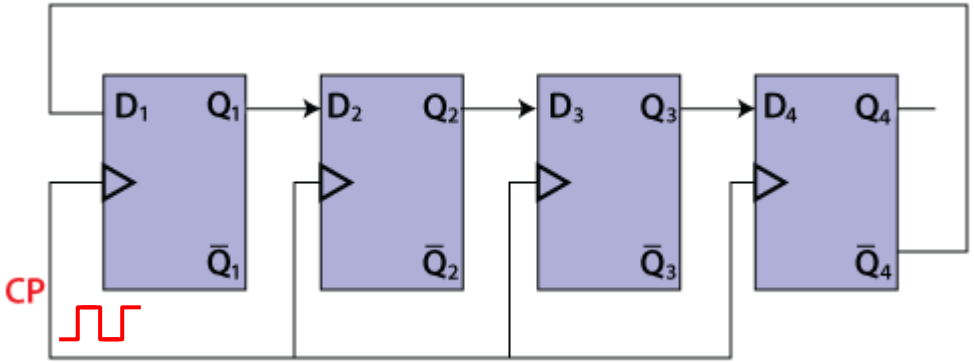
Init	Clk	Q_0	Q_1	Q_2	Q_3
L	X	1	0	0	0
H	↑	0	1	0	0
H	↑	0	0	1	0
H	↑	0	0	0	1
H	↑	1	0	0	0



Johnson Ring Counter

- No. of states in counter = 2*No. of flip-flop used

Clk	Q_1	Q_2	Q_3	Q_4
X	0	0	0	0
↑	1	0	0	0
↑	1	1	0	0
↑	1	1	1	0
↑	1	1	1	1
↑	0	1	1	1
↑	0	0	1	1
↑	0	0	0	1
↑	0	0	0	0



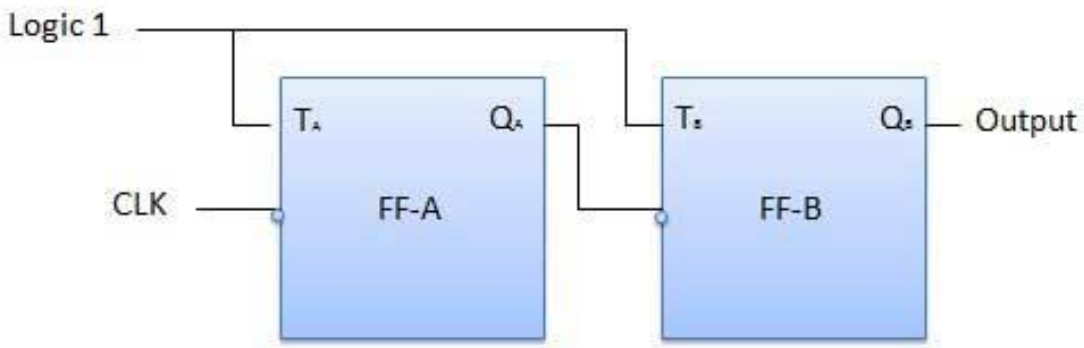
By Nilesh Patidar

Binary Counters

- An ‘N’ bit binary counter consists of ‘N’ flip-flops and it can count the numbers up to $2^N - 1$ (2^N States)
- If the counter counts from 0 to $2^N - 1$, then it is called as binary **up counter**.
- Similarly, if the counter counts down from $2^N - 1$ to 0, then it is called as binary **down counter**.
- There are two types of counters based on the flip-flops that are connected in synchronous or not.
  - Asynchronous counters
  - Synchronous counters

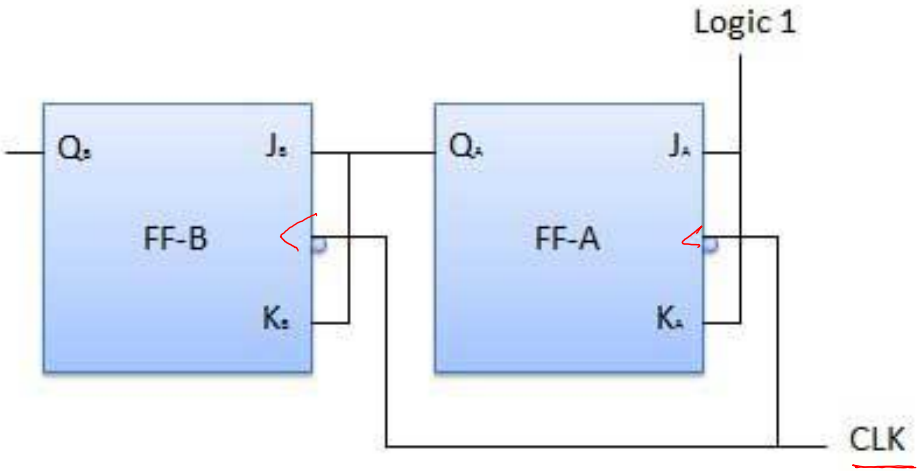
Asynchronous or Ripple Counters

- Asynchronous counters are those whose output is free from the clock signal.
- The first flip-flop is triggered by the clock signal and others successive flip-flops are driven by output of previous flip flops in asynchronous counters.



Synchronous Counters

- Synchronous counters are sometimes called parallel counters as the clock is fed in parallel to all flip-flops.
- The output of all flip-flops are triggered by a single source of clock.



Synchronous v/s Asynchronous

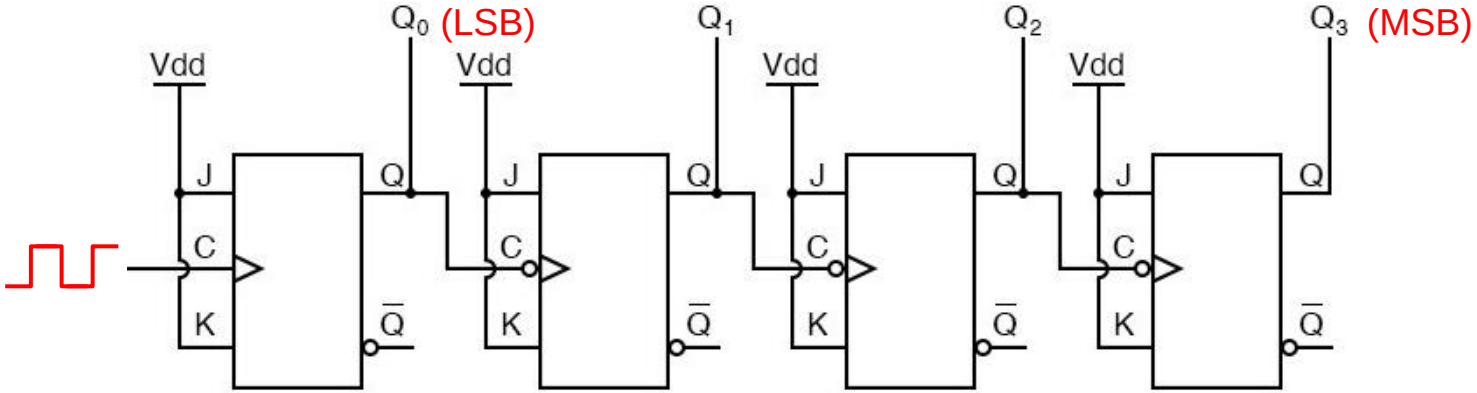
Synchronous

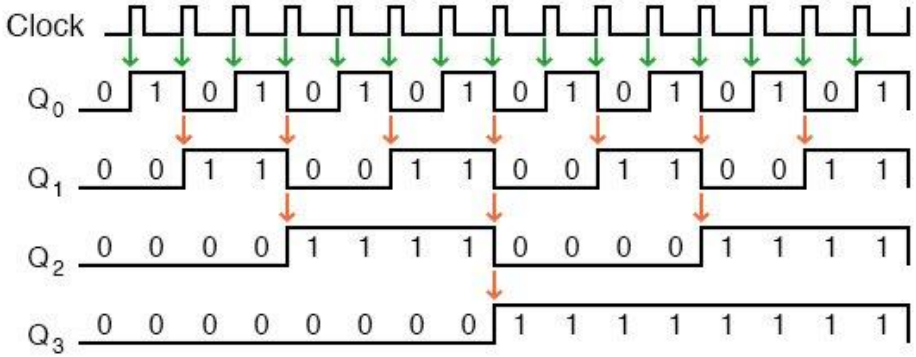
- All flip flops are triggered with same clock simultaneously.
- It is faster in operation
- It does not produce any decoding errors.
- It is also called Parallel Counter.
- The designing as well implementation are complex due to increasing the number of states.
- It will operate in any desired count sequence
- Examples are: Ring counter, Johnson counter.
- Propagation delay is less.

Asynchronous

- Different flip flops are triggered with different clock, not simultaneously.
- It is slower in operation.
- It produces decoding error.
- It is also called Serial Counter.
- The designing as well as implementation is very easy.
- It will operate only in fixed count sequence (UP/DOWN).
- Examples are: Ripple UP counter, Ripple DOWN counter.
- High propagation delay.

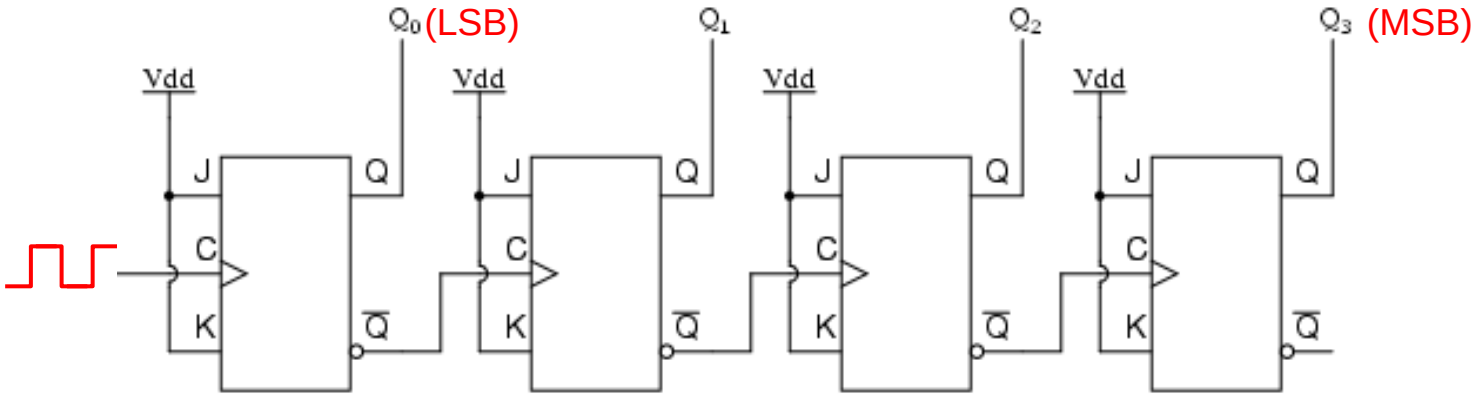
Asynchronous Up Counters (4-bit)

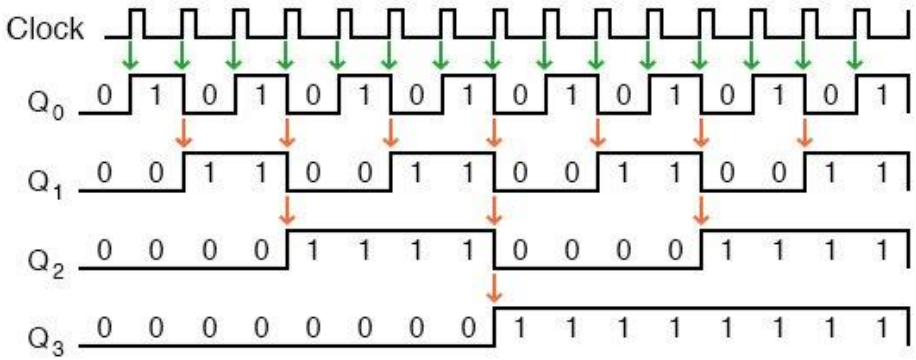




By Nilesh Patidar

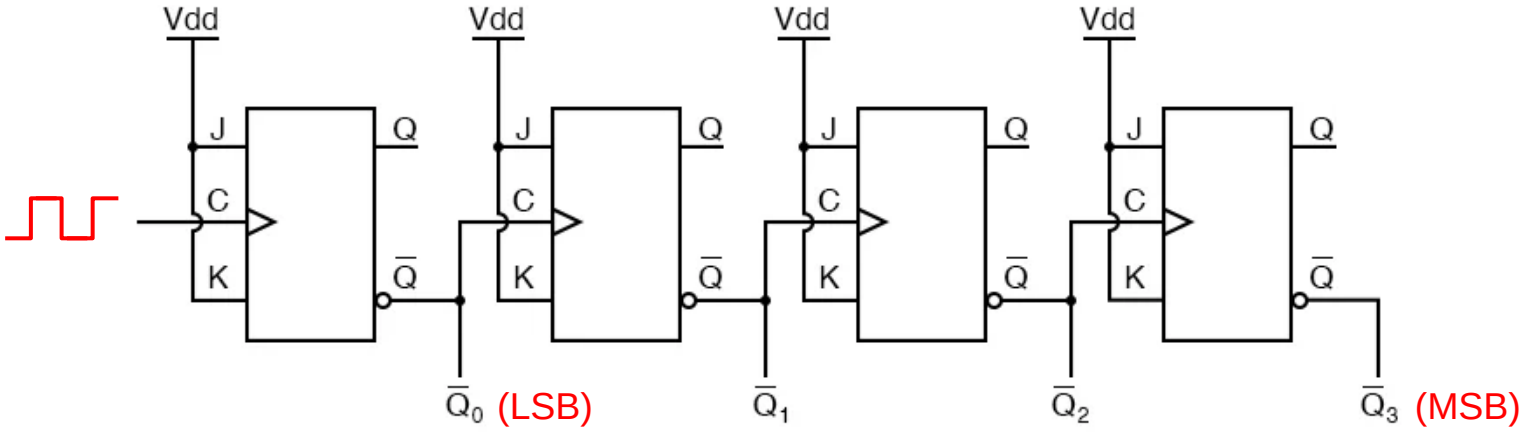
Asynchronous Up Counters (Alternate Configuration)

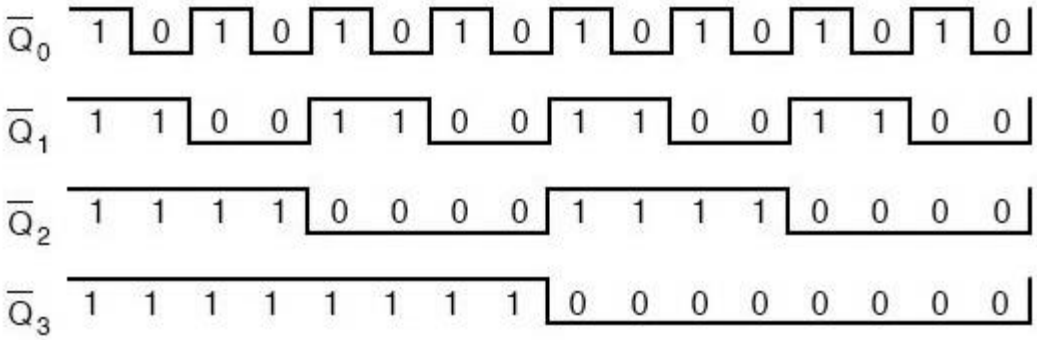




By Nilesh Patidar

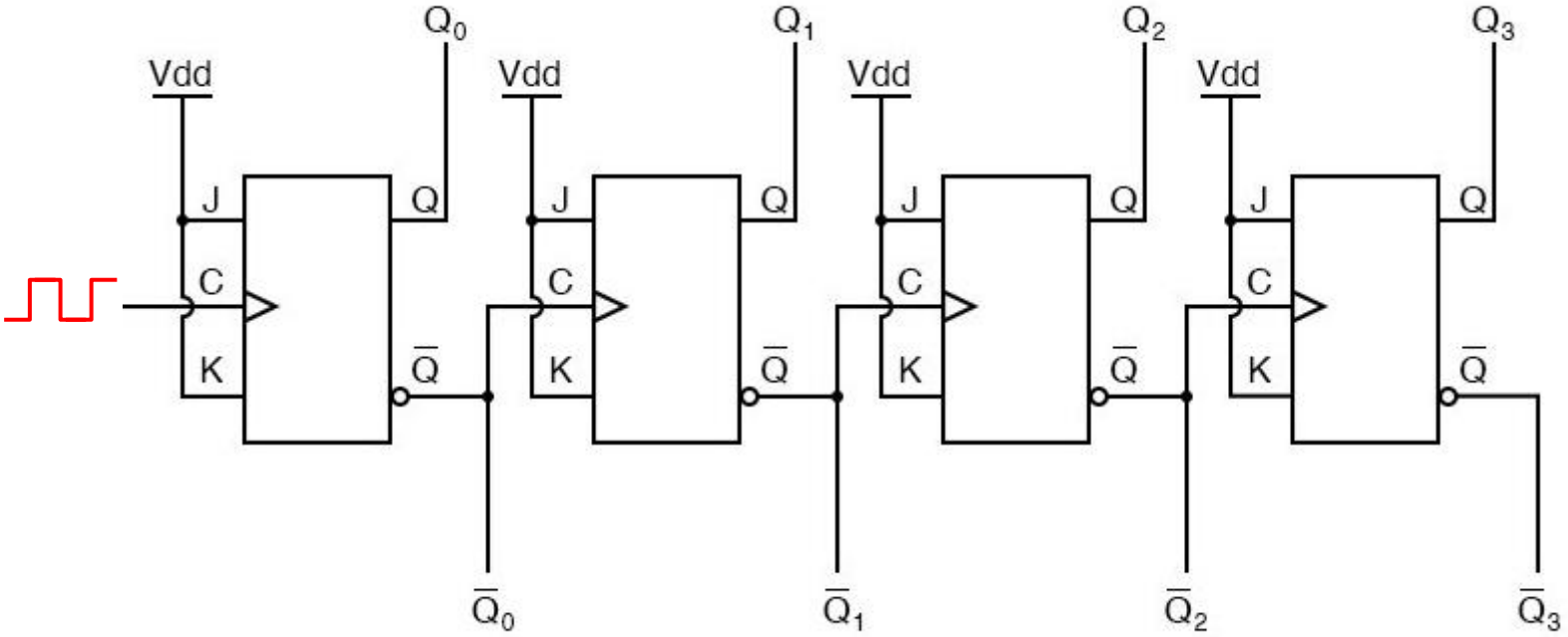
Asynchronous Down Counters (4-bit)





By Nilesh Patidar

Asynchronous Up-Down Counters (4-bit)



By Nilesh Patidar

MOD-*N* Counters

- The number of flip flops used in a ripple counter is depends on the number of states of counter.
- The number of output states of counter is called “Modulus” or “MOD” of the counter. (ex: Mod 4, Mod 2 etc).
- For example, if we have 2 flip flops, the maximum number of outputs of the counter is 4 i.e. 2^2. So it is called as “MOD-4 counter” or “Modulus 4 counter”.
- The 2-bit ripple counter is called as MOD-4 counter and 3-bit ripple counter is called as MOD-8 counter. So, an ***n-bit*** ripple counter is called as **modulo-*N*** counter. Where, MOD number $\boldsymbol{N = 2^n}$

Ripple counter with MOD-N *($N \leq 2^n$)*

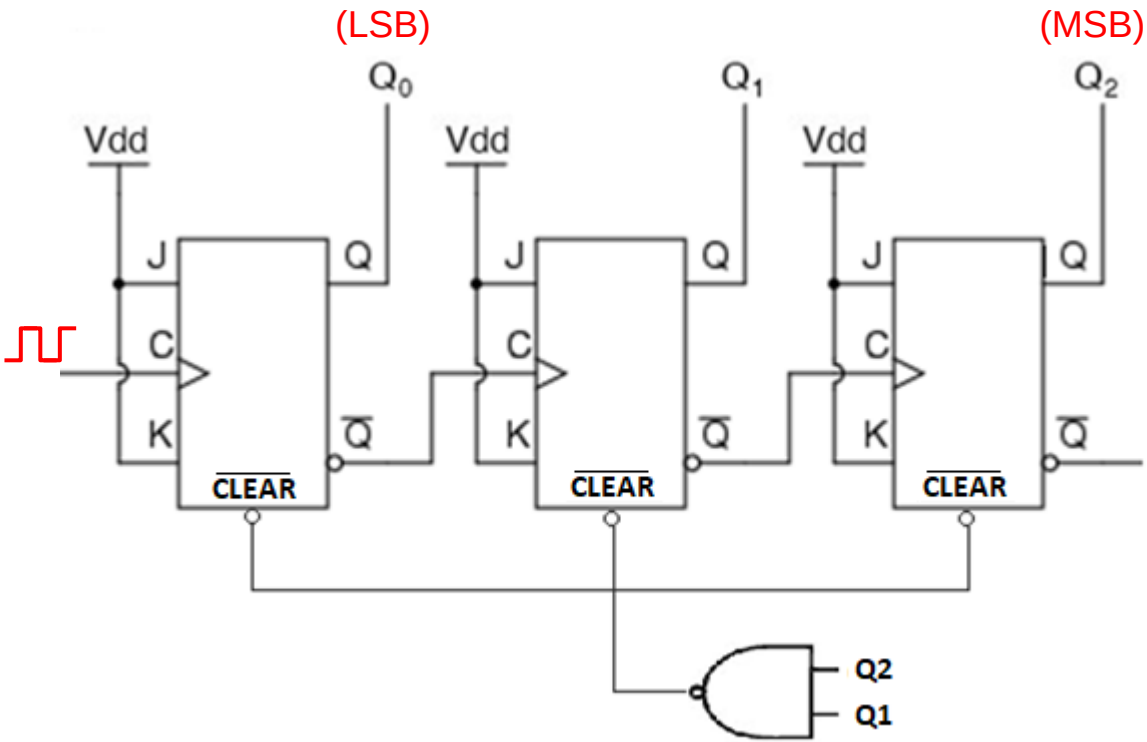
To construct any Mod-N counter, following methods can be used-

1. Find the number of flip-flops (n) required for the desired MOD number (N) using the equation-

$$2^{n-1} \leq N \leq 2^n$$

2. Connect all n flip-flops as a ripple counter.
3. Find the binary number for N.
4. Connect all the flip-flop outputs, for which Q = 1 when the count is N as inputs to NAND gate.
5. Connect the NAND Gates output to clear input of each flip-flop.

Mod-6 counter (Ripple)



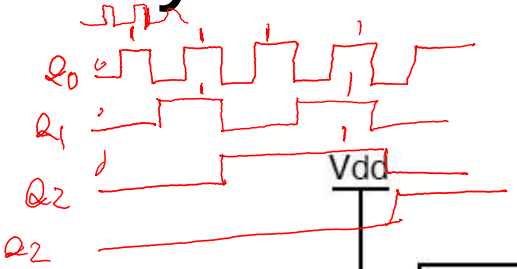
Here N=6
So number flip-flops will be
$2^{n-1} \leq N \leq 2^n$ (so, n=3)

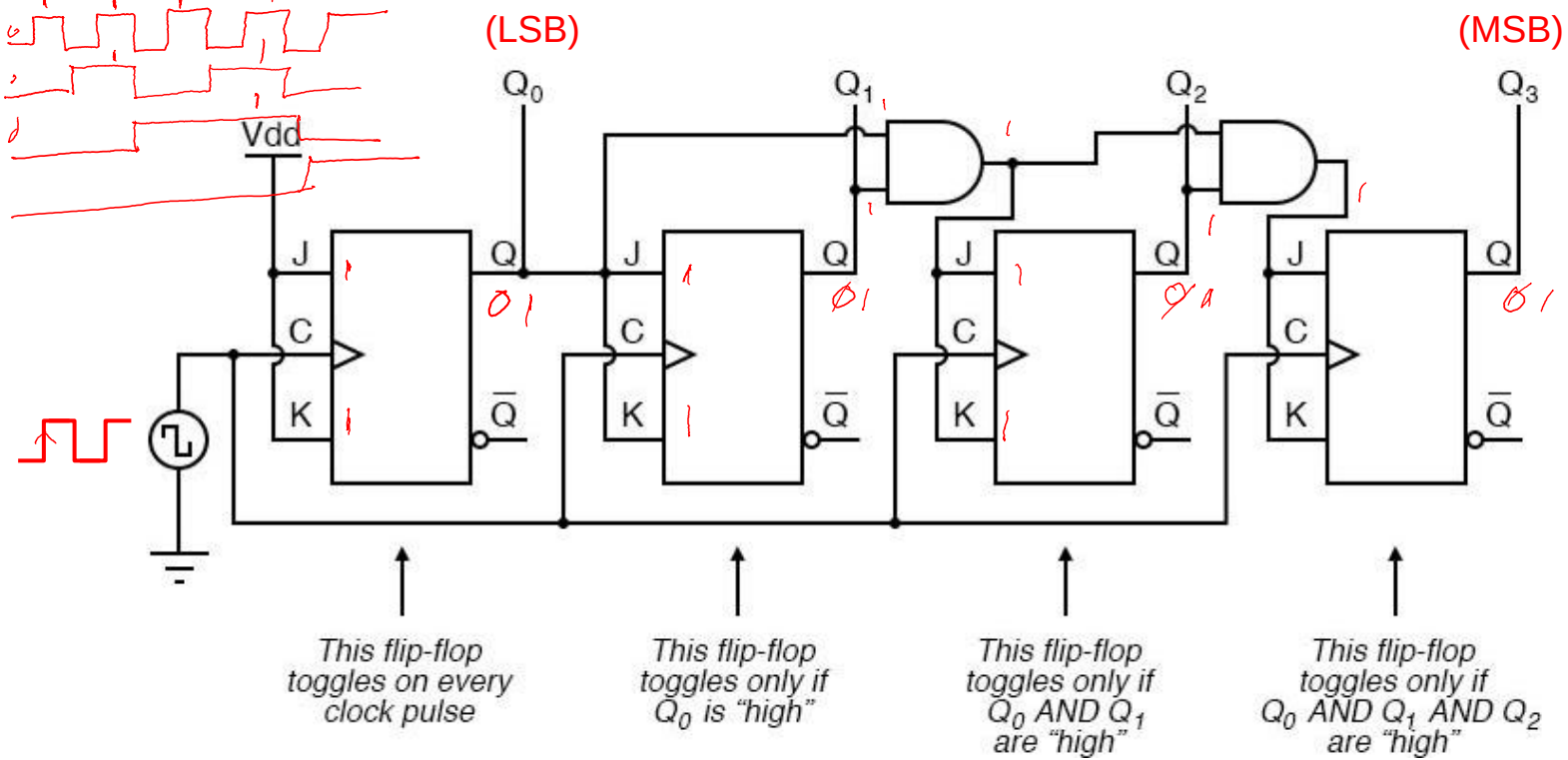
For N=6

Q2	Q1	Q0	
1	1	0	(Binary of 6)

So Q1 and Q2 are inputs of NAND Gate because Q1=Q2=1

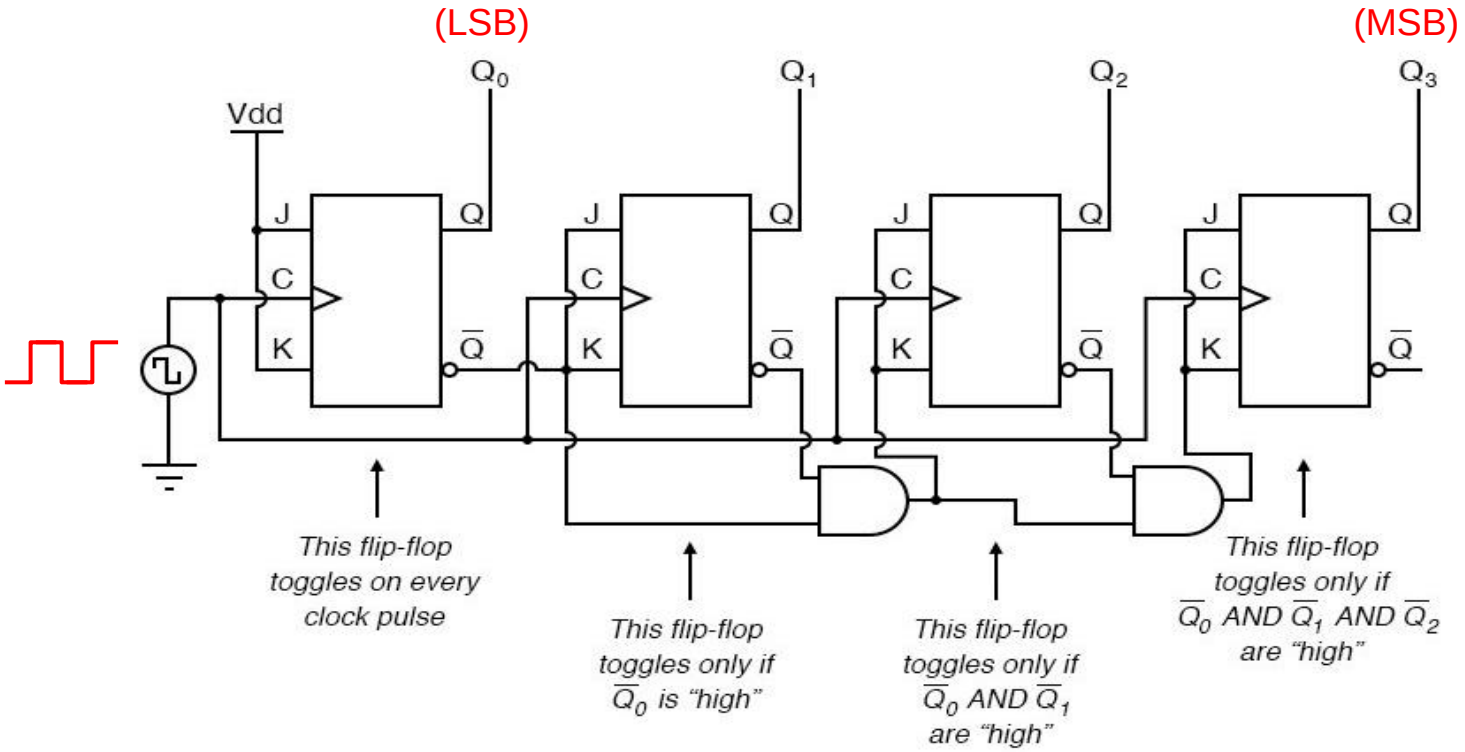
Synchronous Up Counters (4-bit) (0 – 15)





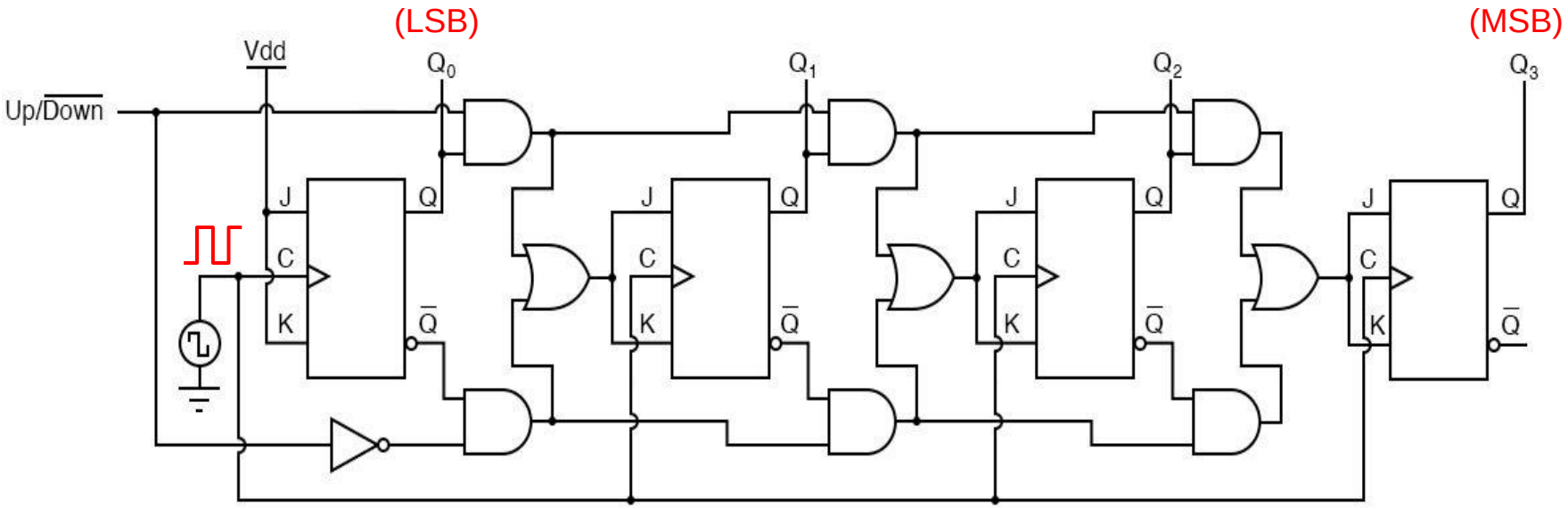
By Nilesh Patidar

Synchronous Down Counters (4-bit)

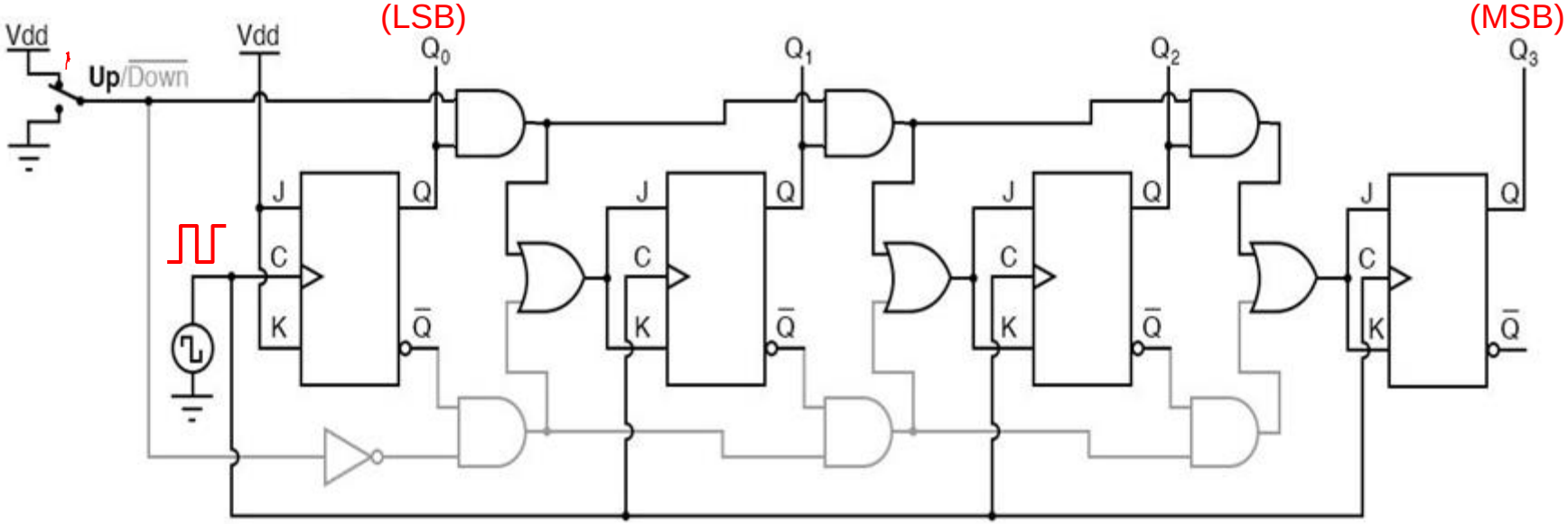


By Nilesh Patidar

Synchronous Up-Down Counters (4-bit)



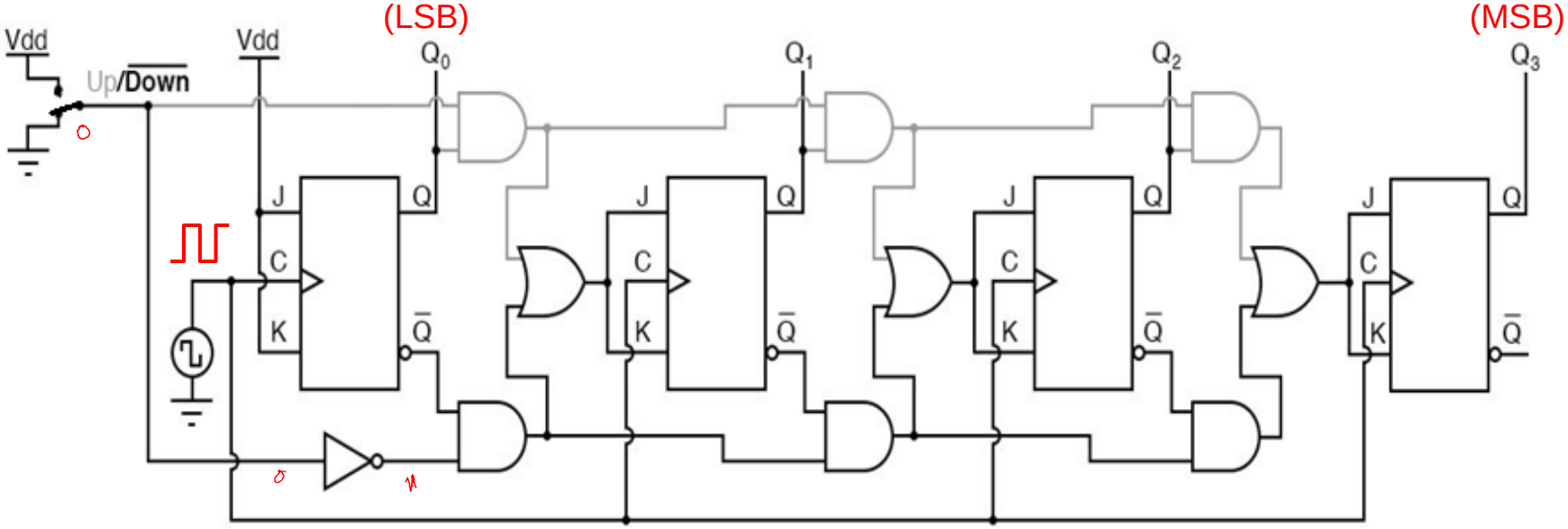
Synchronous Up-Down Counters (4-bit)



When it is acting as Up counter

By Nilesh Patidar

Synchronous Up-Down Counters (4-bit)



When it is acting as Down counter

By Nilesh Patidar

Design of Synchronous Counter

1. Determine the number (n) of FFs needed to support the counting sequence's highest number.

$$2n - 1 \geq \text{Highest number}$$

2. Build a State Transition Diagram. Be sure to include all states.
3. Build a State Table & Excitation Table.
4. Simplify expressions for all inputs for each F/F on K-Maps.
5. Implement the Synchronous Counter/State Machine Circuit.
6. Draw the Timing Diagram (If Needed).

Decade Counter (Synchronous)

- Determine the number (n) of FFs needed to support the counting sequence's highest number. (Count from 0 to 9)

$$2^n - 1 \geq \text{Highest number}$$

$$2^n - 1 \geq 9$$

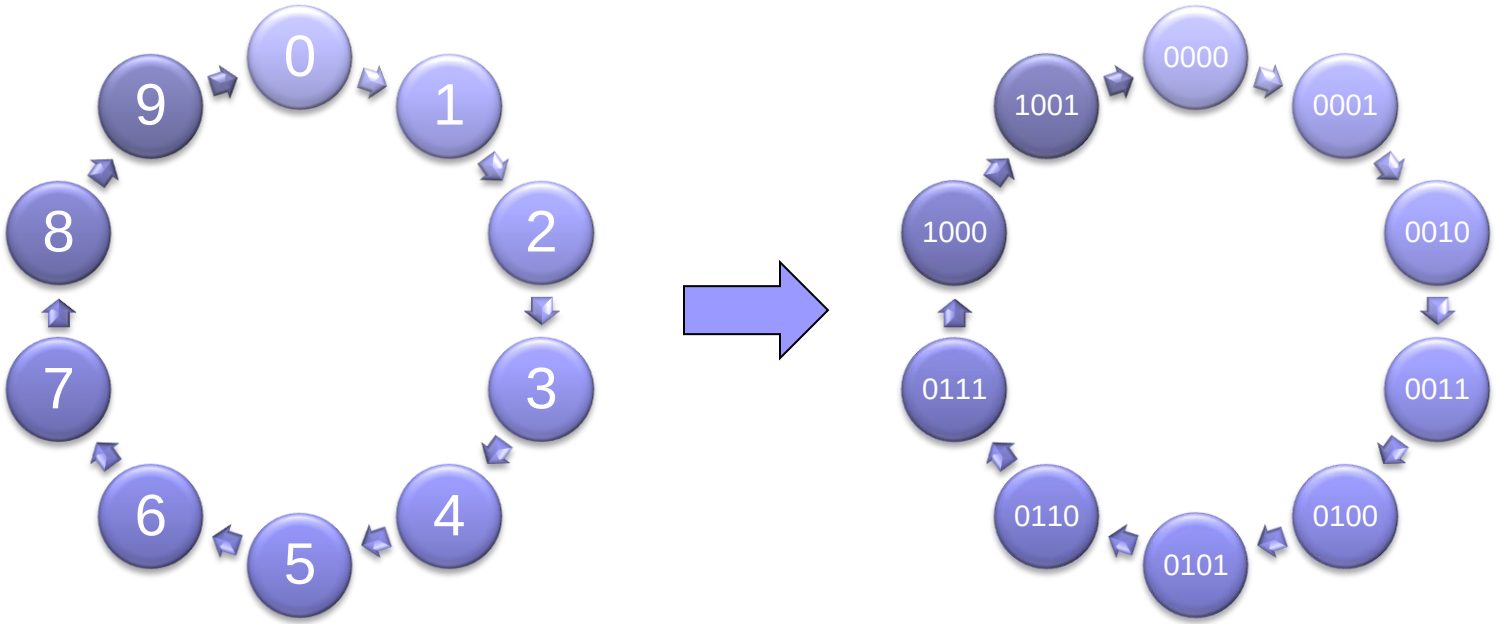
$$2^n \geq 10$$

So, $n = 4$

- It means, there are 4 FFs are required to design Decade Counter (MOD-10 counter that counts from 0 to 9)

Decade Counter (Synchronous)

- Build a State Transition Diagram



By Nilesh Patidar

Decade Counter

PS	NS	SR-F	PS	NS	JK-FF	
Q_t	Q_{t+1}	S	Q_t	Q_{t+1}	J	K
0	0	0	0	0	0	X
0	1	1	0	1	1	X
1	0	0	1	0	X	1
1	1	X	1	1	X	0

Reference Table

- State Table & Excitation Table

	Present State $Q_3\ Q_2\ Q_1\ Q_0$ ($q_3\ q_2\ q_1\ q_0$)	Next State $Q_3\ Q_2\ Q_1\ Q_0$	Excitation Inputs (For JK FF)							
			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
1	0 0 0 1	0 0 1 0	0	X	0	X	1	X	X	1
2	0 0 1 0	0 0 1 1	0	X	0	X	X	0	1	X
3	0 0 1 1	0 1 0 0	0	X	1	X	X	1	X	1
4	0 1 0 0	0 1 0 1	0	X	X	0	0	X	1	X
5	0 1 0 1	0 1 1 0	0	X	X	0	1	X	X	1
6	0 1 1 0	0 1 1 1	0	X	X	0	X	0	1	X
7	0 1 1 1	1 0 0 0	1	X	X	1	X	1	X	1
8	1 0 0 0	1 0 0 1	X	0	0	X	0	X	1	X
9	1 0 0 1	0 0 0 0	X	1	0	X	0	X	X	1

Decade Counter

- Simplify expressions for all inputs for each F/F on K-Maps.

J_3 Q_3Q_2 \ Q_1Q_0	$Q_1'Q_0'$ (00)	$Q_1'Q_0$ (01)	Q_1Q_0 (11)	Q_1Q_0' (10)
$Q_3'Q_2'$ (00)	0 (0)	0 (1)	0 (3)	0 (2)
$Q_3'Q_2$ (01)	0 (4)	0 (5)	1 (7)	0 (6)
Q_3Q_2 (11)	X (12)	X (13)	X (15)	X (14)
Q_3Q_2' (10)	X (8)	X (9)	X (11)	X (10)

$$J_3 = Q_2Q_1Q_0$$

K_3 Q_3Q_2 \ Q_1Q_0	$Q_1'Q_0'$ (00)	$Q_1'Q_0$ (01)	Q_1Q_0 (11)	Q_1Q_0' (10)
$Q_3'Q_2'$ (00)	X (0)	X (1)	X (3)	X (2)
$Q_3'Q_2$ (01)	X (4)	X (5)	X (7)	X (6)
Q_3Q_2 (11)	X (12)	X (13)	X (15)	X (14)
Q_3Q_2' (10)	0 (8)	1 (9)	X (11)	X (10)

$$K_3 = Q_0$$

Decade Counter

- Simplify expressions for all inputs for each F/F on K-Maps.

J_2 Q_3Q_2 \ Q_1Q_0	$Q_1'Q_0'$ (00)	$Q_1'Q_0$ (01)	Q_1Q_0 (11)	Q_1Q_0' (10)
$Q_3'Q_2'$ (00)	0 (0)	0 (1)	1 (3)	0 (2)
$Q_3'Q_2$ (01)	X (4)	X (5)	X (7)	X (6)
Q_3Q_2 (11)	X (12)	X (13)	X (15)	X (14)
Q_3Q_2' (10)	0 (8)	0 (9)	X (11)	X (10)

$J_2 = Q_1Q_0$

K_2 Q_3Q_2 \ Q_1Q_0	$Q_1'Q_0'$ (00)	$Q_1'Q_0$ (01)	Q_1Q_0 (11)	Q_1Q_0' (10)
$Q_3'Q_2'$ (00)	X (0)	X (1)	X (3)	X (2)
$Q_3'Q_2$ (01)	0 (4)	0 (5)	1 (7)	0 (6)
Q_3Q_2 (11)	X (12)	X (13)	X (15)	X (14)
Q_3Q_2' (10)	X (8)	X (9)	X (11)	X (10)

$K_2 = Q_1Q_0$

Decade Counter

- Simplify expressions for all inputs for each F/F on K-Maps.

J_1 Q_3Q_2 \ Q_1Q_0	$Q_1'Q_0'$ (00)	$Q_1'Q_0$ (01)	Q_1Q_0 (11)	Q_1Q_0' (10)
$Q_3'Q_2'$ (00)	0 (0)	1 (1)	X (3)	X (2)
$Q_3'Q_2$ (01)	0 (4)	1 (5)	X (7)	X (6)
Q_3Q_2 (11)	X (12)	X (13)	X (15)	X (14)
Q_3Q_2' (10)	0 (8)	0 (9)	X (11)	X (10)

$J_1 = Q_3'Q_0$

K_1 Q_3Q_2 \ Q_1Q_0	$Q_1'Q_0'$ (00)	$Q_1'Q_0$ (01)	Q_1Q_0 (11)	Q_1Q_0' (10)
$Q_3'Q_2'$ (00)	X (0)	X (1)	1 (3)	0 (2)
$Q_3'Q_2$ (01)	X (4)	X (5)	1 (7)	0 (6)
Q_3Q_2 (11)	X (12)	X (13)	X (15)	X (14)
Q_3Q_2' (10)	X (8)	X (9)	X (11)	X (10)

$K_1 = Q_0$

Decade Counter

- Simplify expressions for all inputs for each F/F on K-Maps.

J_0 Q_3Q_2 \ Q_1Q_0	$Q_1'Q_0'$ (00)	$Q_1'Q_0$ (01)	Q_1Q_0 (11)	Q_1Q_0' (10)
$Q_3'Q_2'$ (00)	1 0	X 1	X 3	1 2
$Q_3'Q_2$ (01)	1 4	X 5	X 7	1 6
Q_3Q_2 (11)	X 12	X 13	X 15	X 14
Q_3Q_2' (10)	1 8	X 9	X 11	X 10

$J_0 = 1$

K_0 Q_3Q_2 \ Q_1Q_0	$Q_1'Q_0'$ (00)	$Q_1'Q_0$ (01)	Q_1Q_0 (11)	Q_1Q_0' (10)
$Q_3'Q_2'$ (00)	X 0	1 1	1 3	X 2
$Q_3'Q_2$ (01)	X 4	1 5	1 7	X 6
Q_3Q_2 (11)	X 12	X 13	X 15	X 14
Q_3Q_2' (10)	X 8	1 9	X 11	X 10

$K_0 = 1$

Decade Counter (Synchronous)

- Implement the Synchronous Counter/State Machine Circuit.

$J_3 = Q_2Q_1Q_0$

$K_3 = Q_0$

$J_2 = Q_1Q_0$

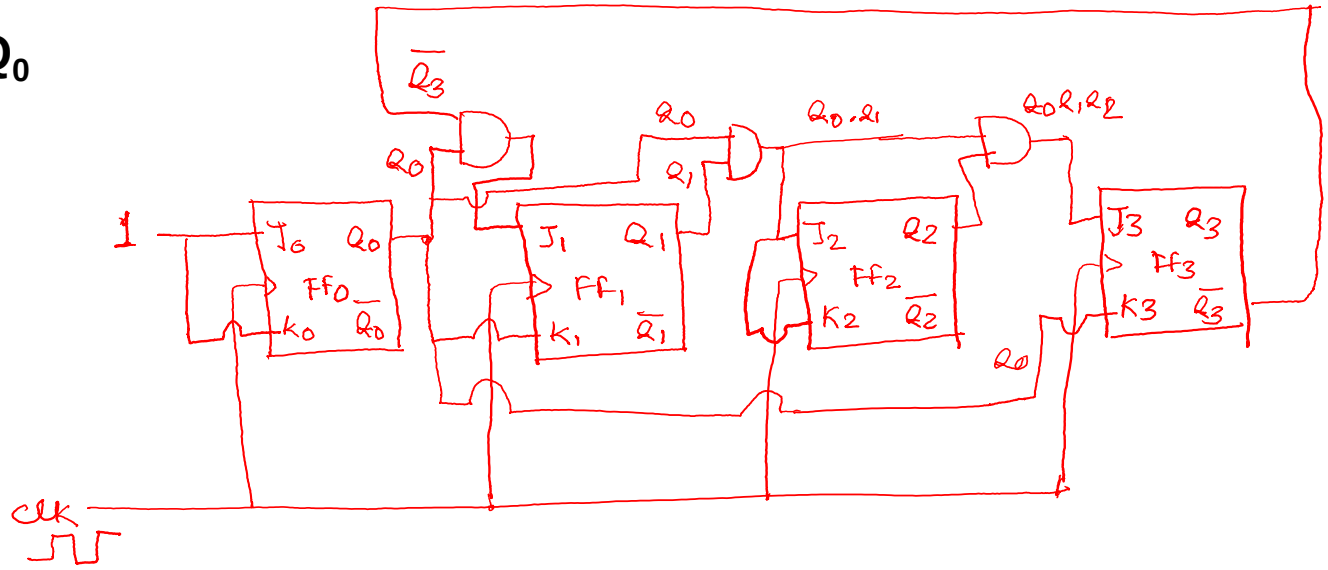
$K_2 = Q_1Q_0$

$J_1 = Q_3'Q_0$

$K_1 = Q_0$

$J_0 = 1$

$K_0 = 1$



Decade counter

Applications of counters

- Frequency counters
- Digital clock
- Time measurement
- A to D converter
- Frequency divider circuits
- Digital triangular wave generator
- Ramp signal generator

Thank You

By Nilesh Patidar