

S3 LCD MOD 4

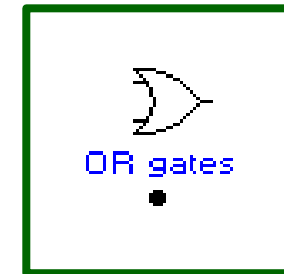
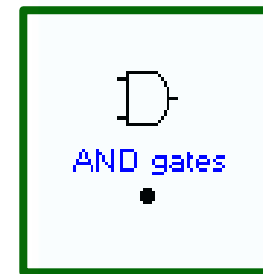
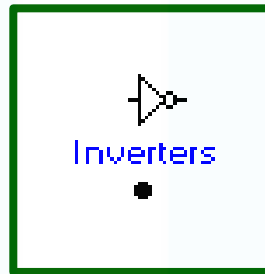
PREPARED BY
ANJALI RAJAN
AP/ECE
IESCE

Logic circuits are classified into two groups:

Combinational logic circuits

Logic gates make decisions

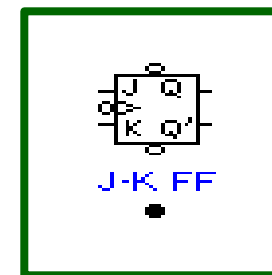
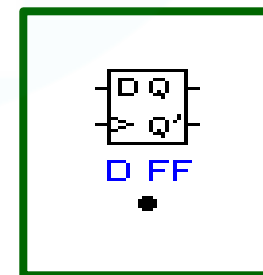
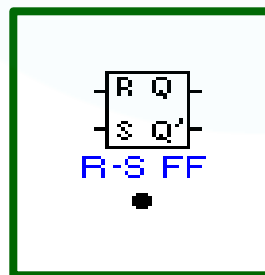
Basic building blocks include Gates:



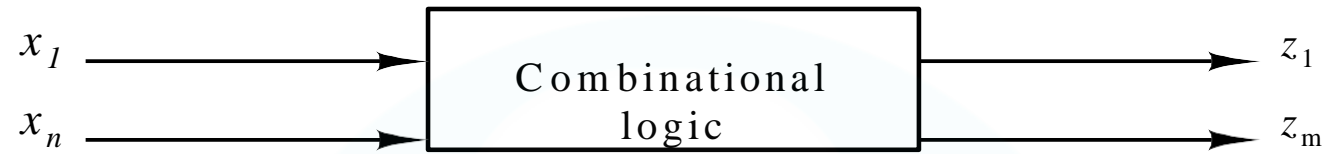
Sequential logic circuits

Flip Flops have memory

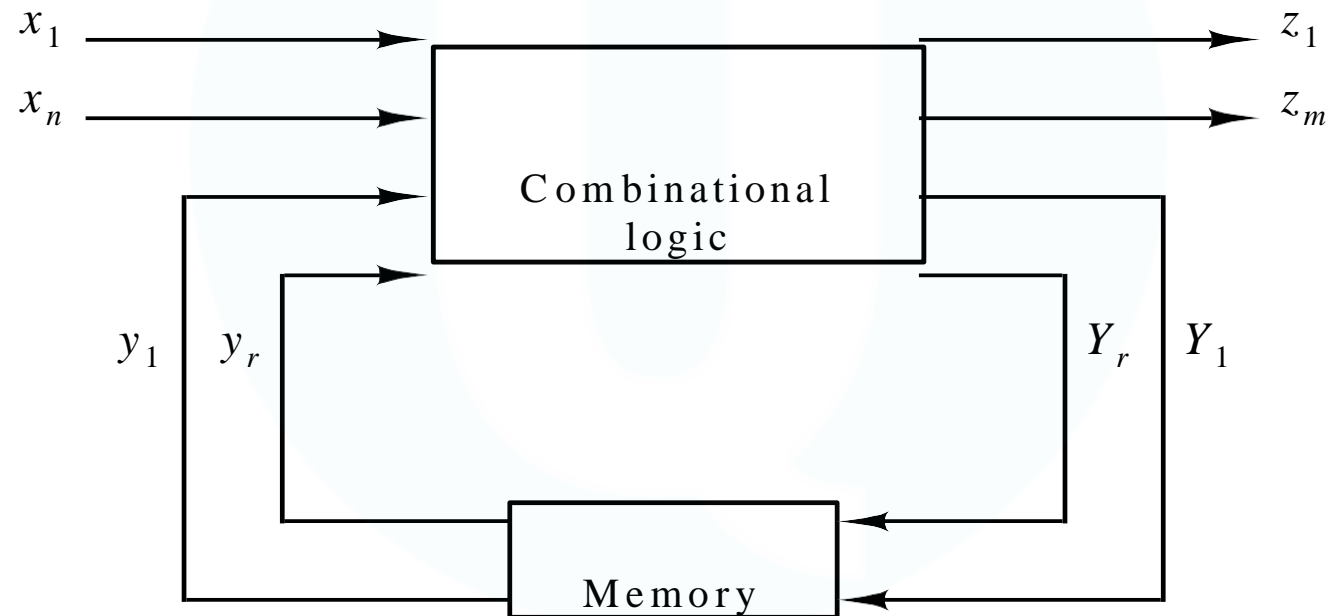
Basic building blocks include FLIP-FLOPS:



Combinational Circuits Vs Sequential Circuits



(a)



(b)

Combinational Circuits Vs Sequential Circuits

Combinational Circuits

- ✓ In combinational circuits, the output variables at any instant of time are dependent only on the present input variables.
- ✓ Memory unit is not required in combinational circuits.

Sequential Circuits

- ✓ In sequential circuits, the output variables at any instant of time are dependent not only on the present input variables but also past output variables.
- ✓ Memory unit is required to store the past history.

Combinational Circuits Vs Sequential Circuits

Combinational Circuits

- ✓ Combinational circuits are faster because the delay between the input and output is due to propagation delay of gates only.
- ✓ Combinational circuits are easy to design.

Sequential Circuits

- ✓ Sequential circuits are slower than combinational circuits.
- ✓ Sequential circuits are comparatively harder to design.

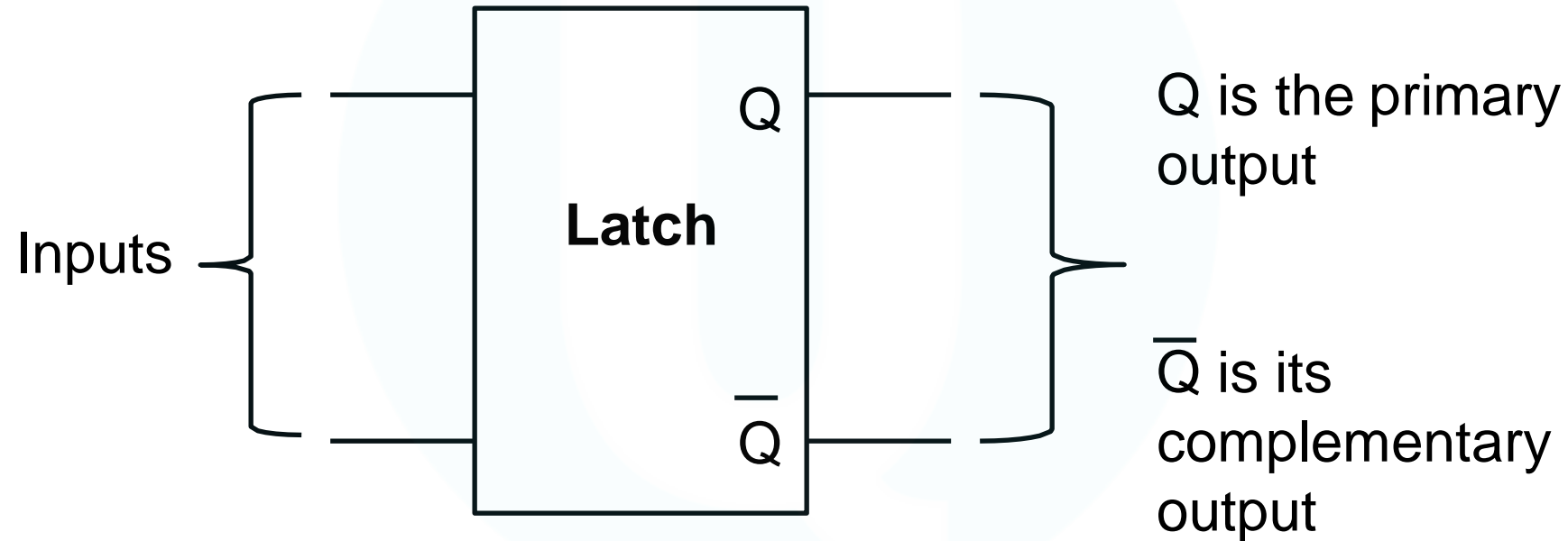
Classification of sequential circuits

1. Synchronous sequential ckts :
2. Asynchronous sequential ckts :

What is exactly Memory?

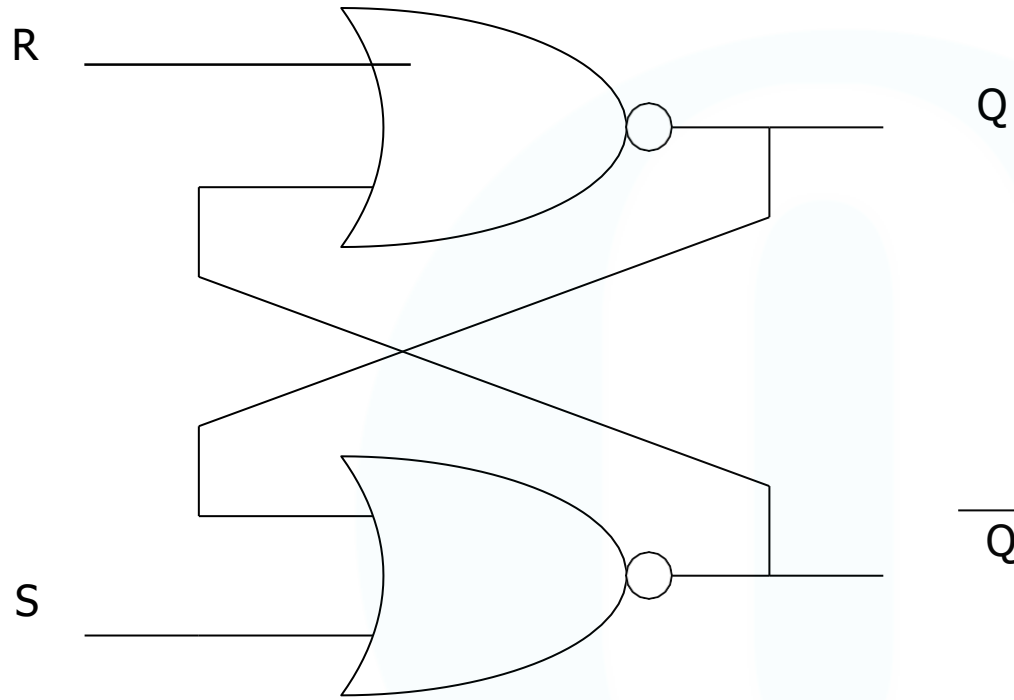
- A memory should support at least three operations:
 - ✓ It should be able to hold a value
 - ✓ You should be able to read the value that is saved
 - ✓ You should be able to change that value

- ✓ Latch are the bi-stable devices which responds to the change of input logic levels as they occur.

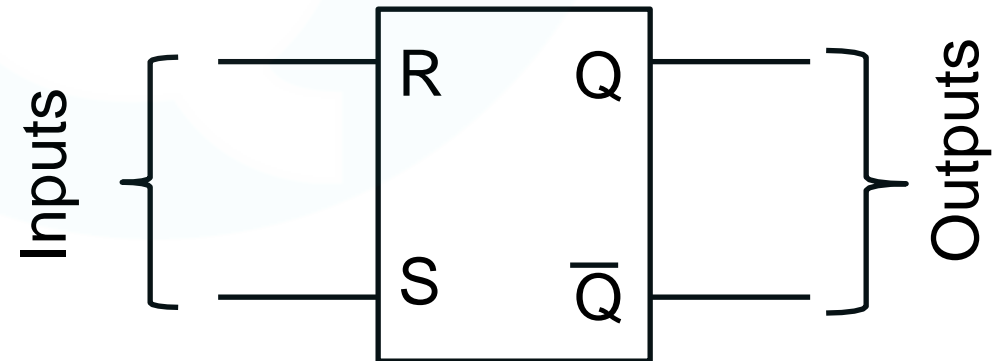


It is said to be in SET state if output Q is High
It is said to be in RESET state if output Q is Low

SR Latch using NOR



Circuit Diagram



Symbol

S	R	Q_n	Q_{n+1}	State
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	Indetermine
1	1	1	X	

Unit IV – Sequential Logic Circuit

✓ **Triggering Methods: Edge Trigger & Level Trigger.**

✓ **SR Flip Flops:** SR Flip Flop, Clocked SR FF with preset & clear,

Drawbacks of SR FF

✓ **JK Flip Flops:** Clocked JK FF with preset & clear, Race around

condition in JK FF, Master Slave JK FF, D and T type Flip Flop,

Excitation Tables of Flip Flops, Block schematic and function table

of IC 7474, IC 7475.

- ✓ A clock is a special device that whose output continuously alternates between 0 and 1.
- ✓ The time it takes the clock to change from 1 to 0 and back to 1 is called the clock period, or clock cycletime.
- ✓ Clocks are often used to synchronize circuits.

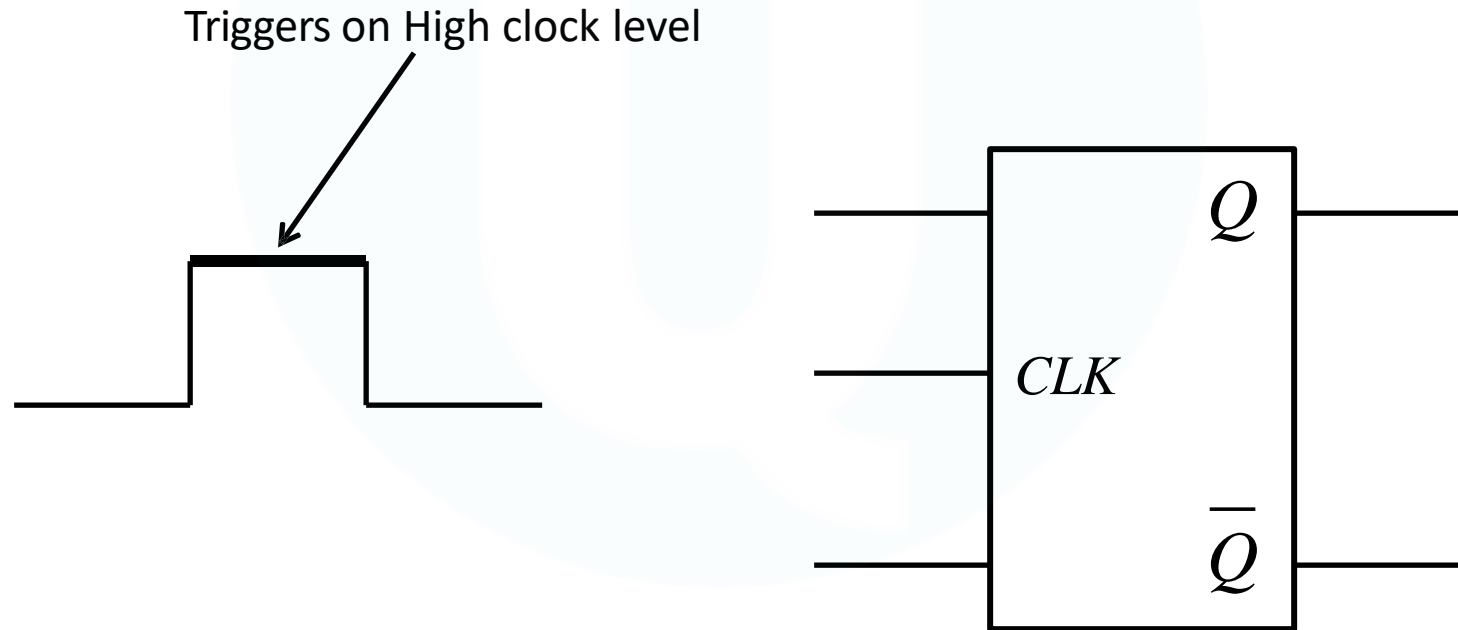
clock period



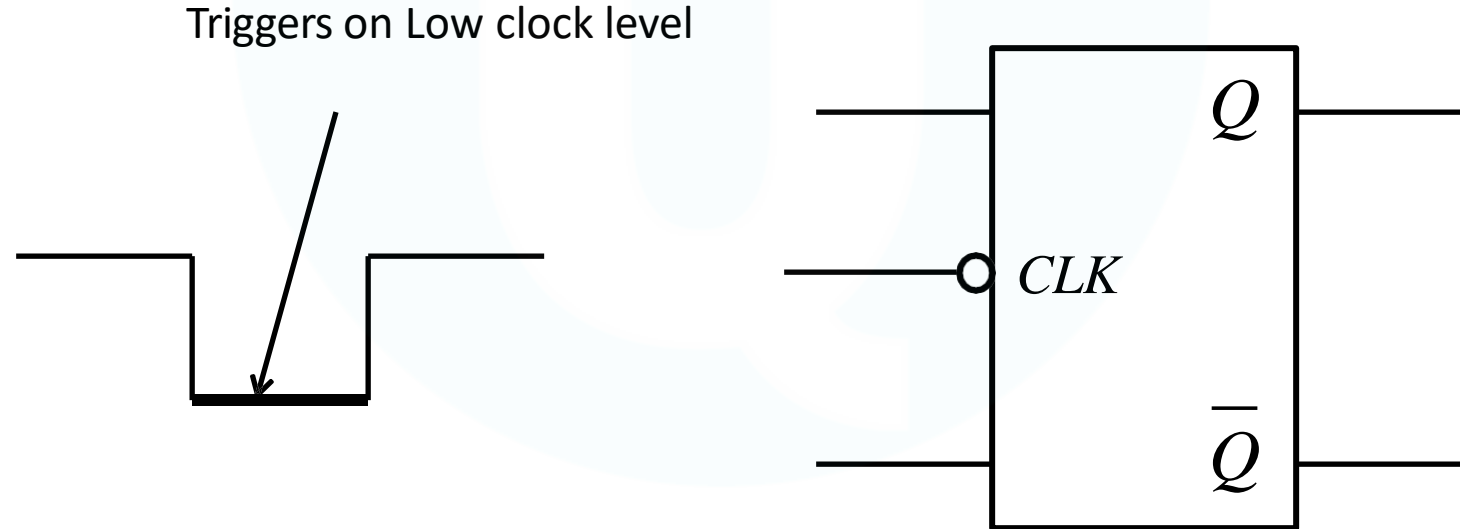
Triggering

- ✓ Sequential circuits are dependent on clock pulses applies to their inputs.
- ✓ The result of flip-flop responding to a clock input is called **clock pulse triggering**, of which there are four types. Each type responds to a clock pulse in one of four ways:-
 - High level triggering
 - Low level triggering
 - Positive edge triggering
 - Negative edge triggering

- ✓ A flip flop who responds to a clock signal during the time at which it is in the logic High state.



- ✓ A flip flop who responds to a clock signal during the time at which it is in the logic Low state.

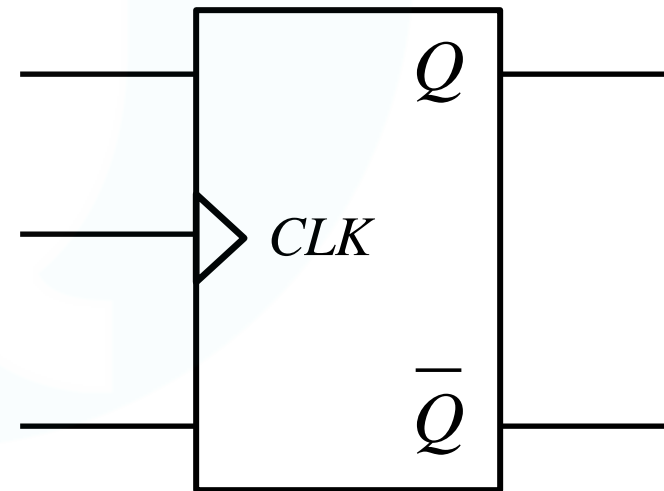
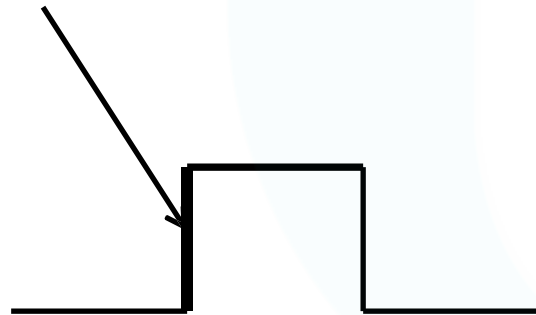


Symbol

Positive Edge Triggering

- ✓ A flip flop who responds to a clock signal during Low to High transition of clock pulse.

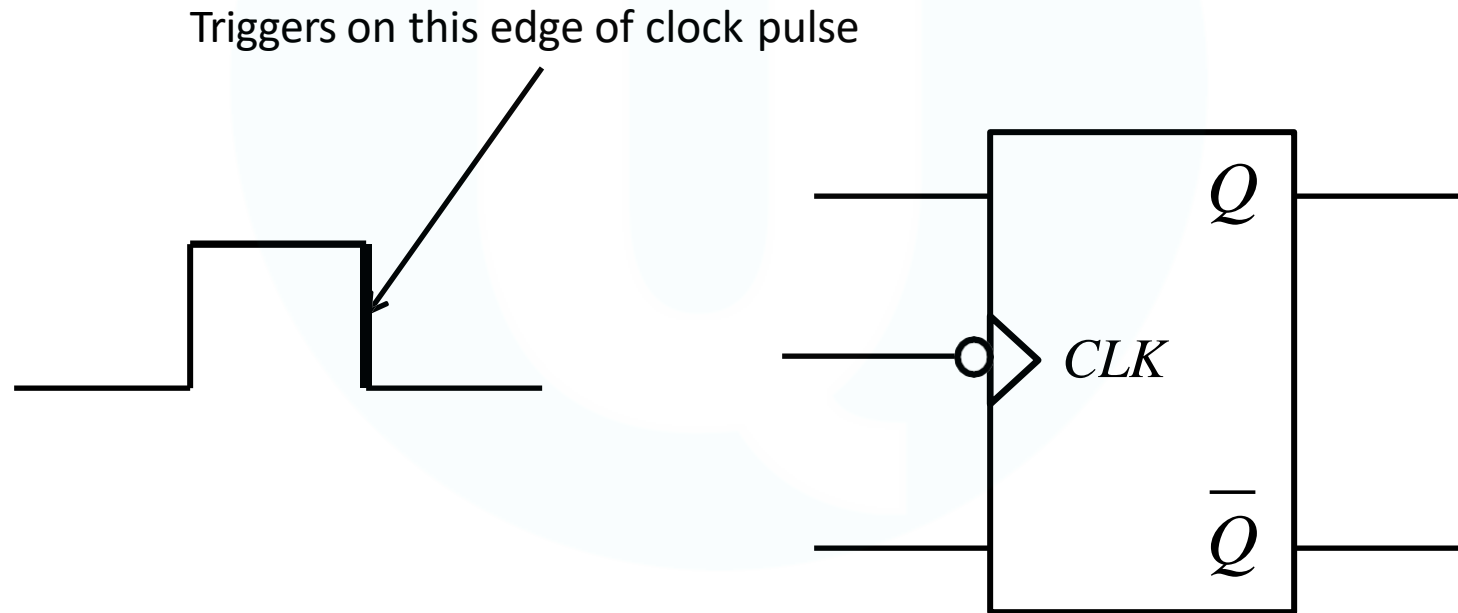
Triggers on this edge of clock pulse



Symbol

Negative Edge Triggering

- ✓ A flip flop who responds to a clock signal during High to Low transition of clock pulse.



Unit IV – Sequential Logic Circuit

✓ **Basic Memory Cell:** RS Latch–using NAND & NOR.

✓ **Triggering Methods:** Edge Trigger & Level Trigger.

✓ **SR Flip Flops:** SR Flip Flop, Clocked SR FF with preset & clear,

Drawbacks of SR FF

✓ **JK Flip Flops:** Clocked JK FF with preset & clear, Race around

condition in JK FF, Master Slave JK FF, D and T type Flip Flop,

Excitation Tables of Flip Flops, Block schematic and function table

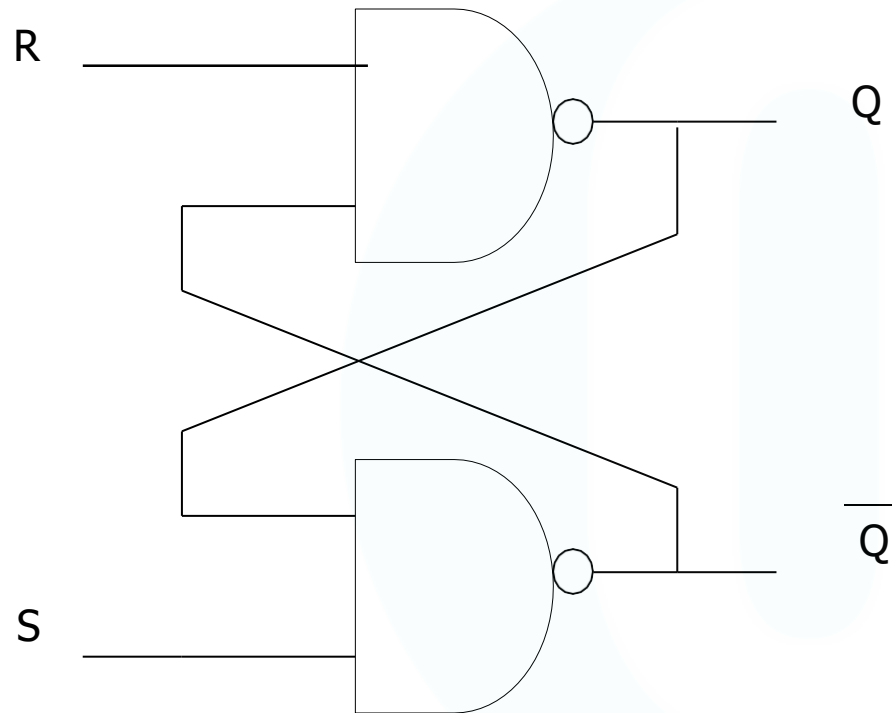
of IC 7474, IC 7475.

- ✓ **Gates** are the building block of the **logic circuits**.
Their primary function is to perform decision making operations.
- ✓ **Flip-flops** are the building blocks of the **digital circuits**. Their primary function is to store the binary bits.

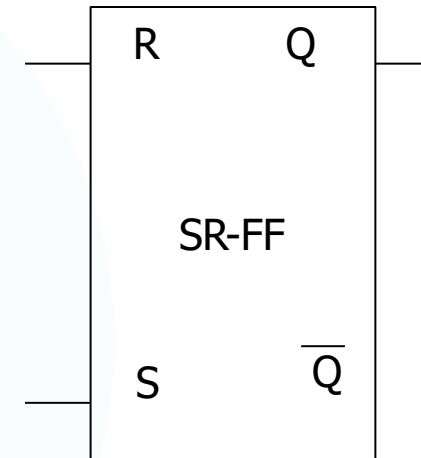
Flip Flops

- ✓ A flip-flop is a bi-stable device, with inputs, that remains in a given state as long as power is applied and until input signals are applied to cause its output to change.
- ✓ There are four basic different types of flip-flops:
 - SR Flip Flop
 - D Flip Flop
 - JK Flip Flop
 - T Flip Flop

SR Flip Flop



Circuit Diagram



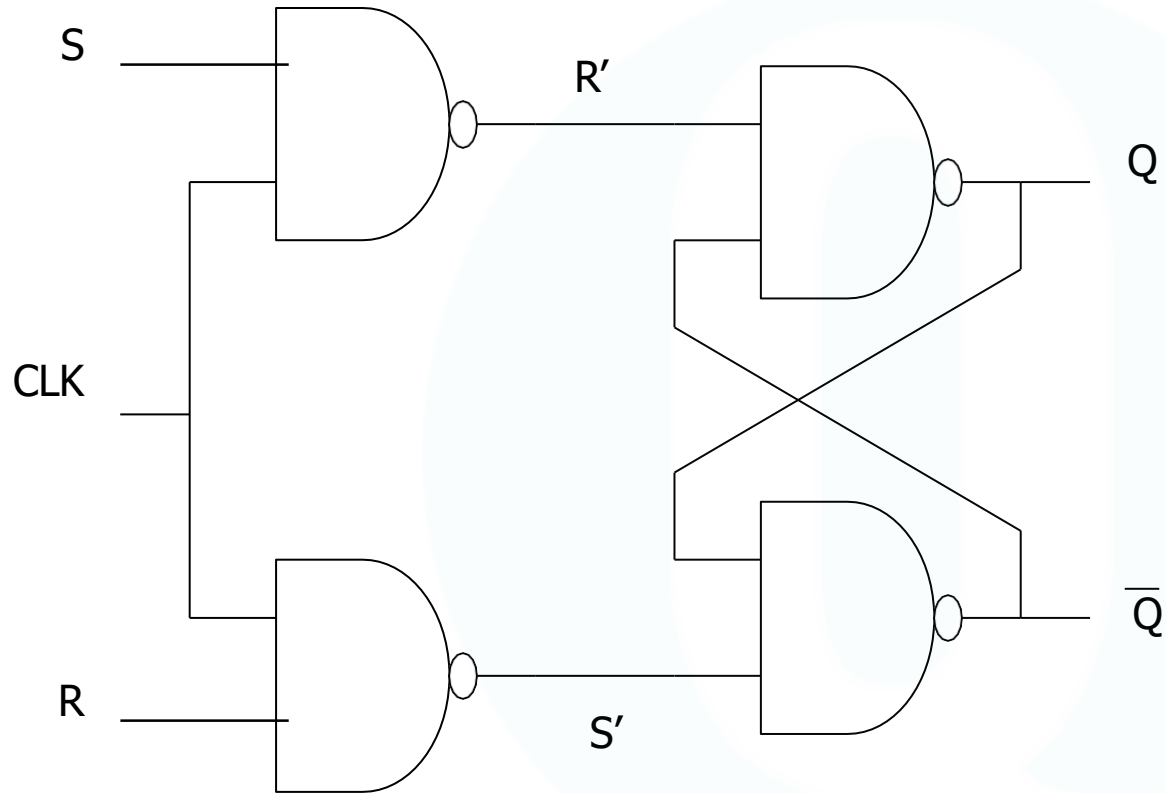
Symbol

SR Flip Flop

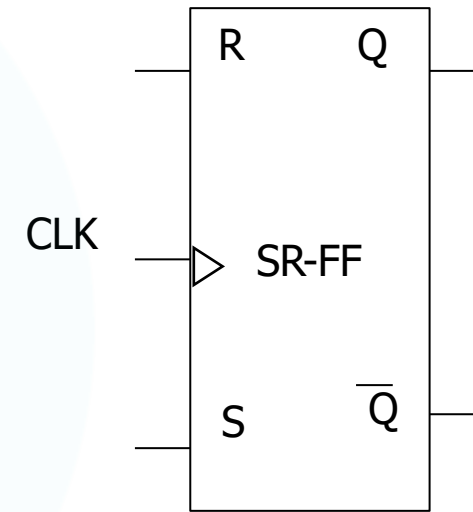
Logic Table

		Present o/p	Next o/p	
S	R	Q_n	Q_{n+1}	State
0	0	0	X	Indetermine
0	0	1	X	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	0	No Change
1	1	1	1	

Clocked SR Flip Flop




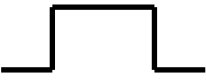



Circuit Diagram



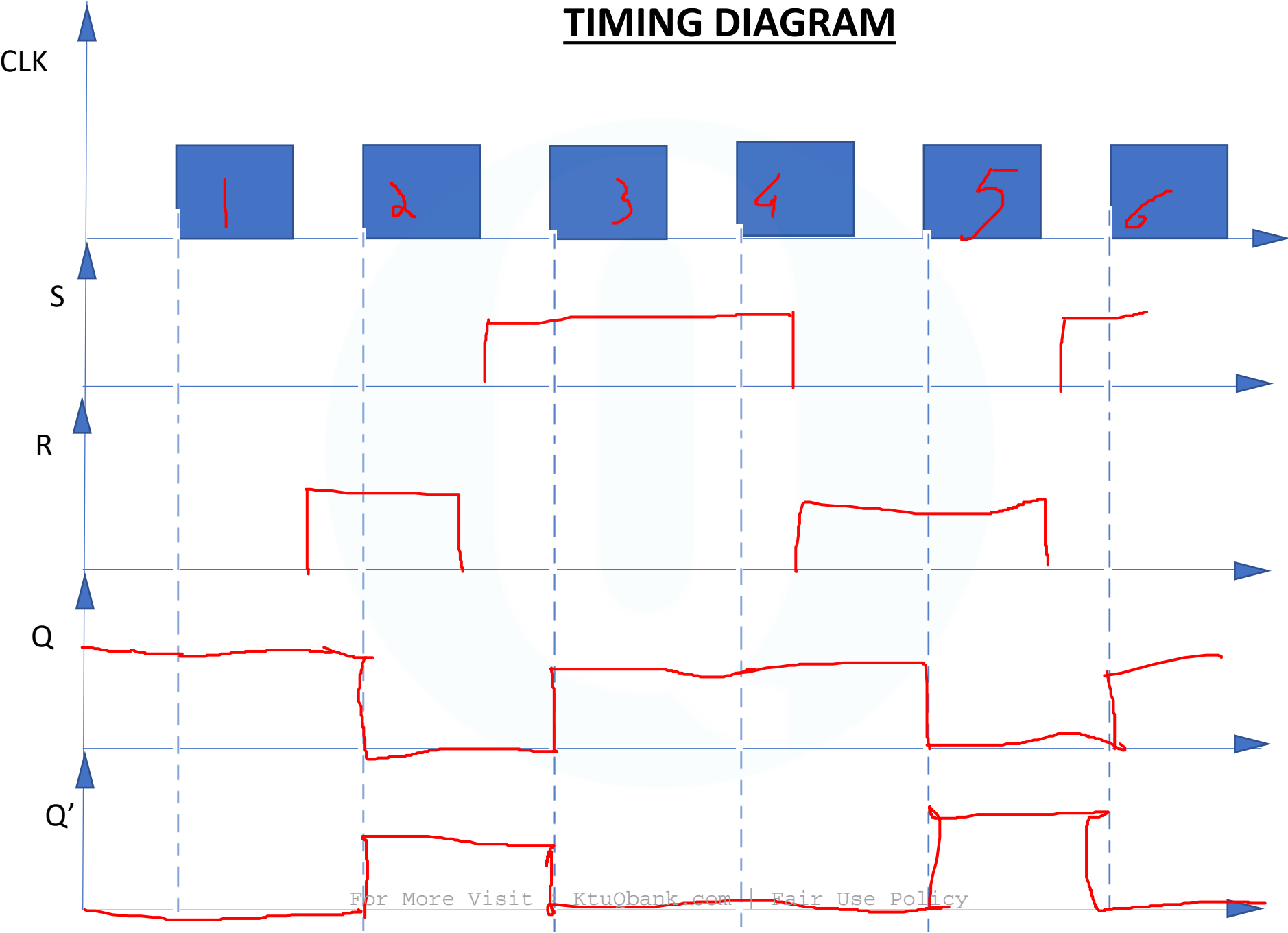
Symbol

Clocked SR Flip Flop

Logic Table

CLK	S	R	Q	\overline{Q}	State
	0	0	Q	\overline{Q}	No Change
	0	1	0	1	Reset
	1	0	1	0	Set
	1	1	X	X	Prohibited
	X	X	Q	\overline{Q}	No Change

TIMING DIAGRAM



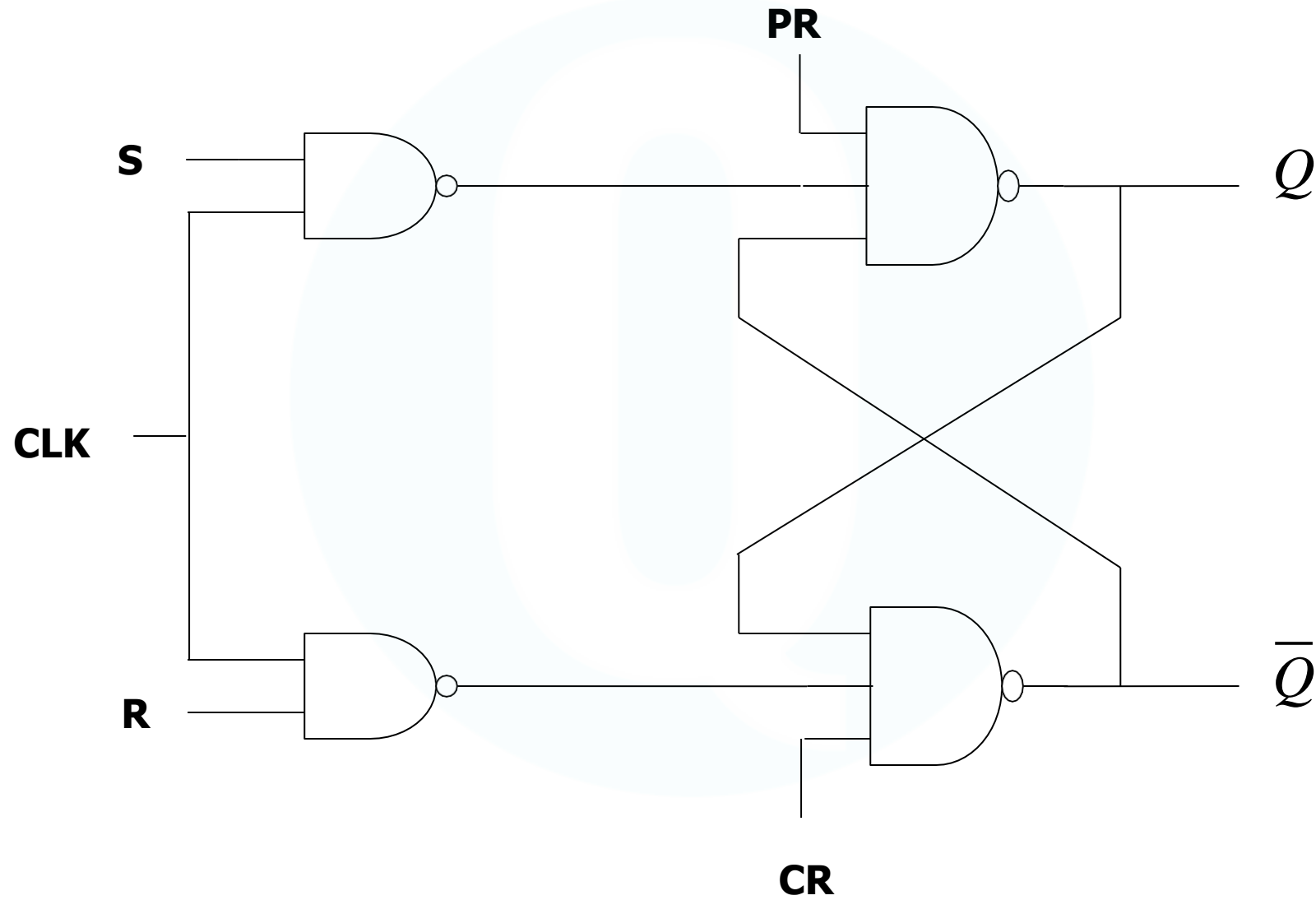
Synchronous Inputs

- ✓ The S and R (for SR FF), D (for D FF), J and K (JK FF), and so on...., inputs are control inputs.
- ✓ These inputs are also called “Synchronous Inputs” because the action of these inputs are synchronized with the action of clock.
- ✓ The flip flop changes state only on the application of clock signal.

Asynchronous Inputs

- ✓ In addition to synchronous inputs the flip flops have one or more asynchronous inputs .
- ✓ These asynchronous inputs operate independently of control and clock input.
- ✓ Two asynchronous inputs are \overline{PRESET} and \overline{CLEAR}
- ✓ These are mostly active LOW inputs.

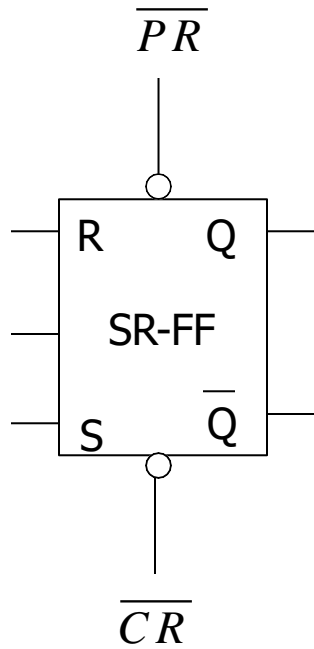
Clocked SR Flip Flop with Clear and Preset Inputs



Asynchronous Inputs: \overline{PRESET} and \overline{CLEAR}

Sr. No.	Action	Function/Operation
1	$\overline{PRESET} = 1$ $\overline{CLEAR} = 1$	Both these asynchronous inputs are inactive. The flip flop responds to synchronous inputs.
2	$\overline{PRESET} = 0$ $\overline{CLEAR} = 1$	The PRESET is activated and Q is immediately set to 1 irrespective of synchronous inputs. The clock input cannot affect the flip flop when PR=0.
3	$\overline{PRESET} = 1$ $\overline{CLEAR} = 0$	The CLEAR is activated and Q is immediately cleared to 0 irrespective of synchronous inputs. The clock input cannot affect the flip flop when CR=0.
4	$\overline{PRESET} = 0$ $\overline{CLEAR} = 0$	This condition should not be used as it leads to race condition

Clocked SR Flip Flop with Clear and Preset Inputs



Inputs					O/P		Comment
CLK	\overline{CR}	\overline{PR}	S	R	Q	\overline{Q}	
	1	1	0	0	Q	Q	No Change
	1	1	0	1	0	1	Reset
	1	1	1	0	1	0	Set
	1	1	1	1	X	X	Invalid
X	0	1	X	X	0	1	Clear
X	1	0	X	X	1	0	Preset
X	0	0	X	X	X	X	Invalid

Synchronous Sequential Circuits Vs Asynchronous Sequential Circuits

Synchronous Seq Circuits

- ✓ In Synchronous circuits, memory elements are clocked FFs.
- ✓ In Synchronous circuits, the change in input signals can affect memory elements upon activation of clock signals.

Asynchronous Seq Circuits

- ✓ In Asynchronous circuits, memory elements are either unclocked FFs or time delay elements.
- ✓ In Asynchronous circuits, the change in input signals can affect memory elements at any instant of time.

Synchronous Sequential Circuits Vs Asynchronous Sequential Circuits

Synchronous Seq Circuits

- ✓ The maximum operating speed of the clock depends on time delays involved
- ✓ Easier to design

Asynchronous Seq Circuits

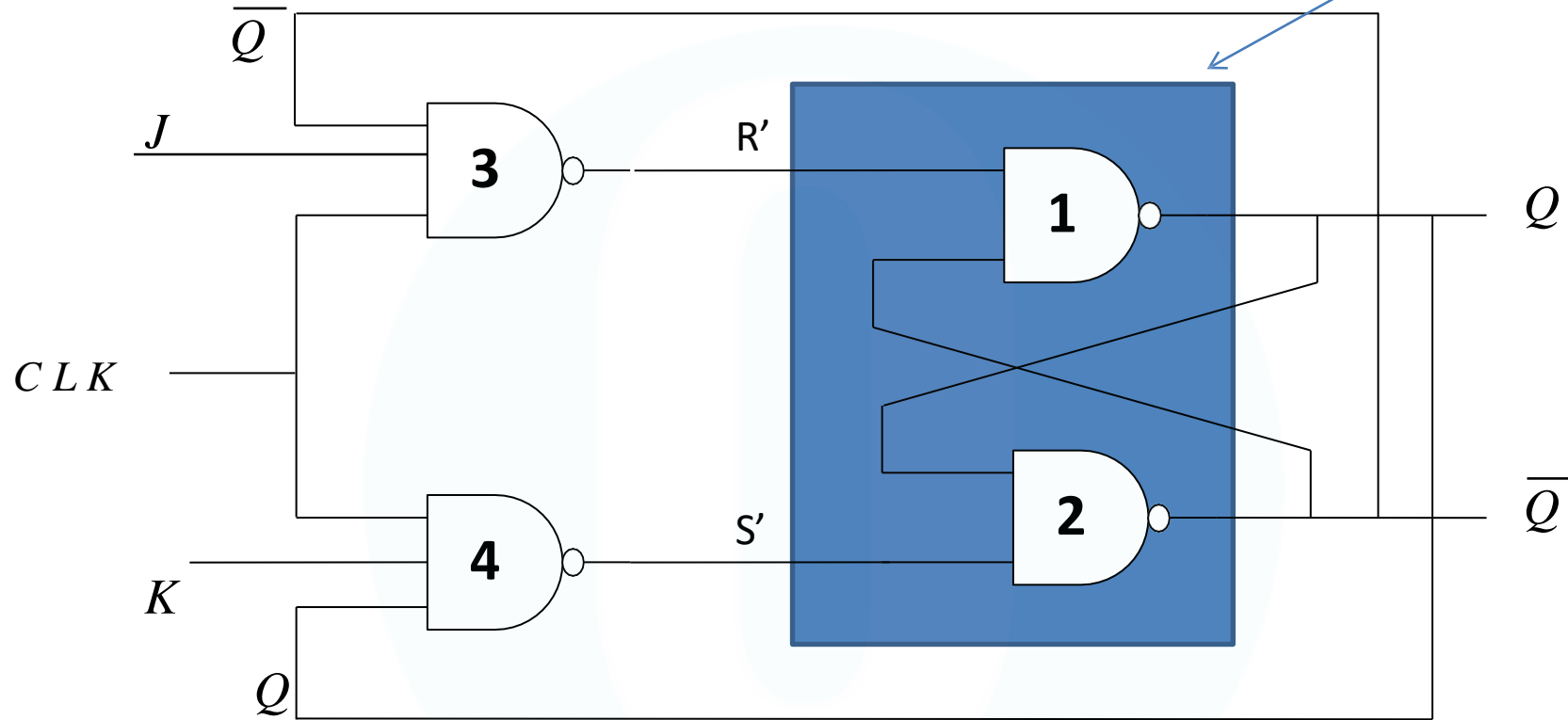
- ✓ Because of the absence of the clock, asynchronous circuits can operate faster than synchronous circuits.
- ✓ More difficult to design

Drawbacks of SR Flip Flop

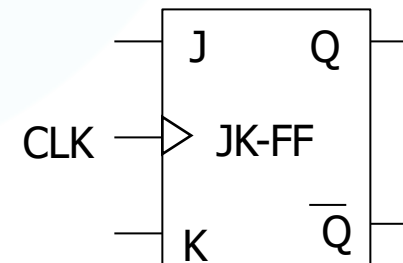
- ✓ If both inputs are pulled down to logic level 0, both outputs will be at logic level 1. This state should not be allowed to occur in flip-flops.

Level Triggered JK Flip Flop

SR Flip Flop



Circuit Diagram of Level Triggered JK Flip Flop

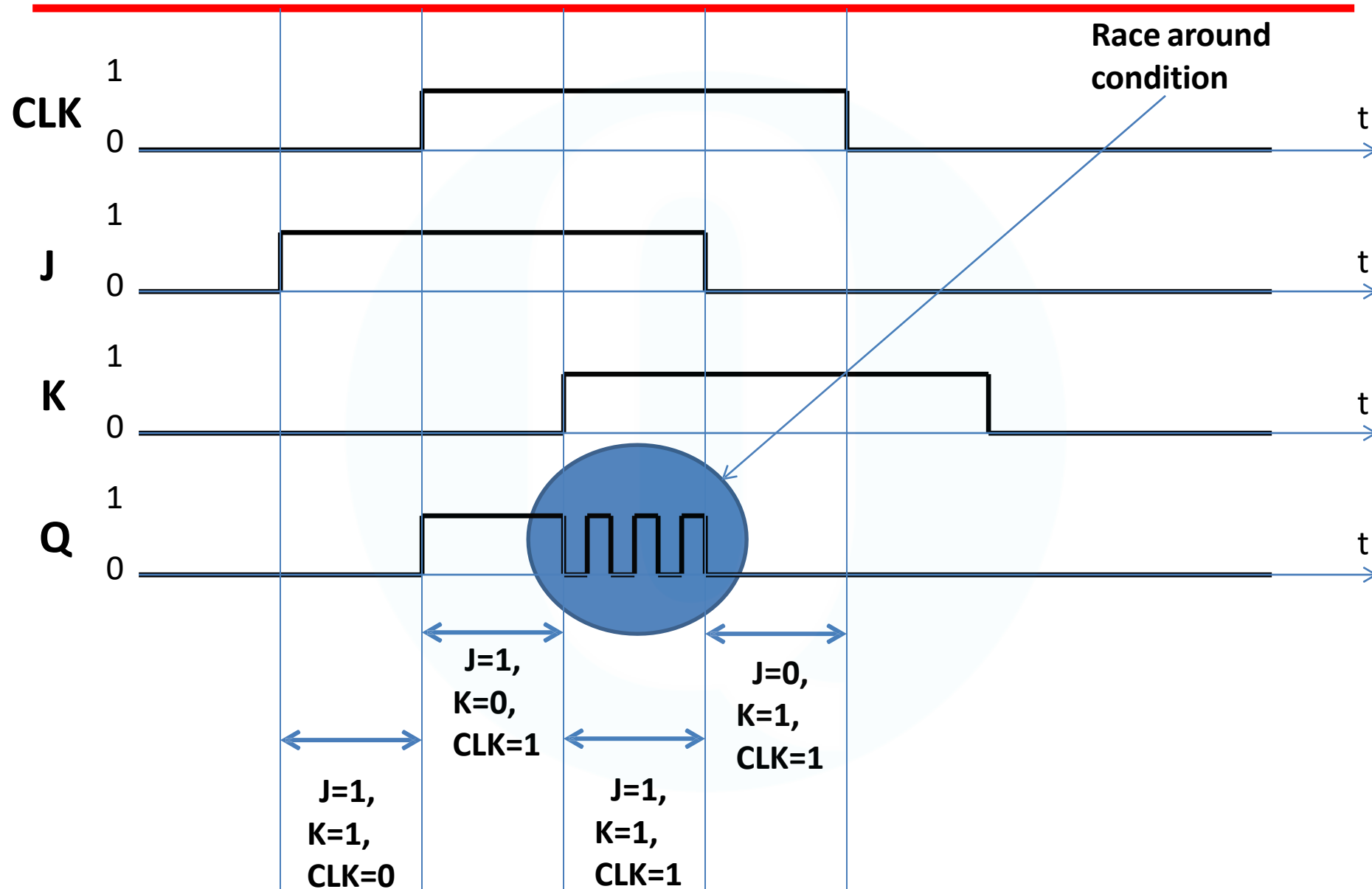


Symbol

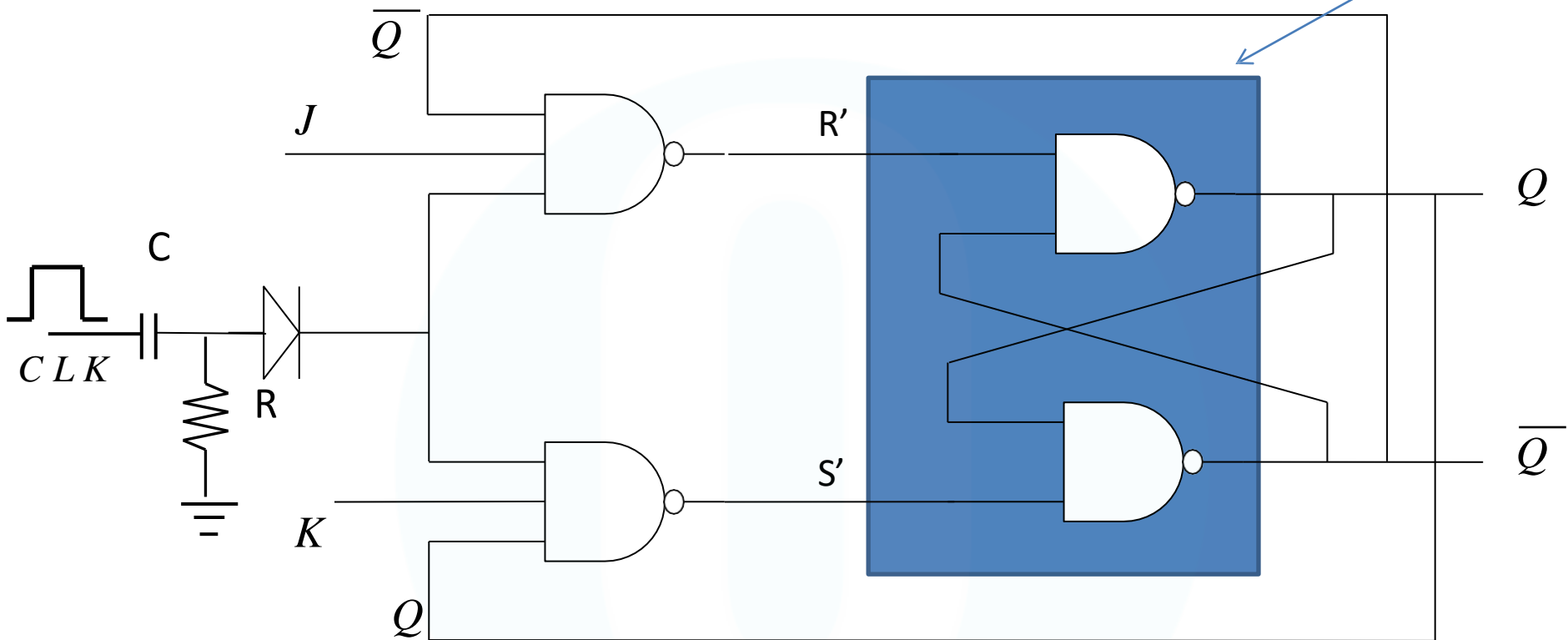
Level Triggered JK Flip Flop

Inputs			Outputs		State
CLK	J	K	Q_{n+1}	$\overline{Q_{n+1}}$	
0	X	X	Q_n	$\overline{Q_n}$	Flip Flop is Disabled (No Change)
1	0	0	Q_n	$\overline{Q_n}$	
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	$\overline{Q_n}$	Q_n	Toggle

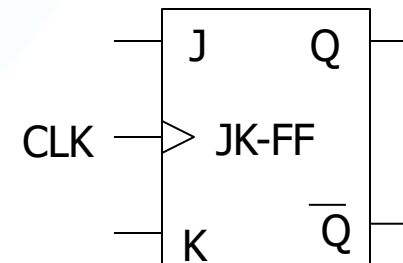
Timing Diagram of Level Triggered JK Flip Flop



Edge Triggered JK Flip Flop



Circuit Diagram of Edge Triggered JK Flip Flop

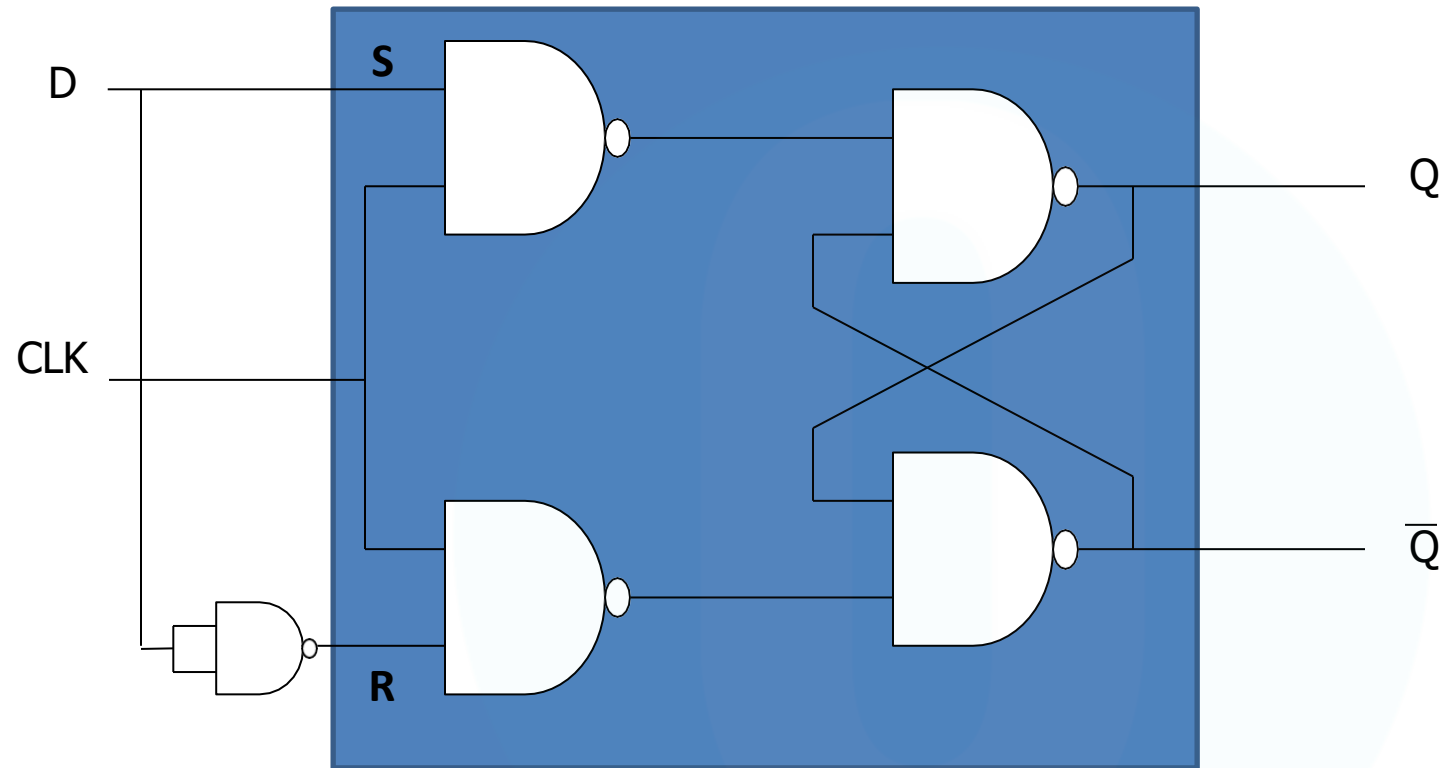


Symbol

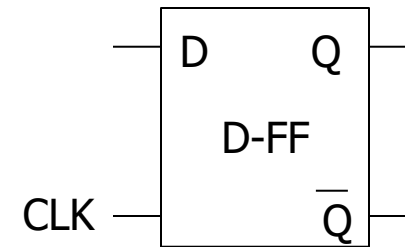
Edge Triggered JK Flip Flop

Inputs			Outputs		State
CLK	J	K	Q_{n+1}	$\overline{Q_{n+1}}$	
0 or 1	X	X	Q_n	$\overline{Q_n}$	Flip Flop is Disabled (No Change)
↓	X	X	Q_n	$\overline{Q_n}$	
↑	0	0	Q_n	$\overline{Q_n}$	
↑	0	1	0	1	Reset
↑	1	0	1	0	Set
↑	1	1	$\overline{Q_n}$	Q_n	Toggle

D Flip Flop



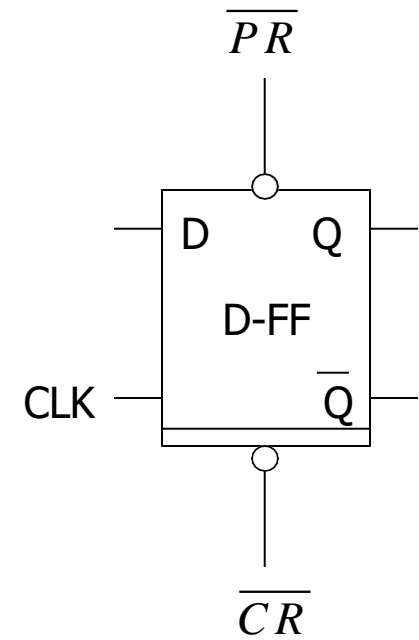
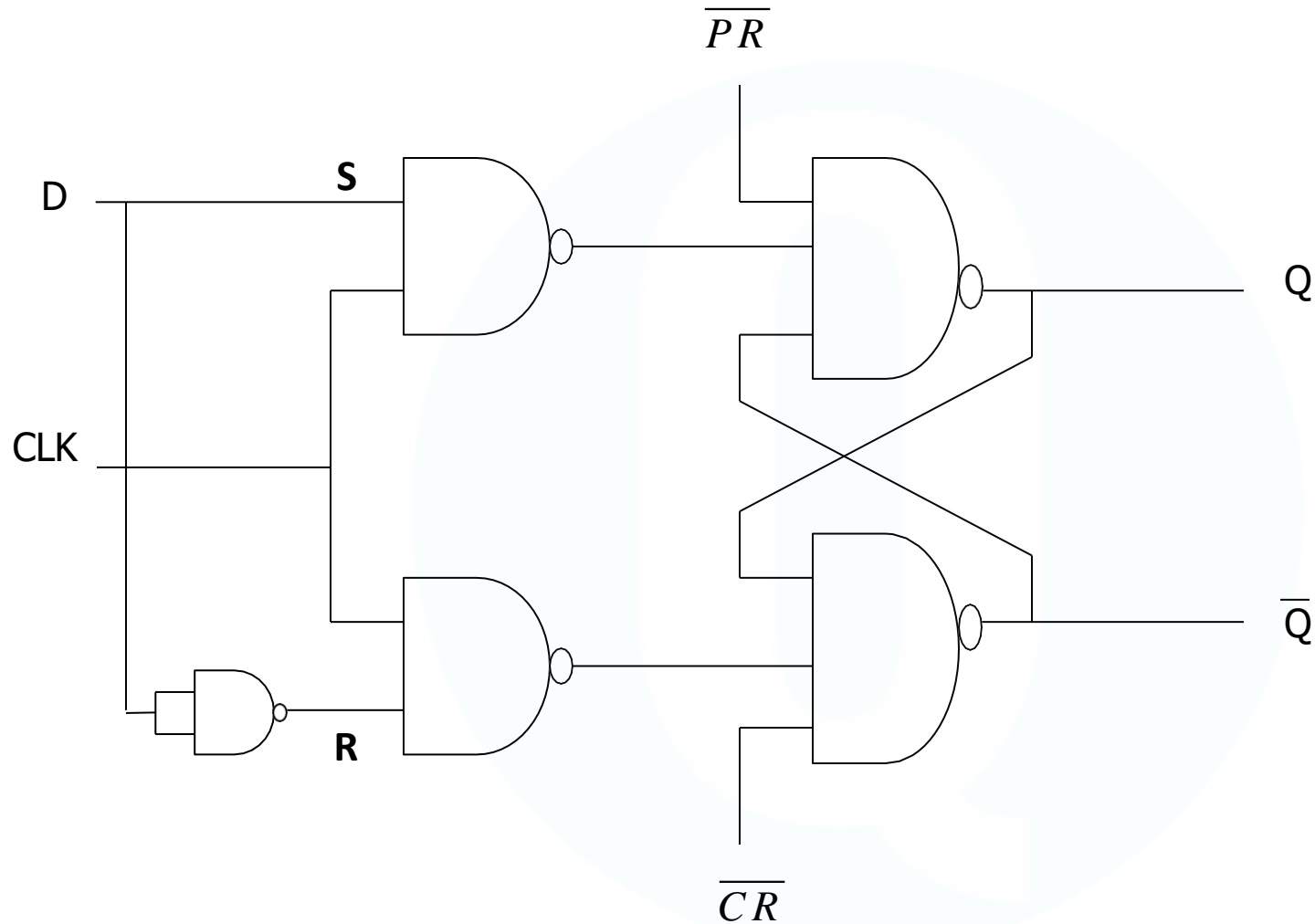
Clocked SR Flip Flop



D Flip Flop

Inputs		Output		Comment
CLK	D	Q_{n+1}	$\overline{Q_{n+1}}$	
0	X	Q_n	$\overline{Q_n}$	Last Value or No Change
1	0	0	1	Reset
1	1	1	0	Set

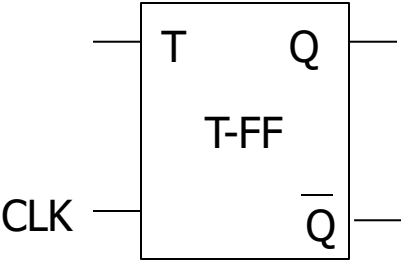
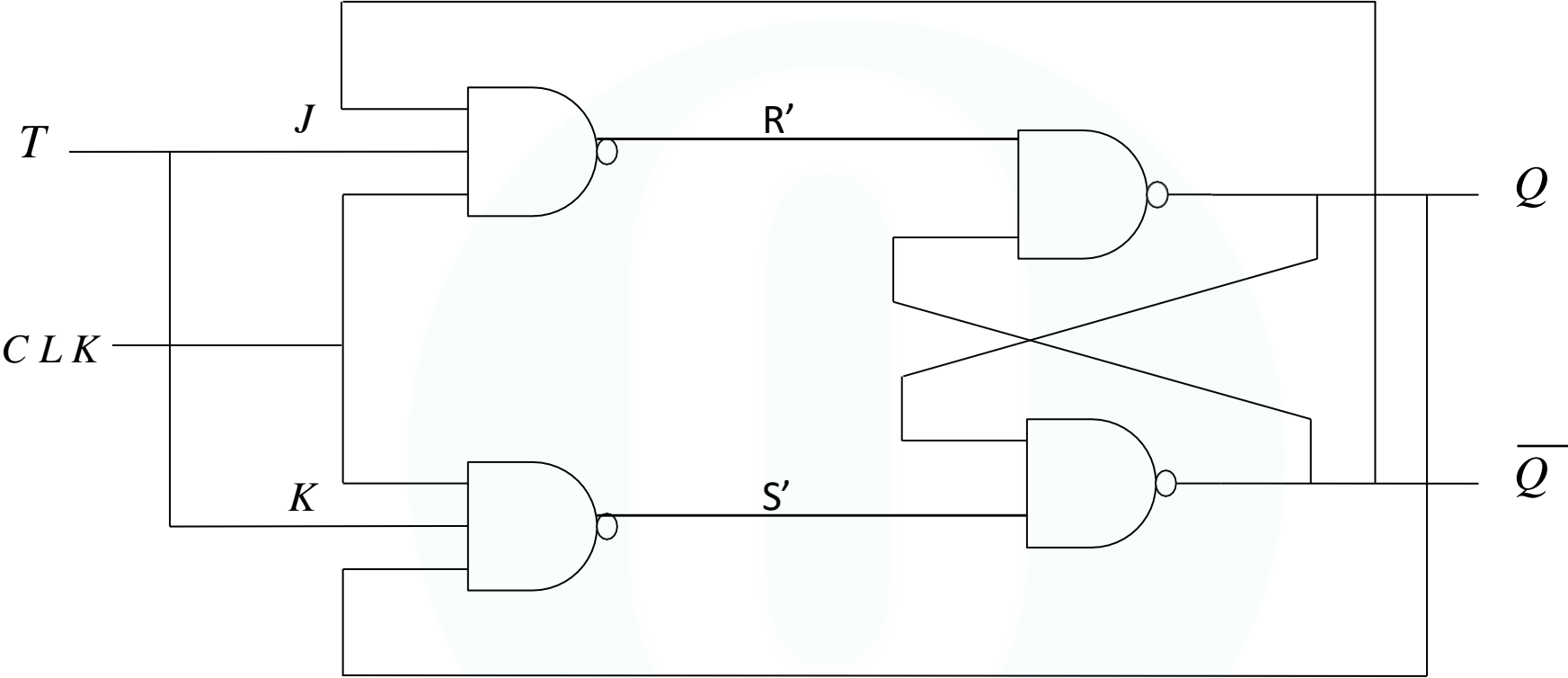
D Flip Flop with Preset and Clear Inputs



D Flip Flop with Preset and Clear Inputs

Inputs				Output		Comment
\overline{PR}	\overline{CR}	CLK	D	Q_{n+1}	$\overline{Q_{n+1}}$	
0	0	X	X	$\overline{Q_n}$	Q_n	Avoid
0	1	X	X	1	0	Preset
1	0	X	X	0	1	Clear
1	1	0	X	Q_n	$\overline{Q_n}$	No Change
1	1	1	0	0	1	Reset
1	1	1	1	1	0	Set

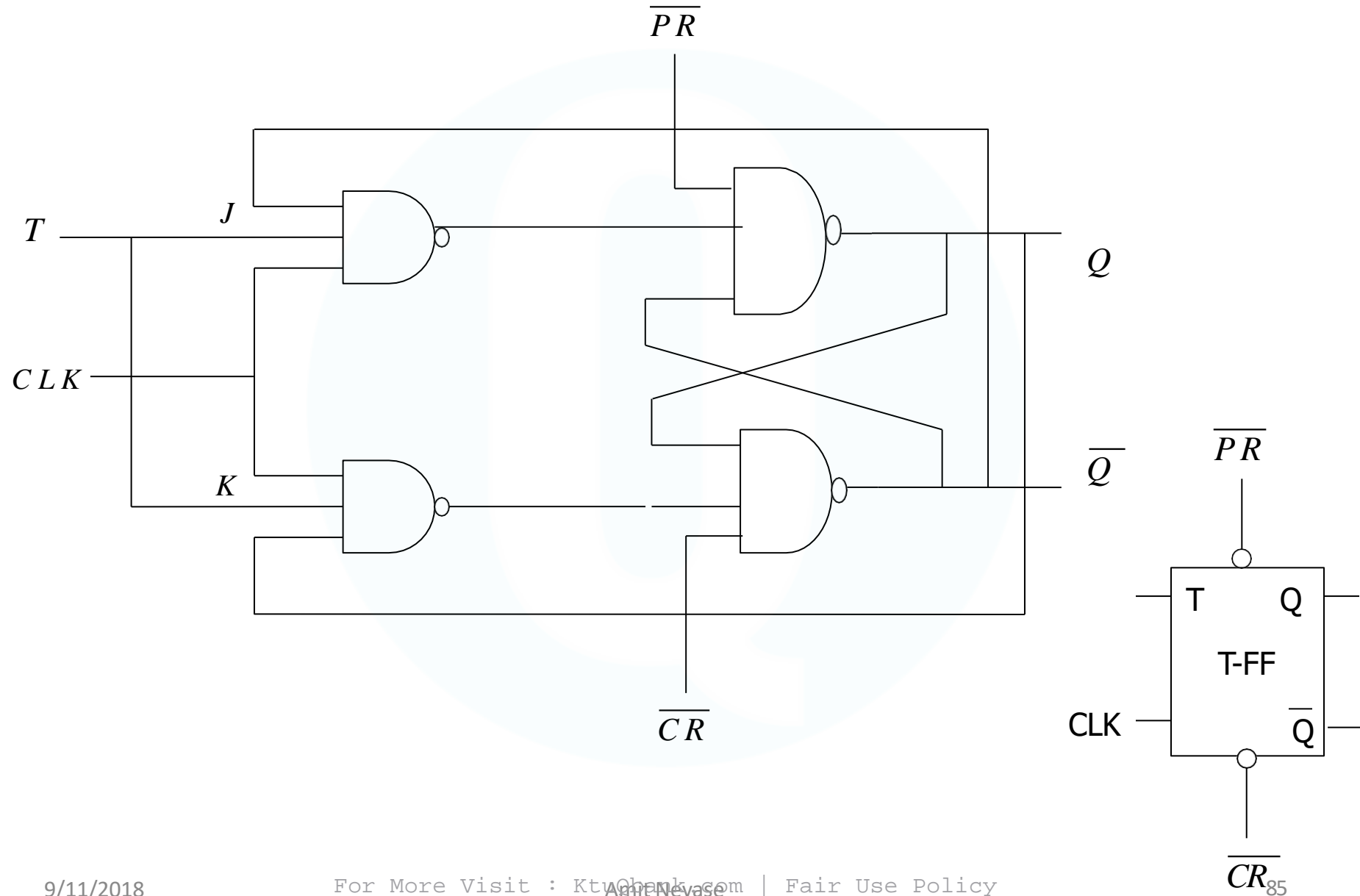
T Flip Flop



T Flip Flop

Inputs		Output		Comment
CLK	T	Q_{n+1}	$\overline{Q_{n+1}}$	
0	X	Q_n	$\overline{Q_n}$	No Change
1	0	Q_n	$\overline{Q_n}$	No Change
1	1	$\overline{Q_n}$	Q_n	Toggle

T Flip Flop with Preset and Clear Inputs



T Flip Flop with Preset and Clear Inputs

Inputs				Output		Comment
\overline{PR}	\overline{CR}	CLK	T	Q_{n+1}	$\overline{Q_{n+1}}$	
0	0	X	X	$\overline{Q_n}$	Q_n	Avoid
0	1	X	X	1	0	Preset
1	0	X	X	0	1	Clear
1	1	0	X	Q_n	$\overline{Q_n}$	No Change
1	1	1	0	Q_n	$\overline{Q_n}$	No Change
1	1	1	1	$\overline{Q_n}$	Q_n	Toggle

Applications of Flip Flops

- ✓ Elimination of Keyboard Debounce
- ✓ As a memory element
- ✓ In various types of registers
- ✓ In counters
- ✓ As delay element
- ✓ Parallel Data storage
- ✓ Serial Data Storage
- ✓ Serial to Parallel Conversion
- ✓ Parallel to Serial Conversion
- ✓ Frequency Division

Excitation Tables of Flip Flops

- ✓ Logic tables show the state of the output(s) of a logic circuit as a function of its inputs **at the same time.**
- ✓ Since, clocked digital systems have **memory**, their behavior depends on inputs **in the past** as well as the present values of the inputs.

Excitation Tables of Flip Flops

- Thus, flip-flops cannot be described by simple truth tables. Instead, we use excitation or transition tables. These show:
 - ✓ output before the clock transition — often labelled Q_n
 - ✓ inputs at the clock transition — such as S and R
 - ✓ occasionally the type of clock transition – positive/negative edge-triggered
 - ✓ the resulting output after the clock transition — often labelled Q_{n+1}
- It is important to remember that Q_n and Q_{n+1} describe the same signal but at different times. The notation can vary, e.g. Q_0 and Q instead.

Excitation Table of SR Flip Flop

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

Truth Table

Excitation Table

Present State O/P	Next State O/P	Required Inputs	
Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

- ✓ $0 \rightarrow 0$ **transition:** If the present state of the FF is 0 and if it has to remain 0 when a clock pulse is applied, the inputs can be either $S=0, R=0$ (no change condition) or $S=0, R=1$ (Reset condition). Thus, S has to be 0 but R can be either 0 or 1. So $SR=0X$ for this transition.
- ✓ $0 \rightarrow 1$ **transition:** If the present state of the FF is 0 and if it has to go 1 state when a clock pulse is applied, the inputs have to be $S=1$ and $R=0$ (set condition). So $SR=10$ for this transition.

Excitation Table of SR Flip Flop

- ✓ 1 \rightarrow 0 transition: If the present state of the FF is 1 and if it has to go to 0 state when a clock pulse is applied, the inputs have to be $S=0$ and $R=1$ (Reset condition). So $SR=01$ for this transition.
- ✓ 1 \rightarrow 1 transition: If the present state of the FF is 1 and if it has to remain 1 when a clock pulse is applied, the inputs can be either $S=0, R=0$ (no change condition) or $S=1, R=0$ (set condition). Thus, R has to be 0 but S can be either 0 or 1. So $SR=X0$ for this transition.

Excitation Table of JK Flip Flop

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q_n}$

Truth Table

Excitation Table

Present State O/P	Next State O/P	Required Inputs	
Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Excitation Table of JK Flip Flop

- ✓ $0 \rightarrow 0$ transition: The present state of the FF is 0 and it has to remain 0 after the clock pulse. This can happen with either $J=0, K=0$ (no change condition) or $J=0, K=1$ (reset condition). Thus, J has to be 0 but K can be either 0 or 1. So $JK=0X$ for this transition.
- ✓ $0 \rightarrow 1$ transition: The present state of the FF is 0 and it has to go 1 state after the clock pulse. This can happen with either $J=1, K=0$ (set condition) or $J=1, K=1$ (toggle condition). Thus, J has to be 1 but K can be either 0 or 1. So $JK=1X$ for this transition.

Excitation Table of JK Flip Flop

- ✓ $1 \rightarrow 0$ transition: The present state of the FF is 1 and it has to go to 0 after the clock pulse. This can happen with either $J=0, K=1$ (reset condition) or $J=1, K=1$ (toggle condition). Thus, K has to be 1 but J can be either 0 or 1. So $JK=X1$ for this transition.
- ✓ $1 \rightarrow 1$ transition: The present state of the FF is 1 and it has to remain in 1 state after the clock pulse. This can happen with either $J=0, K=0$ (no change condition) or $J=1, K=0$ (set condition). Thus, K has to be 0 but J can be either 0 or 1. So $JK=X0$ for this transition.

Excitation Table of D Flip Flop

D	Q_{n+1}
0	0
1	1

Truth Table

Excitation Table

Present State O/P	Next State O/P	Required Inputs
Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Excitation Table of D Flip Flop

- ✓ For a D Flip Flop, the next state is always equal to the D input and it is independent of the present state. Therefore, D must be 0 if Q_{n+1} has to be 1 regardless of the value of Q_n .

Excitation Table of T Flip Flop

T	Q_{n+1}
0	Q_n
1	$\overline{Q_n}$

Truth Table

Excitation Table

Present State O/P	Next State O/P	Required Inputs
Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	1

Excitation Table of T Flip Flop

- ✓ For a T Flip Flop, when the input $T=1$, the state of the Flip flop is complemented and when $T=0$, the state of the Flip Flop remains unchanged.

Thus, for $0 \rightarrow 0$ and $1 \rightarrow 1$ transitions T must be 0 and for $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions T must be 1.

FLIP FLOP CONVERSIONS

- SR to D
- SR to JK
- SR to T
- JK to T
- JK to D
- JK to SR
- D to T
- D to SR
- T to D

PROCEDURE FOR CONVERSION

1. Draw the block diagram of the target flip flop from the given problem.
2. Write truth table for the target flip-flop.
3. Write excitation table for the available flip-flop.
4. Draw k-map for target flip-flop.
5. Draw the block diagram.

SR(Available) to D(Target) Flip flop Conversion

- Truth table**

Input	Present state	Next state
D	Q_n	Q_{n+1}
0	0	0
0	1	0
1	0	1
1	1	1

- Excitation table**

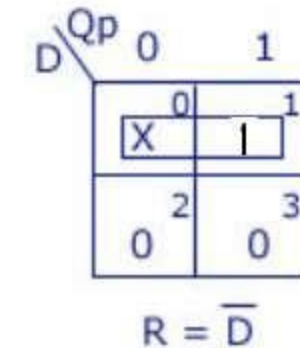
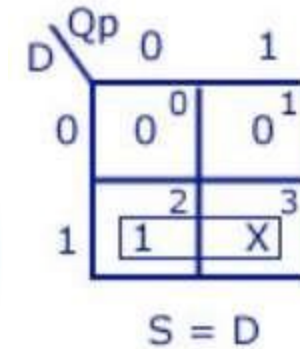
Present state	Next state	Flip flop Inputs	
Q_n	Q_{n+1}	S	R
0	0	0	X
0	0	0	1
0	1	1	0
1	1	X	0

SR to D Flip flop Conversion

Conversion Table

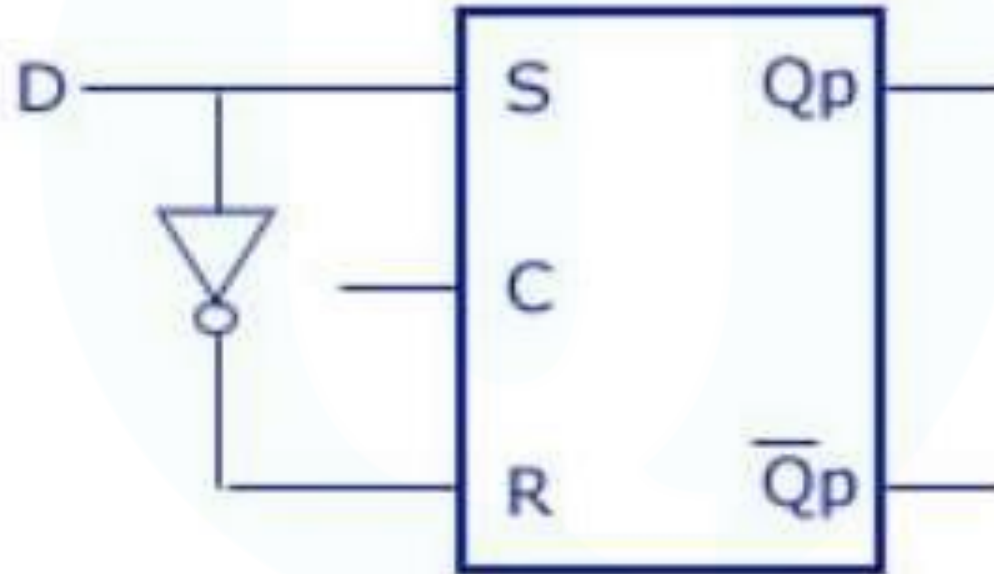
Input	Present state	Next state	Flip flop Inputs	
D	Q_n	Q_{n+1}	S	R
0	0	0	0	X
0	1	0	0	1
1	0	1	1	0
1	1	1	X	0

K- MAP SIMPLIFICATION



SR to D

Logic Diagram



SR(Available) to JK(Target) Flip-Flop

Conversion Table

Input		Present State	Next State	Flip-Flop Inputs	
J	K	Q n	Qn+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

For More Visit : KtuQbank.com | Fair Use Policy

SR to JK

- K-map Simplification

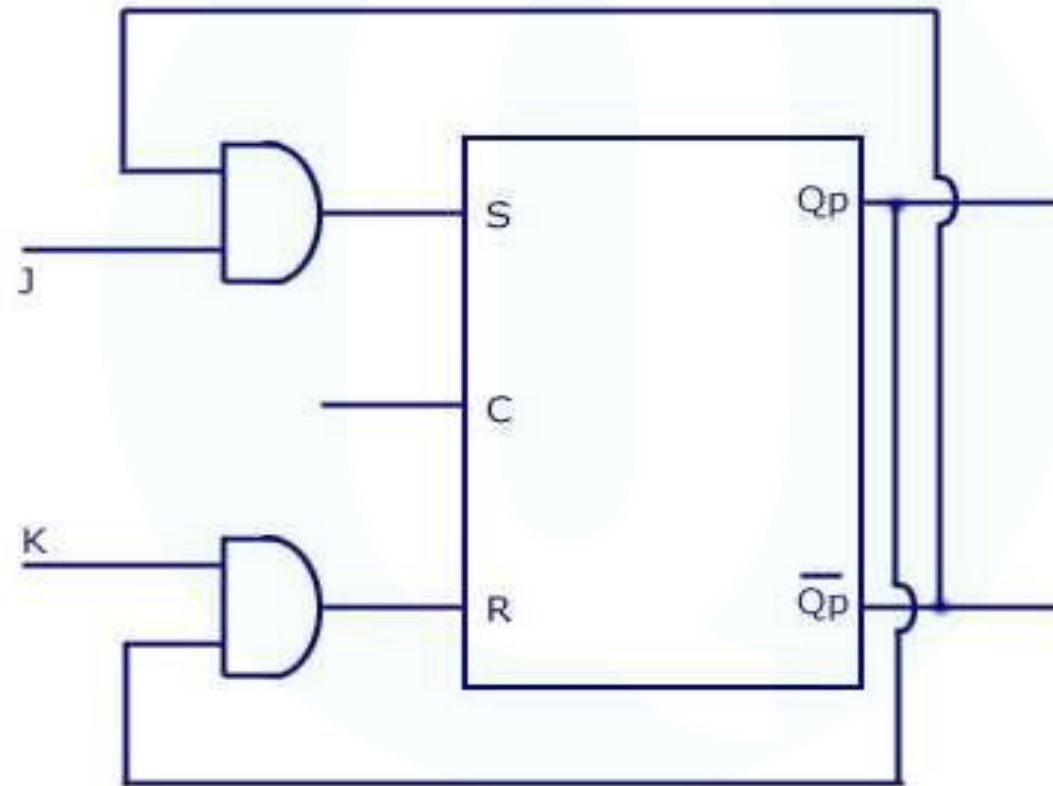
J	KQ _p			
	00	01	11	10
0	0 ⁰	X ¹	0 ³	0 ²
1	1 ⁴	X ⁵	0 ⁷	1 ⁶

$$S = \bar{J}Q_p$$

J	KQ _p			
	00	01	11	10
0	X ⁰	0 ¹	1 ³	X ²
1	0 ⁴	0 ⁵	1 ⁷	0 ⁶

$$R = KQ_p$$

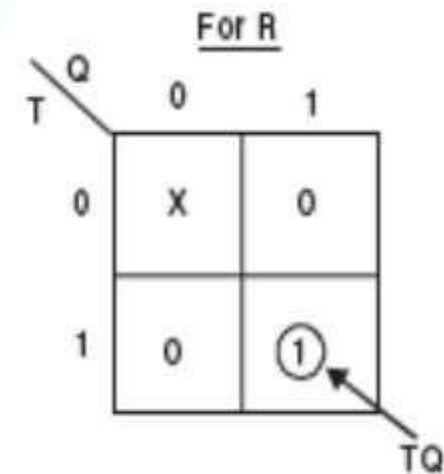
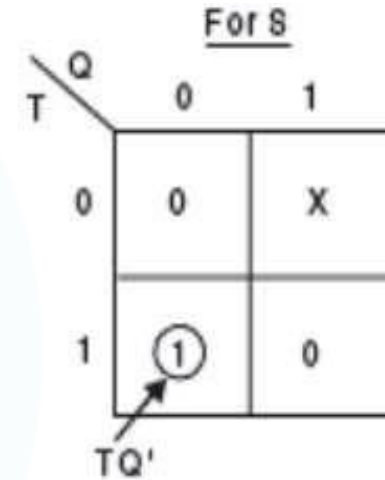
Logic Diagram (SR to JK)



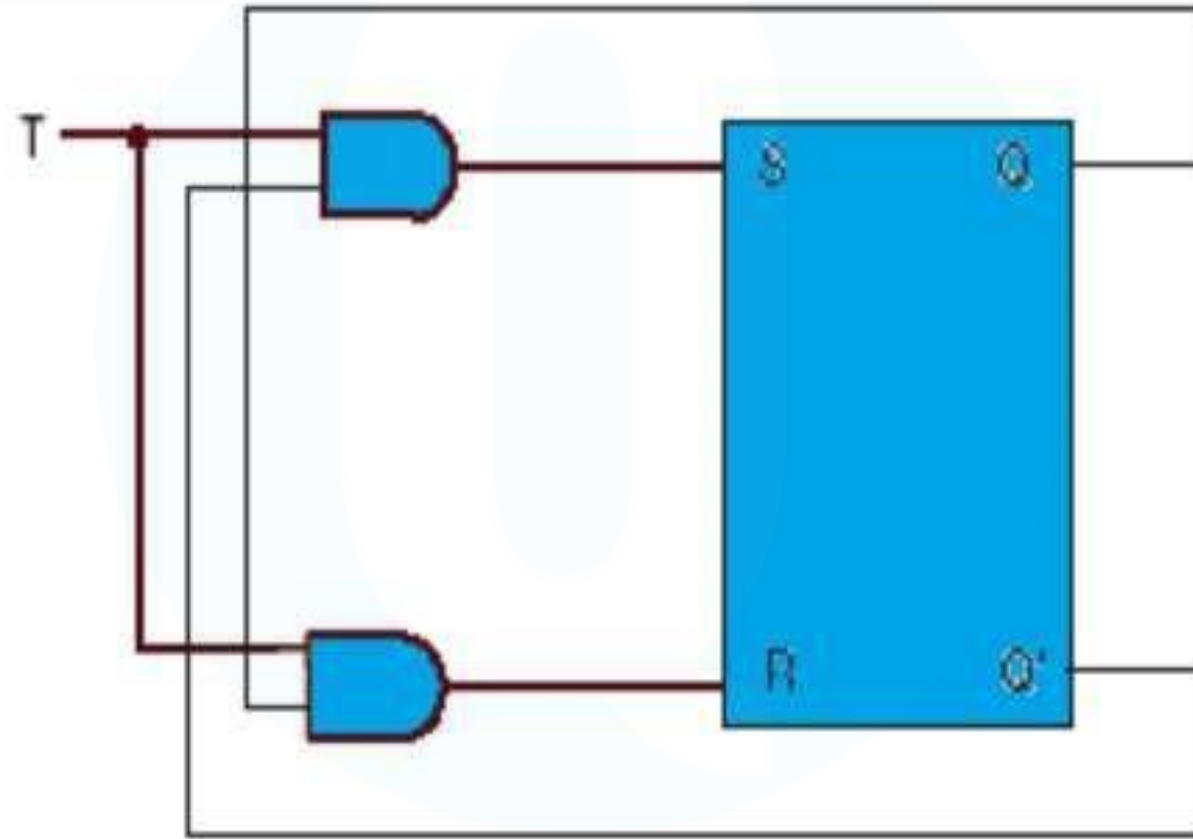
SR(Available) to T(Target) Conversion Table

Input	Present state	Next state	Flip flop Inputs	
T	Q_n	Q_{n+1}	S	R
0	0	0	0	X
0	1	1	X	0
1	0	1	1	0
1	1	0	0	1

K- MAP SIMPLIFICATION



Logic Diagram (SR to T)



A T flip-flop using S-R flip-flop.

JK(Available) to T (Target) Conversion

Conversion Table

Input	Present state	Next state	Flip flop Inputs	
T	Q_n	Q_{n+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 SIMPLIFICATION

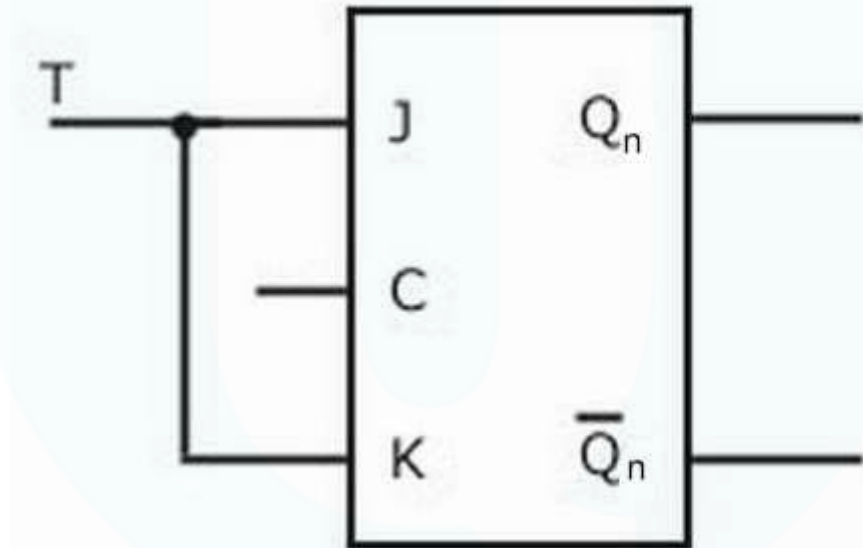
Q_n	0	1
T	0	1
0	0	X
1	1	X

$$J=T$$

Q_n	0	1
T	0	1
0	X	0
1	X	1

$$K=T$$

Logic Diagram (JK to T)

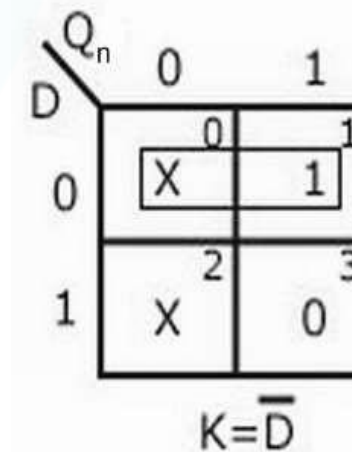
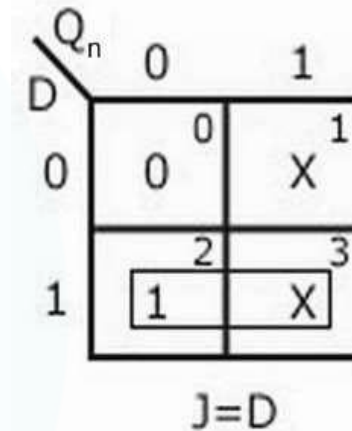


JK(Available) to D(Target)Flip-flop Conversion

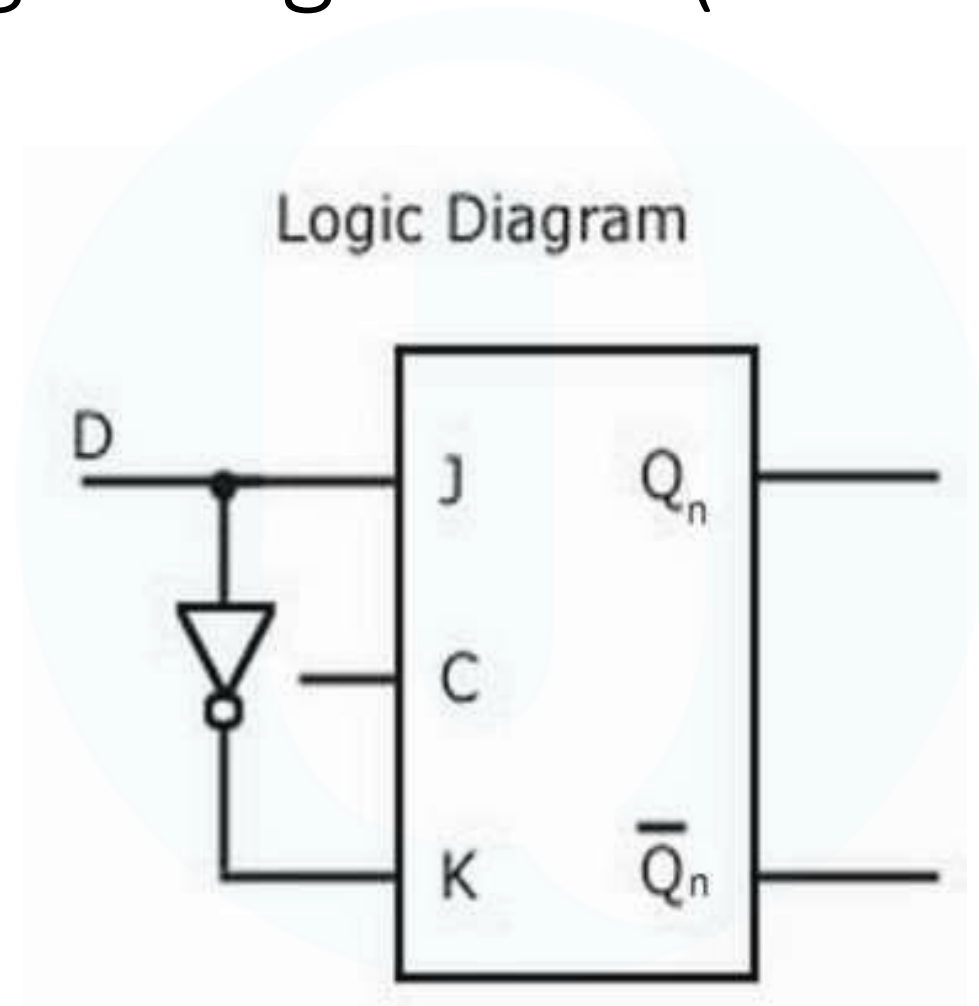
Conversion Table

Input	Present state	Next state	Flip flop Inputs	
D	Q_n	Q_{n+1}	J	K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	x
1	1	1	x	0

K- MAP SIMPLIFICATION



Logic Diagram (JK to D)



D(Available) to T(Target)Flip-Flop

Conversion Table

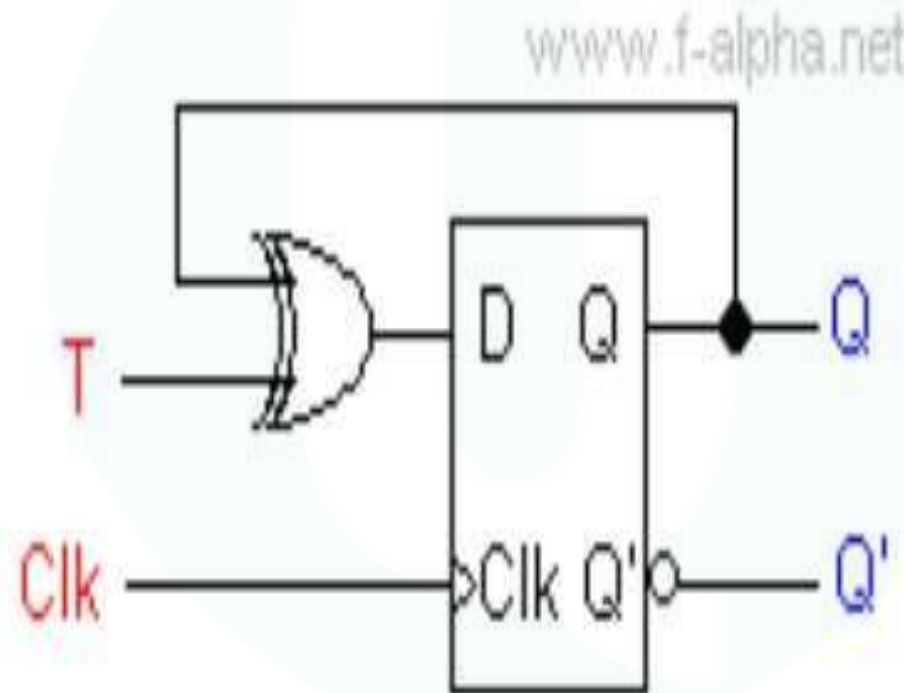
Input	Present state	Next state	Flip flop Inputs
T	Q_n	Q_{n+1}	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

K- MAP SIMPLIFICATION

	Q_n	
	0	1
T		
0	0	1
1	1	0

$$D = T'Q_n + TQ_n'$$

Logic Diagram(D to T)



T (Available) to D(Target) Flip-flop Conversion

Conversion Table

Input	Present state	Next state	Flip flop Inputs
D	Q_n	Q_{n+1}	T
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

K- MAP SIMPLIFICATION

		Qn	
		0	1
D:	0	0	1
	1	1	0

$$T = DQ_n' + D'Q_n$$

JK(Available) to SR(Target) Flip-flop conversion

Conversion Table

Input		Present State	Next State	Flip-Flop Inputs	
S	R	Q _n	Q _{n+1}	J	K
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	X	1
1	0	0	1	1	X
1	0	1	1	X	0
1	1	0	X	X	X
1	1	1	X	X	X

For More Visit : KtuQbank.com | Fair Use Policy

JK(Available) to SR(Target) Flip-flop conversion

SR	Qn	
	0	1
00	0	X
01	0	X
11	X	X
10	1	X

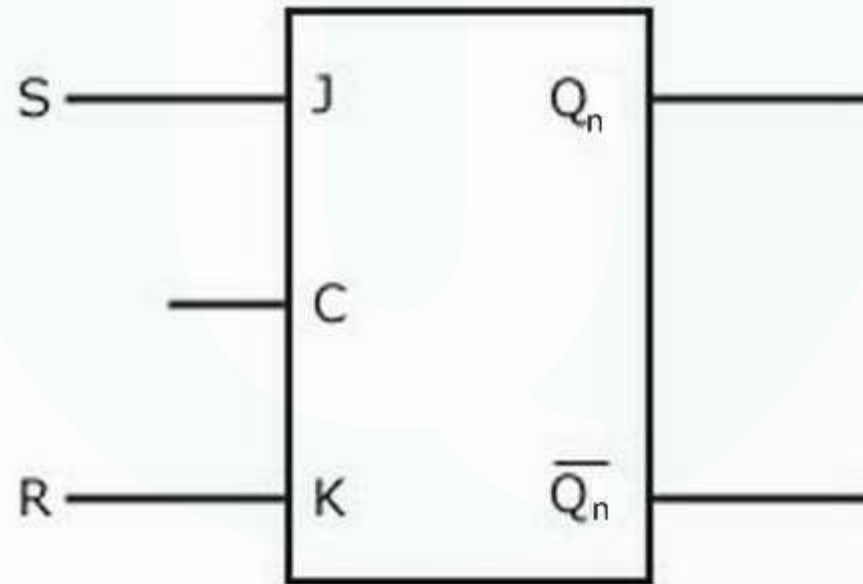
J=S

SR	Qn	
	0	1
00	X	0
01	X	1
11	X	X
10	0	X

K=R

JK to SR

- Logic Diagram



D(Available) to SR(Target) Flip-Flop Conversion

Conversion Table

Input		Present State	Next State	Flip-Flop Inputs	
S	R	Q _n	Q _{n+1}	J	K
0	0	0	0	0	0
0	0	1	1	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	X	X	X
1	1	1	X	X	X

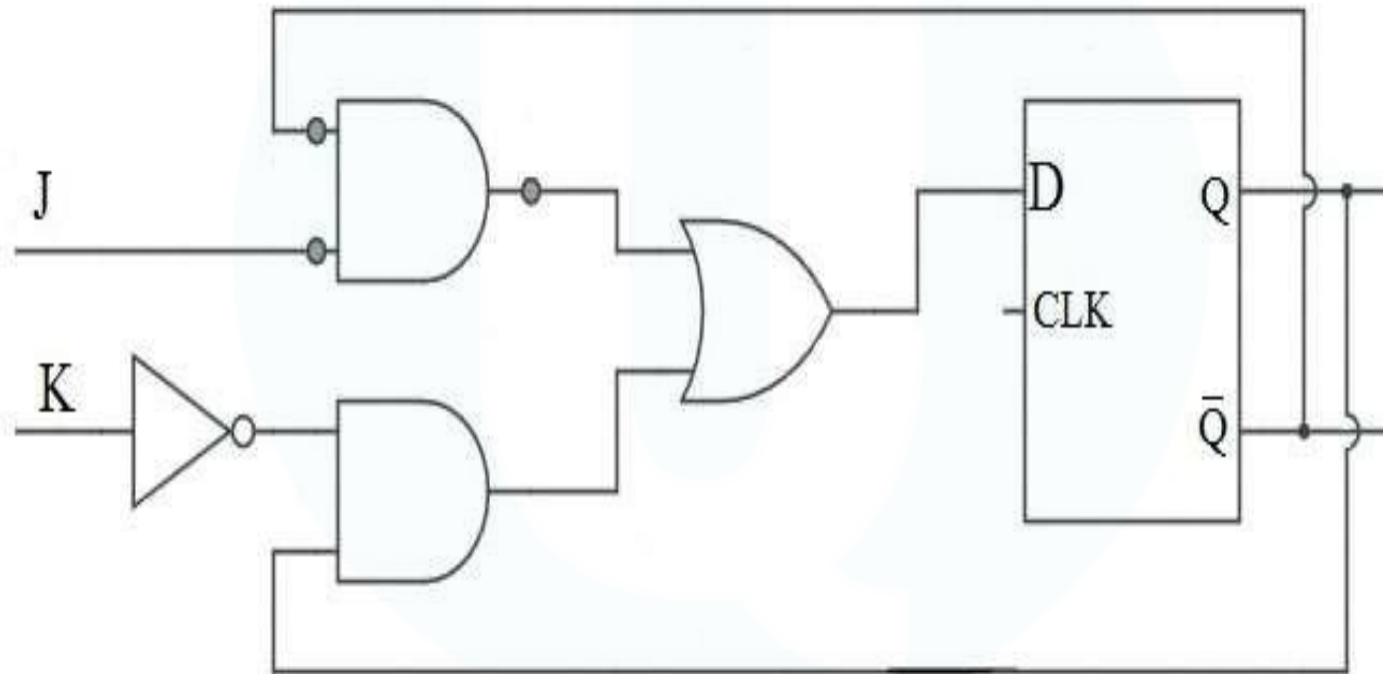
K- MAP SIMPLIFICATION

$\overline{S}R$

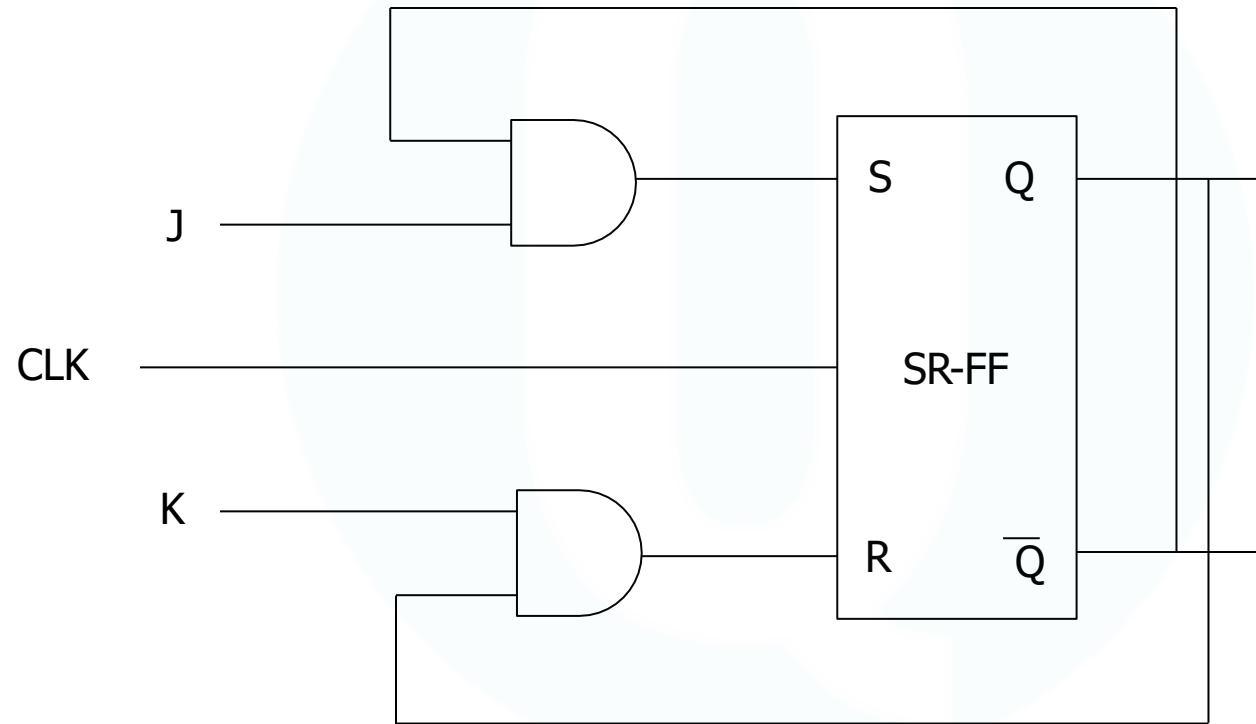
	$R\ Q_n$	00	01	11	10
S	0	0	1	0	0
	1	1	1	X	X

$$D = R'Q_n + S$$

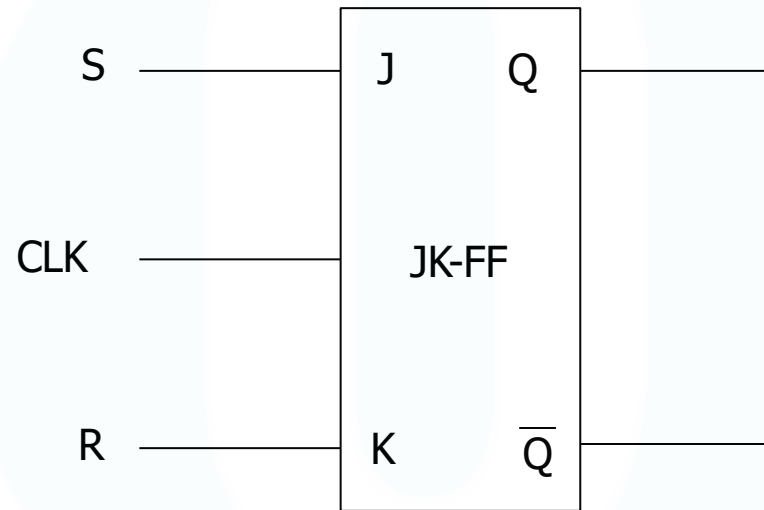
Logic Diagram For D to SR



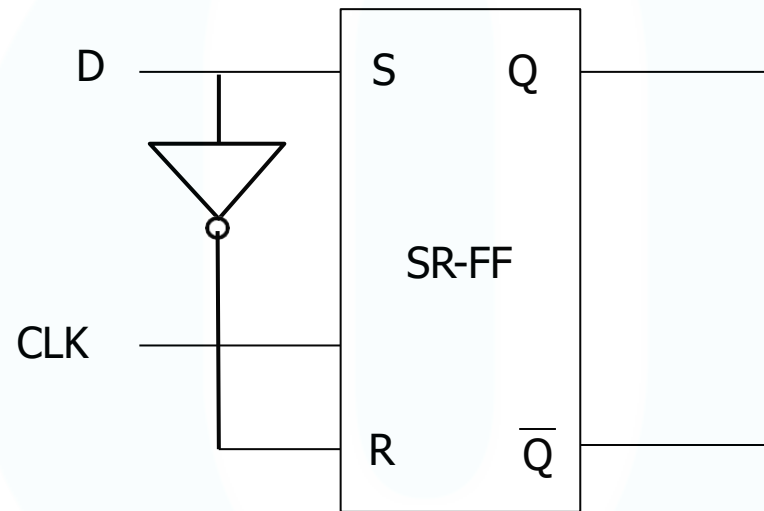
S R Flip to J K Flip Flop:



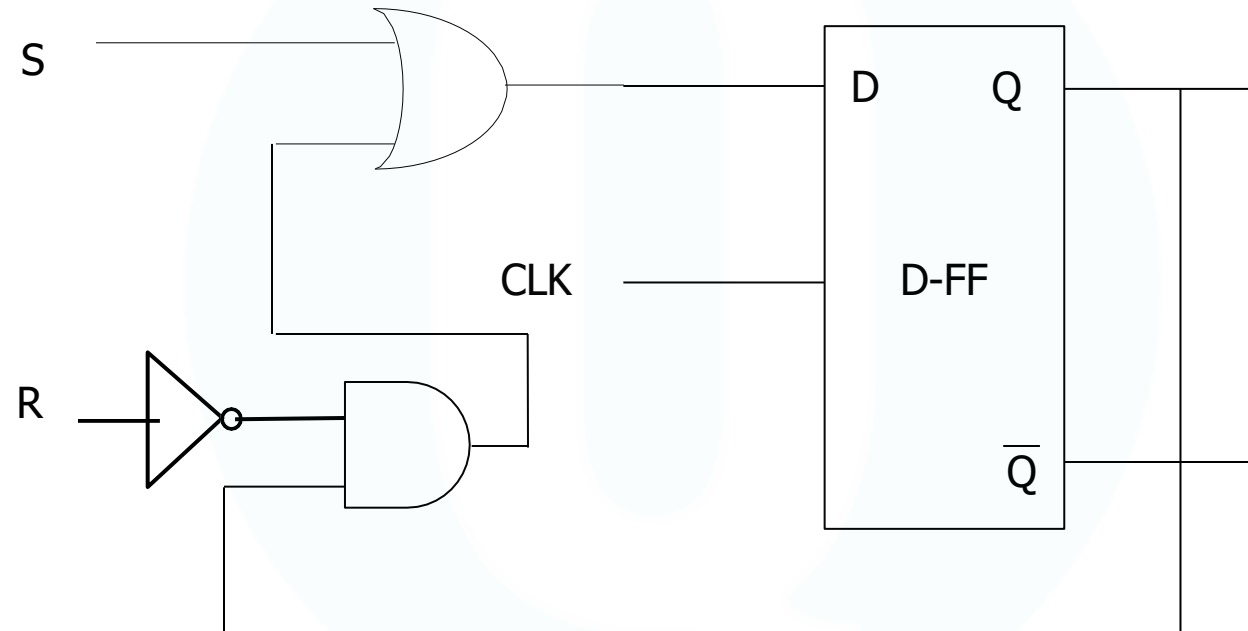
J K Flip to S R Flip Flop:



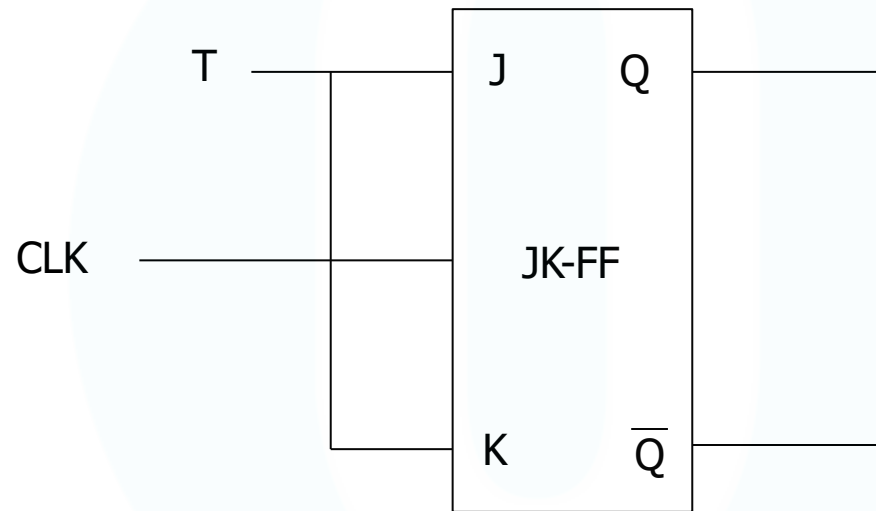
S R Flip to D Flip Flop:



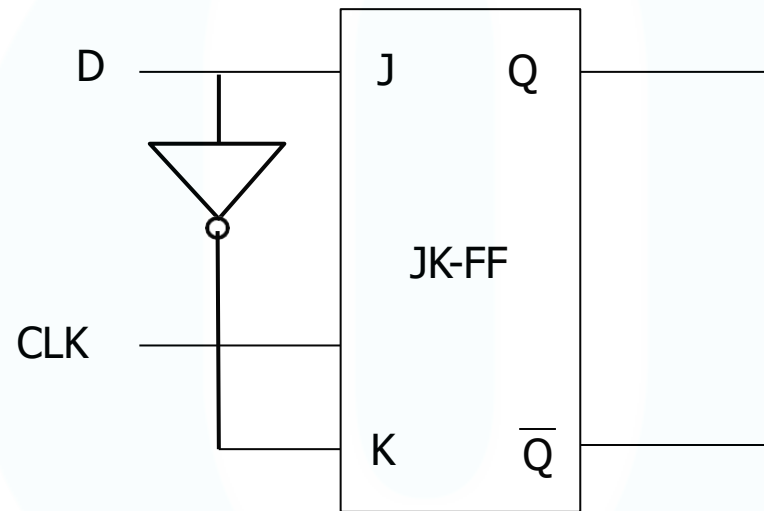
D Flip to S R Flip Flop:



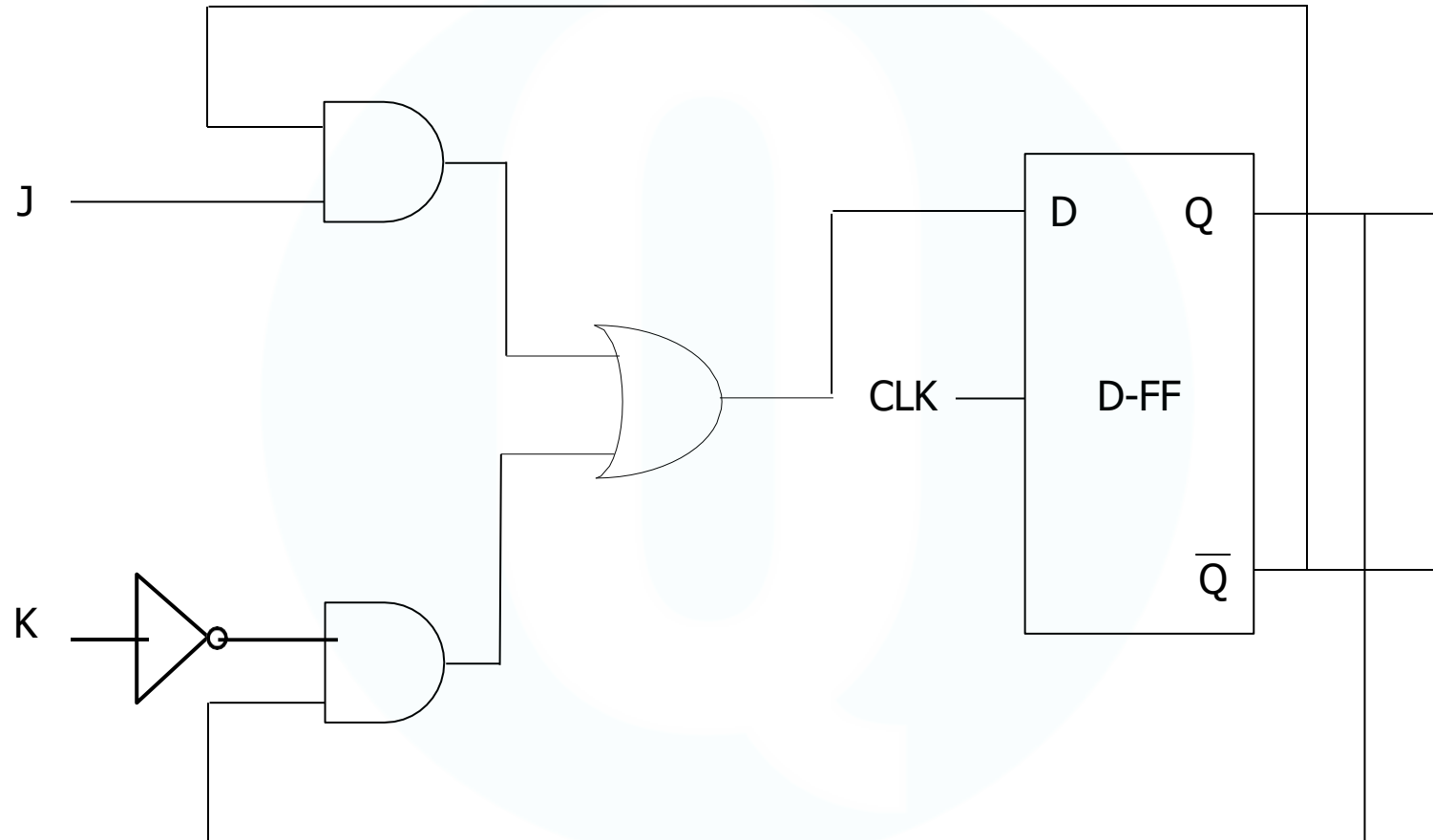
J K Flip to T Flip Flop:



J K Flip to D Flip Flop:



D Flip to J K Flip Flop:



Serial form of Data Vs Parallel Form of Data

- ✓ Data may be available in Parallel form or Serial form.
- ✓ Multi bit data is said to be in parallel form when all the bits are available (accessible) simultaneously.
- ✓ The data is said to be in serial form when data bits appear sequentially (one after another in time) at a single terminal.
- ✓ Data may also be transferred in parallel form or in serial form.

Data Transmission Serial/Parallel

- ✓ Parallel data transfer is the simultaneous transmission of all bits of data from one device to another.
- ✓ Serial data transfer is the transmission of one bit of data at time from one device to another.
- ✓ Serial data must be transmitted under the synchronization of a clock, since clock provides the means to specify the time at which each new bit is sampled

Register

- ✓ As a flip flop can store only one bit of data, a 0 or a 1, it is referred as a single bit register.
- ✓ When more bits of data are to be stored, a number of FFs used.
- ✓ A register is a set of FFs used to store binary data.
- ✓ The storage capacity of a register is the number of bits (1s and 0s) of digital data it can retain.
- ✓ A register may output data either in serial form or in parallel form.

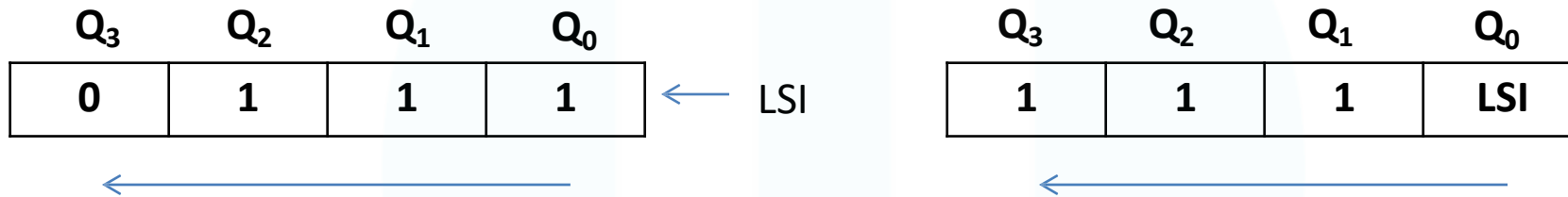
Shift Register

- ✓ A shift register is a very important digital building blocks. It has innumerable applications.
- ✓ Shift registers are a type of logic circuits closely related to counters.
- ✓ They are used basically for storage and transfer of digital data.
- ✓ The basic difference between a shift register and a counter is that, a shift register has no specified sequence of states whereas a counter has a specified sequence of states.

Shift Registers

- Multi-bit register that moves stored data bits left/right (1 bit position per clock cycle)

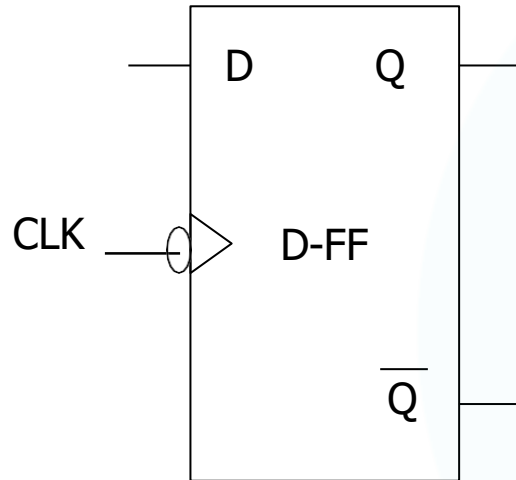
❑ Shift Left is towards MSB



❑ Shift Right (or Shift Up) is towards MSB

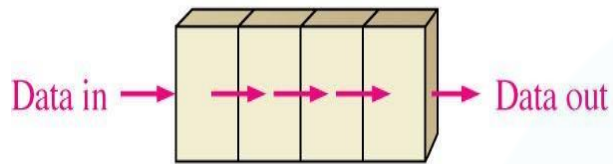


Flip Flop as Storage Element

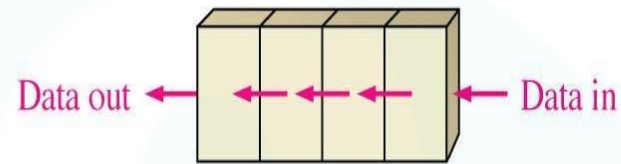


Inputs		Output		Comment
CLK	D	Q_{n+1}	$\overline{Q_{n+1}}$	
0	X	Q_n	$\overline{Q_n}$	Last Value or No Change
↓	0	0	1	Reset
↓	1	1	0	Set

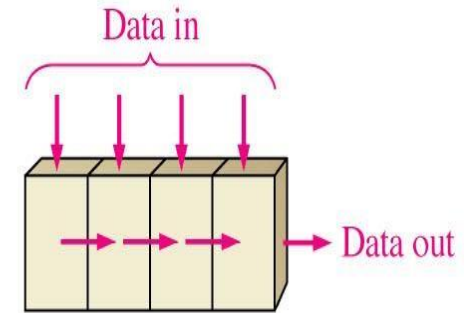
Basic Data Movements in Shift Registers



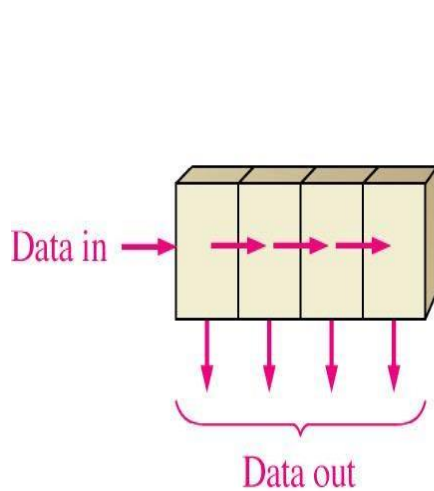
(a) Serial in/shift right/serial out



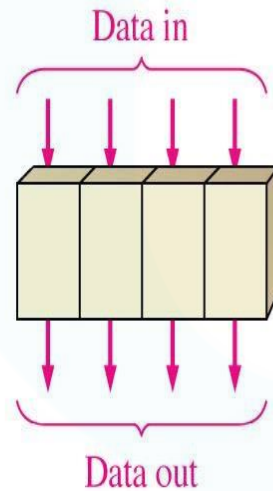
(b) Serial in/shift left/serial out



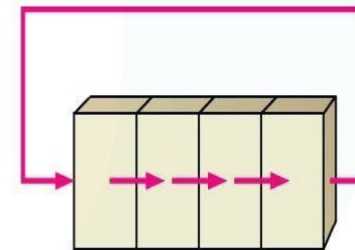
(c) Parallel in/serial out



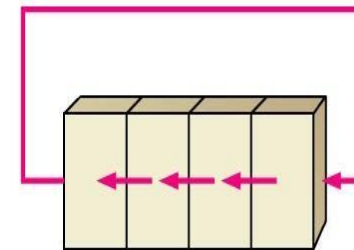
(d) Serial in/parallel out



(e) Parallel in/parallel out



(f) Rotate right

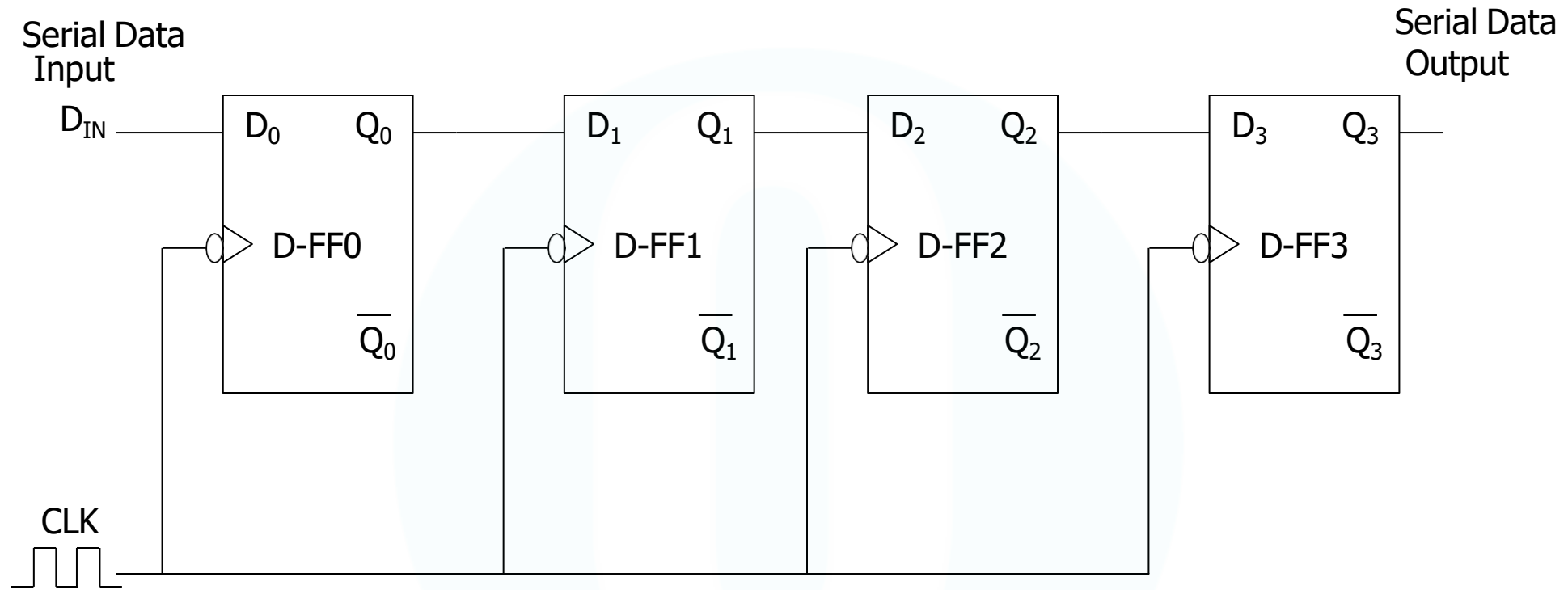


(g) Rotate left

Types of Shift Registers

- ✓ SISO – Serial In Serial Out Shift Register
- ✓ SIPO – Serial In Parallel Out Shift Register
- ✓ PISO – Parallel In Serial Out Shift Register
- ✓ PIPO – Parallel In Parallel Out Shift Register
- ✓ Bi-directional Shift Register
- ✓ Universal Shift Register

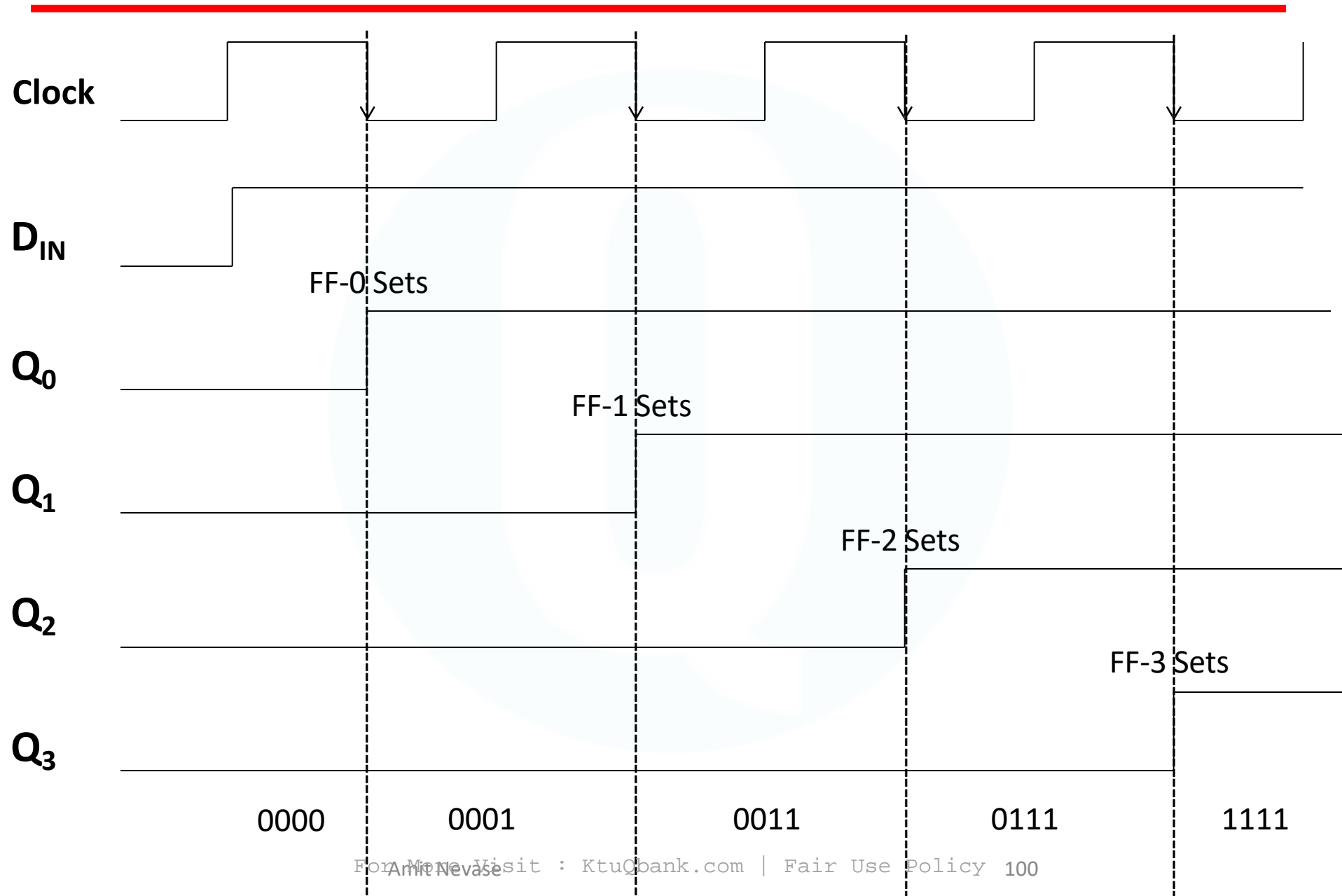
SISO – Serial In Serial Out Shift Register (Shift Left)



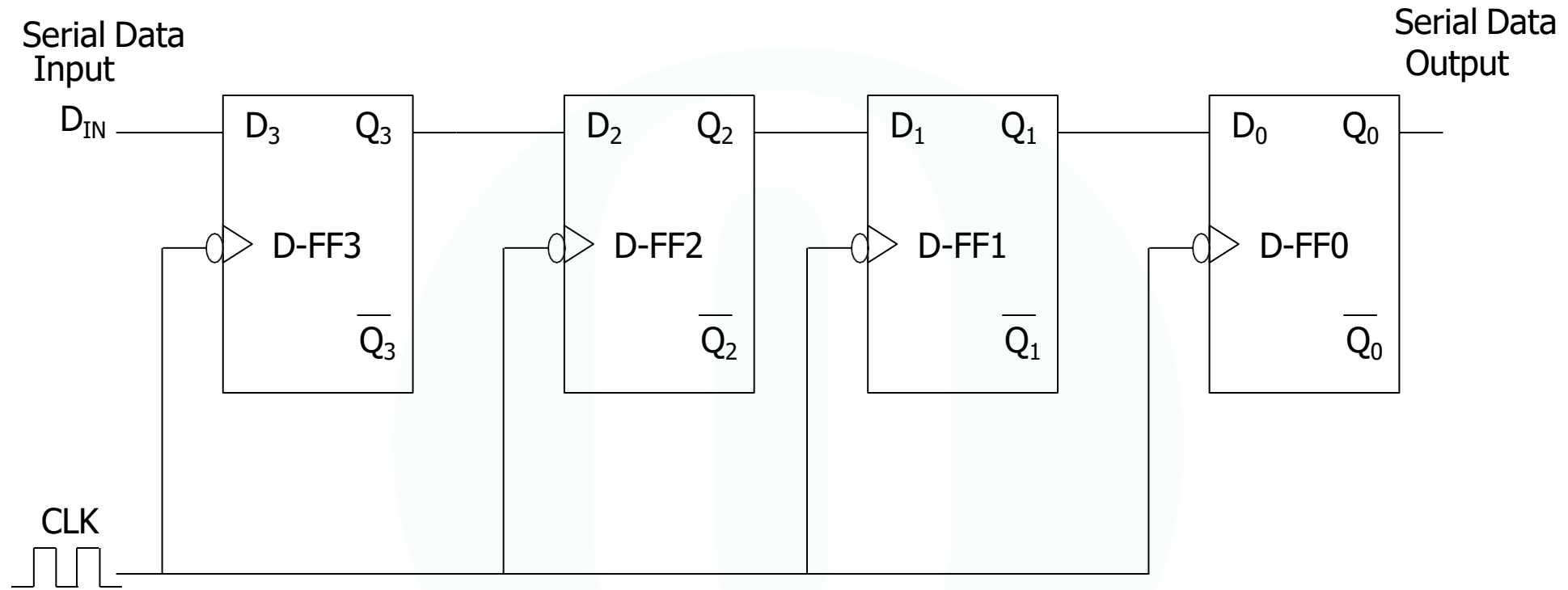
SISO – Serial In Serial Out Shift Register (Shift Left)

CLK	Q_3	Q_2	Q_1	Q_0	Serial Input $D_{IN} = D_0$
Initially	0	0	0	0	
1 st ↓	0	0	0	1	1
2 nd ↓	0	0	1	1	1
3 rd ↓	0	1	1	1	1
4 th ↓	1	1	1	1	1

SISO – Serial In Serial Out Shift Register (Shift Left)



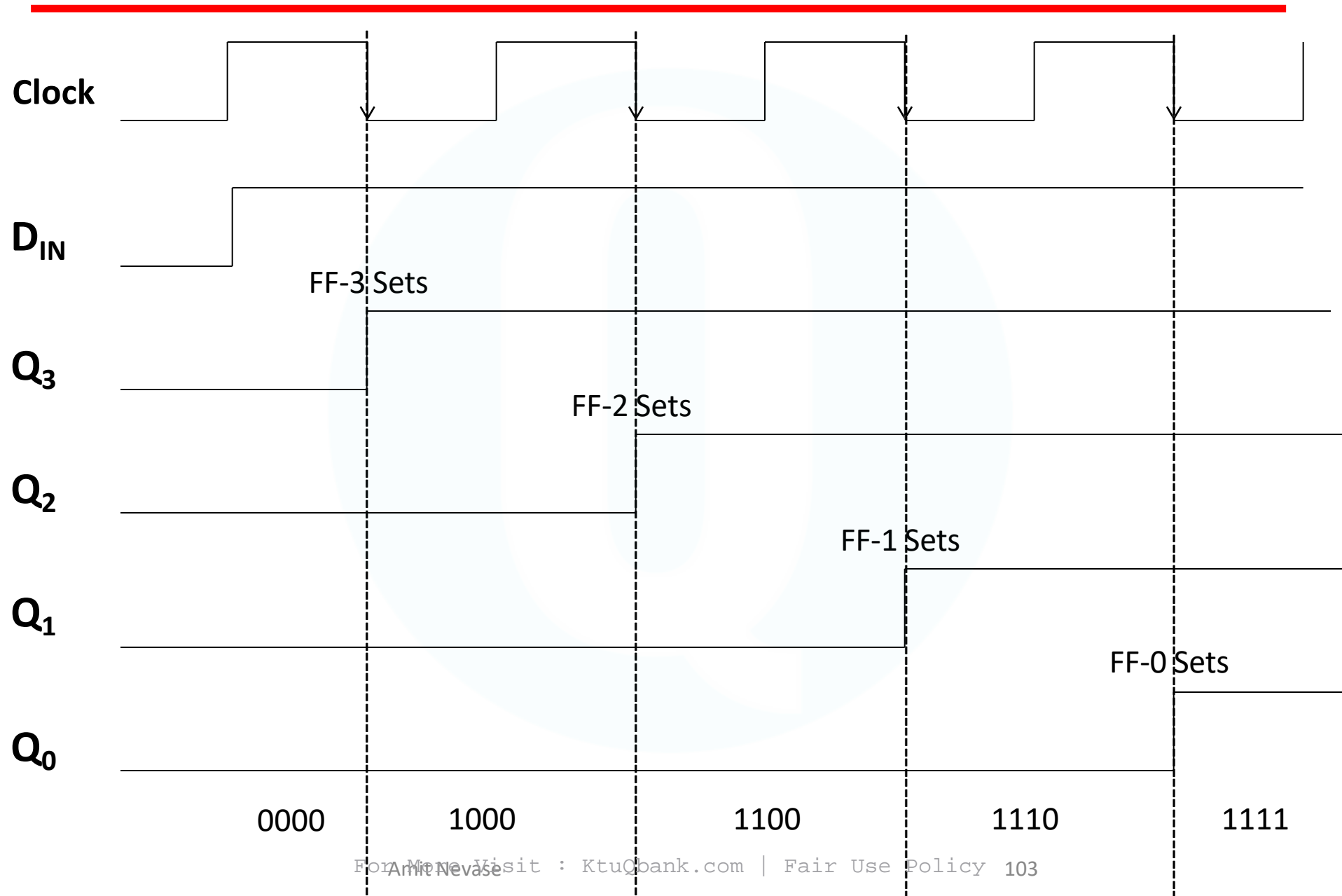
SISO – Serial In Serial Out Shift Register (Shift Right)



SISO – Serial In Serial Out Shift Register (Shift Right)

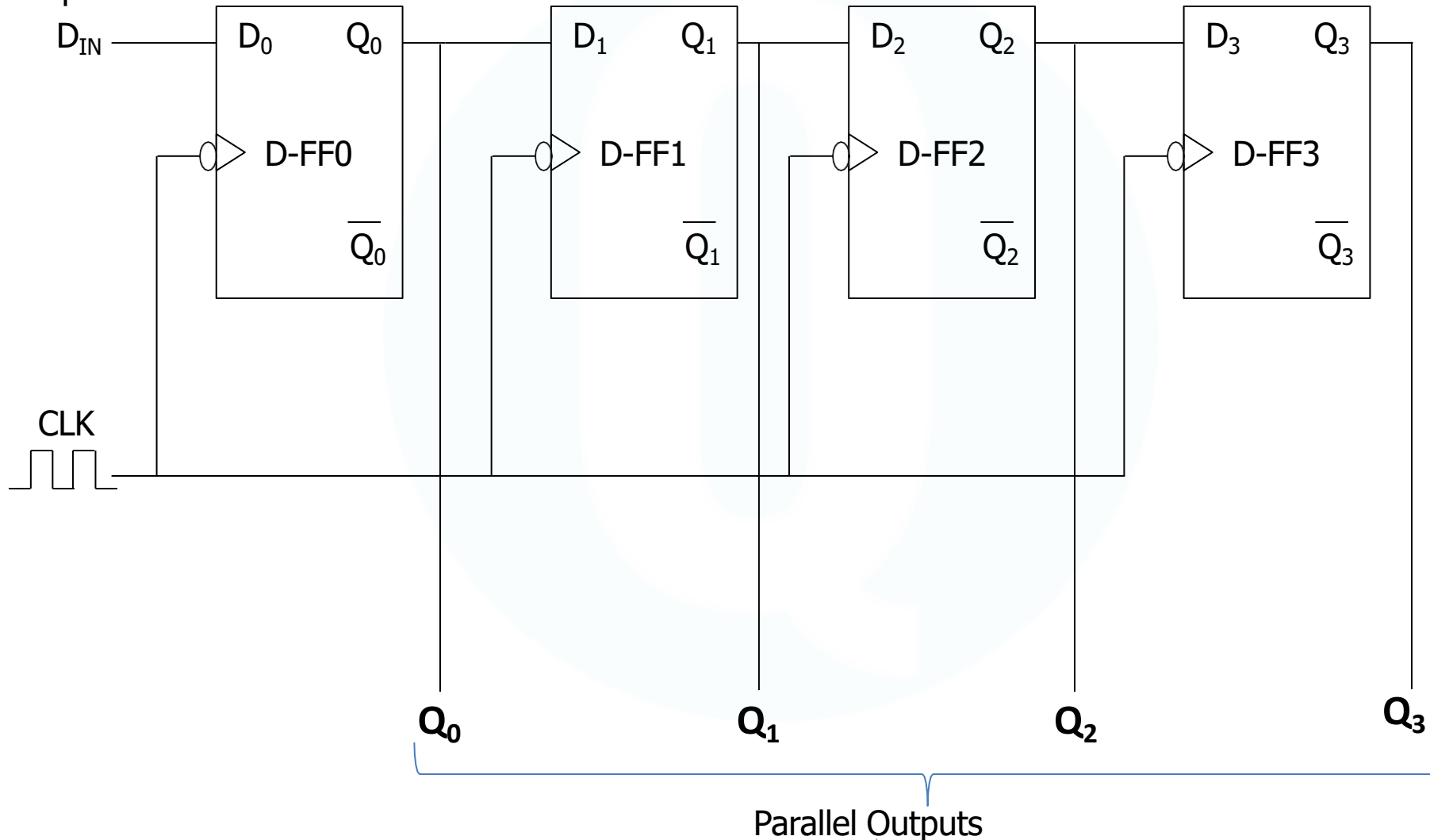
CLK	Serial Input $D_{IN} = D_0$	Q_3	Q_2	Q_1	Q_0
Initially		0	0	0	0
1 st ↓	1 →	1	0	0	0
2 nd ↓	1 →	1	1	0	0
3 rd ↓	1 →	1	1	1	0
4 th ↓	1 →	1	1	1	1

SISO – Serial In Serial Out Shift Register (Shift Right)



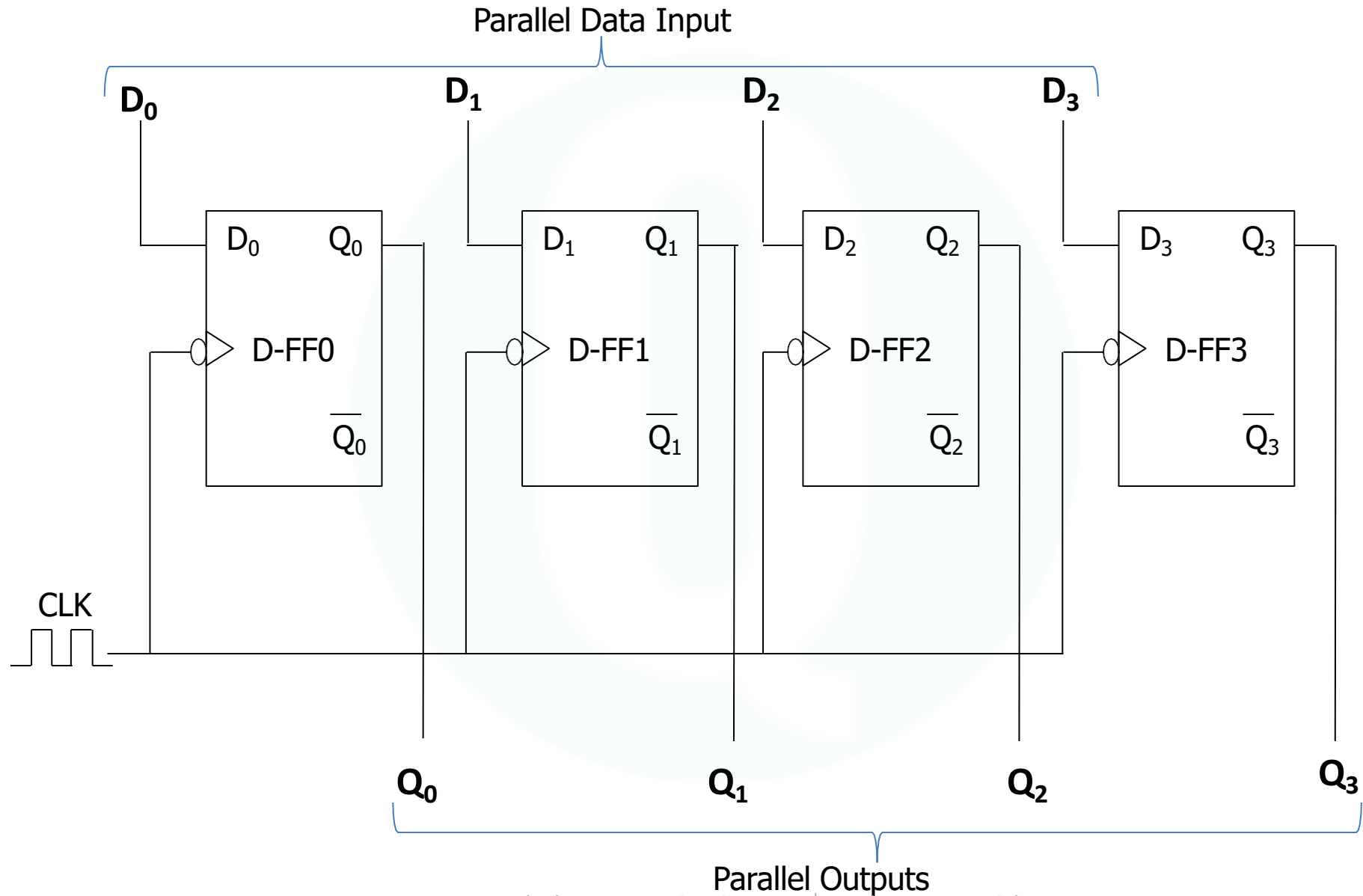
SIPO – Serial In Parallel Out Shift Register

Serial Data
Input

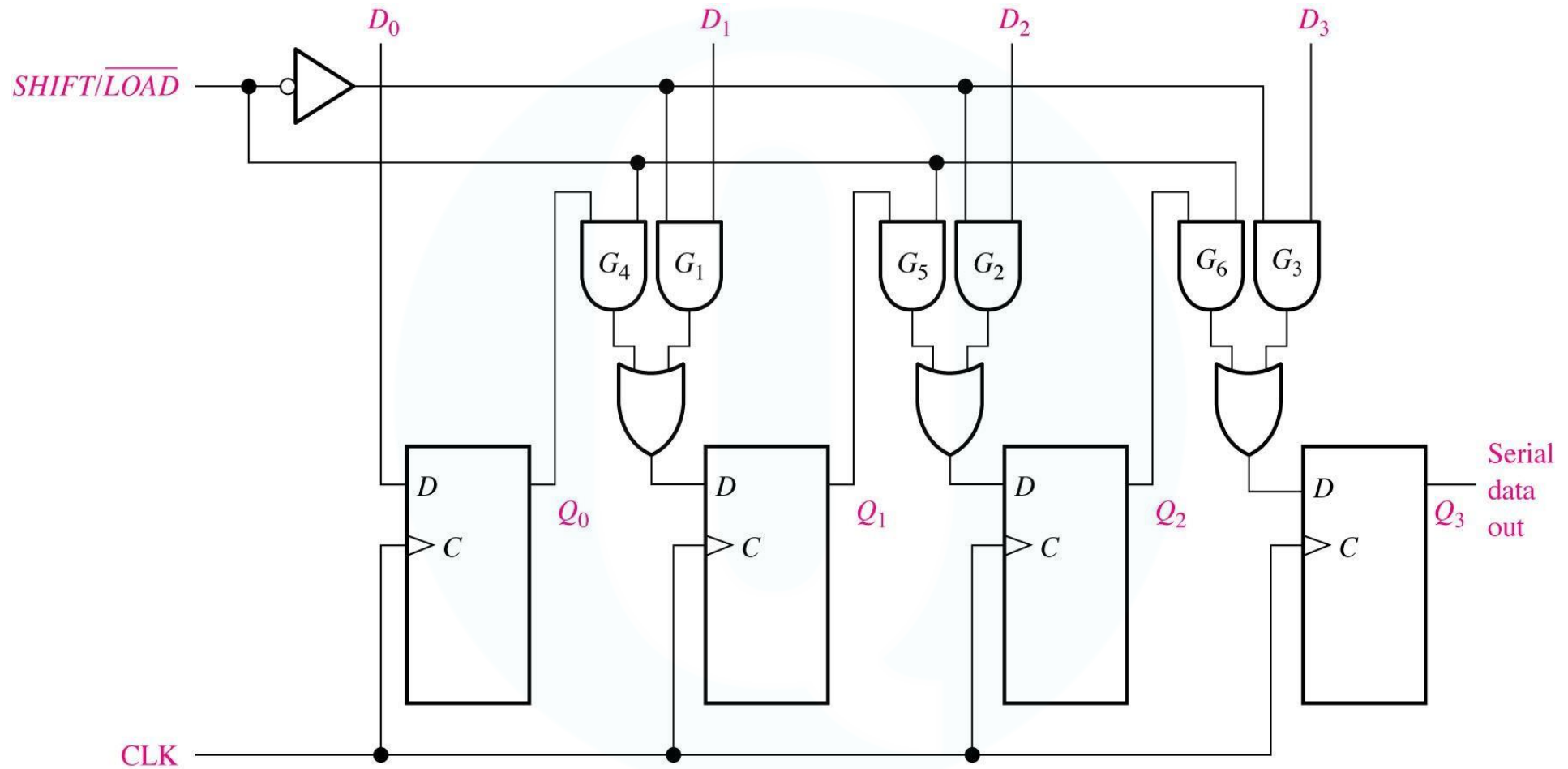


Parallel Outputs

PIPO – Parallel In Parallel Out Shift Register



PISO – Parallel In Serial Out Shift Register



Load Mode:

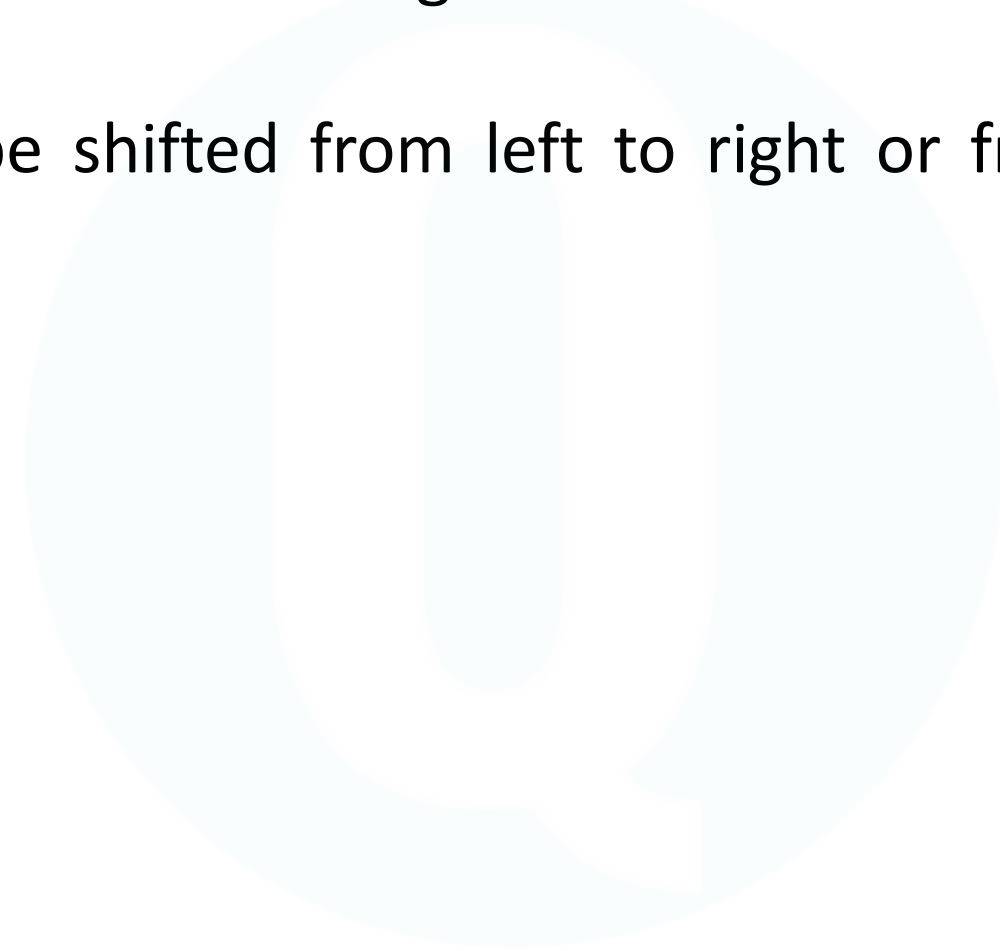
- ✓ When the *Shift / \overline{Load}* line is Low, the AND gates G1, G2 and G3 become active. They will pass D1, D2, and D3 bits to the corresponding Flip Flops.
- ✓ On the low going edge of clock, the binary inputs D0, D1, D2 and D3 will get loaded into corresponding flip flops. Thus parallel loading takes place.

Shift Mode:

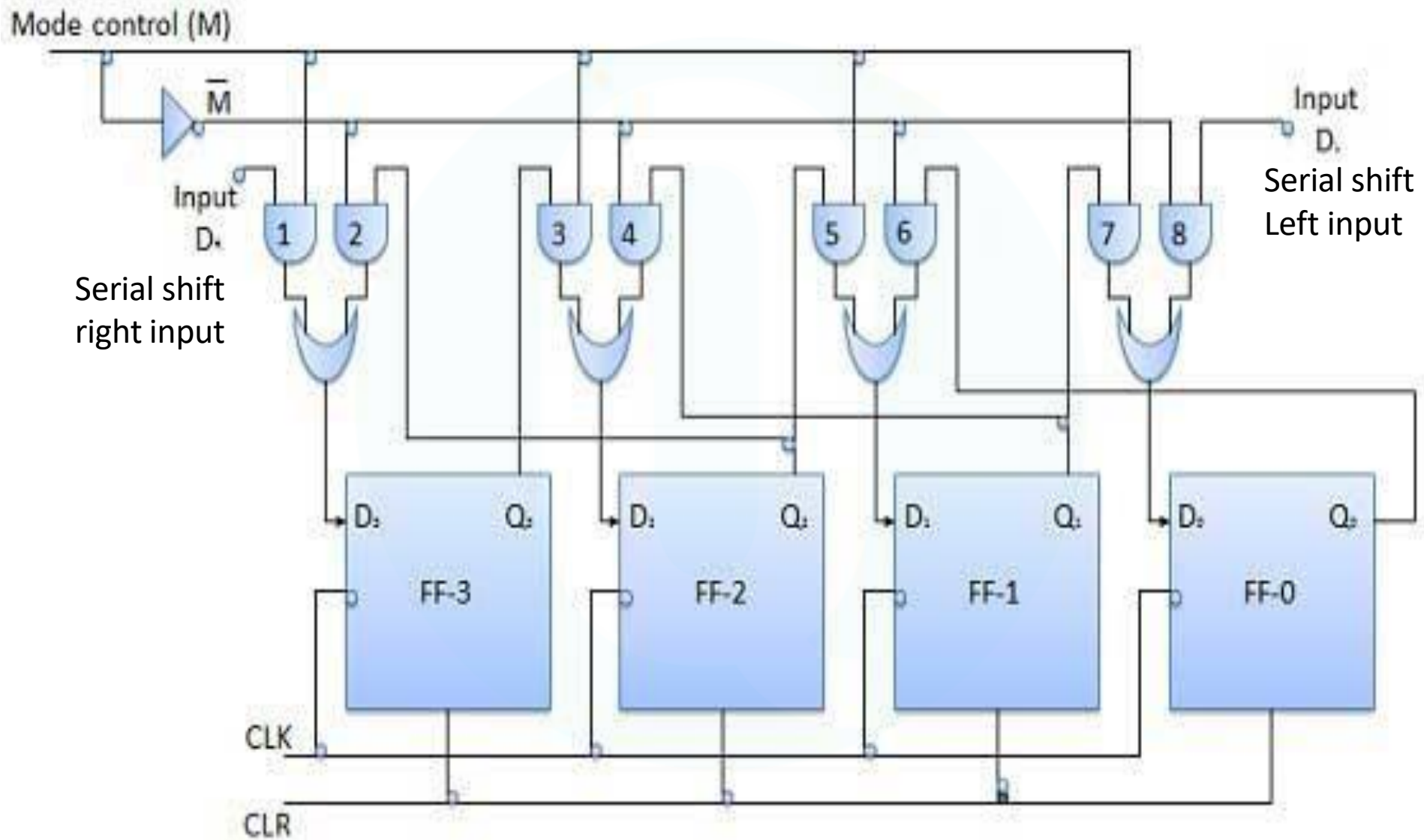
- ✓ When the *Shift / \overline{Load}* line is High, the AND gates G1, G2 and G3 become inactive. Hence parallel loading of data becomes impossible.
- ✓ But AND gates G4, G5 and G6 become active. Therefore the shifting of data from left to right bit by bit on application of clock pulses
- ✓ Thus the parallel in serial out operation takes place

4 Bit Bi-directional Shift Register

- ✓ A bi directional shift register is one in which the data bits can be shifted from left to right or from right to left.



4 Bit Bi-directional Shift Register



4 Bit Bi-directional Shift Register

With $M = 1$: Shift Right Operation

- ✓ If $M=1$, then the AND gates 1,3,5 and 7 are enabled whereas the remaining AND gates 2,4,6 and 8 will be disabled.
- ✓ Hence the data at shift right input is shifted to right bit by bit from FF-3 to FF-0 on the application of clock pulses.
- ✓ Thus with $M=1$ we get the serial right shift operation.

4 Bit Bi-directional Shift Register

With $M = 0$: Shift Left Operation

- ✓ If $M=0$, then the AND gates 2,4,6 and 8 are enabled whereas the remaining AND gates 1,3,5 and 7 will be disabled.
- ✓ Hence the data at shift left input is shifted to left bit by bit from FF-0 to FF-3 on the application of clock pulses.
- ✓ Thus with $M=0$ we get the serial left shift operation.

4 Bit Universal Shift Register

- ✓ A register capable of shifting in one direction only is a **unidirectional shift register**.
- ✓ One that can shift in both directions is a **bidirectional shift register**.
- ✓ If the register has both shifts and parallel load capabilities, it is referred to as a “**Universal Shift Register**”.
- ✓ So universal shift register is a bidirectional shift register, whose input can be either in serial form or in parallel form and whose output also can be either in serial form or parallel form.

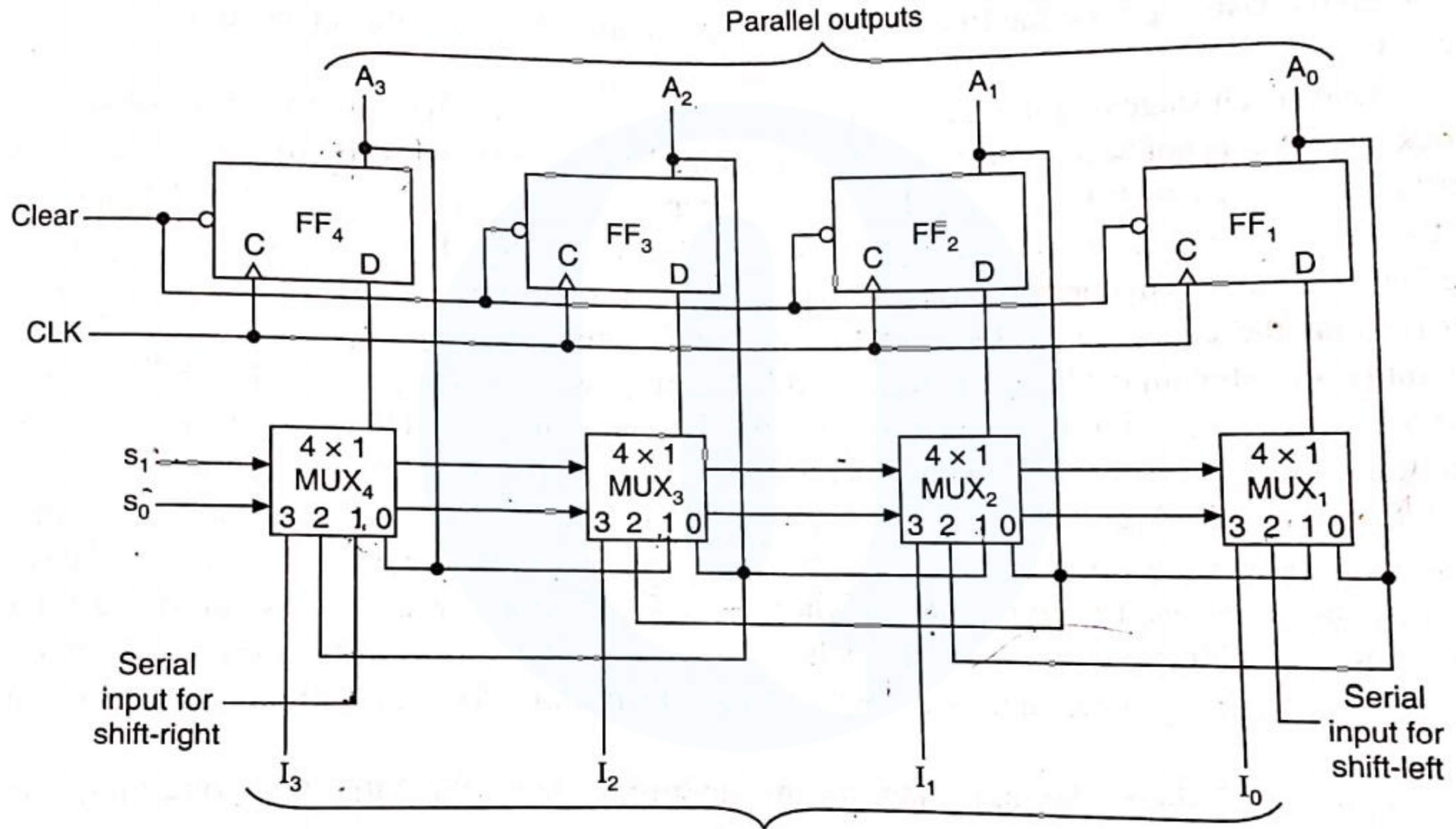


Figure 11.11 4-bit universal shift register.

4 Bit Universal Shift Register

Mode control		
S_1	S_0	Register operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

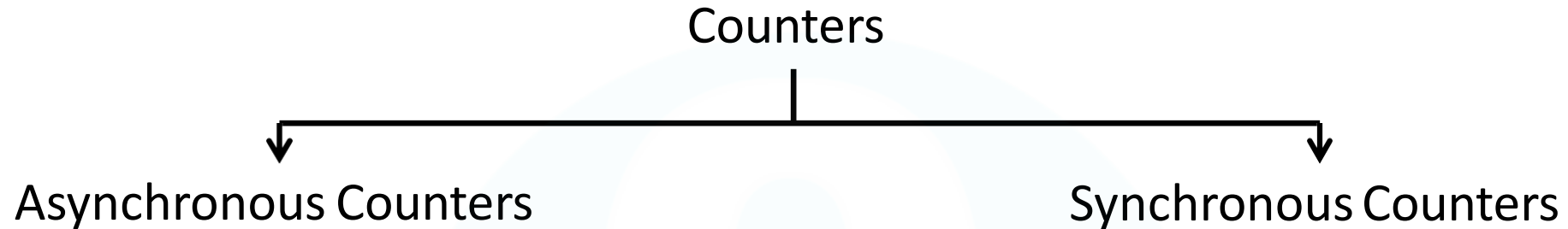
A universal shift register can be realized using multiplexers. Figure 11.11 shows the logic diagram of a 4-bit universal shift register that has all the capabilities listed above. It consists of four D flip-flops and four multiplexers. The four multiplexers have two common selection inputs S_1 and S_0 . Input 0 in each multiplexer is selected when $S_1S_0 = 00$, input 1 is selected when $S_1S_0 = 01$, and input 2 is selected when $S_1S_0 = 10$ and input 3 is selected when $S_1S_0 = 11$. The selection inputs control the mode of operation of the register according to the function entries in Table 11.1. When $S_1S_0 = 00$, the present value of the register is applied to the D inputs of flip-flops. This condition forms a path from the output of each flip-flop into the input of the same flip-flop. The next clock edge transfers into each flip-flop the binary value it held previously, and no change of state occurs. When $S_1S_0 = 01$, terminal 1 of the multiplexer inputs have a path to the D inputs of the flip-flops. This causes a shift-right operation, with the serial input transferred into flip-flop FF_4 . When $S_1S_0 = 10$,

Applications of Shift Registers

- ✓ For temporary data storage
- ✓ As a delay line
- ✓ Parallel to Serial Converter
- ✓ Serial to Parallel Converter
- ✓ Ring Counter
- ✓ Twisted Ring Counter (Johnson Counter)

- ✓ A digital counter is a set of flip flops whose states changes in response to pulses applied at the input to counter.
- ✓ The FFs are interconnected such that their combined state at any time is the binary equivalent of the total number of pulses that have occurred up to that time.
- ✓ Thus, as its name implies, a counter is used to count pulses.

Types of Counters



- ✓ **Asynchronous or Ripple Counter:** For these counters the external clock signal is applied to one Flip Flop and then the output of preceding flip flop is connected to the clock of the next flip flop.
- ✓ **Synchronous Counter:** In synchronous counters all the flip flops receive the external clock pulse simultaneously.

Asynchronous Vs Synchronous Counter

Asynchronous Counter

- ✓ Output of the preceding Flip Flop is connected to clock of the next Flip Flop.
- ✓ All the Flip Flops are not clocked simultaneously.
- ✓ Logic circuit is simple.

Synchronous Counter

- ✓ There is no connection between output of preceding Flip Flop and Clock of next one.
- ✓ All the Flip Flops receive clock signal simultaneously.
- ✓ With increase in number of states, the logic circuit becomes complicated.

Asynchronous Vs Synchronous Counter

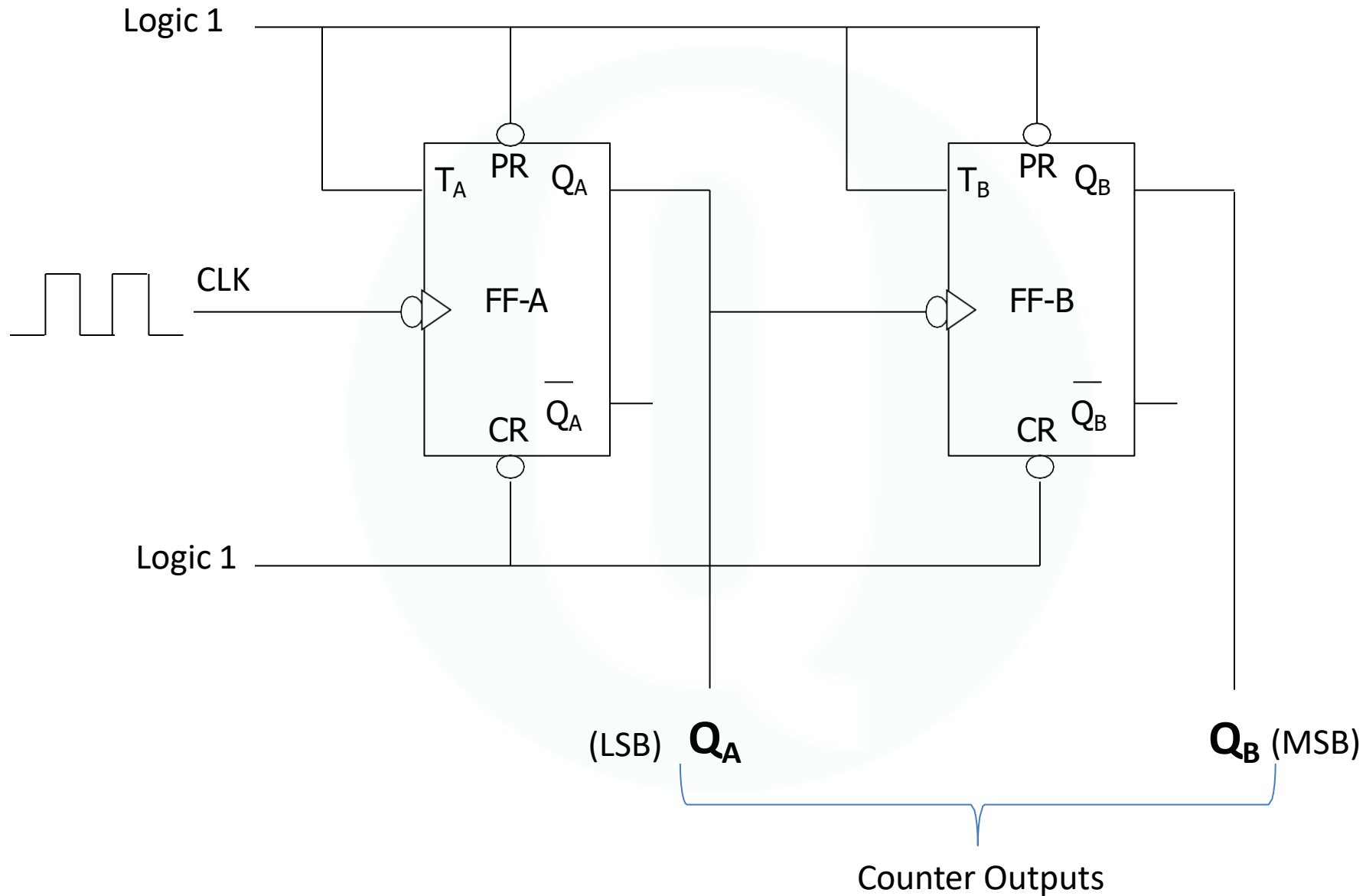
Asynchronous Counter

- ✓ P.D. = $n \times (t_d)$ where n is number of Flip Flops and t_d is propagation delay of flip flop.
- ✓ Frequency of operation is low because of the long propagation delay

Synchronous Counter

- ✓ P.D. = $(t_d)_{FF} + (t_d)_{Gate}$, It is much shorter than that of asynchronous counter.
- ✓ Frequency of operation is high due to shorter propagation delay.

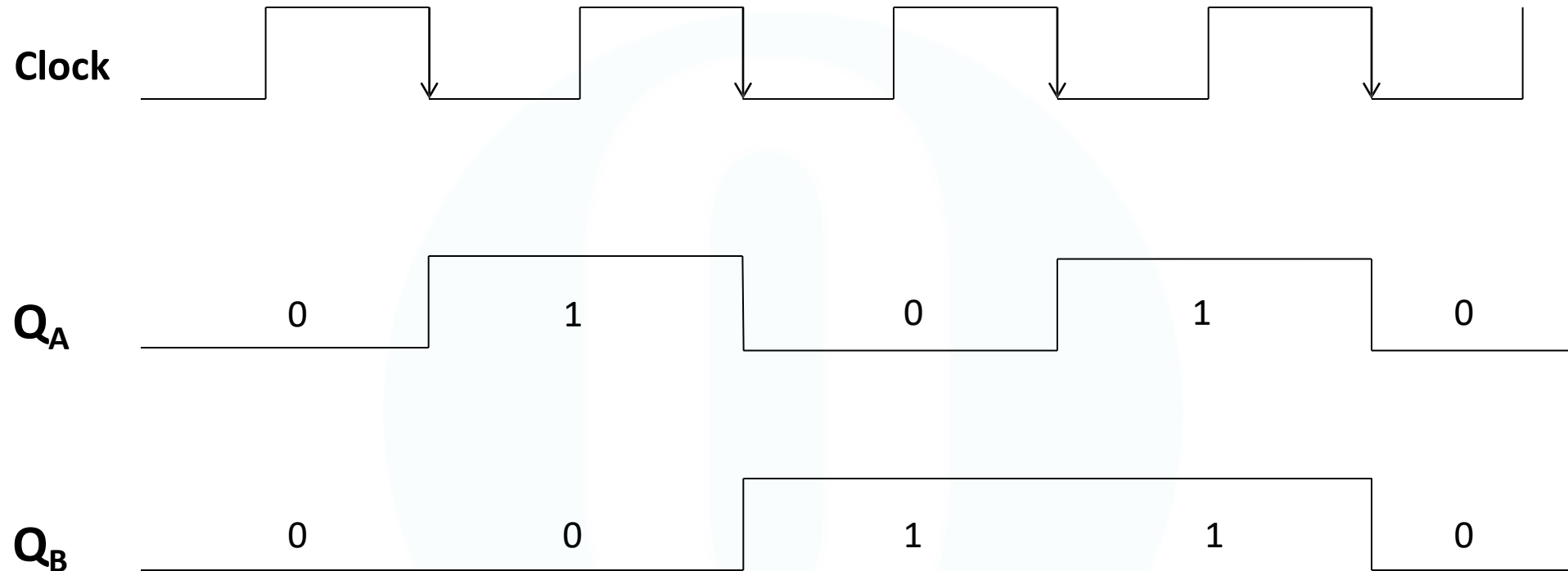
2 Bit Asynchronous or Ripple Up Counter



2 Bit Asynchronous or Ripple Up Counter

Clock	Counter Outputs		State Number	Decimal Equivalent of Counter Output
	Q _B (MSB)	Q _A (LSB)		
Initially	0	0	-	0
1 st ↓	0	1	1	1
2 nd ↓	1	0	2	2
3 rd ↓	1	1	3	3
4 th ↓	0	0	4	0

2 Bit Asynchronous or Ripple Up Counter



Counter O/P

Q _B Q _A	00	01	10	11	00
	0	1	2	3	0

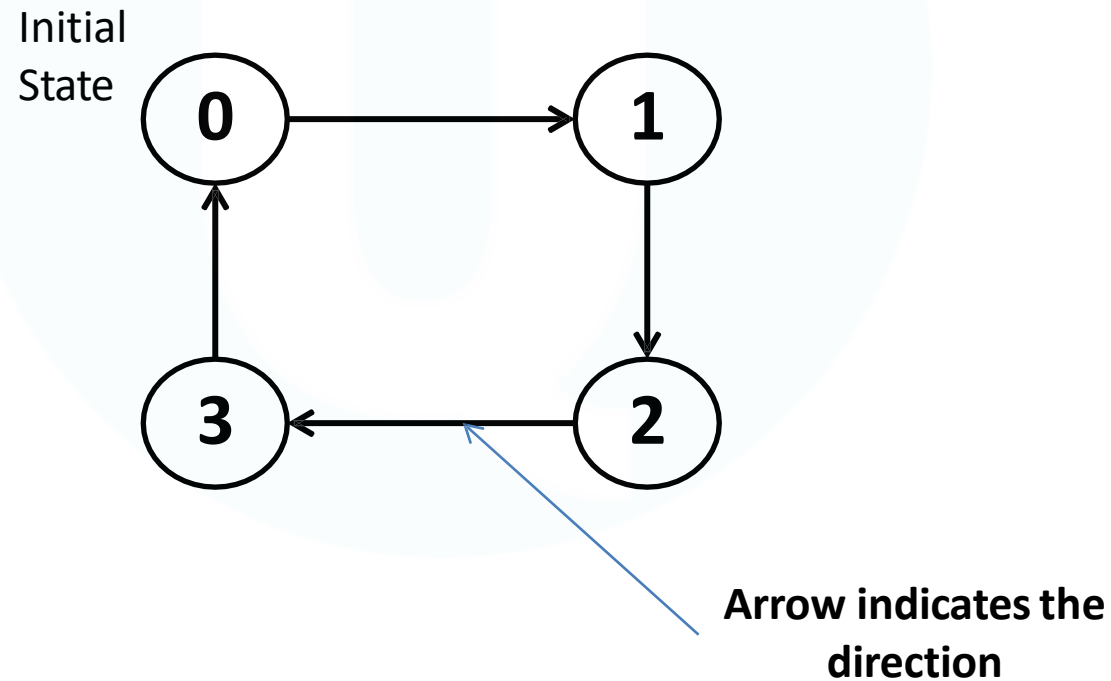
2 Bit Asynchronous or Ripple Up Counter

- ✓ Number of States: 2 Bit ripple counter has four distinct states of outputs namely 00,01,10 and 11.
 - In general the number of states = 2^n , where n is the number of flip flops.

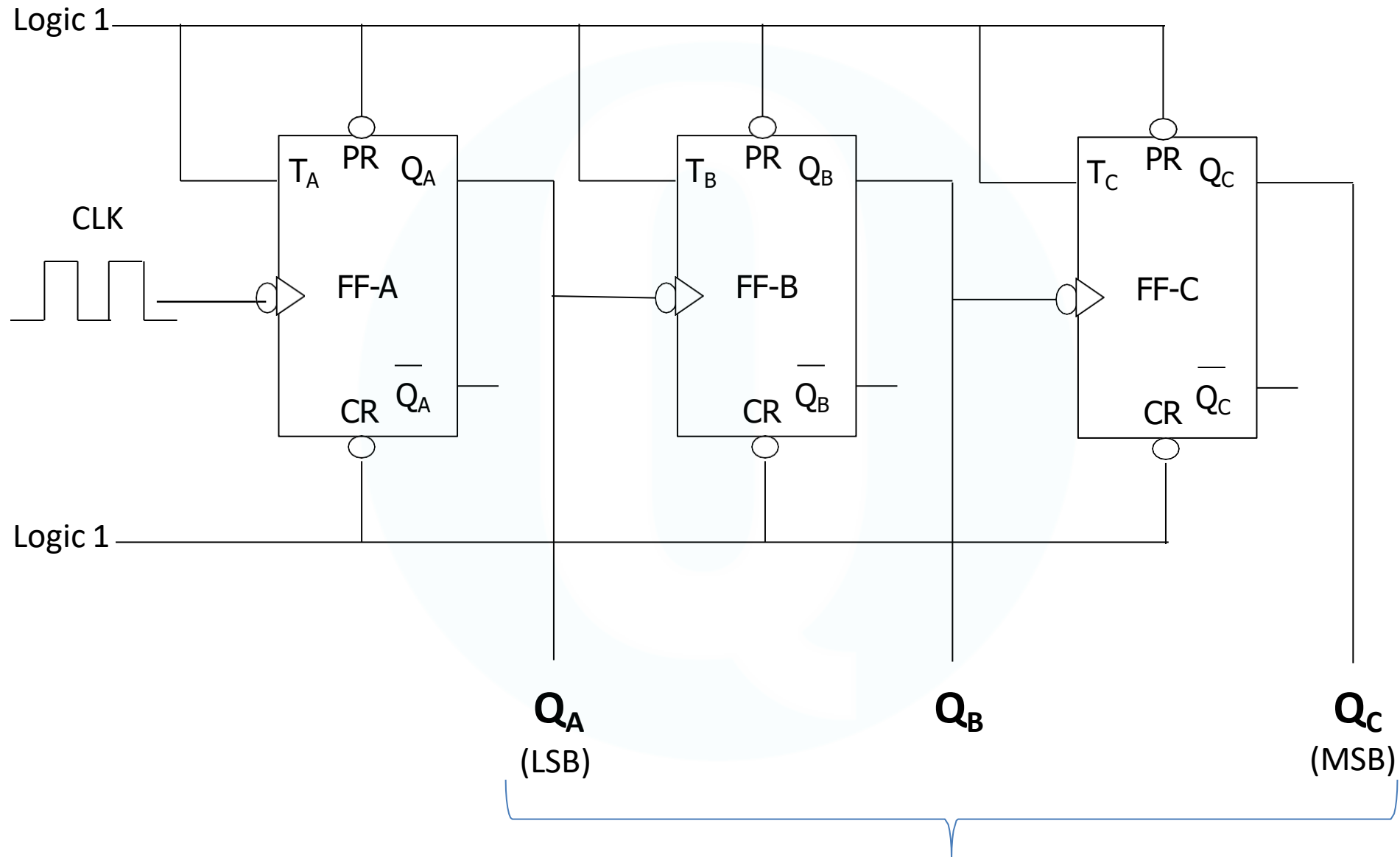
- ✓ Maximum Count is 3 (decimal) i.e. 11 binary
 - In general the maximum count = $(2^n - 1)$

State diagram of 2 bit Ripple Counter

- ✓ The state diagram of a counter represents the states of a counter graphically.
- ✓ For example, for a 2 bit ripple counter the state diagram is shown below.



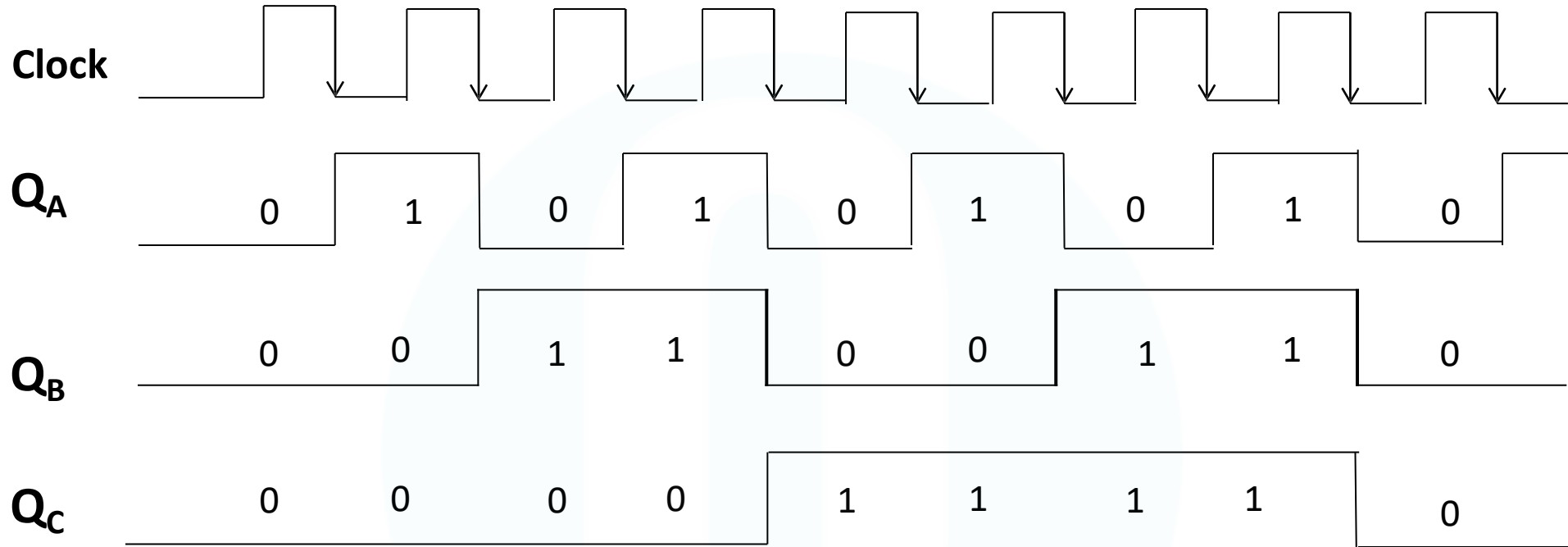
3 Bit Asynchronous or Ripple Up Counter



3 Bit Asynchronous or Ripple Up Counter

Clock	Flip Flop Outputs			State	Decimal Equivalent
	Q _C (MSB)	Q _B	Q _A (LSB)		
Initial	0	0	0	1	0
1 st ↓	0	0	1	2	1
2 nd ↓	0	1	0	3	2
3 rd ↓	0	1	1	4	3
4 th ↓	1	0	0	5	4
5 th ↓	1	0	1	6	5
6 th ↓	1	1	0	7	6
7 th ↓	1	1	1	8	7
8 th ↓	0	0	0	1	0

3 Bit Asynchronous or Ripple Up Counter

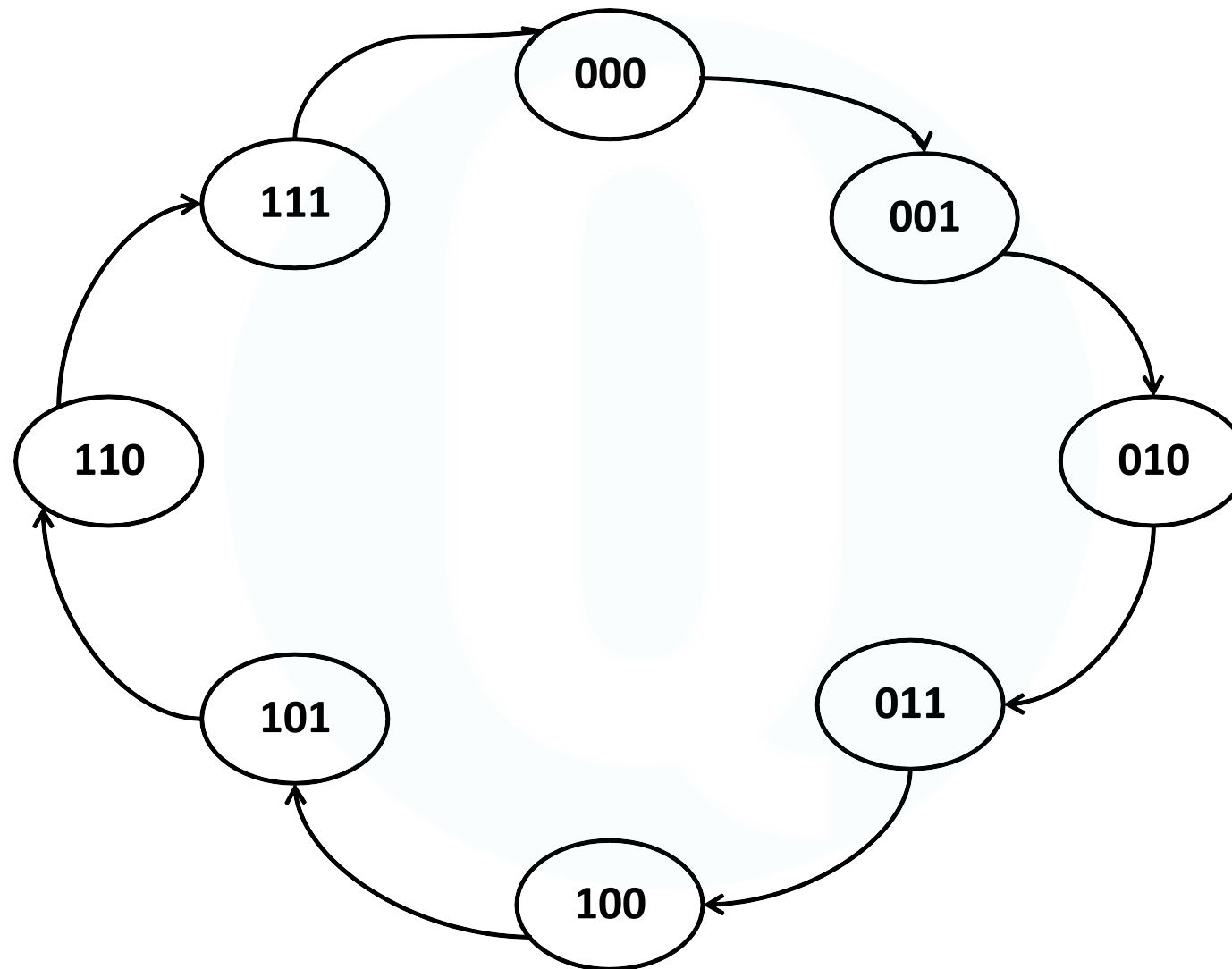


Counter O/P

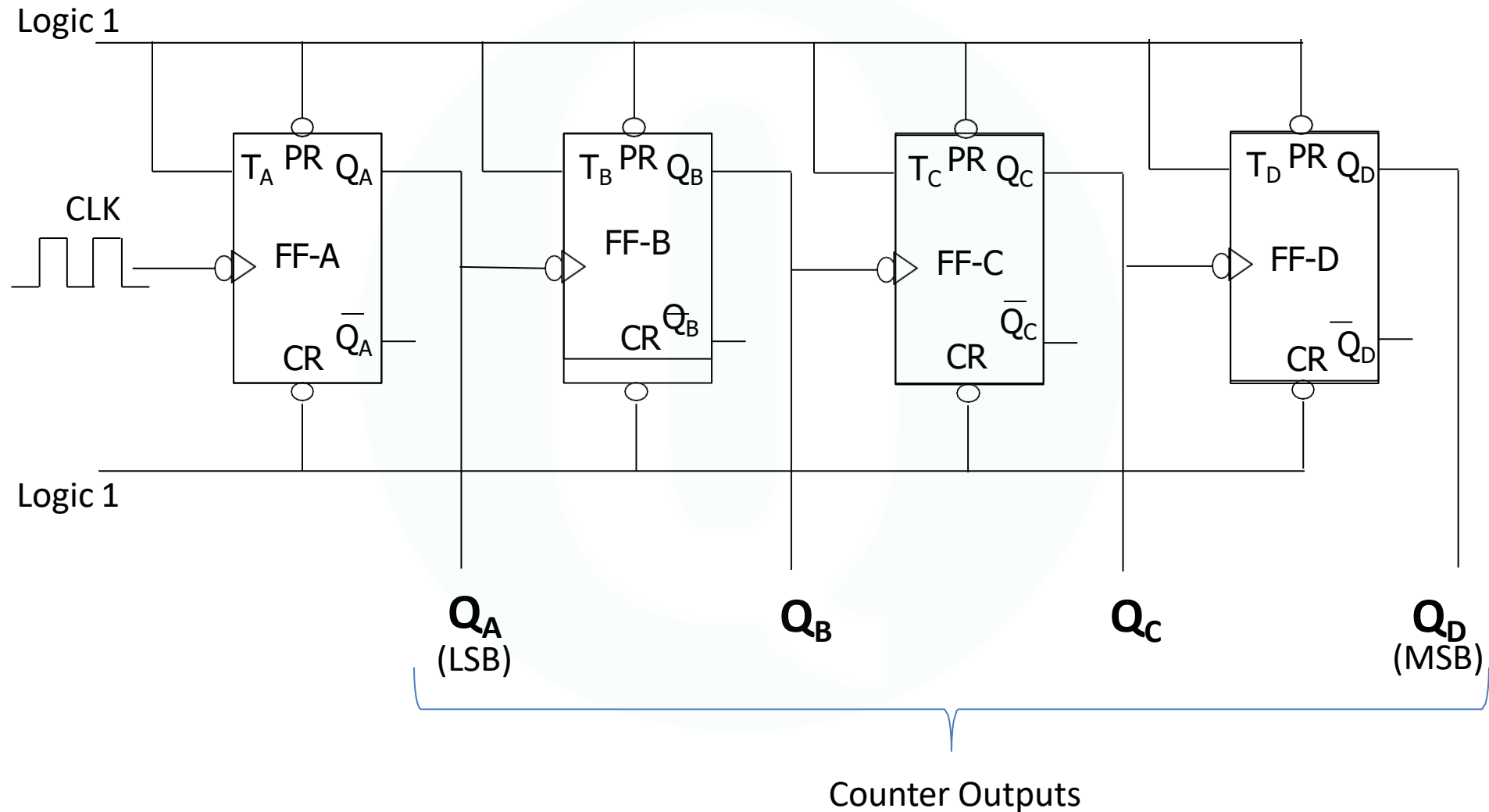
Q_C Q_B Q_A

000	001	010	011	100	101	110	111	000
0	1	2	3	4	5	6	7	0

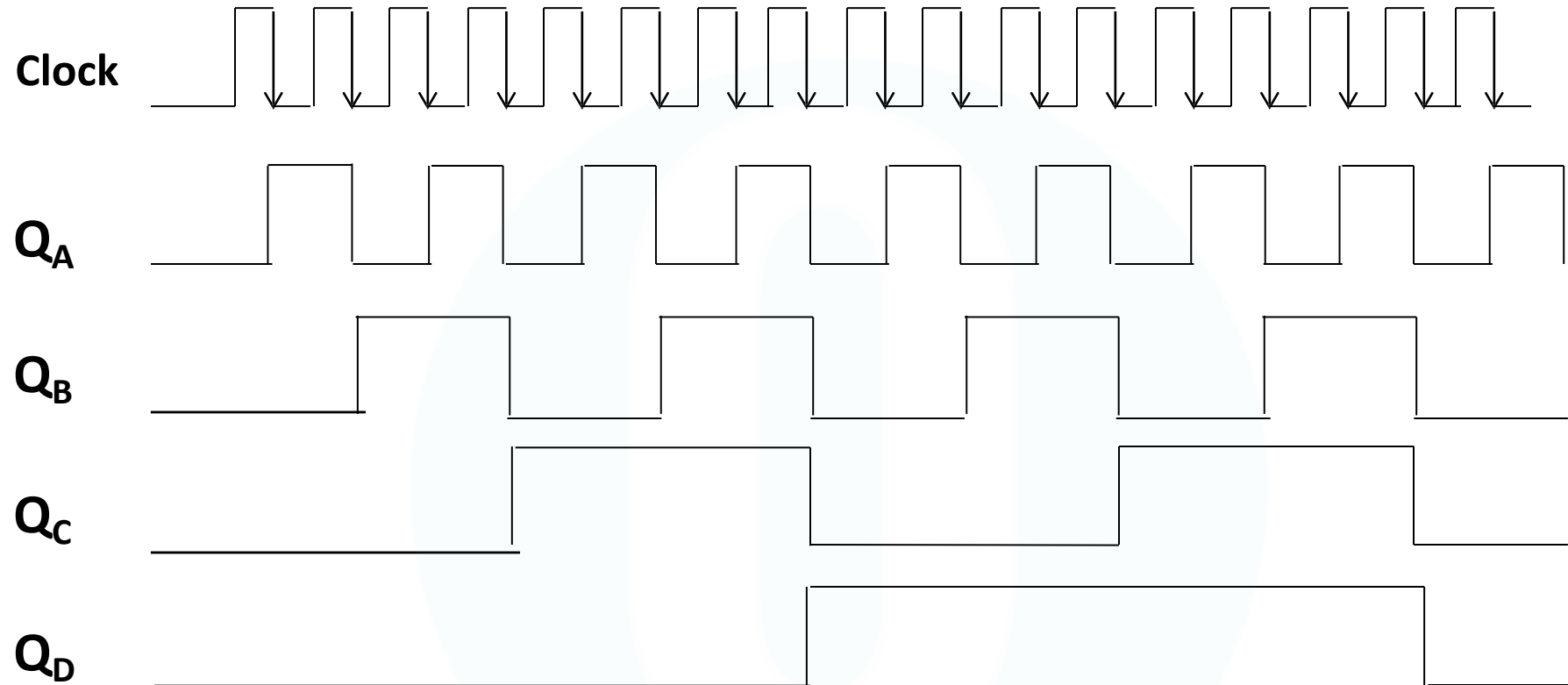
State diagram of 3 bit Asynchronous Up Counter



4 Bit Asynchronous or Ripple Up Counter



4 Bit Asynchronous or Ripple Up Counter

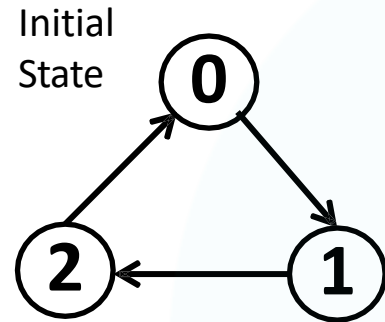


Counter O/P

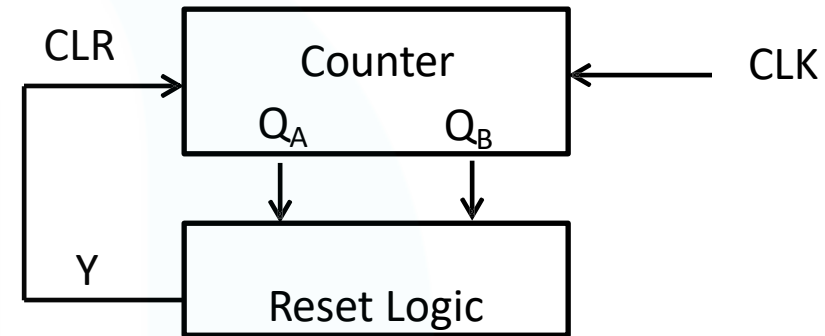
Q _D	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0
Q _C	0	0	0	0	1	1	1	0	0	0	0	1	1	1	1	0
Q _B	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	0
Q _A	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	0
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	00	01	10	11	00	01	10	11	00	01	10	11	00	01	10	11

Design MOD-3 Asynchronous Counter

- ✓ Mod – 3 counter is a counter having three states i.e. 00, 01 and 10. After 10 it will return back to its original state 00.



State Diagram



Block Diagram

- ✓ We have to design the reset logic which is a combinational circuit.
- ✓ The output of reset logic is applied to the clear inputs of flip flops. This is an active Low input.

Design MOD-3 Asynchronous Counter

Truth Table of Reset Logic

FF Outputs		O/P of Reset Logic
Q_B	Q_A	Y
0	0	1
0	1	1
1	0	1
1	1	0

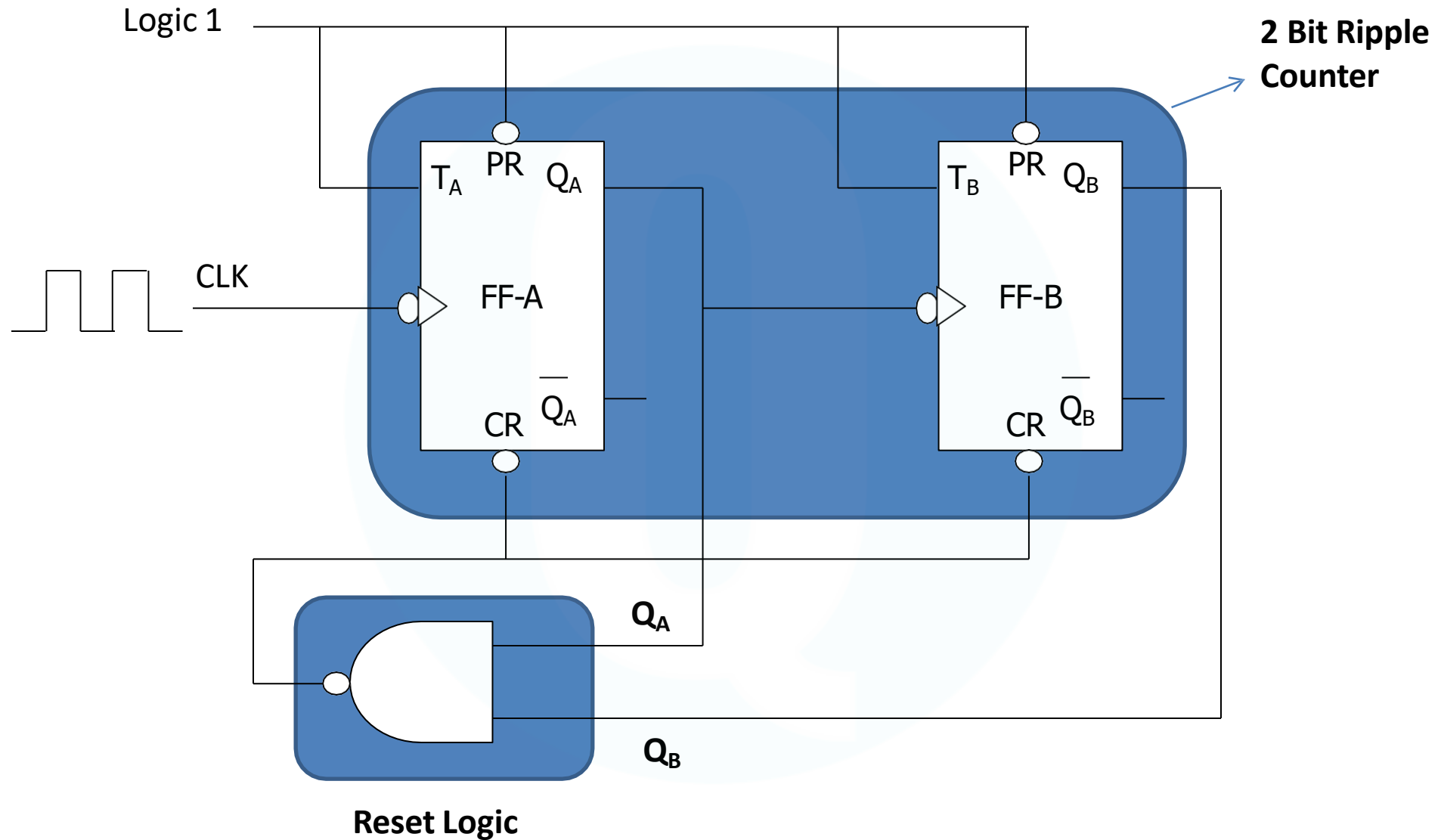
K-map & Simplification for Reset Logic

		Q_A	
		$\overline{Q_A}$	Q_A
Q_B	0	1	1
	1	1	0

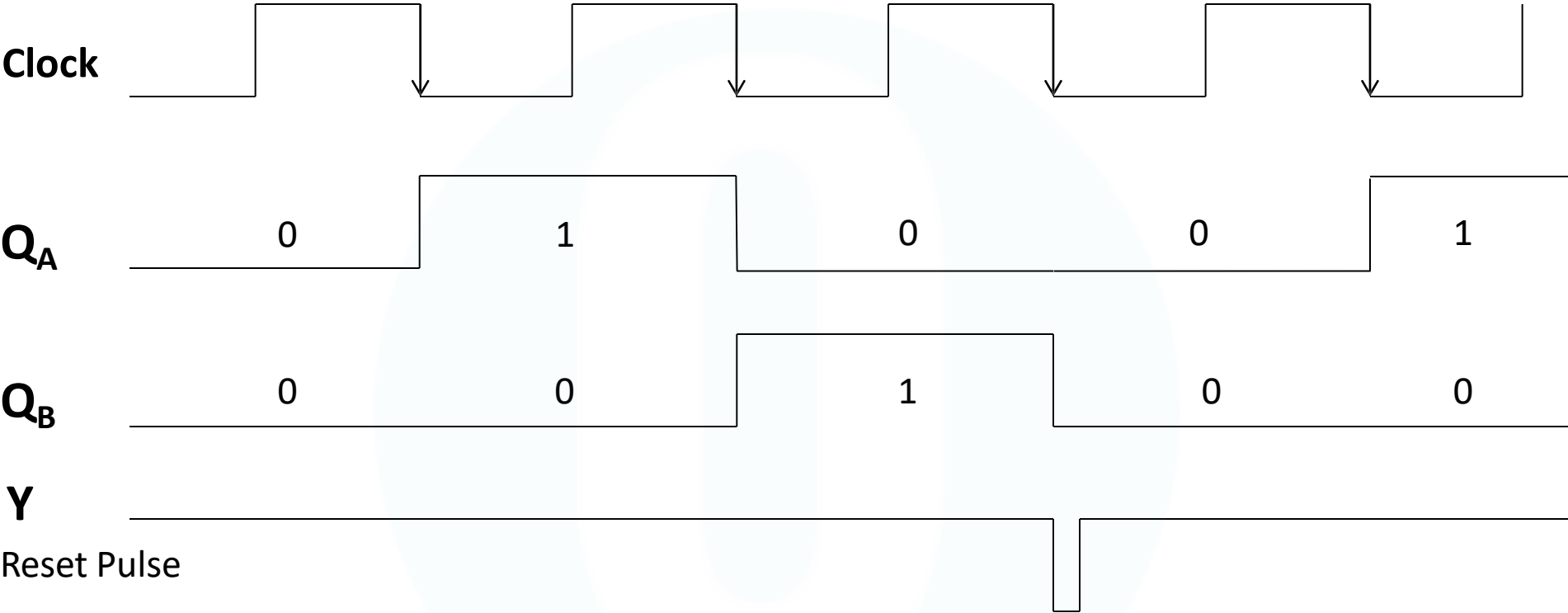
$$Y = \overline{Q_B} + \overline{Q_A}$$

$$\therefore Y = \overline{Q_B Q_A}$$

Design MOD-3 Asynchronous Counter



Design MOD-3 Asynchronous Counter

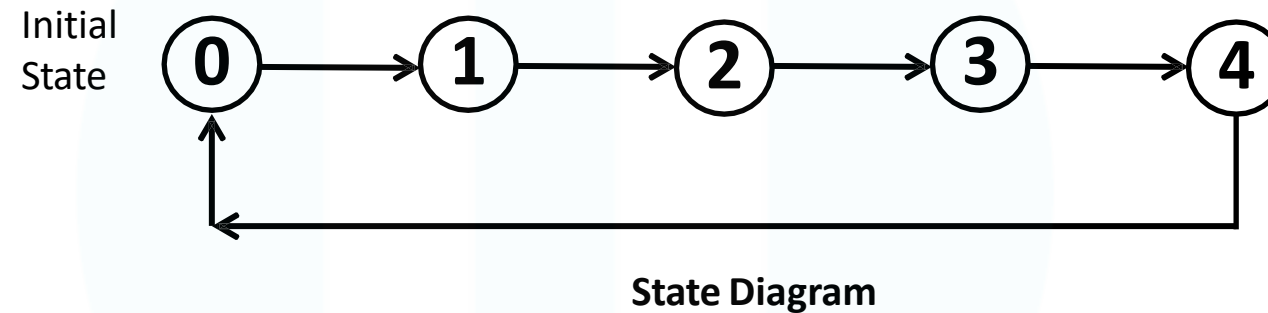


Counter O/P

$Q_B Q_A$	00	01	10	00	01
	0	1	2	0	0

Design MOD-5 Asynchronous Counter

- ✓ Mod – 5 counter is a counter having five states i.e. 000, 001, 010, 011 and 100. After 100 it will return back to its original state 000.



- ✓ We have to design the reset logic which is a combinational circuit.
- ✓ The output of reset logic is applied to the clear inputs of flip flops. This is an active Low input.

Design MOD-5 Asynchronous Counter

Truth Table of Reset Logic

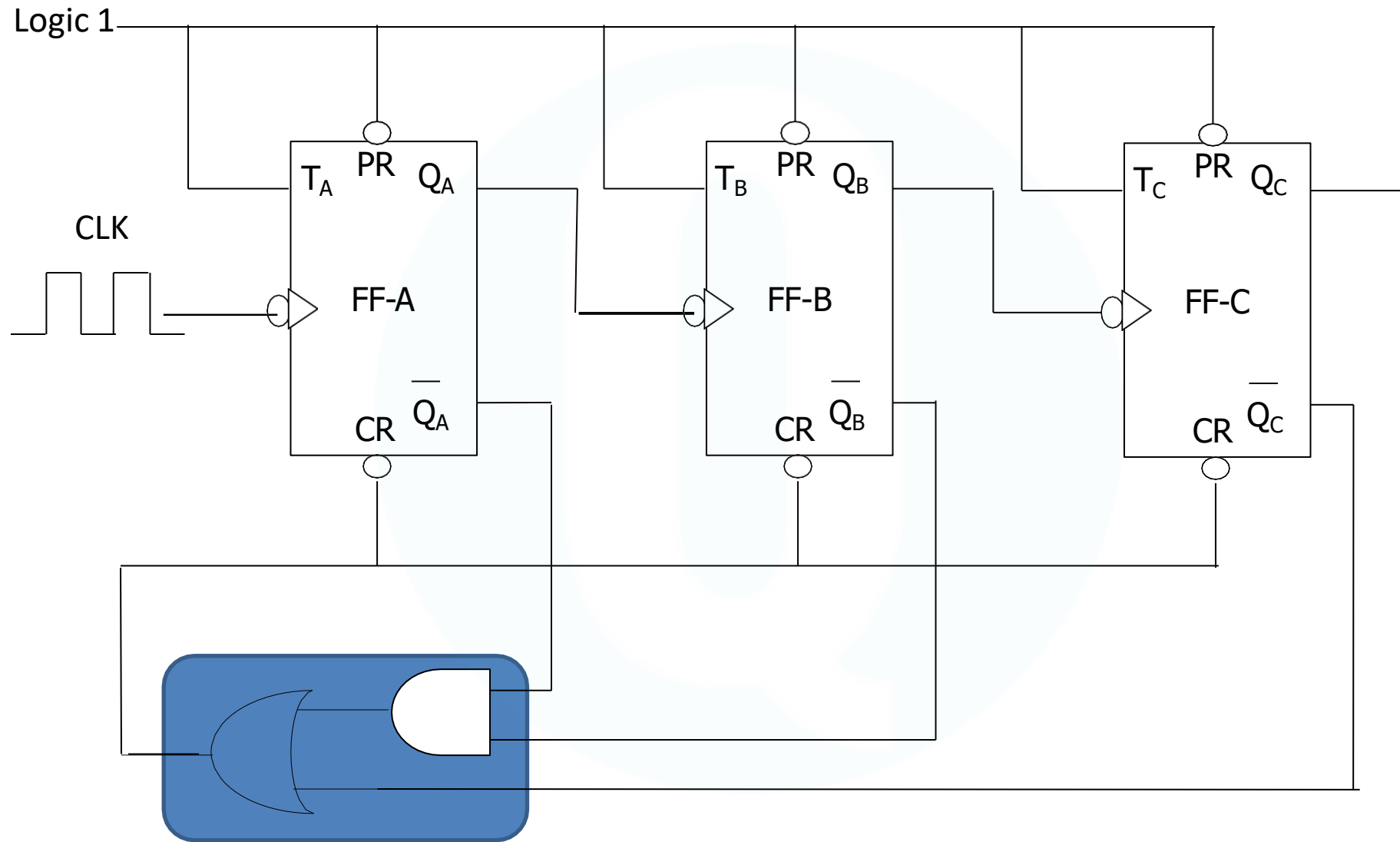
State	FF Outputs			O/P of reset logic Y
	Q_C	Q_B	Q_A	
0	0	0	0	1
1	0	0	1	1
2	0	1	0	1
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

K-map & Simplification for Reset Logic

		$Q_B Q_A$			
		00	01	11	10
Q_C	$\overline{Q_C}$ 0	1	1	1	1
Q_C	1	1	0	0	0

$$Y = \overline{Q_C} + \overline{Q_B} \overline{Q_A}$$

Design MOD-5 Asynchronous Counter

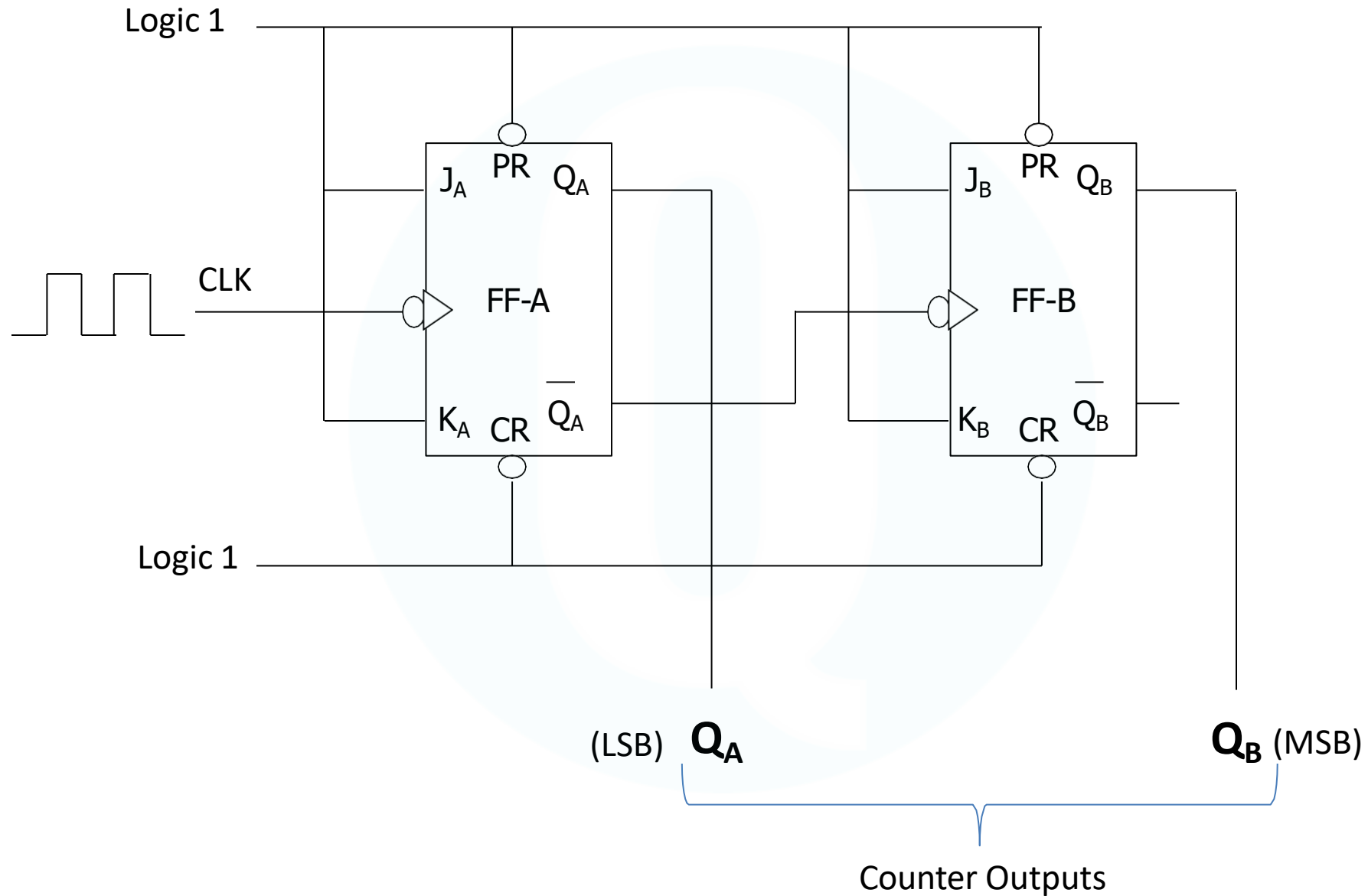


Reset Logic

Up Counter Vs Down Counter

- ✓ The counter which count is in up direction that means the decimal equivalent of the counter output increases (0, 1, 2,.....etc.) as it receives the clock pulses. Hence such counters are called as **“Up Counter”**.
- ✓ The counter which count is in down direction that means the decimal equivalent of the counter output decreases as it receives the clock pulses. Hence such counters are called as **“Down Counter”**.

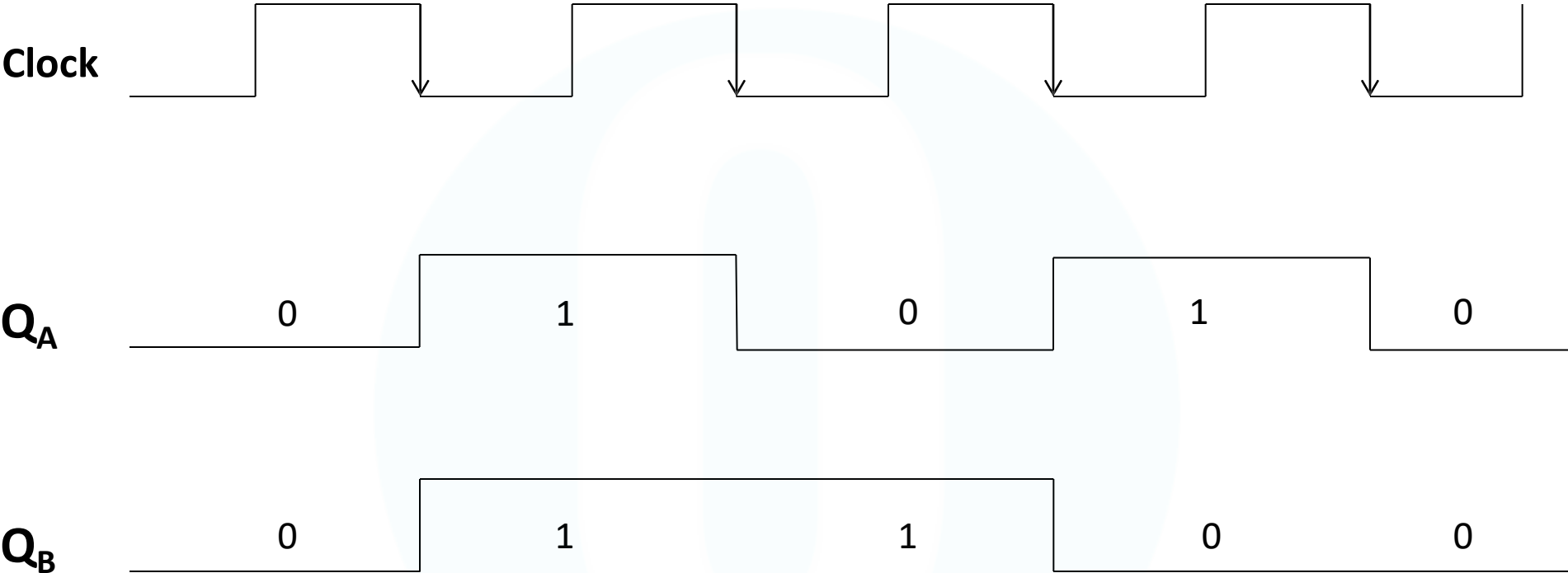
2 Bit Asynchronous or Ripple Down Counter



2 Bit Asynchronous or Ripple Down Counter

Clock	Counter Outputs		State Number	Decimal Equivalent of Counter Output
	Q _B (MSB)	Q _A (LSB)		
Initially	0	0	-	0
1 st ↓	1	1	4	3
2 nd ↓	1	0	3	2
3 rd ↓	0	1	2	1
4 th ↓	0	0	1	0

2 Bit Asynchronous or Ripple Down Counter

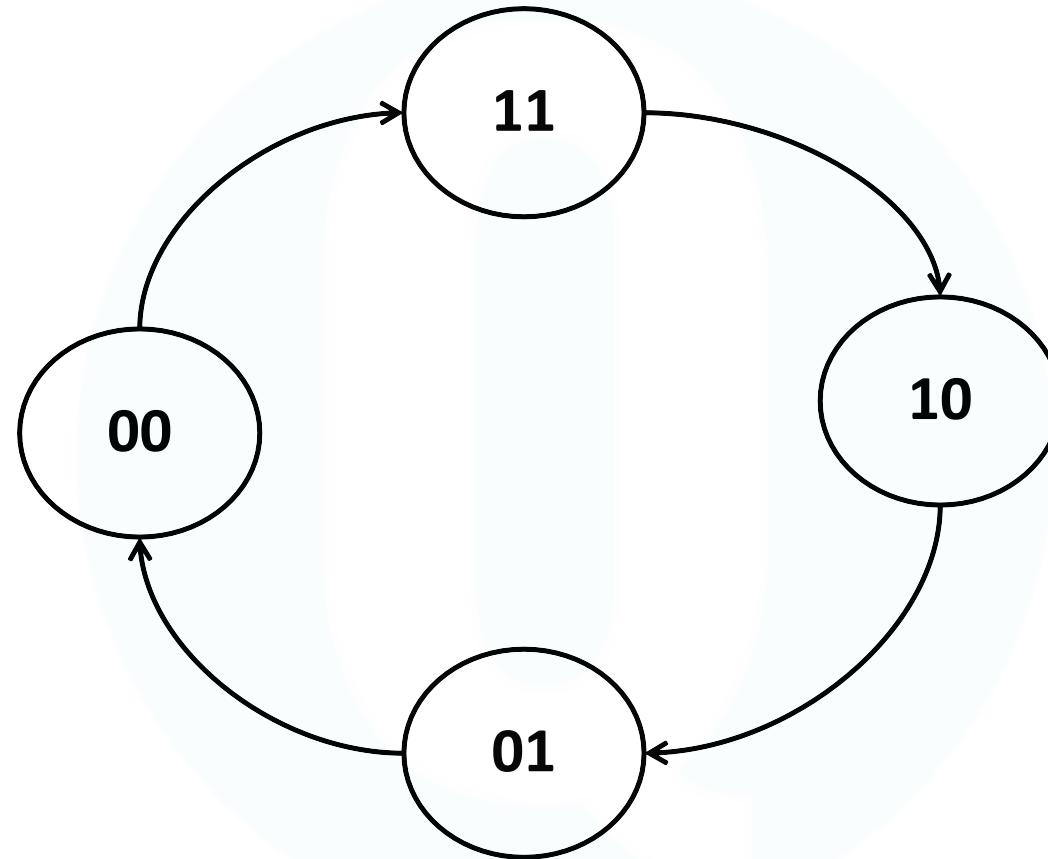


Counter O/P

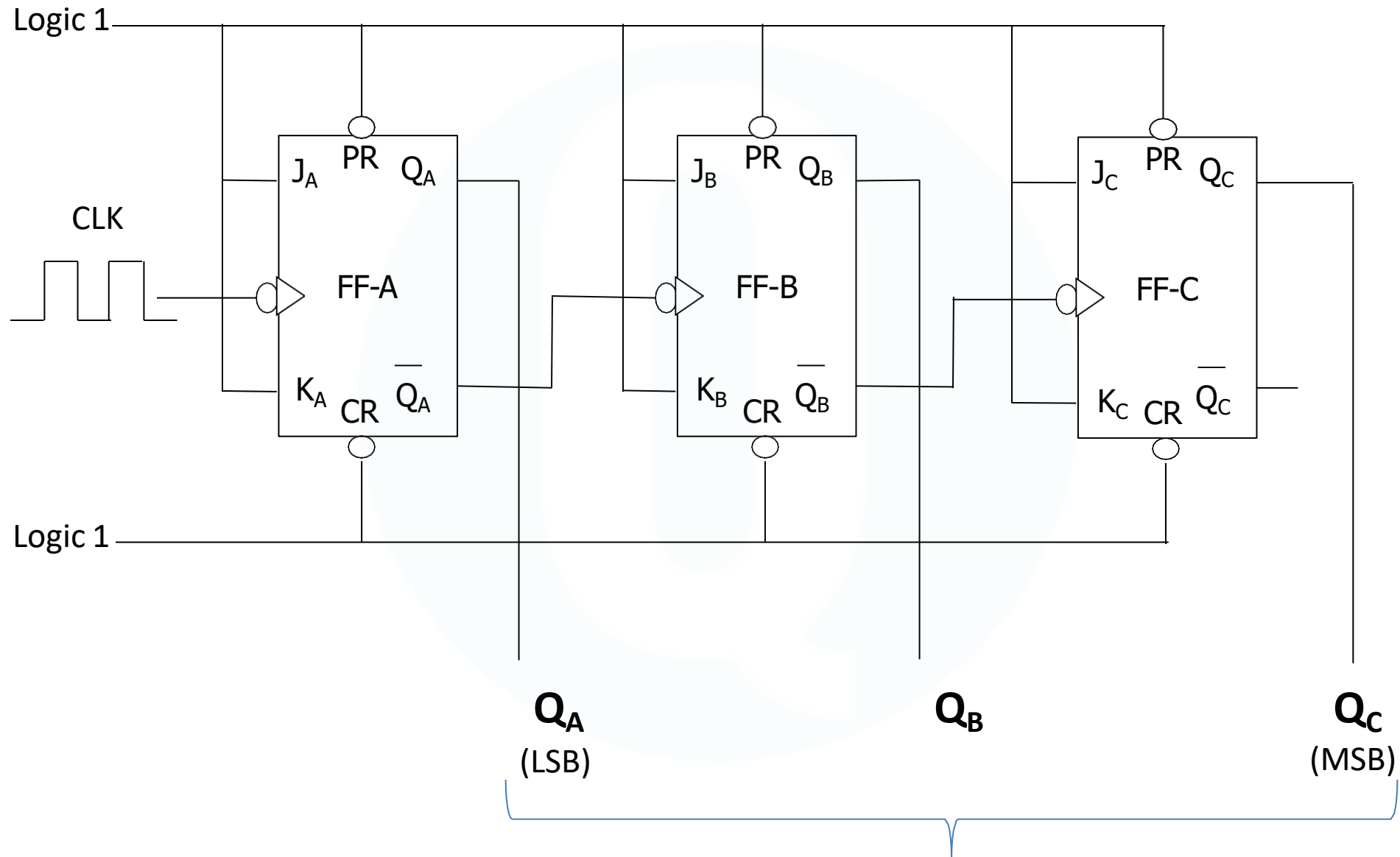
QB QA

11	10	01	00
3	2	1	0

State diagram of 2 bit Ripple Down Counter



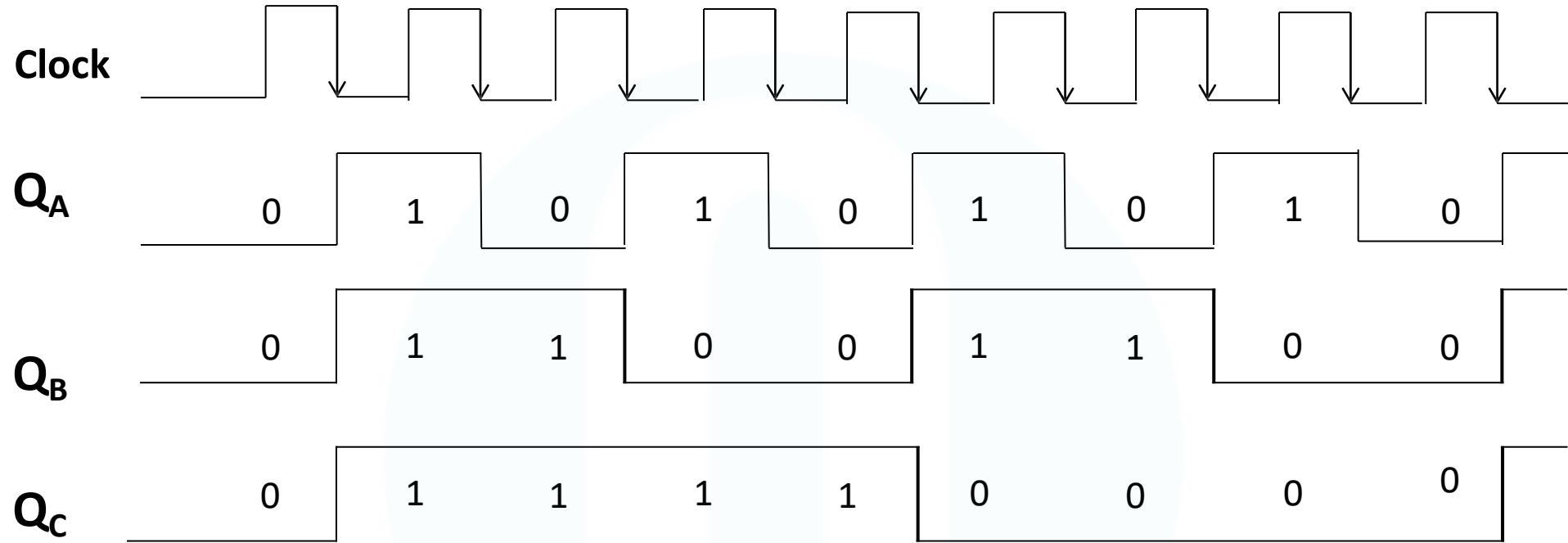
3 Bit Asynchronous or Ripple Down Counter



3 Bit Asynchronous or Ripple Down Counter

Clock	Flip Flop Outputs			State	Decimal Equivalent
	Q _C (MSB)	Q _B	Q _A (LSB)		
Initial	0	0	0	-	0
1 st ↓	1	1	1	8	7
2 nd ↓	1	1	0	7	6
3 rd ↓	1	0	1	6	5
4 th ↓	1	0	0	5	4
5 th ↓	0	1	1	4	3
6 th ↓	0	1	0	3	2
7 th ↓	0	0	1	2	1
8 th ↓	0	0	0	1	0
9 th ↓	1	1	1	8	7

3 Bit Asynchronous or Ripple Down Counter

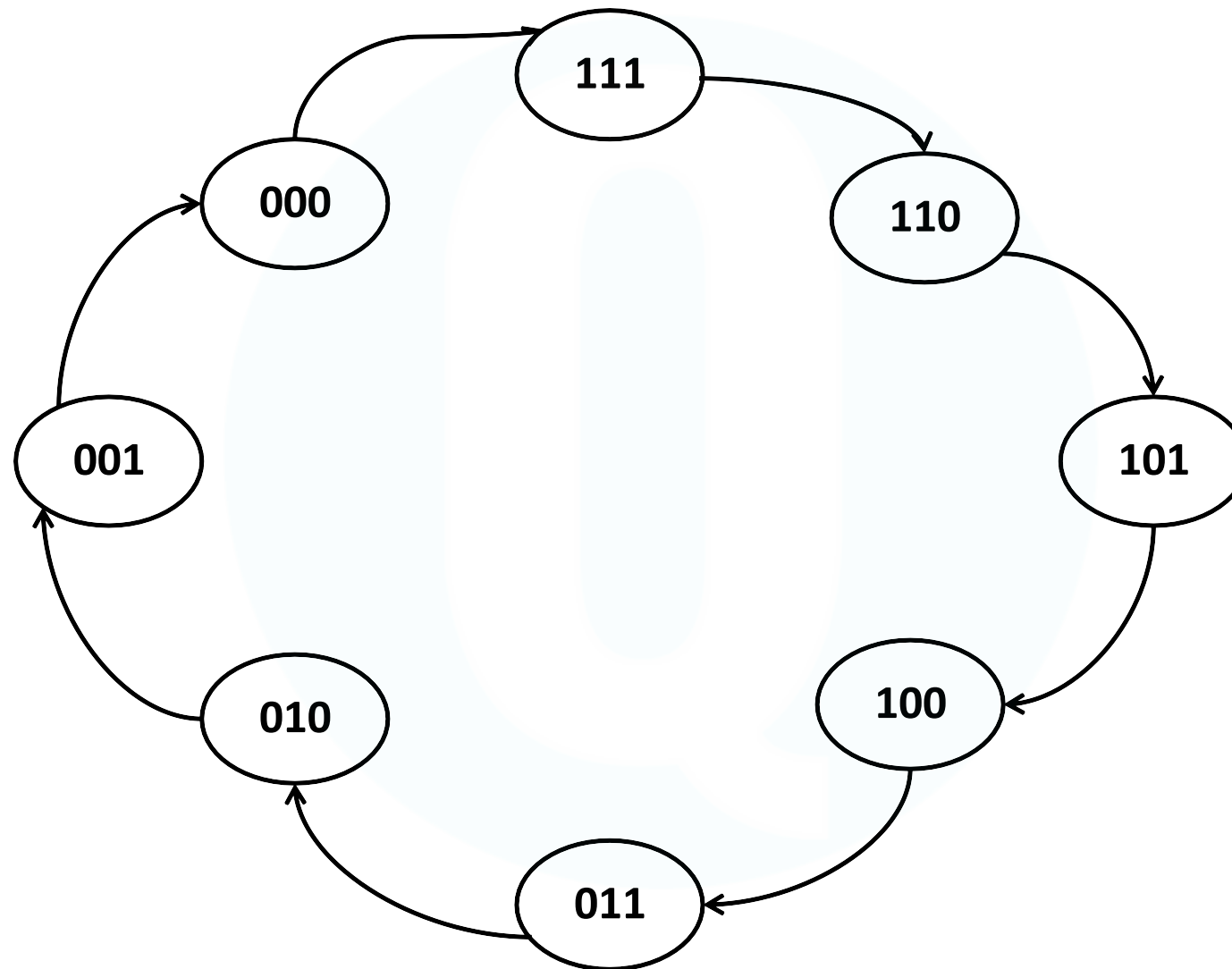


Counter O/P

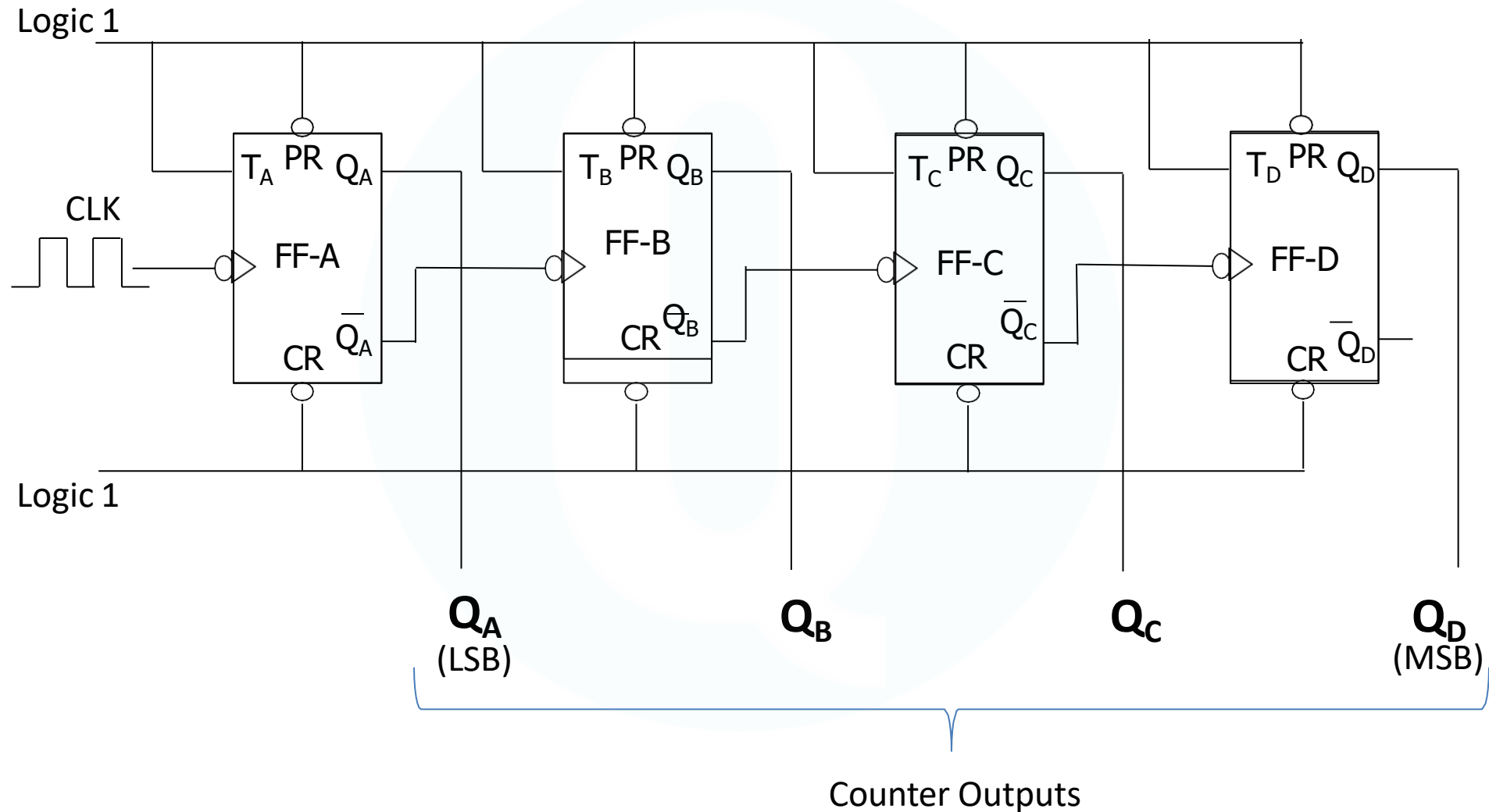
Q_C Q_B Q_A

000	111	110	101	100	011	010	001	000
0	1	2	3	4	5	6	7	0

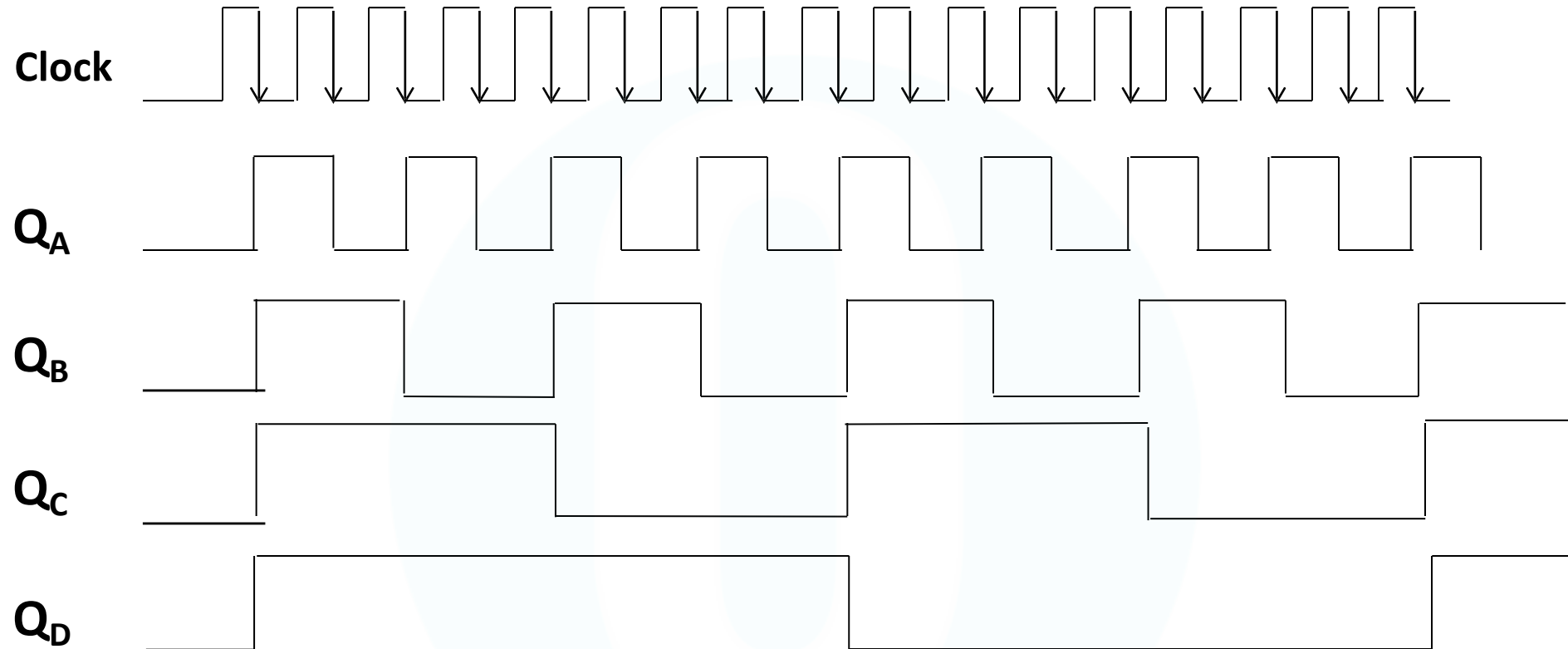
State diagram of 3 bit Asynchronous Down Counter



4 Bit Asynchronous or Ripple Down Counter



4 Bit Asynchronous or Ripple Down Counter



Counter O/P

Q_D	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
Q_C	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
Q_B	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
Q_A	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Up/Down Ripple Counter

- ✓ In Up/Down ripple counter, the counter can be used as UP counter as well as Down Counter.
- ✓ Up Counting Mode (M=0): The Q output of the preceding Flip Flop is connected to the clock of the next stage if up counting is to be achieved. For this mode, the mode select input M is at logic 0.
- ✓ Down Counting Mode (M=1): The \overline{Q} output of the preceding Flip Flop is connected to the clock of the next stage if down counting is to be achieved. For this mode, the mode select input M is at logic 1.



Disadvantages of Ripple Counter

- ✓ Every flip flop has its own propagation delay. In ripple counter the output of the previous flip flop is used as clock for the next flip flop. Hence the propagation delay goes on accumulating.
- ✓ The propagation delay goes on increasing with increase in number of flip flops.
- ✓ This will put a limitation on the maximum clock frequency.
- ✓ The frequency 'f' of a clock pulse for reliable operation of the counter is given by;

$$f \leq \frac{1}{n(t_d) + T_s}$$

Where n = number of flip flops

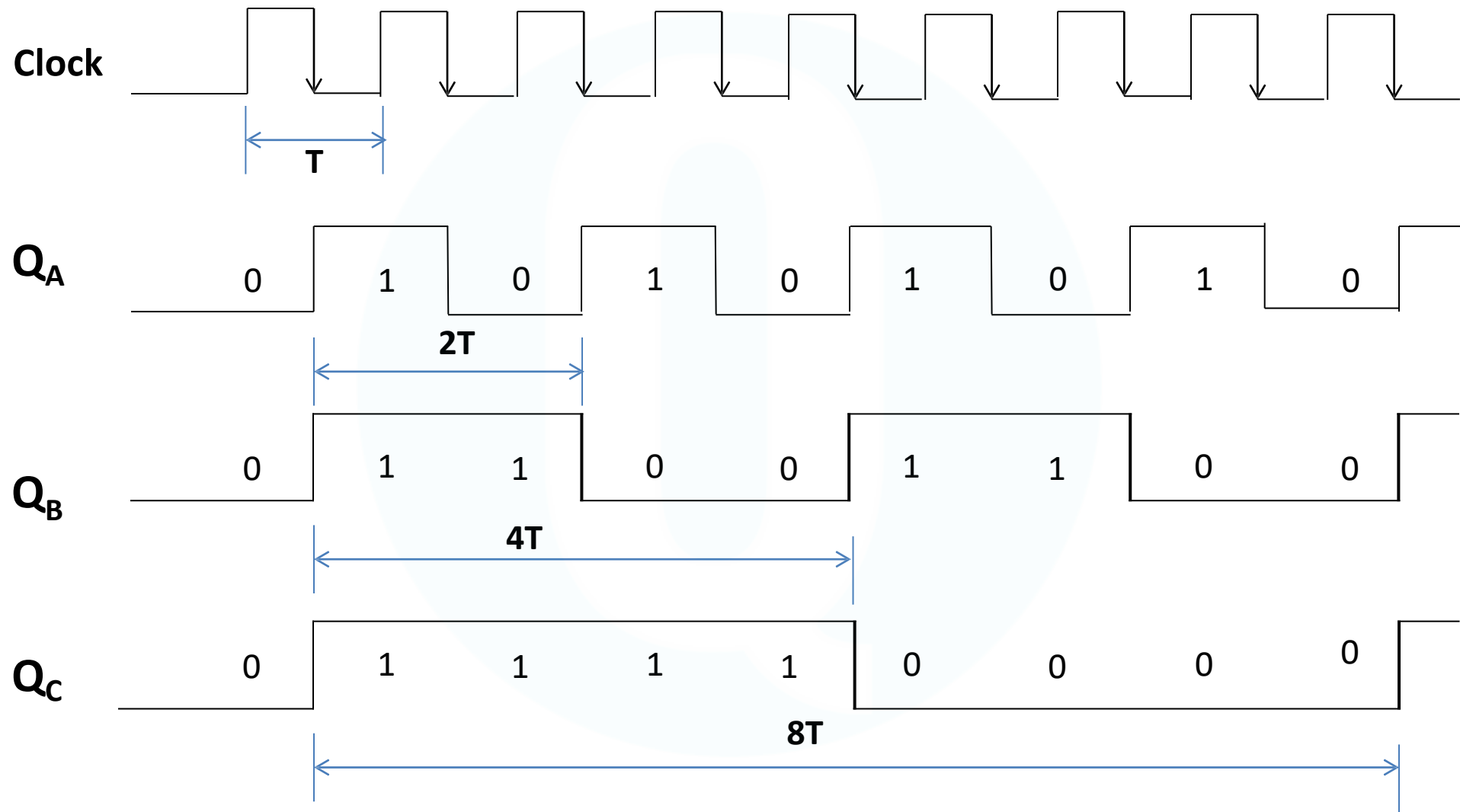
T_s = width of strobe pulse

t_d = propagation delay of one flip flop

Frequency Division in Ripple Counter

- ✓ In ripple counter, a flip flop in toggle mode divides the clock frequency by 2.
- ✓ That means the frequency of Q and Q output of a toggle flip flop is exactly half of the clock frequency.
- ✓ The concept of frequency division is observed in counters where flip flops used in toggle mode.

Frequency Division in Ripple Counter



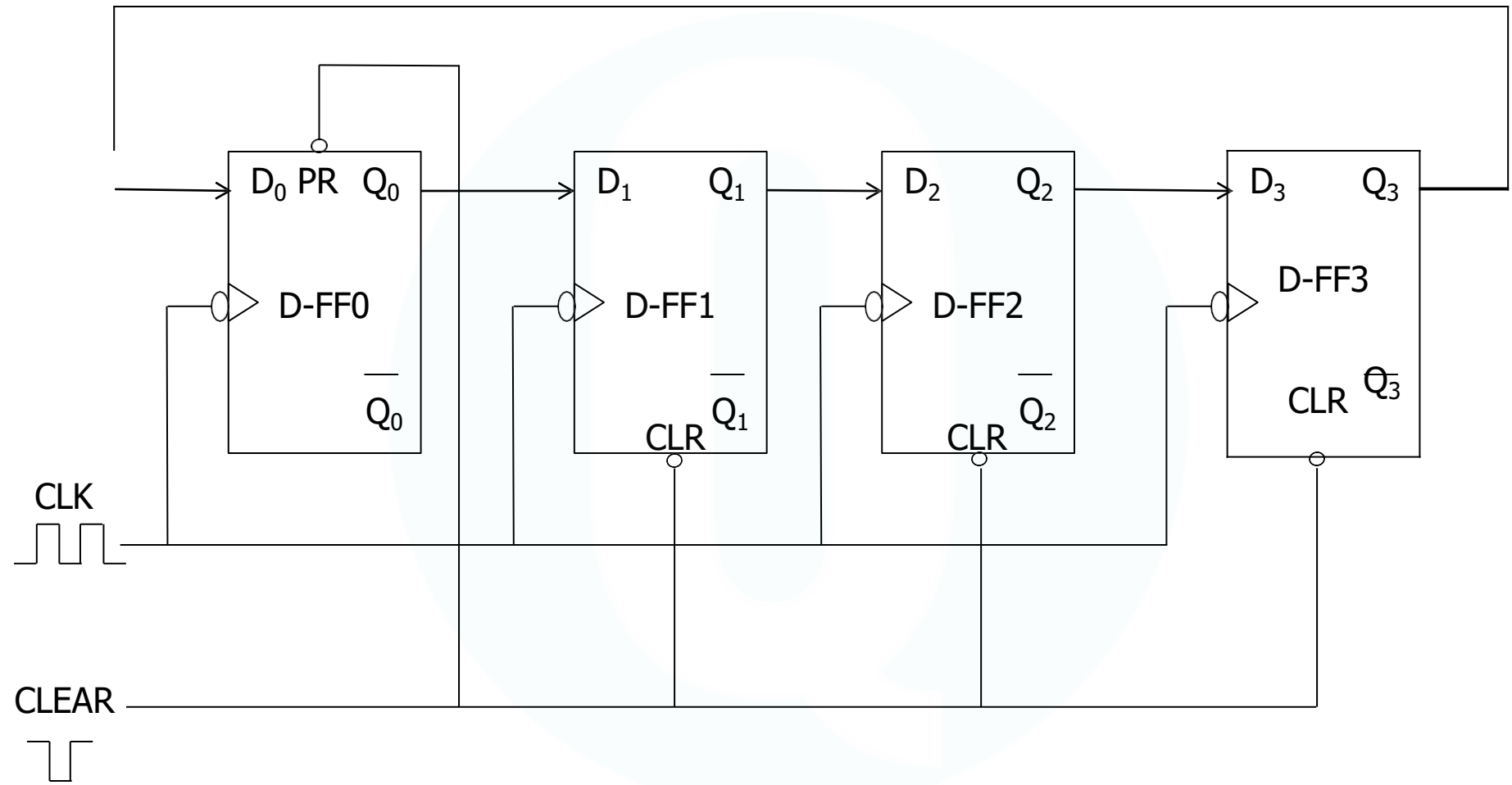
Modulus of Counter

- ✓ The modulus of a counter represents the **number of states** through which the counter progresses during its operation.
- ✓ So in general, an n bit ripple counter has MOD number:

$$\therefore MOD \text{ No.} = 2^n$$

- ✓ For example, 2 bit ripple counter is called as MOD-4 counter because the counter represents 4 states during its operation.
- ✓ For example, 3 bit ripple counter is called as MOD-8 counter because the counter represents 8 states during its operation.

Ring Counter



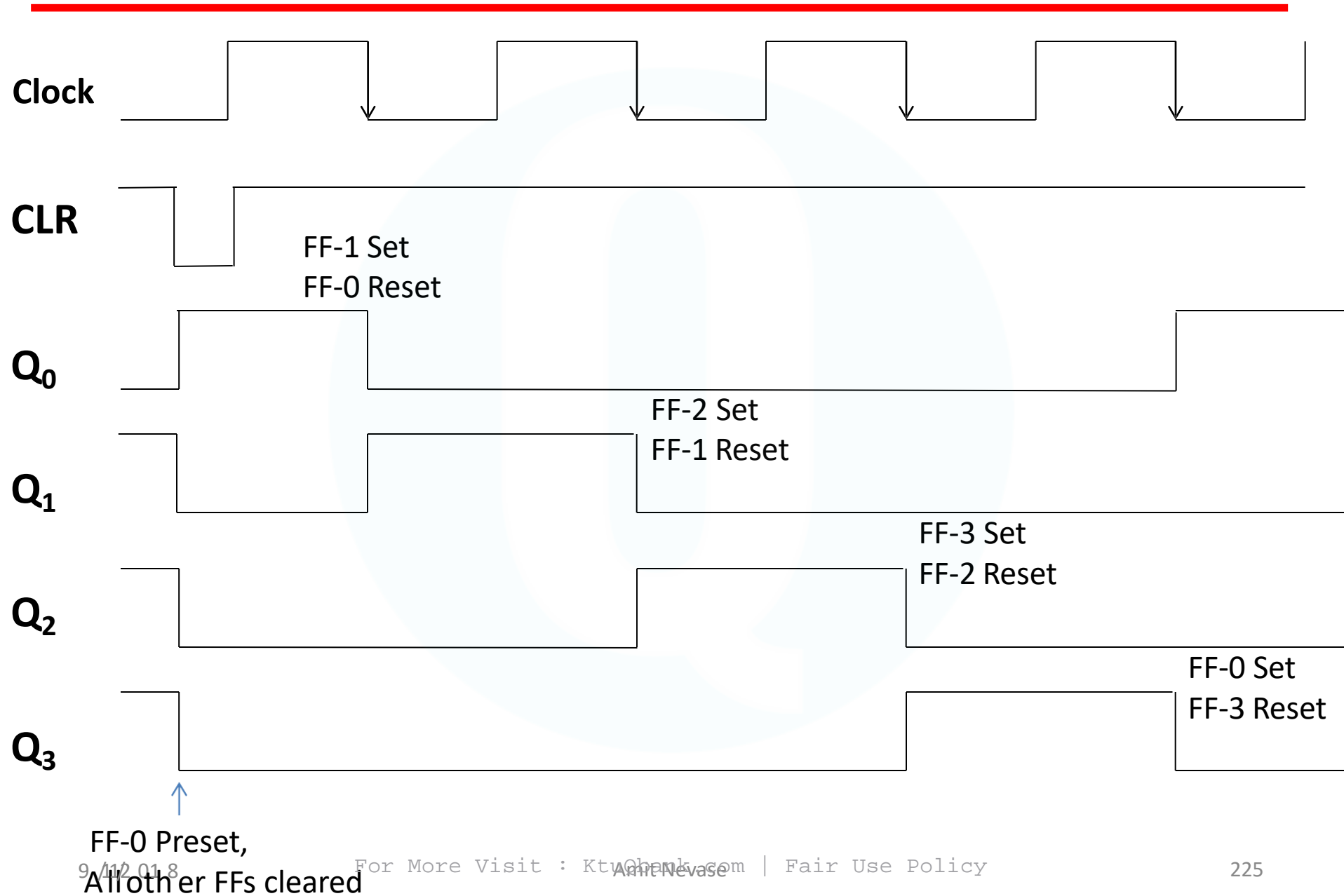
Ring Counter

CLR	CLK	Q_0	Q_1	Q_2	Q_3
1	X	1	0	0	0
1	↓	0	1	0	0
1	↓	0	0	1	0
1	↓	0	0	0	1
1	↓	1	0	0	0

FF-0 Preset,
others cleared

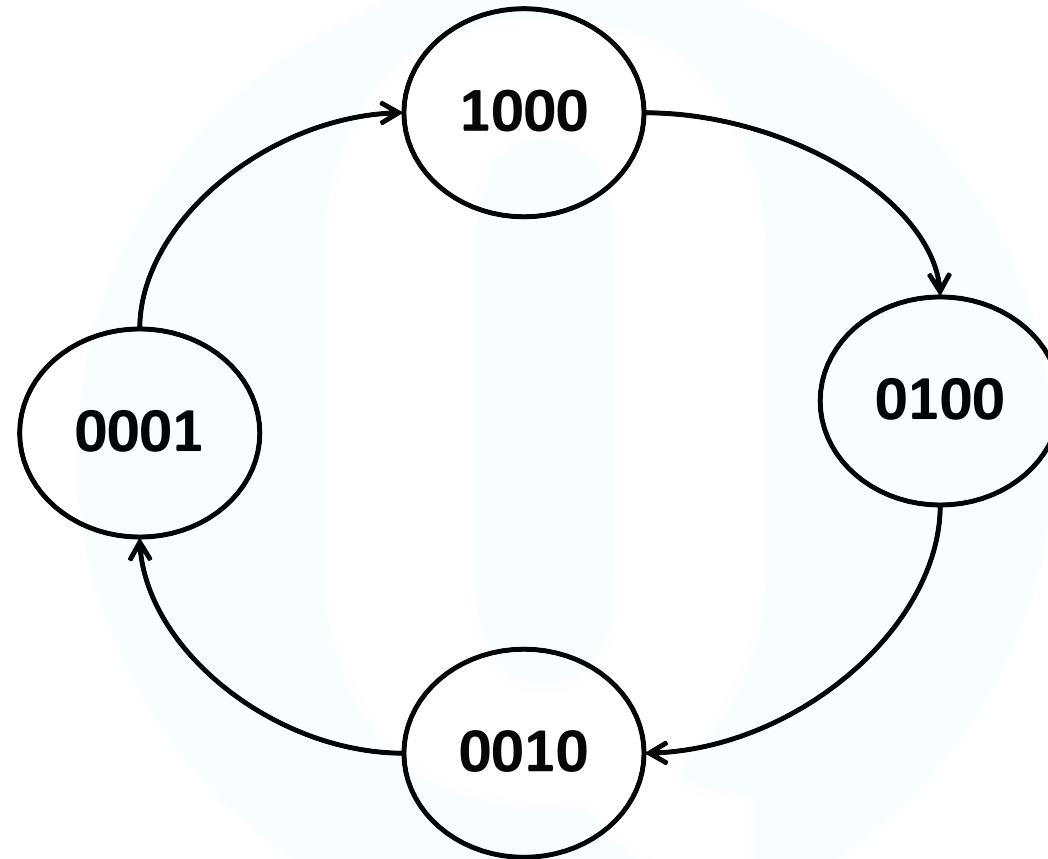
The Presetted 1,
follows a circular
Path to form a ring

Ring Counter

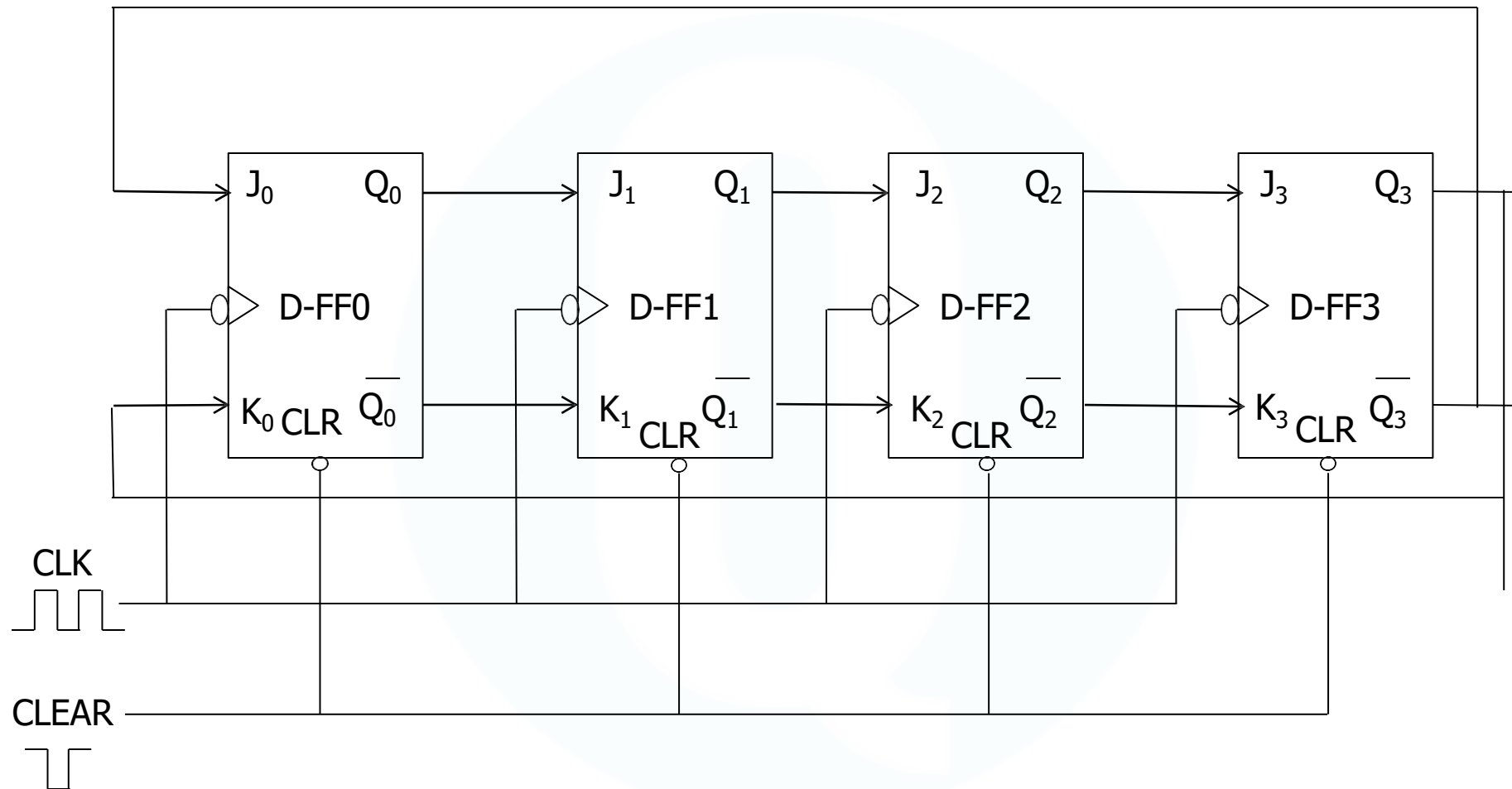


FF-0 Preset,
All other FFs cleared


State diagram of Ring Counter



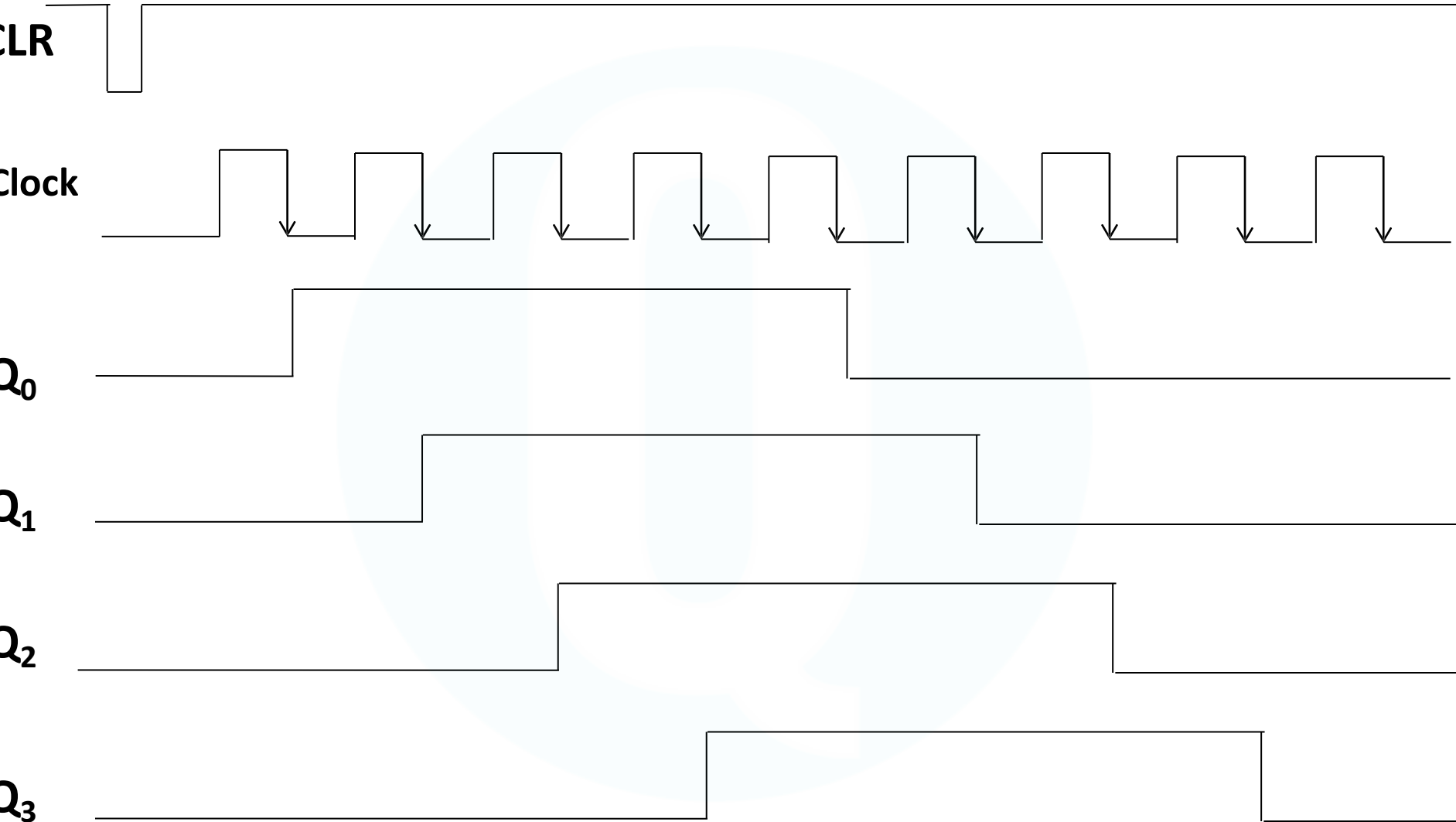
Johnson Counter (Twisted Ring Counter)



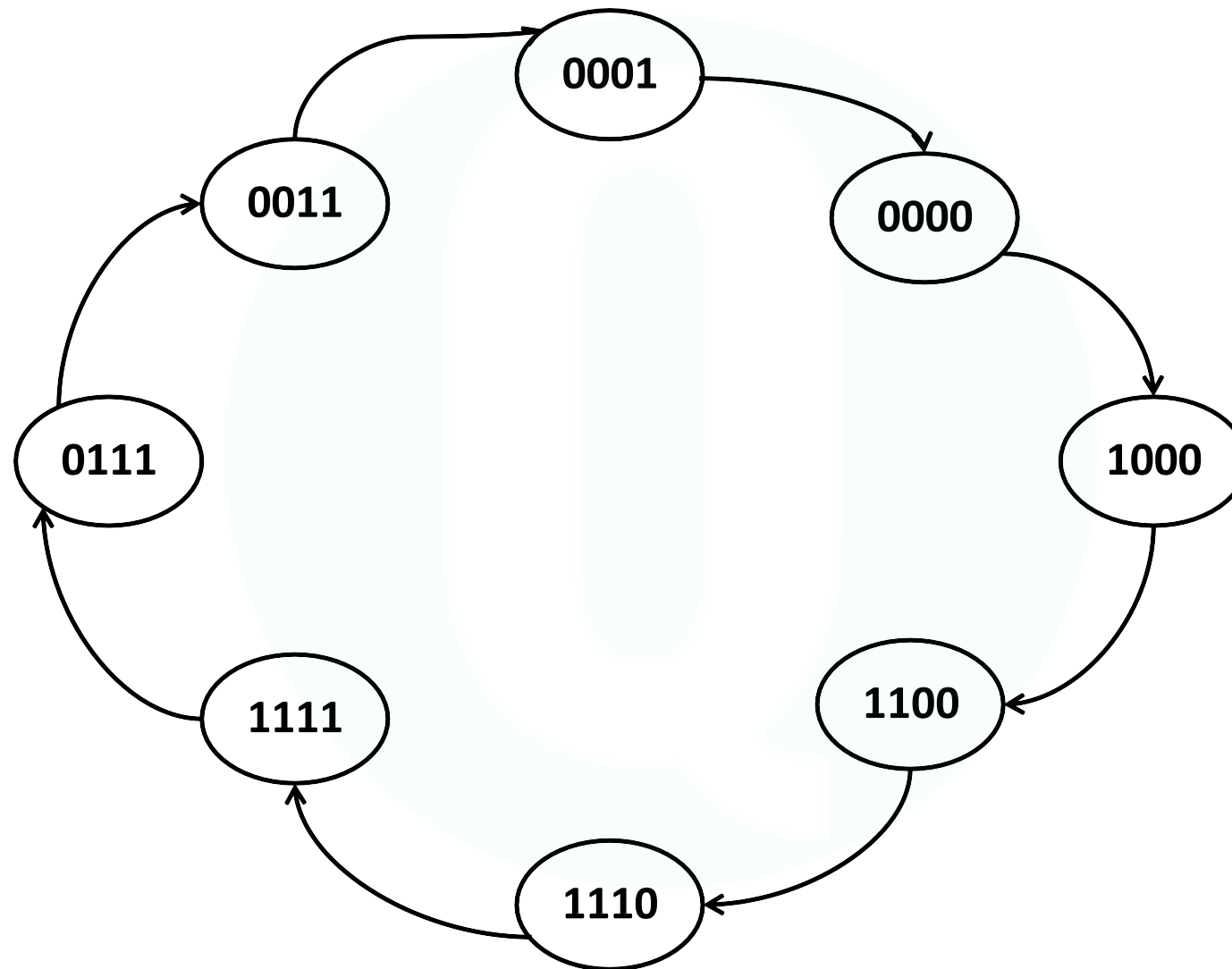
Johnson Counter (Twisted Ring Counter)

CLR	CLK	Q_3	Q_2	Q_1	Q_0	State Number	Decimal Equivalent
	Initial	0	0	0	0	1	0
1	↓	0	0	0	1	2	1
1	↓	0	0	1	1	3	3
1	↓	0	1	1	1	4	7
1	↓	1	1	1	1	5	15
1	↓	1	1	1	0	6	14
1	↓	1	1	0	0	7	12
1	↓	1	0	0	0	8	8
1	↓	0	0	0	0	1	0

Johnson Counter (Twisted Ring Counter)

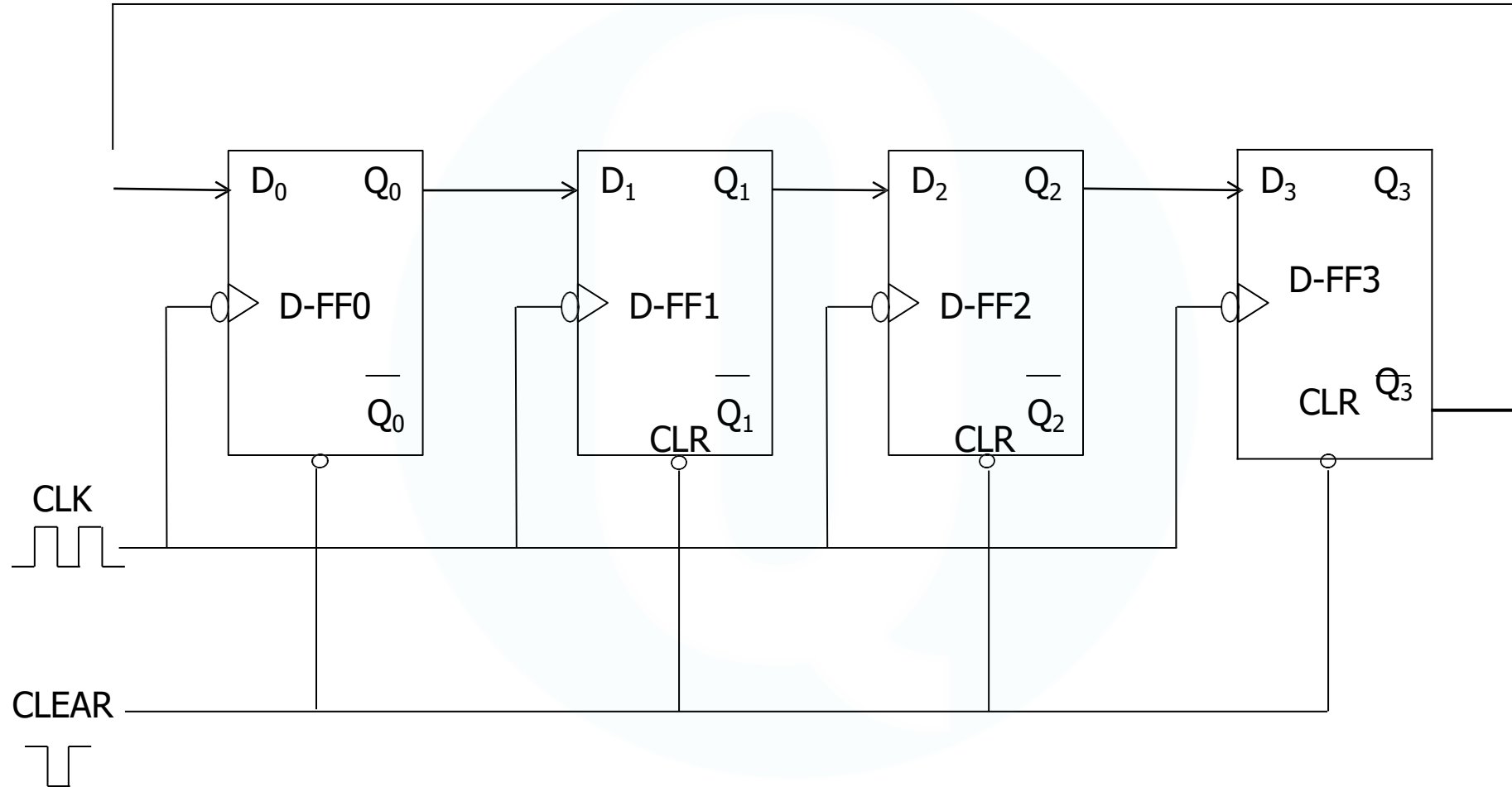
CLR**Clock** **Q_0** **Q_1** **Q_2** **Q_3** 

State diagram of Twisted Ring (Johnson) Counter




Johnson Counter (Twisted Ring Counter)

Using D Flip Flop



Johnson Counter (Twisted Ring Counter)

CLR	CLK	Q_3	Q_2	Q_1	Q_0
	Initial	0	0	0	0
1	↓	0	0	0	1
1	↓	0	0	1	1
1	↓	0	1	1	1
1	↓	1	1	1	1
1	↓	1	1	1	0
1	↓	1	1	0	0
1	↓	1	0	0	0
1	↓	0	0	0	0