DIGITAL DESIGN AND COMPUTER ORGANIZATION

Synchronous Sequential Logic

Team DDCO
Department of Computer Science and Engineering

Introduction- T1- section 5.1

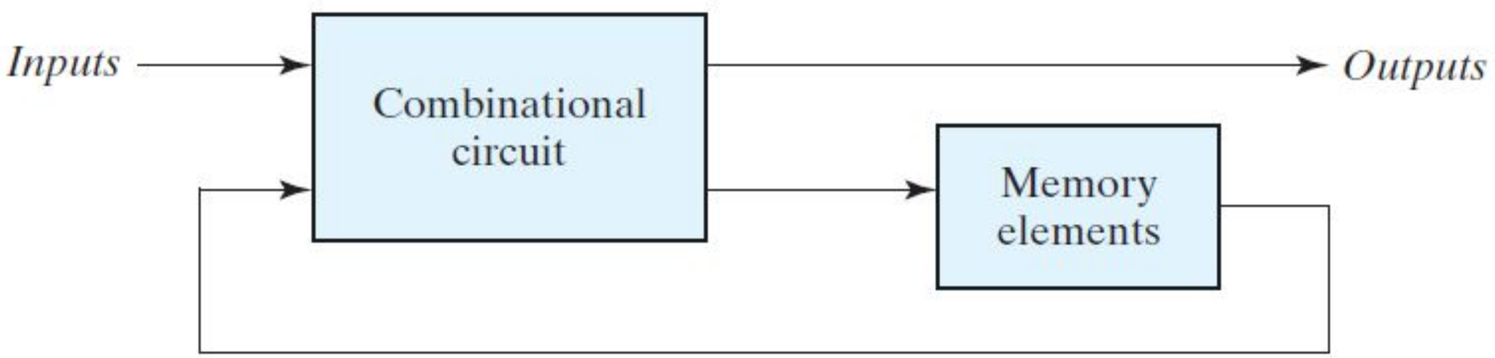
- Hand-held devices, cell phones, navigation receivers, personal computers, digital cameras, personal media players, and virtually all electronic consumer products have the ability to send, receive, store, retrieve, and process information represented in a binary format.
- The technology enabling and supporting these devices is critically dependent on electronic components that can store information, i.e., have memory.
- This chapter examines the operation and control of these devices and their use in circuits.

Introduction

- The digital circuits considered thus far have been **combinational**.
- **Their output depends only and immediately on their inputs—they have no memory, i.e., dependence on past values of their inputs.**

- Sequential circuits, however, act as **storage elements and have memory**.
- They can store, retain, and then retrieve information when needed at a later time.

Sequential Circuits (T1- section 5.2)



Sequential Circuits

- It consists of a ***combinational circuit*** to which ***storage elements*** are connected to form a ***feedback path*.**
- The storage elements are devices capable of storing ***binary information***.
- The binary information stored in these elements at any given time *defines the **state** of the sequential circuit* at that time.
- The sequential circuit receives binary information from external inputs that, together with the present state of the storage elements, determine the binary value of the outputs.
- These **external inputs** also determine the condition for changing the state in the storage elements.
- The ***next state*** of the *storage elements* is also a function of ***external inputs*** and the ***present state*.**

Sequential Circuits: Key Points

- Depends on current and prior input
- Memory element
- Feedback/cycles
- Remember previous input
- Next state= external input + present state
- State of system-binary information stored in system
- State variables

Thus, **a sequential circuit is specified by a time sequence of inputs, outputs, and internal states**.

Types of Sequential Circuits

There are two main types of sequential circuits, and their classification is a function of the timing of their signals.

1. A **synchronous sequential circuit** is a system whose behavior can be defined from the knowledge of its signals at discrete instants of time.
2. The behavior of an **asynchronous sequential circuit** depends upon the **input signals** at any instant of time and the order in which the inputs change.

Synchronous Sequential Circuit

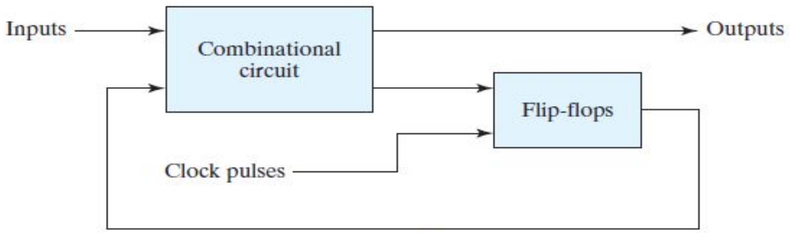
- A synchronous sequential circuit employs signals that affect the storage elements at only discrete instants of time.
- Synchronization is achieved by a timing device called a **clock generator**, which provides a clock signal having the form of a periodic train of clock pulses.
- The clock signal is commonly denoted by the identifiers ***clock*** **and** ***clk***.
- The **clock pulses are distributed throughout the system in such a way that storage elements are affected only with the arrival of each pulse.**

Synchronous Sequential Circuit

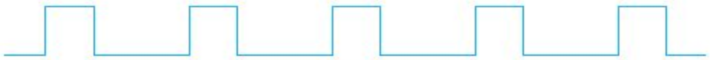
- In practice, the clock pulses determine when computational activity will occur within the circuit, and other signals (external inputs and otherwise) determine what changes will take place affecting the storage elements and the outputs.
- For example, a **circuit that is to add and store two binary numbers would compute their sum from the values of the numbers and store the sum at the occurrence of a clock pulse.**
- Synchronous sequential circuits that use clock pulses to control storage elements are called **clocked sequential circuits**

Synchronous Sequential Circuit

- The storage elements (memory) used in clocked sequential circuits are **called flip flops**. A flip-flop is a binary storage device capable of storing one bit of information. In a stable state, the output of a flip-flop is either **0 or 1**



(a) Block diagram



(b) Timing diagram of clock pulses

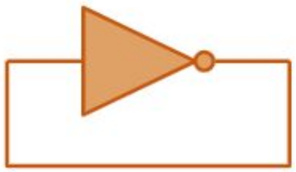
- The new value is stored (i.e., the flip-flop is updated) when a pulse of the clock signal occurs

Synchronous Sequential Circuit

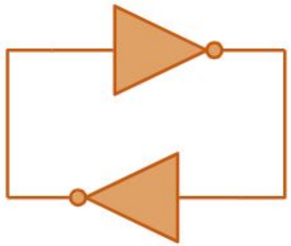
- Propagation delays play an important role in determining the minimum interval between clock pulses that will allow the circuit to operate correctly. **A change in state of the flip-flops is initiated only by a clock pulse transition—for example, when the value of the clock signals changes from 0 to 1**
- When a **clock pulse is not active**, the feedback loop between the value stored in the flip-flop and the value formed at the input to the flip-flop is effectively broken because the flip flop outputs cannot change even if the outputs of the combinational circuit driving their inputs change in value.
- Thus, **the transition** from one state to the next occurs only at predetermined intervals dictated by the **clock pulses.**

How to Implement Memory?

- The inverter is essentially the simplest logic gate
- What happens when we connect its output to input, forming a loop?
- Inverter loop is thus not in a **stable state**



- How about a loop of two inverters?
- A two-inverter loop can be in one of **two stable states**.
- A bit can have one of **two different values**. Thus, a two-inverter loop can **store** (or “remember”) a single bit.
- But how can we **change the bit stored** or the **stable state**?



Storage Using Cross-Coupled Inverters

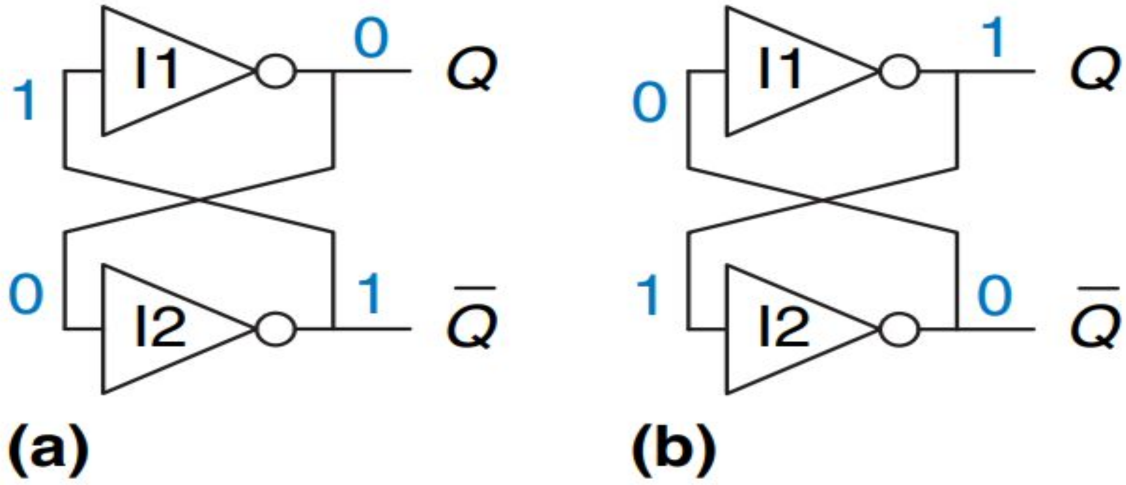
- **Cross-coupled inverters** form a basic memory element with **two stable states**.

- A system with **N stable states** can store up to **$\log_2(N)$ bits of information**.

- Therefore, a **bistable element** (with 2 stable states) can store **1 bit** of information.

Storage Using Cross-Coupled Inverters

Bistable Operation of Cross-Coupled Inverters



Storage Using Cross-Coupled Inverters

- When power is first applied to sequential circuit, initial state is unknown.

- It changes each time clock is on, it is unpredictable which is not desirable.

- **Cross coupled inverters are not practical because user has no input to control the state**
- **Hence we have flipflops as memoy elements**

Asynchronous Sequential Circuit

- The storage elements commonly used in asynchronous sequential circuits are **time-delay devices.**
- The storage capability of a time-delay device varies with the time it takes for the signal to propagate through the device.
- An asynchronous sequential circuit is **one where the outputs (and next states) can change immediately when inputs change, without waiting for a clock pulse.**
- In gate-type asynchronous systems, **the storage elements consist of logic gates whose propagation delay provides the required storage**.
- Thus, an **asynchronous sequential circuit may be regarded as a combinational circuit with feedback**.

Flip-Flops-key points

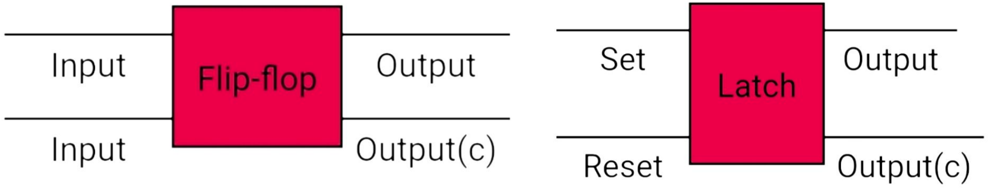
- The storage elements (memory) used in clocked sequential circuits are called flip-flops.
- A flip-flop is a binary storage device capable of storing one bit of information.
- In a stable state, the output of a flip-flop is either 0 or 1
- The new value is stored (i.e., the flip-flop is updated) when a pulse of the clock signal occurs.
- A change in state of the flip-flops is initiated only by a clock pulse transition, eg: when the value of the clock signals changes from 0 to 1
- Thus, the transition from one state to the next occurs only at predetermined intervals dictated by the clock pulses.

Latches

- **A storage element in a digital circuit can maintain a binary state indefinitely** (as long as power is delivered to the circuit), until directed by an input signal to switch states.
- The major differences among various types of storage elements are in the number of inputs they possess and in the manner in which the inputs affect the binary state.
- Storage elements that operate with **signal levels** (rather than signal transitions) are referred to as latches ; those controlled by a **clock transition are flip-flops .**
- Latches are said to be **level sensitive devices**; flip-flops are edge-sensitive devices.

Latches

- The two types of storage elements are related because latches are the basic circuits from which all flip-flops are constructed.
- Although latches are useful for storing binary information and for the design of asynchronous sequential circuits, they are not practical for use as storage elements in synchronous sequential circuits.
- They are the building blocks of flip-flops.



Difference between flip flop and latch

Source: solderingmind.com

Difference b/w Latches & Flip-Flops

Latches	Flip-Flops
Latches are building blocks of sequential circuits, these can be built from **logic gates**	Flip flops are also building blocks of sequential circuits. But, these can be built from the **latches**.
Latch continuously **checks its inputs and changes its output** correspondingly.	Flip flop continuously checks its inputs and changes its output correspondingly only at times determined by **clocking signal**
The latch is **sensitive to the duration of the pulse** and can send or receive the data when the switch is on.	Flipflop is sensitive to a **signal change**. They can transfer data only at the single instant and data cannot be changed until next signal change.Flip flops are used as a register.
It is based on the enable function input.	It works on the basis of clock pulses.
It is a level triggered, it means that the output of the present state and input of the next state depends on the level that is binary input 1 or 0. (**asynchronous circuits)**	It is an edge triggered, it means that the output and the next state input changes when there is a change in clock pulse whether it may a+ve or-ve clock pulse.(**synchronous circuits**)

Difference b/w Latches & Flip-Flops

Example: Traffic Light Controller
Scenario:
A city traffic light controller uses digital storage elements to decide when to change from Green → Yellow → Red.

Using a Latch (Level-triggered)
Suppose the system uses a **latch** to store the current state of the traffic light.
Since a latch is **level-sensitive**, whenever the clock/enable signal is **high**, the latch is "transparent" — it immediately passes input changes to the output.

Problem: If there is noise or a small glitch during the enable-high period, the latch will capture it. This could cause the traffic light to change state unexpectedly (e.g., Green jumps to Red without Yellow).

Application-level interpretation: **Not safe** for traffic control, because continuous input changes during the active level may corrupt the output.

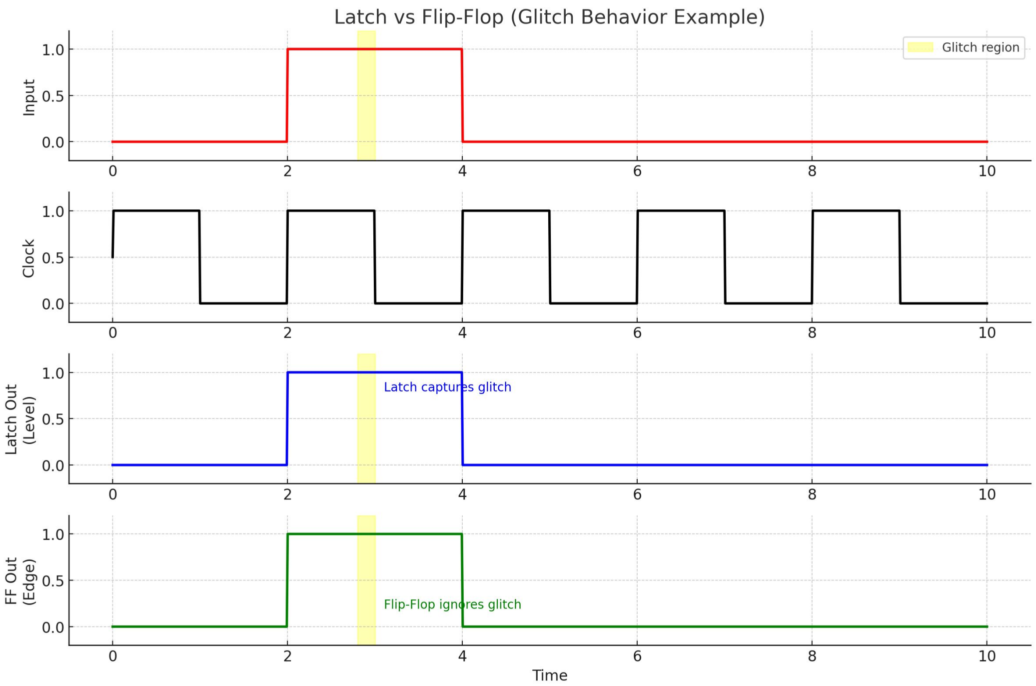
Difference b/w Latches & Flip-Flops

Example: Traffic Light Controller
Scenario:
A city traffic light controller uses digital storage elements to decide when to change from Green → Yellow → Red.

Using a Flip-Flop (Edge-triggered)
If the system uses a **Flip-Flop**, the light state is updated **only at the clock's rising (or falling) edge**.
Input changes happening in between edges are ignored until the next edge.
Advantage: The traffic light changes state in a controlled manner (e.g., every 30 seconds on the rising clock edge). Even if input glitches occur while the clock is low, they won't affect the state.

Application-level interpretation: **Much safer and reliable** for real-world sequential control (traffic lights, elevators, etc.).

Difference b/w Latches & Flip-Flops



Yellow region → glitch in the input.

Latch Output (blue) → immediately captures the glitch while clock is HIGH.

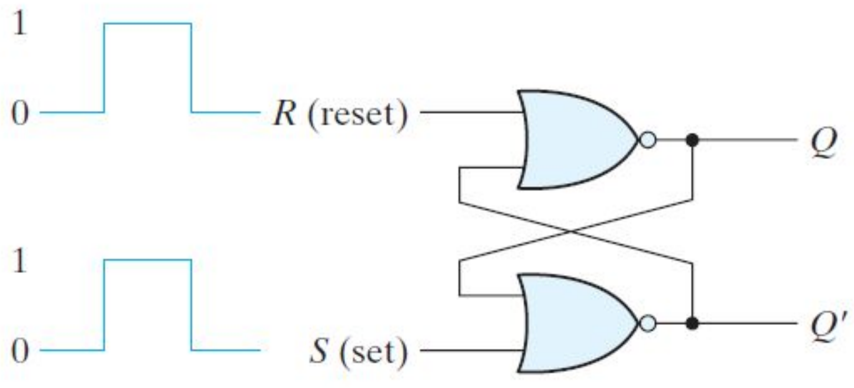
Flip-Flop Output (green) → ignores the glitch, updates only at the clock edge.

SR Latch (Using NOR Gates) T1- section 5.3

- The SR latch is a circuit with two cross-coupled NOR gates, and two inputs labeled S for set and R for reset. The latch has two useful states.
- When output ***Q = 1 and Q' = 0***, the latch is said to be in the ***set state***.
- When ***Q = 0 and Q' = 1***, it is in the ***reset state***.
- Outputs Q and Q are normally the complement of each other.

Truth table of NOR Gate

Input A	Input B	0 = (A + B)'
0	0	1
0	1	0
1	0	0
1	1	0



(a) Logic diagram

S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after $S = 1, R = 0$)
0	1	0	1	
0	0	0	1	(after $S = 0, R = 1$)
1	1	0	0	(forbidden)

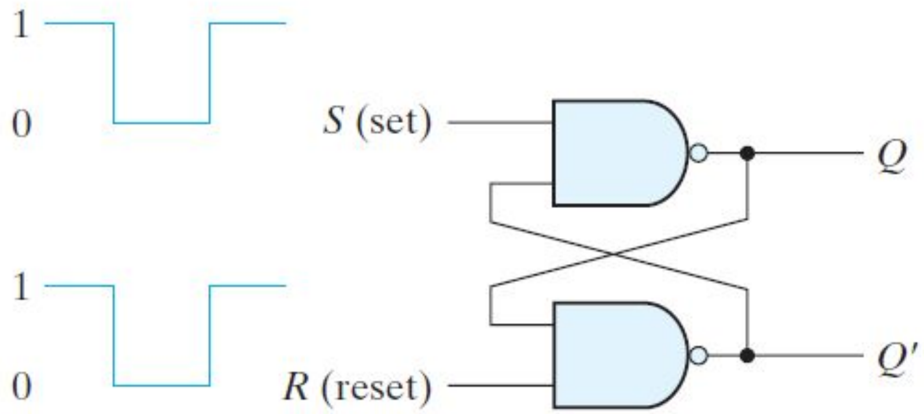
(b) Function table

SR Latch (Using NAND Gates)

- The SR latch is a circuit with two cross-coupled NAND gates and two inputs labeled S for set and R for reset.
- It operates with both inputs **normally at 1**, unless the state of the latch has to be changed.

Truth table of NAND Gate

A	B	Output
0	0	1
1	0	1
0	1	1
1	1	0



(a) Logic diagram

S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	(forbidden)

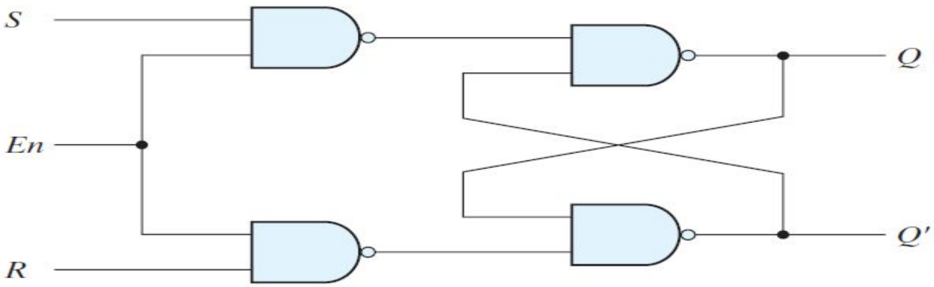
(b) Function table

SR Latch (Using NAND Gates)

- The application of **0 to the S input** causes output Q to go to 1, putting the latch in the **set state**.
- When the S input goes back to 1, the circuit remains in the set state.
- After both inputs go back to 1, we are allowed to change the state of the latch by placing a **0 in the R input**.
- This action causes the circuit to go to the **reset state** and stay there even after both inputs return to 1.
- The condition that is forbidden for the NAND latch is **both inputs being equal to 0 at the same time**, an input combination that should be **avoided**.

SR Latch with Control Input

- In comparing the NAND with the NOR latch, note that the input signals for the NAND require the complement of those values used for the NOR latch. Because the NAND latch requires a 0 signal to change its state, **it is sometimes referred to as an S`R` latch.**
- The operation of the basic SR latch can be modified by providing an **additional input signal** that determines (controls) when the state of the latch can be changed by determining whether S and R (or S' and R') can affect the circuit.
- It consists of the basic SR latch and two additional NAND gates.



(a) Logic diagram

En	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	$Q = 0$; reset state
1	1	0	$Q = 1$; set state
1	1	1	Indeterminate

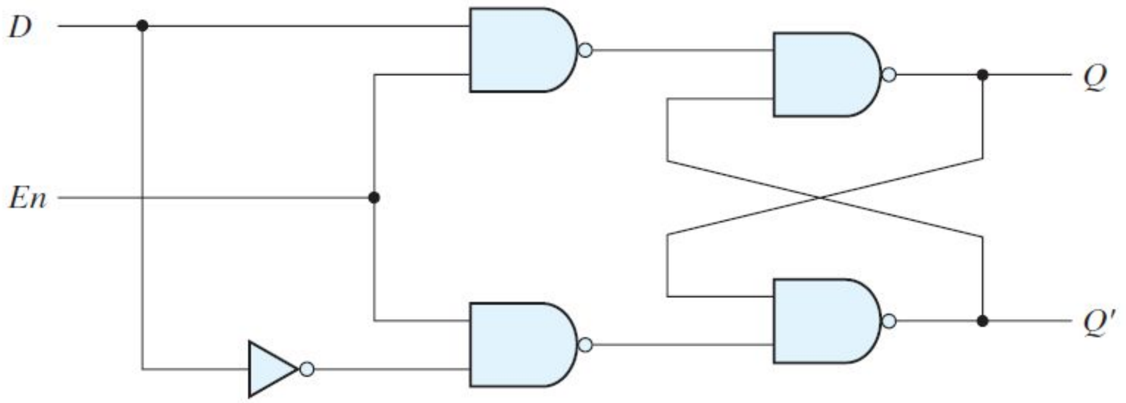
(b) Function table

SR Latch with Control Input

- The control input **E*n* acts as an enable signal** for the other two inputs.
- The outputs of the NAND gates stay at the logic-1 level as long as the enable signal remains at 0.
- This is the quiescent condition for the SR latch.
- When the **enable input goes to 1, information from the S or R input is allowed to affect the latch**.
- The set state is reached with S = 1, R = 0, and En = 1.(active-high enabled).
- To change to the reset state, the inputs must be S = 0, R = 1, and En = 1.
- In either case, **when En returns to 0, the circuit remains in its current state.**
- The control input disables the circuit by applying 0 to En, so that the state of the output does not change regardless of the values of S and R .
- An indeterminate condition occurs when all three inputs are equal to 1.

D Latch(Transparent Latch)

- One way to eliminate the undesirable condition of the indeterminate state in the SR latch is to ensure that inputs S and R are never equal to 1 at the same time. This is done in the D latch.
- This latch has only two inputs: D (data) and En (enable).



(a) Logic diagram

En	D	Next state of Q
0	X	No change
1	0	$Q = 0$; reset state
1	1	$Q = 1$; set state

(b) Function table

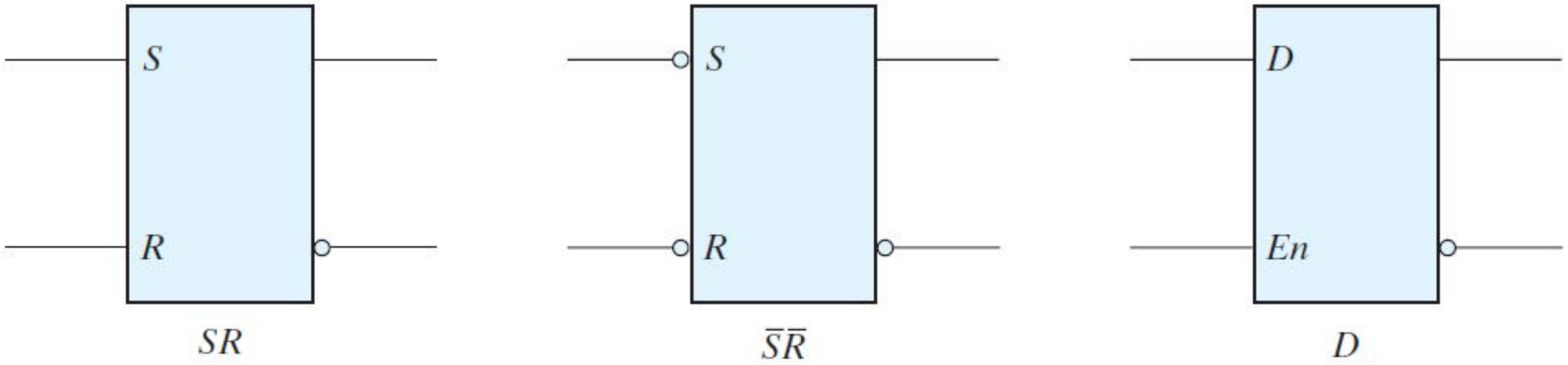
D Latch

PES UNIVERSITY

- As long as the **enable input is at 0,** the cross-coupled SR latch has both inputs at the 1 level and the **circuit cannot change state regardless of the value of D.**
- The binary information present at the data input of the D latch is transferred to the Q output when the enable input is asserted.
- **The output follows changes in the data input as long as the enable input is asserted**.
- This situation provides a path from input D to the output, and for this reason, the circuit is often called a **transparent latch.**
- When the **enable input signal is de-asserted**, the binary information that was present at the data input at the time the transition occurred is **retained** (i.e., stored) at the Q output until the enable input is asserted again.

Graphic Symbols for Latches

- A latch is designated by a rectangular block with inputs on the left and outputs on the right.
- One output designates the normal output, and the other (with the bubble designation) designates the complement output.



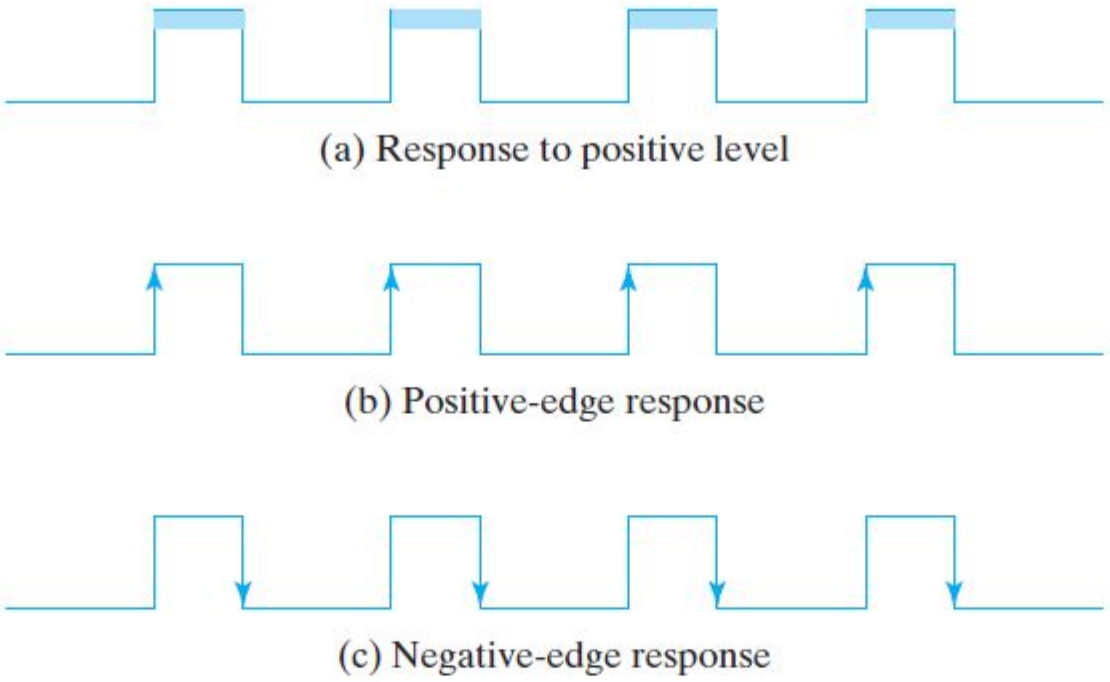
Flip-Flops T1- section 5.4

- When latches are used for the storage elements, a serious difficulty arises. The state transitions of the latches start as soon as the clock pulse changes to the logic-1 level. **The new state of a latch appears at the output while the pulse is still active.**

- **If the inputs applied to the latches change while the clock pulse is still at the logic-1 level,** the latches will respond to new values and a new output state may occur. The result is an unpredictable situation.

- **A flipflop triggers only during a signal transition (from 0 to 1 or from 1 to 0) of the synchronizing signal (clock) and is disabled during the rest of the clock pulse.**

Clock Response in Latch and Flip-Flop



The problem with the latch is that it responds to a change in the level of a clock pulse.

The key to the proper operation of a flip-flop is to trigger it only during a signal transition

A clock pulse goes through two transitions: from 0 to 1 and the return from 1 to 0.

Flip-Flop

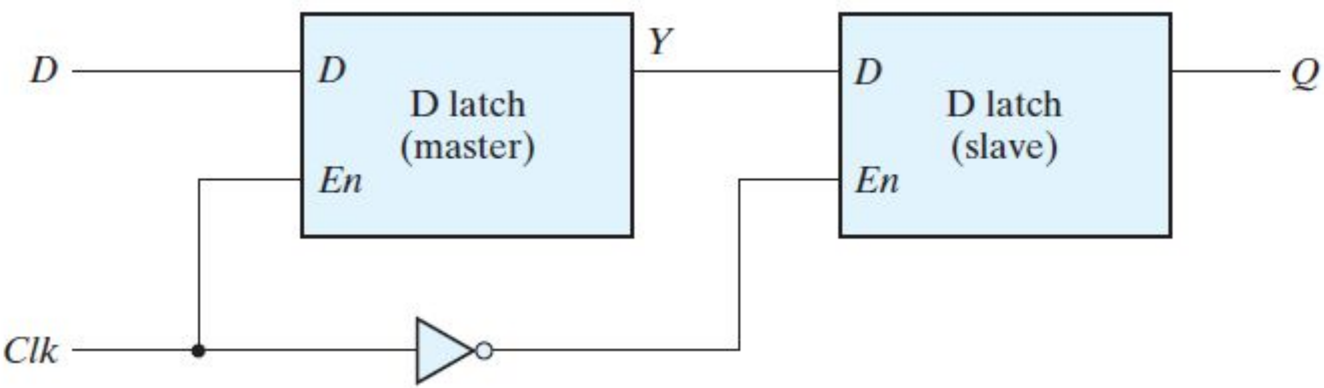
There are two ways that a latch can be modified to form a flip-flop

- One way is to employ two latches in a special configuration that isolates the output of the flip-flop and prevents it from being affected while the input to the flip-flop is changing.

- Another way is to produce a flip-flop that triggers only during a signal transition (from 0 to 1 or from 1 to 0) of the synchronizing signal (clock) and is disabled during the rest of the clock pulse.

Edge-Triggered D Flip-Flop

- The construction of a D flip-flop with two D latches and an inverter:



- Also called **Master Slave D Flip-Flop**.
- The first latch is called the master and the second the slave.
- The circuit samples the D input and changes its output Q only at the negative edge of the synchronizing or controlling clock (designated as Clk).

Edge-Triggered D Flip-Flop

- When clk=1 master is enabled and clk=0 slave is enabled.
- Any change in the input changes the master output at Y, but cannot affect the slave output.

- Thus, a change in the output of the flip-flop can be triggered only by and during the transition of the clock from 1 to 0.

- The behavior of the master–slave flip-flop just described dictates that
    - the output may change only once,
    - a change in the output is triggered by the negative edge of the clock,
    - the change may occur only during the clock’s negative level.

- The value that is produced at the output of the flip-flop is the value that was stored in the master stage immediately before the negative edge occurred .

Edge-Triggered D Flip-Flop

- It is also possible to design the circuit so that the flip-flop output changes on the positive edge of the clock.
- In sum, when the input clock in the positive-edge-triggered flip-flop makes a positive transition, the value of D is transferred to Q . A negative transition of the clock (i.e., from 1 to 0) does not affect the output.
- There is a minimum time called **the setup time** during which the D input must be maintained at a constant value prior to the occurrence of the clock transition
- minimum time called the **hold time** during which the D input must not change after the application of the positive transition of the clock.
- The **propagation delay time** of the flip-flop is defined as the interval between the trigger edge and the stabilization of the output to a new state.

Edge-Triggered D Flip-Flop

Parameter	Definition	Example Value	Illustration
Setup Time	Minimum time `D` must be stable **before** the clock edge	5 ns	If clock ↑ at 20 ns → `D` must be stable from 15 ns onwards
Hold Time	Minimum time `D` must remain stable **after** the clock edge	2 ns	If clock ↑ at 20 ns → `D` must remain unchanged until 22 ns
Propagation Delay	Time taken for output `Q` to update after the clock edge	8 ns	If clock ↑ at 20 ns → output `Q` updates at ~28 ns

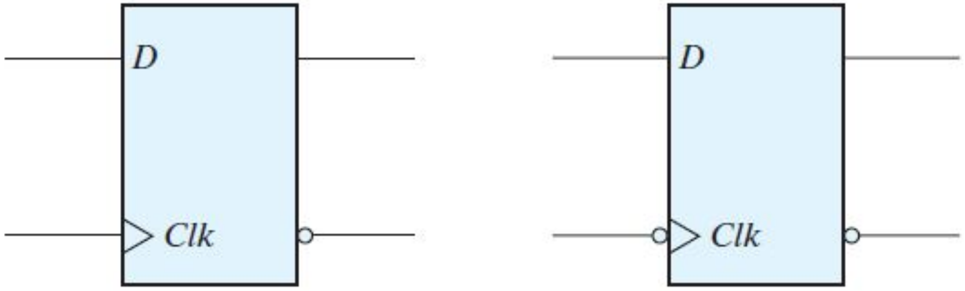
Real-Life Analogy

Think of it like catching a photo with a camera:

- **Setup time** → Subject must pose *before* the shutter clicks.
- **Hold time** → Subject must stay still *just after* the shutter clicks.
- **Propagation delay** → Time it takes for the photo to actually appear on screen after clicking.

Edge-Triggered D Flip-Flop

- Graphic symbol for edge-triggered D flip-flop:



(a) Positive-edge (a) Negative-edge

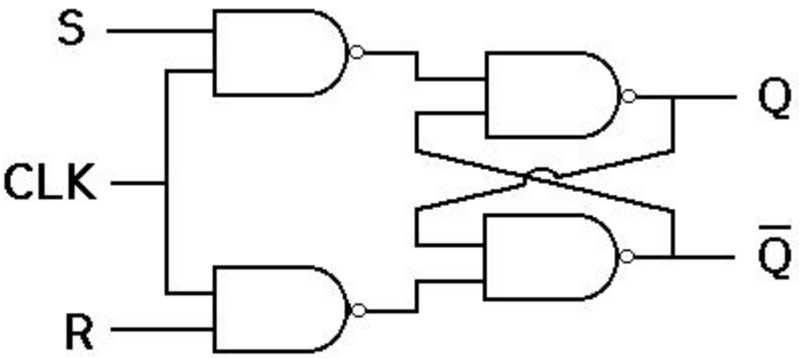
- The dynamic indicator (>) denotes the fact that the flip-flop responds to the edge transition of the clock.
- A bubble outside the block adjacent to the dynamic indicator designates a negative edge for triggering the circuit.
- The absence of a bubble designates a positive-edge response.

Flip-Flops

Flip-Flops have following representation:

- Function table- output Q and Q`
- Characteristic table- They define next state
- Characteristic equation-solve using K map- The logical properties of a flip-flop, as described in the characteristic table, can be expressed algebraically with a characteristic equation.
- Excitation table- predict the i/p based on current state and next state to build state diagram

SR Flip-Flops Characteristic table and Characteristic equation



En	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	$Q = 0$; reset state
1	1	0	$Q = 1$; set state
1	1	1	Indeterminate

(b) Function table

Characteristic table

S	R	Q_N	Q_{N+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	-
1	1	1	-

Characteristic equation by simplifying using K maps

$Q_{n+1} = S + R`Q_n$

Excitation Table

An excitation table shows the minimum inputs that are necessary to generate a particular next state when the current state is known.

The excitation tables are used to determine the inputs of the flip-flop when the present state and the next state to which the flip-flop goes after the occurrence of the clock pulse are known.

Excitation Table and State Diagram

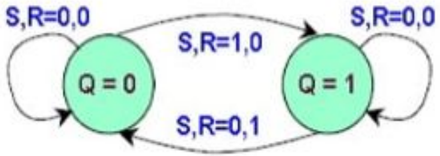
S	R	Present state Q_n	Next state Q_{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	x
1	1	1	x

Invalid states

Truth table of SR flip flop

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

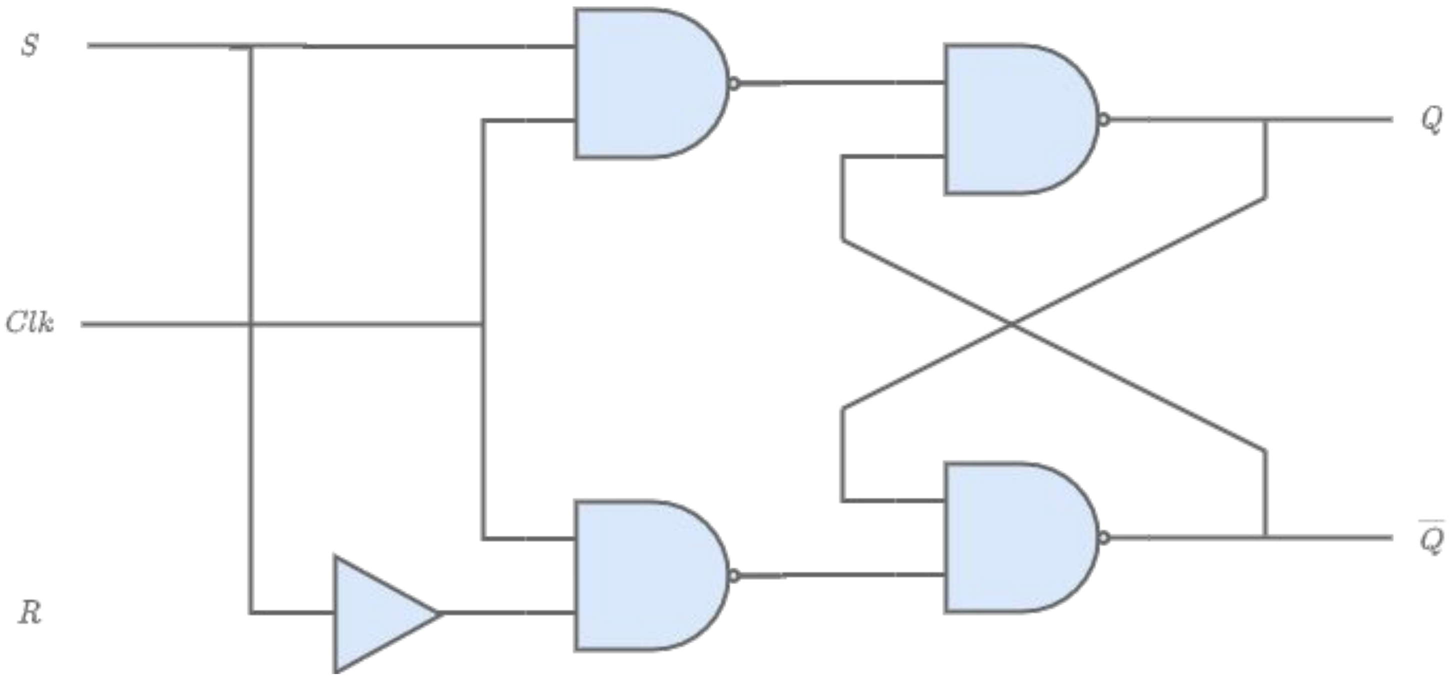
Excitation table of SR flip flop



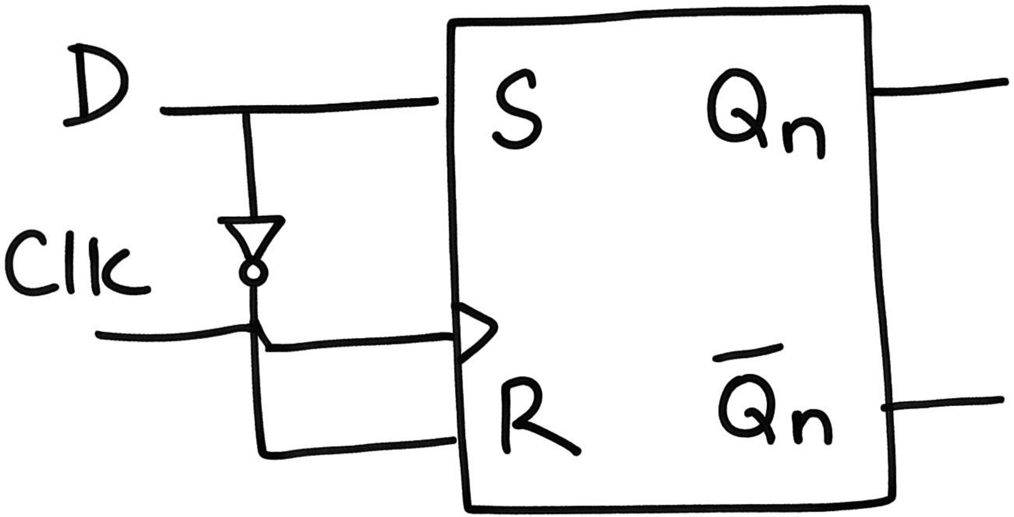
State Diagram

Source: Electrically4u

D Flip-Flop



D Flip-Flop



D Flip-Flop- Characterstic Table and Characterstic Equation

En	D	Next state of Q
0	X	No change
1	0	$Q = 0$; reset state
1	1	$Q = 1$; set state

(b) Function table

D Flip-Flop

D	$Q(t+1)$	
0	0	Reset
1	1	Set

D	Q_n	Q_{n+1}
0	0	0
0	1	0
1	0	1
1	1	1

Simplifying using K maps- characteristic equation is

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

D Flip-Flop- excitation table

Qn	Q(n+1)	D
0	0	0
0	1	1
1	0	0
1	1	1

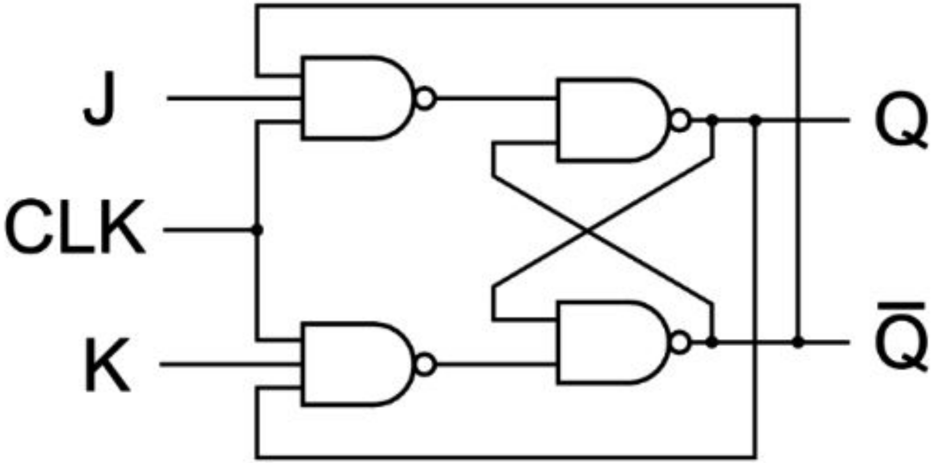
Excitation Table

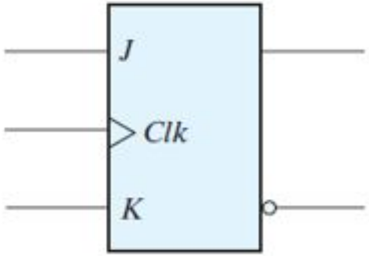
State Diagram

JK Flip-Flop

A **JK flip-flop** is required mainly to overcome the limitations of the **SR flip-flop** and to provide a more versatile storage element in sequential circuits.

In an **SR flip-flop**, when **S=1** and **R=1** simultaneously, the output becomes indeterminate (invalid).





(b) Graphic symbol

JK Flip-Flop-function table

Clk	J	K	Q
0	x	x	Qn
1	0	0	Qn
1	0	1	0
1	1	0	1
1	1	1	$\overline{Qn}$

Memory

Toggle race around condition

JK Flip-Flop

Characteristics table

Q(n)	J	K	Q(n+1)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Solve using K map to get characterstic equation:

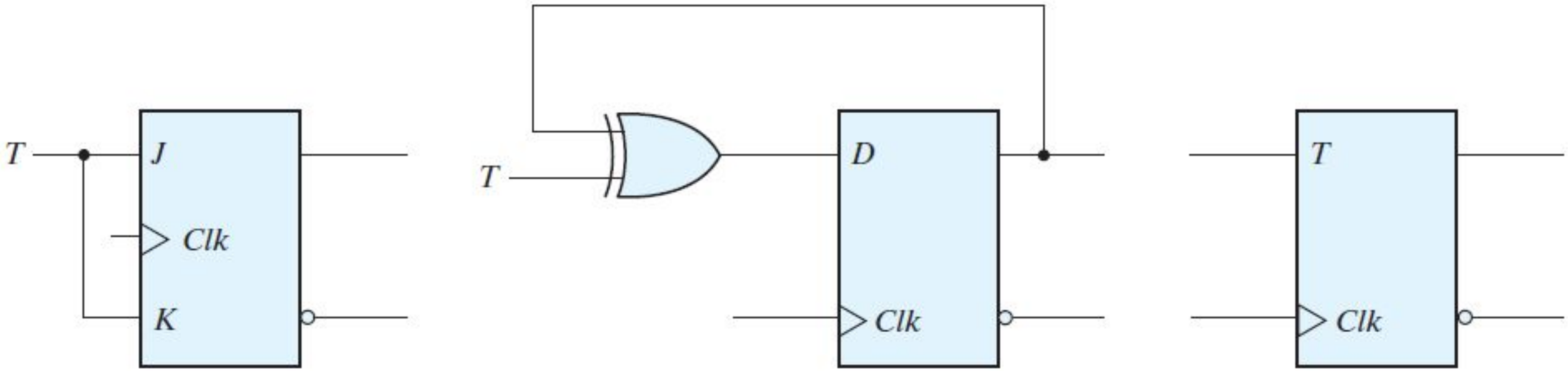
$$Q_{n+1} = Q_n K` + Q_n`J$$

JK Flip-Flop

Excitation table

Q(n)	Q(n+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

T Flip-Flop (Toggle Flip-Flop)



(a) From *JK* flip-flop (b) From *D* flip-flop (c) Graphic symbol

T Flip-Flop (Toggle Flip-Flop)

Truth table- if clk=0 then previous state

CLK	T	Q_{n+1}
↑	0	Q_n
↑	1	Q_n'

T Flip-Flop (Toggle Flip-Flop)

Characteristic table and equations:

T	Qn	Qn+1
0	0	0
0	1	1
1	0	1
1	1	0

$$Q(t+1) = T \oplus Q = TQ' + T'Q$$

Excitation table

Qn	Qn+1	T
0	0	0
0	1	1
1	0	1
1	1	0

Flip-Flops

- JK flip-flop can function as:
- **SR flip-flop** (J = S, K = R)
- **T flip-flop** (J=K=1 → toggles)
- **D flip-flop** (J=D, K=¬D)
- Hence, it's a **universal flip-flop** that can emulate other types.

Characteristics Table -summary

PES UNIVERSITY

Flip-Flop Characteristic Tables

JK Flip-Flop

J	K	$Q(t+1)$	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

D Flip-Flop

D	$Q(t+1)$	
0	0	Reset
1	1	Set

T Flip-Flop

T	$Q(t+1)$	
0	$Q(t)$	No change
1	$Q'(t)$	Complement

Characteristics Equations-summary

The logical properties of a flip-flop, as described in the characteristic table, can be expressed algebraically with a characteristic equation. For the D flip-flop, we have the characteristic equation

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

which states that the next state of the output will be equal to the value of input D in the present state. The characteristic equation for the JK flip-flop can be derived from the characteristic table or from the circuit of Fig. 5.12. We obtain

$$Q(t + 1) = JQ' + K'Q$$

where Q is the value of the flip-flop output prior to the application of a clock edge. The characteristic equation for the T flip-flop is obtained from the circuit of Fig. 5.13:

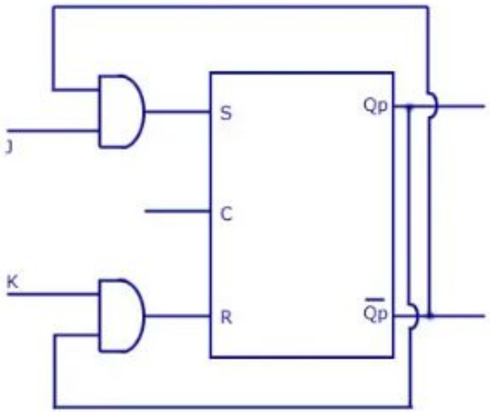
$$Q(t + 1) = T \oplus Q = TQ' + T'Q$$

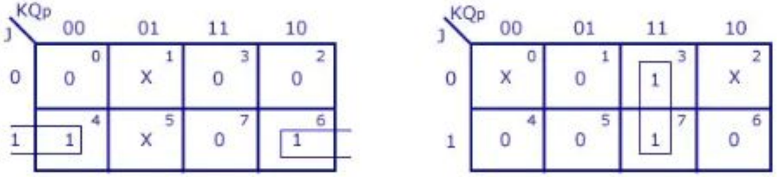
SR Flip-Flop to JK Flip-Flop

Conversion Table

J-K Inputs		Outputs		S-R Inputs	
J	K	Qp	Qp+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

Logic Diagram





$S = \overline{J}Qp$

$R = KQp$

K-Map

www.CircuitsToday.com

https://www.geeksforgeeks.org/digital-logic/flip-flop-types-their-conversion-and-applications/

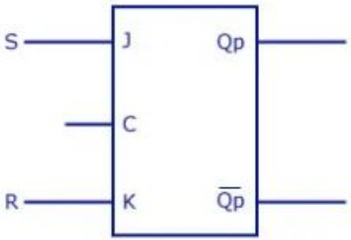
JK Flip-Flop to SR Flip-Flop

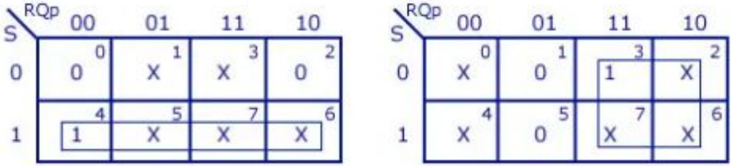
J-K Flip Flop to S-R Flip Flop

Conversion Table

S-R Inputs		Outputs		J-K Inputs	
S	R	Qp	Qp+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	Invalid		Dont care	
1	1	Invalid		Dont care	

Logic Diagram





J=S K-maps K=R

www.CircuitsToday.com

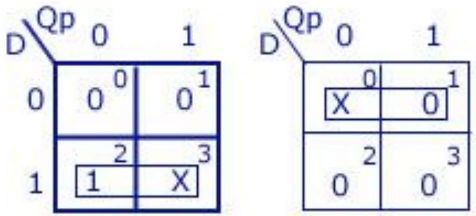
SR Flip-Flop to D Flip-Flop

S-R Flip Flop to D Flip Flop

Conversion Table

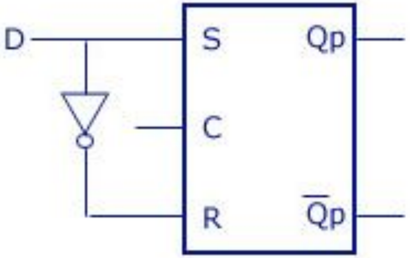
D Input	Outputs		S-R Inputs	
	Qp	Qp+1	S	R
0	0	0	0	X
0	1	0	0	1
1	0	1	1	0
1	1	1	X	0

K-maps



$S = D$ $\quad$ $R = \overline{D}$

Logic Diagram



www.CircuitsToday.com

D Flip-Flop to SR Flip-Flop

Conversion Table

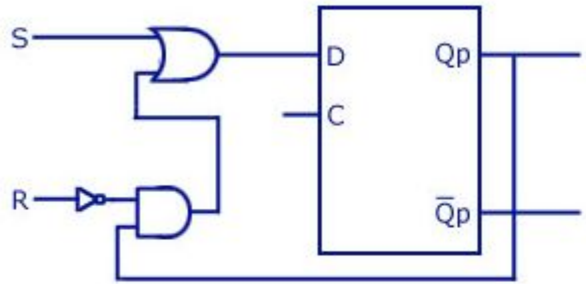
S-R Inputs		Outputs		D Input
S	R	Qp	Qp+1	
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	Invalid		Dont care
1	1	Invalid		Dont care

K-map

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

D = S+$\overline{R}$Qn

Logic Diagram



www.CircuitsToday.com

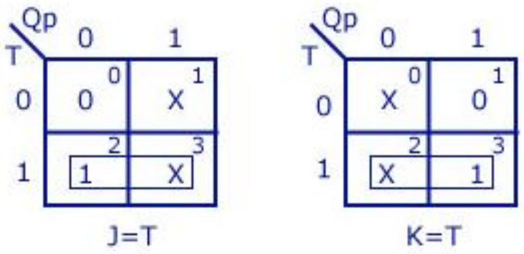
JK Flip-Flop to T Flip-Flop

J-K Flip Flop to T Flip Flop

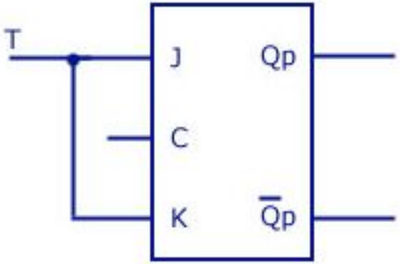
Conversion Table

T Input	Outputs Qp	Outputs Qp+1	J-K Inputs J	J-K Inputs K
0	0	0	0	X
0	1	1	X	0
1	0	1	1	X
1	1	0	X	1

K-maps



Logic Diagram



www.CircuitsToday.com

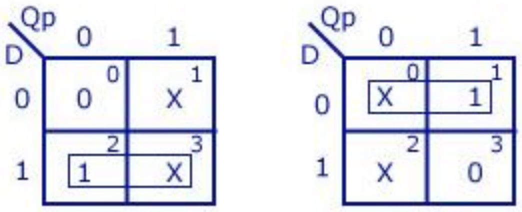
JK Flip-Flop to D Flip-Flop

J-K Flip Flop to D Flip Flop

Conversion Table

D Input	Outputs Qp	Outputs Qp+1	J-K Inputs J	J-K Inputs K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	X
1	1	0	X	0

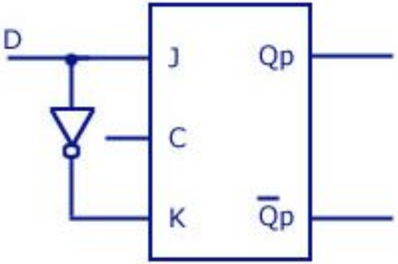
K-maps



J=D

$K=\overline{D}$

Logic Diagram



www.CircuitsToday.com

D Flip-Flop to JK Flip-Flop

D Flip Flop to J-K Flip Flop

Conversion Table

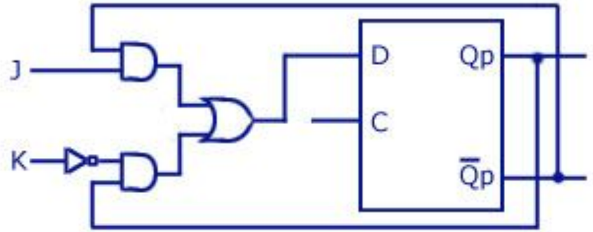
J-K Input J	J-K Input K	Outputs Qp	Outputs Qp+1	D Input
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	0	0

K-map

J \ KQp	00	01	11	10
1	0 (0)	1 (1)	0 (3)	0 (2)
1	1 (4)	1 (5)	0 (7)	1 (6)

$$D = J\overline{Qp} + \overline{K}Qp$$

Logic Diagram



www.CircuitsToday.com

Applications of Flipflops and Latches

- **Electronic Voting Machine**
- **Scenario:** Registering and storing votes securely.
- **How flip-flops help:** Flip-flops store the vote data bit-by-bit, allowing fast, reliable vote counting.

Applications of Flipflops and Latches

Flip-Flop Type	Real-Time Example	How It Works in This Example	Why This Flip-Flop Is Suitable
JK Flip-Flop	**Digital Watch Seconds Counter**	Counts seconds by toggling state every clock pulse, progressing through time steps in binary.	JK flip-flop toggles easily on J=K=1, ideal for counters.
D Flip-Flop	**Computer Register (CPU Storage)**	Holds the current data/instruction being processed, updated only on clock edges for accuracy.	D flip-flop stores data bits reliably, synchronized to clock.
SR Flip-Flop	**Elevator Door Control System**	Sets door to open (set) or closed (reset) state based on user inputs or safety sensors.	Simple set/reset functionality matches door open/close signals.
T Flip-Flop	**LED Blinking Circuit**	Toggles LED ON and OFF on each clock pulse to create blinking effect.	T flip-flop toggles output state each pulse, perfect for blinking LEDs.

MCQ

1. **Which of the following statements about a latch is TRUE?**
   A) A latch is edge-triggered.
   B) A latch is level-sensitive.
   C) A latch can only store analog values.
   D) A latch cannot hold its state without a clock signal.

2. **In an SR latch constructed with NOR gates, what is the output when both S and R inputs are 0?**
   A) Output is set (Q=1)
   B) Output is reset (Q=0)
   C) Output retains previous state
   D) Output is indeterminate/invalid

MCQ

1. **Which of the following statements about a latch is TRUE?**
   A) A latch is edge-triggered.
   B) A latch is level-sensitive.
   C) A latch can only store analog values.
   D) A latch cannot hold its state without a clock signal.
   Answer: B
2. **In an SR latch constructed with NOR gates, what is the output when both S and R inputs are 0?**
   A) Output is set (Q=1)
   B) Output is reset (Q=0)
   C) Output retains previous state
   D) Output is indeterminate/invalid
   Answer: C

MCQ

3. **In a JK flip-flop, when both J and K inputs are 1 and a clock pulse occurs, the output:**

A) Is set to 1

B) Is reset to 0

C) Toggles from its previous state

D) Remains unchanged

4. A D flip-flop can be constructed using:

A) Two SR latches connected in master-slave configuration.

B) Only combinational logic

C) Only NOR gates without feedback.

D) A JK flip-flop with both J and K tied to the D input.

MCQ

3. **In a JK flip-flop, when both J and K inputs are 1 and a clock pulse occurs, the output:**

A) Is set to 1

B) Is reset to 0

C) Toggles from its previous state

D) Remains unchanged

Answer: C

4. A D flip-flop can be constructed using:

A) Two SR latches connected in master-slave configuration.

B) Only combinational logic

C) Only NOR gates without feedback.

D) A JK flip-flop with both J and K tied to the D input.

Answer: A

THANK YOU

Team DDCO
Department of Computer Science and Engineering