Lecture # 8

Introduction to Sequential Circuits/Synchronous versus Asynchronous/Latches/Flip-Flops/Characteristic Tables and Equations

By: Muhammad Zain Uddin

email: zuddin@iba.edu.pk

بسم الله الرحمن الرحيم

Digital Logic Design

Muhammad Zain Uddin
Lecturer,
IBA

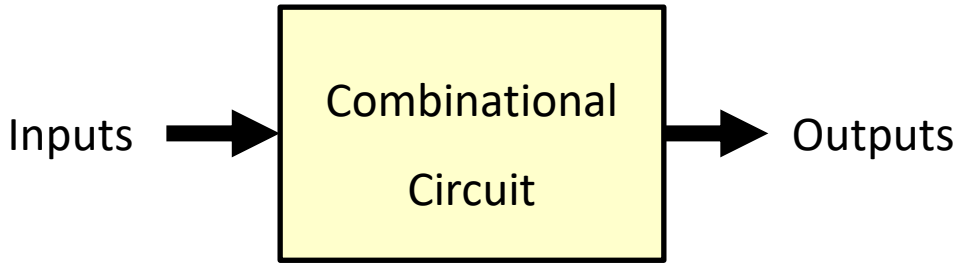
Combinational versus Sequential

Two classes of digital circuits

- Combinational Circuits
- Sequential Circuits

Combinational Circuit

- Outputs = F(Inputs)
- Function of Inputs only
- NO internal memory



Sequential Circuit

- Outputs is a function of Inputs and internal Memory
- There is an internal memory that stores the state of the circuit
- Time is very important: memory changes with time

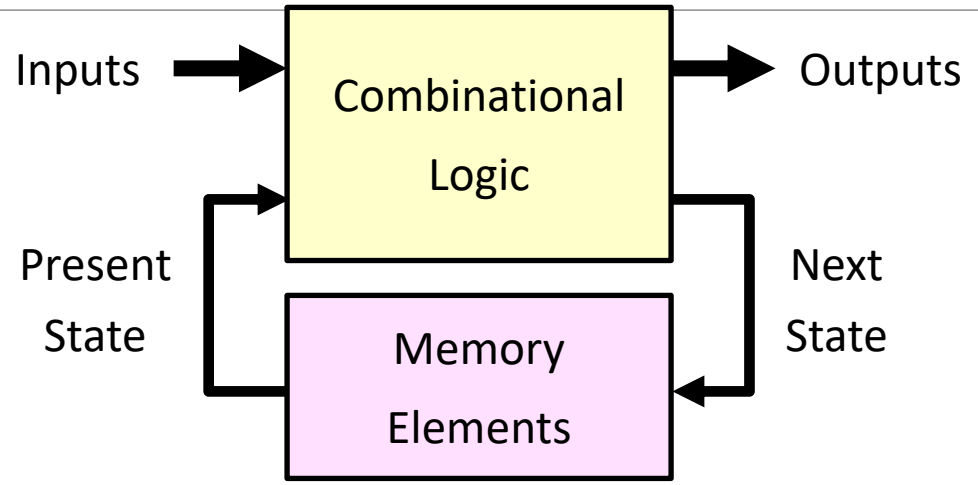
Introduction to Sequential Circuits

A Sequential circuit consists of:

1. Memory elements:
   - **Latches** or **Flip-Flops**
   - Store the **Present State**
2. Combinational Logic
   - Computes the **Outputs** of the circuit

     Outputs depend on Inputs and Current State
   - Computes the **Next State** of the circuit

     Next State also depends on the Inputs and the Present State

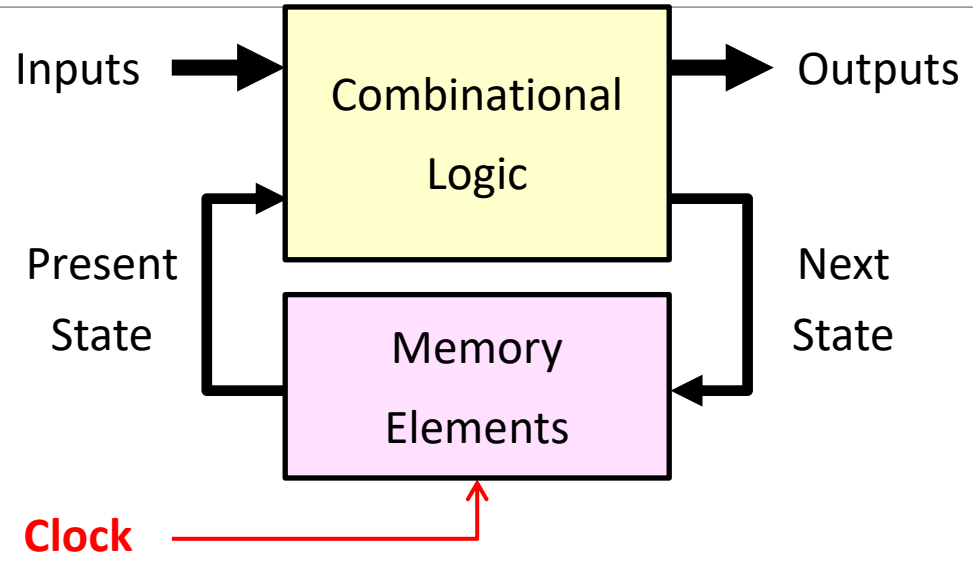


Two Types of Sequential Circuits

1. **Synchronous** Sequential Circuit
   - Uses a clock signal as an additional input
   - Changes in the memory elements are controlled by the clock
   - Changes happen at discrete instances of time
2. **Asynchronous** Sequential Circuit
   - No clock signal
   - Changes in the memory elements can happen at any instance of time

   Our focus will be on Synchronous Sequential Circuits
   - Easier to design and analyze than asynchronous sequential circuits

Synchronous Sequential Circuits



Synchronous sequential circuits use a **clock signal**

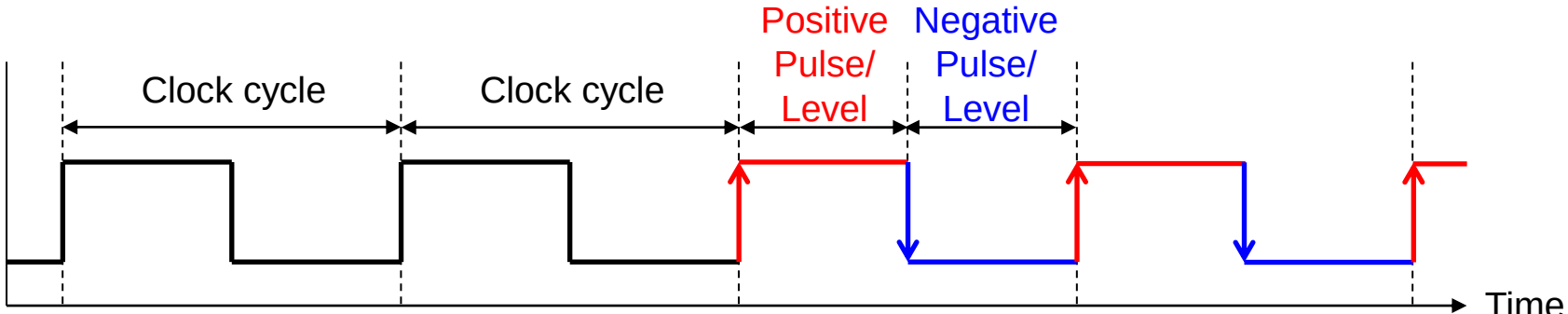
The clock signal is an input to the memory elements

The clock determines **when** the memory should be updated

The **present state** = output value of memory (stored)

The **next state** = input value to memory (not stored yet)

The Clock



Clock is a periodic signal = Train of pulses (1's and 0's)

The same clock cycle repeats indefinitely over time

Positive Pulse: when the **level** of the clock is **1**

Negative Pulse: when the **level** of the clock is **0**

Rising Edge: when the clock goes **from 0 to 1**

Falling Edge: when the clock goes **from 1 down to 0**

Memory Elements

Memory can store and maintain binary state (0's or 1's)

- Until directed by an input signal to change state

Main difference between memory elements

- Number of inputs they have
- How the inputs affect the binary state

Two main types:

- Latches are level-sensitive (the level of the clock)
- Flip-Flops are edge-sensitive (sensitive to the edge of the clock)

Flip-Flips are used in synchronous sequential circuits

Flip-Flops are built with latches

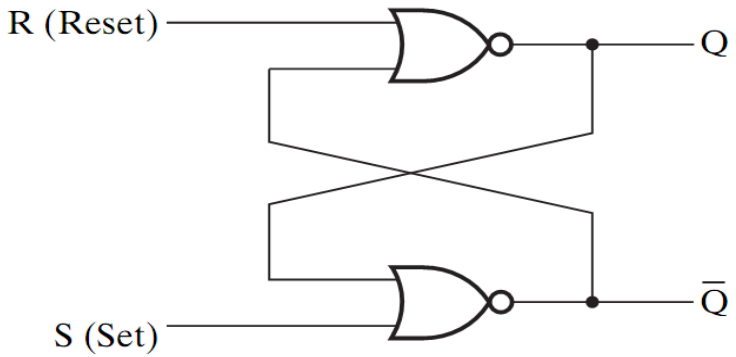
SR Latch

A **latch** is a memory element that can store 0 or 1

An **SR Latch** can be built using two **cross-coupled** NOR gates

Two inputs: S (Set) and R (Reset)

Two outputs: Q and $\overline{Q}$

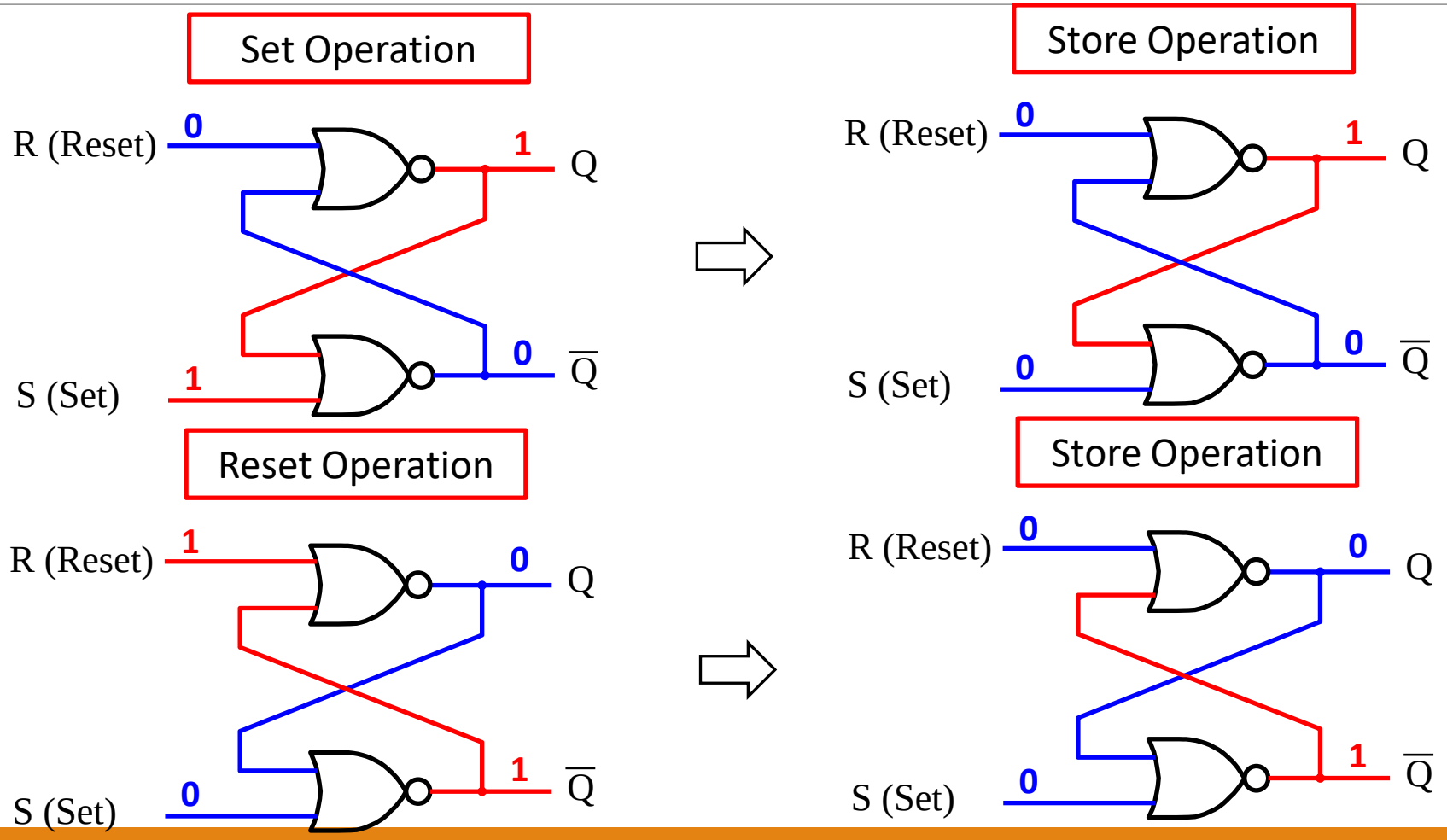


(a) Logic diagram

S	R	Q	$\overline{Q}$	
1	0	1	0	Set state
0	0	1	0	
0	1	0	1	Reset state
0	0	0	1	
1	1	0	0	Undefined

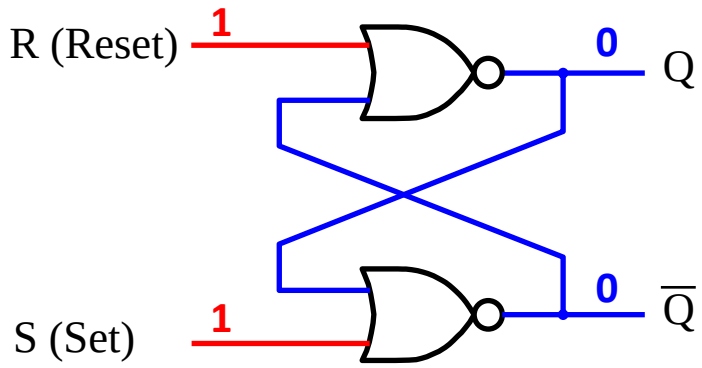
(b) Function table

SR Latch Operation



SR Latch Invalid Operation

Invalid Operation

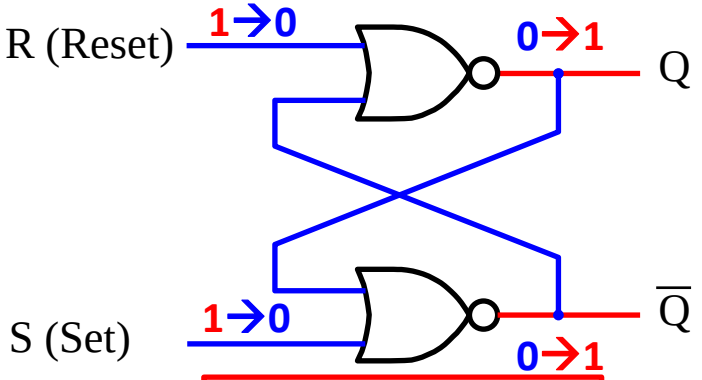


S = R = 1 should never be used

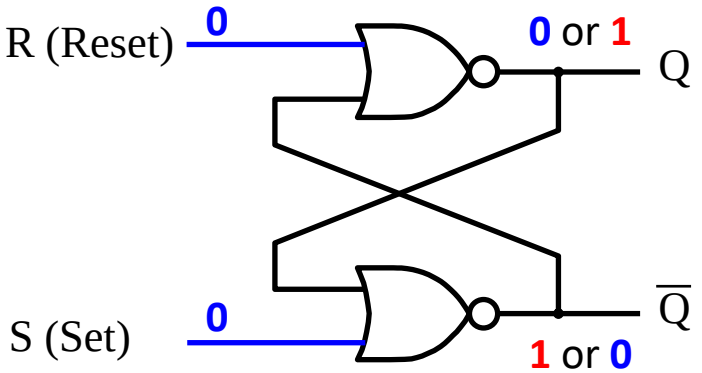
If S and R change from 1 → 0 simultaneously then race condition (oscillation) occurs

Final Q and $\overline{Q}$ are unknown

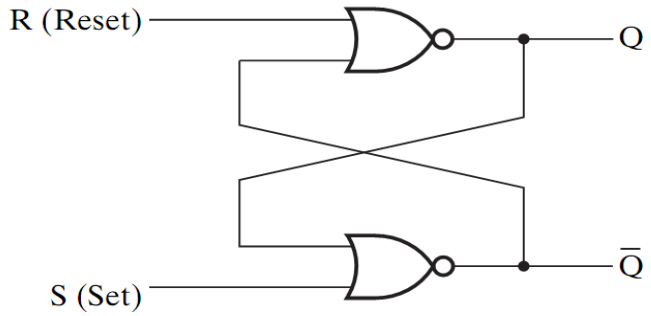
Race Condition



Unknown State



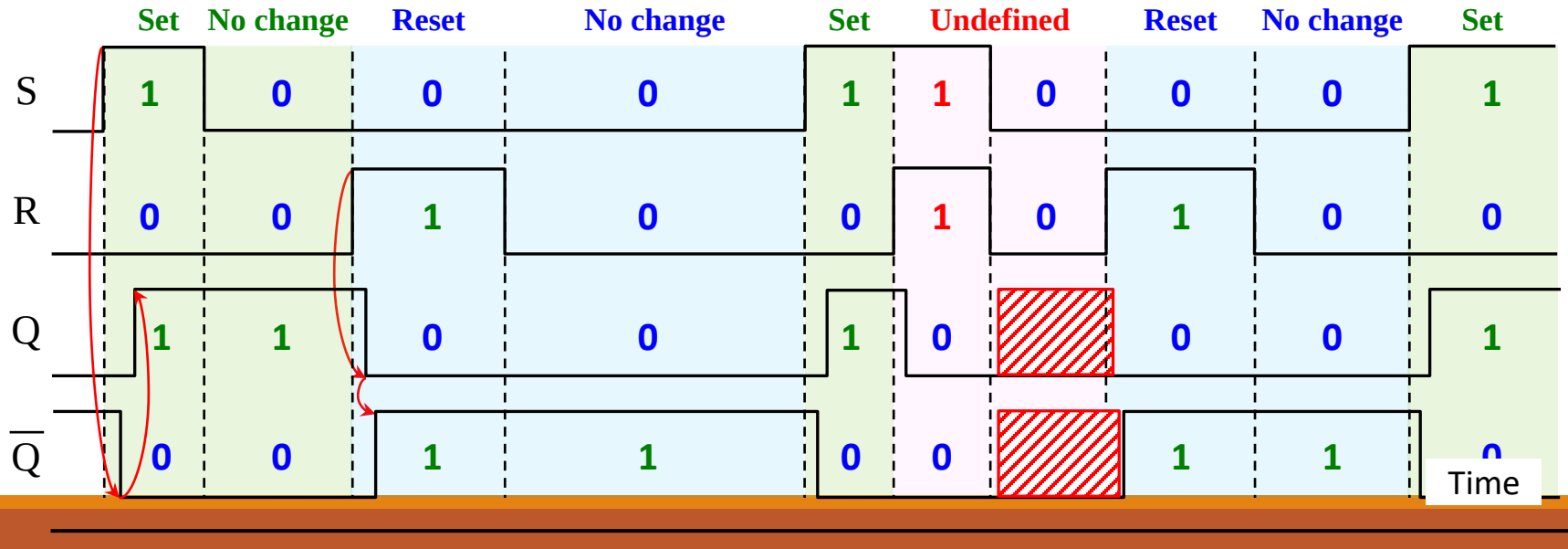
Timing Diagram of an SR Latch



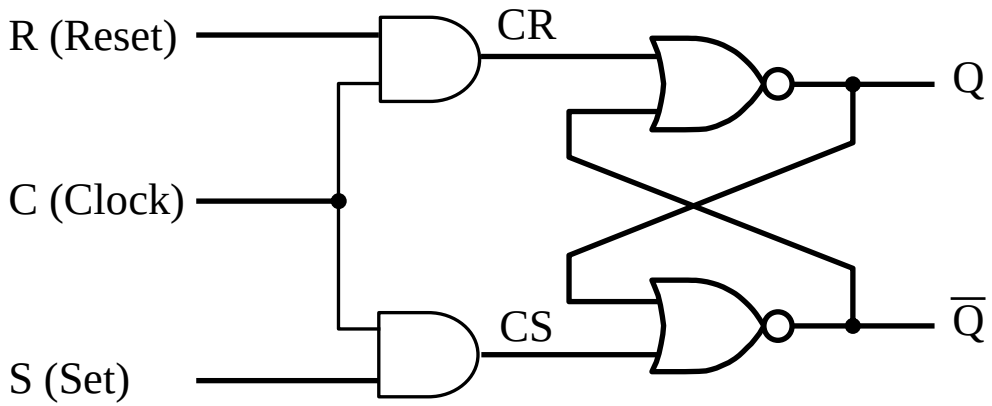
(a) Logic diagram

S	R	Q	$\overline{Q}$	
1	0	1	0	Set state
0	0	1	0	
0	1	0	1	Reset state
0	0	0	1	
1	1	0	0	Undefined

(b) Function table



Gated SR Latch with Clock Enable

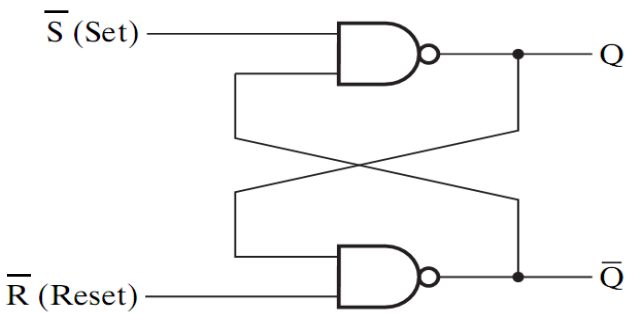


C	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	Q = 0; Reset state
1	1	0	Q = 1; Set state
1	1	1	Undefined

- An additional Clock (enable) input signal **C** is used
- Clock controls **when** the state of the latch can be changed
- When **C=0**, the S and R inputs have no effect on the latch

  The latch will remain in the same state, regardless of S and R
- When **C=1**, then normal SR latch operation

S R Latch with NAND Gates



(a) Logic diagram

$\overline{S}$	$\overline{R}$	Q	$\overline{Q}$	
0	1	1	0	Set state
1	1	1	0	
1	0	0	1	Reset state
1	1	0	1	
0	0	1	1	Undefined

(b) Function table

Known as the $\overline{S}\ \overline{R}$ Latch

If $\bar{S} = 0$ and $\bar{R} = 1$ then **Set** ($Q = 1,\ \overline{Q} = 0$)

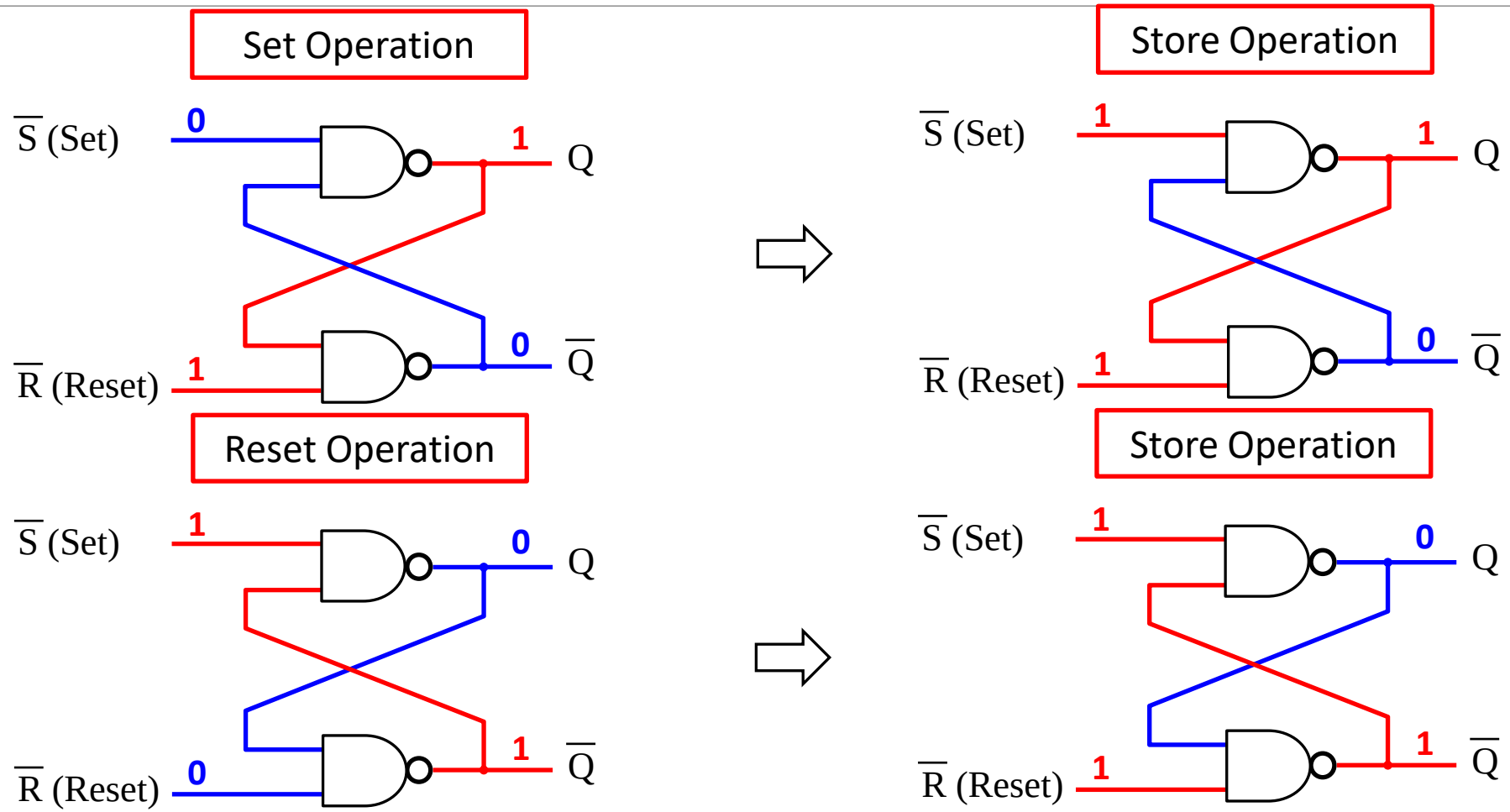
If $\bar{S} = 1$ and $\bar{R} = 0$ then **Reset** ($Q = 0,\ \overline{Q} = 1$)

When $\bar{S} = \bar{R} = 1$, Q and $\overline{Q}$ are unchanged (remain the same)

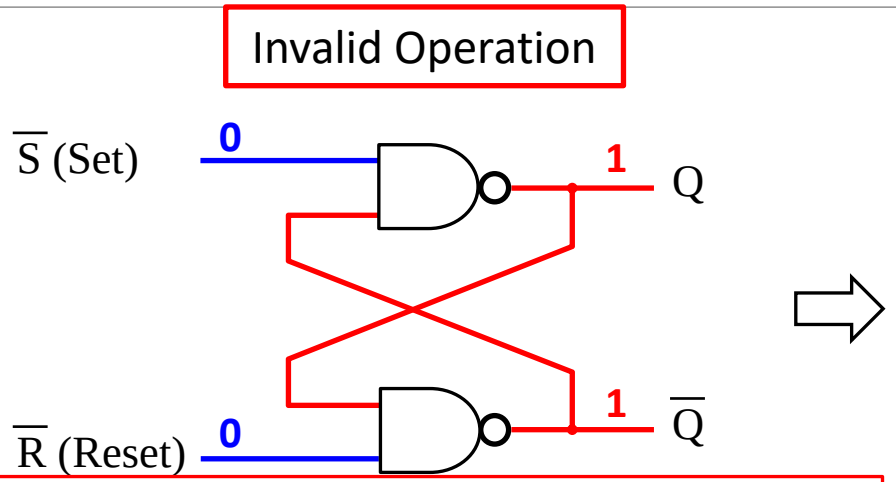
The latch stores its outputs Q and $\overline{Q}$ as long as $\bar{S} = \bar{R} = 1$

When $\bar{S} = \bar{R} = 0$, Q and $\overline{Q}$ are undefined (should never be used)

S R Latch Operation



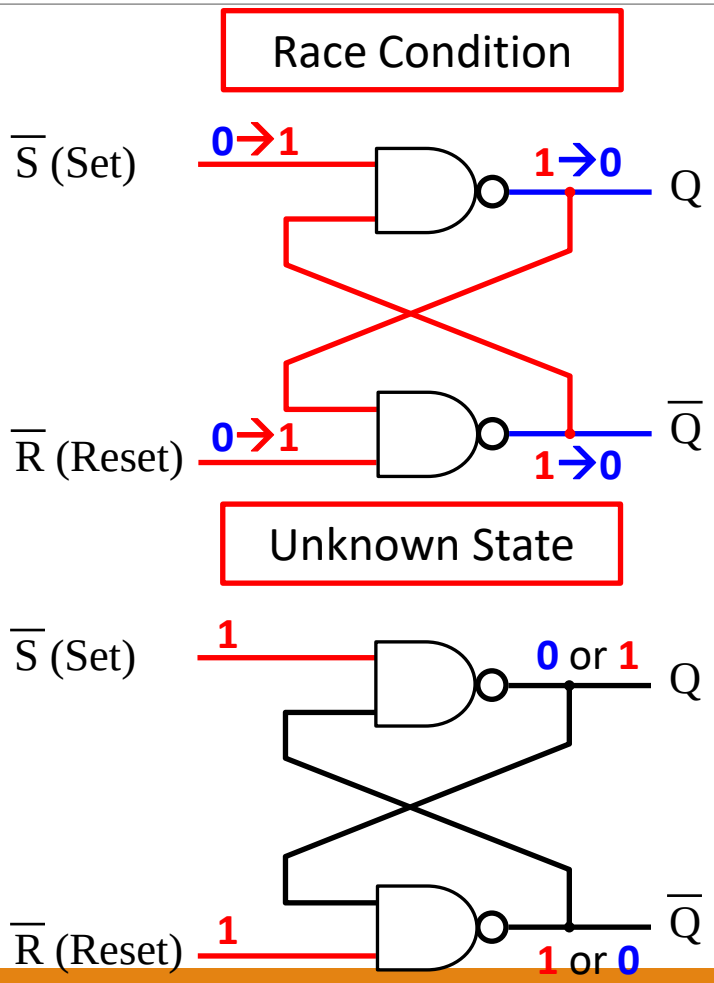
S R Latch Invalid Operation



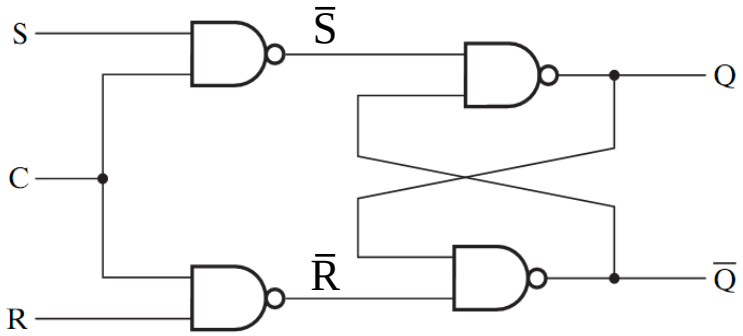
$\overline{S} = \overline{R} = 0$ should never be used

If $\overline{S}$ and $\overline{R}$ change from 0 → 1 simultaneously then race condition (oscillation) occurs

Final Q and $\overline{Q}$ are unknown



Gated SR Latch with Clock Enable



(a) Logic diagram

C	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	Undefined

(b) Function table

An additional Clock (enable) input signal **C** is used

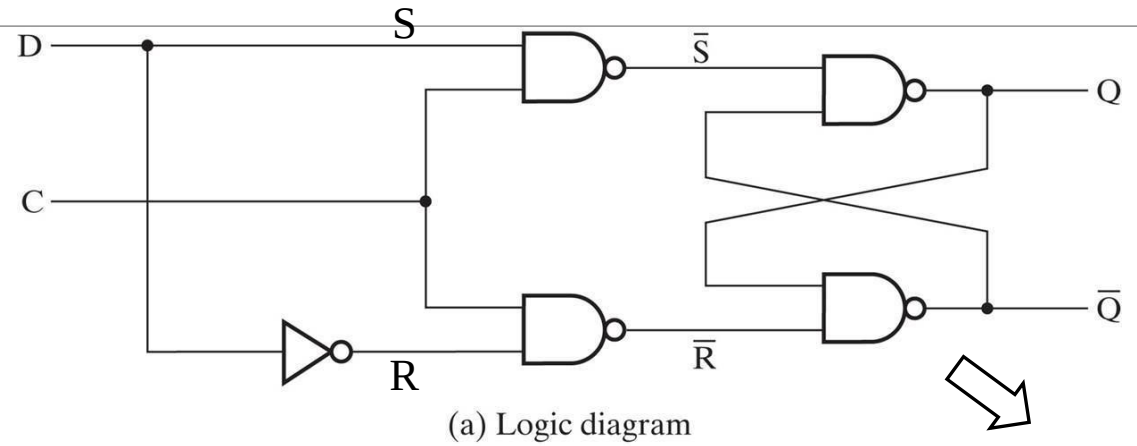
Clock controls **when** the state of the latch can be changed

When **C=0**, the latch remains in the same state

When **C=1**, then normal latch operation

The NAND gates invert the **S** and **R** inputs when **C=1**

D-Latch with Clock Enable



(a) Logic diagram

C	D	Next state of Q
0	X	No change
1	0	Q = 0; Reset state
1	1	Q = 1; Set state

(b) Function table

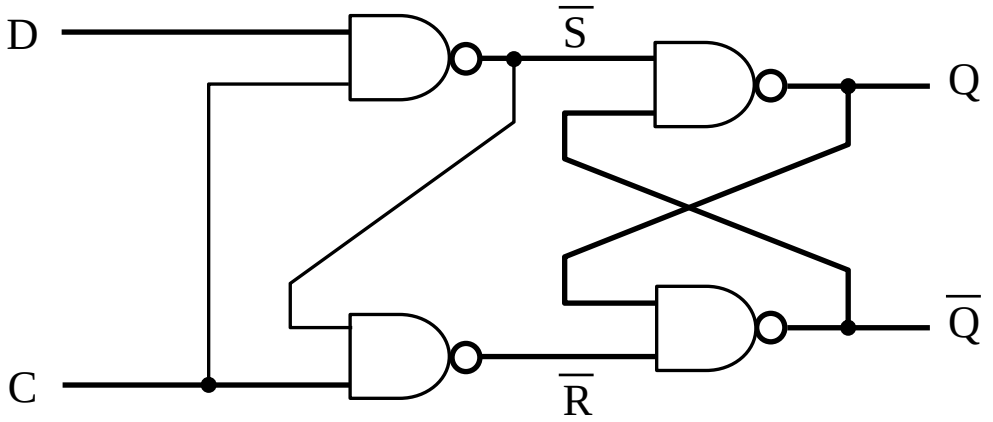
One data input D

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

No undefined state

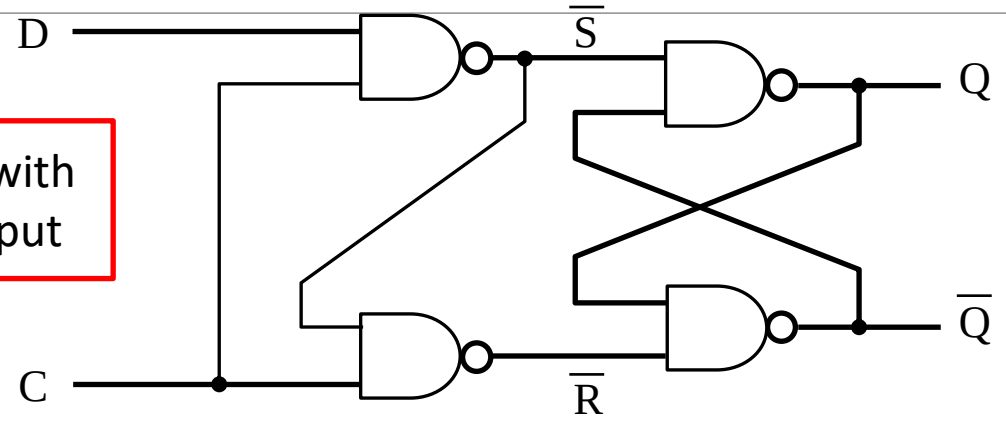
Inverter can be removed

When $C = 1, R = \bar{S} = \overline{D}$



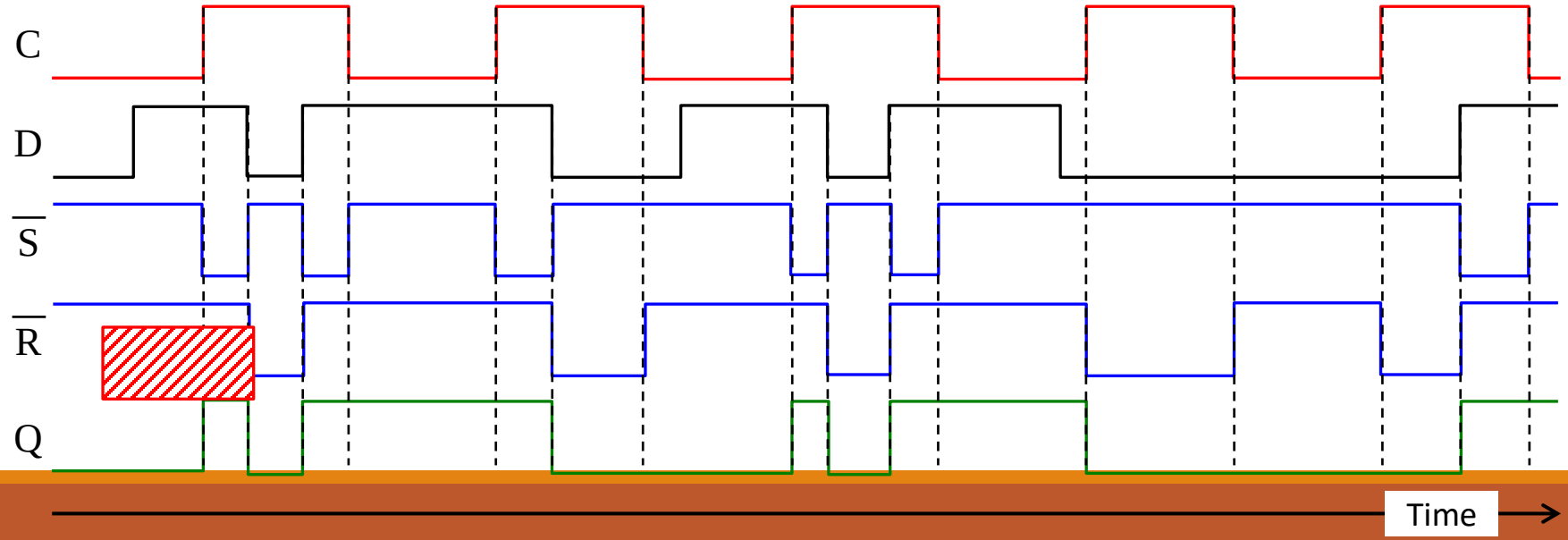
Timing of a D-Latch with Clock Enable

D-Latch with Clock Input

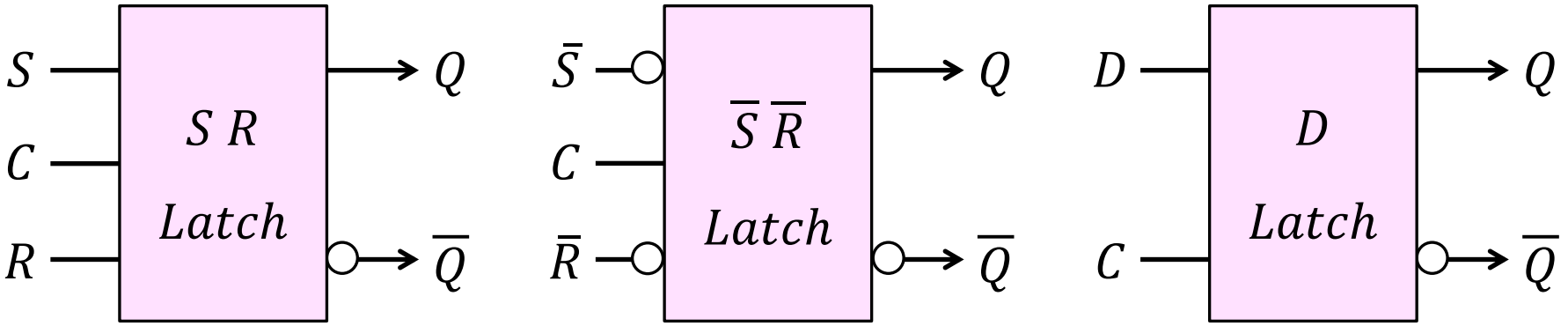


C	D	Next state of Q
0	X	No change
1	0	Q = 0; Reset state
1	1	Q = 1; Set state

Function table



Graphic Symbols for Latches



A bubble appears at the complemented output $\overline{Q}$

Indicates that $\overline{Q}$ is the complement of Q

A bubble also appears at the inputs of an $\overline{S}$ $\overline{R}$ latch

Indicates that **logic-0** is used (not logic-1) to set (or reset) the latch (as in the NAND latch implementation)

Problem with Latches

A latch is **level-sensitive** (sensitive to the level of the clock)

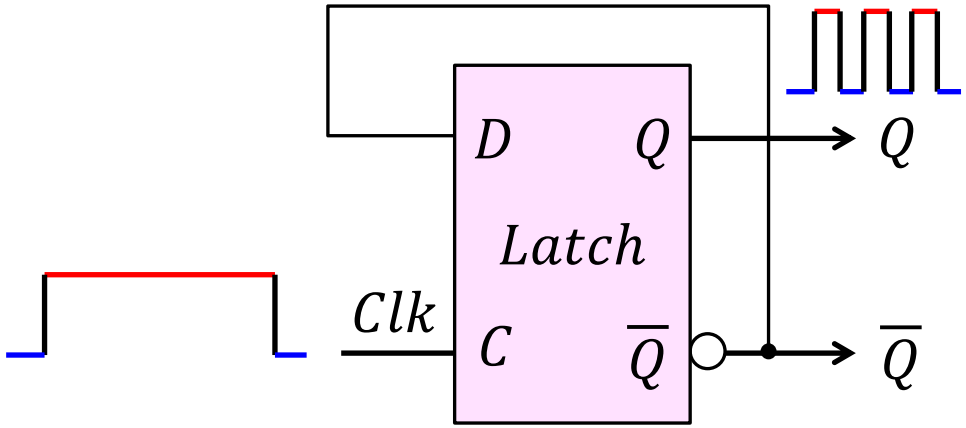
As long as the clock signal is **high** …

Any change in the value of input D appears in the output Q

Output Q keeps changing its value during a clock cycle

Final value of output Q is uncertain

Due to this uncertainty, latches are NOT used as memory elements in synchronous circuits



Flip-Flops

A **Flip-Flop** is a better memory element for synchronous circuits

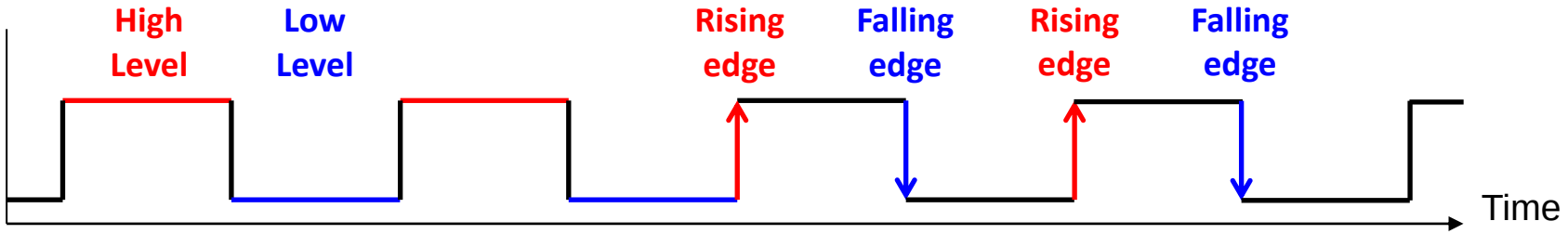
Solves the problem of latches in synchronous sequential circuits

A **latch** is sensitive to the **level** of the clock

However, a **flip-flop** is sensitive to the **edge** of the clock

A flip-flop is called an **edge-triggered** memory element

It changes it output value at the **edge** of the clock



Edge-Triggered D Flip-Flop

Built using two latches in a **master-slave** configuration

A master latch (D-type) receives external inputs

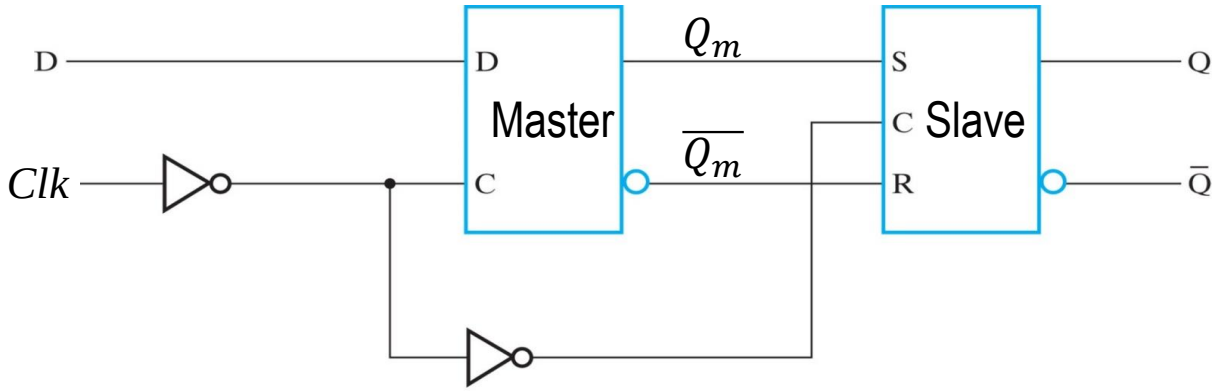
A slave latch (SR-type) receives inputs from the master latch

Only one latch is enabled at any given time

When **Clk=0**, the master is enabled and the D input is latched (slave disabled)

When **Clk=1**, the slave is enabled to generate the outputs (master is disabled)

Outputs change when *Clk* changes **from 0 to 1**



Negative Edge-Triggered D Flip-Flop

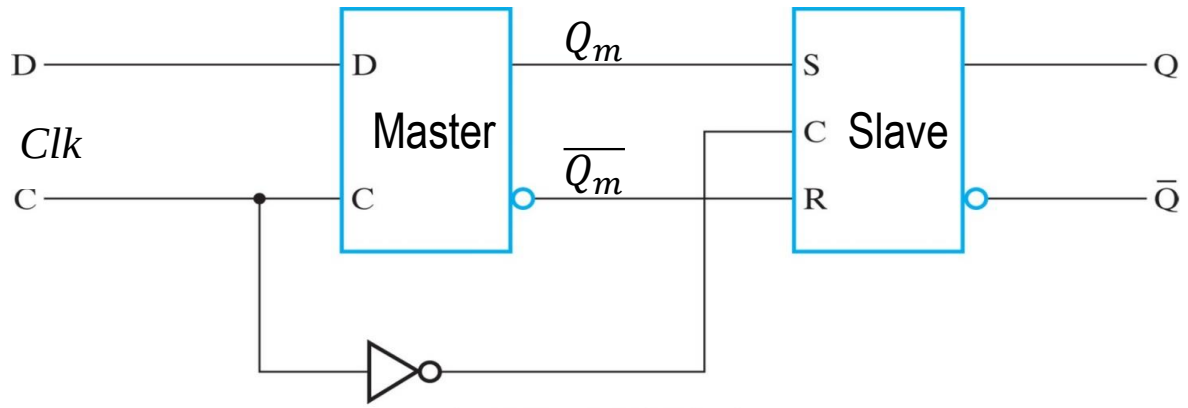
Similar to positive edge-triggered flip-flop

The first inverter at the Master C input is removed

Only one latch is enabled at any given time

When **Clk=1**, the master is enabled and the D input is latched (slave disabled)

When **Clk=0**, the slave is enabled to generate the outputs (master is disabled)



Outputs change when *Clk* changes **from 1 to 0**

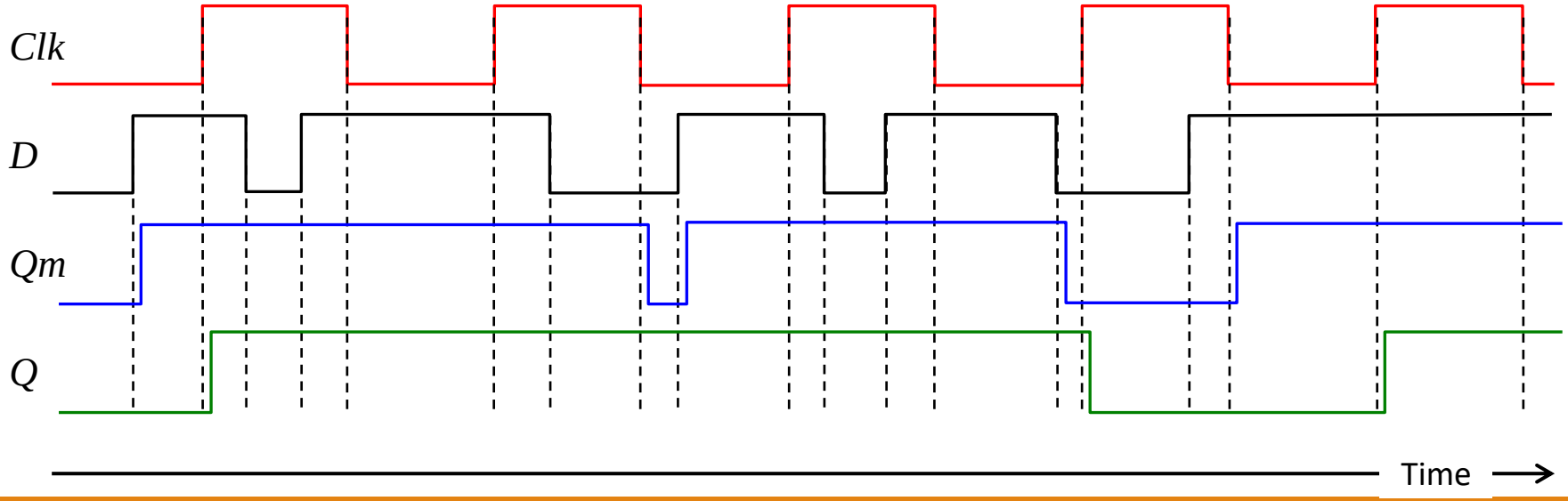
D Flip-Flop Timing Diagram

The diagram shows the timing of a positive-edge D Flip-Flop

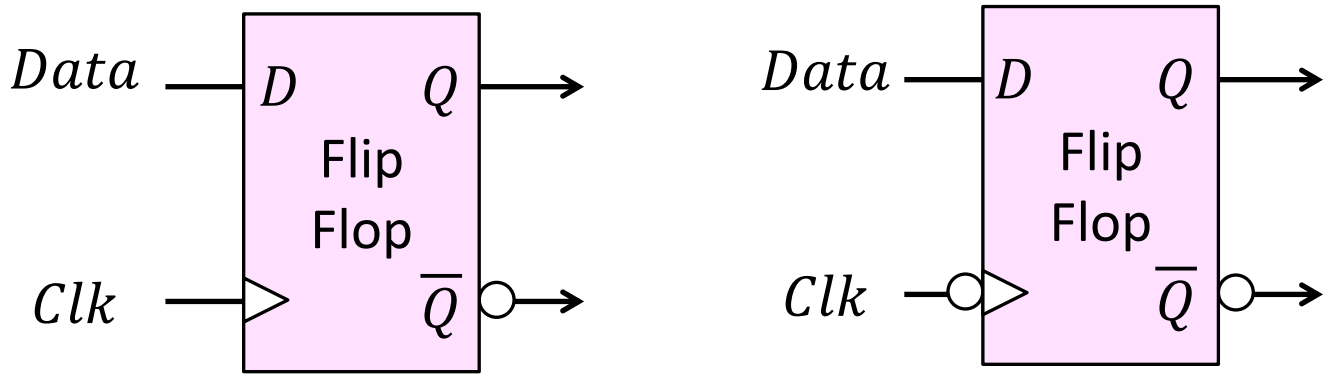
The master latch changes its output *Qm* when the clock *C* is 0

The rising edge of the clock triggers the D Flip-Flop

Notice the slight delay in the output *Q* after the rising edge



Graphic Symbols for Flip-Flops



A Flip-Flop has a similar symbol to a Latch

The difference is the arrowhead at the clock input

The arrowhead indicates sensitivity to the edge of the clock

A circle at the *Clk* input indicates negative edge-triggered FF

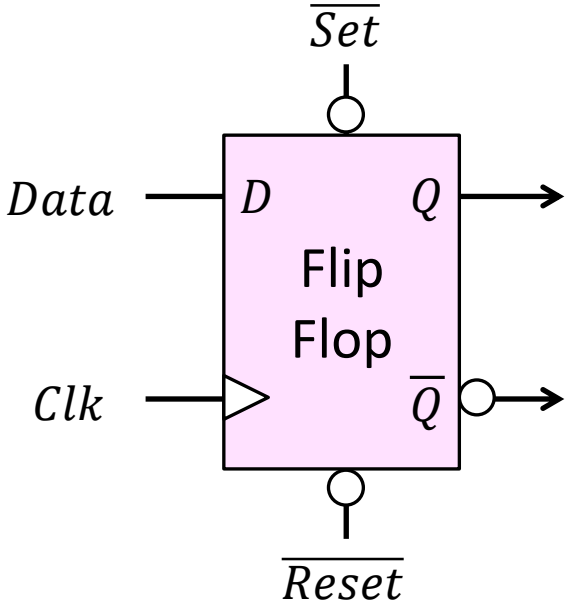
Asynchronous Set and Reset

When Flip-Flops are powered, their initial state is **unknown**

Some flip-flops have an **asynchronous Set** and **Reset** inputs

Set forces Q to become **1**, independently of the clock

Reset forces Q to become **0**, independently of the clock



Inputs				Outputs	
$\overline{Set}$	$\overline{Reset}$	$Data$	Clk	Q	$\overline{Q}$
0	1	X	X	1	0
1	0	X	X	0	1
1	1	0	↑	0	1
1	1	1	↑	1	0

Function Table

JK Flip-Flop

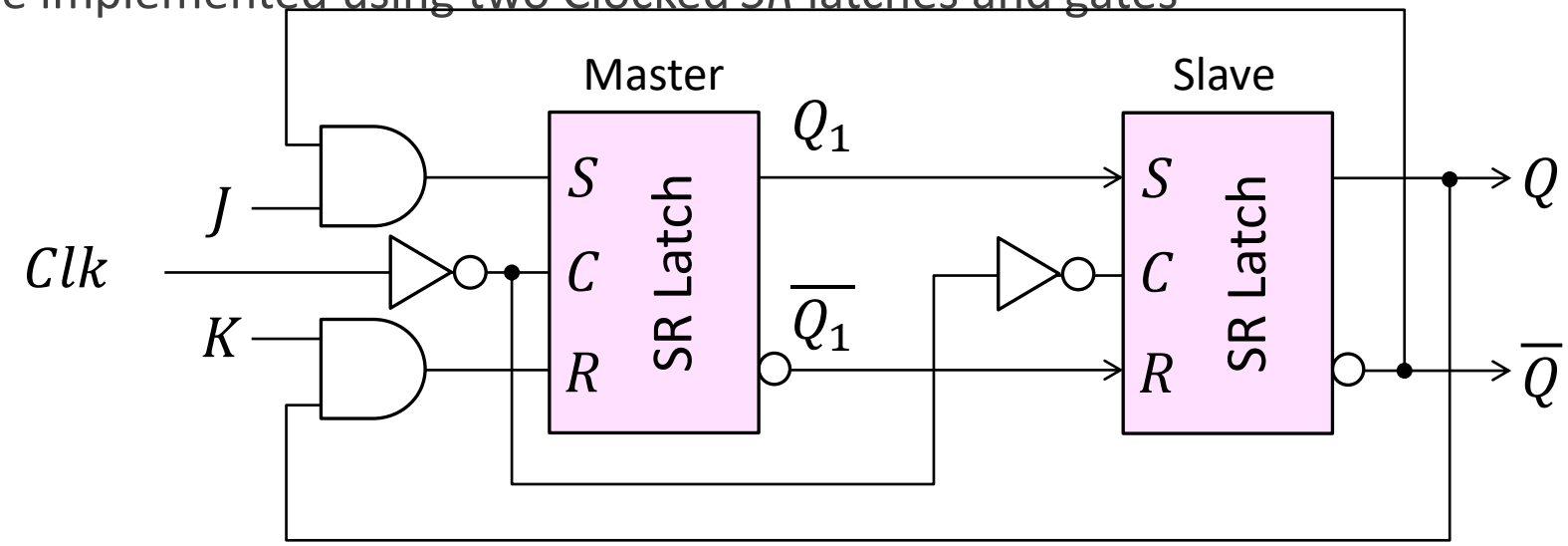
The *D* Flip-Flop is the most commonly used type

The *JK* is another type of Flip-Flop with inputs: *J, K*, and *Clk*

When *JK* = 10 ➔ Set, When *JK* = 01 ➔ Reset

When *JK* = 00 ➔ No change, When *JK* = 11 ➔ Invert outputs

JK can be implemented using two Clocked *SR* latches and gates



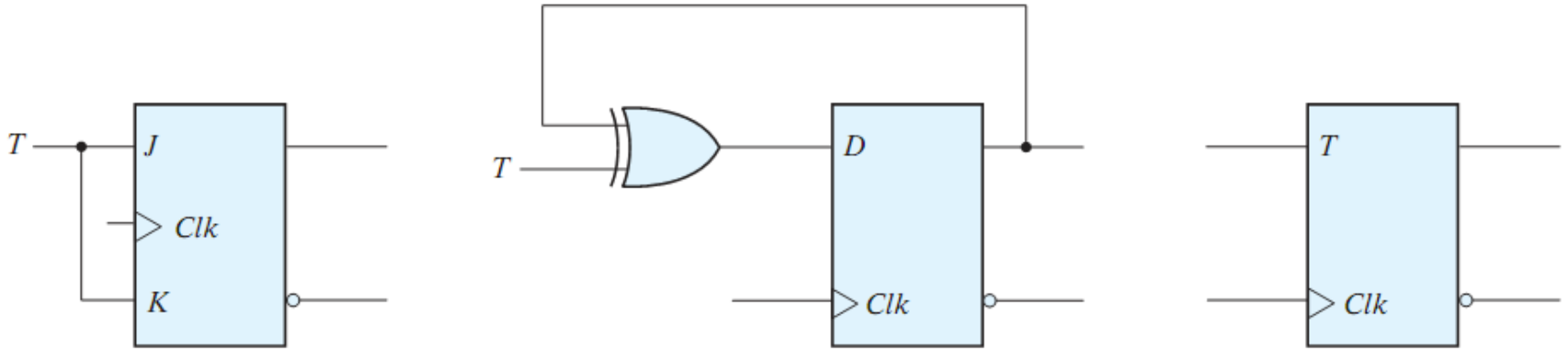
T Flip-Flop

The *T* (Toggle) flip-flop has inputs: *T* and *Clk*

When *T* = 0 ➔ No change, When *T* = 1 ➔ Invert outputs

The *T* flip-flop can be implemented using a *JK* flip-flop

It can also be implemented using a *D* flip-flop and a XOR gate



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

Flip-Flop Characteristic Table

Defines the operation of a flip-flop in a tabular form

Next state is defined in terms of the current state and the inputs

$Q(t)$ refers to current state **before** the clock edge arrives

$Q(t+1)$ refers to next state **after** the clock edge arrives

D Flip-Flop	
D	$Q(t+1)$
0	0 Reset
1	1 Set

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

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

Flip-Flop Characteristic Equation

The characteristic equation defines the operation of a flip-flop

For D Flip-Flop: $Q(t+1) = D$

For JK Flip-Flop: $Q(t+1) = J\,Q'(t) + K'\,Q(t)$

For T Flip-Flop: $Q(t+1) = T \oplus Q(t)$

Clearly, the D Flip-Flop is the simplest among the three

D Flip-Flop	
D	$Q(t+1)$
0	0 Reset
1	1 Set

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

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

Summary

In a sequential circuit there is internal memory

- Output is a function of current inputs and present state
- The stored memory value defines the present state
- Similarly, the next state depends on current inputs and present state

Two types of sequential circuits:

- Synchronous sequential circuits are clocked (easier to implement)
- Asynchronous sequential circuits are not clocked

Two types of Memory elements: Latches and Flip-Flops

Latches are level-sensitive, flip-flops are edge-triggered

Flip-flops are better memory elements for synchronous circuits

A flip-flop is described using a characteristic table and equation