ITI1100 Section Z
Digital Systems I

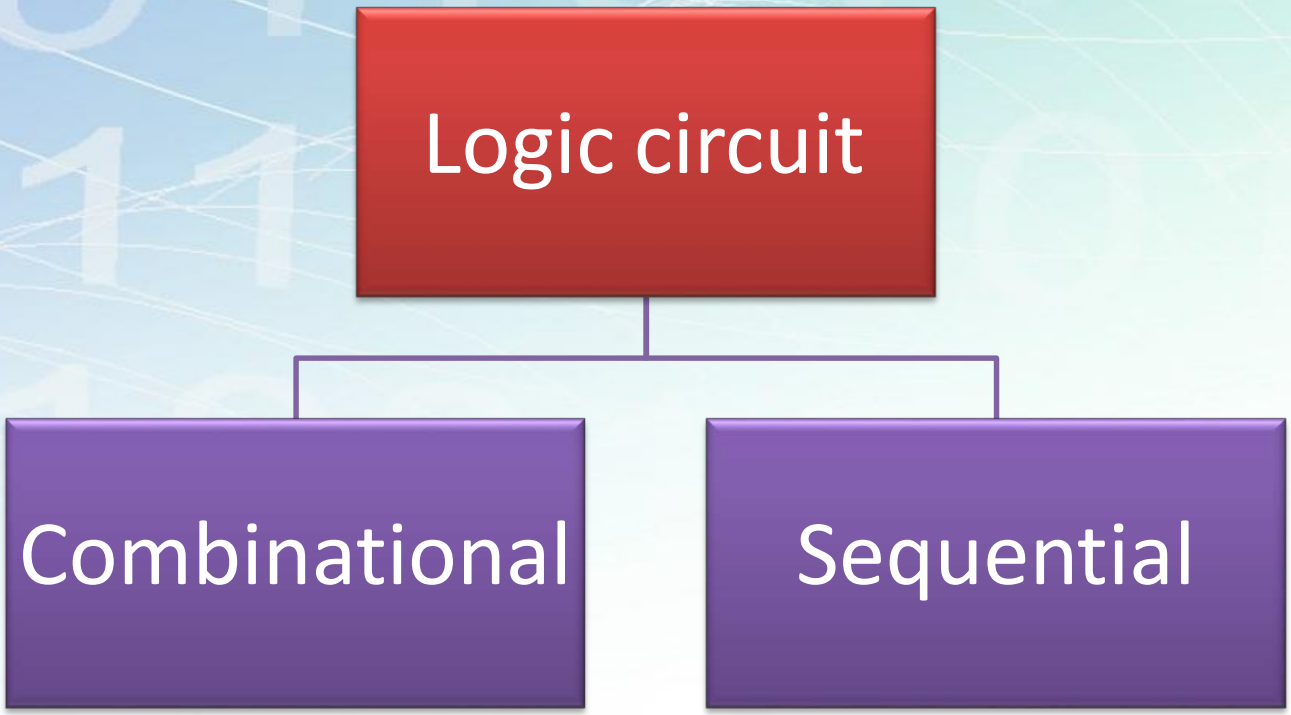
Chapter 5:

Synchronous Sequential Logic (1)

Prof. Mohammad Alja'Afreh

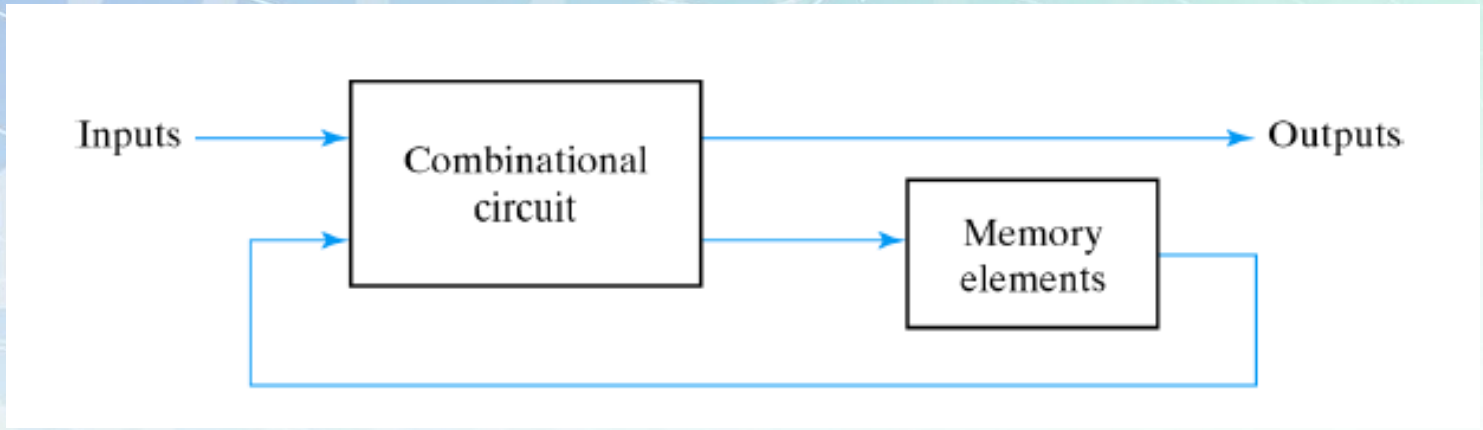
Summer 2019

Logic Circuits



√

Sequential Circuits



- **Outputs** are **functions** of **inputs** and **previous history of circuit** (memory)
- **Feedback** loops

Sequential Circuits (cont'd)

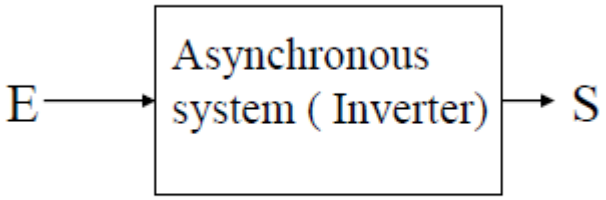
- In contrast to the combinational circuits, **two new parameters** are to be considered
  - **Memory Element:**
    - The function performed by the memory element is to **store previous results in a binary form**
  - **Time:**
    - We have to consider the **propagation delay** of the circuits

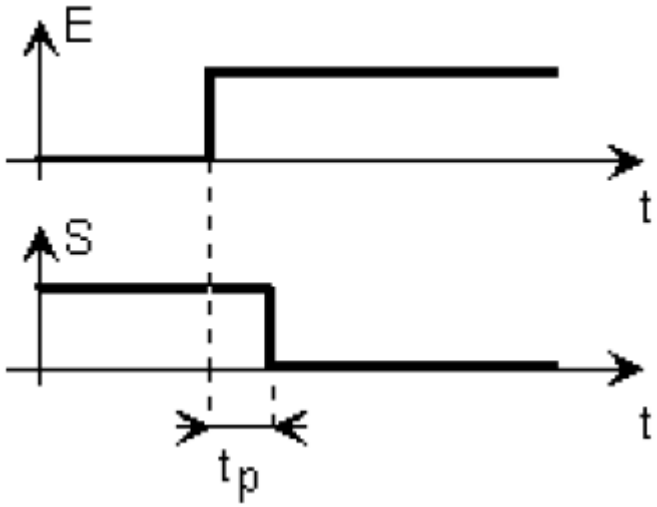
Sequential Circuits (cont'd)

- The **memory cell** that stores one "**bit**" of information is called **flip-flop**
- A **set of flip-flops** can be used to **store several bits** in a **REGISTER**
- Two different **modes**:
  - Asynchronous
  - Synchronous

Sequential Circuits (cont'd): Asynchronous

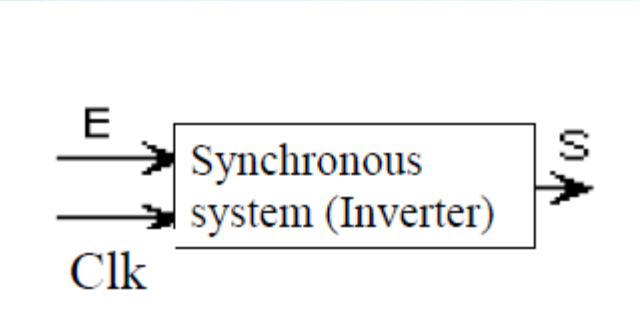
- In this mode of operation, the **output reacts "immediately"** to the **changes** of the **input variables**
  - **Propagation delay**, t_p, of the **gates** are to be **considered**

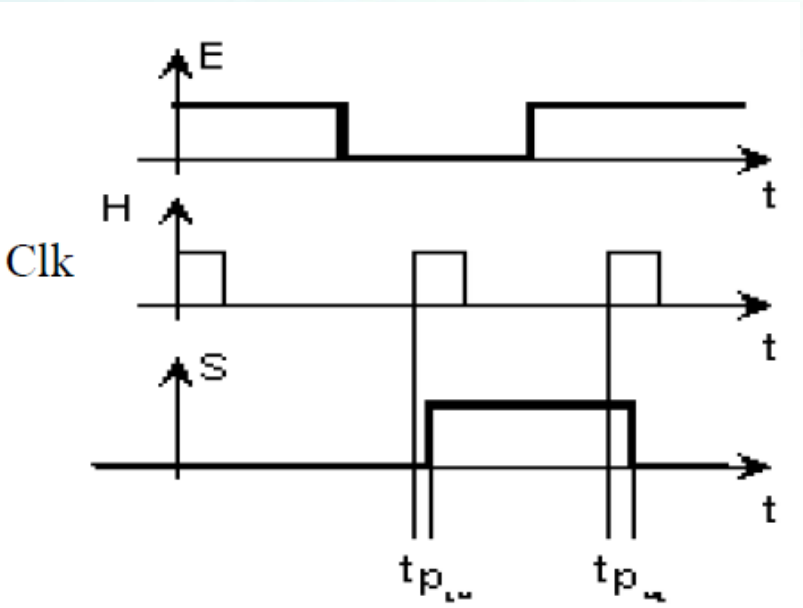




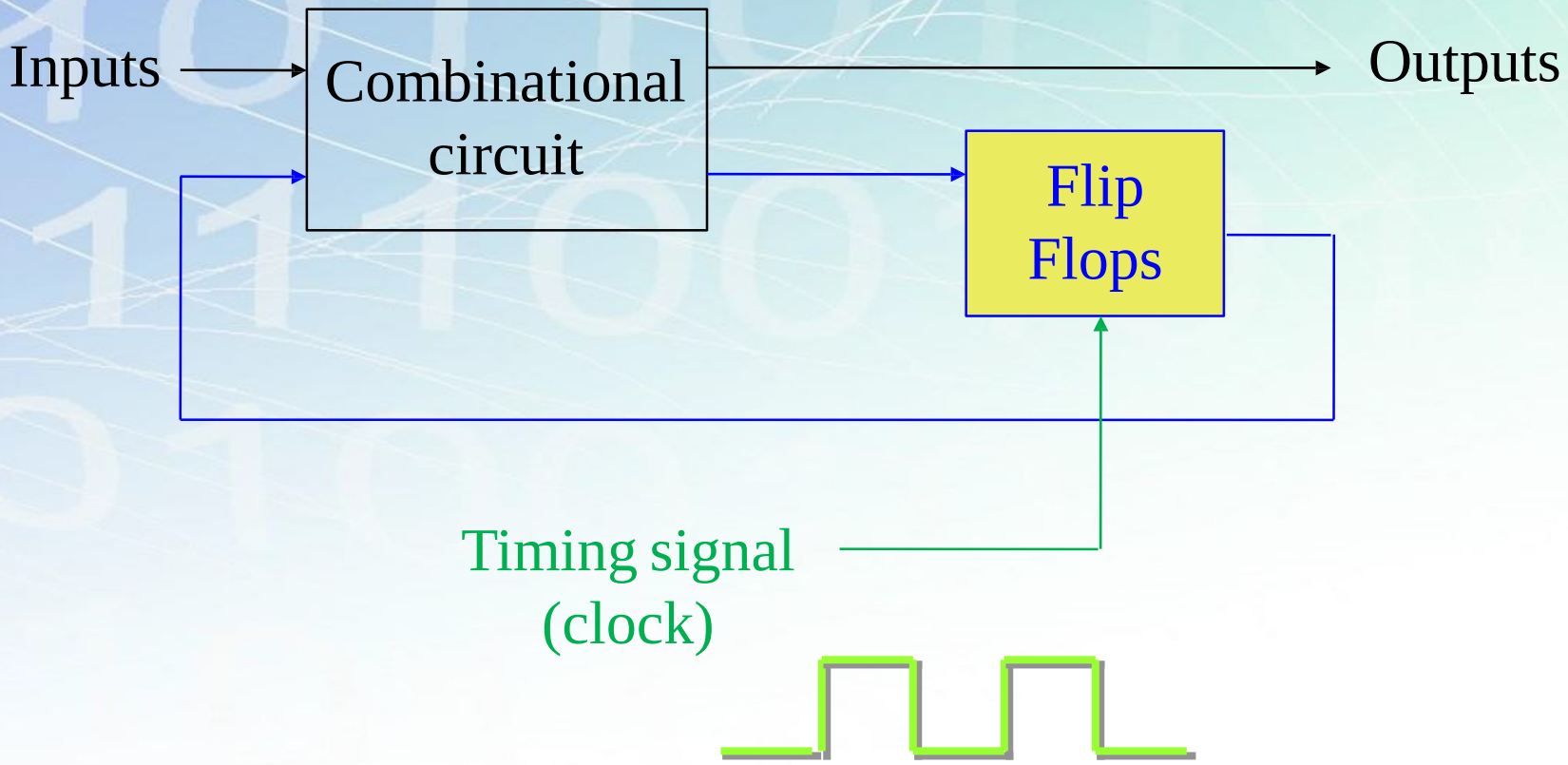
Sequential Circuits (cont'd): Synchronous

- In this mode of operation, **changes** of the **input variables** will be become **"effective"** when the **clock input** is **active**
  - Clock signals are periodic signals





Synchronous Clocked Sequential Circuits



After changed inputs, new outputs appear in the next clock cycle

Clock Signal

- Clock is used in synchronous logic circuits to **trigger the circuit (flip-flop)** allowing it to **switch its states**

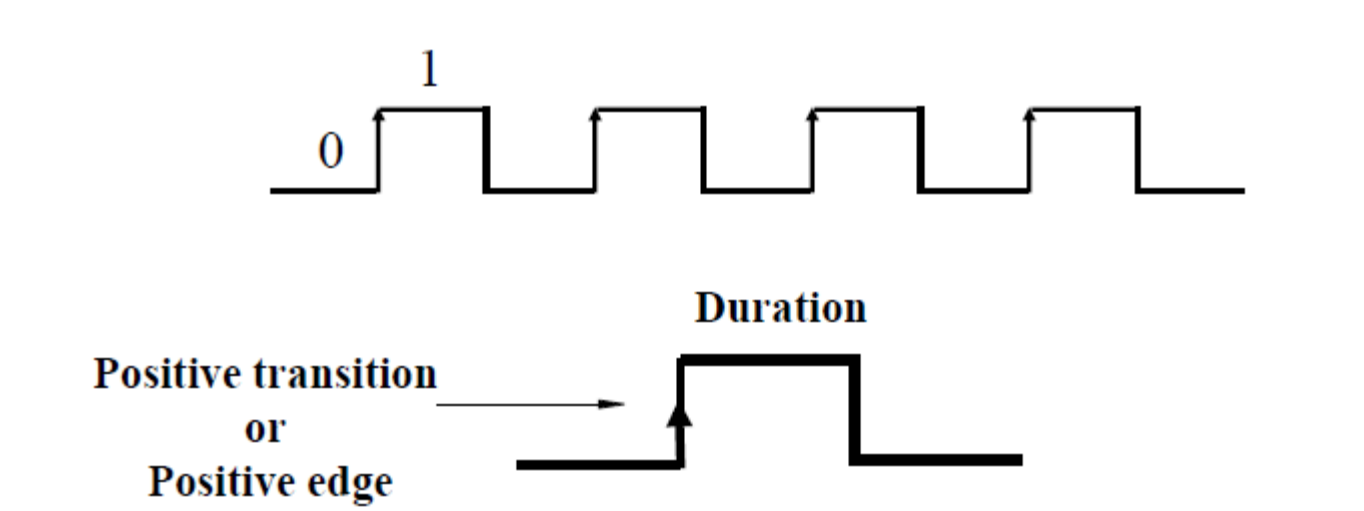
- Clock signal can **trigger a flip-flop** in **three ways**:
  - **During Positive level**
  - **During Positive transition (or positive edge)**
  - **During Negative transition (or negative edge)**

Clock Signal (cont'd)

- **Positive level:**
  - The **flip-flop** is **triggered** while the **clock pulse** is at **logic '1'**
  - The **logic '0'** is therefore **inactive state** where changes to the flip-flop's states are not allowed

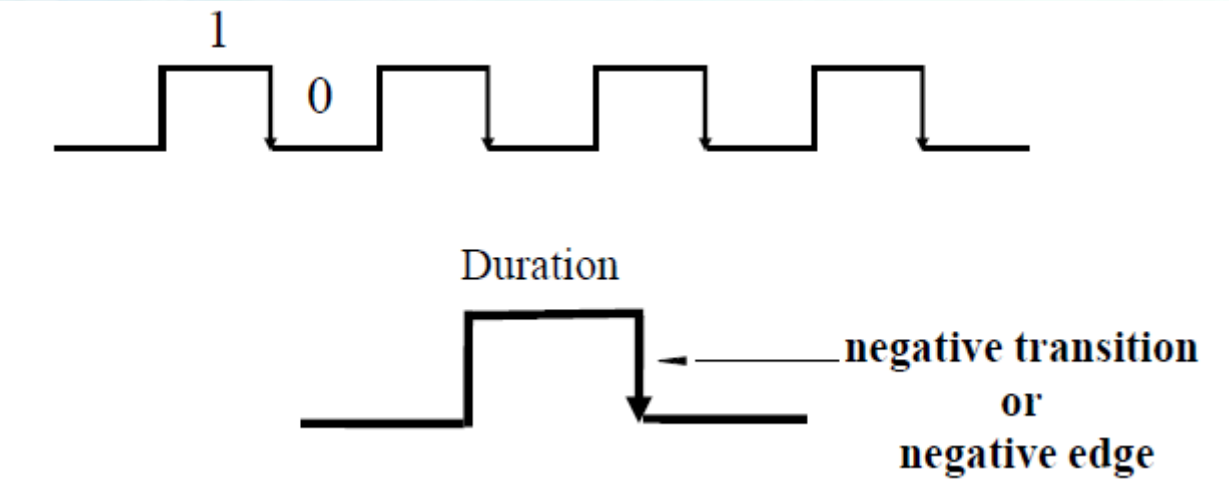
Clock Signal (cont'd)

- **During positive transition (or positive edge):**
  - The **flip-flop** is **triggered only during the signal transition from 0 to 1**; this transition is defined as **positive edge**



Clock Signal (cont'd)

- **During negative transition (or negative edge):**
  - The **flip flop** is **triggered only during the signal transition from 1 to 0**; this transition is defined as **negative edge**

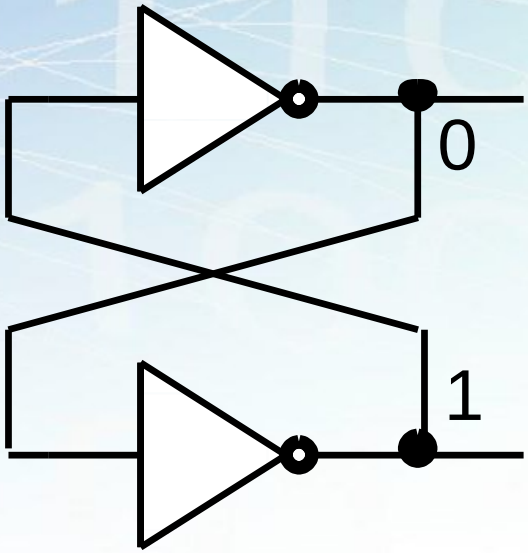


Memory Element: Latch

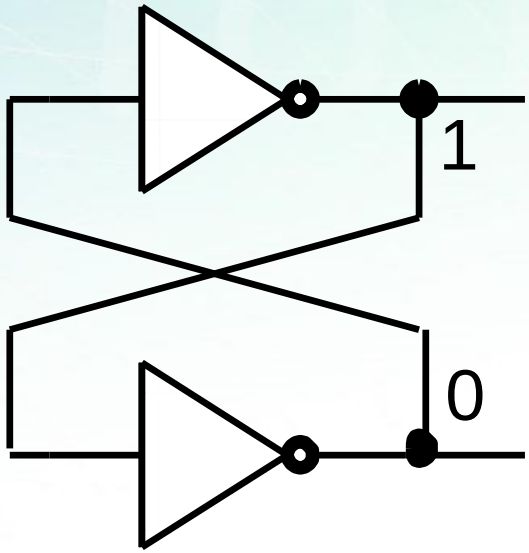
- A **flip-flop** can **maintain a binary state indefinitely until directed by input signals** to **switch states**
- The most **basic types of flip-flops** operate **without clock signal** are referred to as **Latches**
- **Latches** are **basic circuits** from which **all flip-flops** are **constructed**

Memory Element: Latch (cont'd)

- **A stable value can be stored at inverter outputs**



State 1



State 2

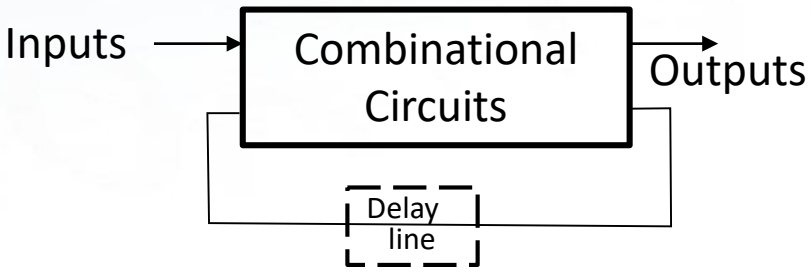
SEQUENTIAL CIRCUITS

The output of sequential circuits is logic function of the present state of external inputs, but also on the state of these inputs in past. Therefore, they are also referred to as circuits with memory.

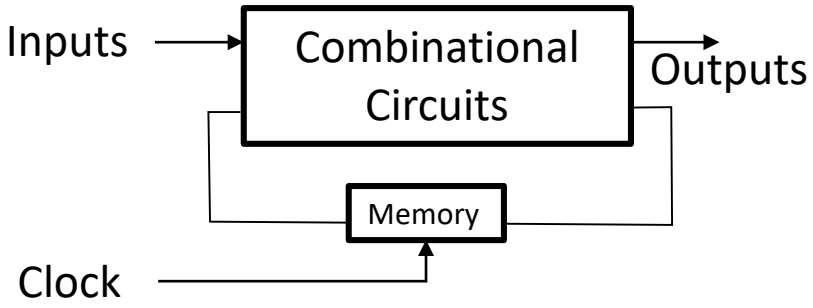
The sequential circuits are formed by additional feedback signals, looped backward from the output to the input of the circuit (referred to as signals of internal state of the circuit). By this backward loop, the dependence on previous values of inputs is implemented as dependence on the current internal state of the machine.

The sequential switching circuits are classified in:

- **Asynchronous** sequential circuits = any (asynchronously) change of an input may trigger an output change



- **Synchronous** sequential circuits = outputs may change at time events defined by a periodical control signal (pulse) called **clock**.



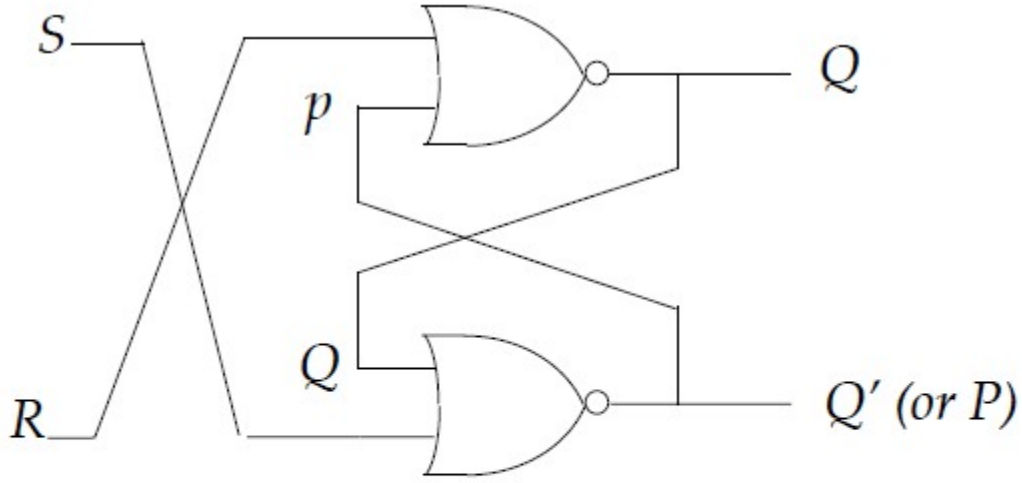
Memory Element: SR Latch

- SR latch has **two input variables** that are defined as **S** and **R**
  - **S** for **SET**
  - **R** for **RESET**
- It also has **two outputs Q** (**normal state**) and **Q'** (**complement state**)

Symbol

Memory Element: SR Latch (cont'd)

- NOR implementation:



$Q = (R+P)' = R'P'$

$P = (S+Q)' = S'Q'$

Memory Element: SR Latch (cont'd)

S	R	Present State Q	New State Q	New State Q' (or P)
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	0
1	0	0	0	0
1	0	1	1	0
1	1	0	0	0
1	1	1	0	0

Unchanged

Reset or Set to "0"

Set to "1"

Not allowed or forbidden

= Inconsistent

Memory Element: SR Latch (cont'd)

		Present State	New State	
S	R	Q	Q	Q' (or P)
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	1	0
1	1	0	X	X
1	1	1	X	X

Unchanged

Reset or Set to "0"

Set to "1"

Not allowed or forbidden

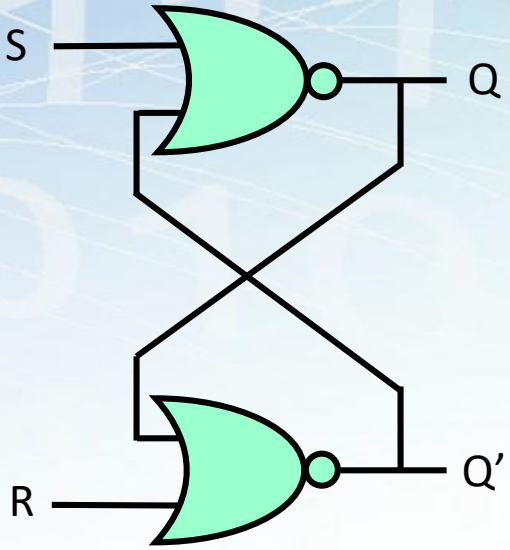
At steady state

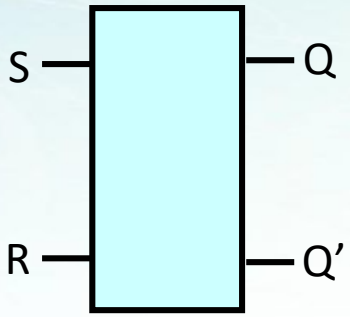
Memory Element: SR Latch (cont'd)

- Function table for SR latch with NOR gates:

S	R	Q	Q'	
1	0	1	0	Set to "1"
0	1	0	1	Set to "0"
0	0	1	0	unchanged (after SR = 10)
0	0	0	1	unchanged (after SR = 01)
1	1	?	?	Not allowed (0 0)

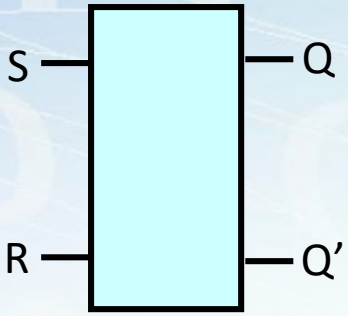
The Set-Reset Latch



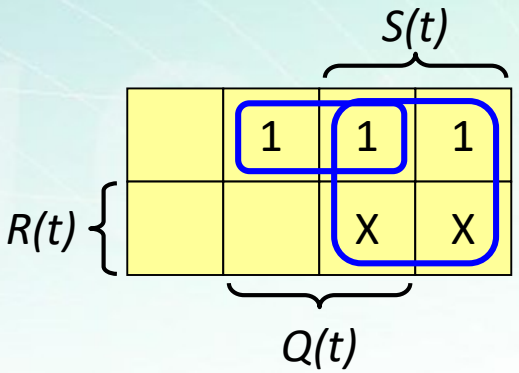


$S(t)$	$R(t)$	$Q(t)$	$Q(t+\varepsilon)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	-
1	1	1	-

The Set-Reset Latch



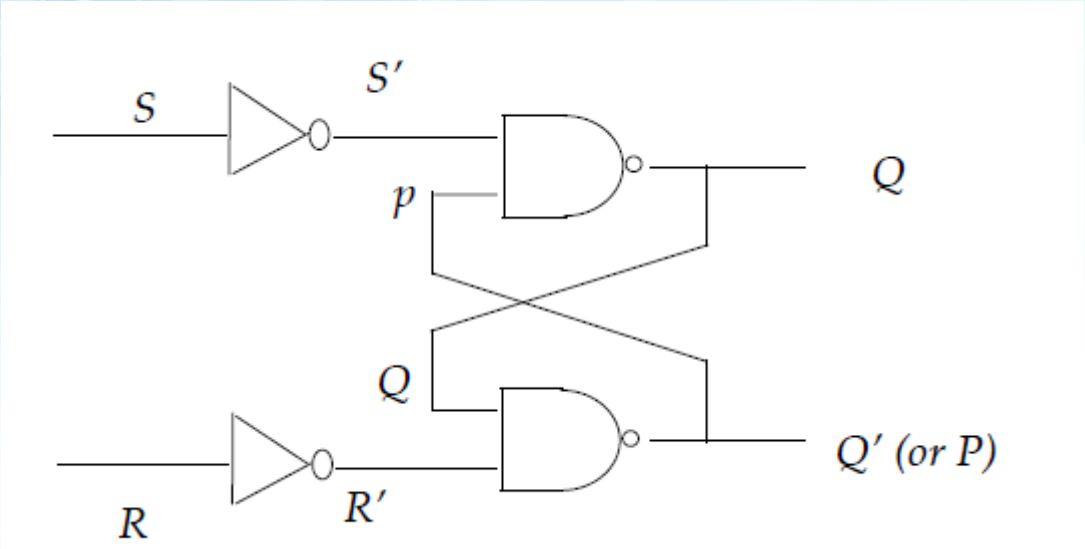
S(t)	R(t)	Q(t)	Q(t+ε)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	-
1	1	1	-



$$Q(t+\varepsilon) = S(t) + Q(t)R'(t)$$

Memory Element: SR Latch (cont'd)

- Inverter and NAND Implementation also called S'R' latch:



$Q = (S'.P)' = S + P'$

$P = (R'Q)' = R + Q'$

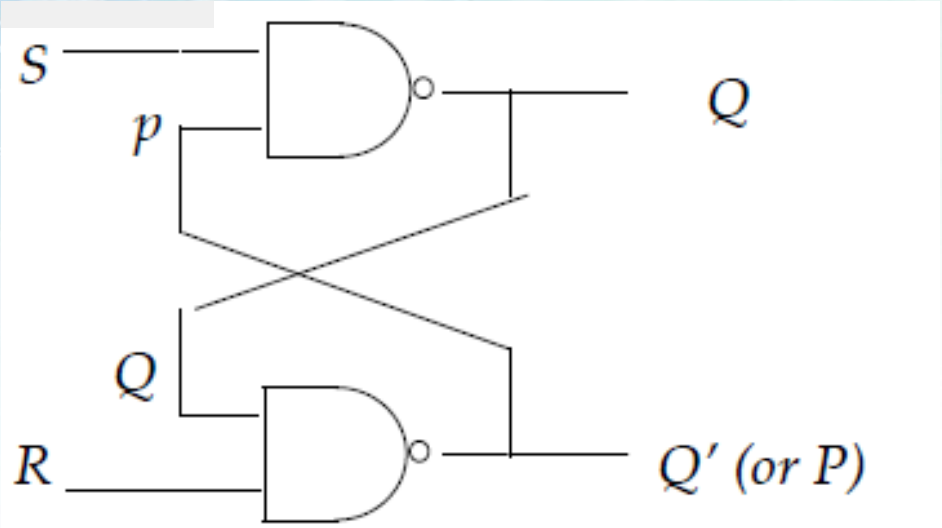
Memory Element: SR Latch (cont'd)

- Function table for SR latch with Inverter and NAND gates:

S	R	Q	Q'	
1	0	1	0	Set to "1"
0	1	0	1	Set to "0"
0	0	1	0	unchanged (after SR = 10)
0	0	0	1	unchanged (after SR = 01)
1	1	?	?	Not allowed (0 0)

Memory Element: SR Latch (cont'd)

- NAND Implementation:



Q=(S.P)'=S'+P'
P=(Q.R)'=Q'+R'

Memory Element: SR Latch (cont'd)

S	R	Present State Q	New State Q	New State Q' (or P)
0	0	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	1
1	1	0	0	1
1	1	1	1	0

Not allowed or forbidden

Set to "1"

Reset or Set to "0"

Unchanged

= Inconsistent

Memory Element: SR Latch (cont'd)

		Present State	New State	
S	R	Q	Q	Q' (or P)
0	0	0	X	X
0	0	1	X	X
0	1	0	1	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

Not allowed or forbidden

Set to "1"

Reset or Set to "0"

Unchanged

At steady state

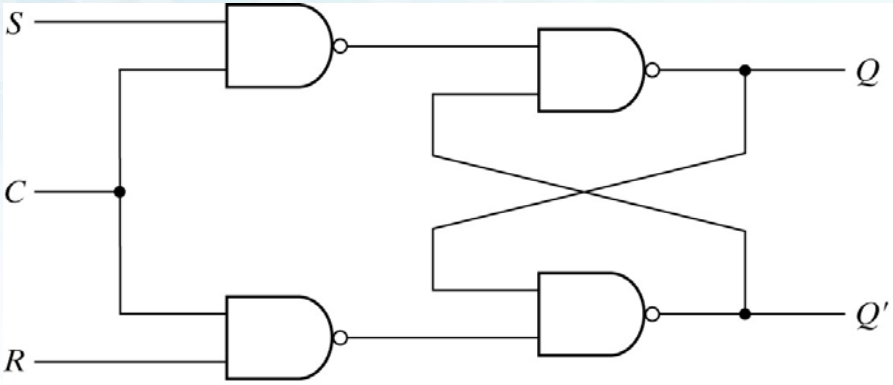
Memory Element: SR Latch (cont'd)

- Function table for SR latch with NAND gates:

S	R	Q	Q'	
1	0	0	1	set to « 0 »
1	1	0	1	unchanged (after SR = 10)
0	1	1	0	set to « 1 »
1	1	1	0	unchanged (after SR = 01)
0	0	?	?	Not allowed! (1 1)

SR Latch with Control Input

- Occasionally, desirable to avoid latch changes
- **Change** of **state** is **regulated** by a **control signal, C**
- **C = 0 disables** all **latch state changes**
- **Control signal enables data change** when **C = 1**
- **Right side** of circuit same as **ordinary SR latch**



(a) Logic diagram

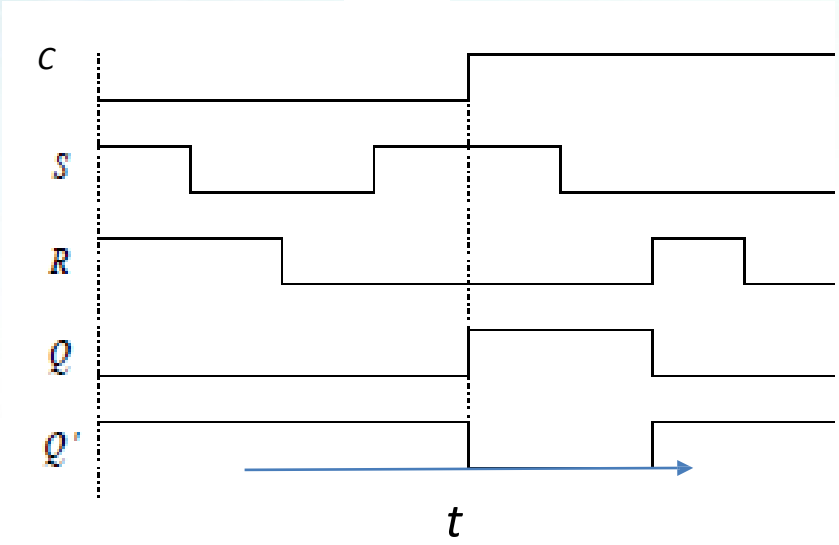
C	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	$Q = 0$; Reset state
1	1	0	$Q = 1$; set state
1	1	1	Indeterminate

(b) Function table

SR latch with inverter and NAND implementation

SR Latch with Control Input (cont'd)

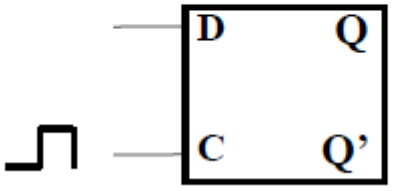
C	S	R	Next state of Q
0	X	X	No change
1	0	0	No change
1	0	1	$Q = 0$; Reset state
1	1	0	$Q = 1$; set state
1	1	1	Indeterminate



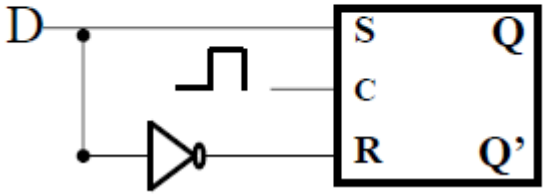
Latches with Control Input: D Latch

- **D** for **Data**. Data is transferred to the Latch
- When **C** is **high**, **D passes** from **input** to **output (Q)**
- **D latch** ensures that inputs **S** and **R** are **never equal 1 at the same time**

Symbol



Implementation with SR latch



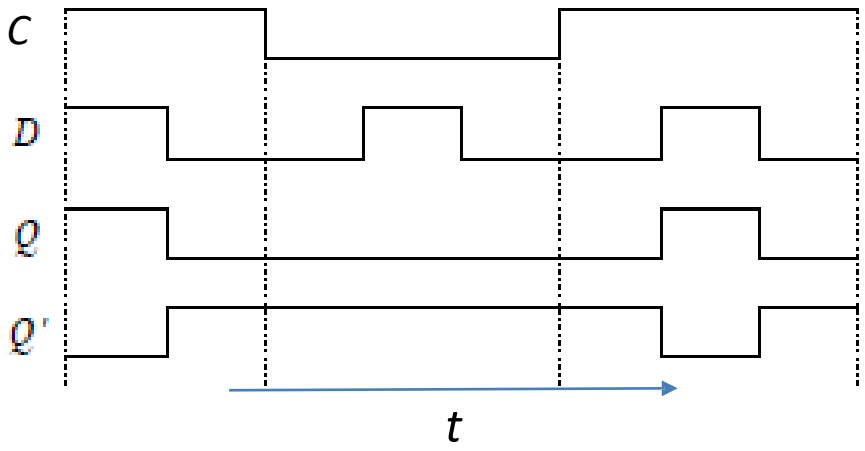
Function table

D	C	Q	Q'
0	1	0	1
1	1	1	0
X	0	Q_n	Q_n'

Note: Q_n indicates the previous state (the previously stored value)

Latches with Control Input: D Latch (cont'd)

D	C	Q	Q'
0	1	0	1
1	1	1	0
X	0	Q_n	Q_n'



Synchronous Sequential Circuits

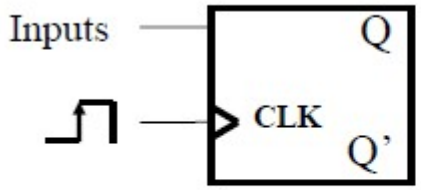
- If we **add to the memory element** a **synchronization signal** (i.e., clock impulse or control input), we would obtain **a synchronous SR latch** (i.e., a **flip-flop**)
- We will examine the following **synchronous flip-flops**:
  - SR
  - D
  - T
  - JK

Edge Sensitive Flip-Flop

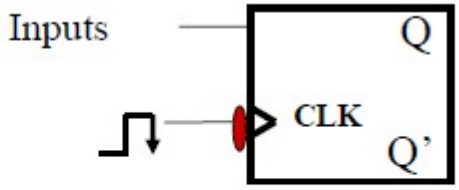
- **Latches respond** to **trigger levels** on **control inputs**
  - Example: If C = 1, input is reflected at output
- **Flip-flips store data** on a **positive or negative edge** of the **control inputs**
  - Example: control input transitions from 0 to 1, data input available at output
- **Data remains stable** in the flip-flop until **next positive/ negative edge**
- **Different types of flip-flops** are **used** for **specific functions**

Edge Sensitive Flip-Flop (cont'd)

- **Flip-flip** is a **latch** with a **clock input**
- **The sequential circuit output changes** when **its CLOCK input detects an edge**. **Edge-sensitive instead of level-sensitive**
- Symbol is a triangle on the CLK (clock) input of a flip-flop
- Flip flop can be triggered on positive or negative edge
- Bubble before Clock (CLK) input indicates negative edge trigger

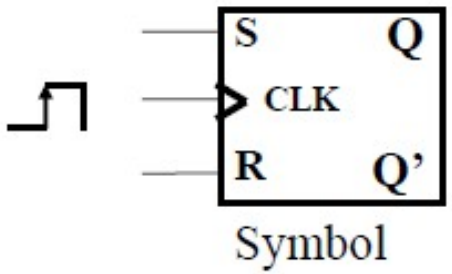


Symbol with positive edge

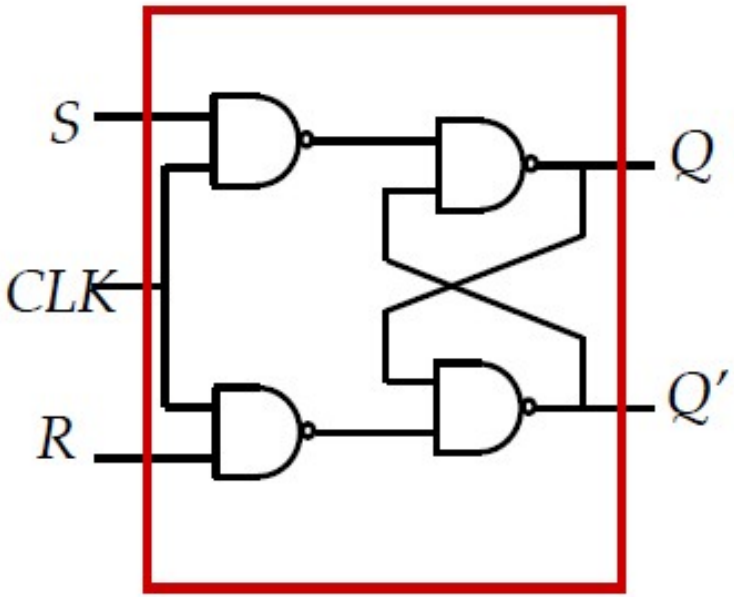


Symbol with negative edge

Edge Sensitive SR Flip-Flop



Symbol



S	R	Q_{n+1}	Function
0	0	Q_n	**No change**
0	1	0	**RESET**
1	0	1	**SET**
1	1	?	**Forbidden**

- Q_n = state *before* positive edge
- Q_{n+1} = state *after* positive edge

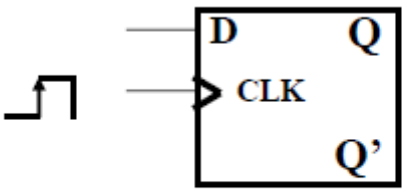
Characteristic Equation:

$$Q^{t+\Delta t} = S^t + Q^t \cdot \overline{R^t}$$

$$\text{condition}: S^t \cdot R^t = 0$$

D Flip-Flop

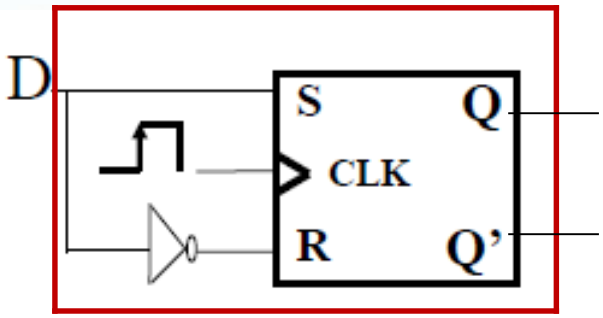
- **Output changes** only **on the clock transition**
- **Input changes at other times** have **no effect on output**



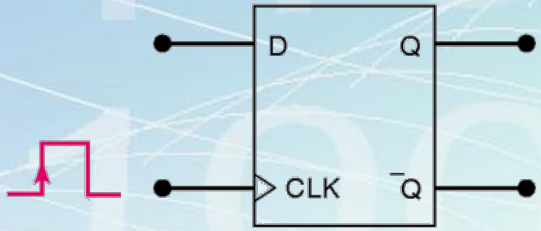
Symbol

D	CLK	Q	Q'
0	↑	0	1
1	↑	1	0
X	0	Q_0	Q_0'

$Q^{n+1} = D^n$

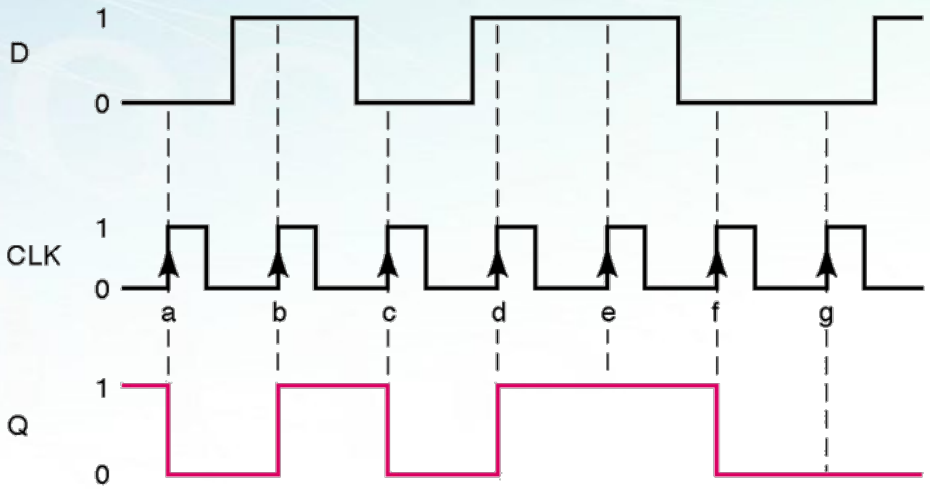


D Flip-Flop (cont'd)



(a)

D	CLK	Q
0	↑	0
1	↑	1



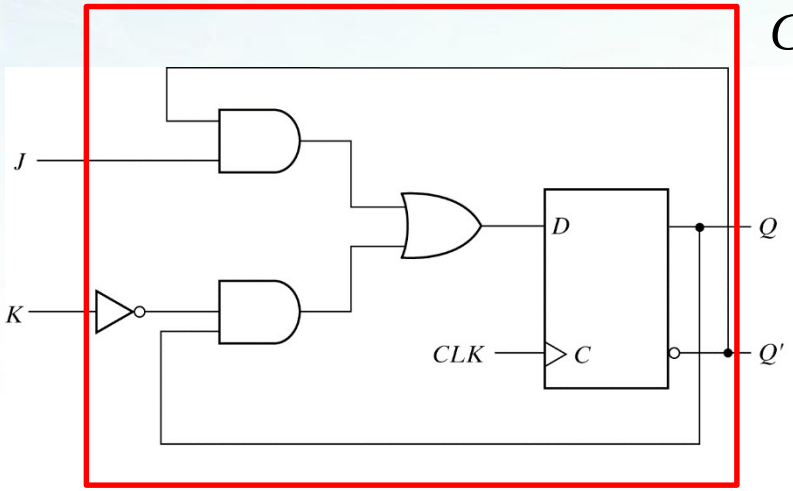
(b)

JK Flip-Flop

- Same as SR **except for K=J=1** the JK flip-flop will **output** the **opposite state** of **Q_n**
  - If $Q_n = 1$ then $Q_{n+1} = 0$
  - If $Q_n = 0$ then $Q_{n+1} = 1$

J	K	Q_{n+1}	Function
0	0	Q_n	**No change**
0	1	0	**RESET**
1	0	1	**SET**
1	1	Q'_n	**complement (also Toggle)**

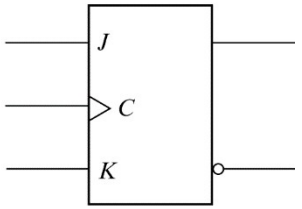
- Q_n = state *before* positive edge
- Q_{n+1} = state *after* positive edge



(a) Circuit diagram

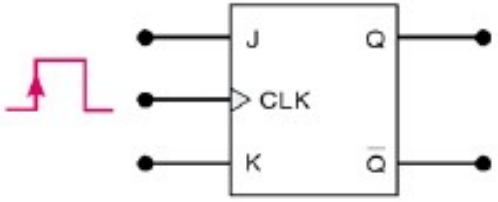
CHARACTERISTIC EQUATION:

$$Q^{n+1} = \overline{Q^n} \cdot J^n + Q^n \cdot \overline{K^n}$$

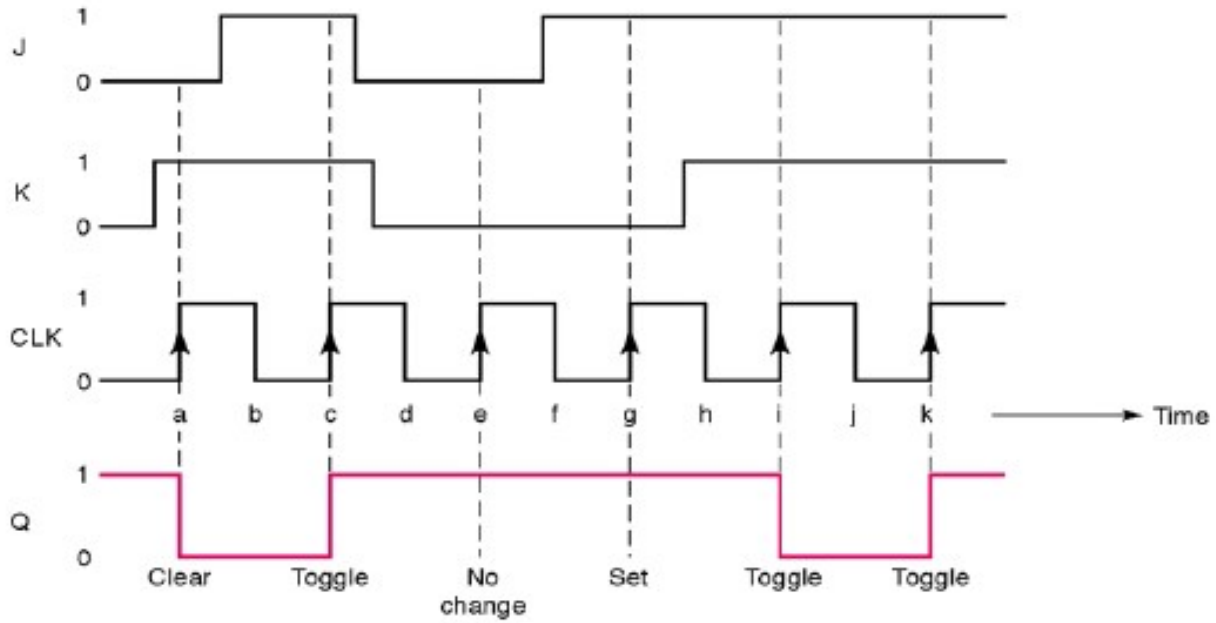


(b) Graphic symbol

JK Flip-Flop (cont'd)

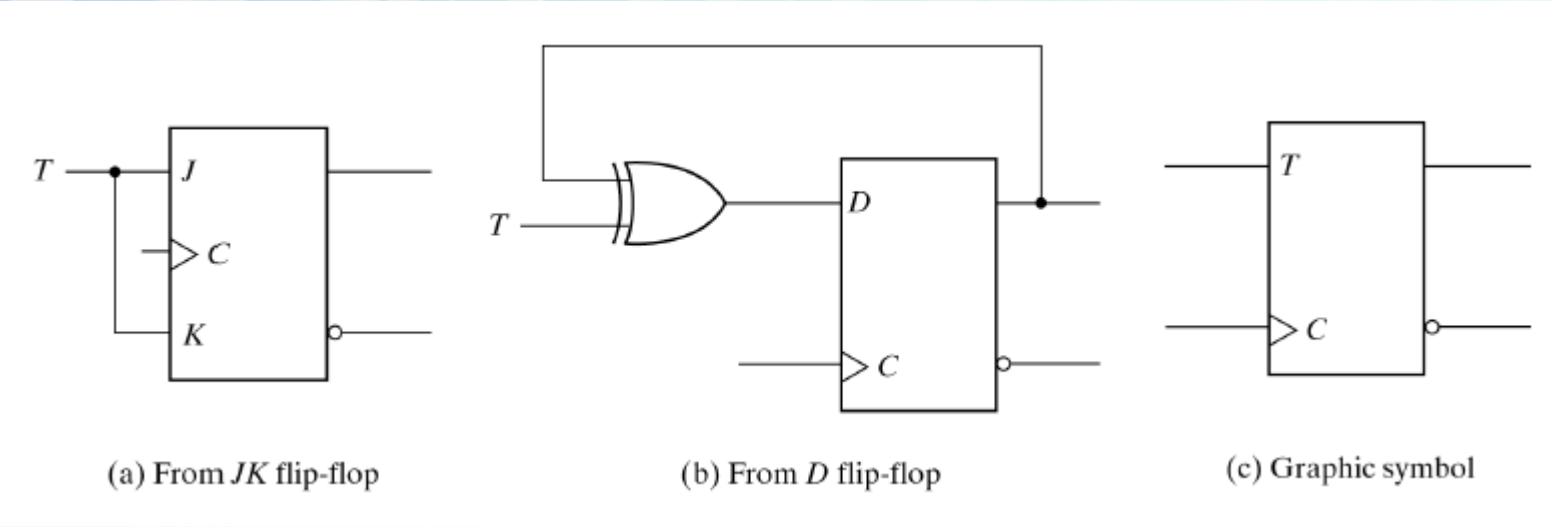


J	K	CLK	Q
0	0	↑	Q_0 (no change)
1	0	↑	1
0	1	↑	0
1	1	↑	$\overline{Q}_0$ (toggles)



T Flip-Flop

- **T** stands for **toggle**
- Can be **created** from **JK** or **D flip-flop**



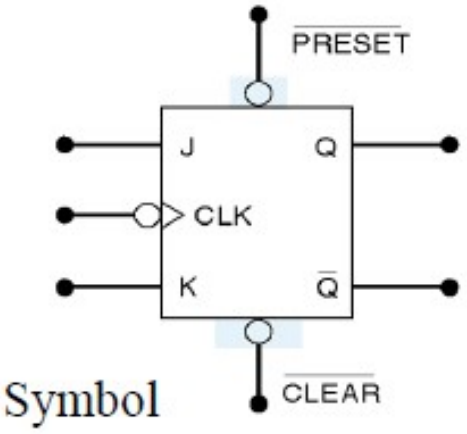
(a) From *JK* flip-flop (b) From *D* flip-flop (c) Graphic symbol

T	Q	Q_{next}	Q_{next}'
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

$$Q^{n+1} = \overline{Q^n} \cdot T + Q^n \cdot \overline{T}$$

Asynchronous Inputs

- So far, we have only considered **synchronous inputs** (e.g. D, S, R, J, K, T)
  - Their **effects** on the **output** are **synchronized** with the **clock** or **control signal**
- Flip-flops have **asynchronous inputs**. They **operate independently of the synchronous inputs and clock**
  - Used to **set** the **Flip-Flop to 1 or 0 state at any time**



Symbol

PRESET	CLEAR	FF response
1	1	Clocked operation*
0	1	Q = 1 (regardless of CLK)
1	0	Q = 0 (regardless of CLK)
0	0	Not used

*Q will respond to J, K, and CLK