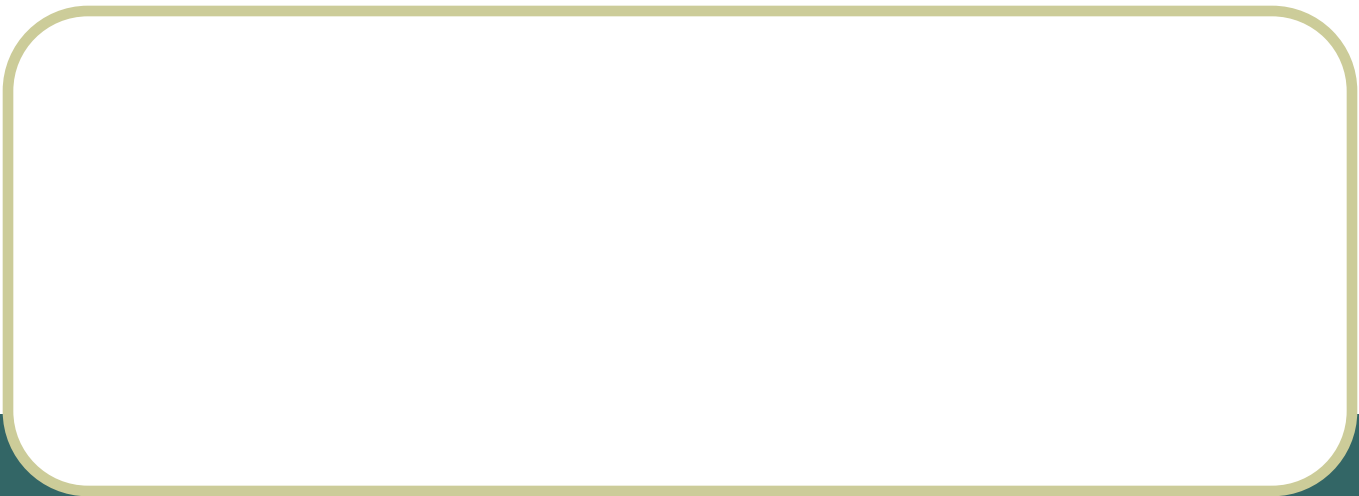


Sequential Circuits



Objectives

- Sequential Circuits
- Storage Elements (Memory)
 - Latches
 - Flip-Flops

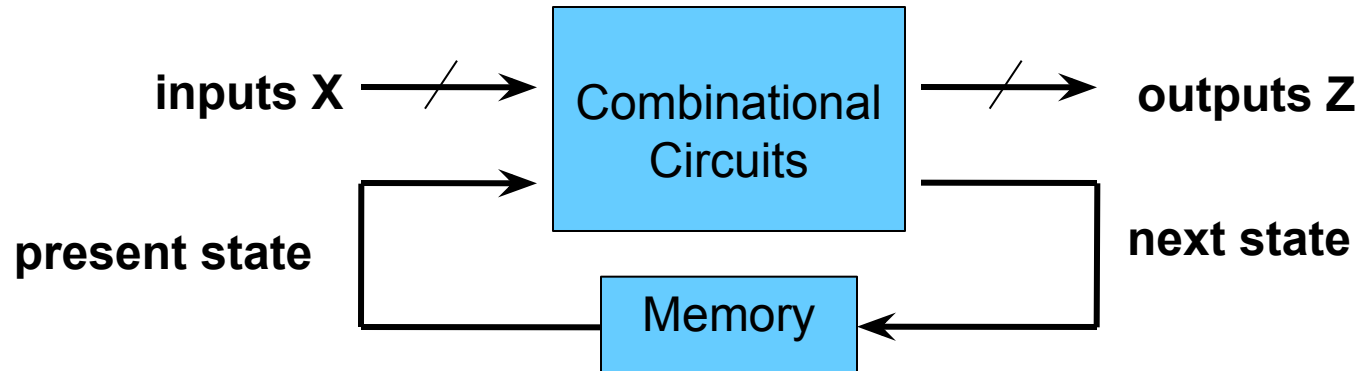
Combinational vs Sequential



A combinational circuit:

- At any time, outputs depends only on inputs
 - Changing inputs changes outputs
- No regard for previous inputs
 - No memory (history)
- Time is ignored !

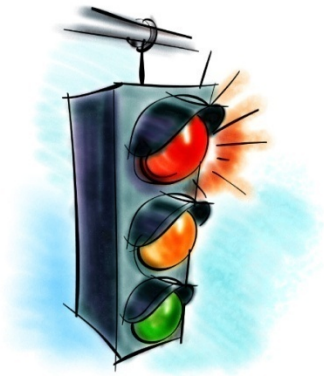
Combinational vs Sequential



A **sequential** circuit:

- A combinational circuit with **feedback** through **memory**
 - The stored information at any time defines a **state**
- Outputs depends on inputs and previous inputs
 - Previous inputs are stored as binary information into memory
- Next state depends on inputs and present state

Examples of sequential systems



Traffic light



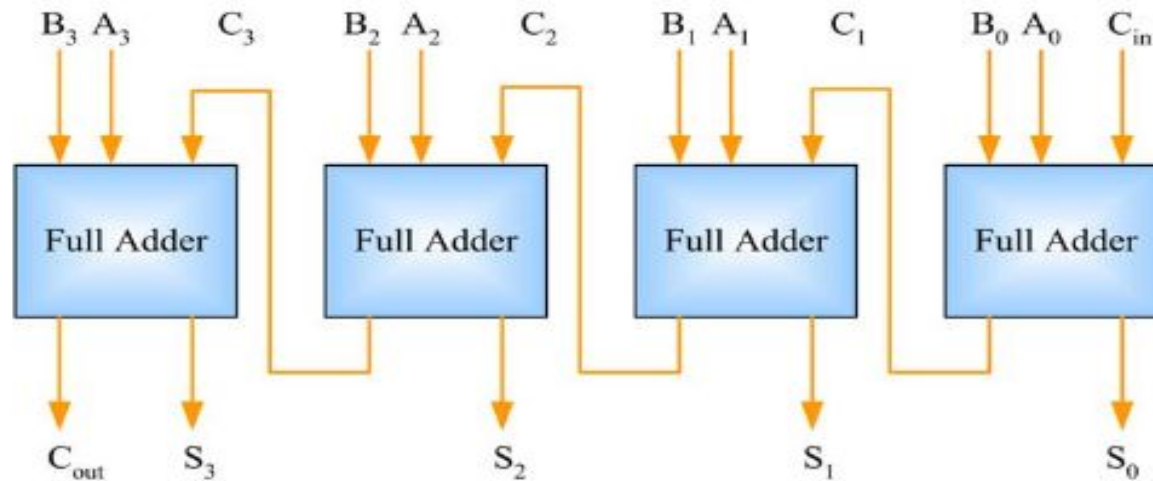
ATM



Vending machine

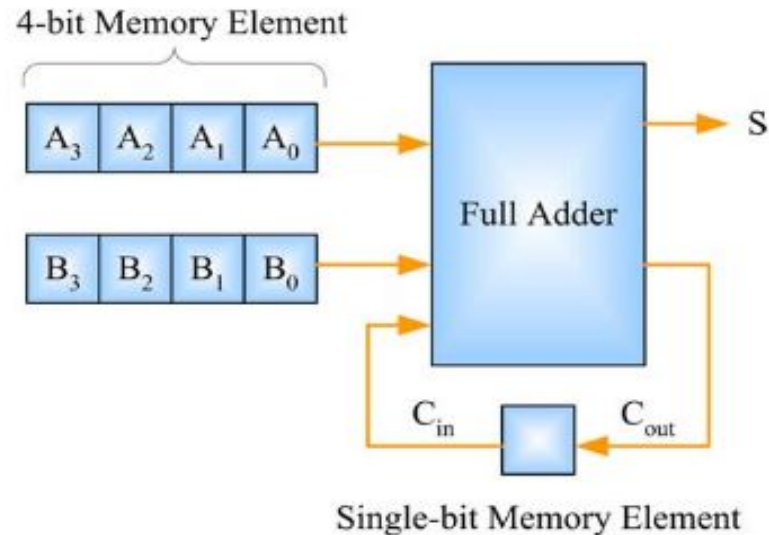
What is common between these systems?

Combinational Adder



- 4-bit adder (ripple-carry)
 - Notice how carry-out propagates
 - One adder is active at a time
- 4 full adders are needed

Sequential Adder

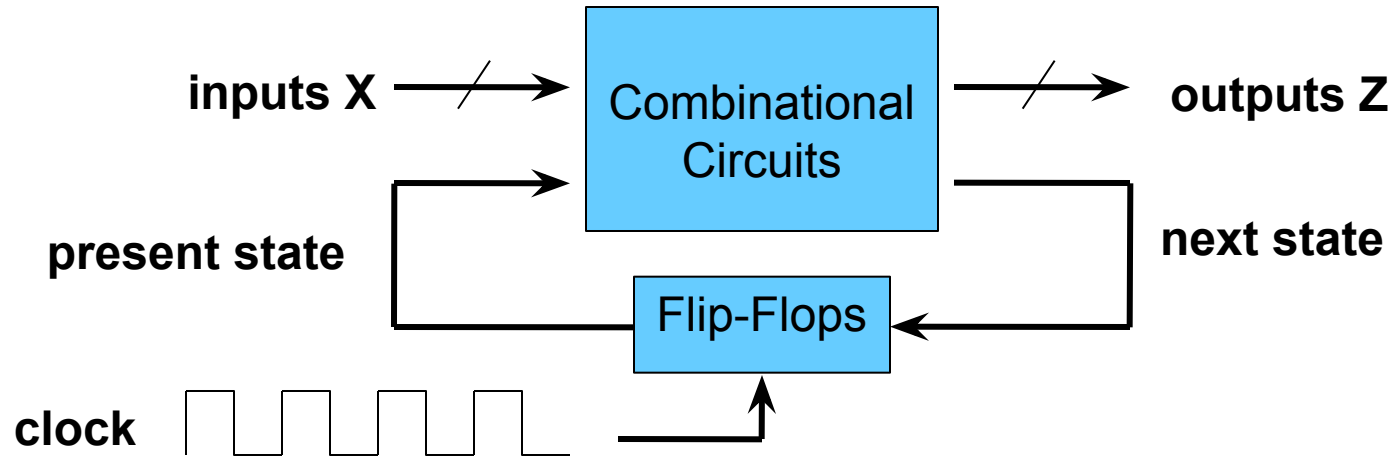


- 1-bit memory and 2 4-bit memory
- Only one full-adder!
- 4 clocks to get the output
- The 1-bit memory defines the circuit state (0 or 1)

Types of Sequential Circuits

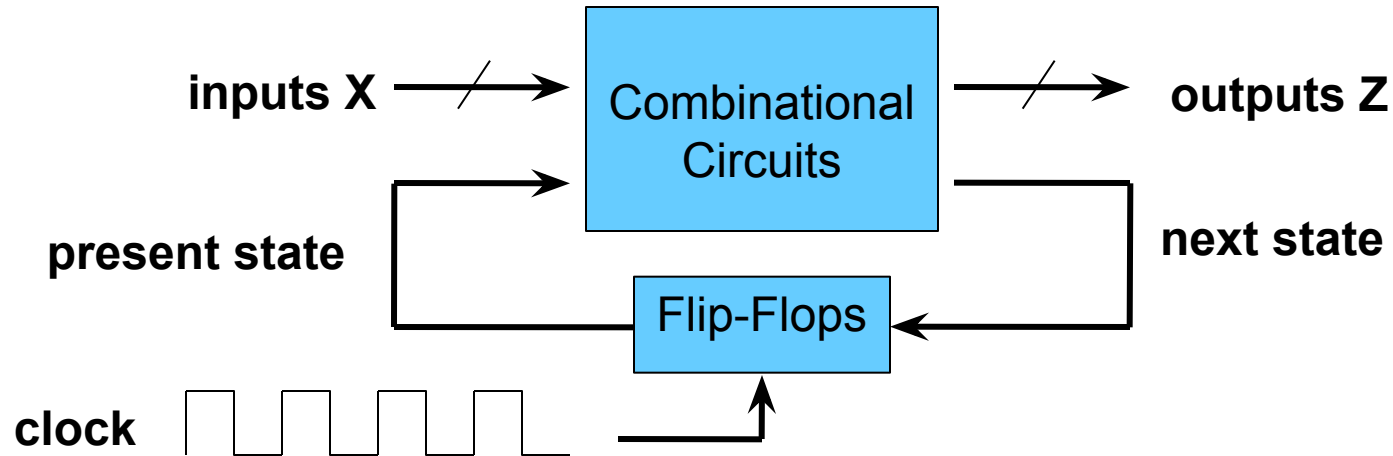
- Two types of sequential circuits:
 - **Synchronous:** The behavior of the circuit depends on the input signal at discrete instances of time (*also called clocked*)
 - **Asynchronous:** The behavior of the circuit depends on the input signals at any instance of time and the order of the inputs change
 - A combinational circuit with feedback

Synchronous Sequential Circuits



- Synchronous circuits employs a synchronizing signal called **clock** (a periodic train of pulses; 0s and 1s)
- A clock determines **when** computational activities occur
- Other signals determines **what** changes will occur

Synchronous Sequential Circuits



- The storage elements (memory) used in clocked sequential circuits are called **flip-flops**
 - Each flip-flop can store one bit of information 0,1
 - A circuit may use many flip-flops; together they define the circuit state
- Flip-Flops (memory/state) update **only** with the clock

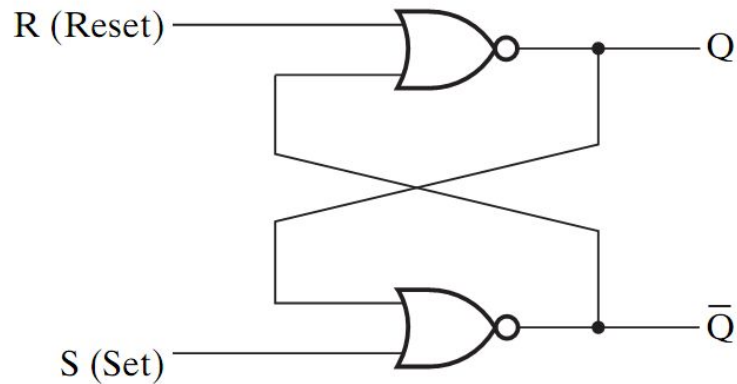
Storage Elements (Memory)

- A storage element can maintain a binary state (0,1) indefinitely, until directed by an input signal to switch state
- Main difference between storage elements:
 - Number of inputs they have
 - How the inputs affect the binary state
- Two main types:
 - **Latches** (level-sensitive)
 - **Flip-Flops** (edge-sensitive)
- Latches are useful in asynchronous sequential circuits
- Flip-Flops are built with latches

Latches

- A **latch** is binary storage element
- Can store a 0 or 1
- The most basic memory
- Easy to build
 - Built with gates (NORs, NANDs, NOT)

SR Latch



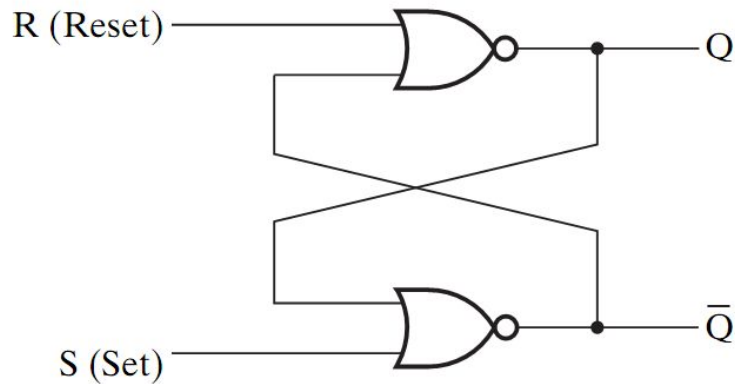
(a) Logic diagram

S	R	Q	\bar{Q}
1	0		
0	0		
0	1		
0	0		
1	1		

(b) Function table

What does this circuit do?

SR Latch



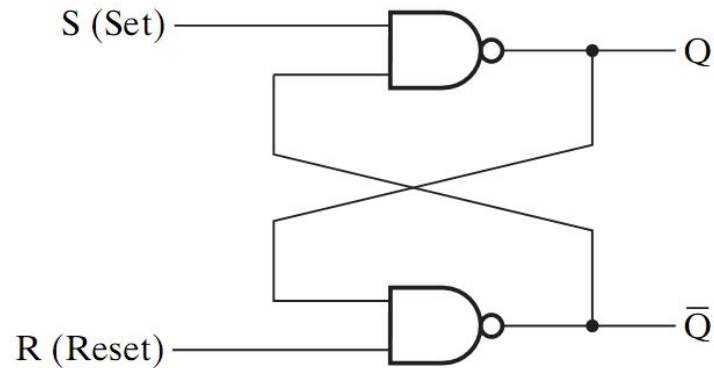
(a) Logic diagram

S	R	Q	\bar{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

- Two states: Set ($Q = 1$) and Reset ($Q = 0$)
- When $S=R=0$, Q remains the same, $S=R=1$ is not allowed!
- Normally, $S=R=0$ unless the state need to be changed (memory?)
- State of the circuit depends not only on the current inputs, but also on the recent history of the inputs

S' R' Latch



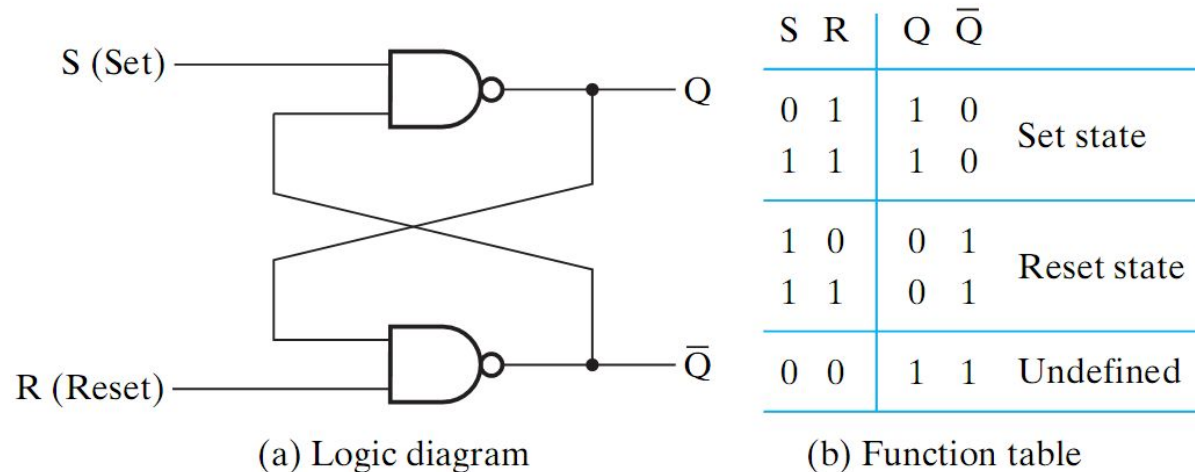
(a) Logic diagram

S	R	Q	\bar{Q}
0	1		
1	1		
1	0		
1	1		
0	0		

(b) Function table

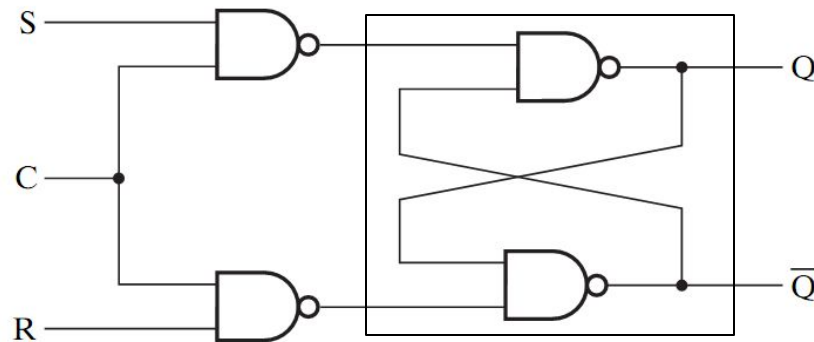
How about this circuit?

S' R' Latch



- Similar to SR latch (complemented)
- Two states: Set ($Q = 0$) and Reset ($Q = 1$)
- When $S=R=1$, Q remains the same
- $S=R=0$ is not allowed!

SR Latch with Clock



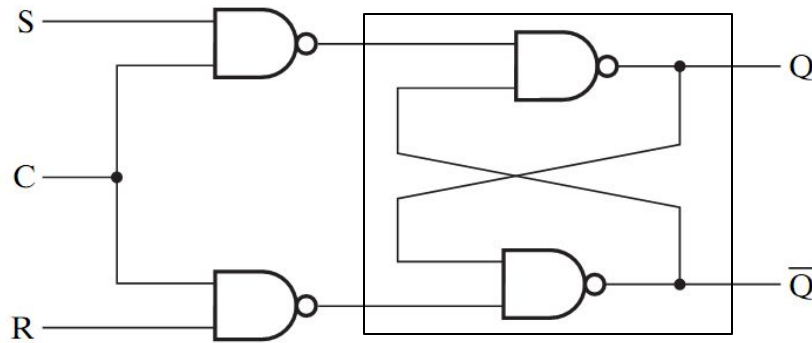
(a) Logic diagram

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 SR Latch can be modified to control **when** it changes
- An additional input signal Clock (C)
- When $C=0$, the S and R inputs have no effect on the latch
- When $C=1$, the inputs affect the state of the latch and possibly the output

SR Latch with Clock (cont.)



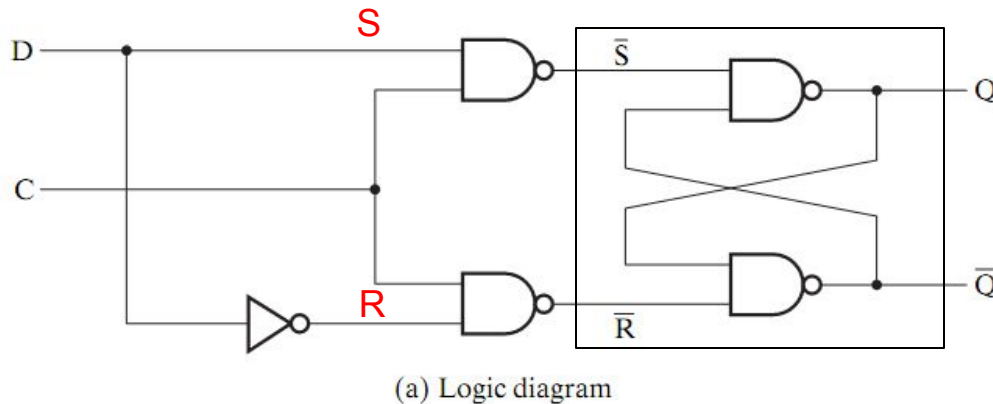
(a) Logic diagram

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

How can we eliminate the undefined state?

D Latch

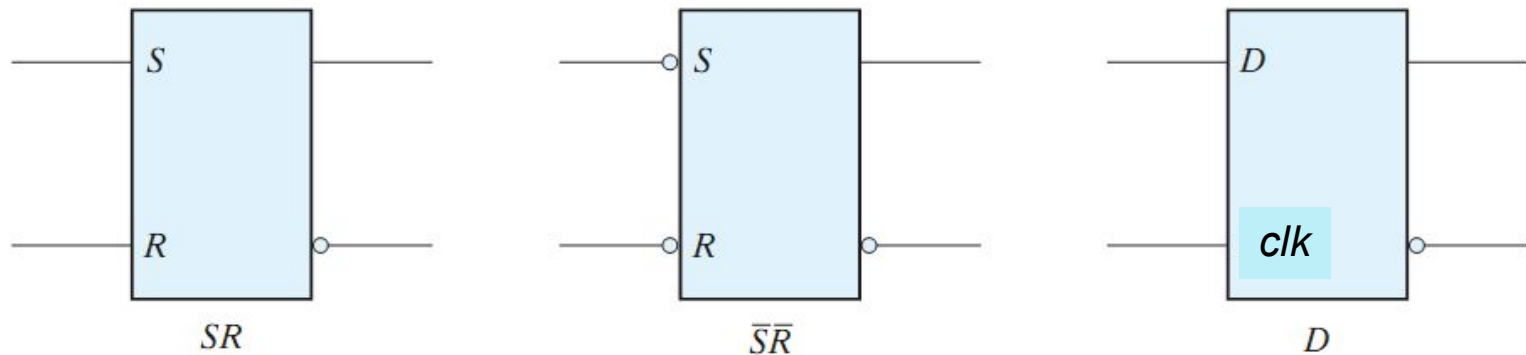


C	D	Next state of Q
0	X	No change
1	0	Q = 0; Reset state
1	1	Q = 1; Set state

(b) Function table

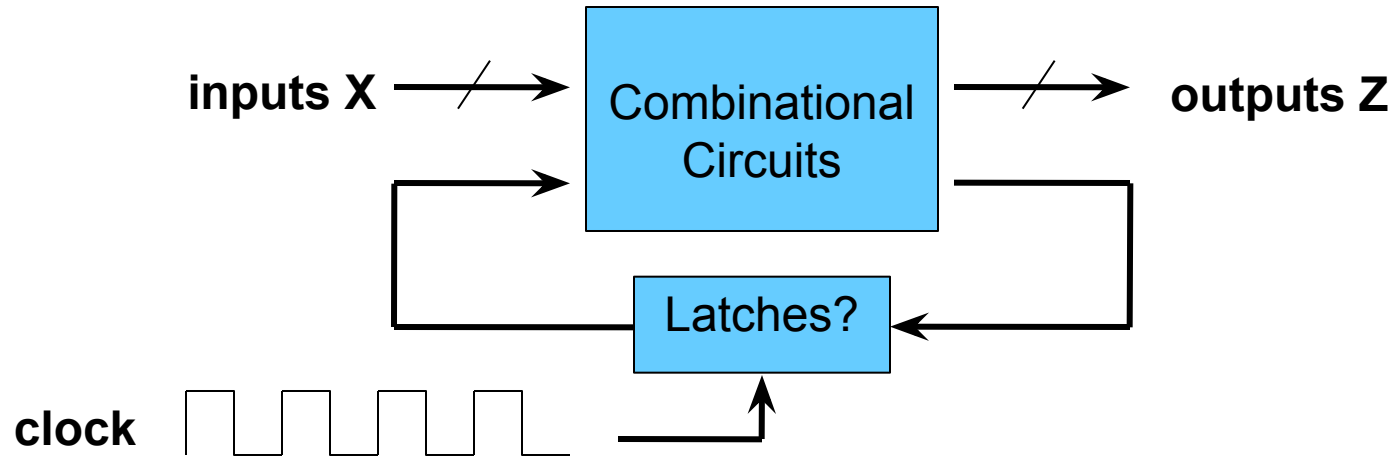
- Ensure S and R are never equal to 1 at the same time
- Add inverter
- Only one input (D)
 - D connects to S
 - D' connects to R
- D stands for data
- Output follows the input when C = 1
 - Transparent
- When C = 0, Q remains the same

Graphic Symbols for Latches



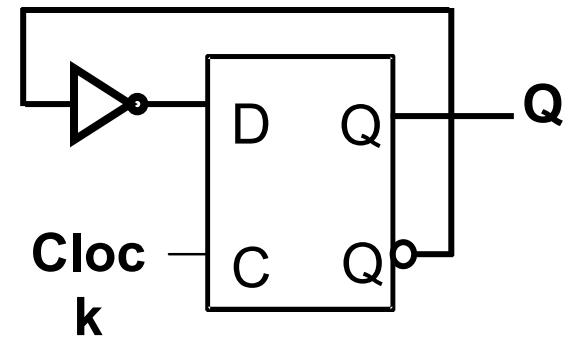
- A latch is designated by a rectangular block with inputs on the left and outputs on the right
- One output designates the normal output, the other (with the bubble) designates the complement
- For $S'R'$ (SR built with NANDs), bubbles added to the input

Problem with Latches



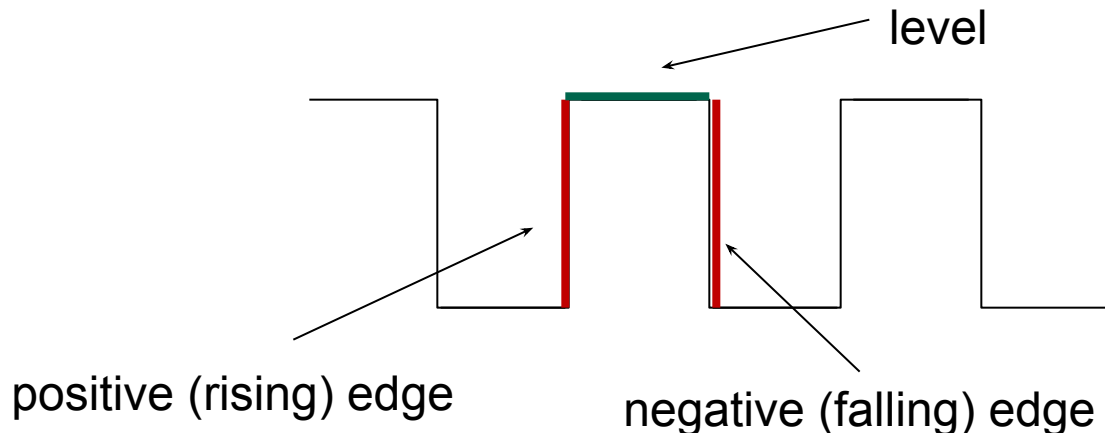
- What happens if Clock=1? What will be the value of Q when Clock goes to 0?
- **Problem:** A latch is **transparent**; state keep changing as long as the clock remains active
- Due to this uncertainty, latches can not be reliably used as storage elements.

Example

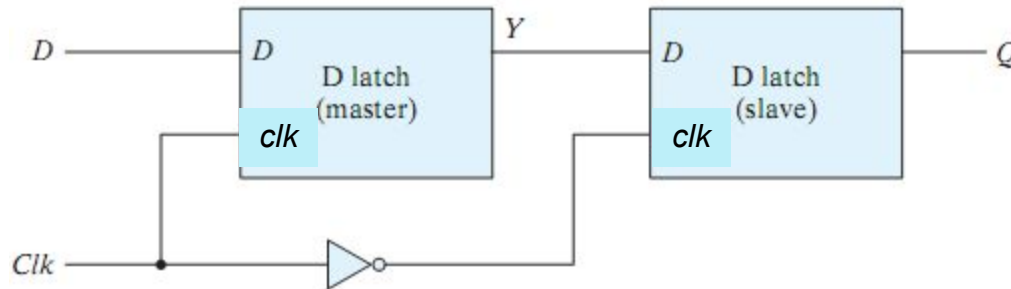


Flip Flops

- A **flip-flop** is a one bit memory similar to latches
- Solves the issue of latch transparency
- Latches are **level** sensitive memory element
 - Active when the clock = 1 (whole duration)
- Flip-Flops are **edge-triggered** or **edge-sensitive** memory element
 - Active only at transitions; i.e. either from 0 \rightarrow 1 or 1 \rightarrow 0

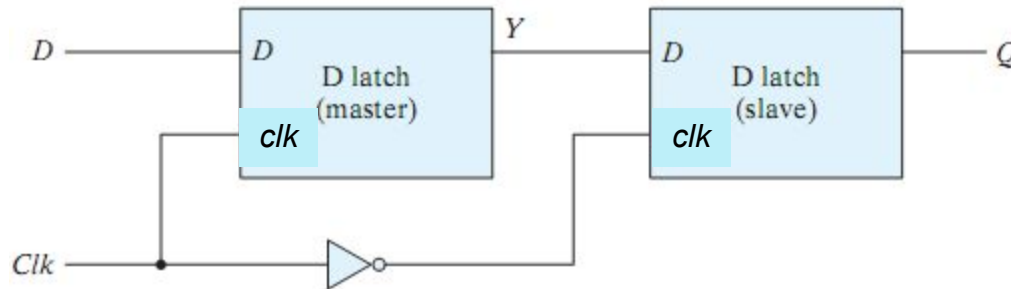


Flip Flops



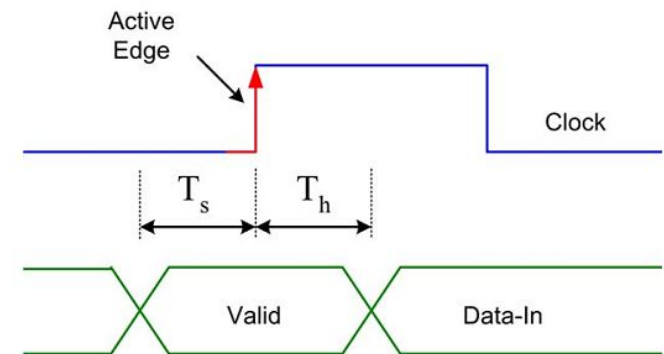
- A flip flop can be built using two latches in a **master-slave** configuration
- A master latch receives external inputs
- A slave latch receives inputs from the master latch
- Depending on the clock signal, only one latch is active at any given time
 - If $\text{clk}=1$, the master latch is enabled and the inputs are latched
 - if $\text{clk}=0$, the master is disabled and the slave is activated to generate the outputs

Flip Flops



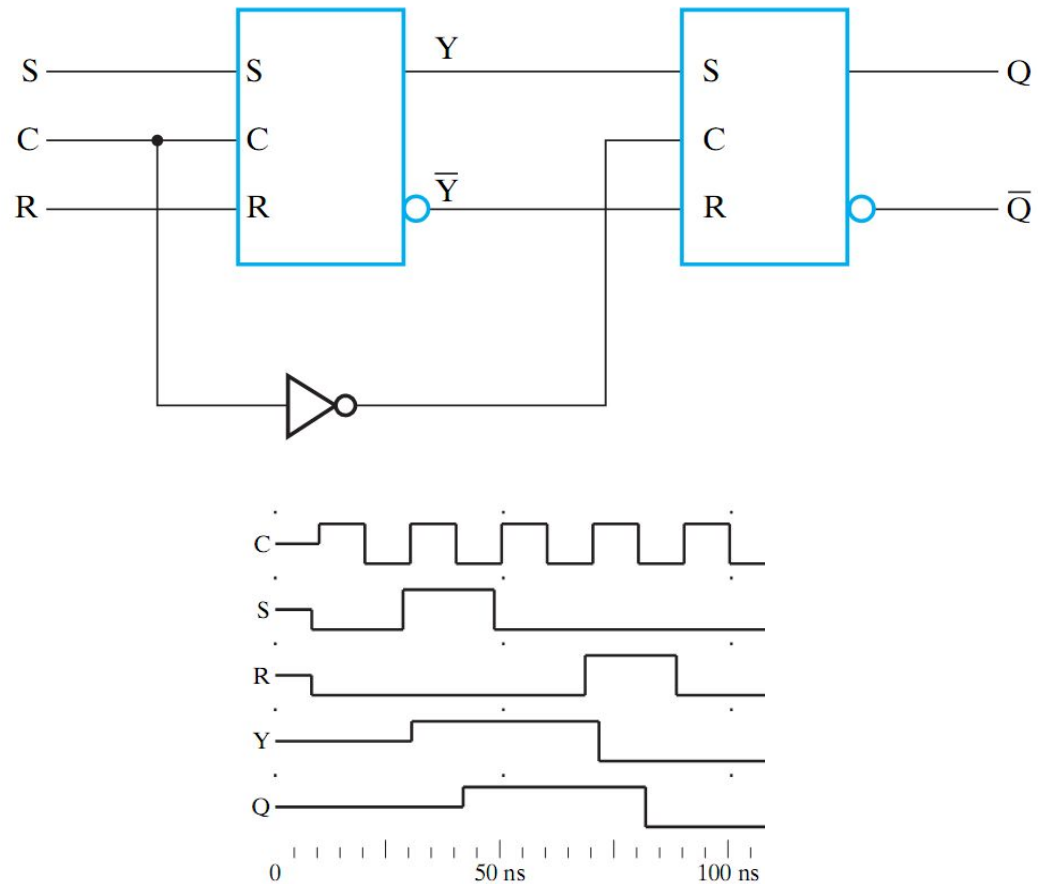
Important Timing Considerations:

- Delay of logic gates inside the flip-flop
- **Setup Time (T_s)**: The minimum time during which D input must be maintained before the clock transition occurs.
- **Hold Time (T_h)**: The minimum time during which D input must not be changed after the clock transition occurs.

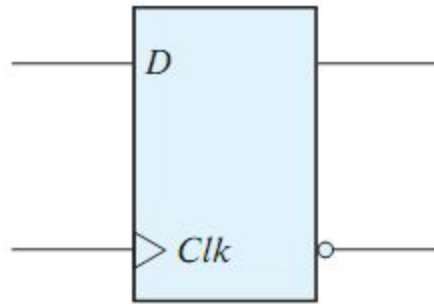


SR Flip Flop

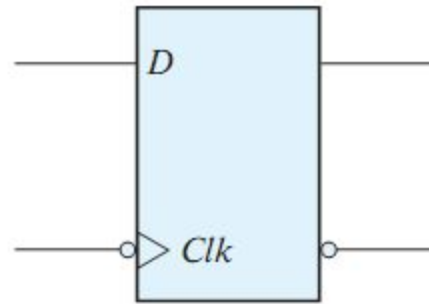
- Built using two latches (Master and Slave)
 - $C = 1$, master is active
 - $C = 0$, slave is active
- Q is sampled at the falling edge
- Data is entered on the rising edge of the clock pulse, but the output does not reflect the change until the falling edge of the clock pulse.



Graphic Symbols for Flip Flops



(a) Positive-edge

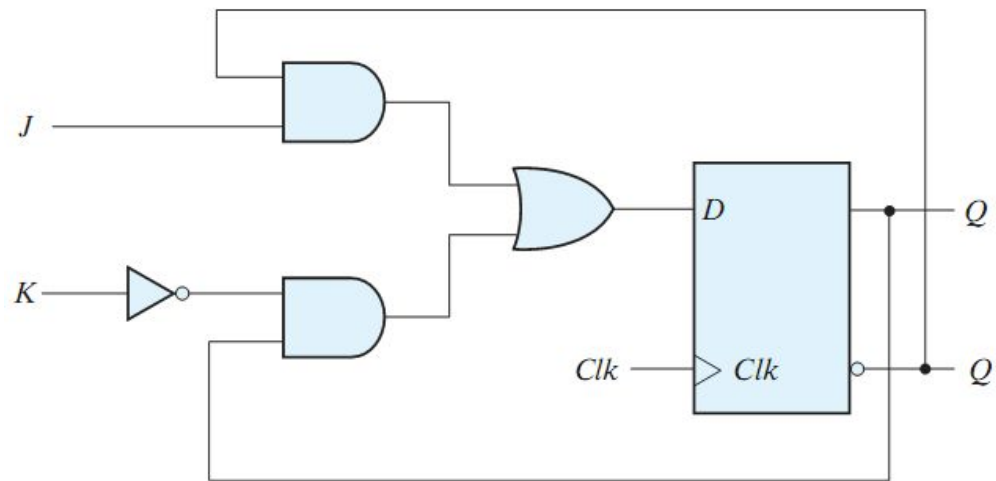


(a) Negative-edge

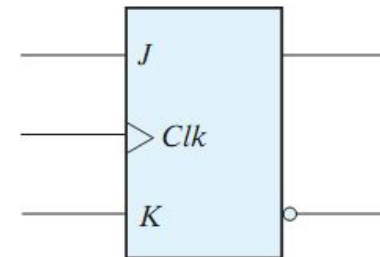
- A Flip Flop is designated by a rectangular block with inputs on the left and outputs on the right (similar to latches)
- The clock is designated with an arrowhead
- A bubble designates a negative-edge triggered flip flops

Other Flip Flops

JK Flip Flop



(a) Circuit diagram

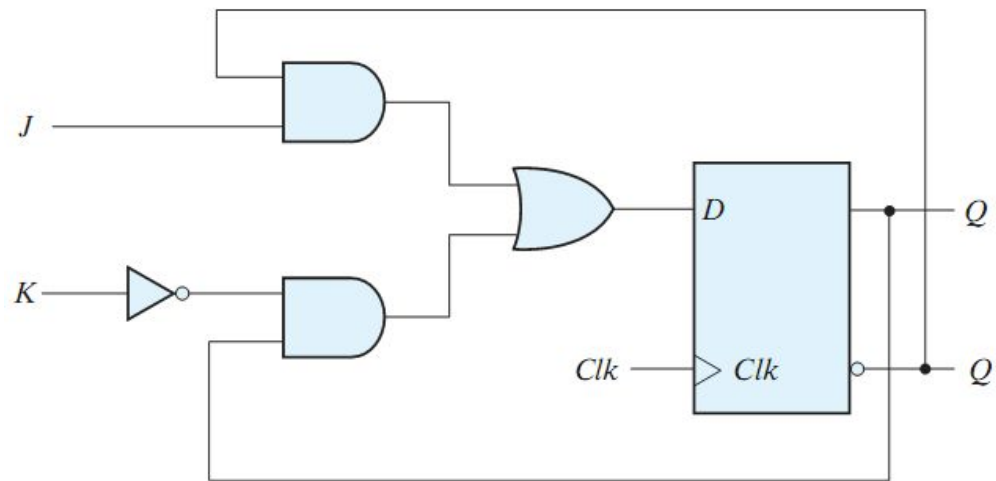


(b) Graphic symbol

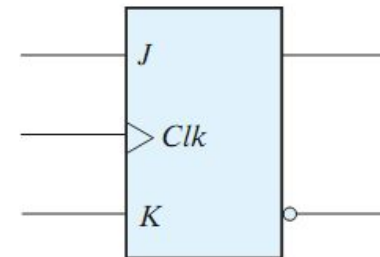
- How does it work?
- Hint: $D = ?$

Other Flip Flops

JK Flip Flop



(a) Circuit diagram

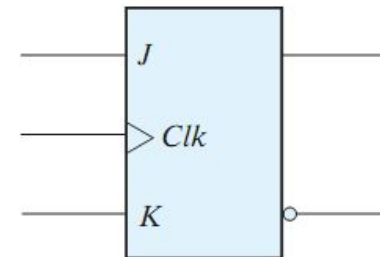
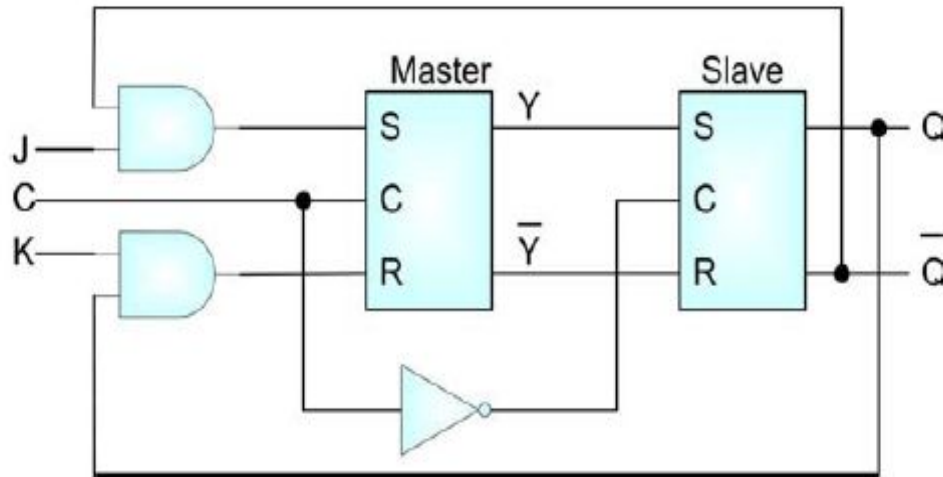


(b) Graphic symbol

- $D = J Q' + K' Q$
- J sets the flip flop (1)
- K reset the flip flop (0)
- When $J = K = 1$, the output is complemented

Other Flip Flops

JK Flip Flop



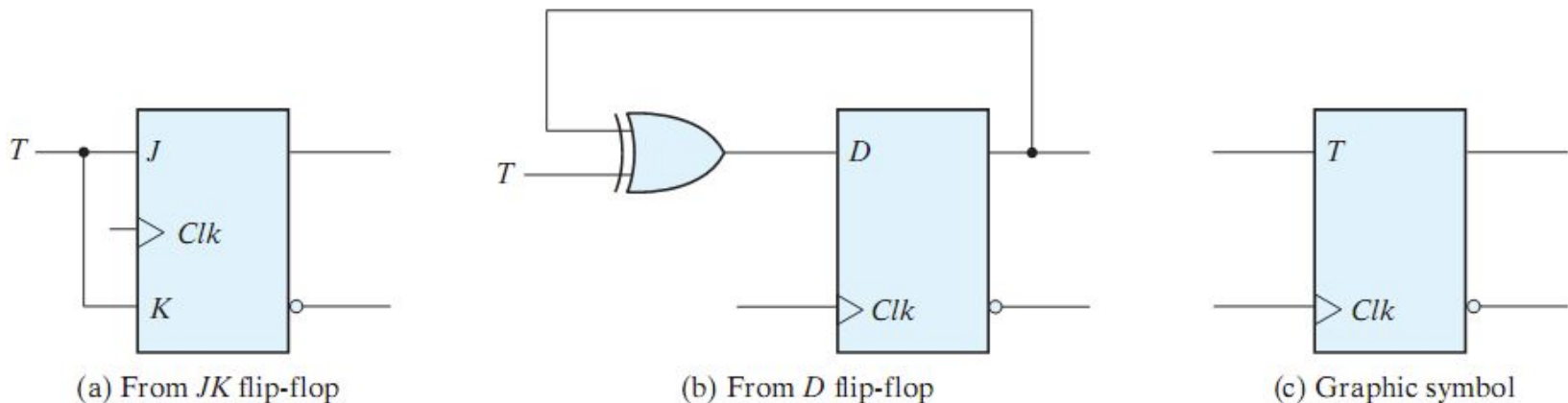
(b) Graphic symbol

- $D = J Q' + K' Q$
- J sets the flip flop (1)
- K reset the flip flop (0)
- When $J = K = 1$, the output is complemented

JK Flip Flop built with SR latches

Other Flip Flops (cont.)

T Flip Flop



- T (toggle) flip flop is a complementing flip flop
- Built with a JK or D flip flop (as shown above)
- $T = 0$, no change,
- $T = 1$, complement (toggle)
- For D-FF implementation, $D = T \oplus Q$

Characteristic Tables

- A **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 arrives)
 - $Q(t+1)$ refers to next state (**after** the clock arrives)
- Similar to the truth table in combinational circuits

Flip-Flop Characteristic Tables

<i>JK Flip-Flop</i>			
<i>J</i>	<i>K</i>	$Q(t + 1)$	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

<i>D Flip-Flop</i>		
<i>D</i>	$Q(t + 1)$	
0	0	Reset
1	1	Set

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

Characteristic Equations

- A **characteristic equation** defines the operation of a flip flop in an algebraic form
- For D-FF
 - $Q(t+1) = D$
- For JK-FF
 - $Q(t+1) = J Q' + K' Q$
- For T-FF
 - $Q(t+1) = T \oplus Q$

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

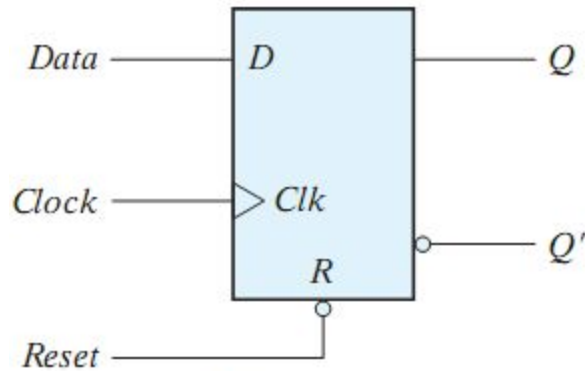
D Flip-Flop

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

Direct Inputs



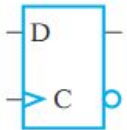
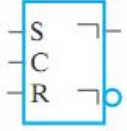
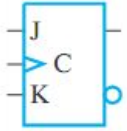
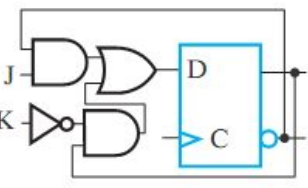
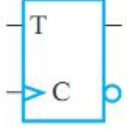
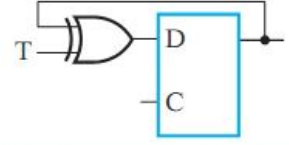
(b) Graphic symbol

R	Clk	D	Q	Q'
0	X	X	0	1
0	\uparrow	0	0	1
0	\uparrow	1	1	0

(b) Function table

- Some flip-flops have asynchronous inputs to set/reset their states independently of the clock.
- **Preset** or **direct set**, sets the flip-flop to 1
- **Clear** or **direct reset**, set the flip-flop to 0
- When power is turned on, a flip-flop state is unknown; Direct inputs are useful to put in a known state
- Figure shows a positive-edge D-FF with active-low asynchronous reset.

Flip Flops Sheet (Mano's Textbook)

Type	Symbol	Logic Diagrams	Characteristic Table			Characteristic Equation	Excitation Table					
D		See Figure 5-12	D	Q(t+1)	Operation	$Q(t+1) = D(t)$	Q(t+1)		D	Operation		
			0 1	0 1	Reset Set		0 1	0 1	Reset Set			
SR		See Figure 5-9	S	R	Q(t+1)	Operation	$Q(t+1) = S(t) + \bar{R}(t) Q(t)$	Q(t)	Q(t+1)	S	R	Operation
			0	0	Q(t)	No change		0	0	0	X	No change
			0	1	0	Reset		0	1	1	0	Set
			1	0	1	Set		1	0	0	1	Reset
			1	1	?	Undefined		1	1	X	0	No change
JK			J	K	Q(t+1)	Operation	$Q(t+1) = J(t) \bar{Q}(t) + \bar{K}(t) Q(t)$	Q(t)	Q(t+1)	J	K	Operation
			0	0	Q(t)	No change		0	0	0	X	No change
			0	1	0	Reset		0	1	1	X	Set
			1	0	1	Set		1	0	X	1	Reset
			1	1	Q̄(t)	Complement		1	1	X	0	No Change
T			T	Q(t+1)	Operation	$Q(t+1) = T(t) \oplus Q(t)$	Q(t+1)		T	Operation		
			0	Q(t)	No change		Q(t)		0	No change		
			1	Q̄(t)	Complement		Q̄(t)		1	Complement		

Summary

- In a sequential circuit, outputs depends on inputs and previous inputs
 - Previous inputs are stored as binary information into memory
 - The stored information at any time defines a state
 - Similarly, next state depends on inputs and present state
- Two types of sequential circuits: Synchronous and Asynchronous
- Two types of Memory elements: Latches and Flip-Flops.
- Flip-flops are built with latches
- A flip-flop is described using characteristic table/equation
- Flips-flops can have direct asynchronous inputs