CSE1003-Digital Logic Design

Module:5 Sequential Circuits-I

Dr.Penchalaiah Palla
Dept. of Micro and Nanoelectronics
School of Electronics Engineering,VIT, Vellore

Module:5 Sequential Circuits-I

Module:5	**SEQUENTIAL CIRCUITS – I**	**6 hours**
Flip Flops - Sequential Circuit: Design and Analysis - Finite State Machine: Moore and Mealy model - Sequence Detector.		
Module:6	**SEQUENTIAL CIRCUITS – II**	**7 hours**
Registers - Shift Registers - Counters - Ripple and Synchronous Counters - Modulo counters - Ring and Johnson counters		

Introduction:Sequential Circuits

- **Combinational**
  - The outputs depend only on the current input values
  - It uses only logic gates
- **Sequential**
  - The outputs depend on the current and past input values
  - It uses logic gates and storage elements
  - Example
    - ✓ Vending machine
  - They are referred as finite state machines since they have a finite number of states

Block Diagram

- Memory elements can store binary information
  - This information at any given time determines the state of the circuit at that time

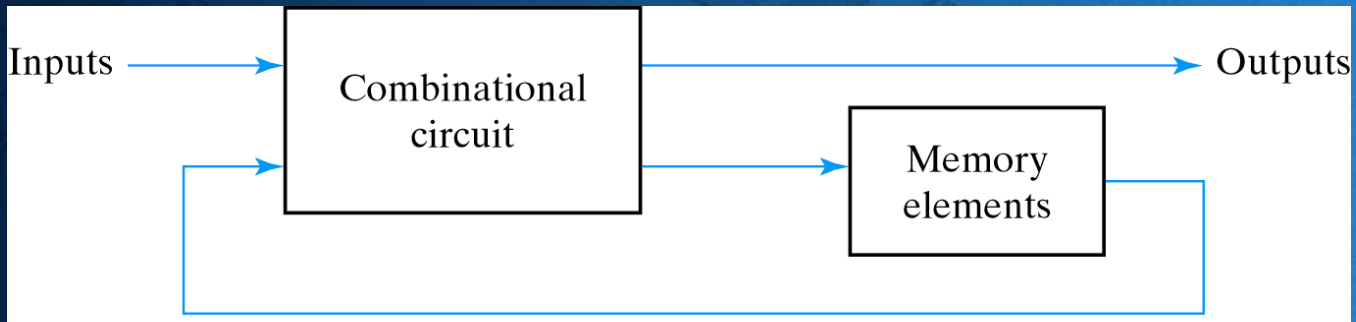
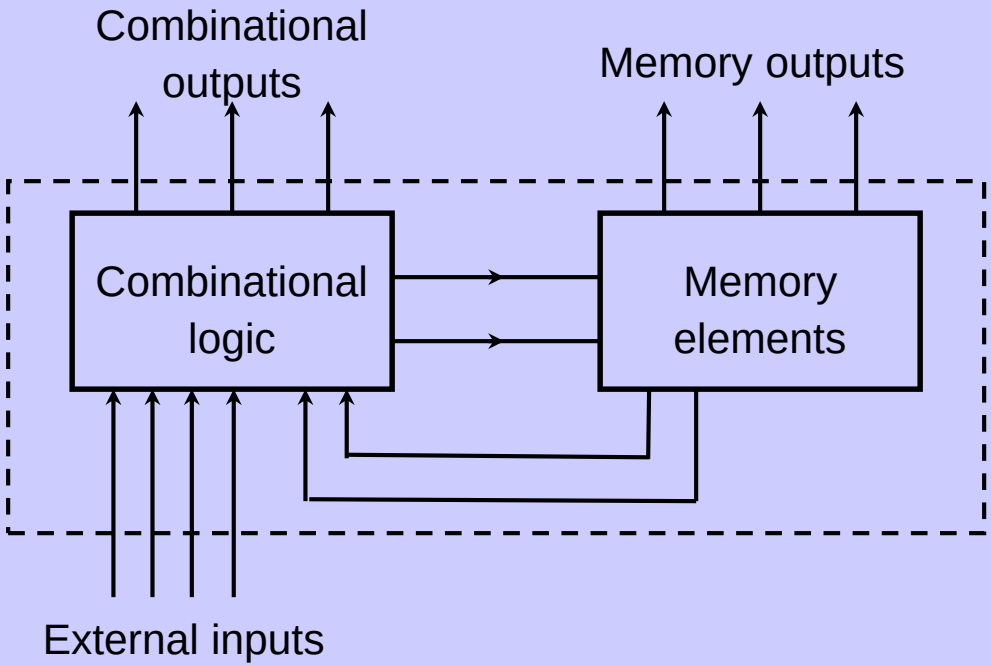


Fig. 5-1 Block Diagram of Sequential Circuit

- A **sequential circuit** consists of a *feedback path*, and employs some *memory elements*.



Sequential circuit = Combinational logic + Memory Elements

Sequential Circuit Types

- **Synchronous**
  - The circuit behavior is determined by the signals at discrete instants of time
  - The memory elements are affected only at discrete instants of time
  - A clock is used for synchronization
    - ✓ Memory elements are affected only with the arrival of a clock pulse
    - ✓ If memory elements use clock pulses in their inputs, the circuit is called
      - Clocked sequential circuit

Sequential Circuit Types

- **ASynchronous**
  - **The circuit behavior is determined by the signals at any instant of time**
  - **It is also affected by the order the inputs change**

IN short……..

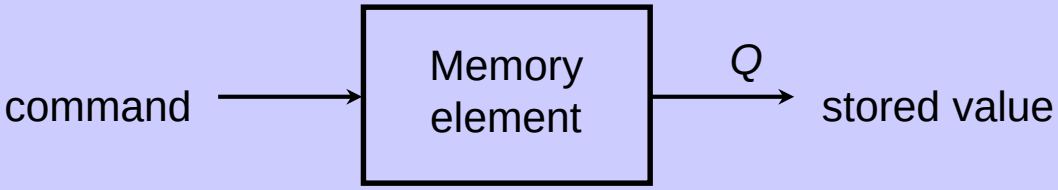
- There are two types of sequential circuits:
  - *synchronous*: outputs change only at specific time
  - *asynchronous*: outputs change at any time
- *Multivibrator*: a class of sequential circuits. They can be:
  - *bistable* (2 stable states)
  - *monostable* or *one-shot* (1 stable state)
  - *astable* (no stable state)
- Bistable logic devices: *latches* and *flip-flops*.
- Latches and flip-flops differ in the method used for changing their state.

Latches & Flip-flops

- Memory Elements
- Pulse-Triggered Latch
  - S-R Latch
  - Gated S-R Latch
  - Gated D Latch
- Edge-Triggered Flip-flops
  - S-R Flip-flop
  - D Flip-flop
  - J-K Flip-flop
  - T Flip-flop
- Asynchronous Inputs

Memory Elements

- **Memory element**: a device which can remember value indefinitely, or change value on command from its inputs.



- Characteristic table:

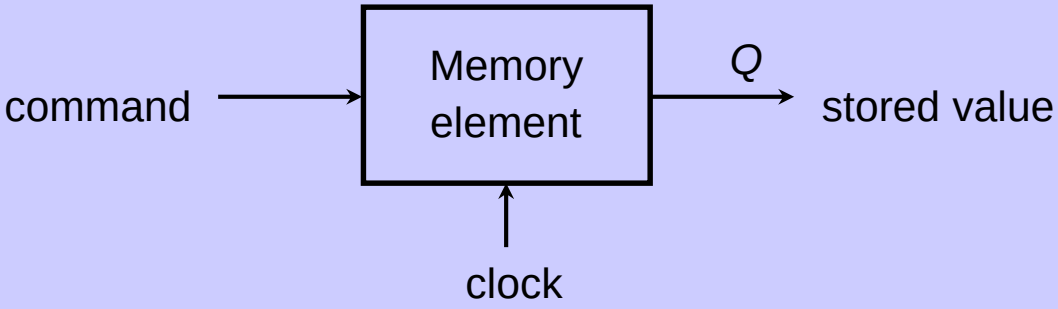
Command (at time *t*)	*Q(t)*	*Q(t+1)*
Set	X	1
Reset	X	0
Memorise / No Change	0	0
	1	1

Q(t): current state

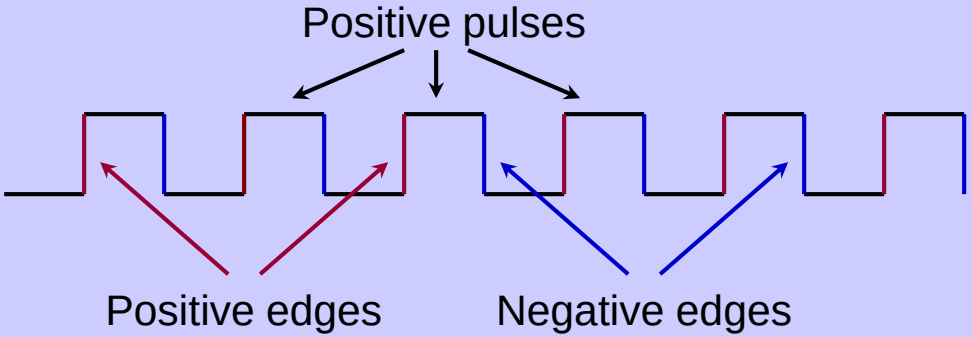
Q(t+1) or Q^+: next state

Memory Elements

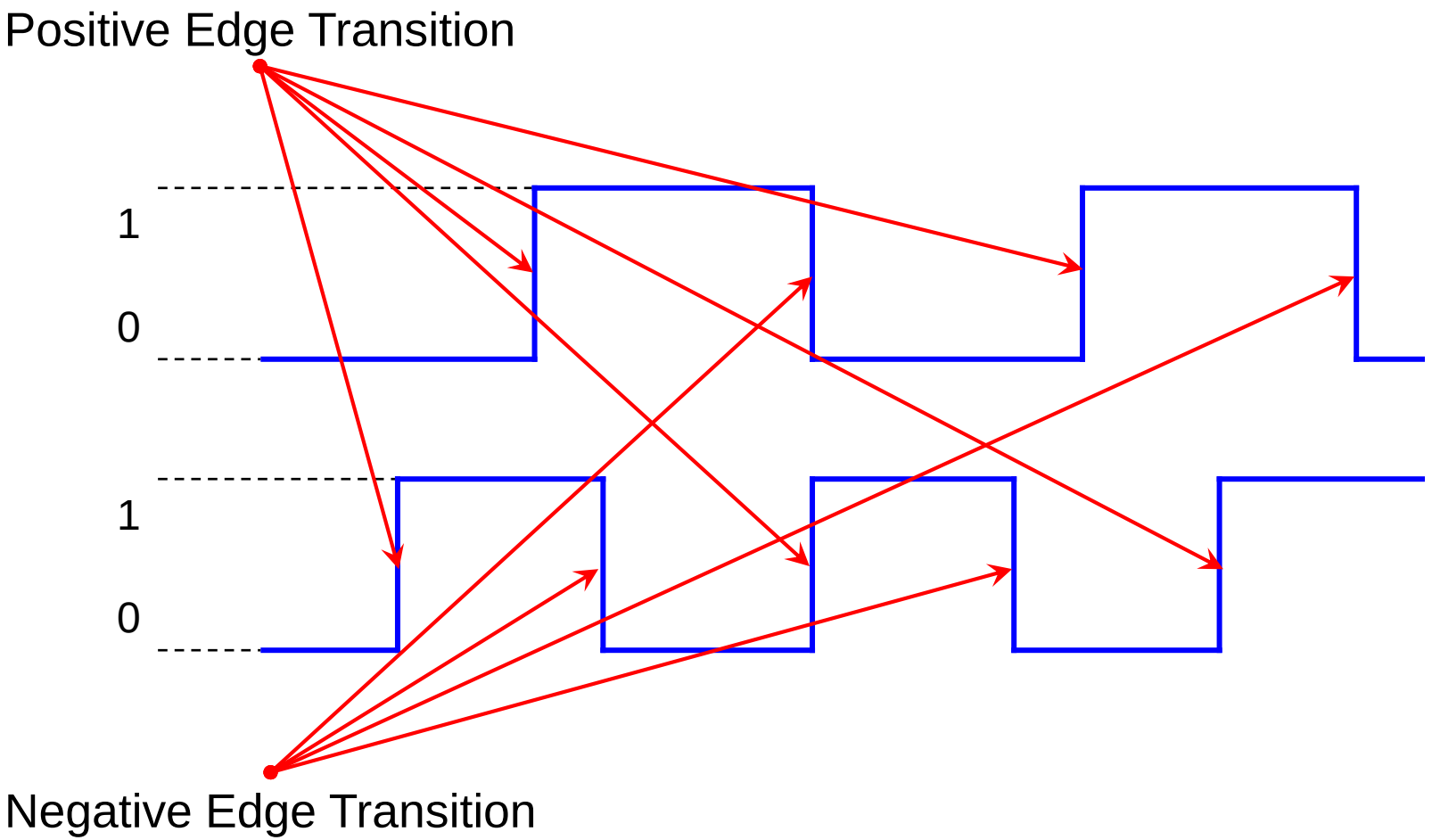
- Memory element with clock. Flip-flops are memory elements that change state on clock signals.



- Clock is usually a square wave.



Clock Edges

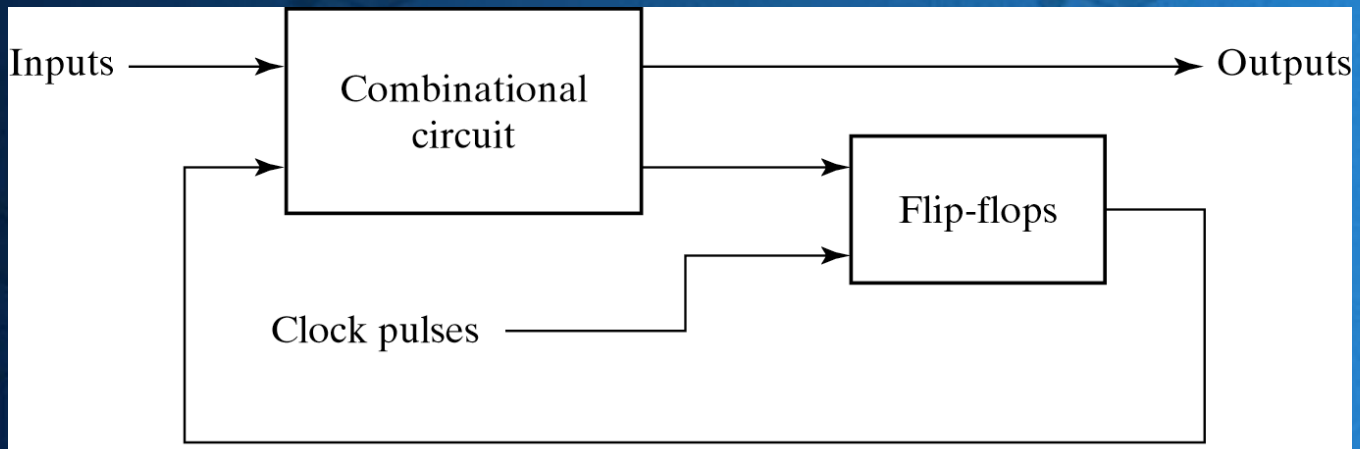


Memory Elements

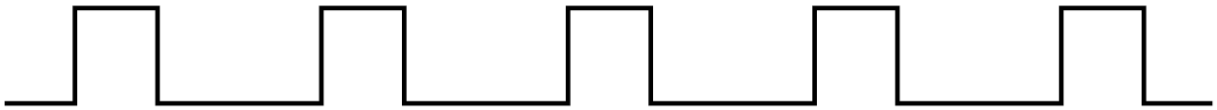
- Two types of triggering/activation:
  - pulse-triggered
  - edge-triggered
- Pulse-triggered
  - latches
  - ON = 1, OFF = 0
- Edge-triggered
  - flip-flops
  - positive edge-triggered (ON = from 0 to 1; OFF = other time)
  - negative edge-triggered (ON = from 1 to 0; OFF = other time)

Flip-Flops

- They are memory elements
- They can store binary information



(a) Block diagram



(b) Timing diagram of clock pulses

Fig. 5-2 Synchronous Clocked Sequential Circuit

Clock

- It emits a series of pulses with a precise pulse width and precise interval between consecutive pulses
- Timing interval between the corresponding edges of two consecutive pulses is known as the clock cycle time, or period

Flip-Flops

- Can keep a binary state until an input signal to switch the state is received
- There are different types of flip-flops depending on the number of inputs and how the inputs affect the binary state

Latches

- The most basic flip-flops
  - They operate with signal levels
- The flip-flops are constructed from latches
- They are not useful for synchronous sequential circuits
- They are useful for asynchronous sequential circuits

S-R Latch

- *Complementary* outputs: *Q* and *Q'*.
- When *Q* is HIGH, the latch is in *SET* state.
- When *Q* is LOW, the latch is in *RESET* state.
- For *active-HIGH input* S-R latch (also known as NOR gate latch),

  R=HIGH (and *S*=LOW) ⇨ RESET state

  S=HIGH (and *R*=LOW) ⇨ SET state

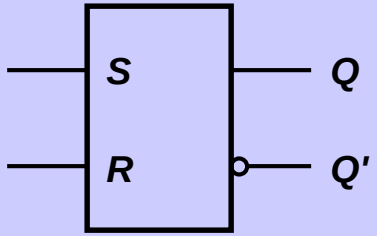
  both inputs LOW ⇨ no change

  both inputs HIGH ⇨ *Q* and *Q'* both LOW (invalid)!

S-R Latch

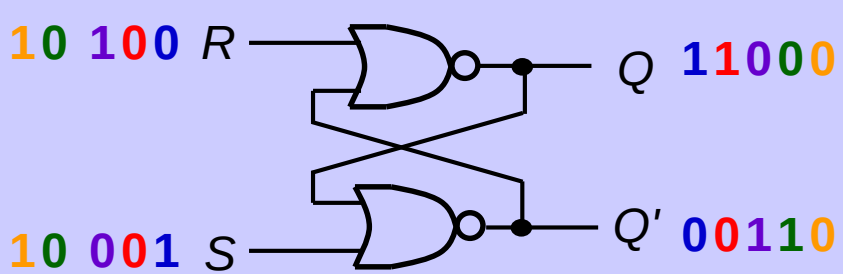
- Characteristics table for active-high input S-R latch:

S	*R*	*Q*	*Q′*	
0	0	NC	NC	No change. Latch remained in present state.
1	0	1	0	Latch SET.
0	1	0	1	Latch RESET.
1	1	0	0	Invalid condition.



S-R Latch with NOR

- Active-HIGH input S-R latch



S	*R*	*Q*	*Q′*	
1	0	1	0	initial
0	0	1	0	(afer *S*=1, *R*=0)
0	1	0	1	
0	0	0	1	(after *S*=0, *R*=1)
1	1	0	0	invalid!

SR Latch with NOR

$S = set$

$R = reset$

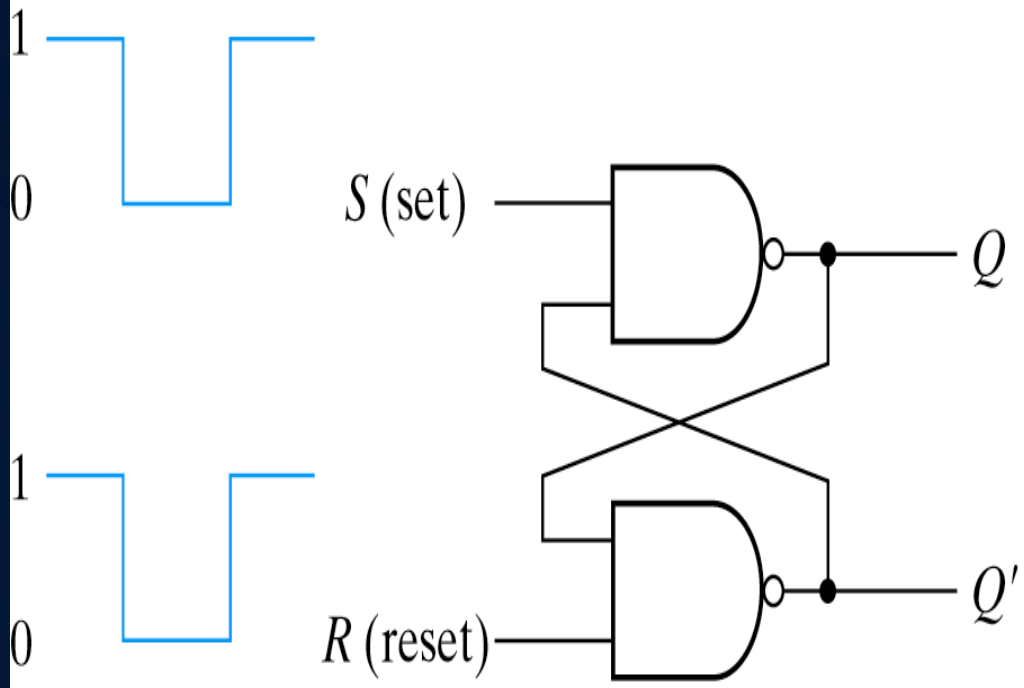
$Q = 1, \quad Q' = 0 \quad \Rightarrow$ set state

$Q = 0, \quad Q' = 1 \quad \Rightarrow$ reset state

$S = 1, \quad R = 1 \quad \Rightarrow$ undefined, Q and Q' are set to 0

In normal conditions, avoid $S = 1, R = 1$

SR Latch with NAND



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	

(a) Logic diagram

(b) Function table

Fig. 5-4 *SR* Latch with NAND Gates

SR Latch with NAND

$S = set$

$R = reset$

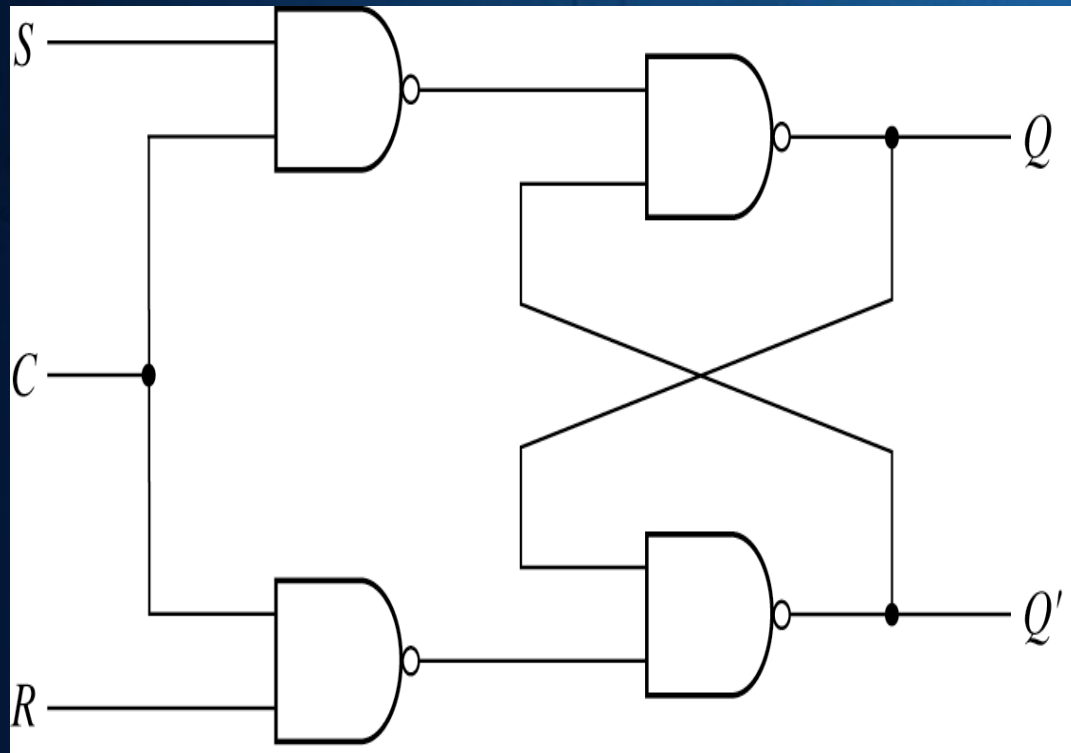
$Q = 0, \quad Q' = 1 \quad \Rightarrow$ set state

$Q = 1, \quad Q' = 0 \quad \Rightarrow$ reset state

$S = 0, \quad R = 0 \quad \Rightarrow$ undefined, Q and Q' are set to 1

In normal conditions, avoid $S = 0, R = 0$

SR Latch with Control Input



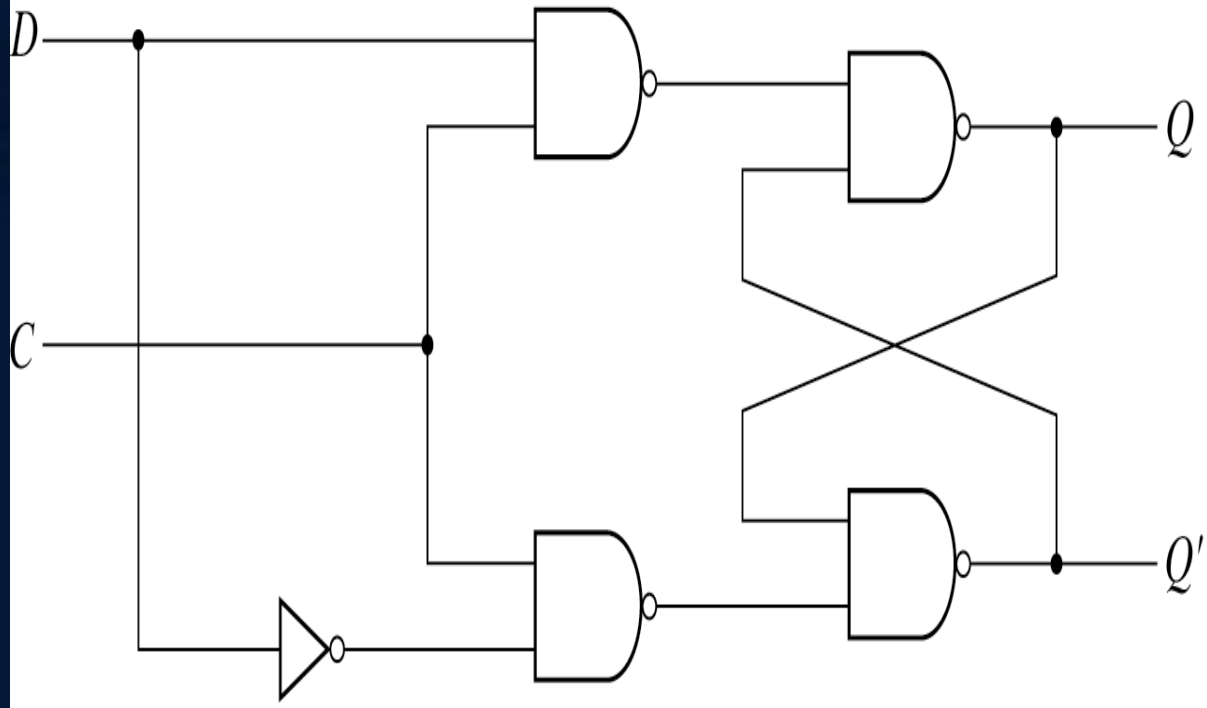
(a) Logic diagram

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

Fig. 5-5 SR Latch with Control Input

D Latch



(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

Fig. 5-6 D Latch

Symbols for Latches

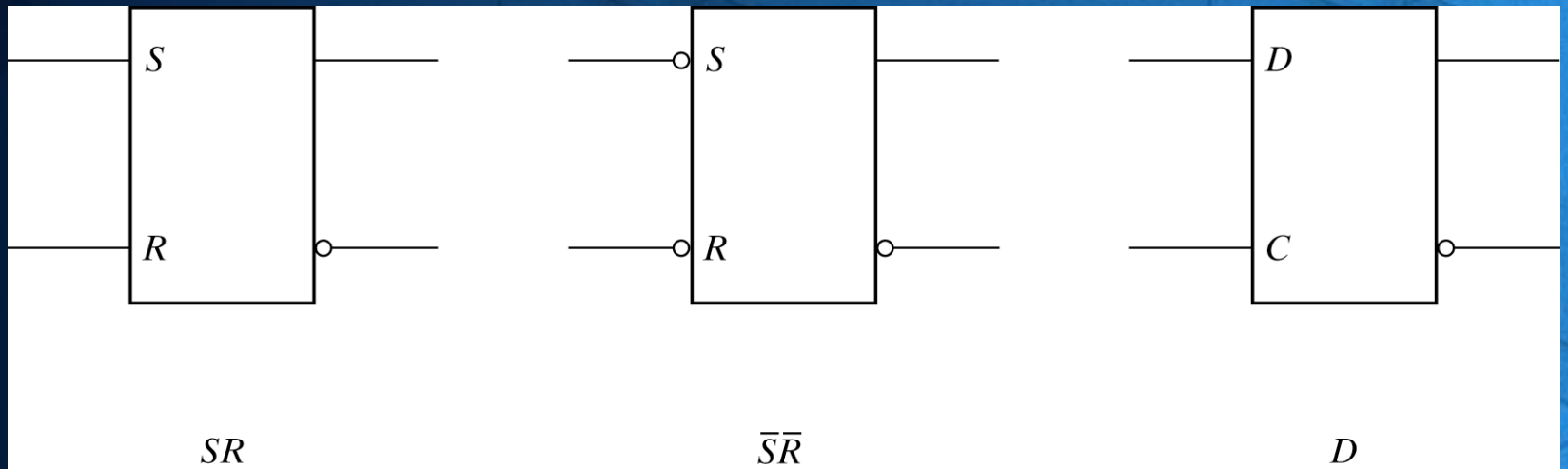


Fig. 5-7 Graphic Symbols for Latches

Note

- The control input changes the state of a latch or flip-flop
- The momentary change is called a trigger
- Example: D Latch
  - It is triggered every time the pulse goes to the logic level 1
  - As long as the pulse remains at the logic level 1, the change in the data (D) directly affects the output (Q)
  - THIS MAY BE A BIG PROBLEM since the state of the latch may keep changing depending on the input (may be coming from a combinational logic network)

How to Solve?

- Trigger the flip-flop only during a signal transition

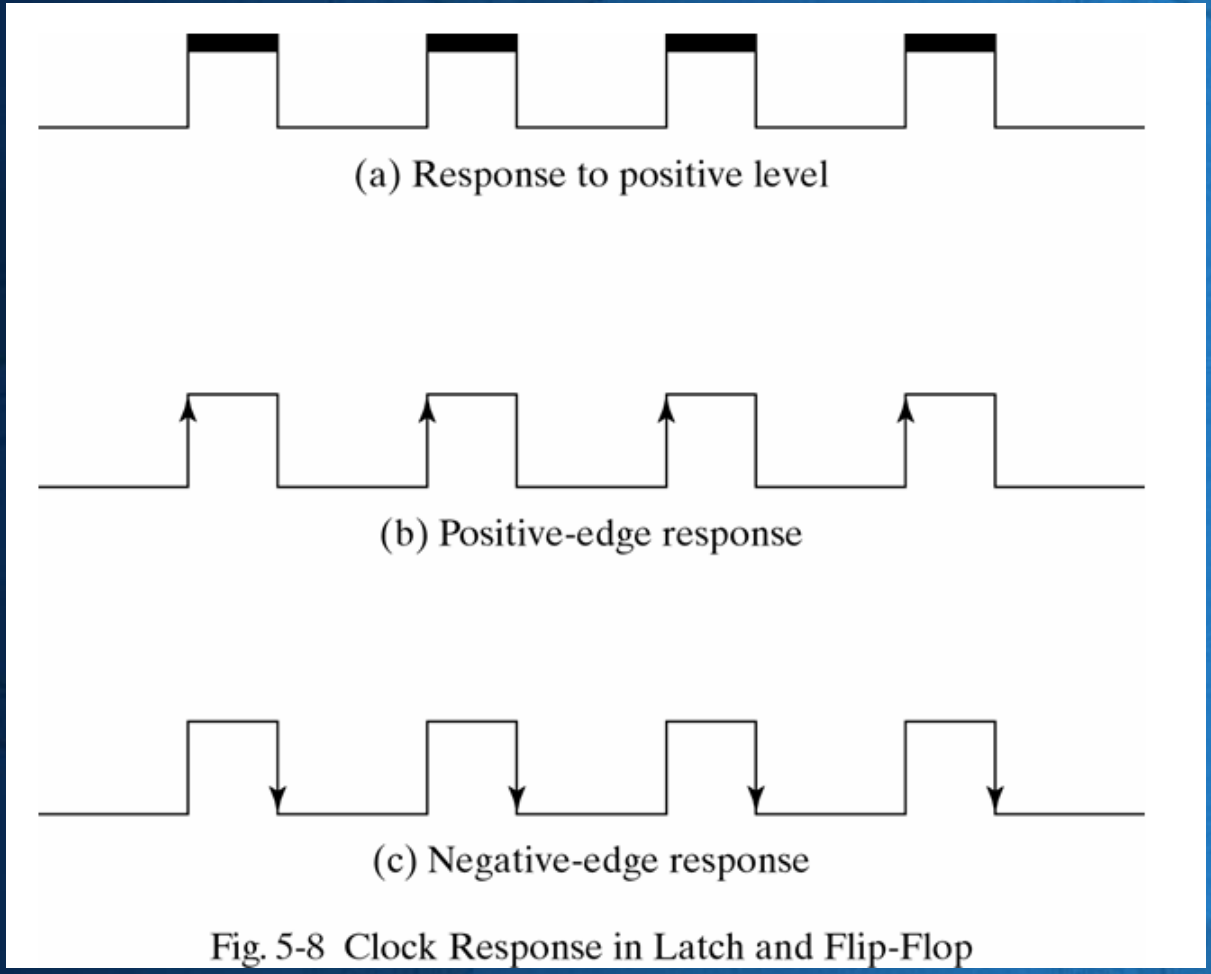


Fig. 5-8 Clock Response in Latch and Flip-Flop

Edge-Triggered D Flip-Flop

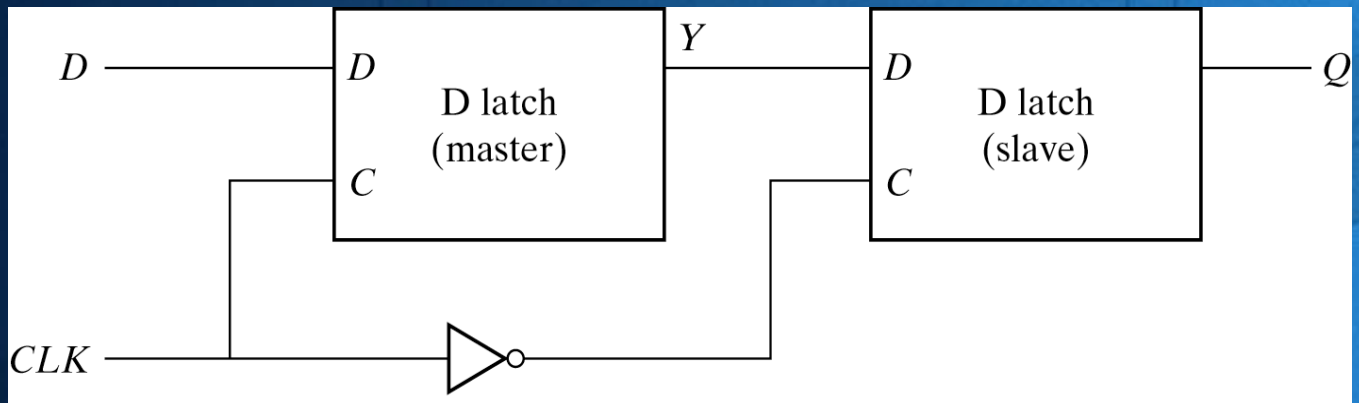


Fig. 5-9 Master-Slave *D* Flip-Flop

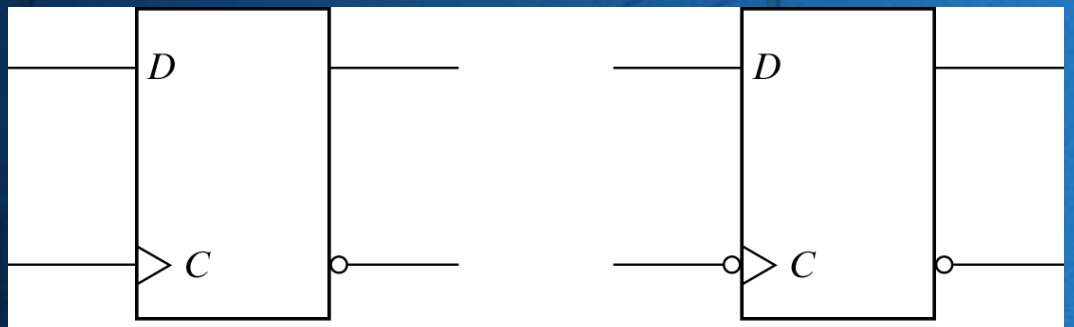
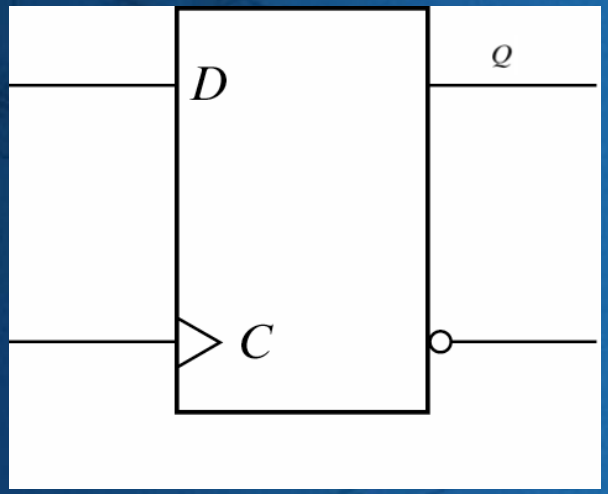


Fig. 5-11 Graphic Symbol for Edge-Triggered *D* Flip-Flop

Characteristics of D Flip-Flop

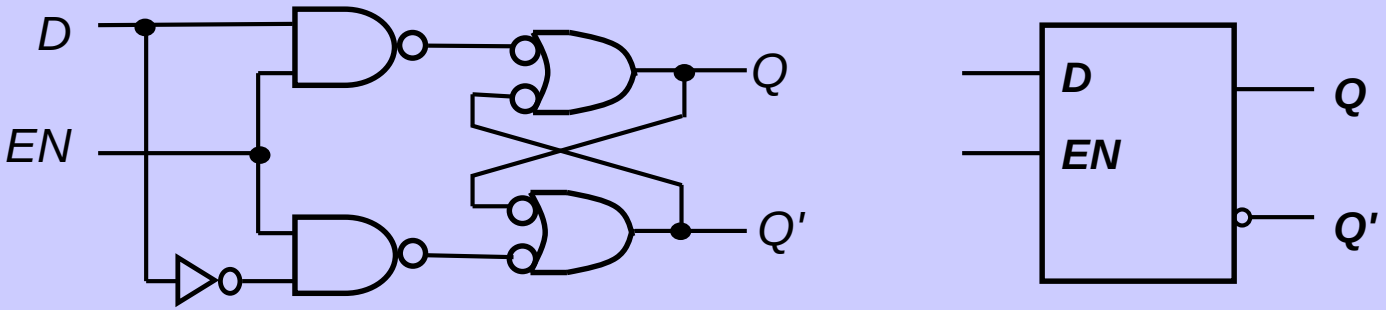


Inputs		Outputs		
D	C	Q	Q'	Comments
0	↑	0	1	RESET
1	↑	1	0	SET

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

Gated D Latch

- Make *R* input equal to *S′* → *gated D latch.*
- *D* latch eliminates the undesirable condition of invalid state in the *S*-*R* latch.



Gated D Latch

- When *EN* is HIGH,
  - *D*=HIGH → latch is SET
  - *D*=LOW → latch is RESET
- Hence when *EN* is HIGH, *Q* 'follows' the *D* (data) input.
- Characteristic table:

EN	*D*	*Q(t+1)*	
1	**0**	**0**	**Reset**
1	**1**	**1**	**Set**
0	**X**	***Q(t)***	**No change**

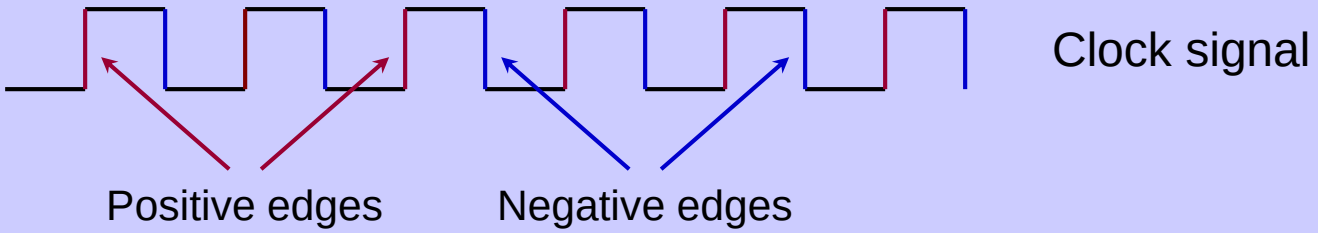
When *EN*=1, *Q(t+1)* = *D*

Latch Circuits: Not Suitable

- Latch circuits are not suitable in synchronous logic circuits.
- When the enable signal is active, the excitation inputs are gated directly to the output Q. Thus, any change in the excitation input immediately causes a change in the latch output.
- The problem is solved by using a special timing control signal called a *clock* to restrict the times at which the states of the memory elements may change.
- This leads us to the edge-triggered memory elements called *flip-flops*.

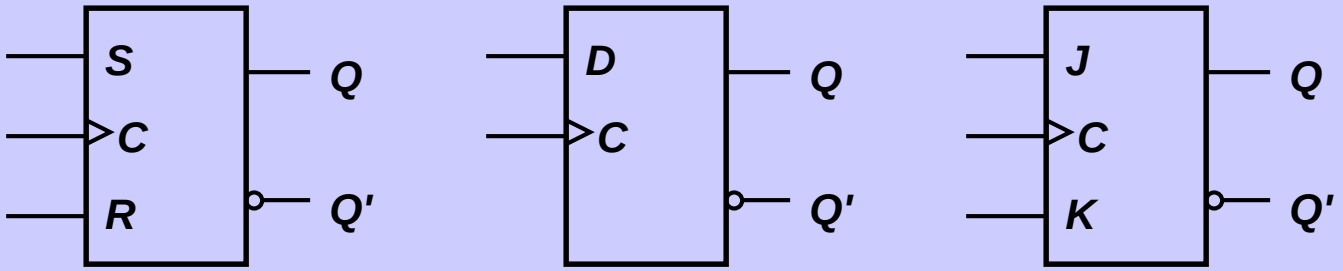
Edge-Triggered Flip-flops

- *Flip-flops*: synchronous bistable devices
- Output changes state at a specified point on a triggering input called the *clock*.
- Change state either at the *positive edge* (rising edge) or at the *negative edge* (*falling edge*) of the clock signal.

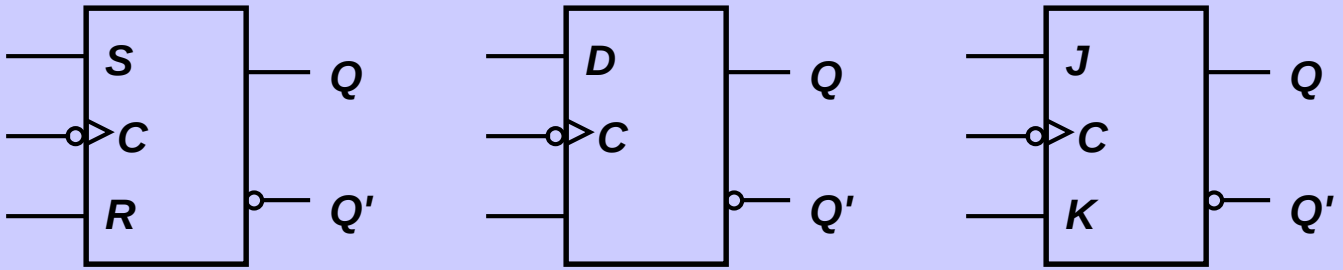


Edge-Triggered Flip-flops

- S-R, D and J-K edge-triggered flip-flops. Note the “>” symbol at the clock input.



Positive edge-triggered flip-flops



Negative edge-triggered flip-flops

S-R Flip-flop

- S-R flip-flop: on the triggering edge of the clock pulse,
  - *S*=HIGH (and *R*=LOW) ⇨ SET state
  - *R*=HIGH (and *S*=LOW) ⇨ RESET state
  - both inputs LOW ⇨ no change
  - both inputs HIGH ⇨ invalid
- Characteristic table of positive edge-triggered S-R flip-flop:

S	*R*	*CLK*	*Q(t+1)*	Comments
0	0	X	*Q(t)*	No change
0	1	↑	0	Reset
1	0	↑	1	Set
1	1	↑	?	Invalid

X = irrelevant (“don’t care”)

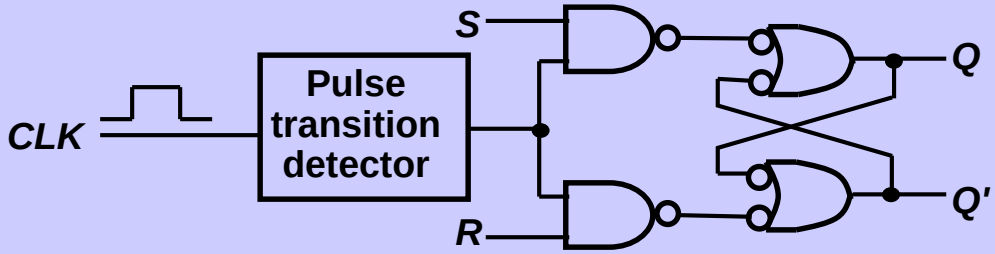
↑ = clock transition LOW to HIGH

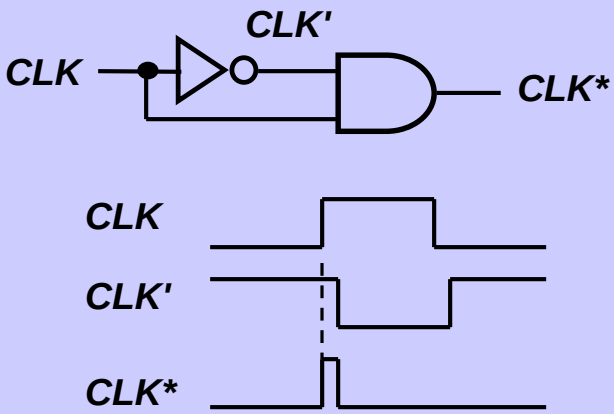
S-R Flip-flop

- It comprises 3 parts:
  - a basic *NAND latch*
  - a *pulse-steering* circuit
  - a *pulse transition detector* (or *edge detector*) circuit
- The pulse transition detector detects a rising (or falling) edge and produces a very *short-duration spike*.

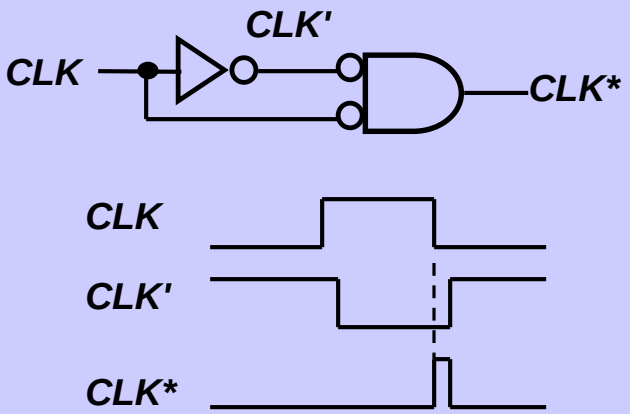
S-R Flip-flop

The pulse transition detector.





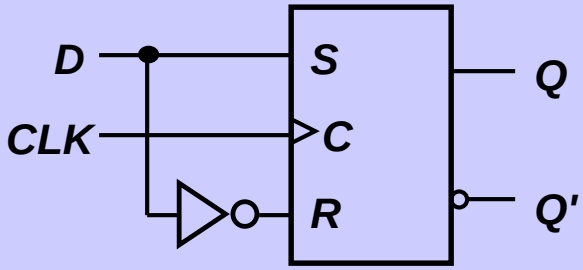
Positive-going transition (rising edge)



Negative-going transition (falling edge)

Edge-Triggered D Flip-Flop

- D flip-flop: single input *D* (data)
  - *D*=HIGH ⇨ SET state
  - *D*=LOW ⇨ RESET state
- *Q* follows *D* at the clock edge.
- Convert S-R flip-flop into a D flip-flop: add an inverter.



A positive edge-triggered D flip-flop formed with an S-R flip-flop.

D	*CLK*	*Q(t+1)*	Comments
1	↑	1	Set
0	↑	0	Reset

↑ = clock transition LOW to HIGH

Edge-Triggered D Flip-Flop

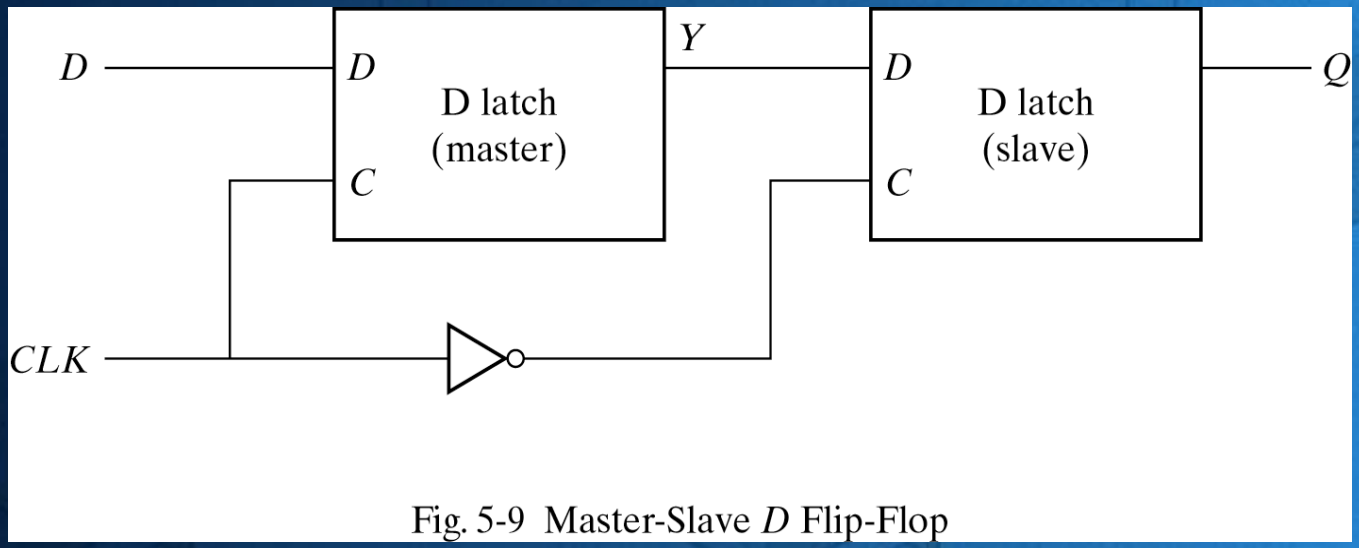
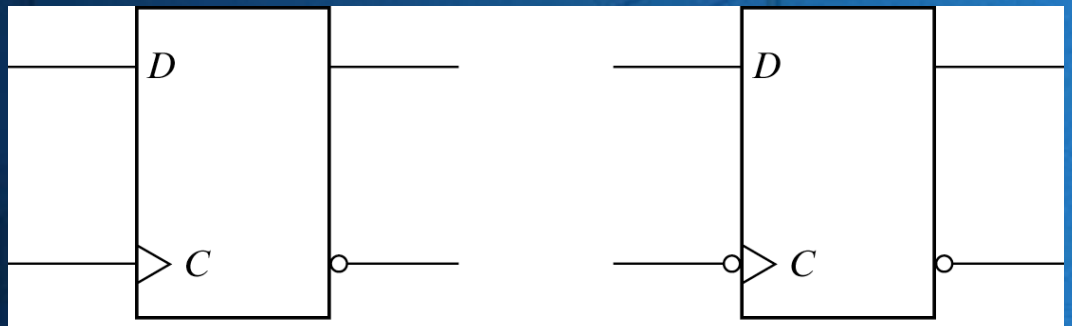


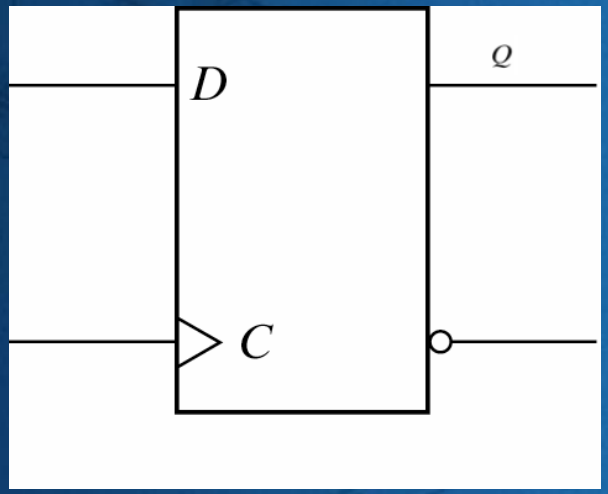
Fig. 5-9 Master-Slave *D* Flip-Flop



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

Fig. 5-11 Graphic Symbol for Edge-Triggered *D* Flip-Flop

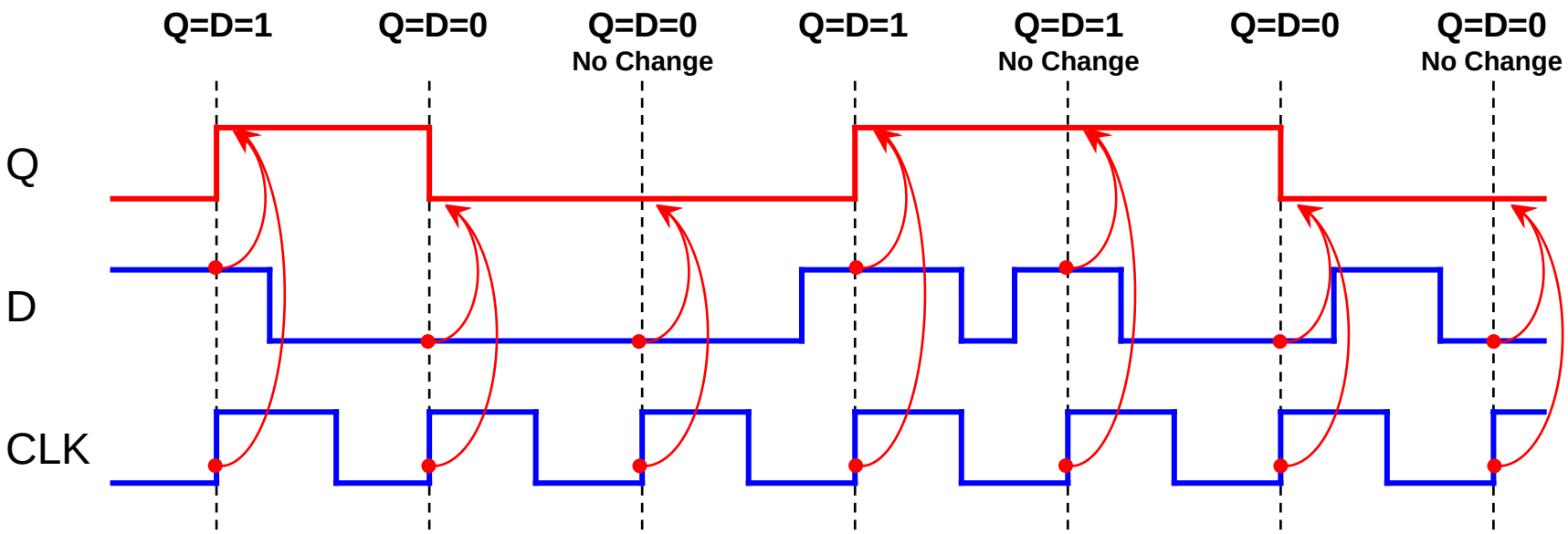
Characteristics of D Flip-Flop



Inputs		Outputs		
D	C	Q	Q'	Comments
0	↑	0	1	RESET
1	↑	1	0	SET

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

D Flip-Flop: Example Timing

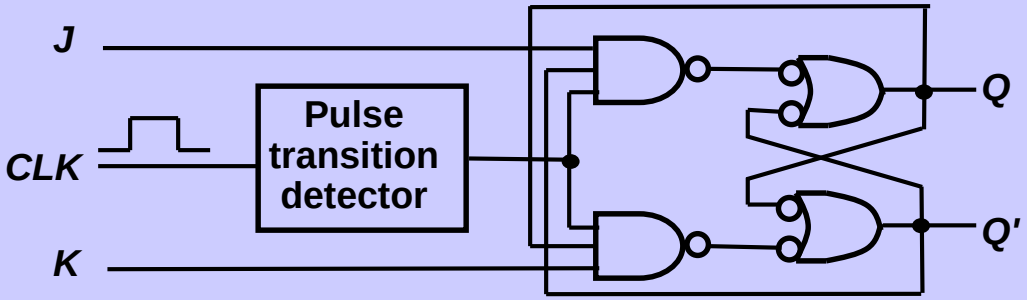


Edge-Triggered J-K Flip-Flop

- J-K flip-flop: Q and Q' are fed back to the pulse-steering NAND gates.
- No invalid state.
- Include a *toggle* state.
  - *J*=HIGH (and *K*=LOW) ⇨ SET state
  - *K*=HIGH (and *J*=LOW) ⇨ RESET state
  - both inputs LOW ⇨ no change
  - both inputs HIGH ⇨ toggle

Edge-Triggered J-K Flip-Flop

- J-K flip-flop.



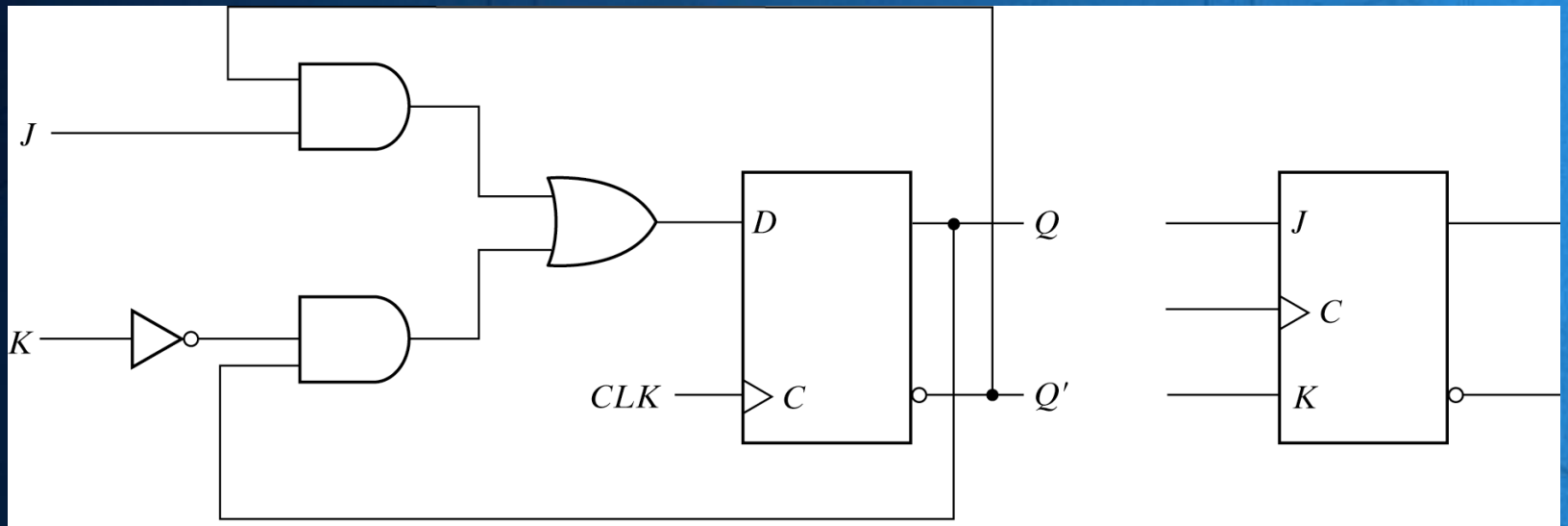
- Characteristic table.

J	K	CLK	Q(t+1)	Comments
0	0	↑	Q(t)	No change
0	1	↑	0	Reset
1	0	↑	1	Set
1	1	↑	Q(t)'	Toggle

Q	J	K	Q(t+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

$$Q(t+1) = J.Q' + K'.Q$$

Edge-Triggered J-K Flip-Flop



(a) Circuit diagram (b) Graphic symbol

Fig. 5-12 *JK* Flip-Flop

Inputs			Outputs		
J	K	C	Q	Q'	Comments
0	0	↑	Q	Q'	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	Q'	Q	Toggle

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

How???????

Excitation Table

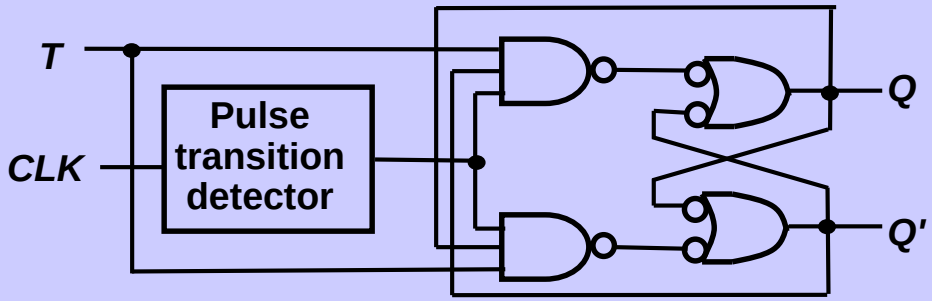
Q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

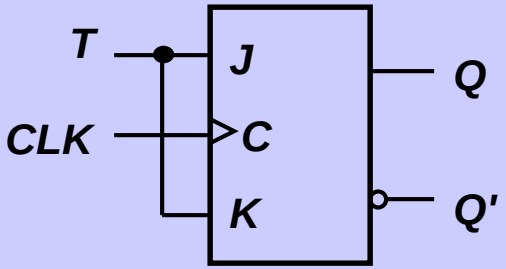
Excitation table - shows the minimum inputs that are required to generate a particular next state or to "excite" it to the next state, when the current state is known.

They are similar to truth tables, except for the rearrangement of the data. Here, the current state and next state are next to each other on the left-hand side of the table, and the inputs needed to make that state change happen are shown on the right side of the table.

Edge-Triggered T Flip-Flop

- T flip-flop: single-input version of the J-K flip flop, formed by tying both inputs together.





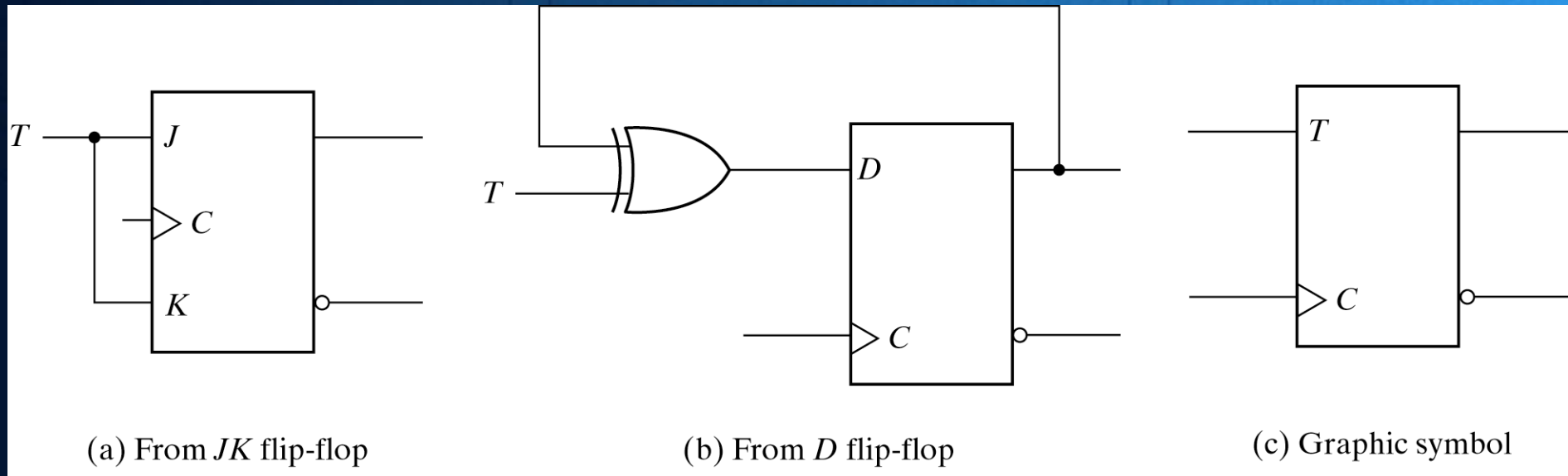
- Characteristic table.

T	*CLK*	*Q(t+1)*	Comments
0	↑	*Q(t)*	No change
1	↑	*Q(t)'*	Toggle

Q	*T*	*Q(t+1)*
0	0	0
0	1	1
1	0	1
1	1	0

$$Q(t+1) = T.Q' + T'.Q$$

Edge-Triggered T Flip-Flop



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

Fig. 5-13 T Flip-Flop

T	$Q(t+1)$
0	$Q(t)$
1	$Q'(t)$

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

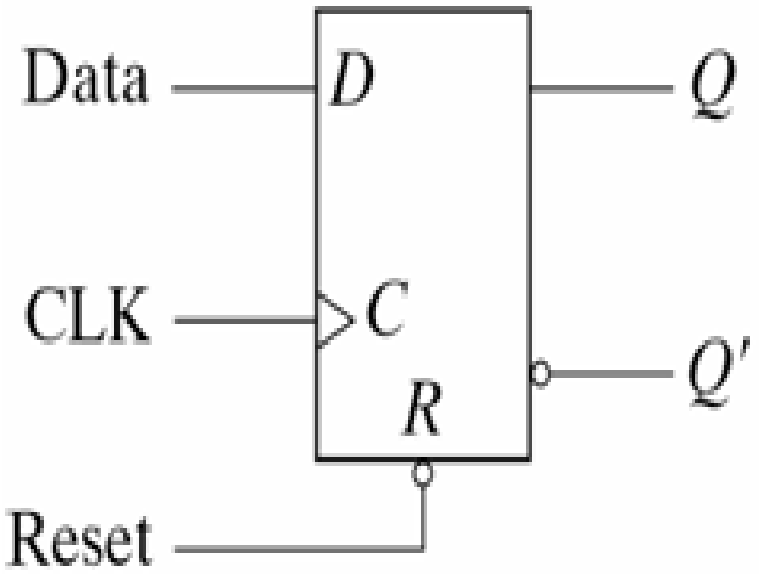
Excitation Table

Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

Direct Inputs

- You can use asynchronous inputs to put a flip-flop to a specific state regardless of the clock
- You can clear the content of a flip-flop
  - The content is changed to zero (0)
  - This is called clear or direct reset
  - This is particularly useful when the power is off
    - The state of the flip-flop is set to unknown

D Flip-Flop with Asynchronous Reset



(b) Graphic symbol

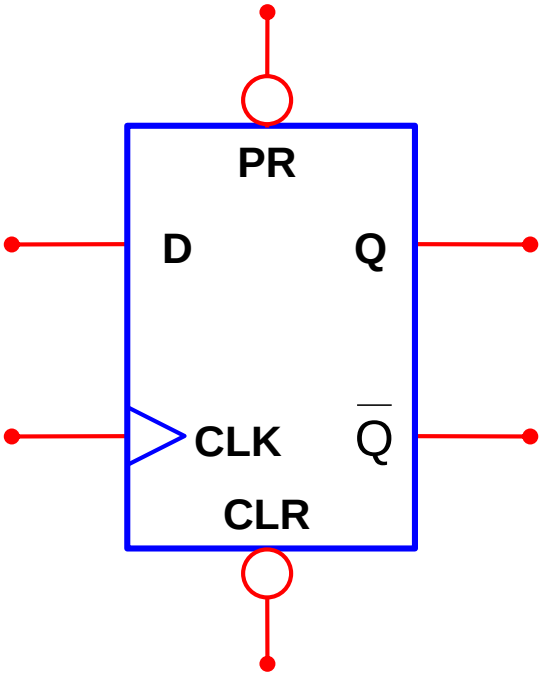
R	C	D	Q	Q′
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0

(b) Function table

Fig. 5-14 *D* Flip-Flop with Asynchronous Reset

Asynchronous Inputs(contd…)

Asynchronous inputs (Preset & Clear) are used to override the clock/data inputs and force the outputs to a predefined state.



The Preset (PR) input forces the output to:

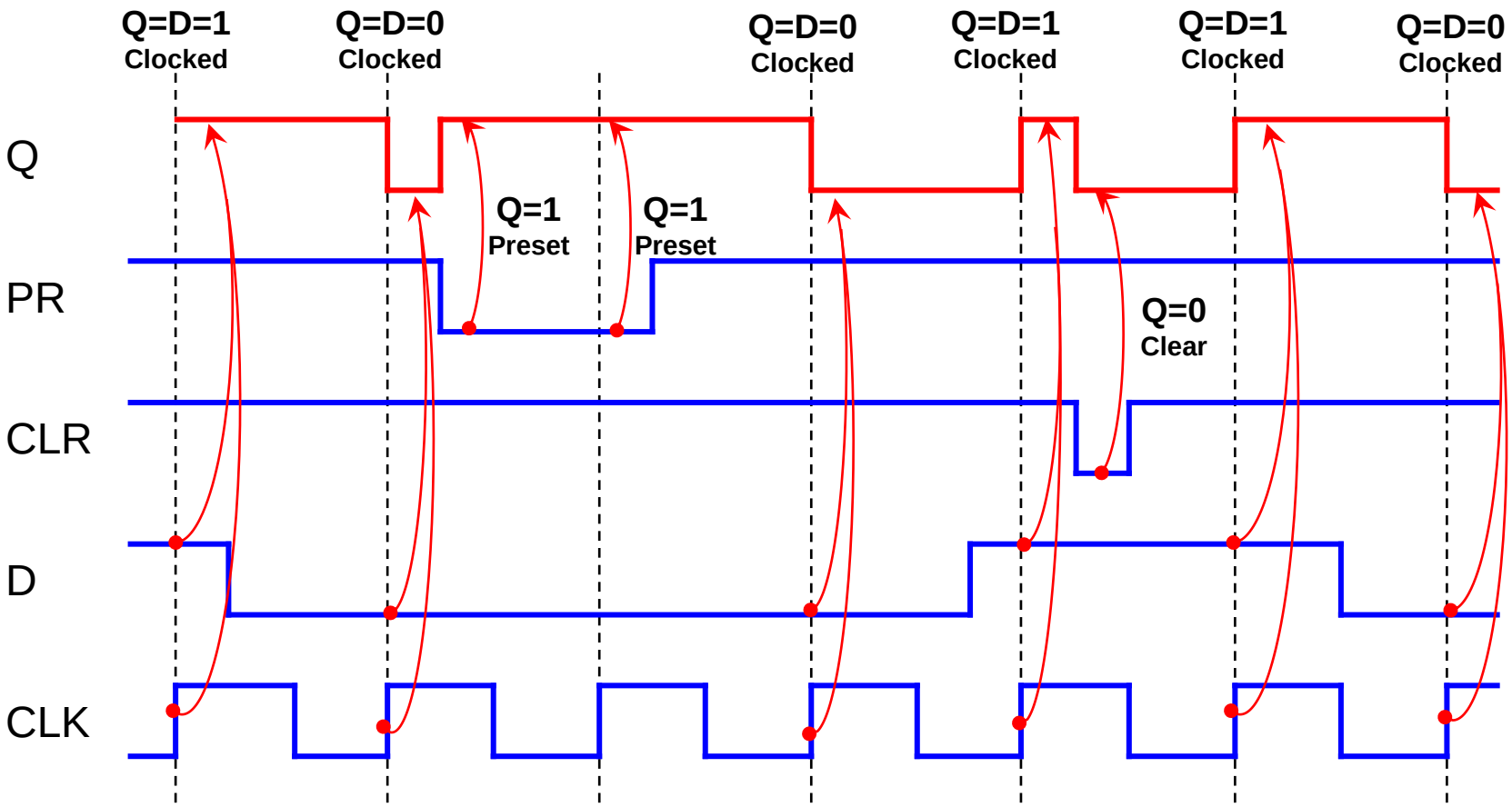
$$Q = 1 \ \& \ \overline{Q} = 0$$

The Clear (CLR) input forces the output to:

$$Q = 0 \ \& \ \overline{Q} = 1$$

PR PRESET	CLR CLEAR	CLK CLOCK	D DATA	Q	$\overline{Q}$	
1	1	↑	0	0	1	
1	1	↑	1	1	0	
0	1	X	X	1	0	Asynchronous Preset
1	0	X	X	0	1	Asynchronous Clear
0	0	X	X	1	1	ILLEGAL CONDITION

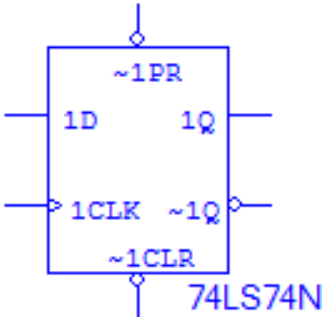
D Flip-Flop: PR & CLR Timing



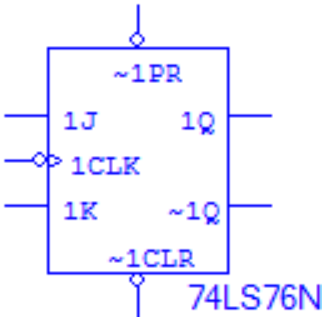
Flip-Flop Vs. Latch

- The primary difference between a D flip-flop and D latch is the EN/CLOCK input.
- The flip-flop's CLOCK input is edge sensitive, meaning the flip-flop's output changes on the edge (rising or falling) of the CLOCK input.
- The latch's EN input is level sensitive, meaning the latch's output changes on the level (high or low) of the EN input.

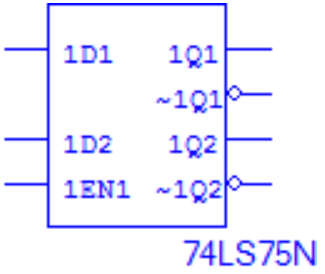
Flip-Flops & Latches



74LS74
Dual Positive-Edge-Triggered D Flip-Flops with Preset, Clear, and Complementary Outputs

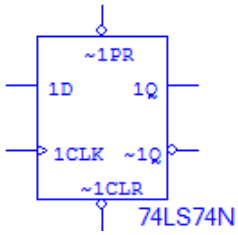


74LS76
Dual Negative-Edge-Triggered J-K Flip-Flops with Preset, Clear, and Complementary Outputs



74LS75
Quad Latch

74LS74: D Flip-Flop



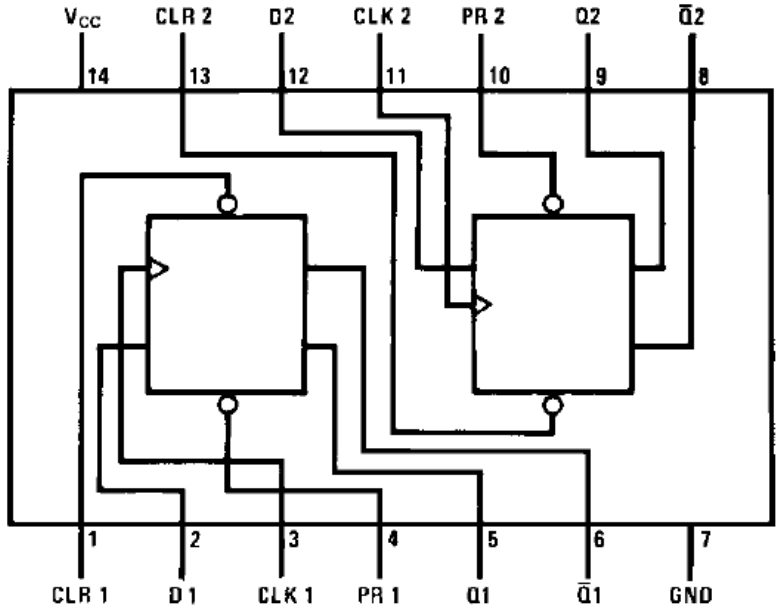
Function Table

Inputs				Outputs	
PR	CLR	CLK	D	Q	$\overline{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H (Note 1)	H (Note 1)
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Q_0	$\overline{Q}_0$

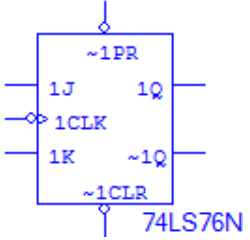
H = HIGH Logic Level
X = Either LOW or HIGH Logic Level
L = LOW Logic Level
↑ = Positive-going Transition
Q_0 = The output logic level of Q before the indicated input conditions were established.

Note 1: This configuration is nonstable; that is, it will not persist when either the preset and/or clear inputs return to their inactive (HIGH) level.

Connection Diagram



74LS76: J/K Flip-Flop



Function Table

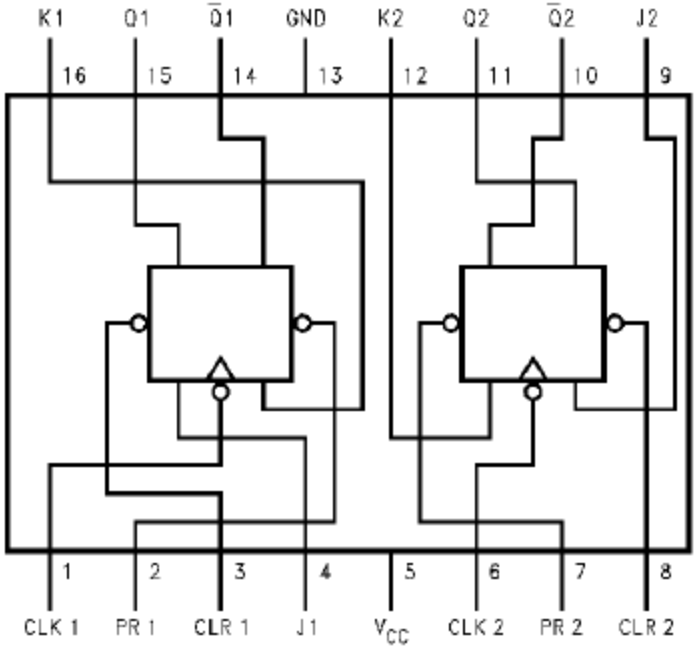
Inputs					Outputs	
PR	CLR	CLK	J	K	Q	$\overline{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H (Note 1)	H (Note 1)
H	H	⎍	L	L	Q_0	$\overline{Q}_0$
H	H	⎍	H	L	H	L
H	H	⎍	L	H	L	H
H	H	⎍	H	H	Toggle	

H = High Logic Level
L = Low Logic Level
X = Either Low or High Logic Level
⎍ = Positive pulse data. The J and K inputs must be held constant while the clock is high. Data is transfered to the outputs on the falling edge of the clock pulse.
Q_0 = The output logic level before the indicated input conditions were established.
Toggle = Each output changes to the complement of its previous level on each complete active high level clock pulse.

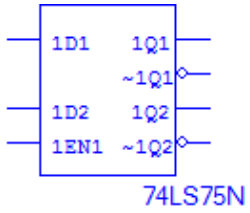
Note 1: This configuration is nonstable; that is, it will not persist when the preset and/or clear inputs return to their inactive (high) level.

Connection Diagram

Dual-In-Line Package



74LS75: D Latch



Function Table (Each Latch)

Inputs		Outputs	
D	Enable	Q	$\overline{Q}$
L	H	L	H
H	H	H	L
X	L	Q_0	$\overline{Q}_0$

H = HIGH Level
L = LOW Level
X = Don't Care
Q_0 = The Level of Q Before the HIGH-to-LOW Transition of ENABLE

Connection Diagram