Latches & Flip-flops

Sequential Logic
Latches & Flip-flops

- Introduction
- 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

Flip-Flops & Latches

- Review sequential logic and the flip-flop.
- Introduce the D flip-flop and provide an excitation table and a sample timing analysis.
- Introduce the J/K flip-flop and provide an excitation table and a sample timing analysis.
- Review flip-flop clock parameters.
- Introduce the transparent D-latch.
- Discuss flip-flop asynchronous inputs.

Flip-Flops & Latches

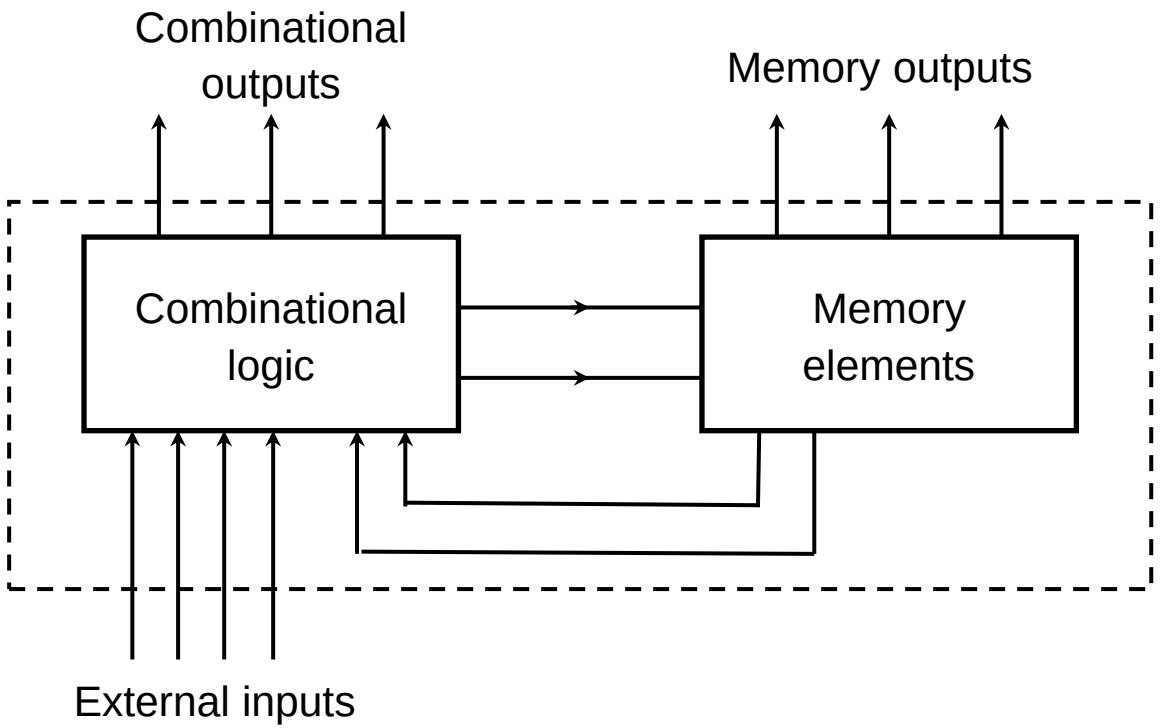
- Sequential switching network
  - Output depends on present input and past sequence of inputs.
  - Need to remember past history.
  - Flip-flop (latch) is a memory that has a pair of complementary outputs.

Sequential Circuits

- Give sequence to events
- Have memory (short-term)
- Use feedback from output to input to store information

Introduction

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



Sequential circuit = Combinational logic + Memory Elements

Introduction

- 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*.

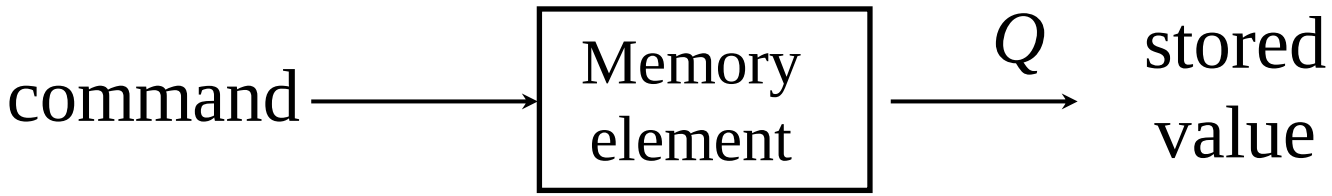
- Latches and flip-flops differ in the method used for changing their state.

Introduction

- Outputs of sequential logic depend on current *and* prior input values – it has ***memory***.
- Some definitions:
  - **State:** all the information about a circuit necessary to explain its future behavior
  - **Latches and flip-flops:** state elements that store one bit of state
  - **Synchronous sequential circuits:** combinational logic followed by a bank of flip-flops

Memory Elements

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



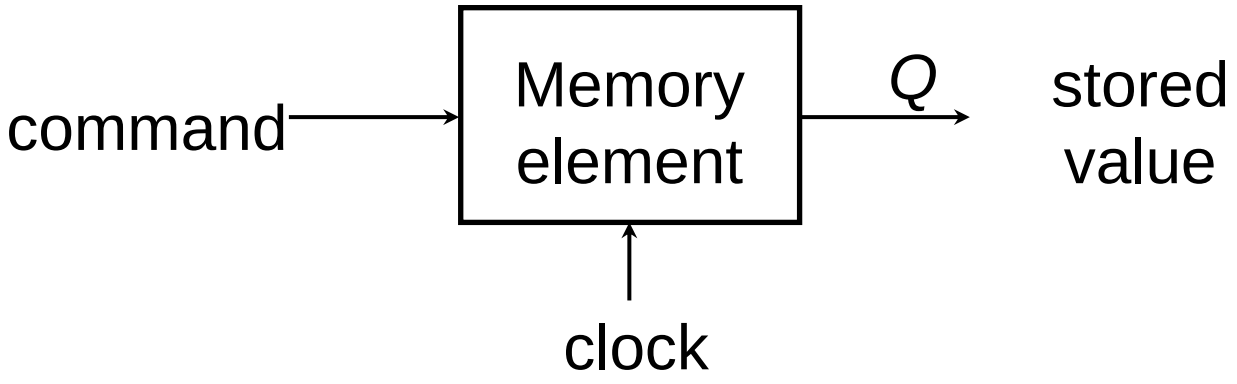
- Characteristic table:

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

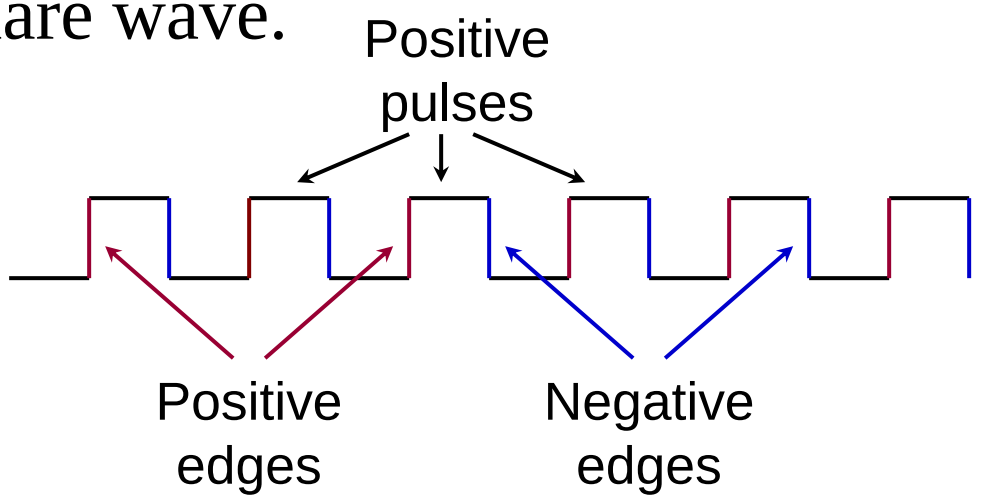
Q(t): current state
Q(t+1) or Q^+: next state

Memory Elements

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



- Clock is usually a square wave.



Memory Elements

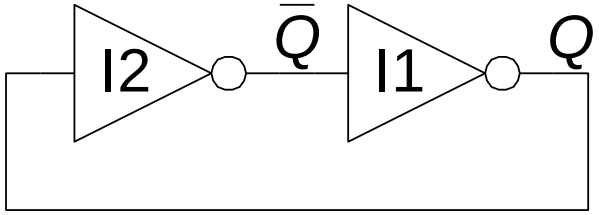
- Two types of triggering/activation:
  - ❖pulse-triggered
  - ❖edge-triggered
- Pulse-triggered
  - ❖latches
  - ❖ON = 1, OFF = 0
- Edge-triggered
  - ❖flip-flops
  - ❖positive edge-triggered (ON = from 0 to 1; OFF = other time)
  - ❖negative edge-triggered (ON = from 1 to 0; OFF = other time)

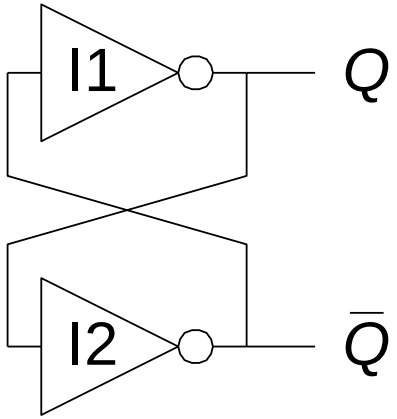
State Elements

- The state of a circuit influences its future behavior
- State elements store state
  - Bistable circuit
  - SR Latch
  - D Latch
  - D Flip-flop

Bistable Circuit

- Fundamental building block of other state elements
- Two outputs: Q, $\overline{Q}$
- No inputs

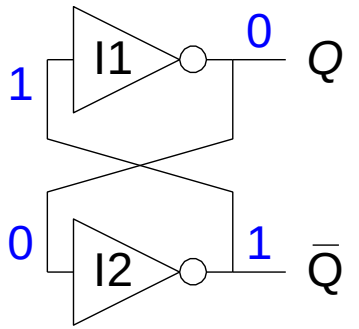




Bistable Circuit Analysis

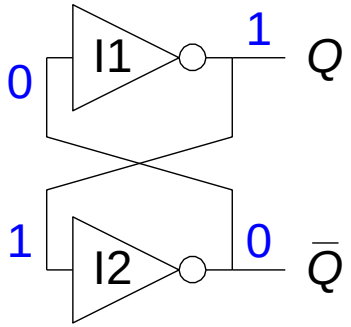
- Consider the two possible cases:
  - **Q = 0:**

    then Q = 1, $\bar{Q}$ = 0 (consistent)



  - **Q = 1:**

    then Q = 0, $\bar{Q}$ = 1 (consistent)



- Stores 1 bit of state in the state variable, Q (or $\bar{Q}$)
- But there are **no inputs to control the state**

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) a RESET state

  S=HIGH (and *R*=LOW) a SET state

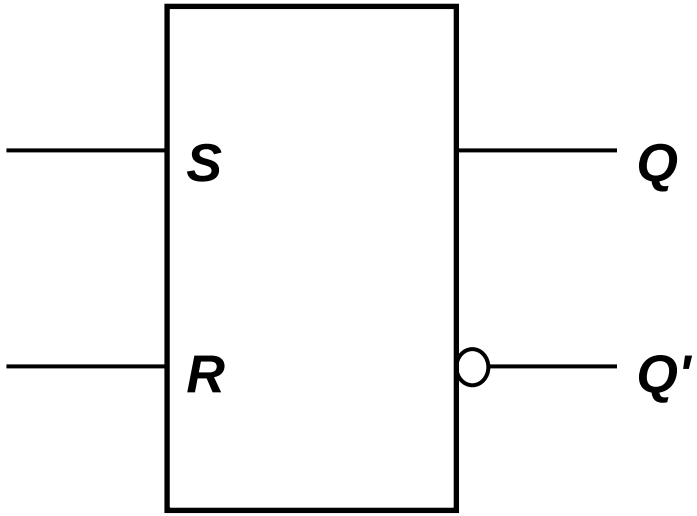
  both inputs LOW a no change

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

S-R Latch

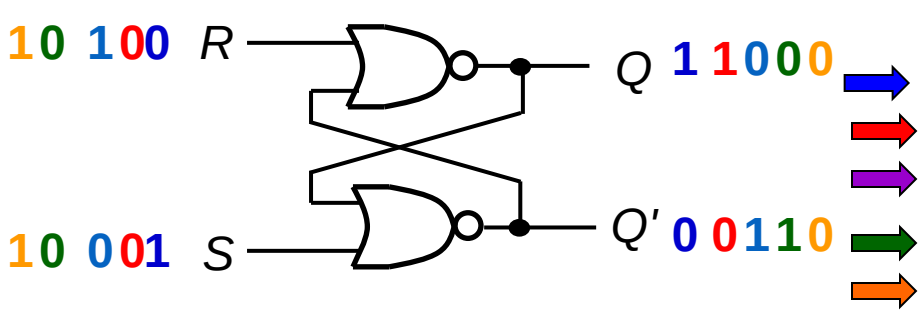
- Characteristics table for active-high input S-R latch:

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

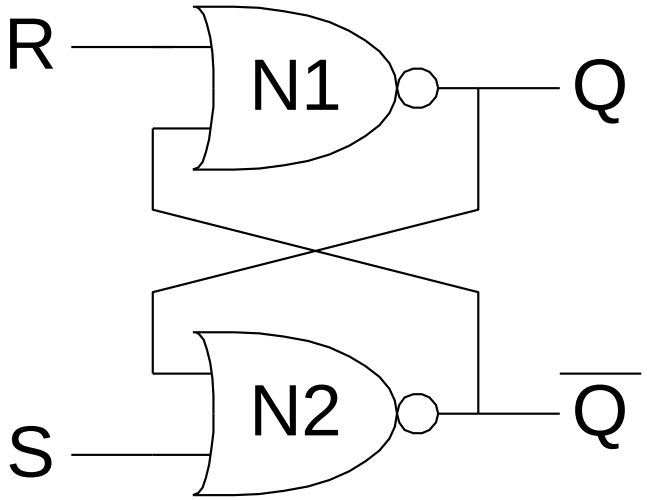
- Active-HIGH input S-R latch



S	R	Q	Q′	
1	0	1	0	initial
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	invalid!

SR (Set/Reset) Latch

- SR Latch

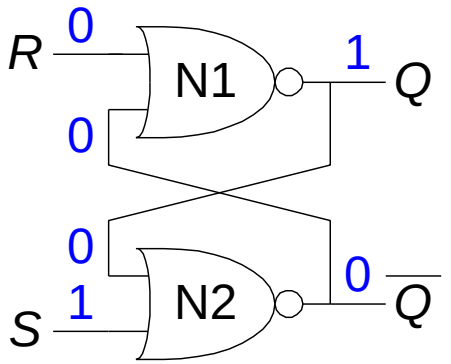


- Consider the four possible cases:
  - $S = 1, R = 0$
  - $S = 0, R = 1$
  - $S = 0, R = 0$
  - $S = 1, R = 1$

SR Latch Analysis

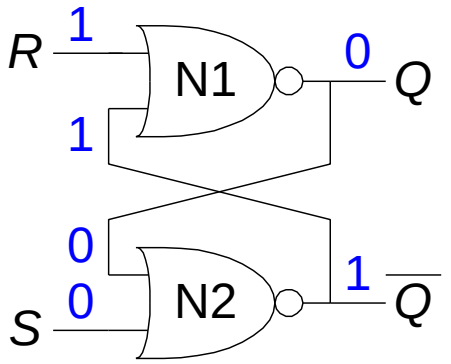
– **$S = 1, R = 0$:**

then $Q = 1$ and $\overline{Q} = 0$



– **$S = 0, R = 1$:**

then $\overline{Q} = 1$ and $Q = 0$

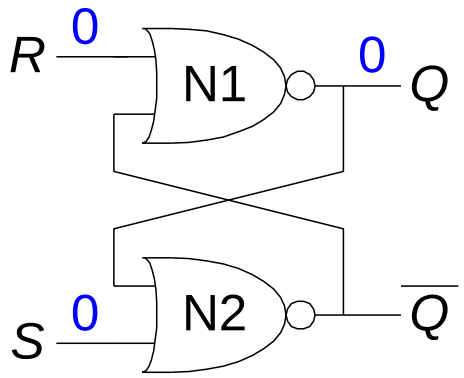


SR Latch Analysis

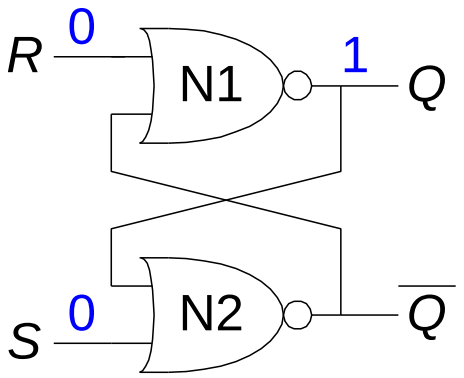
– **$S = 0, R = 0$:**

then $Q = Q_{prev}$

$Q_{prev} = 0$

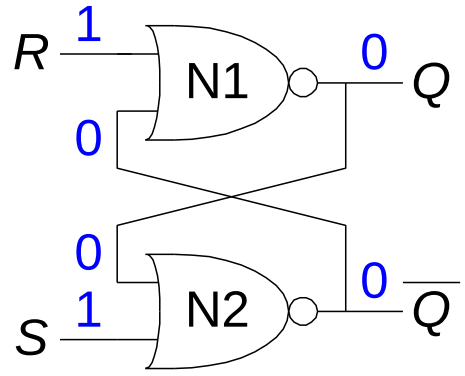


$Q_{prev} = 1$



– **$S = 1, R = 1$:**

then $Q = 0, \bar{Q} = 0$



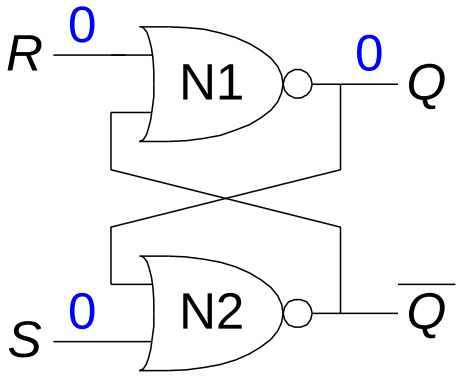
SR Latch Analysis

– $S = 0, R = 0$:

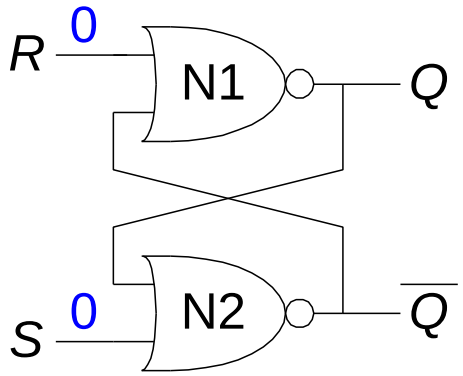
then $Q = Q_{prev}$

– **Memory!**

$Q_{prev} = 0$



$Q_{prev} = 1$

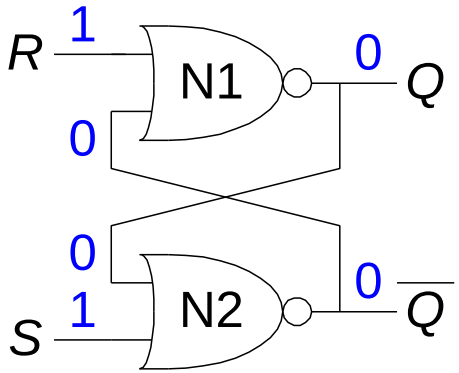


– $S = 1, R = 1$:

then $Q = 0, \bar{Q} = 0$

– **Invalid State**

$\bar{Q} \neq$ NOT Q



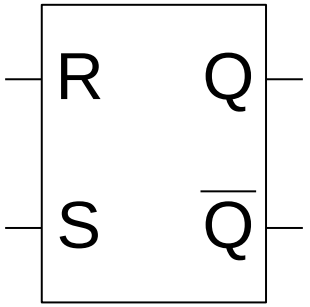
SR Latch Symbol

- SR stands for Set/Reset Latch
  - Stores one bit of state (Q)
- Control what value is being stored with S, R inputs
  - **Set:** Make the output 1

    ($S = 1, R = 0, Q = \mathbf{1}$)
  - **Reset:** Make the output 0

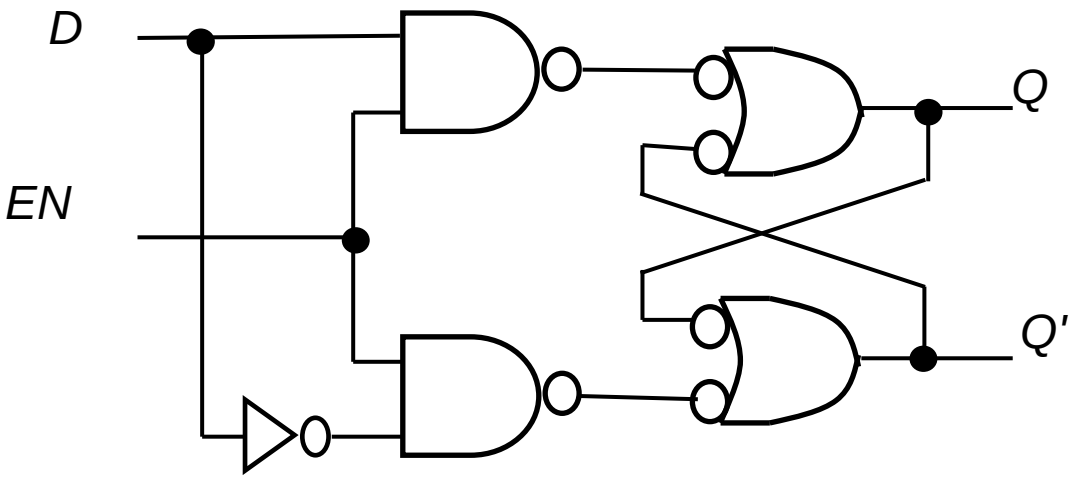
    ($S = 0, R = 1, Q = \mathbf{0}$)
- **Must do something to avoid invalid state (when $S = R = 1$)**

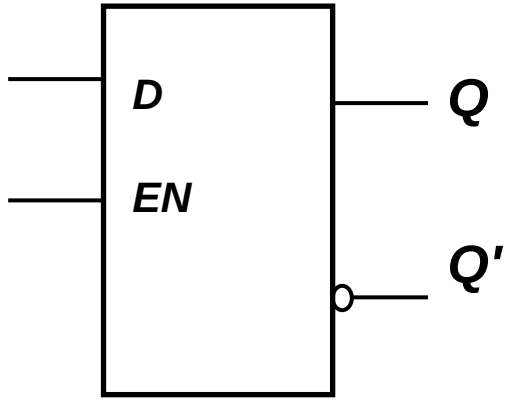
SR Latch Symbol



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.





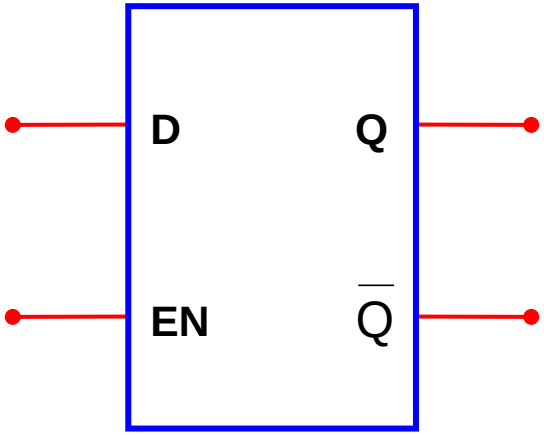
Gated D Latch

- 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*

Transparent D-Latch



EN	D	Q	$\overline{Q}$
0	X	Q_0	$\overline{Q}_0$
1	0	0	1
1	1	1	0

EN: Enable

D Latch

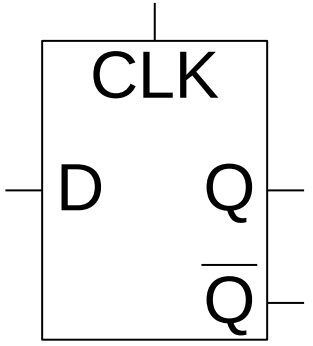
- Two inputs: *CLK*, *D*
  - ***CLK*:** controls *when* the output changes
  - ***D*** (the data input): controls *what* the output changes to
- Function
  - When ***CLK* = 1**,

    D passes through to *Q* (*transparent*)
  - When ***CLK* = 0**,

    Q holds its previous value (*opaque*)
- Avoids invalid case when

  $Q \neq \text{NOT } \overline{Q}$

D Latch Symbol

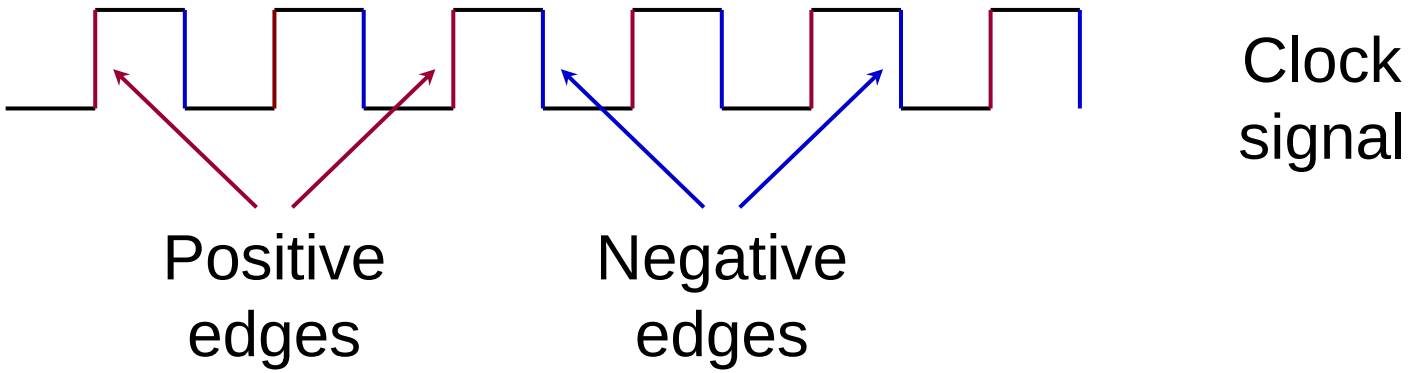


Latch Circuits: Not Suitable

- 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.

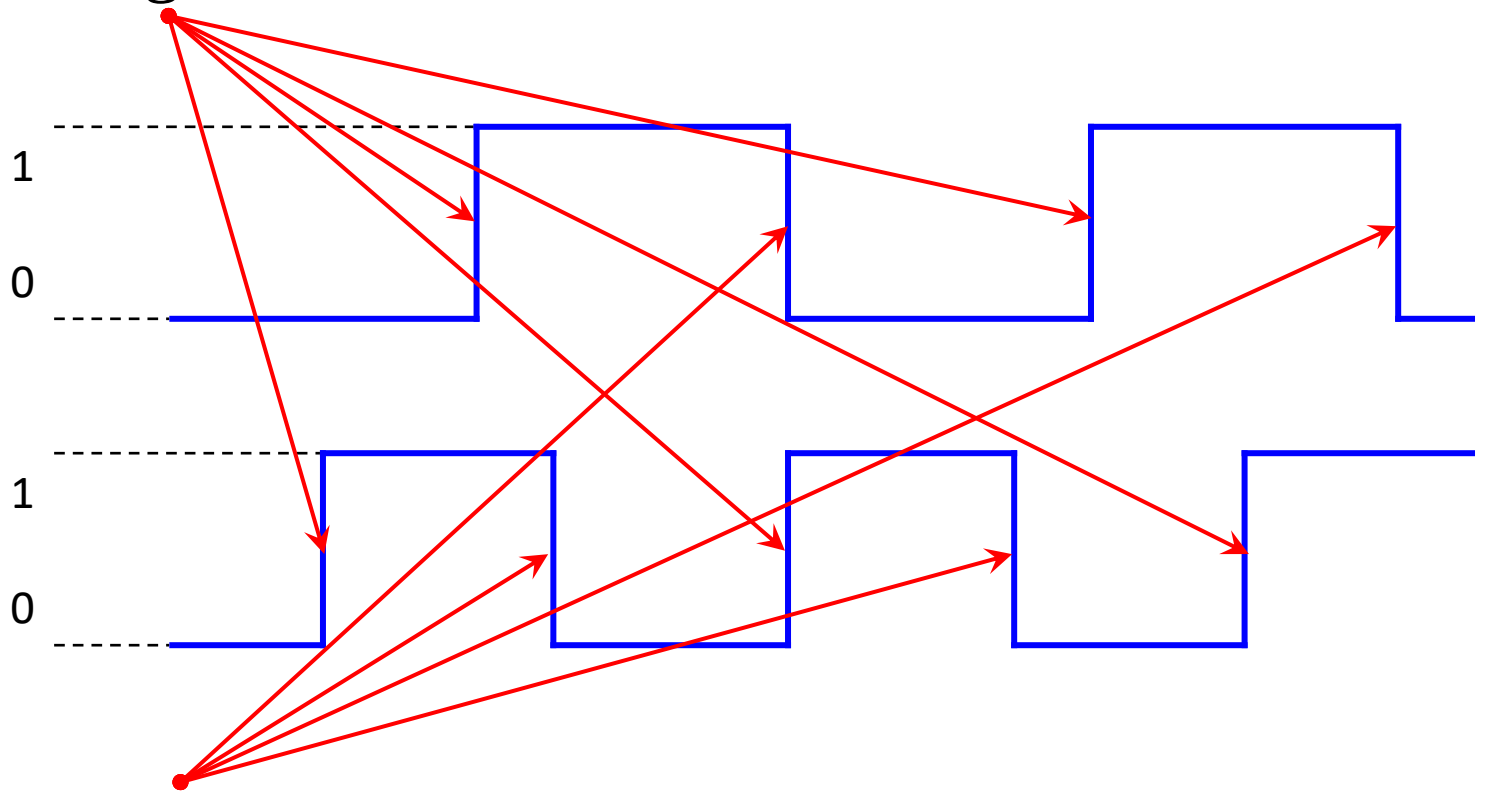
Edge-Triggered Flip-flops

- *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.



Clock Edges

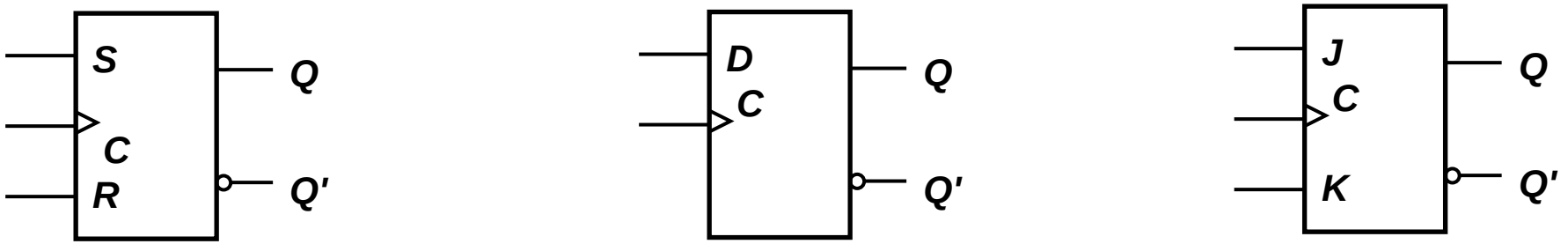
Positive Edge Transition



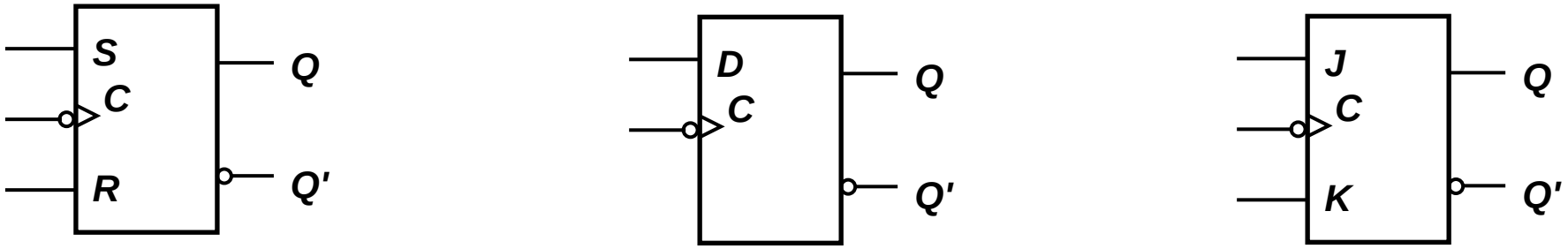
Negative Edge Transition

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) a SET state
  - ❖*R*=HIGH (and *S*=LOW) a RESET state
  - ❖both inputs LOW a no change
  - ❖both inputs HIGH a 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”)

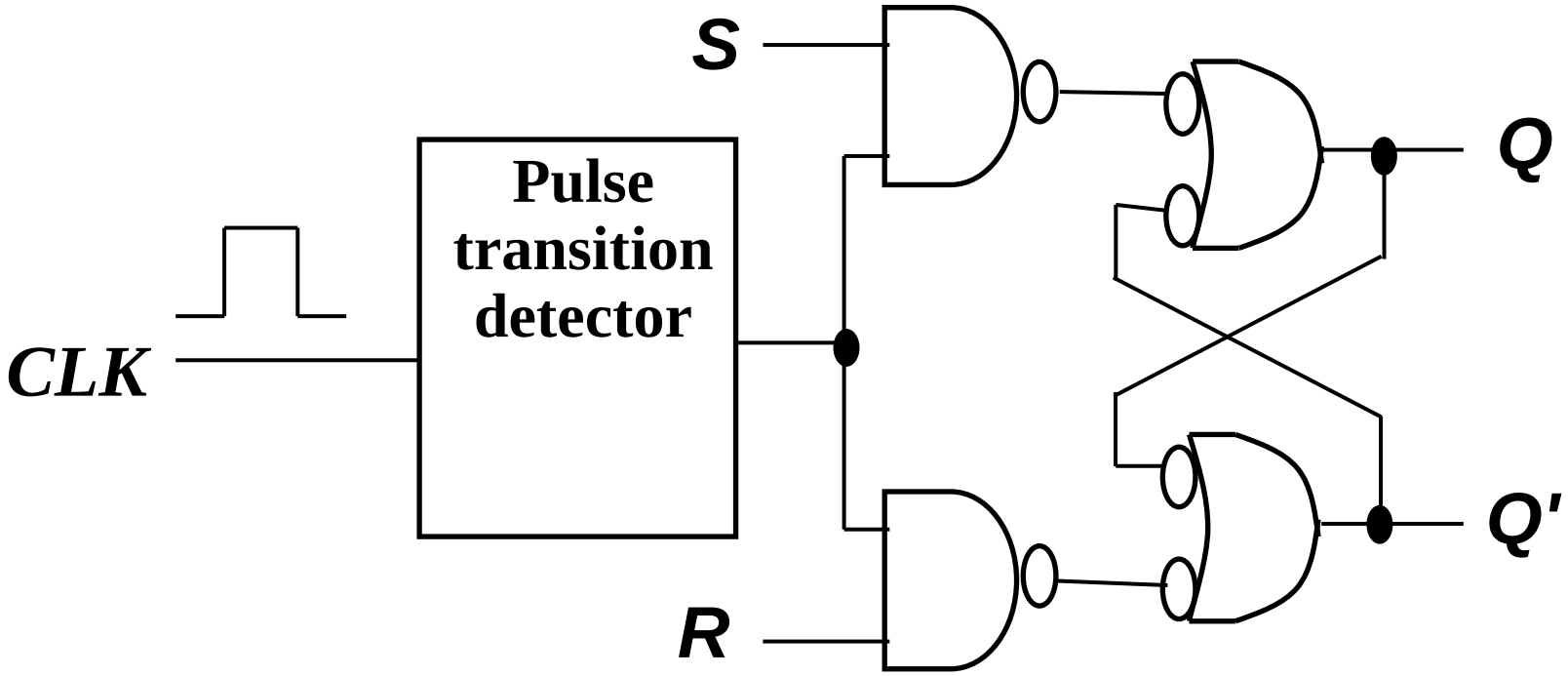
↑ = clock transition LOW to HIGH

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*.

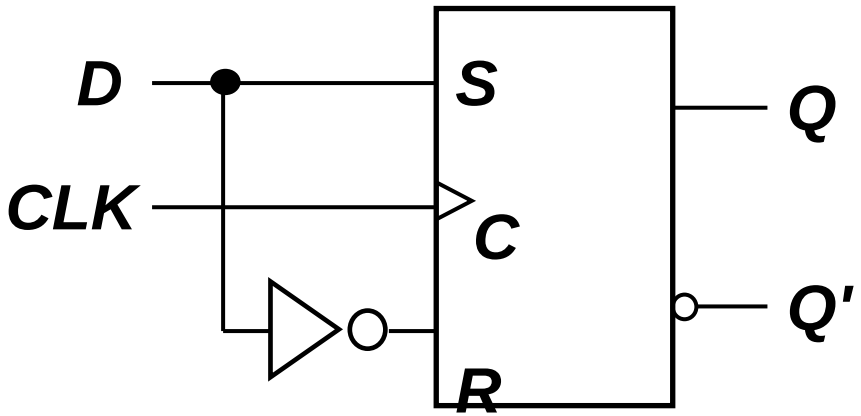
S-R Flip-flop

The pulse transition detector.



D Flip-flop

- D flip-flop: single input *D* (data)
  - ❖*D*=HIGH a SET state
  - ❖*D*=LOW a 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

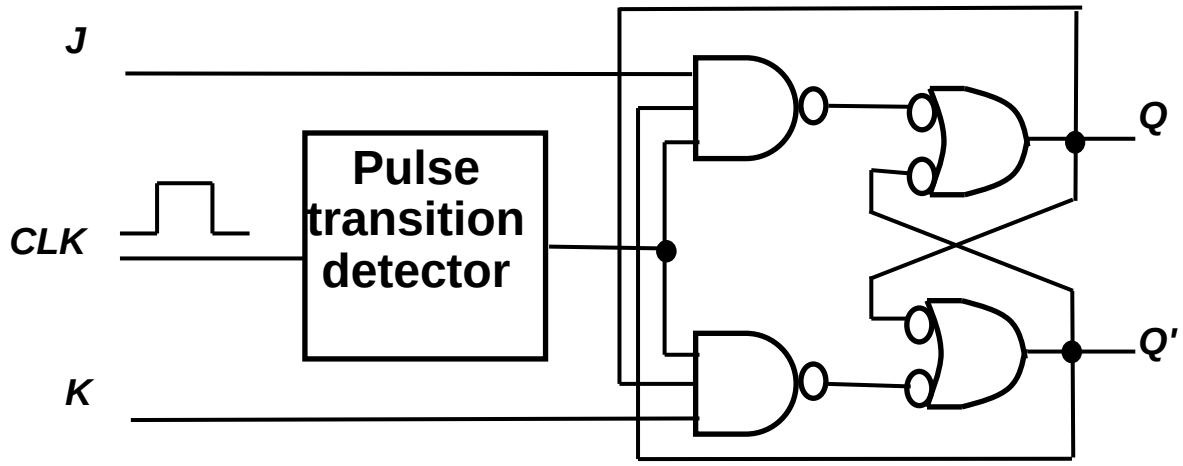
↑ = clock transition LOW to HIGH

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) a SET state
  - *K*=HIGH (and *J*=LOW) a RESET state
  - both inputs LOW a no change
  - both inputs HIGH a toggle

J-K Flip-flop

- J-K flip-flop.



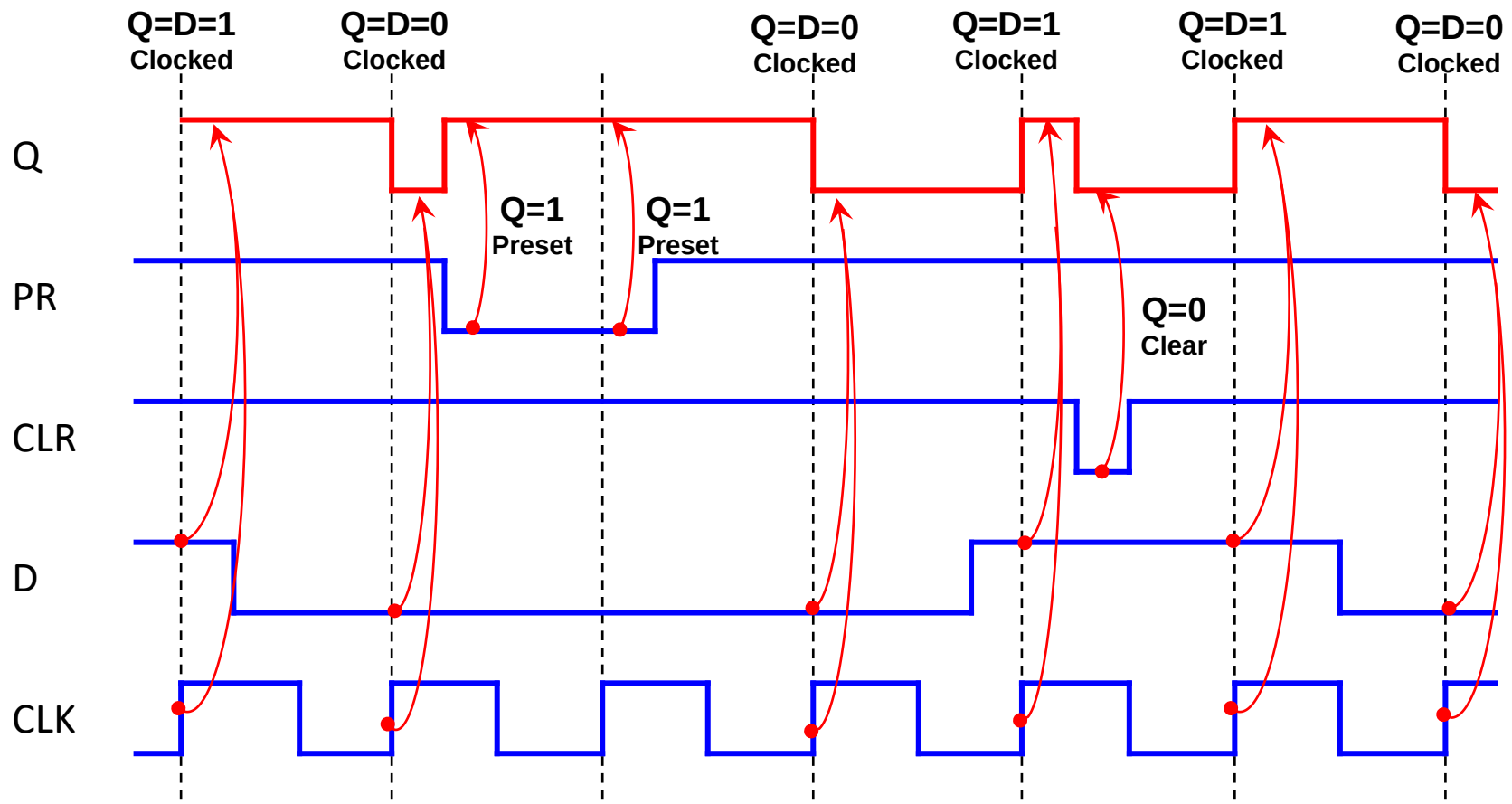
- Characteristic table.

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

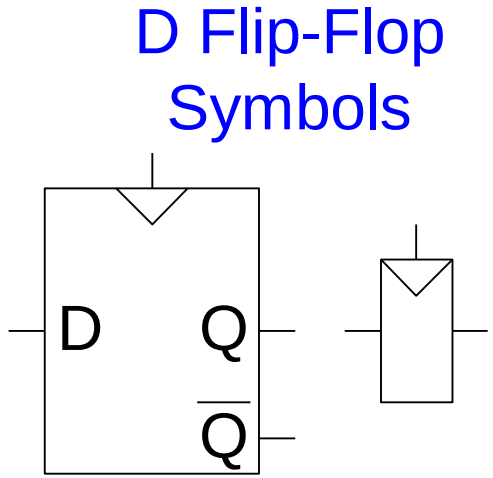
$$Q(t+1) = J.Q' + K'.Q$$

D Flip-Flop: PR & CLR Timing

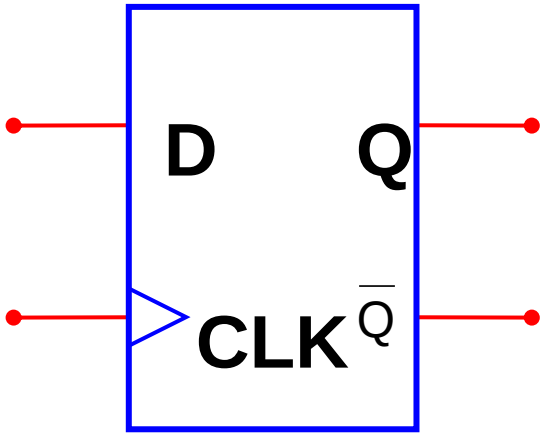


D Flip-Flop

- **Inputs:** *CLK*, *D*
- **Function**
  - Samples *D* on rising edge of *CLK*
    - When *CLK* rises from 0 to 1, *D* passes through to *Q*
    - Otherwise, *Q* holds its previous value
  - *Q* changes only on rising edge of *CLK*
- Called *edge-triggered*
- Activated on the clock edge



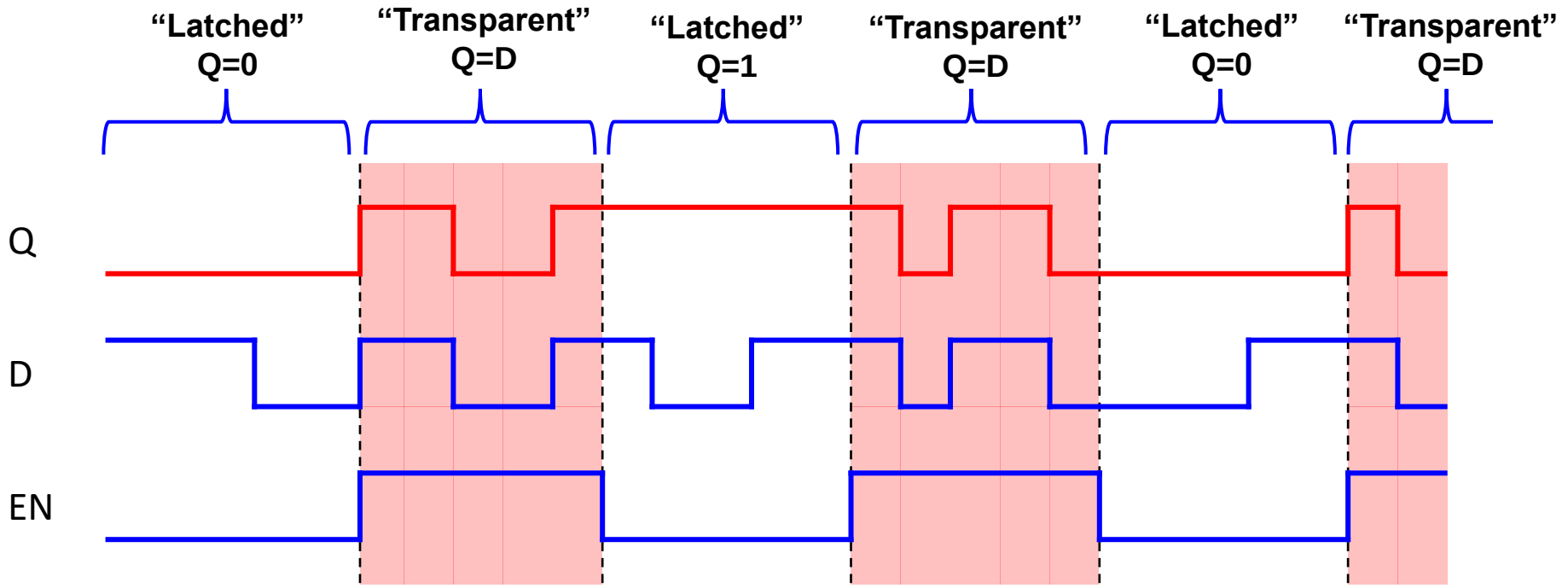
D Flip-Flop: Excitation Table



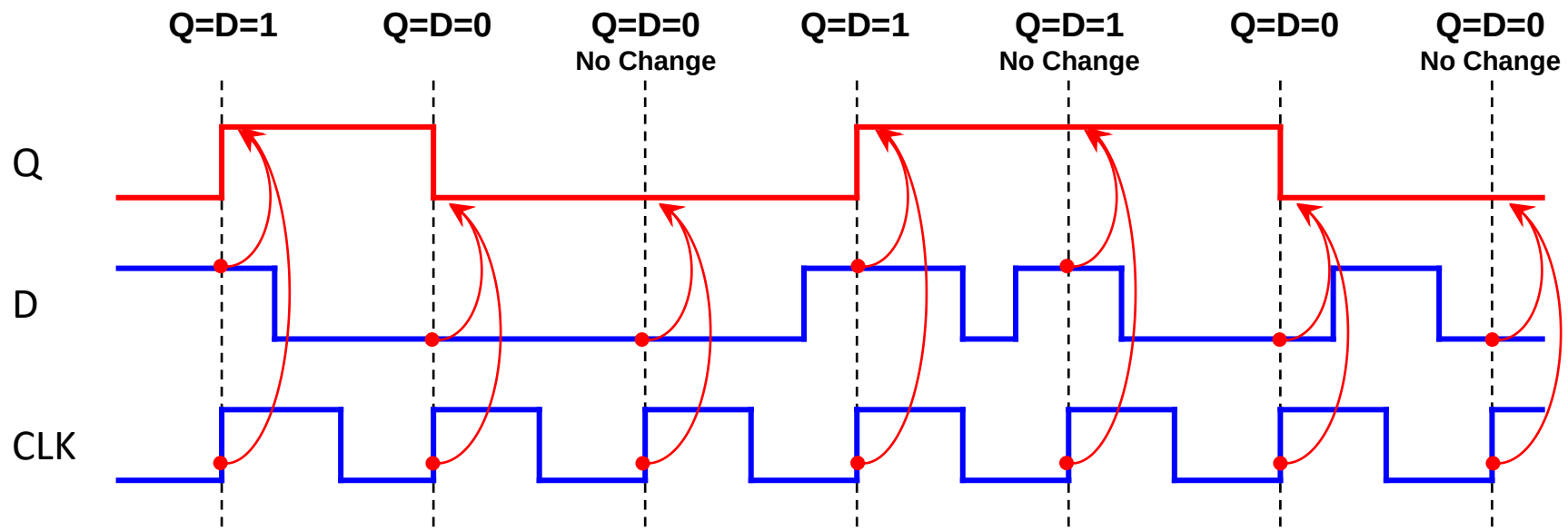
D	CLK	Q	$\overline{Q}$
0	↑	0	1
1	↑	1	0

↑ : Rising Edge of Clock

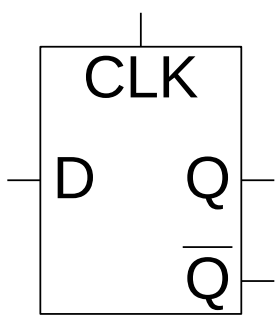
Transparent D-Latch: Example Timing

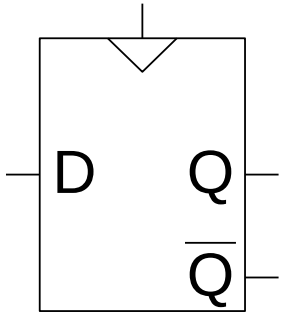


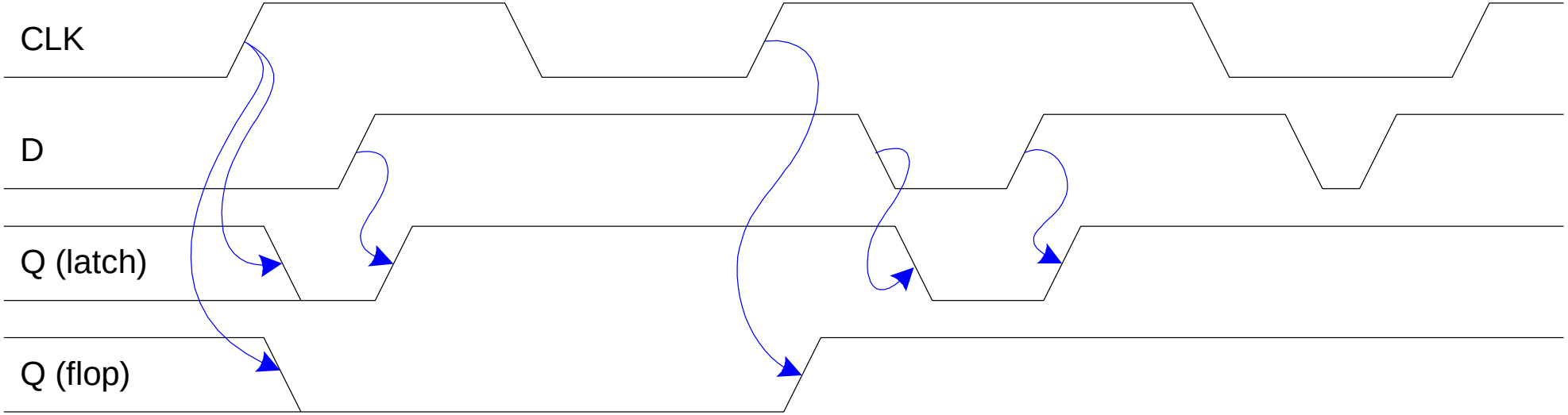
D Flip-Flop: Example Timing



D Latch vs. D Flip-Flop

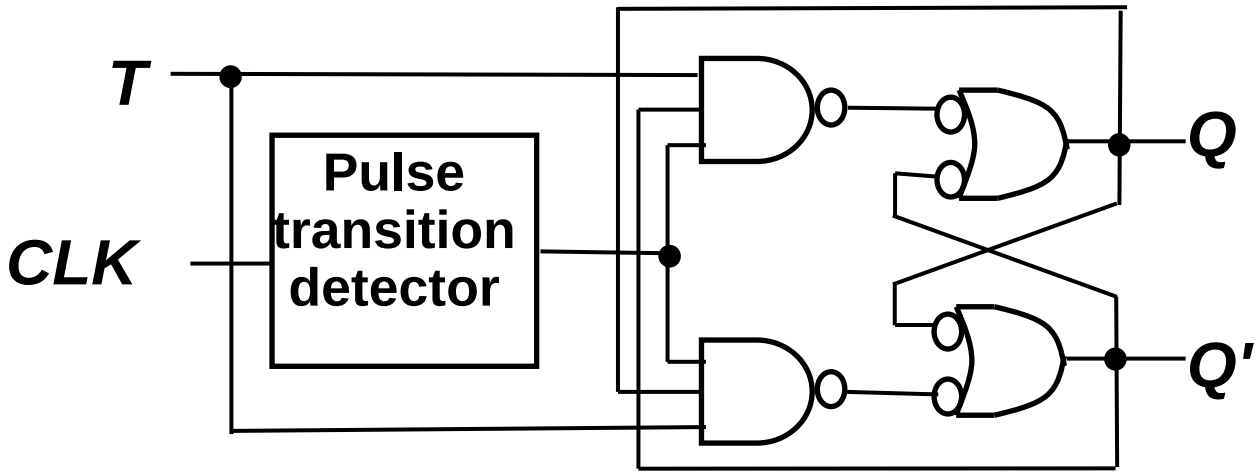


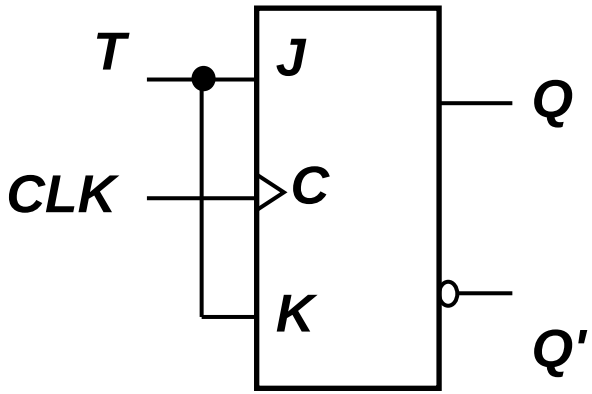




T Flip-flop

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





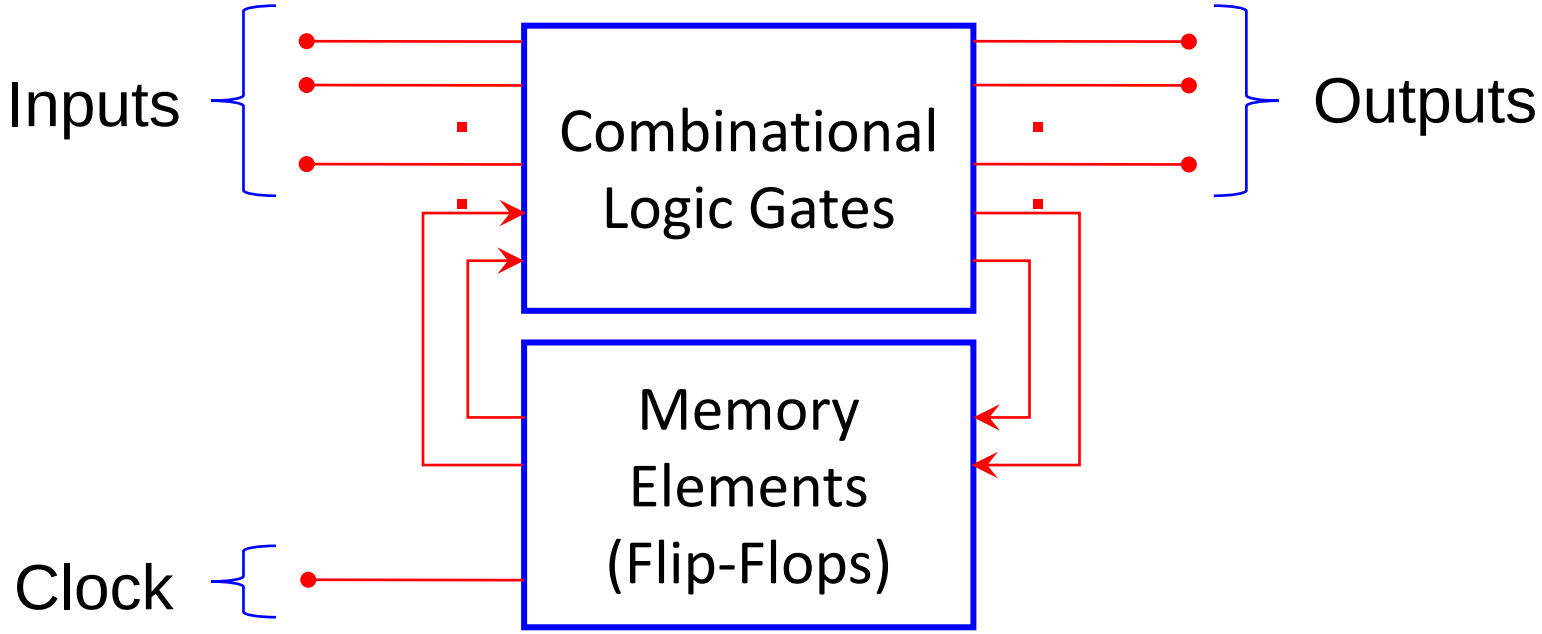
- Characteristic table.

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

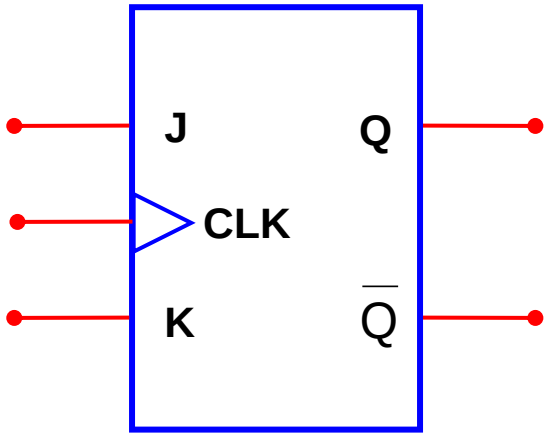
$$Q(t+1) = T.Q' + T'.Q$$

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

Sequential Logic & The Flip-Flop



J/K Flip-Flop: Excitation Table

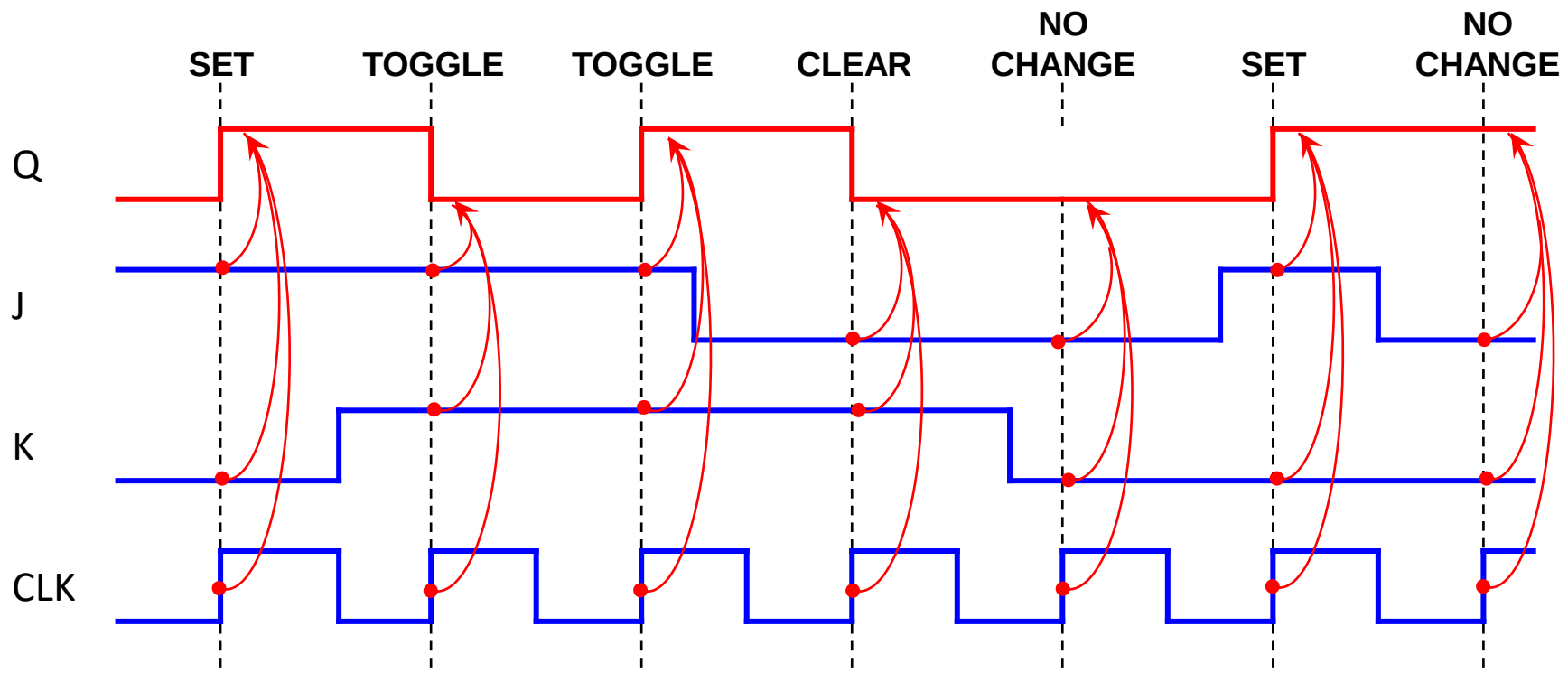


J	K	CLK	Q	
0	0	↑	Q_0	No Change
0	1	↑	0	Clear
1	0	↑	1	Set
1	1	↑	$\overline{Q}_0$	Toggle

↑ : Rising Edge of Clock

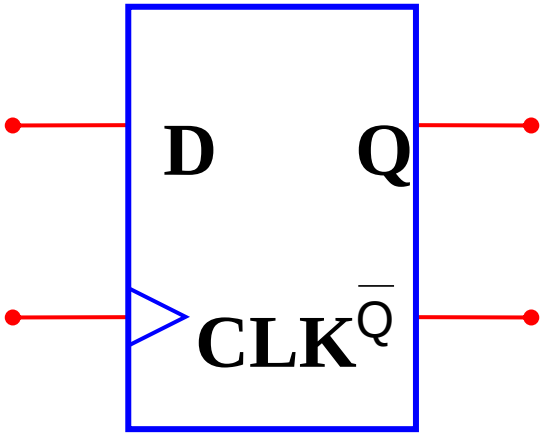
$\overline{Q}$: Complement of Q

J/K Flip-Flop: Example Timing



POS & NEG Edge Triggered D

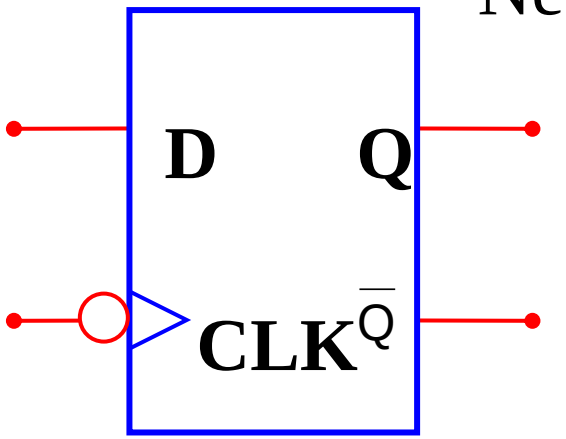
Positive Edge Trigger



D	CLK	Q	$\overline{Q}$
0	↑	0	1
1	↑	1	0

↑ : Rising Edge of Clock

Negative Edge Trigger

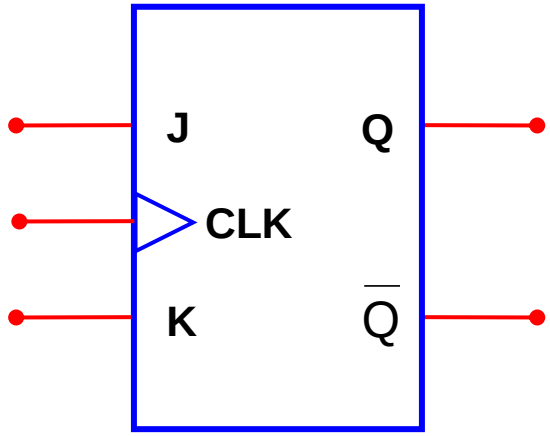


D	CLK	Q	$\overline{Q}$
0	↓	0	1
1	↓	1	0

↓ : Falling Edge of Clock

POS & NEG Edge Triggered J/K

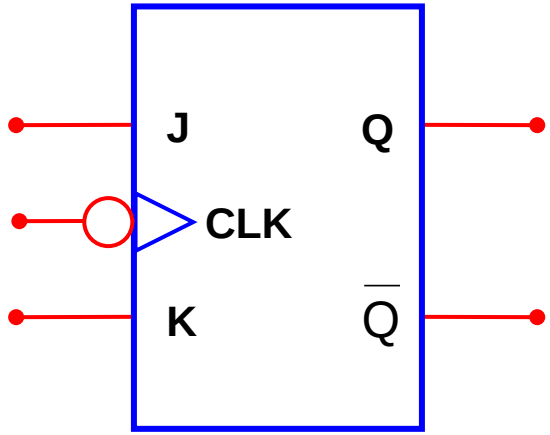
Positive Edge Trigger



J	K	CLK	Q
0	0	↑	Q_0
0	1	↑	0
1	0	↑	1
1	1	↑	$\overline{Q}_0$

↑ : Rising Edge of Clock

Negative Edge Trigger



J	K	CLK	Q
0	0	↓	Q_0
0	1	↓	0
1	0	↓	1
1	1	↓	$\overline{Q}_0$

↓ : Rising Edge of Clock

Asynchronous Inputs

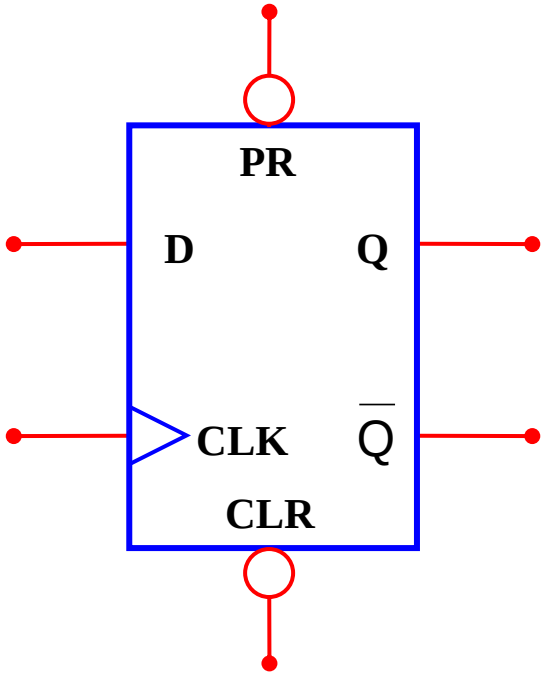
Asynchronous inputs (Preset & Clear) are used to override the clock/data inputs and force the outputs to a predefined state.

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

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.