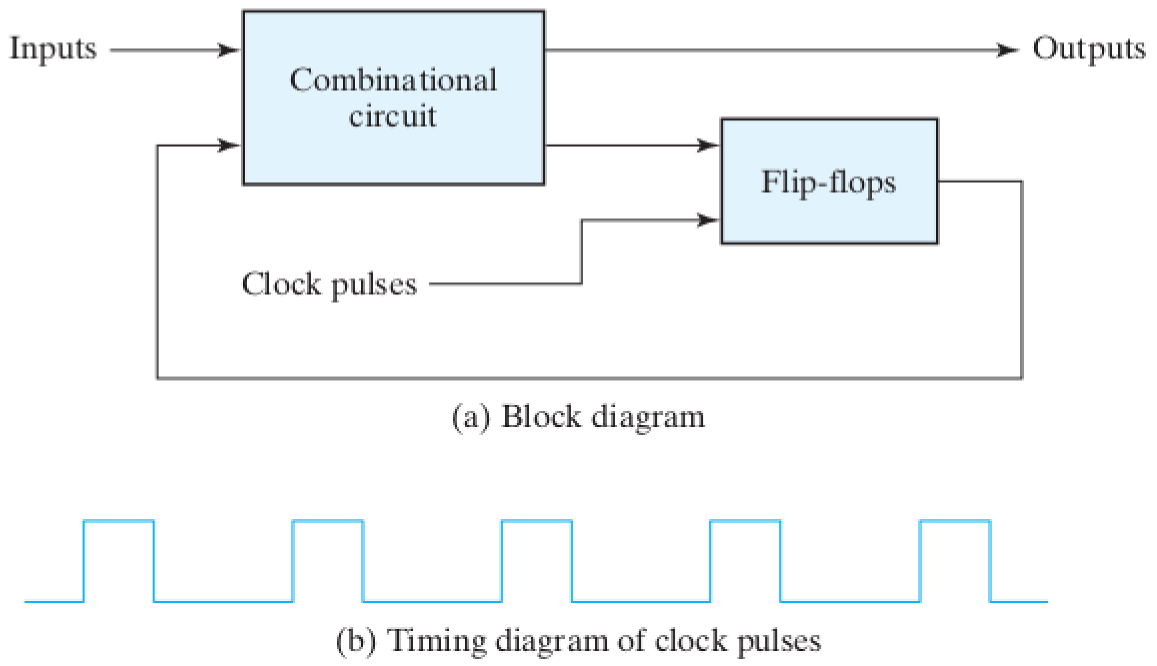


(a) Block diagram

(b) Timing diagram of clock pulses

Synchronous clocked sequential circuit

The outputs are formed by a combinational logic function of the inputs to the circuit or the values stored in the flip-flops (or both).

The value that is stored in a flip-flop when the clock pulse occurs is also determined by the inputs to the circuit or the values presently stored in the flip-flop (or both).

The new value is stored (i.e., the flip-flop is updated) when a pulse of the clock signal occurs. Prior to the occurrence of the clock pulse, the combinational logic forming the next value of the flip-flop must have reached a stable value.

Consequently, the speed at which the combinational logic circuits operate is critical.

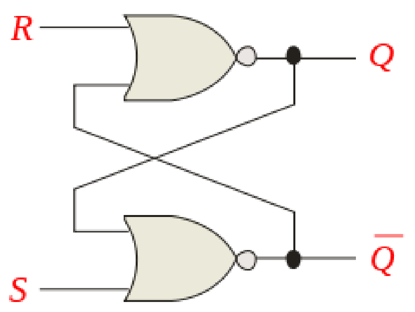
If the clock (synchronizing) pulses arrive at a regular interval, as shown in the timing diagram  the combinational logic must respond to a change in the state of the flip-flop in time to be updated before the next pulse arrives.

Propagation delays play an important role in determining the minimum interval between clock pulses that will allow the circuit to operate correctly.

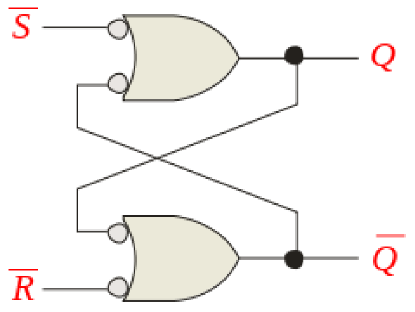
## Latches

A **latch** is a temporary storage device that has two stable states (bistable). It is a basic form of memory.

The S-R (Set-Reset) latch is the most basic type. It can be constructed from NOR gates or NAND gates. With NOR gates, the latch responds to active-HIGH inputs; with NAND gates, it responds to active-LOW inputs.



NOR Active-HIGH Latch



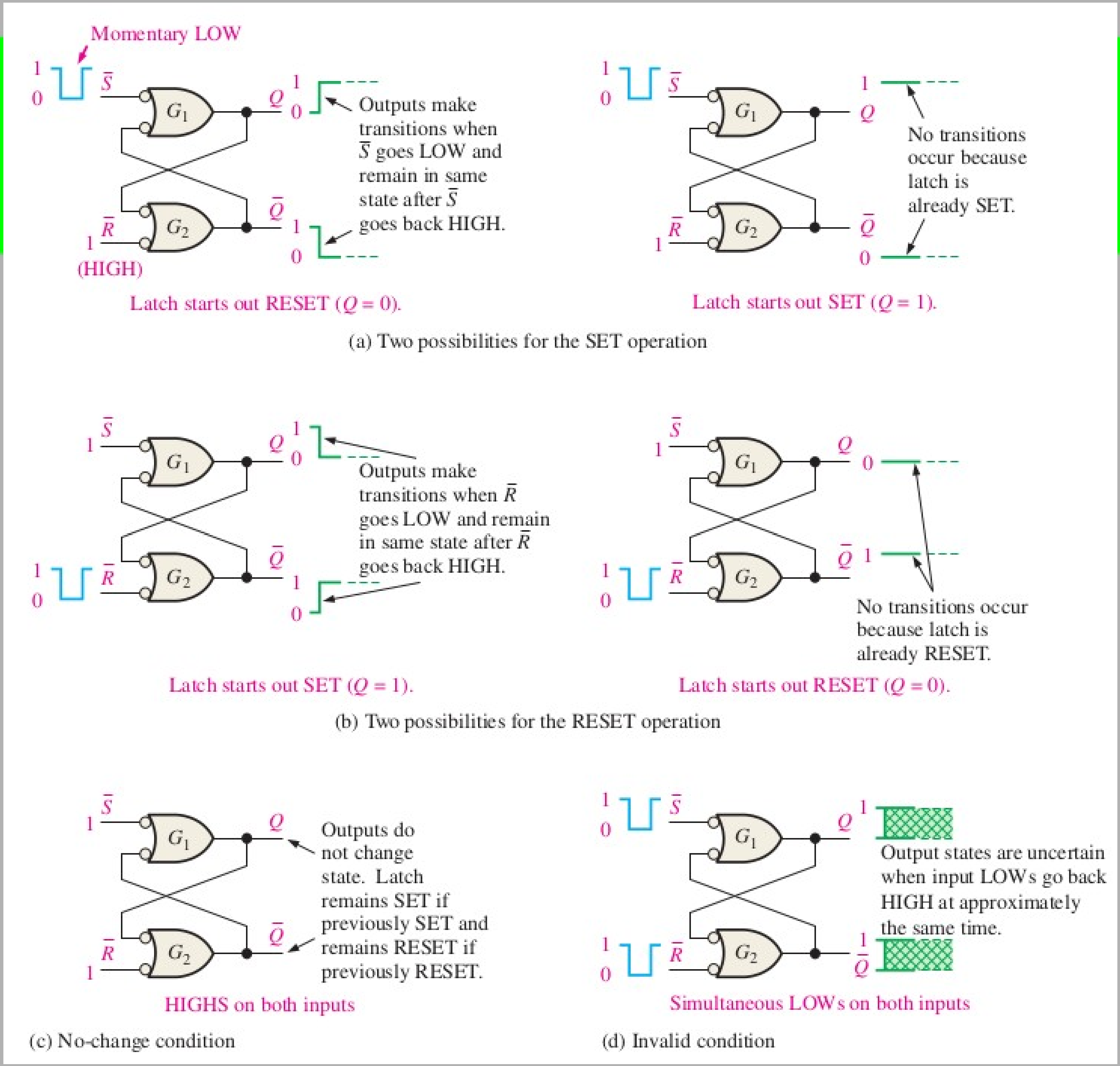
NAND Active-LOW Latch

Storage elements that operate with signal levels (rather than signal transitions) are referred to as latches; those controlled by a clock transition are flip-flops.

Latches are said to be level sensitive devices; flip-flops are edge-sensitive devices.

The two types of storage elements are related because latches are the basic circuits from which all flip-flops are constructed.

Although latches are useful for storing binary information and for the design of asynchronous sequential circuits, they are not practical for use as storage elements in synchronous sequential circuits.


Momentary LOW
Outputs make transitions when $\bar{S}$ goes LOW and remain in same state after $\bar{S}$ goes back HIGH.
(HIGH)
Latch starts out RESET ($Q = 0$).
No transitions occur because latch is already SET.
Latch starts out SET ($Q = 1$).
(a) Two possibilities for the SET operation
Outputs make transitions when $\bar{R}$ goes LOW and remain in same state after $\bar{R}$ goes back HIGH.
Latch starts out SET ($Q = 1$).
No transitions occur because latch is already RESET.
Latch starts out RESET ($Q = 0$).
(b) Two possibilities for the RESET operation
Outputs do not change state. Latch remains SET if previously SET and remains RESET if previously RESET.
HIGHS on both inputs
(c) No-change condition
Output states are uncertain when input LOWs go back HIGH at approximately the same time.
Simultaneous LOWs on both inputs
(d) Invalid condition

When both inputs are low at once, however, there is a problem: it is being told to simultaneously produce a high Q and a low Q.

This produces a "race condition" within the circuit - whichever gate succeeds in changing first will feedback to the other and assert itself. Ideally, both gates are identical and this is "metastable", and the device will be in an undefined state for an indefinite period.

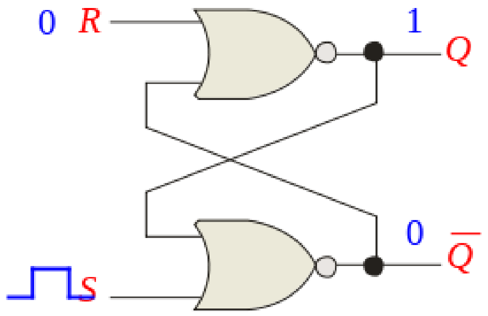
In real life, due to manufacturing methods, one gate will always win, but it's impossible to tell which it will be for a particular device from an assembly line.

The state of S = R = 1 is therefore "illegal" and should never be entered.

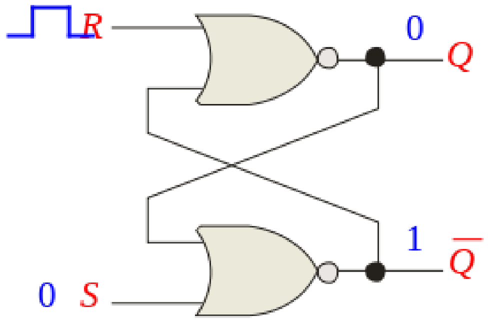
## Latches

The active-HIGH *S*-*R* latch is in a stable (latched) condition when both inputs are LOW.

Assume the latch is initially RESET ($Q = 0$) and the inputs are at their inactive level (0). To SET the latch ($Q = 1$), a momentary HIGH signal is applied to the *S* input while the *R* remains LOW.



To RESET the latch ($Q = 0$), a momentary HIGH signal is applied to the *R* input while the *S* remains LOW.



An invalid condition in the operation of an active-LOW input S-R latch occurs when LOWs are applied to both S and R at the same time.

As long as the LOW levels are simultaneously held on the inputs, both the Q and Q outputs are forced HIGH, thus violating the basic complementary operation of the outputs.

Also, if the LOWs are released simultaneously, both outputs will attempt to go LOW.

Since there is always some small difference in the propagation delay time of the gates, one of the gates will dominate in its transition to the LOW output state.

This, in turn, forces the output of the slower gate to remain HIGH.

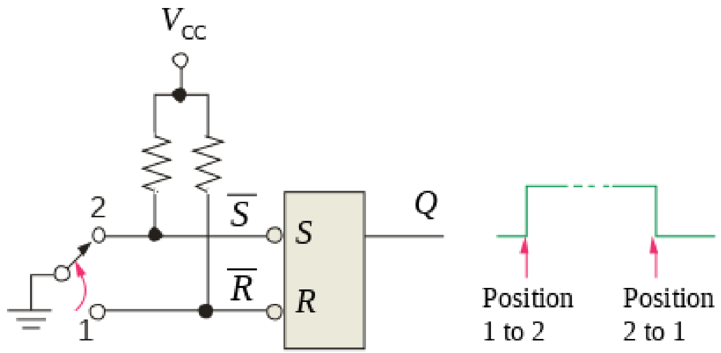
In this situation, you cannot reliably predict the next state of the latch.

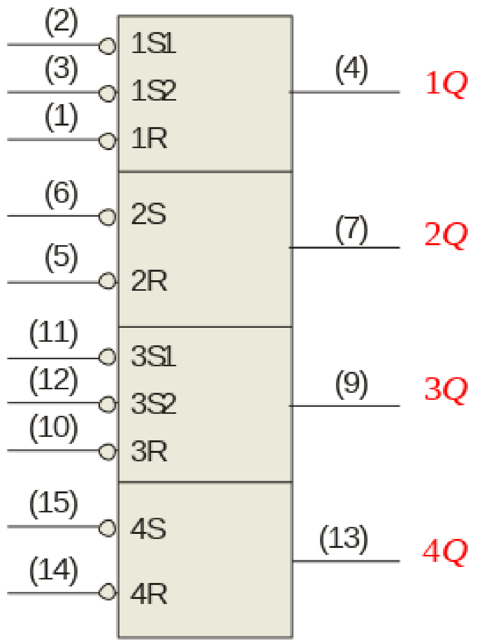
## Latches

The active-LOW $\overline{S}$-$\overline{R}$ latch is available as the 74LS279A IC.

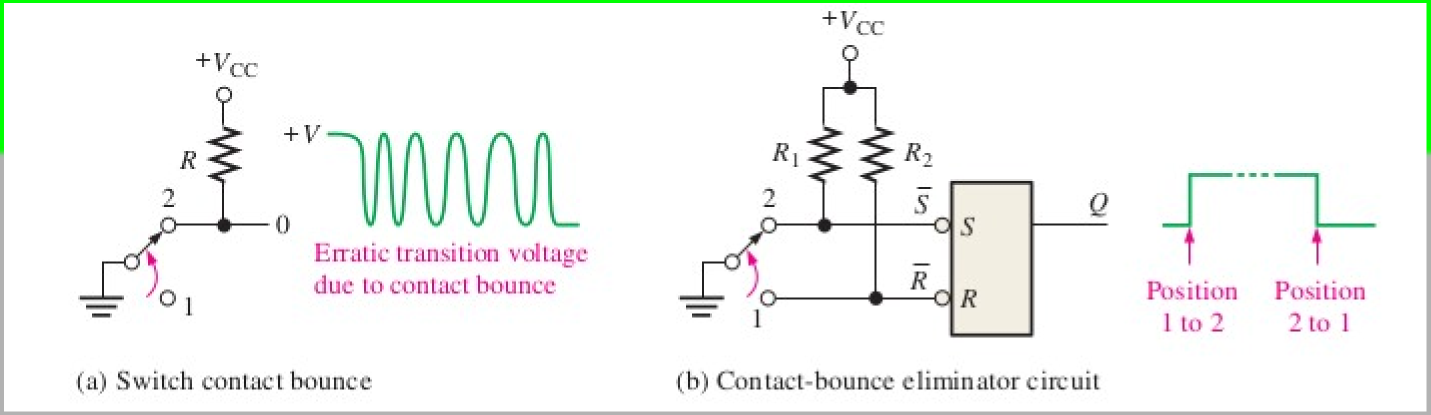
It features four internal latches with two having two $\overline{S}$ inputs. To SET any of the latches, the $\overline{S}$ line is pulsed low. It is available in several packages.

$\overline{S}$-$\overline{R}$ latches are frequently used for switch debounce circuits as shown:





74LS279A



$+V_{CC}$
$R$
2
0
1
$+V$
Erratic transition voltage due to contact bounce
(a) Switch contact bounce
$+V_{CC}$
$R_1$
$R_2$
2
1
$\overline{S}$
$\overline{R}$
$S$
$R$
$Q$
Position 1 to 2
Position 2 to 1
(b) Contact-bounce eliminator circuit

An S-R latch can be used to eliminate the effects of switch bounce.

The switch is normally in position 1, keeping the R input LOW and the latch RESET. When the switch is thrown to position 2, R goes HIGH because of the pull-up resistor to VCC, and S goes LOW on the first contact. Although S remains LOW for only a very short time before the switch bounces, this is sufficient to set the latch. Any further voltage spikes on the S input due to switch bounce do not affect the latch, and it remains SET.

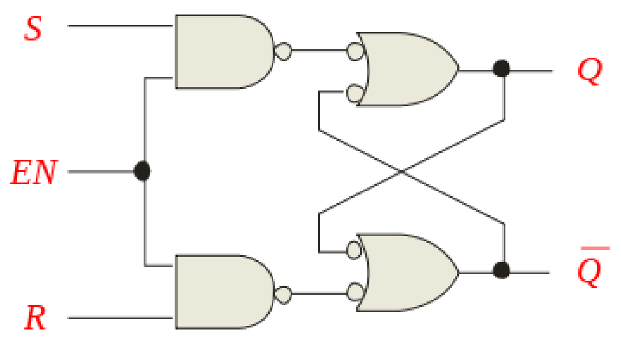
Notice that the Q output of the latch provides a clean transition from LOW to HIGH, thus eliminating the voltage spikes caused by contact bounce.

Similarly, a clean transition from HIGH to LOW is made when the switch is thrown back to position 1.

## Latches

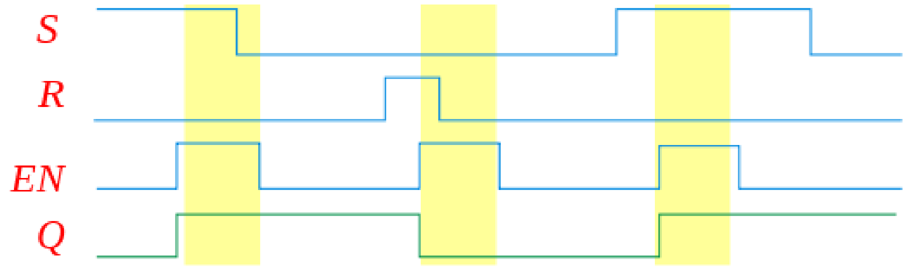
A gated latch is a variation on the basic latch.

The gated latch has an additional input, called enable (*EN*) that must be HIGH in order for the latch to respond to the *S* and *R* inputs.



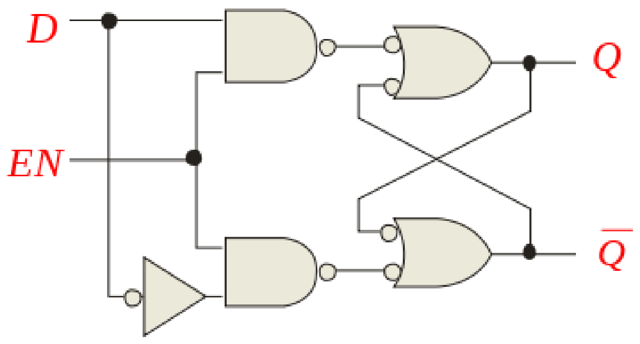
**Example** Show the *Q* output with relation to the input signals. Assume *Q* starts LOW.

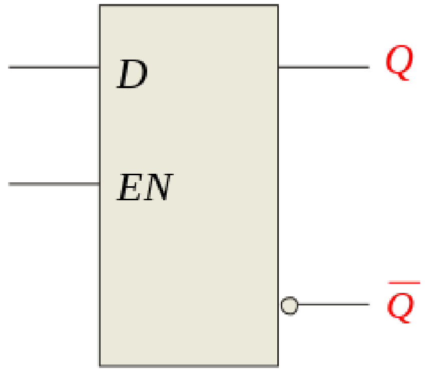
**Solution** Keep in mind that *S* and *R* are only active when *EN* is HIGH.



## Latches

The *D* latch is an variation of the *S-R* latch but combines the *S* and *R* inputs into a single *D* input as shown:



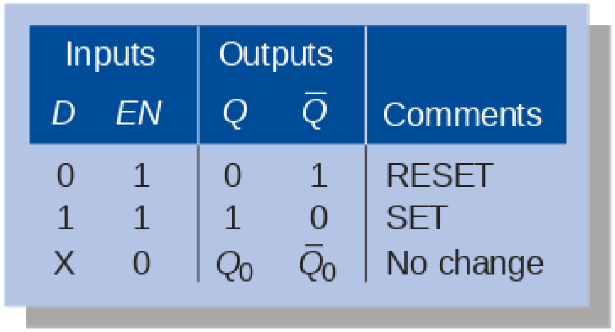


A simple rule for the *D* latch is:

*Q* follows *D* when the Enable is active.

## Latches

The truth table for the *D* latch summarizes its operation. If *EN* is LOW, then there is no change in the output and it is latched.

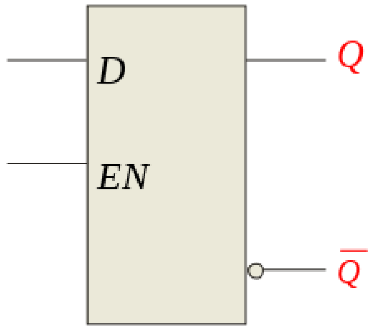


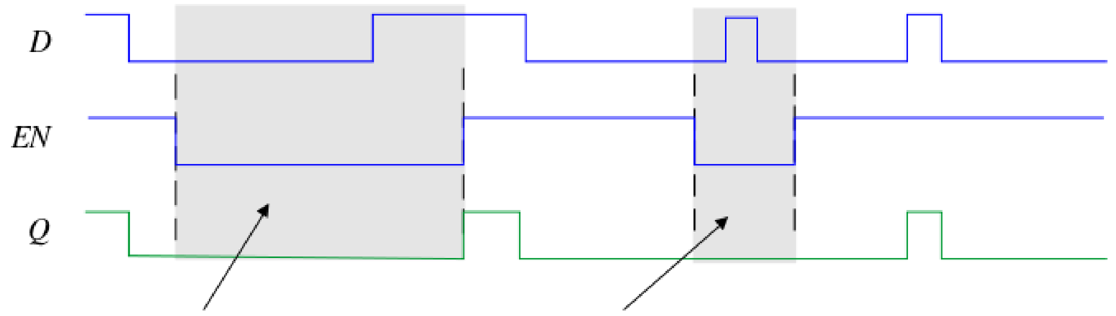
| Inputs | | Outputs | | |
|---|---|---|---|---|
| $D$ | $EN$ | $Q$ | $\overline{Q}$ | Comments |
| 0 | 1 | 0 | 1 | RESET |
| 1 | 1 | 1 | 0 | SET |
| X | 0 | $Q_0$ | $\overline{Q}_0$ | No change |

Latches

Example

Determine the $Q$ output for the $D$ latch, given the inputs shown.





Notice that the Enable is not active during these times, so the output is latched.

The issue with level triggering is that while the clock level is high, inputs change the outputs. In circuits that have feedback (the outputs are connected back to the inputs) level triggering causes chaos, because the level is wide enough (half a clock cycle) that the output can feed back to the inputs within the same period.

So by the time the well-defined moment occurs when the clock falls and every device is supposed to snapshot and hold it state until the next level, chaos has already occurred and the circuits are in unpredictable states. This is unacceptable.

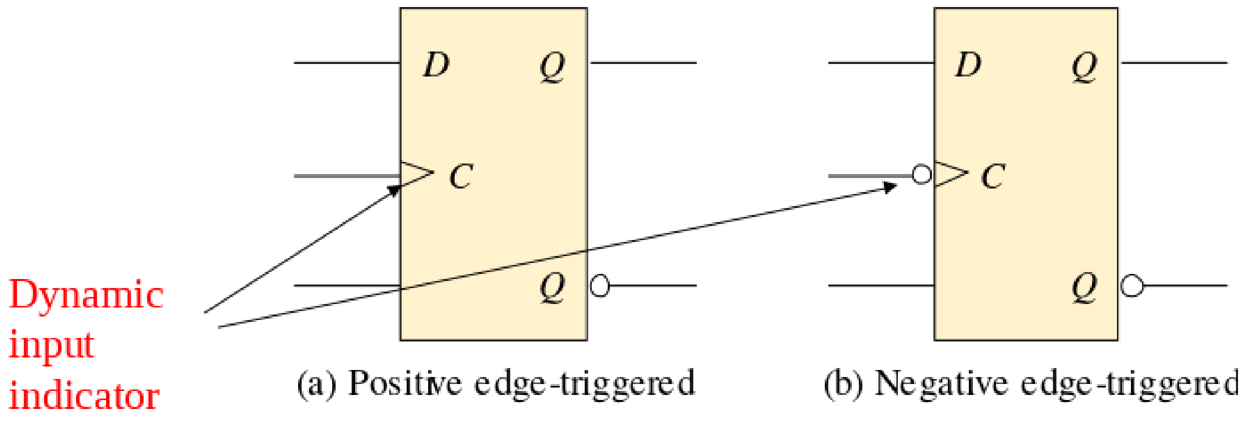
In sequential circuits, we want the outputs produced in clock period t
to only come into consideration for computing the states of clock period t+1.

## Flip-flops are edge-triggered or edge-sensitive whereas gated latches are level-sensitive

### Flip-flops

A flip-flop differs from a latch in the manner it changes states. A flip-flop is a clocked device, in which only the clock edge determines when a new bit is entered.

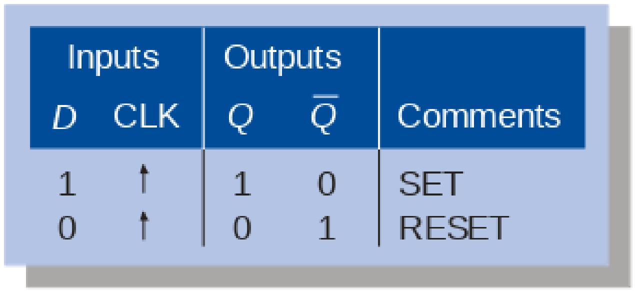
The active edge can be positive or negative.



(a) Positive edge-triggered (b) Negative edge-triggered

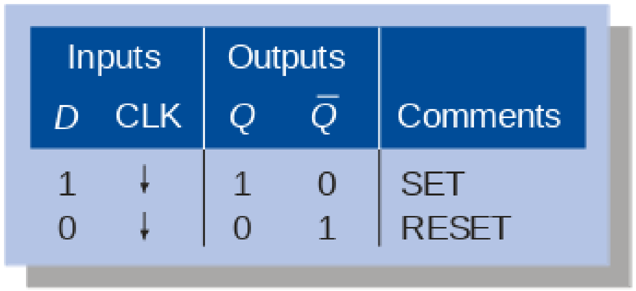
## Flip-flops

The truth table for a positive-edge triggered D flip-flop shows an up arrow to remind you that it is sensitive to its *D* input only on the rising edge of the clock; otherwise it is latched. The truth table for a negative-edge triggered D flip-flop is identical except for the direction of the arrow.



| Inputs | | Outputs | | |
|---|---|---|---|---|
| *D* | CLK | *Q* | $\overline{Q}$ | Comments |
| 1 | ↑ | 1 | 0 | SET |
| 0 | ↑ | 0 | 1 | RESET |

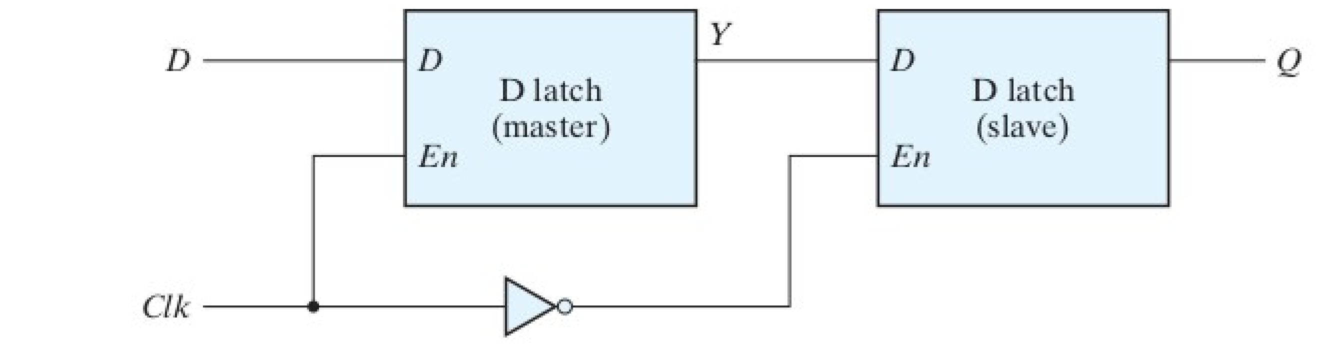
(a) Positive-edge triggered



| Inputs | | Outputs | | |
|---|---|---|---|---|
| *D* | CLK | *Q* | $\overline{Q}$ | Comments |
| 1 | ↓ | 1 | 0 | SET |
| 0 | ↓ | 0 | 1 | RESET |

(b) Negative-edge triggered

Thus, a change in the output of the flip-flop can be triggered only by and during the transition of the clock from 1 to 0.



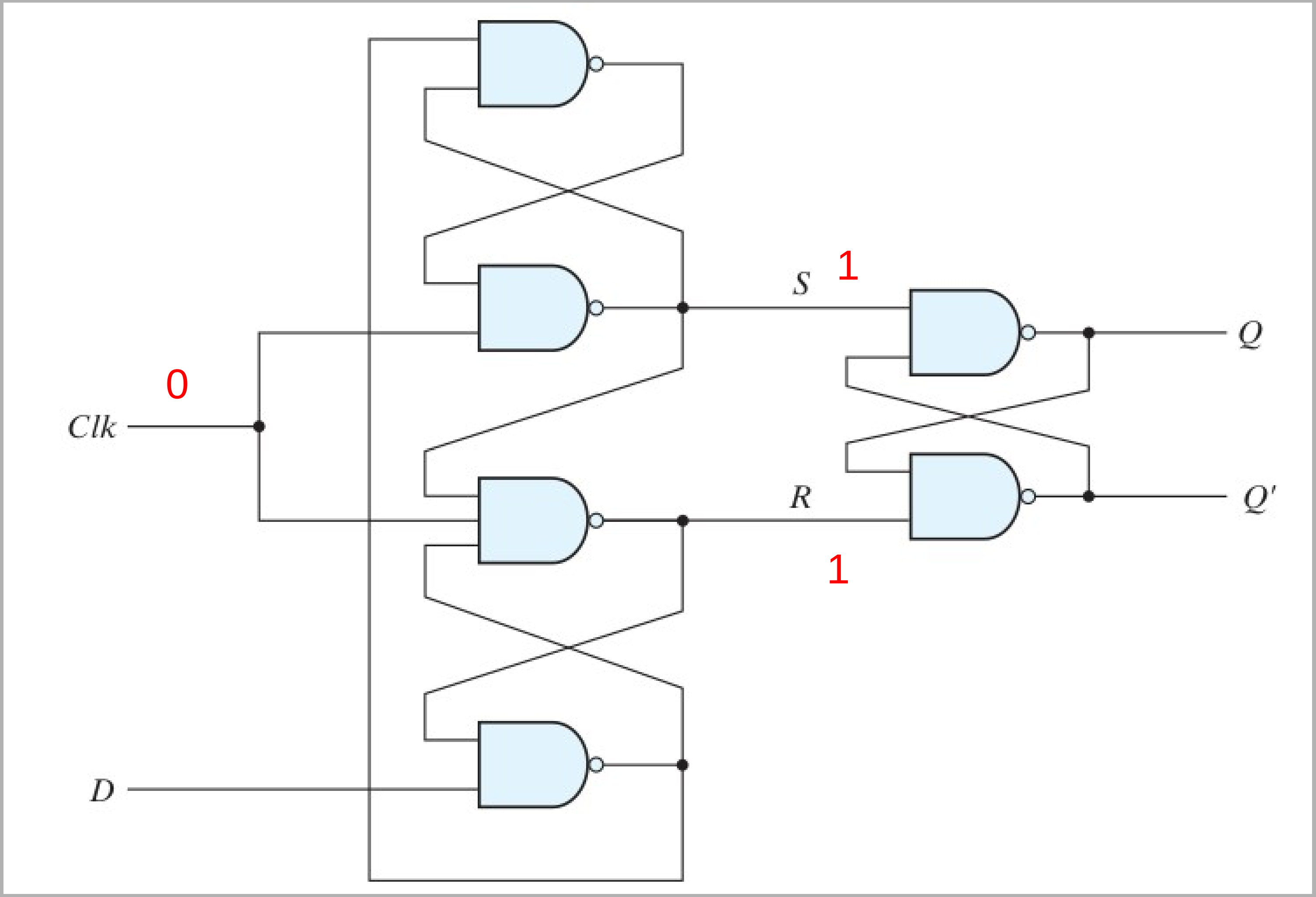
Master–slave *D* flip-flop

(1) the output may change only once

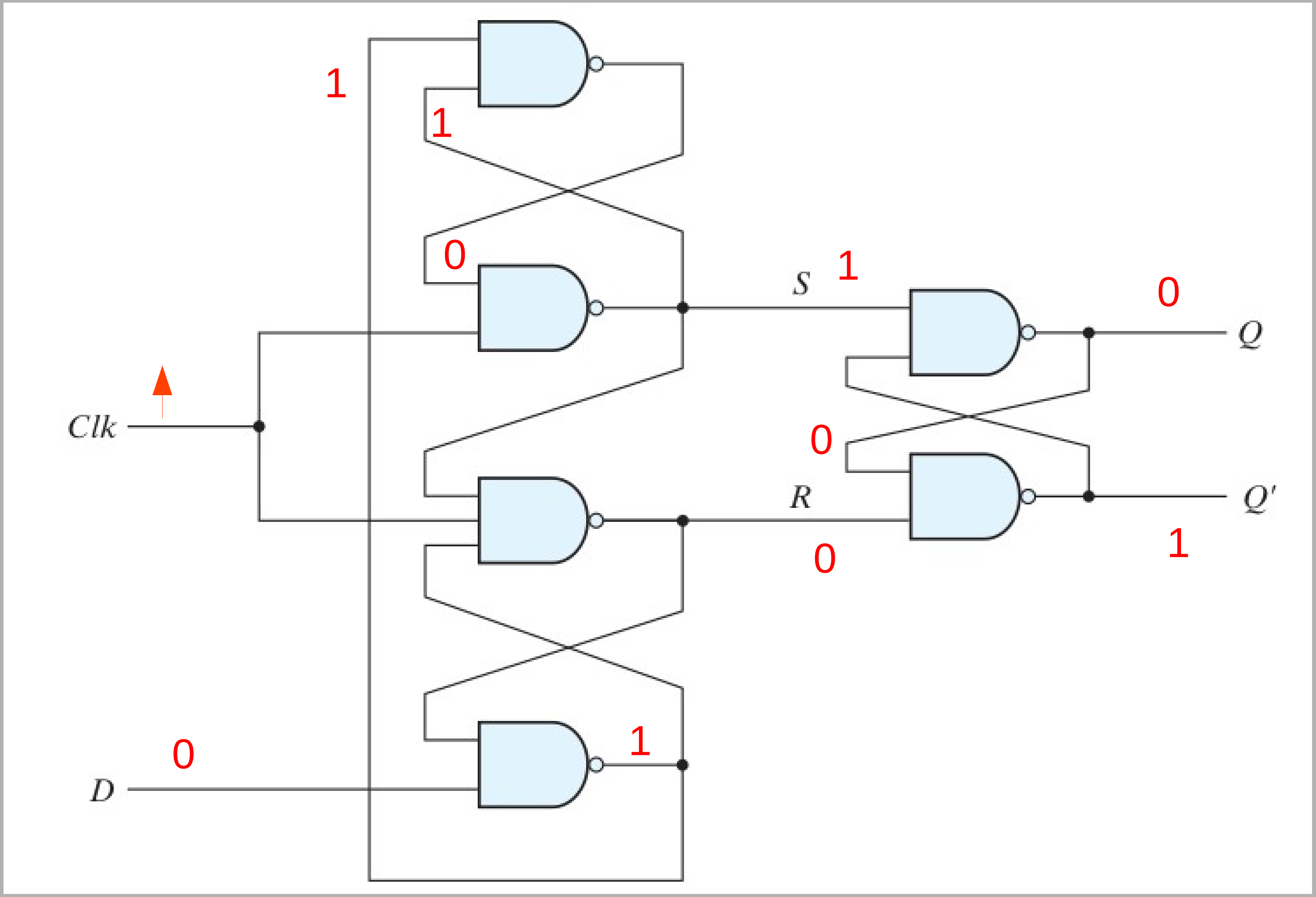
(2) a change in the output is triggered by the negative edge of the clock, and

(3) the change may occur only during the clock’s negative level. The value that is produced at the output of the flip-flop is the value that was stored in the master stage immediately before the negative edge occurred.

The advantage of this circuit is that it
uses only 6 NAND gates (24 transistors) as opposed to 10 gates (36 transistors)

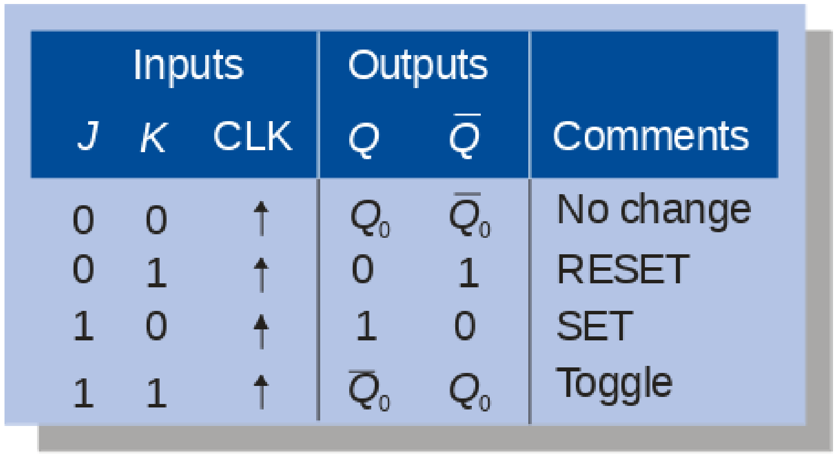


If there is a change in the D input while Clk = 1, terminal R remains at 0 because Q is 0. Thus, the flip-flop is locked out and is unresponsive to further changes in the input.

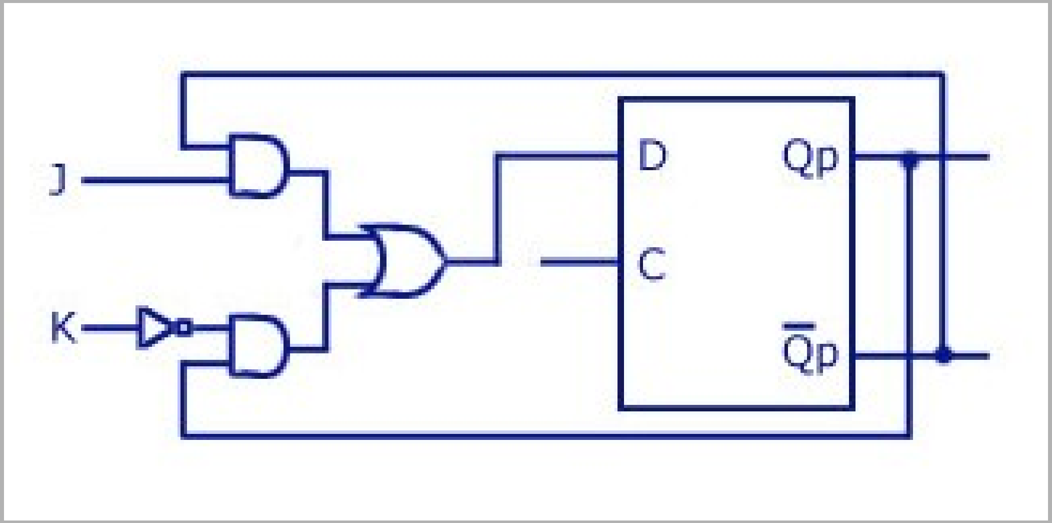


## Flip-flops

The J-K flip-flop is more versatile than the D flip flop. In addition to the clock input, it has two inputs, labeled *J* and *K*. When both *J* and *K* = 1, the output changes states (toggles) on the active clock edge (in this case, the rising edge).



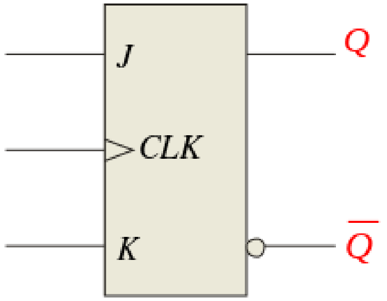
| Inputs | | | Outputs | | |
|---|---|---|---|---|---|
| $J$ | $K$ | CLK | $Q$ | $\overline{Q}$ | Comments |
| 0 | 0 | ↑ | $Q_0$ | $\overline{Q}_0$ | No change |
| 0 | 1 | ↑ | 0 | 1 | RESET |
| 1 | 0 | ↑ | 1 | 0 | SET |
| 1 | 1 | ↑ | $\overline{Q}_0$ | $Q_0$ | Toggle |



Flip-flops

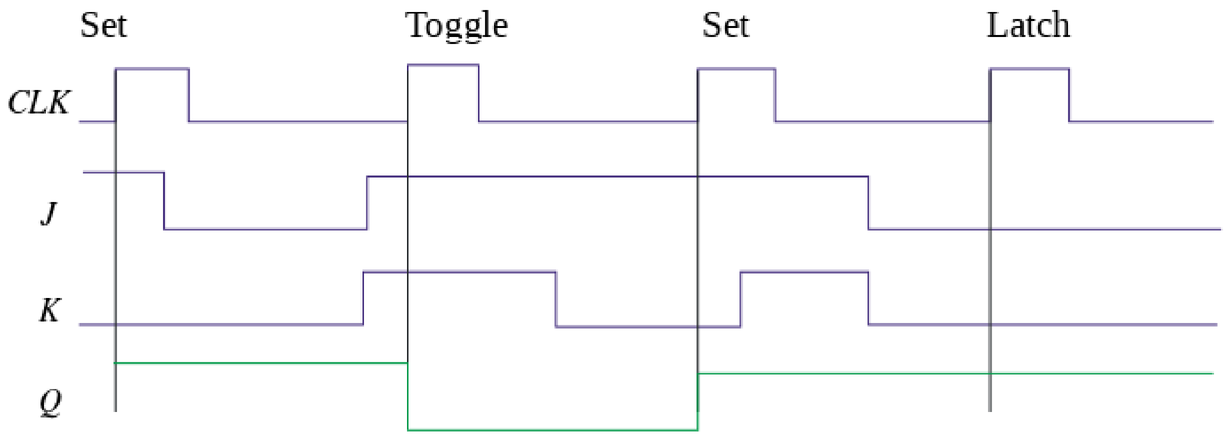
## Example

Determine the *Q* output for the *J*-*K* flip-flop, given the inputs shown.



Notice that the outputs change on the leading edge of the clock.

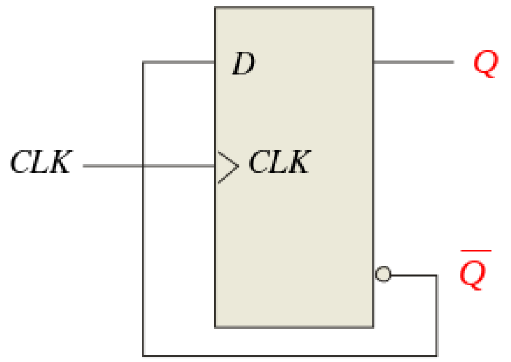
## Solution



## Flip-flops

A D-flip-flop does not have a toggle mode like the J-K flip-flop, but you can hardwire a toggle mode by connecting $\overline{Q}$ back to $D$ as shown. This is useful in some counters as you will see

For example, if $Q$ is LOW, $\overline{Q}$ is HIGH and the flip-flop will toggle on the next clock edge. Because the flip-flop only changes on the active edge, the output will only change once for each clock pulse.

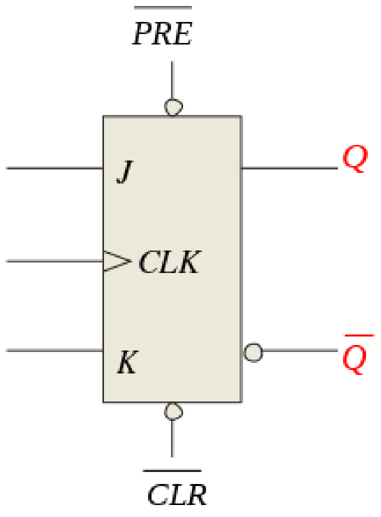


D flip-flop hardwired for a toggle mode

## Flip-flops

Synchronous inputs are transferred in the triggering edge of the clock (for example the *D* or *J-K* inputs). Most flip-flops have other inputs that are *asynchronous*, meaning they affect the output independent of the clock.

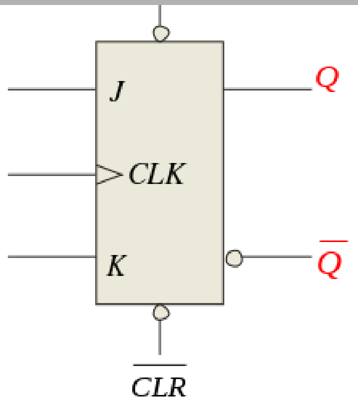
Two such inputs are normally labeled preset (*PRE*) and clear (*CLR*). These inputs are usually active LOW. A J-K flip flop with active LOW preset and CLR is shown.



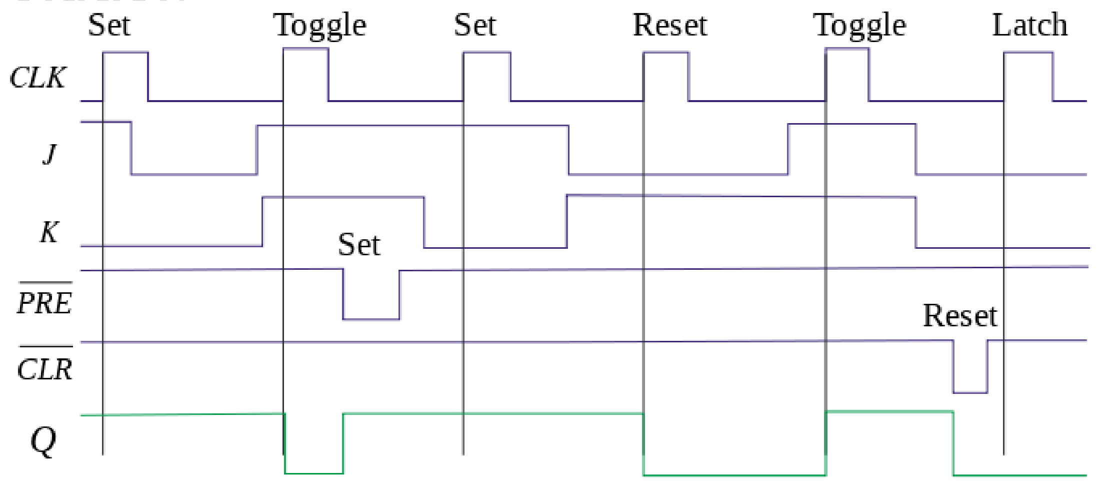
## Flip-flops

**Example**

Determine the *Q* output for the *J-K* flip-flop, given the inputs shown.

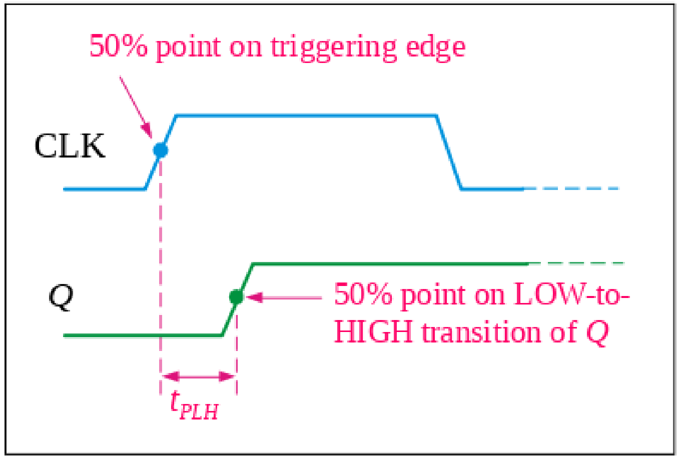


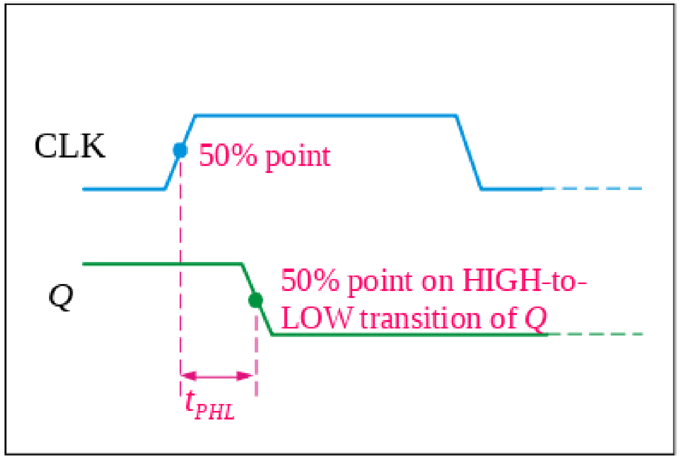
**Solution**



## Flip-flop Characteristics

**Propagation delay time** is specified for the rising and falling outputs. It is measured between the 50% level of the clock to the 50% level of the output transition.

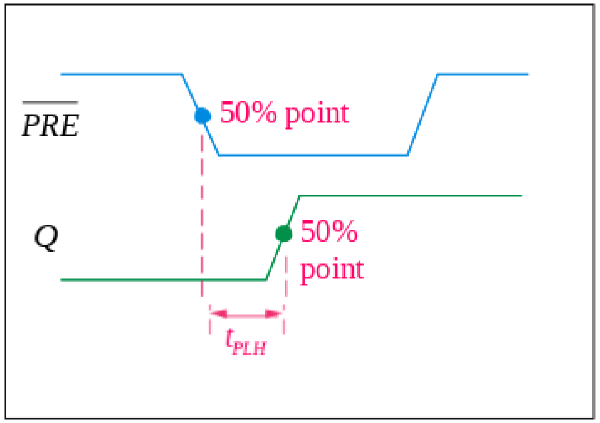

50% point on triggering edge
CLK
Q
50% point on LOW-to-HIGH transition of Q
$t_{PLH}$

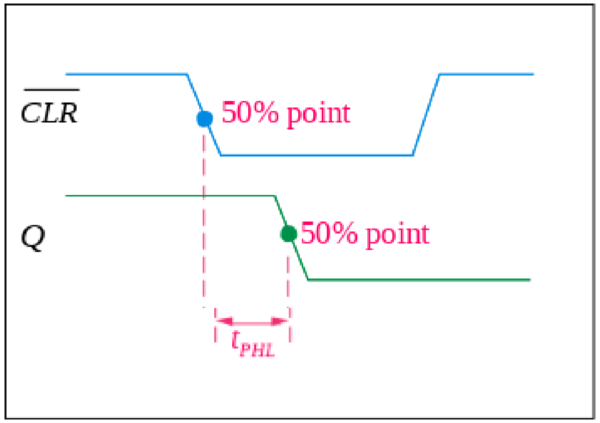


The typical propagation delay time for the 74AHC family (CMOS) is 4 ns. Even faster logic is available for specialized applications.

## Flip-flop Characteristics

Another **propagation delay time** specification is the time required for an *asynchronous* input to cause a change in the output. Again it is measured from the 50% levels. The 74AHC family has specified delay times under 5 ns.

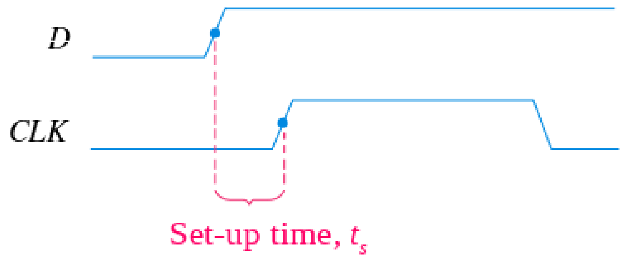




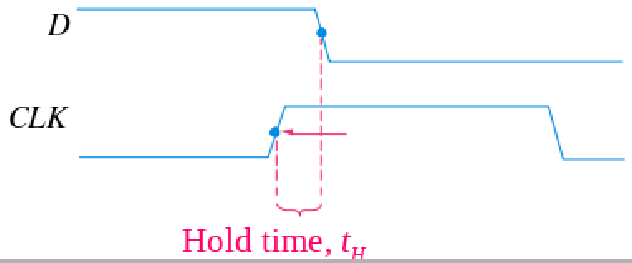
## Flip-flop Characteristics

**Set-up time** and **hold time** are times required before and after the clock transition that data must be present to be reliably clocked into the flip-flop.

**Setup time** is the minimum time for the data to be present *before* the clock.



**Hold time** is the minimum time for the data to *remain* after the clock.



## Maximum Clock Frequency

The maximum clock frequency (f max ) is the highest rate at which a flip-flop can be reliably triggered. At clock frequencies above the maximum, the flip-flop would be unable to respond quickly enough, and its operation would be impaired.

Pulse Widths

Minimum pulse widths (tw) for reliable operation are usually specified by the manufacturer for the clock, preset, and clear inputs. Typically, the clock is specified by its minimum HIGH time and its minimum LOW time.

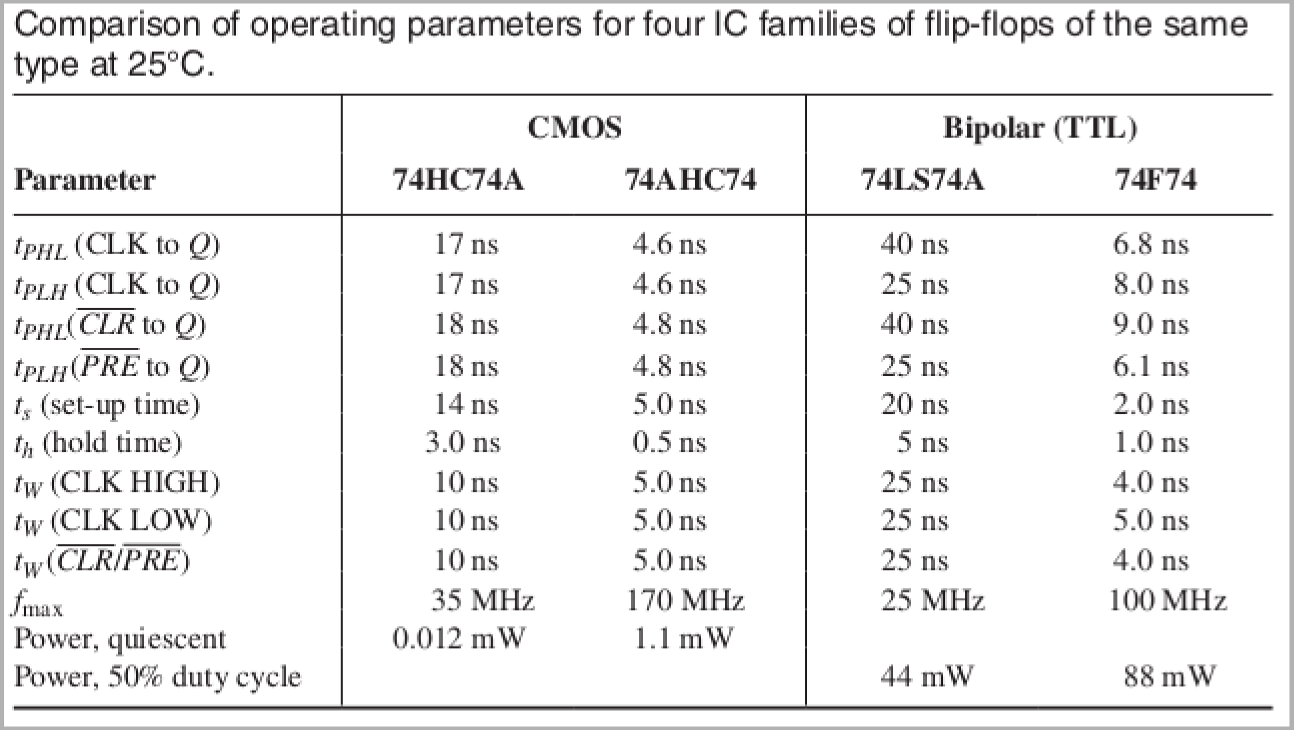
## Flip-flop Characteristics

Other specifications include maximum clock frequency, minimum pulse widths for various inputs, and power dissipation. The power dissipation is the product of the supply voltage and the average current required.

A useful comparison between logic families is the **speed-power product** which uses two of the specifications discussed: the average propagation delay and the average power dissipation. The unit is energy.

Example What is the speed-power product for 74AHC74A?

Solution the average propagation delay is 4.6 ns. The quiescent power dissipated is 1.1 mW. Therefore, the speed-power product is 5 pJ



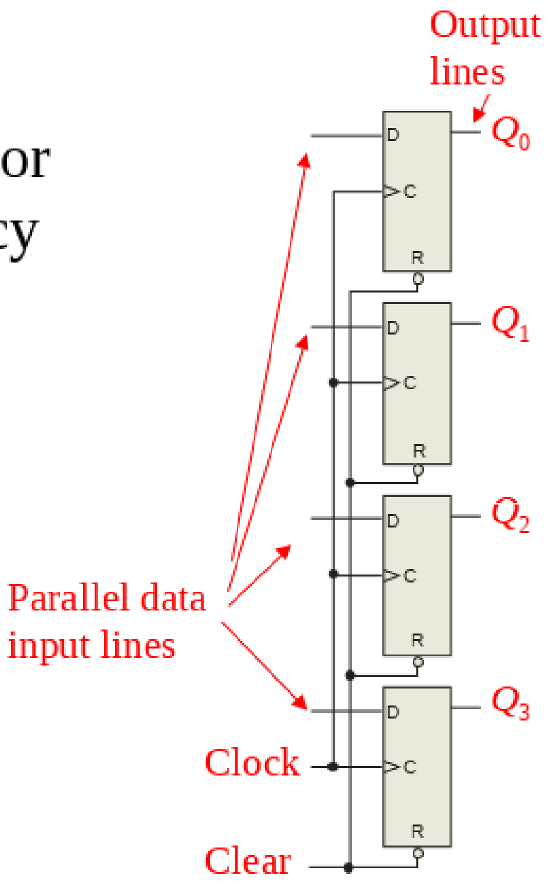
Comparison of operating parameters for four IC families of flip-flops of the same type at 25°C.

| | CMOS | | Bipolar (TTL) | |
|---|---|---|---|---|
| Parameter | 74HC74A | 74AHC74 | 74LS74A | 74F74 |
| $t_{PHL}$ (CLK to $Q$) | 17 ns | 4.6 ns | 40 ns | 6.8 ns |
| $t_{PLH}$ (CLK to $Q$) | 17 ns | 4.6 ns | 25 ns | 8.0 ns |
| $t_{PHL}$($\overline{CLR}$ to $Q$) | 18 ns | 4.8 ns | 40 ns | 9.0 ns |
| $t_{PLH}$($\overline{PRE}$ to $Q$) | 18 ns | 4.8 ns | 25 ns | 6.1 ns |
| $t_s$ (set-up time) | 14 ns | 5.0 ns | 20 ns | 2.0 ns |
| $t_h$ (hold time) | 3.0 ns | 0.5 ns | 5 ns | 1.0 ns |
| $t_W$ (CLK HIGH) | 10 ns | 5.0 ns | 25 ns | 4.0 ns |
| $t_W$ (CLK LOW) | 10 ns | 5.0 ns | 25 ns | 5.0 ns |
| $t_W$($\overline{CLR}$/$\overline{PRE}$) | 10 ns | 5.0 ns | 25 ns | 4.0 ns |
| $f_{\max}$ | 35 MHz | 170 MHz | 25 MHz | 100 MHz |
| Power, quiescent | 0.012 mW | 1.1 mW | | |
| Power, 50% duty cycle | | | 44 mW | 88 mW |

## Flip-flop Applications

Principal flip-flop applications are for temporary data storage, as frequency dividers, and in counters

Typically, for **data storage** applications, a group of flip-flops are connected to parallel data lines and clocked together. Data is stored until the next clock pulse.

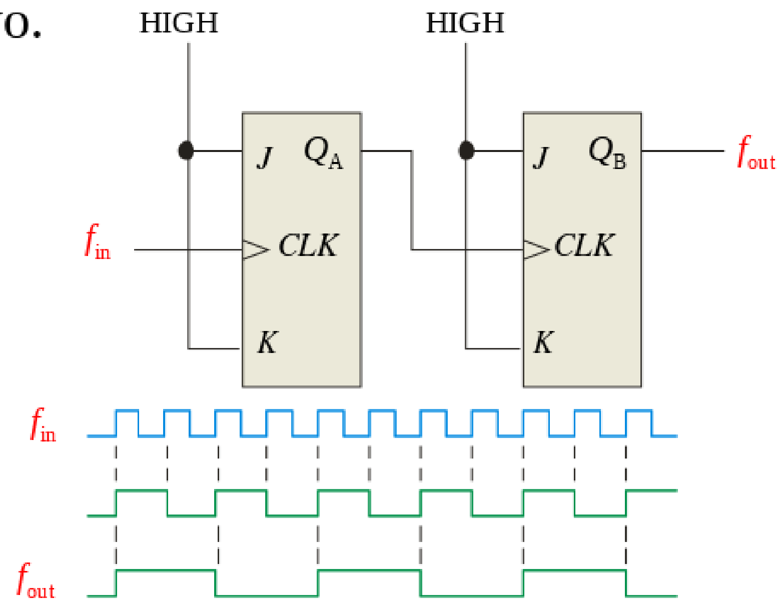


## Flip-flop Applications

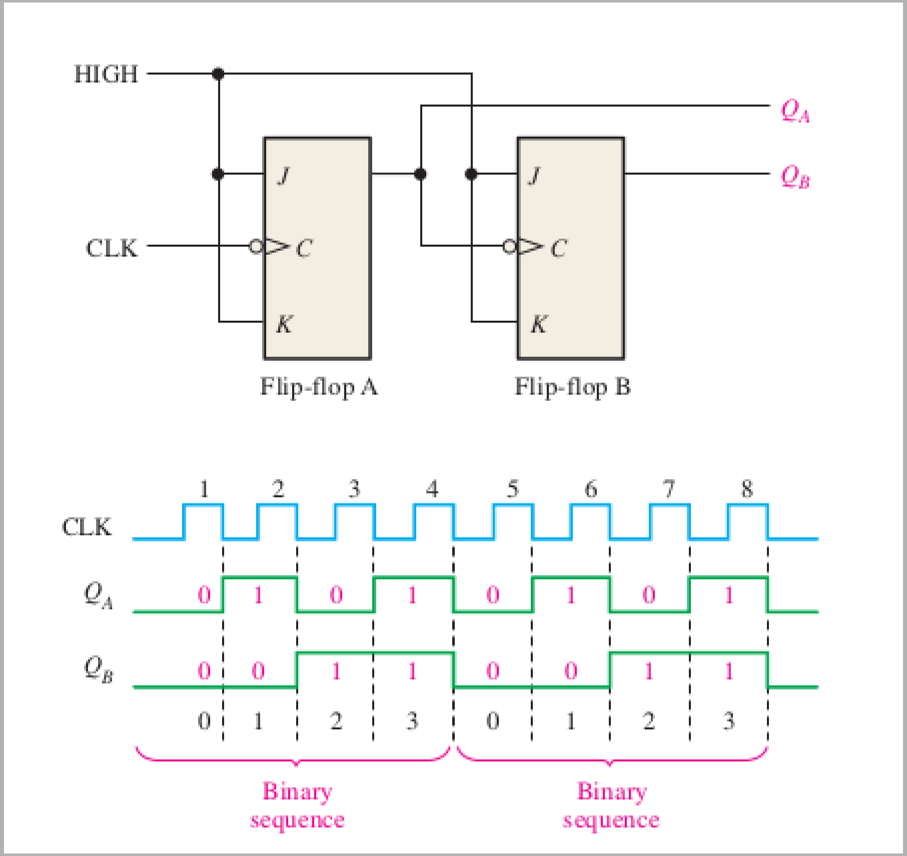
For **frequency division**, it is simple to use a flip-flop in the toggle mode or to chain a series of toggle flip flops to continue to divide by two.

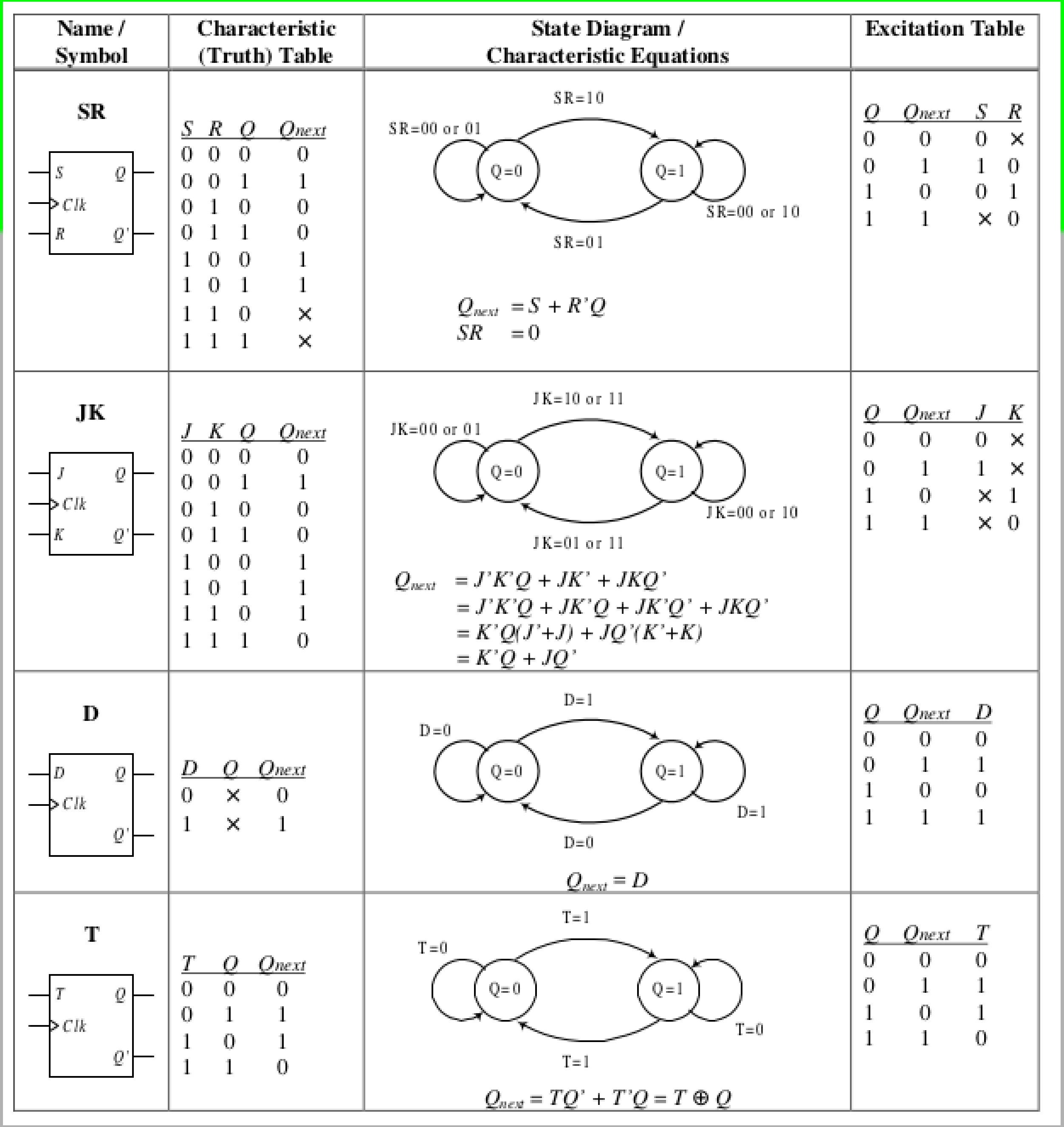
One flip-flop will divide $f_{in}$ by 2, two flip-flops will divide $f_{in}$ by 4 (and so on). A side benefit of frequency division is that the output has an exact 50% duty cycle.

Waveforms:



J-K flip-flops used to generate a binary count sequence (00, 01, 10, 11). Two repetitions are shown.





| Name / Symbol | Characteristic (Truth) Table | State Diagram / Characteristic Equations | Excitation Table |
|---|---|---|---|

**SR**

S
Q
Clk
R
Q'

| S | R | Q | $Q_{next}$ |
|---|---|---|---|
| 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 | × |

$$\begin{tikzcd}
Q=0 \arrow[loop left, "SR=00\text{ or }01"] \arrow[r, bend left, "SR=10"] & Q=1 \arrow[loop right, "SR=00\text{ or }10"] \arrow[l, bend left, "SR=01"]
\end{tikzcd}$$

$$Q_{next} = S + R'Q$$
$$SR = 0$$

| Q | $Q_{next}$ | S | R |
|---|---|---|---|
| 0 | 0 | 0 | × |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | × | 0 |

**JK**

J
Q
Clk
K
Q'

| J | K | Q | $Q_{next}$ |
|---|---|---|---|
| 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 | 1 | 0 |

$$\begin{tikzcd}
Q=0 \arrow[loop left, "JK=00\text{ or }01"] \arrow[r, bend left, "JK=10\text{ or }11"] & Q=1 \arrow[loop right, "JK=00\text{ or }10"] \arrow[l, bend left, "JK=01\text{ or }11"]
\end{tikzcd}$$

$$\begin{aligned}
Q_{next} &= J'K'Q + JK' + JKQ' \\
&= J'K'Q + JK'Q + JK'Q' + JKQ' \\
&= K'Q(J'+J) + JQ'(K'+K) \\
&= K'Q + JQ'
\end{aligned}$$

| Q | $Q_{next}$ | J | K |
|---|---|---|---|
| 0 | 0 | 0 | × |
| 0 | 1 | 1 | × |
| 1 | 0 | × | 1 |
| 1 | 1 | × | 0 |

**D**

| D | Q | $Q_{next}$ |
|---|---|---|
| 0 | × | 0 |
| 1 | × | 1 |

$$\begin{tikzcd}
Q=0 \arrow[loop left, "D=0"] \arrow[r, bend left, "D=1"] & Q=1 \arrow[loop right, "D=1"] \arrow[l, bend left, "D=0"]
\end{tikzcd}$$

$$Q_{next} = D$$

| Q | $Q_{next}$ | D |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

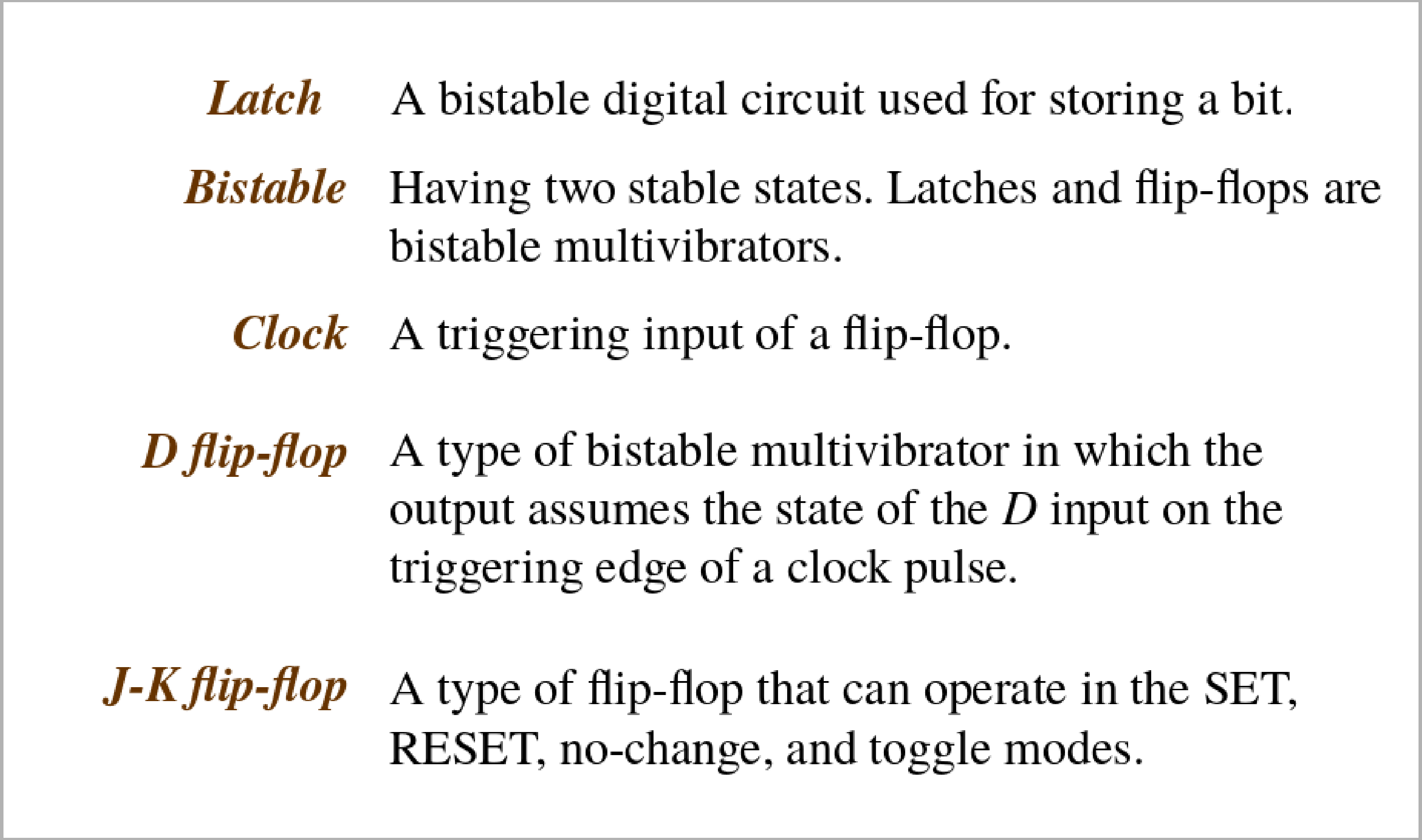
**T**

| T | Q | $Q_{next}$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

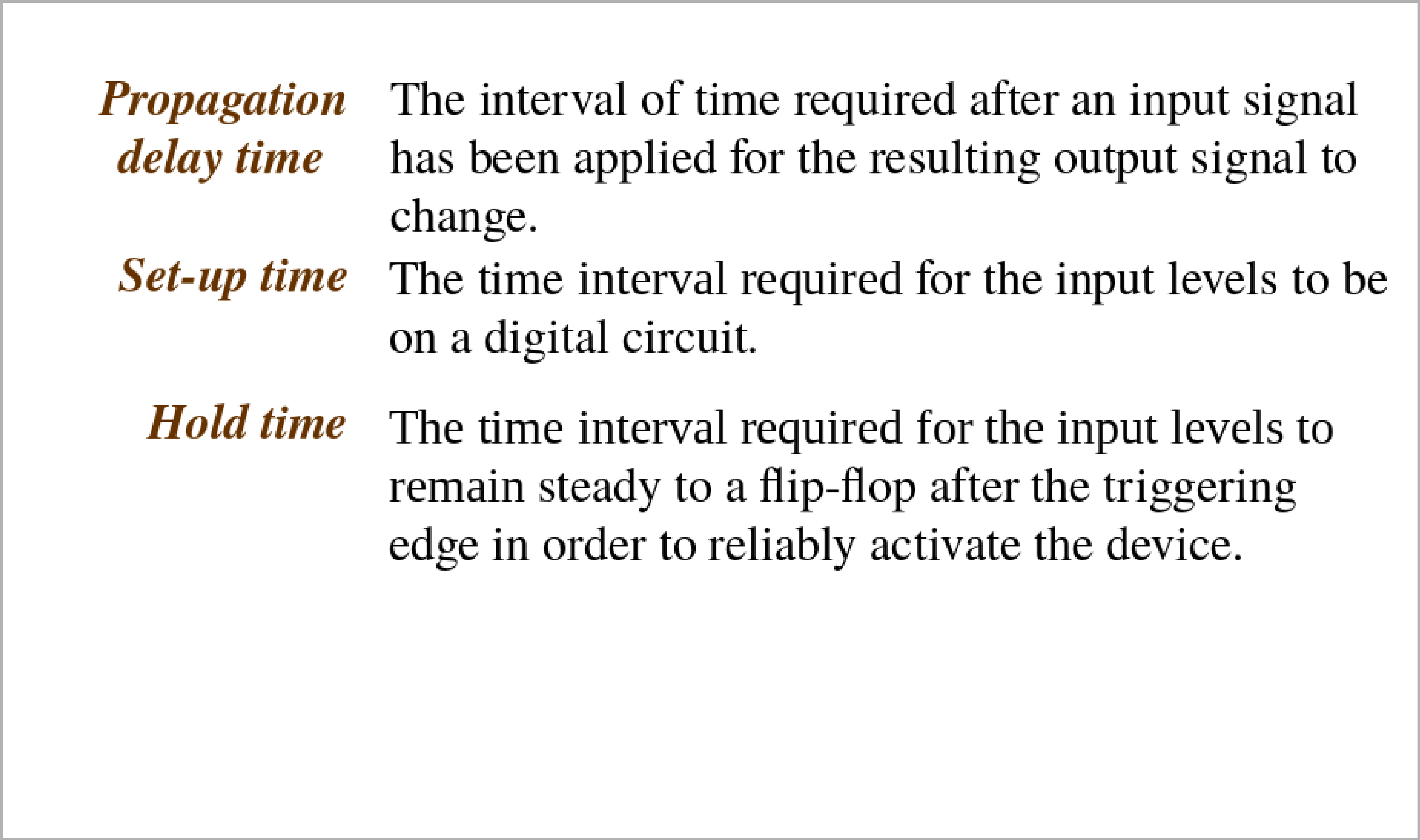
$$\begin{tikzcd}
Q=0 \arrow[loop left, "T=0"] \arrow[r, bend left, "T=1"] & Q=1 \arrow[loop right, "T=0"] \arrow[l, bend left, "T=1"]
\end{tikzcd}$$

$$Q_{next} = TQ' + T'Q = T \oplus Q$$

| Q | $Q_{next}$ | T |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |



| | |
|---|---|
| ***Latch*** | A bistable digital circuit used for storing a bit. |
| ***Bistable*** | Having two stable states. Latches and flip-flops are bistable multivibrators. |
| ***Clock*** | A triggering input of a flip-flop. |
| ***D flip-flop*** | A type of bistable multivibrator in which the output assumes the state of the *D* input on the triggering edge of a clock pulse. |
| ***J-K flip-flop*** | A type of flip-flop that can operate in the SET, RESET, no-change, and toggle modes. |



| | |
|---|---|
| ***Propagation delay time*** | The interval of time required after an input signal has been applied for the resulting output signal to change. |
| ***Set-up time*** | The time interval required for the input levels to be on a digital circuit. |
| ***Hold time*** | The time interval required for the input levels to remain steady to a flip-flop after the triggering edge in order to reliably activate the device. |