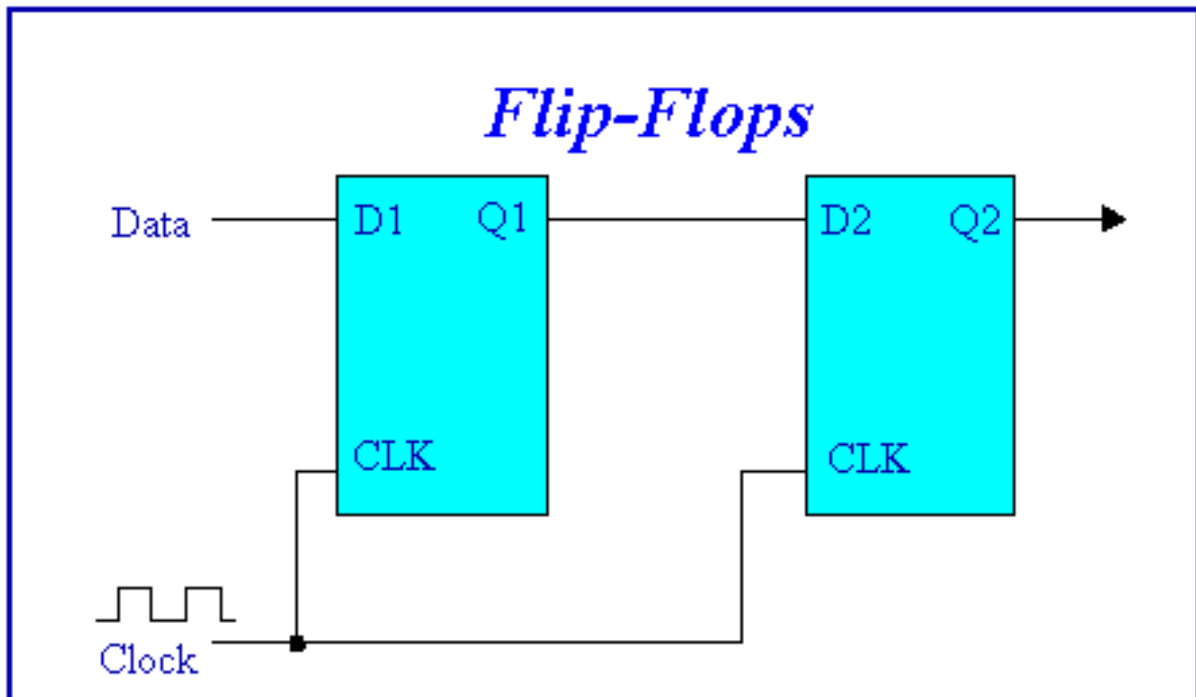


5. Flip – Flops and other Devices



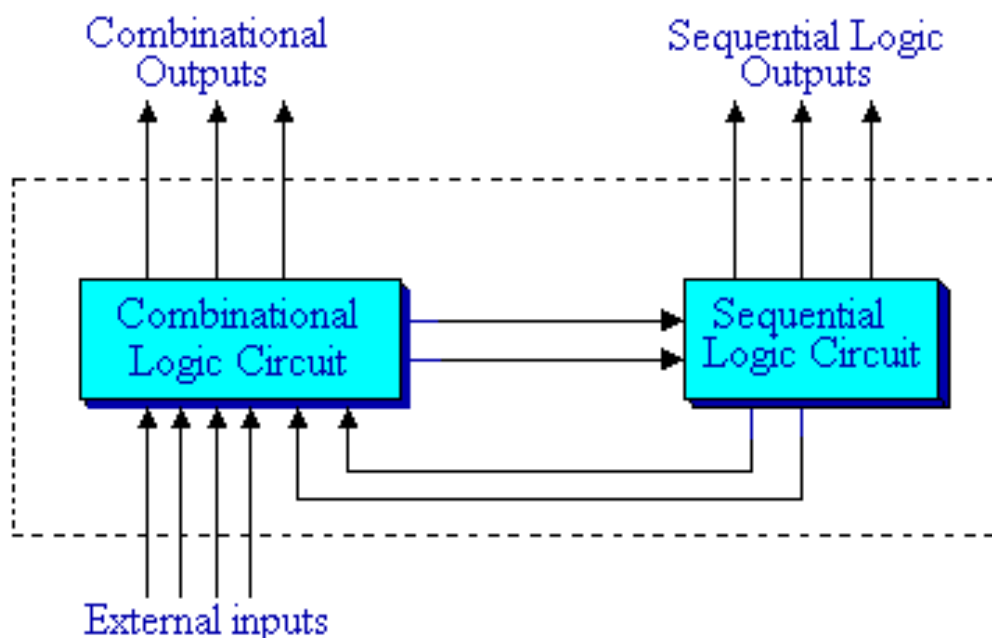
Sequential Logic

Objectives

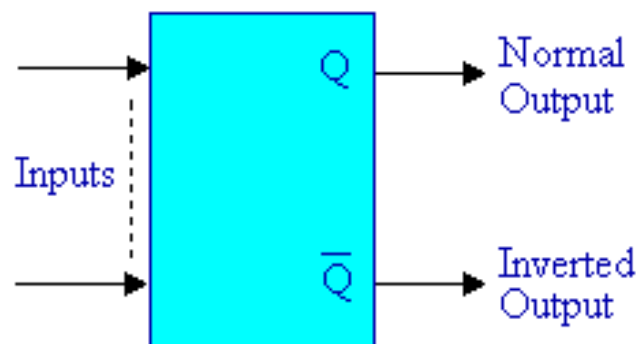
- Analyse the operation of the basic latch that is constructed using the NAND or NOR gates.
- Describe the differences between synchronous and asynchronous systems.
- Understand several types of edge triggered flip-flops, such as the J-K, D-type, and S-C.
- Understand the various flip-flop timing parameters specified by manufacturers.
- Draw the output timing waveforms of the flip-flops as responses to a given set of input signals.
- State and describe various flip-flop applications.
- Understand the characteristics of a Schmitt trigger device.
- Describe the operation of a free running oscillator using the Schmitt trigger inverter.
- Understand the operation of the 'One-shot' or monostable multivibrator

Introduction

- Logic circuits we have discussed so far are called ***Combinational Logic Circuits***.
- There is another category of Logic circuits where the output responses do not just depend on the combination of input levels applied, but also on the sequence in which the input logic levels are applied.
→ ***Sequential Logic Circuits***.
- Most digital systems are constructed using both combinational and sequential logic circuits



- The most important sequential logic element is the Flip-flop.
- It is also called a Bistable circuit because its output Q is able to stay indefinitely in one of two stable states.
- A flip-flop is also a basic memory element as it is able to remember or store 1 bit of data.
- The general Flip-flop symbol is as shown:



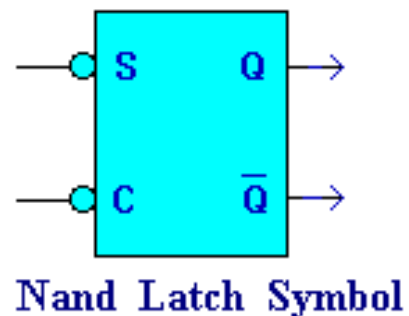
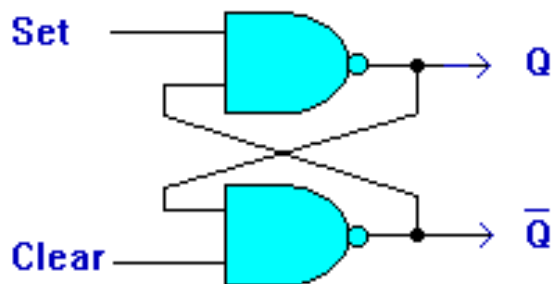
- When the Q output is High, i.e. $Q = 1$, the Flip-flop is said to be in the *SET* or *PRESET* state.
- When the Q output is Low, i.e. $Q = 0$, the Flip-flop can be described as being in the *CLEAR* or *RESET* state.

NAND Latch

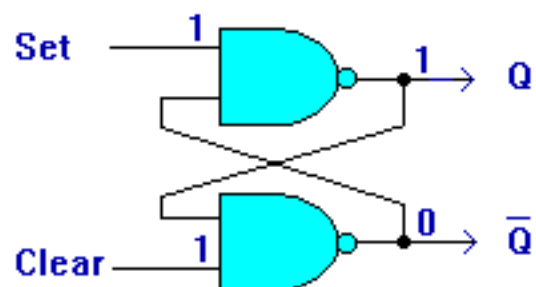
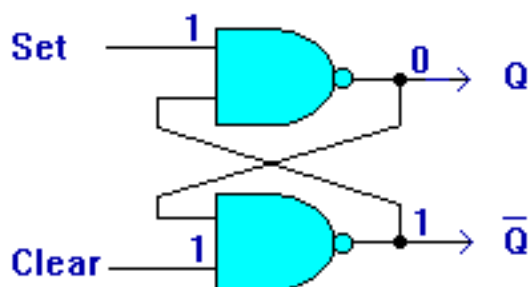
The circuit of the NAND gate latch and its symbol is shown below.

There are 2 inputs: Set & Clear and, 2 outputs Q and Not Q.

As their name implies, the Set input is basically used for setting Q to logic H and the Clear input is to force Q to low.



Under normal conditions, both Set and Clear would be in the resting state which is logic High and Q and Not Q outputs would each be in one of two possible states as, illustrated below.

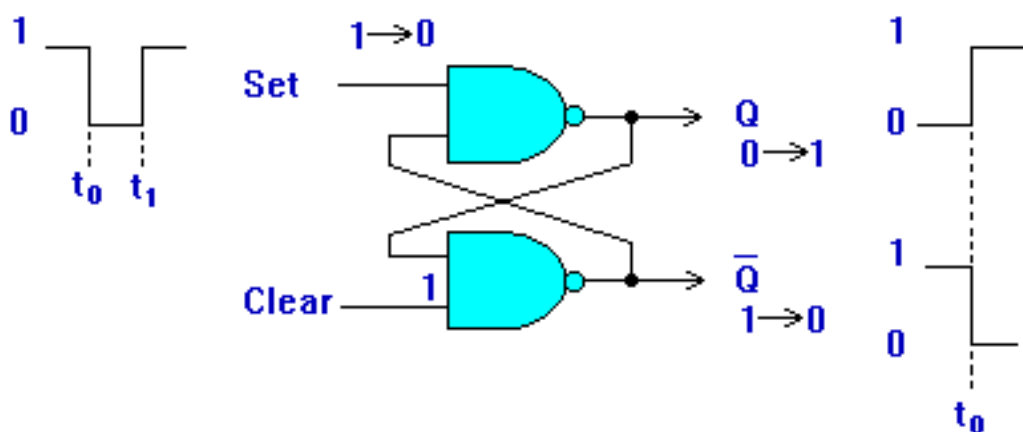


Setting the Latch (FF)

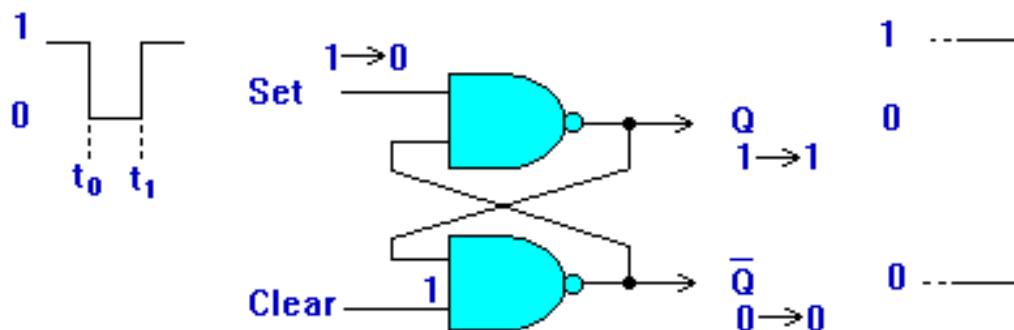
This is accomplished by pulsing the set input to logic Low.

Two possible scenarios exist:

Case (i), if the Q output was previously at logic Low, it switches to logic High. Conversely, the \bar{Q} output switches from High to Low.



Case (ii), if Q was previously at logic H, then pulsing Set to L does not result in any change, i.e. Q remains at logic H and \bar{Q} remains at logic Low.



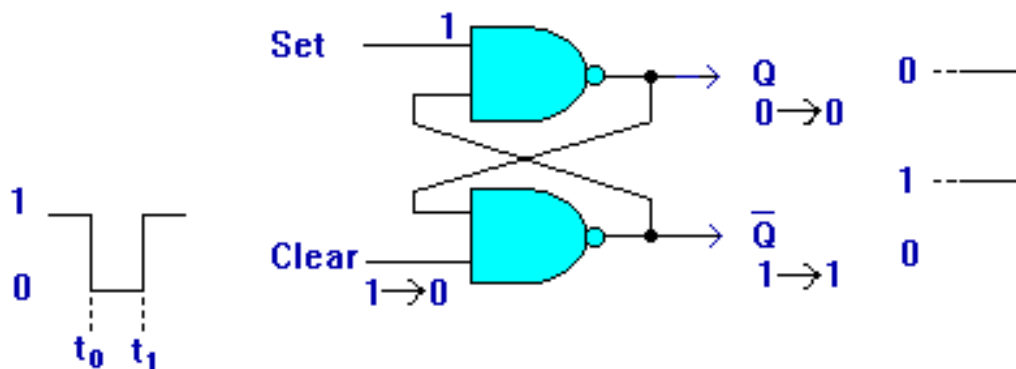
Thus, pulsing the SET input to Low will always make $Q = \text{High}$

Clearing the Latch (FF)

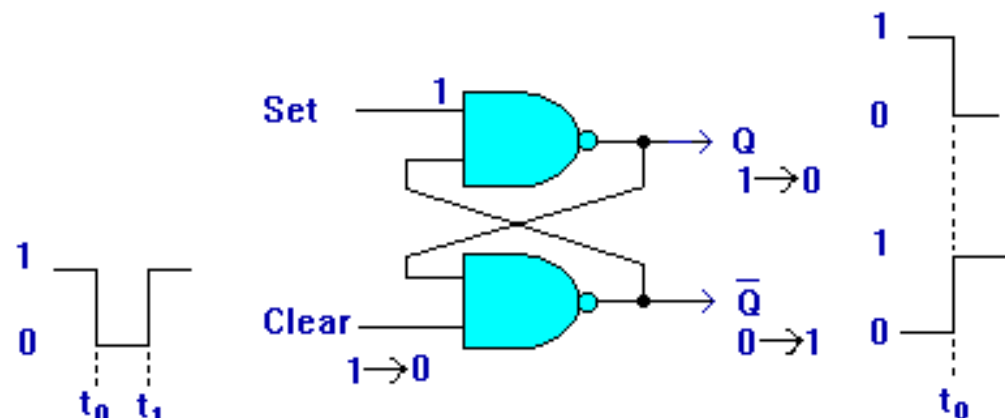
Clearing the NAND latch = forcing Q to logic L, is accomplished by pulsing the Clear input to L.

Two possible scenarios to consider:

Case (i), if Q is already at logic L, then activating the Clear input does not result in any change. The latch remains in the Clear state.



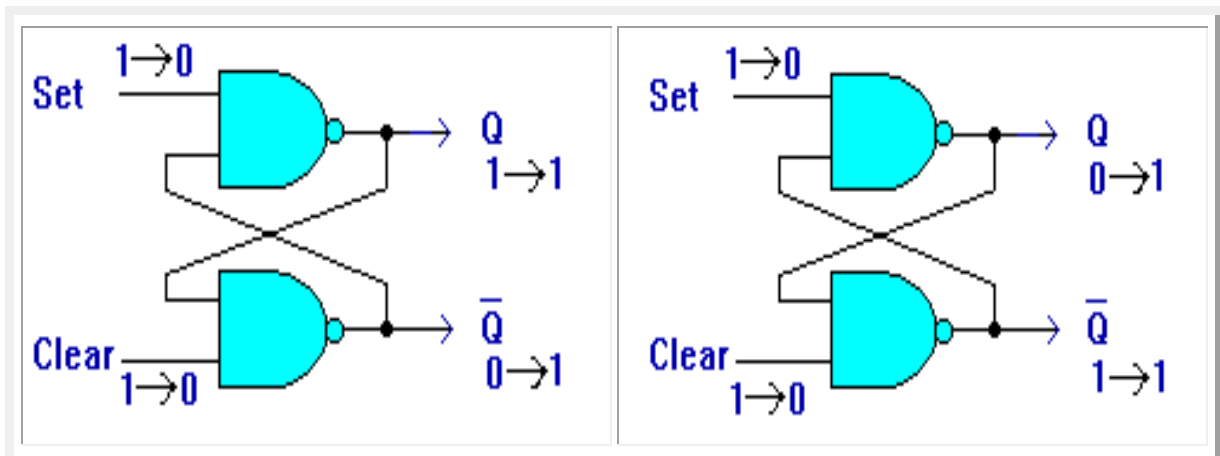
Case (ii), if the Q was previously in the H state, then activating Clear forces the Q output to switch to the logic L state and \bar{Q} output to switch to the logic H state.



Simultaneously Setting and Clearing:

If Set goes Low it will make $Q = 1$

If Clear goes Low it will make $Q = 0$



Therefore both NAND outputs will be H and $Q = \bar{Q} = 1$

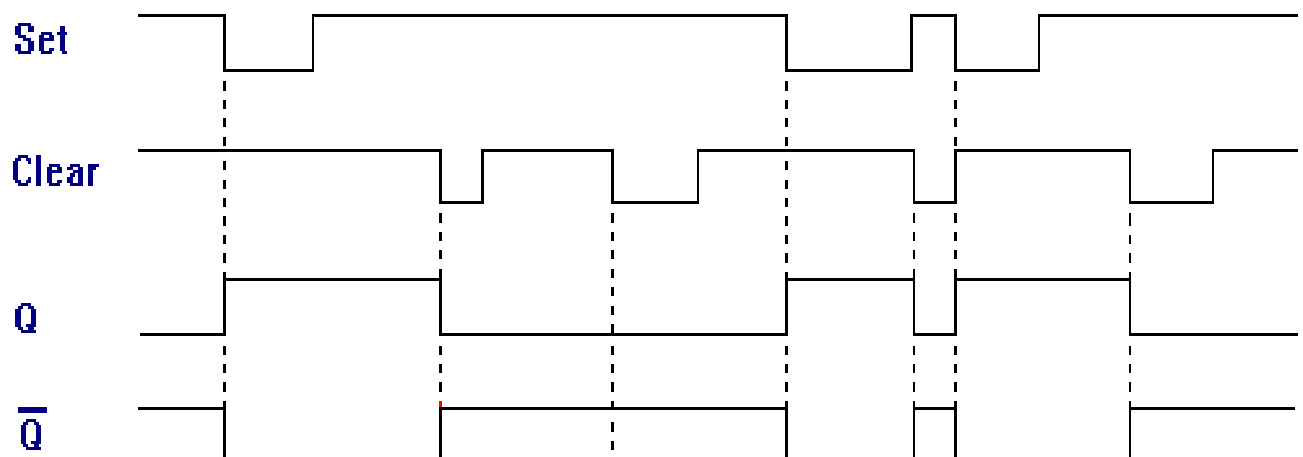
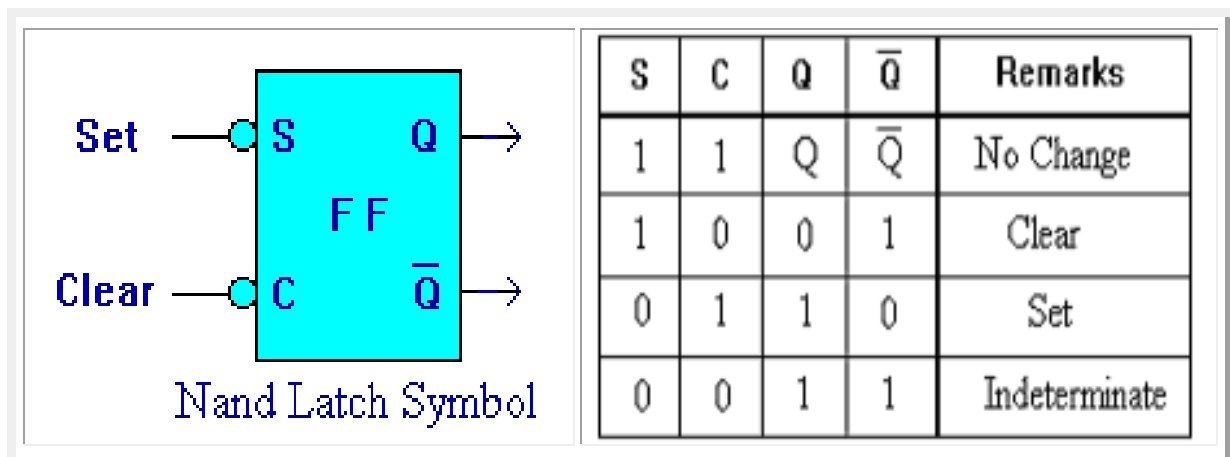
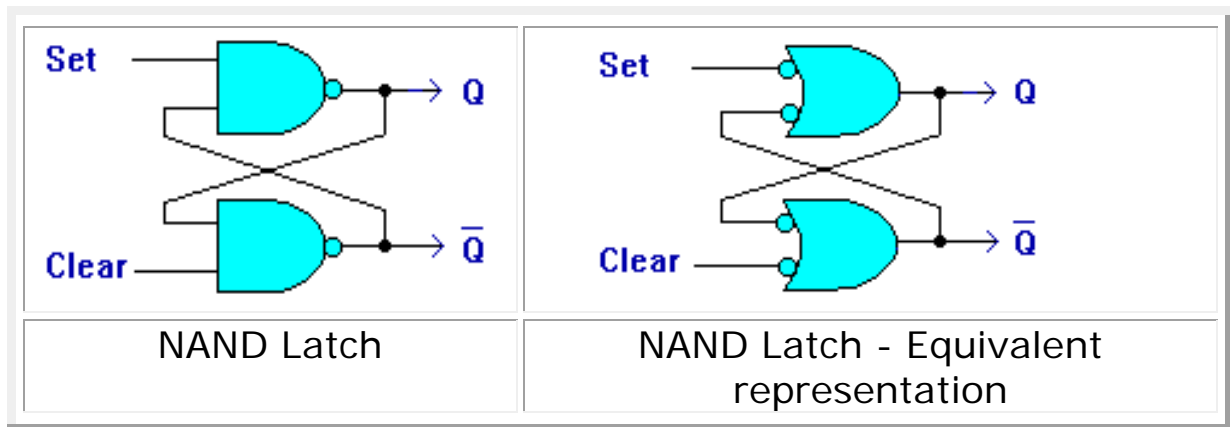
This condition is undesirable.

When Set and Clear return High resulting output state will be dependant upon which of the inputs goes High first.

Question? If Set and Clear go H together what will be the output state?

If Set and Clear goes High at the same time, then whether Q or Not Q reacts first depends basically on the gate that respond faster, i.e. whichever gate that has the shortest propagation delay will determine the appropriate output state.

Summary



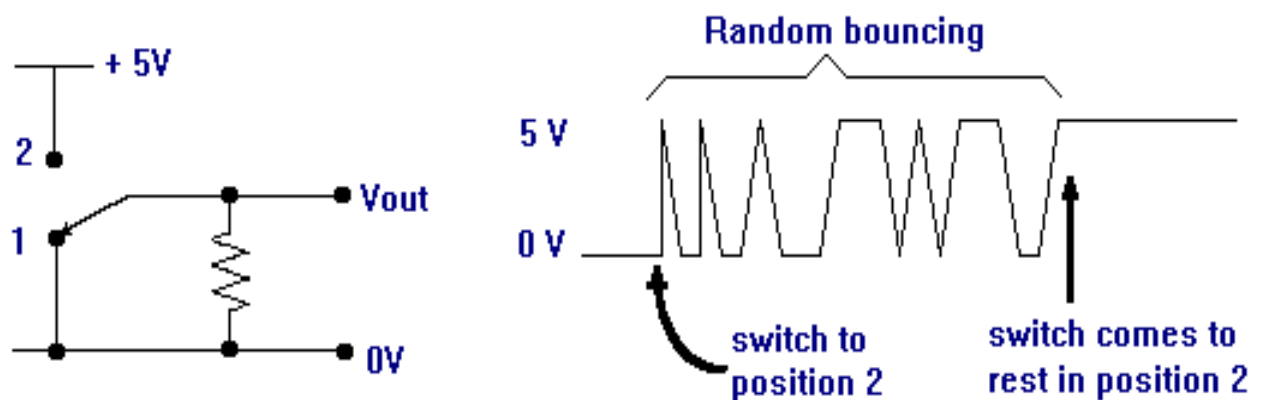
Eg: SC Flip-Flop Timing diagram

NAND Latch Application example

All mechanical switches produces switch bounce.

Switch bounce typically lasts over no more than 20 milli-Seconds.

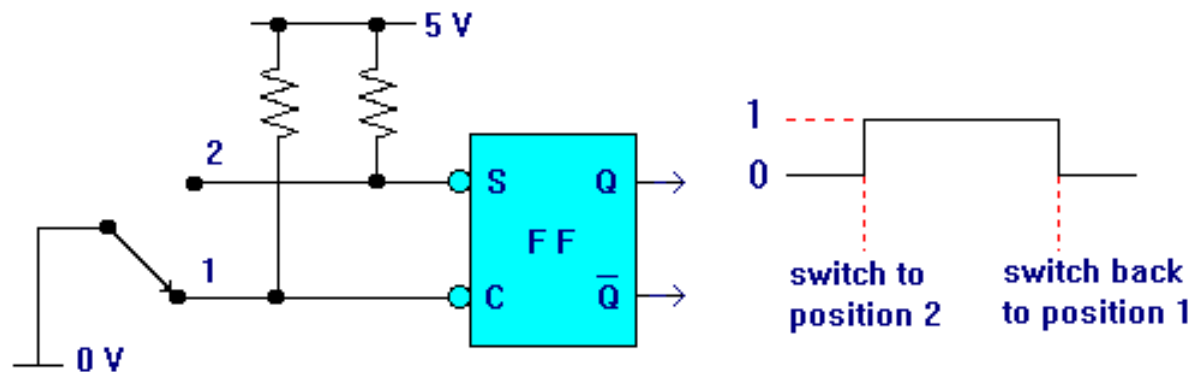
This switch bounce leads to generation of rapid and random signal transitions between High and Low levels.



Effects of switch bounce on combinational logic circuits and level-triggered circuits are minimal.

When applied to edge triggered sequential logic circuits, unpredictable response results as each random signal transitions between H and L is equivalent to an active edge.

Solution is to use a NAND latch as a circuit for mechanical switch debouncing.



When the switch is in position 1, Clear (C) is active while Set (S) is disabled.

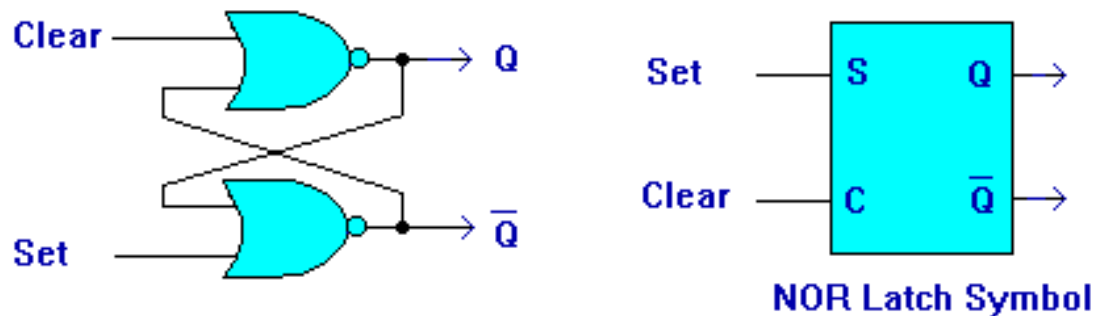
Output Q is therefore in the logic Low state.

When the switch is set to position 2, input S will be activated on first contact and Q will be forced High.

- switch bounce at contact 2 causes only S and C to be High.
- therefore no changes occur at output Q.

NOR gate Latch

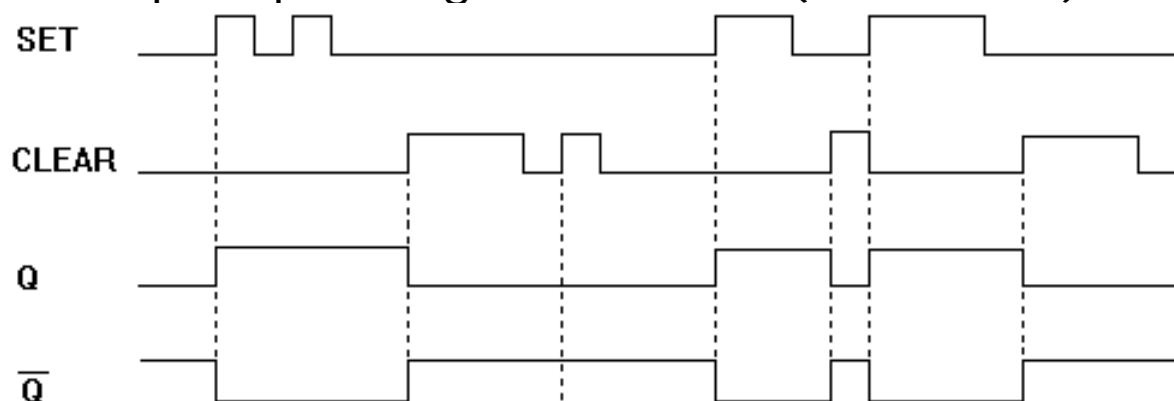
The NOR gate Latch uses NOR gates in a similar cross-coupled connection.



The function table describing the behaviour of the Latch is as shown below:

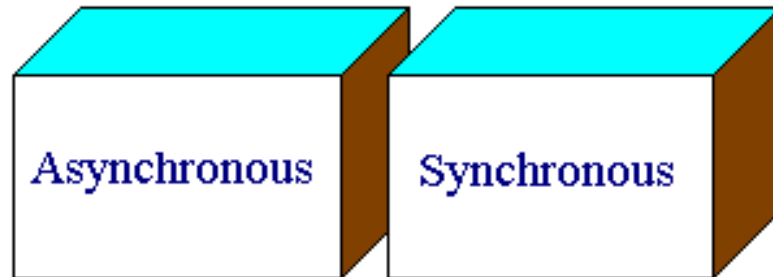
Set	Clear	Q	\bar{Q}	Remarks
0	0	Q_0	Q_0	no change
1	0	1	0	Set
0	1	0	1	Clear
1	1	0	0	Invalid

S-C Flip-Flop timing waveforms (NOR Latch)



Clocked Signals

Digital systems can operate either:



Asynchronous

- The outputs of logic circuits can change state at any time when one or more of the inputs change.
- Difficult to design and troubleshoot.
- Signs of a bad design.

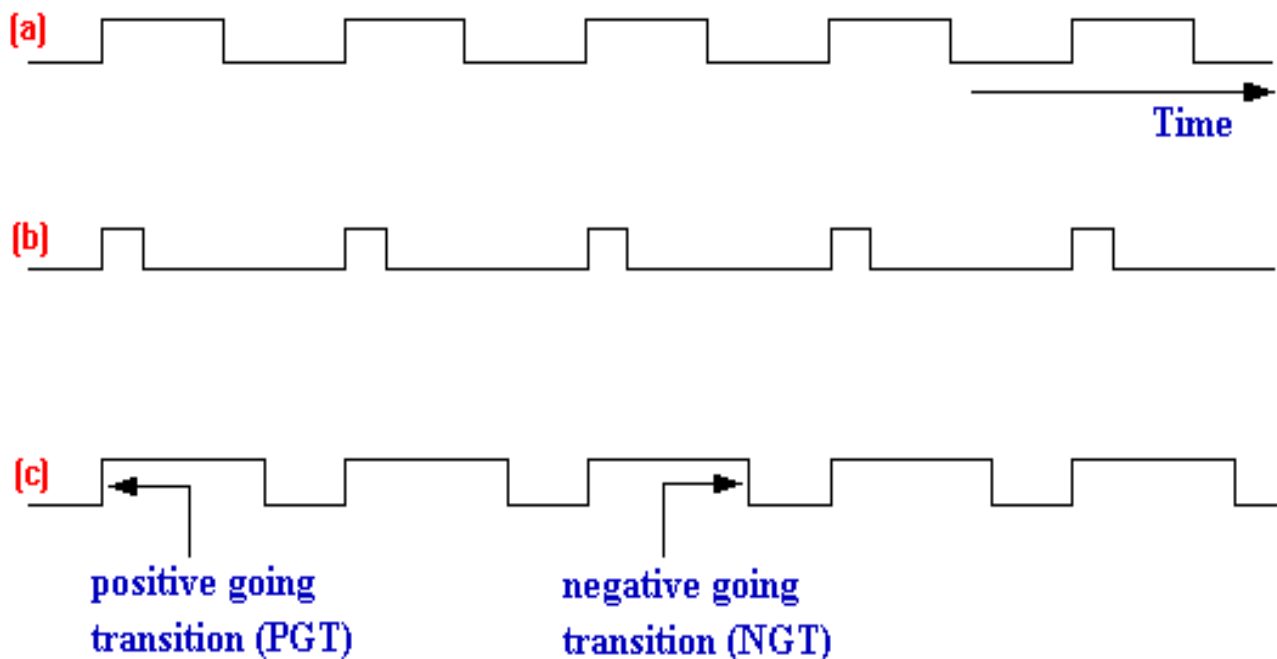
Synchronous

- The exact time at which any output can change state is determined signal commonly called the clock.
- Relatively easy to design and troubleshoot.
- Signs of a good design.
- Generally uses more gates than equivalent asynchronous circuits.
- Most digital systems are synchronous.

Clocked Signals

A clock signal is a periodic square wave that repeats at regular intervals.

Three examples of which are shown below:



Question

Which of the clock signals shown has the highest frequency?

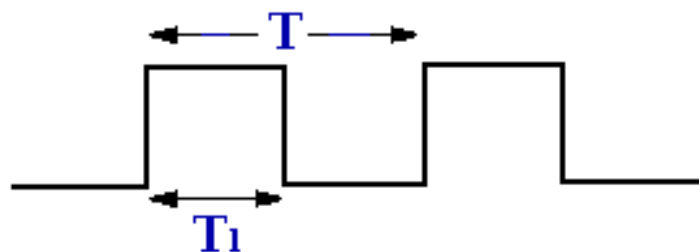
Answer by selecting (a), (b), (c), or (d) for same frequency.

Answer to previous question is (d), i.e. all the 3 waveforms have the same frequency because they have the same period!

What is different amongst the 3 signal waveforms shown previously is the **Duty cycle**, which is defined as:

$$(\text{Mark Duration/Period of CLK}) * 100 \%$$

Where, the Mark duration is defined as the time duration of the High Pulse.



$$\text{Duty Cycle} = \frac{T_1}{T} * 100 \%$$

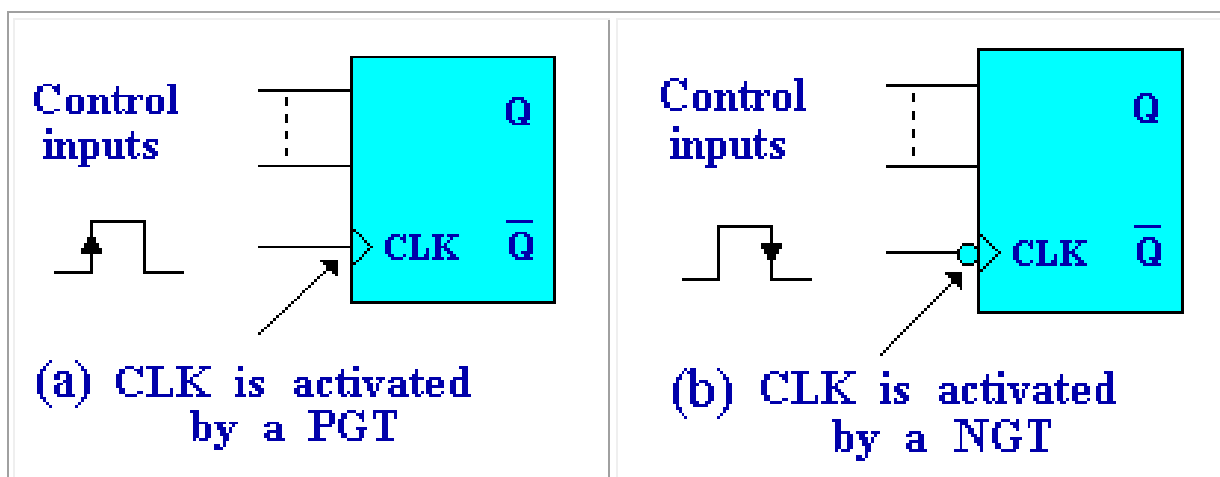
Question

What is the duty cycle of a symmetrical or perfect square wave?

What is your **Answer**?

Clocked Flip-Flops

- Clocked flip-flops have a *clock* input labelled: CLK, CK, or CP.
- Most flip-flops are *edge-triggered*.
- The Flip-flops can either be PGT or NGT activated.



Clocked flip-flops typically have one or more control inputs, which are called the synchronous control inputs.

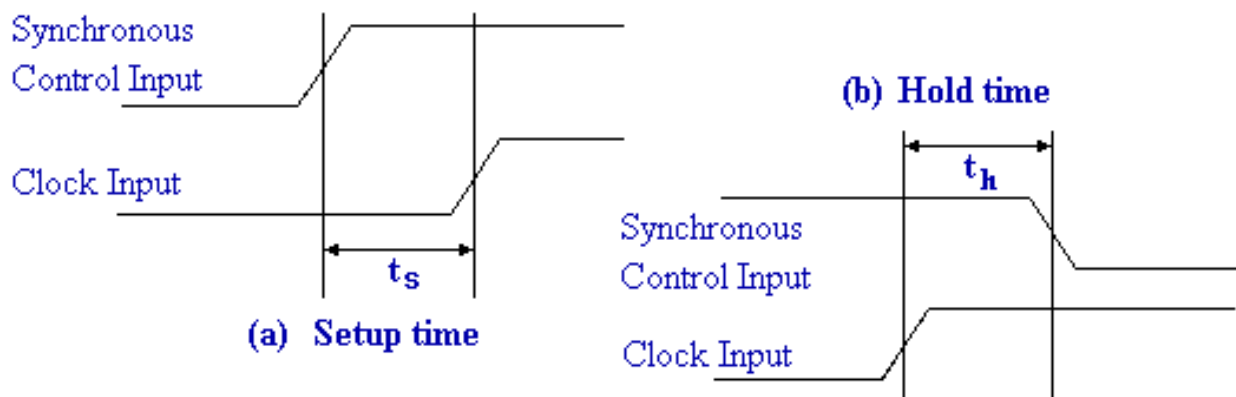
1. The control inputs only effect the Q output when an active clock transition occurs.
2. The control inputs are said to prepare FF output for a change.
3. The arrival of the active CLK edge actually *triggers* the change.

Setup and Hold Times

These two timing requirements must be met for successful FF operation.

All timing measurements are with respect to the active edge of the CLK.

The time is measured between the 50% points on the transitions.



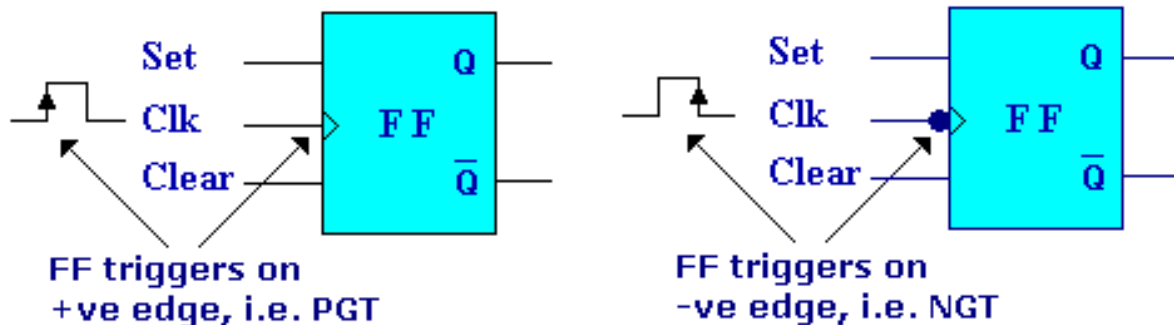
Setup time (t_s) is the time interval immediately before the active transition of the CLK during which the synchronous inputs **MUST NOT** change.

Hold time (t_h) is the time interval immediately after the active transition of the CLK during which the synchronous inputs **MUST NOT** change.

Control i/p must be held stable for (a) a setup time (t_s) prior to active clock edge and for (b) a time (t_h) after the active clock transition.

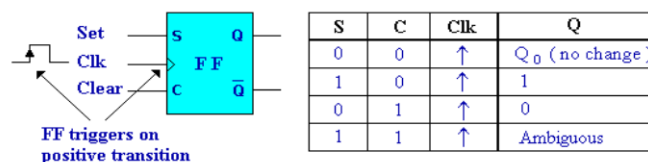
Clocked S-C flip-flop

In this FF, Set and Clear can only affect the output only in the presence of an active clock edge.

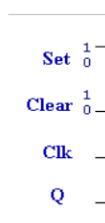


Double click on the power-point attachment on a Clocked SC FF example.

Clock SC Flip-flop – PGT CLK

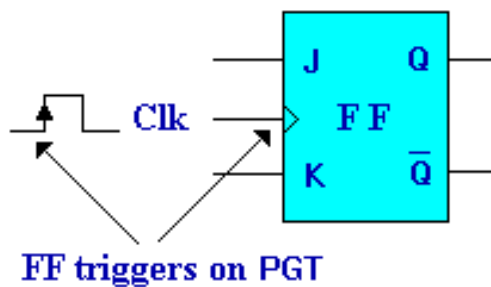


Assume that Q is initially Low or 0.



Clocked J-K flip – flop

Has 2 synchronous control inputs - J K.



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

The J-K inputs control the output states of the FF in almost the same way as the S-C inputs of the clocked S-C FF.

The J-K inputs control the output states of the FF in almost the same way as the S-C inputs of the clocked S-C FF.

The J-K FF is much more useful than the S-C FF because it has NO undefined states.

If $J = K = 1$ the output will toggle.

Double click on the power-point attachment for JK-waveform example ↓

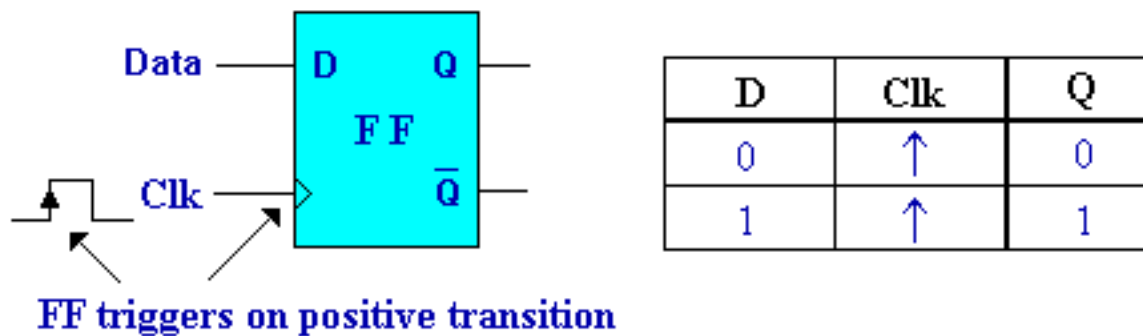


JK_flipflop.ppt

Clocked D flip – flop

The D flip-flop has only one synchronous input called the D input.

The Q output takes on the logic value of the D input on the active edge of the clock.



Output $Q = D$ when the active clock transition is applied.

Application examples of D flip-flops includes:

- Parallel transfer of data.
- Synchronization of asynchronous signals.
- Counters

Double click on the power-point attachment for D flip-flop waveform example ↓

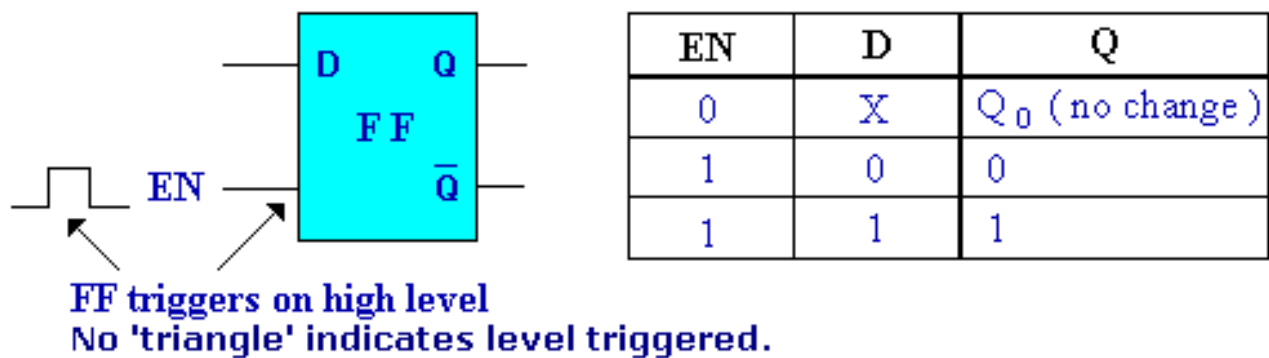


DFF1.ppt

D Latch

The circuit is level triggered.

The latch responds to input levels NOT input edges - there is NO edge detection circuit.



When the circuit is enabled and the output will be the D input.

When the circuit becomes disabled it "stores" the value of D just before the end of the active level of enable.

Setup and hold times must be met for correct operation of this latch.

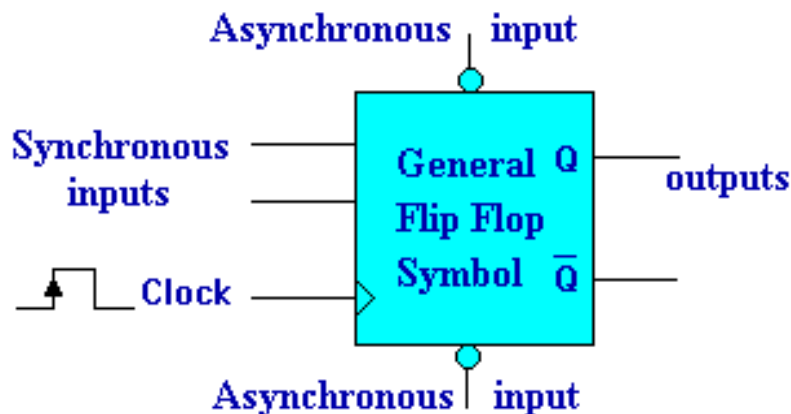
Double click on the power-point attachment for D flip-flop waveform example ↓



DLatch.ppt

Asynchronous Inputs

The general flip-flop symbol shows a flip-flop has both synchronous (e.g. J,K) and asynchronous inputs.

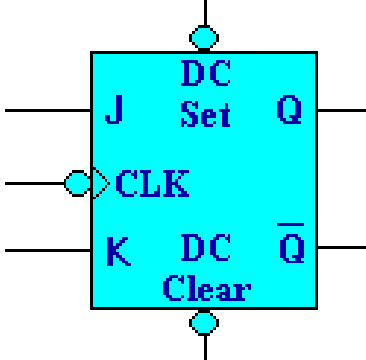


Synchronous inputs	Effects on the FF output are synchronized with the clock input.
Asynchronous inputs	Effects on the FF output are independent of the clock and synchronous inputs.

These *asynchronous* inputs are used to either:

- CLEAR/RESET the FF and make $Q = 0$,
- Or,
- SET/PRESET the FF making $Q = 1$.

The *asynchronous* inputs are normally active Low and they **OVERRIDE** the synchronous inputs.

	<p><u>DC</u> Set</p> <p>1</p> <p>0</p> <p>1</p> <p>0</p>	<p><u>DC</u> Clear</p> <p>1</p> <p>1</p> <p>0</p> <p>0</p>	<p><u>Response</u></p> <p>CLK operation</p> <p>Q = 1</p> <p>Q = 0</p> <p>Not used</p>
---	--	--	---

The asynchronous inputs respond to DC input levels and NOT edges.

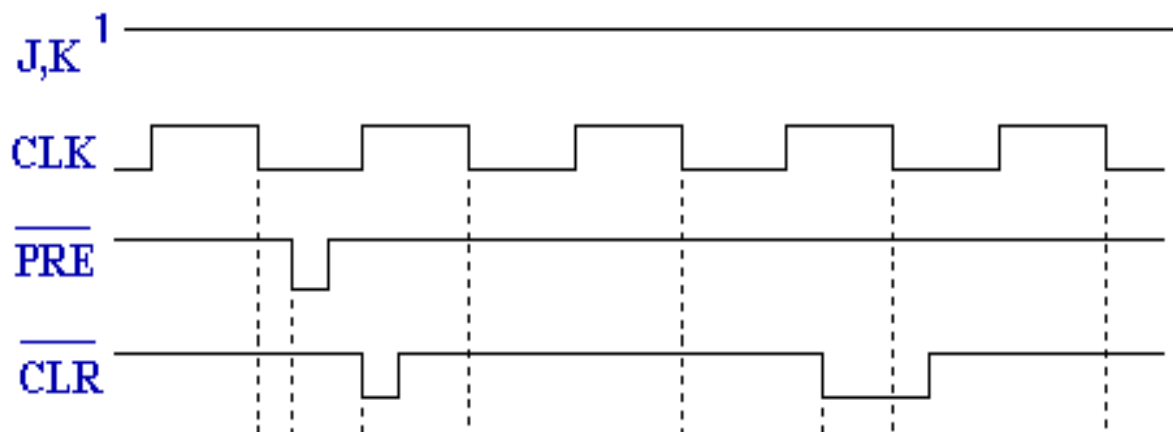
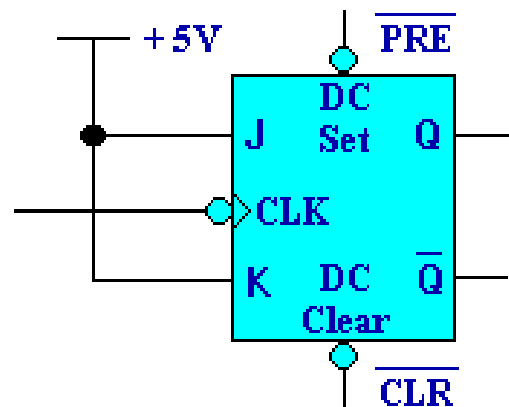
The asynchronous inputs on a FF are typically usually used to SET the flip-flop into a known state before it is used.

Other terms that are used synonymously with DC Set and DC Clear are:

○	DC SET	DC CLEAR
○	PRESET or PRE	CLEAR or CLR
○	SET	RESET
○	SD (direct set)	CD (direct clear)

Example:

Given the JK Flip-flop shown and the input waveforms for PRE and CLR as shown below, what is the output waveform at Q, assuming that it is initially High?



Double Click on the ICON for the solution.



Answer.bmp

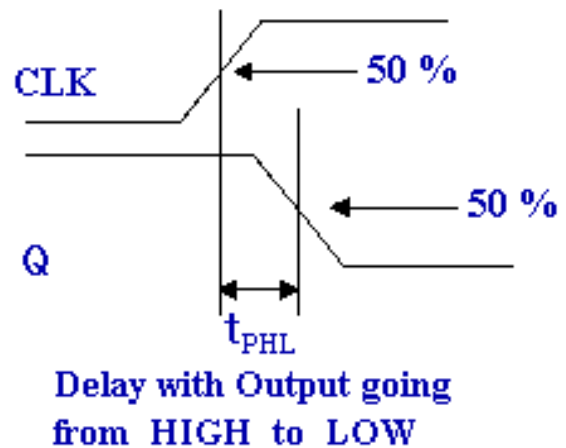
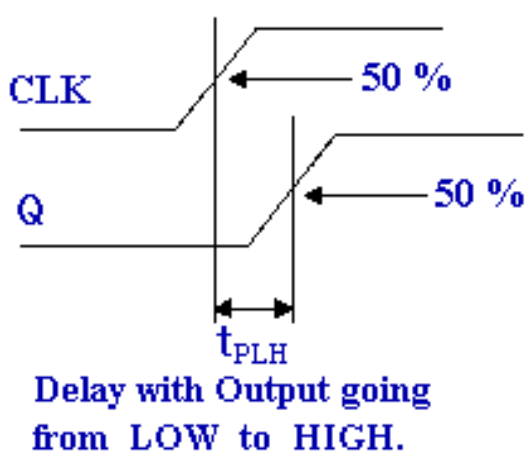
PRE = CLR = 1	Asynchronous inputs are inactive and FF will respond to J, K, and CLK inputs.
PRE = 0; CLR = 1	PRE is active and Q = 1 regardless of the status of the other inputs.
PRE = 1; CLR = 0	CLR is active and Q = 0 regardless of the status of the other inputs.
PRE = CLR = 0	This condition should not be used, since it results in an ambiguous response.

Flip-Flop timing Consideration

*Setup and Hold times: this has already been described**

Propagation Delays

When a FF or logic gate input changes the output will change too. However, the o/p does not change immediately - there is a small delay. This is called propagation delay.



Propagation delay basically measures the response times of the devices. Two values are quoted: t_{PLH} and t_{PHL} .

t_{PLH} is the delay that results when the output state changes from Low to High in response to the input stimulus.

t_{PHL} is the delay when the output changes from High to Low. These two delays are usually different & IC manufactures specify their maximum values.

Maximum Clocking Frequency, f_{MAX}

This is the highest frequency that can be applied to the FF CLK input and still have it trigger reliably.

IC manufactures will only specify minimum value of f_{MAX} .

Clock Pulse HIGH and Clock Pulse LOW Times

$t_{W(L)}$ - The minimum time duration for which the CLK signal *must* remain LOW before it goes HIGH.

$t_{W(H)}$ - The minimum time duration for which the CLK signal *must* remain HIGH before it goes LOW.

Asynchronous Active Pulse Width

The minimum time for which PRESET or CLEAR inputs *must* remain in their active state to reliably set or clear the FF.

Question

Use table 5-2 to determine the following:

- How long can it take for the Q output of a 74LS112 to switch from 0 to 1 in response to an active clock transition?
- Which flip-flop requires the control inputs to remain stable for the longest time after the active CLK transition?
- What is the narrowest pulse that can be applied to the PRE input of a 7474 flip-flop?

	nS	7474	74LS112	74C74	74HC112
t_s	Set-up time	20	20	60	25
t_h	Hold time	5	0	0	0
t_{PHL}	CLK to Q	40	24	200	31
t_{PLH}	CLK to Q	25	16	200	31
t_{PHL}	$\overline{\text{CLR}}$ to Q	40	24	225	41
t_{PLH}	$\overline{\text{PRE}}$ to Q	25	16	225	41
t_{w(L)}	CLK L time	37	15	100	25
t_{w(H)}	CLK H time	30	20	100	25
t_{w(L)}	$\overline{\text{CLR}}$ or $\overline{\text{PRE}}$	30	20	100	25
f_{max}	in MHz	15	30	5	20

Examples of Flip-Flop Applications

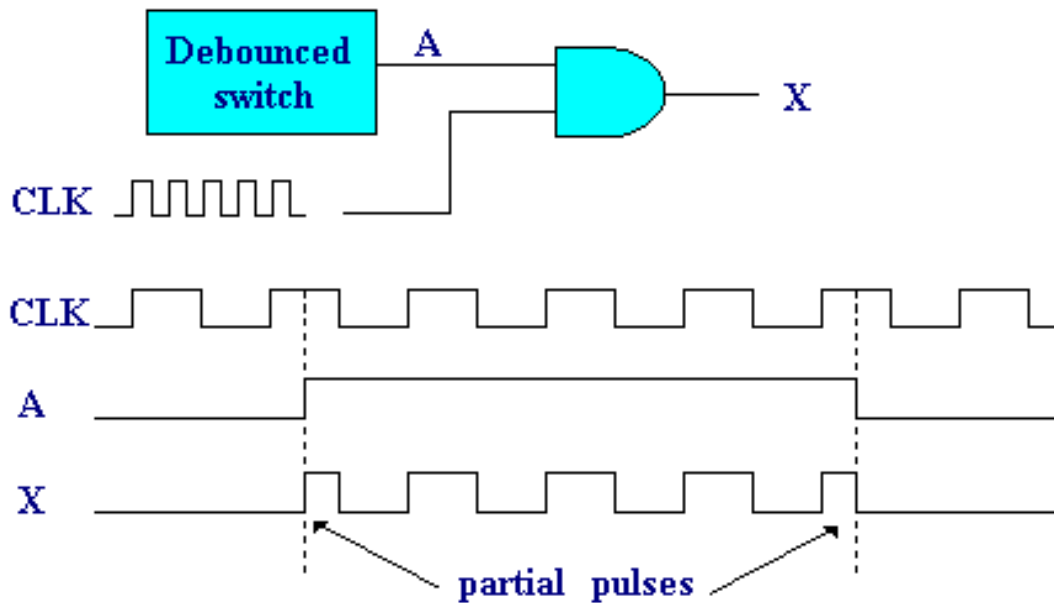
- Flip-flops are very versatile devices that have numerous applications, some of these includes:

- Synchronisation of Signals
- Detecting an input sequence
- Data Storage and Transfer
 - Synchronous
 - Asynchronous
- Frequency division and counting

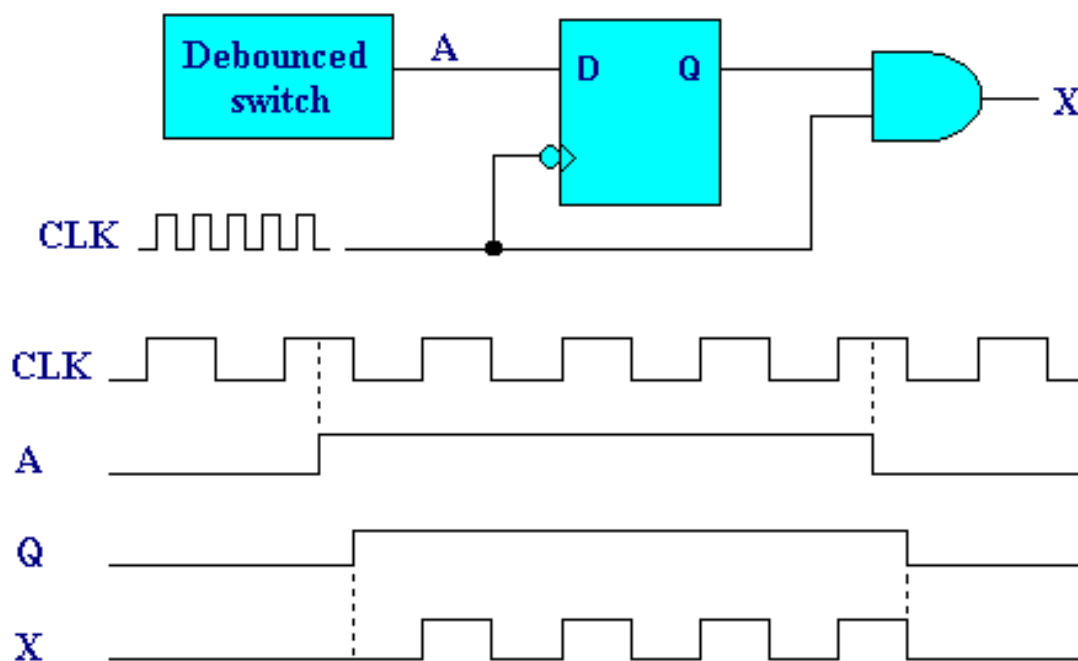
Etc.

Flip Flop synchronisation

A manually operated switch to enable a pulse train will produce partial pulses at output X:



An edge-triggered D flip-flop is used to synchronize the switch output and the pulse train.



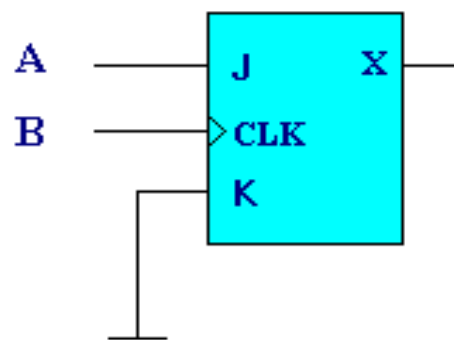
Detecting an input sequence

To detect sequence of signals applied, use of just gates will not do.

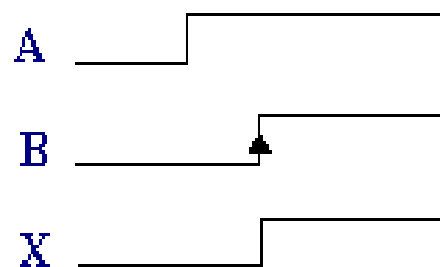
Clocked J-K or D flip-flops is needed to respond to a particular sequence of inputs.

In the circuit shown output X can only go High if the signals applied at A and B is of a particular sequence.

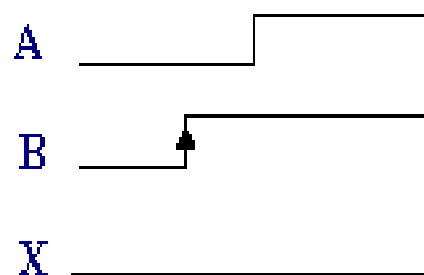
Assuming X starts at 0,



- A goes High before B goes High, Output X will be High



- However if B goes High prior to A, then X remains at Logic Low.



Data storage and transfer

Most common use for FF is for the storage of data or information.

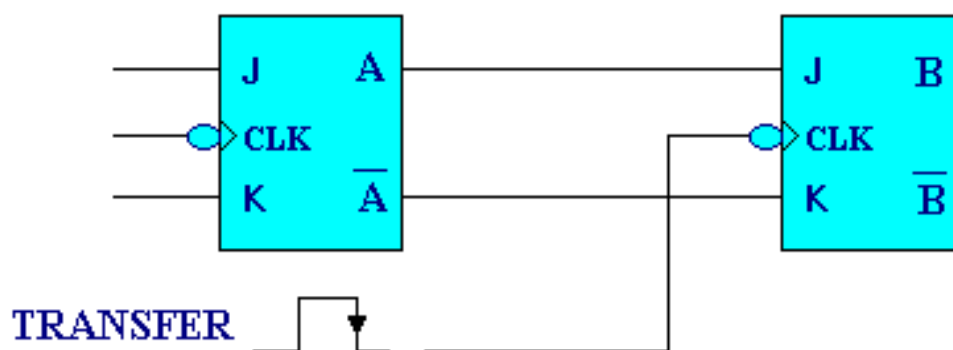
Data is stored in groups of FF called *registers*.

Common operation of registers is data transfer.

Transfer operations are either **Synchronous** or **Asynchronous**.

Synchronous transfer is accomplished using a clock and FF synchronous inputs.

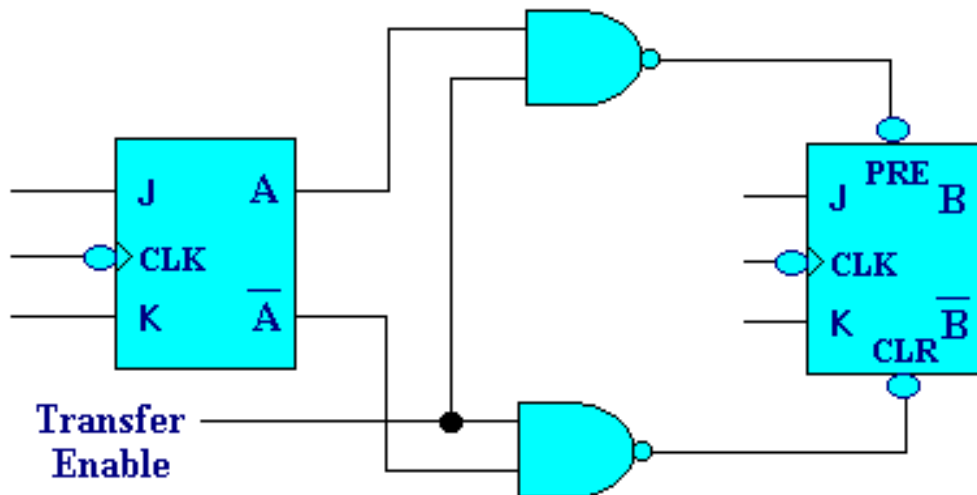
Asynchronous transfer is accomplished using the FF asynchronous inputs (PRE & CLR).



Shown above is an example of a 1-bit synchronous transfer.

Data from A is copied over to B on the falling edge of the transfer clock.

Asynchronous data transfer operation



In asynchronous transfer, the asynchronous inputs PRE and CLR are used to perform the transfer.

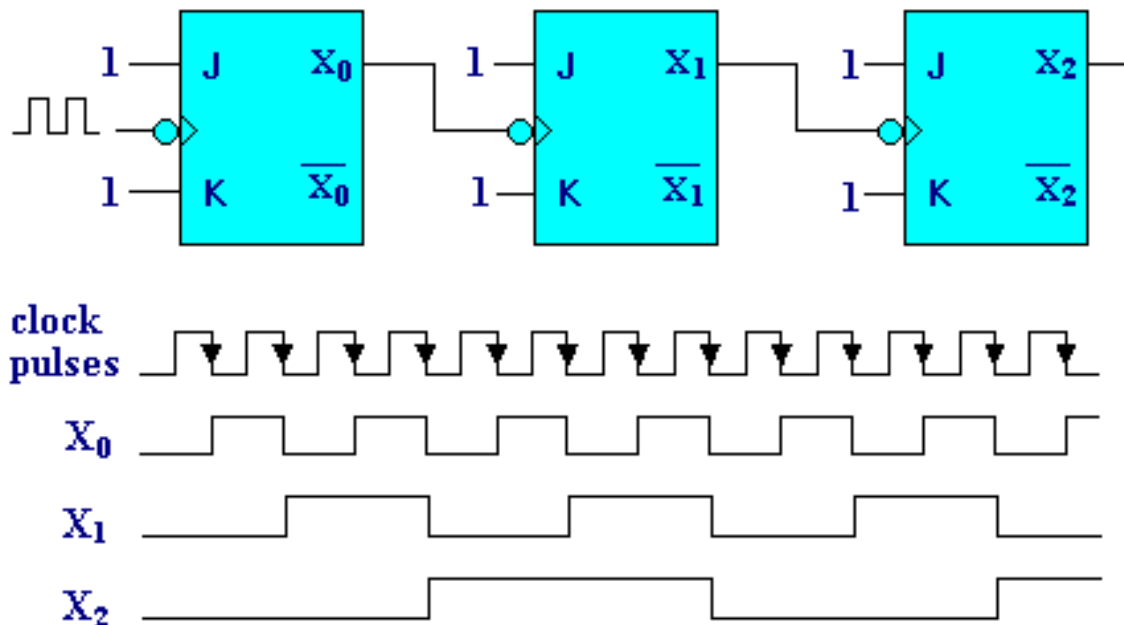
When the transfer enable is H, the 2 NAND gates will be enabled and the data at A will activate either PRE or CLR in the B flip-flop.

E.g. *With Enable = H and A = H, PRE at flip-flop B will be L or active, while CLR will be H or in active. Therefore B goes H, effectively copying the value of A.*

Conversely if A = L with Enable = H, CLR at flip-flop B will be L or active and PRE will be H or in active.

Frequency division and counting

Most common application example on the use of flip-flops is counters.



An example is a cascaded connection of 3 toggle J-K flip-flops wired as a 3-bit binary counter (MOD-8).

Each FF divides its input frequency by 2.

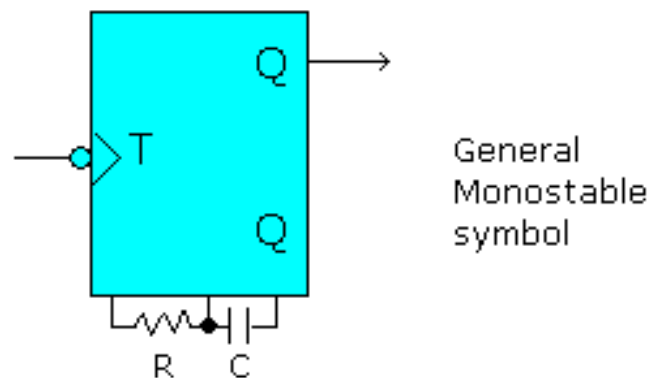
2^2	2^1	2^0	
X_2	X_1	X_0	
0	0	0	Before applying clock pulses
0	0	1	After pulse #1
0	1	0	After pulse #2
1	0	0	After pulse #3
1	0	0	After pulse #4
1	0	1	After pulse #5
1	1	0	After pulse #6
1	1	1	After pulse #7
0	0	0	After pulse #8 recycles to 000

One - Shot (monostable multivibrators)

A monostable output only has ONE stable state.

When triggered, the output will remain in a *quasi-stable* state for a fixed period of time.

It will remain in the quasi-stable state for a period of time, t_p , which is determined by an external CR network.



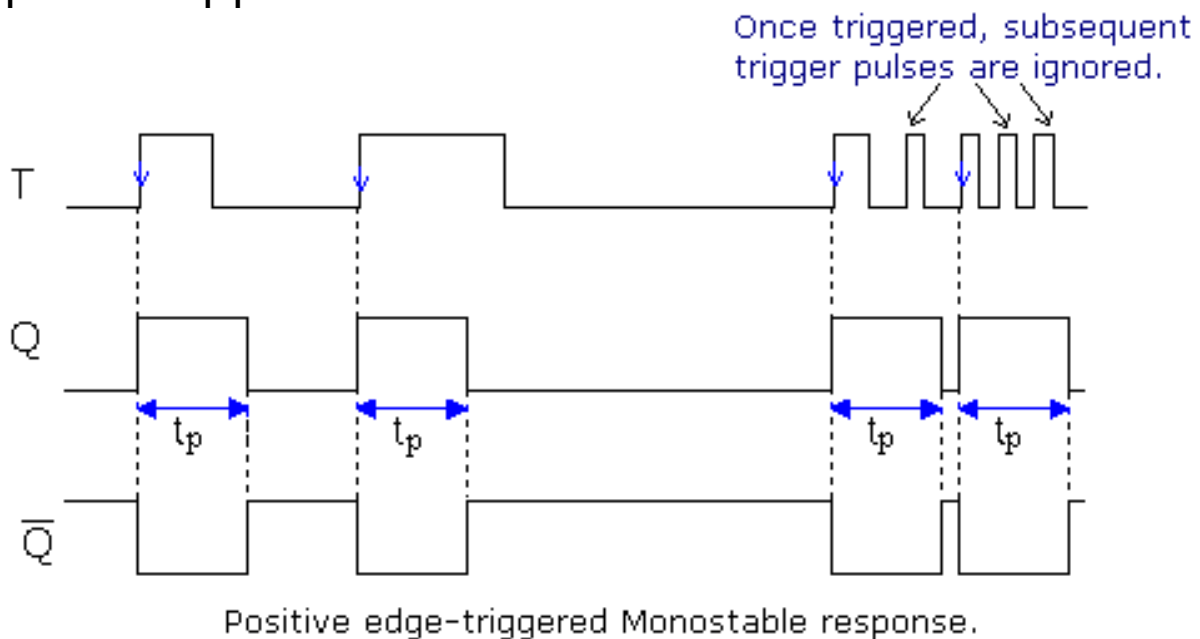
All monostables are edge-triggered devices.

There are two types of One-Shot or monostable multivibrators: -

- a) Non-retriggerable
- b) Retriggerable

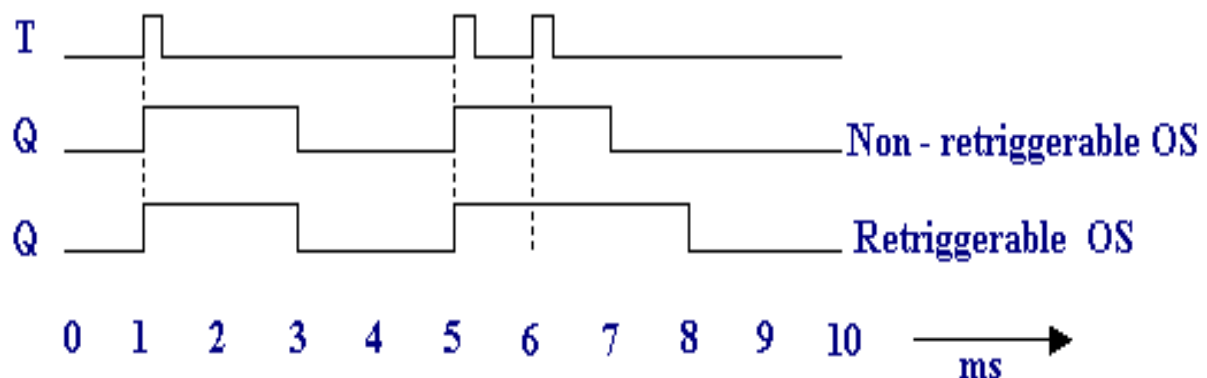
Non-retriggerable One-shot

When the output goes into its triggered state, i.e. within t_p seconds, any subsequent triggered pulses applied has no effects.



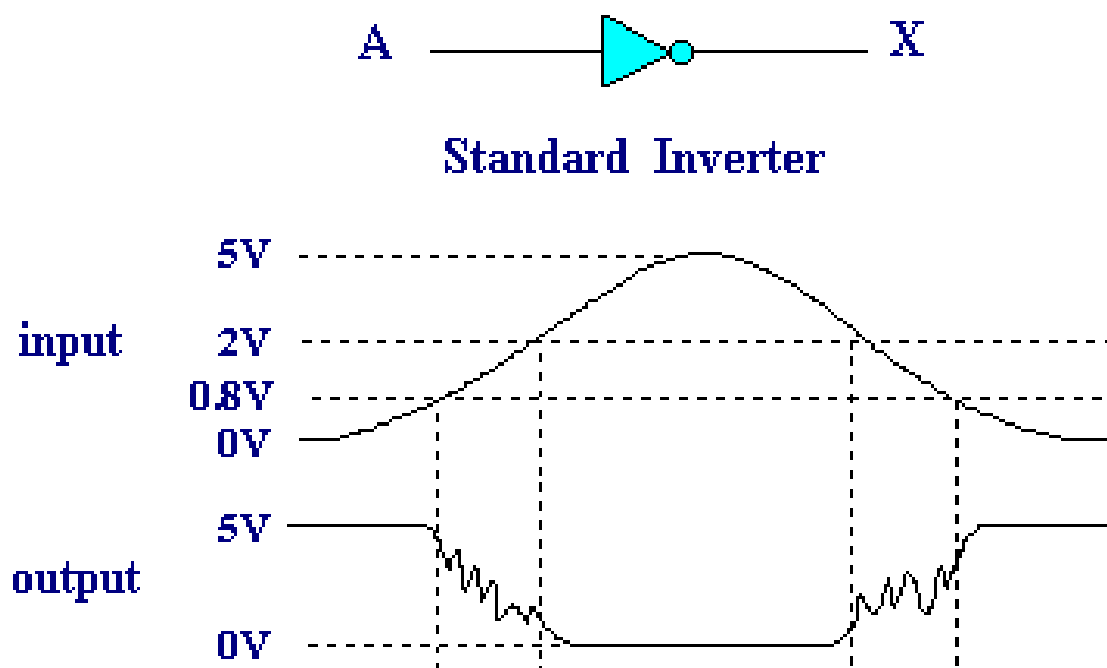
Retriggerable One-shot

Similar to non-retriggerable OS except that it can be *retriggered* (for a further t_p seconds) while it is in the quasi-stable state.



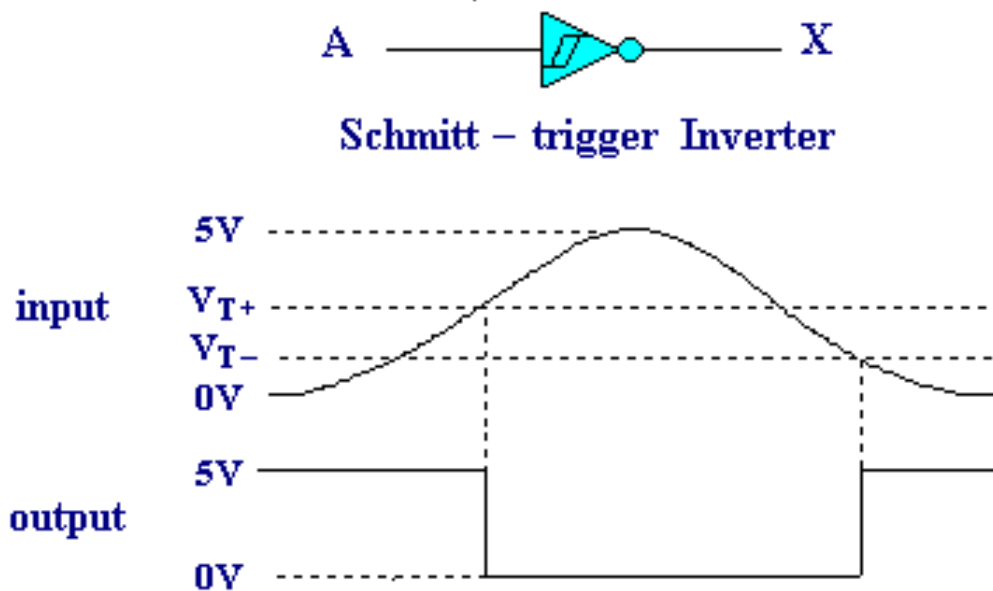
Schmitt trigger is not a FF, but does exhibit a type of memory characteristic.

When a signal with a slow transition time is applied to a normal gate or FF i/p it may cause oscillations at the o/p as the signal passes through the intermediate region.



Oscillations may occur on an output if input transition times are too long.

A device with a *Schmitt trigger* i/p can accept slow changing signals and produce an o/p that is oscillation free.



The o/p rise and fall times are usually very fast (typically $<10\text{ns}$) and are independent of i/p signal characteristics.

There is no spurious oscillation in the output waveform.

Output has clean, fast transitions independent of input transition times.

Astable Multivibrators

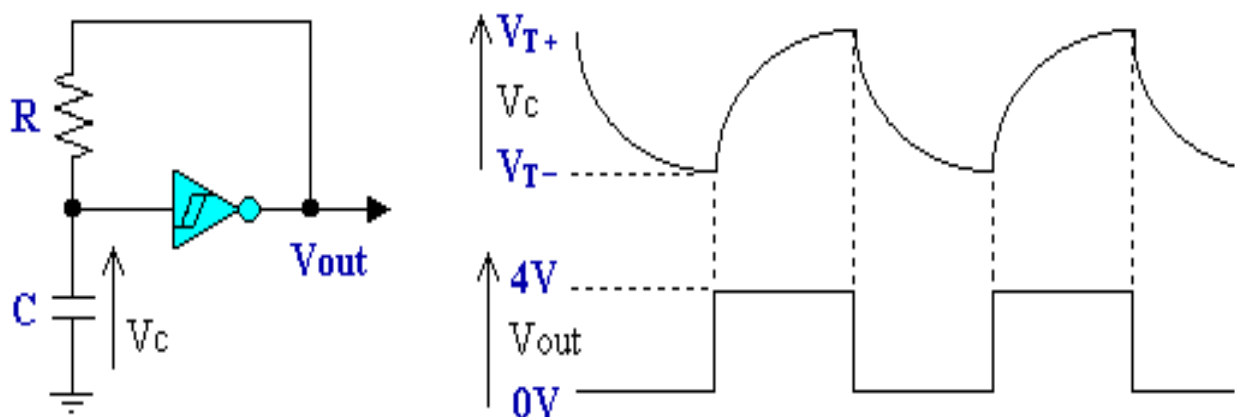
The output has 2 unstable states.

The output switches back and forth between the unstable output states.

This process is called *oscillations*.

The circuit is often referred to an oscillator.

These types of circuits are often used to provide clock signals for synchronous digital circuits.



IC	Frequency	
7414	$\gg 0.8/RC$	$R \leq 500\Omega$ $C > 100\text{pF}$
74LS14	$\gg 0.8/RC$	$R \leq 2 \text{ K}\Omega$
74HC14	$\gg 1.2/RC$	$R \leq 10 \text{ M}\Omega$