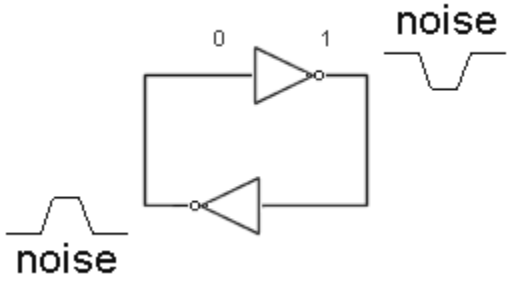
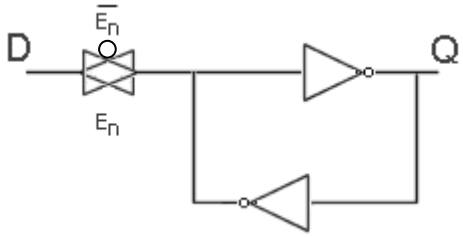
CMOS Sequential Circuits

- **Static CMOS Latch:**
  - A basic memory element (2 back-to-back CMOS inverters)



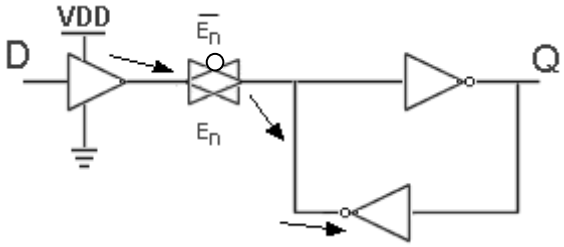
Back to Back Inverters will hold data as long as there is a supply voltage. Also will resist noise

  - To add writing capability, we use:



- when **En** = 1, **Q** = **D**
- when **En** = 0, **Q** stays the same

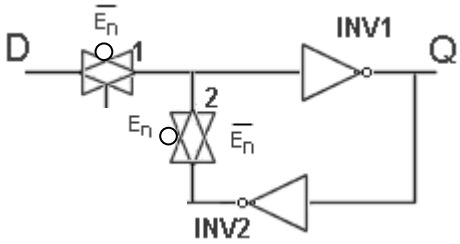
⇨ We get contention between the gate trying to write the new data and the backward inverter which is trying to retain the old data:



⇨ To eliminate the contention, we add a 2nd transmission gate to break the feedback loop during writing:

- When **En** = 0 → **TG_1** is OFF & **TG_2** is ON, **INV_1** & **INV_2** are connected back to back, **Q** stay the same.

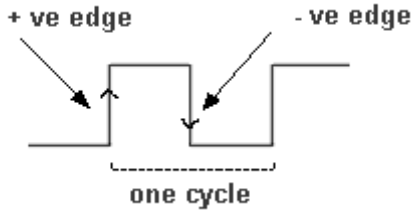
- When **En** = 1 → **TG_1** is ON & **TG_2** is OFF, **Q** = **D**, and no contention since the O/P of **INV_2** is isolated from the I/P



⇨ This is a transparent, level sensitive latch; i.e. it is enabled by the level of the signal En:
If **En** = 1, **Q** "sees" **D** ; whatever happens to **D** is transferred to **Q**

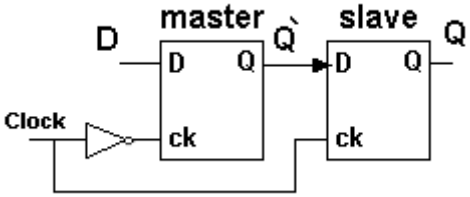
⇨ So, **Latches** cannot be used in **Synchronous Circuits, since they will cause several state transitions per clock cycle**.

↳ What we need is a memory element that is **Edge-Trigged**, since the **clock** has only one positive edge & one negative edge per cycle:



→ So, what we need is an Edge-**Trigged Flip Flop (FF).**

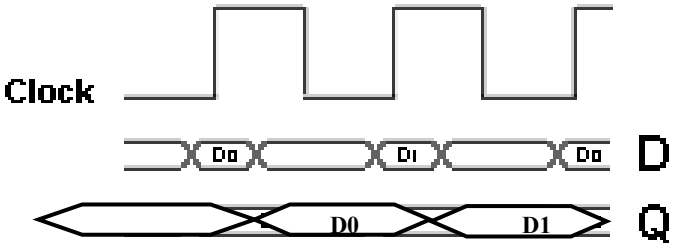
- The edge trigged effect could be obtained by using two **Latches** connected as **Master & Slave**:



- When **CK** = 0, **Master** is enabled while **Slave** is disabled,
Q stays the same ⇨ **D → Q'**

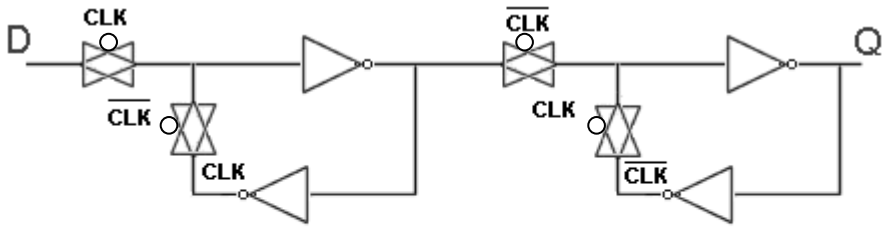
- As soon as **CK** = 1, **Master** is disabled while **Slave** is enabled,
Q' stays constant and **Q ← D** this happens with the clock edge (since D is already at Q'), so it appears that Q is edge-triggered. Q will change only once per clock period.

o Time Diagram:



❖ here **Q** "never" sees D → no transparency

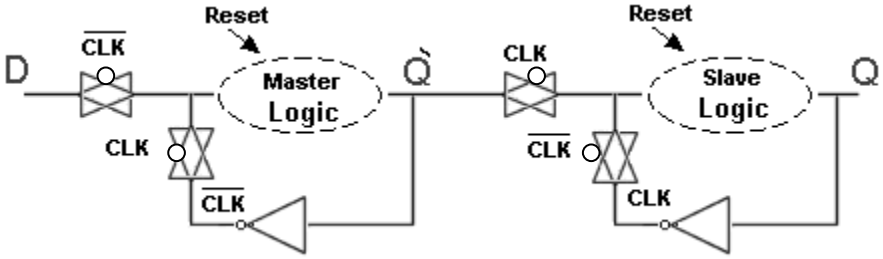
- **The Edge-Trigged FF:**
  - **The Positive Edge-Trigged**:



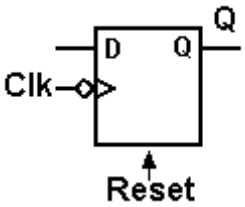
For negative edge-triggered Just change $\overline{CLK} \rightarrow CLK$

$CLK \rightarrow \overline{CLK}$

- **FFs with Direct (i.e. asynchronous) Inputs:**
  - Direct I/Ps can be added by replacing the forward inverters in **Master & Slave**, by inverting logic that gives us the function we want.



  - In a **Negative Edge-Trigged FF** with **Active Low Direct Reset:**



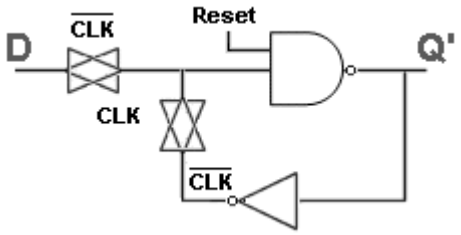
CLK	Reset	Q+
X	0	0
1	1	Q
↓	1	D

❖ to **Reset a FF**, we need to **Reset the Slave** & **Set the Master**.

- **Master Logic:**

D	Reset	Q'	
X	0	1	→ when Reset is active, Q' = 1 ~ whatever D is
0	1	1	} → when Reset is inactive, it works like an inverter
1	1	0	

$$Q' = \overline{D} + \overline{Reset} = \overline{D.\,Reset}$$



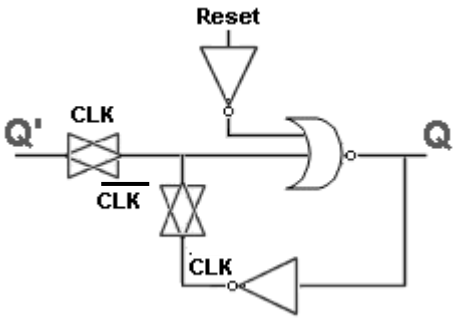
- **Slave Logic:**

Q'	Reset	Q
X	0	0
0	1	1
1	1	0

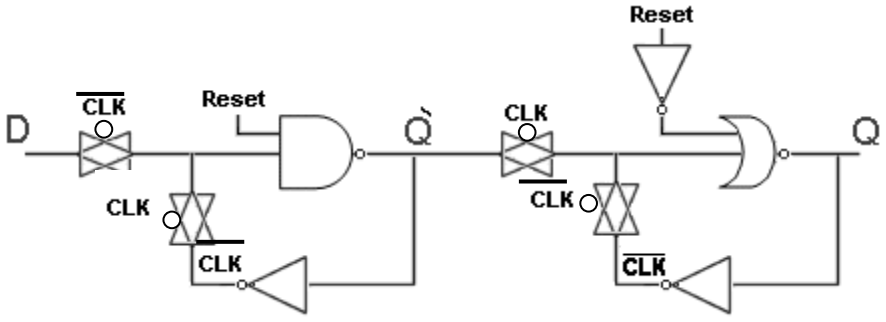
acts like an inverter

$$Q = \overline{Q'}.Reset$$

$$Q = \overline{Q' + \overline{Reset}}$$



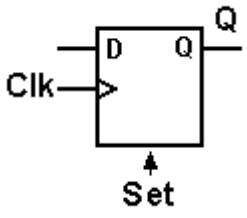
- **The final solution:**



Example 1:

Implement a **Positive Edge-Trigged FF** with **Active High Direct Set**

Sol:

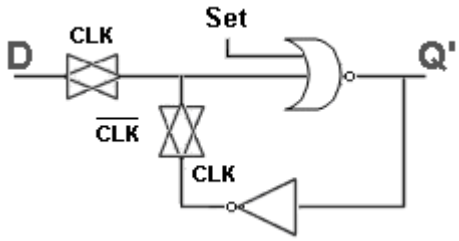


CLK	Set	Q+
X	1	1
0	0	Q
↓	0	D

- **Master Logic:**

D	Set	Q'
X	1	0
0	0	1
1	0	0

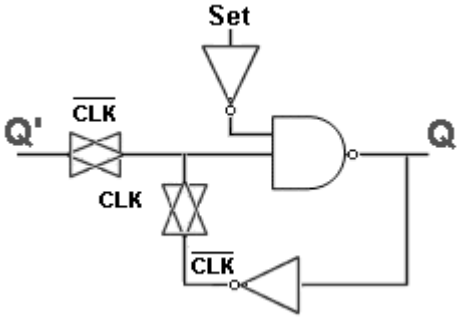
$Q' = \overline{D}.\overline{Set} = \overline{D+Set}$



- **Slave Logic:**

Q'	Set	Q
X	1	1
0	0	1
1	0	0

$Q = \overline{Q'} + Set = \overline{Q'.\overline{Set}}$



- **The final solution:**