

Lecture 11:

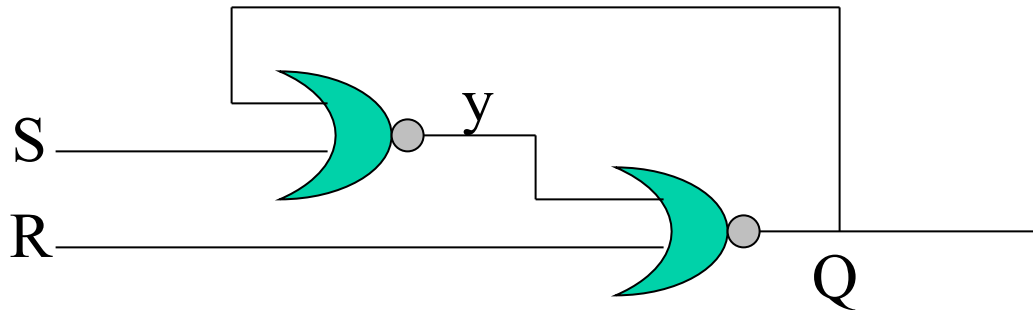
Sequential Networks – Latches and Flip flops

CSE 140: Components and Design Techniques for
Digital Systems

Diba Mirza
Dept. of Computer Science and Engineering
University of California, San Diego

Flip-flop Components

SR latch (Set-Reset)



Inputs: S, R

State: (Q, y)

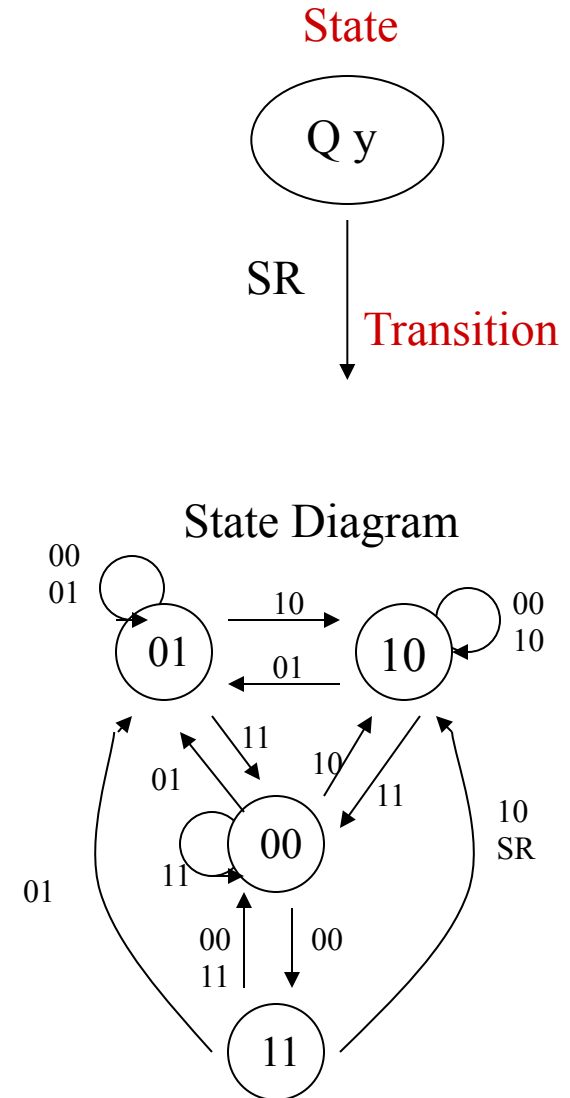
SR=01, (Q,y) = (0,1)

SR=10, (Q,y) = (1,0)

SR=11, (Q,y) = (0,0)

SR = 00 => if (Q,y) = (0,0) or (1,1), the output keeps toggling

Id	Q(t)	y(t)	S	R	Q(t ₁)	y(t ₁)	Q(t ₂)	y(t ₂)	Q(t ₃)	y(t ₃)
0	0	0	0	0	1	1	0	0	1	1
1	0	0	0	1	0	1	0	1	0	1
2	0	0	1	0	1	0	1	0	1	0
3	0	0	1	1	0	0	0	0	0	0
4	0	1	0	0	0	1	0	1	0	1
5	0	1	0	1	0	1	0	1	0	1
6	0	1	1	0	0	0	1	0	1	0
7	0	1	1	1	0	0	0	0	0	0
8	1	0	0	0	1	0	1	0	1	0
9	1	0	0	1	0	0	0	1	0	1
10	1	0	1	0	1	0	1	0	1	0
11	1	0	1	1	0	0	0	0	0	0
12	1	1	0	0	0	0	1	1	0	0
13	1	1	0	1	0	0	0	1	0	1
14	1	1	1	0	0	0	1	0	1	0
15	1	1	1	1	0	0	0	0	0	0



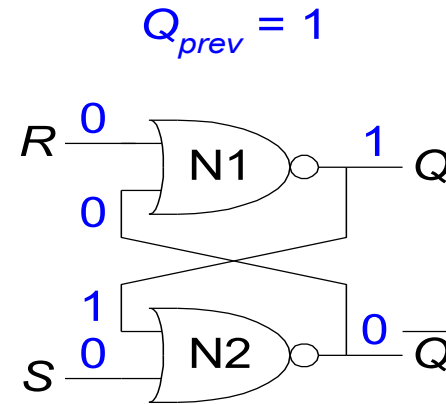
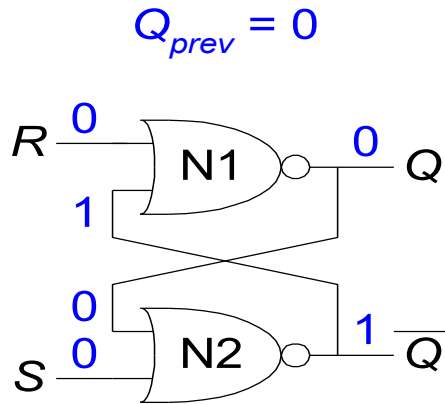
Q: Which of the following is a good solution to avoid the output from toggling?

A) Avoid the input $SR = (0,0)$

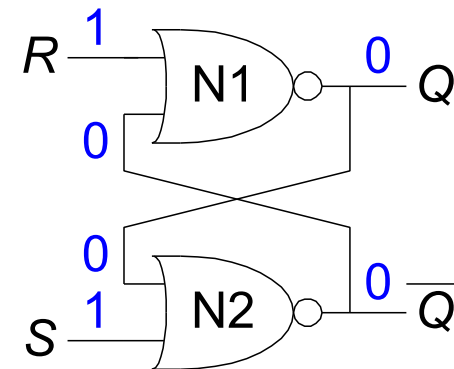
B) Avoid the input $SR = (1,1)$

SR Latch Analysis

- $S = 0, R = 0$: then $Q = Q_{prev}$ and $\bar{Q} = \bar{Q}_{prev}$ (**memory!**)



- $S = 1, R = 1$: then $Q = 0$ and $\bar{Q} = 0$ (**invalid state: $Q \neq \text{NOT } \bar{Q}$**)

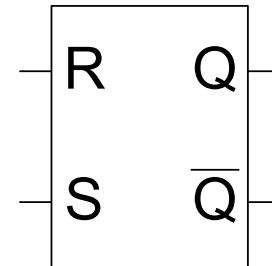


State table

inputs		SR			
		00	01	11	10
PS	Q(t)	0	0	X	1
	1	1	0	X	1

↗

SR Latch Symbol



$Q(t+1)$ NS (next state)

Characteristic Expression

$$Q(t+1) = S(t) + R'(t)Q(t)$$

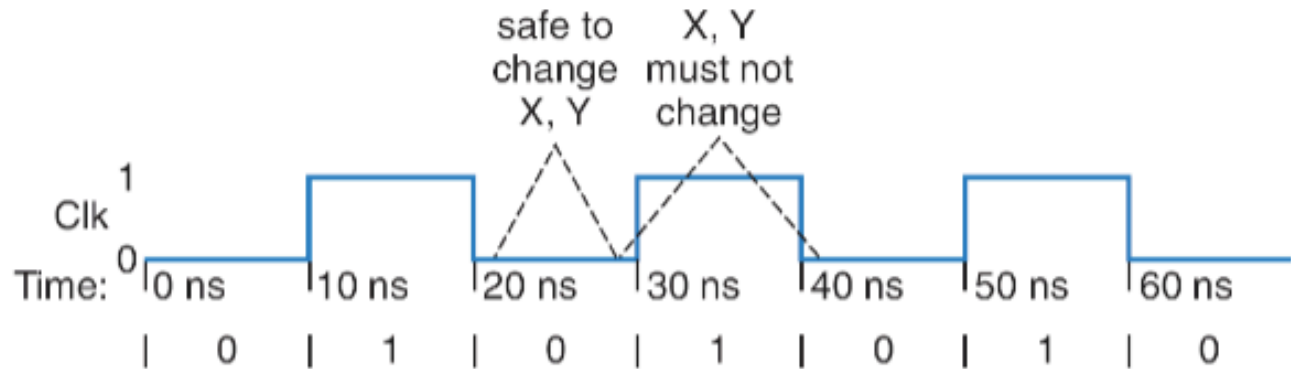
SR Latch Symbol

- 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 ($S = R = 1$)

Clocks

Freq	Period
100 GHz	0.01 ns
10 GHz	0.1 ns
1 GHz	1 ns
100 MHz	10 ns
10 MHz	100 ns



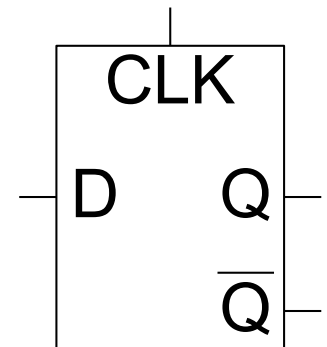
- **Clock** -- Pulsing signal for enabling latches; ticks like a clock
- **Synchronous** circuit: sequential circuit with a clock
- **Clock period**: time btwn pulse starts
 - Above signal: period = 20 ns
- **Clock cycle**: one such time interval
 - Above signal shows 3.5 clock cycles
- **Clock duty cycle**: time clock is high
 - 50% in this case
- **Clock frequency**: $1/\text{period}$
 - Above : $\text{freq} = 1 / 20\text{ns} = 50\text{MHz}$;

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 (the latch is *transparent*)
 - When $CLK = 0$, Q holds its previous value (the latch is *opaque*)
- Avoids invalid case when $Q \neq \text{NOT } Q$

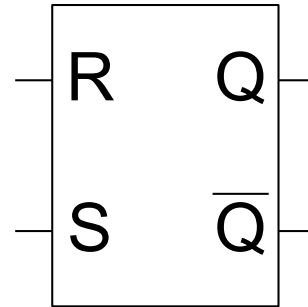


D Latch
Symbol



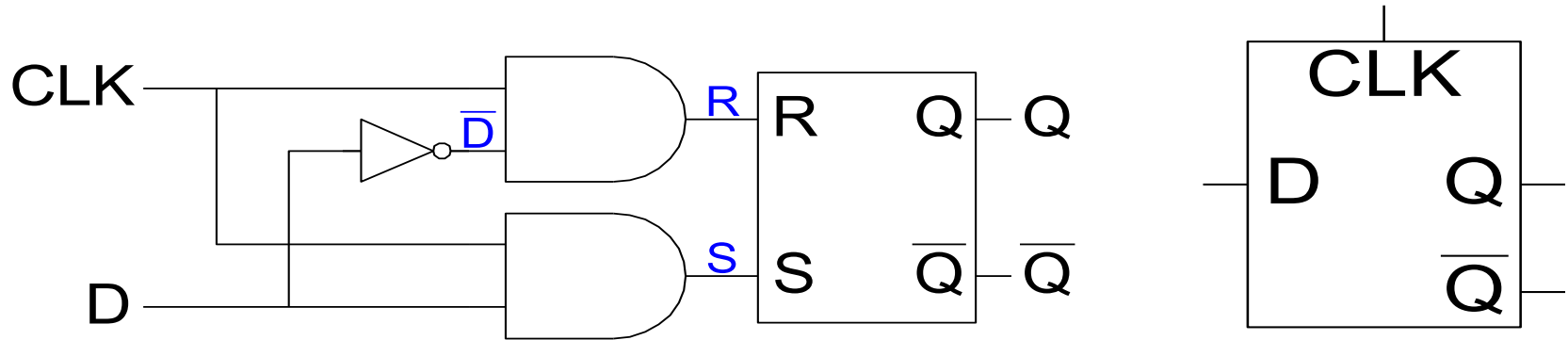
D Latch Internal Circuit

SR Latch



CLK	D	S	R	Q	Q'
0	0				
0	1				
1	0				
1	1				

D Latch Internal Circuit



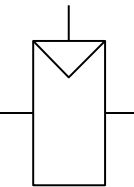
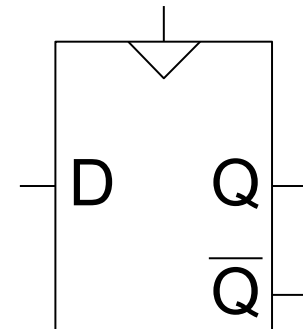
CLK	D	$\overline{D}$	S	R	Q	$\overline{Q}$
0	X	$\overline{X}$	0	0	Q_{prev}	$\overline{Q}_{prev}$
1	0	1	0	1	0	1
1	1	0	1	0	1	0

D Flip-Flop

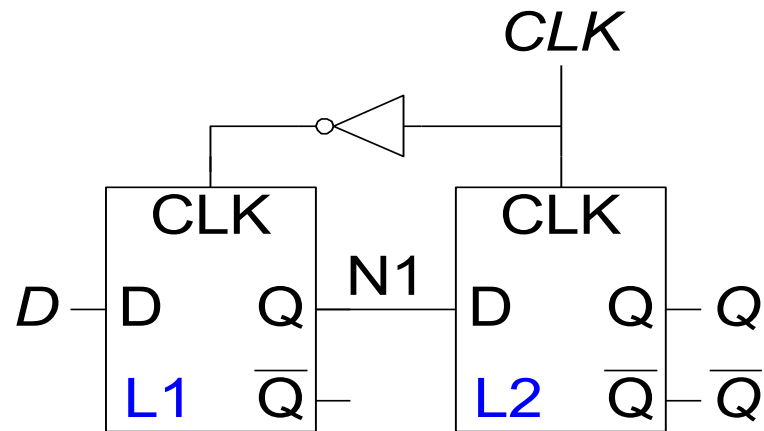
- Two inputs: CLK , D
- *Function*
 - The flip-flop “samples” D on the rising edge of CLK
 - When CLK rises from 0 to 1, D passes through to Q
 - Otherwise, Q holds its previous value
 - Q changes only on the rising edge of CLK
- A flip-flop is an *edge-triggered* device because it is activated on the clock edge (when CLK rises from 0 to 1)



D Flip-Flop
Symbols



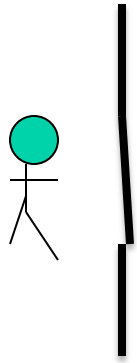
D Flip-Flop Internal Circuit



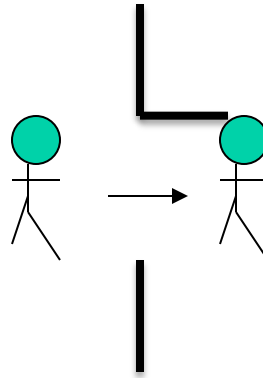
- When $CLK = 0$
 - L1 is transparent, L2 is opaque
 - D passes through to N1
- When $CLK = 1$
 - L2 is transparent, L1 is opaque
 - N1 passes through to Q

Latch and Flip-flop (two latches)

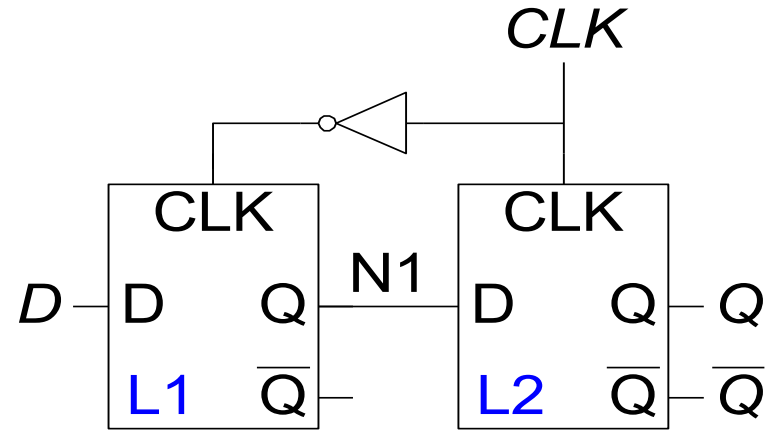
A latch can be considered as a door



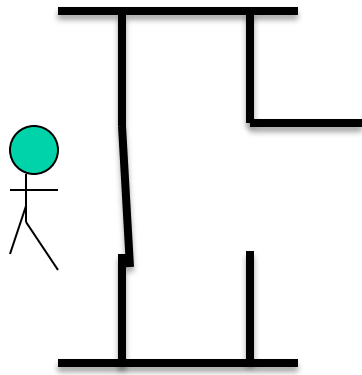
$CLK = 0$, door is shut



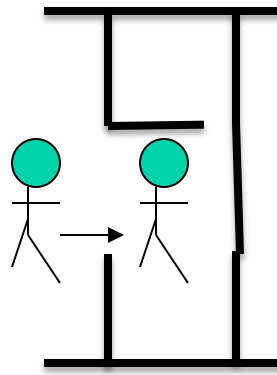
$CLK = 1$, door is unlocked



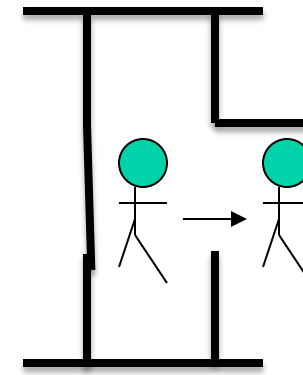
A flip-flop is a two door entrance



$CLK = 1$

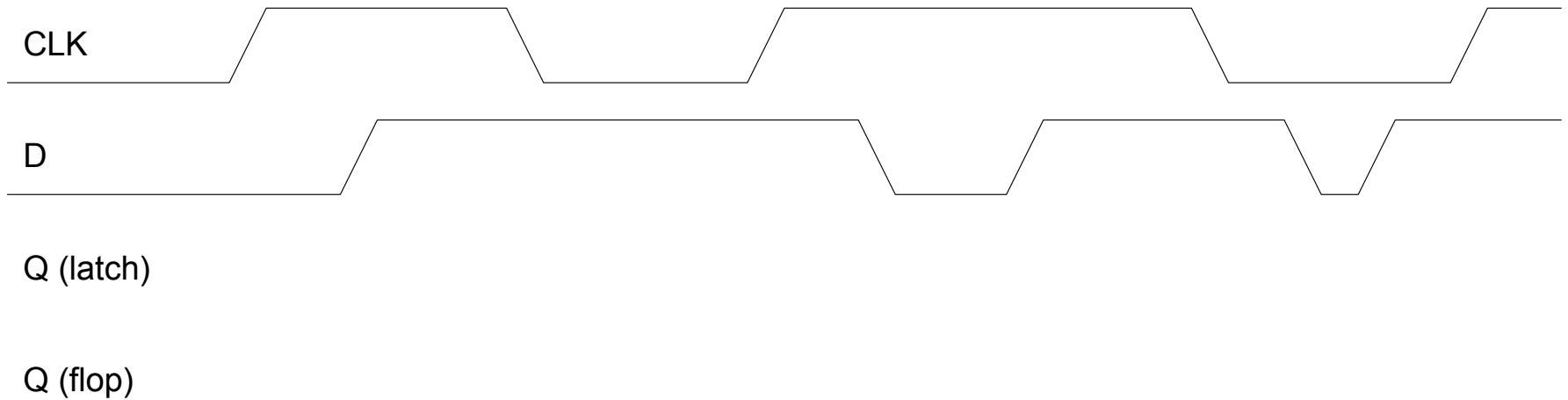
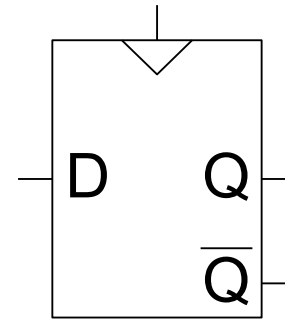
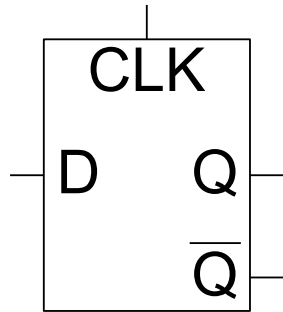


$CLK = 0$

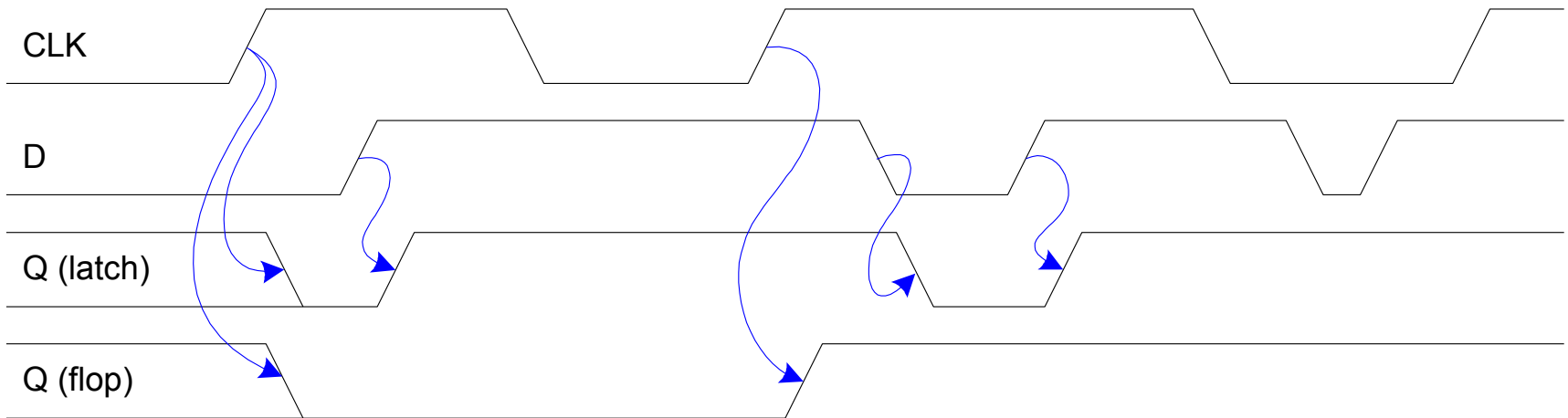
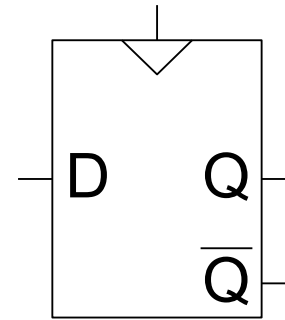
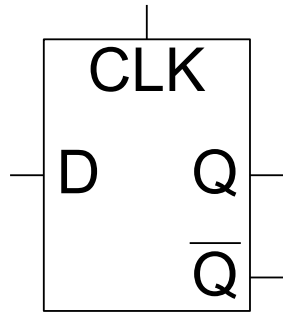


$CLK = 1$

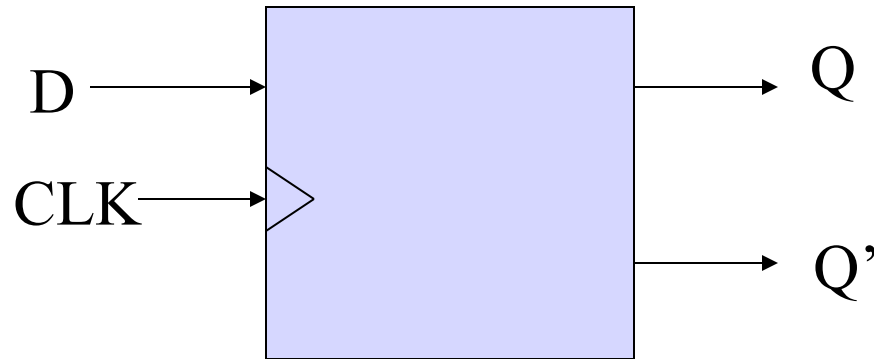
D Flip-Flop vs. D Latch



D Flip-Flop vs. D Latch



D Flip-Flop (Delay)



Id	D	Q(t)	Q(t+1)
0	0	0	0
1	0	1	0
2	1	0	1
3	1	1	1

State table

PS \ D	0	1
0	0	1
1	0	1

NS = Q(t+1)

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

What does the equation mean?



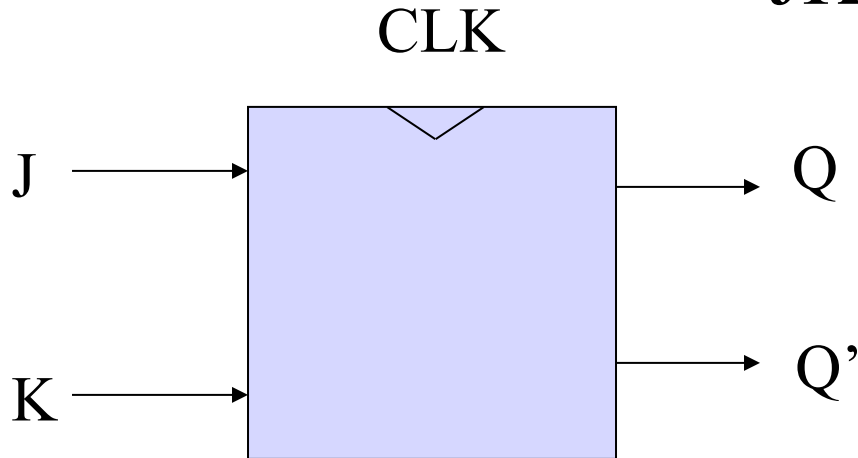
iClicker

Can D flip-flop serve as a memory component?

A. Yes

B. No

JK F-F



State table

PS \ JK	JK			
	00	01	11	10
0	0	0	1	1
1	1	0	0	1

↖ Q(t+1)

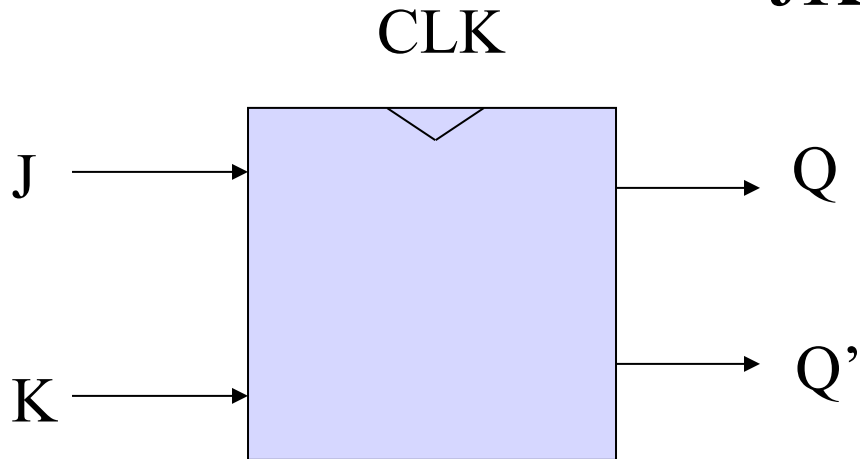
Sounds a lot like a
latch I know...



Characteristic Expression
 $Q(t+1) = Q(t)K'(t) + Q'(t)J(t)$

JK F-F

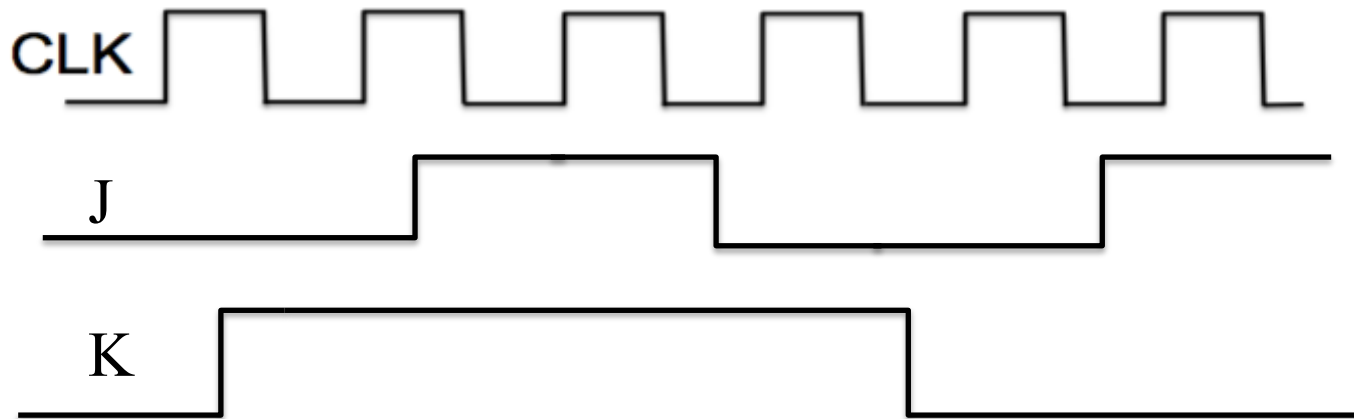
Characteristic Expression
 $Q(t+1) = Q(t)K'(t) + Q'(t)J(t)$



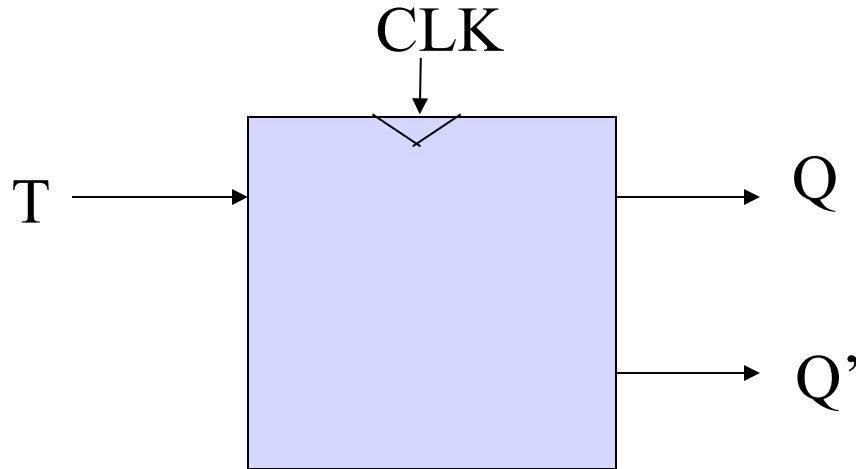
State table

PS \ JK		JK			
		00	01	11	10
0	0	0	0	1	1
	1	1	0	0	1

Q(t+1)



T Flip-Flop (Toggle)



State table

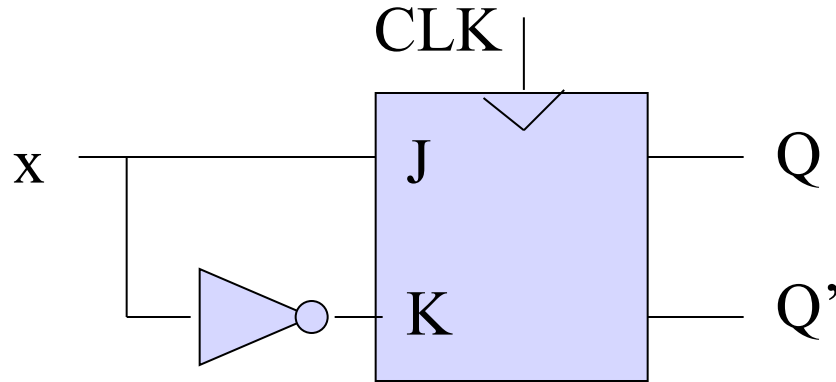
PS \ T	0	1
0	0	1
1	1	0

Q(t+1)

Characteristic Expression

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

Using a JK F-F to implement a D and T F-F

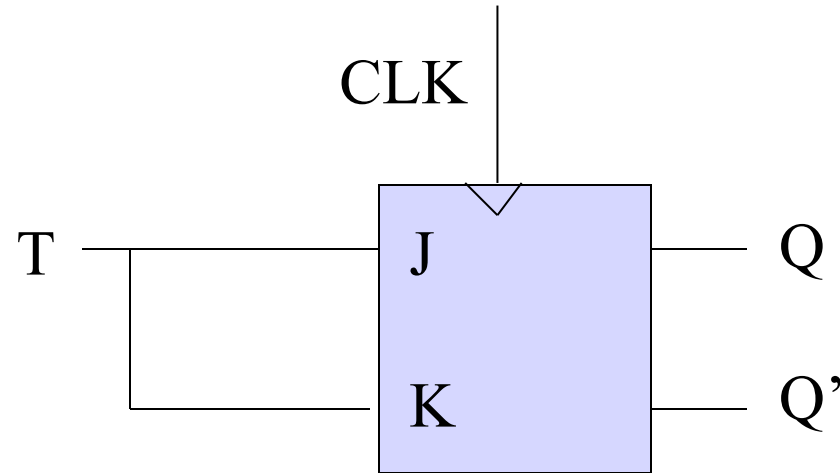


iClicker

The above circuit behaves as which of the following flip flops?

- A. D F-F
- B. T F-F
- C. None of the above

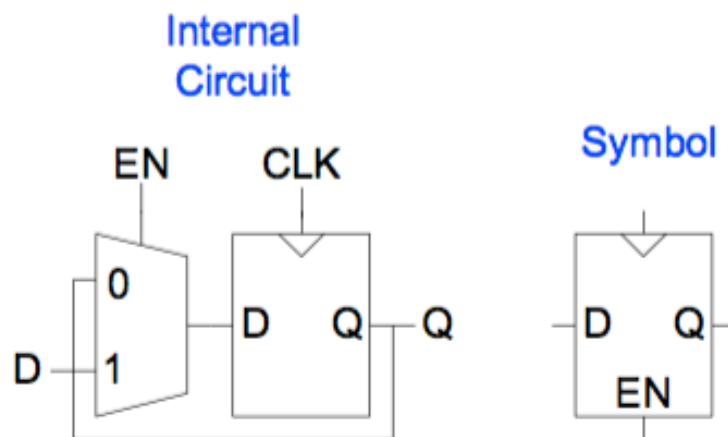
Using a JK F-F to implement a D and T F-F



T flip flop

Enabled Flip-Flops

- **Inputs:** CLK , D , EN
 - The enable input (EN) controls when new data (D) is stored
- **Function**
 - $EN = 1$: D passes through to Q on the clock edge
 - $EN = 0$: the flip-flop retains its previous state



Resettable Flip-Flops

Symbols

- **Inputs:** CLK , D , $Reset$
- **Function:**
 - **$Reset = 1$:** Q is forced to 0
 - **$Reset = 0$:** flip-flop behaves as ordinary D flip-flop
- Two types:
 - **Synchronous:** resets at the clock edge only
 - **Asynchronous:** resets immediately when $Reset = 1$
- Asynchronously resettable flip-flop requires changing the internal circuitry of the flip-flop
- Synchronously resettable flip-flop circuit:
- There are also synch/asynch settable FFs

