



Iran University of Science & Technology
IUST

Digital Logic Design

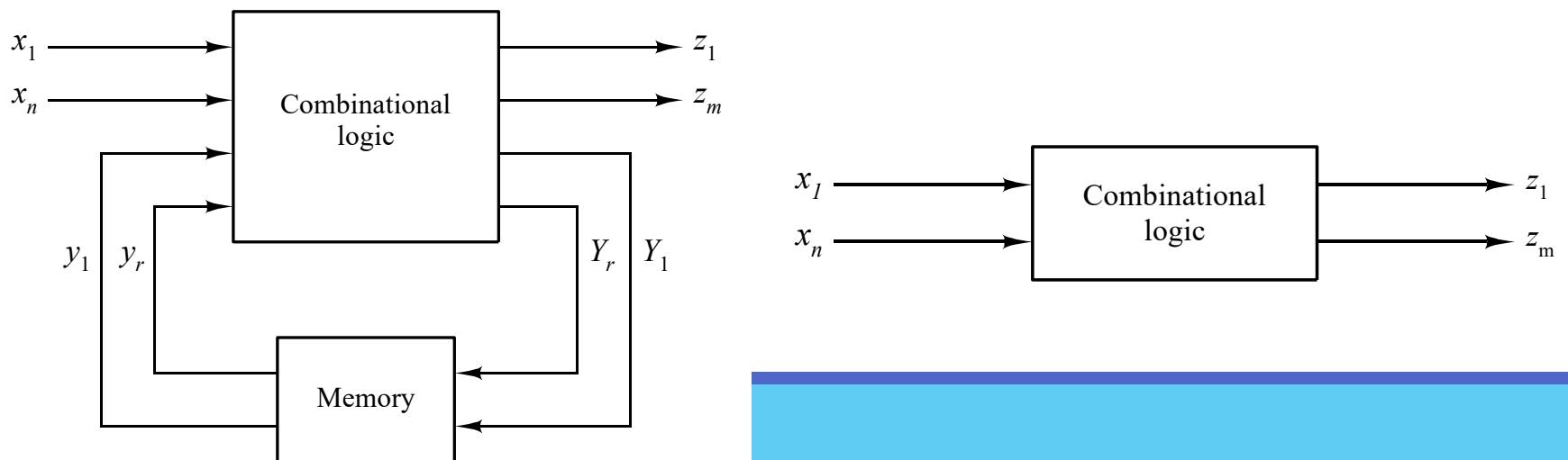
Hajar Falahati

Department of Computer Engineering
IRAN University of Science and Technology

hfalahati@iust.ac.ir

Circuit Logics

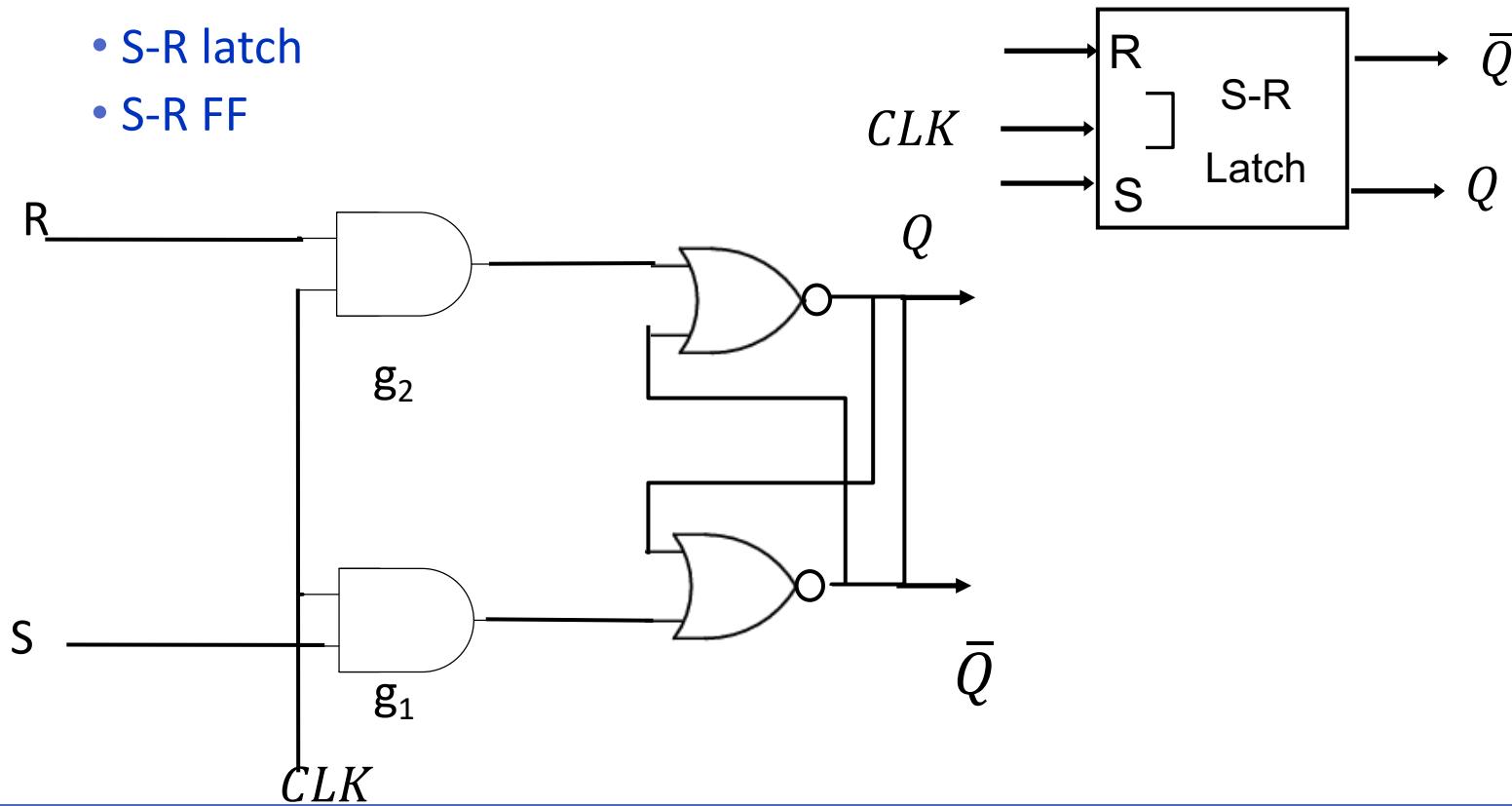
- Combinational logic
 - Always **inputs** produce the **output**
 - Combinational logics are **memory less**
 - Memory-less circuits **do not** contain any **feedback lines**
- Sequential logic
 - Always **inputs** and current state produce the **output**
 - They **have memory**
 - They **contain feedback lines**



Memory Unit

- An element which **stores one bit** data

- S-R latch
- S-R FF



Outline

- Memory devices
 - Flip Flops (FF)



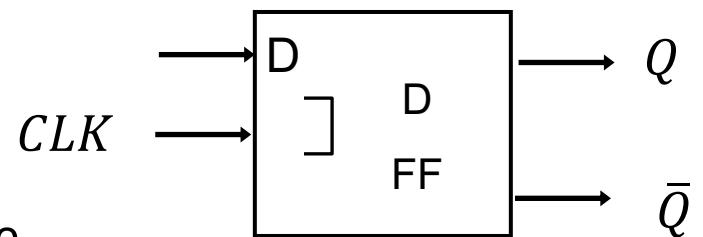
Flip Flops

Delay Flip Flop (D-FF)

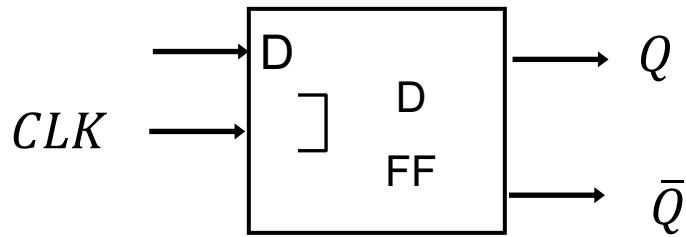
- **Input**
 - D
 - CLK
- **Output**
 - Send D to the output

CLK	D	Q	\bar{Q}
0	X	Q	\bar{Q}
1	0	0	1
1	1	1	0

Hold the value
Send D to Q
Send D to Q



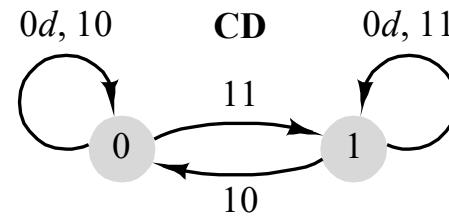
D-FF: Characteristics



Enable input	Excitation input	Present state	Next state	
C	D	Q	Q^*	
0	x	0	0	Hold
0	'	1	1	
1	0	0	0	Store 0
1	0	1	0	
1	1	0	1	Store 1
1	1	1	1	

(a)

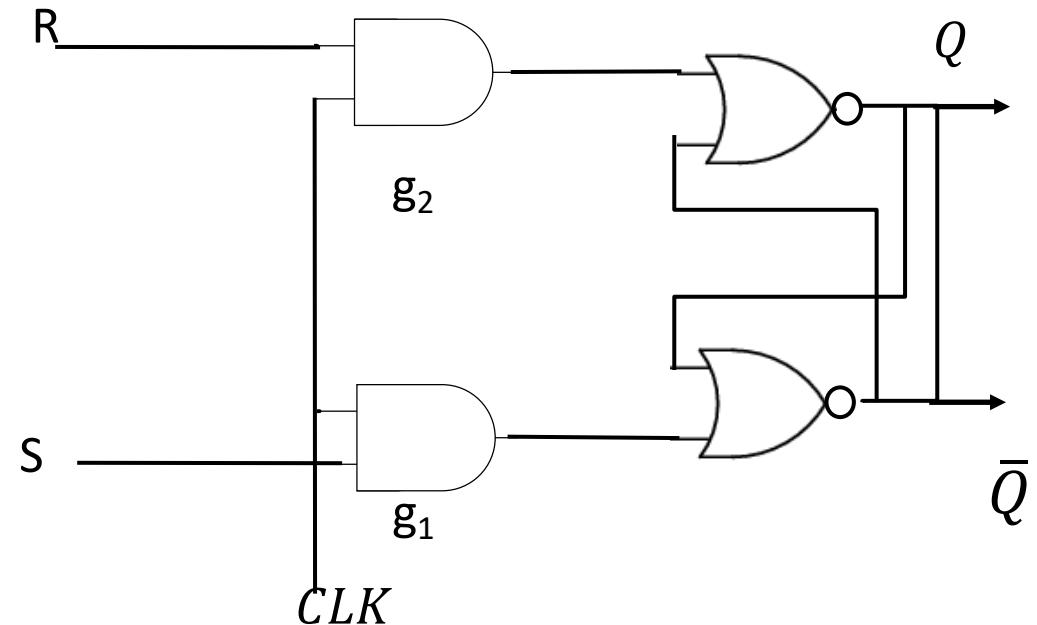
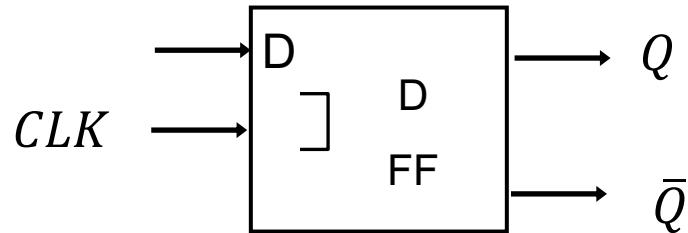
$$Q^* = DC + C'Q$$



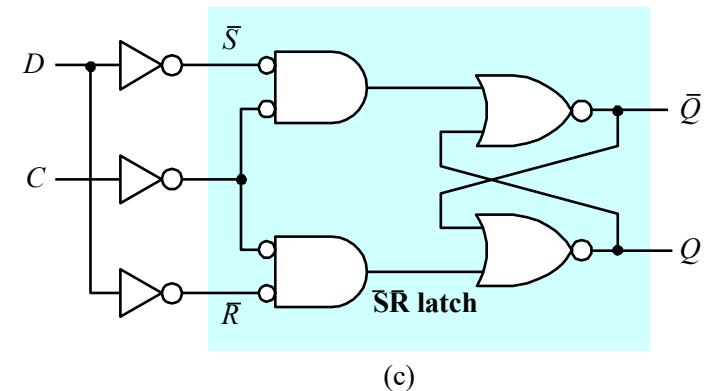
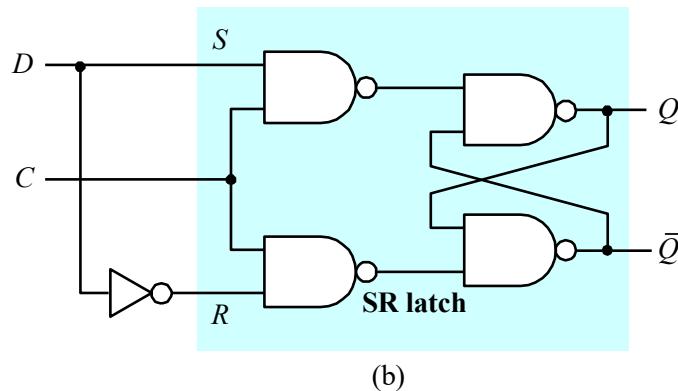
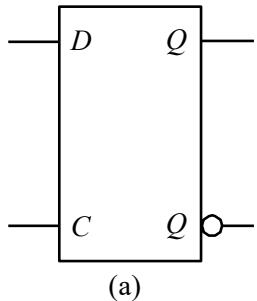
(b)

D-FF: Implementation

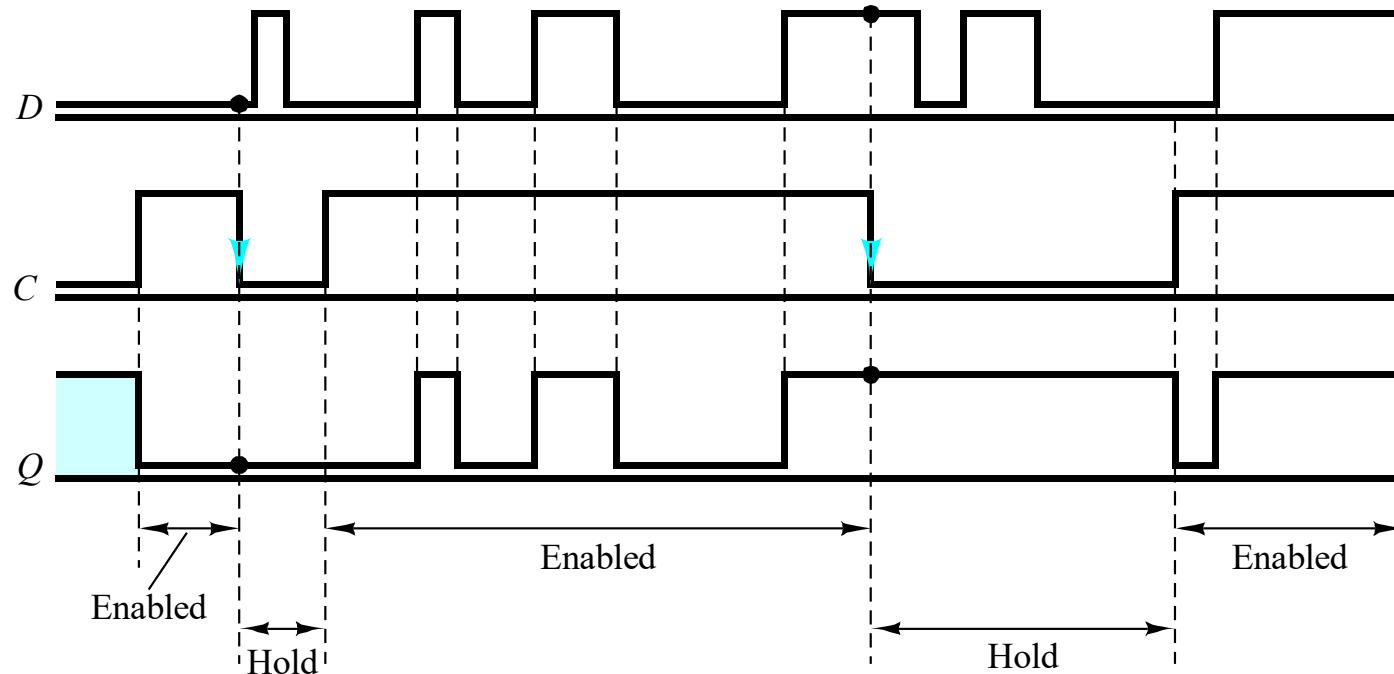
- Using S-R FF



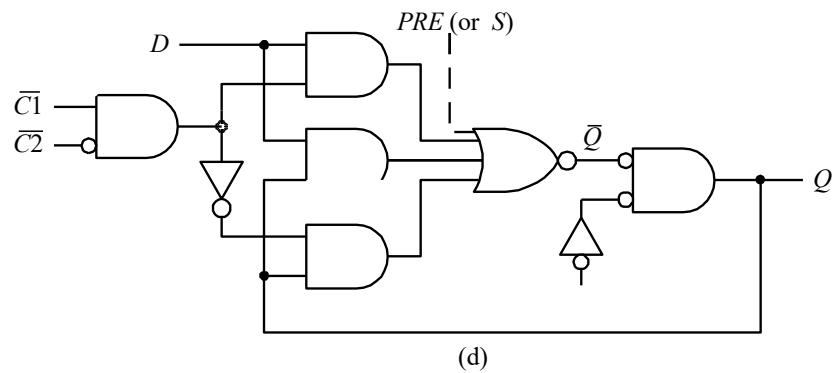
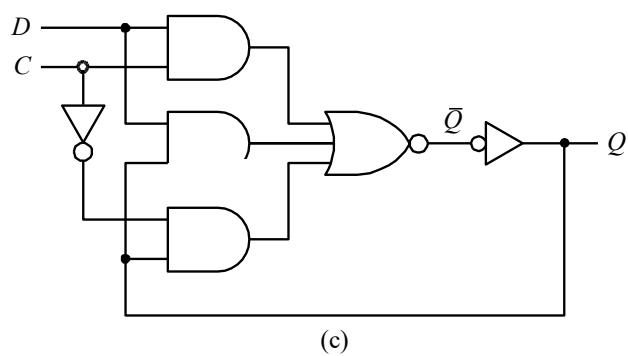
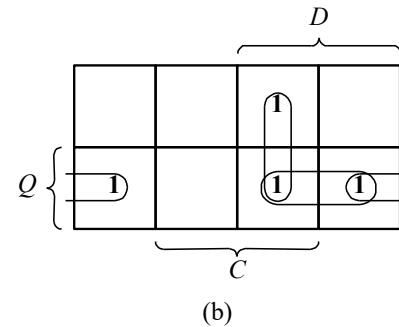
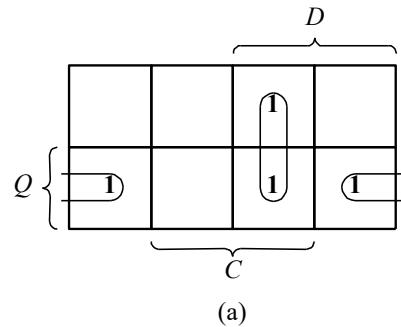
D-FF: Implementation (cont'd)



D-FF: Timing Diagram



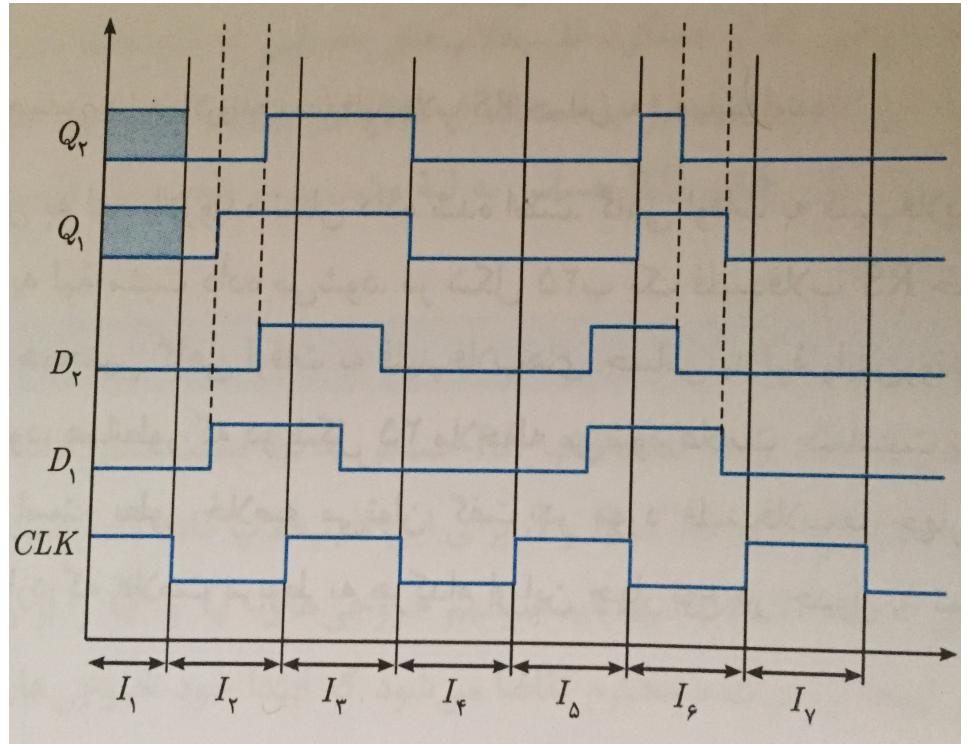
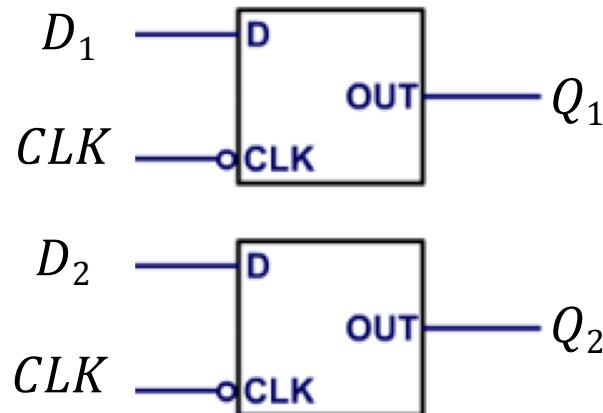
D FF: Hazard-free



$$Q^* = DC + C'Q + DQ$$

Level-Triggered D-FF

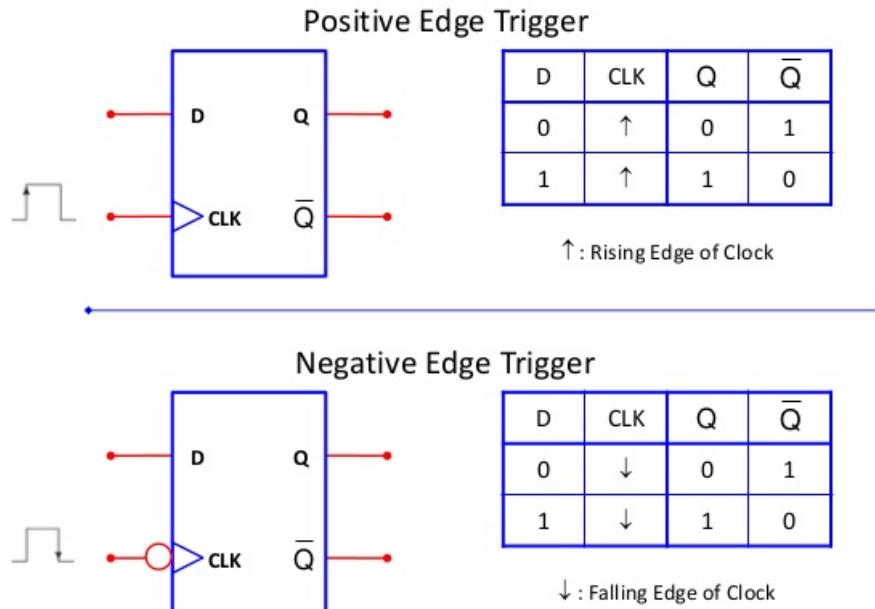
- CLK signal aim to manage changes in latches
- Inputs changes at different times



Edge-Triggered D-FF

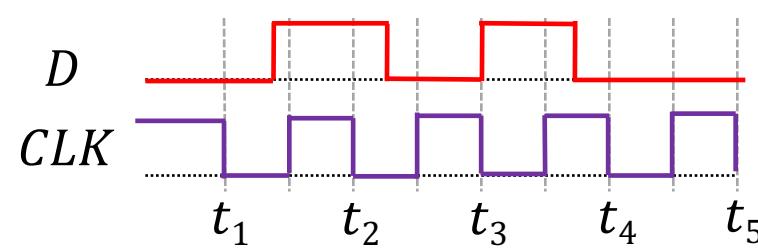
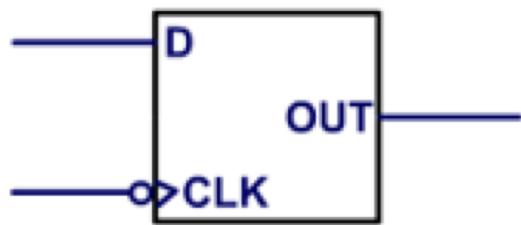
- Output changes only at the positive or negative edge of CLK signal

POS & NEG Edge Triggered **D**

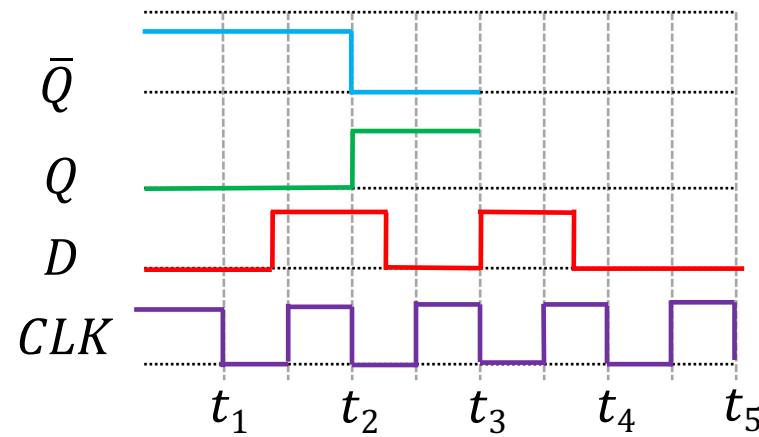
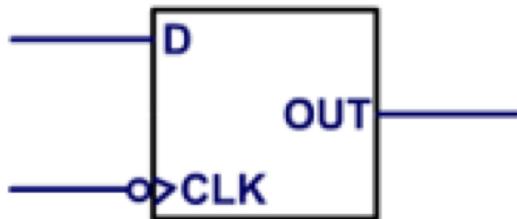


Edge-Triggered D-FF: Timing Analysis

- Determine Q

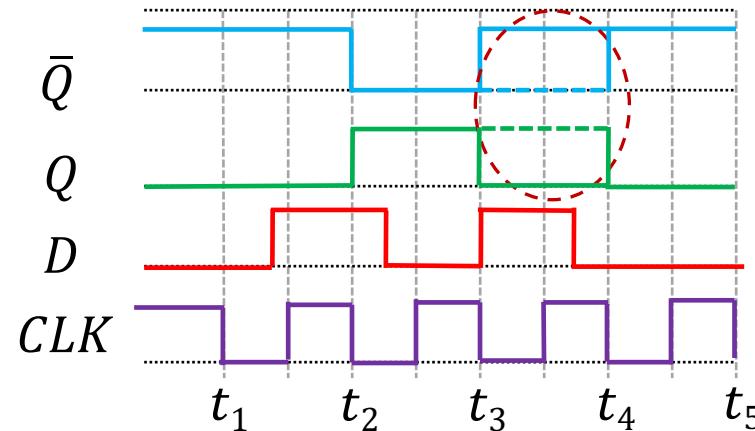
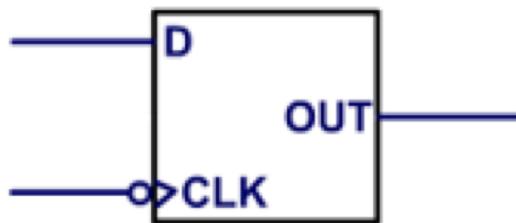


Edge-Triggered D-FF: Timing Analysis (cont'd)



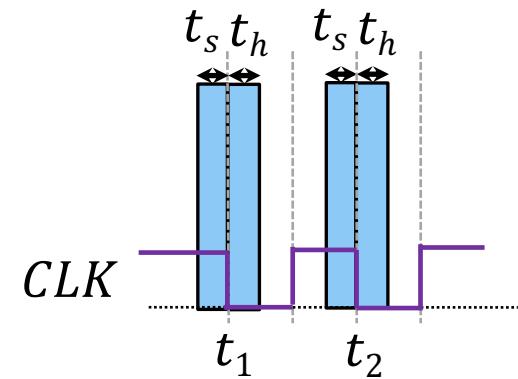
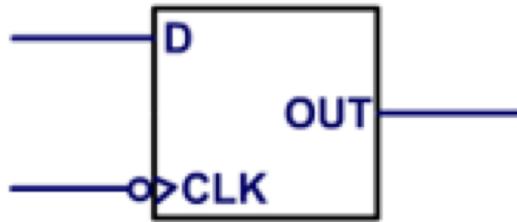
Edge-Triggered D-FF: Timing Analysis (cont'd)

- What happens at t_3 ?
 - There is a race between D and CLK
- D wins
 - FF becomes 1 (dashed lines)
- CLK wins
 - FF becomes 0



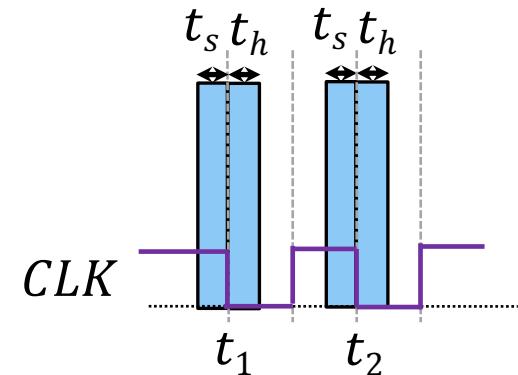
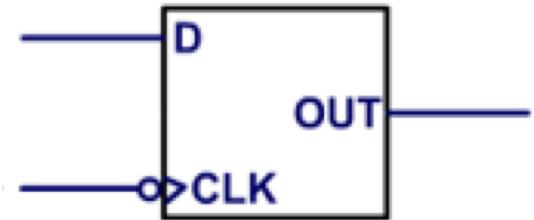
Critical Race

- Overcome the critical race between CLK and input
 - Setup time (t_s)
 - Hold time (t_h)

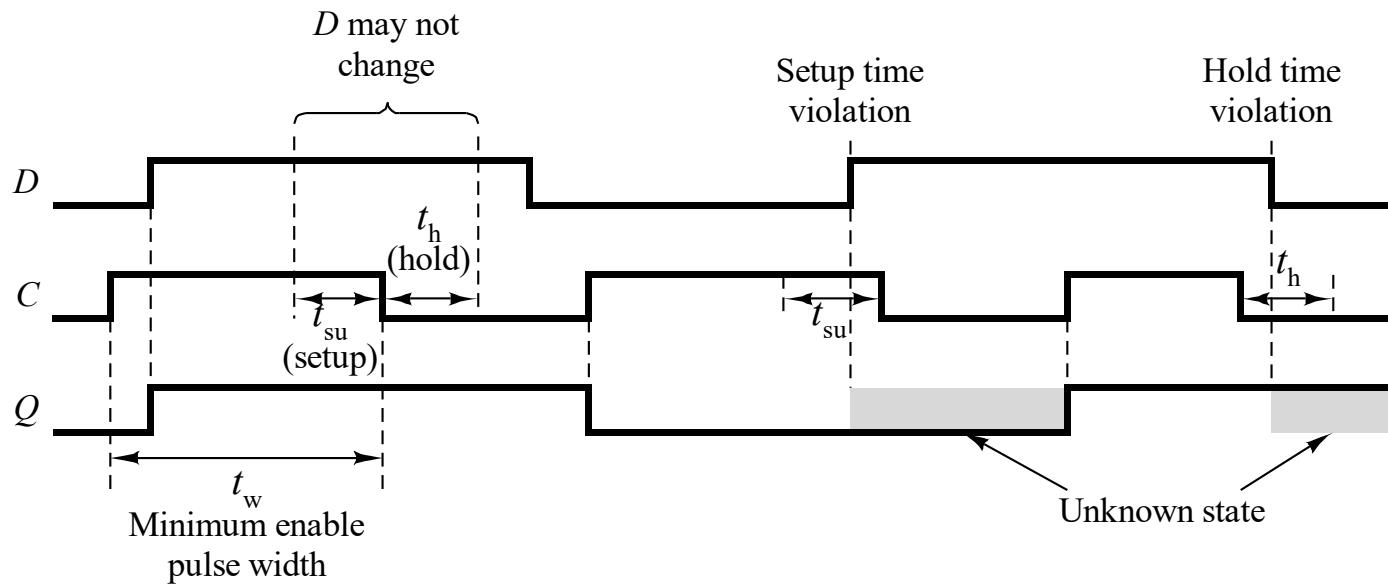


Hold Time and Setup Time

- Setup time (t_s)
 - Minimum time interval
 - Before the sampling event of the clock
 - Input signal must be **stable** (unchanging)
 - To be recognized correctly
- Hold time (t_h)
 - Minimum time interval
 - After the sampling event of the clock
 - Input signal must be **stable** (unchanging)
 - To be recognized correctly



D-FF: Timing Constraints



Reset

- **Reset input**

- Reset the FF value
- $Q = 0$
- A.k.a., clear (CLR)

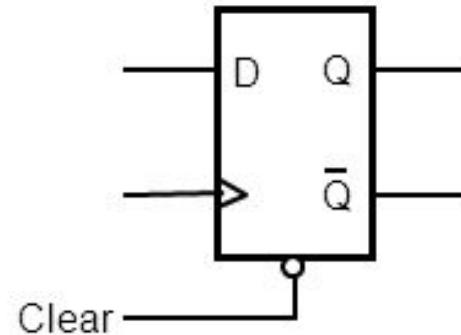
- **Types**

- **Synchronous**

- Dependent to CLK signal
- Reset the value only when the CLK signal is active
 - Next clock edge arrives

- **Asynchronous**

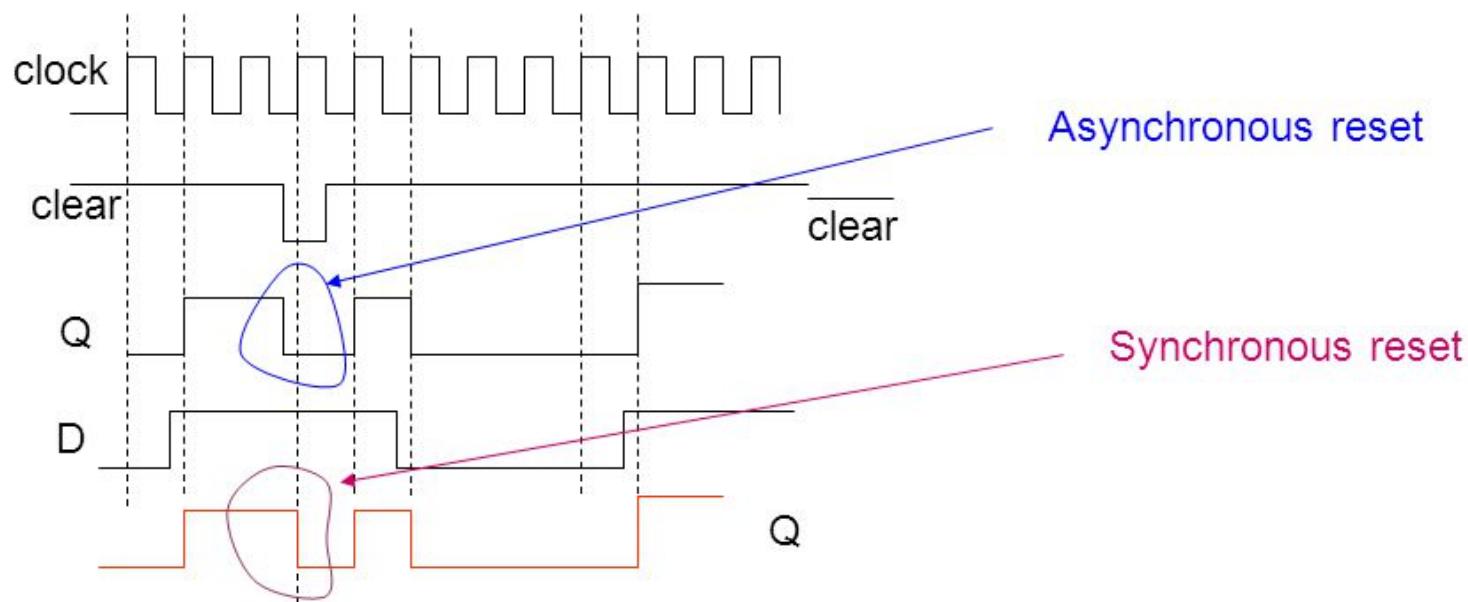
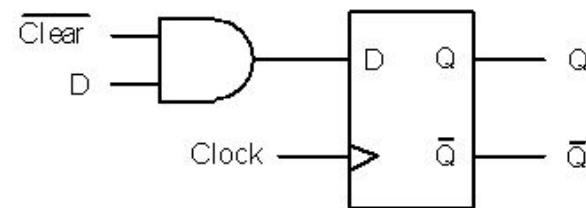
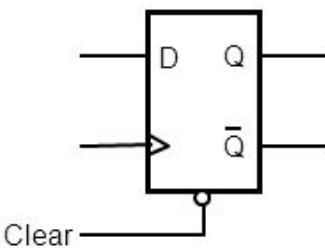
- Independent to CLK signal
- Reset the value only without considering the CLK signal state
- Quick but dangerous



Clear

R	C	D	Q	Q'
0	X	X	0	1
1	↑	0	0	1
1	↑	1	1	0

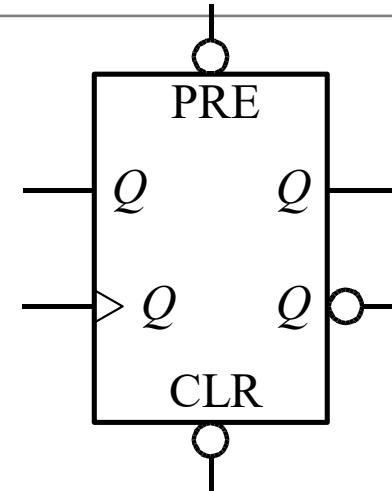
Reset (cont'd)



Preset

- Preset input
 - Reset the FF value
 - $Q = 1$
 - A.k.a., PRE

- Types
 - Synchronous
 - Dependent to CLK signal
 - Asynchronous
 - Independent to CLK signal



Inputs				Outputs		Mode
\overline{PRE}	\overline{CLR}	D	CLK	Q	\overline{Q}	
L	H	x	x	H	L	Set
H	L	x	x	L	H	Clear
L	L	x	x	H	H	Not allowed
H	H	H	↑	H	L	Clocked operation
H	H	L	↑	L	H	Clocked operation
H	H	x	L	Q_0	\overline{Q}_0	Hold

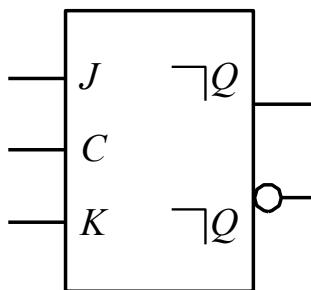
JK Flip Flop (JK-FF)

- **Input**

- J
- K
- CLK

- **Output**

- 4 states

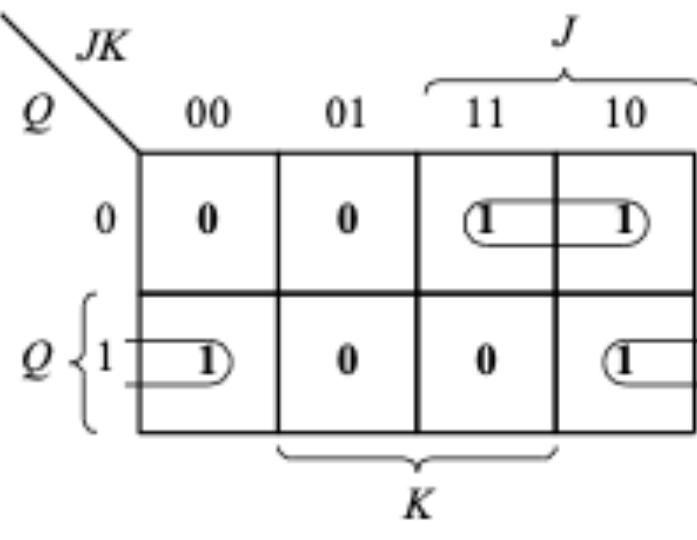
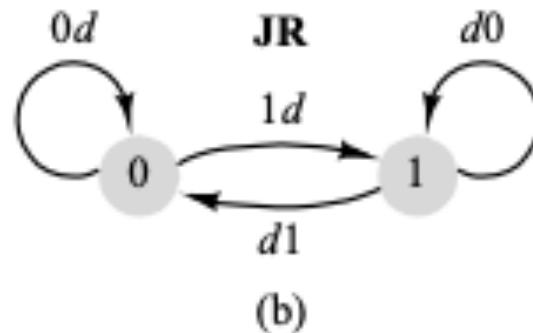


J	K	Q	C	Q^*	
0	0	0	↑	0	Hold
0	0	1	↑	1	
0	1	0	↑	0	Reset
0	1	1	↑	0	
1	0	0	↑	1	Set
1	0	1	↑	1	
1	1	0	↑	1	Toggle
1	1	1	↑	0	

JK-FF: Characteristics

J	K	Q	C	Q^*
0	0	0		0 Hold
0	0	1		1
0	1	0		0 Reset
0	1	1		0
1	0	0		1 Set
1	0	1		1
1	1	0		1 Toggle
1	1	1		0

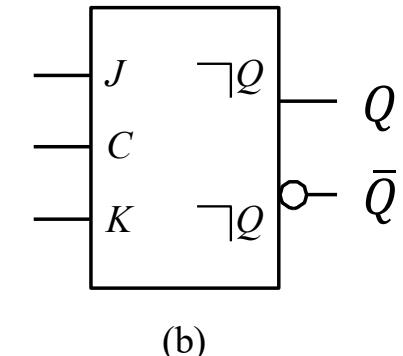
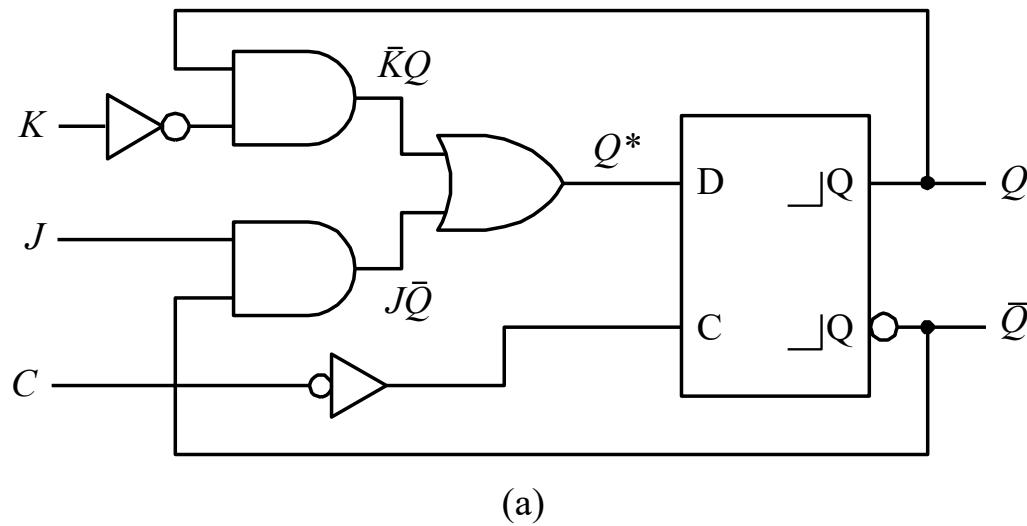
(a)



$$Q^* = K'Q + JQ'$$

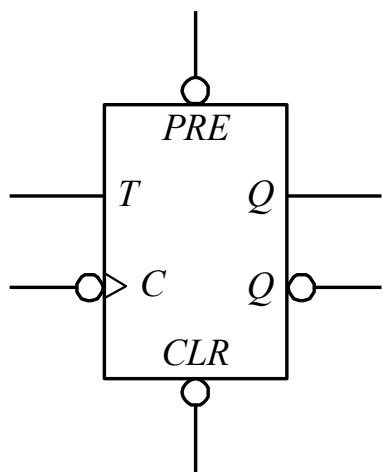
JK-FF: Implementation

- Using D-FF



T Flip Flop (T-FF)

- Input
 - T
 - CLK
- Output
 - Toggling
 - 2 states

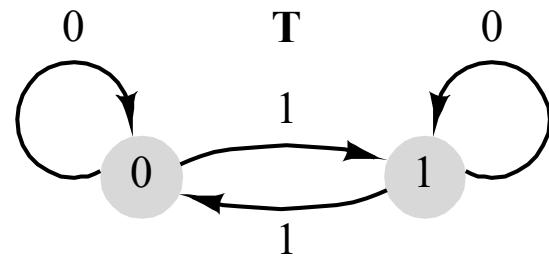


T	Q	C	Q*
0	0	↓	0 Hold
0	1	↓	1
<hr/>			1 Toggle
1	0	↓	
1	1	↓	0

$$Q^* = T'Q + TQ'$$

T-FF: Characteristics

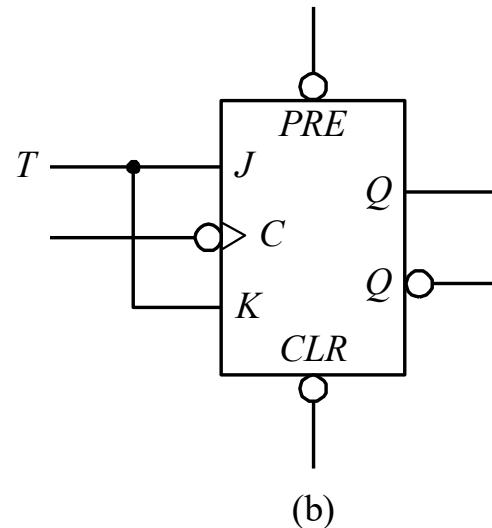
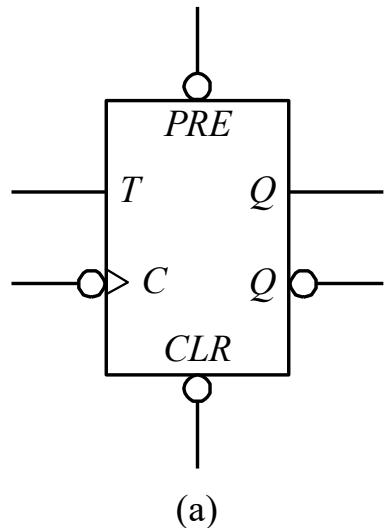
T	Q	C	Q^*
0	0	↓	0 Hold
0	1	↓	1
			1 Toggle
1	1	↓	0



$$Q^* = T'Q + TQ'$$

T FF: Implementation

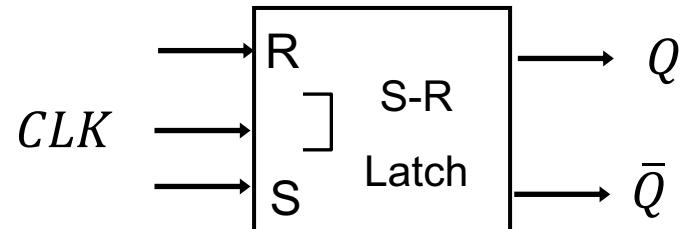
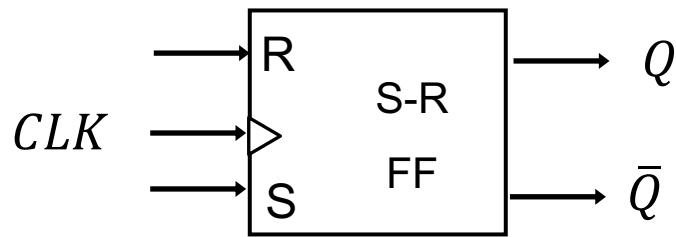
- Using JK FF



Master Slave

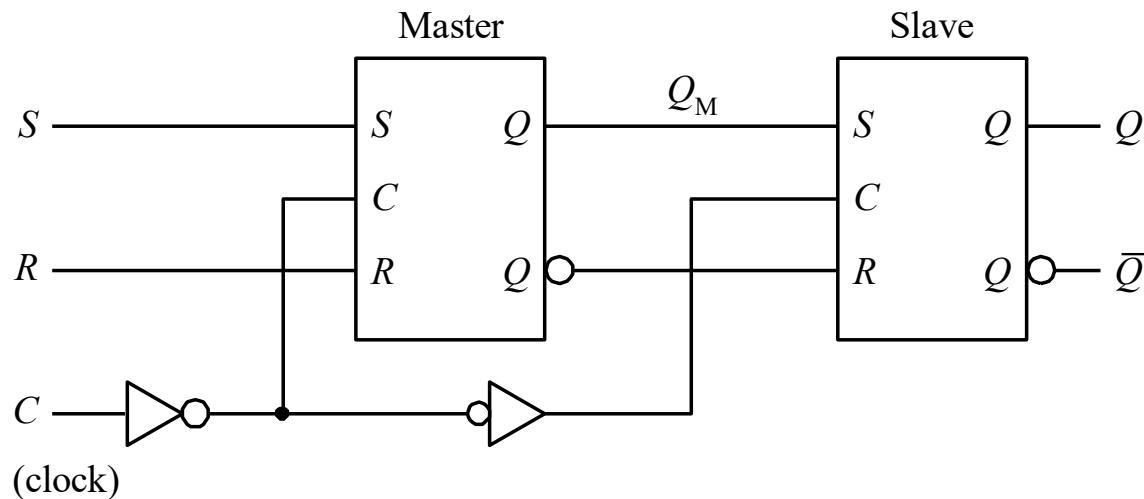
Edge-Triggered FF

- How to implement an edge triggered flip flop?

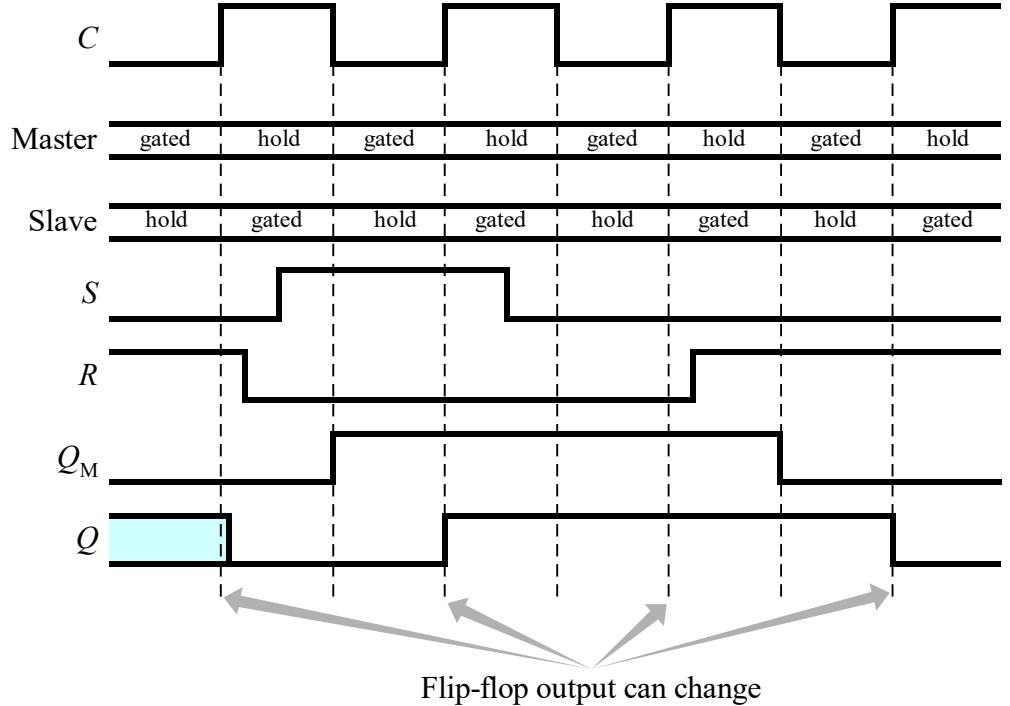
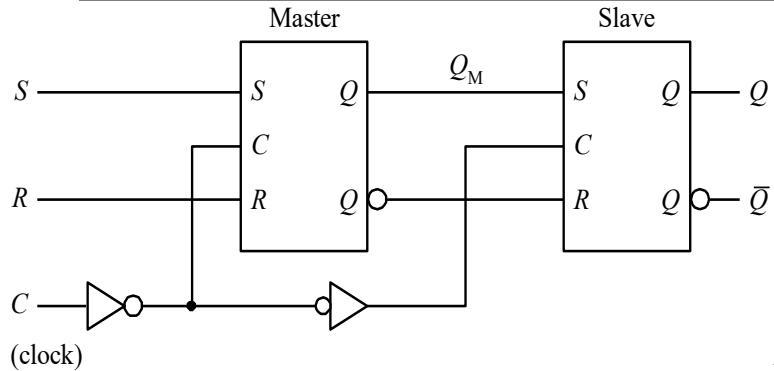


Master-Slave SR FF

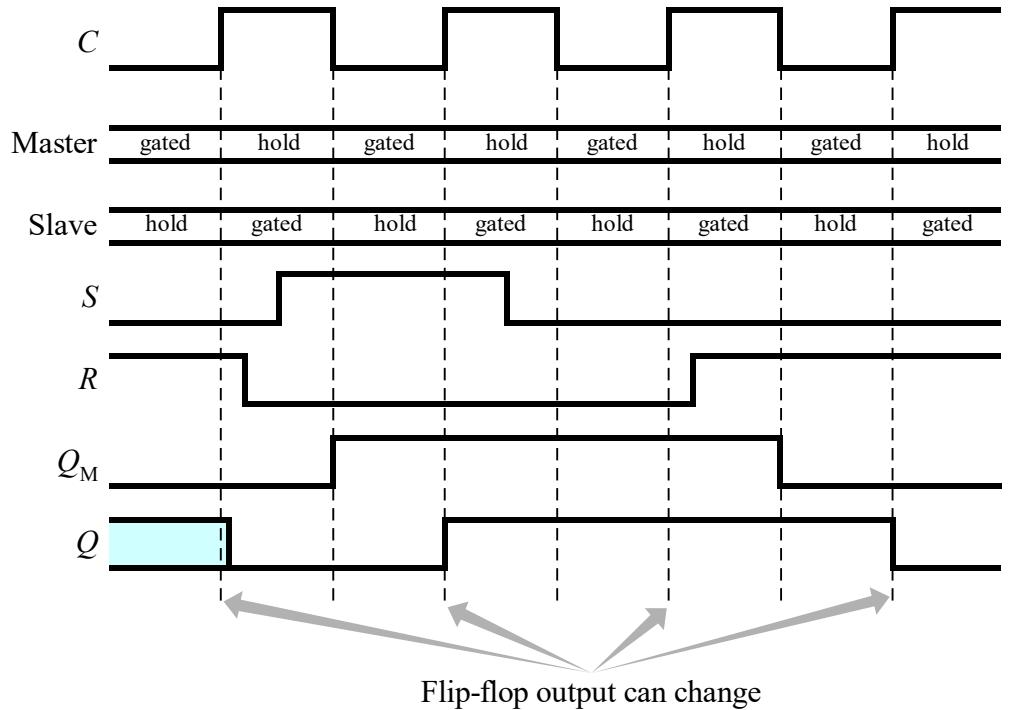
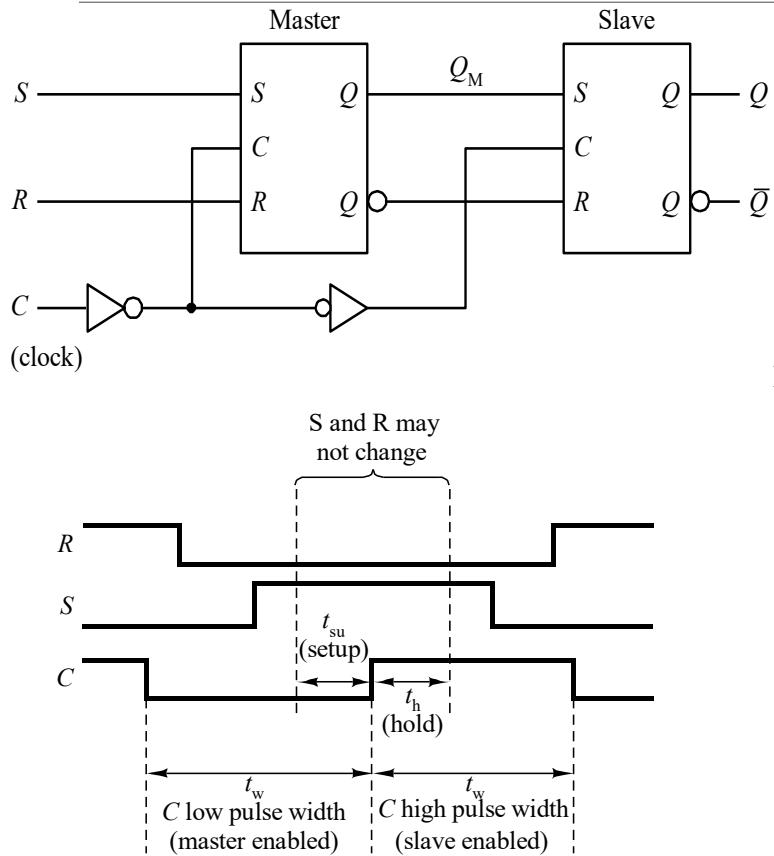
- One solution is using two level triggered flipflop



Master-Slave SR FF (cont'd)

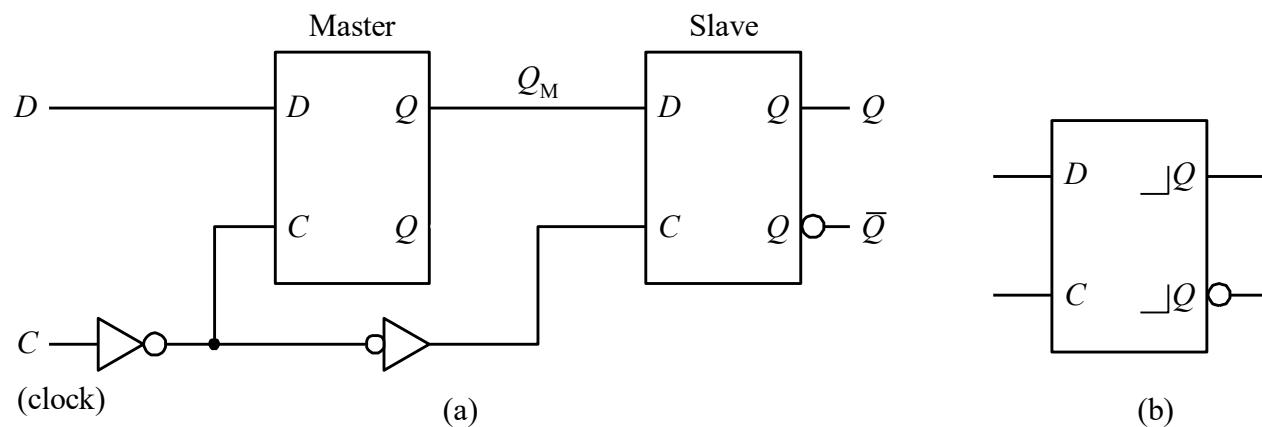


Master-Slave SR FF (cont'd)



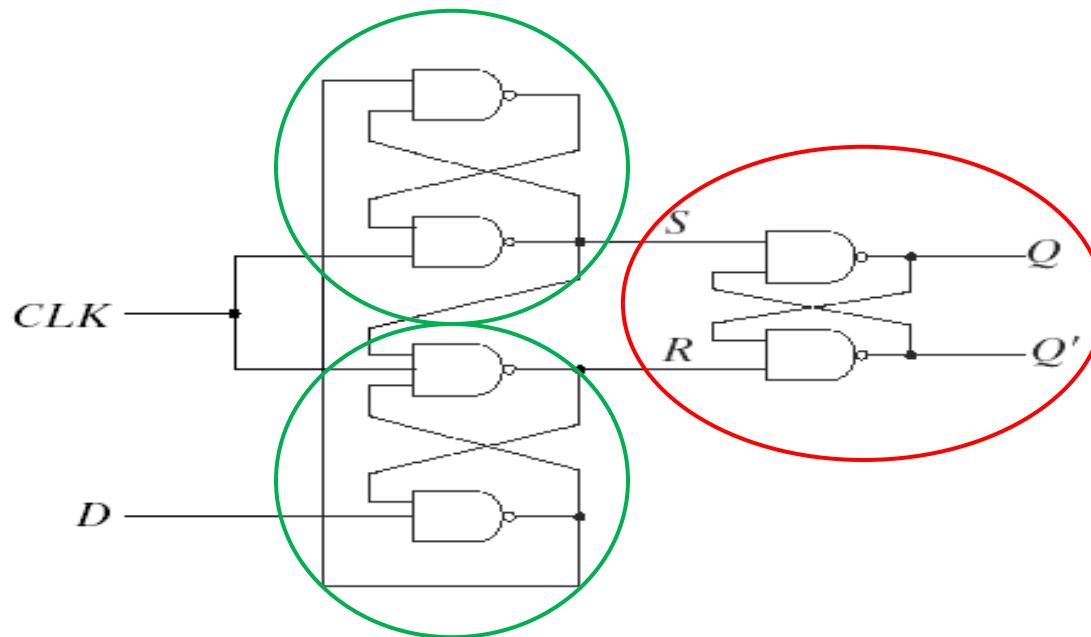
Master-Slave D Flip-flop

- One solution is using two level triggered flipflop



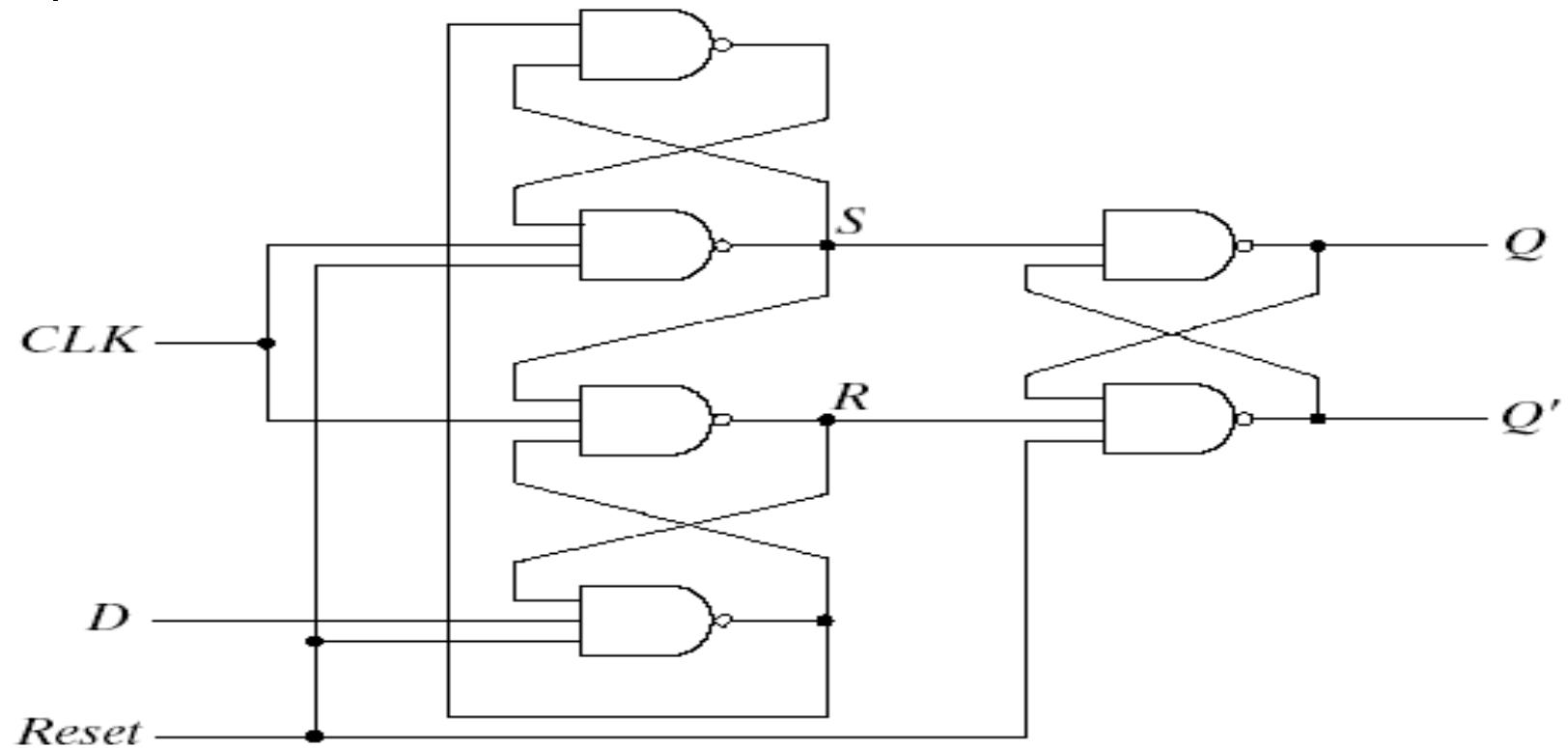
Positive Edge Triggered D FF

- Another solution is using three SR latches
 - **Two** latches respond to **inputs**
 - Data and CLK
 - **One** latch provides the **output**



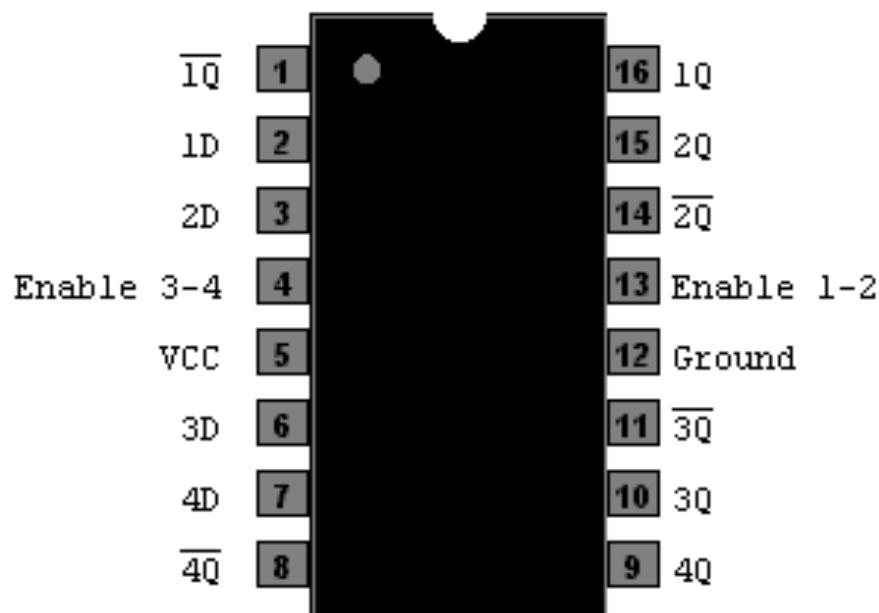
Positive Edge Triggered D FF + Reset

- Asynchronous reset



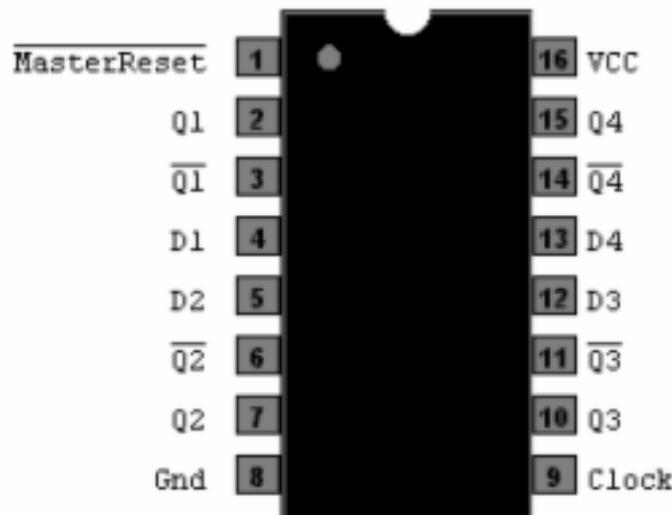
Standard MSI D Latch

- 74LS75 quad D latch



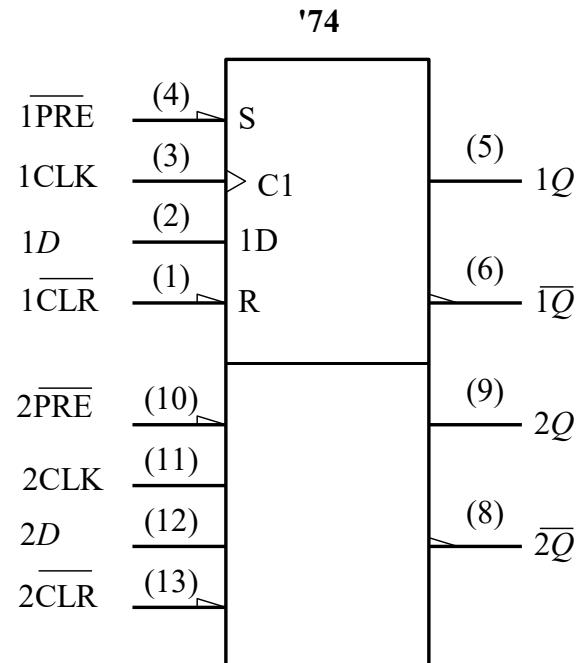
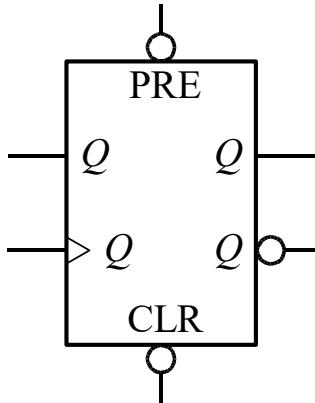
Standard MSI D FF

- 74LS175 quad D FF



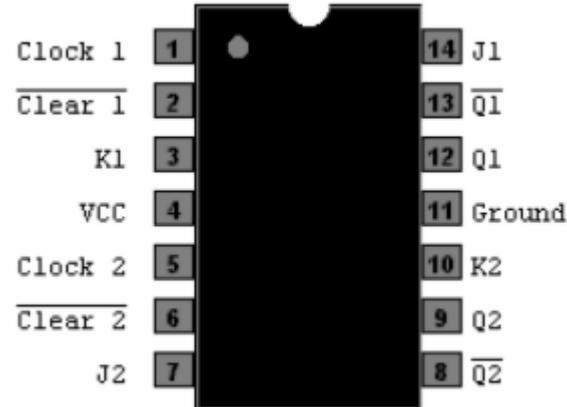
Standard MSI D FF (cont'd)

- SN7474 dual positive-edge-triggered D Flip-Flop



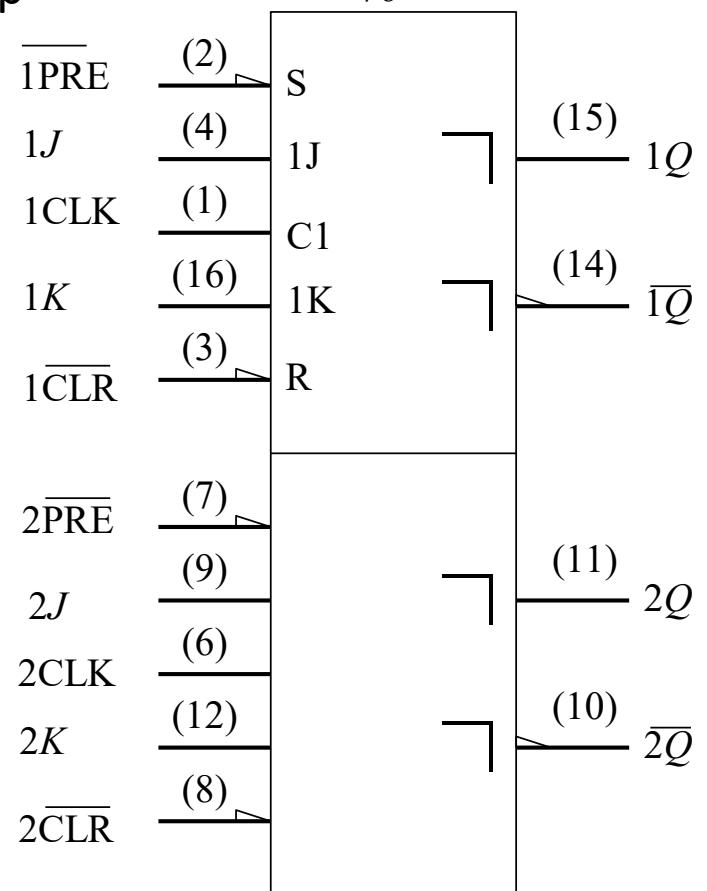
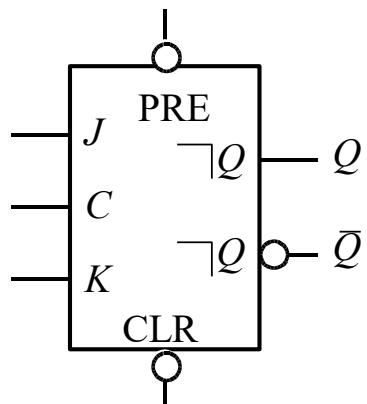
Standard MSI JK FF

- 74LS73A Dual Negative-edge-triggered JK FF



Standard MSI JK FF (cont'd)

- SN7476 Dual Pulse-Triggered JK Flip-Flop



Thank You

