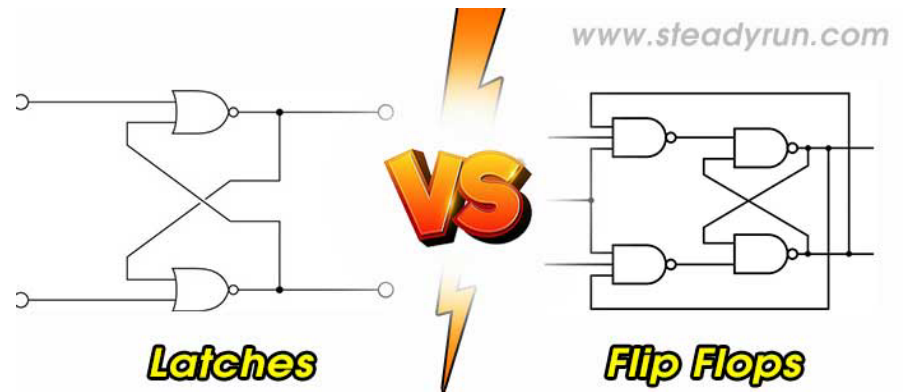


EE2000 Logic Circuit Design

Lecture 7 – Latch and Flip-Flop Circuits



Outline

- 7.1 Latches – Active High & Active Low
- 7.2 Flip-Flop (FF) – SR-, D-, JK-, T-FFs
- 7.3 Master-Slave FF
- 7.4 Edge-Triggered FF
- 7.5 Synchronous and Asynchronous Inputs

Two Classes of Logic Circuits

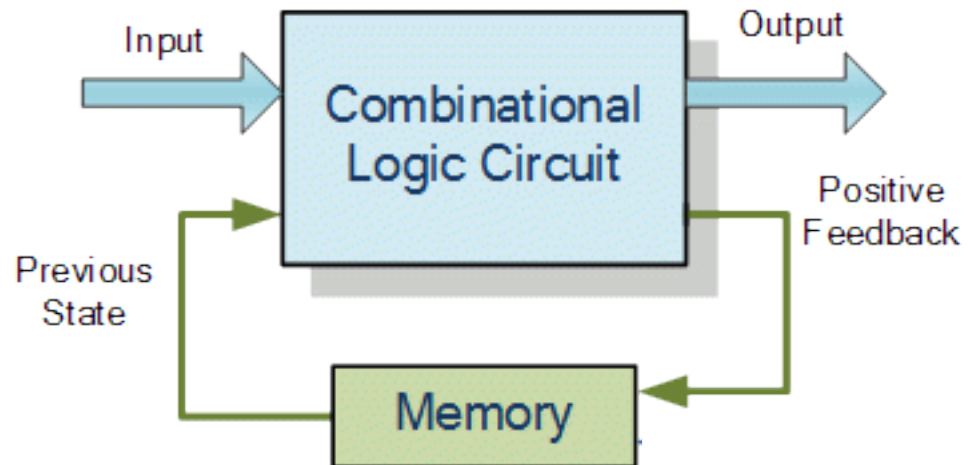
Combinational logic circuit

Output depends only on the inputs (As discussed in previous lectures)

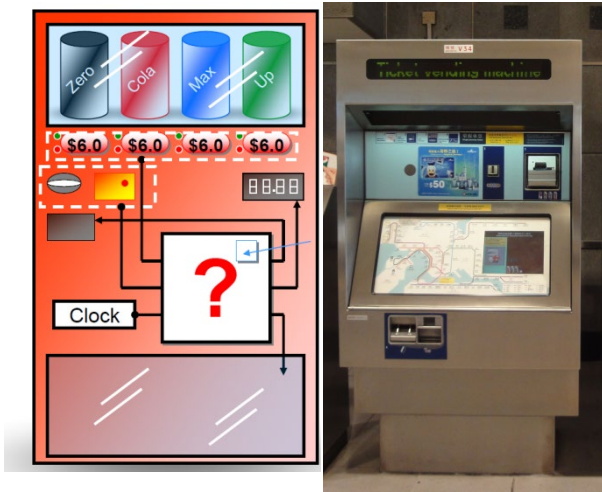
Sequential logic circuit

Output depends on **present input** + **past history**

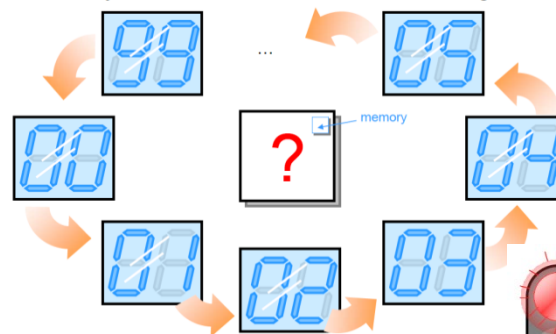
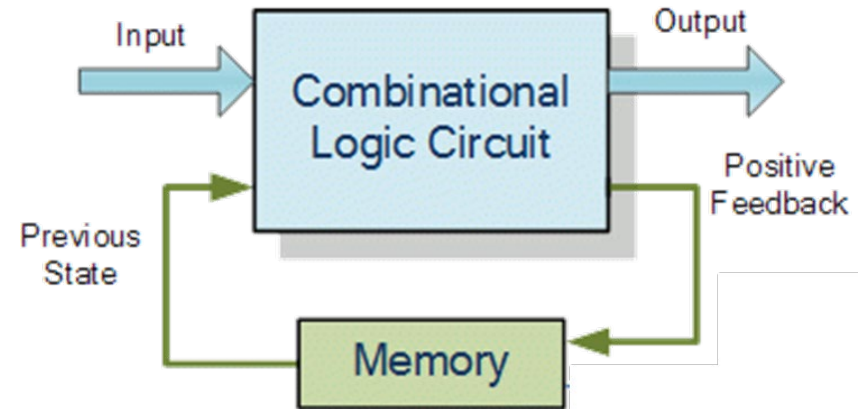
Memory circuit (to store previous STATE information) is required



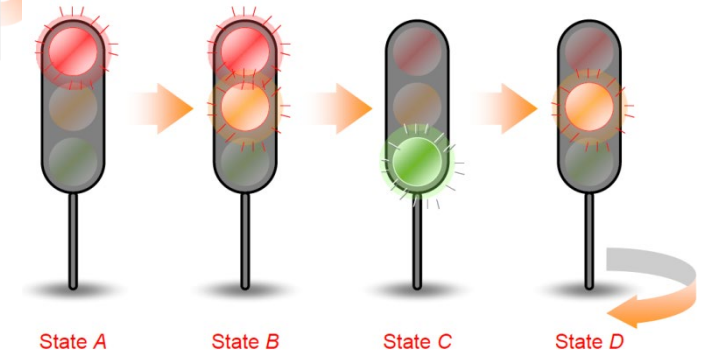
Sequential Logic Circuit



Vending Machine
Food/ drink / MTR tickets



Digital counter / clock

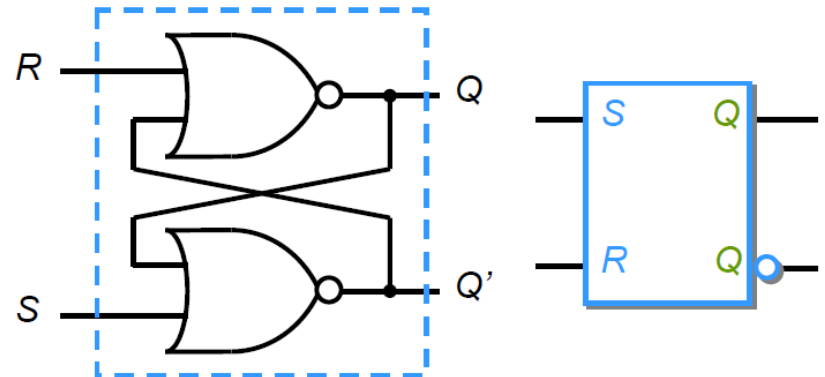


Traffic light controller

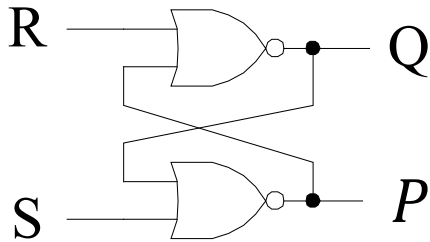
Latches and **Flip-Flops** are building blocks of sequential logic circuit

7.1 Latches

- Latch: A binary storage device, composed of logic gates with feedback
- The output state is controlled by its excitation input signals
- Two inputs: R stands for reset, S stands for set
- Two Outputs: Q stores the state of the latch (binary value)
- Bistable circuit: Two states

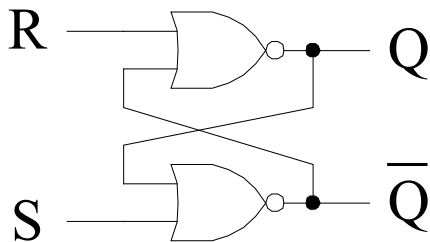


Active High Latch



$$Q = (R + P)'$$

$$P = (S + Q)'$$



- When S and R are 0,

$$Q = P' \quad \text{and} \quad P = Q' = \bar{Q}$$

The latch can store either

$$0 \quad (Q = 0, \bar{Q} = 1) \quad \text{or}$$

$$1 \quad (Q = 1, \bar{Q} = 0)$$

- When $S = 1, R = 0$

$$Q = 1, \bar{Q} = 0$$

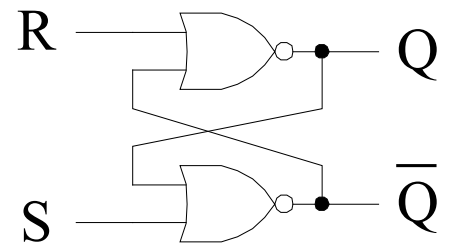
- When $S = 0, R = 1$

$$Q = 0, \bar{Q} = 1$$

- When S and R are 1,

$$Q = 0, \bar{Q} = 0$$

Active High Latch

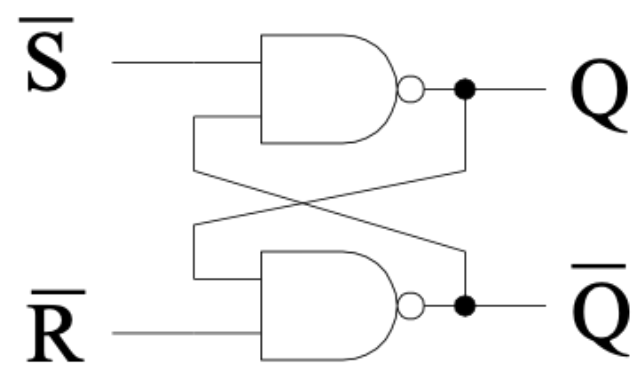


Next state of Q

S	R	Q_{t+1}	Q'_{t+1}	State
0	0	Q_t	Q'_t	Hold state
0	1	0	1	Reset state
1	0	1	0	Set state
1	1	0	0	Undefined state

t	1	2	3	4	5	6	7	8	9
S	1	0	0	0	0	1	1	1	0
R	0	0	1	0	0	0	1	0	0
Q	1	1	0	0	0	1	0	1	1
\bar{Q}	0	0	1	1	1	0	0	0	0
State	S	H	R	H	H	S	U	S	H

Active Low Latch

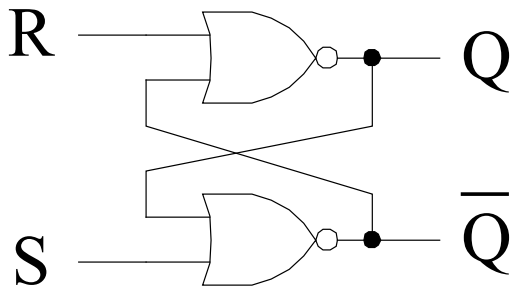


S'	R'	Q_{t+1}	Q'_{t+1}	State
1	1	Q_t	Q'_t	Hold state
1	0	0	1	Reset state
0	1	1	0	Set state
0	0	1	1	Undefined state

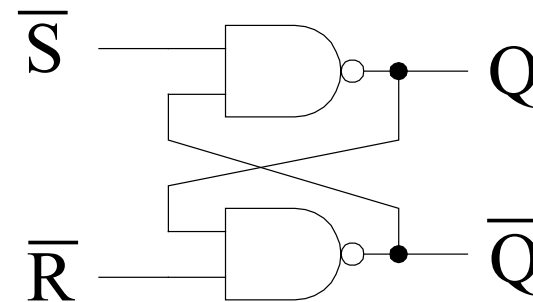
t	1	2	3	4	5	6	7	8	9
\bar{S}	1	1	0	1	1	1	0	1	1
\bar{R}	0	1	1	1	1	0	0	0	1
Q	0	0	1	1	1	0	1	0	0
\bar{Q}	1	1	0	0	0	1	1	1	1
State	R	H	S	H	H	R	U	R	H

Summary

Note that the input is active high for NOR gate implementation, whereas the input is active low for NAND gate implementation



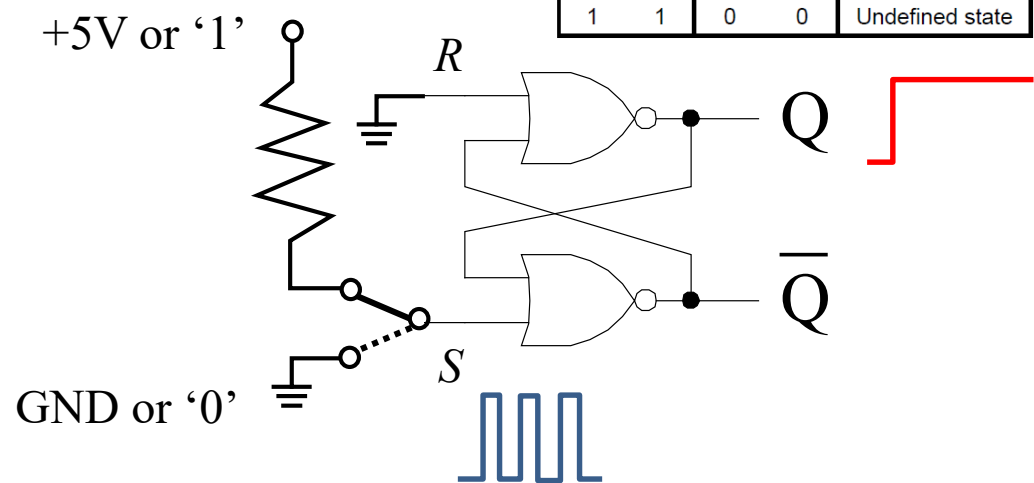
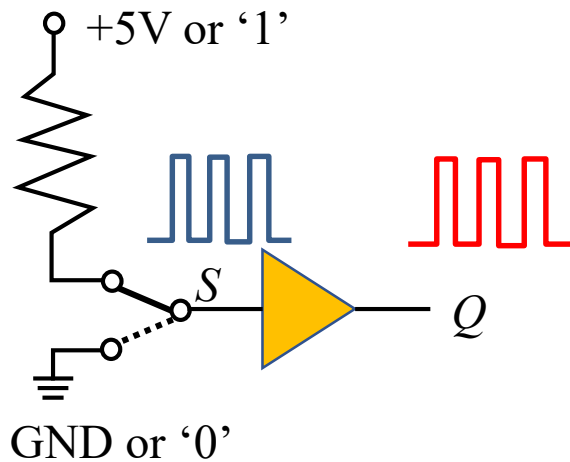
Require a signal "1" to change its state



Require a signal "0" to change its state

Application – Key Debouncing Circuit

When pressing a key or switch, its internal spring may cause the contact point to vibrate for a short period of time, causing a series of “ON” and “OFF” state before settling down



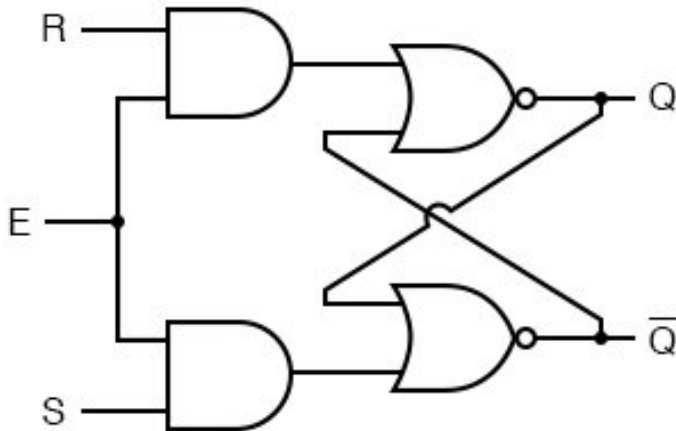
Next state of Q

S	R	Q_{t+1}	Q'_{t+1}	State
0	0	Q_t	Q'_t	Hold state
0	1	0	1	Reset state
1	0	1	0	Set state
1	1	0	0	Undefined state

Drawback - Noise

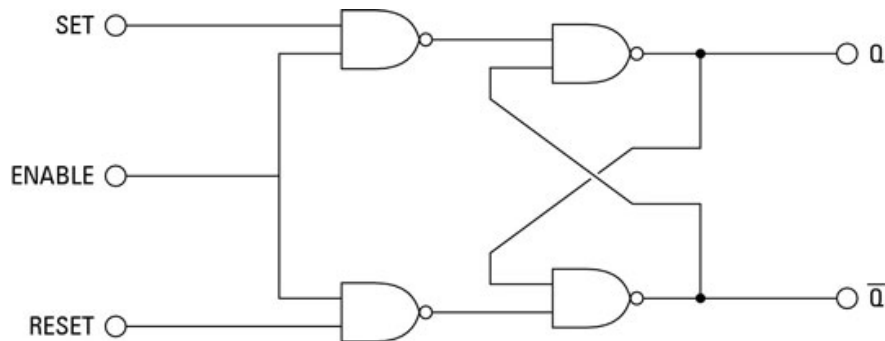
If inputs S , R are unstable (or noise), output produces an unstable short pulse

Solution: Gated Latch!



E	S	R	Q_{t+1}	$\overline{Q_{t+1}}$	State
0	x	x	Q_t	$\overline{Q_t}$	Hold
1	0	0	Q_t	$\overline{Q_t}$	Hold
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	0	0	Undefined

Gated Active Low Latch

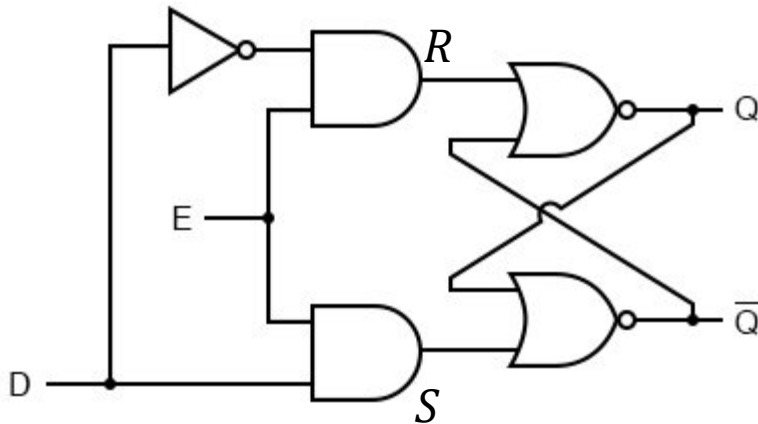


E	S	R	Q_{t+1}	$\overline{Q_{t+1}}$	State

Drawback – Undefined State

The latches will enter **undefined state** for some inputs (e.g. $R = S = 1$)

Solution: **D Latch!**



E	D	Q_{t+1}	\bar{Q}_{t+1}	State
0	x	Q_t	\bar{Q}_t	Hold
1	1	1	0	Set
1	0	0	1	Reset

Remember in this way:

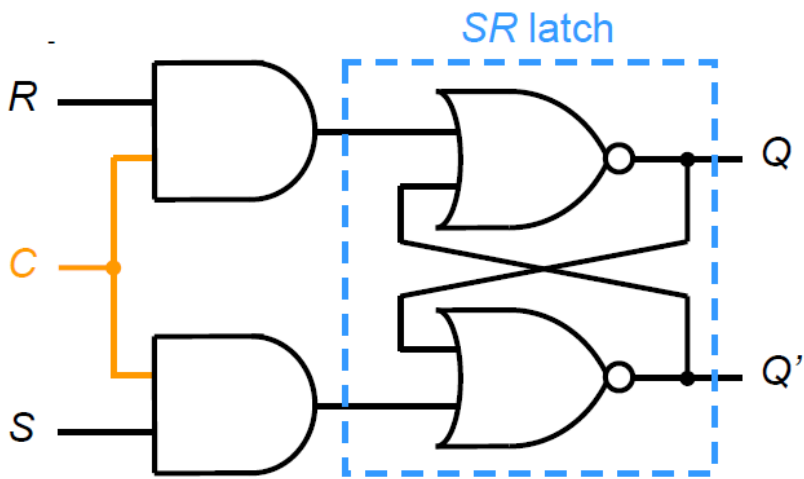
The value of D is transferred to the Q output

7.2 Flip-Flops

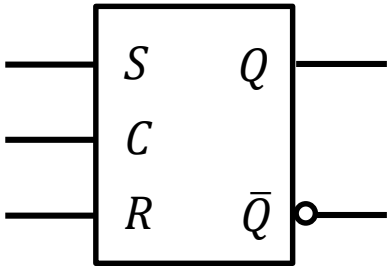
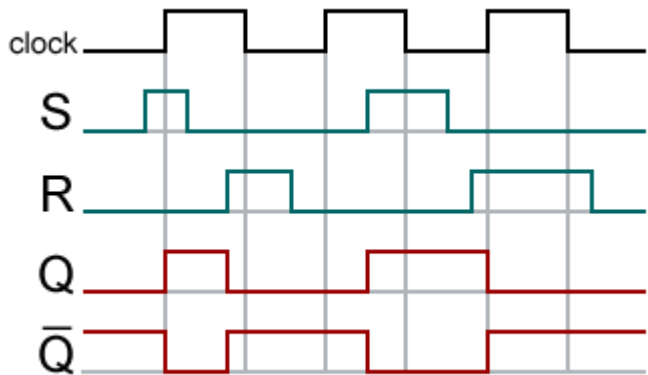
- Flip-Flop: A **clocked** binary storage device that the change of output state is controlled by the clock signal (synchronized)
- We will discuss the following four types of FF
 - Set-Reset FF (SR FF)
 - Data/Delay FF (D FF)
 - Jack Kilby FF (JK FF)
 - Toggle FF (T FF)

Set-Reset FF

A clocked signal is connected to the Enable input of a gated latch

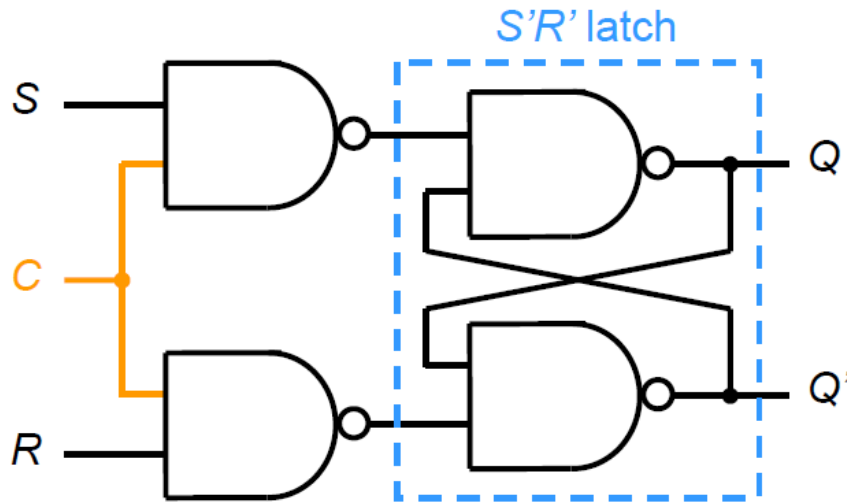


<i>C</i>	<i>S</i>	<i>R</i>	Q_{t+1}	\overline{Q}_{t+1}	State
0	x	x	Q_t	$\overline{Q_t}$	Hold
1	0	0	Q_t	$\overline{Q_t}$	Hold
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	0	0	Undefined

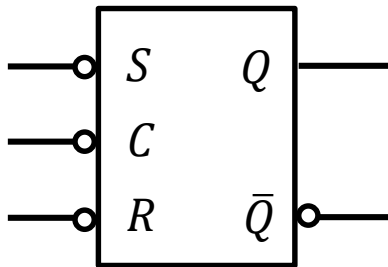


Logic
symbol

SR FF (Active Low)



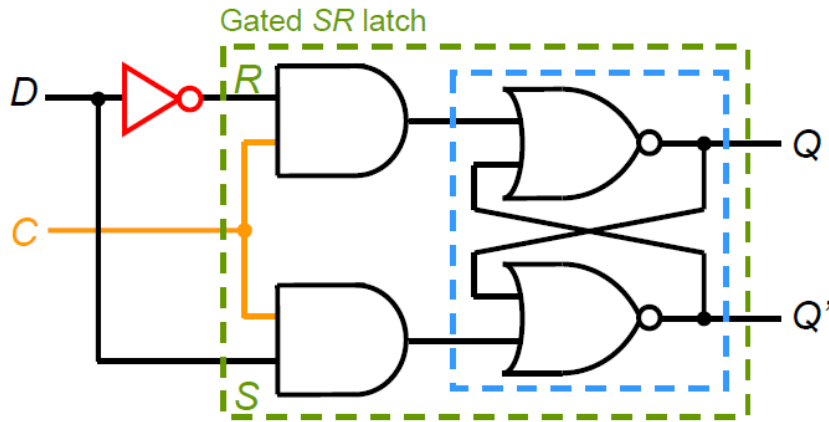
C	S	R	Q_{t+1}	$\overline{Q_{t+1}}$	State
1	x	x	Q_t	$\overline{Q_t}$	Hold
0	1	1	Q_t	$\overline{Q_t}$	Hold
0	1	0	0	1	Reset
0	0	1	1	0	Set
0	0	0	1	1	Undefined



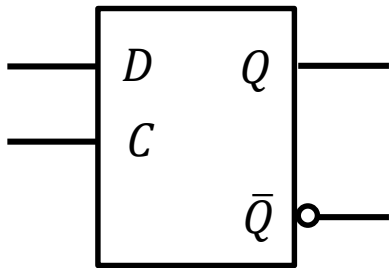
Logic
symbol

Data/Delay FF

A clocked signal is connected to the Enable input of a D latch



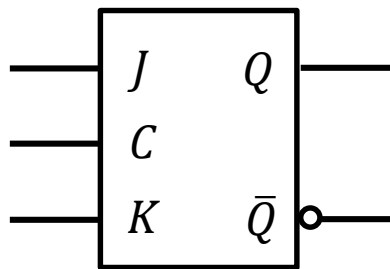
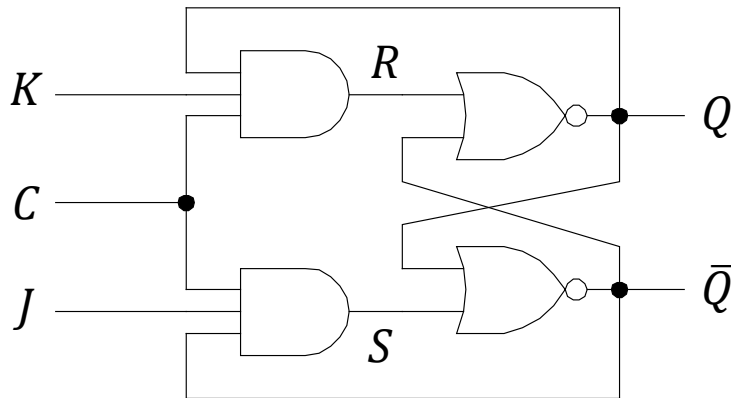
C	D	Q_{t+1}	\overline{Q}_{t+1}	State
0	x	Q_t	\overline{Q}_t	Hold
1	1	1	0	Set
1	0	0	1	Reset



Logic
symbol

Jack-Kilby FF

Refined SR FF whereby the undetermined state of SR FF is defined; ***J*** is set and ***K*** is reset



Logic
symbol

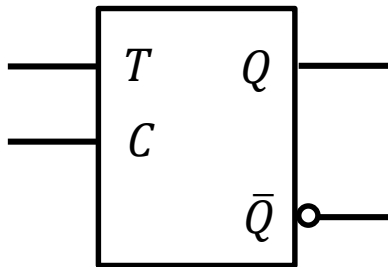
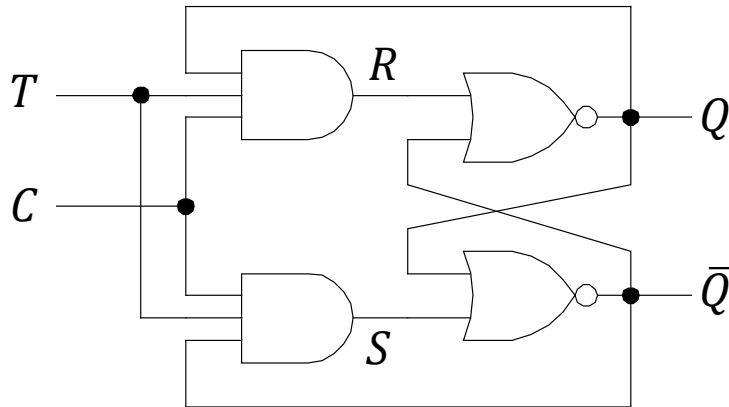
C	J	K	Q_{t+1}	$\overline{Q_{t+1}}$	State
0	x	x	Q_t	$\overline{Q_t}$	Hold
1	0	0	Q_t	$\overline{Q_t}$	Hold
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	$\overline{Q_t}$	Q_t	Toggle



C	J	K	Q_t	$\overline{Q_t}$	Q_{t+1}	$\overline{Q_{t+1}}$
1	1	1	0	1	1	0
1	1	1	1	0	0	1

Toggle FF

Output state changes for each clock pulse when ***T*** is in its active state



Logic
symbol

C	T	Q_{t+1}	$\overline{Q_{t+1}}$	State
0	x	Q_t	$\overline{Q_t}$	Hold
1	0	Q_t	$\overline{Q_t}$	Hold
1	1	$\overline{Q_t}$	Q_t	Toggle



C	T	Q_t	$\overline{Q_t}$	Q_{t+1}	$\overline{Q_{t+1}}$
1	1	0	1	1	0
1	1	1	0	0	1

Summary

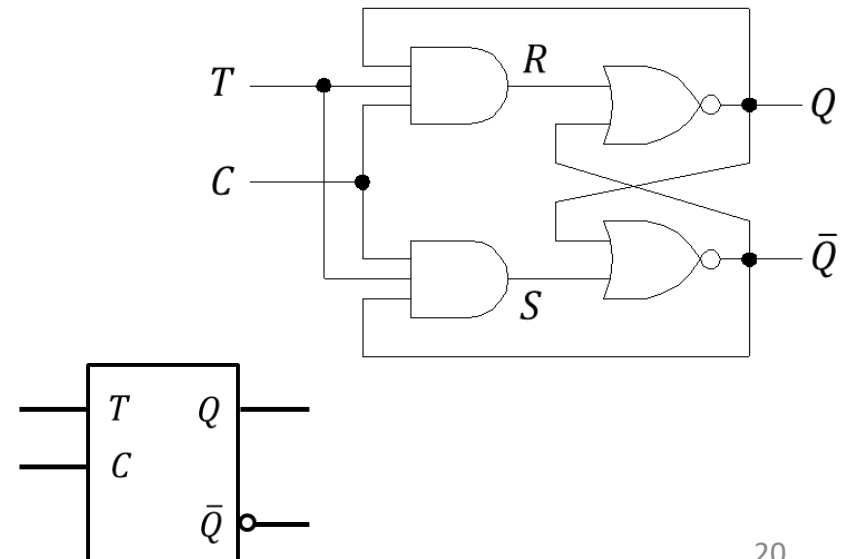
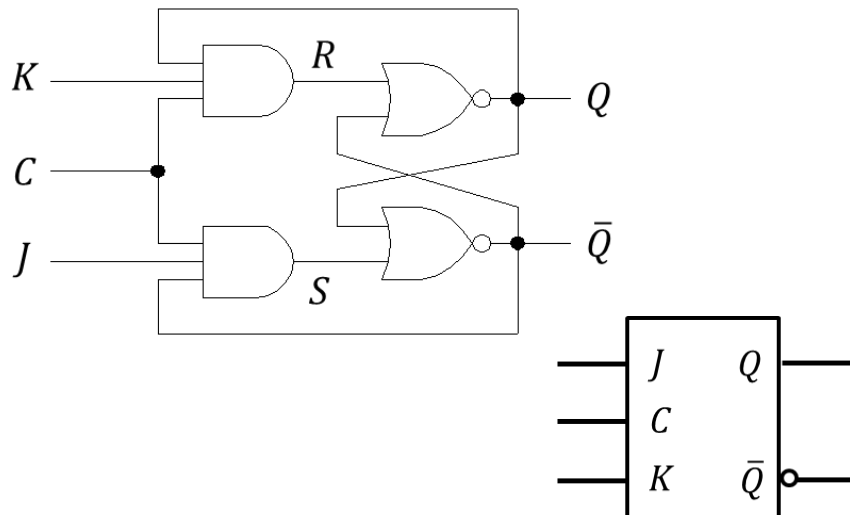
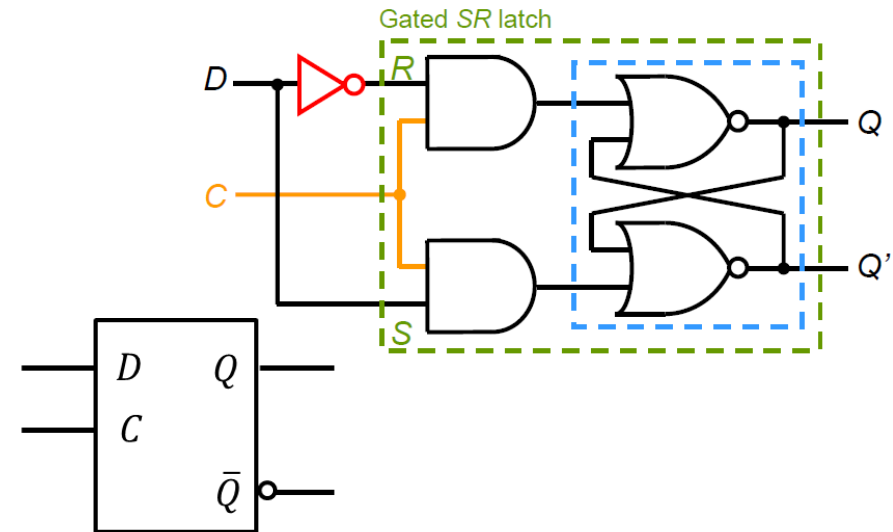
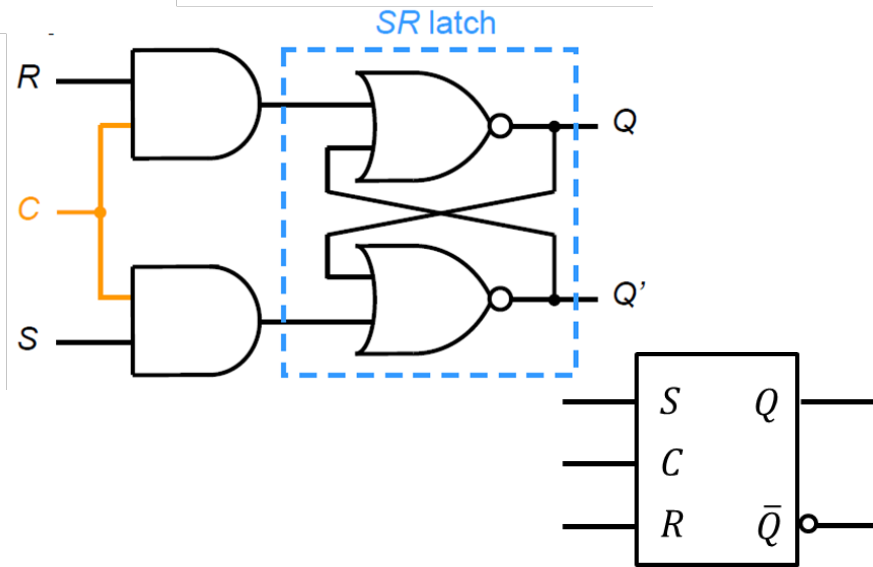


Table Given (Test and Exam)

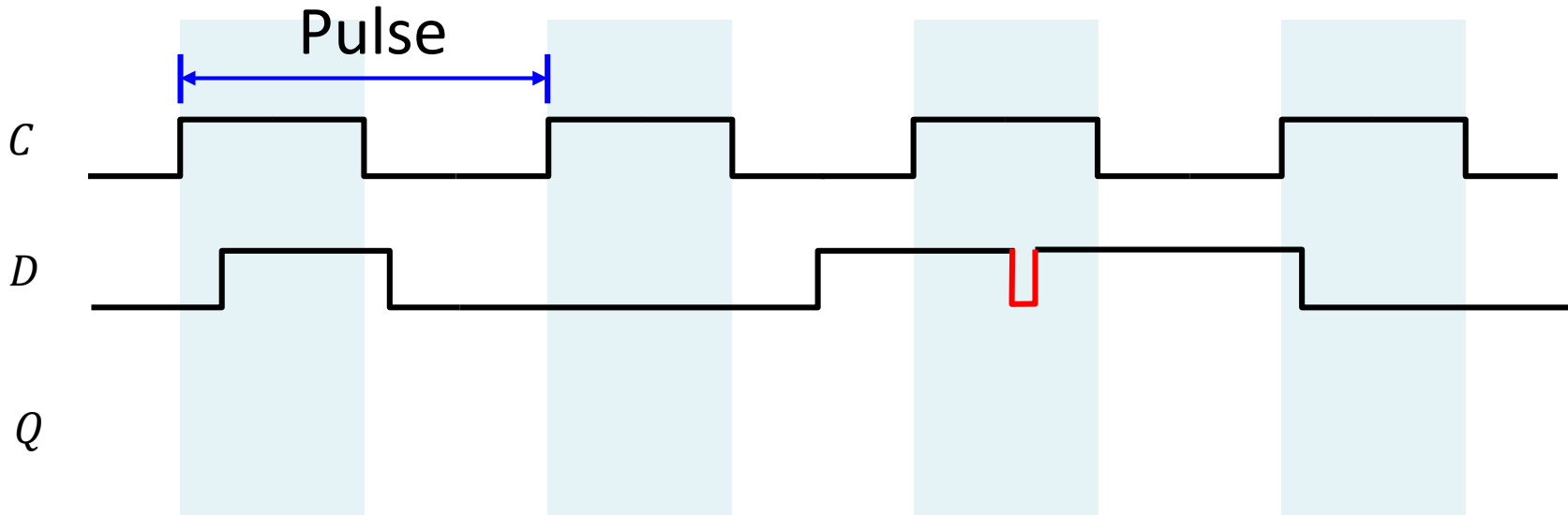
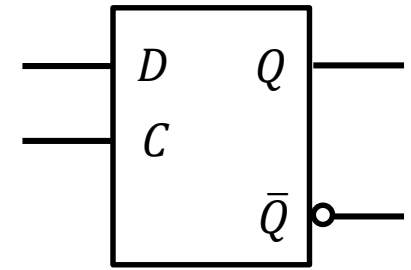
S	R	Q_{t+1}	$\overline{Q_{t+1}}$	State
0	0	Q_t	$\overline{Q_t}$	Hold
0	1	0	1	Reset
1	0	1	0	Set
1	1	1	1	Undefined

J	K	Q_{t+1}	$\overline{Q_{t+1}}$	State
0	0	Q_t	$\overline{Q_t}$	Hold
0	1	0	1	Reset
1	0	1	0	Set
1	1	$\overline{Q_t}$	Q_t	Toggle

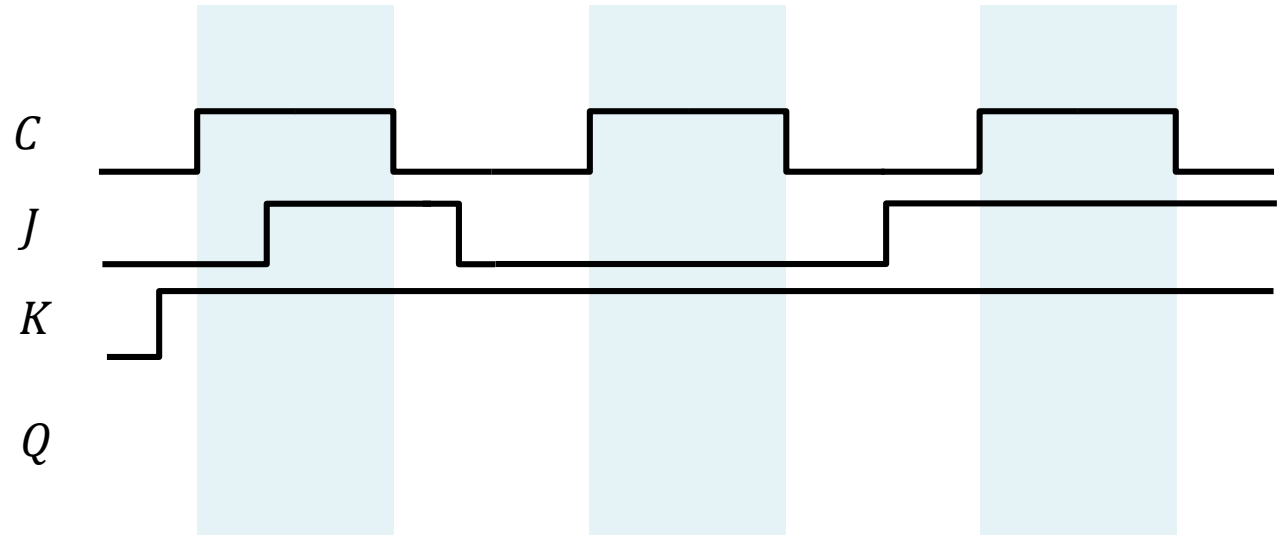
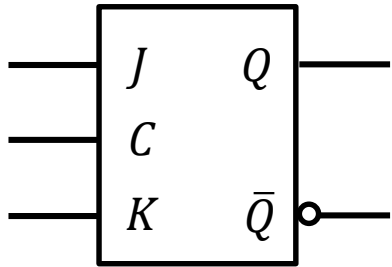
D	Q_{t+1}	$\overline{Q_{t+1}}$	State
1	1	0	Set
0	0	1	Reset

T	Q_{t+1}	$\overline{Q_{t+1}}$	State
0	Q_t	$\overline{Q_t}$	Hold
1	$\overline{Q_t}$	Q_t	Toggle

Exercise

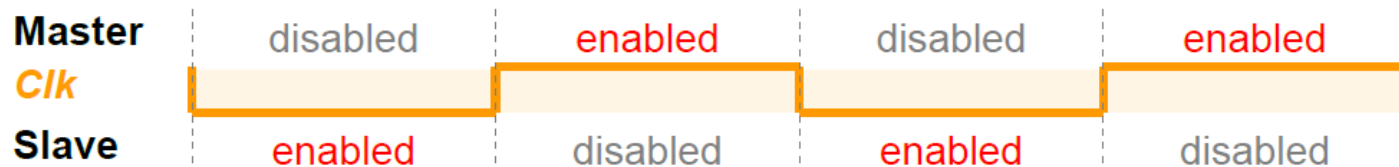
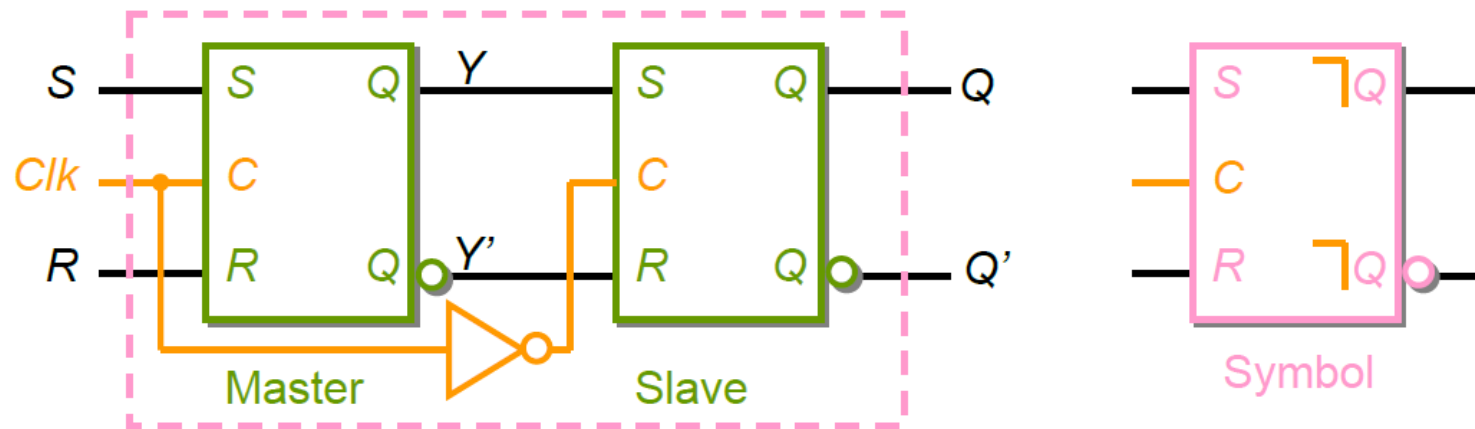


Exercise

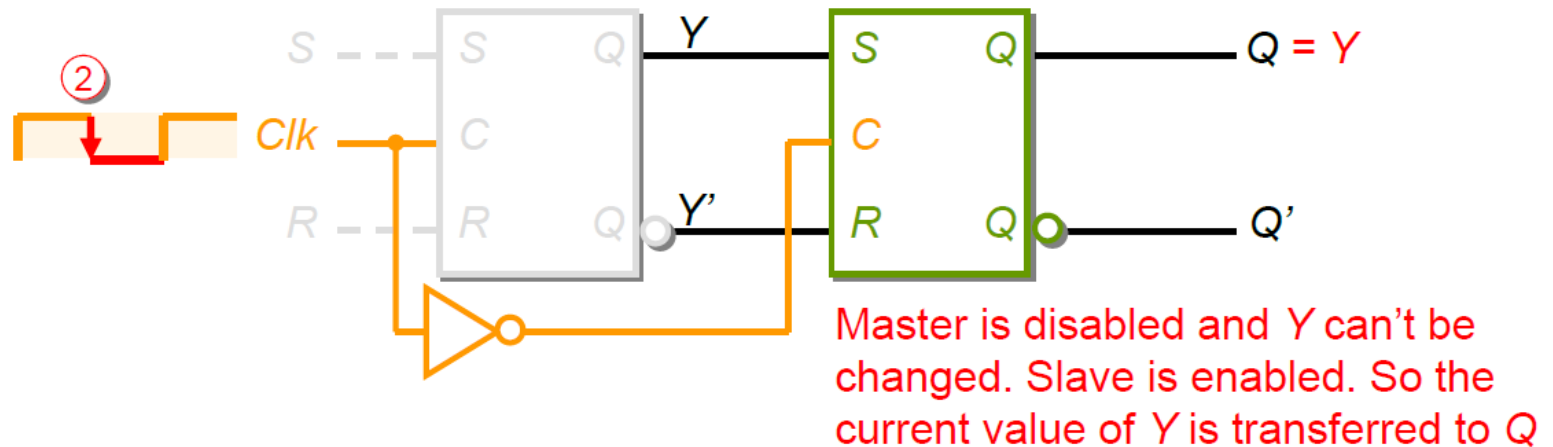
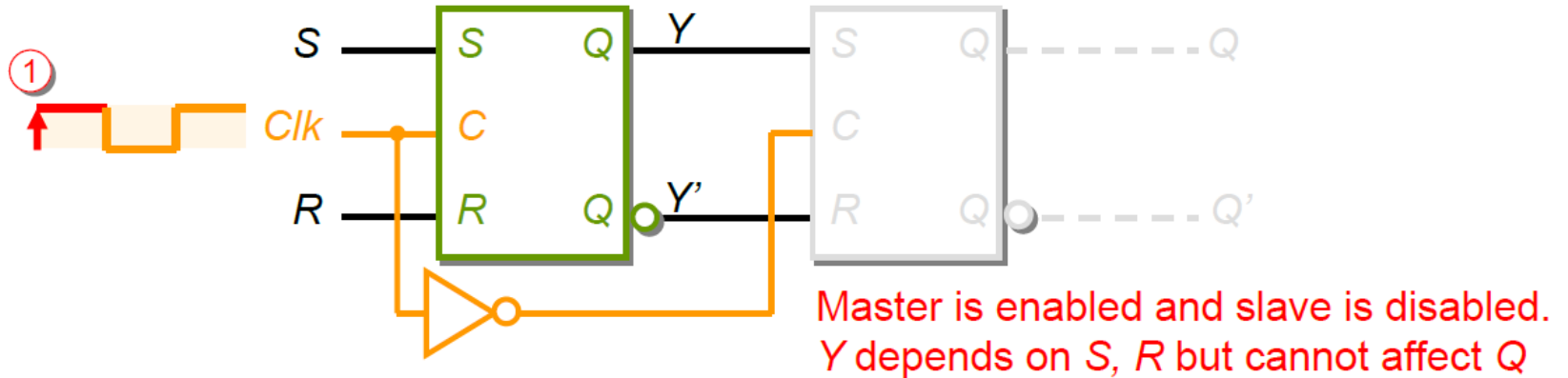


7.3 Master-Slave Flip-Flop

- A combination of two flip-flops together in a series configuration
- At any time, only one flip-flop is enabled



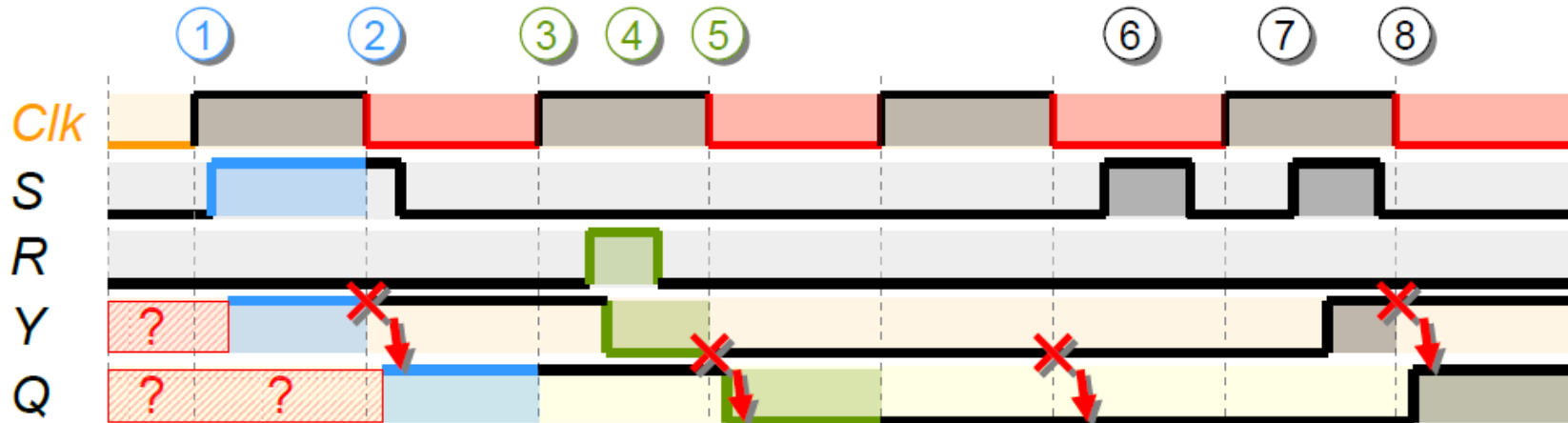
How Does It Work?



Timing Diagram

Initial inputs: S and R are LOW

Initial state of Y and Q unknown



① Clk switches to 1; master FF enabled; $S = 1, R = 0, \therefore Y = 1$

② Clk switches to 0; slave FF enabled; $Y = 1, \therefore Q = 1$

③ Clk switches to 1; master FF enabled; $S = 0, R = 0$, HOLD

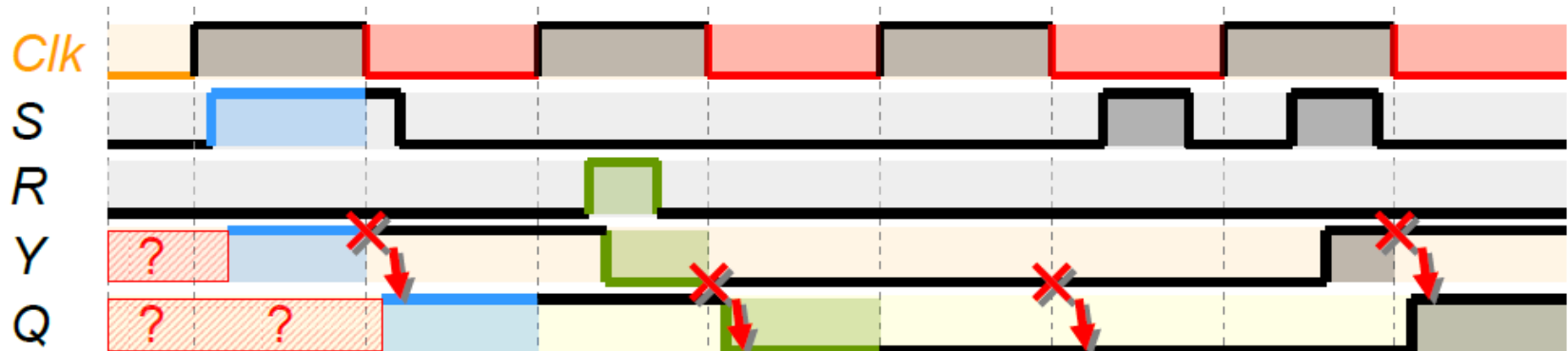
④ $S = 0, R = 1, \therefore Y = 0$ ⑤ Clk switches to 0; slave FF enabled; $Y = 0, \therefore Q = 0$

⑥ Clk is 0; master FF disabled; no change on Y .

⑦ Clk switches to 1; master FF enabled; $S = 1, R = 0, \therefore Y = 1$

⑧ Clk switches to 0; slave FF enabled; $Y = 1, \therefore Q = 1$

Characteristics

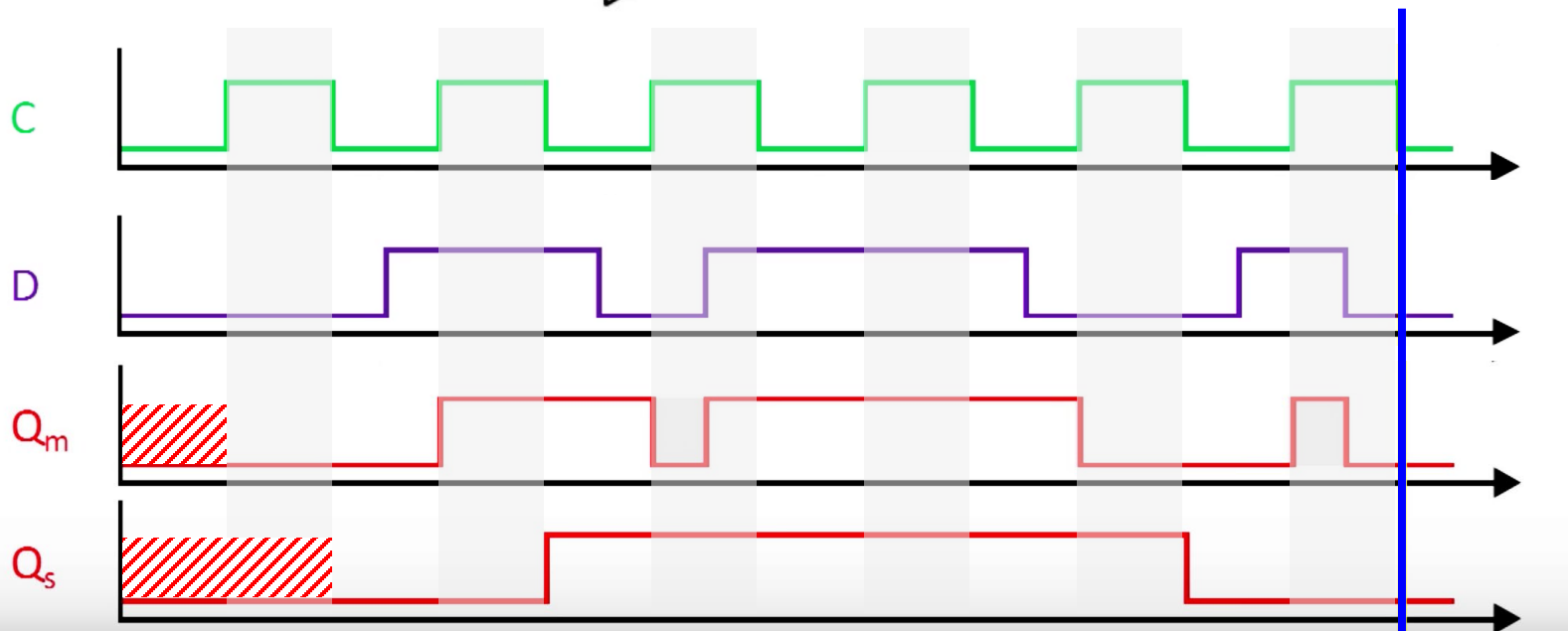
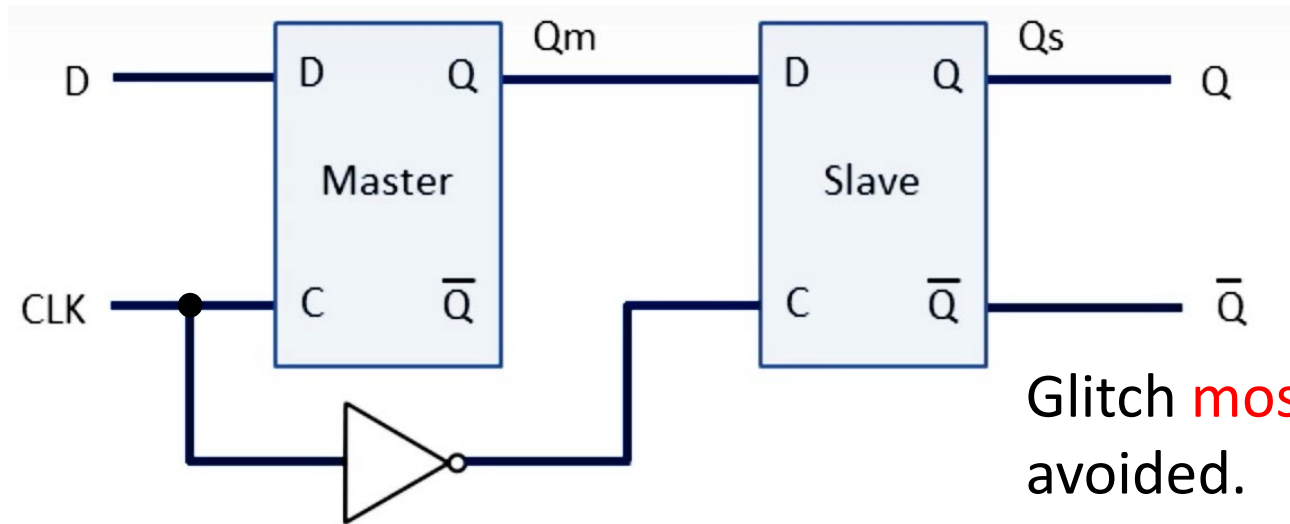


- Y is updated when Clk goes to and remains at 1.
- Q is updated when Clk goes to 0 and remains unchanged until the next pulse.

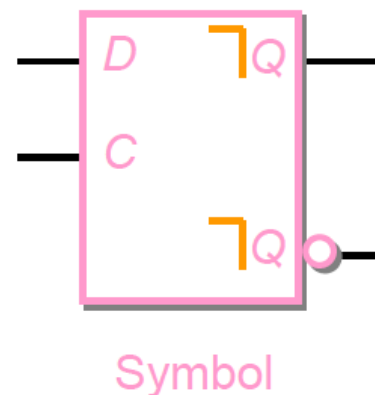
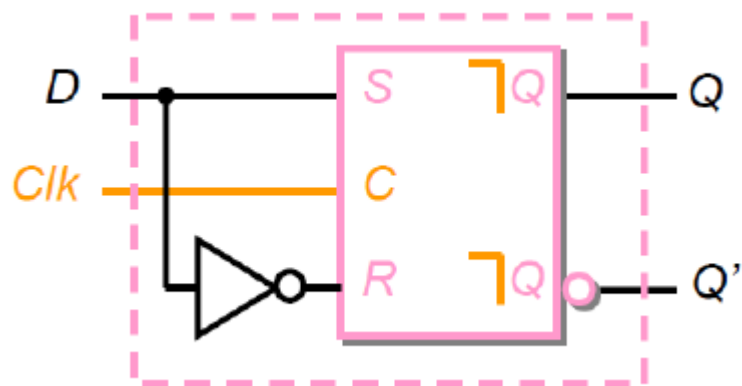
Therefore,


- For each clock pulse, Q only changes state at most **once**.
- State change only occurs **during the transition** of Clk **from 1 to 0**.

Master-Slave D Flip-Flop

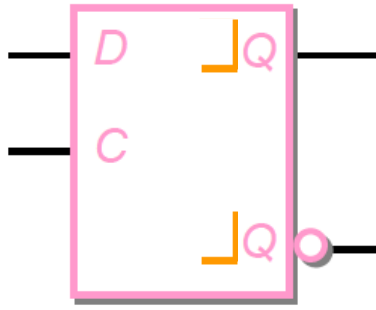


Master-Slave D Flip-Flop

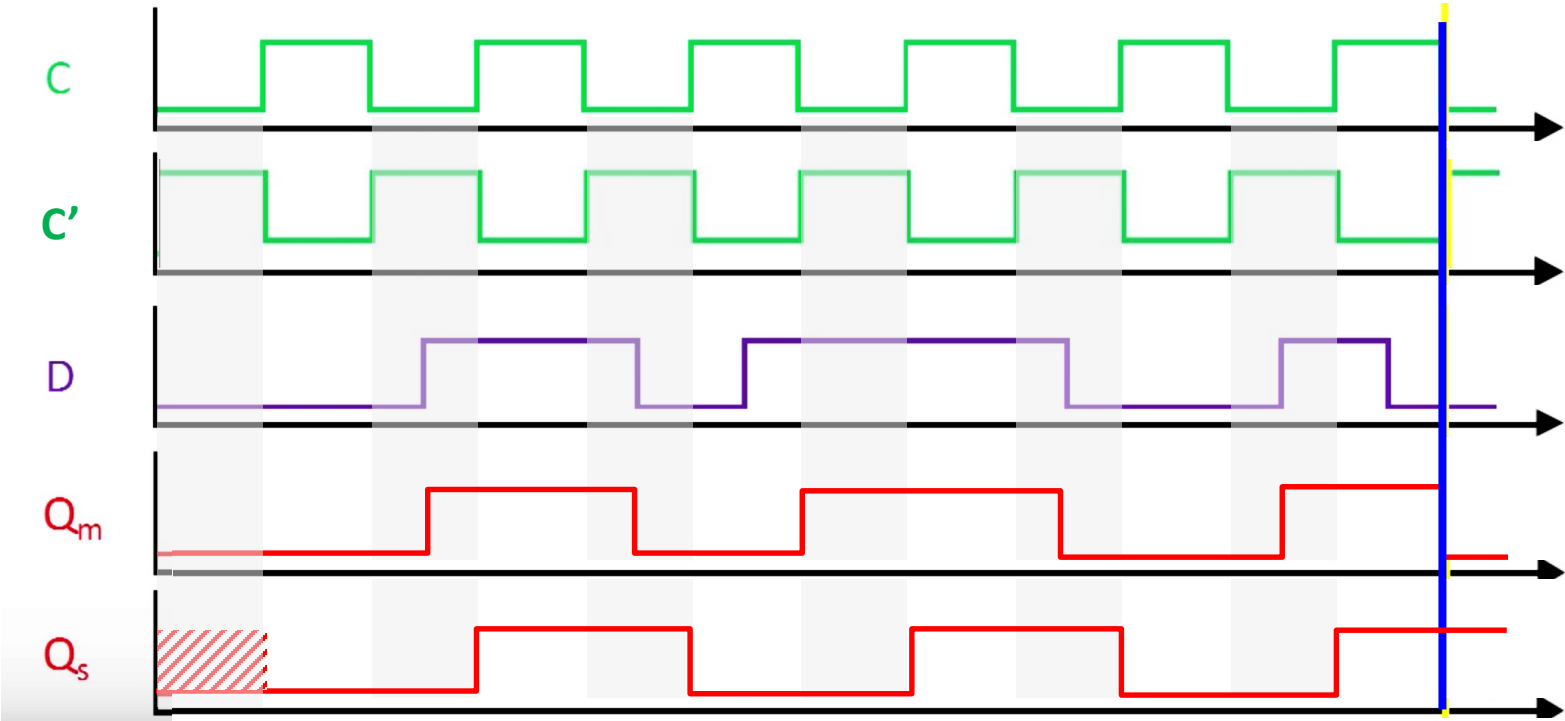
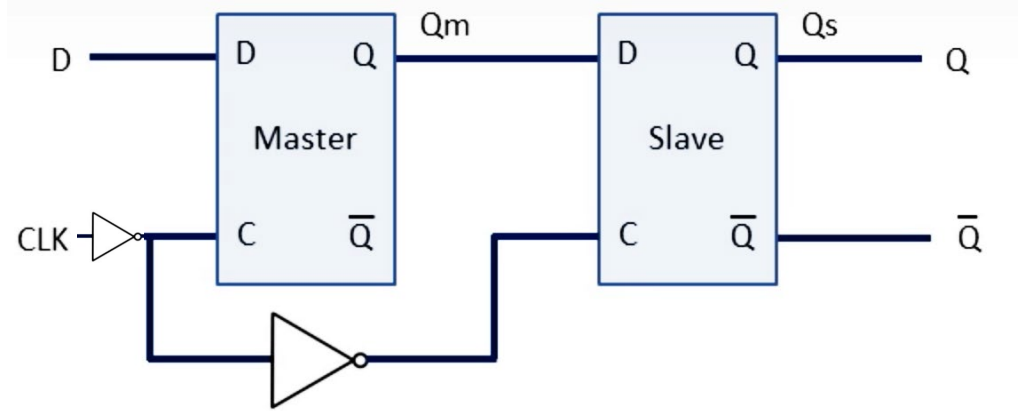


Inputs		Outputs		
D	Clk	Next Q	Next Q'	Meaning
x	0	Q	Q'	Hold
x		D	D'	Set / Reset

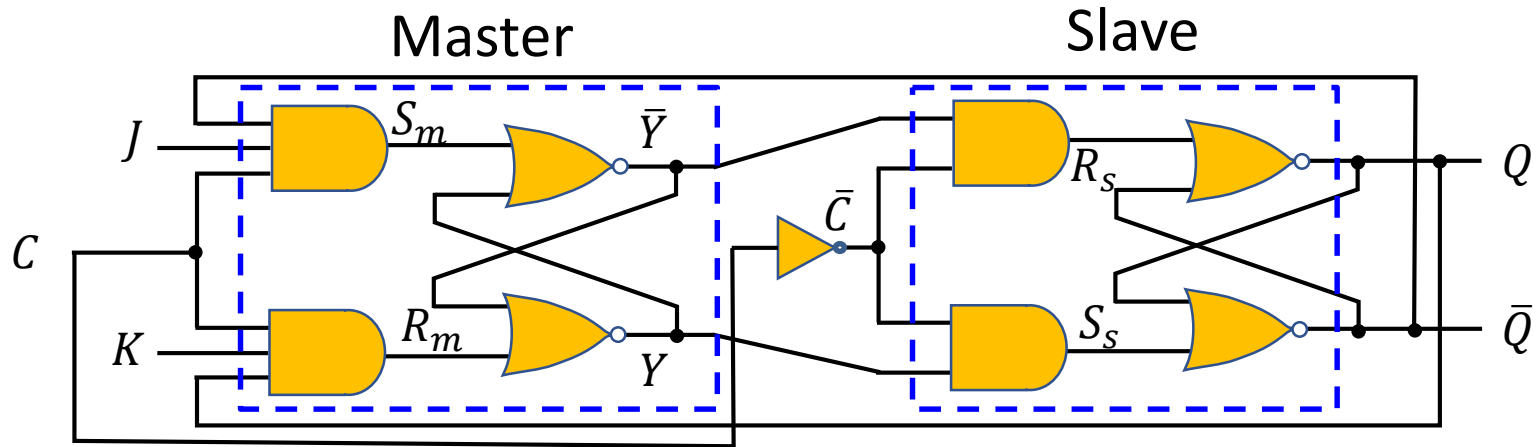
Master-Slave D Flip-Flop



Symbol

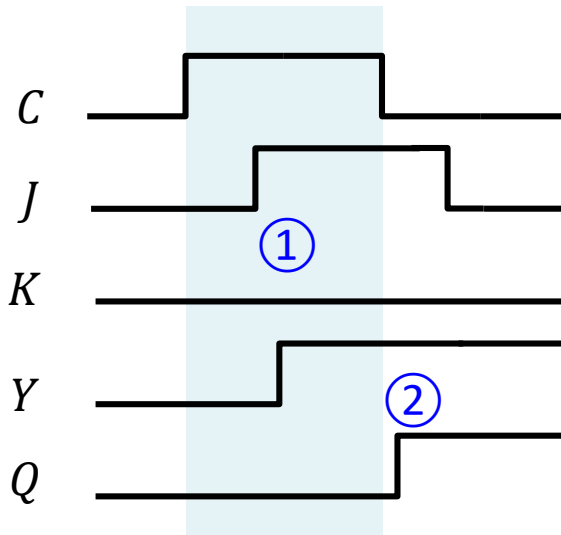
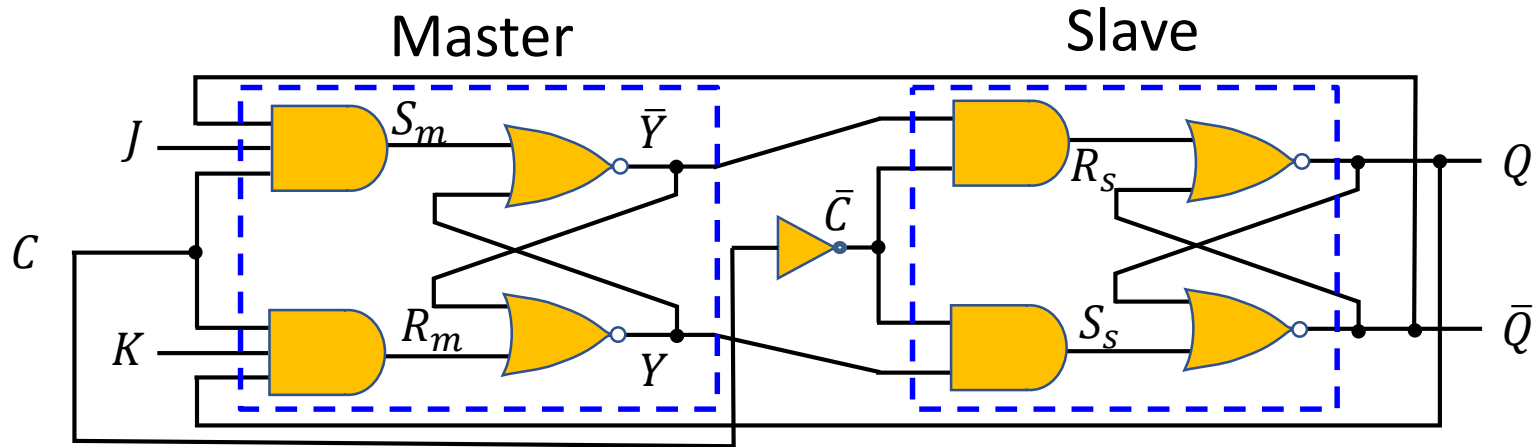


Master-Slave JK Flip-Flop



- Two JK FFs connected in series
- Outputs are fed back to Master's inputs $Q \rightarrow K$ & $\bar{Q} \rightarrow J$
- J is SET and K is RESET
- Y is connected to SET of slave FF

Master-Slave JK Flip-Flop



$$\textcircled{1} J = 1, K = 0,$$

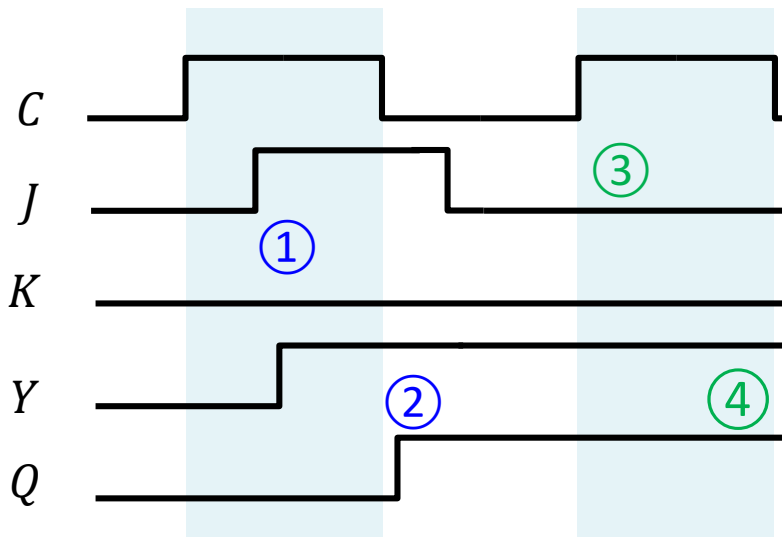
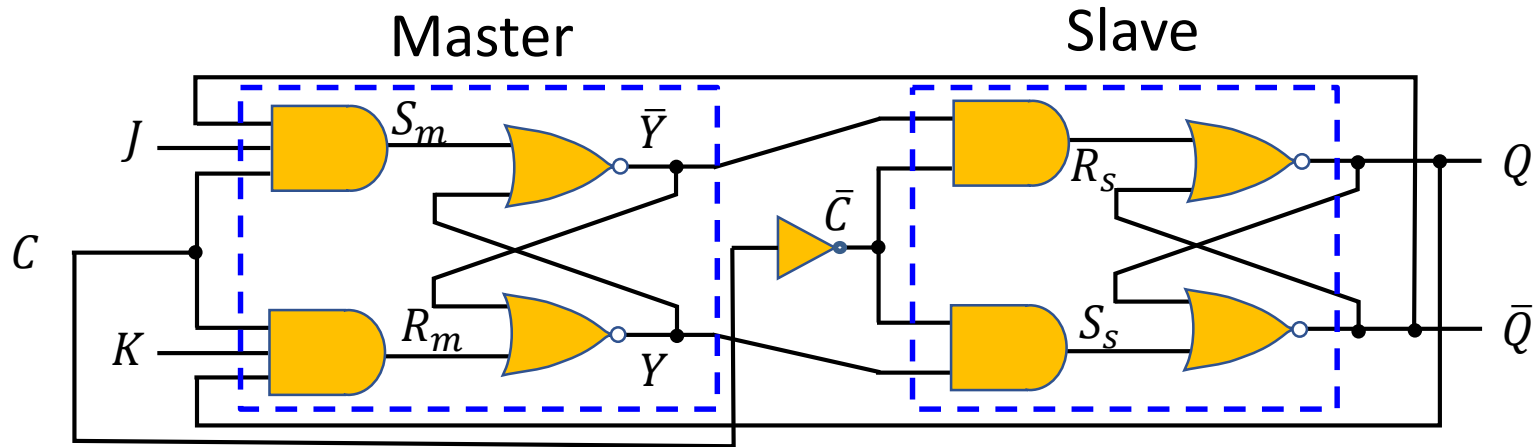
$$S_m = \bar{Q} = 1, R_m = 0 (\text{SET})$$

$$\therefore Y = 1$$

$$\textcircled{2} Y(S_s) = 1, \bar{Y}(R_s) = 0 (\text{SET})$$

$$\therefore Q = 1$$

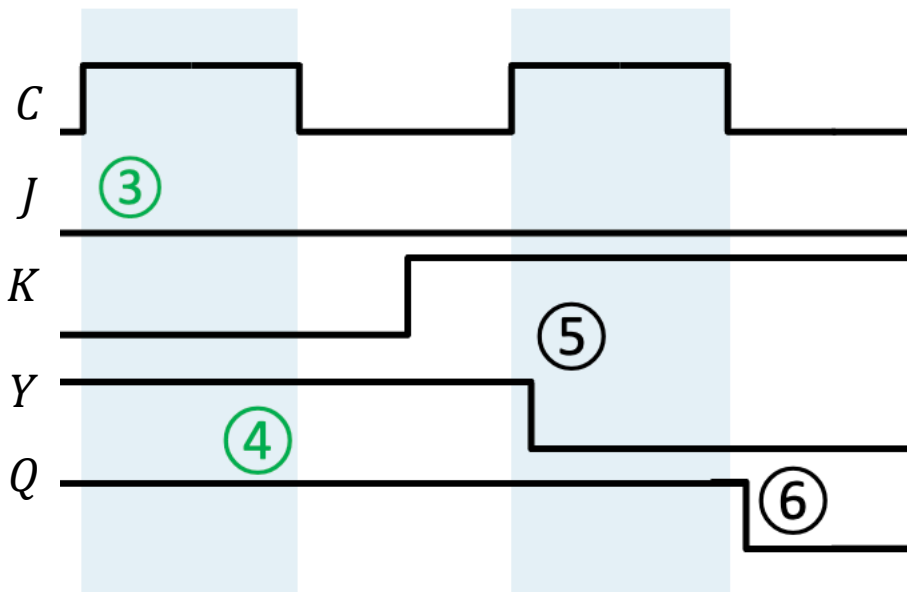
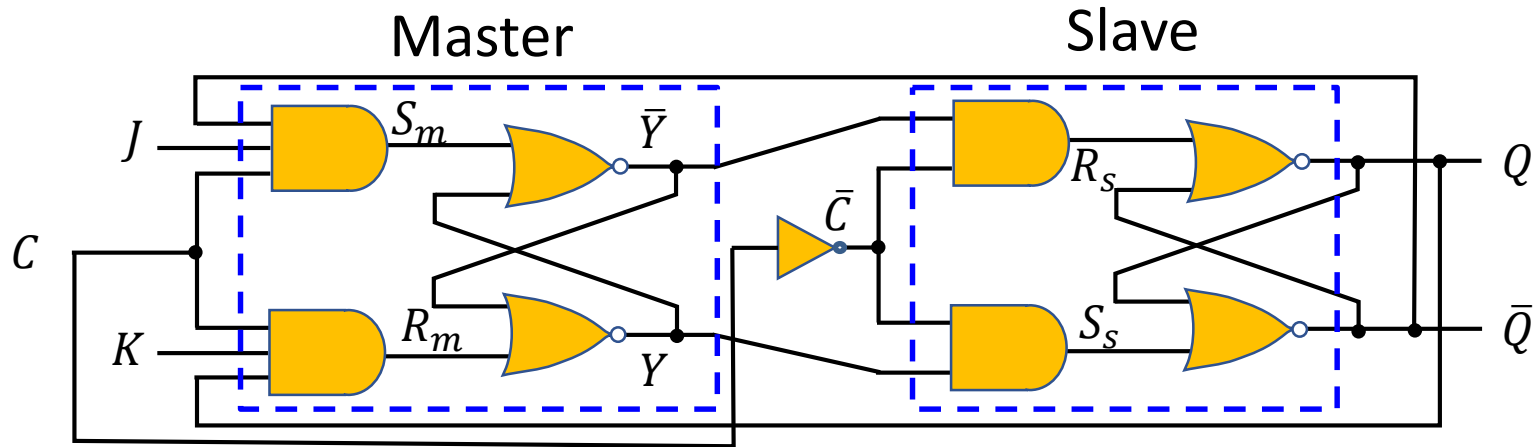
Master-Slave JK Flip-Flop



③ $J = 0, K = 0$ (HOLD)
 $\therefore Y = 1$

④ $Y(S_s) = 1, \bar{Y}(R_s) = 0$ (SET)
 $\therefore Q = 1$

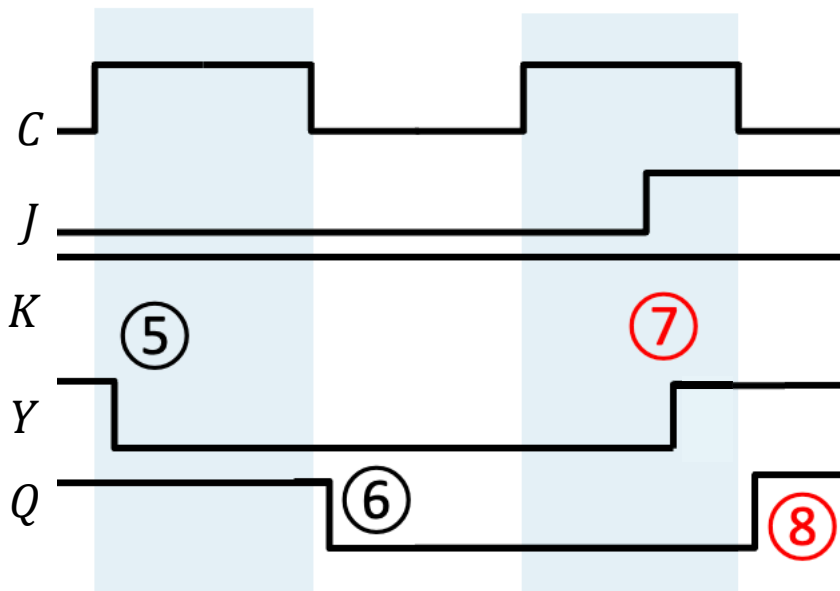
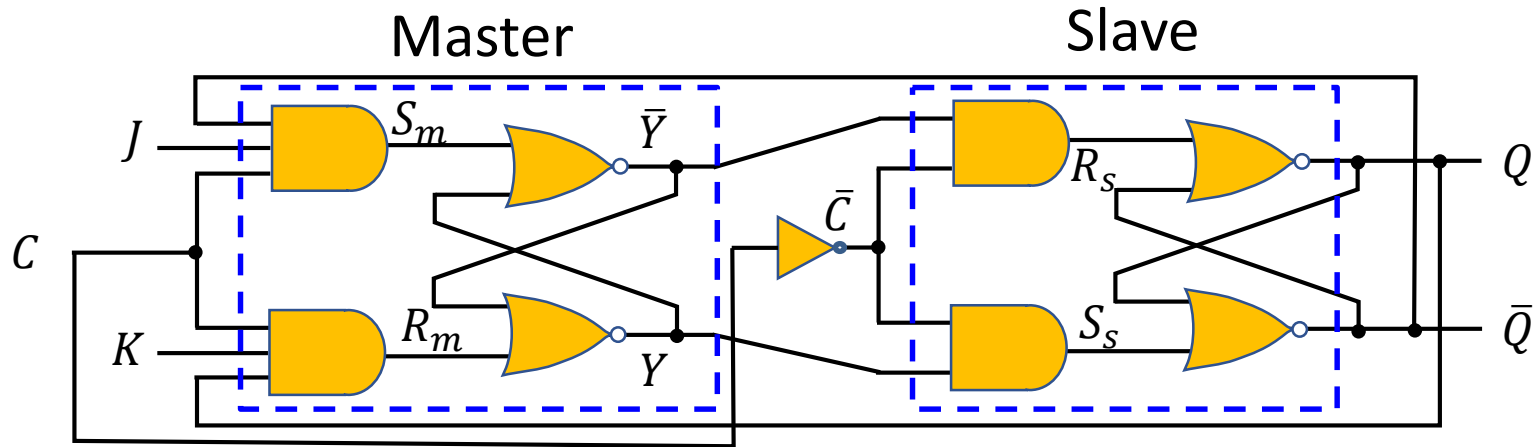
Master-Slave JK Flip-Flop



⑤ $J = 0, K = 1$ (RESET)
 $\therefore Y = 0$

⑥ $Y(S_s) = 0, \bar{Y}(R_s) = 1$
 (RESET)
 $\therefore Q = 0$

Master-Slave JK Flip-Flop



⑦ $J = 1, K = 1$

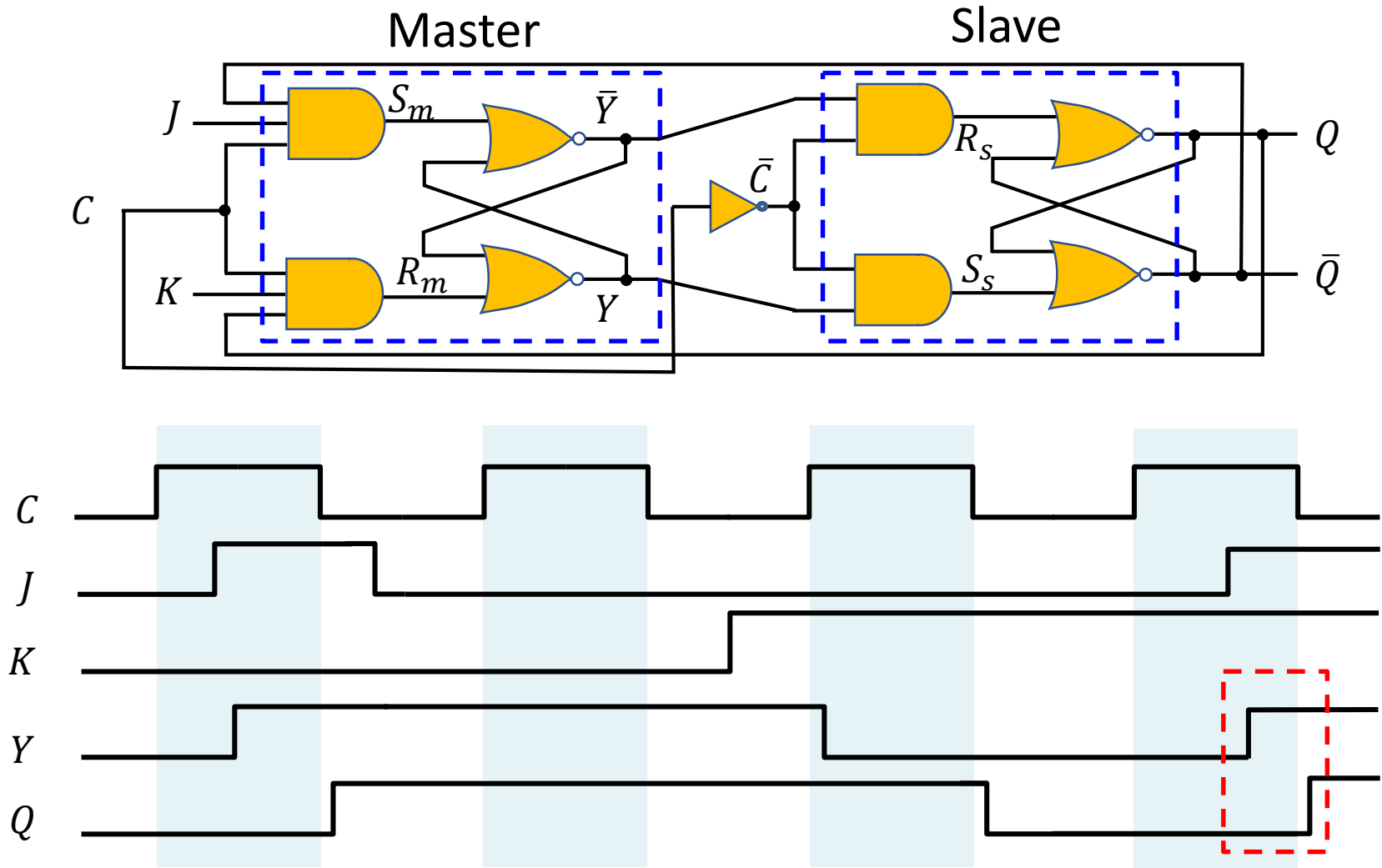
$S_m = \bar{Q} = 1, R_m = 0$ (SET)

$\therefore Y = 1$ at the point $C \rightarrow 0$

⑧ $Y(S_s) = 1, \bar{Y}(R_s) = 0$
(SET)

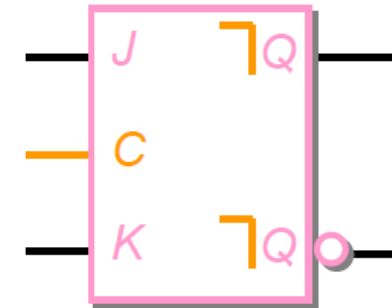
$\therefore Q = 1$

Master-Slave JK Flip-Flop







No racing!

Master-Slave JK Flip-Flop



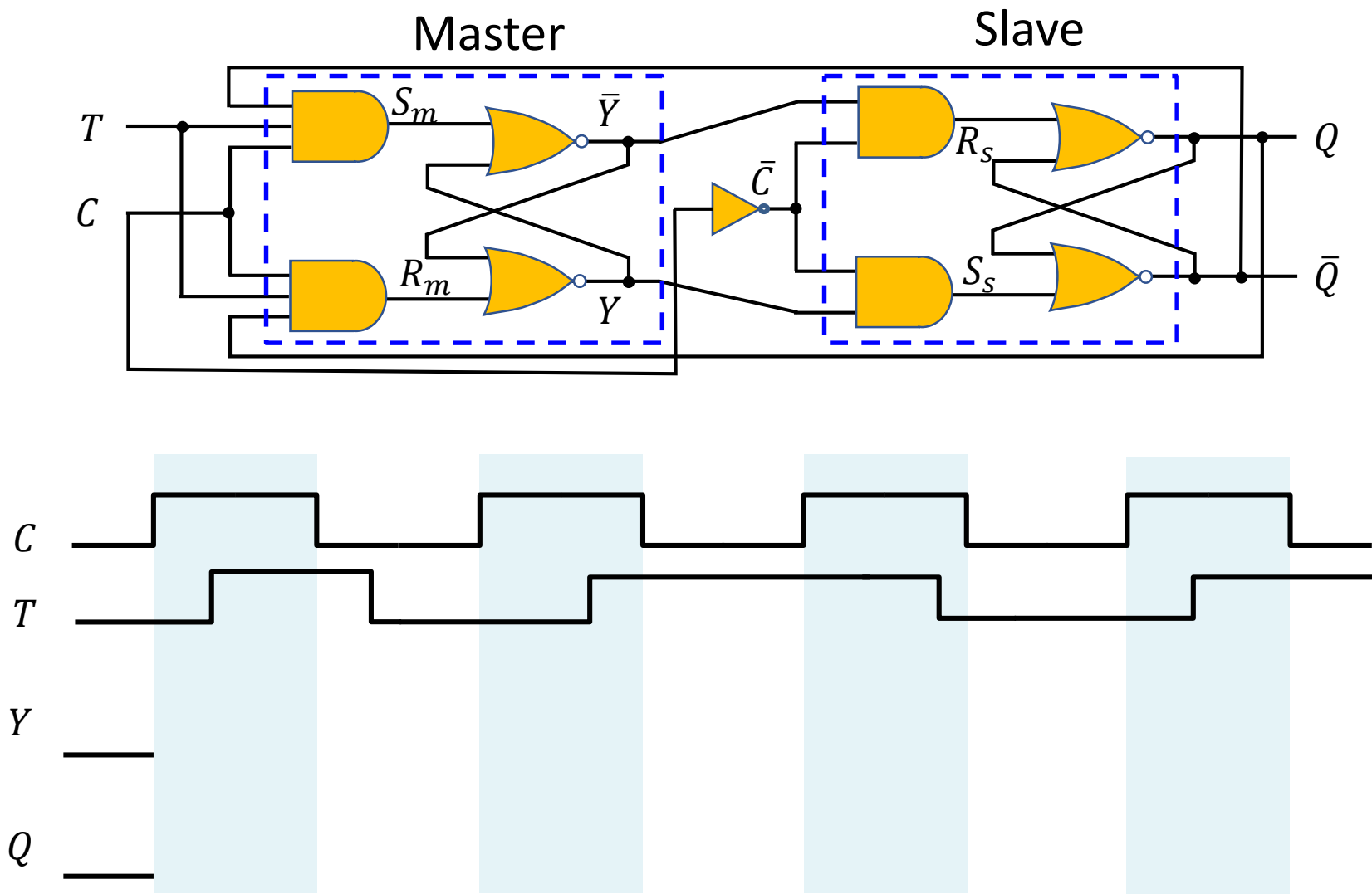
Symbol

Inputs			Outputs		
<i>J</i>	<i>K</i>	<i>Clk</i>	Next <i>Q</i>	Next <i>Q'</i>	Meaning
x	x	0	<i>Q</i>	<i>Q'</i>	Hold
0	0		<i>Q</i>	<i>Q'</i>	Hold
0	1		0	1	Reset
1	0		1	0	Set
1	1		<i>Q'</i>	<i>Q</i>	Toggle

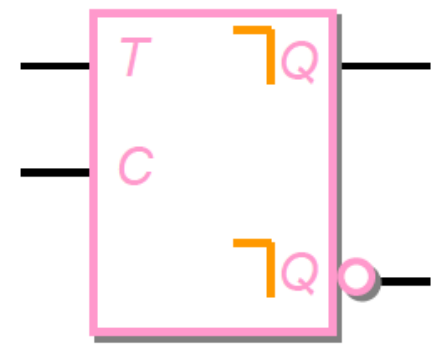
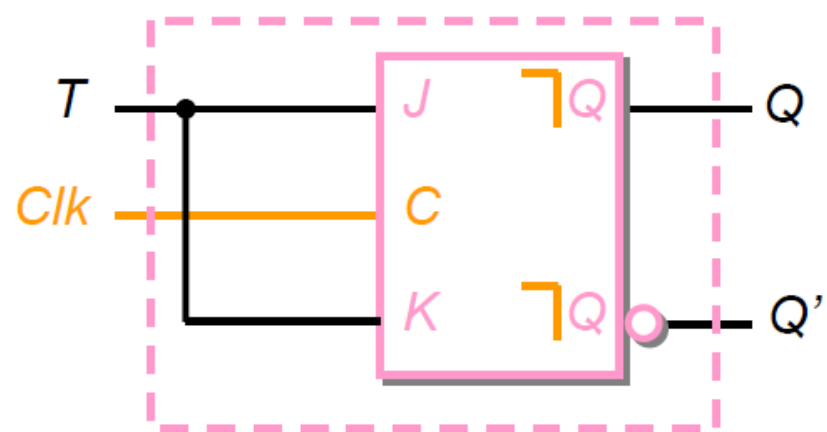
The same behaviors as
master-slave *SR* flip-flop

Eliminated the undefined state

Exercise



Master-Slave Toggle Flip-Flop



Symbol

Inputs		Outputs		
T	Clk	Next Q	Next Q'	Meaning
x	0	Q	Q'	Hold
0		Q	Q'	Hold
1		Q'	Q	Toggle

Problem

- For the presented MSFF, master is enabled during the period when the clock pulse is 1 (Level triggered)
- Undesired behavior produced when inputs values of S and R (or J , K , D , T) change at that period, especially just before the clock pulse changes to 0
- Undesired output due to glitch could not be fully avoided
- Better solution: Edge-triggered FFs

7.4 Edge-triggered Flip-Flop

- Use one of the edges of the clock signal to read the input values
 - either positive or negative transition
 - triggering edge
- The response to the triggering edge at the output of the flip-flop is almost immediate
- The flip-flops remains unresponsive to the input change until the next triggering edge

Triggering edge

- Triggered only during a signal transition from 0 to 1 (or from 1 to 0) on the **clock**

- Positive (rising, leading) edge-triggering: 0-to-1 transition

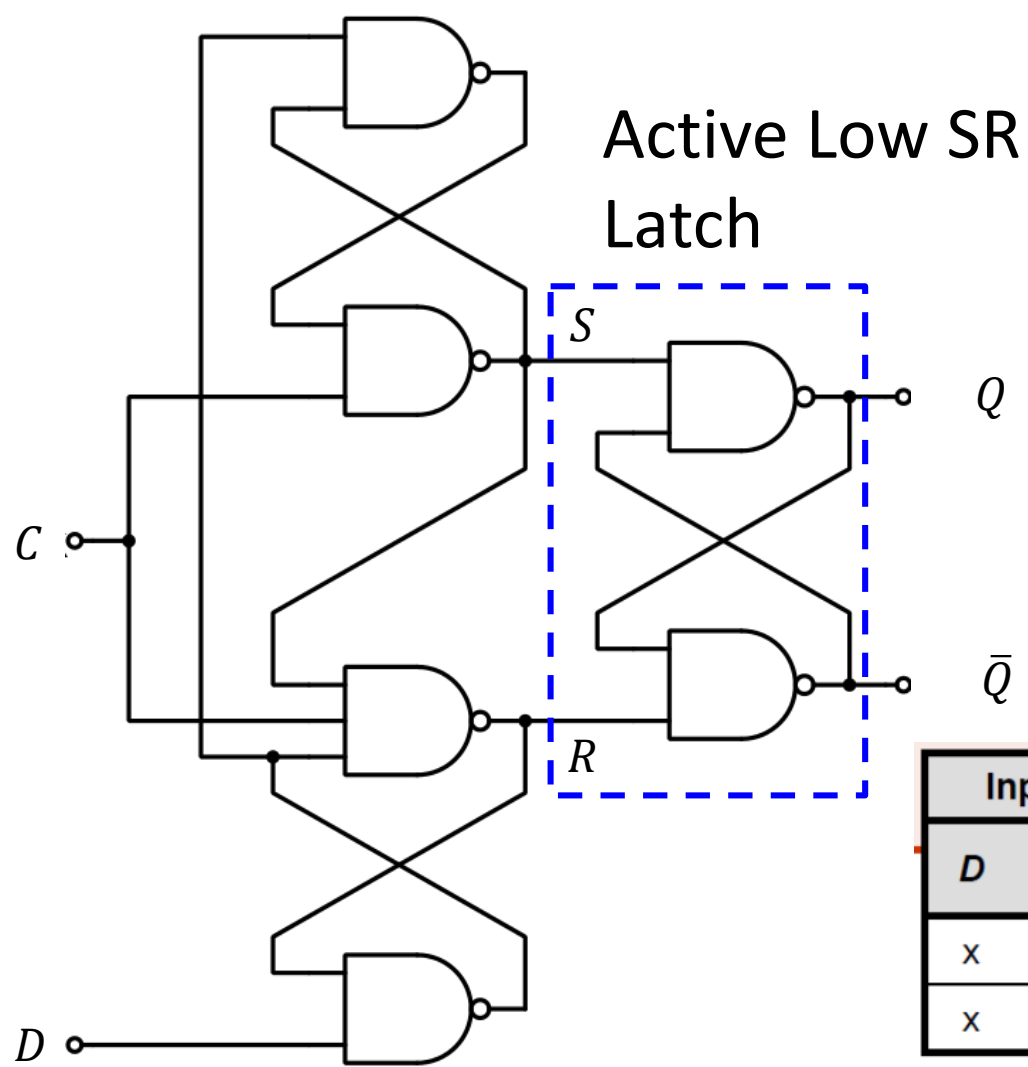


- Negative (falling, trailing) edge-triggering: 1-to-0 transition



- Additional circuit is included to ensure it will only response to the input at the transition edge of the clock pulse

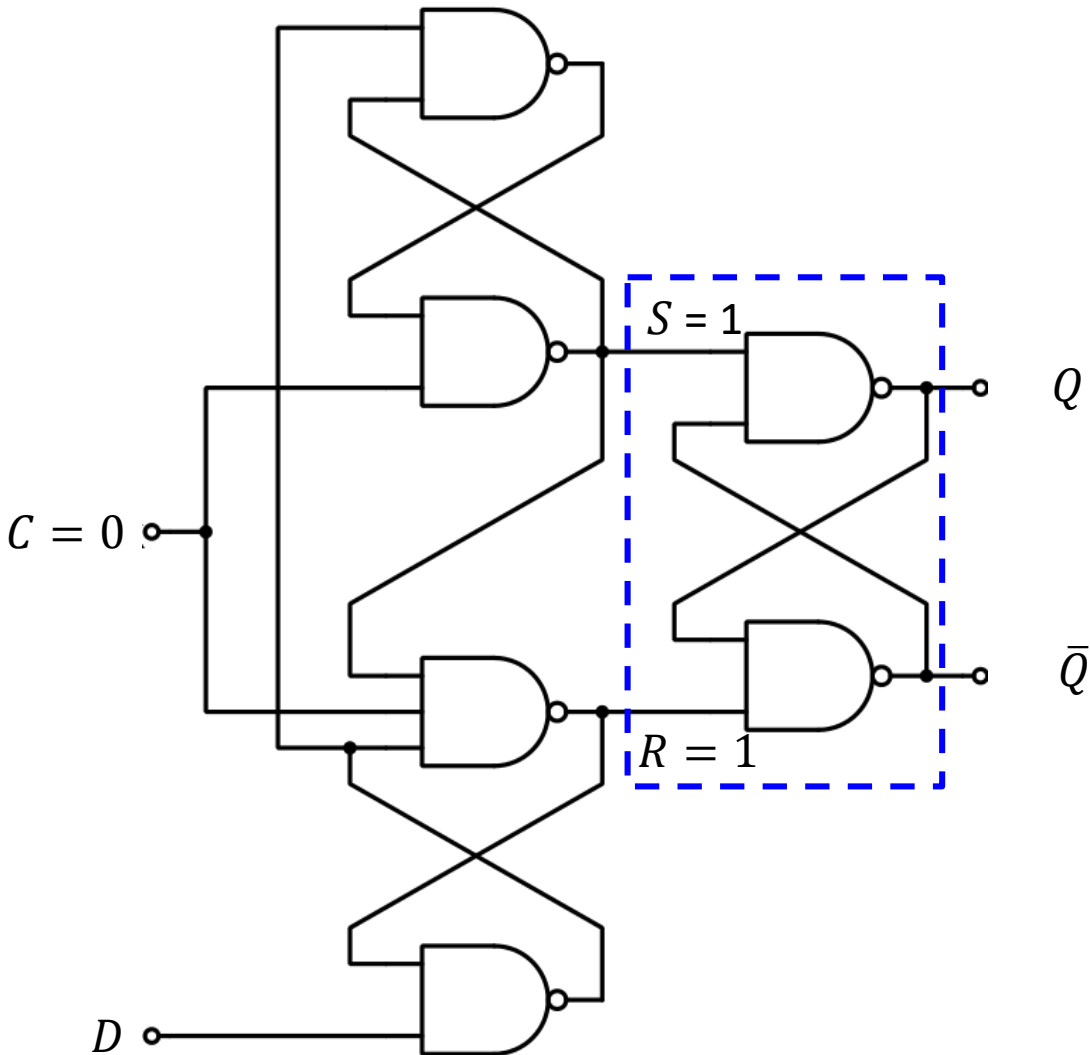
Positive-Edge-Triggered D-FF



S	R	Q_{t+1}	\overline{Q}_{t+1}
1	1	Q_t	\overline{Q}_t
1	0	0	1
0	1	1	0
0	0	1	1

Inputs		Outputs		
D	Clk	Next Q	Next Q'	State
x	$\downarrow, 0, 1$	Q	Q'	Hold
x	\uparrow	D	D'	Set / Reset

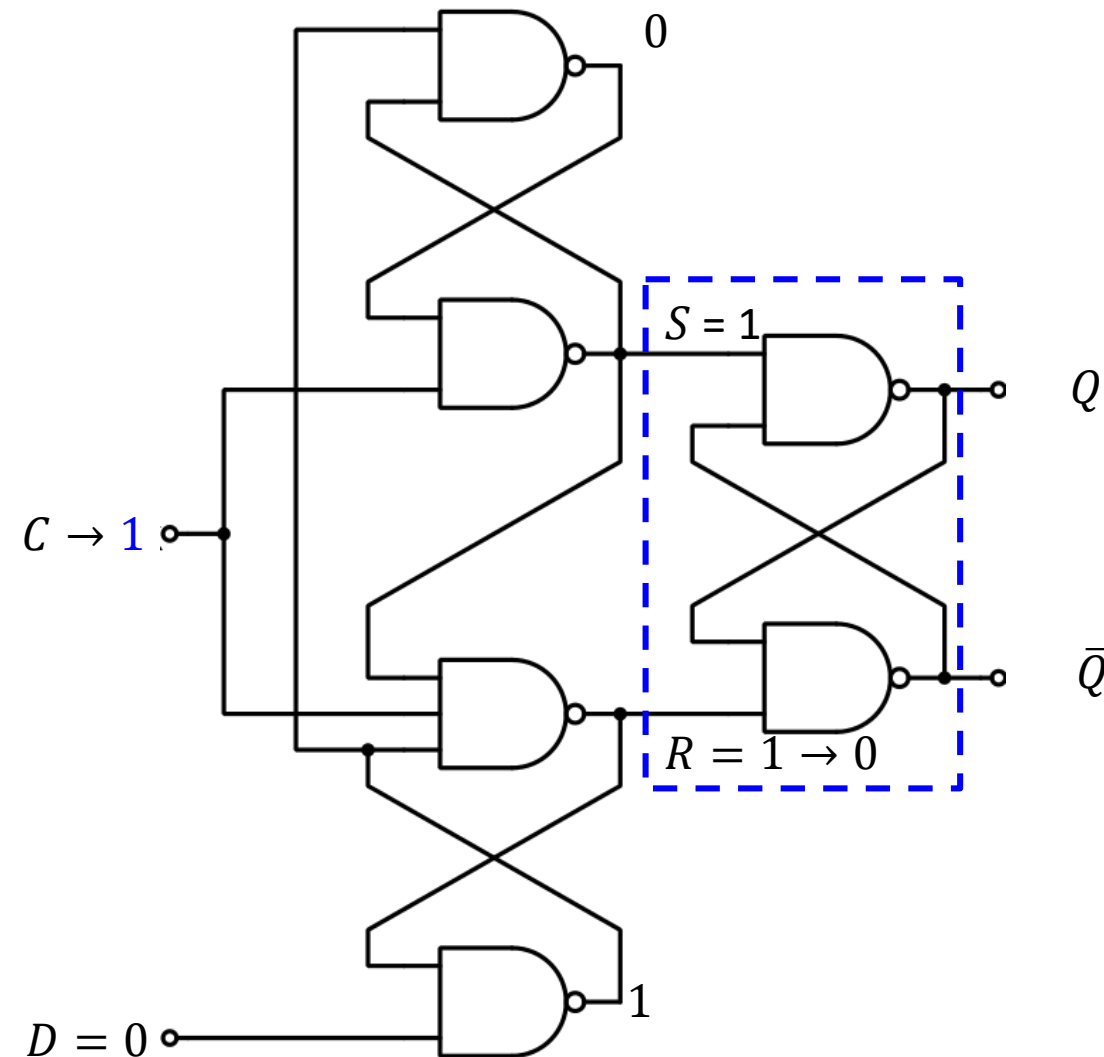
When $C = 0$



S	R	Q_{t+1}	$\overline{Q_{t+1}}$
1	1	Q_t	$\overline{Q_t}$
1	0	0	1
0	1	1	0
0	0	1	1

HOLD!

When $C = 0 \rightarrow 1$ & $D = 0$

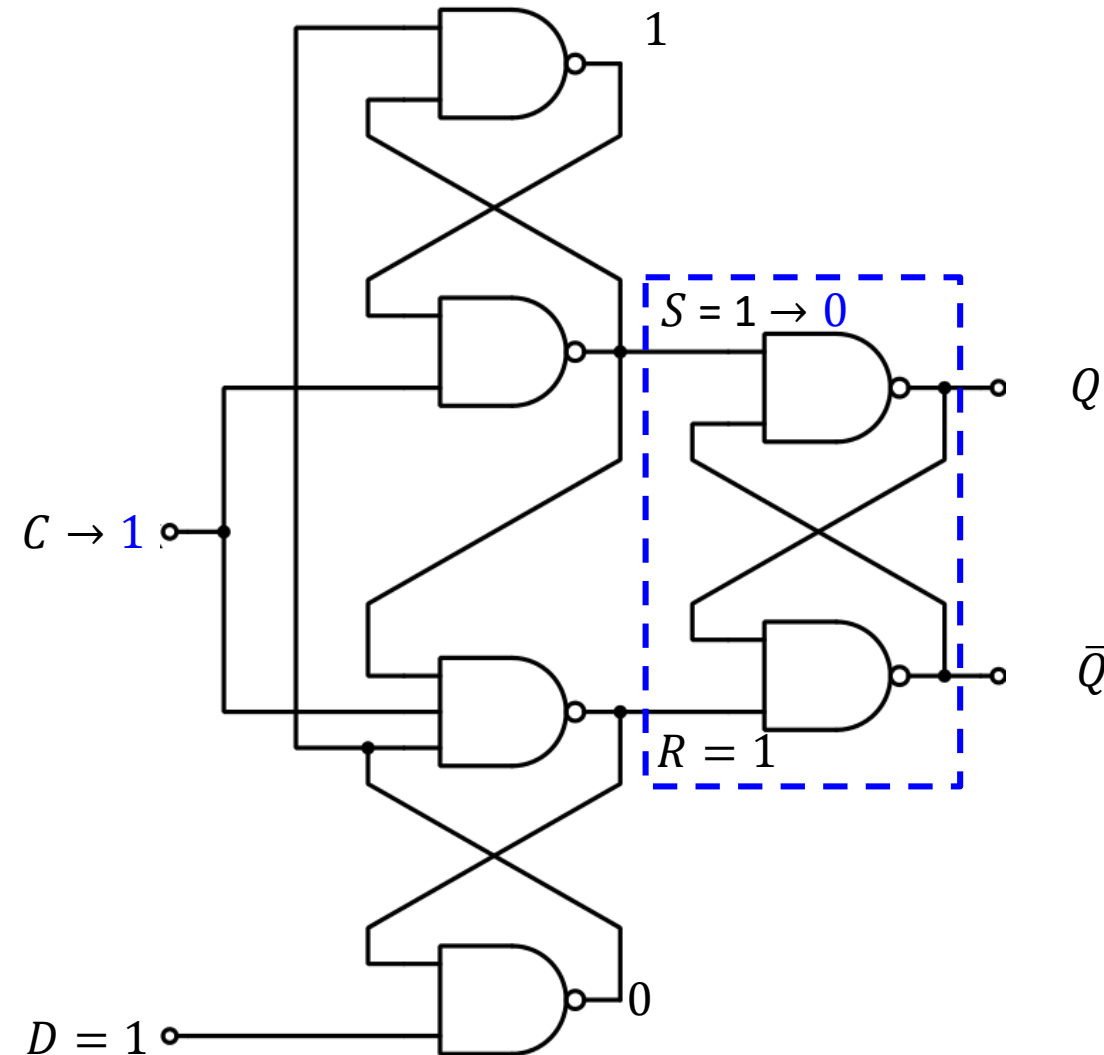


S	R	Q_{t+1}	$\overline{Q_{t+1}}$
1	1	Q_t	$\overline{Q_t}$
1	0	0	1
0	1	1	0
0	0	1	1

HOLD \rightarrow RESET

When $C = 0 \rightarrow 1$ & $D = 0$, $Q \rightarrow 0$

When $C = 0 \rightarrow 1$ & $D = 1$

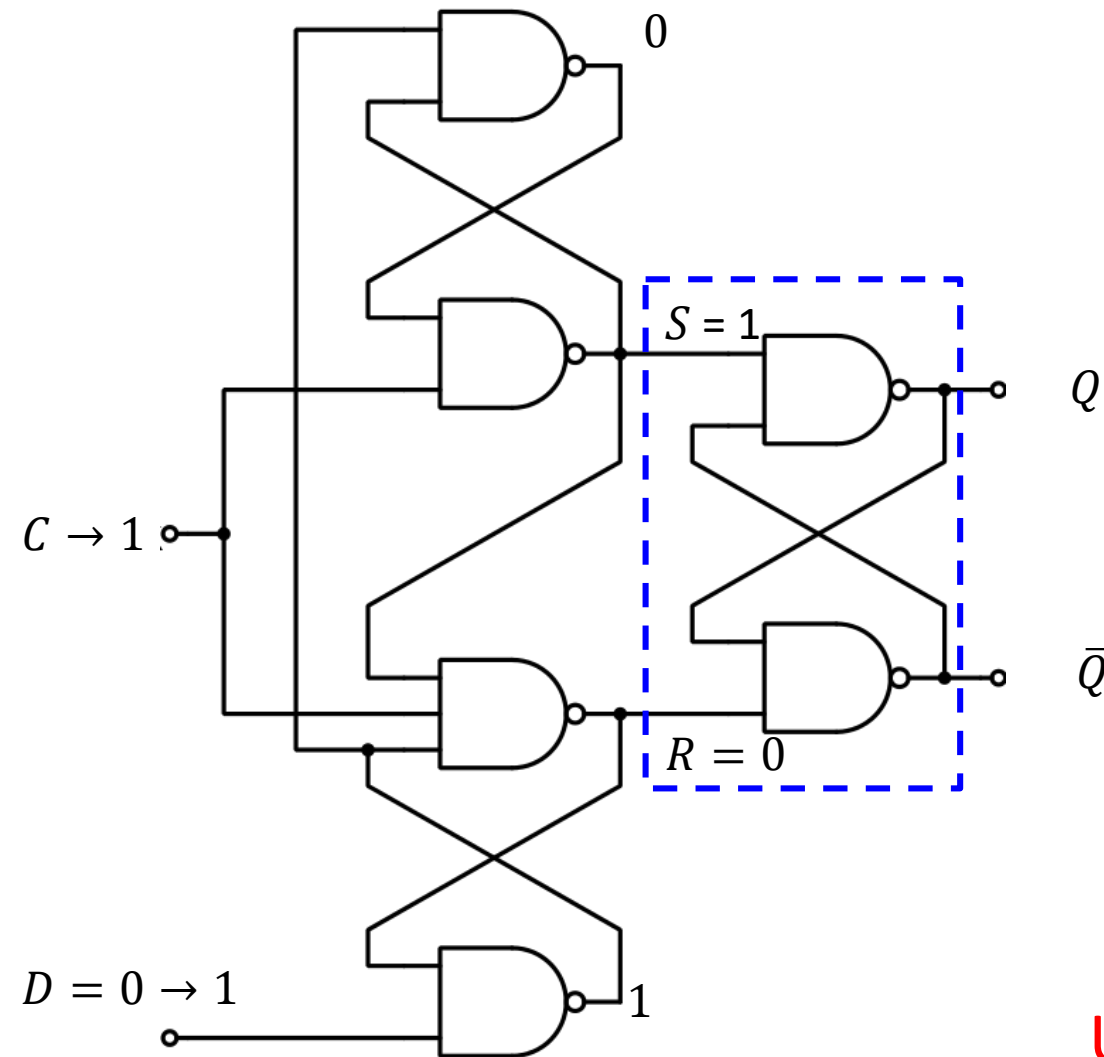


S	R	Q_{t+1}	\bar{Q}_{t+1}
1	1	Q_t	\bar{Q}_t
1	0	0	1
0	1	1	0
0	0	1	1

HOLD \rightarrow SET

When $C = 0 \rightarrow 1$ & $D = 1$, $Q \rightarrow 1$

When $C = 1$ & $D = 0 \rightarrow 1$



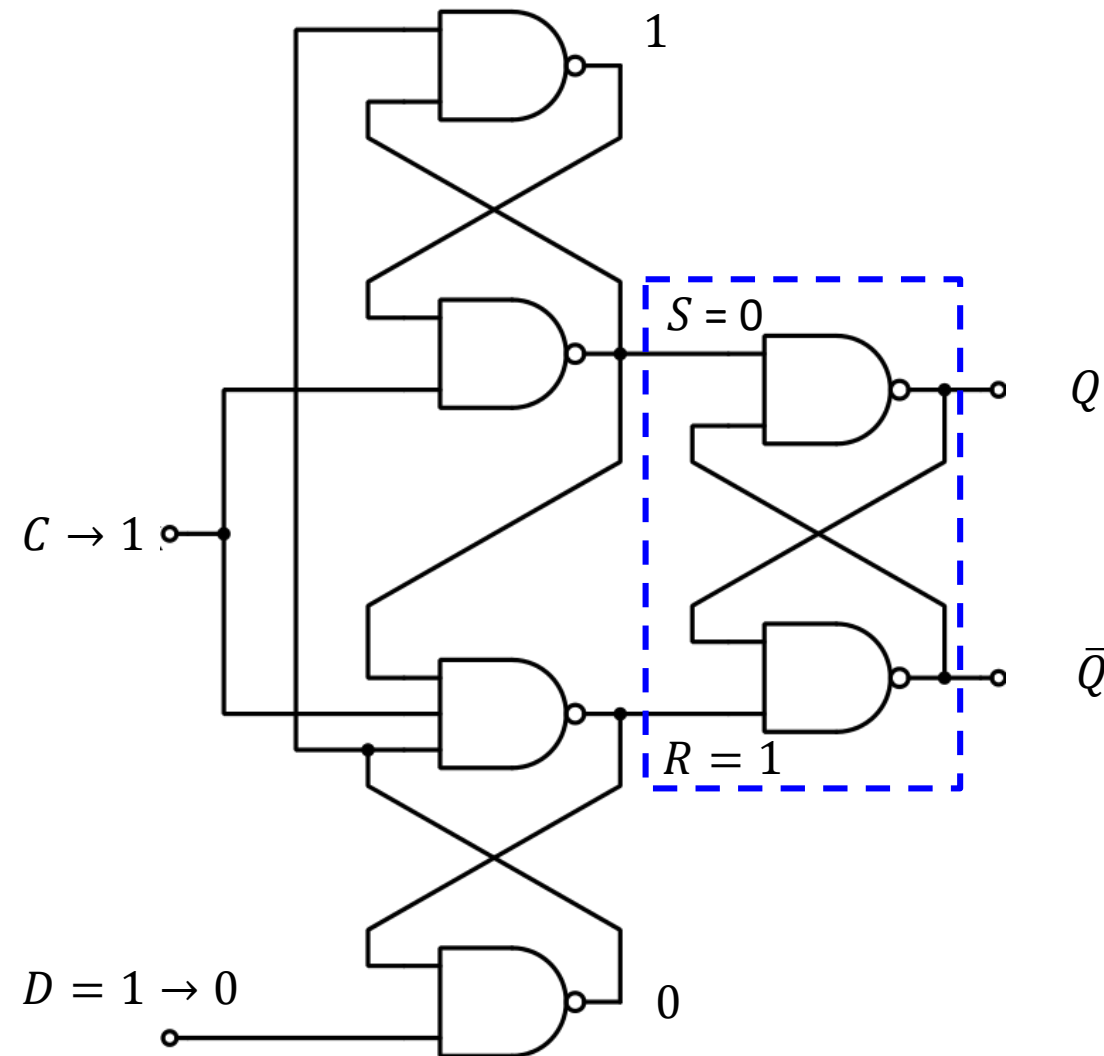
S	R	Q_{t+1}	\bar{Q}_{t+1}
1	1	Q_t	\bar{Q}_t
1	0	0	1
0	1	1	0
0	0	1	1

RESET \rightarrow RESET

When $C = 1$ & $D = 0$
 $\rightarrow 1, Q = 0$

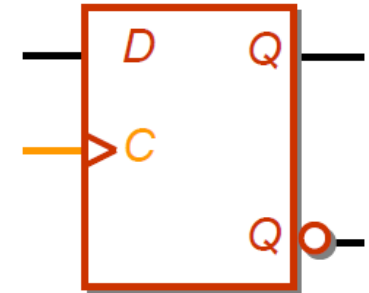
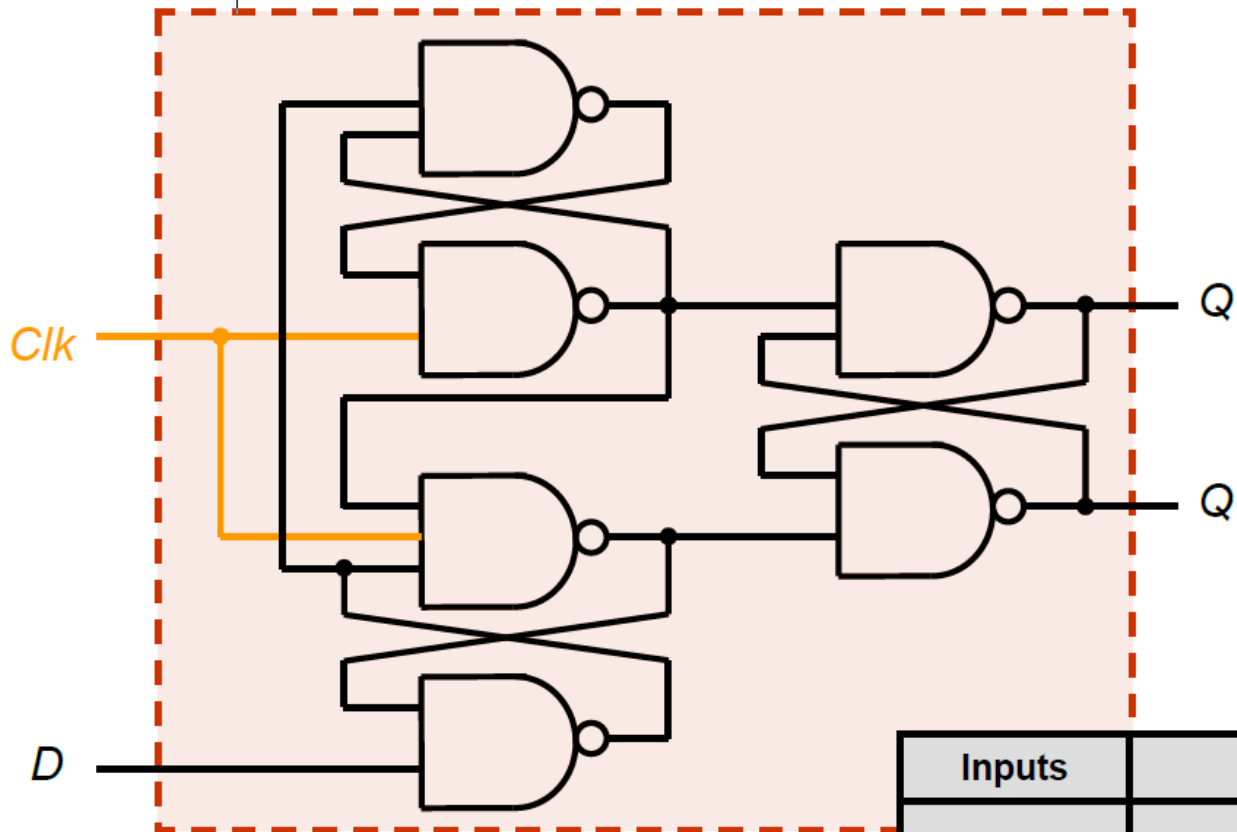
Unresponsive!

When $C = 1$ & $D = 1 \rightarrow 0$



S	R	Q_{t+1}	$\overline{Q_{t+1}}$
1	1	Q_t	$\overline{Q_t}$
1	0	0	1
0	1	1	0
0	0	1	1

Positive-Edge-Triggered D-FF

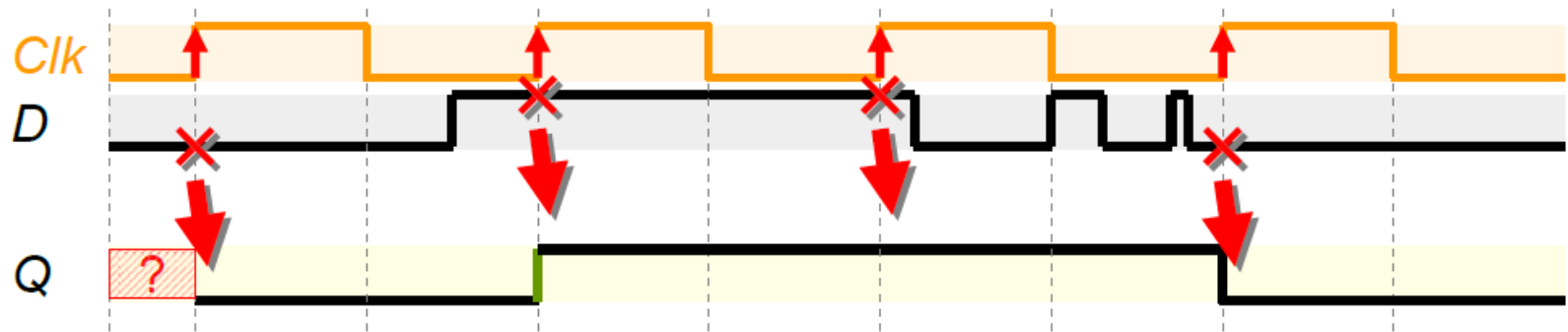


Remember the symbol only!

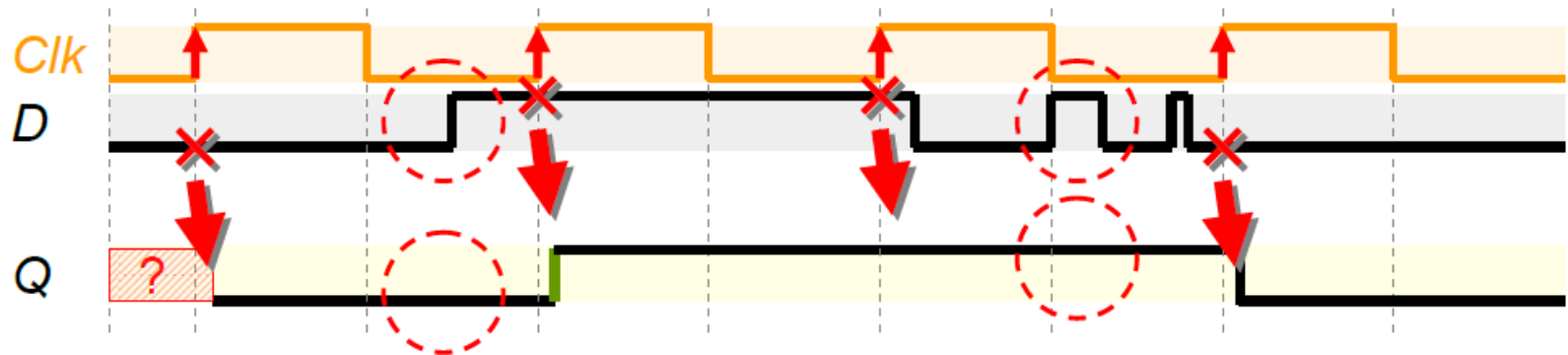
Inputs		Outputs		
D	Clk	Next Q	Next Q'	State
x	$\downarrow, 0, 1$	Q	Q'	Hold
x	\uparrow	D	D'	Set / Reset

Timing Diagram

Timing Diagram of PET D FF (if no output delay)



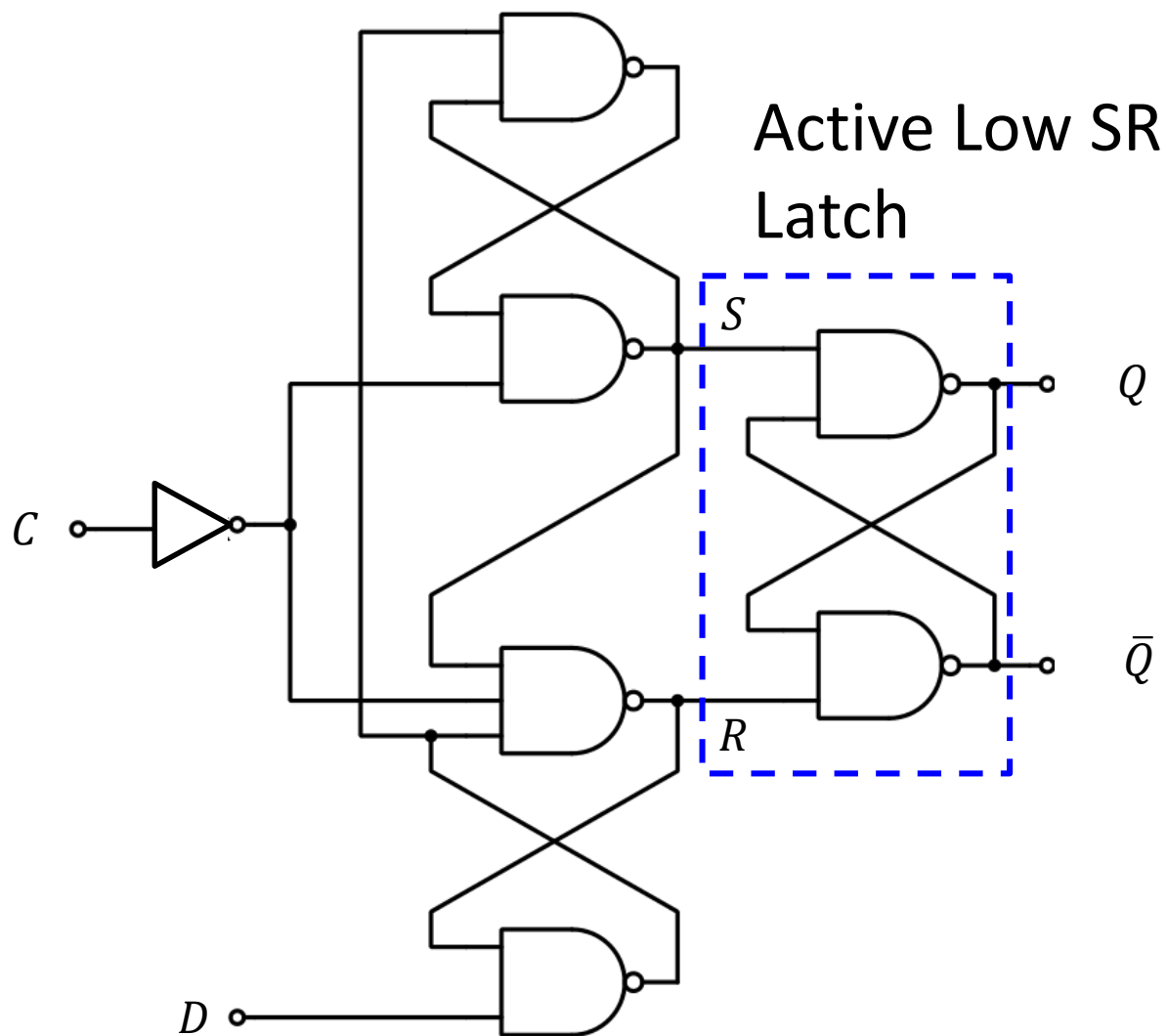
Timing Diagram of PET D FF



Thought

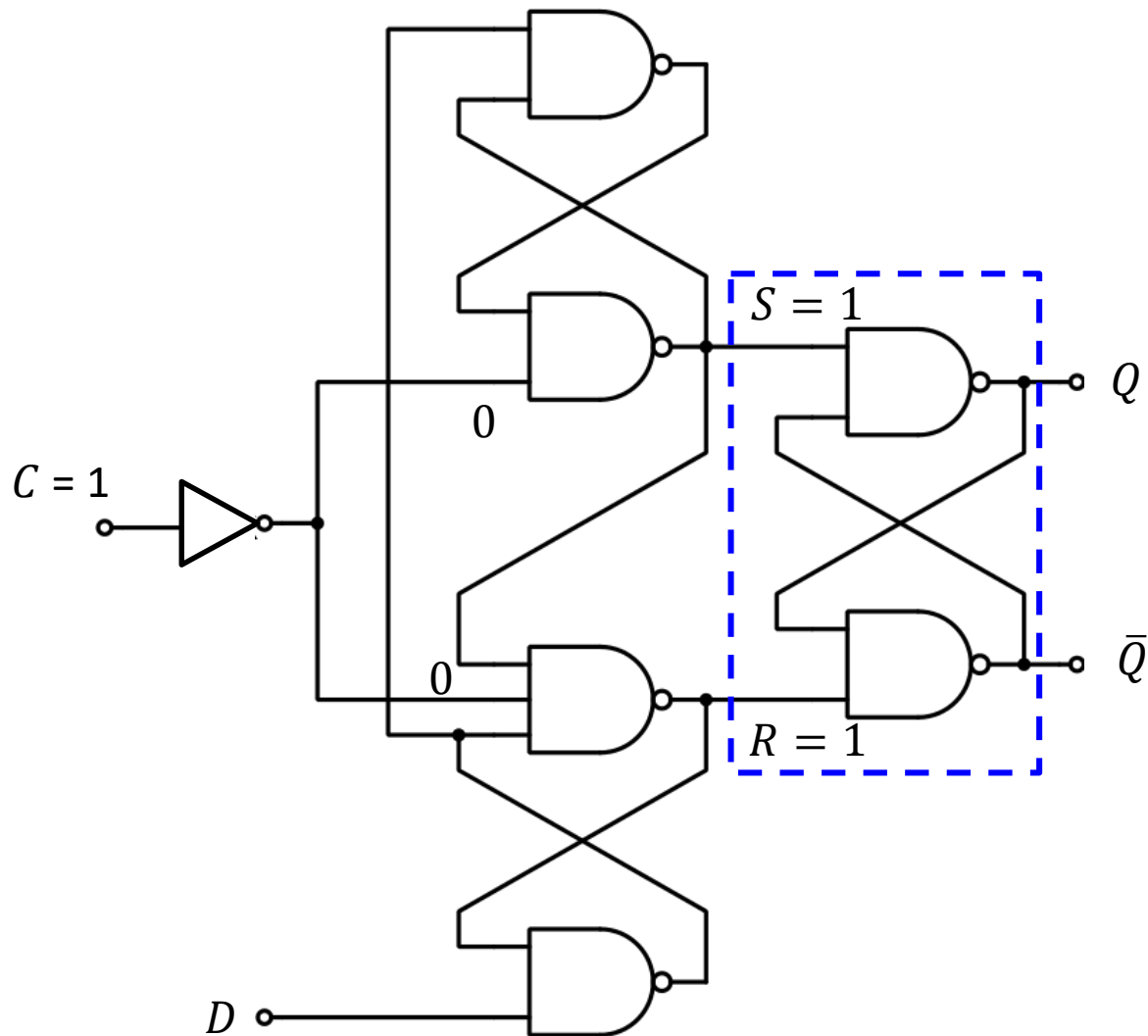
- How to build a negative-edge triggered D-FF?

Negative-Edge-Triggered D-FF



S	R	Q_{t+1}	$\overline{Q_{t+1}}$
1	1	Q_t	$\overline{Q_t}$
1	0	0	1
0	1	1	0
0	0	1	1

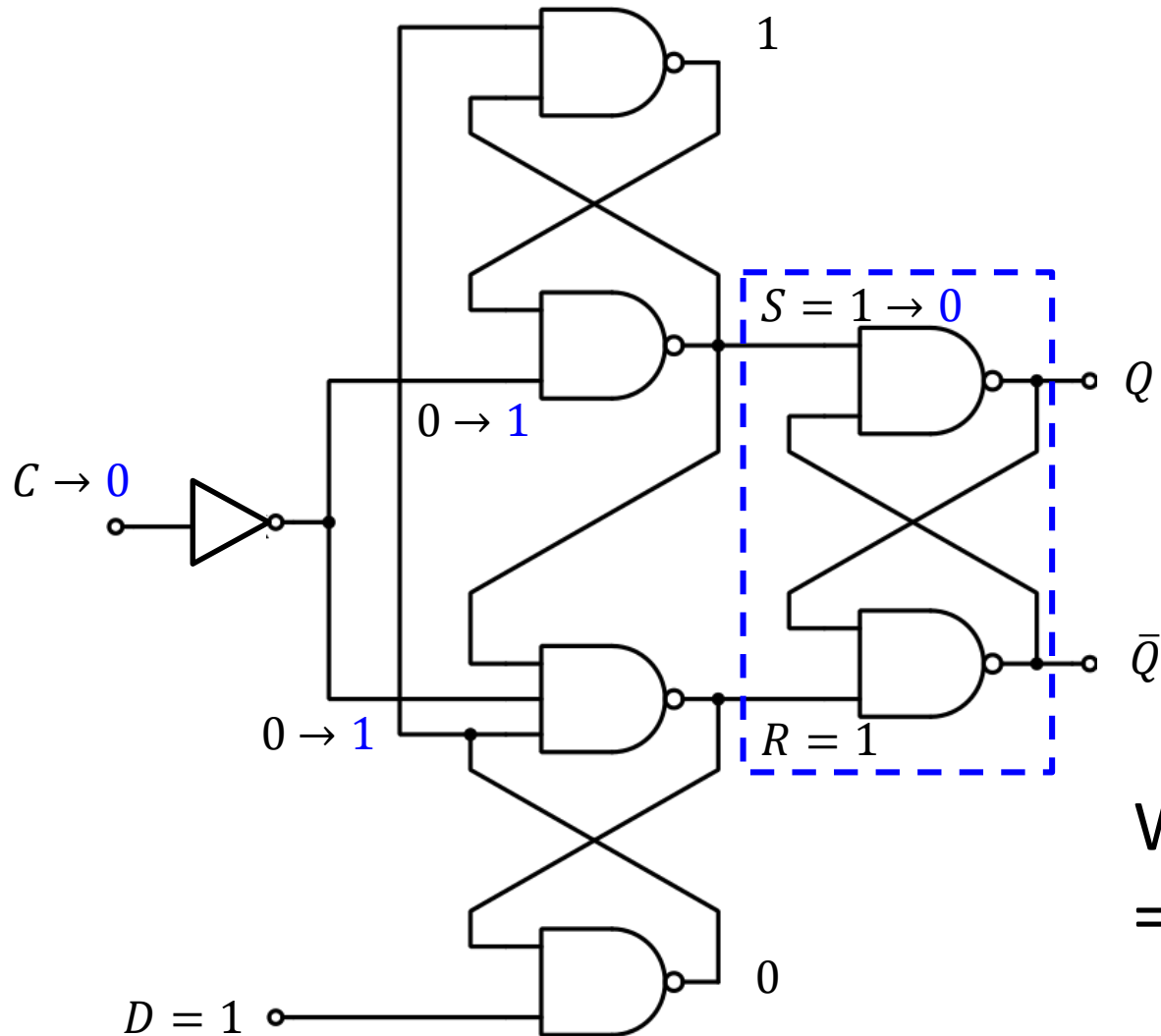
When $C = 1$



S	R	Q_{t+1}	$\overline{Q_{t+1}}$
1	1	Q_t	$\overline{Q_t}$
1	0	0	1
0	1	1	0
0	0	1	1

HOLD!

When $C = 1 \rightarrow 0$ & $D = 1$

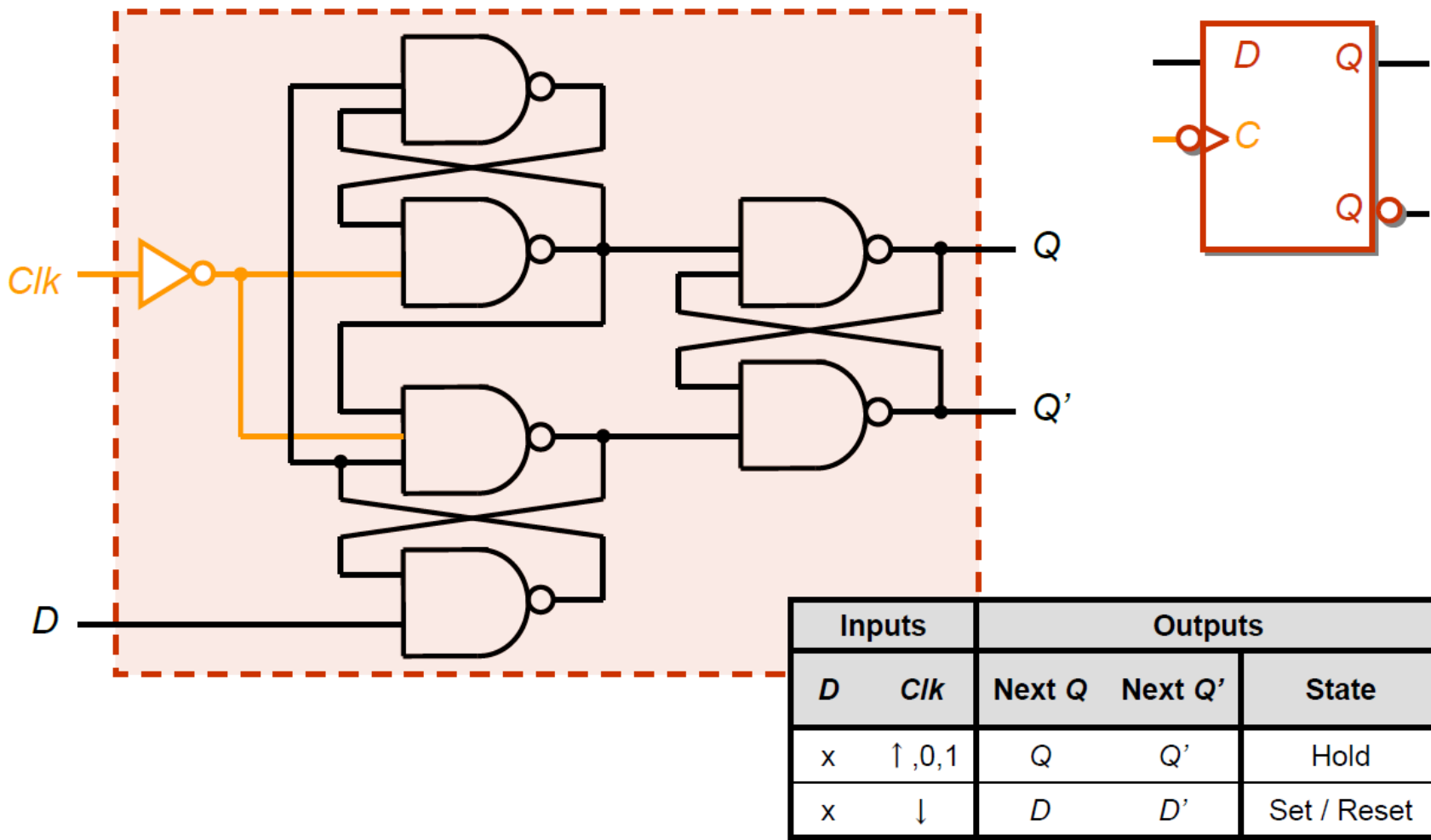


S	R	Q_{t+1}	\bar{Q}_{t+1}
1	1	Q_t	\bar{Q}_t
1	0	0	1
0	1	1	0
0	0	1	1

HOLD \rightarrow SET

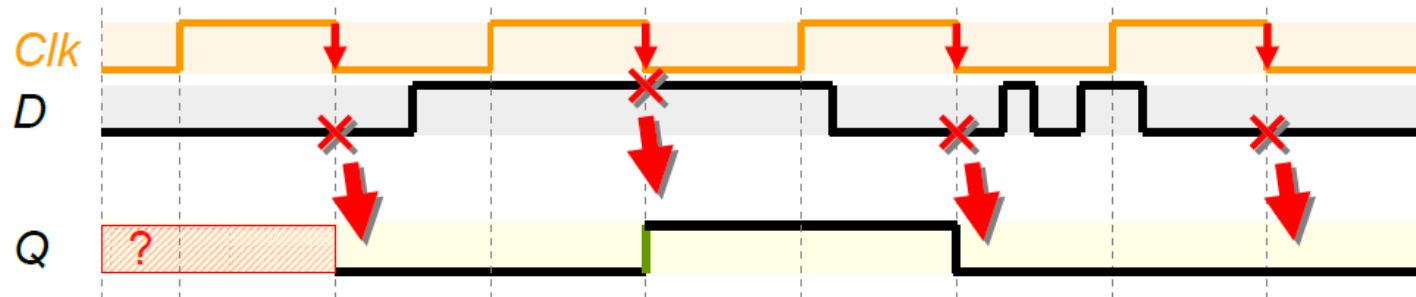
When $C = 1 \rightarrow 0$ & $D = 1$, $Q \rightarrow 1$

Negative-Edge-Triggered D-FF

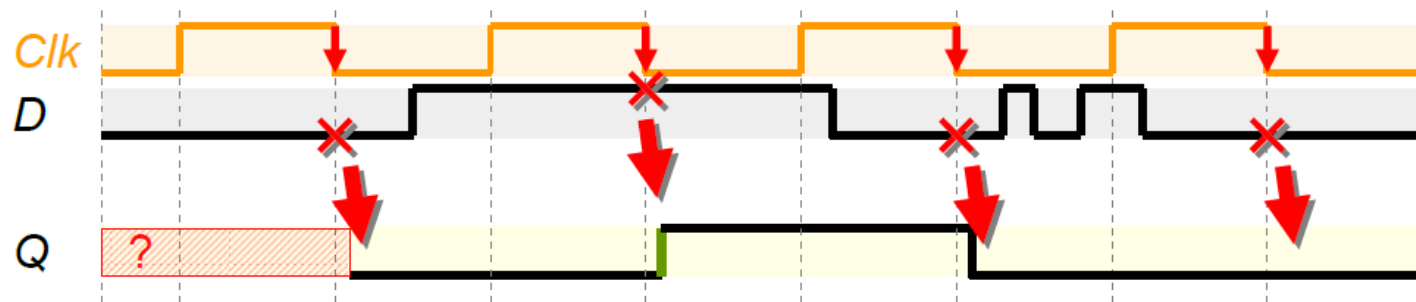


Timing Diagram

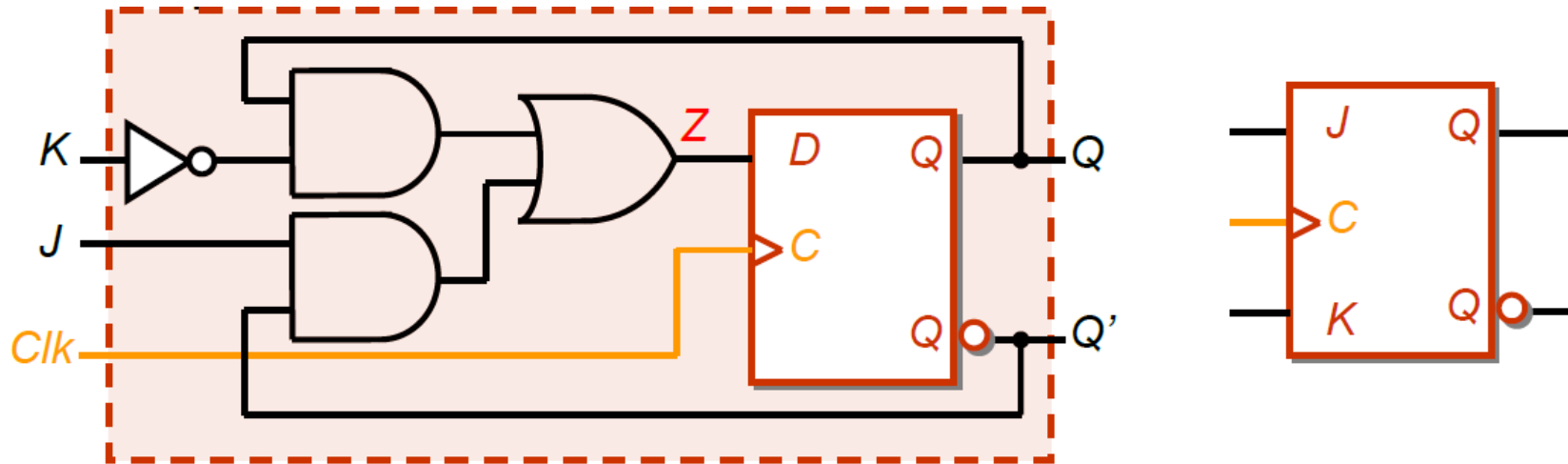
Timing Diagram of NET D FF (if no output delay)



Timing Diagram of NET D FF



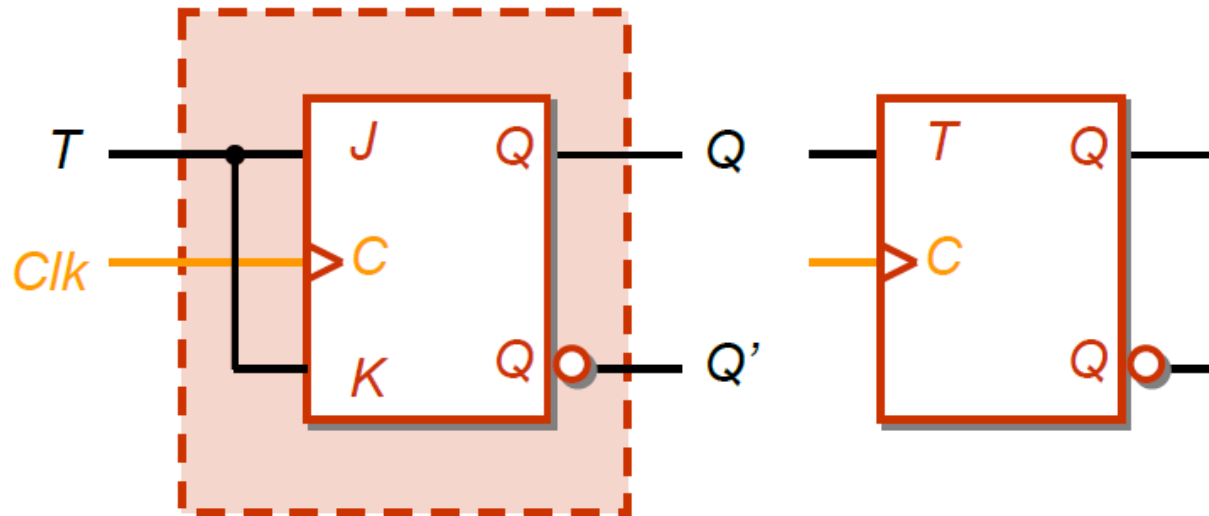
Edge-Triggered JK-FF



$$D = Q\bar{K} + J\bar{Q}$$

Inputs			Outputs		
<i>J</i>	<i>K</i>	<i>Clk</i>	Next <i>Q</i>	Next <i>Q'</i>	State
x	x	↓,0,1	<i>Q</i>	<i>Q'</i>	Hold
0	0	↑	<i>Q</i>	<i>Q'</i>	Hold
0	1	↑	0	1	Reset
1	0	↑	1	0	Set
1	1	↑	<i>Q'</i>	<i>Q</i>	Toggle

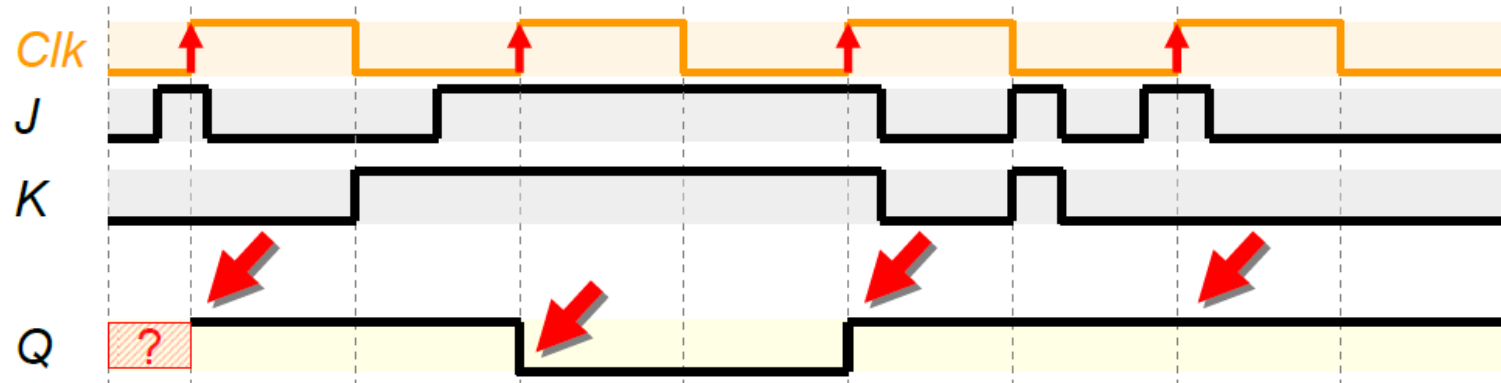
Edge-Triggered T-FF



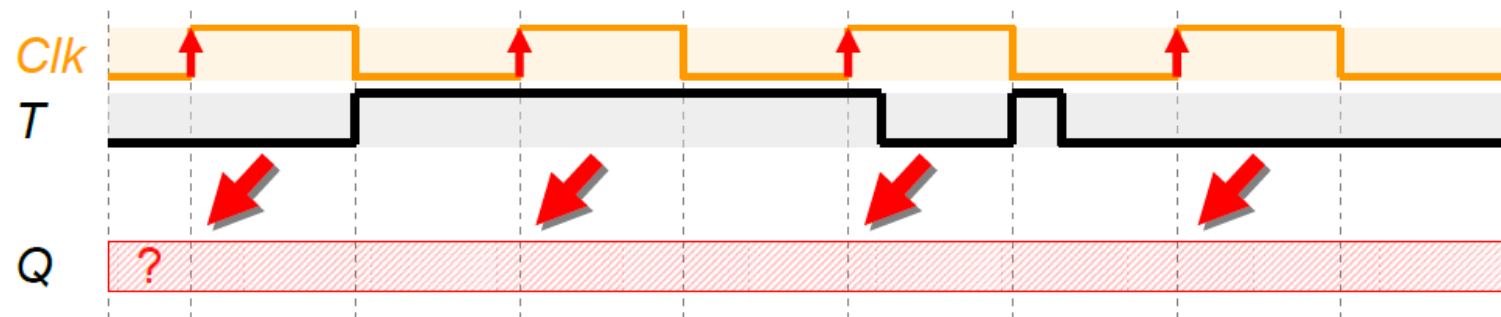
Inputs		Outputs		
T	Clk	Next Q	Next Q'	State
x	$\downarrow, 0, 1$	Q	Q'	Hold
0	\uparrow	Q	Q'	Hold
1	\uparrow	Q'	Q	Toggle

Timing Diagram

Timing Diagram of PET JK FF (if no output delay)

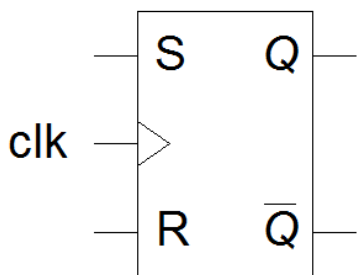


Timing Diagram of PET T FF (if no output delay)

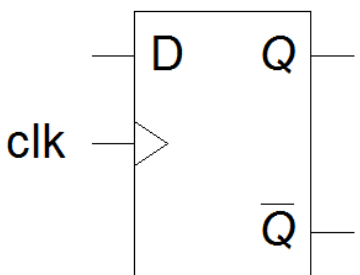


Logic Symbols

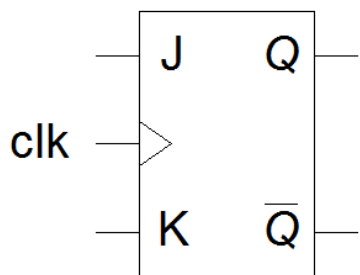
Positive-Edge



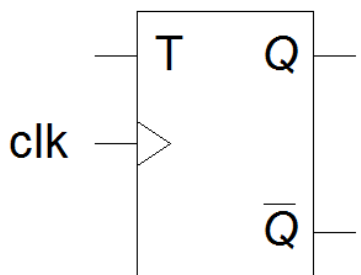
RS flip-flop



D flip-flop

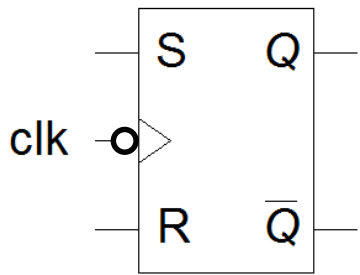


JK flip-flop

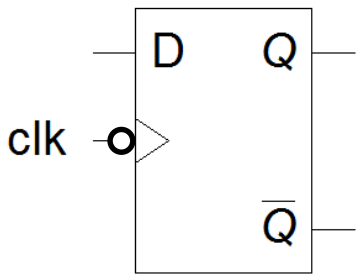


T flip-flop

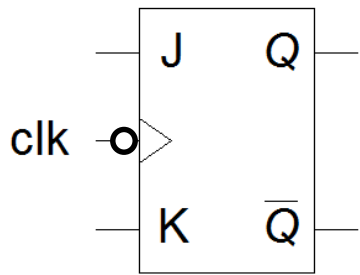
Negative-Edge



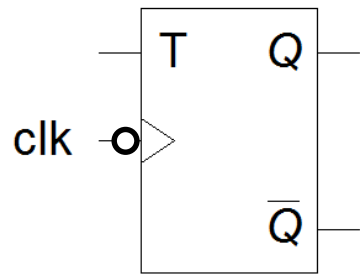
RS flip-flop



D flip-flop



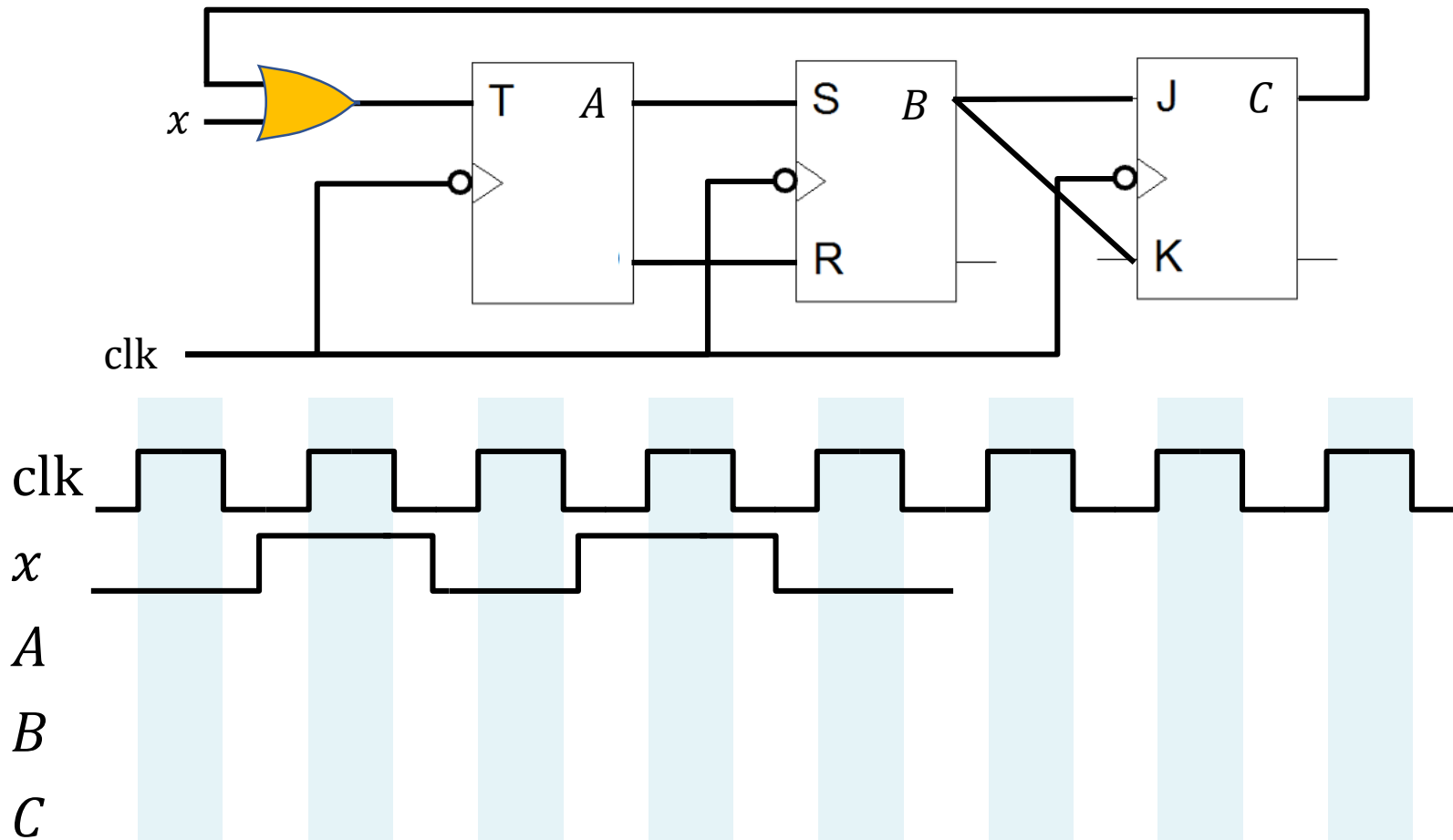
JK flip-flop



T flip-flop

Exercise

Assume all FFs are initially 0 and no propagation delay, work out the timing diagram of the output of each FF (A , B , C).



7.5 Synchronous & Asynchronous Inputs

- Inputs of flip-flops are categorized into two types
- **Synchronous:** Only affect the output in conjunction with a clock pulse (all inputs to FFs presented so far)
- **Asynchronous:** Inputs that can affect the output state independent of the clock pulse

Asynchronous Inputs

- The output will response to the sync. inputs only if all async. inputs are inactive
- Particularly useful for bringing a FF into a desired initial state prior to normal clocked operation
- The asynchronous (or direct) inputs usually called - **Preset** (denoted by *PRE*) and **Clear** (denoted *CLR*)
 - Used to forcibly set and reset the state of flip-flop

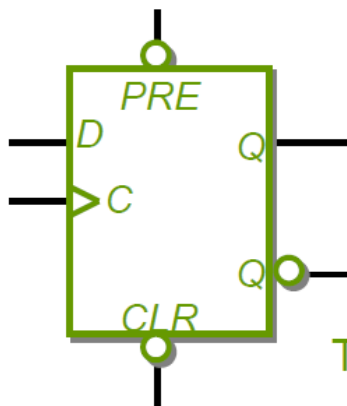
Example

The corresponding function table

Inputs				Outputs		
PRE'	CLR'	Clk	D	Next Q	Next Q'	Meaning
0	1	x	x	1	0	Preset
1	0	x	x	0	1	Clear
0	0	x	x	1*	1*	Undefined
1	1	\uparrow	x	D	D'	Set / Reset
1	1	$\downarrow, 0, 1$	x	Q	Q'	Hold

*Unpredictable behavior will result if both PRE' and CLR' set to 0 simultaneously

} The same behaviors as D flip-flops



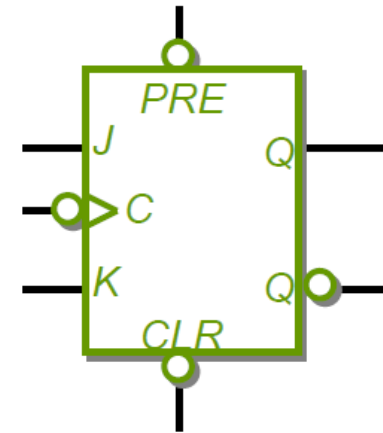
The logic symbol

Example

The function table of an negative-edge-triggered JK flip-flop with asynchronous preset and clear inputs

Inputs					Outputs		
PRE'	CLR'	Clk	J	K	Next Q	Next Q'	Meaning
L	H	x	x	x	H	L	Preset
H	L	x	x	x	L	H	Clear
L	L	x	x	x	H^*	H^*	Undefined
H	H	\downarrow	L	L	Q	Q'	Hold
H	H	\downarrow	H	L	H	L	Set
H	H	\downarrow	L	H	L	H	Reset
H	H	\downarrow	H	H	Q'	Q	Toggle
H	H	$\uparrow, 0, 1$	x	x	Q	Q'	Hold

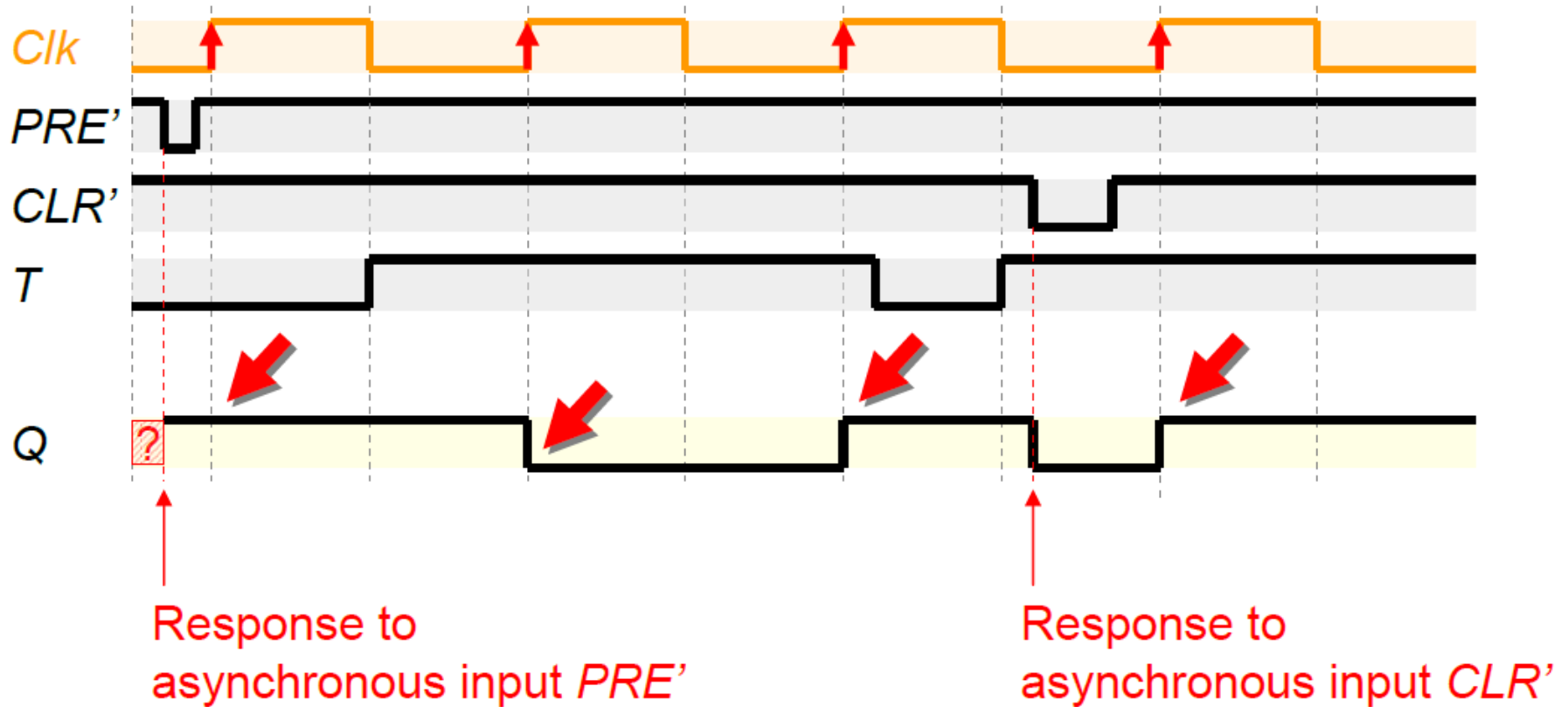
The logic symbol



The same behaviors as JK flip-flops

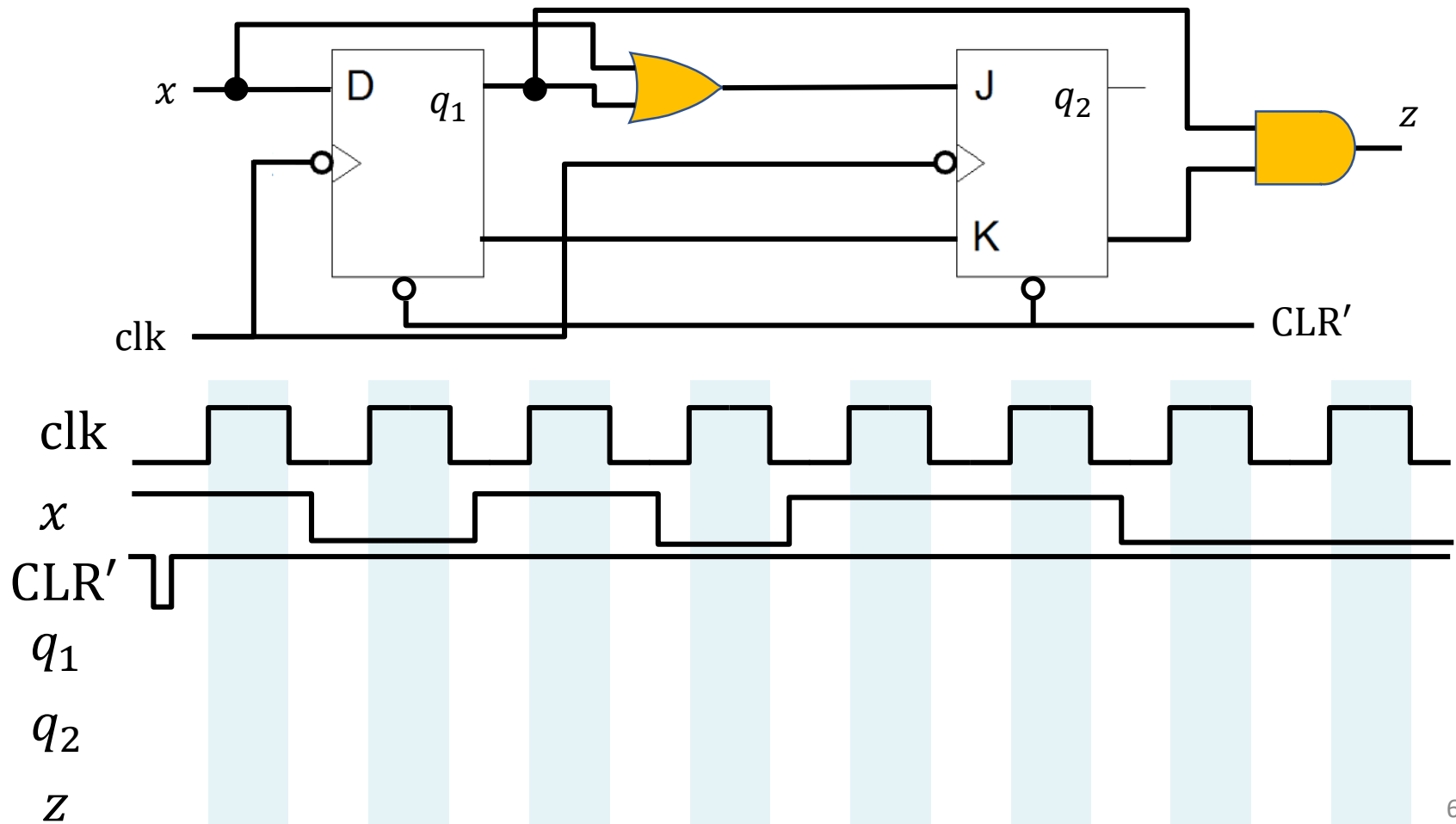
Timing Diagram

Timing Diagram of PET T FF with asynchronous preset and clear inputs (if no output delay)



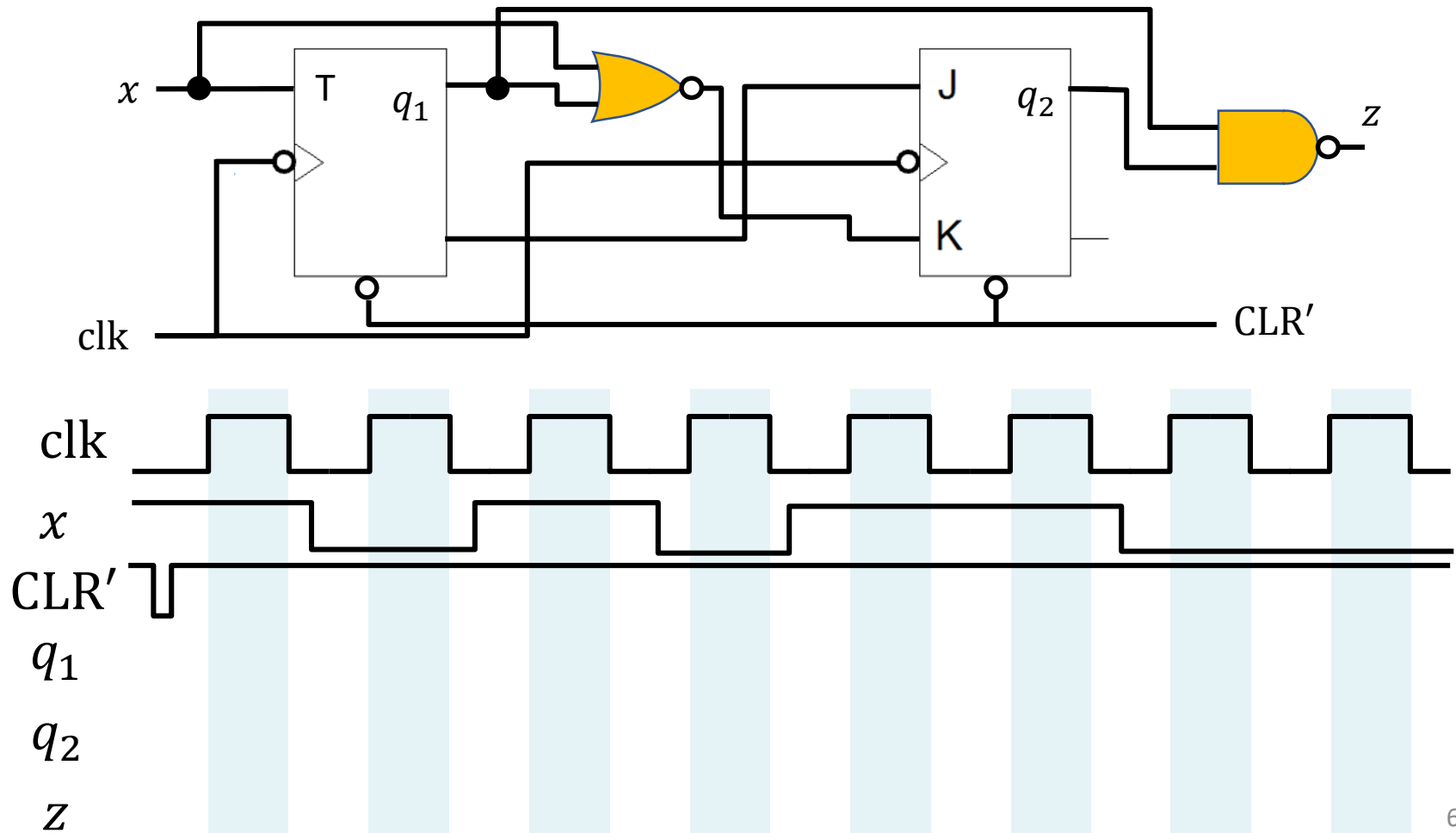
Exercise

Work out the timing diagram for the state of each flip-flop and the output (q_1 , q_2 , z). (Ignore delay)



Exercise

Work out the timing diagram for the state of each flip-flop and the output (q_1 , q_2 , z). (Ignore delay)



Summary

- Latches & Flip-Flops: Basic building block (Memory) of Sequential system
- Flip-Flop (FF): One FF can store 1-bit data
 - Types: SR, D, JK, T, Master-Slave
 - Control inputs to change, set, reset or hold the value of Q
 - Triggering: Pulse vs Edge
 - Asynchronous inputs