DIGITAL LOGIC

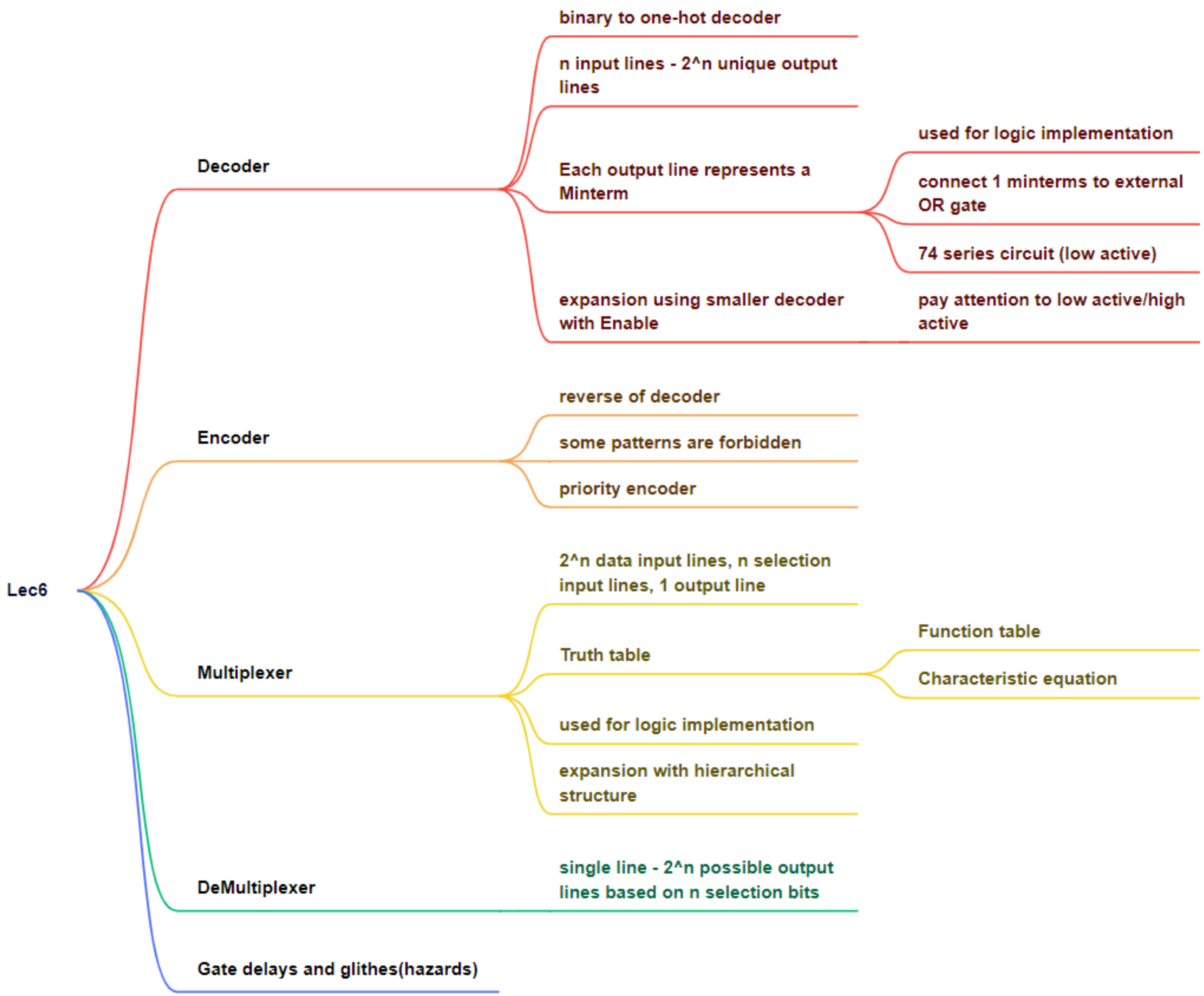
Chapter 5 part1:
Latches and Flip-flops

2023 Fall

Today's Agenda

- Recap
- Context
  - Latches
  - Flip-Flops
- Reading: Textbook, Chapter 5.1-5.4

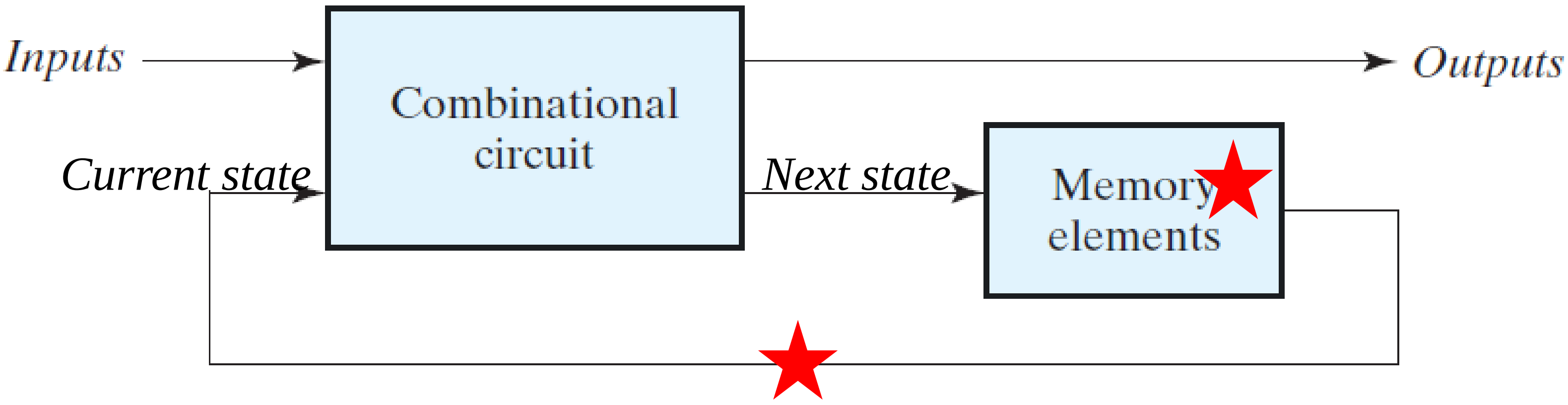
Recap



Lec6
Decoder
binary to one-hot decoder
n input lines - 2^n unique output lines
Each output line represents a Minterm
used for logic implementation
connect 1 minterms to external OR gate
74 series circuit (low active)
expansion using smaller decoder with Enable
pay attention to low active/high active
Encoder
reverse of decoder
some patterns are forbidden
priority encoder
Multiplexer
2^n data input lines, n selection input lines, 1 output line
Truth table
Function table
Characteristic equation
used for logic implementation
expansion with hierarchical structure
DeMultiplexer
single line - 2^n possible output lines based on n selection bits
Gate delays and glithes(hazards)

Sequential Circuits

- A sequential circuit consists of a combinational circuit to which storage elements are connected to form a feedback path.
  - The binary information stored in the memory elements at any given time defines the state of the sequential circuit.
  - (inputs, current state) => (outputs, next state)– The behavior is specified by a time sequence of inputs and internal states.

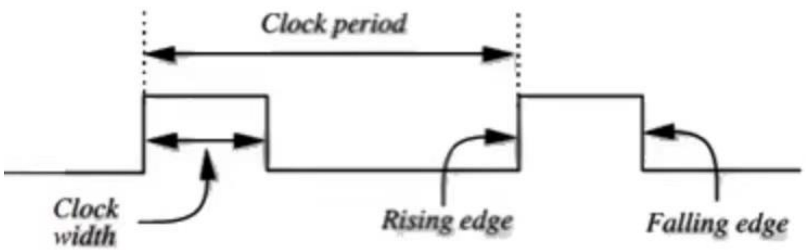


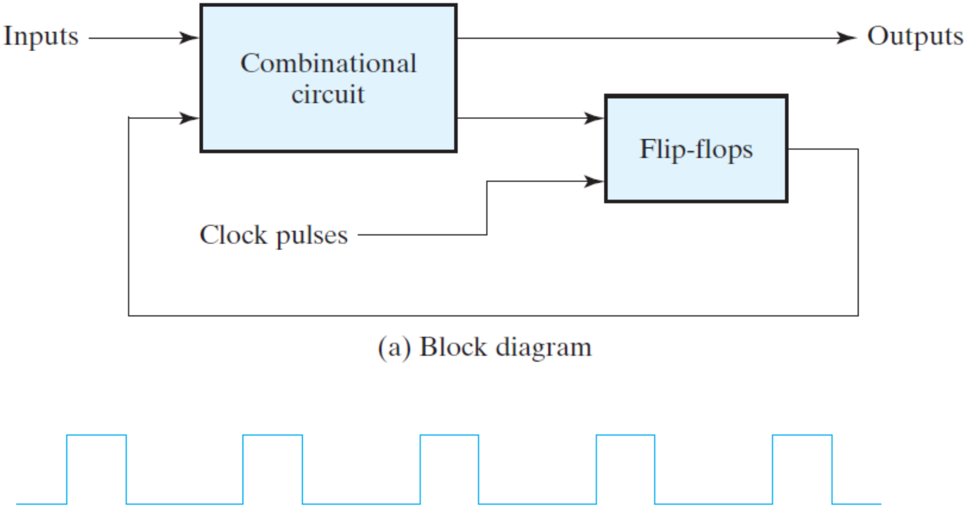
Synchronous vs. Asynchronous Sequential Circuits

- Sequential circuits are broadly classified into two main categories depending on the timing of their signals.
  - **Synchronous sequential circuit** (同步时序电路): A system whose behavior can be defined from the knowledge of its signals at discrete instants of time.
  - **Asynchronous sequential circuit** (异步时序电路): A system whose behavior depends upon input signals at any instant of time and the order in which the inputs change.
- Asynchronous circuit properties
  - Commonly used storage devices are time-delay devices, and the propagation delay of the logic gates (time-delay devices) provides the required storage.
  - Can be viewed as combinational circuit with feedback–May unstable at times

Synchronous Sequential Circuits

- Synchronization usually is achieved by a timing device: clock generator.
  - The outputs are affected only with the application of a clock pulse.
- Clock generator generates a periodic train of clock pulses distributed throughout the system to trigger the memory elements





(a) Block diagram

(b) Timing diagram of clock pulses

Storage Elements

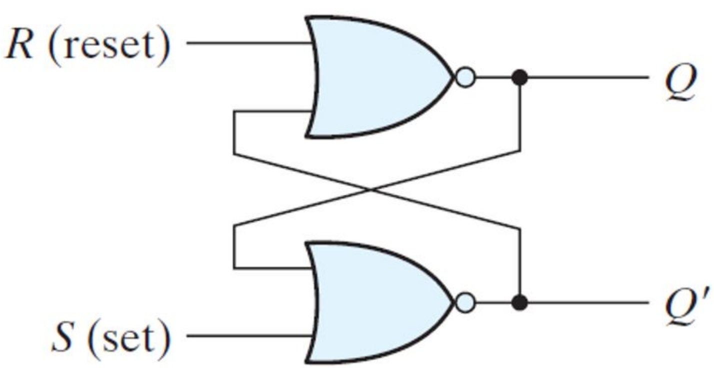
- Some definitions:
  - State: all the information about a circuit necessary to explain its future behavior
  - Latches and flip-flops: state elements that store one bit of state
    - SR Latch (SR锁存器)
    - D Latch (D锁存器)
    - D Flip-flop (DFF, D触发器)

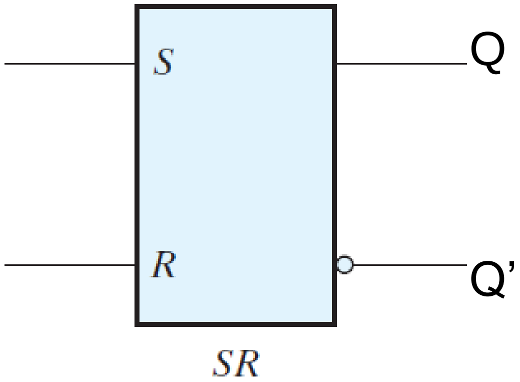
Outline

- **Latches**
- FlipFlops

SR (Set/Reset) Latch

- Latch is an asynchronous sequential circuit (state changes whenever inputs change).
- SR latch is a basic memory element which can store one bit of information
  - Consists of two cross-coupled NOR gates or two cross-coupled NAND gates
  - Two input signals: set (S)/reset(R)
  - Two output signals: Q/Q’
  - Two useful states: set state (Q=1,Q’=0)/reset state(Q=0,Q’=1)

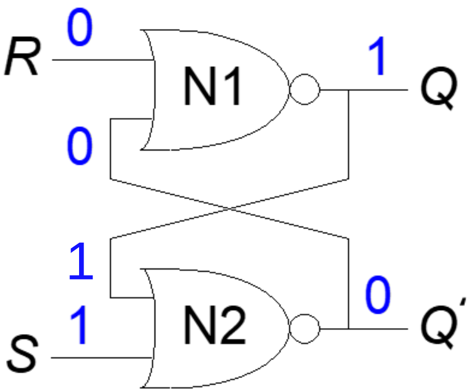




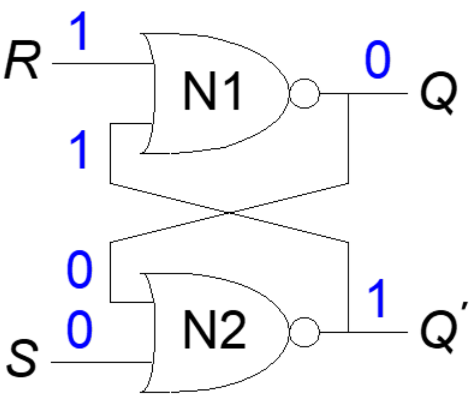
Basic SR Latch with NOR Gate

- Consider the four possible cases:
    1. S = 1, R = 0
    2. S = 0, R = 1
    3. S = 0, R = 0
    4. S = 1, R = 1

- 1. S = 1, R = 0:
    - then Q = 1 and Q’ = 0
    - Set the output



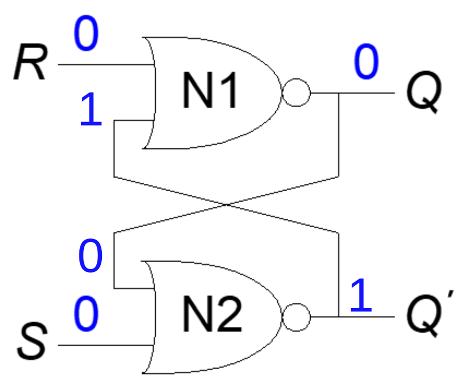
- 2. S = 0, R = 1:
    - then Q = 0 and Q’ = 1
    - Reset the output



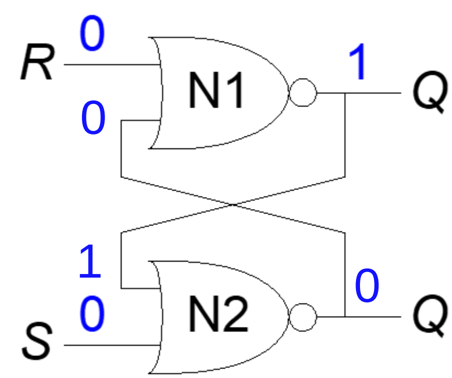
Basic SR Latch with NOR Gate

- Consider the four possible cases:
  1. S = 1, R = 0
  2. S = 0, R = 1
  3. S = 0, R = 0
  4. S = 1, R = 1

Q_{prev} = 0

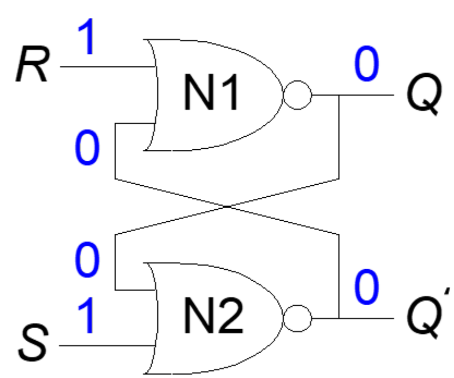


Q_{prev} = 1



- 3. S = 0, R = 0:
  - then Q = Q_{prev}
  - Memory!

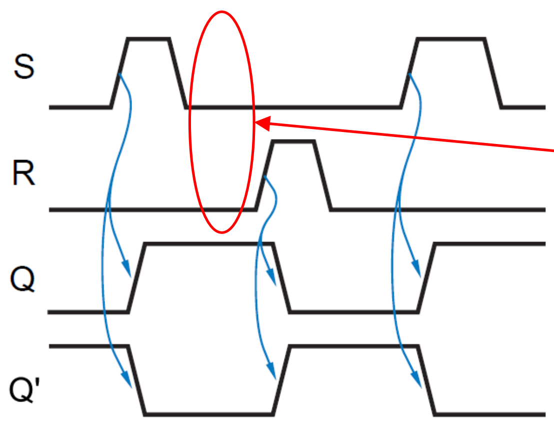
- 4. S = 1, R = 1:
  - then Q = 0 and Q’ = 0
  - Invalid State!
  - Q' ≠ NOT Q



Function Table of SR Latch

- SR Latch's functionality:
  - S is active → Set
  - R is active → Reset
  - S,R are inactive → Memory
  - S,R are active → Forbidden

S	R	Q	Q'	
0	0	last Q	last Q'	no change
0	1	0	1	reset state
1	0	1	0	set state
1	1	0	0	forbidden

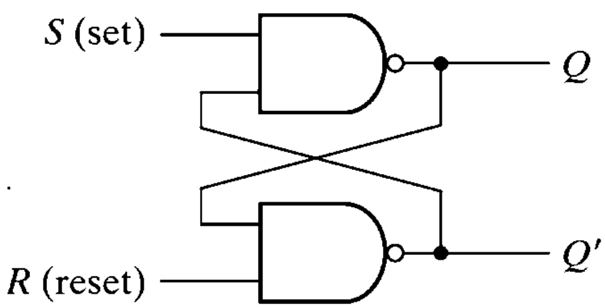


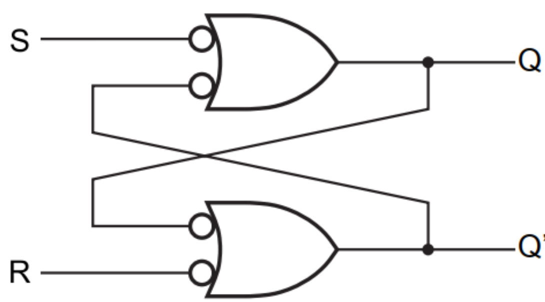
(S,R) must go back to (0,0) before any other change to avoid the occurrence of the undefined state

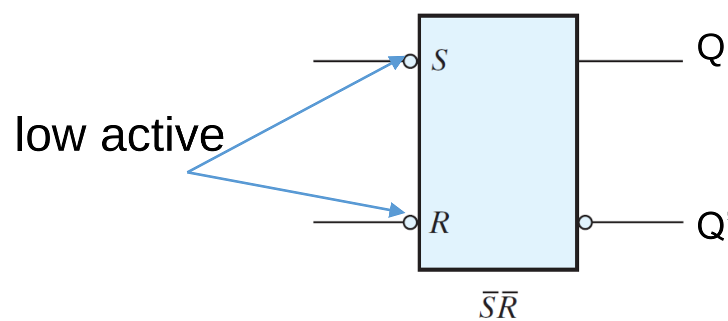
S'R' Latch with NAND Gates

- **Low Active** Set/Reset inputs
  - S is active → Set
  - R is active→ Reset
  - S,R are inactive → Memory
  - S,R are active → Forbidden

S	R	Q	Q'	
0	0	1	1	forbidden
0	1	1	0	set state
1	0	0	1	reset state
1	1	last Q	last Q'	no change





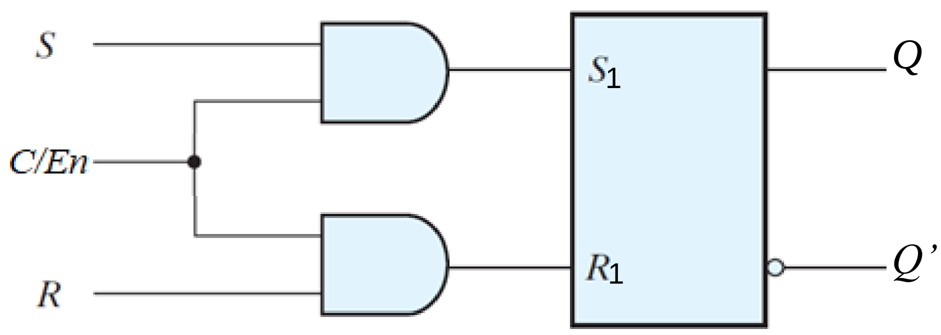


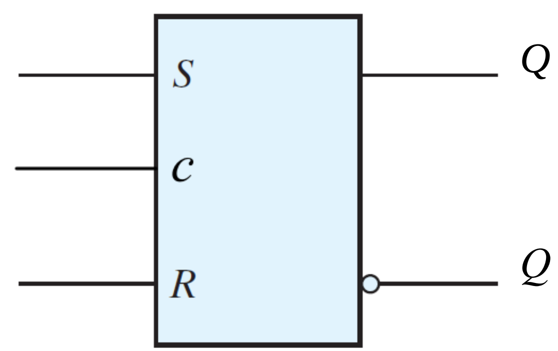
graphic symbol or S'R' Latch

Clocked SR Latch

- Use Clock (or En) to enable/disable the SR latch
  - C=0, no change (disabled)
  - C=1, operates as normal SR latch (enabled)
- Level sensitive SR Latch
  - However, undefined state when
    C = S= R = 1

C	S	R	Q	Q’	
0	X	X	last Q	last Q’	no change
1	0	0	last Q	last Q’	no change
1	0	1	0	1	reset state
1	1	0	1	0	set state
1	1	1	0	0	forbidden

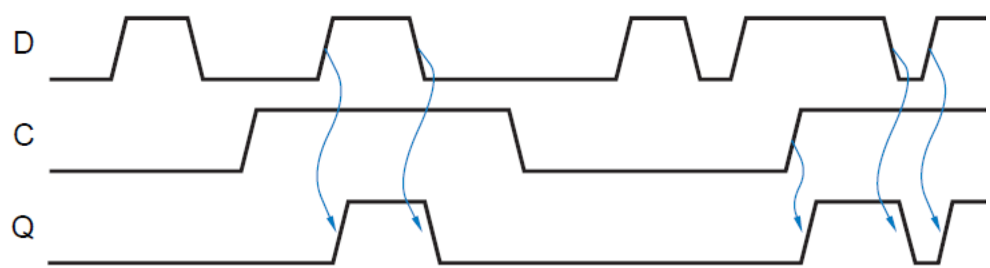




D Latch

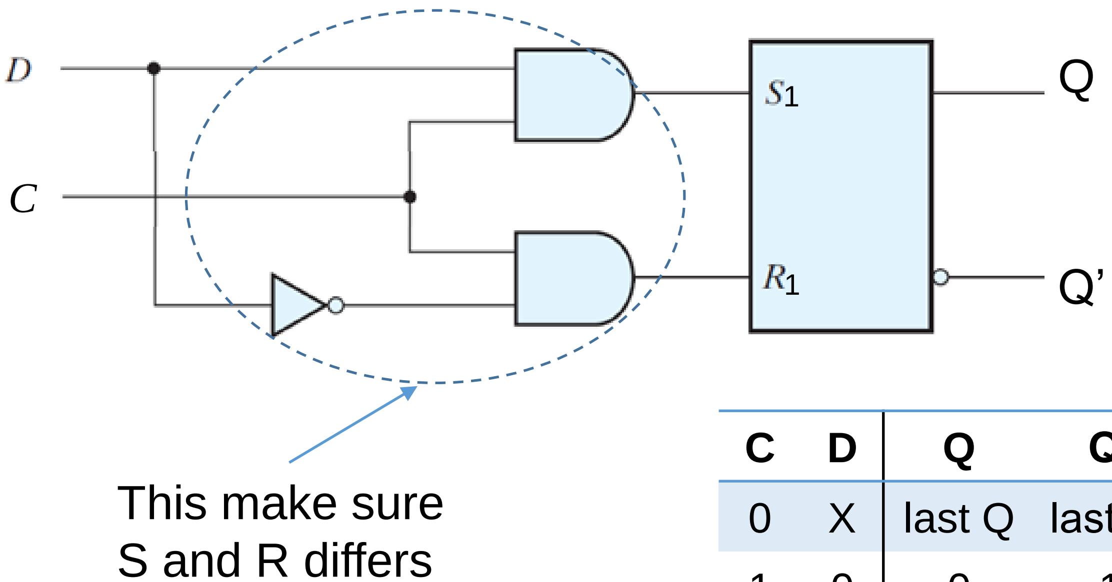
- Two inputs: Clk(En), D
  - Clk(En): controls when the output changes
  - D (the data input): controls what the output changes to
- Level sensitive D Latch
  - When CLK = 1,
    - D passes through to Q (transparent)
  - When CLK = 0,
    - Q holds its previous value (opaque)
  - Can avoid invalid case when
    - Q’ ≠ NOT Q

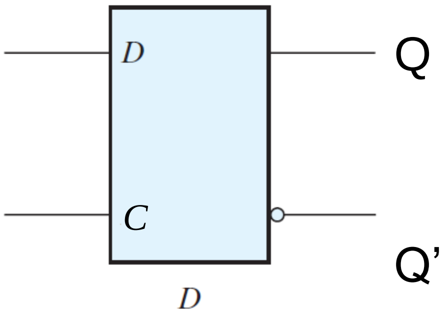
C	D	Q	Q’	
0	X	last Q	last Q’	no change
1	0	0	1	Q follows D
1	1	1	0	Q follows D



D Latch Structure

- D latch is also called Transparent Latch
  - Constructed from a gated SR latch by connecting the D input to S input and D’ to R.
  - Avoiding undesirable condition (S=R=1)



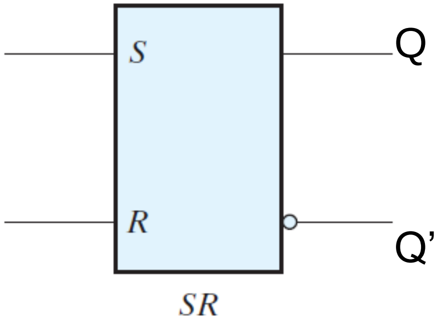


D Latch symbols

C	D	Q	Q’	
0	X	last Q	last Q’	no change
1	0	0	1	Q follows D
1	1	1	0	Q follows D

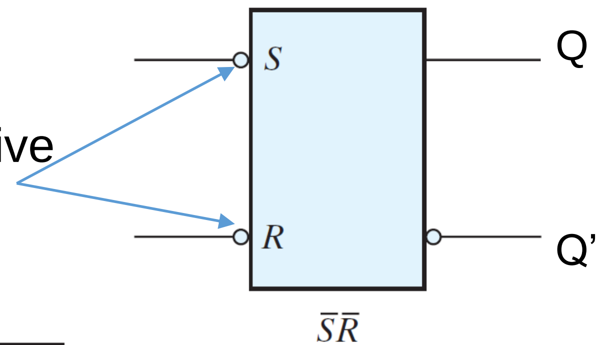
Latch Summary

- SR Latch with NOR gate (SR)
  - Set: Make the output 1
    (S = 1, R = 0, Q = 1)
  - Reset: Make the output 0
    (S = 0, R = 1, Q = 0)
  - Memory: S=R=0

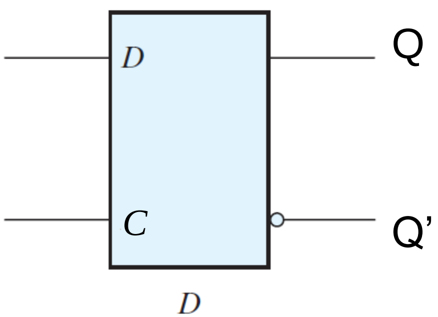


- S'R' Latch with NAND gate (S'R')
  - Set: Make the output 1
    (S = 0, R = 1, Q = 1)
  - Reset: Make the output 0
    (S = 1, R = 0, Q = 0)
  - Memory: S=R=1

low active

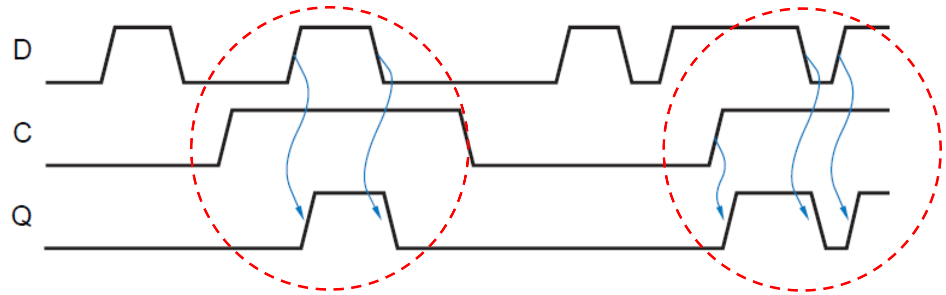


- D Latch (D)
  - Make D passes through to Q



Latches Pros and Cons

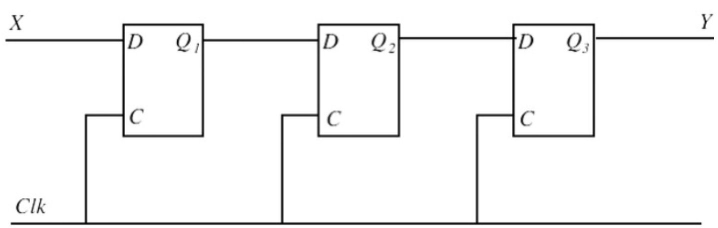
- A latch is enabled whenever C=1. (level-sensitive)
  - At any point during C=1, any input changes will propagate to the output (with some small delay)



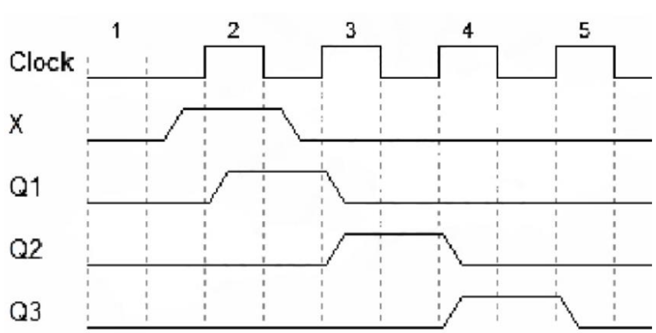
- Pros
  - Useful for level-sensitive applications
  - Latches are faster because they have fewer gates and less delay.
  - Latches are simpler which means they require less hardware to implement and consume less power.

Latches Pros and Cons

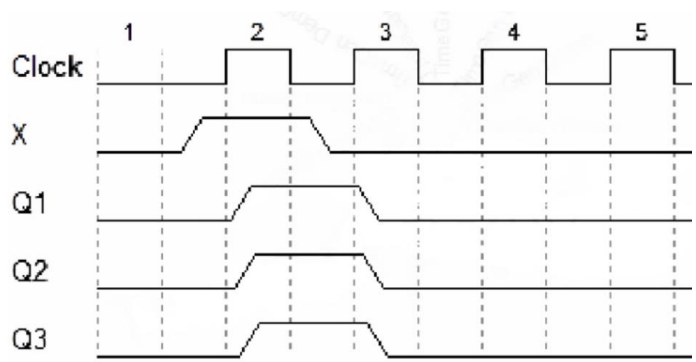
- A latch is enabled whenever C=1. (level-sensitive)
  - At any point during C=1, any input changes will propagate to the output (with some small delay)
- Cons
  - Difficult to retain value
  - Causing erroneous shifting



Unable to shift Q_2 and Q_3 with clock cycles delay



expected



real

Outline

- Latches
- **FlipFlops**

Flip-Flops (FF)

- Latch or Flip-Flop(FF) state changes by a control input.
  - The event is called trigger.
- A Flip-Flop is a more advanced storage element, typically constructed using two D latches or more complex circuitry.
- Example: D Flip Flop
  - Inputs: CLK, D
  - For a Positive-edge triggered D Flip Flop:
  - Samples D on 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 rising edge of CLK

Trigger

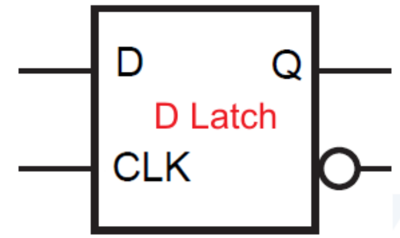
- Trigger
  - The state of a latch or flip-flop is switched by a change of the control input
- Level sensitive (level-triggered) (latches)
  - The state transition starts as soon as the clock is during logic 1 (positive level-sensitive) or logic 0 (negative level-sensitive) level
- Edge-triggered (flip-flops)
  - The state transition starts only at positive (positive edge-triggered) or negative edge (negative edge-triggered) of the clock signal

(a) Response to positive level

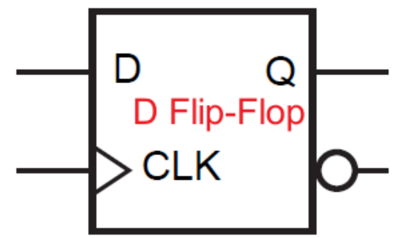
(b) Positive-edge response

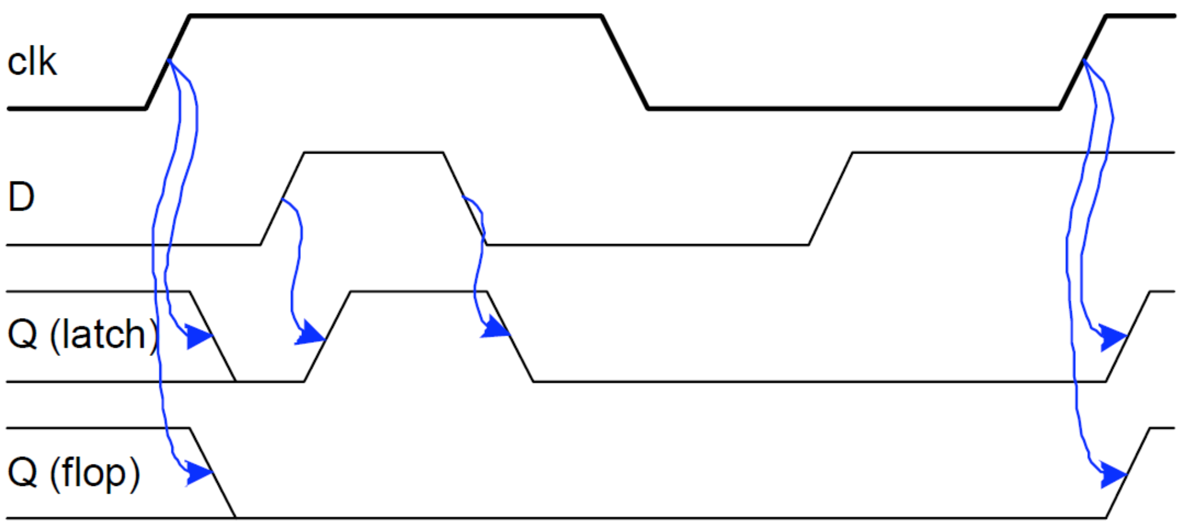
Level-Sensitive vs. Edge-Triggered

(a) Response to positive level



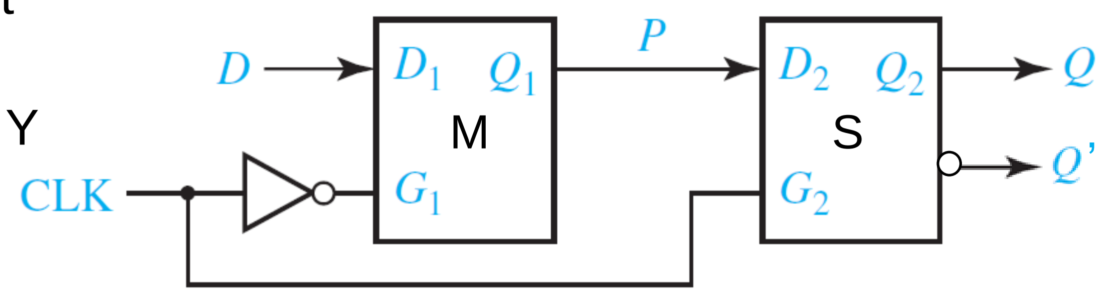
(b) Positive-edge response





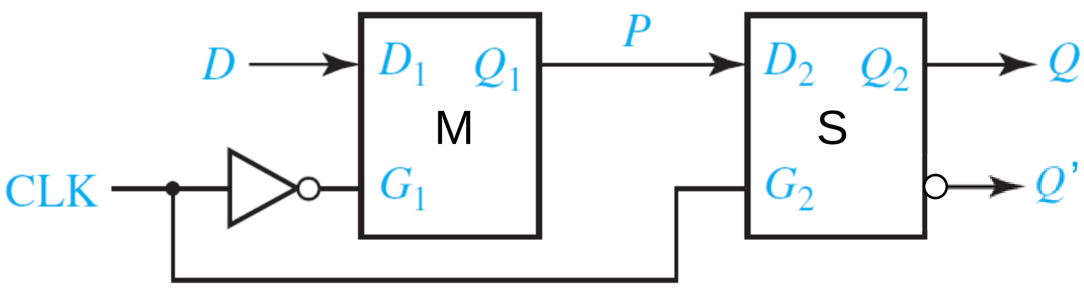
DFF Structure

- A D flip-flop is formed by two separate latches
  - A master D latch (negative level sensitive)
  - A slave D latch (positive level sensitive)
- Positive-edge-triggered D flip-flop
  - When CLK = 0
    - master is transparent
    - slave is opaque
    - D passes through to Y
  - When CLK = 1
    - master is opaque
    - slave is transparent
    - Y passes through to Q
  - Thus, on the rising edge of the clock (CLK rises from 0→1)
    - D passes through to Q

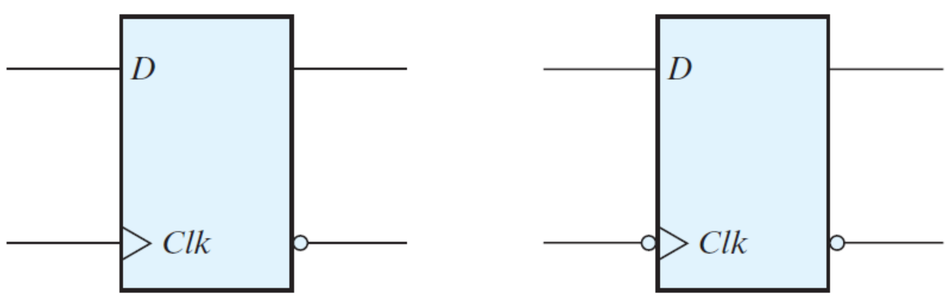


Function Table of DFF

- Function table



C	D	Q	Q’	
0	X	last Q	last Q’	no change
1	X	last Q	last Q’	no change
↑	0	0	1	Q follows D
↑	1	1	0	Q follows D



(a) Positive-edge (a) Negative-edge

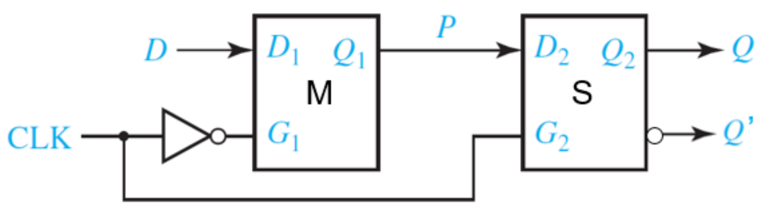
D Flip-Flop Symbols

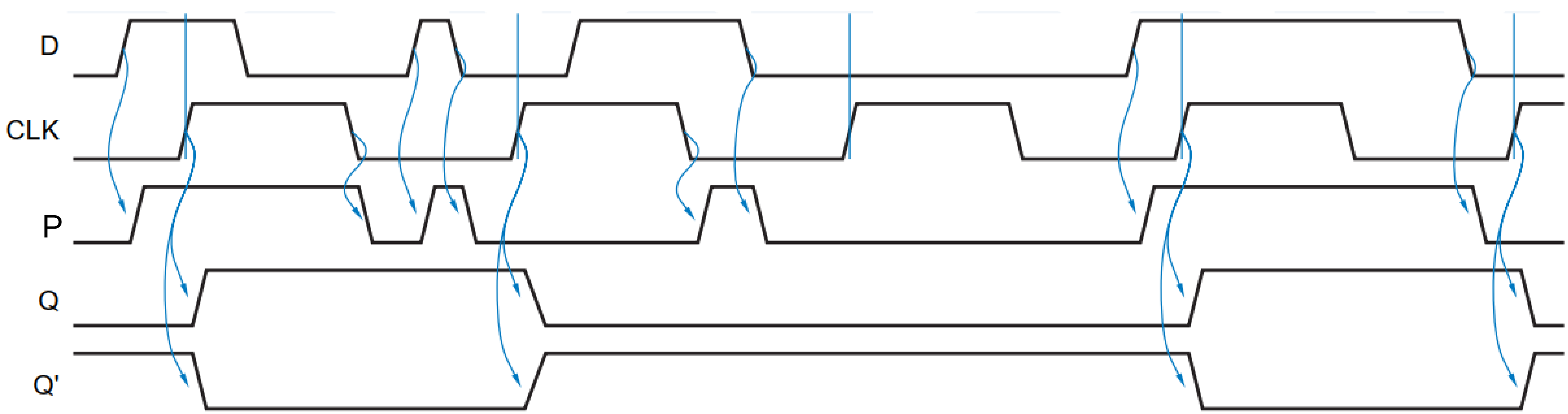
Question: How to design Negative-edge-triggered flip-flop (refer to textbook)

Timing Graph of DFF

- Positive-edge-triggered DFF
  - Q stays stable between two rising edges of the clock signal

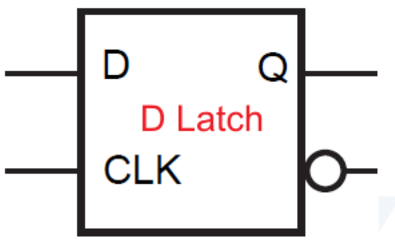
C	D	Q	Q’	
0	X	last Q	last Q’	no change
1	X	last Q	last Q’	no change
↑	0	0	1	Q follows D
↑	1	1	0	Q follows D

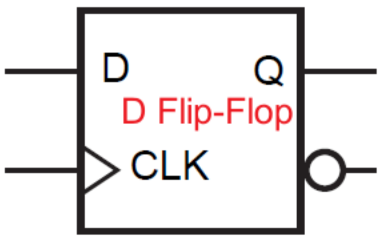


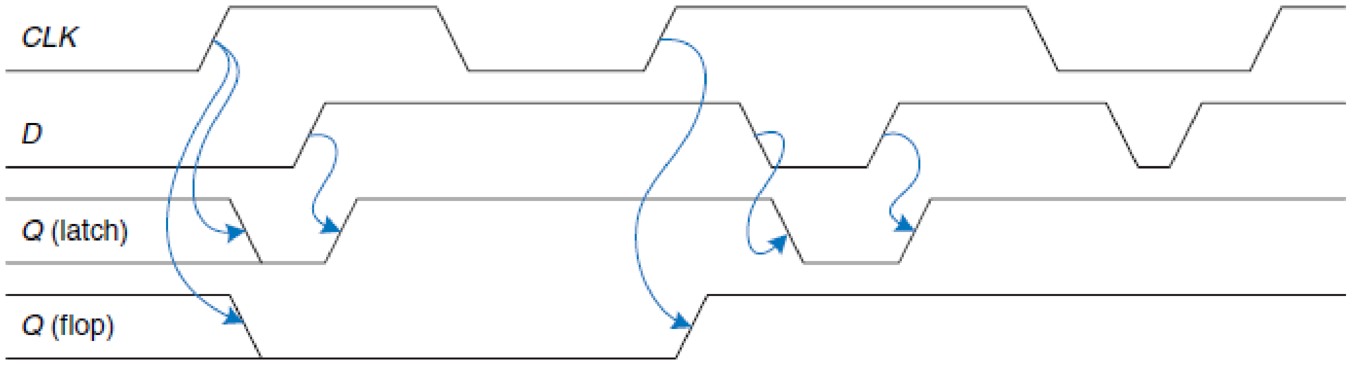


Flip-flop and Latch Comparison

- Apply the D and CLK inputs to a D latch and a D flip-flop, determine the output, Q, of each device.

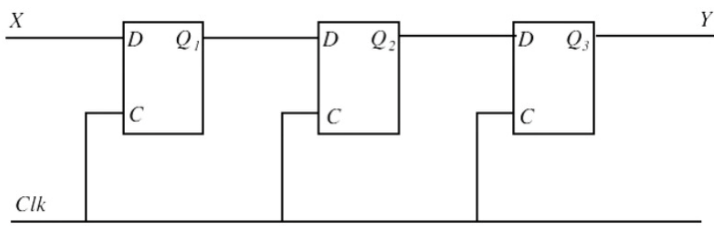




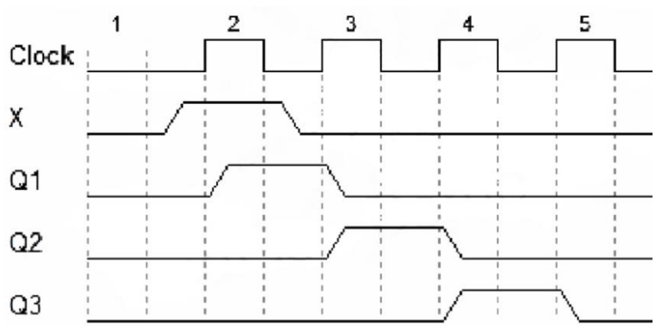


Shifting

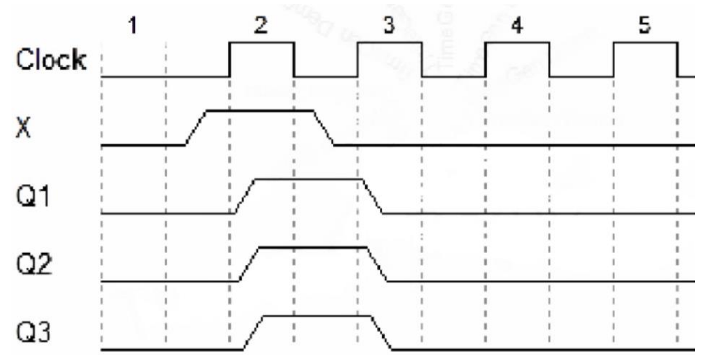
- Remember we wanted to shift input with clock cycles delay, but latches didn't work out



Unable to shift Q_2 and Q_3 with clock cycles delay

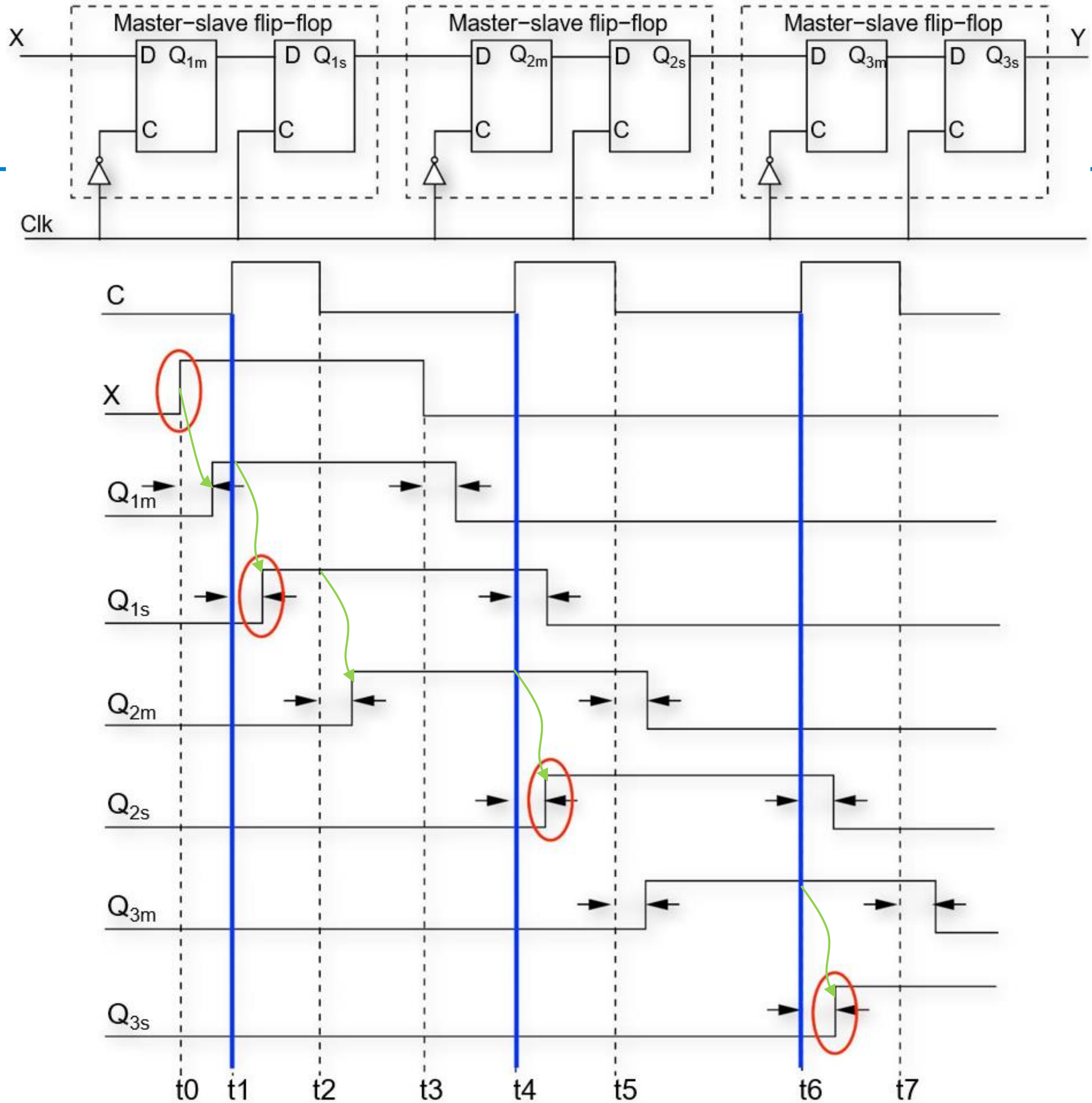


expected

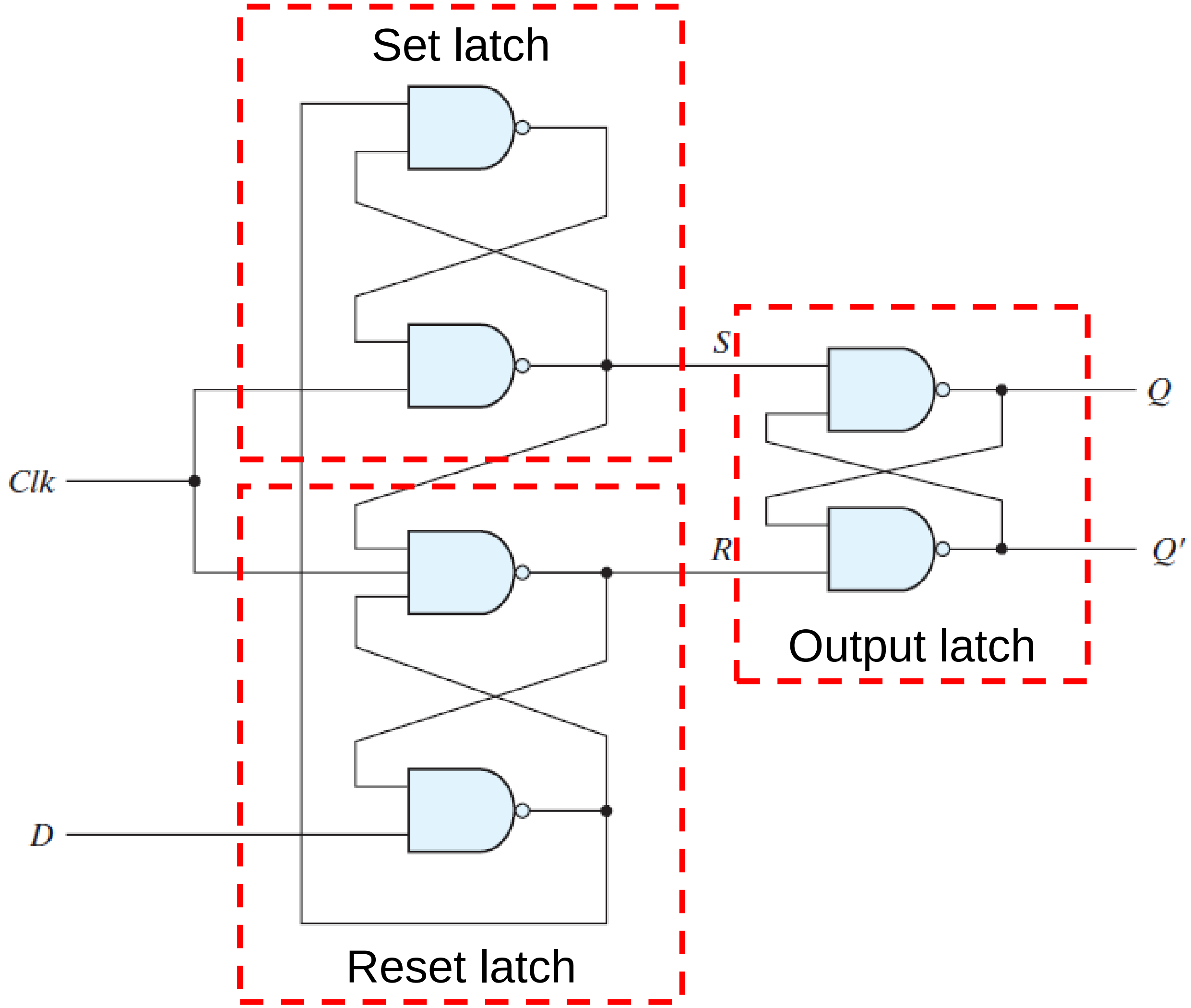


real

- Now we can use FlipFlops


Master–slave flip–flop
Master–slave flip–flop
Master–slave flip–flop
X
Y
D
Q1m
C
D
Q1s
C
D
Q2m
C
D
Q2s
C
D
Q3m
C
D
Q3s
C
Clk
C
X
Q1m
Q1s
Q2m
Q2s
Q3m
Q3s
t0
t1
t2
t3
t4
t5
t6
t7

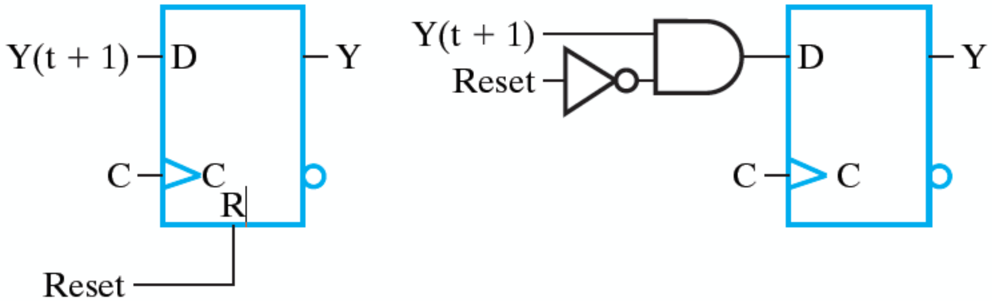
DFF with three SR Latches



Positive-edge-triggered D flip-flop

DFF with Reset

- The state of FFs are unknown when power is on. A direct input can force the FFs to a known state before the system starts.
  - E.g. when Reset = 1, FF's output is forced to 0
- Synchronous vs. asynchronous resettable Flip Flop
  - Asynchronous: resets immediately when Reset = 1
  - Synchronous: resets at the clock edge only



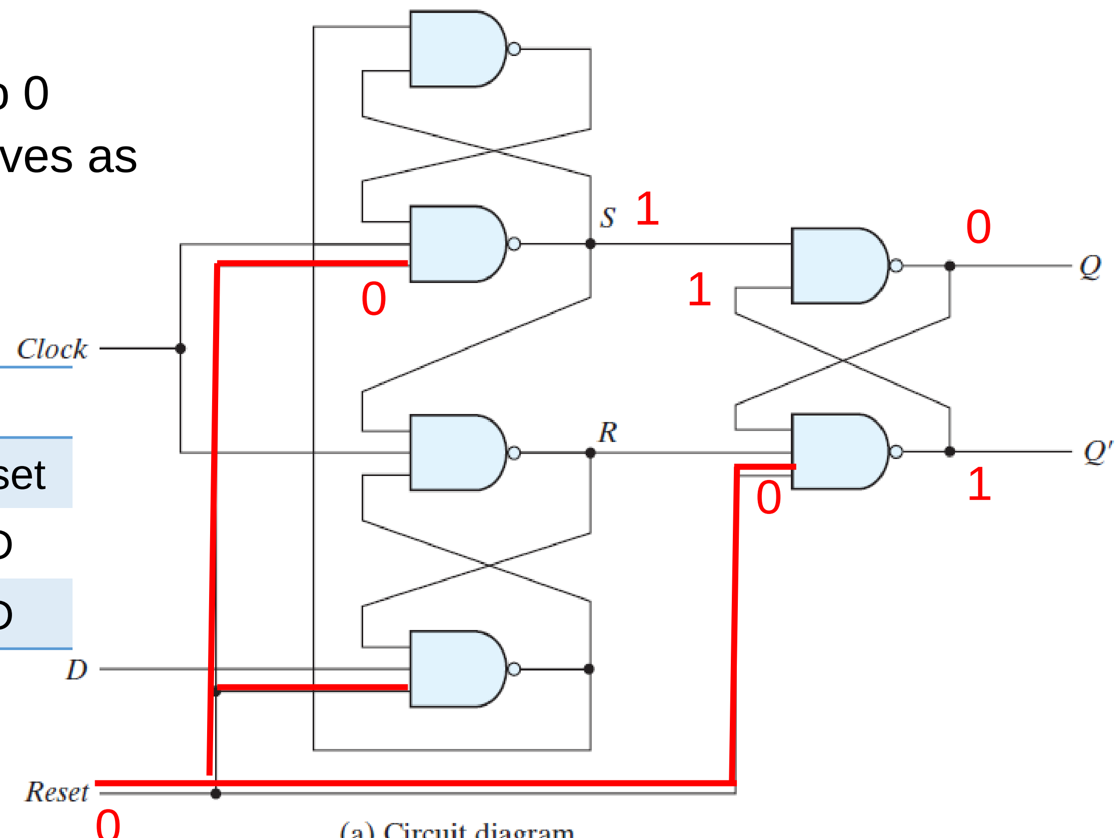
(a) Asynchronous Reset (b) Synchronous Reset

DFF with Asynchronous Reset

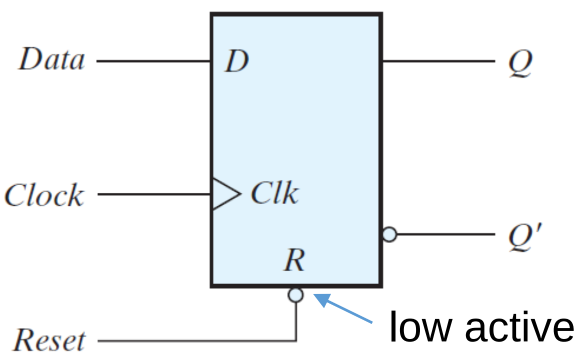
- Active low reset
  - Reset = 0: Q is forced to 0
  - Reset = 1: flip-flop behaves as ordinary D flip-flop

Function table

R	C	D	Q	Q’	
0	X	X	0	1	power on reset
1	↑	0	0	1	Q follows D
1	↑	1	1	0	Q follows D



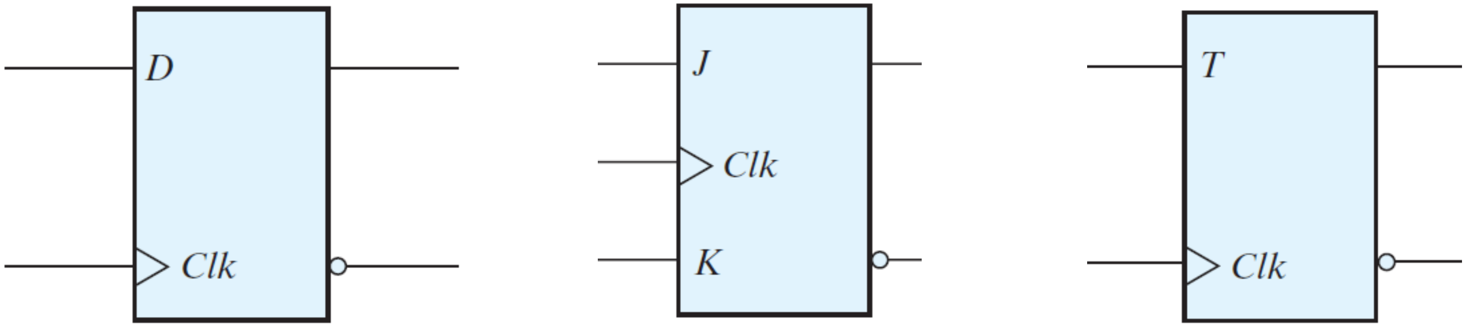
(a) Circuit diagram



(b) Graphic symbol

Types of Flip-Flops

- Flip-flops are more suitable for synchronous sequential circuits and are often used as the basic memory elements, since they only respond to a transition on a clock input.
- Other types of flip-flops can be constructed by using the DFF and external logic.
- Major FFs
    - D(data), JK, T (toggle) FFs
    - Assume only positive-edge-triggered FFs, block diagram are as follow



Characteristic Table

- **Characteristic table:** describe the behavior of a flip-flop based on its input and current state Q(t) just before the rising edge of the clock, and the resulting next state Q(t+1) after the clock transition.

Function table

Clk	D	Q	Q'	
0	X	last Q	last Q'	no change
1	X	last Q	last Q'	no change
	0	0	1	Q follows D
	1	1	0	Q follows D

Characteristic table of DFF

Characteristic table
***D* Flip-Flop**

D	**Q(*t* + 1)**	
0	0	Reset
1	1	Set

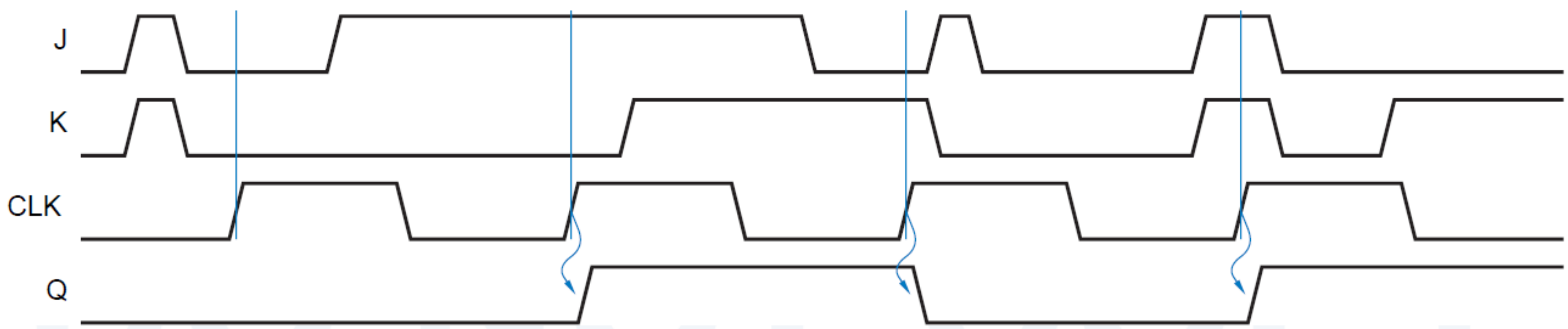
- **Characteristic equation:** derived from the characteristic table using the map method
  - Characteristic equation of DFF: $\boxed{Q_{t+1} = D}$

J-K Flip-Flop

- Positive edge-triggered JKFF
  - **At rising edge** of clock
    - J = K = 0, Q is unchanged
    - J = K = 1, Q toggles
    - J = 1, K = 0, Q is set to 1
    - J = 0, K = 1, Q is reset to 0
  - At level of clock
    - Q is unchanged

Function Table

Clk	**J**	**K**	**Q**	**Q'**
0	x	x	last Q	last Q'
1	x	x	last Q	last Q'
↑	0	0	last Q	last Q'
↑	0	1	0	1
↑	1	0	1	0
↑	1	1	last Q'	last Q



J-K Flip-Flop

Characteristic table

JK Flip-Flop

J	K	Q(t + 1)	
0	0	$Q(t)$	No change
0	1	0	Reset
1	0	1	Set
1	1	$Q'(t)$	Complement

Derive the Truth table

J	K	Q(t)	Q(t+1)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Characteristic equation of JKFF

$$\boxed{Q(t+1) = JQ(t)' + K'Q(t)}$$

- **Algebraically**:

Q(t+1)
= J'K'Q(t) + J'K•0 + JK'•1 + JKQ(t)'
= J'K'Q(t) + JK' + JKQ(t)'
= J'K'Q(t) + JK'Q(t) + JK'Q(t)' + JKQ(t)'
= JQ(t)' + K'Q(t)

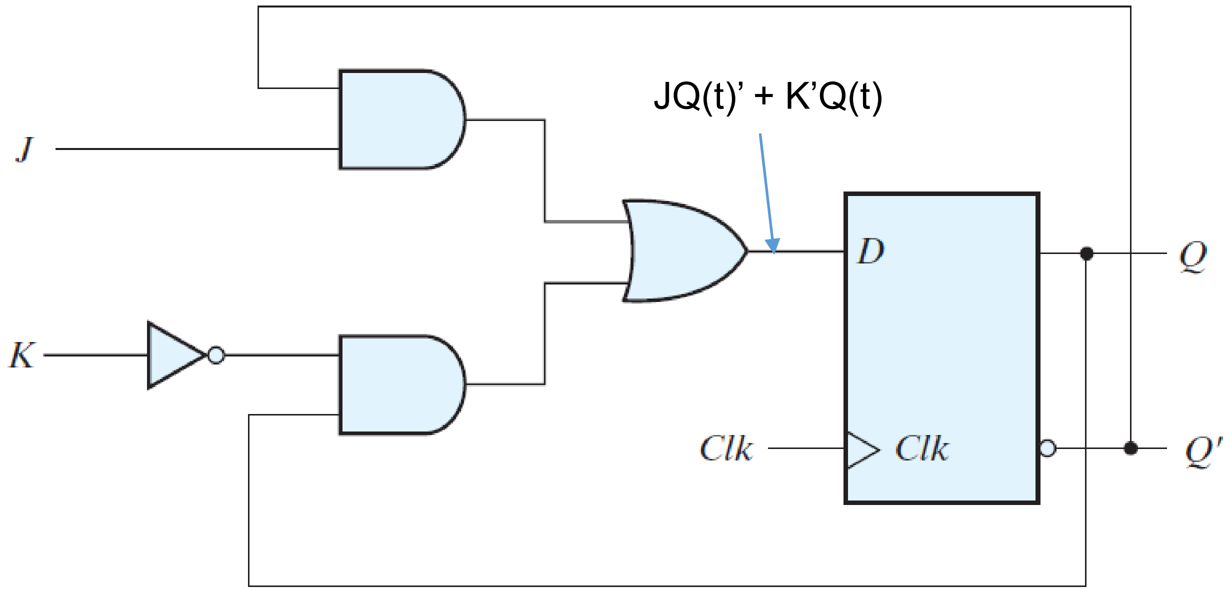
- **K-Map**

J \ kQ(t)	00	01	11	10
0	m_0 0	m_1 1	m_3 0	m_2 0
1	m_4 1	m_5 1	m_7 0	m_6 1

k (columns 11, 10); J (row 1); Q(t) (columns 01, 11)

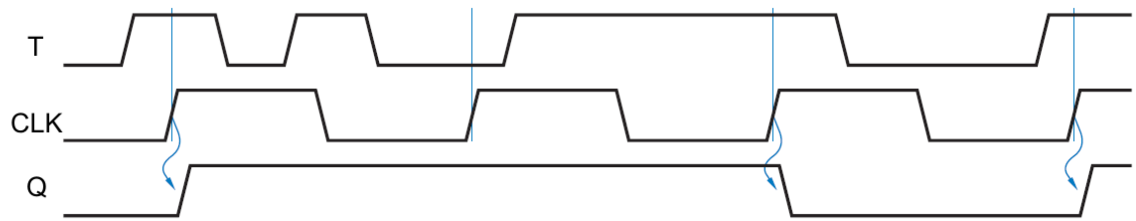
J-K Flip-Flop

- Implement JKFF using DFF
  - Because in DFF, $Q_{t+1} = D$
  - Thus, according to JKFF's characteristic equation, Q(t+1) = JQ(t)' + K'Q(t) , we can derive D = JQ(t)' + K'Q(t)



T Flip-Flop

- T: Toggle



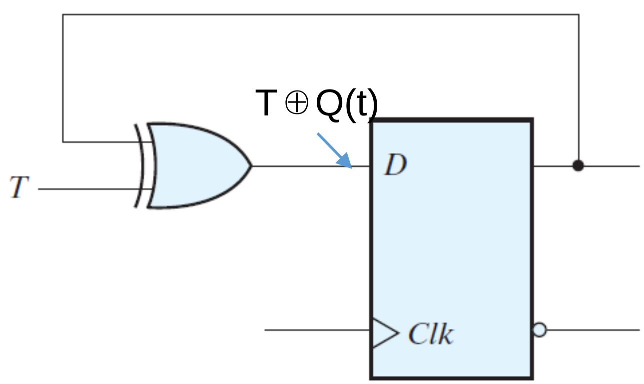
T Flip-Flop

T	$Q(t+1)$	
0	$Q(t)$	No change
1	$Q'(t)$	Complement

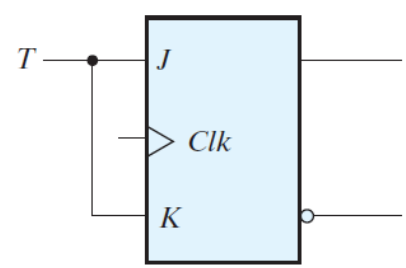
T	Q(t)	Q(t+1)
0	0	0
0	1	1
1	0	1
1	1	0

Characteristic equation of TFF

$$\boxed{\begin{aligned} Q(t+1) &= T'Q(t) + TQ(t)' \\ &= T \oplus Q(t) \end{aligned}}$$



From D flip-flop



From JK flip-flop