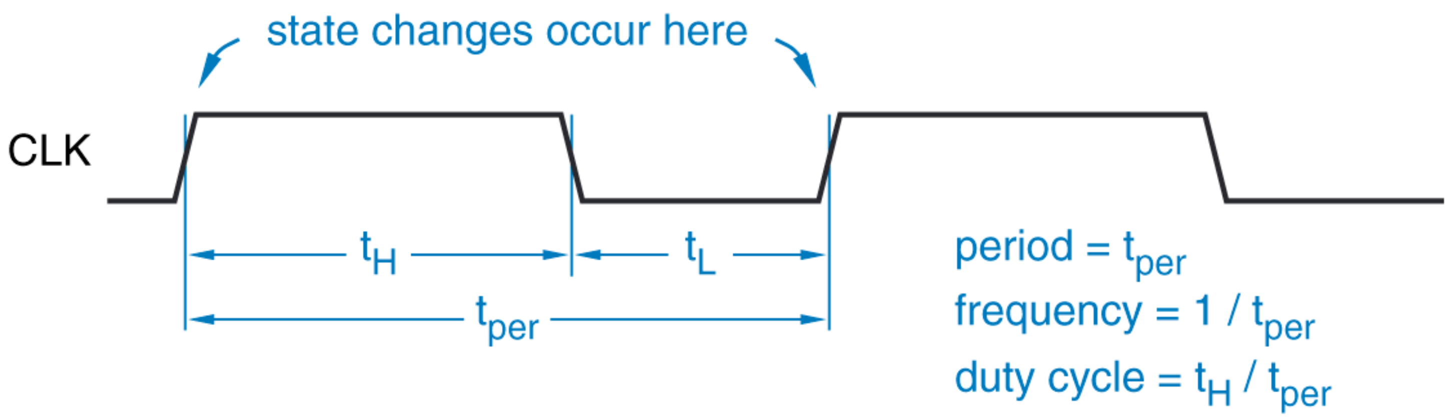
Sequential Circuits: Latches and Flip-Flops

Sequential circuits

- Output depends on current input ***and*** past sequence of input(s)
- How can we tell if the input is current or from the past?
- A clock pulse can cause state changes in sequential circuits.



- What about an active low clock, i.e. CLK_L?
- Sequential circuits can remember past inputs
  - "Memory" is needed to remember the past

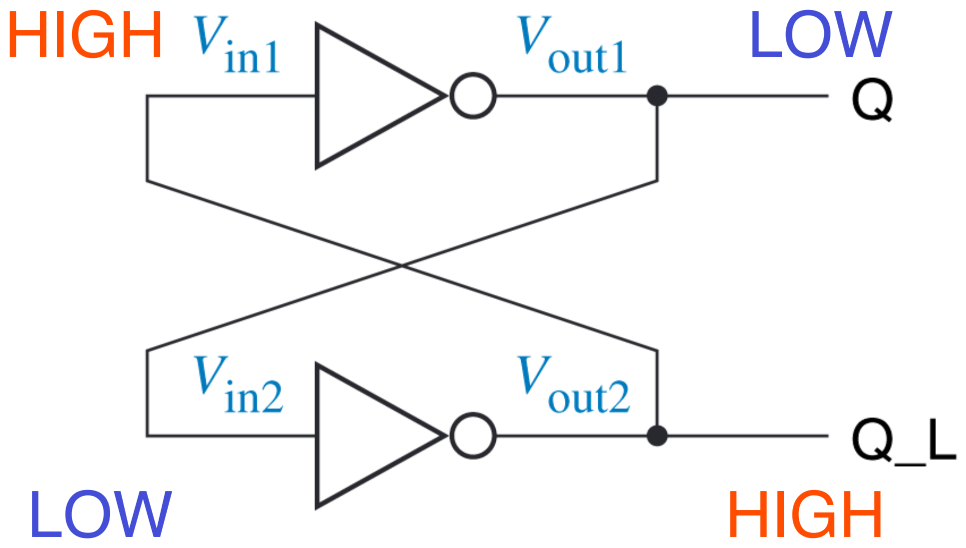
Practical discrete designs

- Feedback sequential circuits
  - Use ordinary gates and feedback loops to obtain memory in a logic circuit
  - Thus creating sequential-circuit building blocks, e.g. latches and flip-flops (many kinds)

- Clocked synchronous state machines
  - Use the building blocks from above, esp. edge-triggered D flip-flops, to create sequential circuits
  - Whose inputs are examined and whose outputs change according to a controlling clock signal (the "triggering")

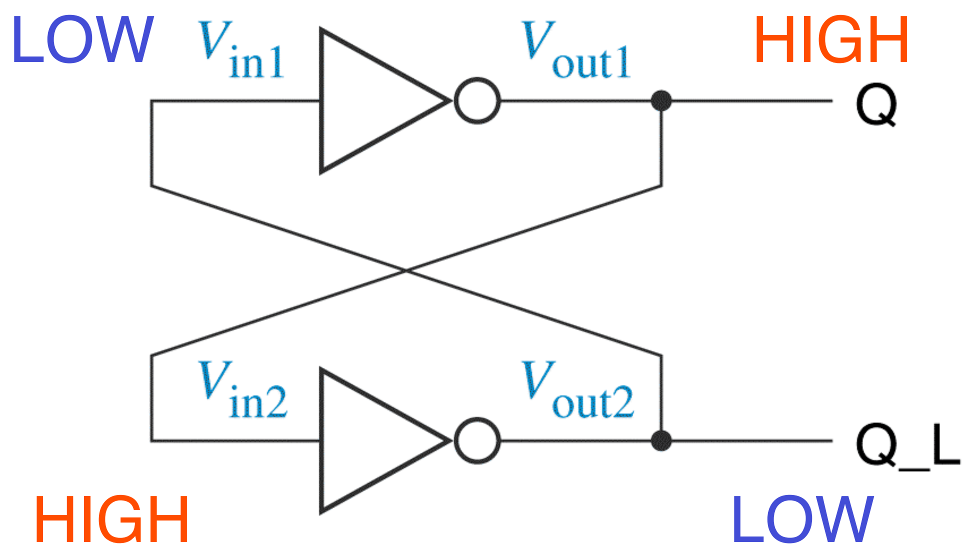
Bistable element

- Simplest sequential circuit – 2 inverters forming a feedback loop
- Has two states
  - 1 state variable, Q, represents 2 states, i.e. **Q = 0** and Q = 1



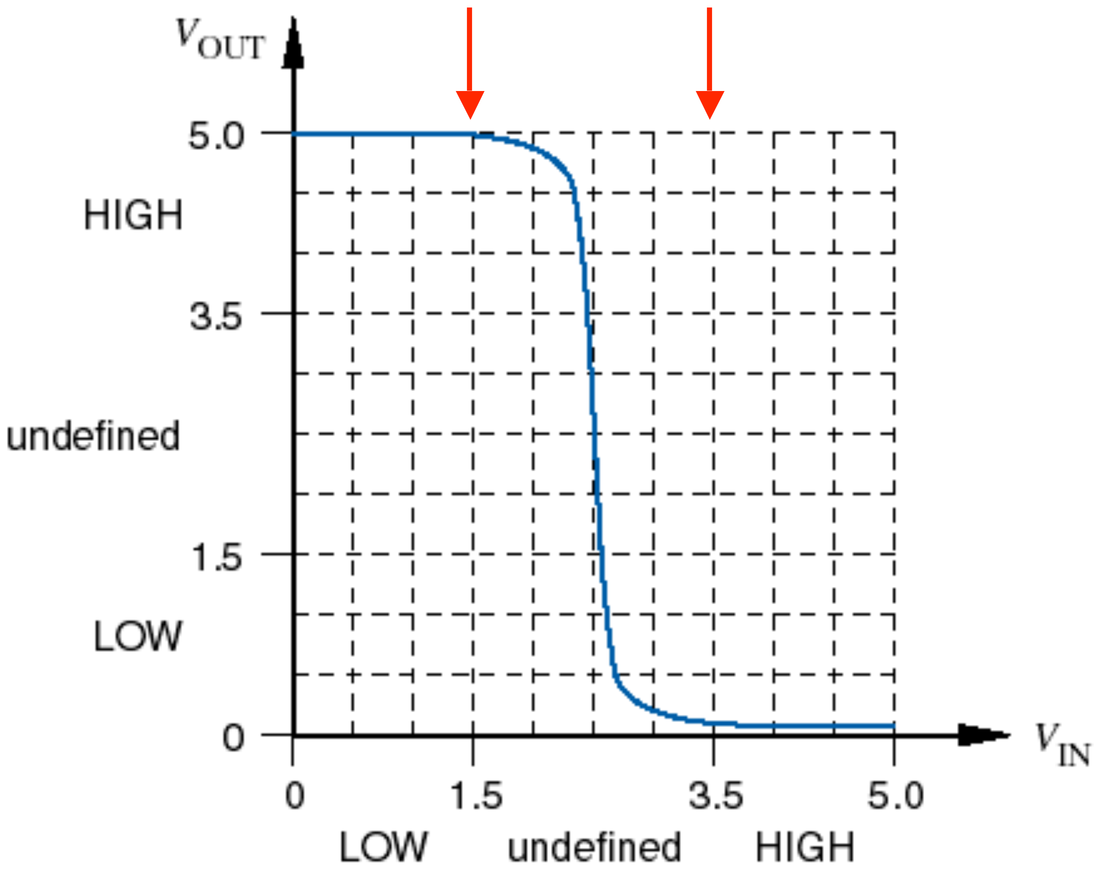
Bistable element

- Simplest sequential circuit – 2 inverters forming a feedback loop
- Has two states
  - 1 state variable, Q, represents 2 states, i.e. Q = 0 and **Q = 1**



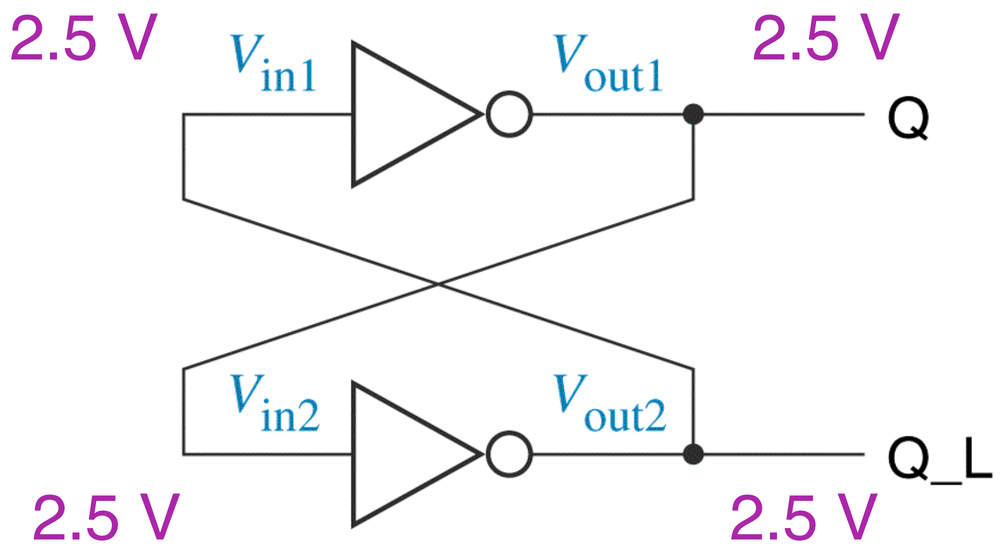
Analog analysis

- Assume pure CMOS thresholds, 5V rail
- Theoretical threshold center is 2.5 V



Analog analysis

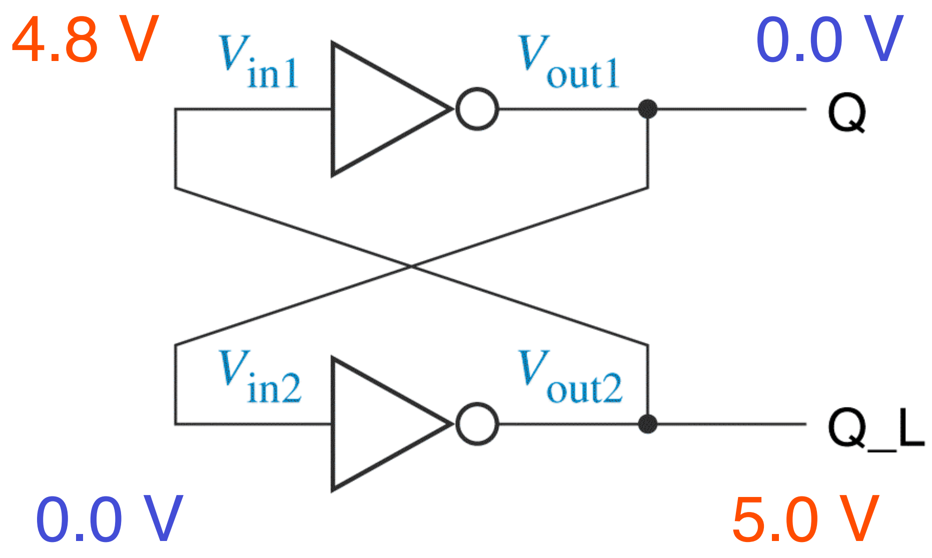
- Assume pure CMOS thresholds, 5V rail
- Theoretical threshold center is 2.5 V

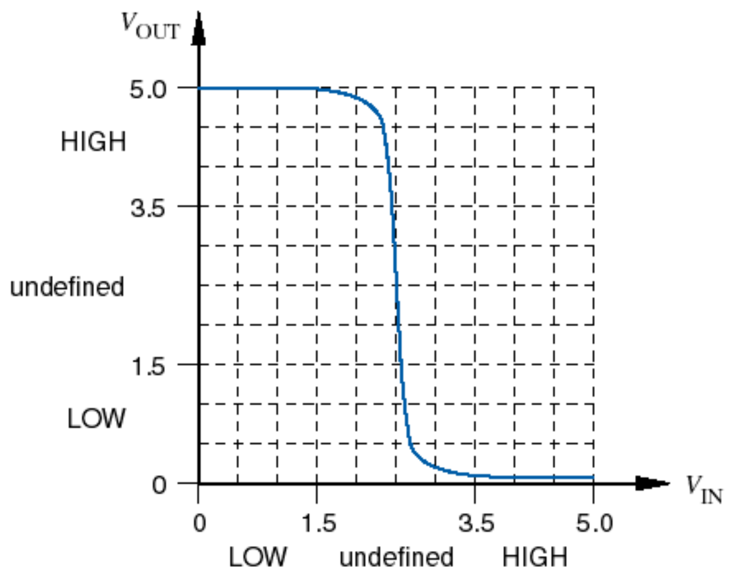


- Q and Q_L are not valid at 2.5 V.
- But in theory they can stay as such indefinitely – metastable!

Analog analysis

- Assume pure CMOS thresholds, 5V rail
- Theoretical threshold center is 2.5 V

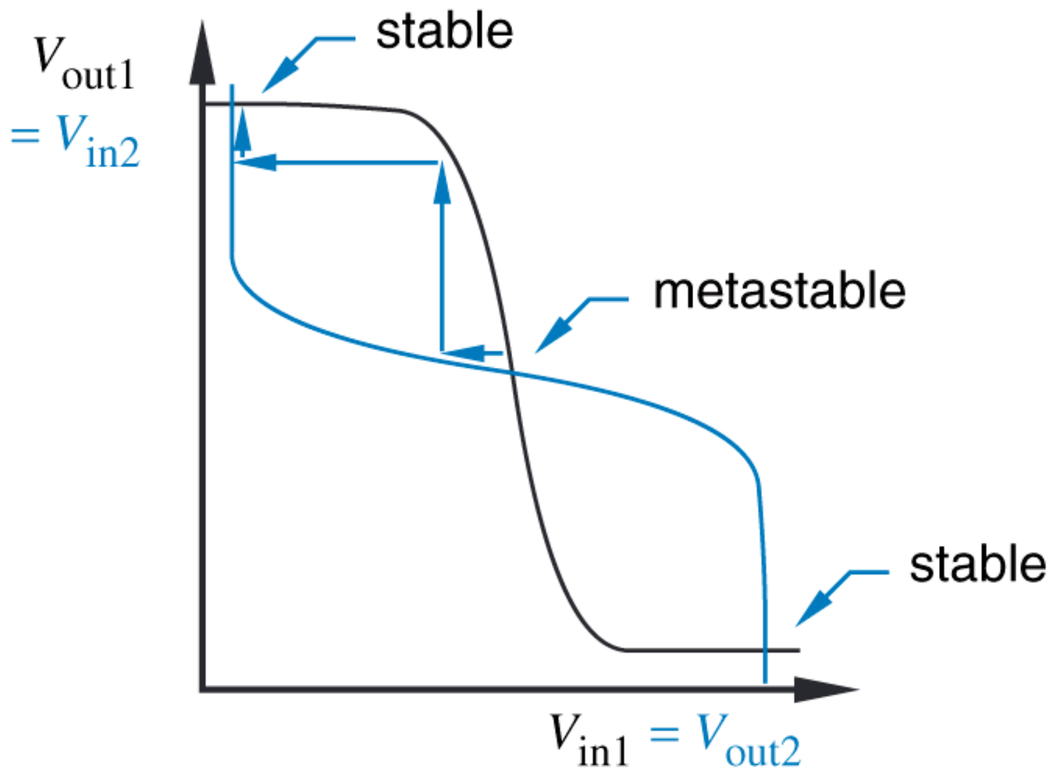




In metastable state

Metastability

- Metastability is inherent in a bistable element

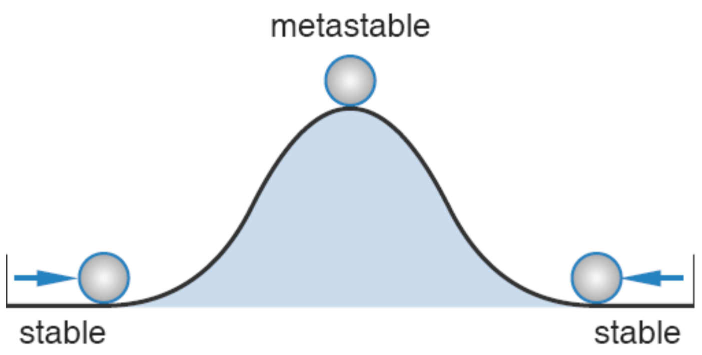


Transfer function:

$$V_{out1} = T(V_{in1})$$

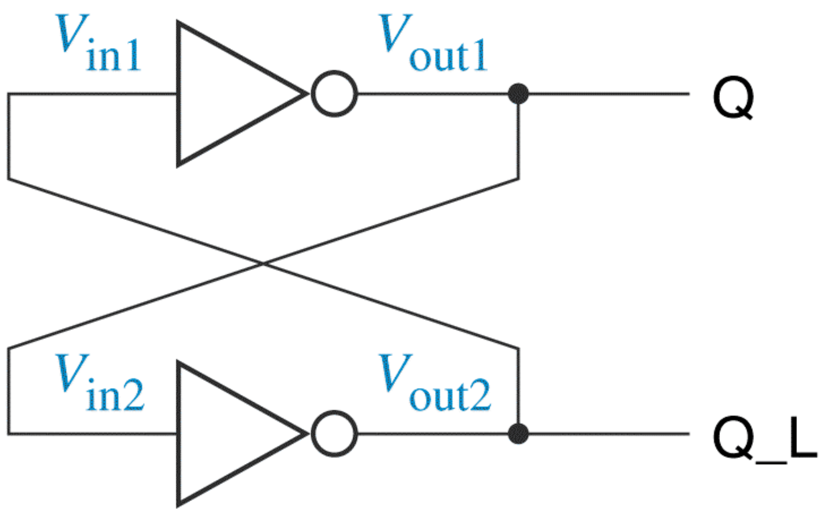
$$V_{out2} = T(V_{in2})$$

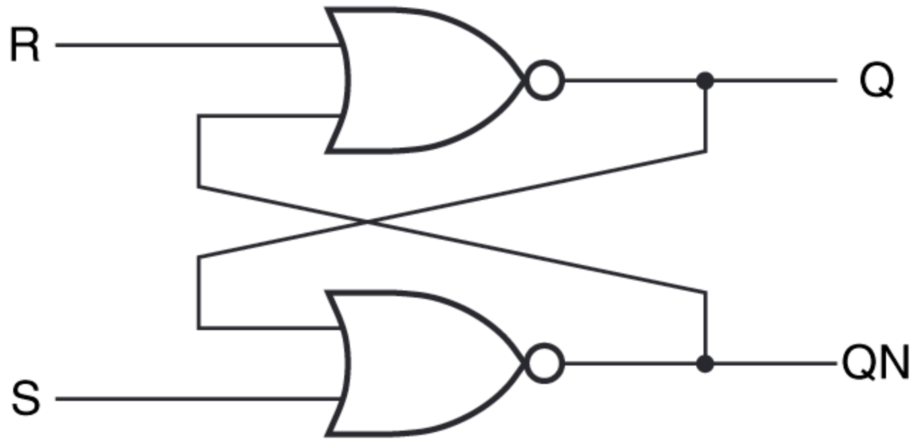
- Two stable points, one metastable point



Control the inputs for the bistable element?

- How do we control its inputs?
  - Add control inputs S and R
- Now we have an S-R latch
- When S=R=0, Q "stores" the last Q (S: set, R: reset)
- 2 NOR gates are used



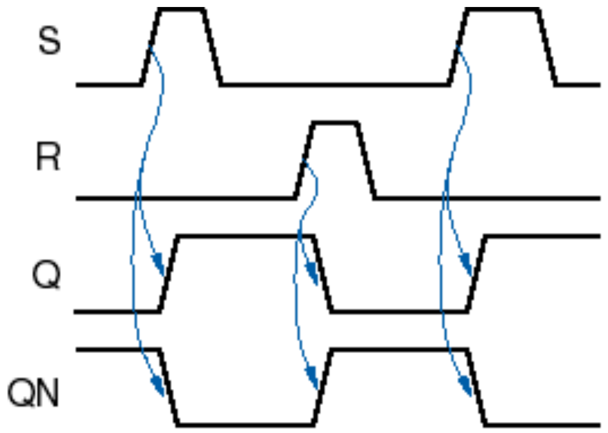


S	R	Q	QN
0	0	last Q	last QN
0	1	0	1
1	0	1	0
1	1	0	0

S-R latch operation

- When S=R=0, the latch behaves like a bistable element
  - Because $(x+0)' = x' \cdot 0' = x' \cdot 1 = x'$
- When S=R=1, we have both Q and QN equal to 0
  - Because $(x+1)' = x' \cdot 1' = x' \cdot 0 = 0$
- Abnormal state for S-R latch is reached when S=R=1
  - Because Q=QN=0, but Q should $\neq$ QN, logically
- Can you trace the remaining rows of the function table?
  - That is, when (S=0, R=1) and (S=1, R=0), what are Q and QN?

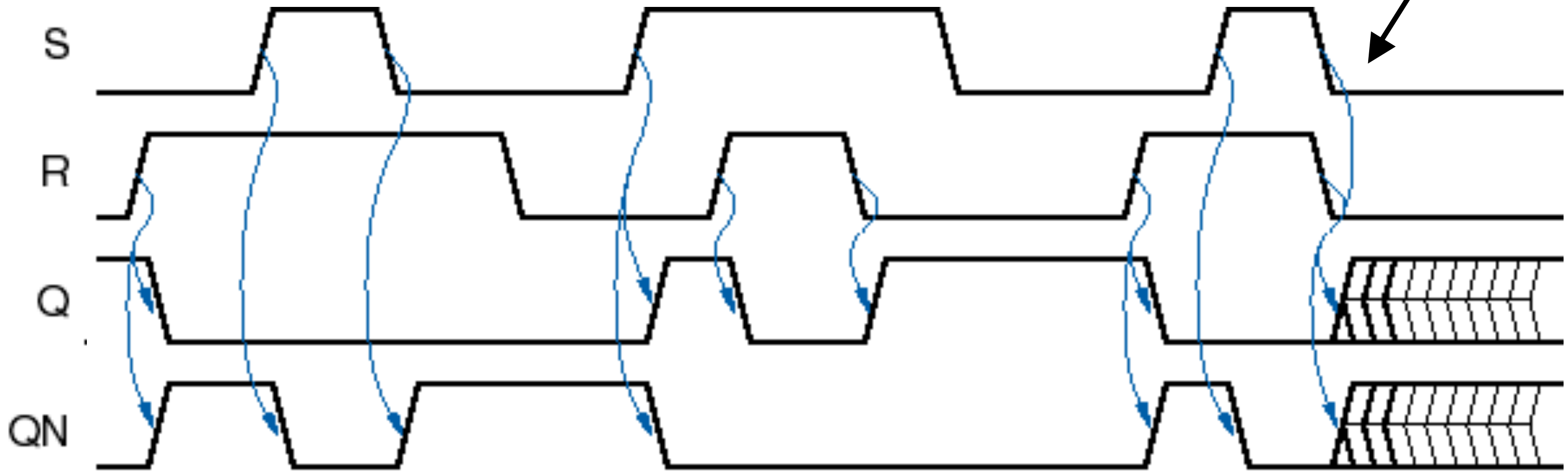
S-R latch operation (blue arrows indicate causality)



Normal operation, i.e. S, R not both 1

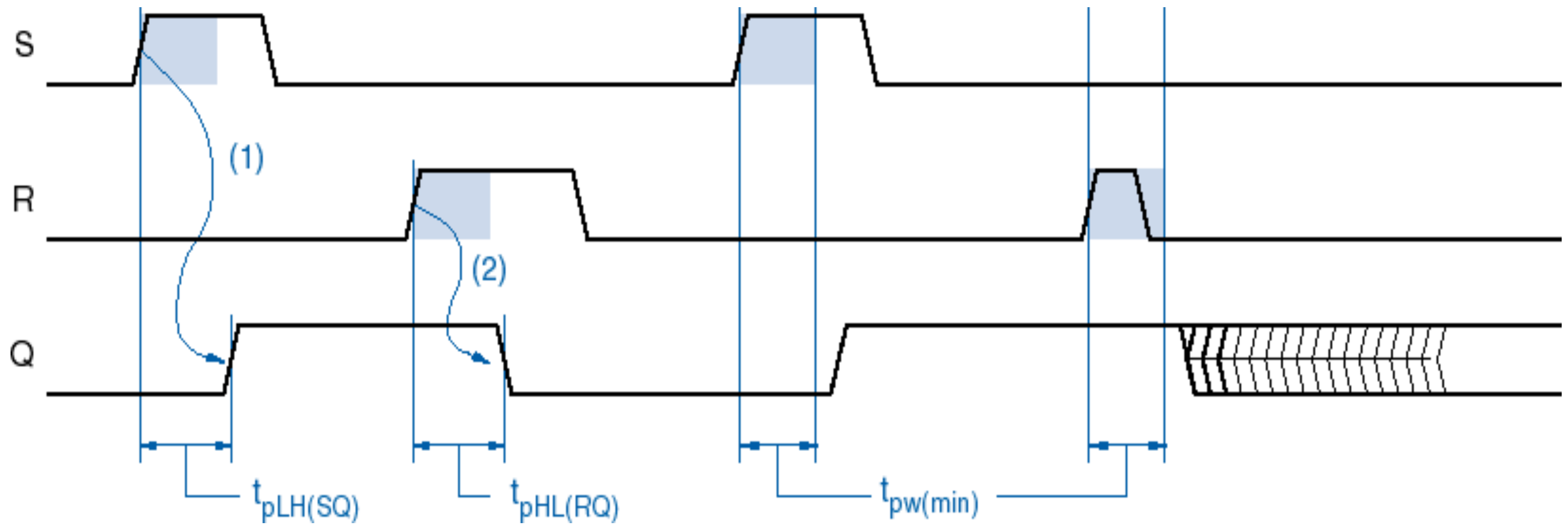
Abnormal State: Q=QN=0

Metastability is possible if S and R are negated simultaneously!

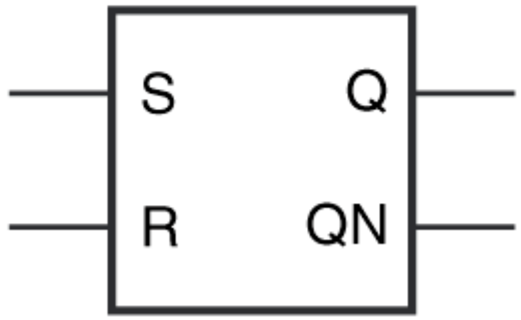


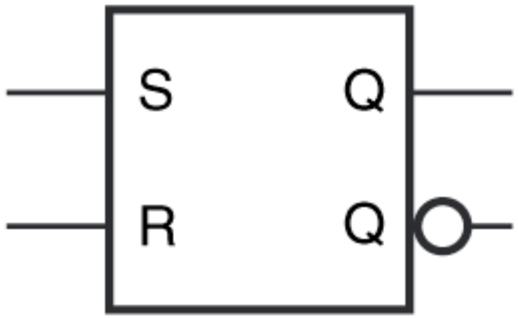
S-R latch timing parameters

- Propagation delay, e.g. LO-HI transition on S causes LO-HI on Q
- Minimum pulse width, $t_{pw(min)}$: minimum duration for an S (or R) transition to cause a “stable” transition of Q
- Latch remains metastable for a random duration of time
- Wait for at least $t_{pw(min)}$ after asserting S/R to avoid metastability

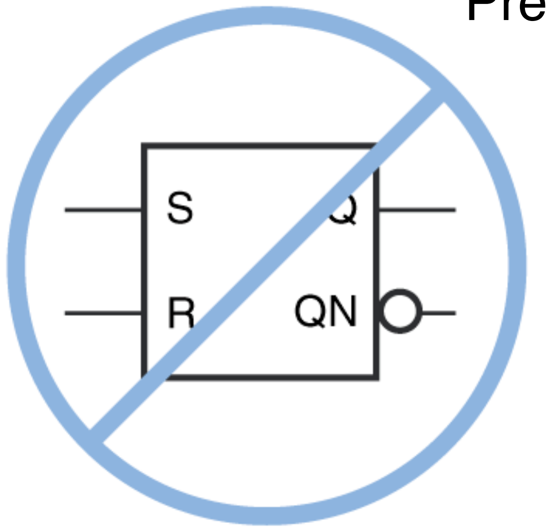


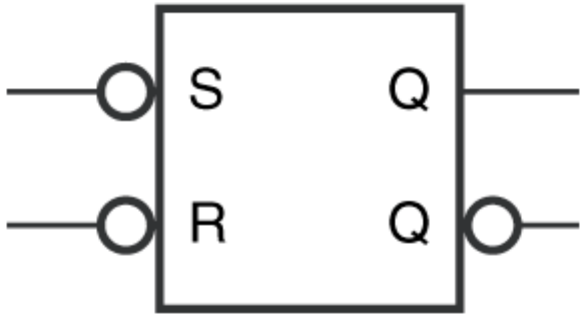
S-R latch symbols





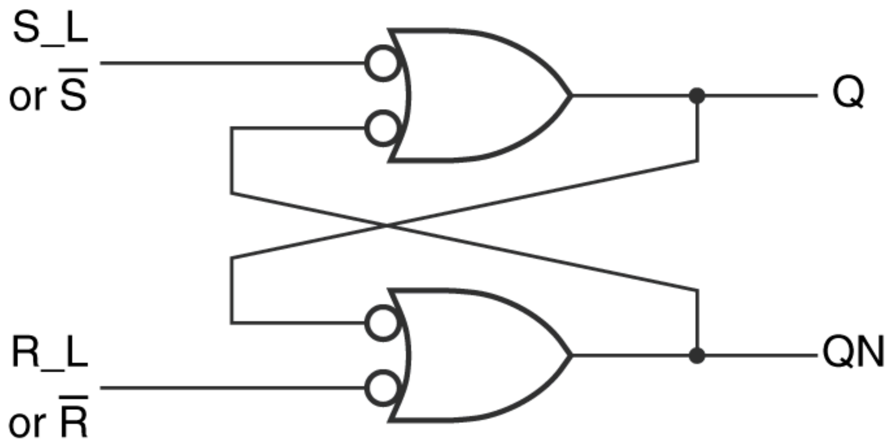
Preferred symbol

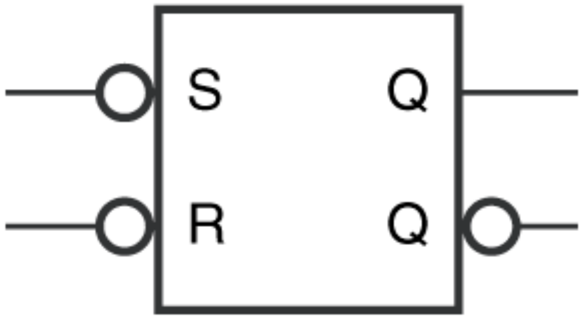




What’s this?!

S-R latch using NAND gates (a.k.a. S-bar-R-bar latch)





- S and R are active-low (not Q/QN)
- Can be built with NAND gates
- Much more popular than NOR logic
- Because NAND is faster than NOR
- Abnormal state: S_L = R_L = 0

S_L	R_L	Q	QN
0	0	1	1
0	1	1	0
1	0	0	1
1	1	last Q	last QN

S-R latch with enable (a.k.a. gated/clocked S-R latch)

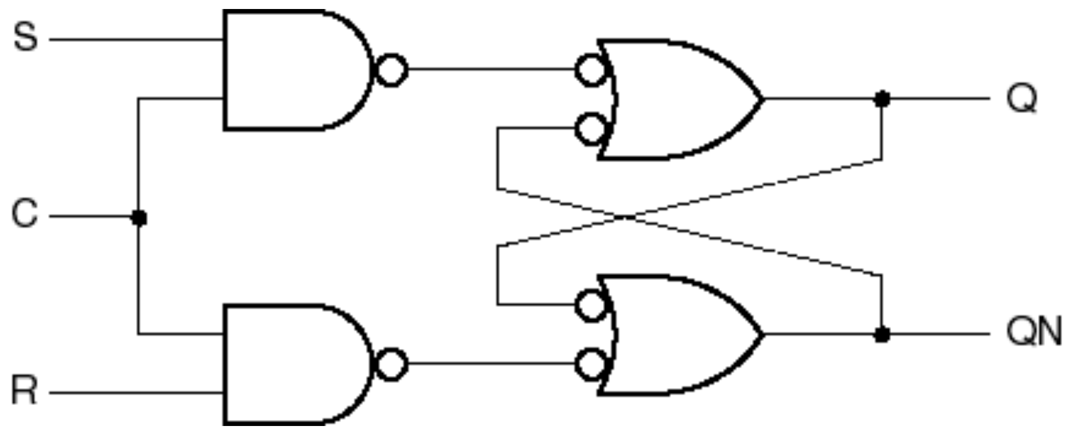
S	R	C	Q	QN
0	0	1	last Q	last QN
0	1	1	0	1
1	0	1	1	0
1	1	1	1	1
x	x	0	last Q	last QN

Let C decide whether S and R can result in a bistable element.

C can be a clock signal!

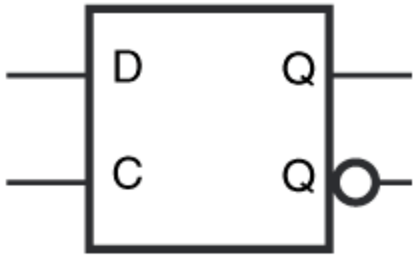
Now S and R are active-high again.

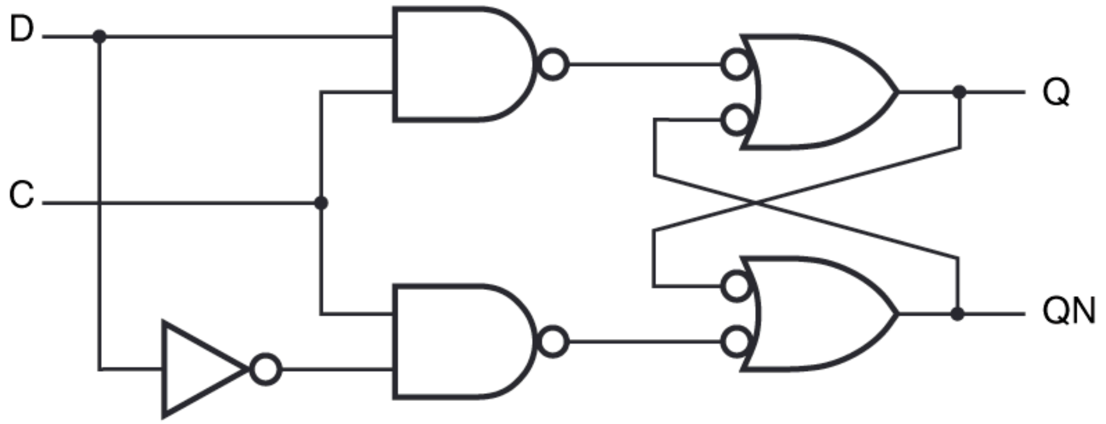
Metastability occurs when S=R=1, and C transitions from 1 to 0.



D latch (D for Data)

C	D	Q	QN
1	0	0	1
1	1	1	0
0	x	last Q	last QN

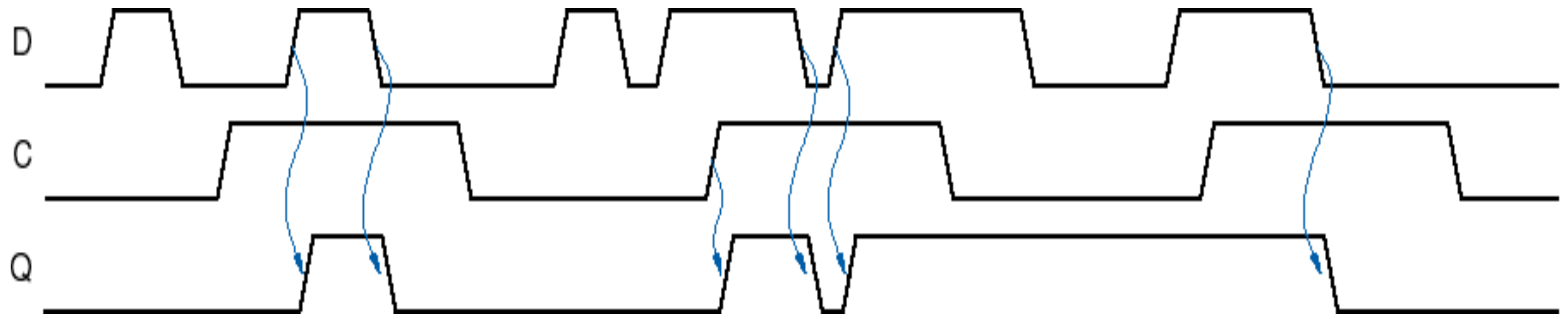




S	R	C	Q	QN
0	0	1	last Q	last QN
0	1	1	0	1
1	0	1	1	0
1	1	1	1	1
x	x	0	last Q	last QN

Think of this as a gated S-R latch with R=S’ ⇒ Only 3 rows of the FT remain.

D latch operation



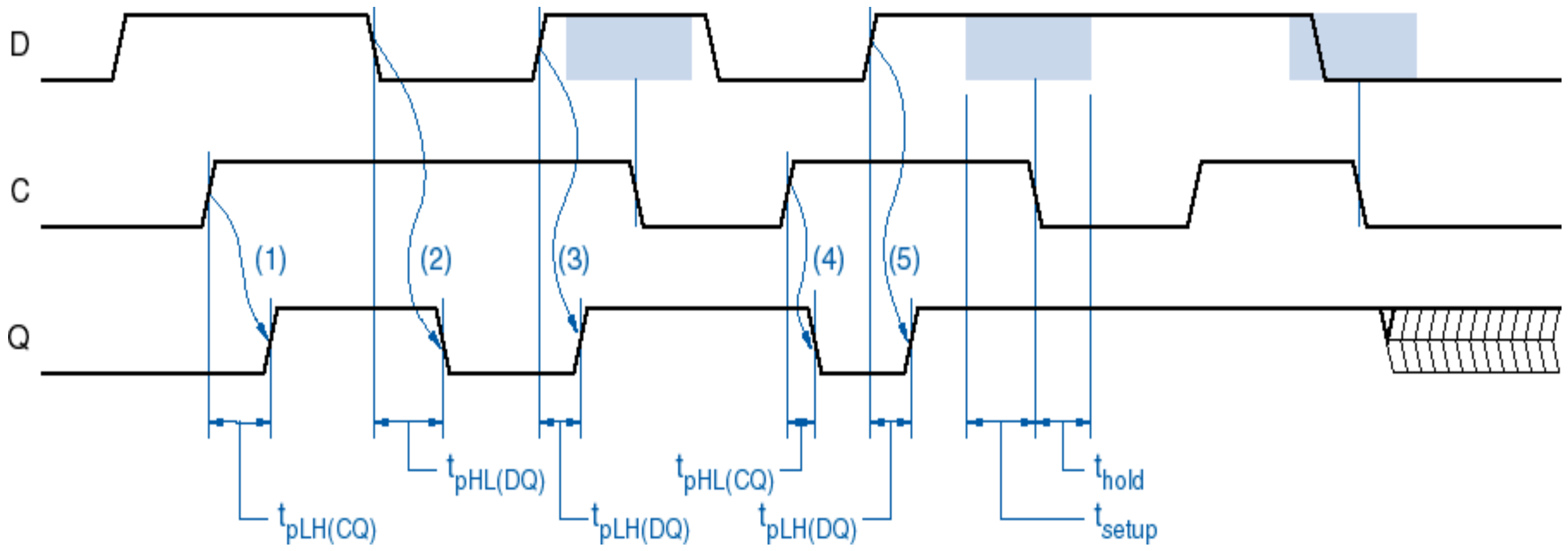
When C = 1, Q = D, i.e. Q “follows” D

When C = 0, Q does not change, i.e. Q “stores” the last D

D latch is (clock) level-triggered, not edge-triggered

D latch timing parameters

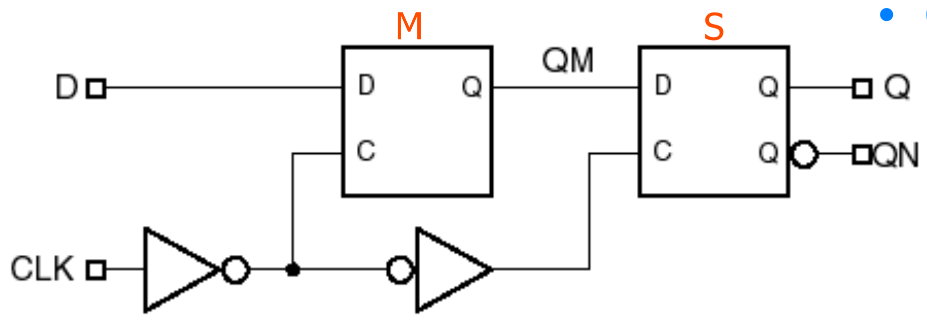
- When C = 1, Q follows D with delay
  - Propagation delays (in Q caused by C or D), i.e. t_pLH(**) and t_pHL(**)
- When C = 0, Q remembers D during C's 1→ 0 transition
  - Setup time (D before C's falling edge), i.e. t_{setup}
  - Hold time (D after C's falling edge), i.e. t_{hold}
  - If D does not remain the same for > (t_{setup}+t_{hold}) ⇒ metastability!

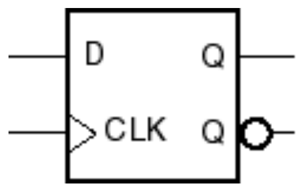


Positive edge-triggered D flip-flop (uses 2 D latches)

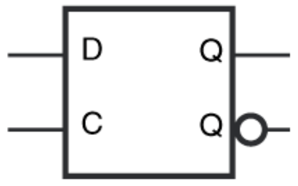
	D	CLK	Q	QN
1	0	↑	0	1
2	1	↑	1	0
3	x	0	last Q	last QN
4	x	1	last Q	last QN

- Row 1: M closed, S open, Q = QM = D = 0
- Row 2: M closed, S open, Q = QM = D = 1
- Row 3: M open and follows D, S closed
- Row 4: S open and follows QM, M closed
  - Row 1/2: M open → close, S close → open
  - Row 3 and Row 4 keep the **last Q** and **QN**
- S is open while CLK=1, but only changes at the beginning of this interval, because M is closed (unchanged) during rest of interval.
- Q only changes during CLK transitions



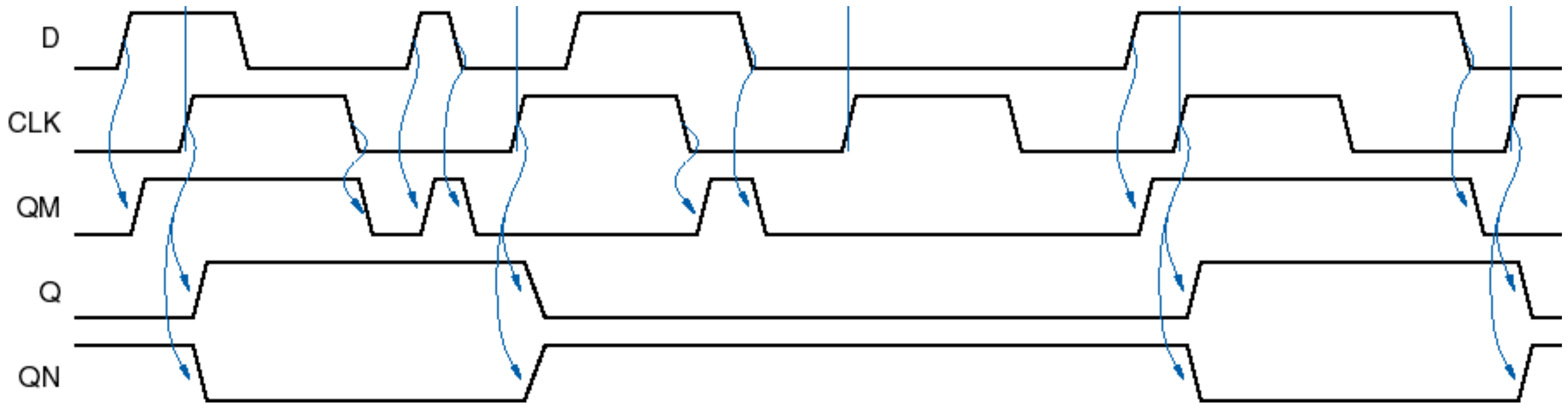


C	D	Q	QN
1	0	0	1
1	1	1	0
0	x	last Q	last QN



D latch

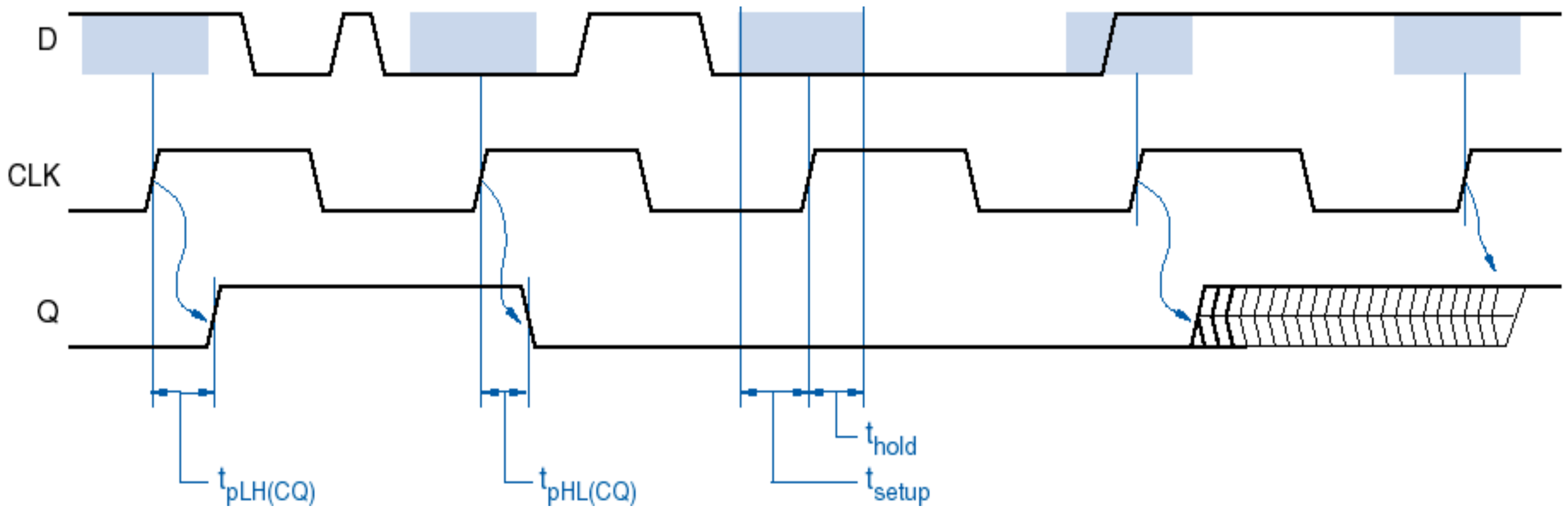
Positive edge-triggered D flip-flop functional behavior



CLK (f/ M)	CLK_L (f/ S)	Latch M status	QM	Latch S status	Q
1	0	closed	D_{last}@ ↑	open	QM = D_{last}@ ↑
↓	↑	closed → open	D_{last}@ ↑	open → closed	D_{last}@ ↑
0	1	open	D (QM	closed	D_{last}@ ↑
↑	↓	open → closed	D	closed → open	D_{last}@ ↑
1	0	closed	D@ ↑	open	QM = D@ ↑

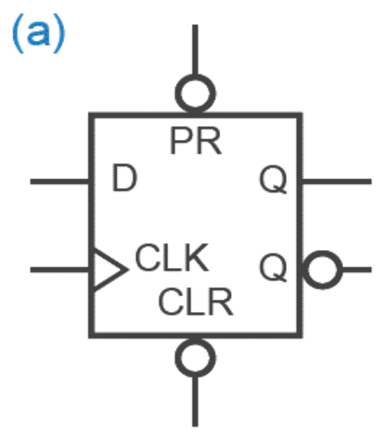
D flip-flop timing parameters

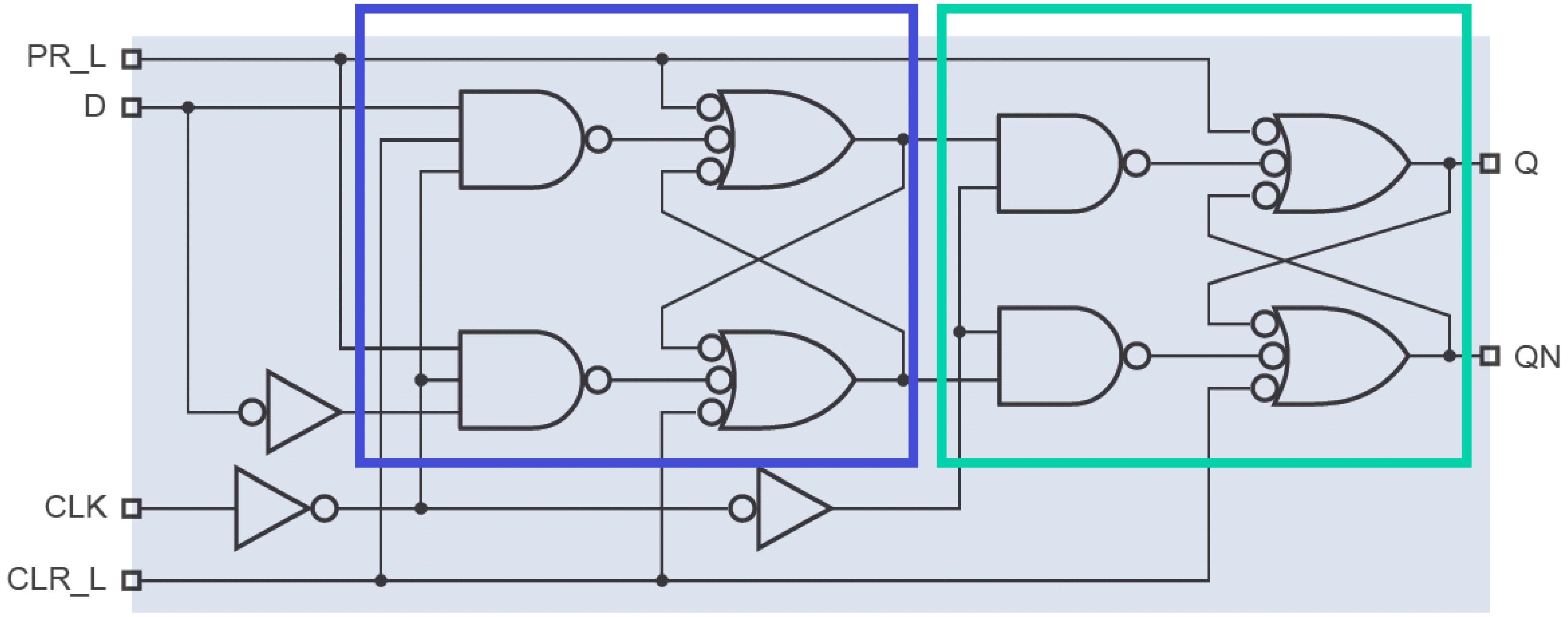
- Propagation delay (from CLK threshold to Q threshold, i.e. “CQ”)
- Setup time (D before CLK)
- Hold time (D after CLK)
- D must not change within (Setup + Hold) window
- Window occurs around the triggering edge of CLK



Positive edge-triggered D flip-flop with Preset and Clear

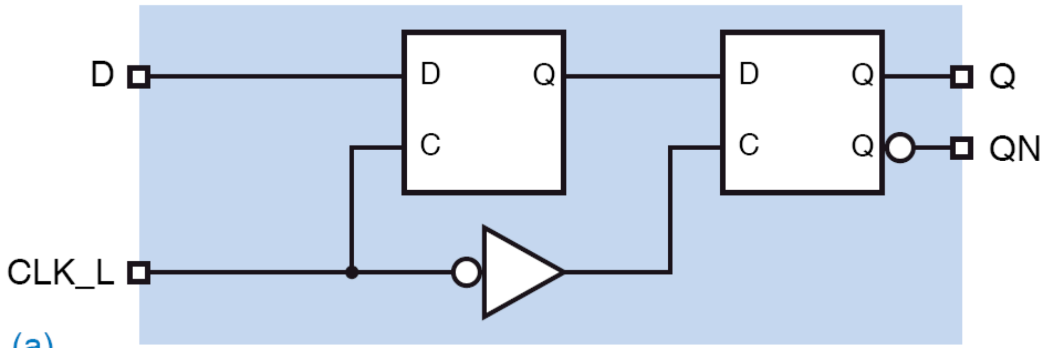
- Preset and clear inputs
  - Works like S-R latch





Negative edge-trigged D flip-flop

- Invert the input CLK signal



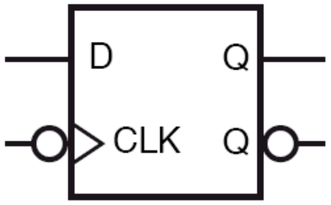
(a)

D	CLK	Q	QN
0	↑	0	1
1	↑	1	0
x	0	last Q	last QN
x	1	last Q	last QN

+ve edge-triggered D FF

D	CLK_L	Q	QN
0	↓	0	1
1	↓	1	0
x	0	last Q	last QN
x	1	last Q	last QN

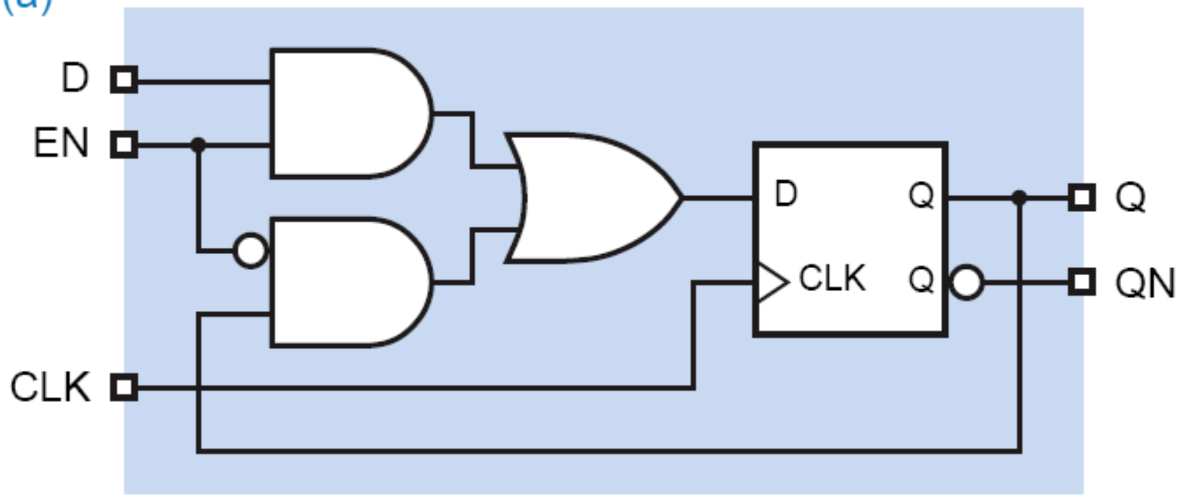
(b)



(c)

Positive-edge-triggered D flip-flop with enable

(a)



EN = 1: D = external D

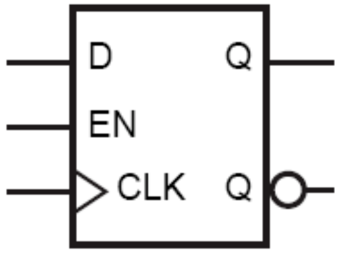
EN = 0: D = last Q

CLK = 0/1: keep last Q

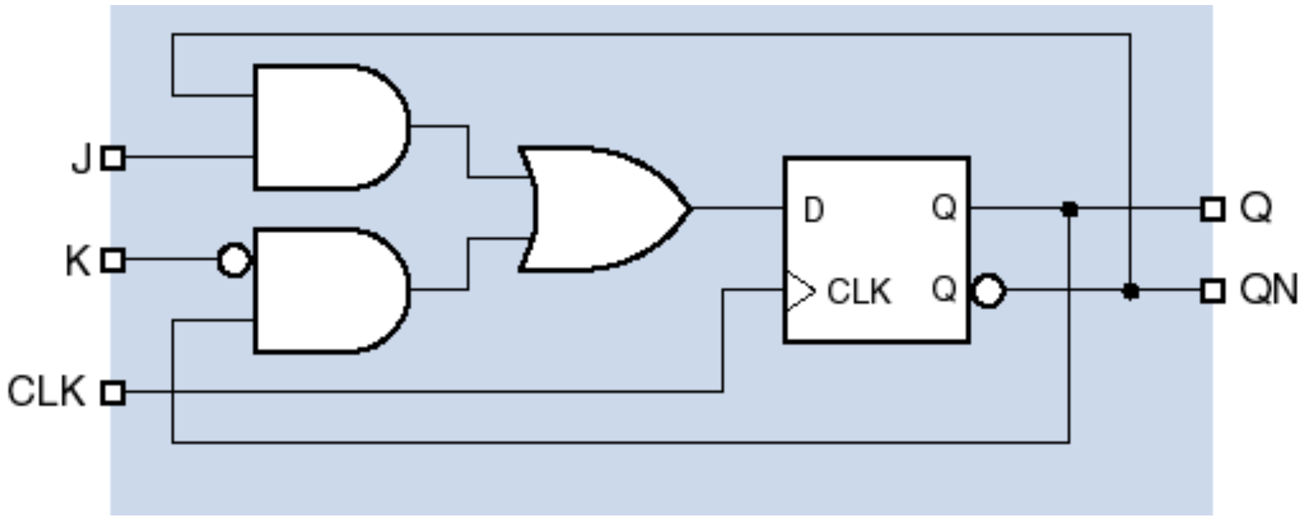
(b)

D	EN	CLK	Q	QN
0	1	↑	0	1
1	1	↑	1	0
x	0	↑	last Q	last QN
x	x	0	last Q	last QN
x	x	1	last Q	last QN

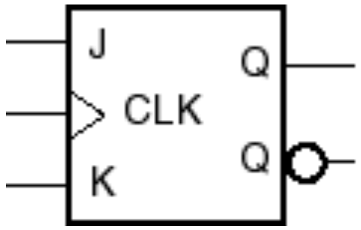
(c)



Edge-Triggered J-K flip-flop

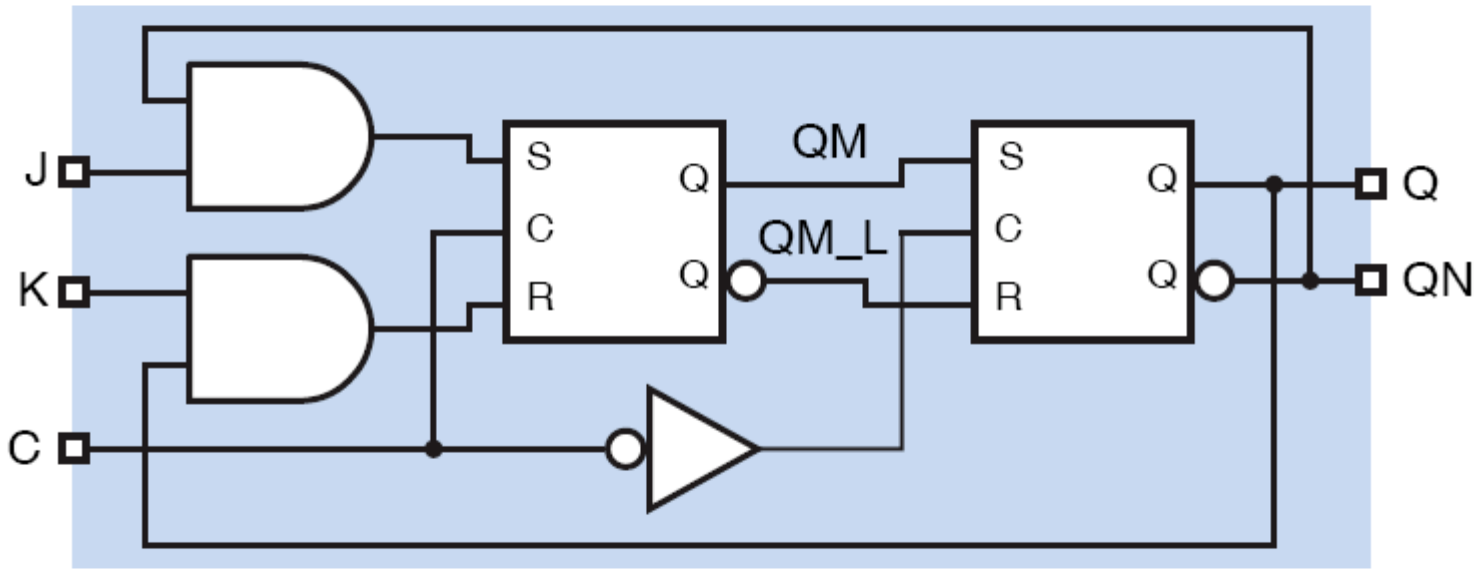


J	K	CLK	Q	QN
x	x	0	last Q	last QN
x	x	1	last Q	last QN
0	0	↑	last Q	last QN
0	1	↑	0	1
1	0	↑	1	0
1	1	↑	last QN	last Q

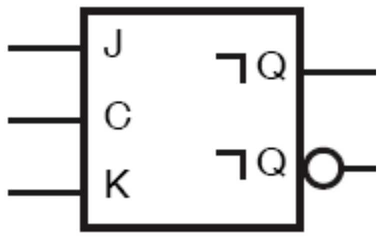


- Not used much anymore
- Don’t worry about them

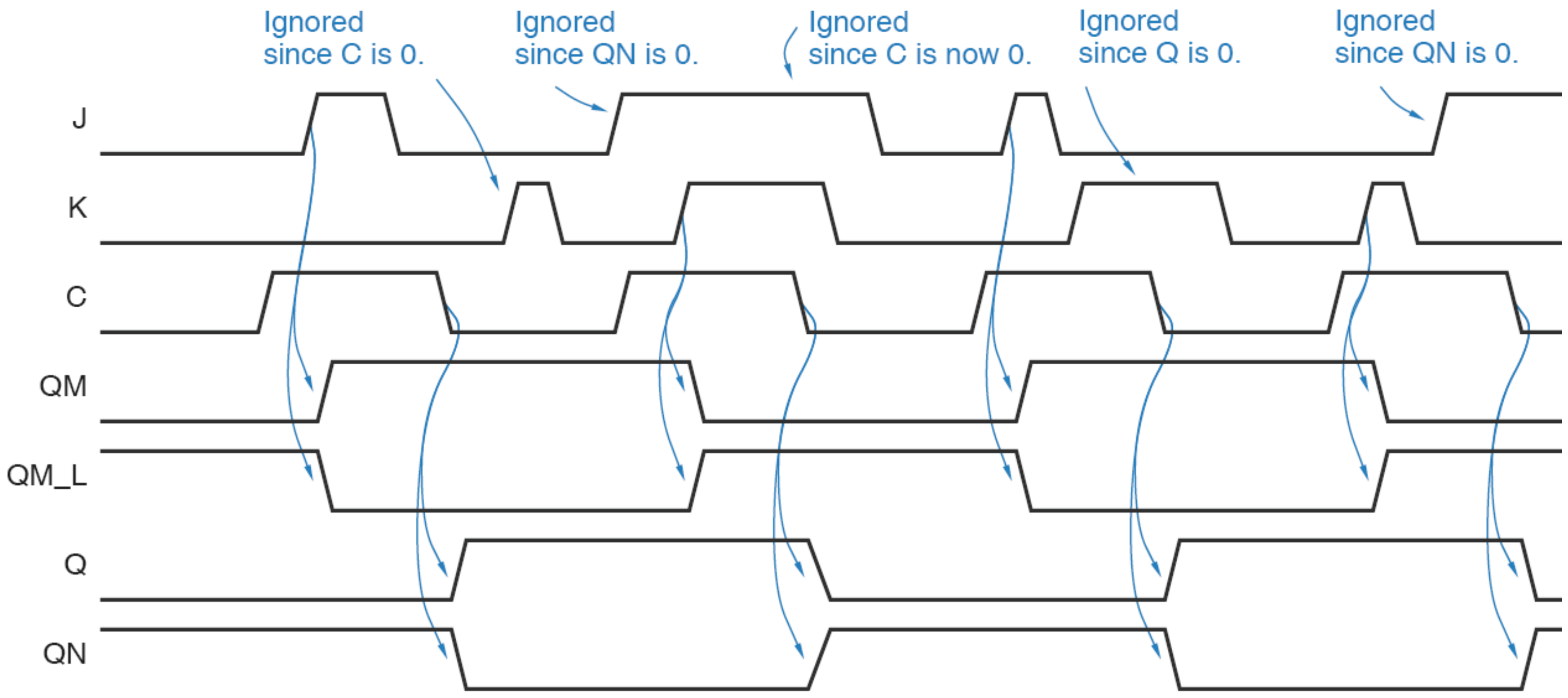
Master/slave J-K flip-flop (pulse-triggered)



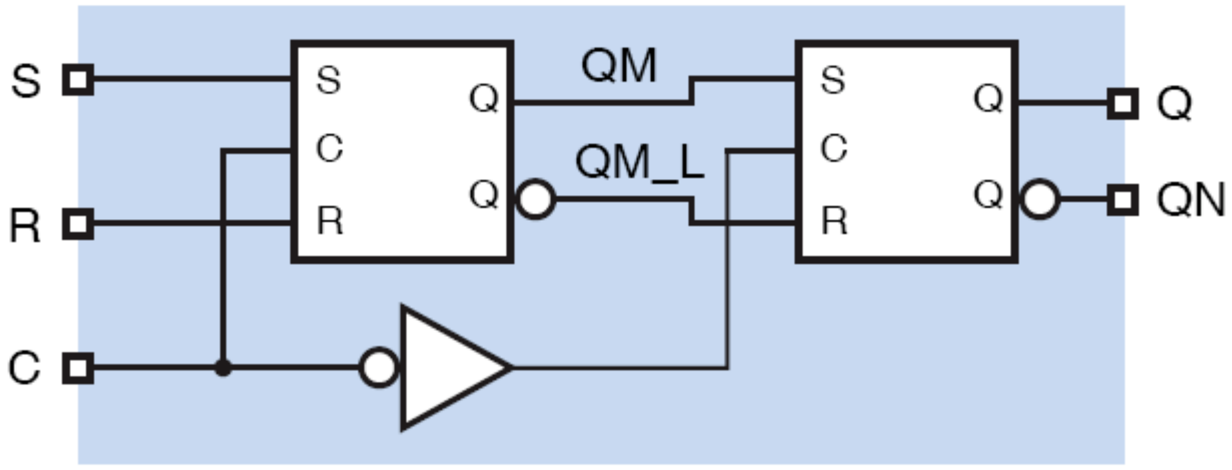
J	K	C	Q	QN
x	x	0	last Q	last QN
0	0	⎍	last Q	last QN
0	1	⎍	0	1
1	0	⎍	1	0
1	1	⎍	last QN	last Q



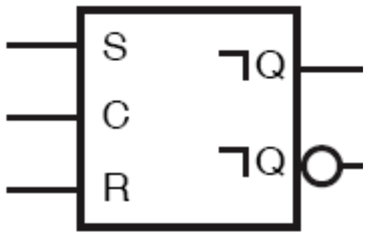
Timing diagram of master/slave J-K flip-flop



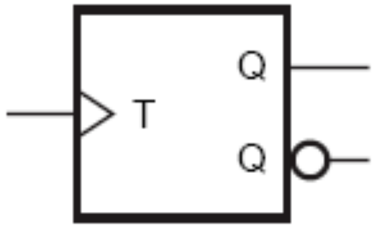
Master/slave S-R flip-flop

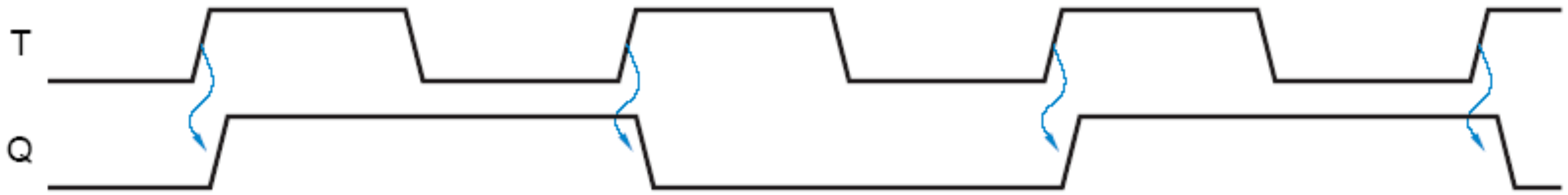


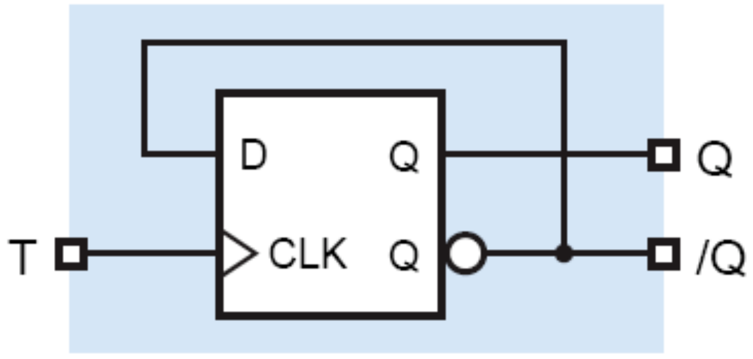
S	R	C	Q	QN
x	x	0	last Q	last QN
0	0	⎍	last Q	last QN
0	1	⎍	0	1
1	0	⎍	1	0
1	1	⎍	undef.	undef.

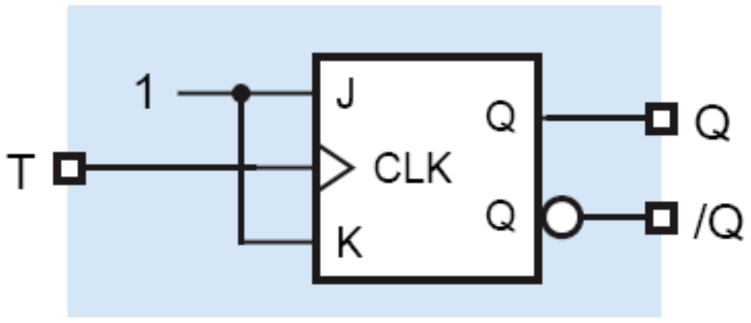


T flip-flops



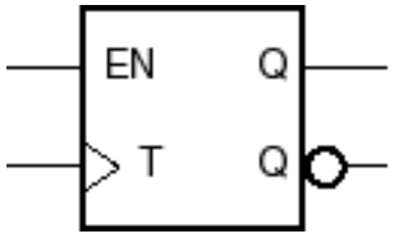


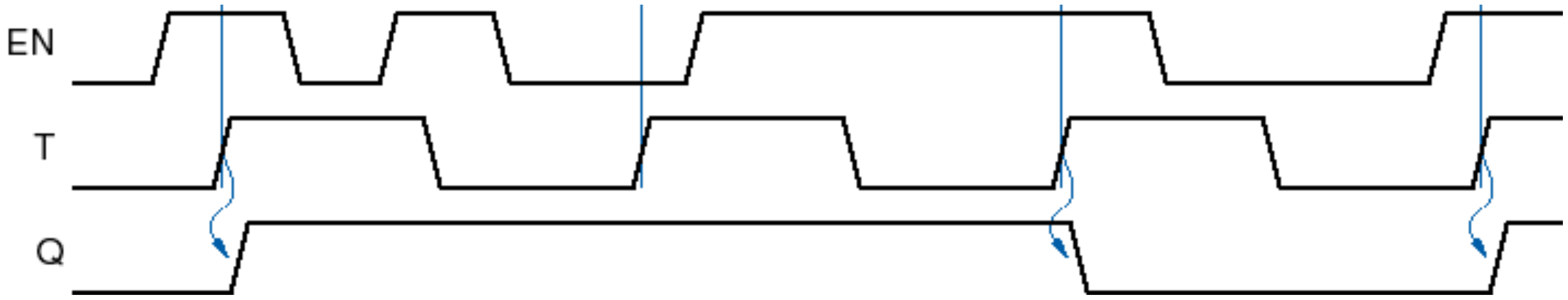


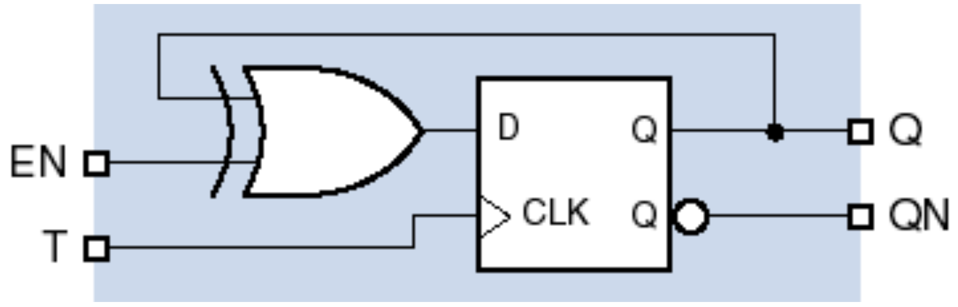


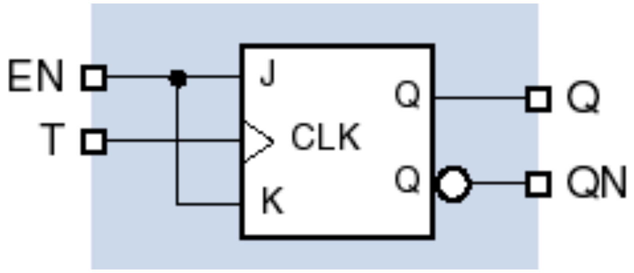
T flip-flops with enable

- Important for counters









Many types of latches and flip-flops

- S-R latch
- S_L-R_L latch
- S-R latch with enable
- D latch
- Edge-triggered D flip-flop
- Edge-triggered D flip-flop with enable
- Edge-triggered D flip-flop with preset and clear
- Scan flip-flop
- Edge-triggered J-K flip-flop
- Master/slave S-R flip-flop
- Master/slave J-K flip-flop
- T flip-flop
- T flip-flop with enable