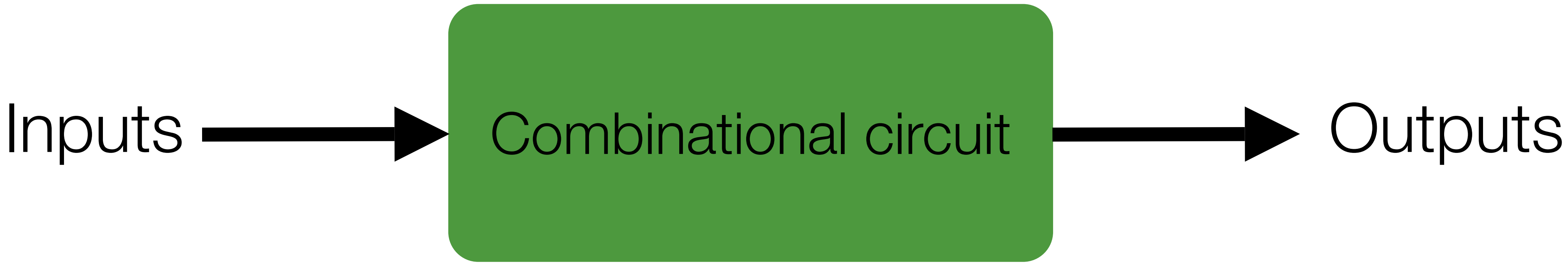
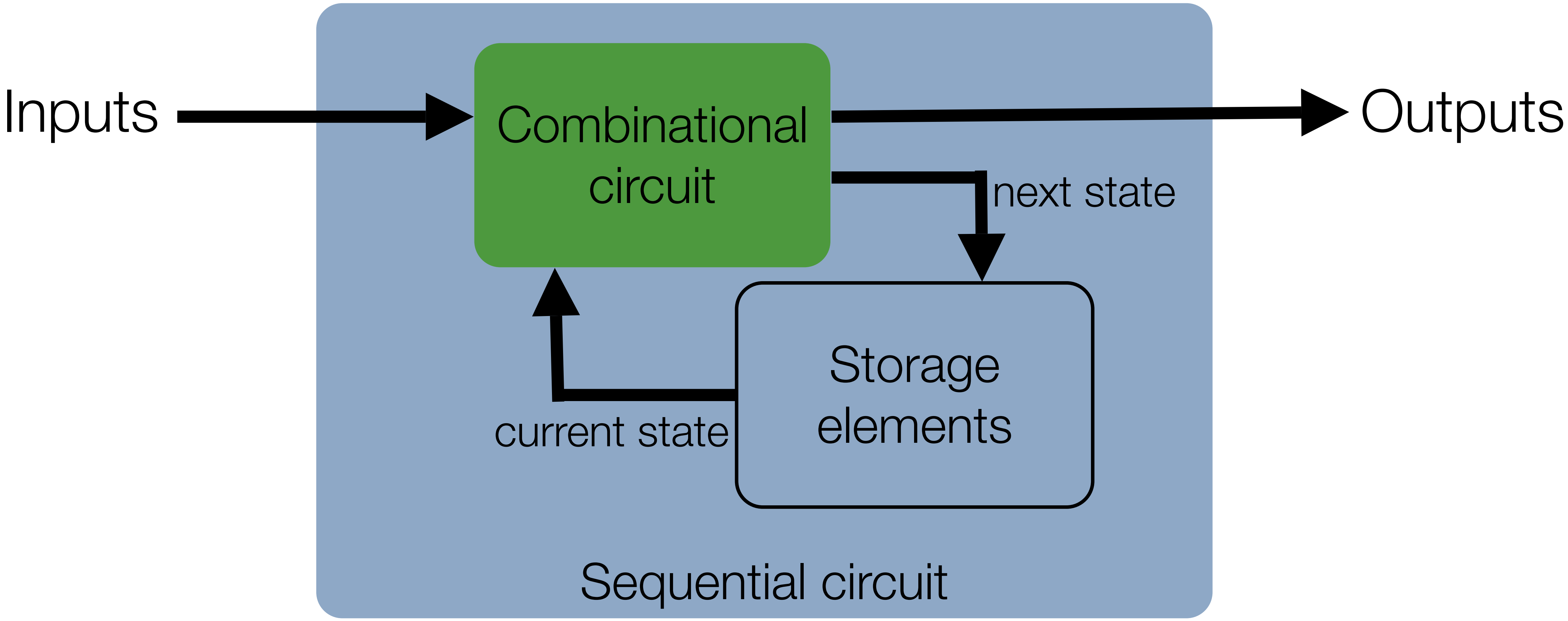
CSEE 3827: Fundamentals of Computer Systems

Latches and Flip Flops

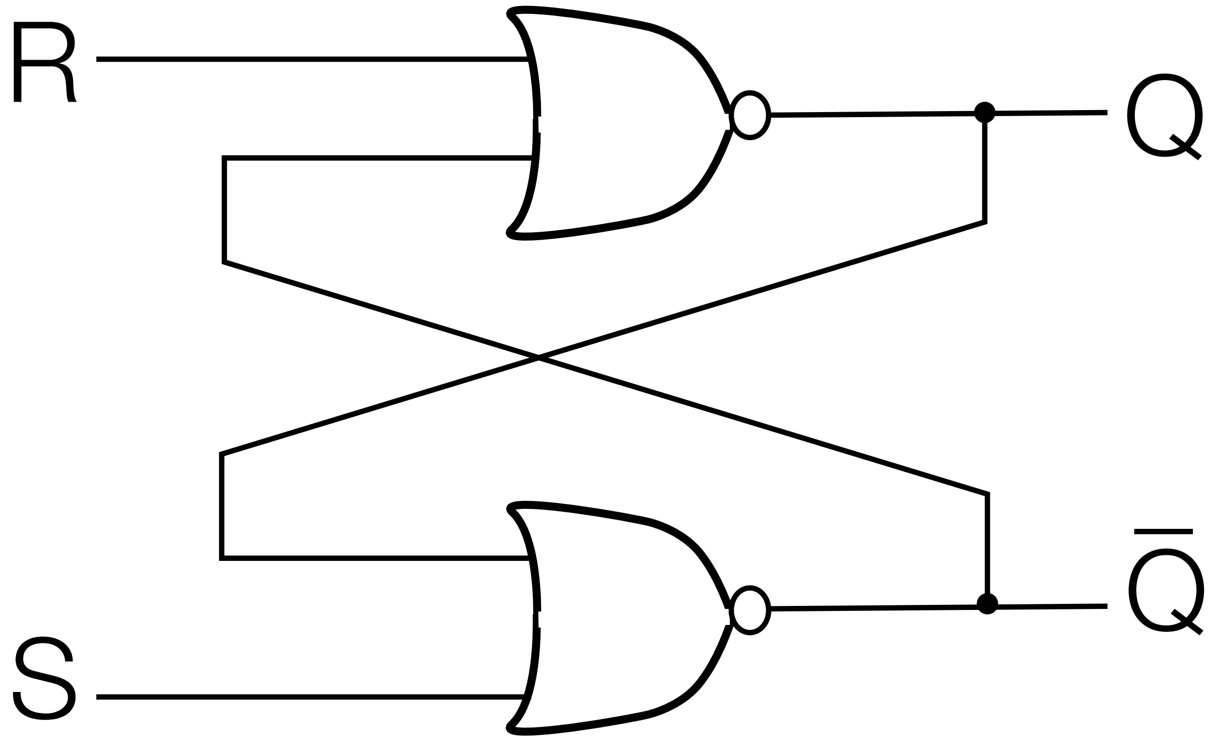
Combinational v. sequential logic





SR latch

- Latch constructed of cross-coupled NOR gates



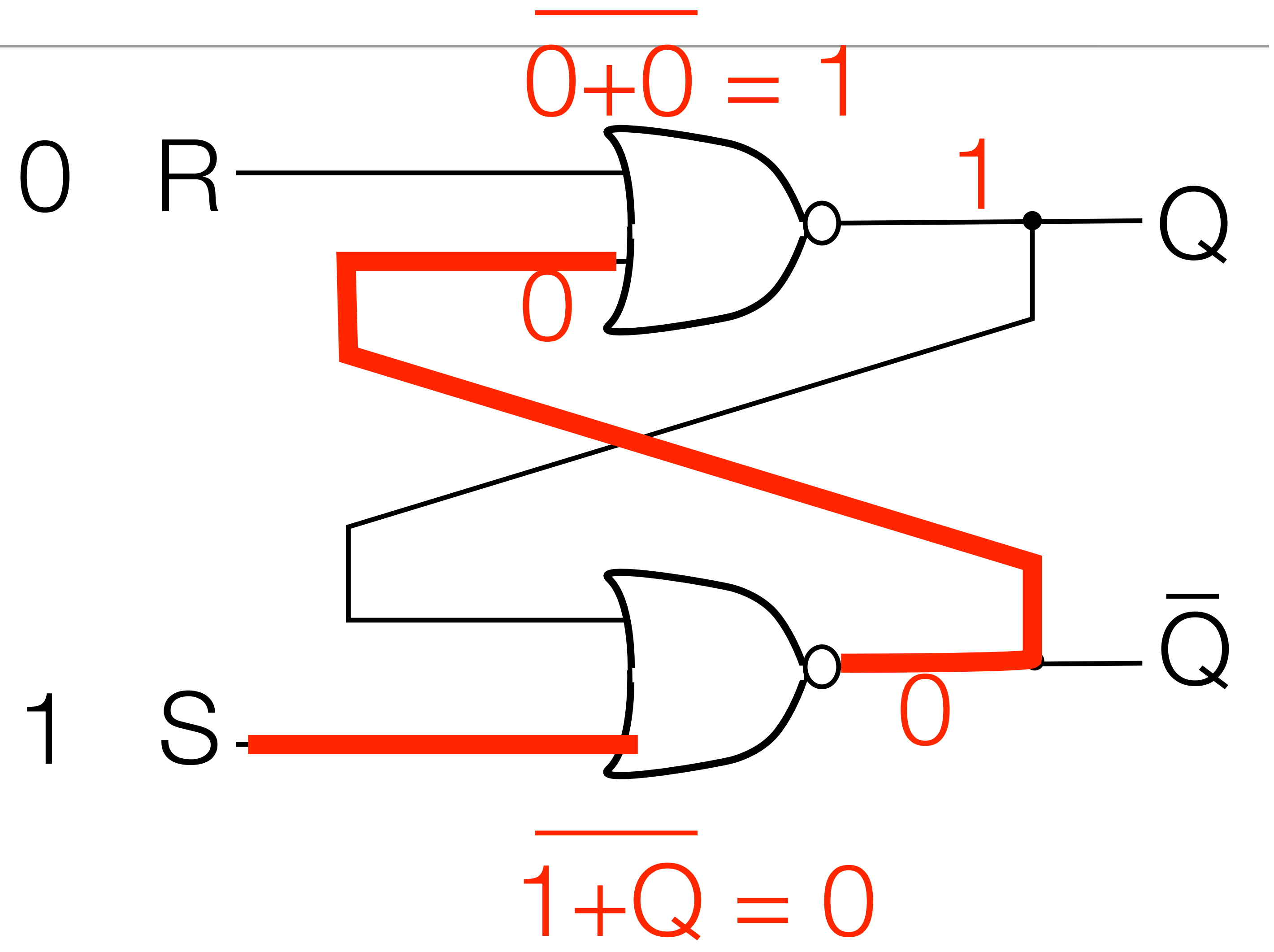
"R" for reset

"S" for set

outputs are complements of each other

- What's so new? The wires "loop back" (output feeding back into circuit)

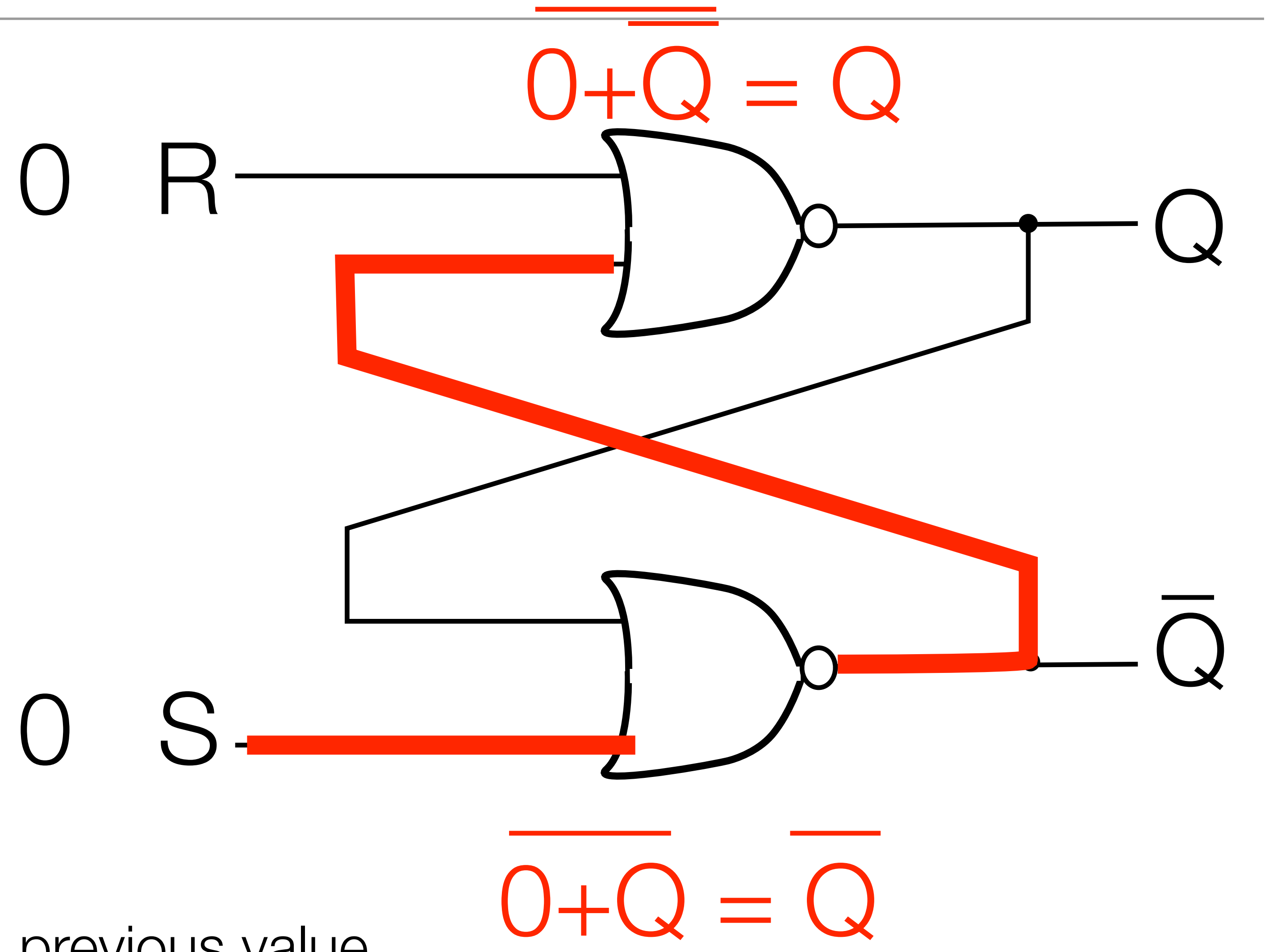
SR latch - “set”, “reset”



R	S	Q	$\overline{Q}$
0	0		
0	1	1	0
1	0	0	1
1	1		

← By symmetry along horizontal cut of latch

SR latch - “hold”



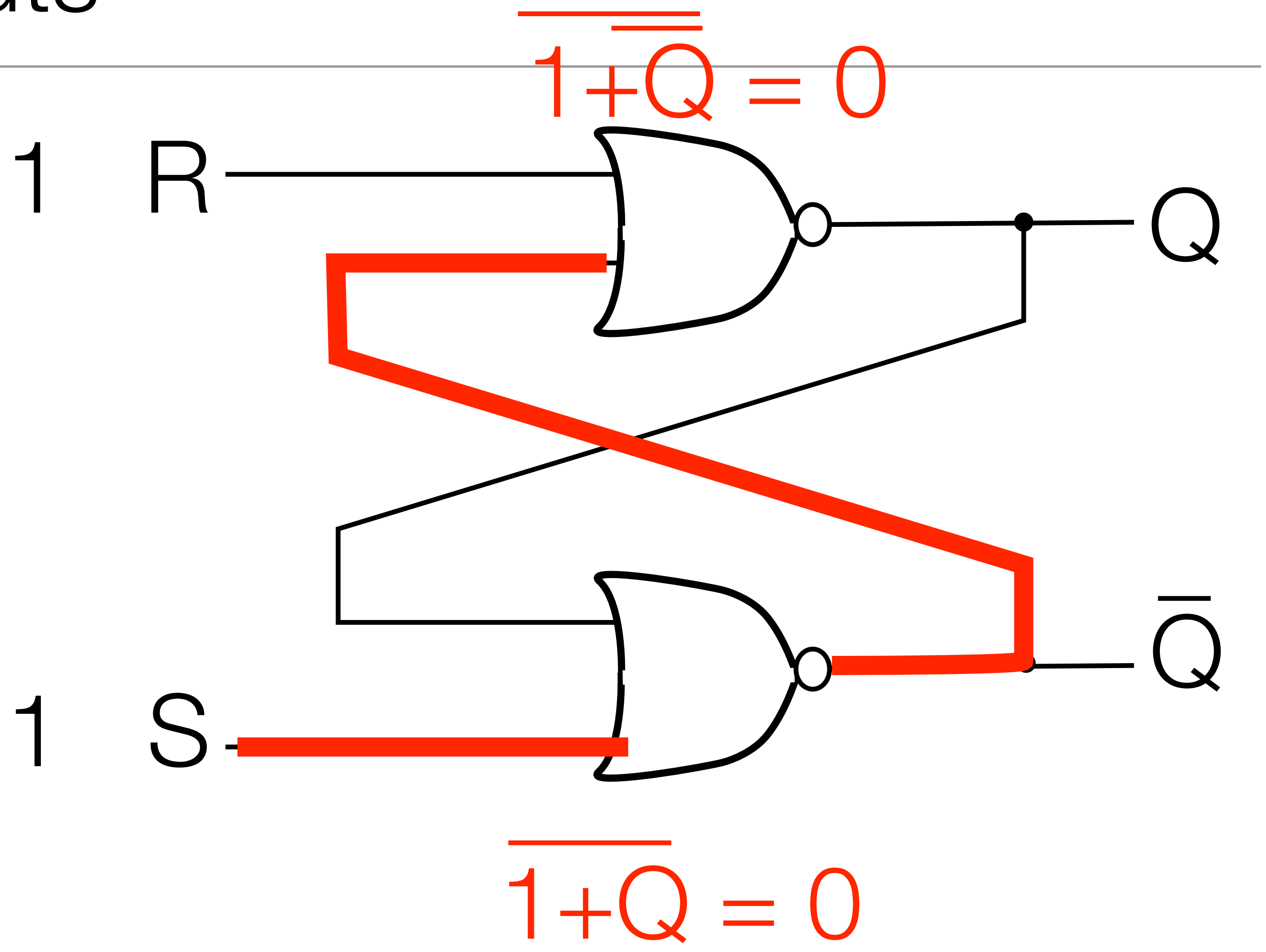
$\overline{0+\overline{Q}} = Q$

$\overline{0+Q} = \overline{Q}$

R	S	Q	$\overline{Q}$
0	0	Q	$\overline{Q}$
0	1	1	0
1	0	0	1
1	1		

Hold previous value

SR latch - “invalid inputs”

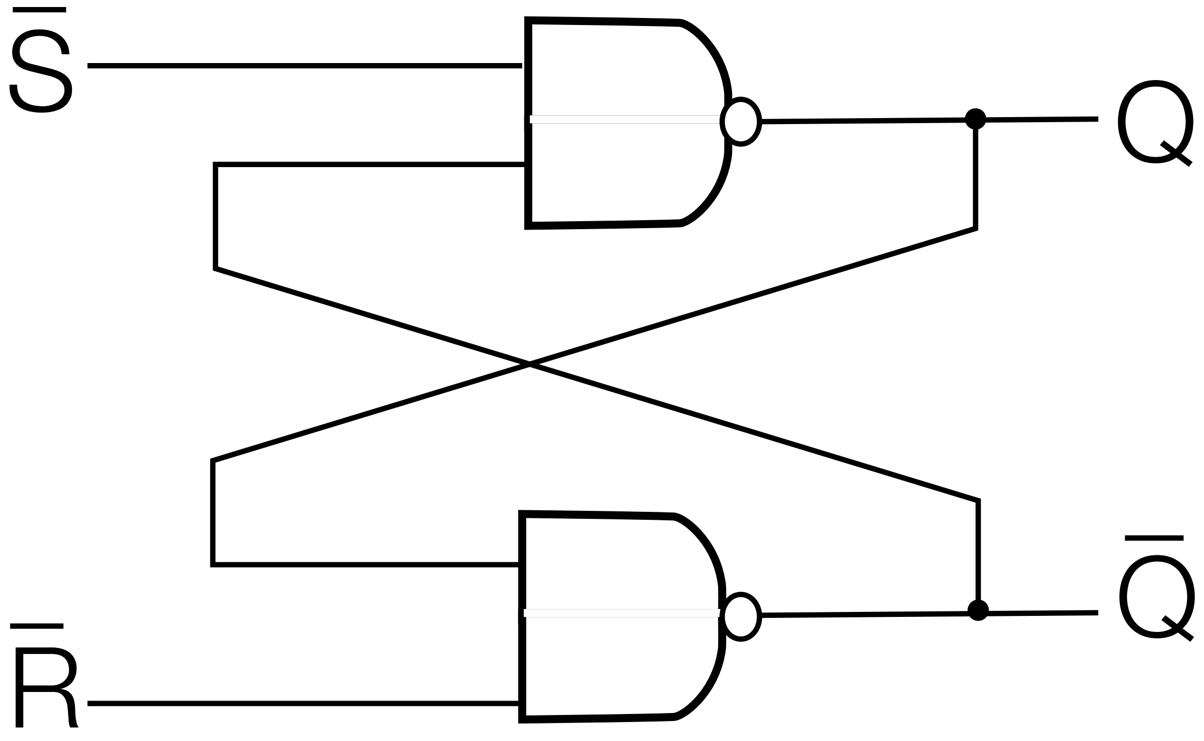


R	S	Q	$\overline{Q}$	
0	0	Q	$\overline{Q}$	*No change*
0	1	1	0	*Set (Q=1)*
1	0	0	1	*Reset (Q=0)*
1	1	0	0	

Bad - do not use!

$\overline{S}\overline{R}$ latch

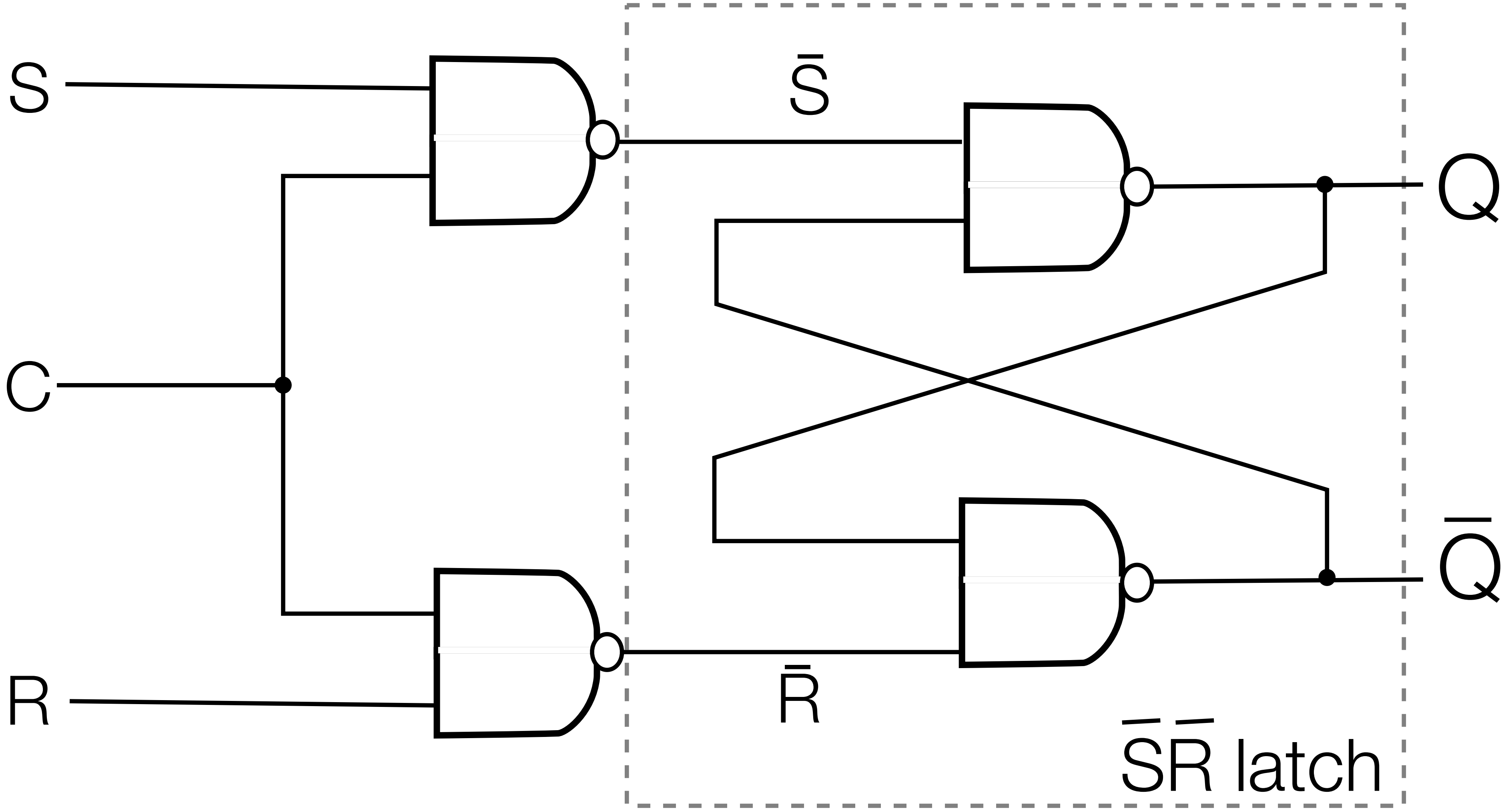
- Latch constructed of cross-coupled **NAND** gates



$\overline{R}$	$\overline{S}$	Q	$\overline{Q}$
0	0	1	1
0	1	0	1
1	0	1	0
1	1	Q	$\overline{Q}$

$\overline{R}$ and $\overline{S}$ have reverse behavior of SR latch,
so R and S have same behaviors

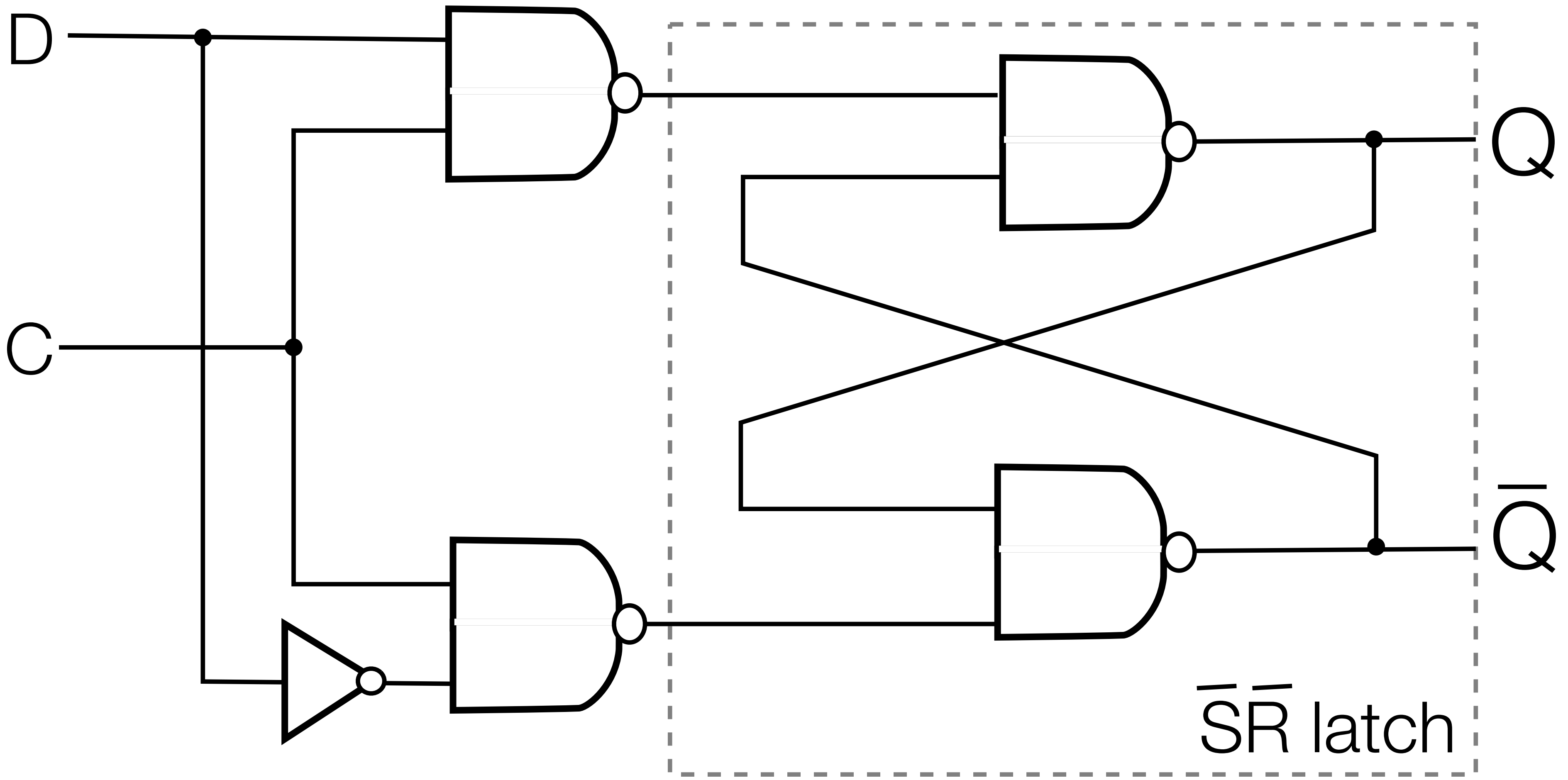
SR Latch with control



- C=0 $\overline{S}\overline{R}$ latch receives $\overline{S}$=1, $\overline{R}$=1, values “hold”
- C=1, first set of NAND gates invert S & R inputs to $\overline{S}$ & $\overline{R}$

C	R	S	Q	$\overline{Q}$
0	X	X	Q	$\overline{Q}$
1	0	0	Q	$\overline{Q}$
1	0	1	1	0
1	1	0	0	1
1	1	1	1	1

D Latch with control

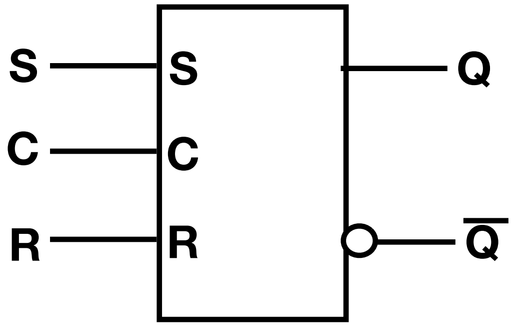


- With the control (C), no reason to ever have S=R
- C=0, latch holds value, C=1, Q=D

C	D	Q	$\overline{Q}$
0	X	Q	$\overline{Q}$
1	0	0	1
1	1	1	0

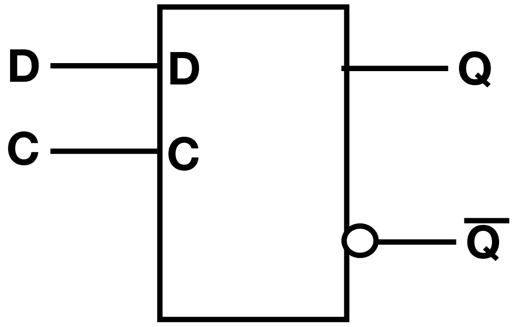
Circuit diagrams for latches

SR Latch with control



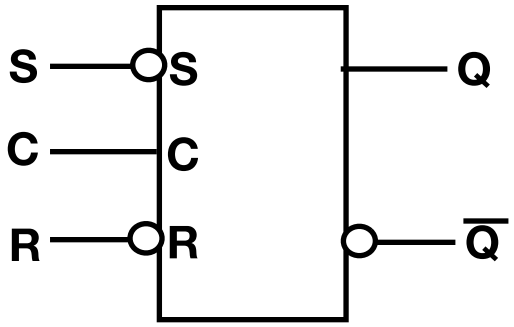
C	R	S	Q	Q
0	X	X	Q	$\overline{Q}$
1	0	0	Q	$\overline{Q}$
1	0	1	1	0
1	1	0	0	1
1	1	1	0	0

D Latch with control



C	D	Q	Q
0	X	Q	$\overline{Q}$
1	0	0	1
1	1	1	0

$\overline{S}\overline{R}$ Latch with control



C	R	S	Q	Q
0	X	X	Q	$\overline{Q}$
1	0	0	1	1
1	0	1	0	1
1	1	0	1	0
1	1	1	Q	$\overline{Q}$

Where we are, where we are headed

Latches are circuits that can store “state”

- set the latch to a value (0 or 1)
- put the latch in a “same value” mode to hold the value

To do complicated computations

- intermediate “state” must be maintained
- various steps of the computation must be coordinated

Q: How to coordinate computations and the changing of state values across lots of different parts of a circuit

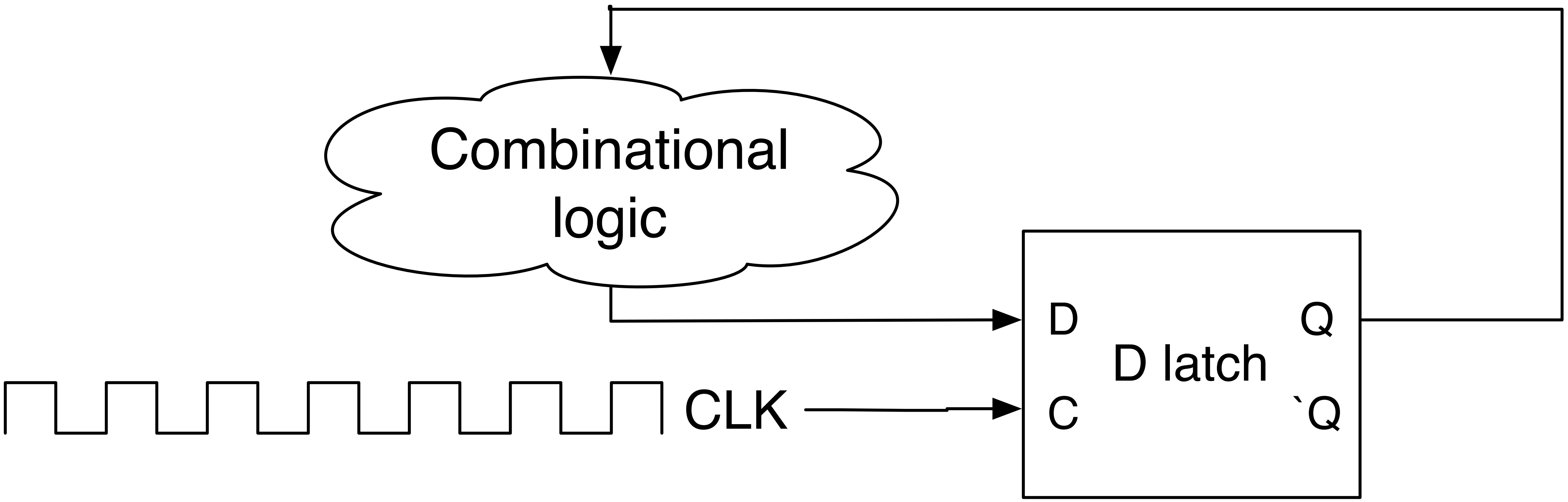
A: Introduce a **clocking** mechanism

- each clock pulse, combinational computations can be performed, results stored (in latches)

Q: How to introduce clocks into latches?

flip flops: latches on a clock

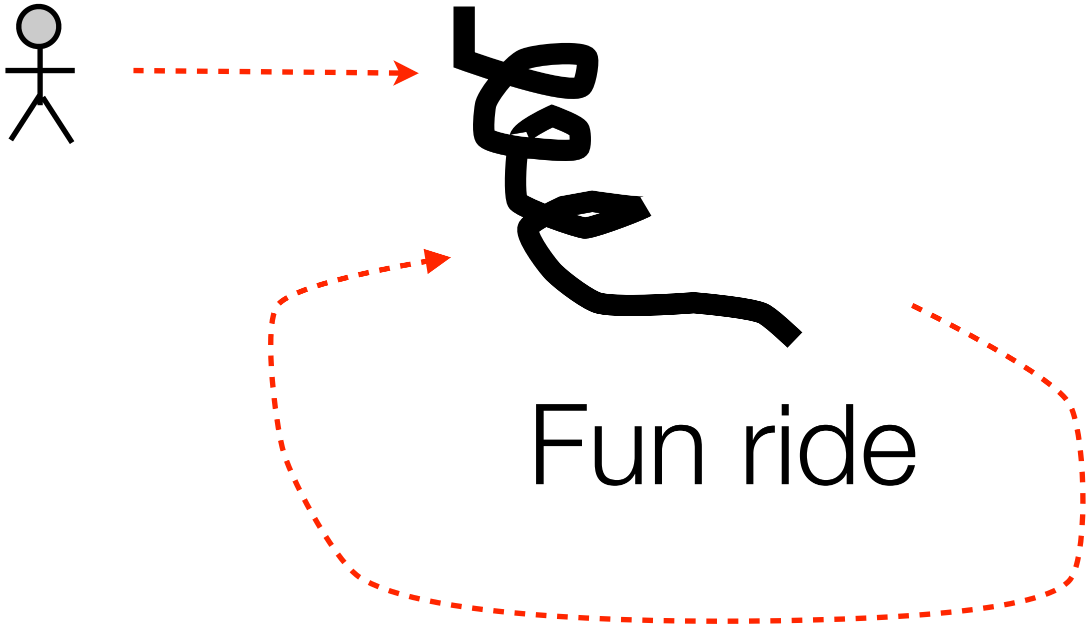
- A straightforward latch is not safely (i.e., predictably) synchronous



- The problem is transparency of latches: as soon as the input changes, at some time later the output will change

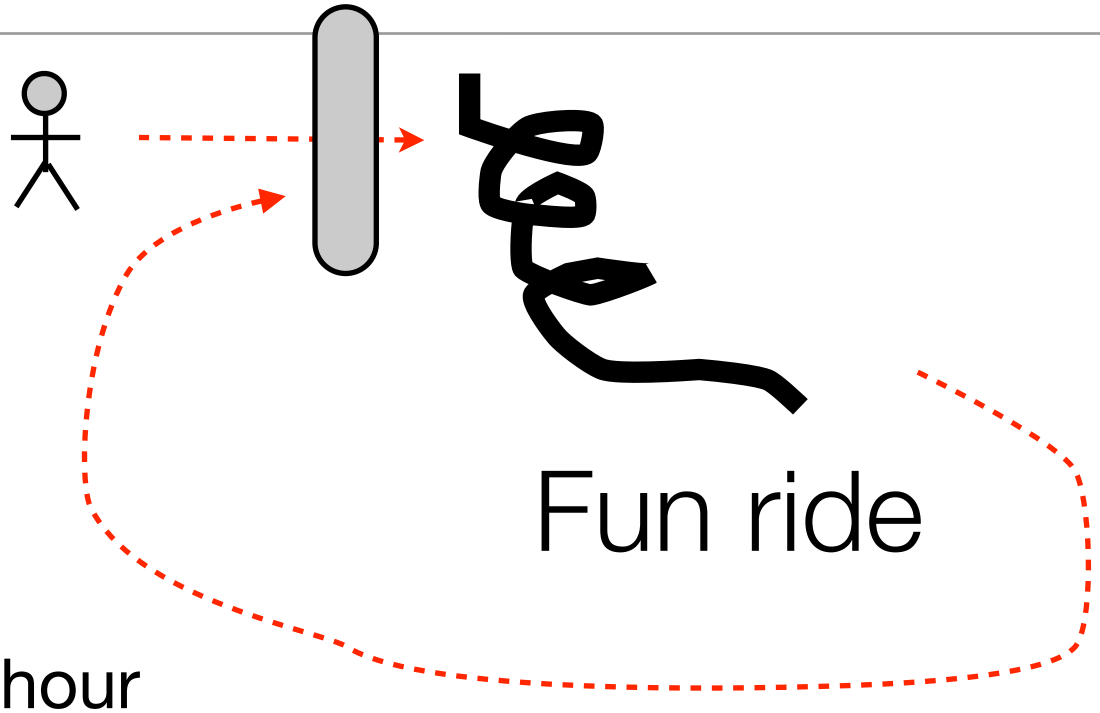
- Flip flops are designed so that outputs will **not** change within a single clock pulse

Implementing the 1-ride-per-hour



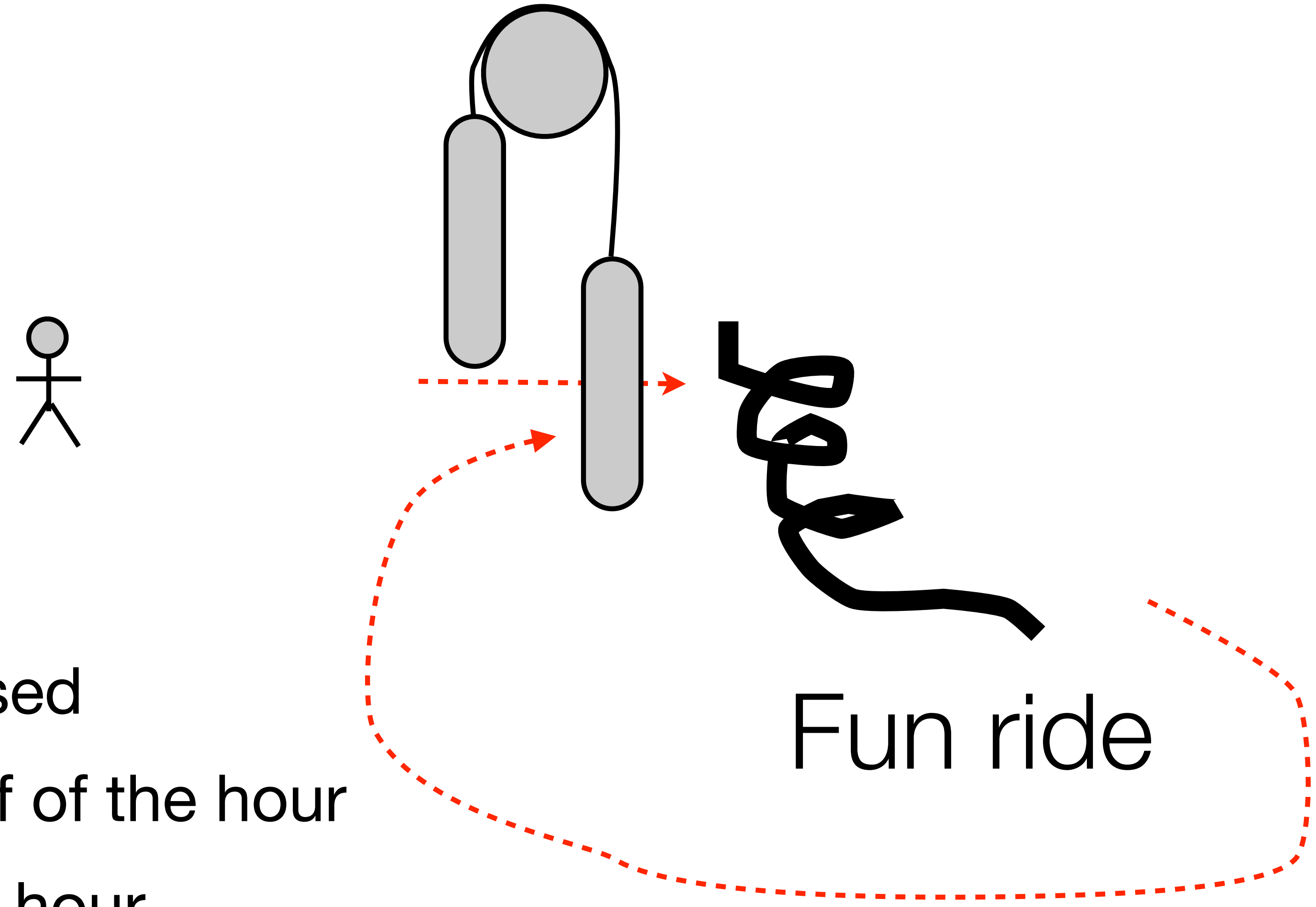
- Suppose there is
  - Fun ride
  - People should only ride at most once per hour
  - How to stop someone from riding too often?

Solution #1: Build a gate



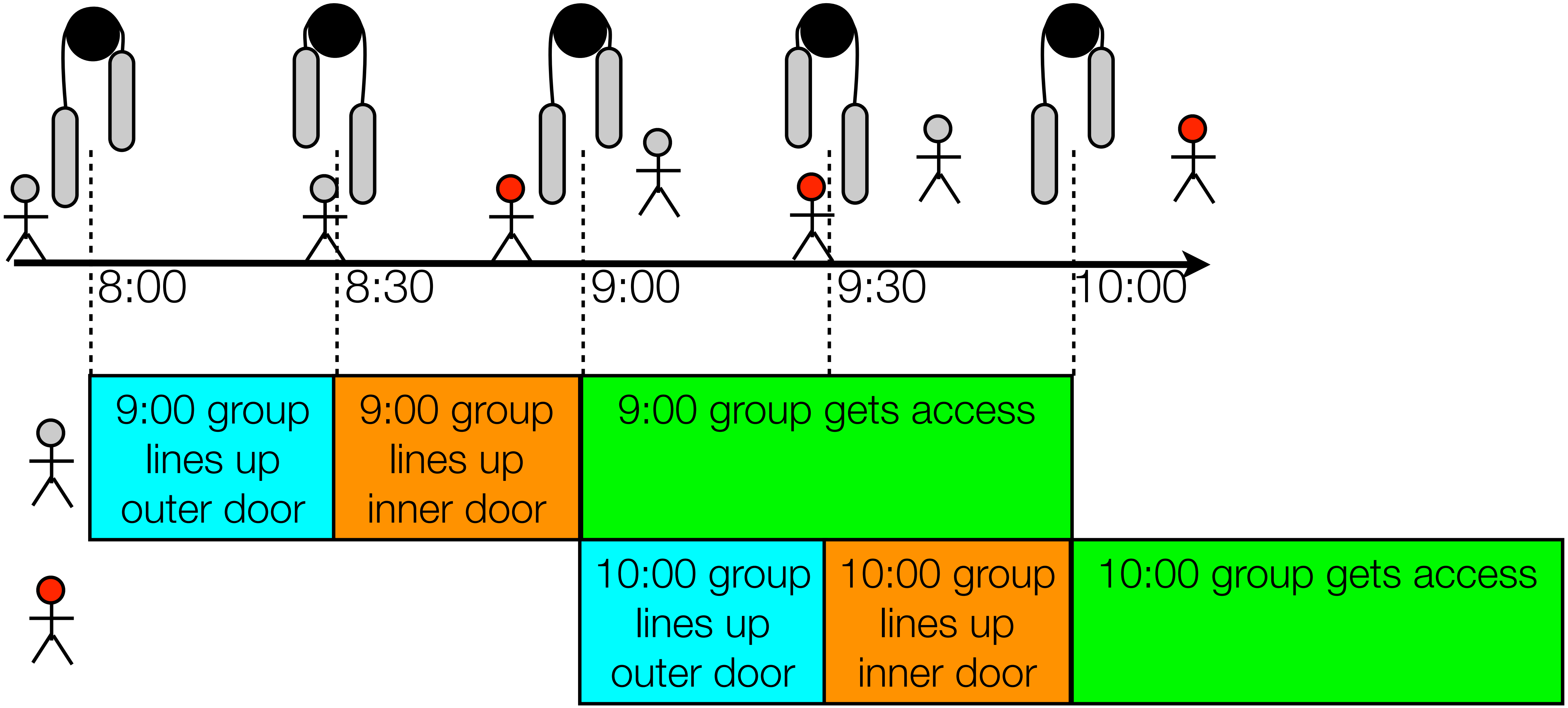
- Gate opens once per hour
  - Problem: how long to leave gate open?
  - Too short: not everyone might make it through in time (limits rideability)
  - Too long: “fast” person can go through, ride, and get through gate again

Solution #2: Pair of alternating gates



- Gates alternate being open and closed
  - 1st gate: open on the bottom half of the hour
  - 2nd gate: open on top half of the hour
  - Anyone lined up from X:00 to X:59 can ride the ride once from (X+1):00 to (X+1):59
    - X:00 - 1st gate closes, people can start waiting in front for ride
    - X:30 - 1st gate opens allowing people into middle region
    - (X+1):00 - anyone who showed up between X:00-X:59 gets through 2nd

2 Door system concluded



9:00 group lines up outer door

9:00 group lines up inner door

9:00 group gets access

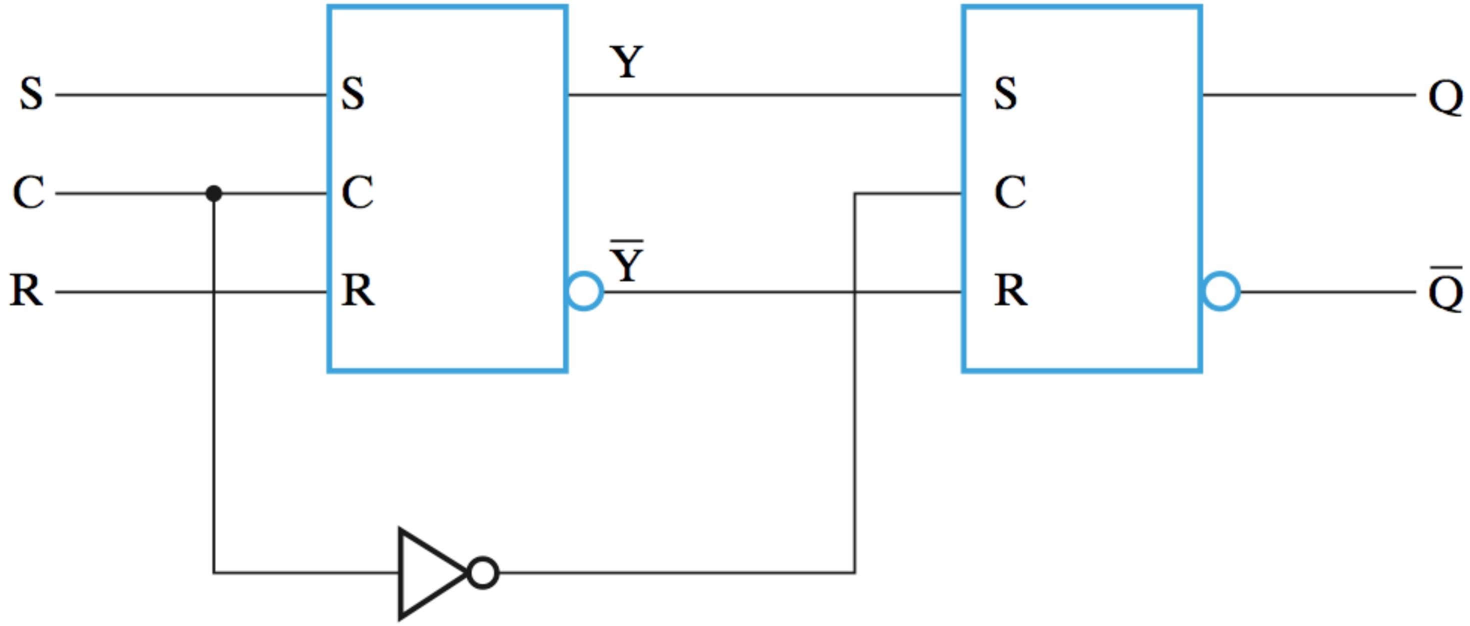
10:00 group lines up outer door

10:00 group lines up inner door

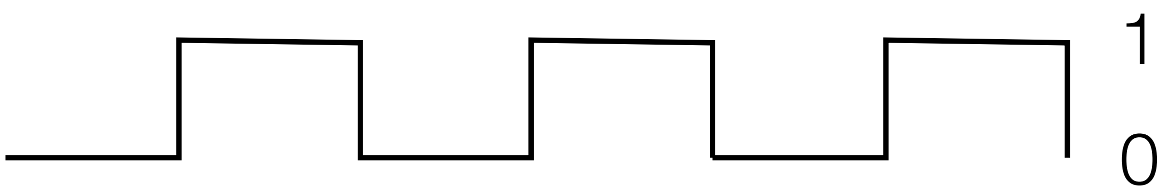
10:00 group gets access

Flip-Flop

2 SR Latches
with control



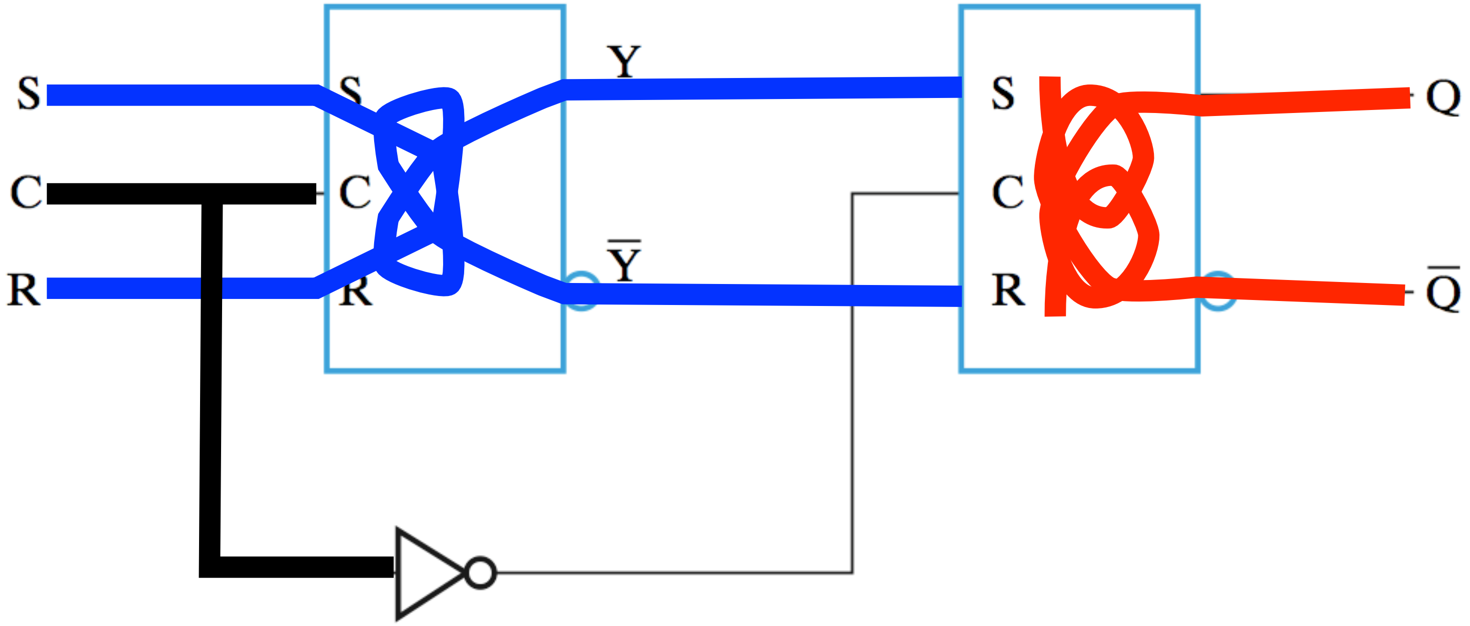
Clock:



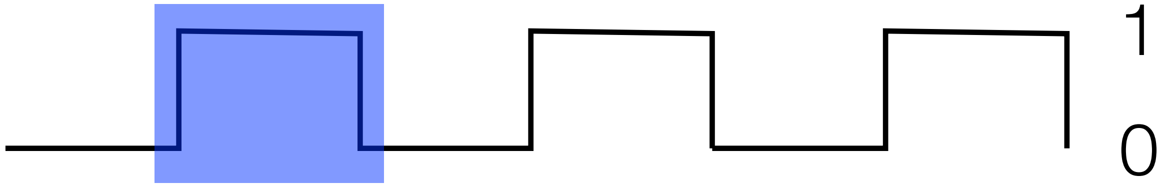
- C (Control) is fed a clock pulse (alternates between 0 and 1 with fixed period)
    - C=1: Master latch “on”, Slave latch “off”
        - New S & R inputs read into master
        - Previous Q values still emitted (not affected by new S&R inputs)
    - C=0: Master latch “off”, Slave latch “on”
        - Changing S & R inputs has no effect on Master (or Slave) latch
        - S&R inputs from last time C=1 stored safely in Master and transferred into Slave

Flip-Flop

2 SR Latches with control



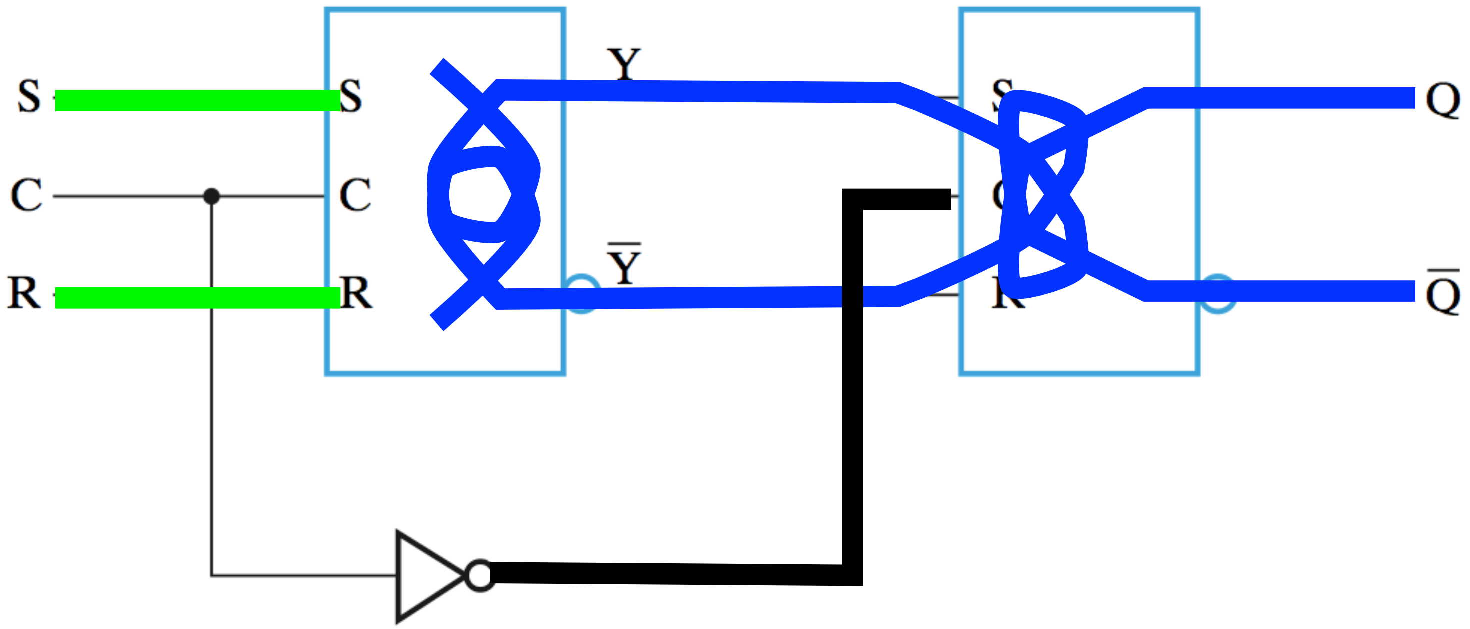
Clock:



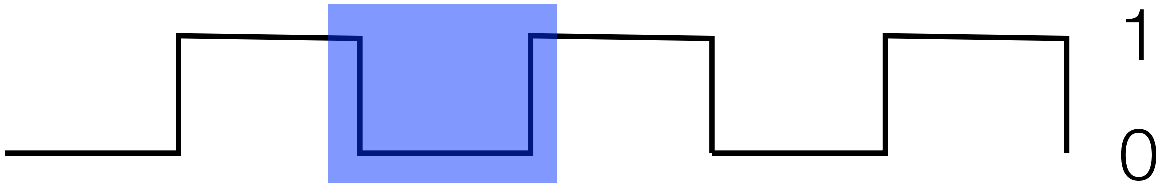
- C (Control) is fed a clock pulse (alternates between 0 and 1 with fixed period)
  - C=1: Master latch “on”, Slave latch “off”
    - New S & R inputs read into master
    - Previous Q values still emitted (not affected by new S&R inputs)
  - C=0: Master latch “off”, Slave latch “on”
    - Changing S & R inputs has no effect on Master (or Slave) latch
    - S&R inputs from last time C=1 stored safely in Master and transferred into Slave

Flip-Flop

2 SR Latches with control

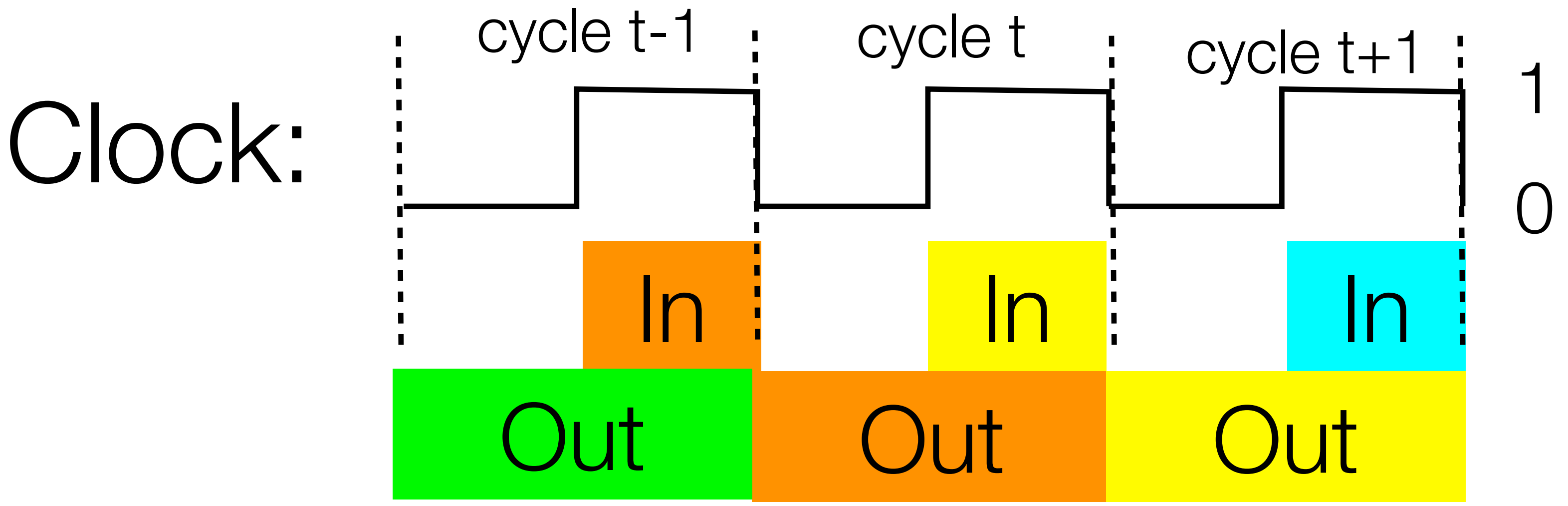


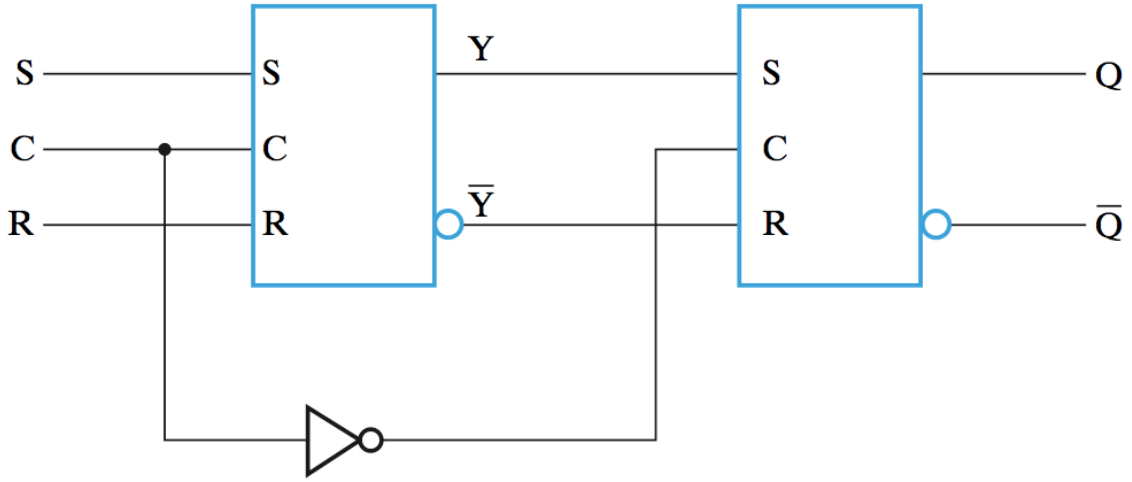
Clock:



- C (Control) is fed a clock pulse (alternates between 0 and 1 with fixed period)
    - C=1: Master latch “on”, Slave latch “off”
        - New S & R inputs read into master
        - Previous Q values still emitted (not affected by new S&R inputs)
    - C=0: Master latch “off”, Slave latch “on”
        - Changing S & R inputs has no effect on Master (or Slave) latch
        - S&R inputs from last time C=1 stored safely in Master and transferred into Slave

Flip-Flop Activation v. time



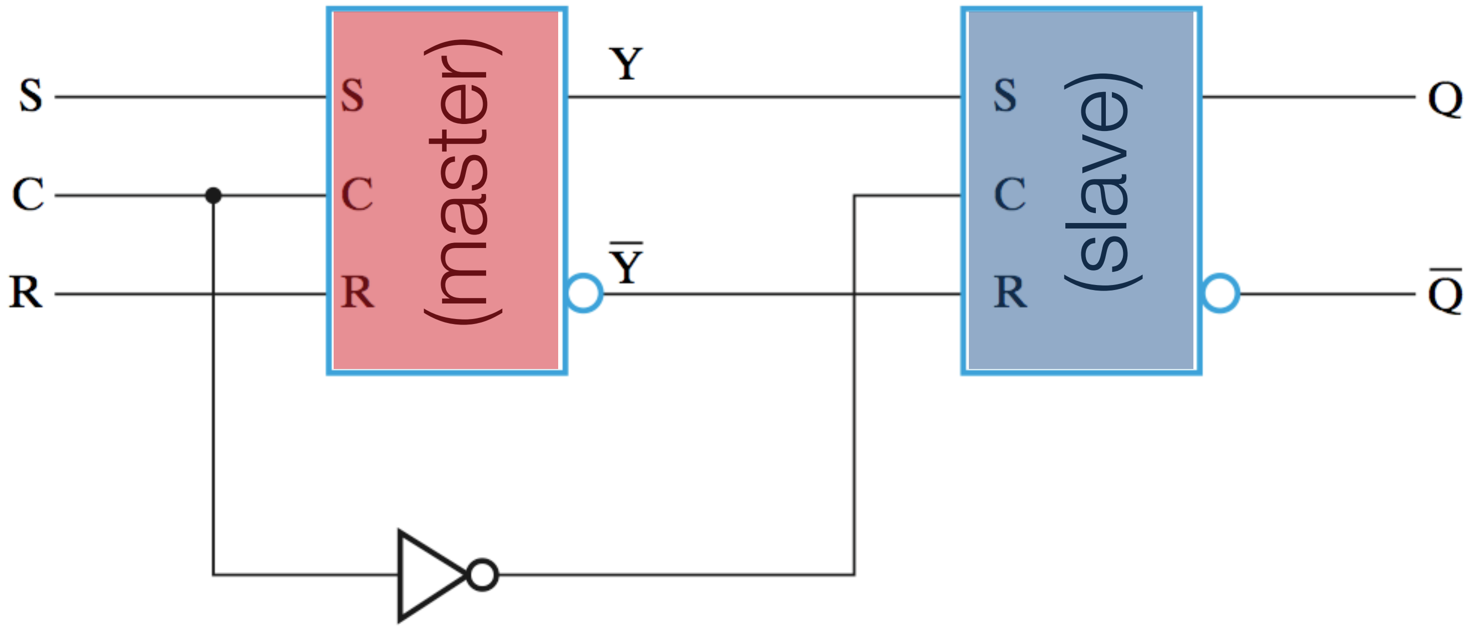


- Q(t): value output by Flip-Flop during the tth clock cycle (clock =0, then 1 during a full cycle)
- Depends on input during end of t-1st cycle

SR master-slave flip-flop

Internal state (Y) updated when CLK=1

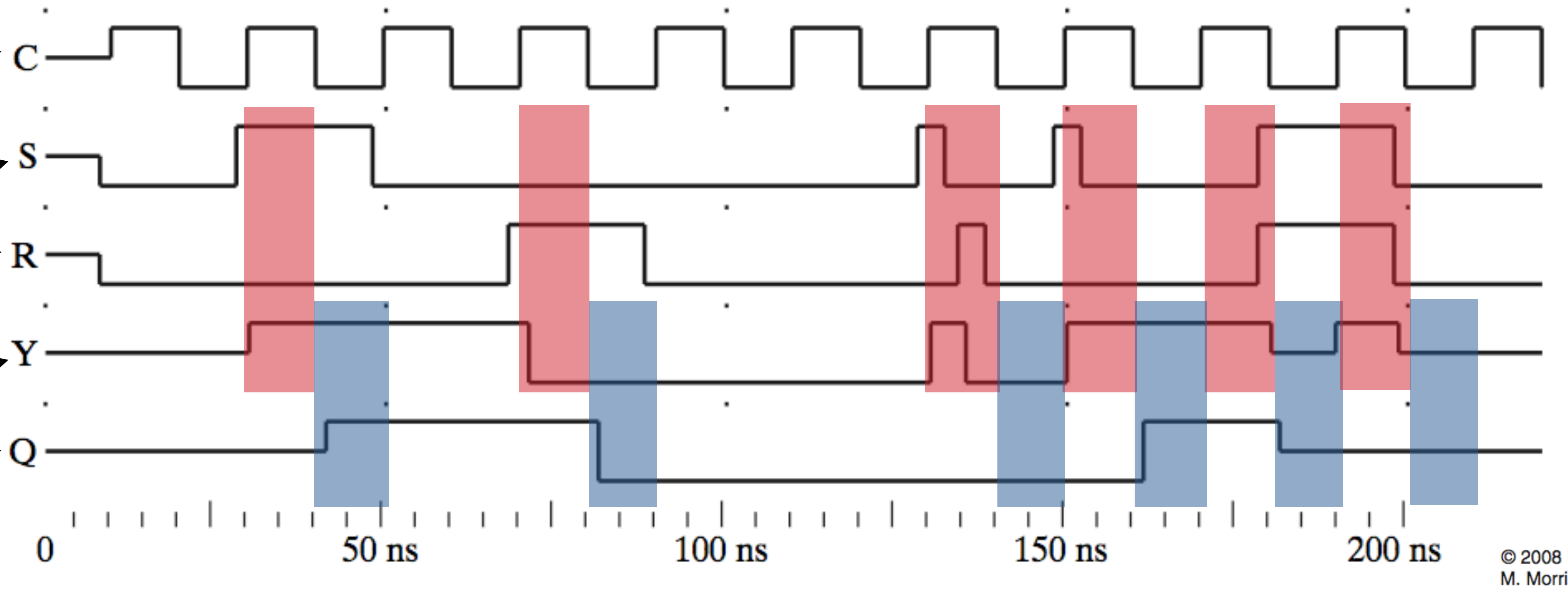
External state (Q) updated when CLK=0



Clock

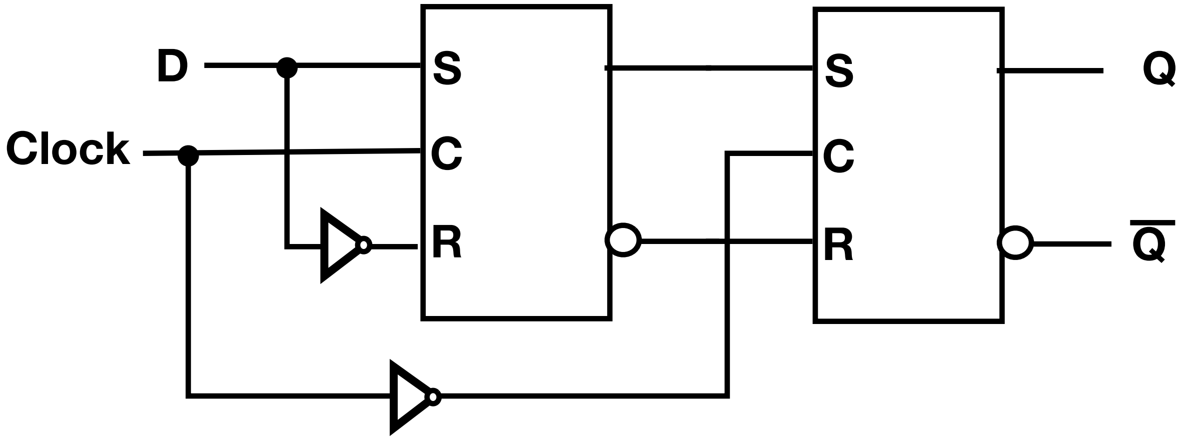
Aribtrary Inputs

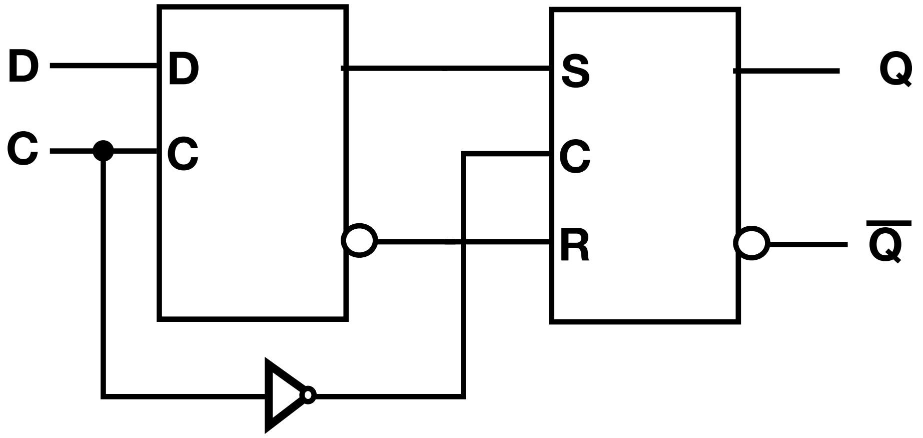
Resulting Outputs

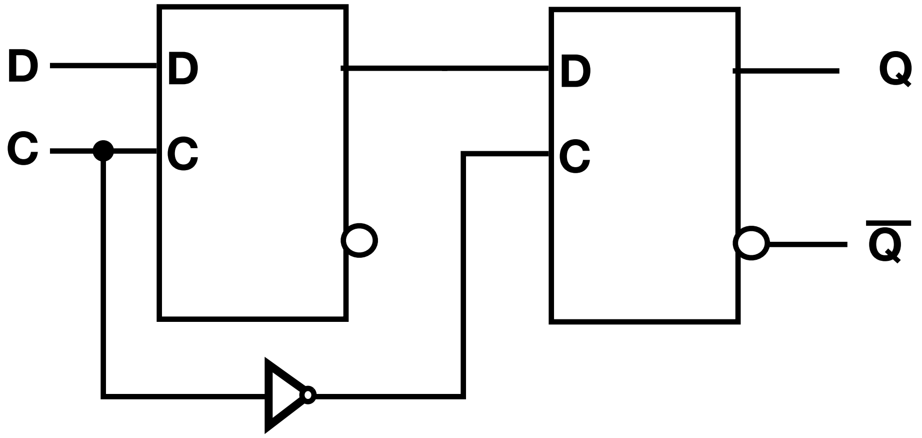


D Flip-Flop

- Can build lots of ways - here are three

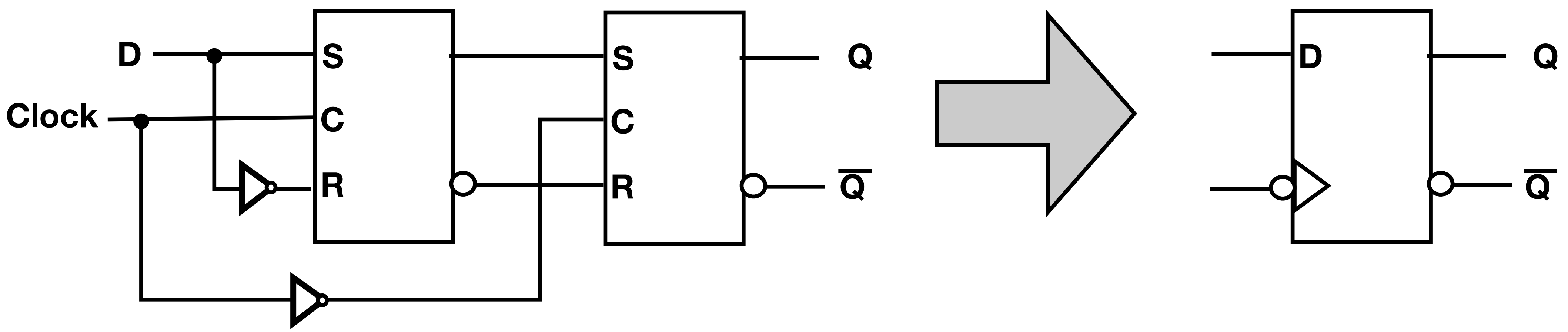




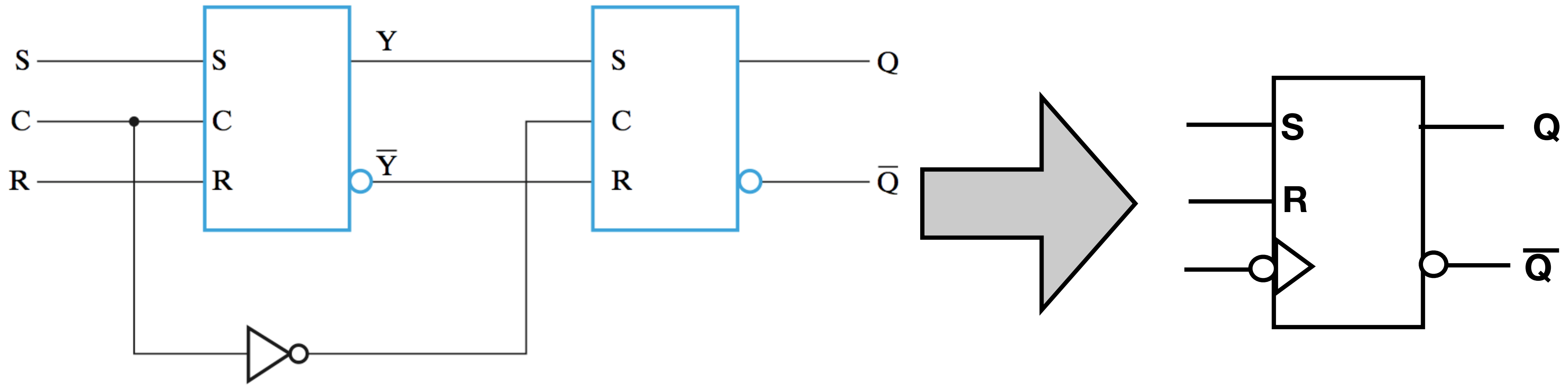


Circuit Diagram for Flip-Flops

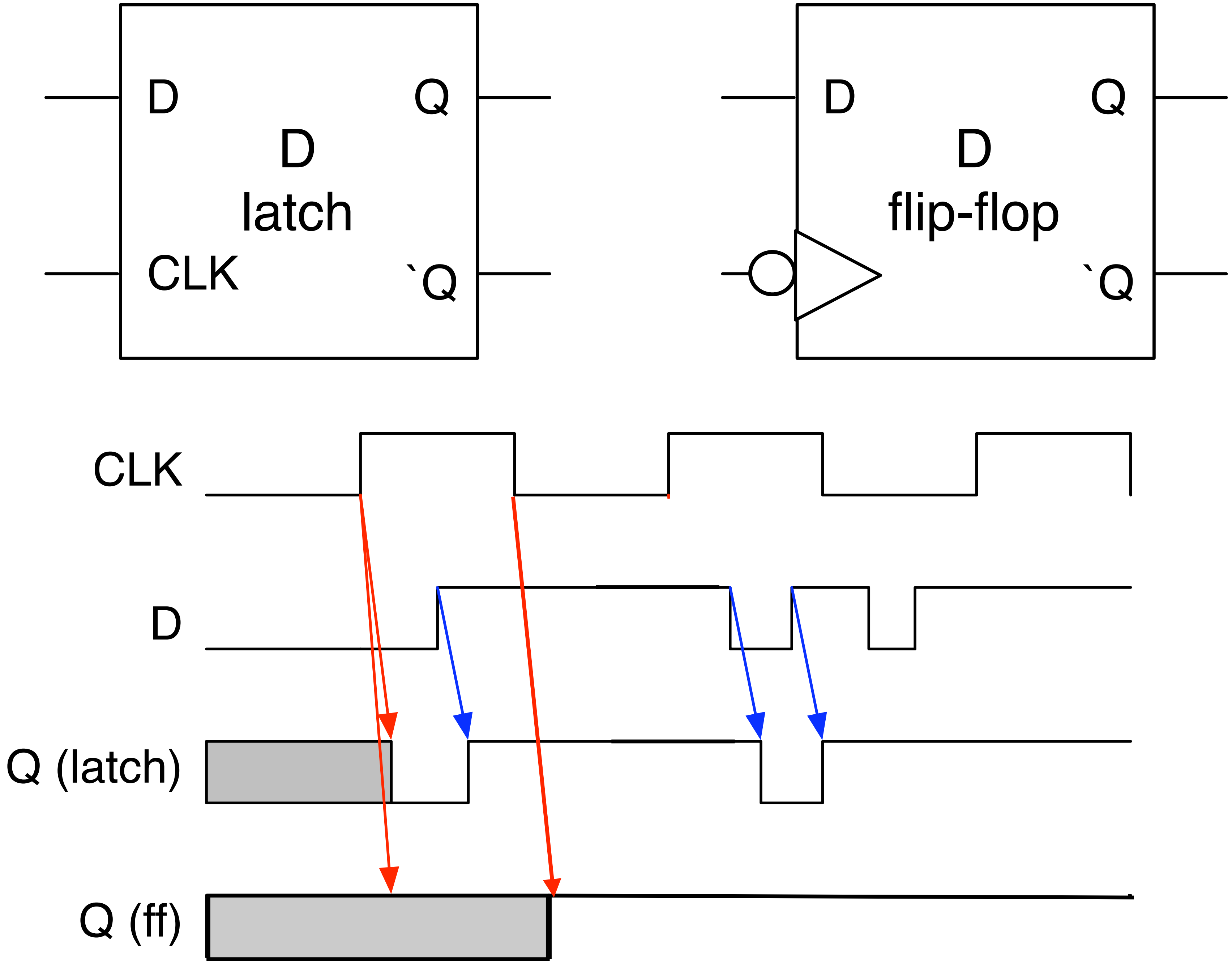
- D



- SR



D latch v. D flip-flop



Latch outputs change at any time, flip-flops only during clock transitions

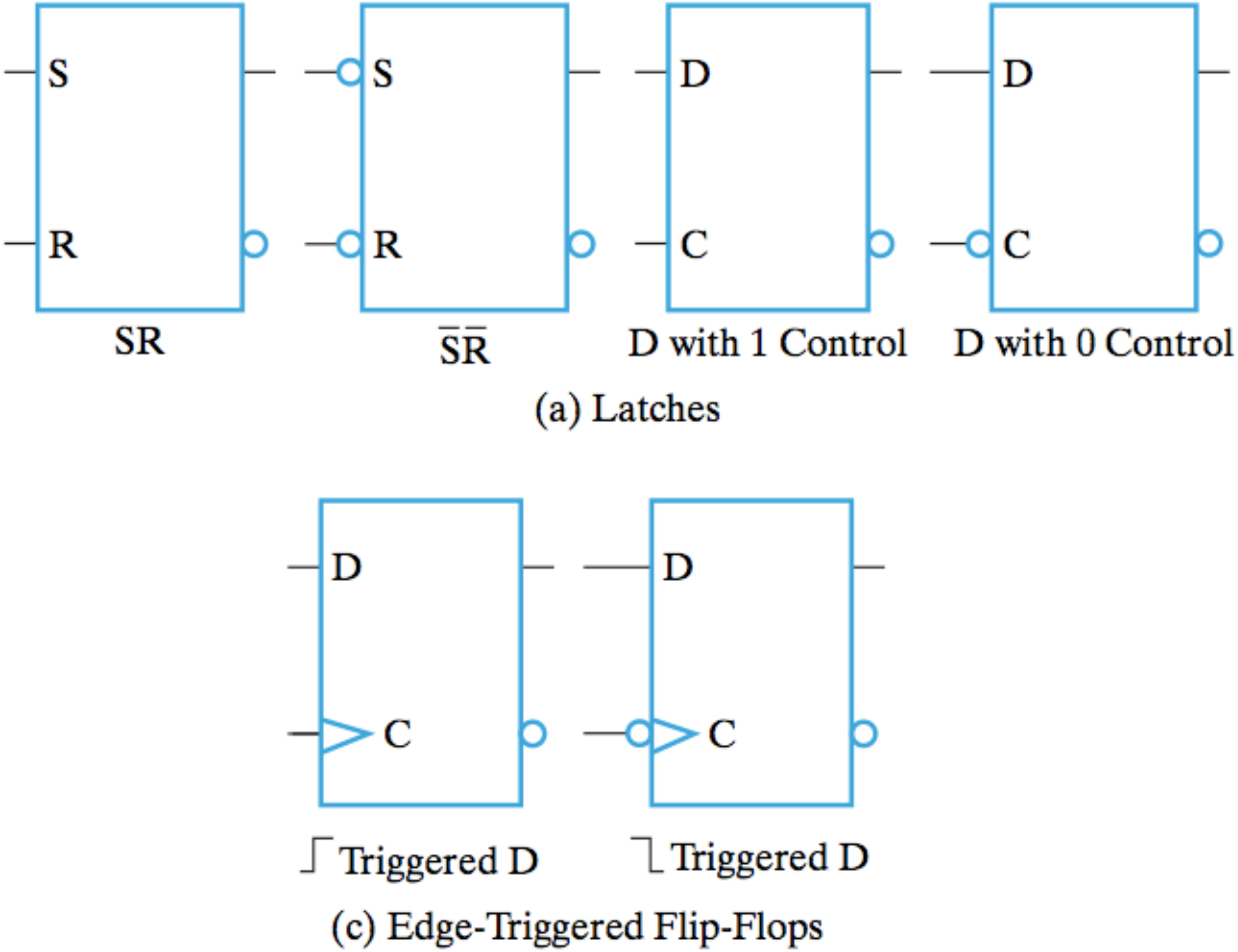
Edge v. Pulse triggered FF’s

- **Edge triggered**: the output value of the FF depends only on the inputs at the instant in time when the clock transitions in value
- **Pulse triggered**: the output value of the FF can depend on the sequence of input values during the interim of the pulse
- Positive or Negative:
    - **Positive Edge**: output value depends on the input during the 0-to-1 transition
    - **Negative Edge**: output value depends on the input during the 1-to-0 transition
    - **Positive Pulse**: Pulse Triggered and Master active when C=1
    - **Negative Pulse**: Pulse Triggered and Master active when C=0

- D FF’s are negative edge triggered (take on whatever value D is set to when clock “flops” from 1 to 0

- SR FF’s are positive pulse triggered (e.g., S=1, R=0 at start of pulse, then switch to S=0, R=0 before end).

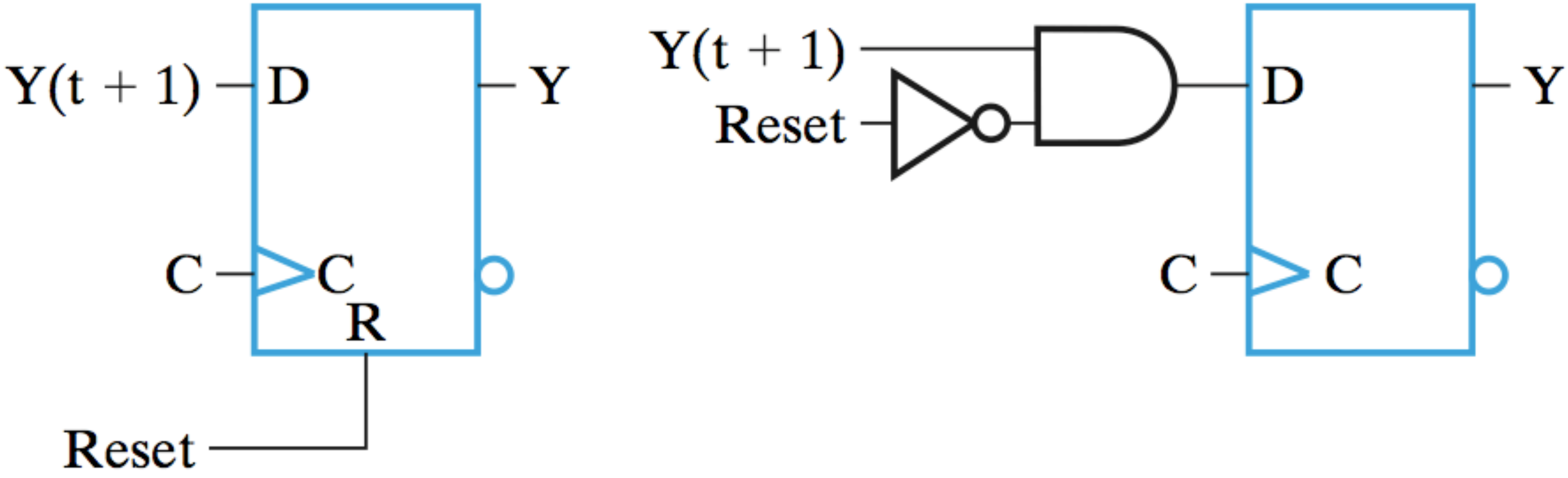
Some notes on notation



(a) Latches

(c) Edge-Triggered Flip-Flops

Adding reset signals



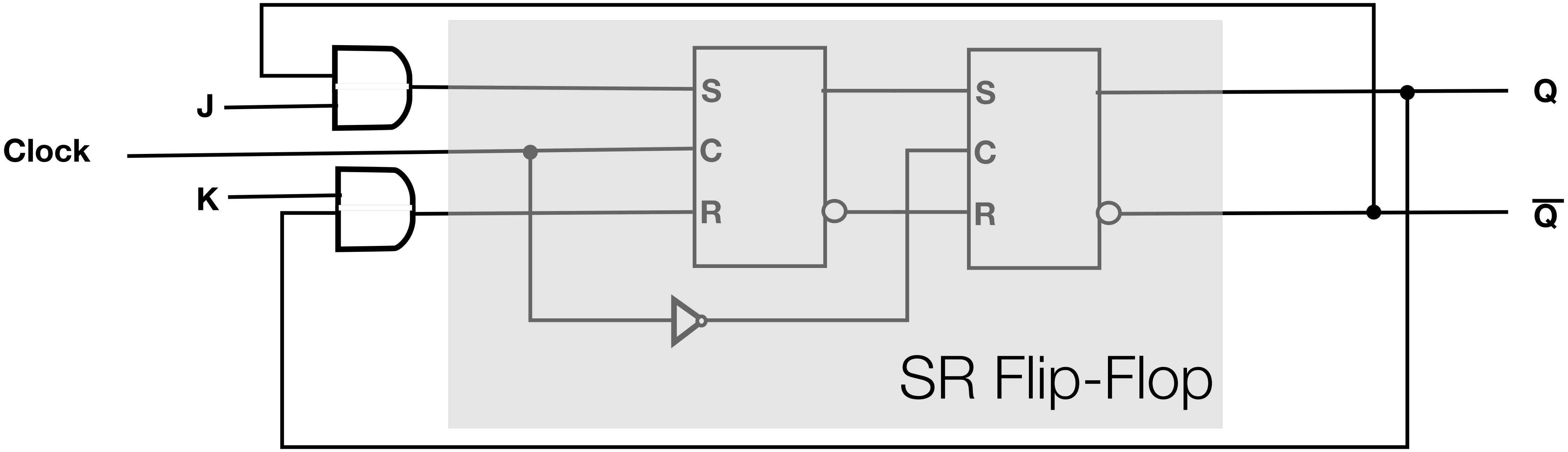
(a) Asynchronous Reset

(b) Synchronous Reset

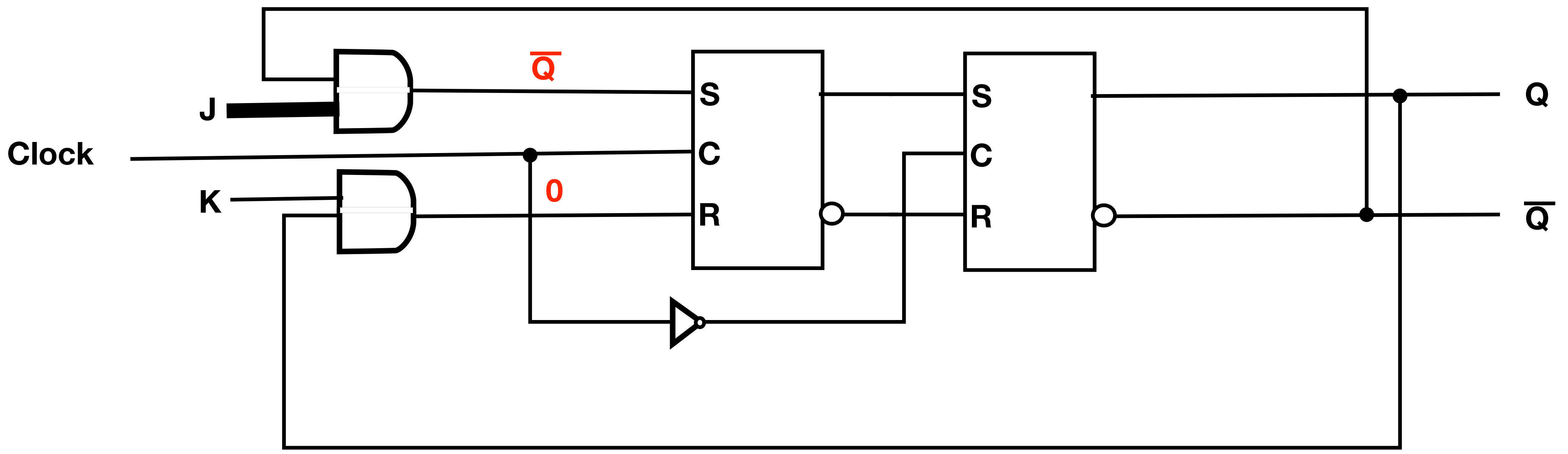
(resets immediately)

(resets at clock edge only)

JK Flip Flop from SR Flip Flop

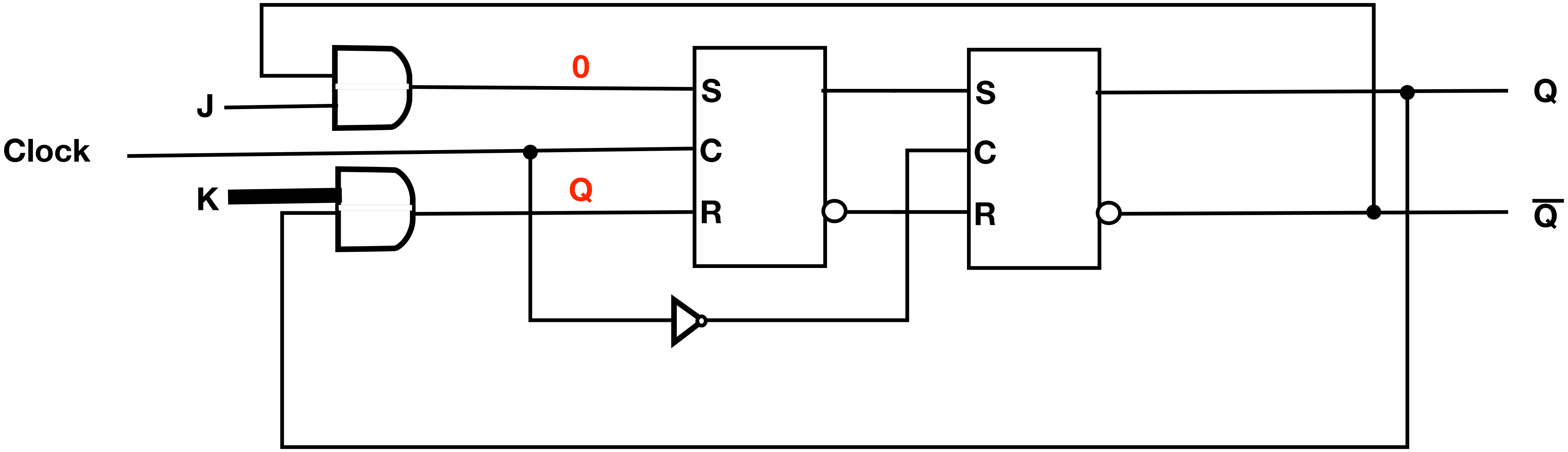


JK Flip Flop from SR Flip Flop



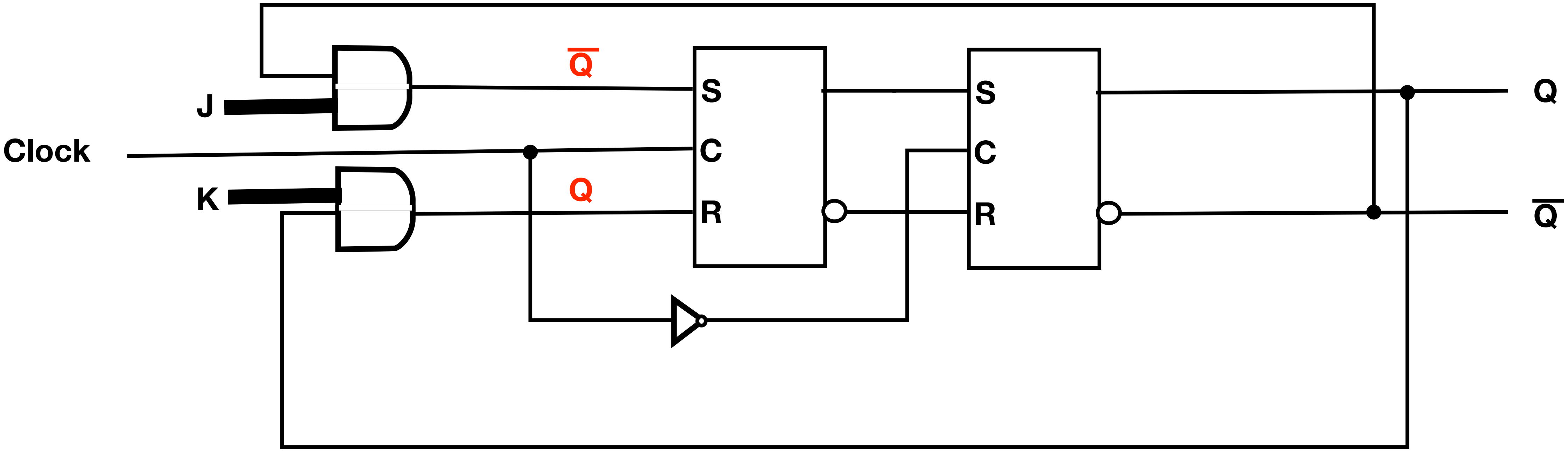
- J=1, K=0
  - Q(t-1)=1, $\overline{Q}$(t-1)=0 SR F.F. fed S=0, R=0, stays the same: Q(t)=1
  - Q(t-1)=0, $\overline{Q}$(t-1)=1, SR F.F. fed S=1, R=0, set: Q(t)=1
  - So regardless of Q(t-1) value, J=1, K=0 **sets** the JK F.F.: Q(t) = 1

JK Flip Flop from SR Flip Flop



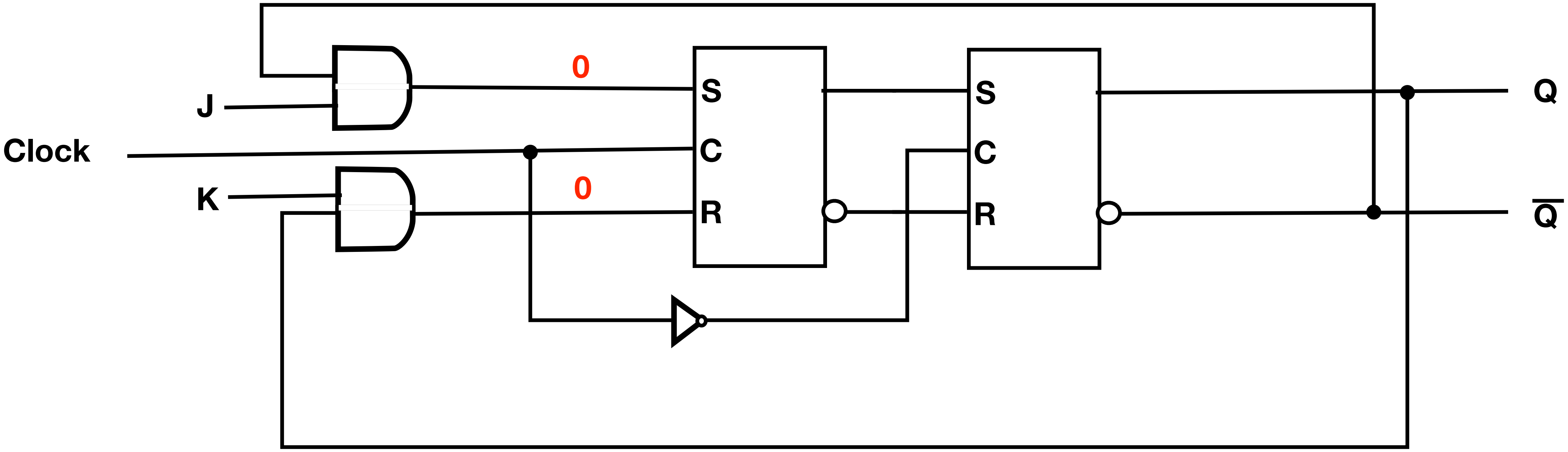
- J=0, K=1
  - Q(t-1)=1, SR F.F. fed S=0, R=1, reset: Q(t)=0
  - Q(t-1)=0, SR F.F. fed S=0, R=0, stay same: Q(t) = 0
  - So regardless of Q(t-1) value, J=0, K=1 **sets** the JK F.F.: Q(t) = 0

JK Flip Flop from SR Flip Flop



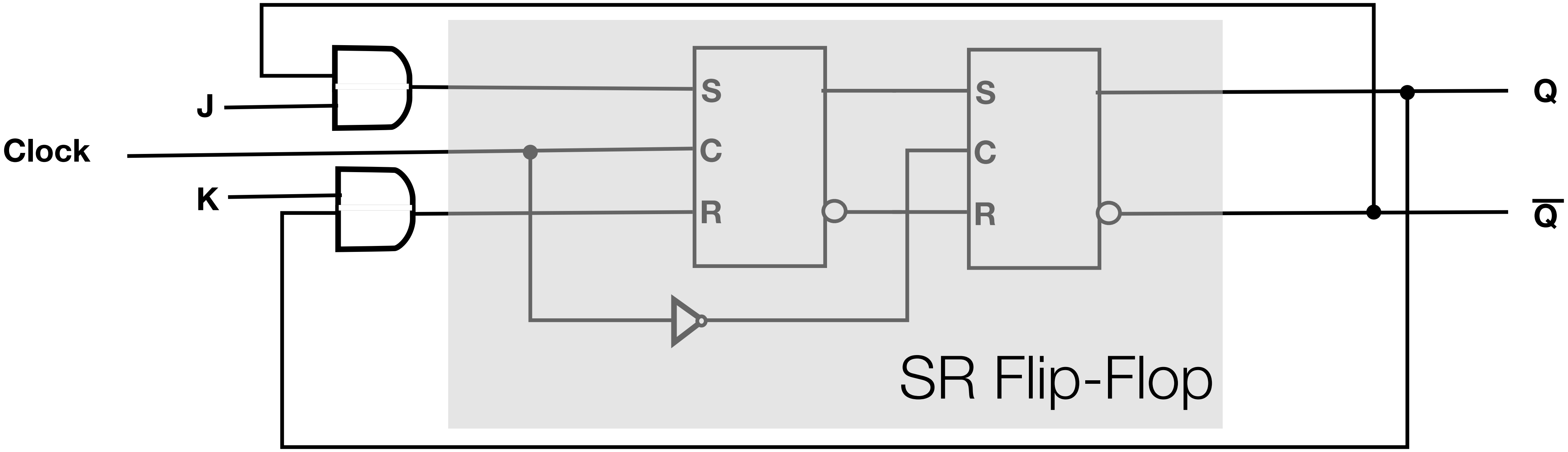
- J=1, K=1
  - Q(t-1)=1, $\overline{Q}$(t-1)=0 SR F.F. fed S=0, R=1, reset: Q(t)=0
  - Q(t-1)=0, $\overline{Q}$(t-1)=1, SR F.F. fed S=1, R=0, set: Q(t)=1
  - So J=1, K=1 **compliments** the JK F.F.: Q(t) = $\overline{Q}$(t-1)

JK Flip Flop from SR Flip Flop



- J=0, K=0
  - S=0, R=0, regardless of J,K values, reset: Q(t)=Q(t-1)
  - J=0, K=0, F.F. **stays same**

JK Flip Flop from SR Flip Flop

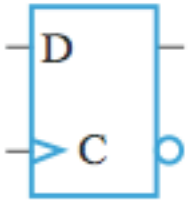
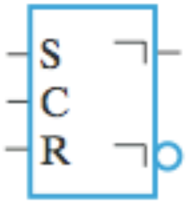
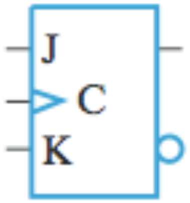
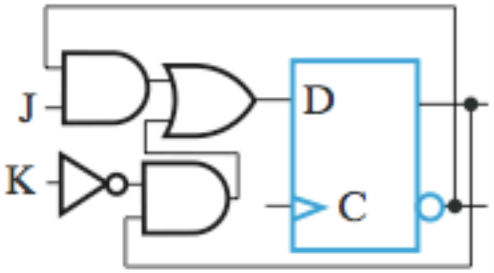
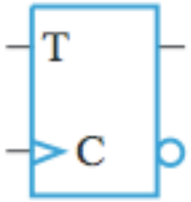
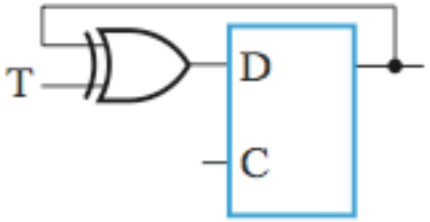


JK Flip-Flop Characteristic Table

J	K	Q(t+1)
0	0	Q(t)
0	1	0
1	0	1
1	1	$\overline{Q}$(t)

Q: Edge or pulse triggered?

Summary + T Flip Flop

Type	Symbol	Logic Diagrams	Characteristic Table				Characteristic Equation	Excitation Table				
D		See Figure 5-12	D		Q(t+1)	Operation	$Q(t+1) = D(t)$		Q(t+1)	D		Operation
			0		0	Reset			0	0		Reset
			1		1	Set			1	1		Set
SR		See Figure 5-9	S	R	Q(t+1)	Operation	$Q(t+1) = S(t) + \bar{R}(t)\,Q(t)$	Q(t)	Q(t+1)	S	R	Operation
			0	0	$Q(t)$	No change		0	0	0	X	No change
			0	1	0	Reset		0	1	1	0	Set
			1	0	1	Set		1	0	0	1	Reset
			1	1	?	Undefined		1	1	X	0	No change
JK			J	K	Q(t+1)	Operation	$Q(t+1) = J(t)\,\bar{Q}(t) + \bar{K}(t)\,Q(t)$	Q(t)	Q(t +1)	J	K	Operation
			0	0	$Q(t)$	No change		0	0	0	X	No change
			0	1	0	Reset		0	1	1	X	Set
			1	0	1	Set		1	0	X	1	Reset
			1	1	$\bar{Q}(t)$	Complement		1	1	X	0	No Change
T				T	Q(t+1)	Operation	$Q(t+1) = T(t) \oplus Q(t)$		Q(t +1)	T		Operation
				0	$Q(t)$	No change			$Q(t)$	0		No change
				1	$\bar{Q}(t)$	Complement			$\bar{Q}(t)$	1		Complement