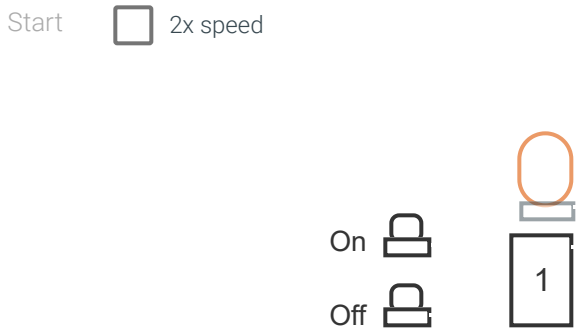
7.6 SR latches

Storing a bit

A ***sequential circuit***'s output is dependent on the present and the past *sequence* of input values, which necessarily means t at least one bit. In contrast, a ***combinational*** circuit's output is dependent only on the present *combination* of input values.

The following system stores a bit to control a lamp.

PARTICIPATION ACTIVITY 7.6.1: A lamp controlled by on and off buttons stores a bit to remember whether keep the lamp on or off.



PARTICIPATION ACTIVITY 7.6.2: Lamp that stores a bit.

Consider the example above.

1) When the On button is pressed, what value should be stored in the lamp?
   - ○ 0
   - ○ 1

2) When the On button is pressed and then released, what value should be stored in the lamp?
   - ○ 0
   - ○ 1

3) Suppose a user pressed On and released, then pressed Off and released. What should be stored in the lamp?
   - ○ 0
   - ○ 1

4) Without knowing any history of the button presses, a designer observes that neither button is currently being pressed. Does the designer know whether the lamp should be illuminated?
   - ○ Yes
   - ○ No

SR latch

The simplest circuit for storing a bit is called a ***latch***. An ***SR latch*** stores one bit, with an input s to set the latch to 1, an inpu latch to 0, and with the stored bit appearing on output q. S and s are for "set", and R and r for "reset".

Below is a circuit for an SR latch. Reminder: NOR outputs 0 if any input is 1.

PARTICIPATION ACTIVITY 7.6.3: An SR latch stores a bit.

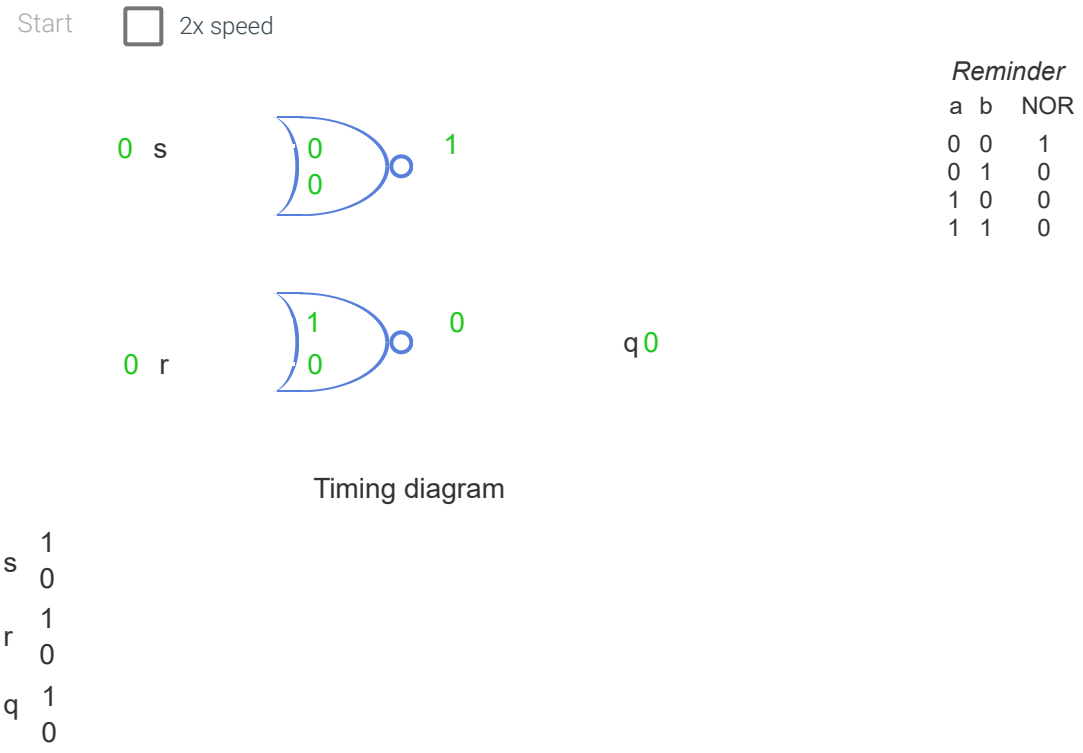


Figure 7.6.1: SR latch behavior.

s	r	q
0	0	Previously-stored bit
0	1	0 ("Reset")
1	0	1 ("Set")
1	1	Unknown

PARTICIPATION ACTIVITY 7.6.4: SR latch.

Indicate q's present value for the given input sequence. "s: 0..1" means s was 0 and is presently 1.

1) s: 0
r: 1
   - ○ 1
   - ○ 0

2) s: 0..0
r: 1..0
   - ○ 1
   - ○ 0

3) s: 0..0..1
r: 1..0..0
   - ○ 1
   - ○ 0

4) s: 0..0..1..0
r: 1..0..0..0

- 1
- 0

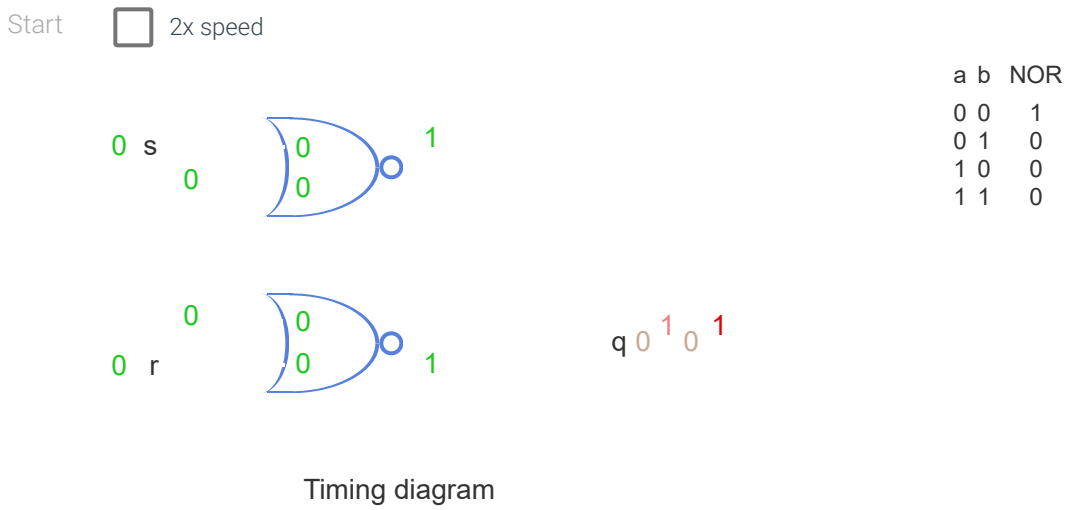
5) s: 0..0..1..0..1
r: 1..0..0..0..0

- 1
- 0

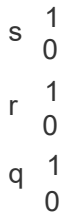
SR = 11 is problematic

s = 1 and r = 1 causes a problem. The 1's cause the NOR gates to output 0's. When s and r return to 0's, then both gates ou 1's propagate back to the gate inputs, causing the gates to output 0's. Those 0's propagate back, causing 1's again. No bit v instead the latch oscillates. ***Oscillate*** means to change from 0 to 1 to 0 to 1 repeatedly. Due to different gate and wire delay latch will settle into a stored 0 or 1, but which one is unknown.

PARTICIPATION ACTIVITY 7.6.5: SR latch problem: sr = 11 doesn't store a bit and will cause oscillation.



Timing diagram



PARTICIPATION ACTIVITY 7.6.6: SR latch when s = 1, r = 1.

1) Setting s and r to 1's simultaneously initially sets q to 1.
   - ○ True
   - ○ False

2) q oscillates while s and r are both 1's.
   - ○ True
   - ○ False

3) If s and r are both 1's, and then both change to 0's, q may oscillate.
   - ○ True
   - ○ False

SR latches are uncommon

SR latches were previously common when gates were expensive. But with gates far

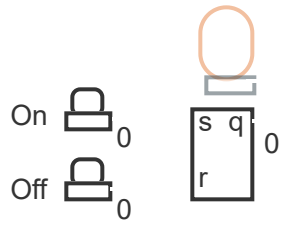
cheaper (and smaller) today, the more robust D latch (discussed later), which extends an SR latch, is more common.

Example: Lamp with on/off buttons

The earlier lamp example can be implemented using an SR latch.

PARTICIPATION ACTIVITY 7.6.7: Lamp with on/off buttons implemented using an SR latch.

Start 2x speed



PARTICIPATION ACTIVITY 7.6.8: Lamp implemented using an SR latch.

Consider the example above.

1) While the On button is being pressed, does the lamp illuminate?
   - Yes
   - No

- Oscillates

2) If the On button is pressed and then released, does the lamp stay illuminated?
   - Yes
   - No
   - Oscillates

3) If the user presses both On and Off buttons and then releases both of them, does the lamp illuminate?
   - Yes
   - No
   - May oscillate

Provide feedback on this section