(*) Building an 8-bit Breadboard Computer

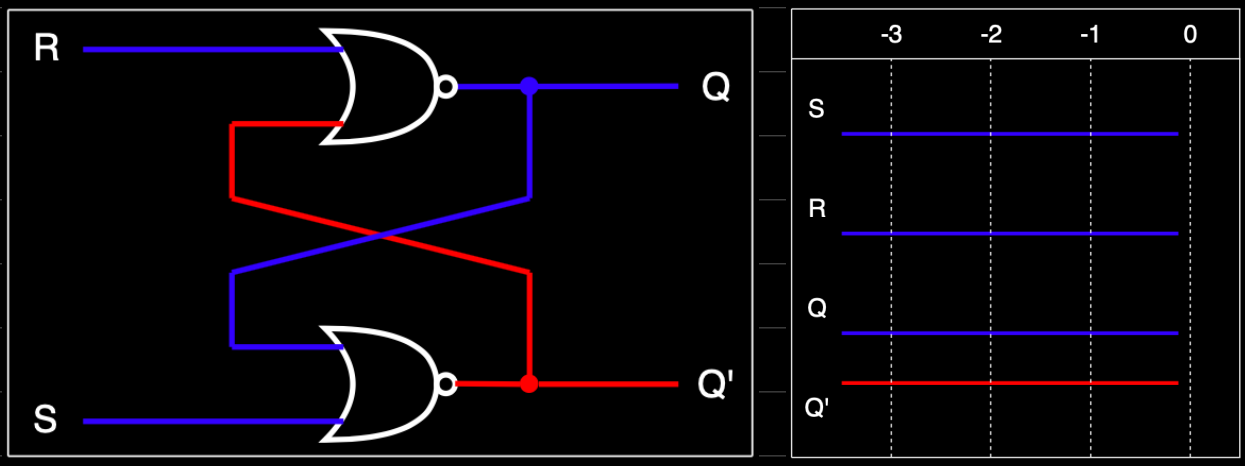
by Ben Eater

(https://youtube.com/playlist?list=PLowKtXNTBypGqImE405J2565dvjafglHU)

^ SR Latch

(https://youtu.be/KM0DdEaY5sY)

Let the initial STABLE state be S = 0, R = 0, Q = 0 and Q' = 1 (CLEAR).



Let the PROPAGATION delays of the upper and lower NOR gates be 2 units and 3 units, respectively.

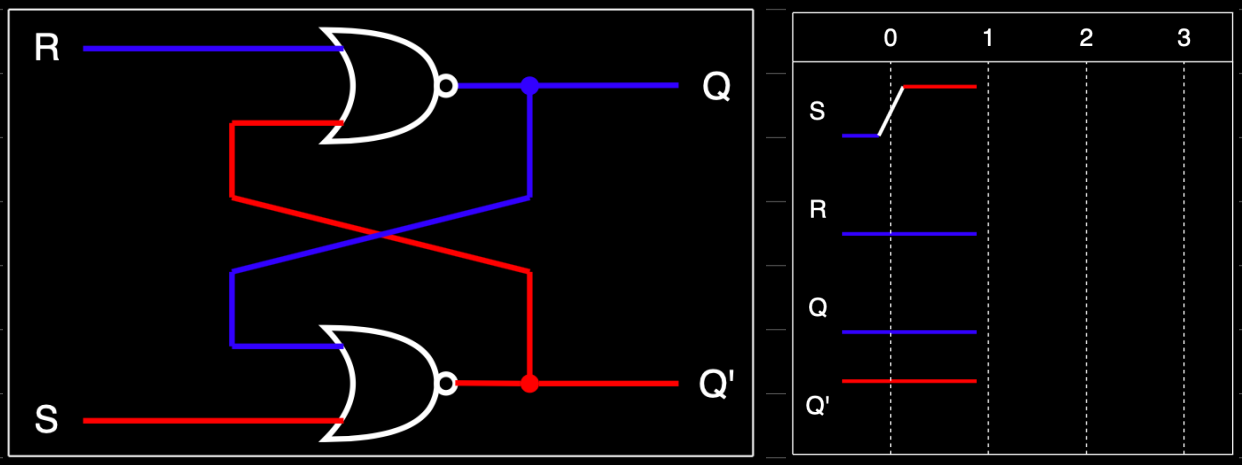
So, the EFFECTS of the changed inputs will be visible only AFTER the propagation delays.

In order to draw the TIMING diagrams, for every logic gate, look at its inputs at time (t - T), where T is its propagation delay.

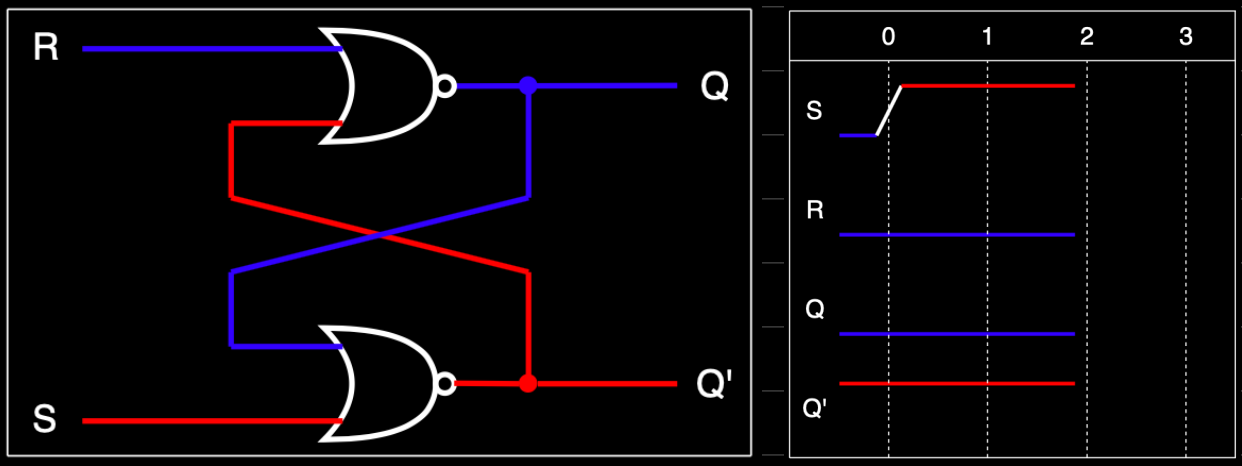
For eg., if the propagation delay of a logic gate is 3 units, then its OUTPUT at t = 3.5 units (say) depends upon its INPUTS at t = 0.5 units.

Now, let S get changed to 1 at t = 0 (in order for the latch to become SET).

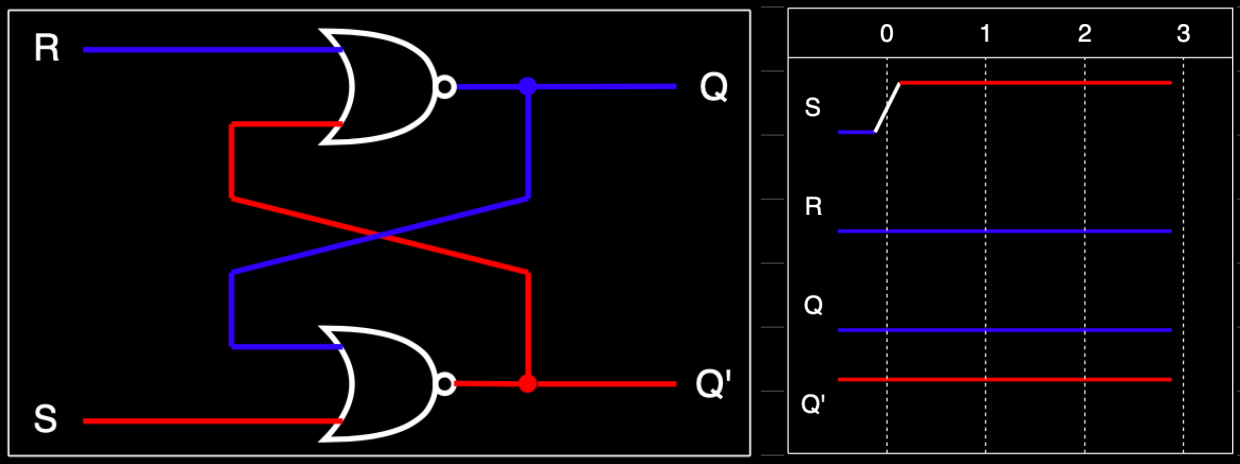
From t = 0 to t = 1 :-



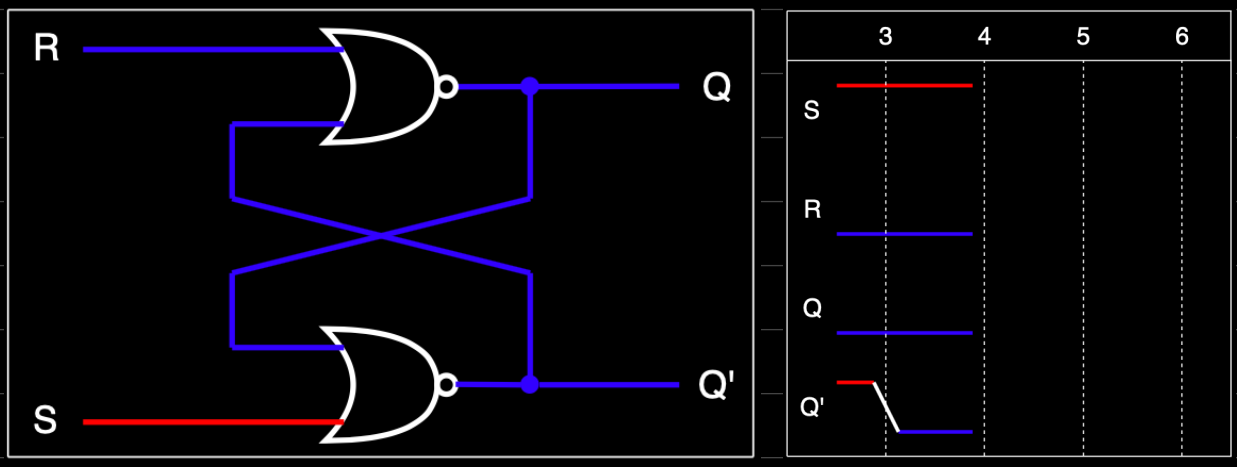
From t = 1 to t = 2 :-



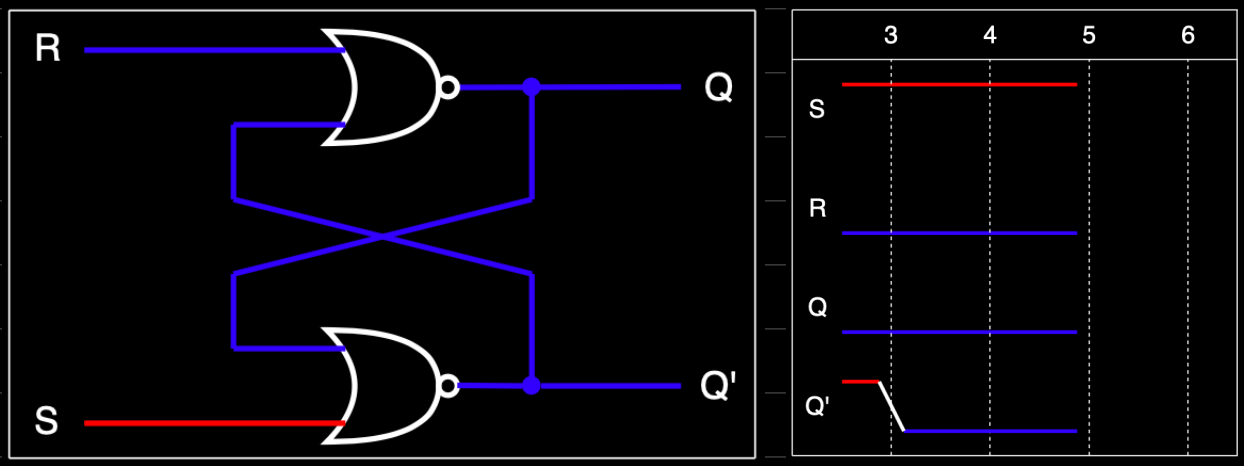
From t = 2 to t = 3 :-



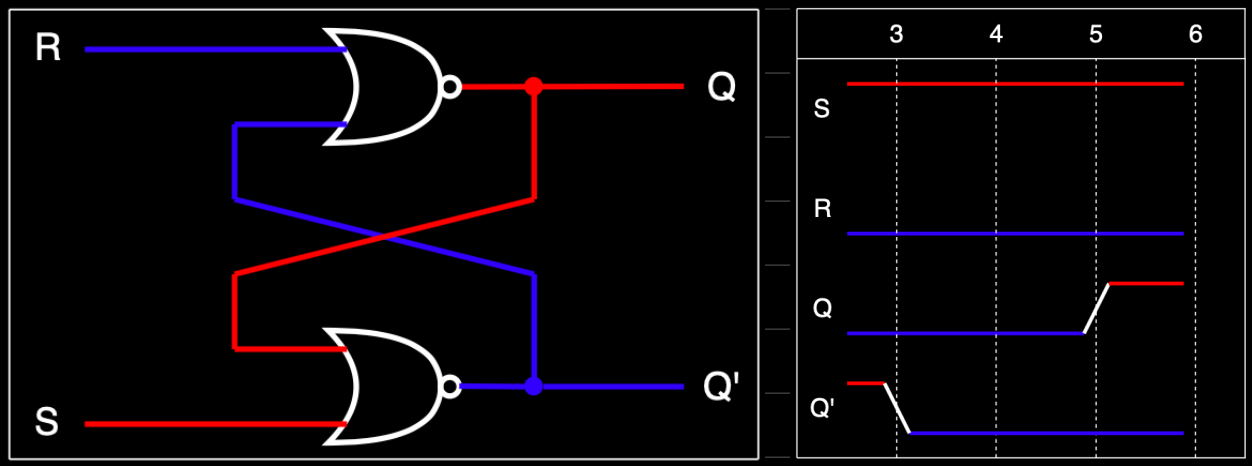
From t = 3 to t = 4 :-



From t = 4 to t = 5 :-



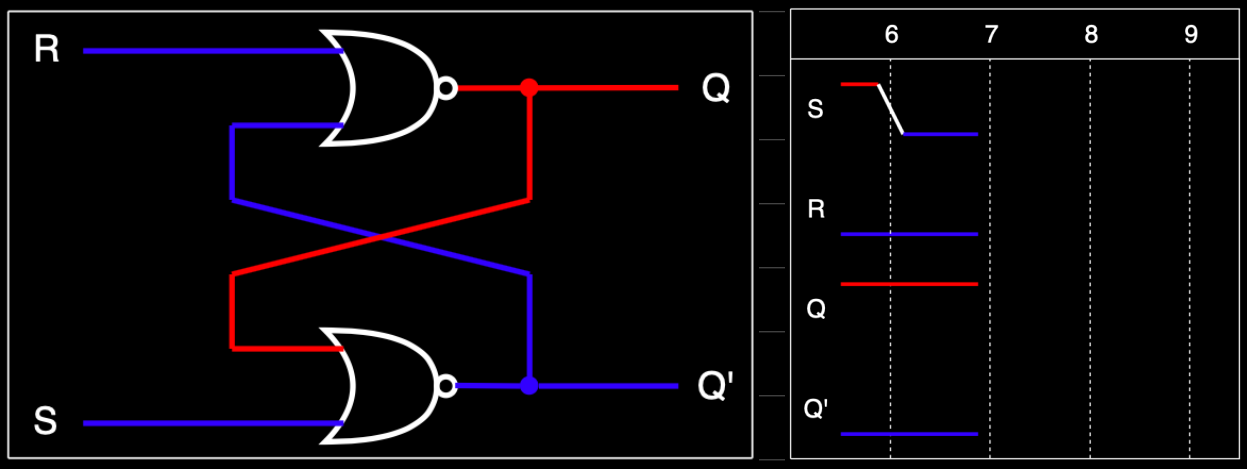
From t = 5 to t = 6 :-



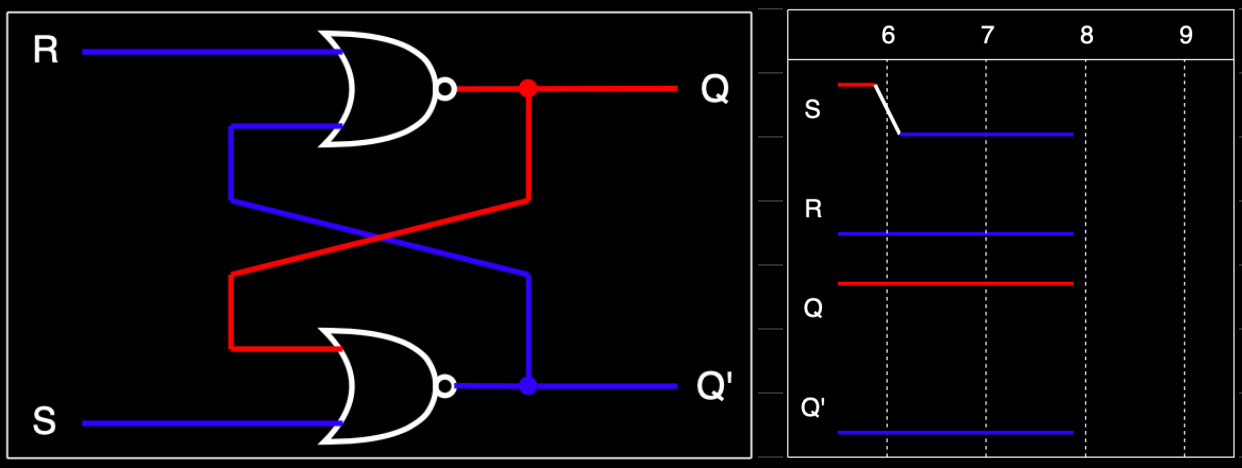
Now, the latch will stay STABLE in the SET state, even if S gets changed back to 0.

Let S get changed back to 0 at t = 6.

From t = 6 to t = 7 :-



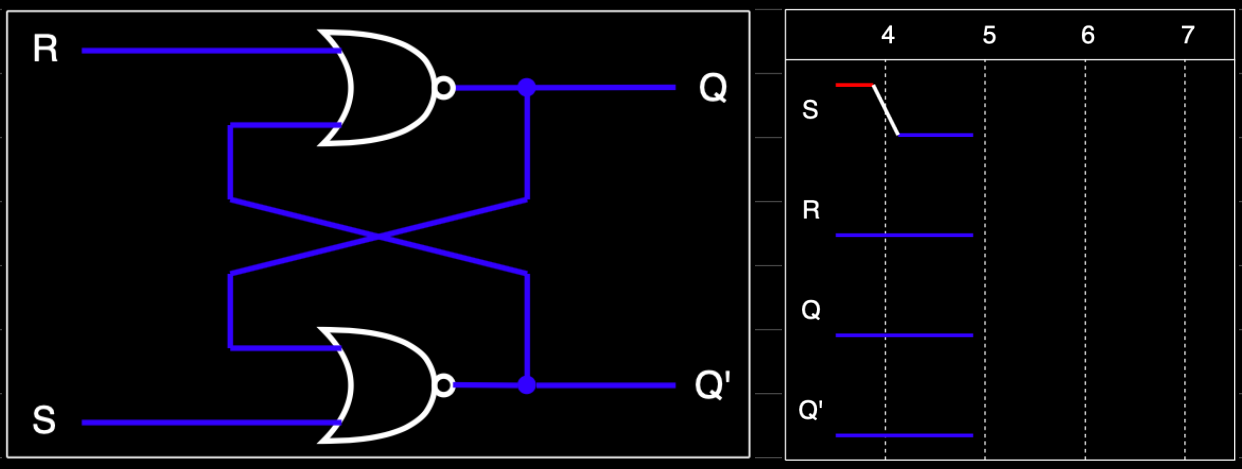
From t = 7 to t = 8 :-



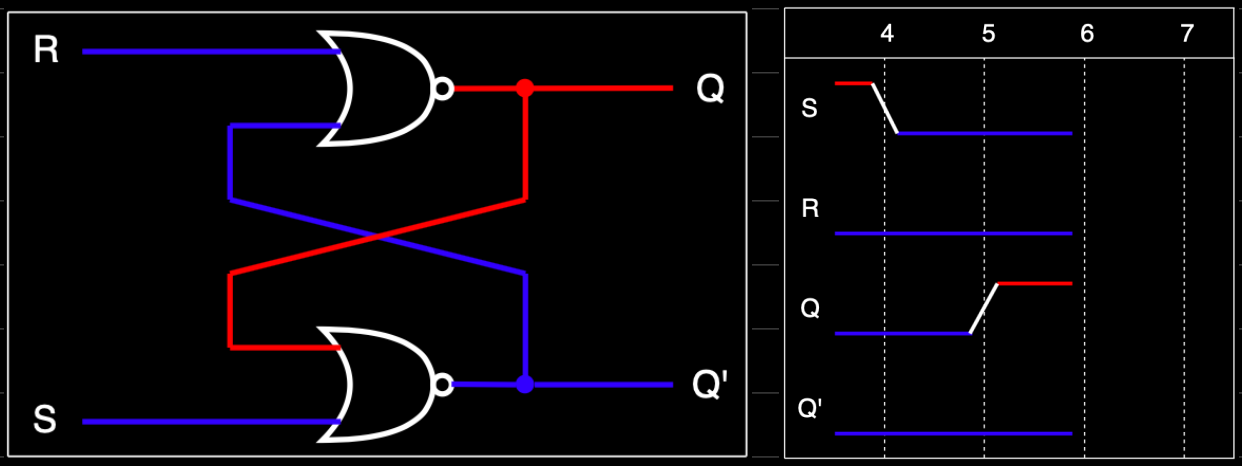
Hence, the latch will stay STABLE in the SET state, as long as S was maintained at 1 for a LONG ENOUGH time, i.e. if S gets changed back to 0 AFTER t = 5, then the latch will result in a STABLE SET state.

Now, let S get changed back to 0 at t = 4, instead of at t = 6.

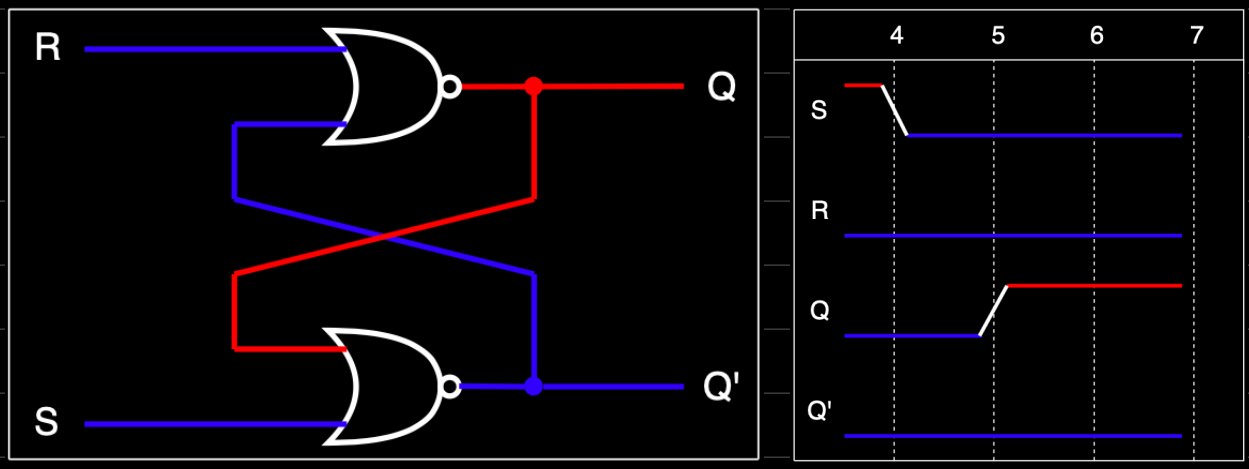
From t = 4 to t = 5 :-



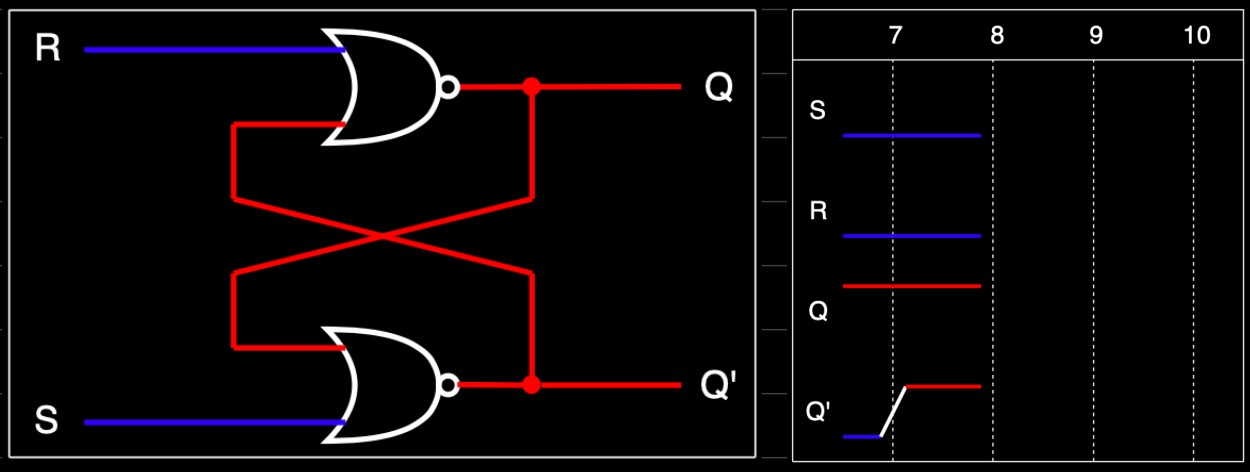
From t = 5 to t = 6 :-



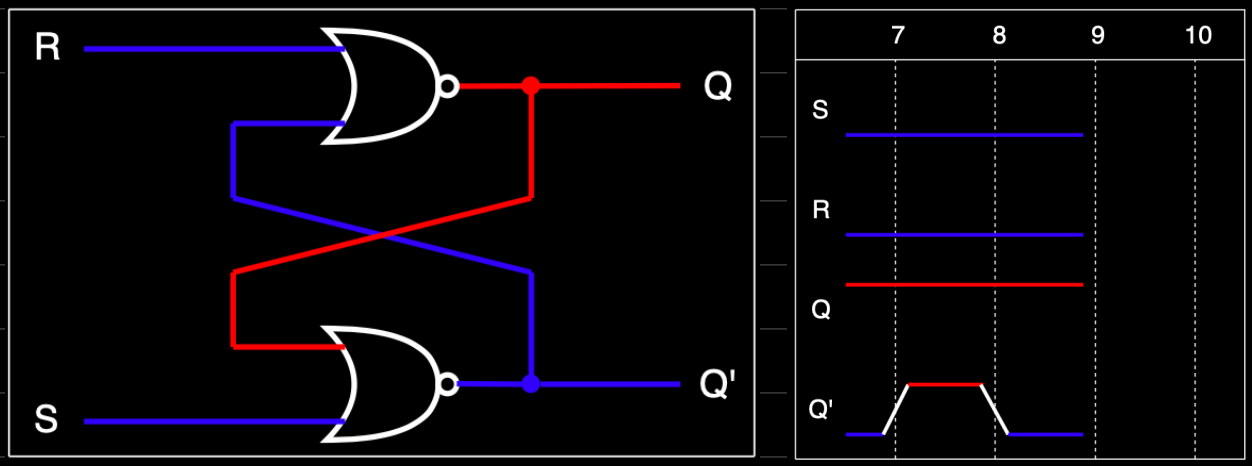
From t = 6 to t = 7 :-



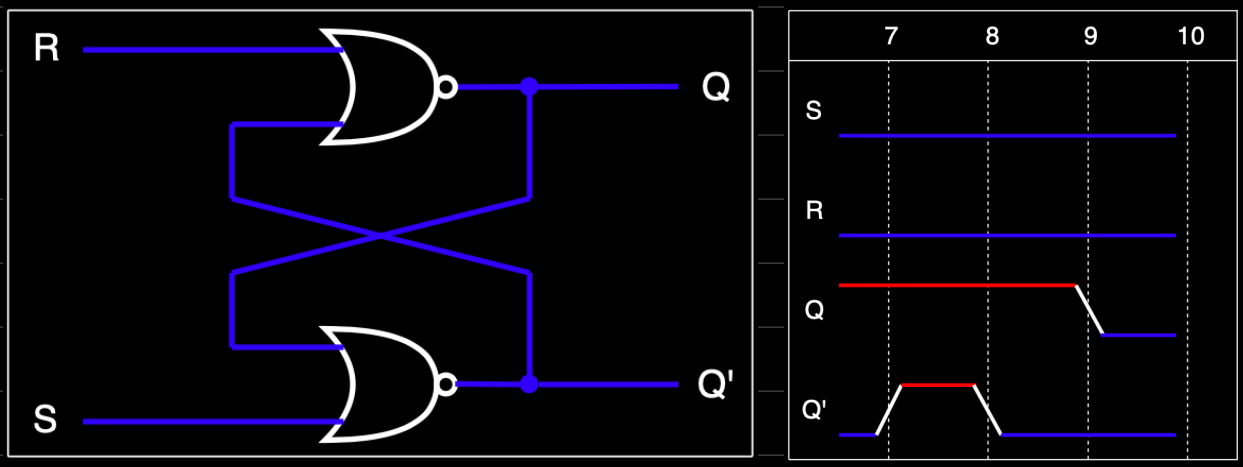
From t = 7 to t = 8 :-



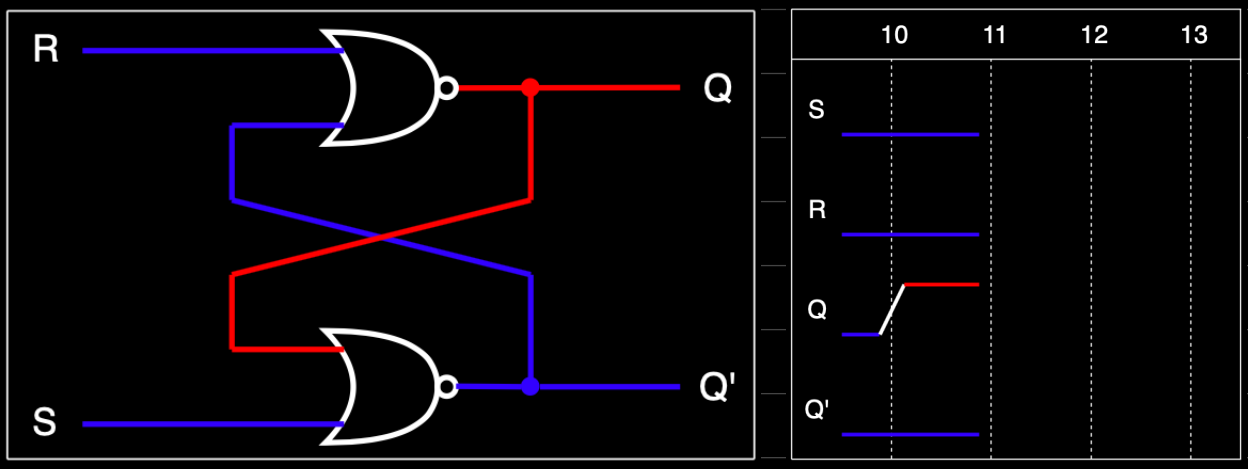
From t = 8 to t = 9 :-



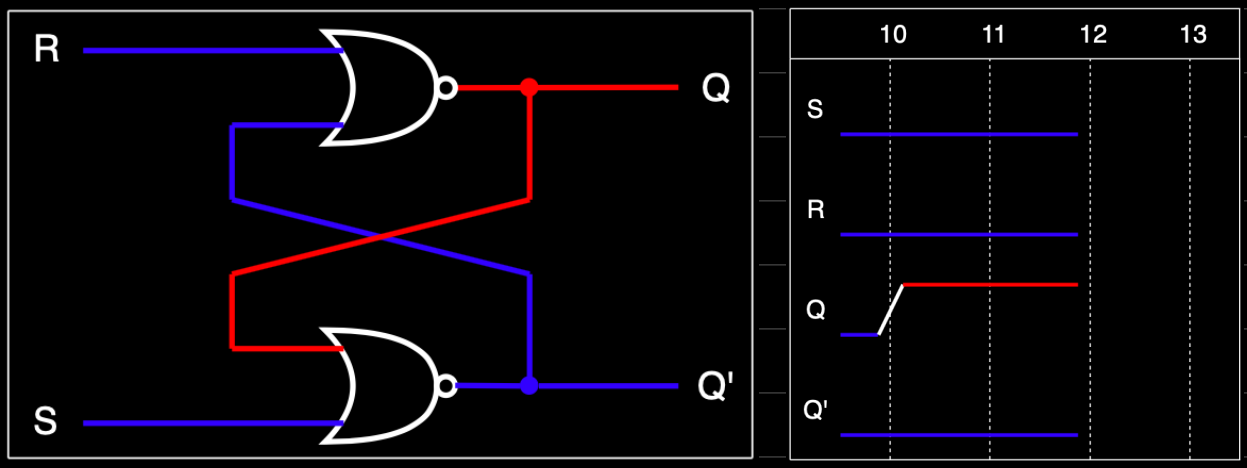
From t = 9 to t = 10 :-



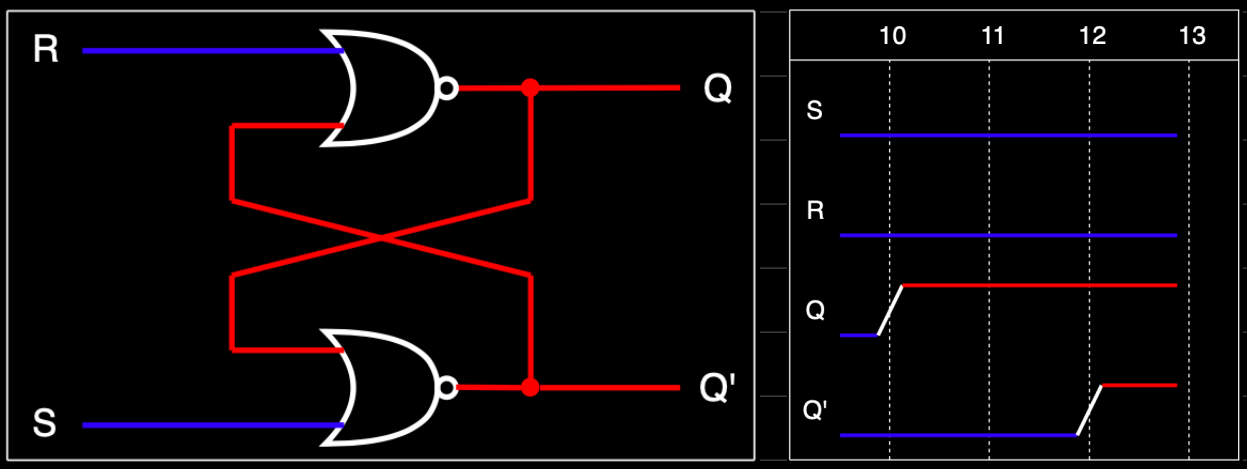
From t = 10 to t = 11 :-



From t = 11 to t = 12 :-



From t = 12 to t = 13 :-



The latch will keep oscillating like this FOREVER.

Previously, when the latch was being SET, it temporarily switched to an INVALID state, i.e. Q = Q' = 0, but then it became STABLE in the SET state.

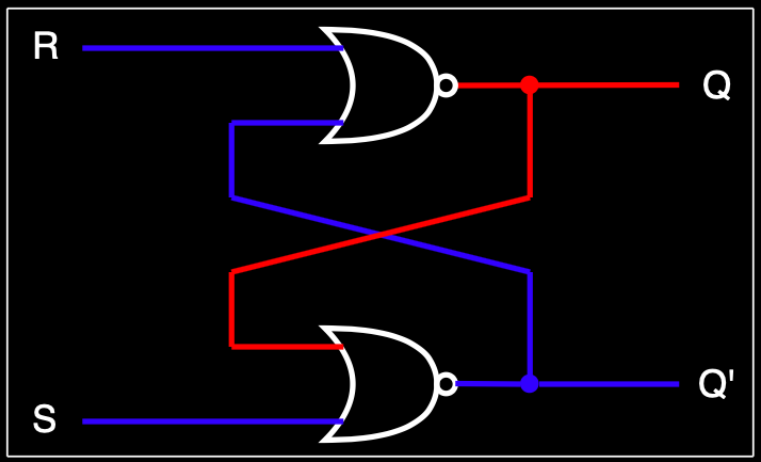
But, in this case, the latch will NEVER become STABLE in ANY state.

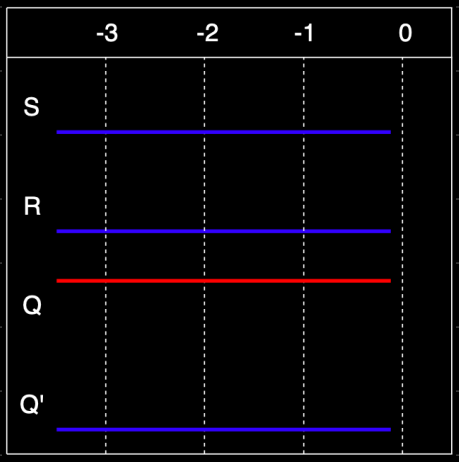
Therefore, if S is NOT maintained at 1 for a LONG ENOUGH time, then problems like this may occur.

It should be noted that the above analysis corresponds to an IDEAL version of logic gates. In real-world circuits, for EDGE cases like this, logic gates may NOT work as expected. For eg., even though ideally the latch should keep oscillating FOREVER, in real-world circuits, the latch MAY or MAY NOT keep oscillating due to the time it takes to turn on and off transistors, to charge internal capacitors, etc.

In any case, if used CORRECTLY, i.e. by keeping S/R high for a LONG ENOUGH time when setting/resetting the latch and by NOT keeping S & R high at the same time, then the latch will work AS EXPECTED in ideal as well as real-world circuits.

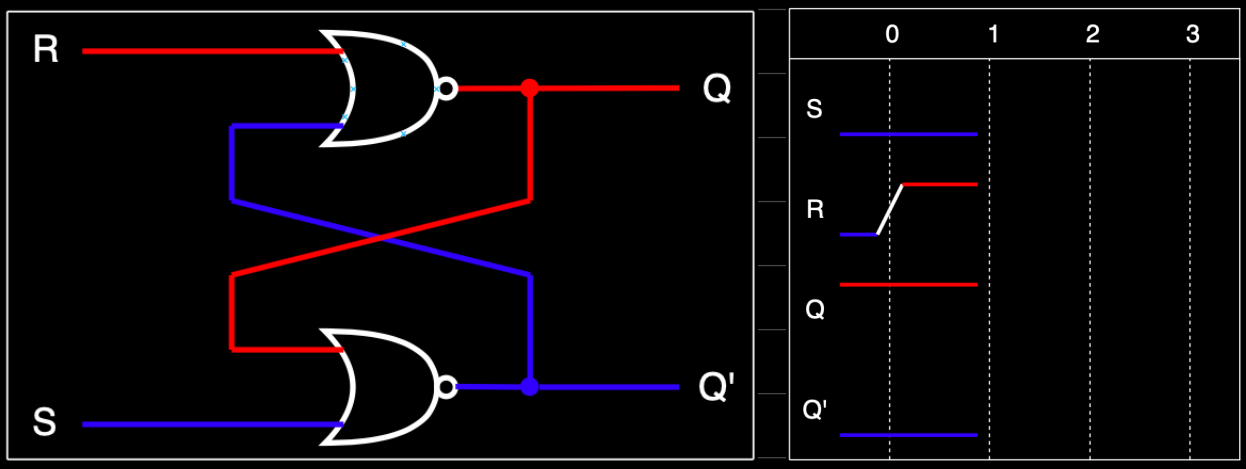
Now, Let the initial STABLE state be S = 0, R = 0, Q = 1 and Q' = 0 (SET).



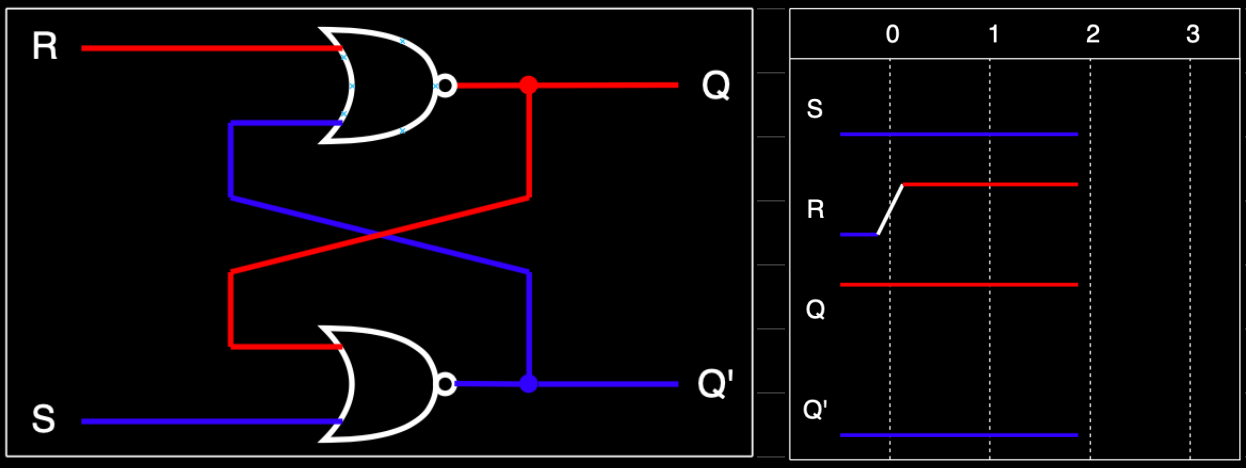


Let R get changed to 1 at t = 0 (in order for the latch to become CLEAR).

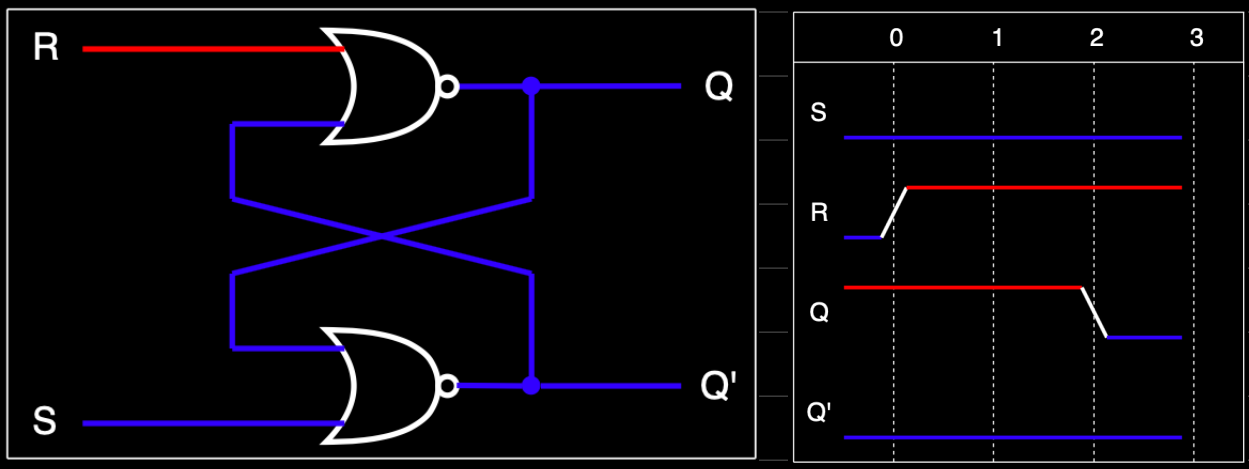
From t = 0 to t = 1 :-



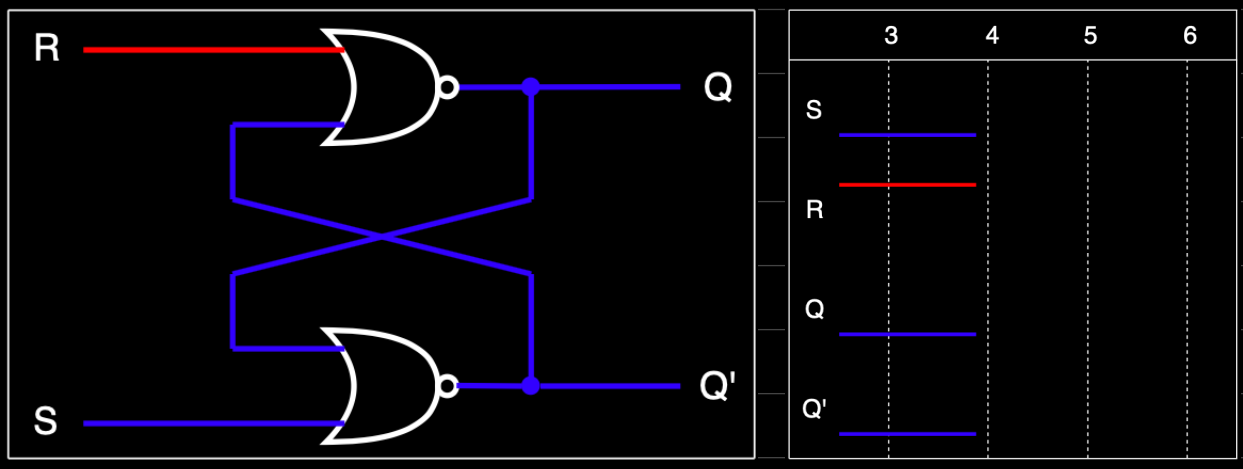
From t = 1 to t = 2 :-



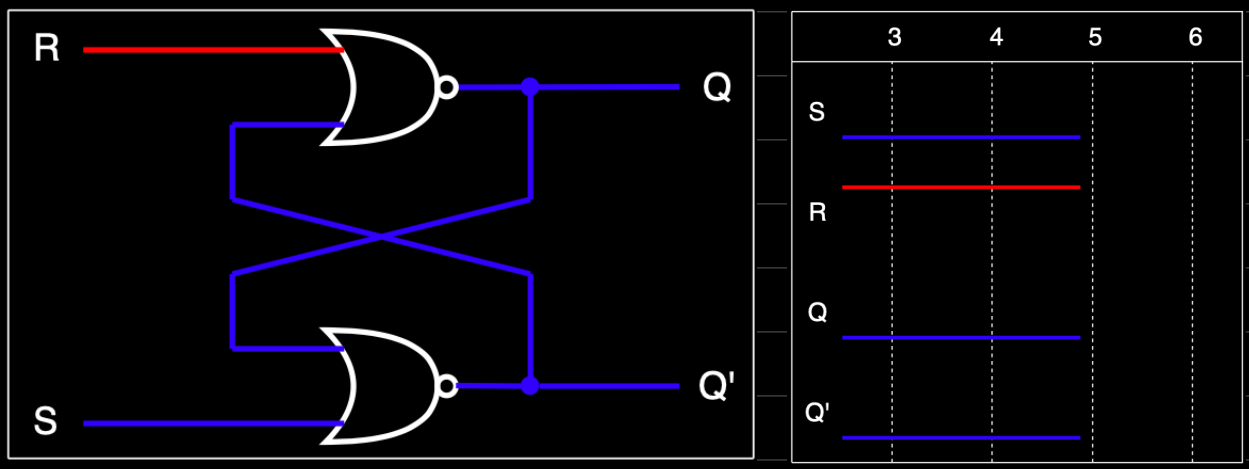
From t = 2 to t = 3 :-



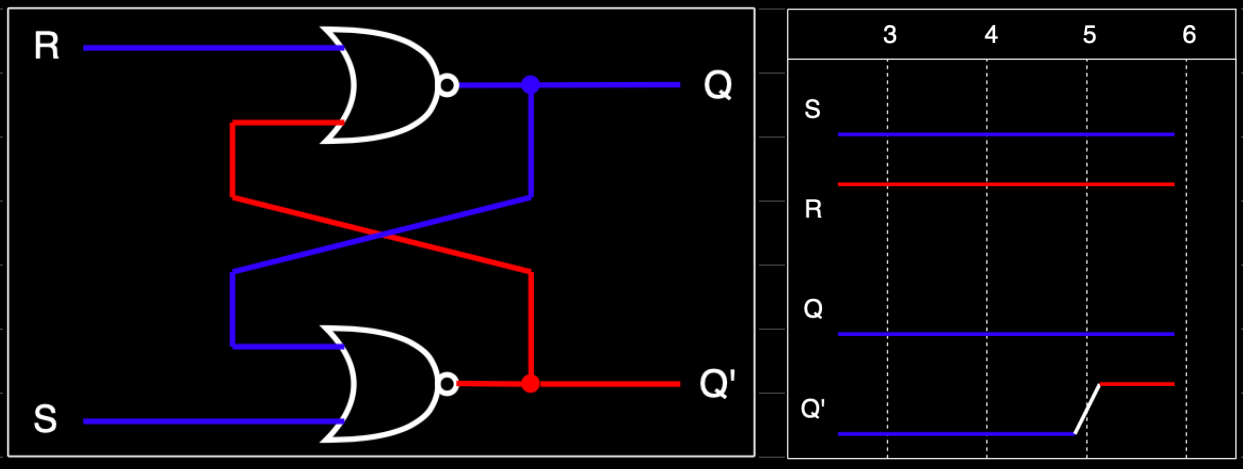
From t = 3 to t = 4 :-



From t = 4 to t = 5 :-



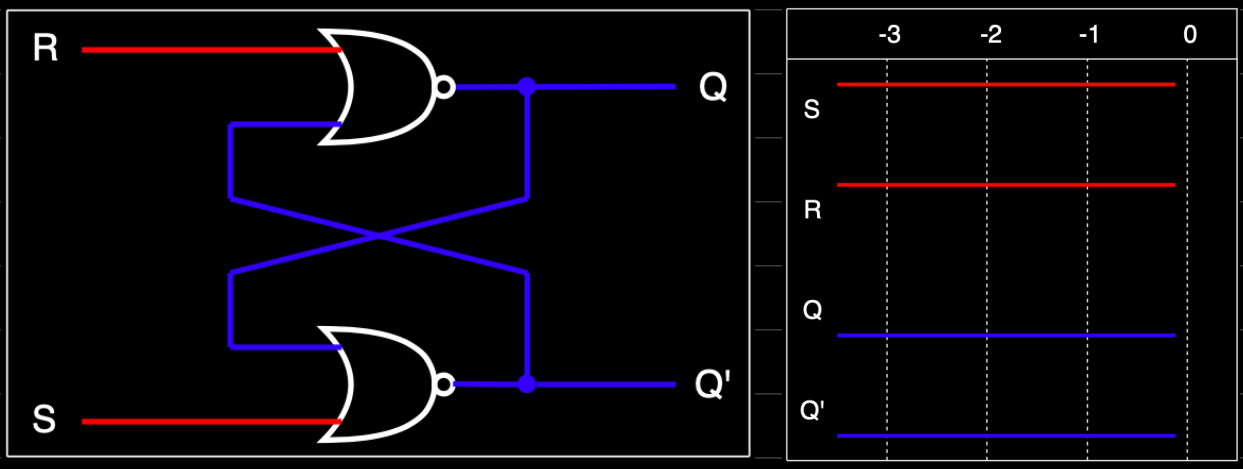
From t = 5 to t = 6 :-



Now, the latch will stay STABLE in the CLEAR state, even if R gets changed back to 0, as long as R was maintained at 1 for a LONG ENOUGH time, i.e. if R gets changed back to 0 AFTER t = 5, then the latch will result in a STABLE CLEAR state.

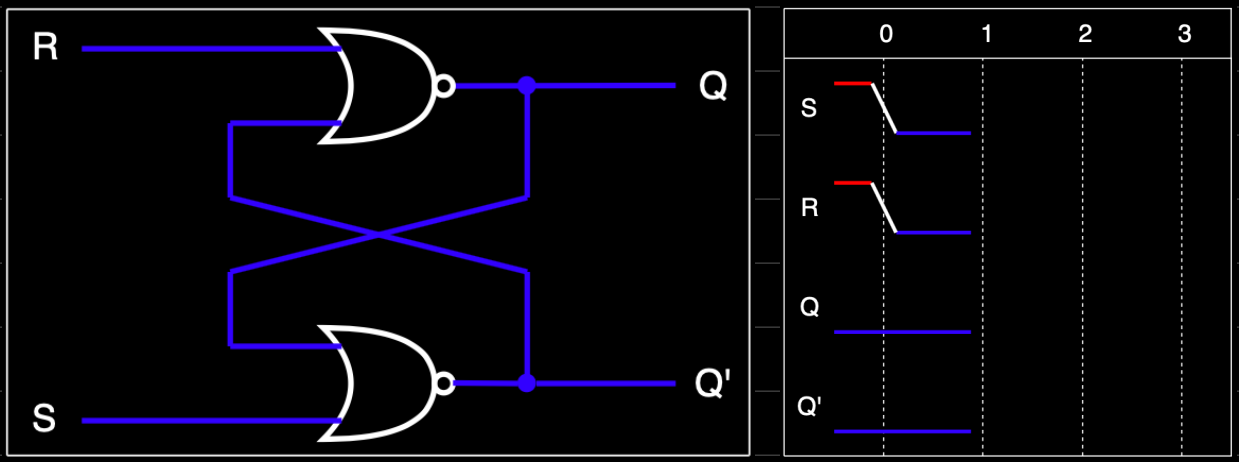
Therefore, if R is NOT maintained at 1 for a LONG ENOUGH time, then similar problems may occur like when S is NOT maintained at 1 for a LONG ENOUGH time when SETTING the latch.

Now, Let the initial STABLE state be S = 1, R = 1, Q = 0 and Q' = 0 (INVALID).

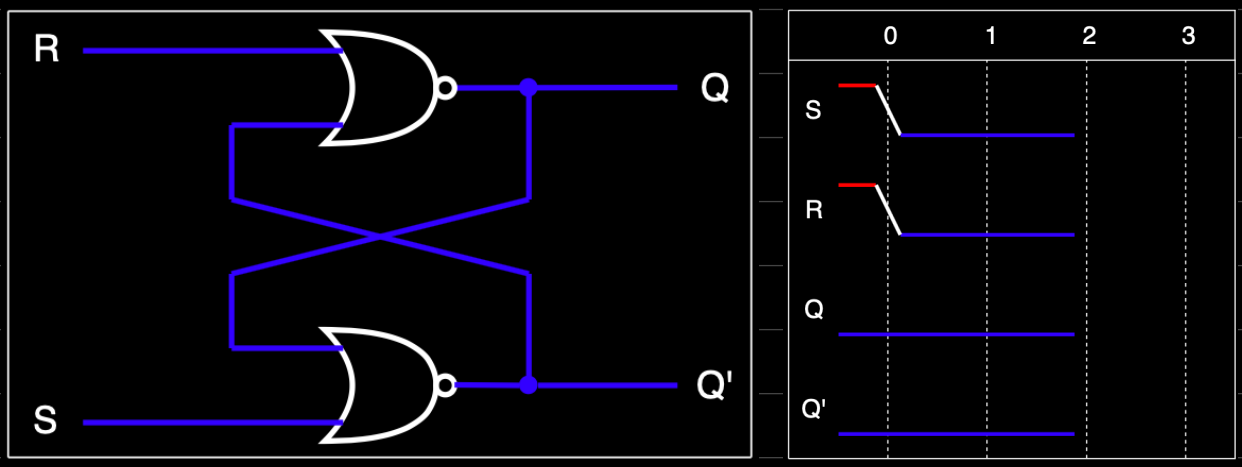


Let S & R both get changed to 0 at t = 0.

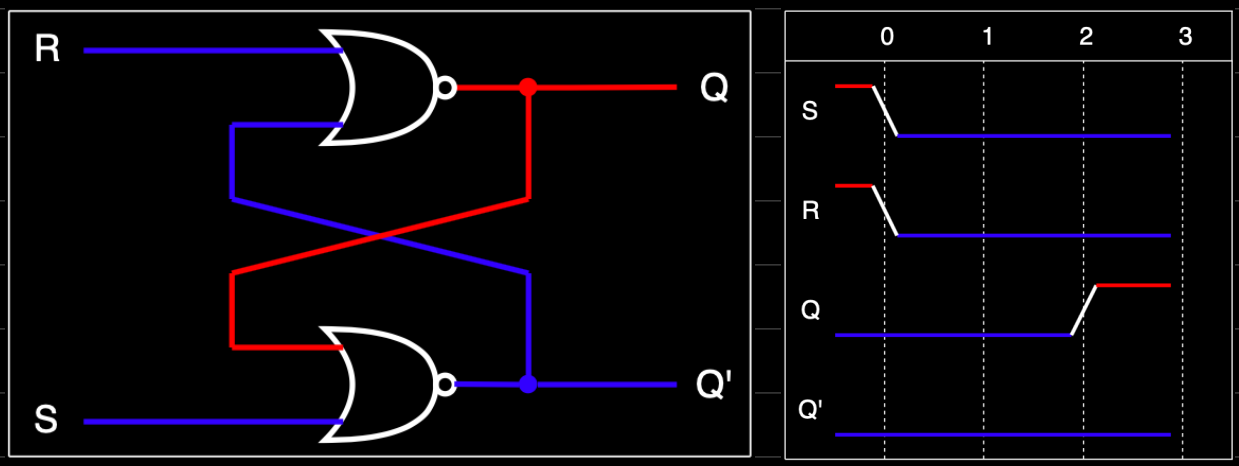
From t = 0 to t = 1 :-



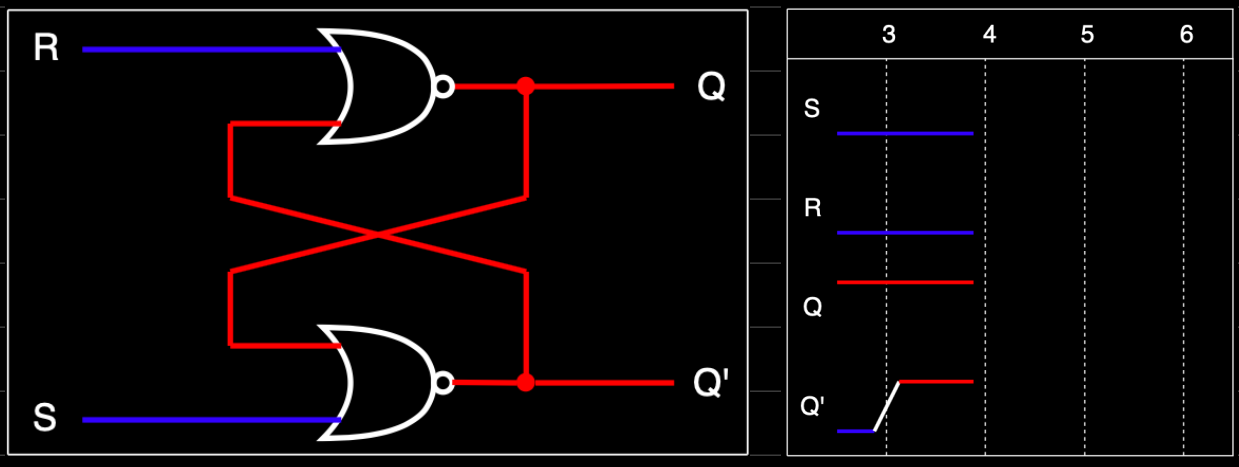
From t = 1 to t = 2 :-



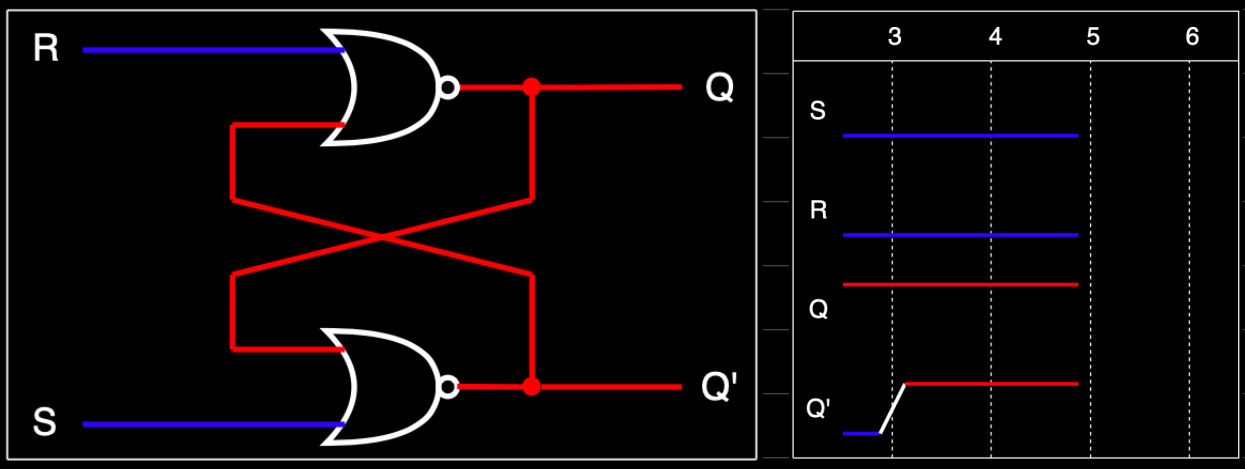
From t = 2 to t = 3 :-



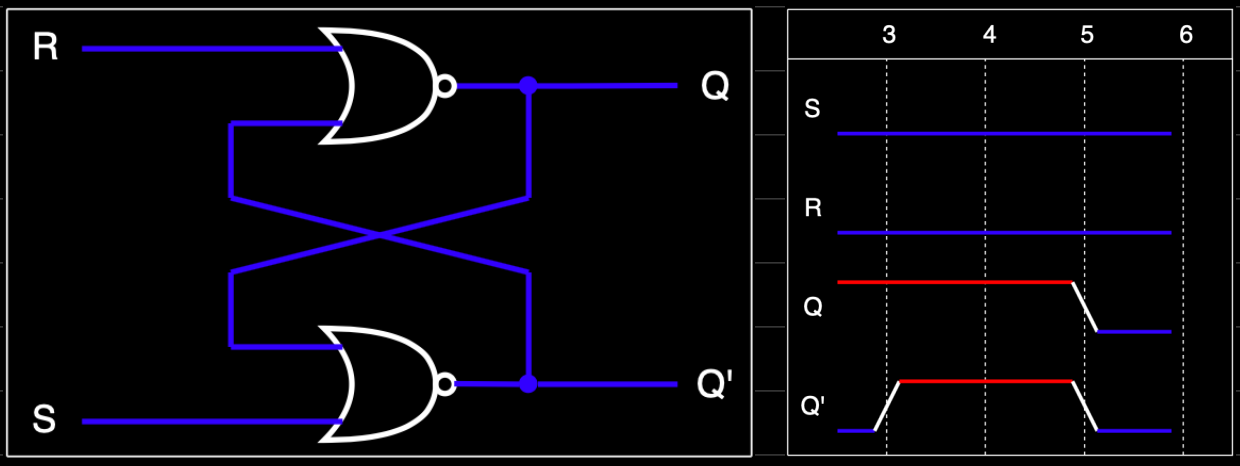
From t = 3 to t = 4 :-



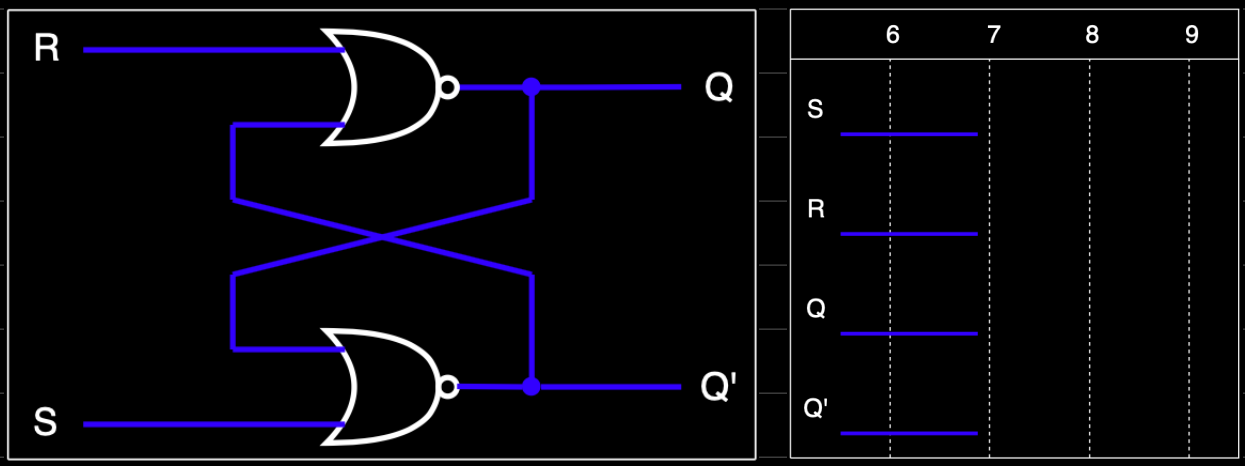
From t = 4 to t = 5 :-



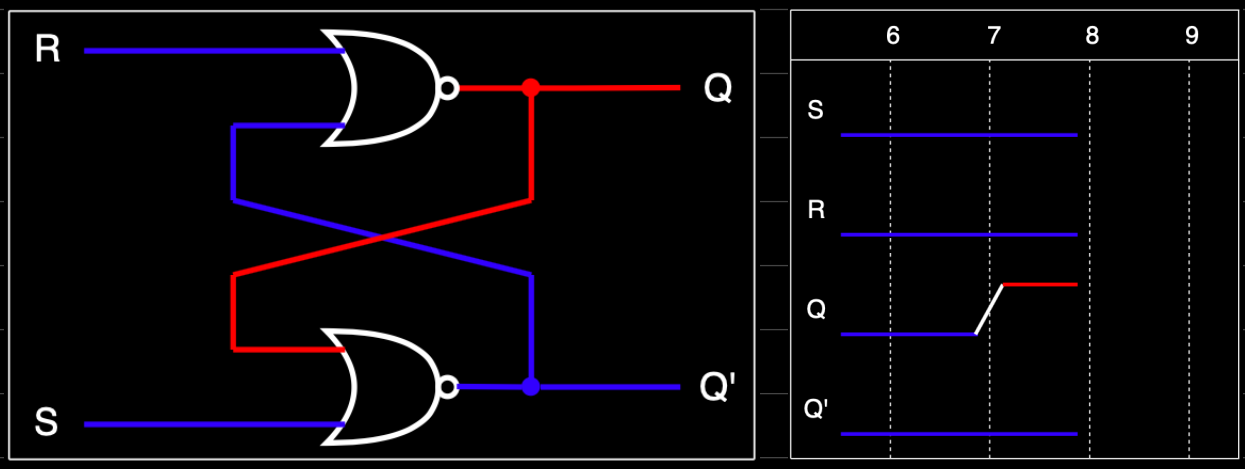
From t = 5 to t = 6 :-



From t = 6 to t = 7 :-



From t = 7 to t = 8 :-



The latch will keep oscillating like this FOREVER.

Therefore, if S and R both get changed to 0 after the latch is STABLE in the INVALID state of S = 1, R = 1, Q = 0 and Q' = 0, then problems like this may occur.

As stated previously, in real-world circuits, the latch MAY or MAY NOT keep oscillating.
In any case, the state of S = 1, R = 1, Q = 0 and Q' = 0 should be AVOIDED in ideal as well as real-world circuits.

It should be noted that if the initial STABLE state is S = 0, R = 0, Q = 1 and Q' = 0 (SET), and if S gets changed to 1, then the latch will REMAIN in the SET state.

Similarly, if the initial STABLE state is S = 0, R = 0, Q = 0 and Q' = 1 (CLEAR), and if R gets changed to 1, then the latch will REMAIN in the CLEAR state.

To be answered later (?) :-

After a stable state of S = 0, R = 0, Q = 1/0 and Q' = 0/1, if S & R both get changed to 1 and then back to 0 without waiting for a long enough time, then what happens?

After a stable state of S = 1/0, R = 0/1, Q = 1/0 and Q' = 0/1, if R/S gets changed to 1 and then after waiting for a long enough time or without waiting for a long enough time, S/R gets changed to 0 and stays at 0 for a long enough time, then what happens?
– The latch goes into a stable clear/set state.
This is because after S and R become constant at 1/0 and 0/1, respectively, then no matter the current state of the latch, stable or oscillating, the latch will result in a stable/clear state after a long enough time.

After a stable state of S = 1, R = 1, Q = 0 and Q' = 0, if only S/R gets changed to 0, then what happens? – The latch goes into a stable clear/set state.
But, if the other input gets changed to 0 as well without waiting for a long enough time for the latch to go into a stable clear/set state, then what happens?
Etc.

^ D latch

(https://youtu.be/peCh_859q7Q?si=M2RUMPAdl8BvUnTm)

To be answered later (?) :-

When the enable/clock is high and the latch is stable, then after changing D and keeping it constant for a long enough time, why does the NOT gate's propagation delay not cause problems?

The answer to this question is similar to the answer to the following question for the SR latch –

After a stable state of S = 1/0, R = 0/1, Q = 1/0 and Q' = 0/1, if R/S gets changed to 1 and then after waiting for a long enough time or without waiting for a long enough time, S/R gets changed to 0 and stays at 0 for a long enough time, then what happens?

– The latch goes into a stable clear/set state.

This is because after S and R become constant at 1/0 and 0/1, respectively, then no matter the current state of the latch, stable or oscillating, the latch will result in a stable set/clear state after a long enough time.

^ D flip-flop

(https://youtu.be/YW-_GkUguMM)

Ben uses the RESISTOR-CAPACITOR method for EDGE-TRIGGERING.
However, the ICs that he later uses for D flip-flops are based on the undermentioned method for EDGE-TRIGGERING.

Let the PROPAGATION delay of every NAND gate be 1 unit, and the DURATION of a CLOCK CYCLE be 20 units, with the clock staying at HIGH & LOW for 10 units each and the clock transitioning from LOW to HIGH at t = 20 T, where T is an integer.

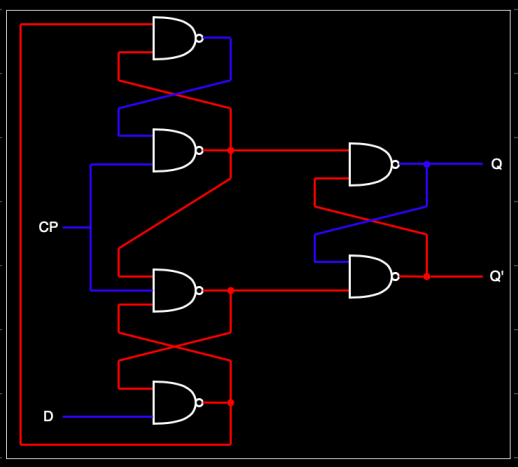
In real-world circuits, the duration of a clock cycle is MUCH MORE than this, because the circuits are quite LARGE and the clock cycle duration is decided such that the ENTIRE circuit becomes STABLE BEFORE the clock transitions from low/high to high/low.

The SETUP time (i.e. the DURATION for which D must be CONSTANT while CP is at 0 IMMEDIATELY BEFORE CP gets changed from 0 to 1) is 2 seconds.
The HOLD time (i.e. the DURATION for which D must be CONSTANT while CP is at 1 IMMEDIATELY AFTER CP gets changed from 0 to 1) is 1 second.
Hence, D must be constant from t = (20 T − 2) to t = (20 T + 1).
[These SETUP and HOLD times were calculated AFTER understanding the undermentioned transitions]

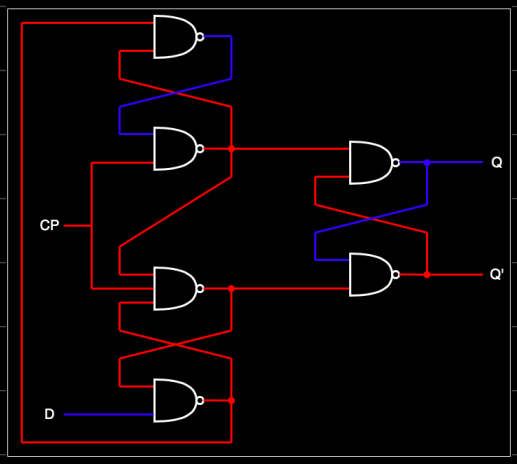
Let the initial STABLE state be CP = 0, D = 0, Q = 0 and Q' = 1 (CLEAR), and let the NEXT data to be stored be 0 (which is present at the D input at t = 0 and whose SETUP time is completed), then 1, then 1, and then 0.

Let the next data to be stored arrive at the D input WHILE the clock is STILL at 1 such that the HOLD time constraint is not violated, say at t = (20 T + 7).

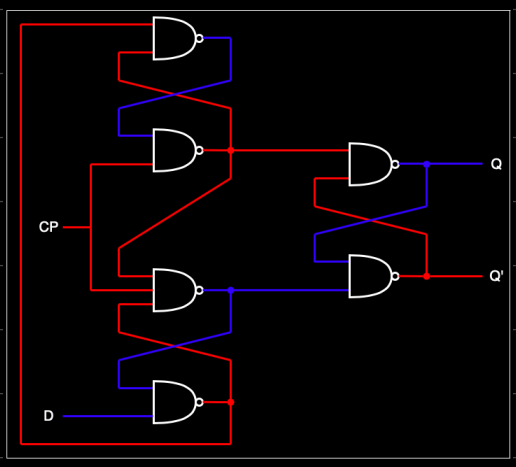
Till t = 0 :-



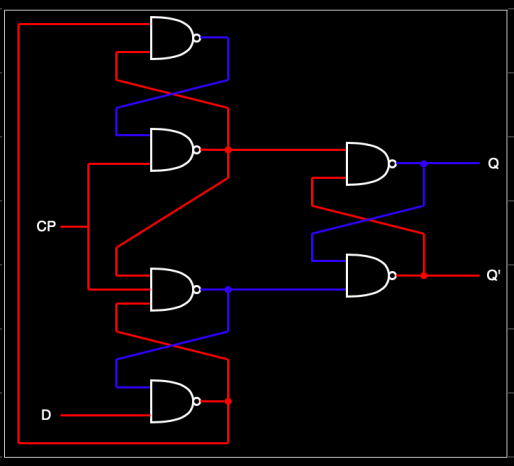
From t = 0 to t = 1 :-



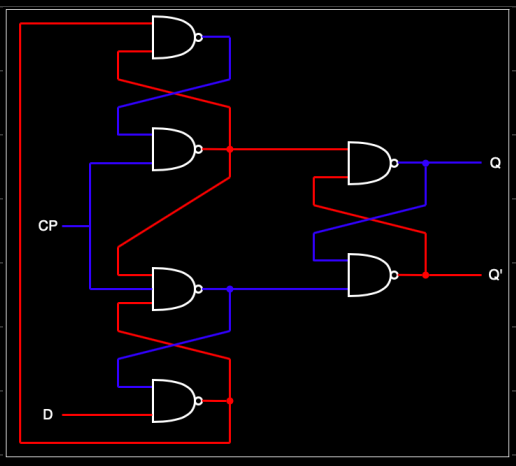
From t = 1 to t = 7 :-



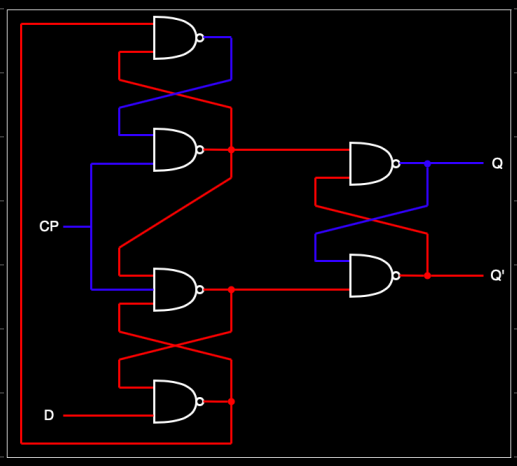
From t = 7 to t = 10 :-



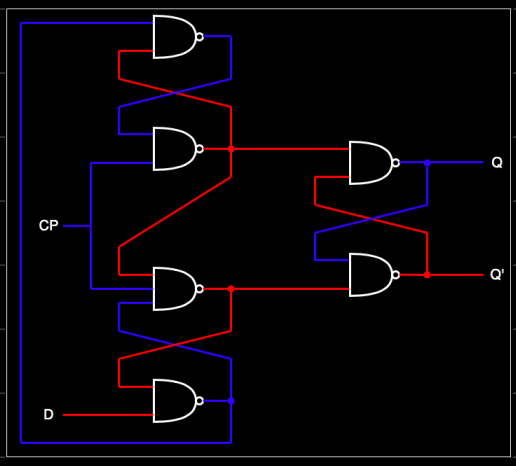
From t = 10 to t = 11 :-



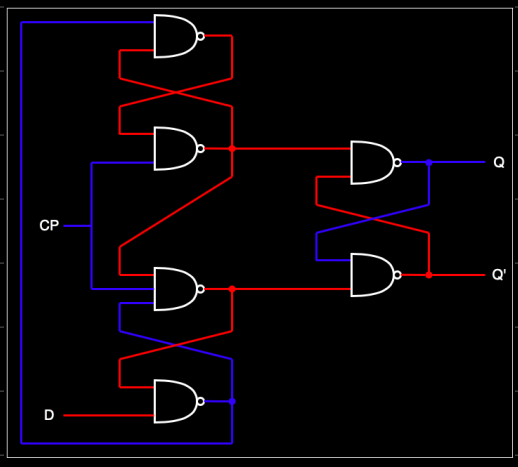
From t = 11 to t = 12 :-



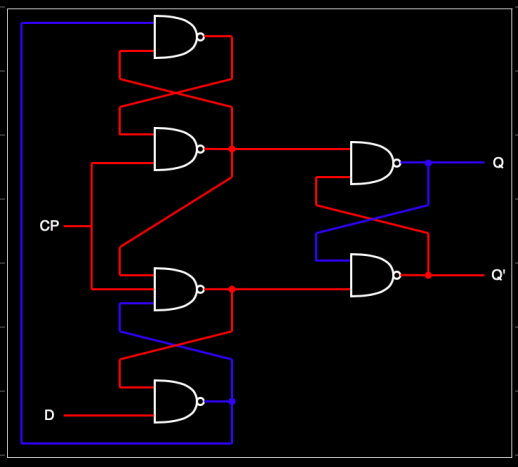
From t = 12 to t = 13 :-



From t = 13 to t = 20 :-



From t = 20 to t = 21 :-



From t = 21 to t = 22 :-

From t = -- to t = -- :-

From t = -- to t = -- :-