LATCH

Storage
Bi-stability
Latches

This is

THE END

of Combinatorial Circuits

and

the beginning

of sequential circuits

Sequential Circuits

- The output of a Combinatorial Circuit depends only on the current inputs

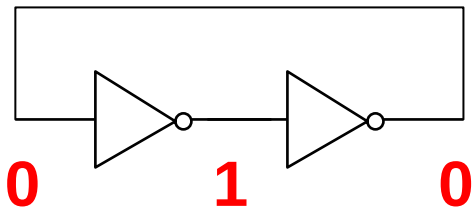
- The output of a Sequential Circuit can remember something about the past

Sequential Applications

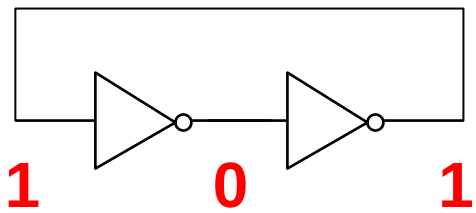
- Do 'X', then do 'Y' 3 times, then do 'Z'
- Take the dot product of 2 vectors
  - One element at a time
- Control a car wash
  - Rinse
  - Soap
  - Rinse
  - Dry

- All of these require *memory*
  - To remember where in the process they are…

Bi-Stability = Key to Memory



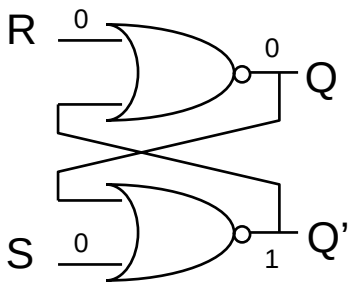
This is a stable state –
it will sit like this forever



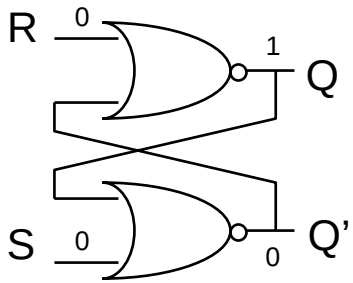
This is also a stable state –
it will sit like this forever

There are 2 stable states -
a bi-stable circuit…

SR Latch – A Bi-Stable Circuit

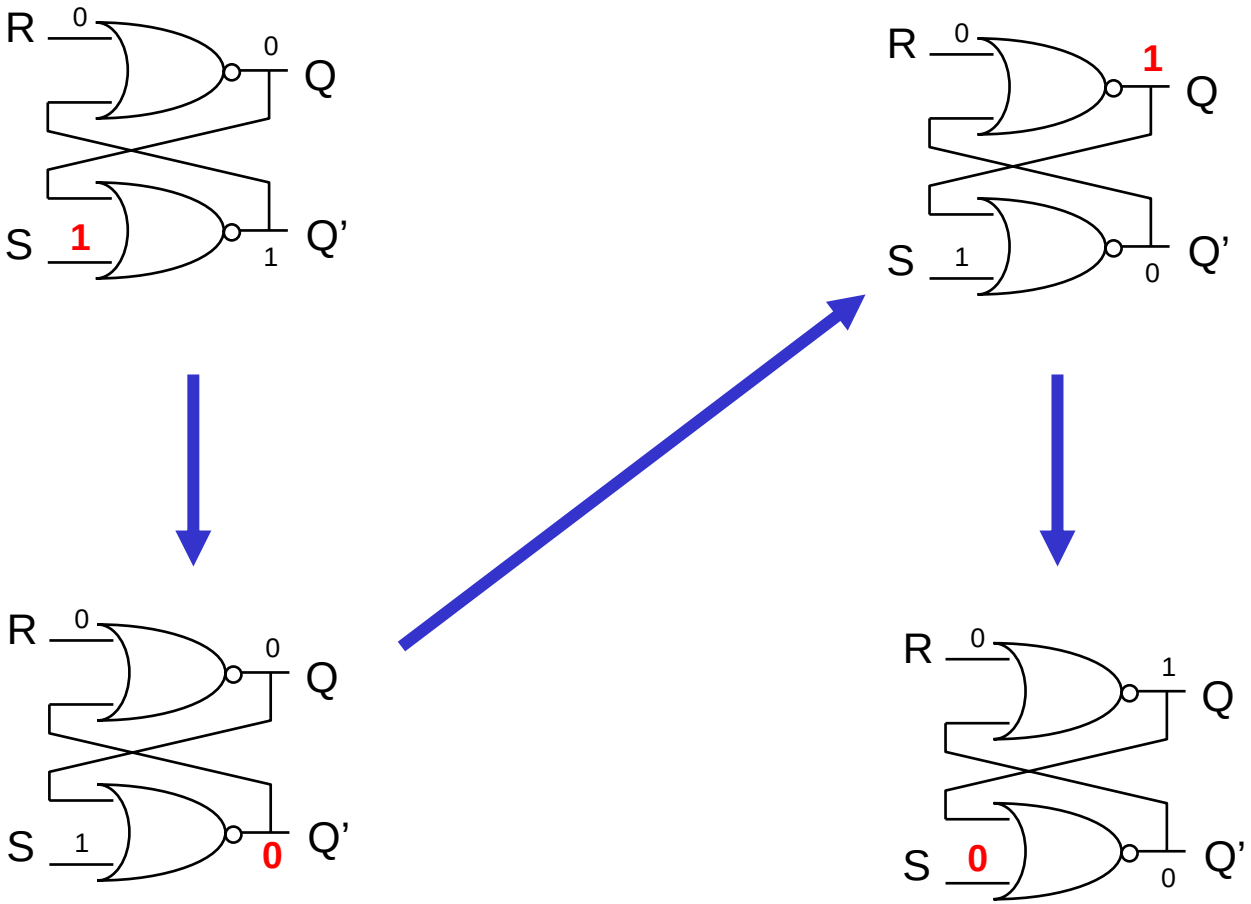


This is a stable state –
it will sit like this forever

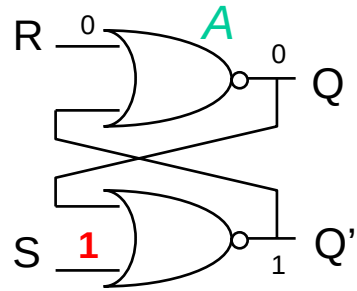


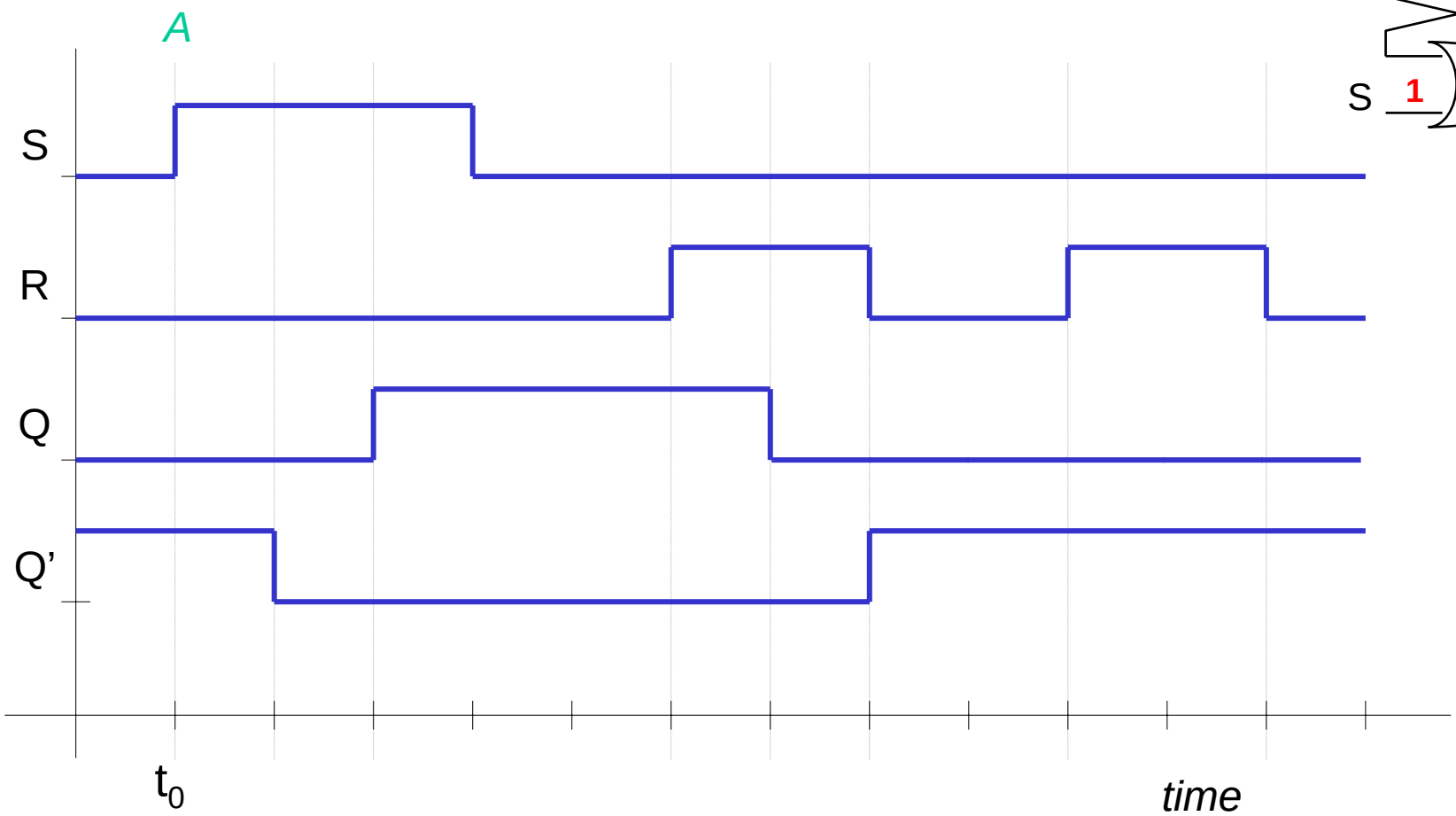
This is also a stable state –
it will sit like this forever

SR Latch Transition

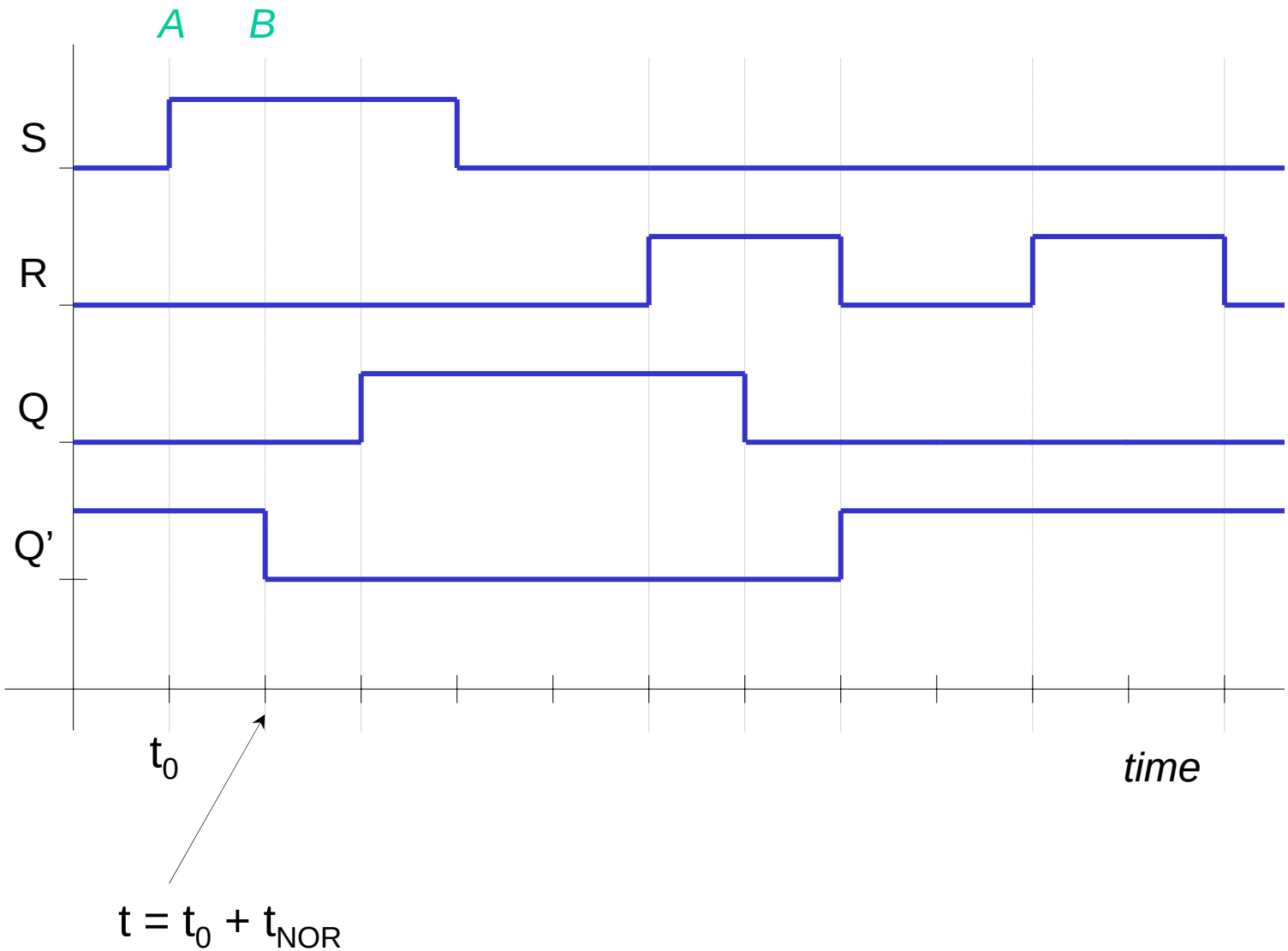


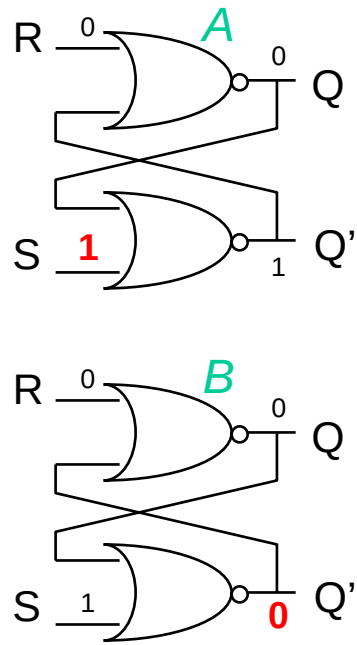
SR Latch Timing



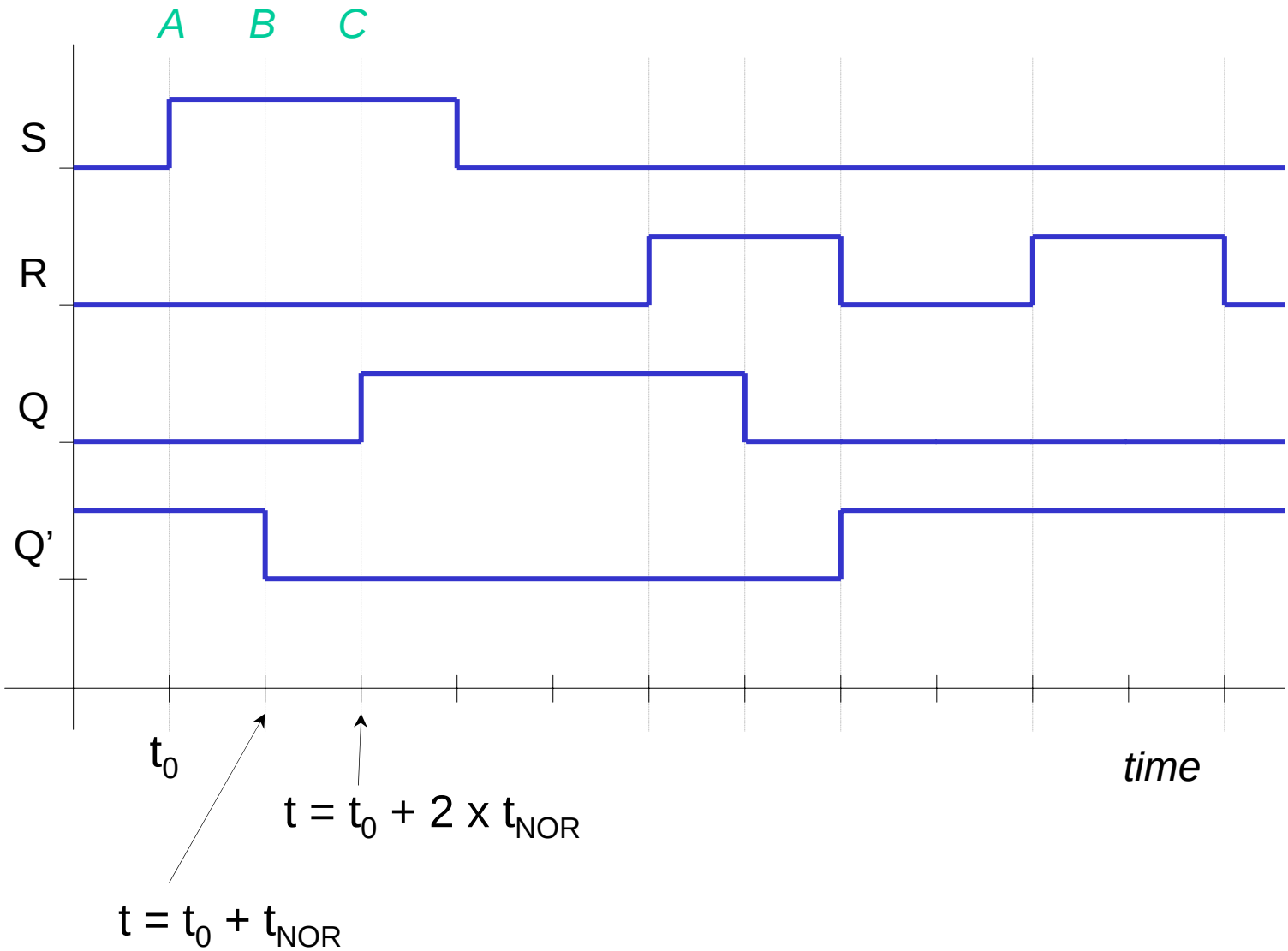


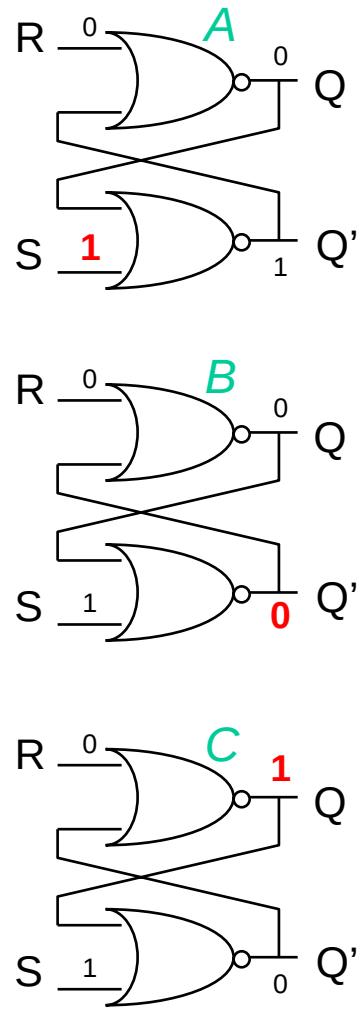
SR Latch Timing



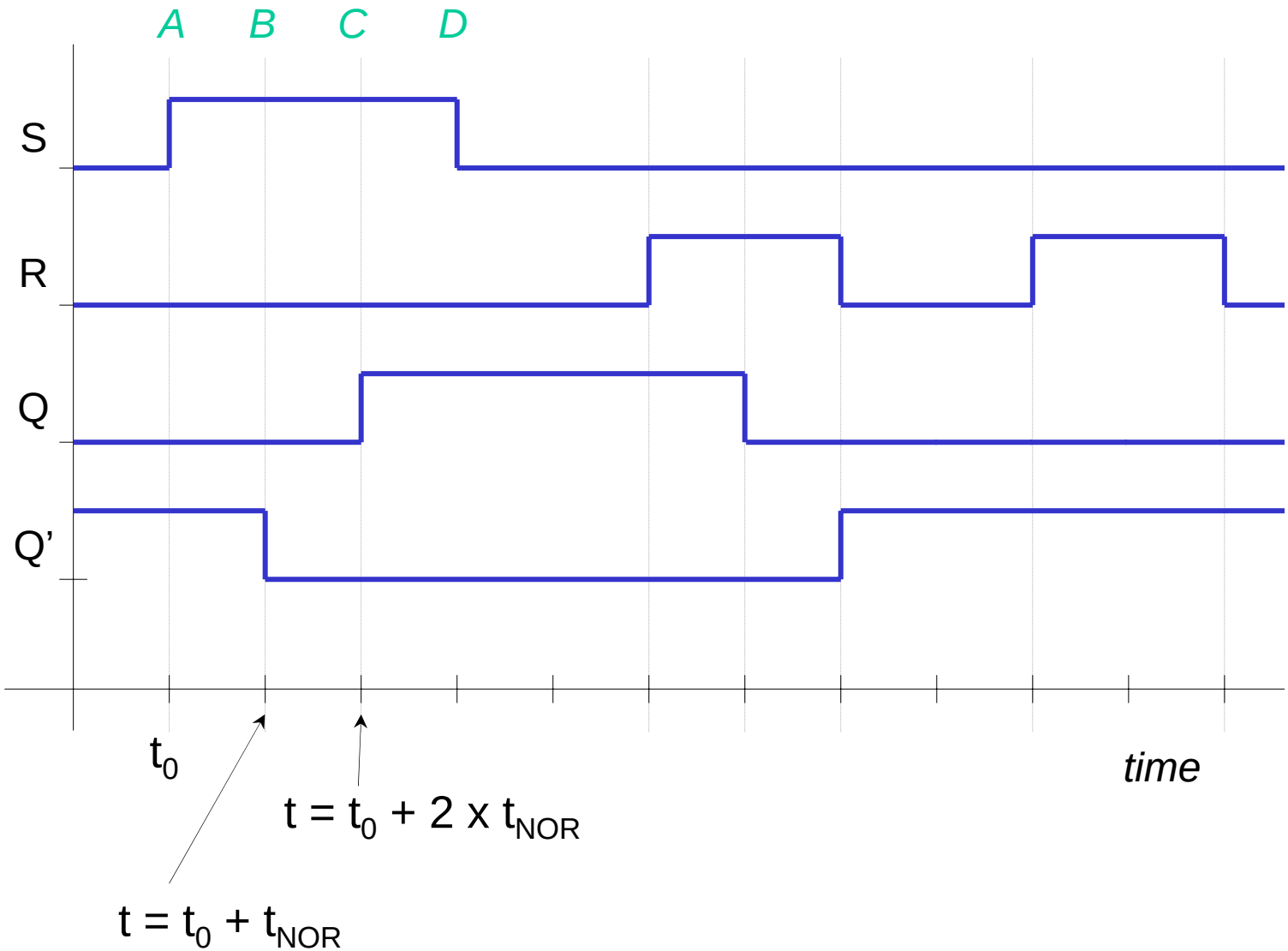


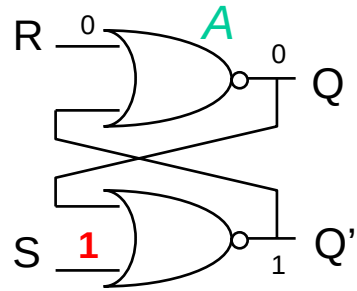
SR Latch Timing

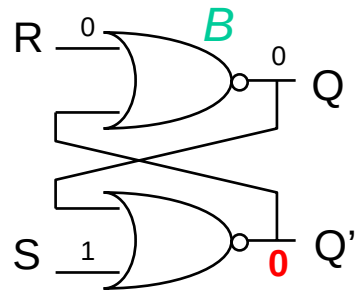


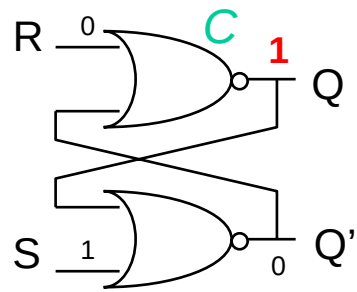


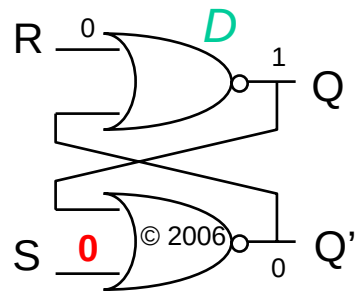
SR Latch Timing



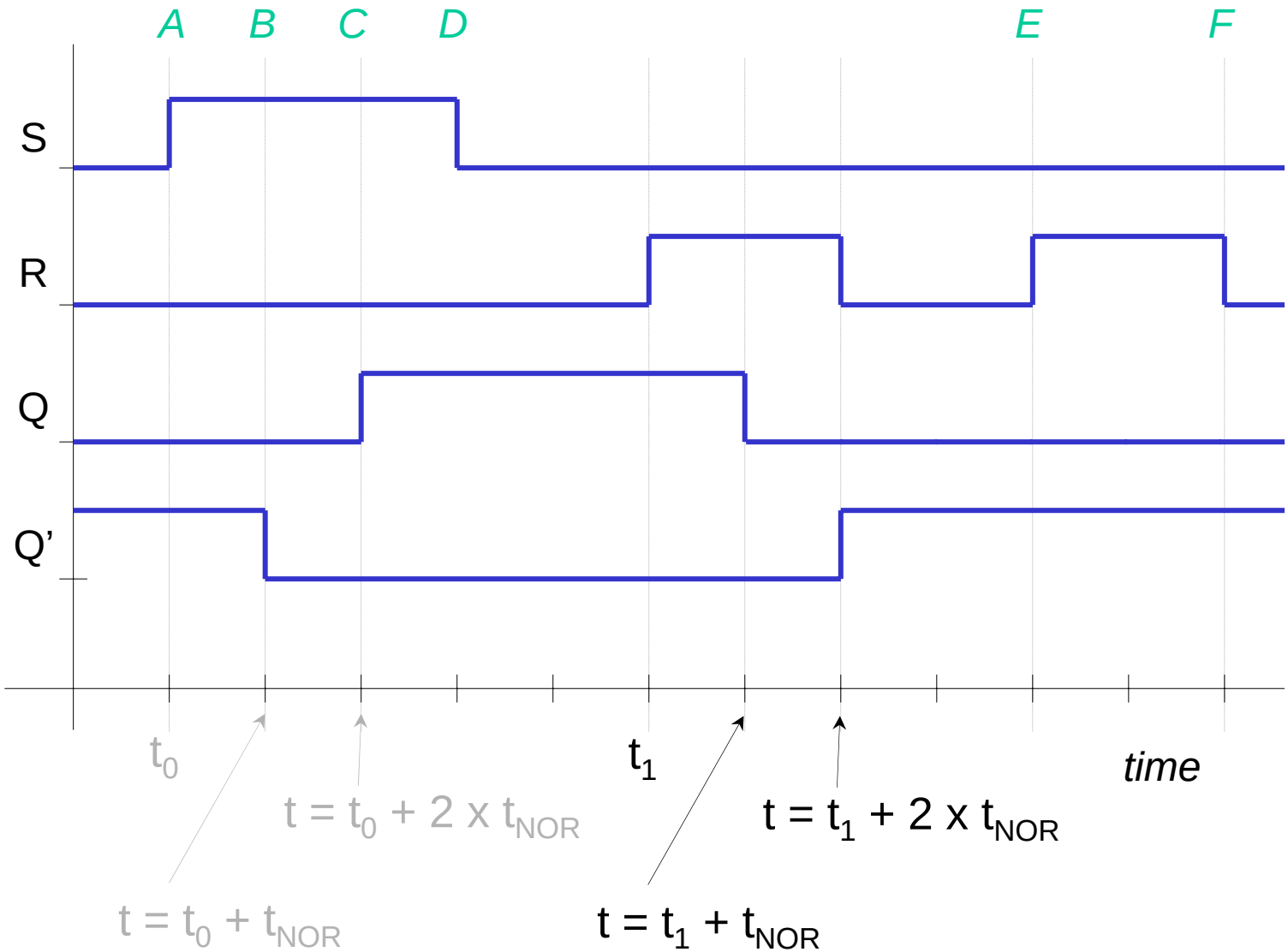


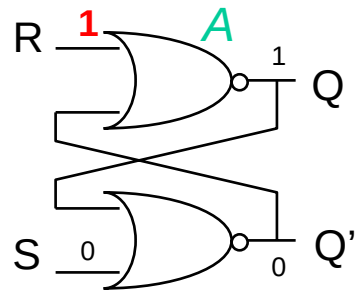


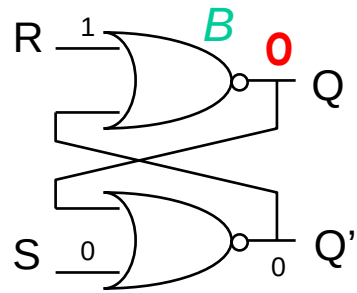


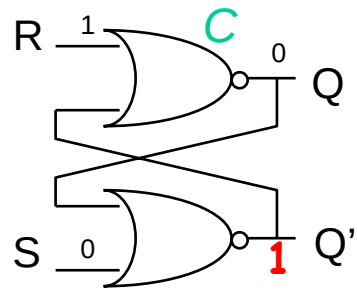


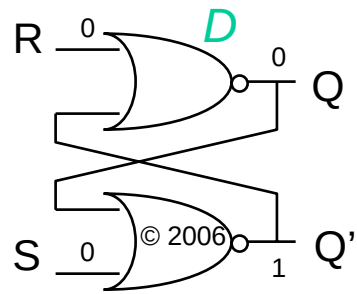
SR Latch Timing



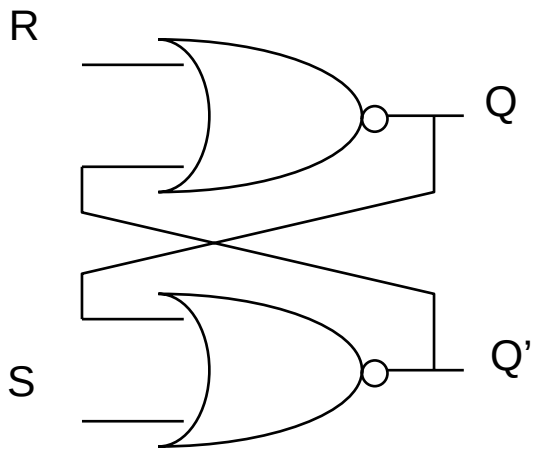








SR Latch Transition Table

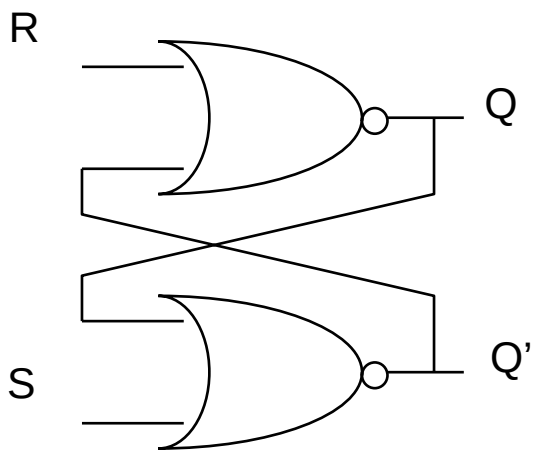


S	R	Q	Q+
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	?
1	1	1	?

The current state of the Q output

The next state of the Q output
(what it will change to)…

SR Latch Transition Table

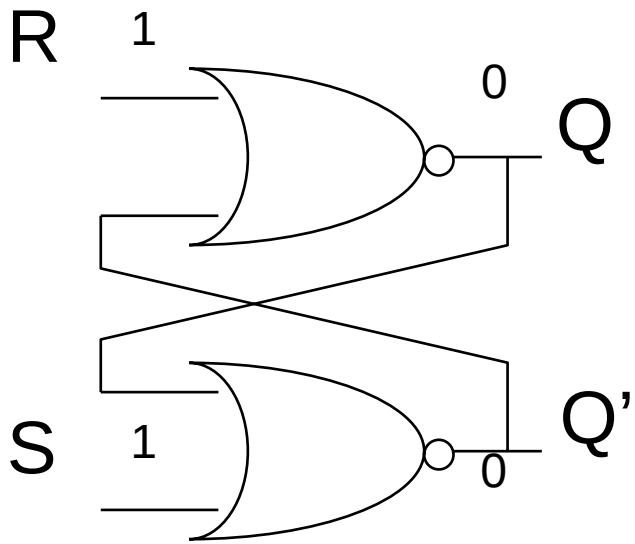


S	R	Q	Q+	
0	0	0	0	No change
0	0	1	1	
0	1	0	0	Reset it
0	1	1	0	
1	0	0	1	Set it
1	0	1	1	
1	1	0	?	
1	1	1	?	

The current state of the Q output

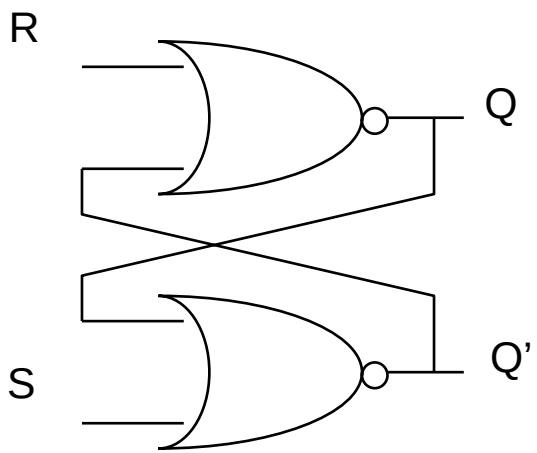
The next state of the Q output
(what it will change to)…

SR Latch: S=R='1'



Is the latch *SET*??? => no
Is the latch *RESET*??? => no
What is it? => neither
We avoid this input combination in normal usage, mainly because it makes no sense

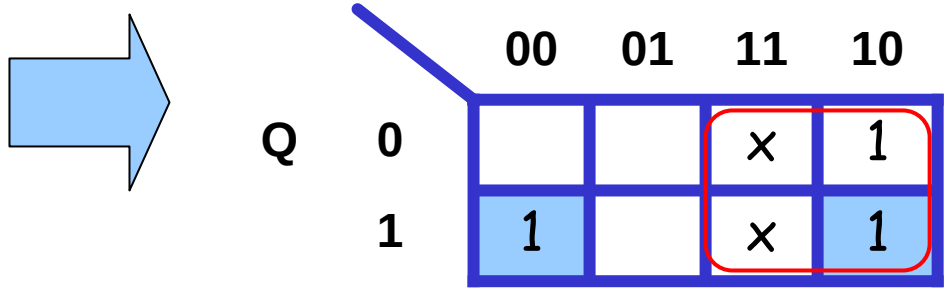
SR Latch Transition Table



S	R	Q	Q+	
0	0	0	0	No change
0	0	1	1	
0	1	0	0	Reset it
0	1	1	0	
1	0	0	1	Set it
1	0	1	1	
1	1	0	N/A	
1	1	1	N/A	

SR Latch – Next State Equation

S	R	Q	Q+
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	X
1	1	1	X



$$Q+ = S + R'Q$$

The next state of Q will be true any time:
+ you SET it
or
+ it is already true and you don't RESET it

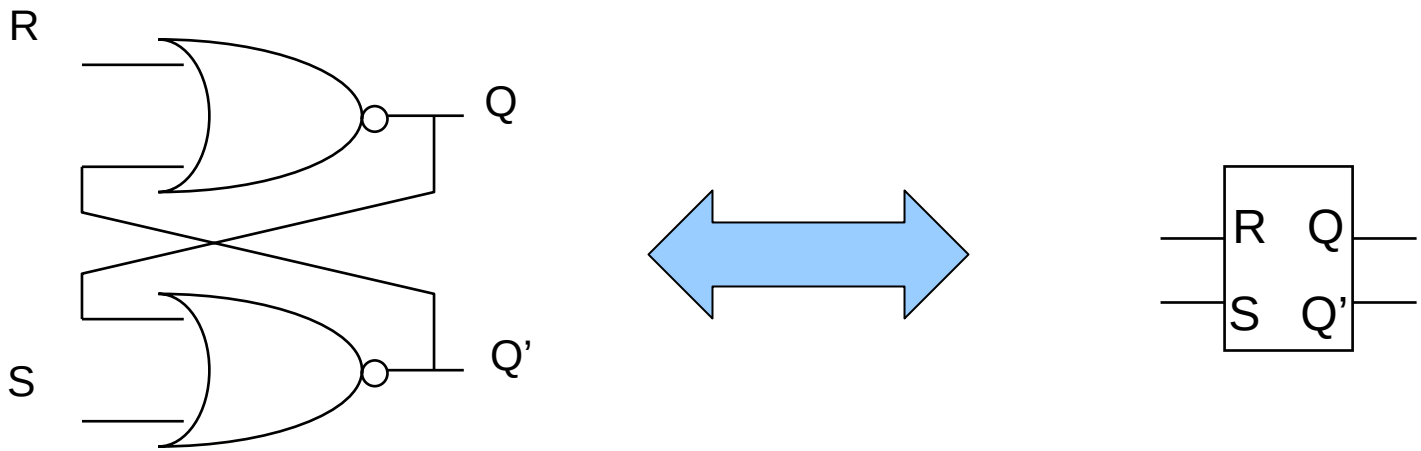
SR Latches – So What?

- Illustrate simple notion of *bi-stability*
  - Two stable states
  - S and R inputs move latch between them
- A memory
  - when Q='1' ⇔ latch storing a '1'
  - when Q='0' ⇔ latch storing a '0'
- Will hold its value indefinitely

SR Latches – What Are They Used For?

- Mainly to explain simple storage in digital design textbooks ☺
- Simple SR latch not used in designs very much
  - Due to some timing issues we will learn later…
- Simple SR latch forms basis for most other kinds of storage elements we will study

Symbology



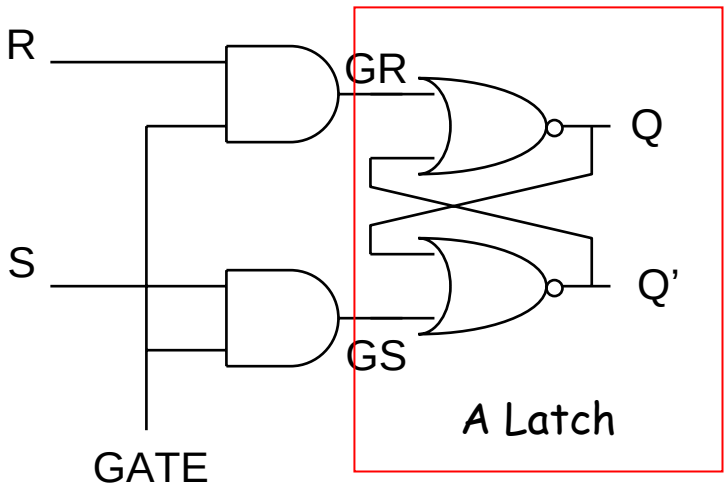
GLATCH

Gated Latches

SR Latches Always Sampling Inputs

- An SR latch will:
  - Respond to S/R input changes
    - *ALWAYS!*
- Sometimes we want to:
  - Control when a storage element loads
    - Use a *gated latch*

The Gated SR Latch

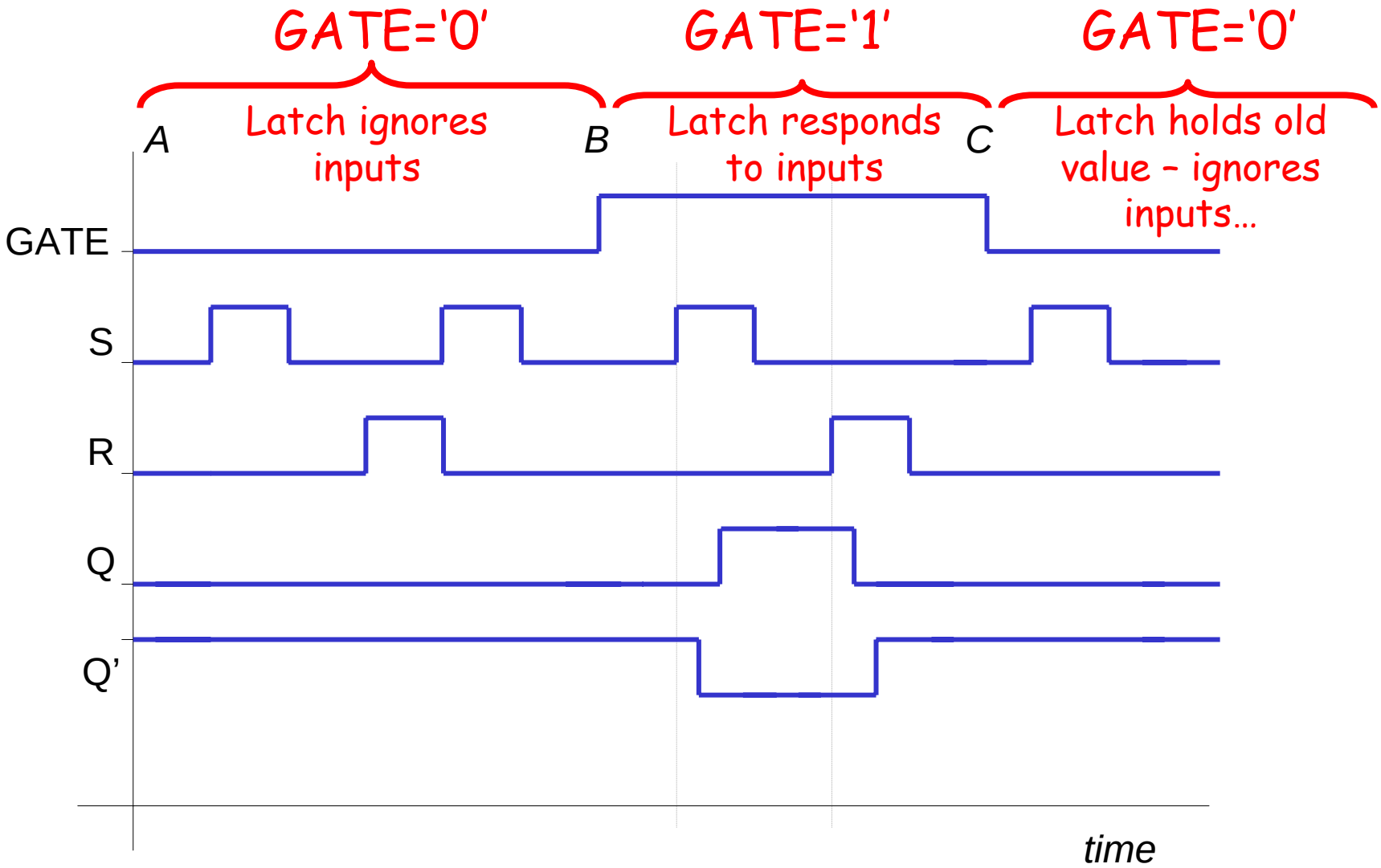


When GATE='0' ⇔
GR=GS='0' ⇔
latch cannot be modified

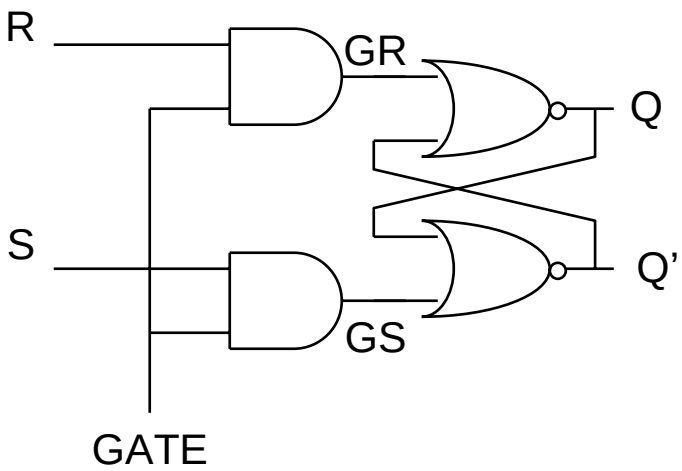
When GATE='1' ⇔
GR=R, GS=S ⇔
works like an SR latch

The GATE signal allows us to control
when
the latch will be loaded with a new value

Gated SR Latch Timing



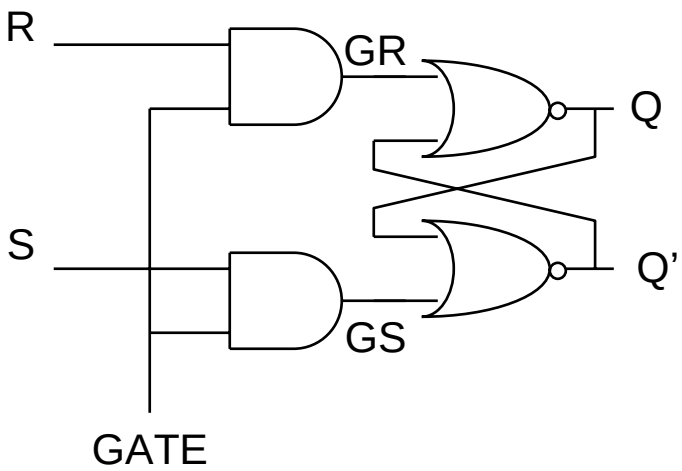
Gated SR Latch Next State Equation



How many inputs are in the transition table?

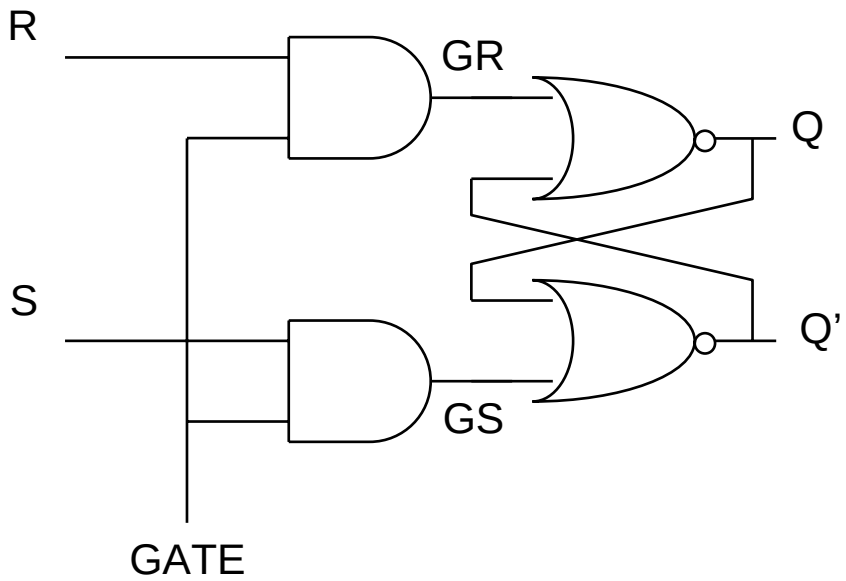
Can you draw it?

Gated SR Latch Truth Table



Gate	S	R	Q	Q+	
0	0	0	0	0	No
0	0	0	1	1	
0	0	1	0	0	change
0	0	1	1	1	
0	1	0	0	0	possible
0	1	0	1	1	
0	1	1	0	X	
0	1	1	1	X	
1	0	0	0	0	
1	0	0	1	1	Like
1	0	1	0	0	
1	0	1	1	0	an
1	1	0	0	1	
1	1	0	1	1	SR
1	1	1	0	X	
1	1	1	1	X	latch

Gated SR Latch Next State Equation

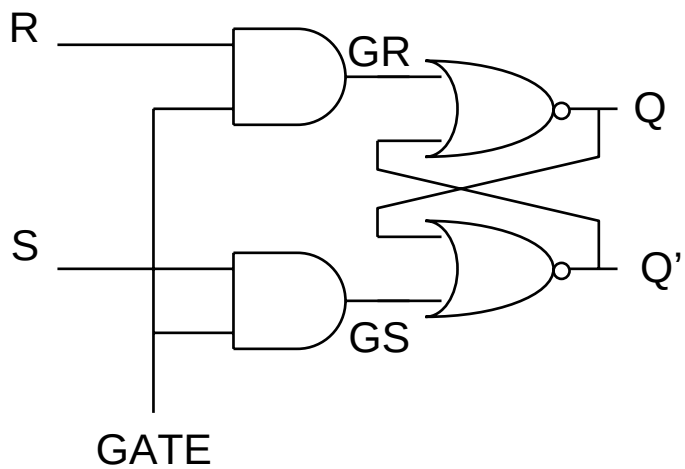


R Q \ Gate S	00	01	11	10
00			1	
01	1	1	1	1
11	1	x	x	
10		x	x	

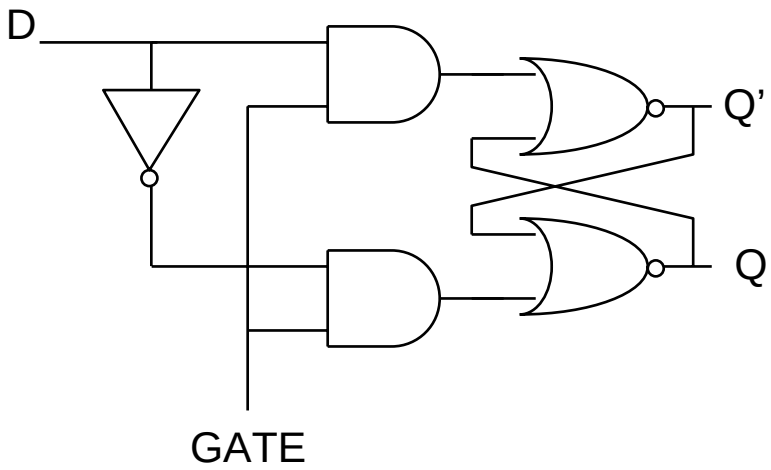
Q+ = GATE•S + R'•Q + GATE'•Q

Gated SR Latch

- Sometimes known as a *loadable SR latch*
  - Can be *loaded* with new value



The Gated D Latch

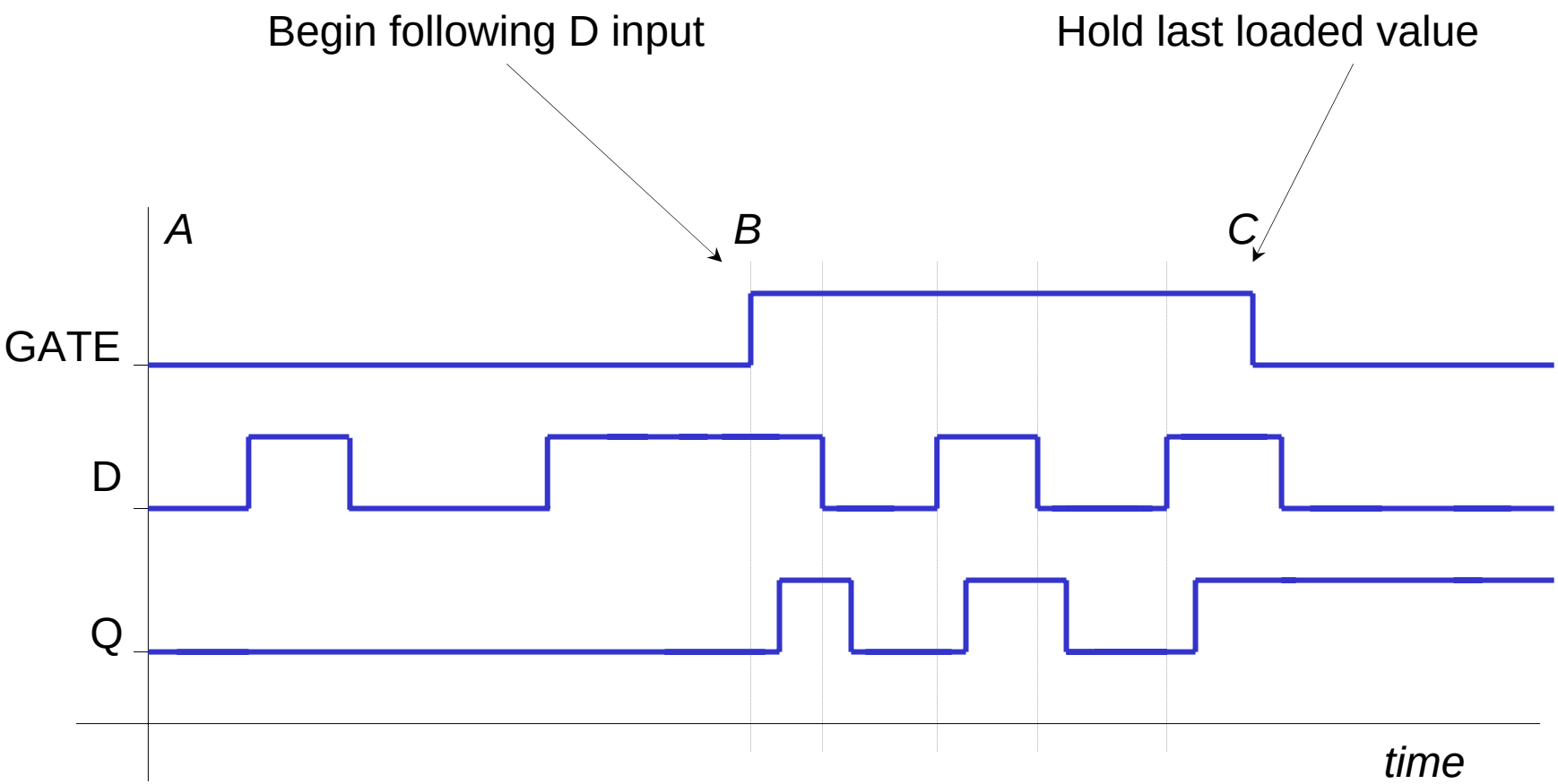


GATE	D	Q	Q+
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Q+ = GATE•D + GATE'•Q

When GATE='1' ⇔ Q follows D (storage)
When GATE='0' ⇔ Q retains old value (retention)

Gated D Latch - Timing

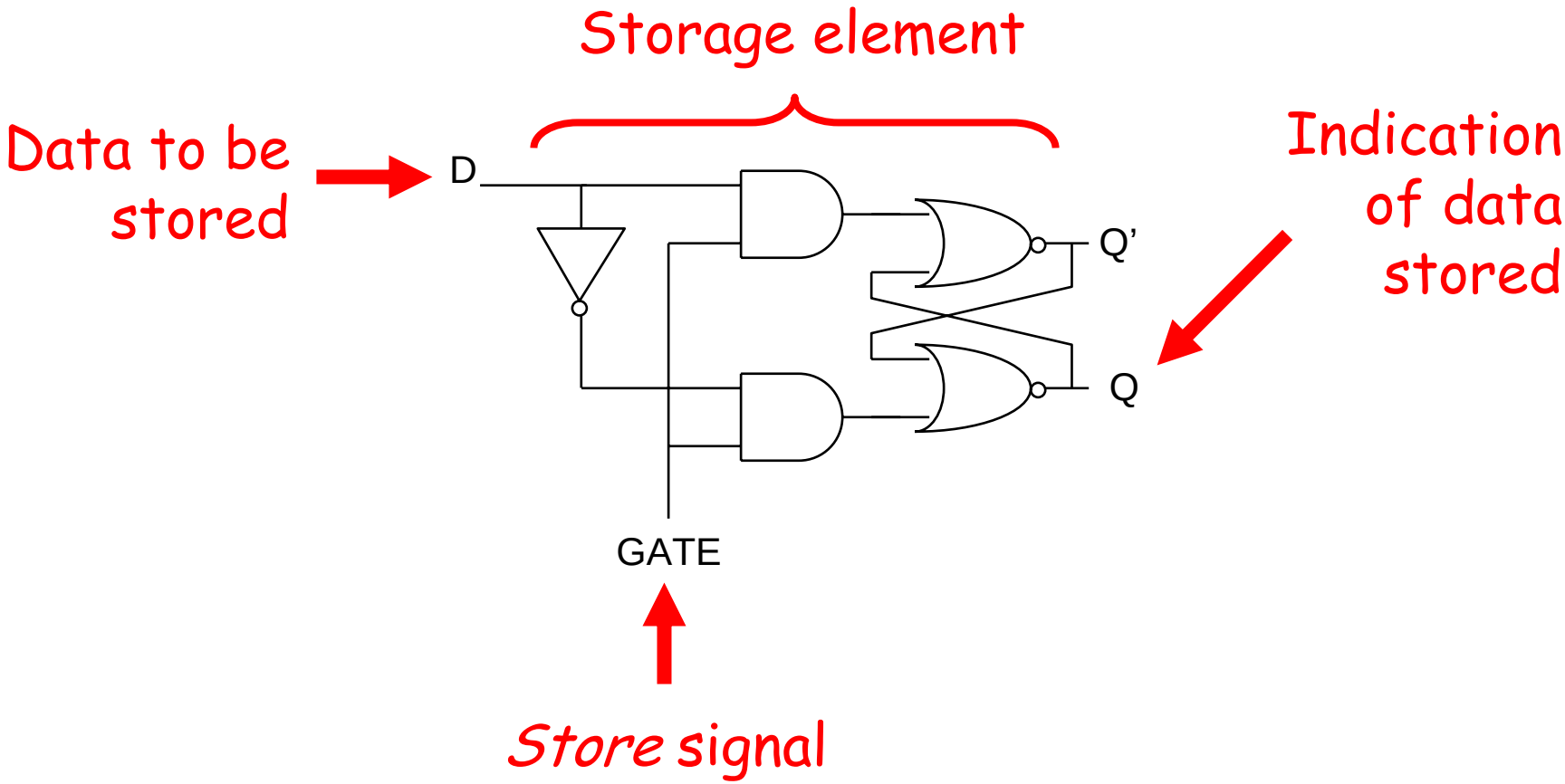


Gated D Latches

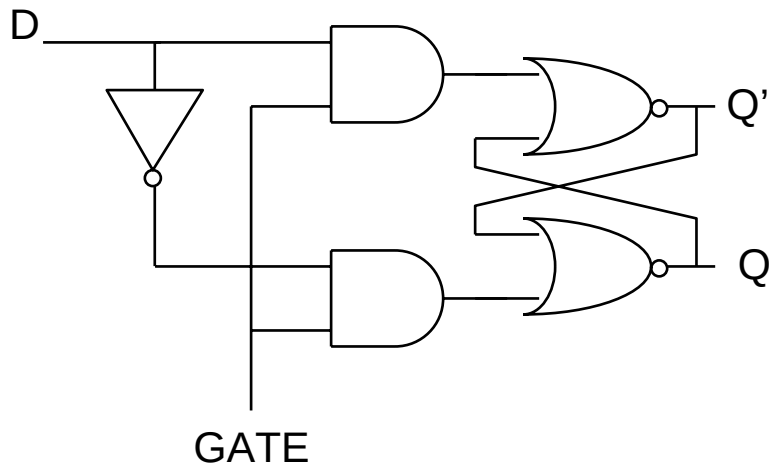
- Sometimes called a *transparent* latch
  - When GATE='1':
    - Q follows D
    - D is reflected on Q output

- Allows us to control *when* to store new data into latch
  - D = data to be stored
  - GATE = control signal

Gated D Latch – Example of Use



Gated D Latches - Delays



If GATE='1',

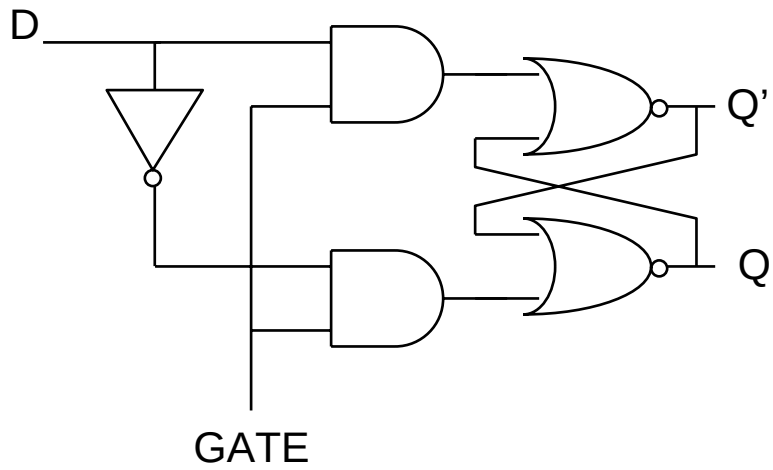
When D changes, Q/Q' will change after:

$t_{D \rightarrow Q} = t_{NOT}+t_{AND}+t_{NOR}+t_{NOR} = t_{NOT}+t_{AND}+2x\ t_{NOR}$

Why t_{NOT}?

Why 2 x t_{NOR} ??

Gated D Latches – More Delays



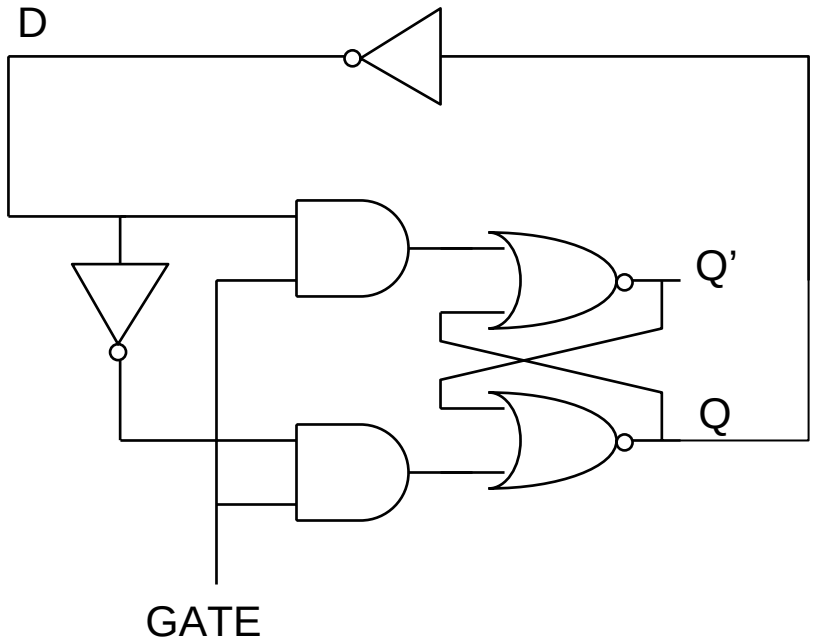
If D is constant:

When GATE ➔ '1', Q/Q' will change after:

$t_{GATE \rightarrow Q} = t_{AND}+t_{NOR}+t_{NOR} = t_{AND}+2x\ t_{NOR}$

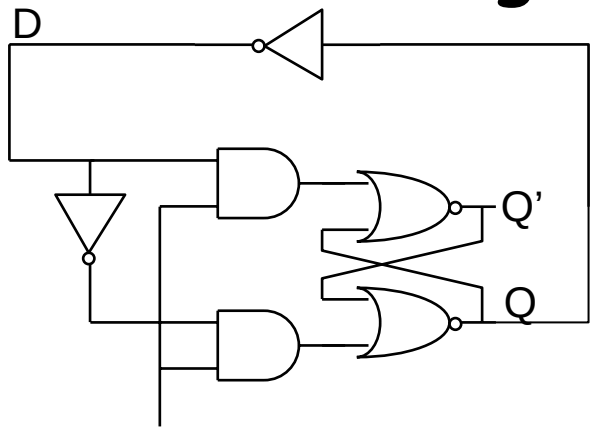
Why no t_{NOT}?

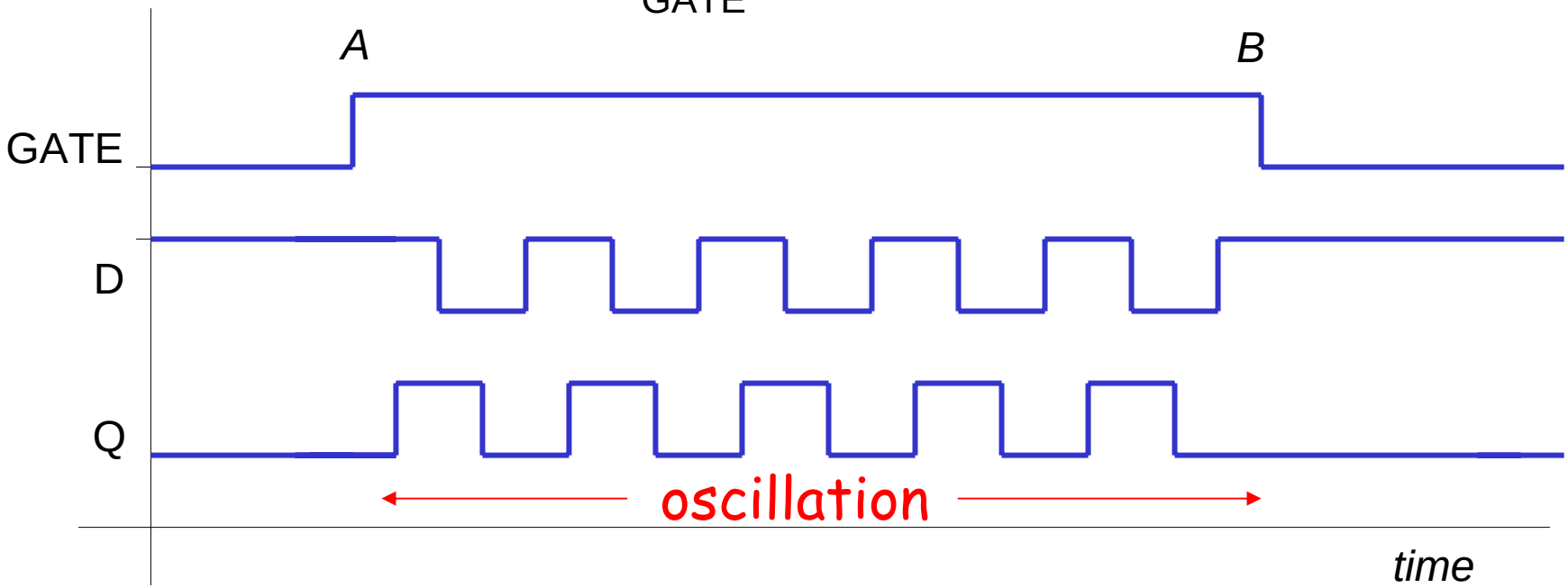
An Example Gated D Latch Circuit



When GATE ⟶ '1':
Latch loads Q' (toggles)
'0' ⟶ '1' or '1' ⟶ '0'

Toggle Circuit Timing Diagram



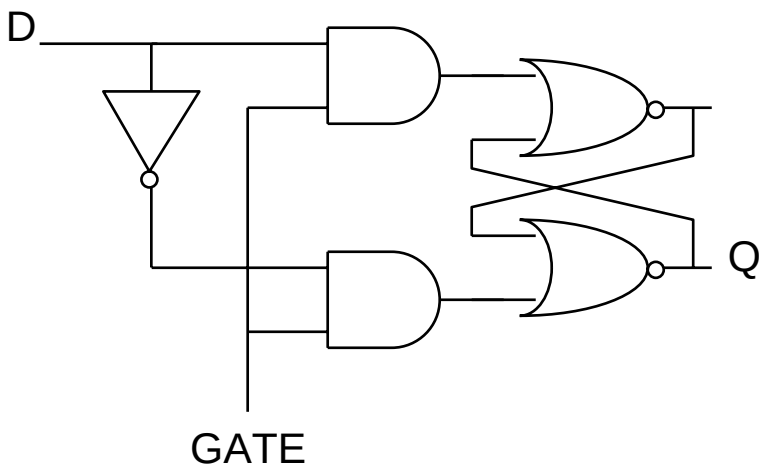


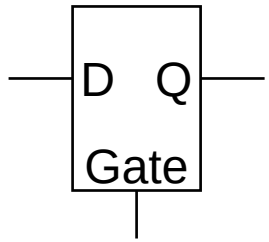
Toggle Circuit Problem

- As long as GATE='1'
  - Latch will repeatedly load new values
    0 → 1 → 0 → 1 → 0 → 1 …

- Solution #1:
  - Make GATE='1' for a very short time
  - Hard to do reliably in a LARGE system
- Solution #2:
  - Build a new storage element out of gated D latches (the flip flop) ⇔ next lecture

Symbology





or…