6 Flip-Flop

Devices Used in Digital Systems

Devices used in digital systems:

- Logical Devices:
  - Gate circuits
  - Combination circuits
- Memory Devices:
  - Flip-flops
  - Sequential Circuits

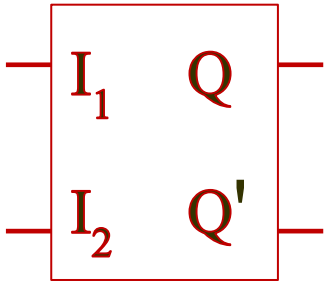
Flip-Flop

Flip-Flop

- is a device with a pair of inputs, and one output *Q* with its inverting *Q*'.
- We can program the output *Q* to be either 0 or 1;
- By activating the pair of inputs;
- The output *Q* could be hold its state until the new inputs are activated;

Flip-Flop can be used to make:

- Registers;
- Counters;
- Sequential Circuits.



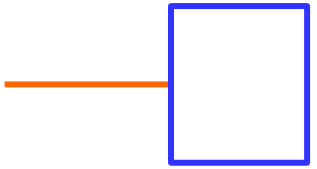
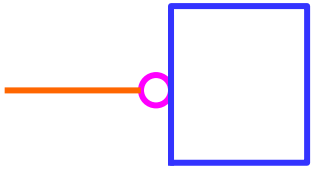
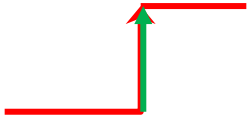
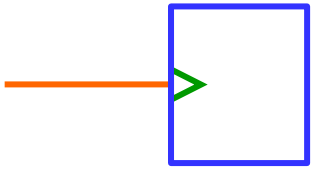
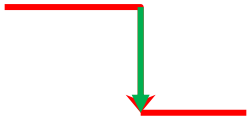
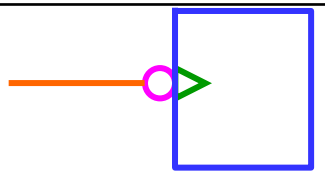
Flip-Flop: Types

Flip-Flop may be categorized by following characteristics

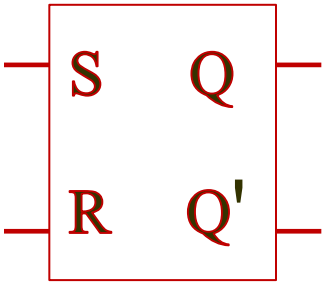
- Controlling signal can be divided into
  - Flip-Flop with Clock and
  - Flip-Flop without Clock.
- Input activation can be divided into 6 types:

- SR Flip-Flop
- D (Data) Flip-Flop
- Master/Slave Flip-Flop
- JK Flip-Flop
- T (Toggle) Flip-Flop
- Preset and Clear Flip-Flop

Flip-Flop Types: Clock Activation

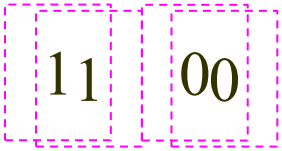
Types of Clock Activation		Symbols
	Positive **Level**	
	Negative **Level**	
	Positive **Edge** Triggered	
	Negative **Edge** Triggered	

Set-Reset Flip-Flop



SR Flip-Flop

- There are 2 inputs: Reset (R) and Set (S).
- **Setting** is to make the output to 1, and
- **Resetting** is to make the output to 0.



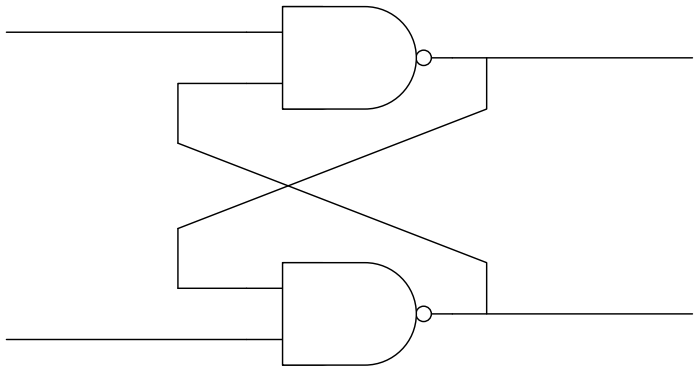
- **Setting** can get by activating the inputs: S = 1 and R = 0.
- **Resetting** can get by activating the inputs: S = 0 and R = 1.
- When BOTH inputs are **not** activated: S = 0 and R = 0,
  the output will remain its previous logic, or **NO CHANGE**.
- When BOTH inputs are activated: S = 1 and R = 1,
  the output will be ambiguous, so **UNUSEABLE**.

Set-Reset Flip-Flop: Excitation Table

Input		Output		Status
SET	$RESET$	Q	$\overline{Q}$	
0	0	Q	$\overline{Q}$	Unchanged State
0	1	0	1	Reset State
1	0	1	0	Set State
1	1			**Unused State**

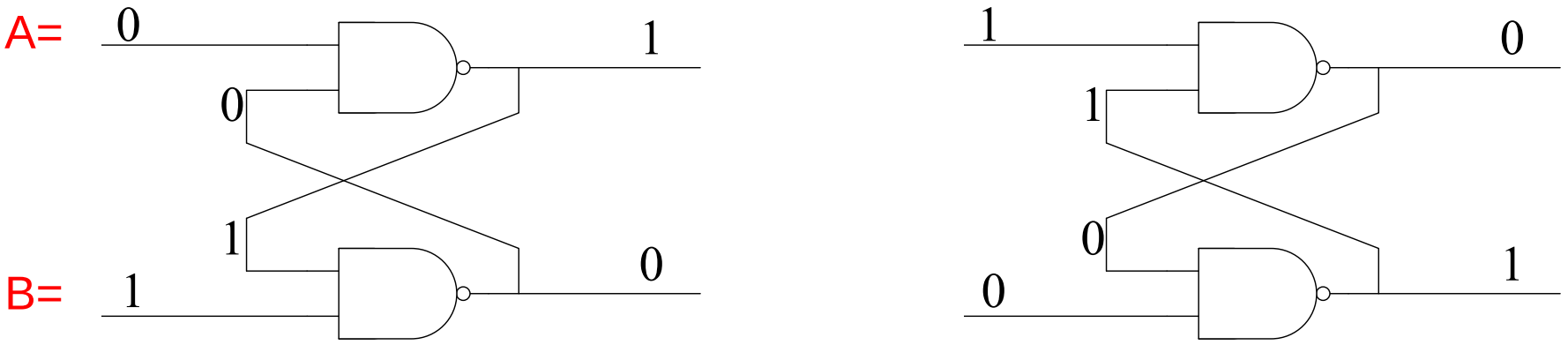
Set-Reset Flip-Flop: Configuration

It may be constructed from either 2 NAND gates or 2 NOR gates with one's output feeding to the other input;



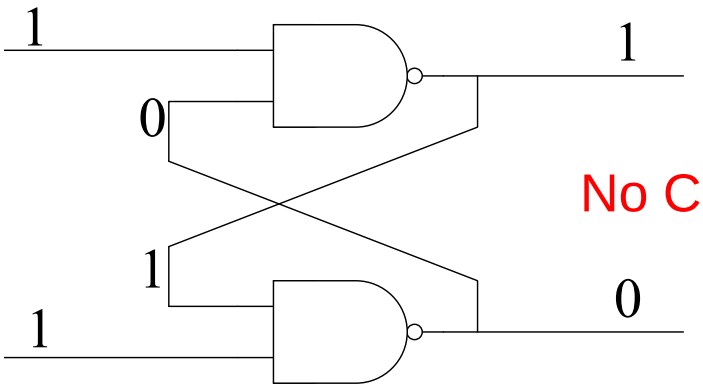
Let's see what happens when applying all possible inputs:

Crossed NAND $\overline{\text{Set}}$-$\overline{\text{Reset}}$ Flip-Flop [1]

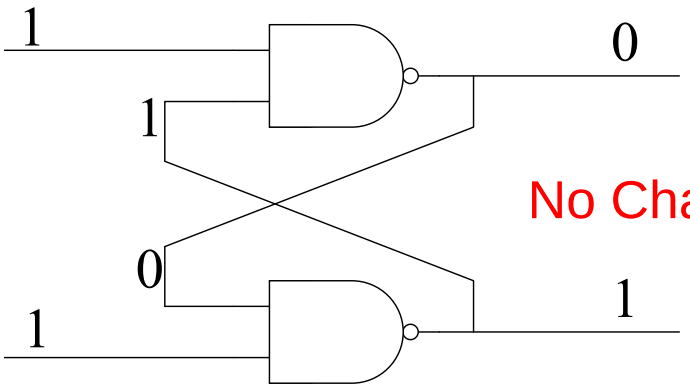


A	*B*	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Crossed NAND $\overline{\text{Set}}$-$\overline{\text{Reset}}$ Flip-Flop [2]



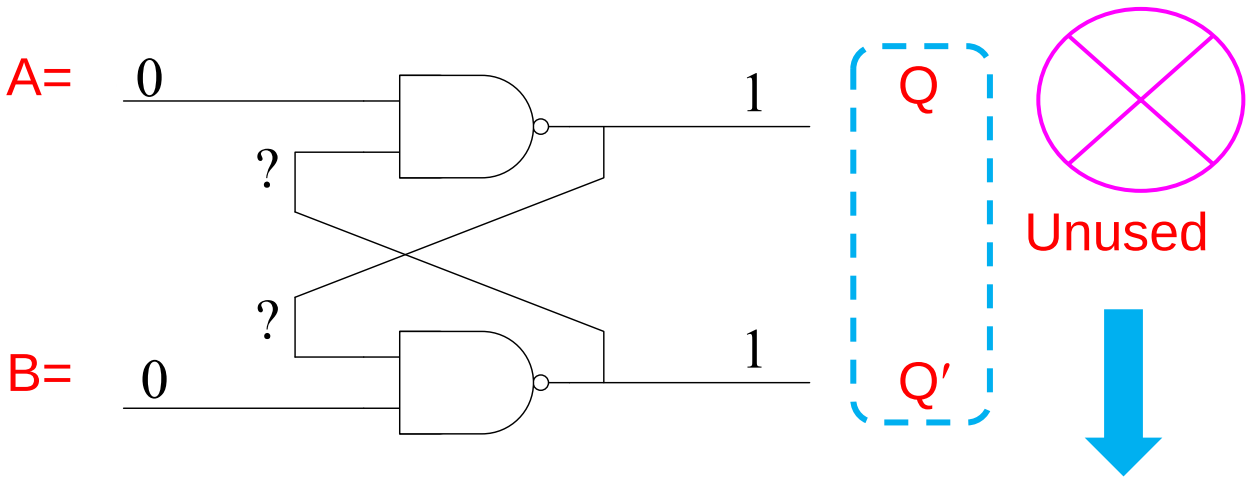
No Change



No Change

A	*B*	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Crossed NAND $\overline{\text{Set}}$-$\overline{\text{Reset}}$ Flip-Flop [3]



A	B	Upper
0	**0**	**1**
0	**1**	**1**
1	0	0
1	1	Q

Activated by 0

Unused

Set

Reset

Unchanged

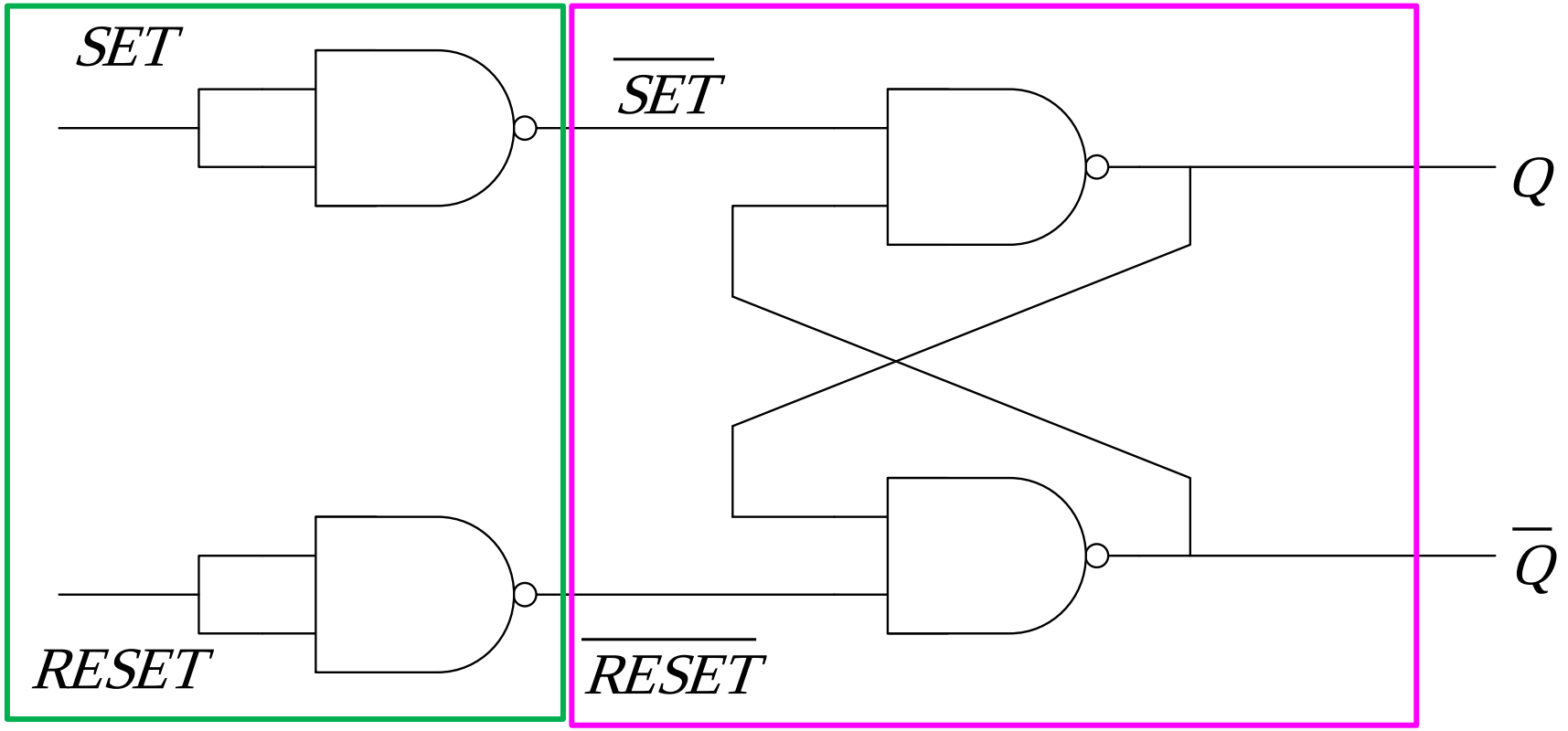
(Previous case)

It is an SR Flip-Flop with active low (0) inputs.

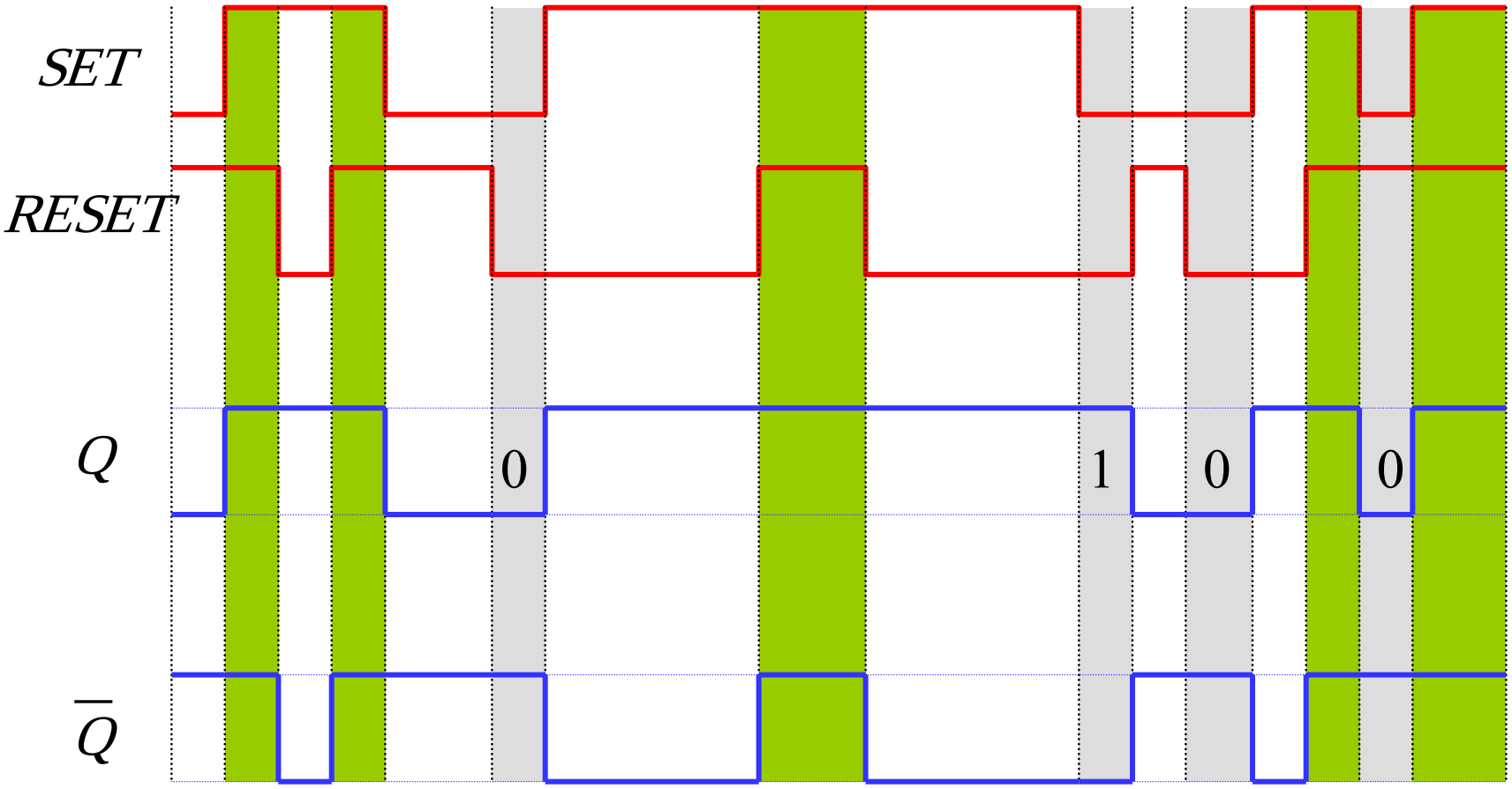
Crossed NAND $\overline{\text{Set}}$-$\overline{\text{Reset}}$ Flip-Flop [4]

SET	$RESET$	$\overline{SET}$	$\overline{RESET}$	Q	$\overline{Q}$	
1	1	0	0	1	1	Unused State
1	0	0	1	1	0	Set State
0	1	1	0	0	1	Reset State
0	0	1	1	Q	$\overline{Q}$	Unchanged State

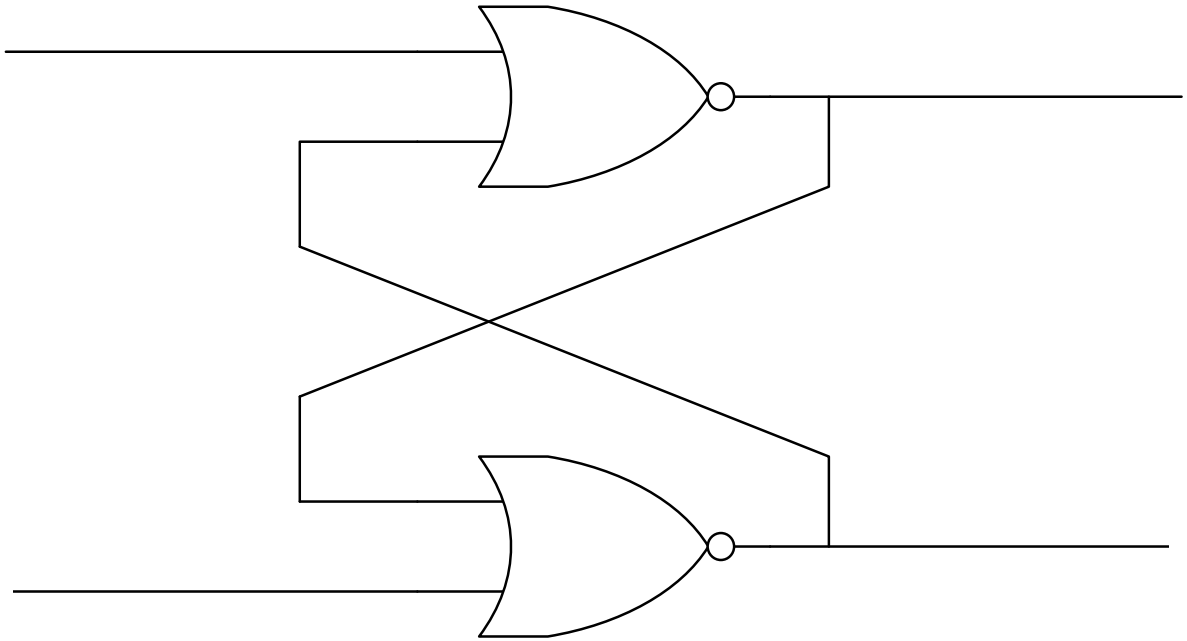
Crossed NAND Set-Reset Flip-Flop [5]



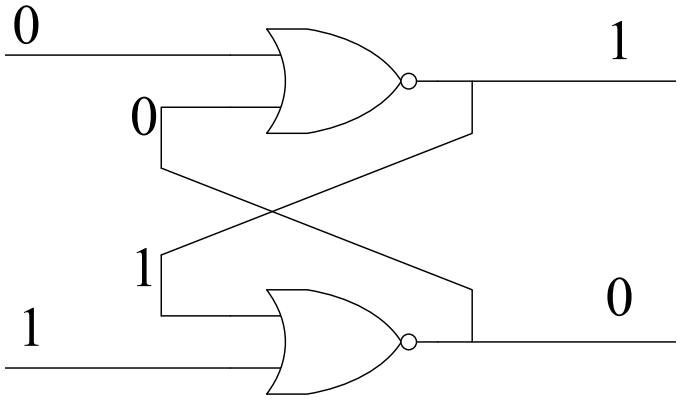
Crossed NAND Set-Reset Flip-Flop [6]

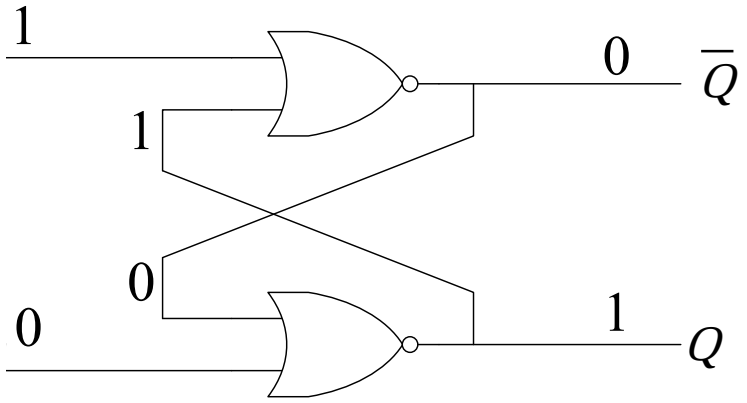


Crossed NOR Set-Reset Flip-Flop [1]



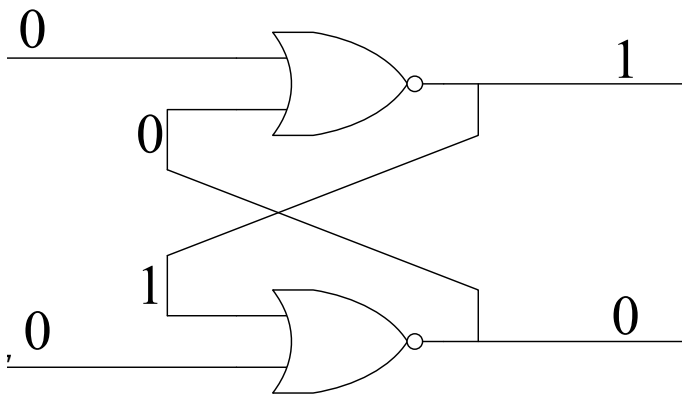
Crossed NOR Set-Reset Flip-Flop [2]

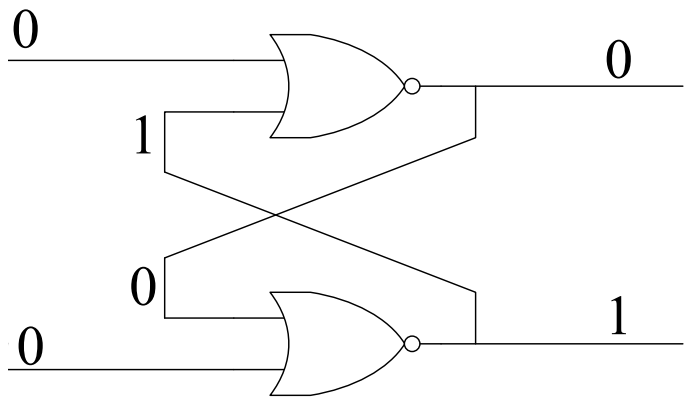




A	B	$\overline{A+B}$
0	0	1
0	1	0
1	**0**	**0**
1	**1**	**0**

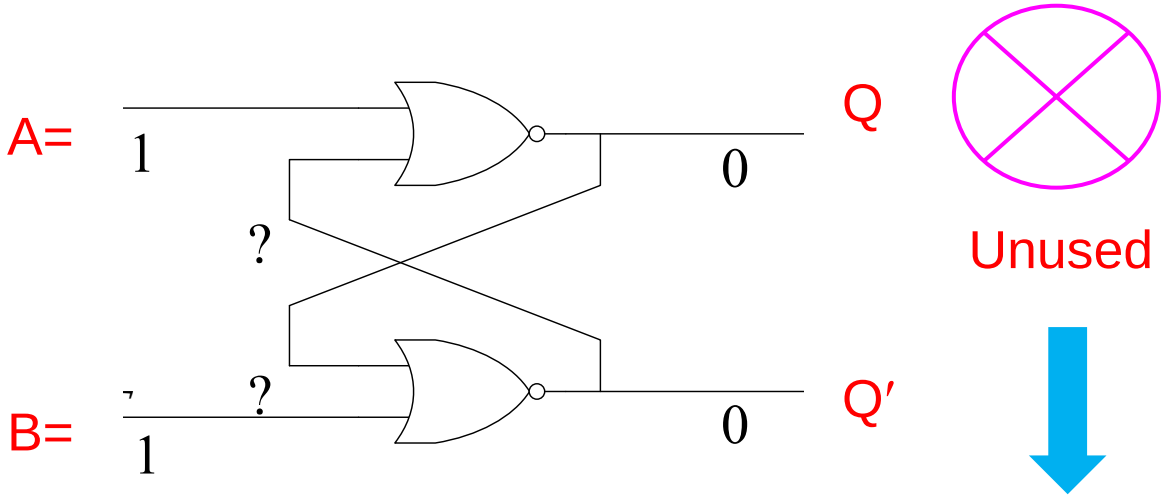
Crossed NOR Set-Reset Flip-Flop [3]





A	B	$\overline{A+B}$
0	0	1
0	1	0
1	**0**	**0**
1	**1**	**0**

Crossed NOR Set-Reset Flip-Flop [4]

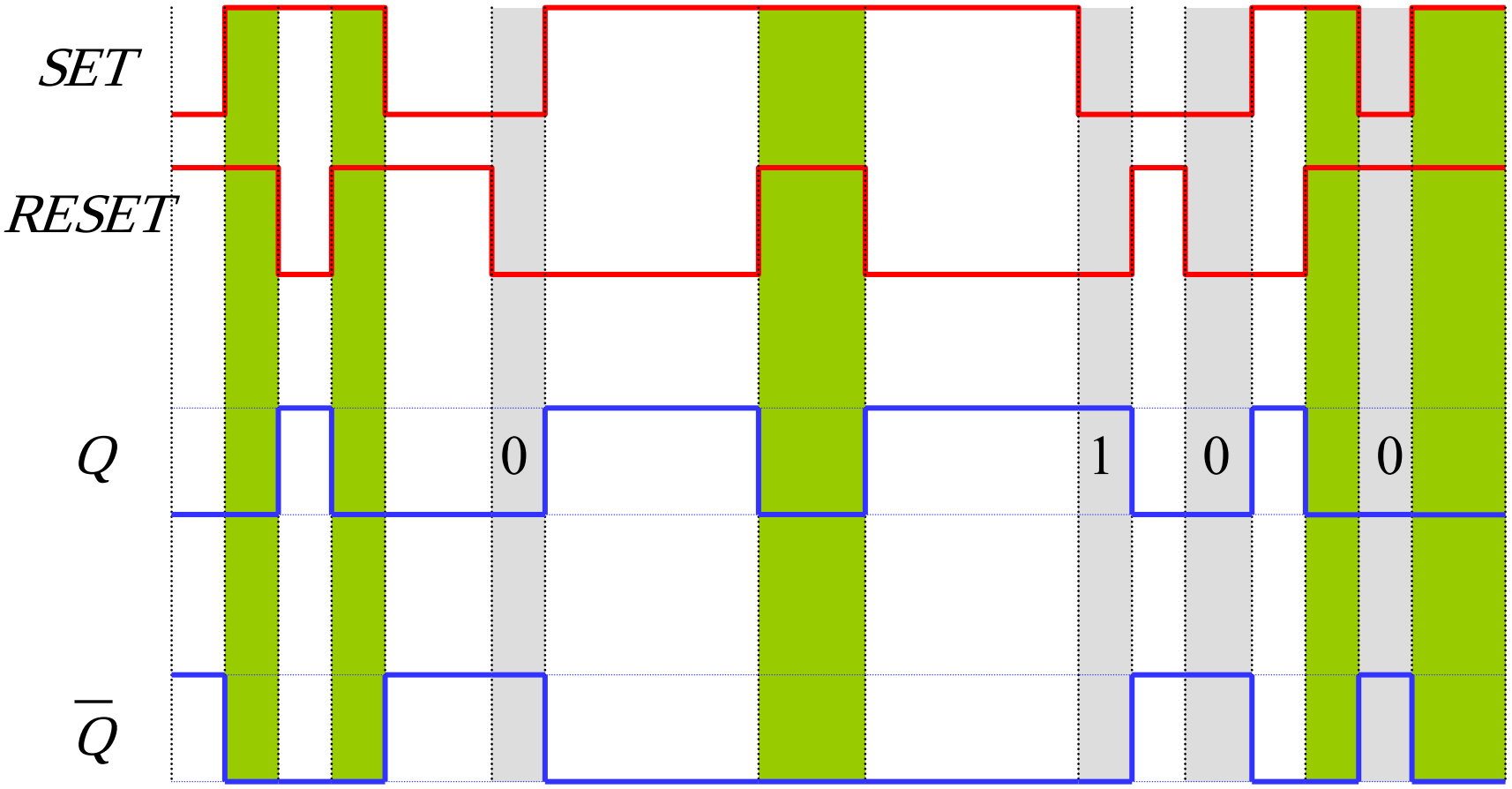


A	B	Upper
0	0	Q
0	1	1
1	**0**	**0**
1	**1**	**0**

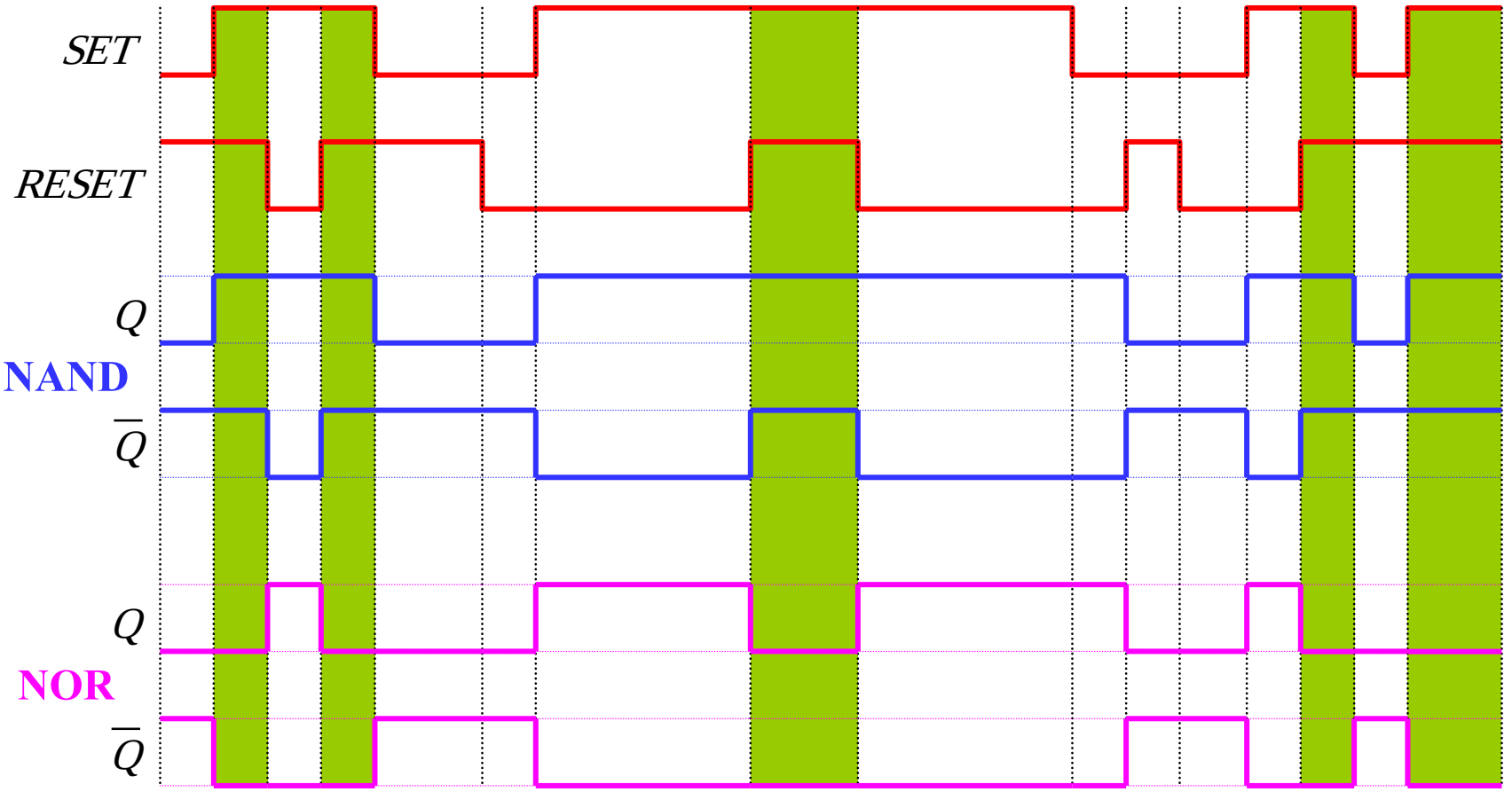
Crossed NOR Set-Reset Flip-Flop [5]

Set	Reset	Q	$\overline{Q}$	State
1	1	0	0	Unused
1	0	1	0	Set
0	1	0	1	Reset
0	0	Q	$\overline{Q}$	Unchanged

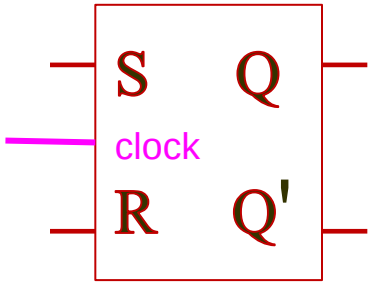
Crossed NOR Set-Reset Flip-Flop [6]



Waveform Crossed SR Flip-Flop



Gated Set-Reset Flip-Flop



- Sometimes called Clocked SR Flip-Flop.
- With a clock used to activate the Flip-Flop operation.

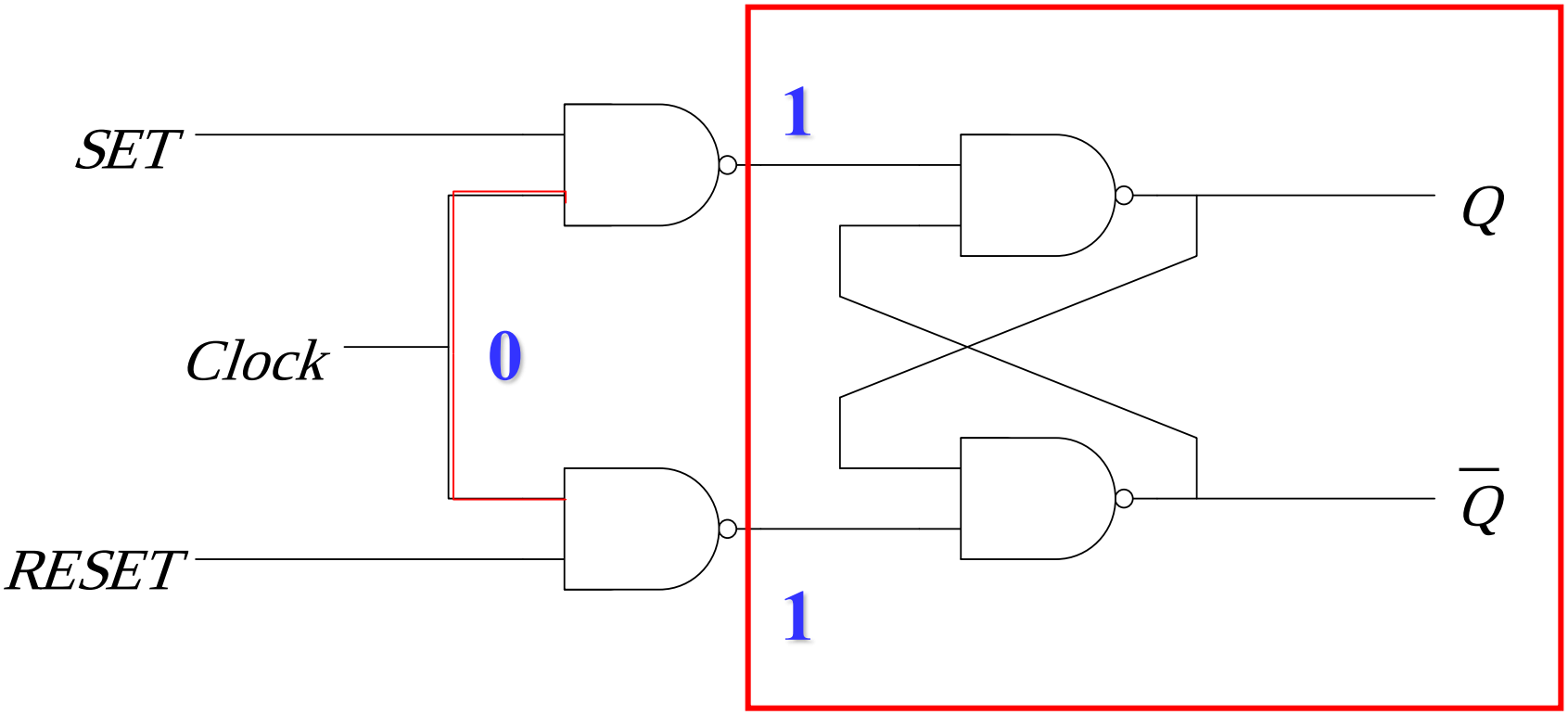
➢ When no clock,

- The flip floppy will not work,
- The output value will maintain the previous logic state.

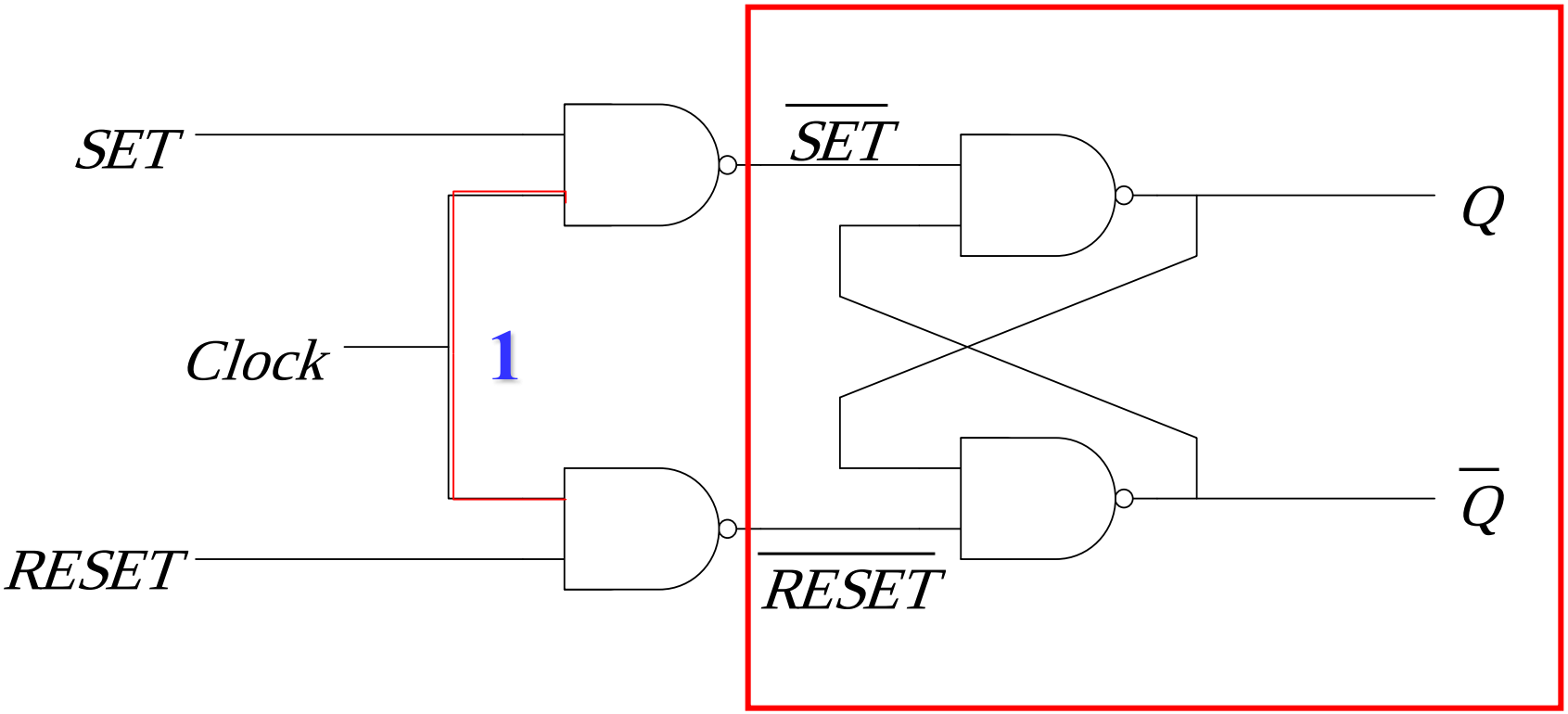
➢ Presenting with a clock,

- It would operate as the normal SR Flip-Flop;
- Depends on the activation at S and R.

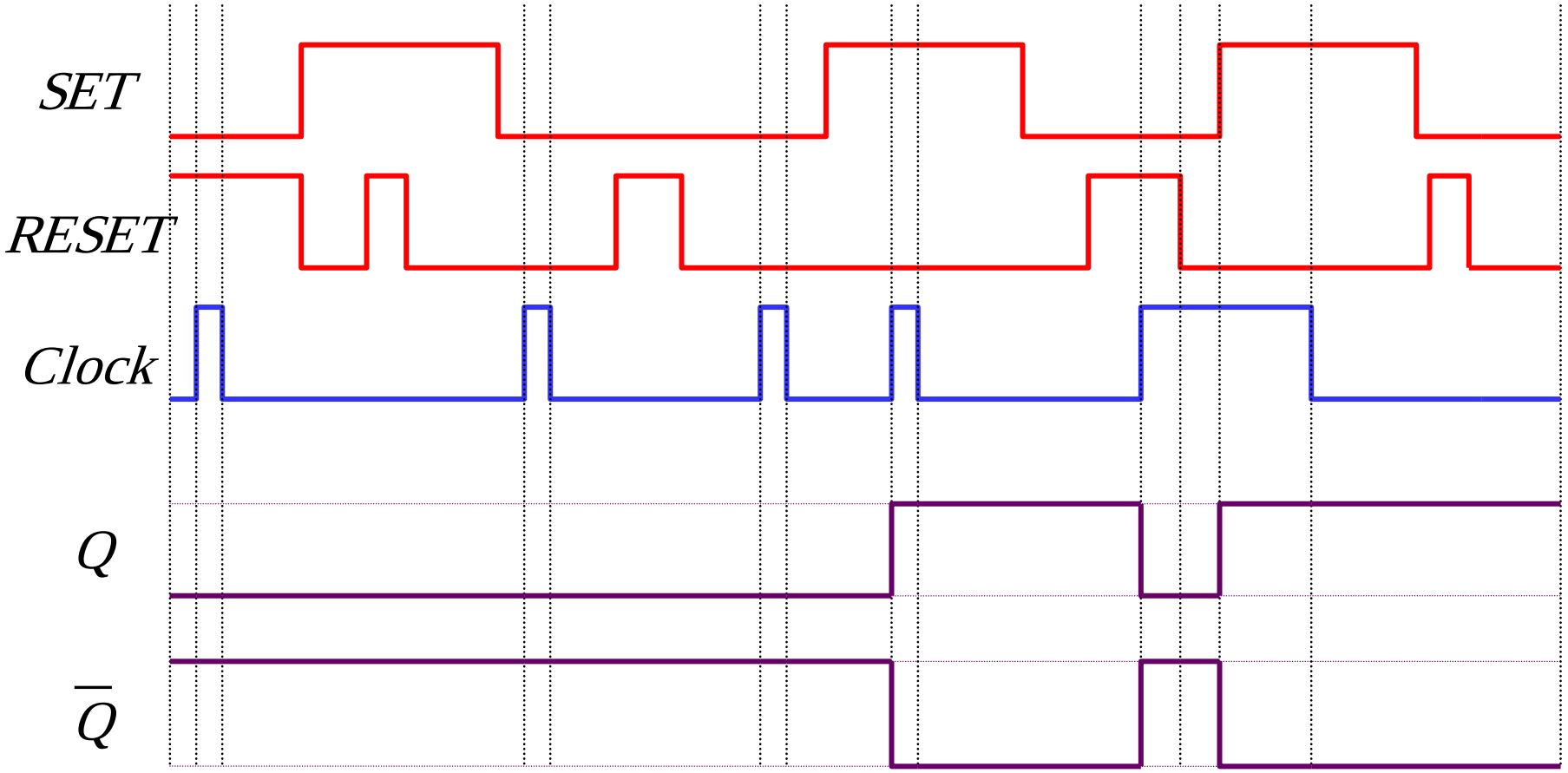
Gated Set-Reset Flip-Flop



Gated Set-Reset Flip-Flop



Gated Set-Reset Flip-Flop



Gated Set-Reset Flip-Flop

Clock	*SET*	*RESET*	Q	$\overline{Q}$	Status
———	0	0	Q	$\overline{Q}$	Unchanged
———	0	1	Q	$\overline{Q}$	
———	1	0	Q	$\overline{Q}$	
———	1	1	Q	$\overline{Q}$	
	0	0	Q	$\overline{Q}$	
	0	1	0	1	Reset
	1	0	1	0	Set
	1	1	1	1	Unused

JK Flip-Flop

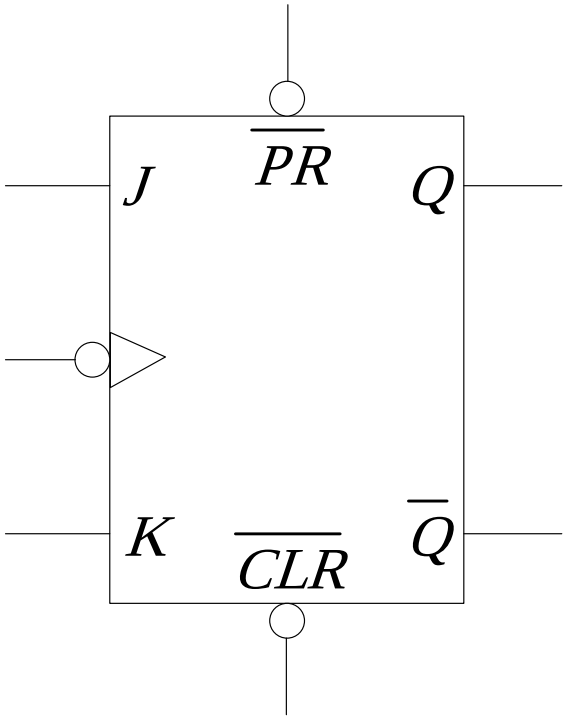
- JK Flip-Flop is the flip flop modified from the SR Flip-Flop.
- Modify the case for both inputs activated or the unused case in the SR Flip-Flop.
- The JK Flip-Flop circuit has 2 inputs: *J* and *K*.
- By comparing the input signals *J* as *S*, and *K* as *R*.
- In the case of the modified state with the inputs of $J = 1$ and $K = 1$.
- Call the state as the Toggle or Complement state.
- The operation will give the complemented output i.e.
- If the present $Q = 0$, it will change to the new state of $Q = 1$;
- if the present $Q = 1$, it will change to the new state of $Q = 0$.

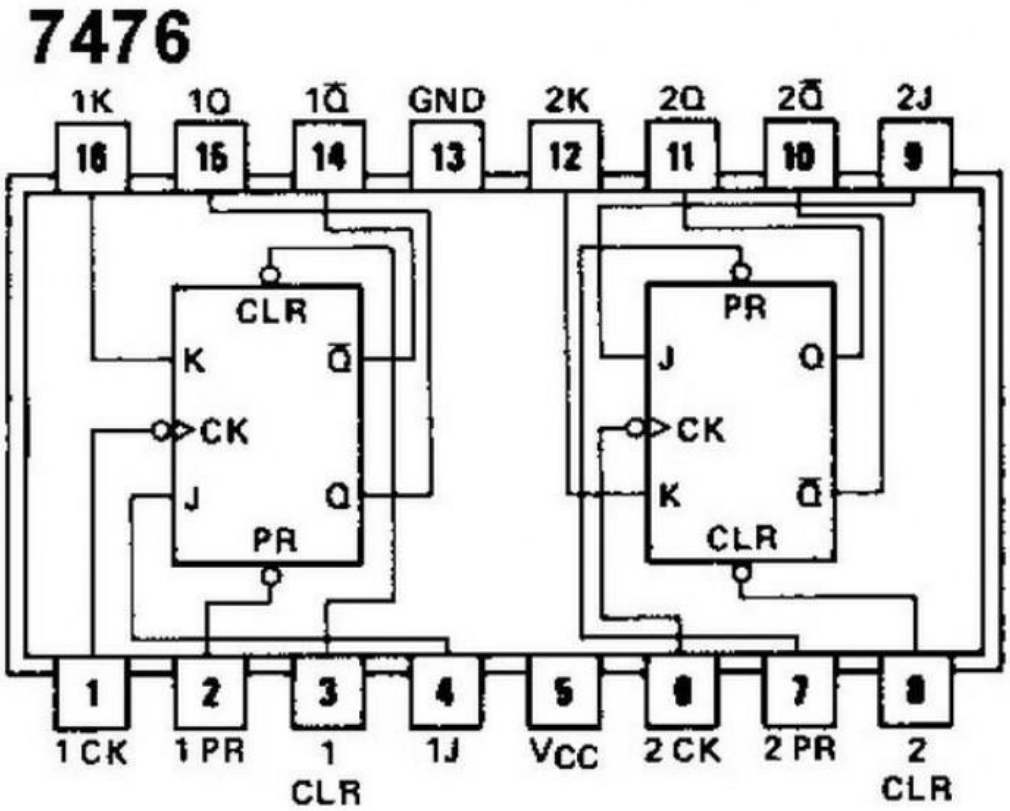
JK Flip-Flop

- In practice, The JK Flip-Flop will have 2 additional control inputs:
  - Preset (PR) and
  - Clear (CLR)
- Which can control the operation of the Flip-Flop
- independent to the activation at the input *J, K* and Clock;
- Both controlling pins are 'active low'.
- The operation table is presented below:

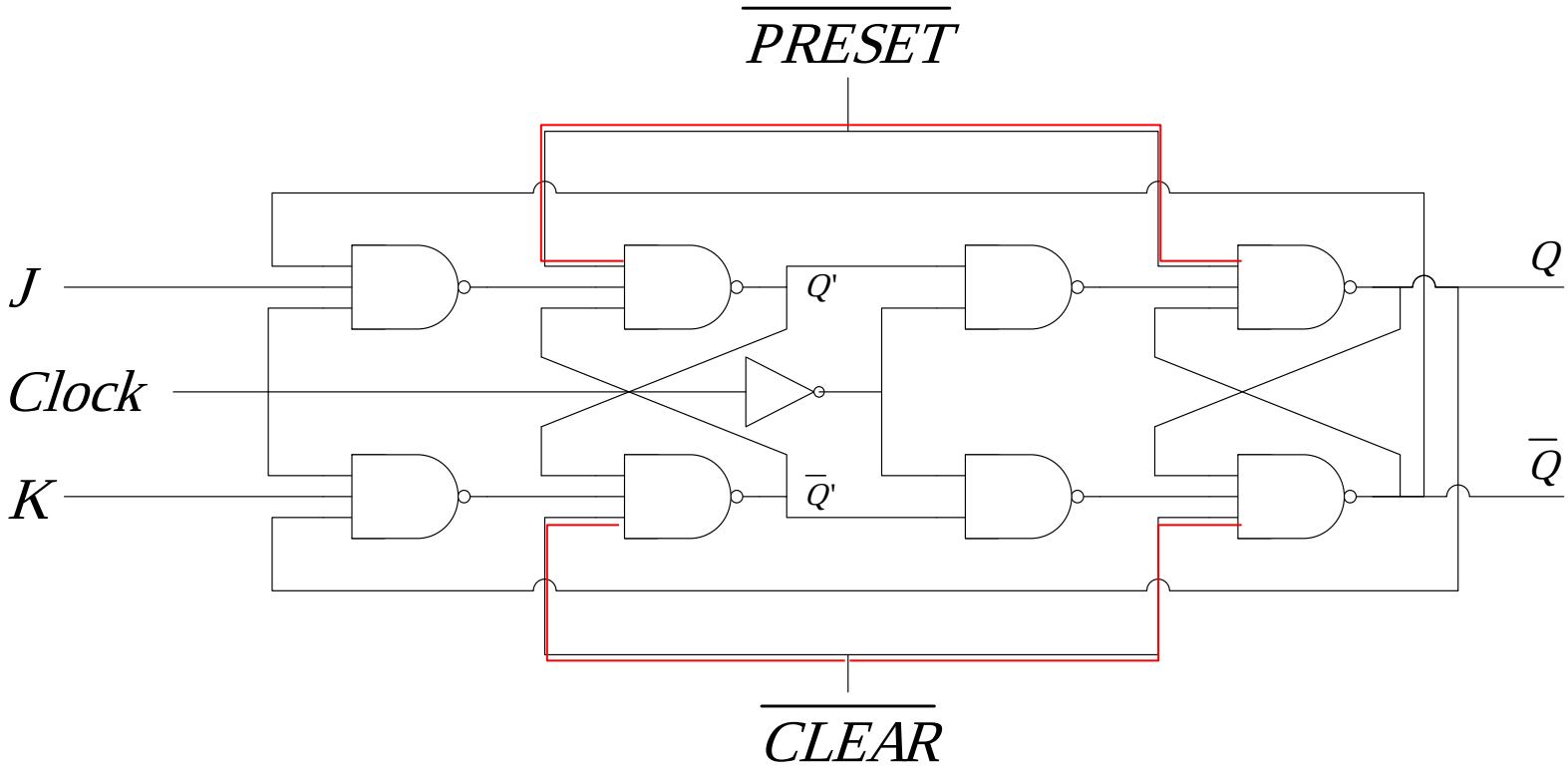
$\overline{\text{Preset}}$	$\overline{\text{Clear}}$	J	K	Clk	Q	$\overline{Q}$	State
0	1	X	X	X	1	0	Set
1	0	X	X	X	0	1	Reset
0	0	X	X	X	1	1	Unused State
1	1	0	1	↓	0	1	
1	1	1	0	↓	1	0	
1	1	0	0	X	Q	$\overline{Q}$	Unchanged
1	1	1	1	↓	Complemented		Toggle

JK Flip-Flop

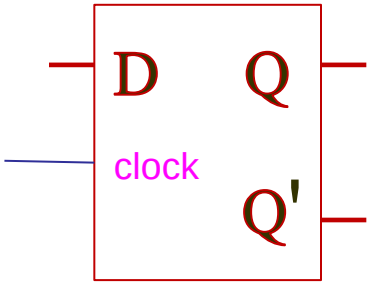

J
$\overline{PR}$
Q
K
$\overline{CLR}$
$\overline{Q}$


7476
1K
1Q
1$\overline{Q}$
GND
2K
2Q
2$\overline{Q}$
2J
16
15
14
13
12
11
10
9
CLR
K
$\overline{Q}$
CK
J
Q
PR
PR
J
Q
CK
K
$\overline{Q}$
CLR
1
2
3
4
5
6
7
8
1 CK
1 PR
1 CLR
1J
Vcc
2 CK
2 PR
2 CLR

JK Flip-Flop

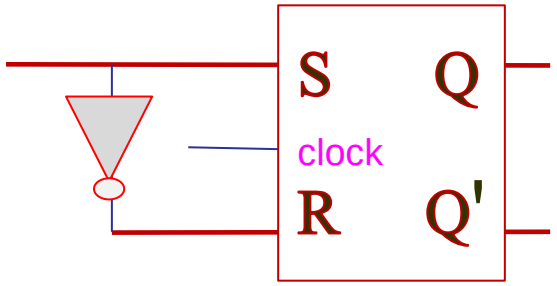


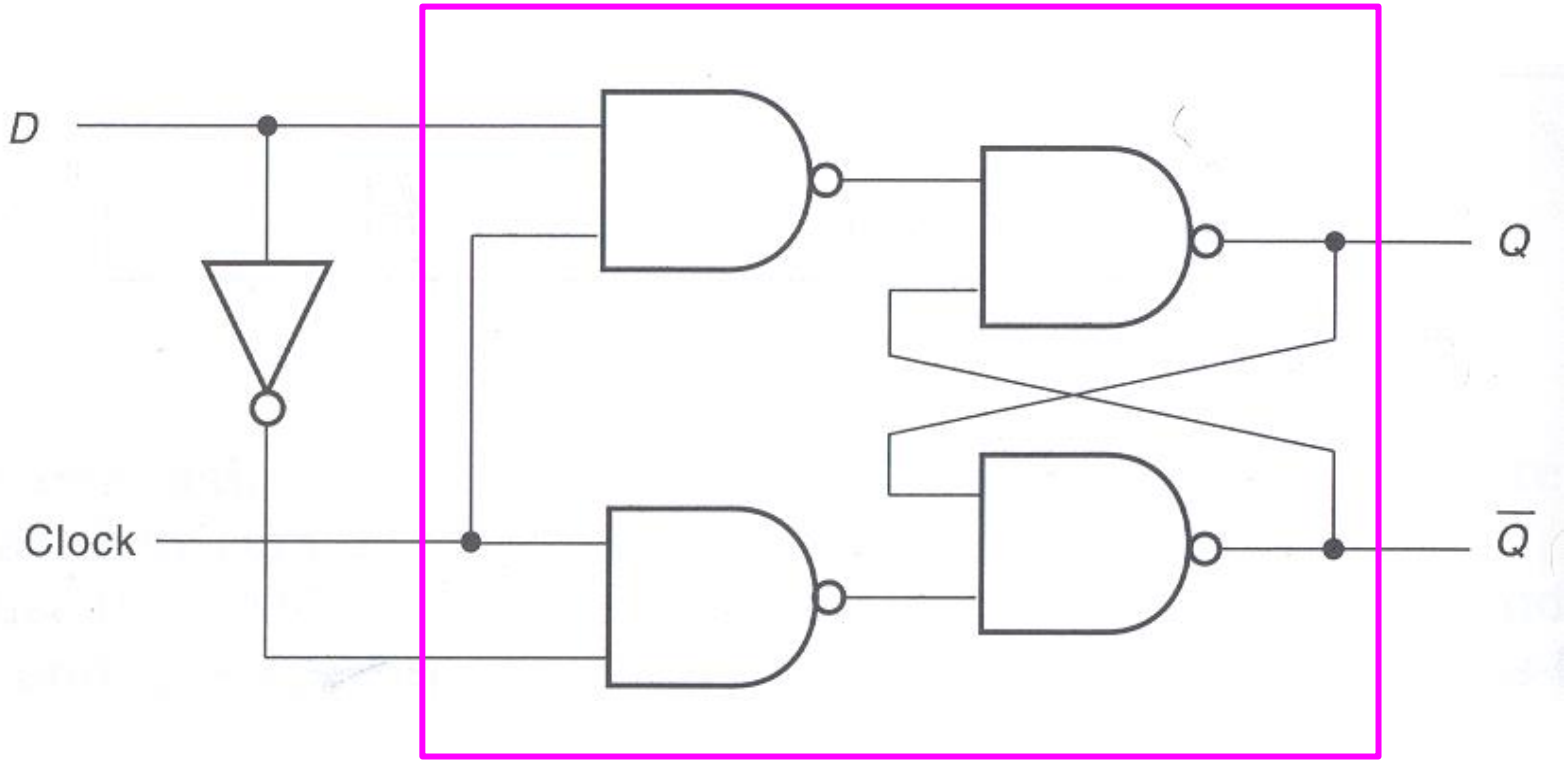
D (Data) Flip-Flop



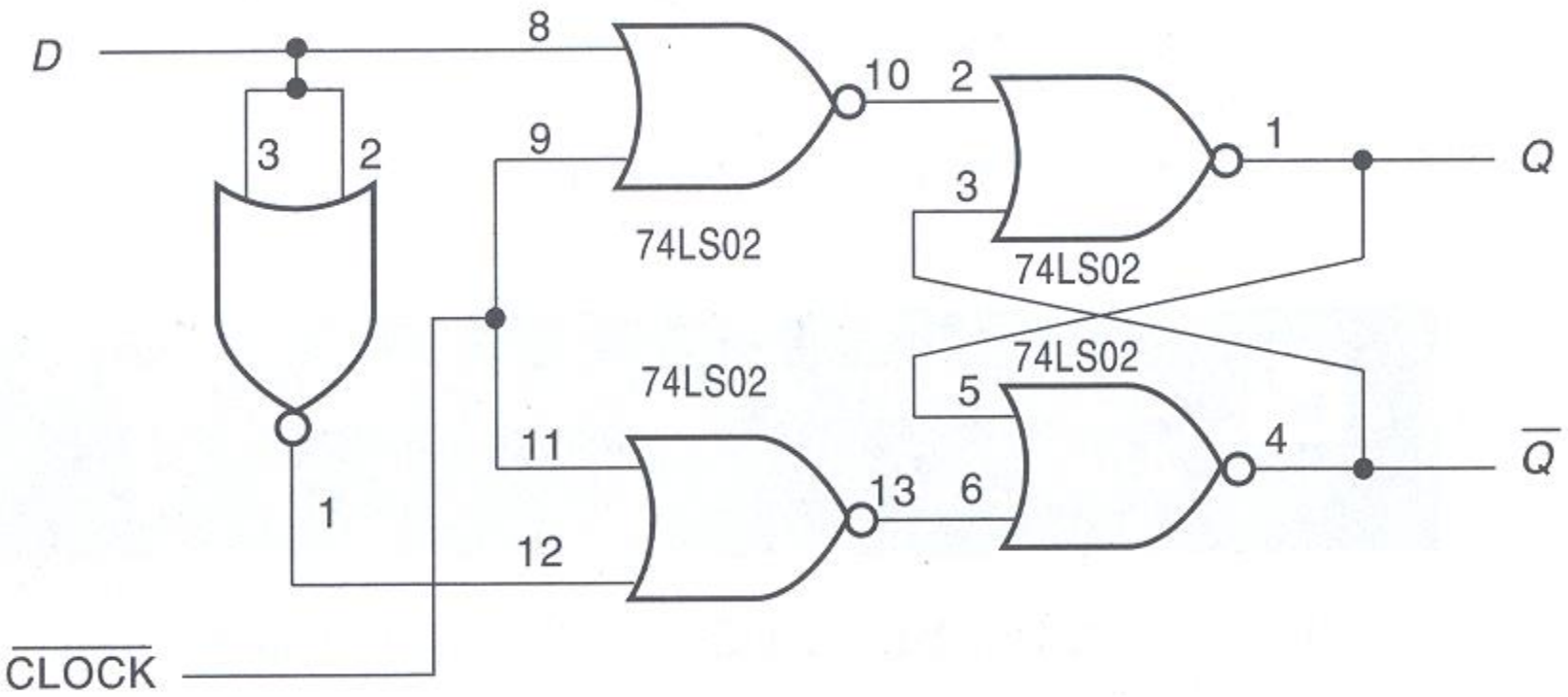
- is a flip-flop with only **one** input *D*;
- Work with the controlling signal – the Clock signal.
- When the circuit is triggered by a clock, the input D would be passed to output:
- when *D* = 0 gives *Q* = 0;
- And if *D* = 1, then *Q* = 1.
- Therefore, D Flip-Flops is used to memorize the input values..
- This principle applies to memorize the desired information.

D (Data) Flip-Flop

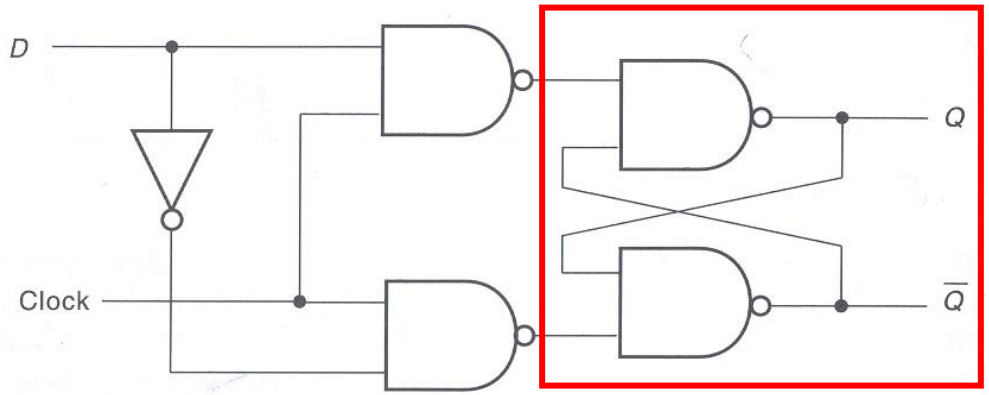




D Flip-Flop using ONLY NOR Gates



D Flip-Flop with CLOCK Activation



D	Clock	*Q*	$\overline{Q}$	Status
0	0	*Q*	$\overline{Q}$	Unchanged
1	0	*Q*	$\overline{Q}$	
0	1	0	1	Data Flip-flop
1	1	1	0	

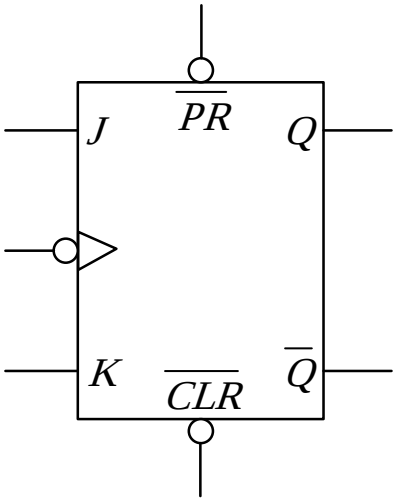
D Flip-Flop from JK Flip-Flop

- D Flip-Flops can be configured from JK Flip-Flops;
- By use the *J* input as the *D* input;
- And putting an inverter between the *J* and *K*;
- So, the *J* and *K* inputs will always be opposite logic.

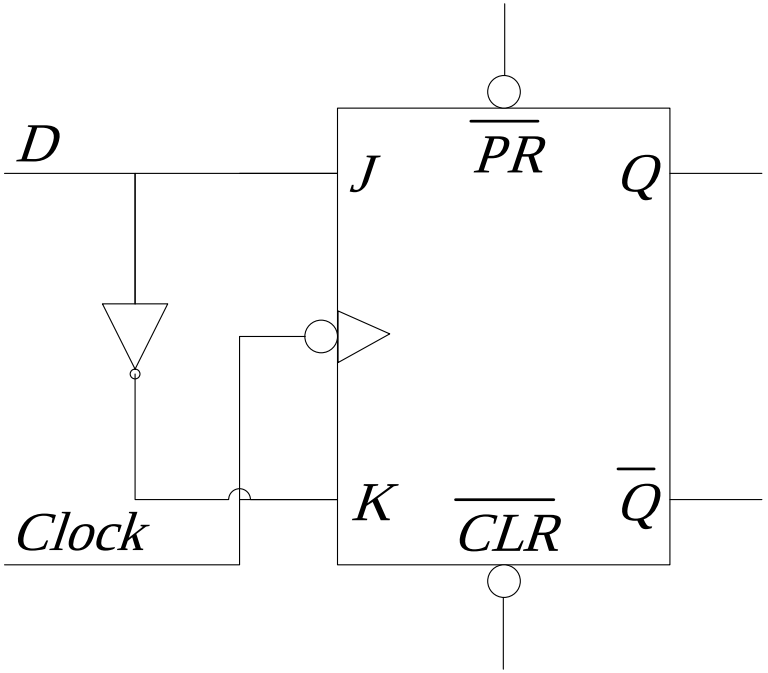
➢ If $D = J = 1$ and ($K = 0$), the output will be 1;

➢ If $D = J = 0$ and ($K = 1$), the output will be 0;

D Flip-Flop from JK Flip-Flop



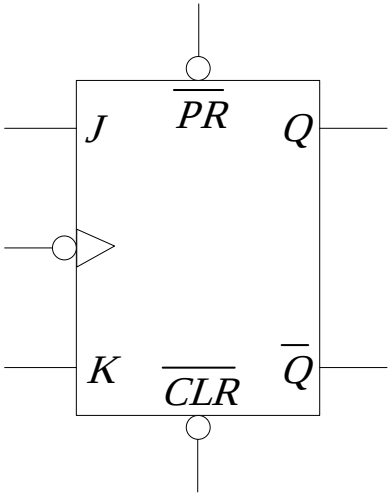
J	K	Q	$\overline{Q}$
0	0	Q	$\overline{Q}$
0	1	0	1
1	0	1	0
1	1	Toggle	



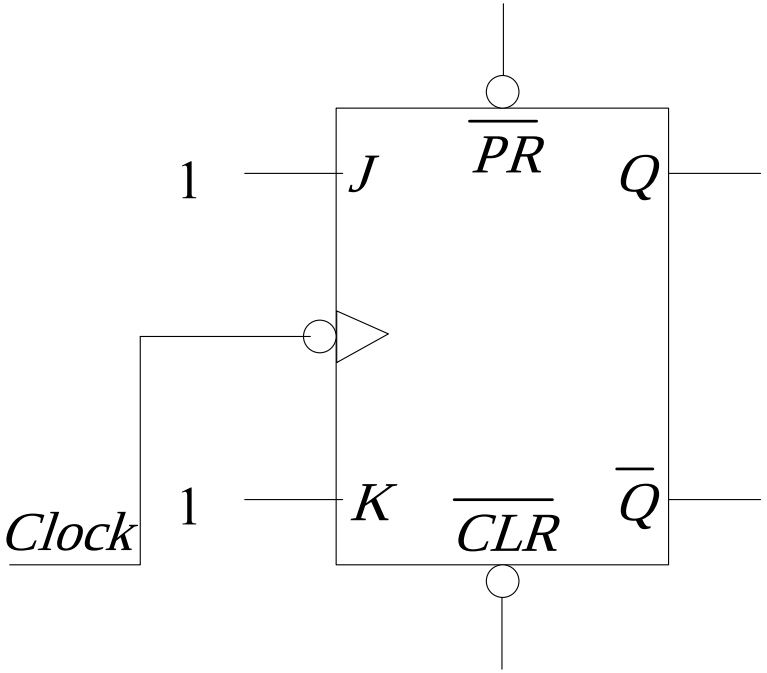
T (Toggle) Flip-Flop

- T Flip-Flop is another type of flip flop used in counting circuits.
- T Flip-Flop can be made from JK flip flop by
  - Applying 1 to both *J* and *K*.
  - And active the toggle by triggering from a clock;
  - The output will be inverted or opposite to the present logic.

T (Toggle) Flip-Flop

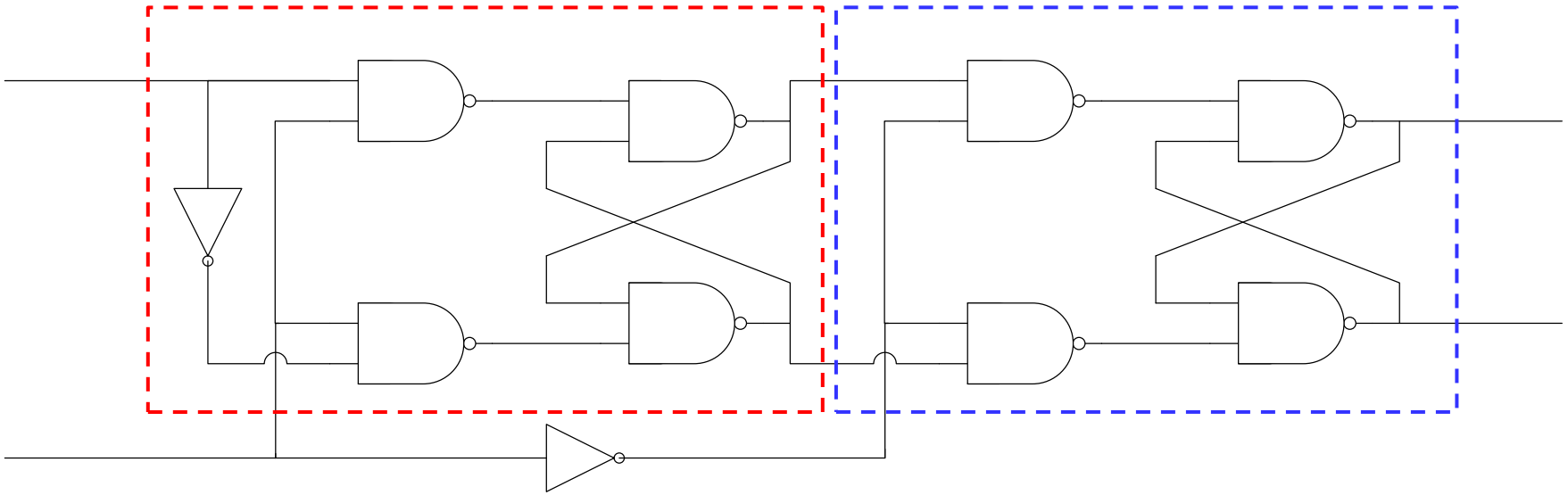


J	K	Q	$\overline{Q}$
0	0	Q	$\overline{Q}$
0	1	0	1
1	0	1	0
1	1	**Toggle**	



Master-Slave Flip-Flop

Master-Slave Flip-Flop is a pair of flip-flops as illustrated.



Master-Slave Flip-Flops may be made from

- D Flip-Flops
- JK Flip-Flop

Master-Slave Flip-Flop

At the rising edge of the clock and during the clock is high:

- The first Flip-Flop works by passing the input *D* to its output.
- The second Flip-Flop is not activated since its clock is 0.

At the falling edge of the clock:

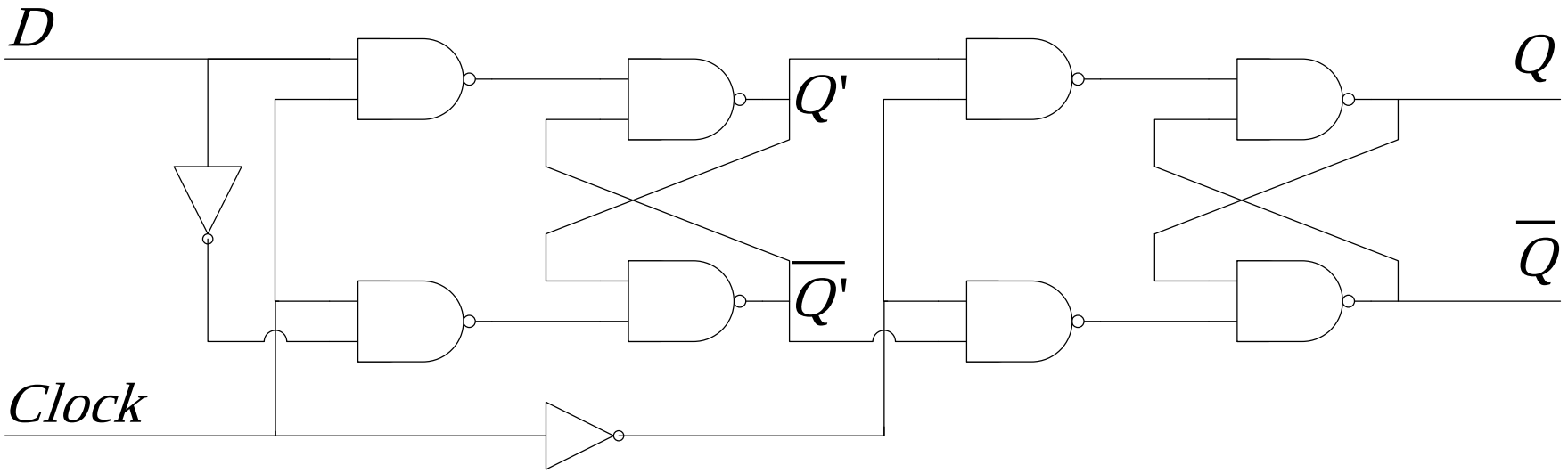
- The first Flip-Flop is not activated since its clock is 0.
- The second Flip-Flop works by passing the input *D* to its output.

➔ Negative Edge Triggered Data Transfer.

➔ Use tiny time at the negative edge for the data manipulation;

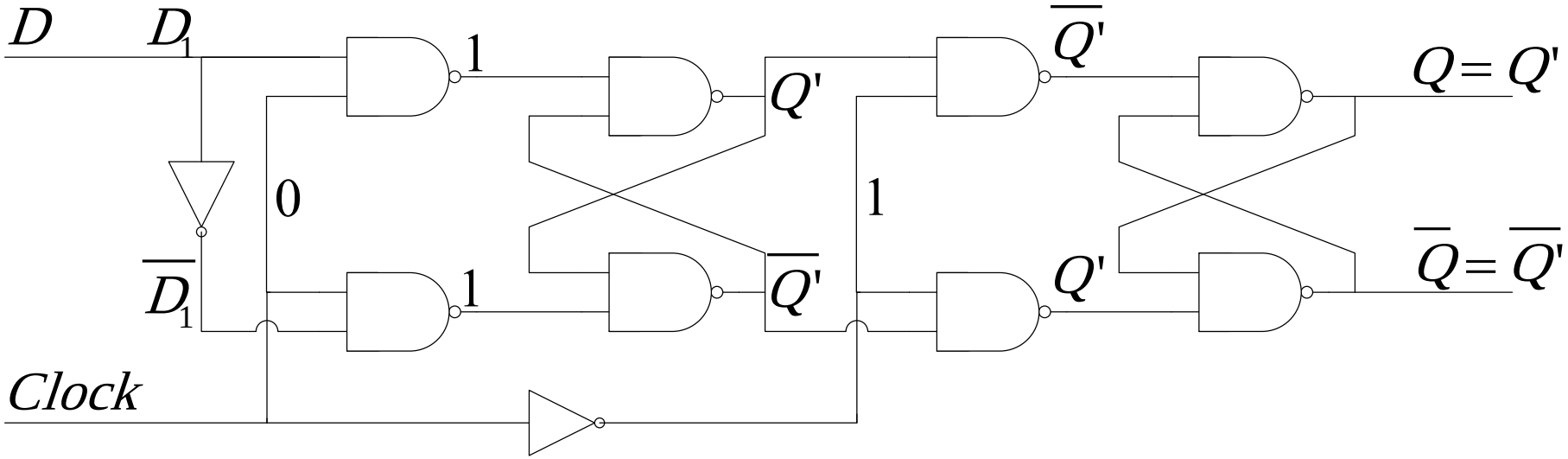
➔ Fast processing.

Master-Slave D Flip-Flop



Negative Edge-Triggered Master-Slave D Flip-Flop

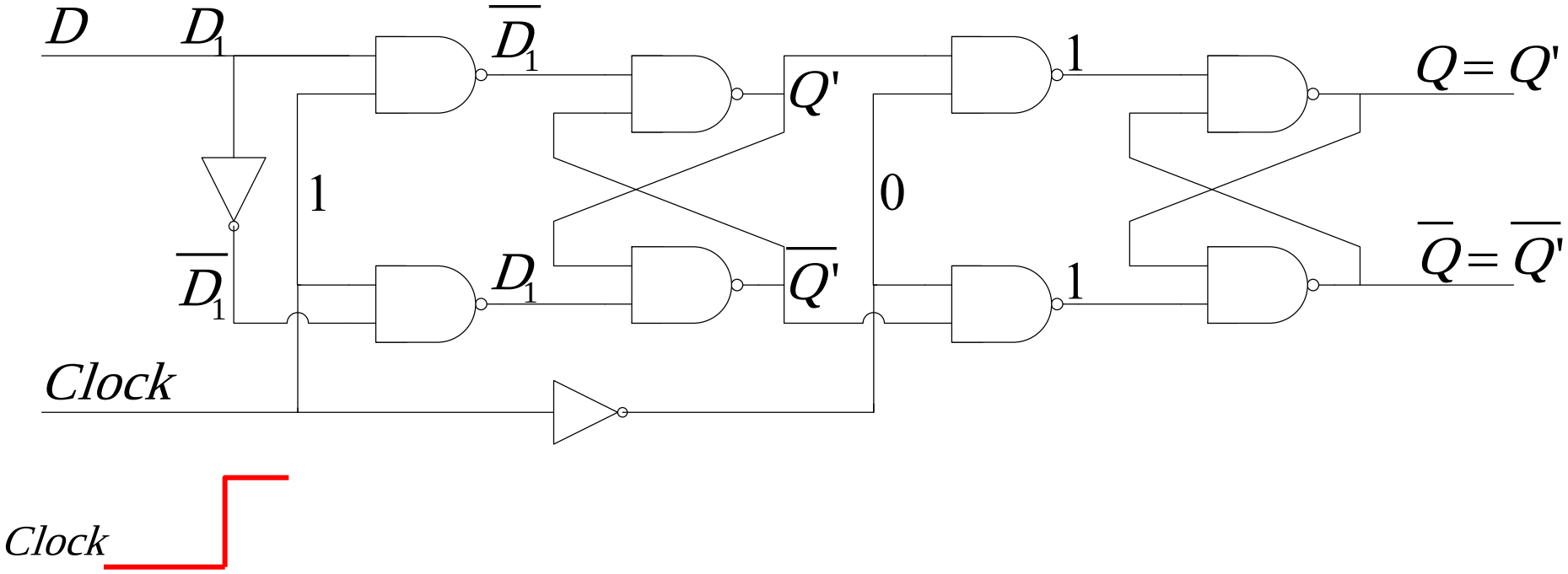
Master-Slave D Flip-Flop



Clock

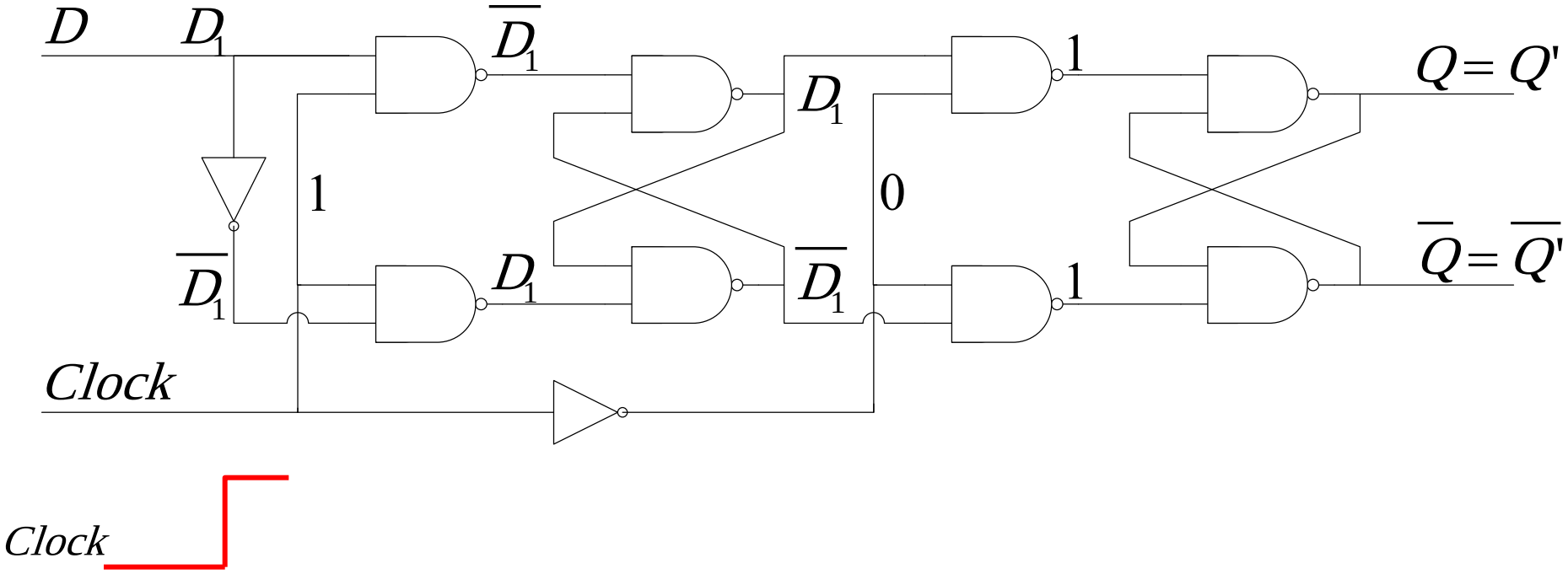
Negative Edge-Triggered Master-Slave D Flip-Flop

Master-Slave D Flip-Flop



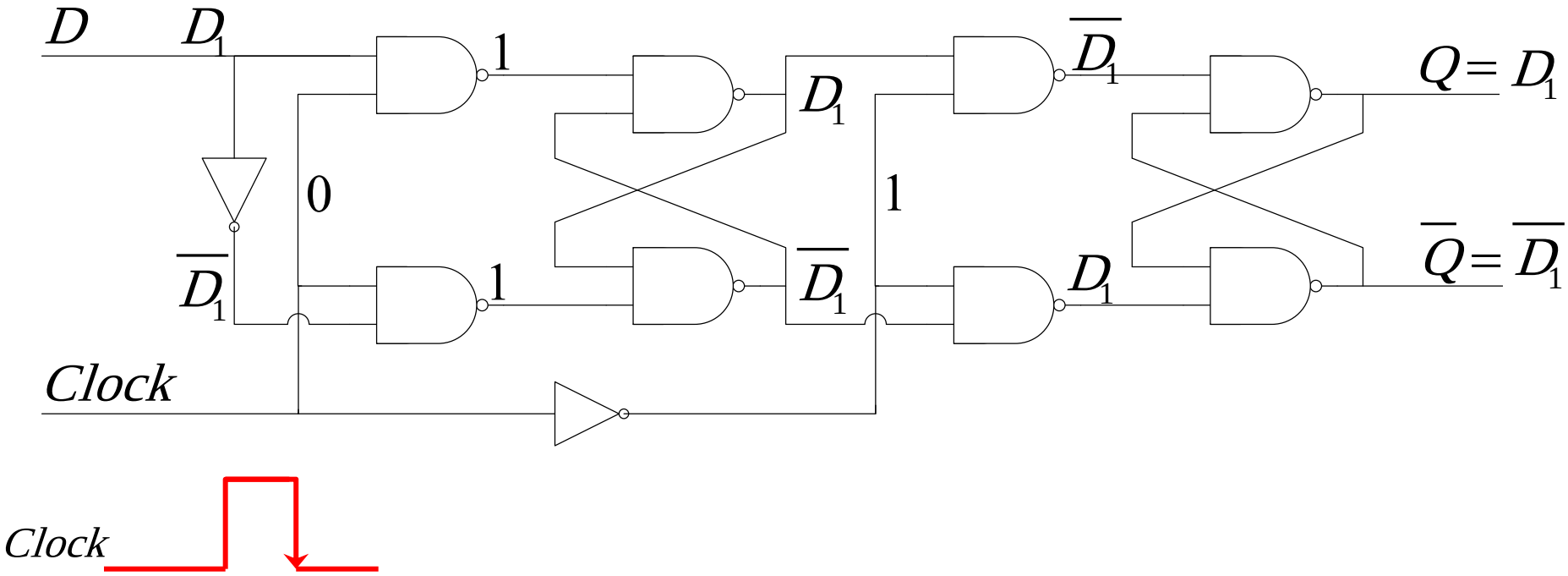
Negative Edge-Triggered Master-Slave D Flip-Flop

Master-Slave D Flip-Flop



Negative Edge-Triggered Master-Slave D Flip-Flop

Master-Slave D Flip-Flop



Negative Edge-Triggered Master-Slave D Flip-Flop

Flip-Flop : Implementation

Flip-Flop can be used in many applications:

- Memorize the assigned logic (D Flip-Flop);

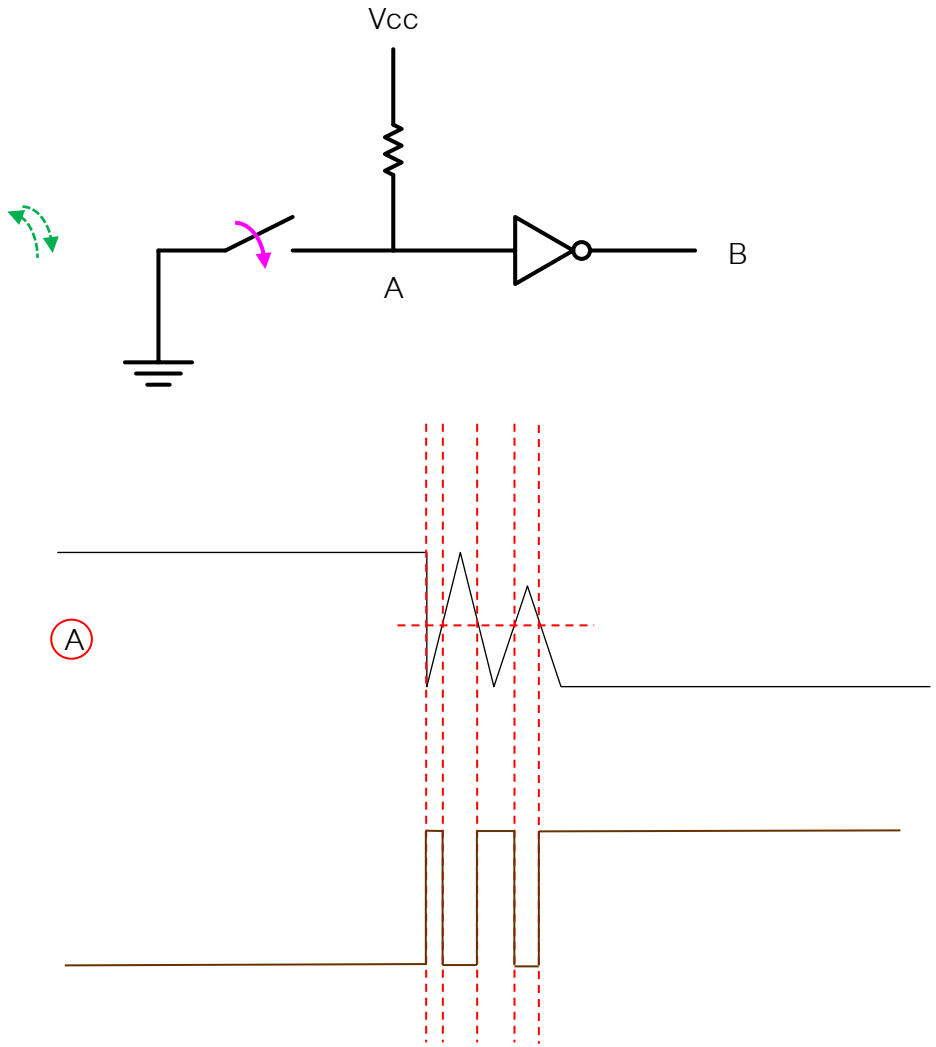
➔ Register = A number of D Flip-Flops to memorize a byte of data.

- Changing the opposite logic (T Flip-Flop):

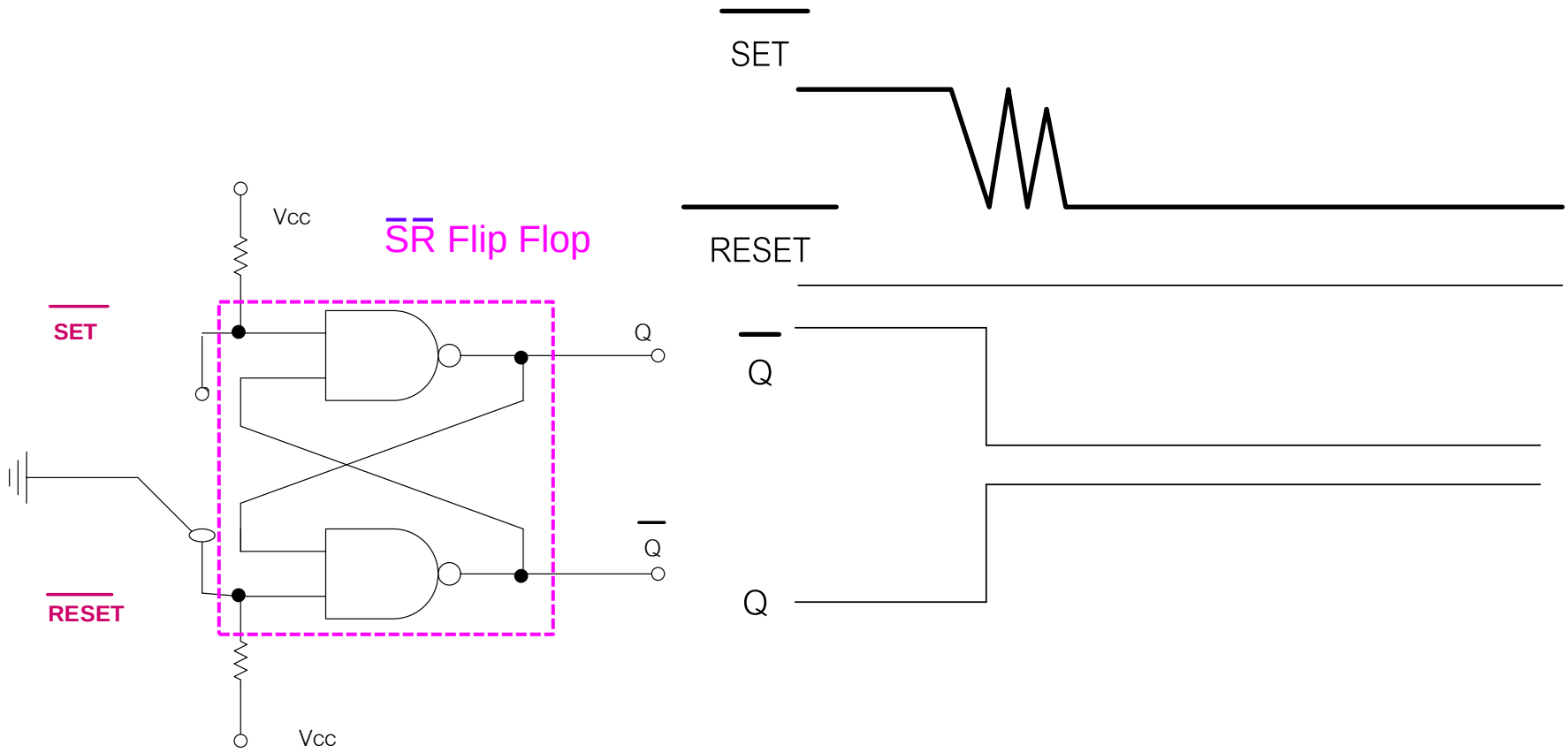
➔ Counter = A number of T Flip-Flops.

- Another application is the switch deboucer, as the circuit below:

Using A $\overline{\text{Set}}$-$\overline{\text{Reset}}$ Flip-Flop as a Switch Debouncer [1]



Using A $\overline{\text{Set}}$-$\overline{\text{Reset}}$ Flip-Flop as a Switch Debouncer [2]



Using A $\overline{\text{Set}}$-$\overline{\text{Reset}}$ Flip-Flop as a Switch Debouncer [3]