Lecture 34 :

Sequential Circuits

Flipflops


GO CLASSES
GATE OVERFLOW
GO Test Series
is now
GATE Overflow + GO Classes
2-IN-1 TEST SERIES
Most Awaited
GO Test Series
is Here
REGISTER NOW
http://tests.gatecse.in/
100+ More than 100 Quality Tests.
15 Mock Tests.
FROM
14th April
+91 - 6302536274
+91 9499453136
GO CLASSES
GATE OVERFLOW
LIVE
GATE 2023
Live + Recorded Lectures
Daily Home Work + Solution
Watch Any Time + Any Number of Times
Summary Lectures For Every Topic
Practice Sets From Standard Resources
Enroll Now
+91 - 6302536274
www.goclasses.in
linkedin.com/company/go-classes
instagram.com/goclasses_cs

Join **GO+ GO Classes Combined Test Series** for BEST quality tests, matching GATE CSE Level:

Visit www.gateoverflow.in website to join Test Series.

1. **Quality Questions:** No Ambiguity in Questions, All Well-framed questions.
2. Correct, **Detailed Explanation**, Covering Variations of questions.
3. **Video Solutions.**

NOTE :

Complete Discrete Mathematics & C-Programming Courses,

by GO Classes, are **FREE** for ALL learners.

Visit here to watch : https://www.goclasses.in/s/store/

SignUp/Login on Goclasses website for free and start learning.

Join GO Classes Resources, Notes, Content, information Telegram Channel:

Public Username: goclasses_cse

Join GO Classes **Doubt Discussion** Telegram Group :

Username: GATECSE_Goclasses

(Any doubt related to Goclasses Courses can also be asked here.)

Join GATEOverflow Doubt Discussion Telegram Group :

Username: gateoverflow_cse

Recap:

Sequential Circuits

Latches (SR Latch)

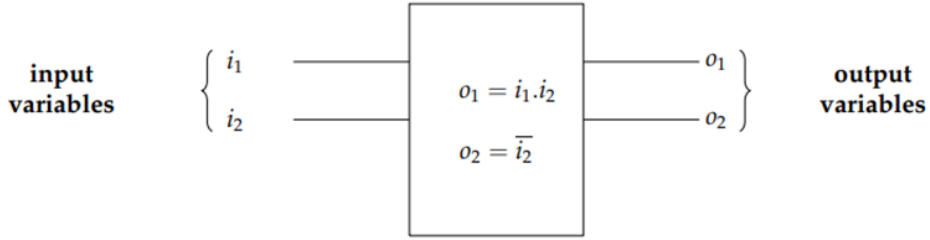
Sequential Logic

- The logic circuits discussed previously are known as *combinational*, in that the output depends only on the condition of the latest inputs
- However, we will now introduce a type of logic where the output depends not only on the latest inputs, but also on the condition of earlier inputs. These circuits are known as *sequential*, and implicitly they contain *memory* elements

Properties of Sequential Circuits

- So far we have seen Combinational Logic
  - the output(s) depends only on the current values of the input variables
- Here we will look at Sequential Logic circuits
  - the output(s) can depend on present and also past values of the input and the output variables

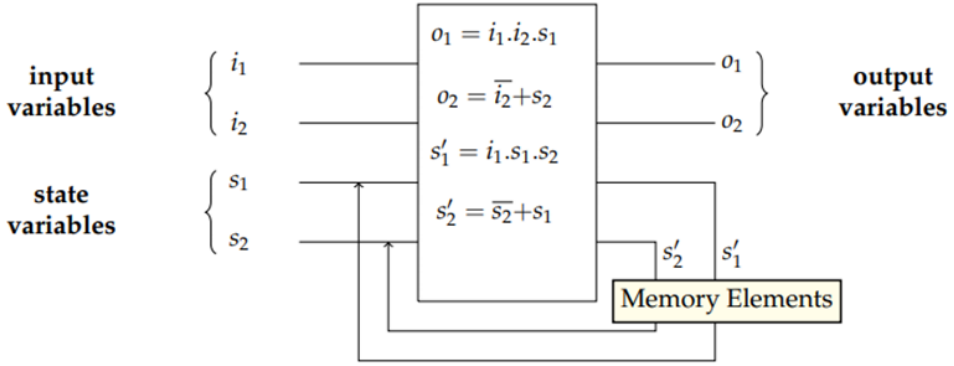
Combinational Circuits



Combinational circuits:

- Consists only of logic gates
- The output are determined by the present value of input
- Circuit behavior specified by a set of Boolean functions, Truth-tables, K-maps
- We have learned techniques to analyze and synthesize such circuits
- Examples: Adder, Multiplexers, Encoders, Decoders, etc.

Sequential Circuits



Sequential Circuits:

- Consists of logic gates and storage elements
- The output are determined by the present value of the input and the state of the storage elements
- In effect, the output may depend on past values of the input via the state of the storage elements

Motivation for Sequential Circuits :

So far, our circuits have been converting inputs to outputs without storing any data.

But To do more advanced things (e.g., to build computers), we need components that can store data.

Can we make a component that "remembers" using the components that we know already?

– Memory Element/Device

Can we make a component that can store a single bit using the components that we know already?

– Memory Element/Device.

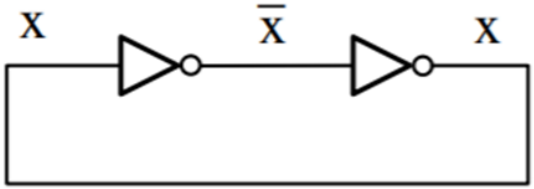
Basic Single Bit Storage Devices:

1. Cross Coupled Inverters
2. SR Latch (with Cross Couple NOR gates)
3. SR Latch with Cross Couple NAND gates

A simple memory element with NOT Gates

Basic Single Bit Storage Devices:

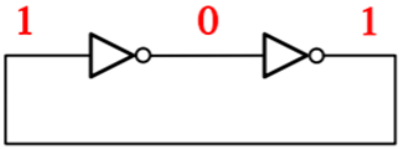
1. Cross Coupled Inverters



A simple memory element with NOT Gates

Basic Single Bit Storage Devices:

1. Cross Coupled Inverters

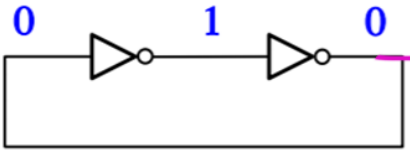


The circuit will stay in this state indefinitely.

A simple memory element with NOT Gates

Basic Single Bit Storage Devices:

1. Cross Coupled Inverters



o/p = Q = 0

Storing 0

Output = 0

State = 0

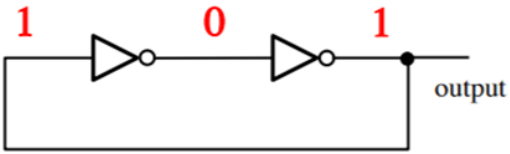
} Same

The circuit will stay in this state indefinitely.

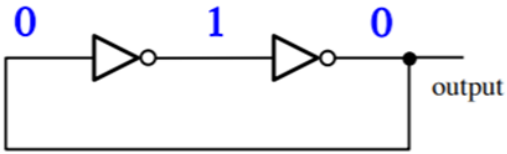
This circuit can be in two possible states

Basic Single Bit Storage Devices:

1. Cross Coupled Inverters

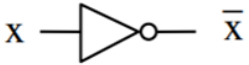


used to store a 1

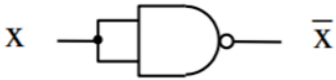


used to store a 0

Building a NOT gate with a NAND gate



x	$\overline{x}$
0	1
1	0

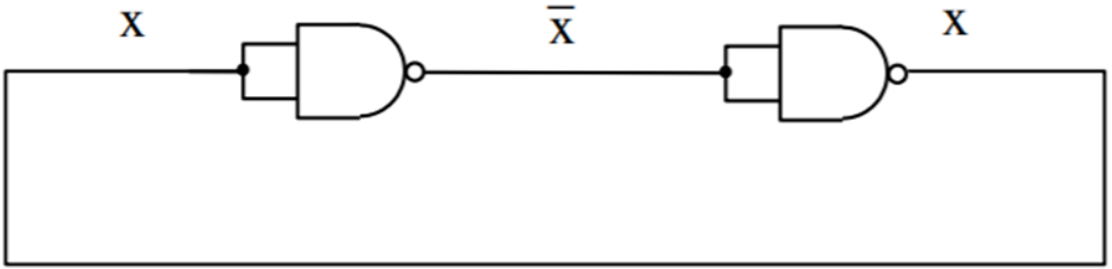


x	x	f
0	0	1
1	1	0

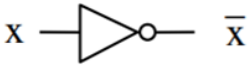
impossible combinations

Thus, the two truth tables are equal!

A simple memory element with NAND Gates



Building a NOT gate with a NOR gate



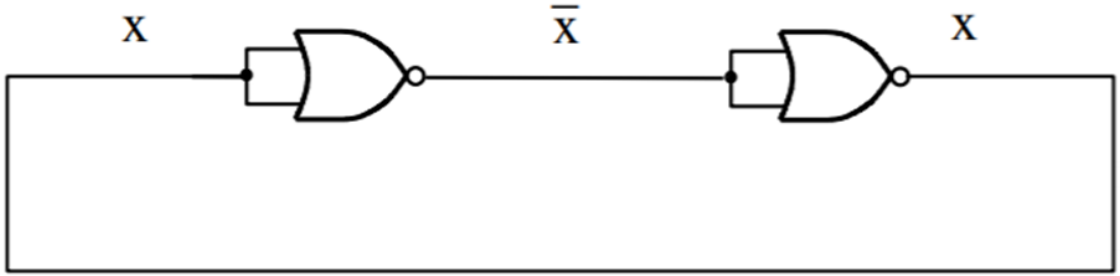
x	$\overline{x}$
0	1
1	0

x	x	f
0	0	1
1	1	0

impossible combinations

Thus, the two truth tables are equal!

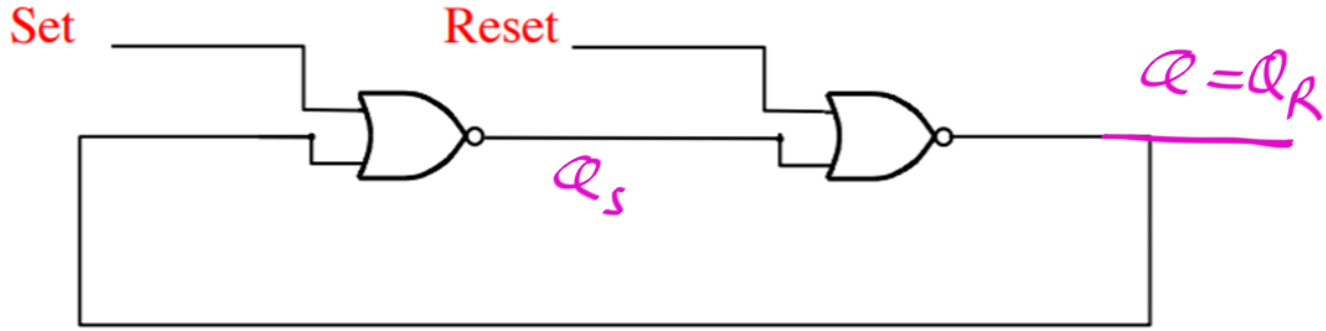
A simple memory element with NOR Gates



A simple memory element with NOR Gates

Basic Single Bit Storage Devices:

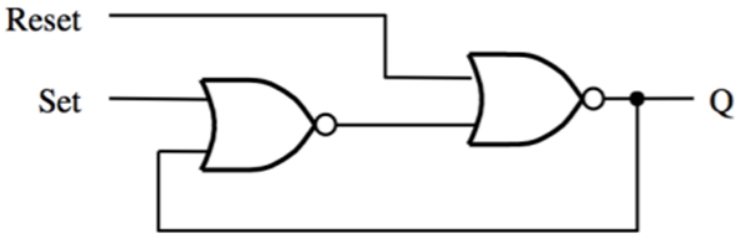
2. SR Latch with Cross Coupled NOR gates

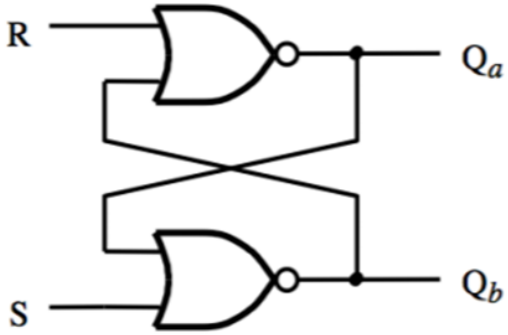


Two Different Ways to Draw the Same Circuit

Basic Single Bit Storage Devices:

2. SR Latch with Cross Coupled NOR gates

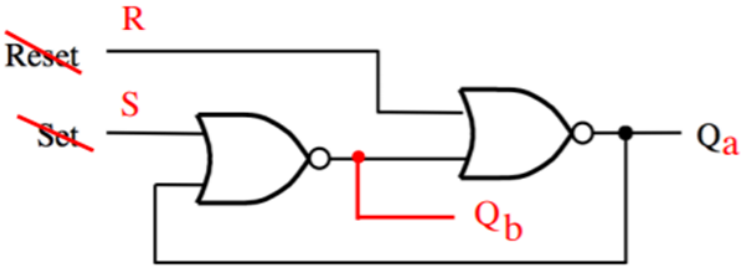


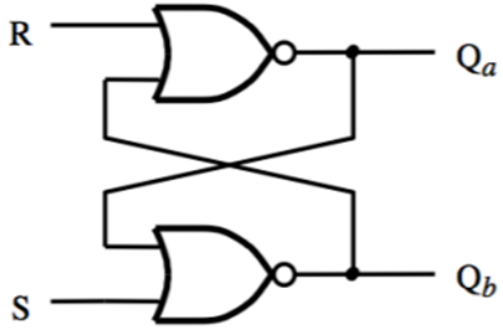


Two Different Ways to Draw the Same Circuit

Basic Single Bit Storage Devices:

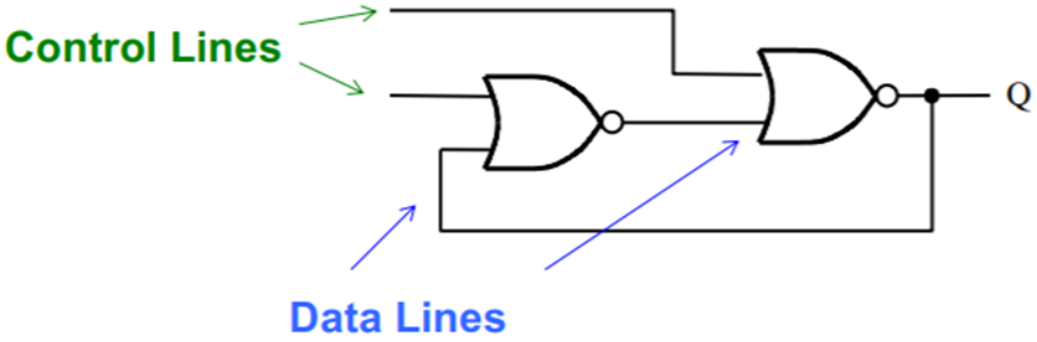
2. SR Latch with Cross Coupled NOR gates





What is a latch?

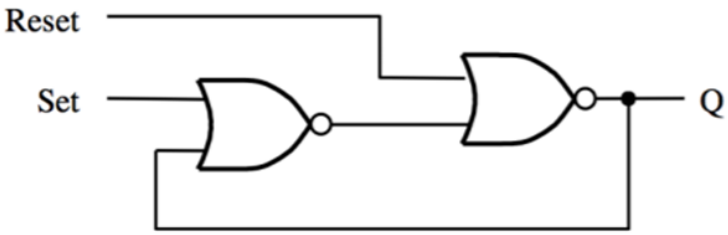
The Basic Latch

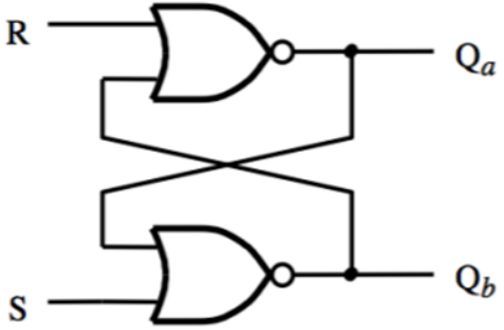


Two Different Ways to Draw the Same Circuit

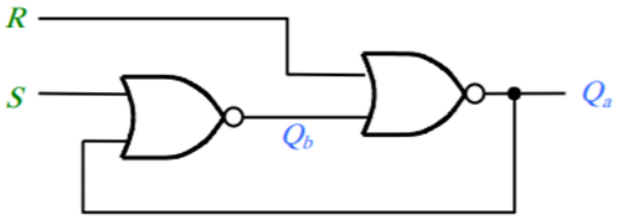
Basic Single Bit Storage Devices:

2. SR Latch with Cross Coupled NOR gates



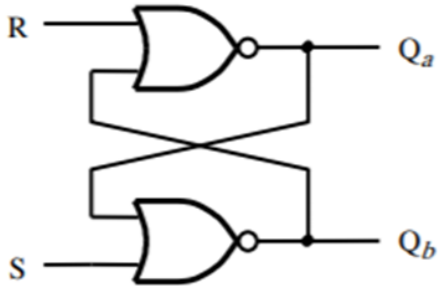


Behavior of the Basic Latch



S	R	$Q_a(t+1)$	$Q_b(t+1)$	
0	0	$Q_a(t)$	$Q_b(t)$	Latch
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Undesirable

Circuit and Characteristic Table



(a) Circuit

S	R	Q_a	Q_b	
0	0	0/1	1/0	(no change)
0	1	0	1	
1	0	1	0	
1	1	0	0	

Note that Q_a and Q_b are inverses of each other!

(b) Characteristic table

Oscillations and Undesirable States

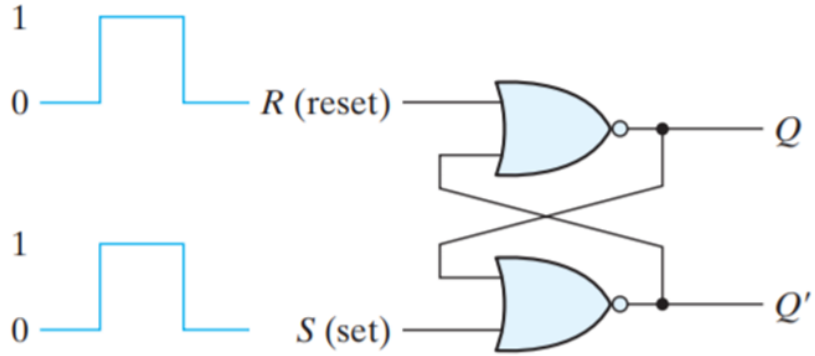
- When S=1 and R=1 both outputs of the latch are equal to 0, i.e., Q_a=0 and Q_b=0.
- Thus, the two outputs are no longer complements of each other.
- This is undesirable as many of the circuits that we will build later with these latches rely on the assumption that the two outputs are always complements of each other.
- (This is obviously not the case for the basic latch, but we will patch it later to eliminate this problem).

Oscillations and Undesirable States

- An even bigger problem occurs when we transition from S=R=1 to S=R=0.
- When S=R=1 we have $Q_a=Q_b=0$. After the transition to S=R=0, however, we get $Q_a=Q_b=1$, which would immediately cause $Q_a=Q_b=0$, and so on.
- If the gate delays and the wire lengths are identical, then this oscillation will continue forever.
- In practice, the oscillation dies down and the output settles into either $Q_a=1$ and $Q_b=0$ or $Q_a=0$ and $Q_b=1$.
- The problem is that we can't predict which one of these two it will settle into.

SR Latch

The *SR* latch is a circuit with two cross-coupled NOR gates or two cross-coupled NAND gates, and two inputs labeled *S* for set and *R* for reset. The *SR* latch constructed with two cross-coupled NOR gates is shown in Fig. 5.3. The latch has two useful states. When output $Q = 1$ and $Q' = 0$, the latch is said to be in the *set state*. When $Q = 0$ and $Q' = 1$, it is in the *reset state*. Outputs Q and Q' are normally the complement of each other. However, when both inputs are equal to 1 at the same time, a condition in which both outputs are equal to 0 (rather than be mutually complementary) occurs. If both inputs are then switched to 0 simultaneously, the device will enter an unpredictable or undefined state or a meta-stable state. Consequently, in practical applications, setting both inputs to 1 is forbidden.



(a) Logic diagram

S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(after $S = 1, R = 0$)
0	1	0	1	
0	0	0	1	(after $S = 0, R = 1$)
1	1	0	0	(forbidden)

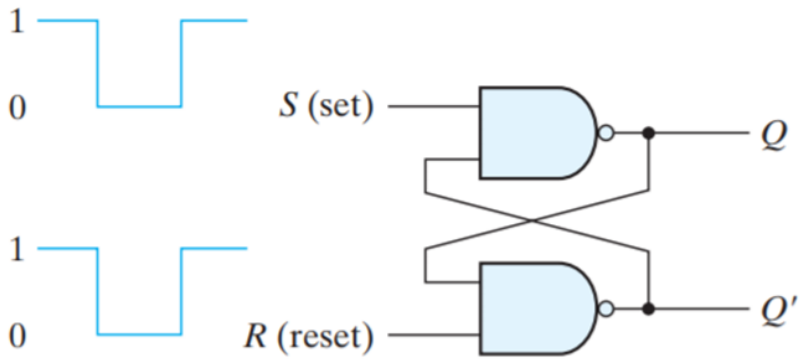
(b) Function table

FIGURE 5.3
SR latch with NOR gates

Next Topic:

SR Latch with Cross Coupled NAND Gates

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



(a) Logic diagram

S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after $S = 1, R = 0$)
0	1	1	0	
1	1	1	0	(after $S = 0, R = 1$)
0	0	1	1	(forbidden)

(b) Function table

FIGURE 5.4
***SR* latch with NAND gates**

In comparing the NAND with the NOR latch, note that the input signals for the NAND require the complement of those values used for the NOR latch. Because the NAND latch requires a 0 signal to change its state, it is sometimes referred to as an $S'R'$ latch. The primes (or, sometimes, bars over the letters) designate the fact that the inputs must be in their complement form to activate the circuit.

Oscillations and Undesirable States

- The basic latch with NAND gates also suffers form oscillation problems, similar to the basic latch implemented with NOR gates.

In an SR latch made by cross-coupling two NAND gates, if both S and R inputs are set to 0, then it will result in

A. $Q = 0, Q' = 1$
B. $Q = 1, Q' = 0$
C. $Q = 1, Q' = 1$
D. Indeterminate states

GATE CSE 2004 | Question: 18, ISRO2007-31

Which of the following input sequences for a cross-coupled $R - S$ flip-flop realized with two $NAND$ gates may lead to an oscillation?

A. $11, 00$
B. $01, 10$
C. $10, 01$
D. $00, 11$

GATE IT 2007 | Question: 7

In an SR latch made by cross-coupling two NOR gates, if both S and R inputs are set to 1, then it will result in

A. $Q = 0, Q' = 0$
B. $Q = 1, Q' = 0$
C. $Q = 1, Q' = 1$
D. Indeterminate states

Which of the following input sequences for a cross-coupled $R - S$ flip-flop realized with two $NAND$ gates may lead to an oscillation?

A. $11, 00$
B. $01, 10$
C. $10, 01$
D. $00, 11$

GATE IT 2007 | Question: 7

In an SR latch made by cross-coupling two NOR gates, if both S and R inputs are set to 0, then it will result in

A. $Q = 0, Q' = 1$
B. $Q = 1, Q' = 0$
C. $Q = 1, Q' = 1$

GATE CSE 2004 | Question: 18, ISRO2007-31

D - Unchanged state

Which of the following input sequences for a cross-coupled $R - S$ flip-flop realized with two NOR gates may lead to an oscillation?

A. $11, 00$
B. $01, 10$
C. $10, 01$
D. $00, 11$

GATE IT 2007 | Question: 7

In an SR latch made by cross-coupling two NOR gates, if both S and R inputs are set to 0, then it will result in

SR latch

A. $Q = 0, Q' = 1$
B. $Q = 1, Q' = 0$
C. $Q = 1, Q' = 1$
D. Unchanged State

GATE CSE 2004 | Question: 18, ISRO2007-31

Which of the following input sequences for a cross-coupled $R - S$ flip-flop realized with two NOR gates may lead to an oscillation?

11 → 00

A. $11, 00$
B. $01, 10$
C. $10, 01$
D. $00, 11$

GATE IT 2007 | Question: 7

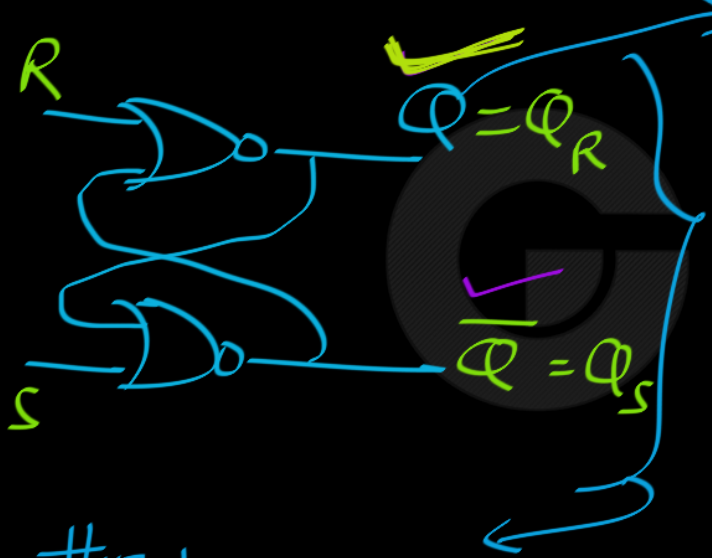
$(a+b)^2 = a^2 + 2ab + b^2$ ✓ In Exam.

$(a+b)^2 = (a+b)(a+b) = a^2 + ab + ba + b^2$

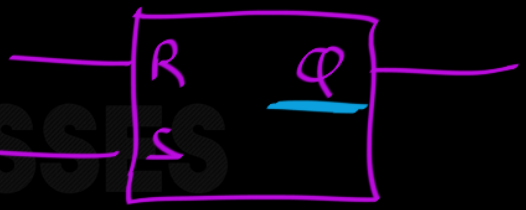
$= a^2 + 2ab + b^2$

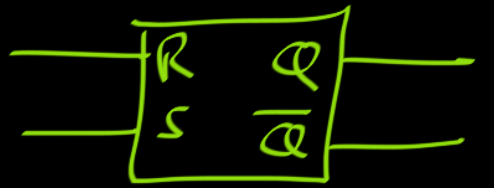
Not for exam
for preparation

SR latch : (SR latch with Cross Coupled NOR gates)



Block Diagram





#outputs = 1 = Q

If we want $\overline{Q}$, then No need to use extra inverter.

Just take Connection from $\overline{Q}$.

S	R	Q_{t+1}
0	0	Q_t (Retain) (unchanged) (memory)
0	1	0 (Reset) (Resetting the o/p/state)
1	0	1 (Set) (Setting the state/o/p)
1	1	0 (forbidden)

o/p Q at t+1 time

Characteristic Table (Behaviour Table)

$Q_{t+1} = f(R, S, Q_t)$

Q_{t+1} : Next output

Q_t : present o/p

In Truth Table of $Q_{t+1} \Rightarrow$ #Rows = 8 Rows.

Truth Table:

Truth Table

R	S	Q_t	Q_{t+1}	
0	0	0	0	Retain/latch
0	0	1	1	
0	1	0	1	set
0	1	1	1	
1	0	0	0	Reset
1	0	1	0	
1	1	0	0	forbidden
1	1	1	0	

$\bar{S}\bar{R}$ latch (SR latch with Cross Coupled NAND gates)

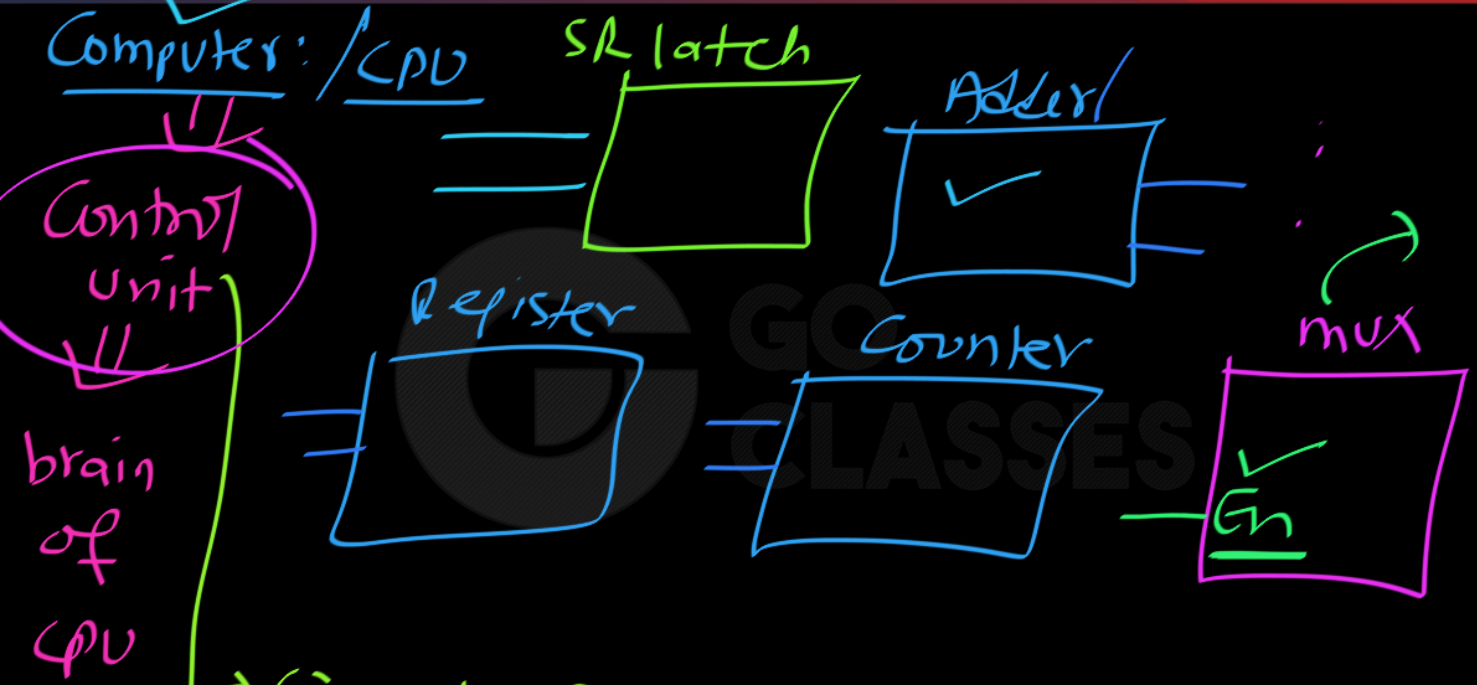
Active low latch

S	R	Q_{t+1}	
1	1	Q_t	Retain
0	1	1	set
1	0	0	Reset
0	0	1	for-bidden

Characteristic Table

Truth Table:

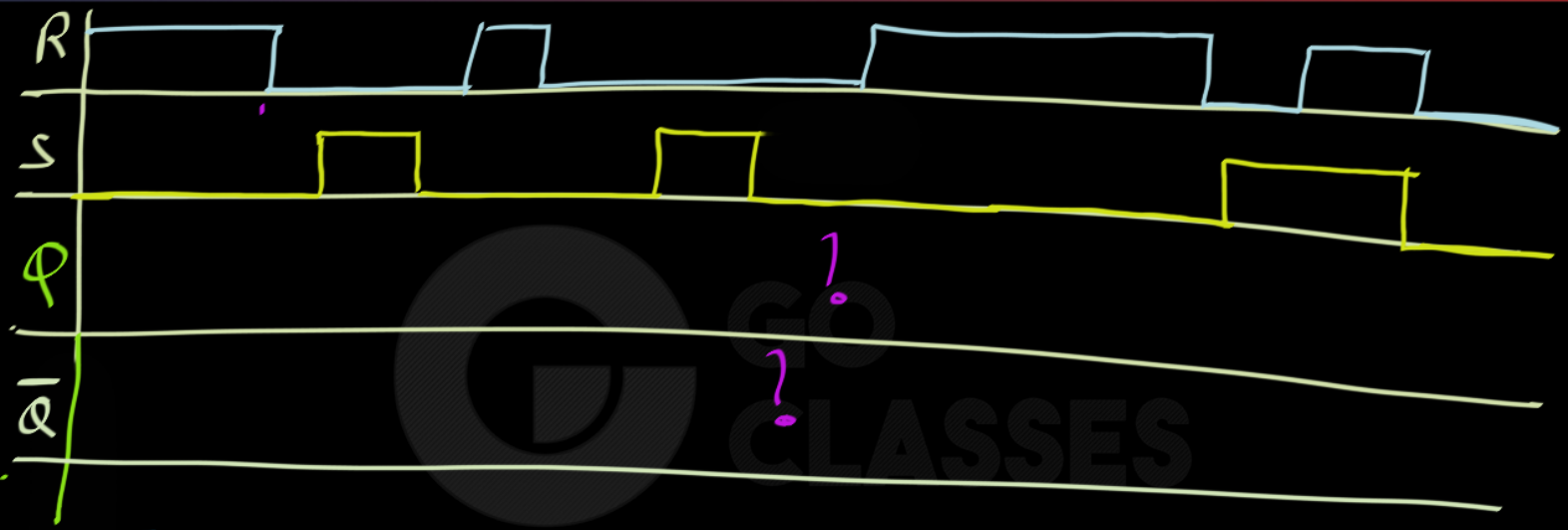
R	S	Q_t	$Q_{t+1} = f(R, S, Q_t)$	
1	1	0	0	latch / Retain / memory / unchanged
1	1	1	1	
1	0	0	1	set
1	0	1	1	
0	1	0	0	Reset
0	1	1	0	
0	0	0	1	forbidden
0	0	1	1	


Computer: /CPU
SR latch
Adder/
Control unit
Register
Counter
mux
brain of CPU
Gives the Command

Problem with SR (or $\bar{S}\bar{R}$) latch :

As soon as you apply/change inputs, o/p will change.

SR latch :⟶ No Controlling input to Control its working.



Timing Diagram of SR latch

Timing Diagram of SR latch

$Q = Q_R$ $\bar{Q} = Q_S$ ✓

SR latch:

S	R	$Q = Q_n$	$Q_s = \overline{Q}$	
1	1	0	0	forbidden
0	0	Q_t	$\overline{Q_t}$	Valid Cases
0	1	0	1	
1	0	1	0	

SR Latch / $\overline{S}\overline{R}$ latch :

Problem: No Controlling input

Next Topic:

Flipflops

Motivation

- The basic latch changes its state when the input signals change.
- It is hard to control when these input signals will change and thus it is hard to know when the latch may change its state.
- We want to have something like an Enable input.
- In this case it is called the "Clock" input because it is desirable for the state changes to be synchronized.

for Seq. ckts

||

"Clock"

||

will control working of Seq. ckt.

for Comb. Ckt

||

Enable Signal

Flipflop = Latch + Clock

- Clocked Latch
- Latch Controlled by Clock.

Next Topic:

Clock
(System Clock)

Clock

- The basic latch changes its state when the input signals change.
- It is hard to control when these input signals will change and thus it is hard to know when the latch may change its state.
- We want to have something like an Enable input.
- In this case it is called the "Clock" input because it is desirable for the state changes to be synchronized.

Clock signals control the outputs of the sequential circuit .That is it determines when and how the memory elements change their outputs . If a sequential circuit is not having any clock signal as input, the output of the circuit will change randomly.

Purchasing Computer: (System Clock frequency)

16 GB RAM, 2.4 GHz

CPU

Control Unit (manager) ⇒ Brain of CPU

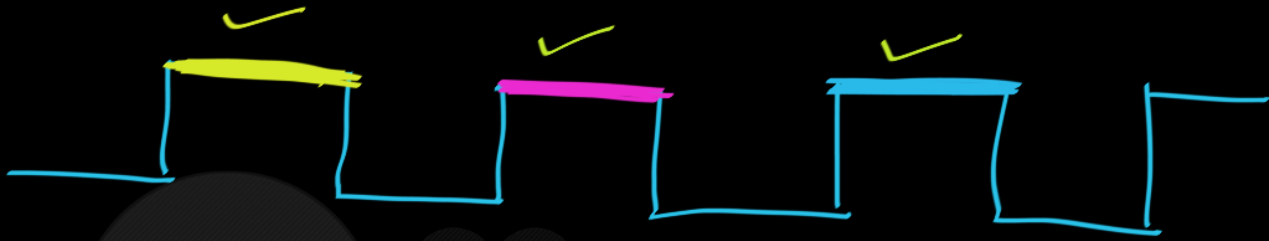
produce signal

Periodic signals

mux
Register

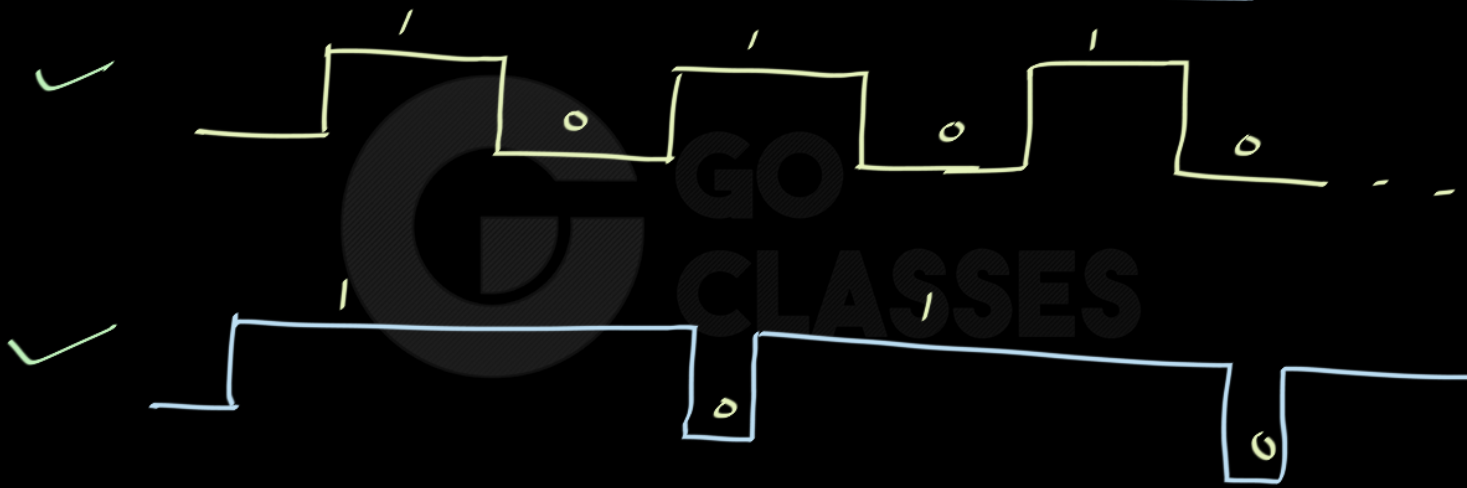
Clock:

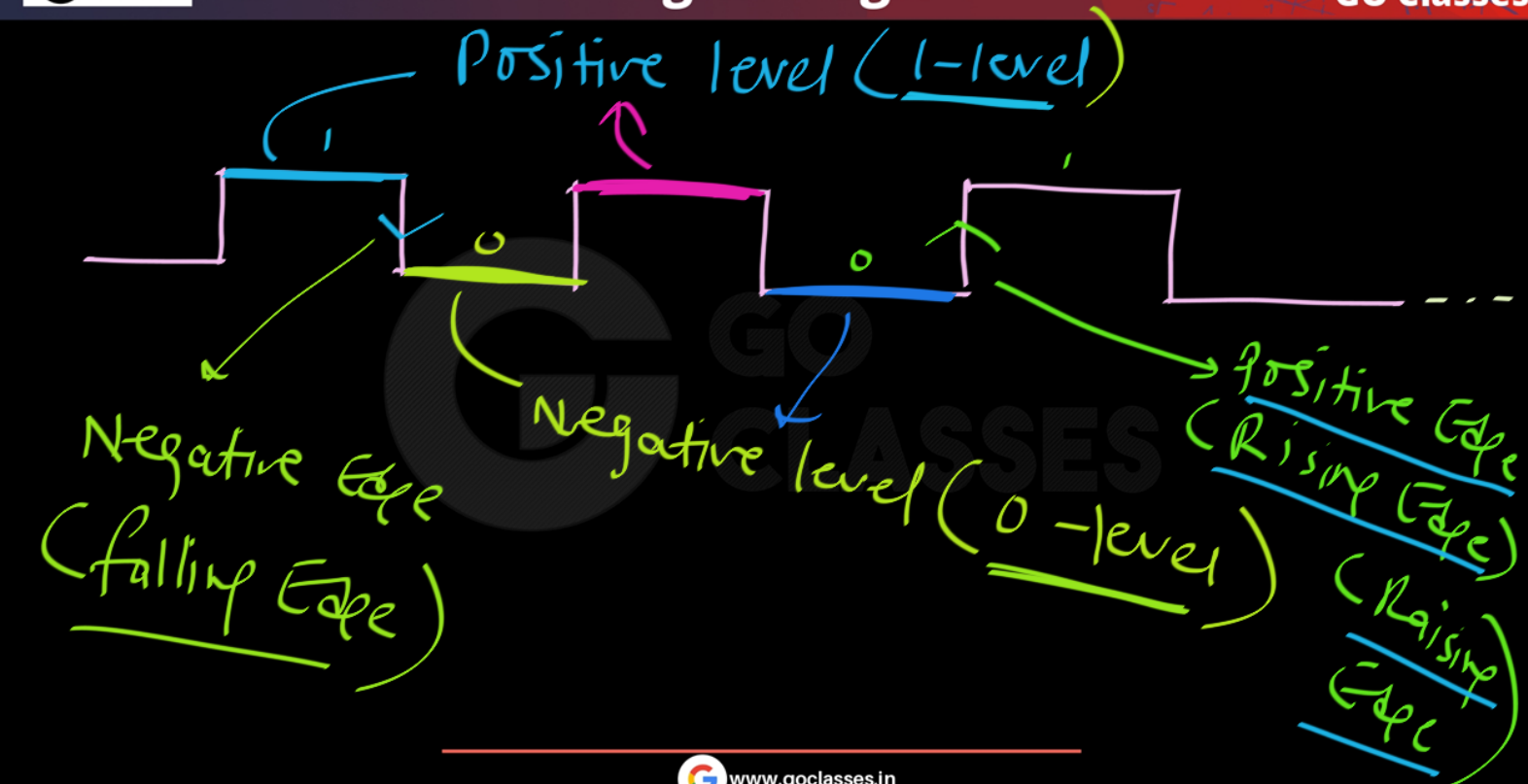
1 GHz



3 Hz

Clock: Continuous oscillating signal of 0, 1.




Positive level (1-level)
1
0
0
1
Negative Edge
(falling Edge)
Negative level (0-level)
Positive Edge
(Rising Edge)
(Raising
Edge

Time Period: (t) = time Duration b/w Consecutive Rising Edges

Time Period of Clock = 5ns

5ns

Time Period: (t) = time Duration b/w Consecutive Falling Edges

Frequency : How frequently ↓ Signal is Changing?

Clock

freq: High → low time period

freq low → High time Period

freq. Vs time period :

$$f = \frac{1}{t}$$

$t \uparrow \quad f \downarrow$

$t \downarrow \quad f \uparrow$

Note: Unit of time period: seconds (minutes, Hours)

$t = 5ns, 2\mu s, \ldots$

Unit of frequency: Hz (Hertz)

Hz = Per second

$\frac{1}{sec} = Hz$

$t = 5\,ns \qquad f = ?$

$$f = \frac{1}{5\,ns} = \frac{10^9}{5}\,\frac{1}{sec} = Hz$$

10^{-9}

$= 0.2\,GHz$

nano	10^{-9}
micro	10^{-6}
milli	10^{-3}

Giga	10^{9}
mega	10^{6}
kilo	10^{3}

$5\,\mu s = 5 \times 10^{-6}$ sec

$2\,GHz = 2 \times 10^{9}$ Hz

$f = 2.4\ GHz \Rightarrow t$ of Clock ?

System

$$t = \frac{1}{f} = \frac{1}{2.4\ GHz} = \frac{10^{-9}}{2.4}\left(\frac{1}{Hz}\right) sec$$

10^9

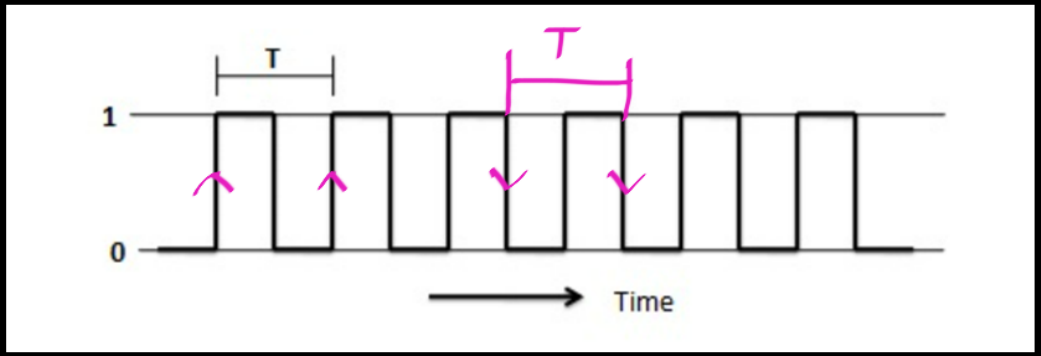
$$= \left(\frac{1}{2.4}\right) n\,sec = 0.416\ nSec$$

Clock Signal in Sequential Circuits

• The clock signal plays a crucial role in sequential circuits. A clock is a signal, which oscillates between logic level 0 and logic level 1, repeatedly. Square wave with constant frequency is the most common form of clock signal. A clock signal has “edges”. These are the instants at which the clock changes from 0 to 1 (a positive edge) or from 1 to 0 (a negative edge).

Clock signals control the outputs of the sequential circuit .That is it determines when and how the memory elements change their outputs .

If a sequential circuit is not having any clock signal as input, the output of the circuit will change randomly.


T
T
1
0
Time

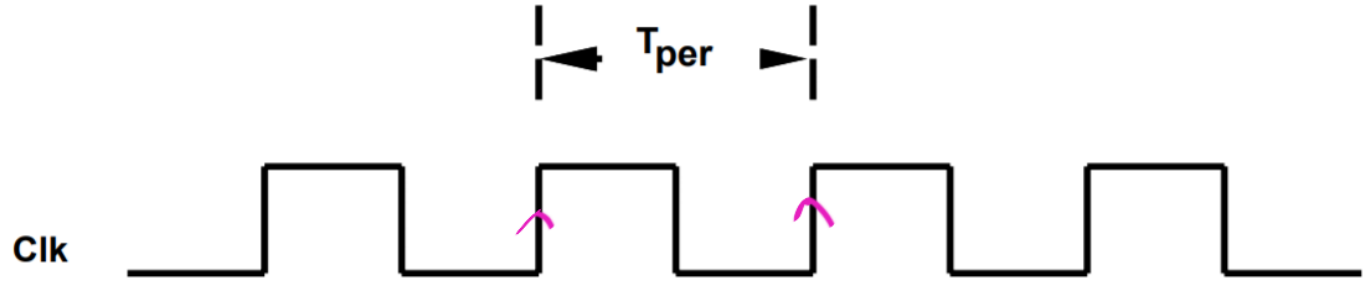


Figure 1: Periodic Clock Signal

Next Topic:

Back to Flipflops

Flipflop = Clocked Latch

Latch

Standard latches

- SR latch
- $\overline{S}\,\overline{R}$ latch

flipflop ≡ Clocked Latch

Standard ffs:

- SR ff
- D ff
- T ff
- JK ff

Next Topic:

1. SR Flipflop

SR Flipflop = Clocked SR Latch

Gated SR Latch

SR ff:

||

behaviour (Idea)

||

implementations

Behaviour / Characteristic of SR ff :

Clock	S	R	Q_{t+1} → Next o/p
0	X	X	Q_t (No Change) ← present o/p
1	0	0	Q_t Retain / Unchanged
1	0	1	0 Reset
1	1	0	1 Set
1	1	1	x forbidden → Never occurs

SR ff : Clocked SR latch

→ SR latch with Clock

→ studied

When Clock = 0 then ff Does not Respond to change in inputs.

$f(a,b) = x$ iff Don't Care

(1) input Does not occur / input Never occurs

↳ Don't Care Combination

(2) input Can occur, But function value Does not matter.

SR latch :

- SR latch with NOR → by Default
- " " " NAND

SR flipflop : ⟶ implementations { NOR, NAND, ... }

fixes Definition

Same as SR Latch

SR ff → fixed behaviour

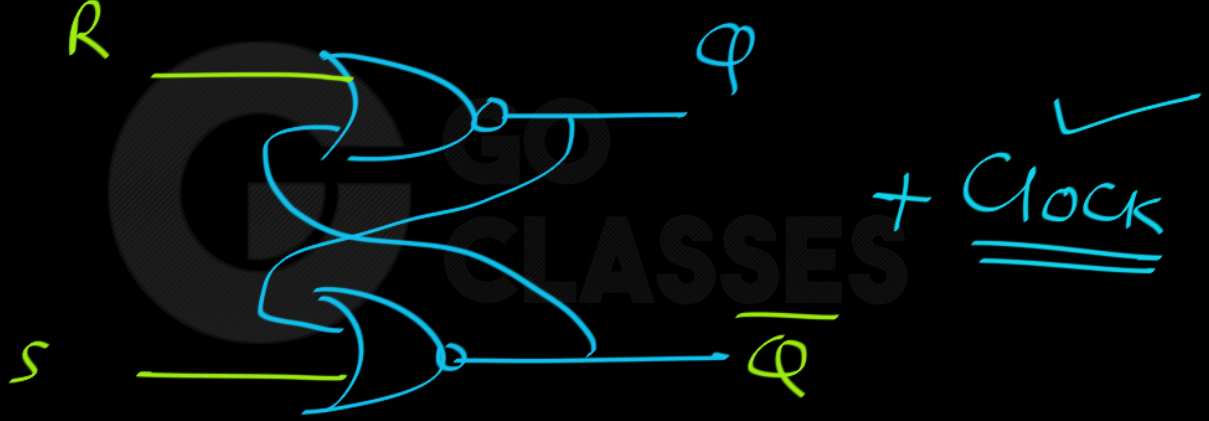
Clock	S	R	Q_{t+1}
0	X	X	Q_t (No change)
1	0	0	Q_t
1	0	1	0
1	1	0	1
1	1	1	X

Implementation :

SR latch ⟹ SR flipflop

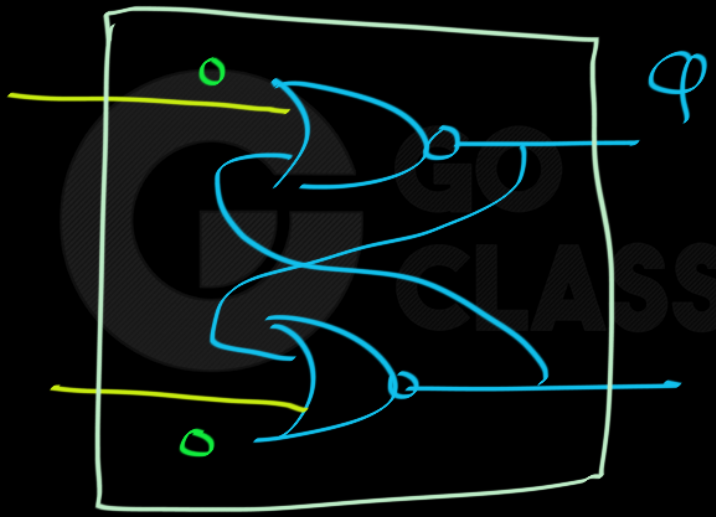
+ clock

SR latch ⟹ SR flipflop :



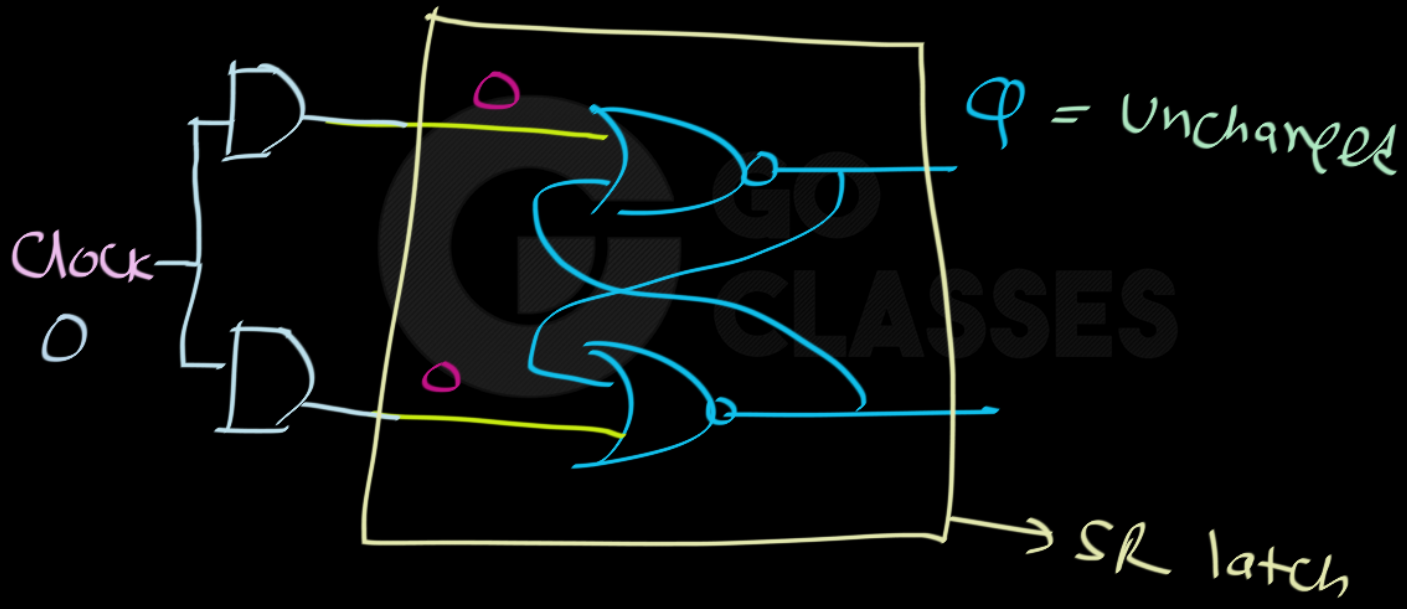
SR latch

SR latch ⟹ SR flipflop :

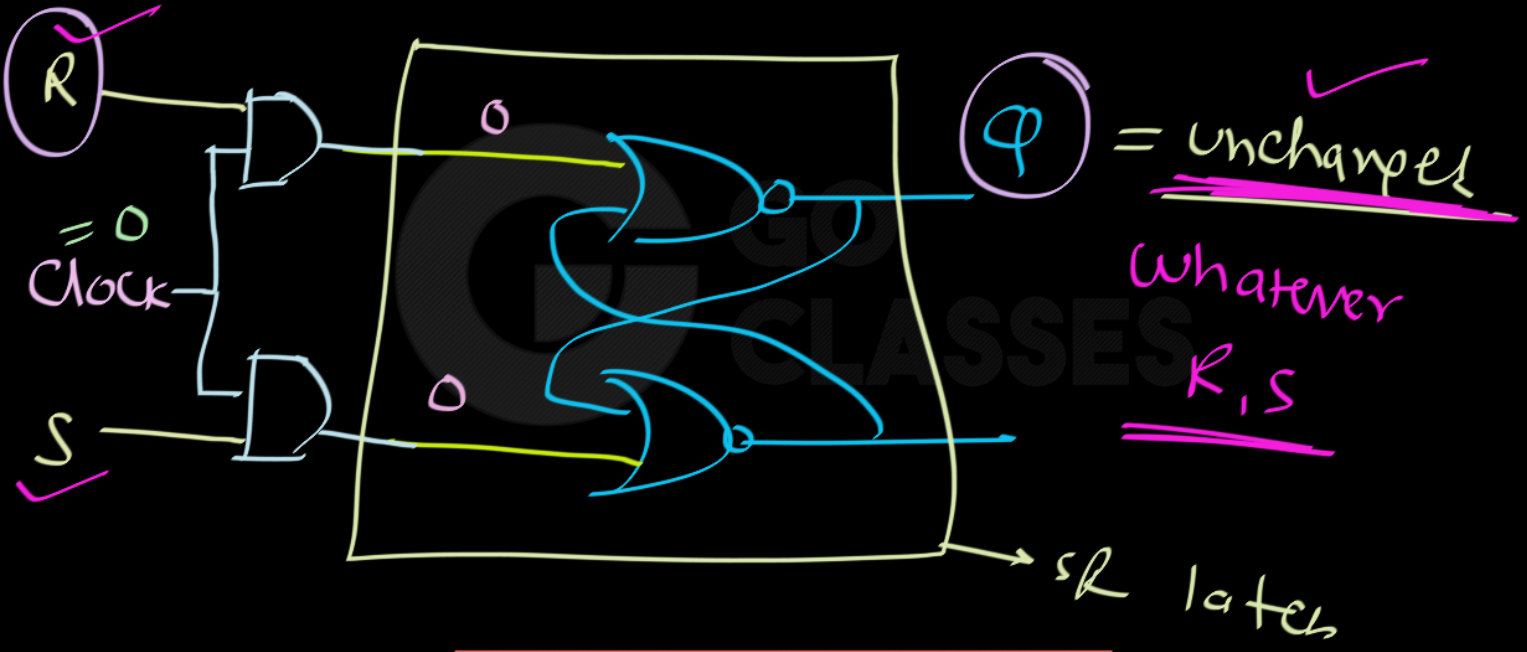


When
Clock = 0

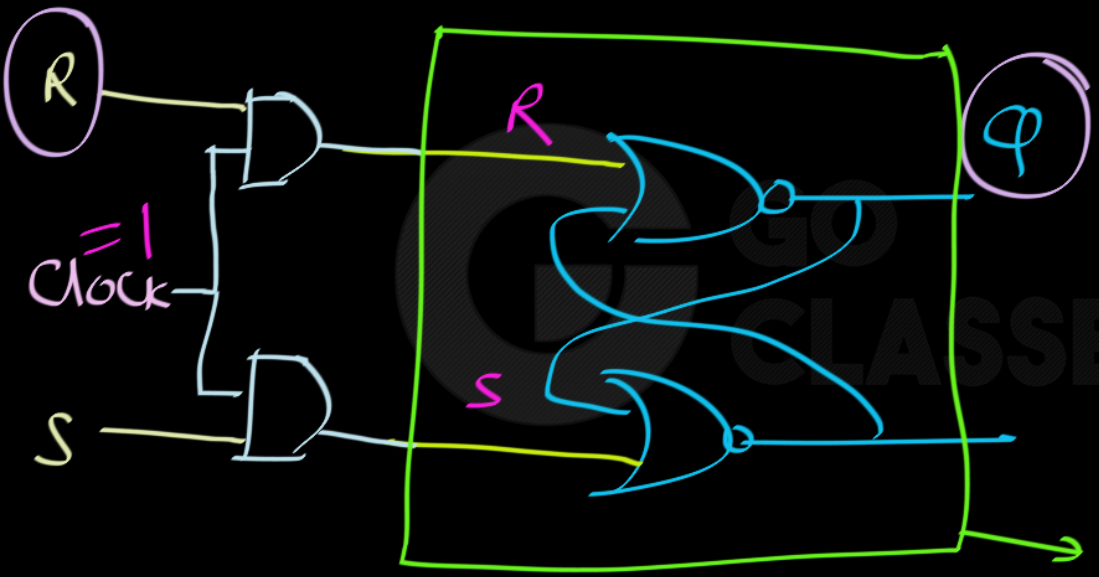
SR latch ⟹ SR flipflop :



SR latch ⟹ SR flipflop :



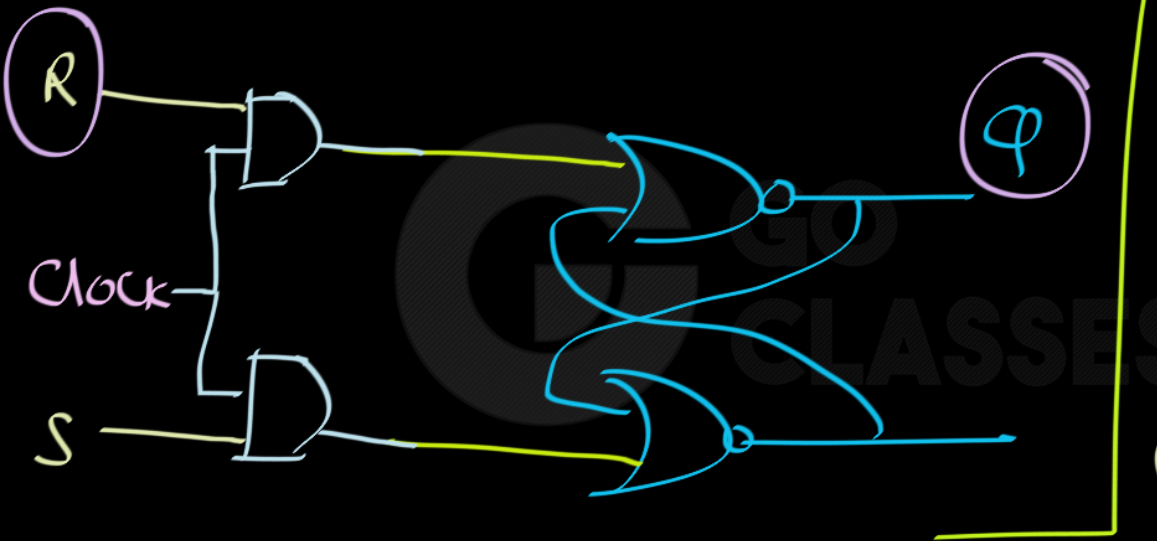
SR latch ⟹ SR flipflop :



If Clock = 1

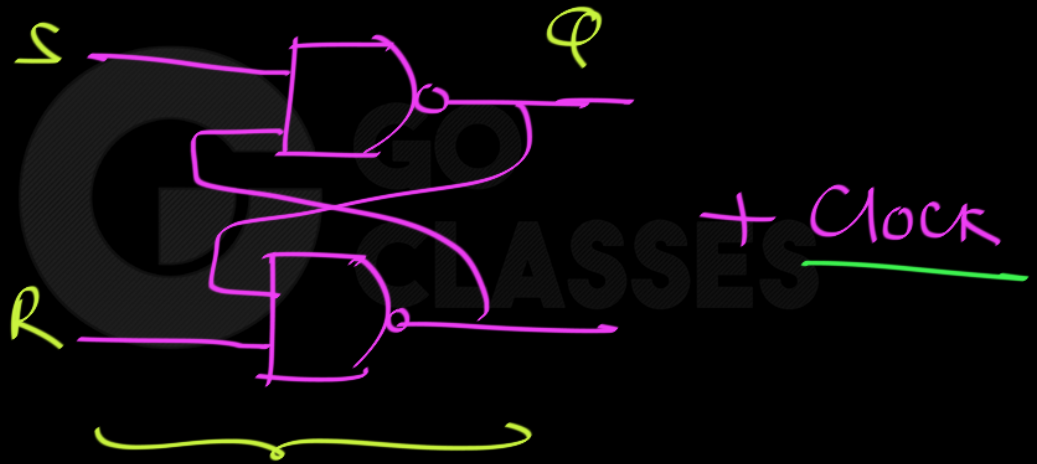
R	S	Q_{t+1}
0	0	Q_t
0	1	1
1	0	0
1	1	X

SR latch ⟹ SR flipflop :

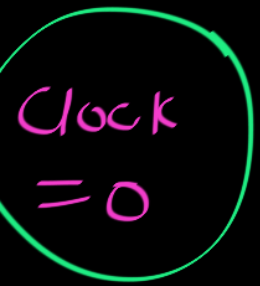


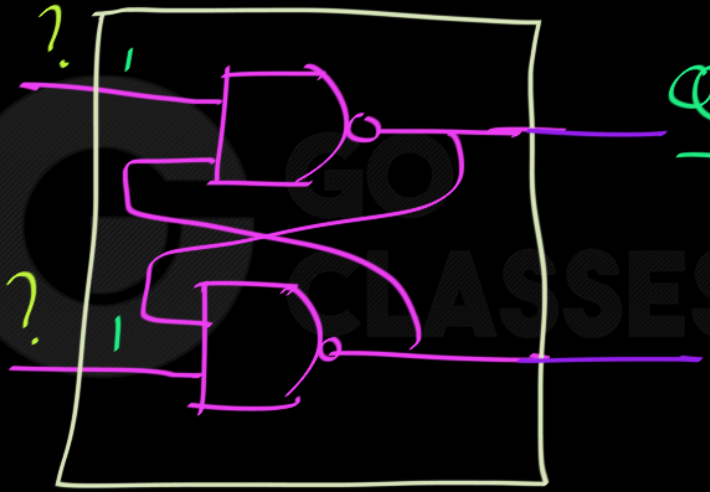
SR flipflop using SR latch (with NOR)

$\overline{S}\,\overline{R}$ latch $\Longrightarrow$ SR flip flop



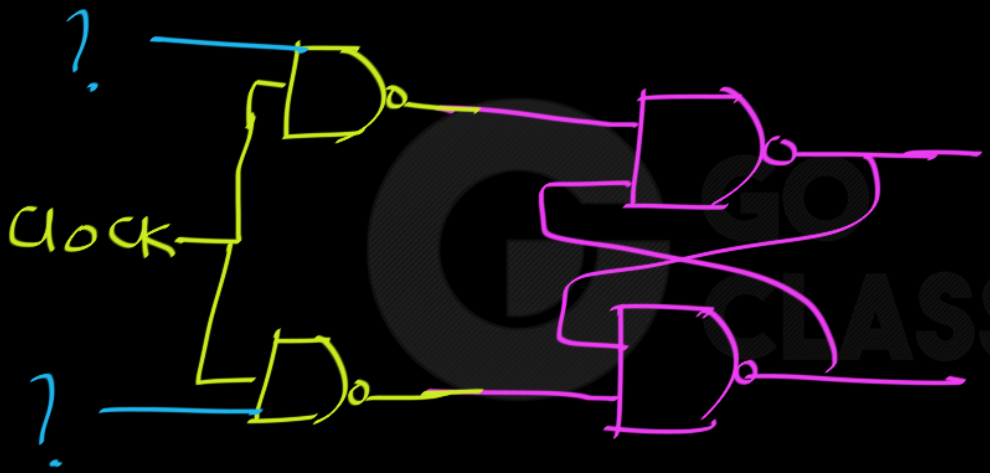
$\overline{S}\,\overline{R}$ latch $\Longrightarrow$ SR flip flop



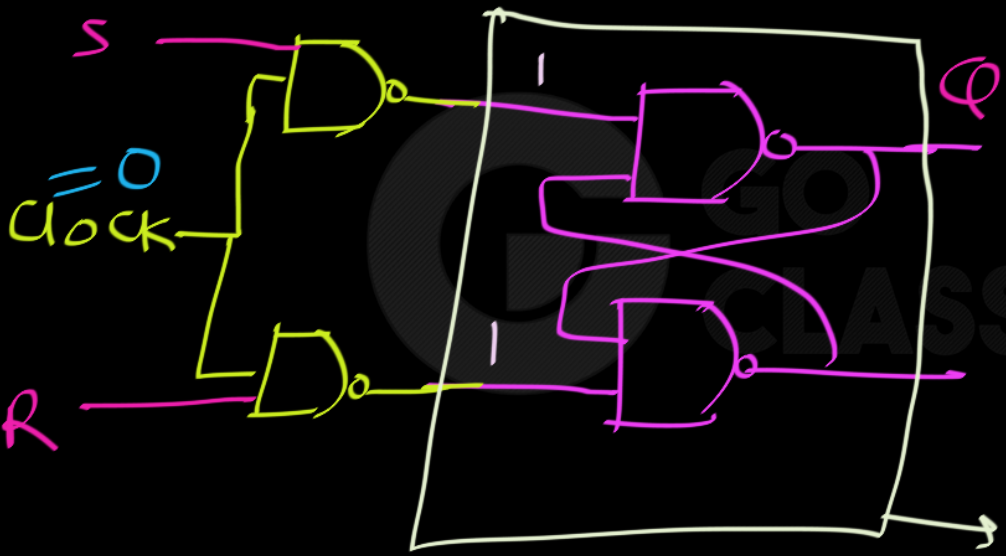


Q = Unchanges

$\overline{S}\,\overline{R}$ latch $\Longrightarrow$ SR flip flop



$\bar{S}\bar{R}$ latch $\Longrightarrow$ SR flip flop

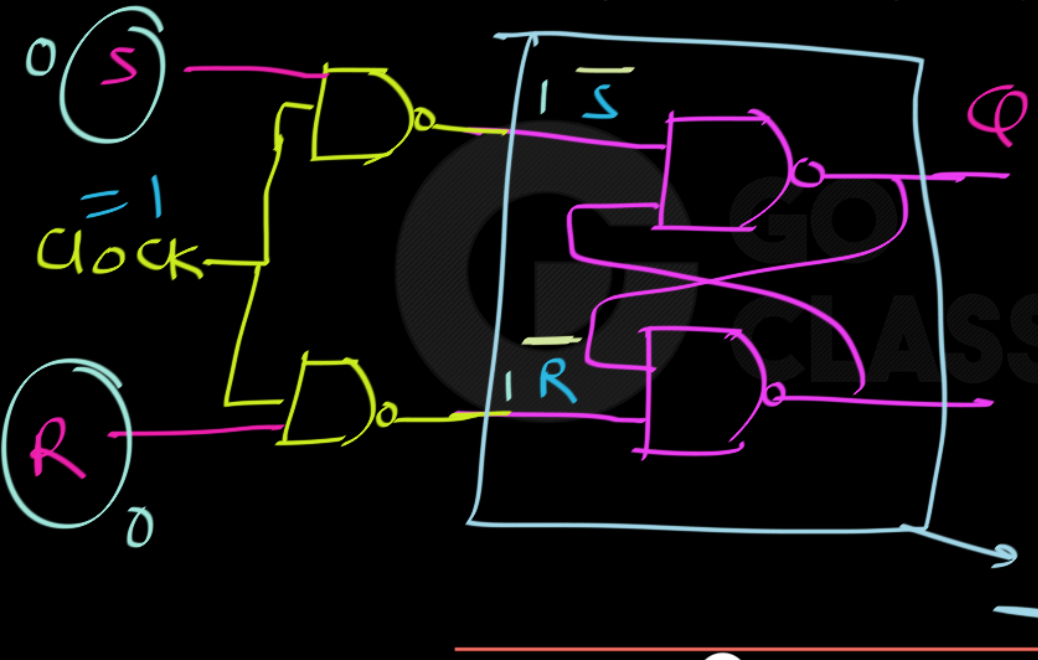


Q = Unchanged

Whatever R, S.

$\bar{S}\bar{R}$ latch

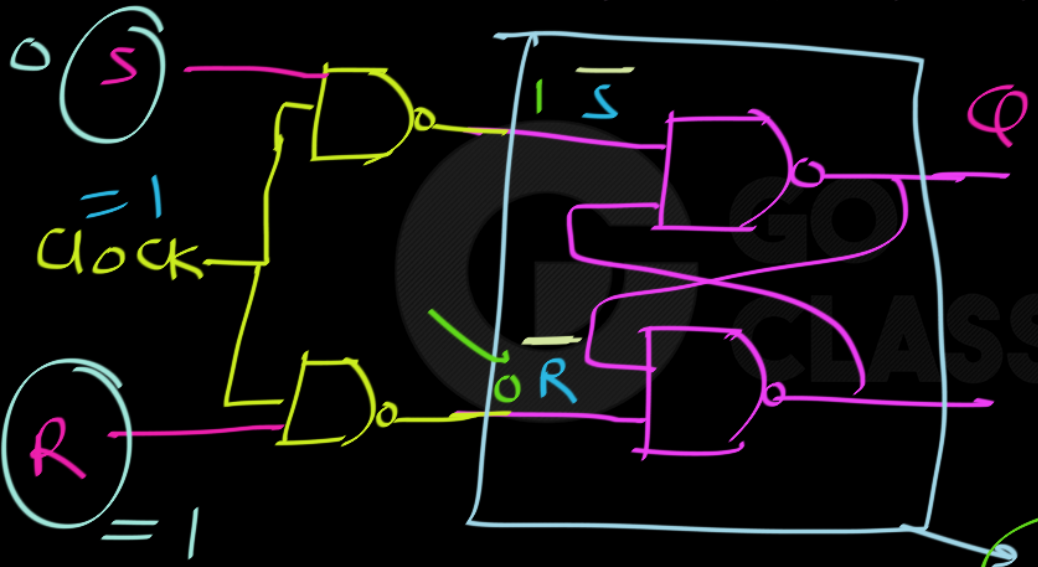
$\bar{S}\bar{R}$ latch $\Longrightarrow$ SR flip flop



If Clock = 1

S	R	Q_{t+1}
0	0	Q_t

$\bar{S}\bar{R}$ latch $\Longrightarrow$ SR flip flop

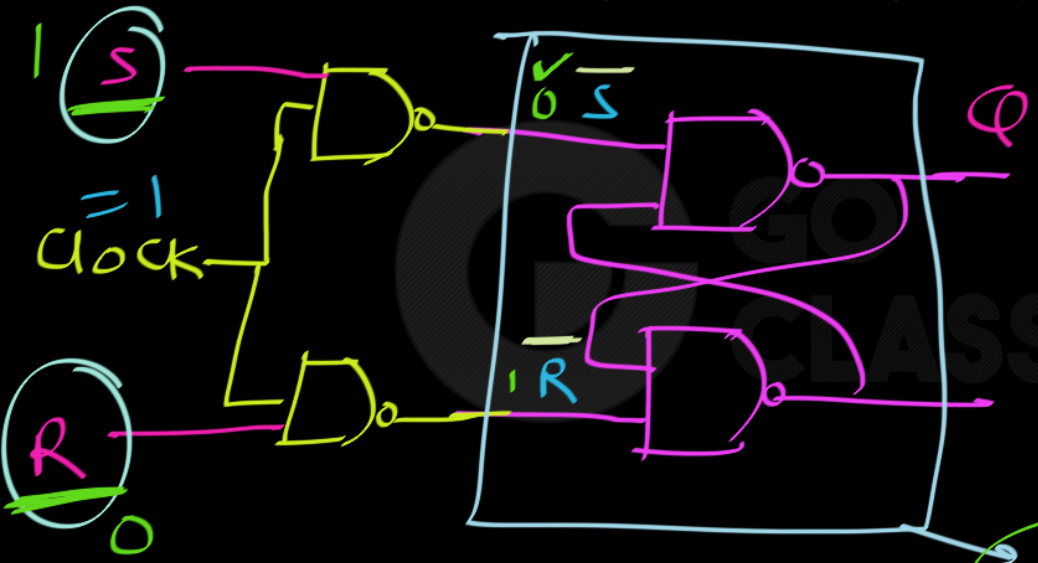


If Clock = 1

S	R	Q_{t+1}
0	0	Q_t
0	1	0

$\bar{S}\bar{R}$ latch ⇒ Active low

$\bar{S}\bar{R}$ latch $\Longrightarrow$ SR flip flop

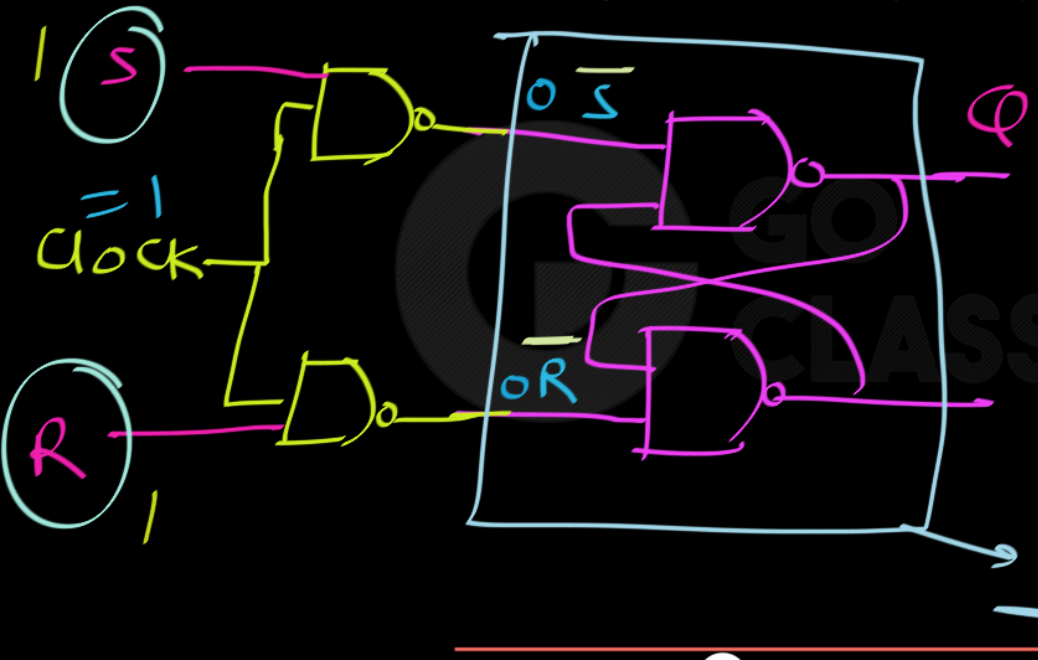


If Clock = 1

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1

$\bar{S}\bar{R}$ latch → Active low

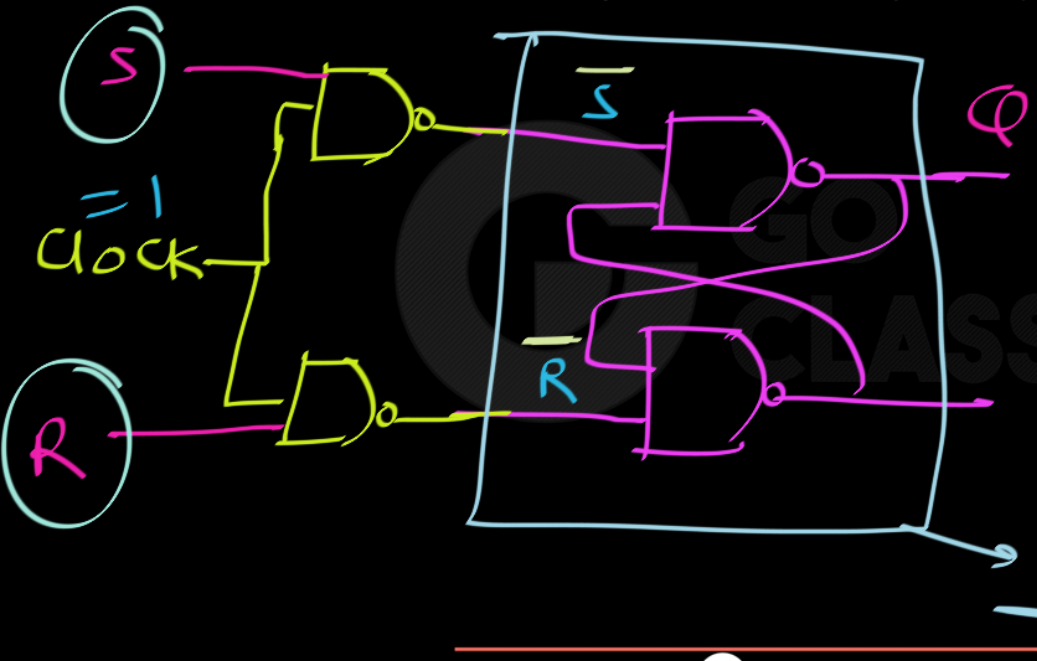
$\bar{S}\bar{R}$ latch $\Longrightarrow$ SR flip flop



If Clock = 1

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	X

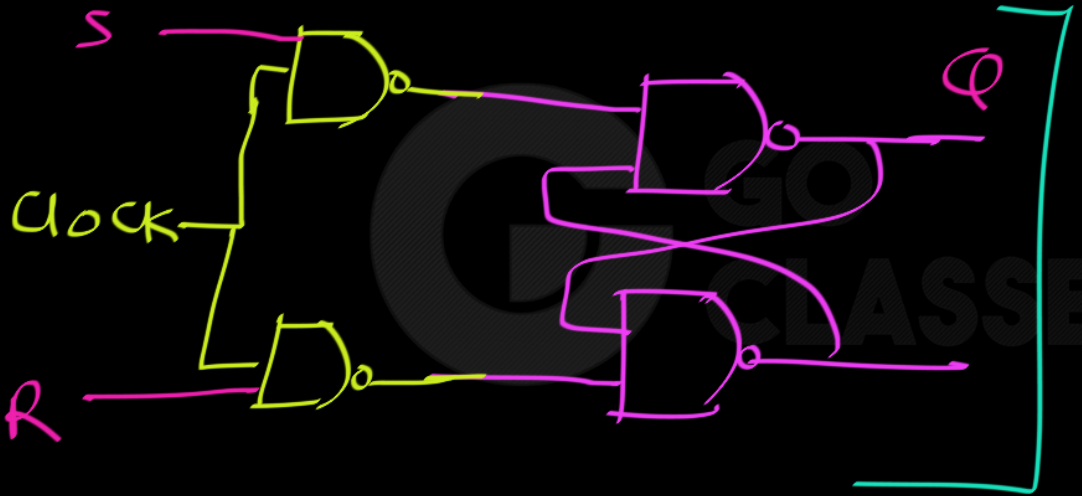
$\bar{S}\bar{R}$ latch $\Longrightarrow$ SR flip flop



If Clock = 1

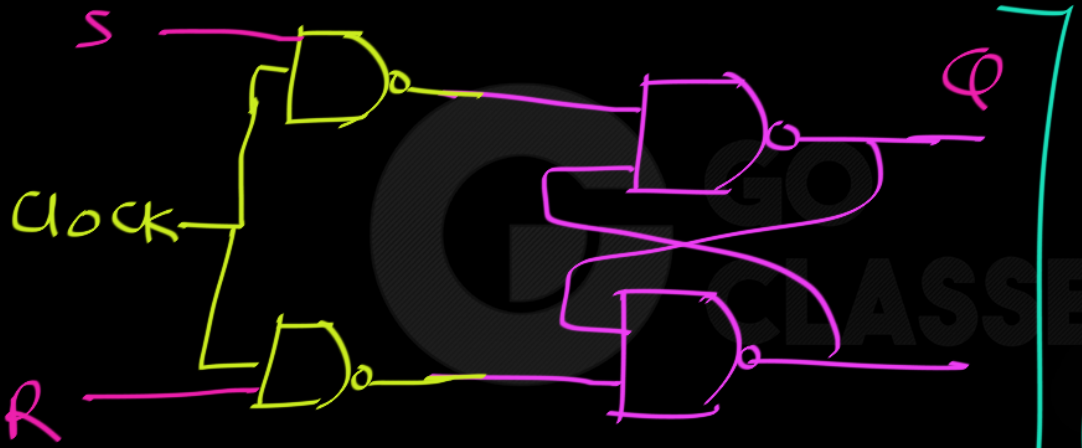
S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	X

$\overline{S}\,\overline{R}$ latch $\Longrightarrow$ SR flip flop



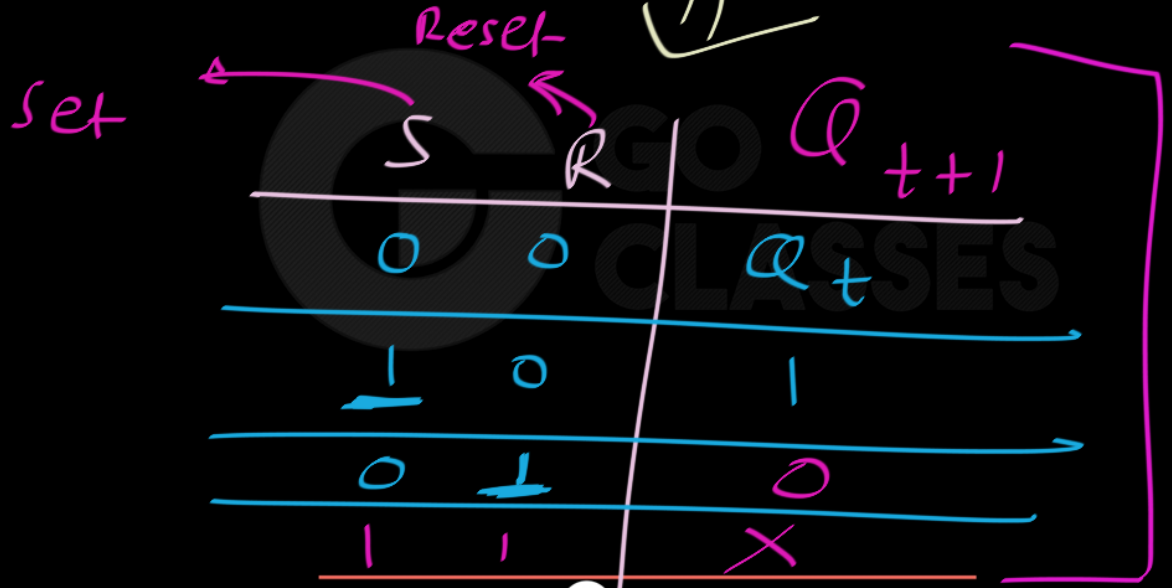
SR ff using $\overline{S}\,\overline{R}$ latch

$\overline{S}\,\overline{R}$ latch $\Longrightarrow$ SR flip flop

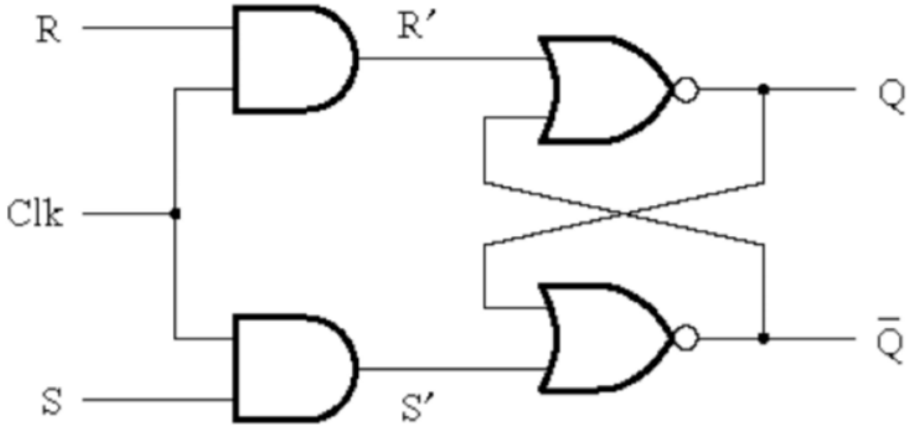


SR ff using (SR latch of NAND gates)

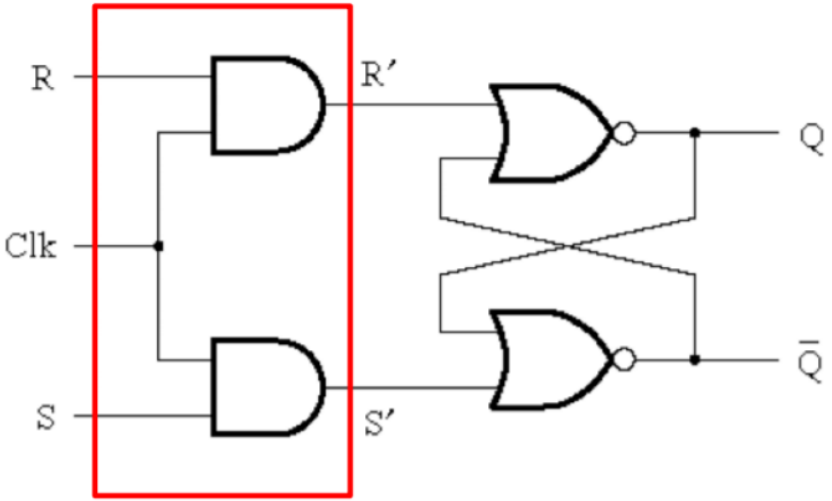
SR ff ⇒ fixed behaviour



Circuit Diagram for the Gated SR Latch

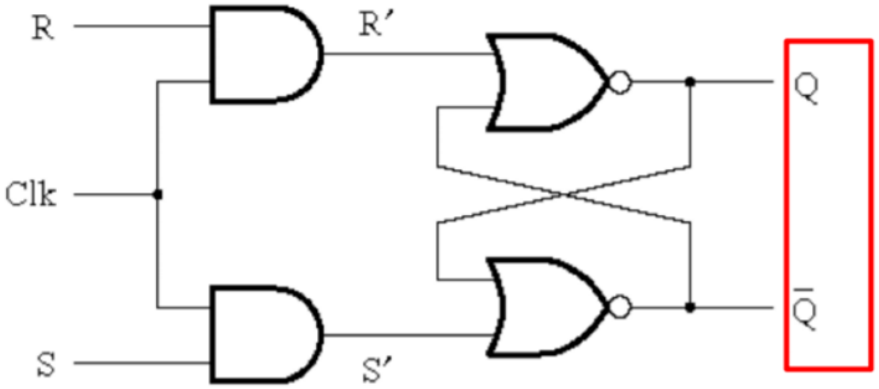


Circuit Diagram for the Gated SR Latch



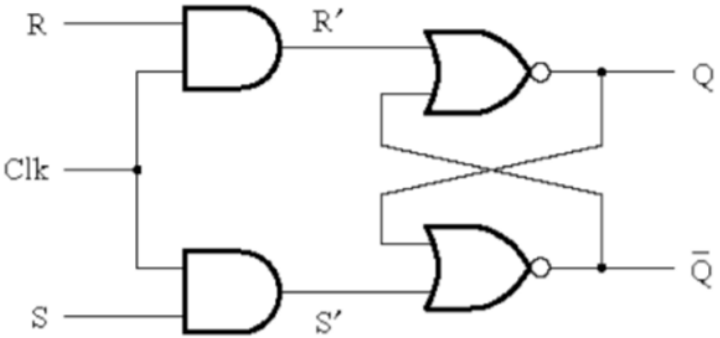
This is the "gate"
of the gated latch

Circuit Diagram for the Gated SR Latch



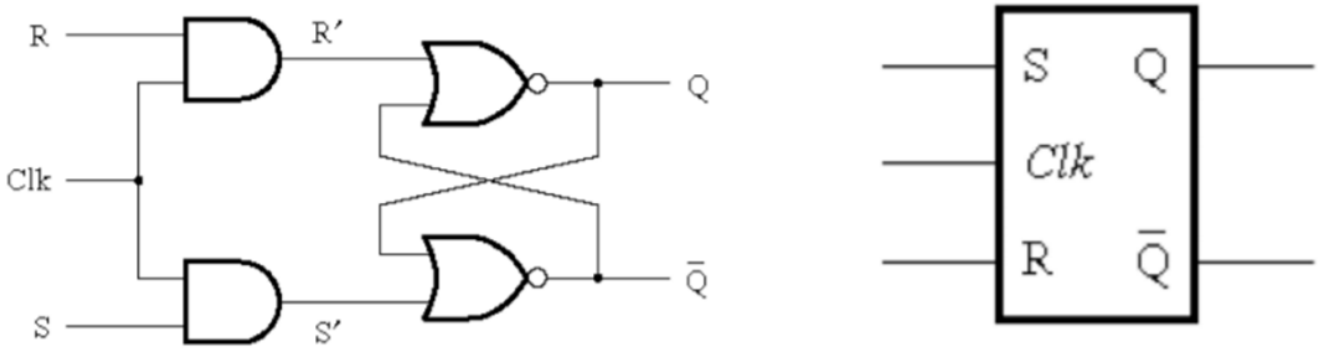
Notice that these are complements of each other

Circuit Diagram and Characteristic Table for the Gated SR Latch

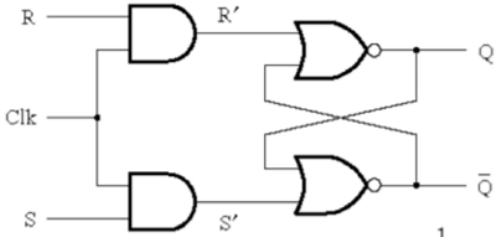


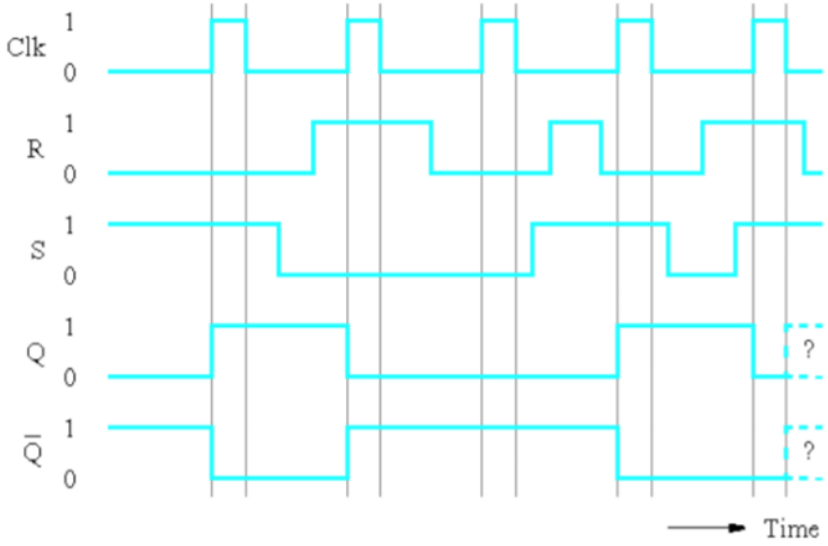
Clk	S	R	Q(t + 1)
0	x	x	Q(t) (no change)
1	0	0	Q(t) (no change)
1	0	1	0
1	1	0	1
1	1	1	x

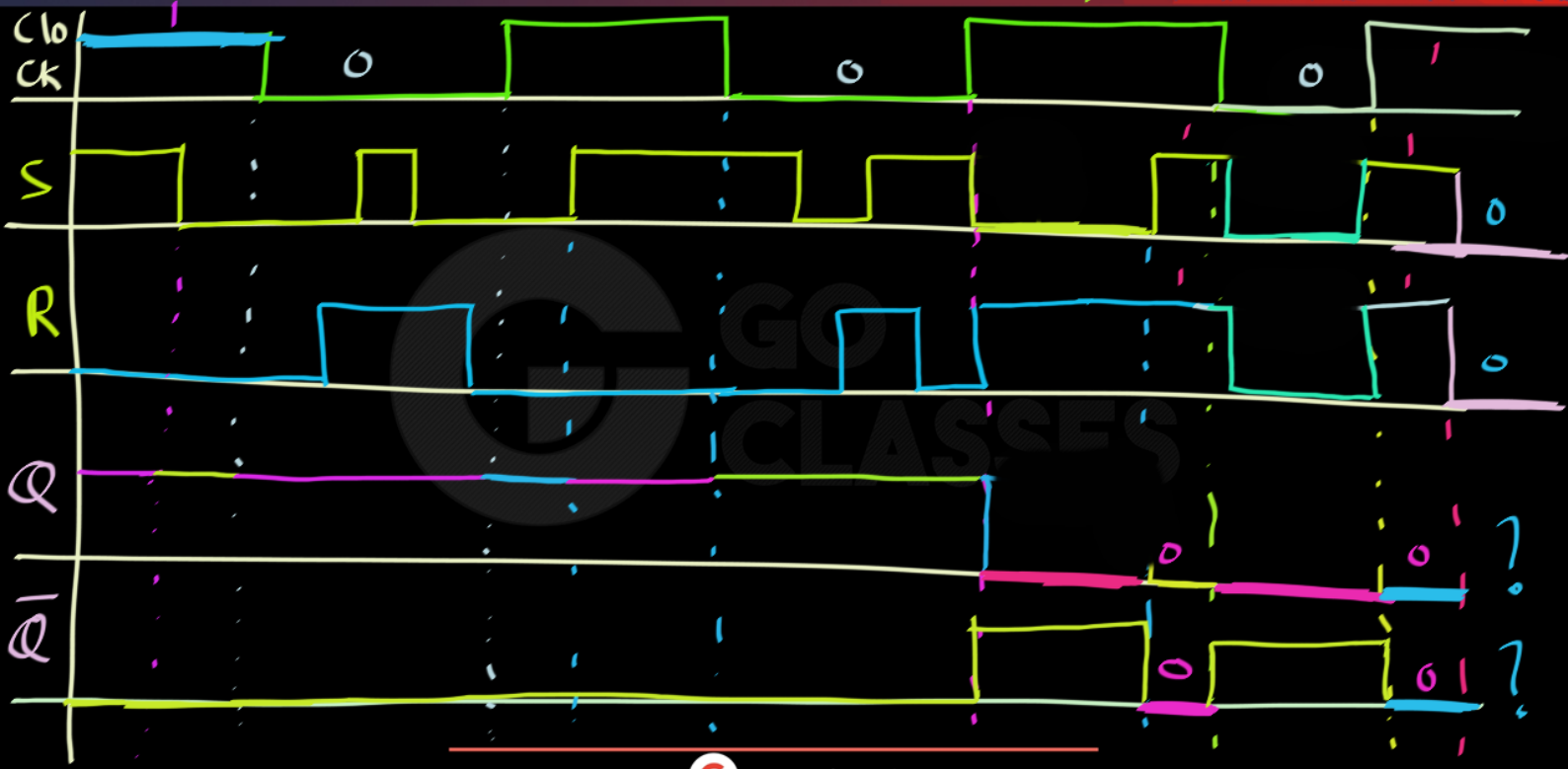
Circuit Diagram and Graphical Symbol for the Gated SR Latch



Timing Diagram for the Gated SR Latch






Clock
CK
S
R
Q
Q̄
0
0
0
0
0
0
0
0
1
?
?

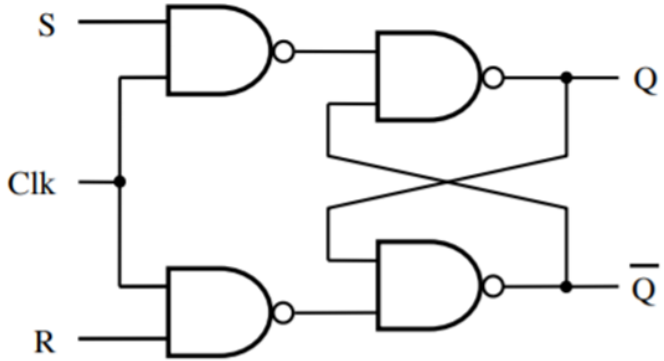
When Clock = 0 In SR flip flop

S R	S R
1 1 ⟹	0 0

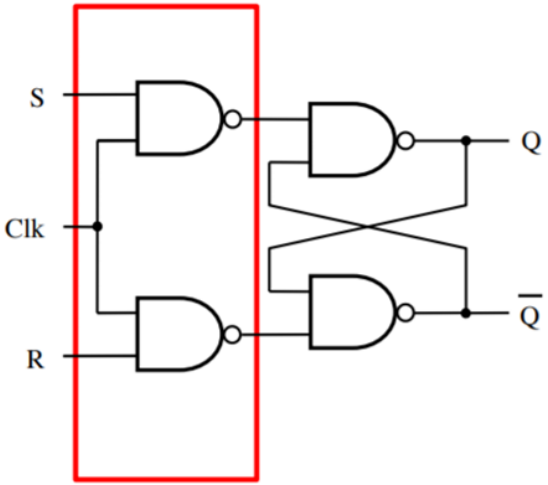
No problem

→ ff Does not Respond to input Change.

Gated SR latch with NAND gates

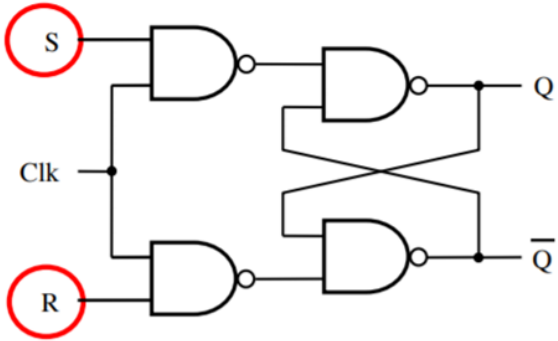


Gated SR latch with NAND gates



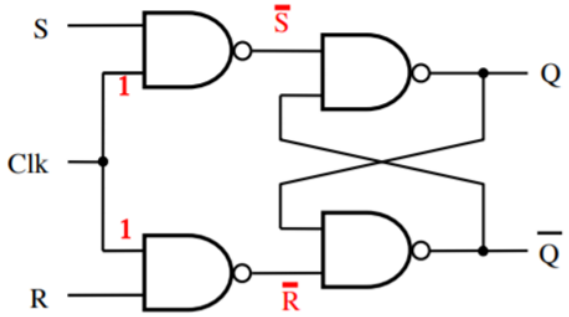
In this case the "gate" is constructed using NAND gates! Not AND gates.

Gated SR latch with NAND gates



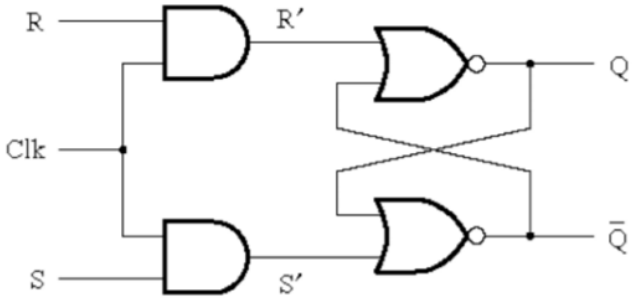
Also, notice that the positions of S and R are now swapped.

Gated SR latch with NAND gates

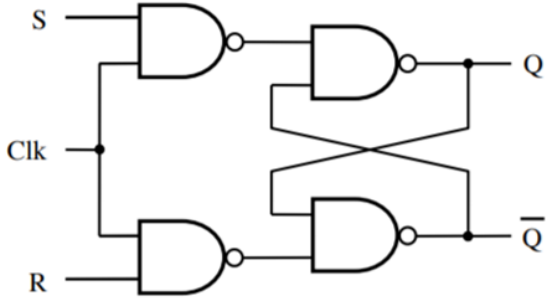


Finally, notice that when Clk=1 this turns into the basic latch with NAND gates, i.e., the $\overline{S}\overline{R}$ Latch.

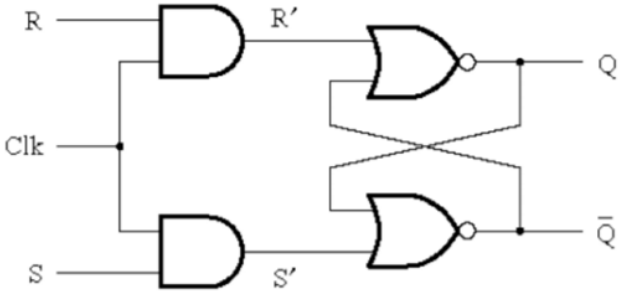
Gated SR latch with NOR gates

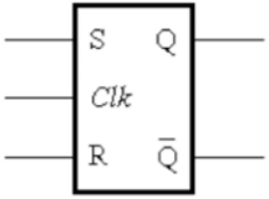


Gated SR latch with NAND gates

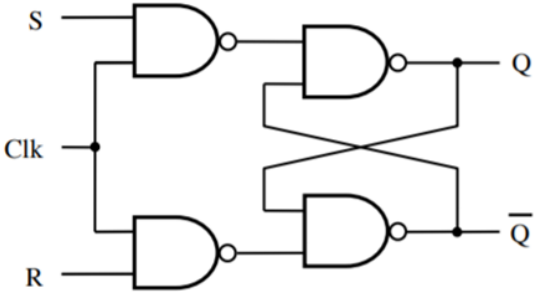


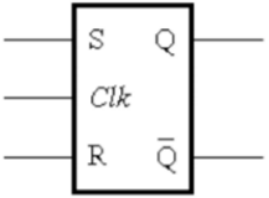
Gated SR latch with NOR gates





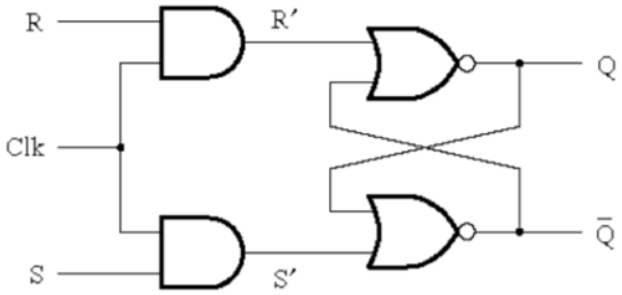
Gated SR latch with NAND gates





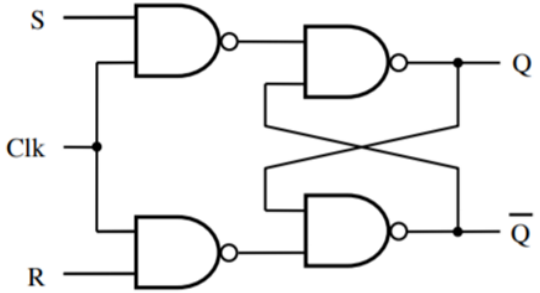
Graphical symbols are the same

Gated SR latch with NOR gates



Clk	S	R	Q(t + 1)
0	x	x	Q(t) (no change)
1	0	0	Q(t) (no change)
1	0	1	0
1	1	0	1
1	1	1	x (undesirable)

Gated SR latch with NAND gates



Clk	S	R	Q(t + 1)
0	x	x	Q(t) (no change)
1	0	0	Q(t) (no change)
1	0	1	0
1	1	0	1
1	1	1	x (undesirable)

Characteristic tables are the same

Next Topic:

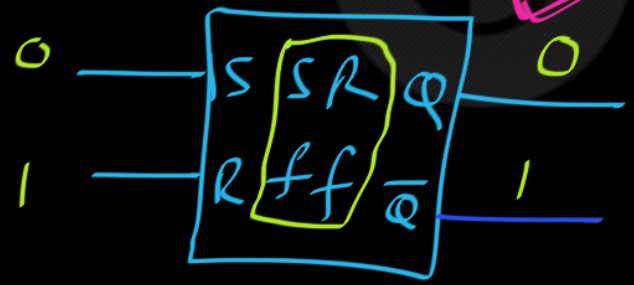
1. SR Flipflop

Truth Table, Characteristic Table,

Excitation Table, State Diagram

In Sequential Circuits :

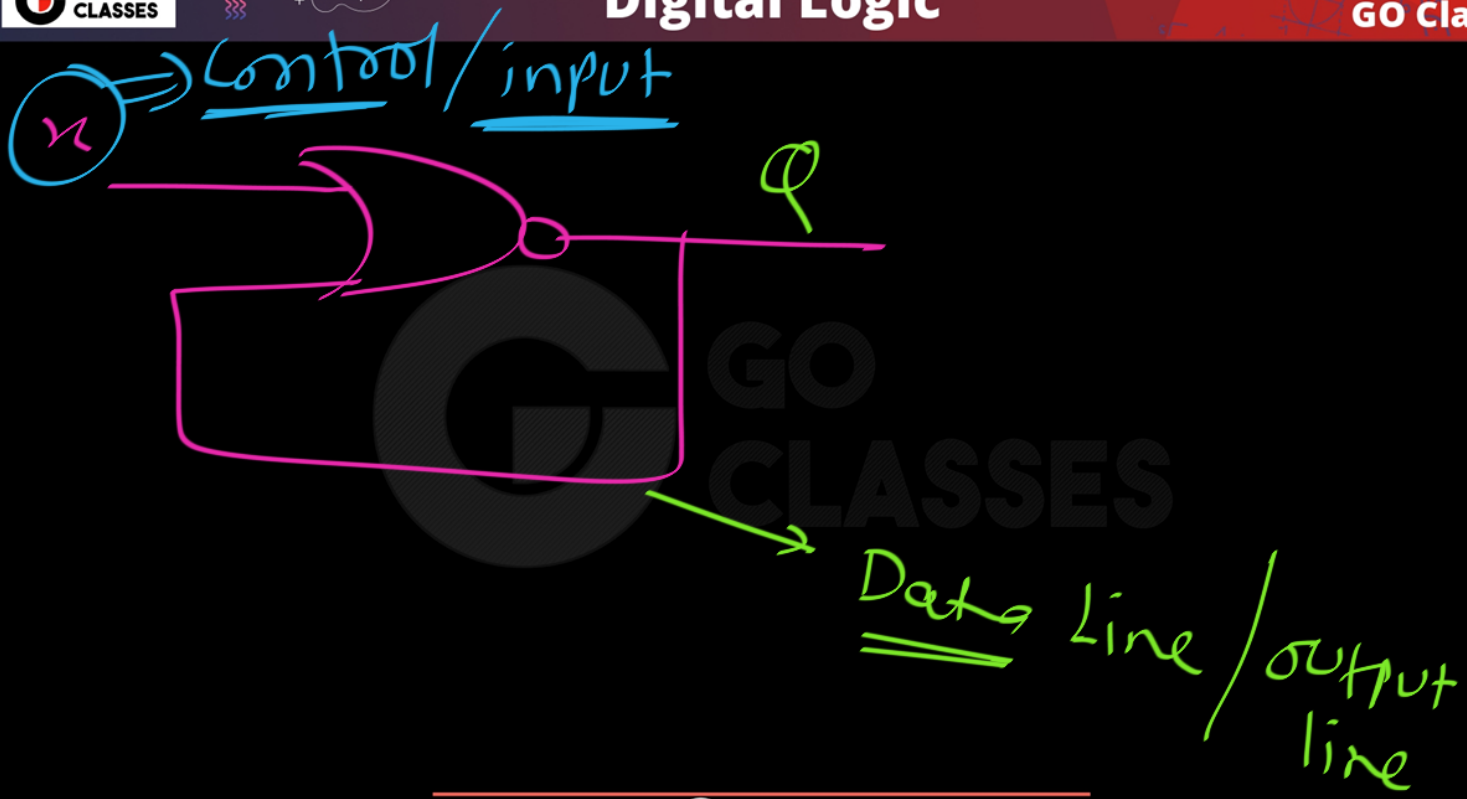
(1) State = Output = (Stored) Data

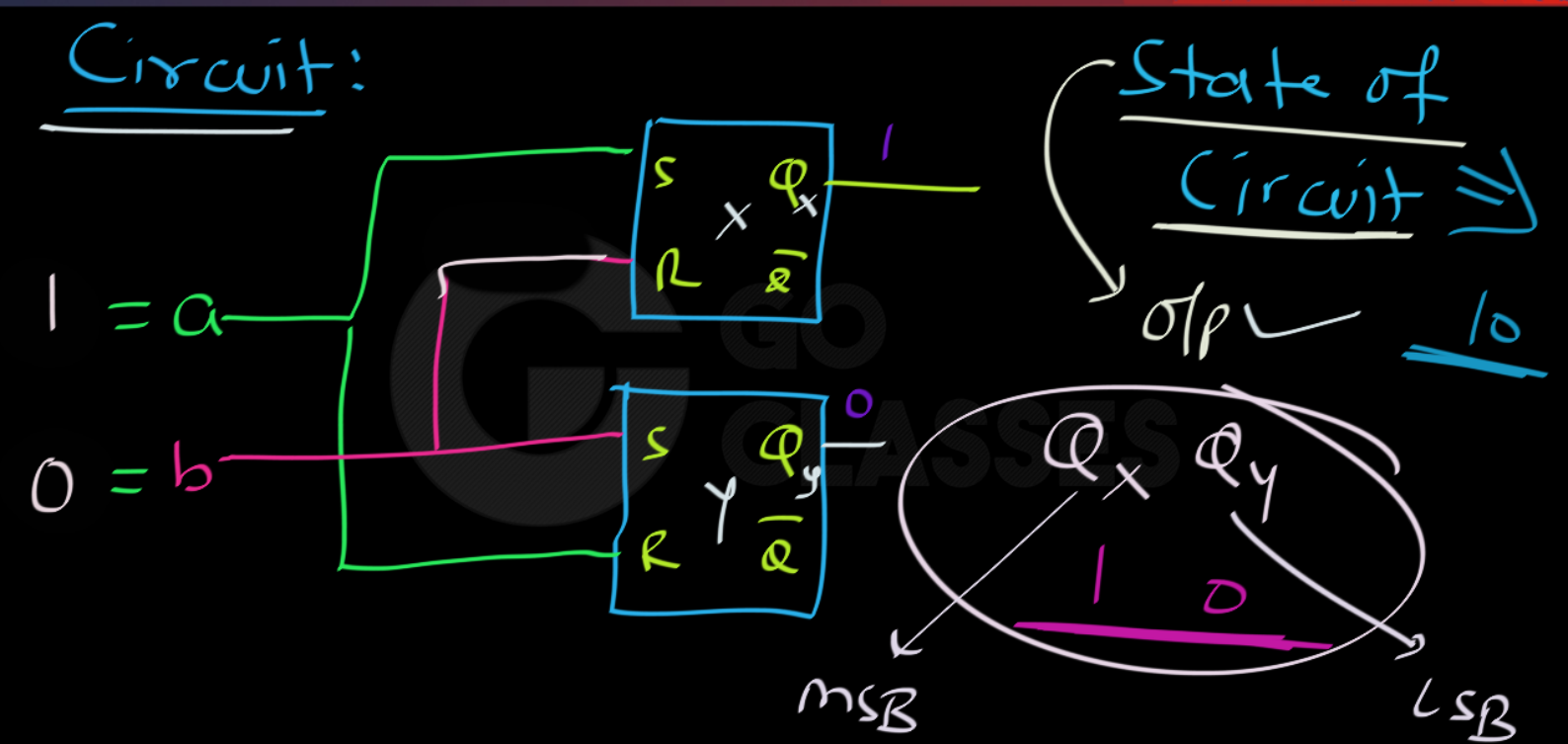


State of ff = 0

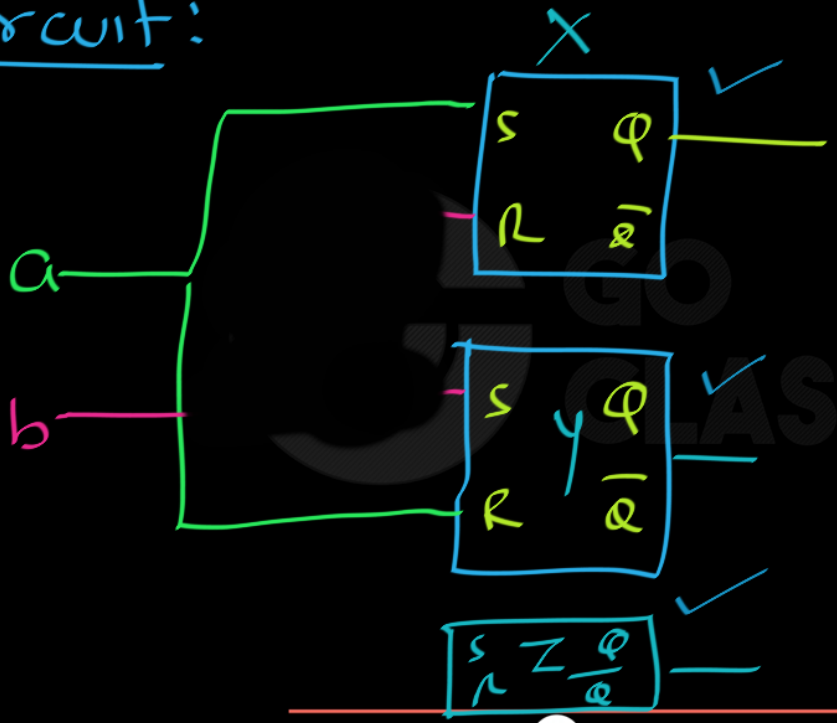
output " " = 0

Data (Stored) = 0


x
control / input
Q
Data line / output line


Circuit:
1 = a
0 = b
S
R
Qx
Q̄x
X
1
S
R
Qy
Q̄y
Y
0
State of Circuit ⇒
o/p
10
Qx Qy
1 0
MSB
LSB

Circuit:



o/p

states = 8

Q_x	Q_y	Q_z
0	0	0
0	1	0

Coming back to SR ff :

Clock	S	R	Q_{t+1}	
0	X	X	Q_t	Unchanged
1	0	0	Q_t	Retain
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	X	

⇒ Characteris-tic table (behaviour)

Truth Table: $Q_{(t+1)} = f(R, S, Q_t, \text{Clock})$

16 Rows

6 Rows ← 0 X X X | Q_t

Normally ff works

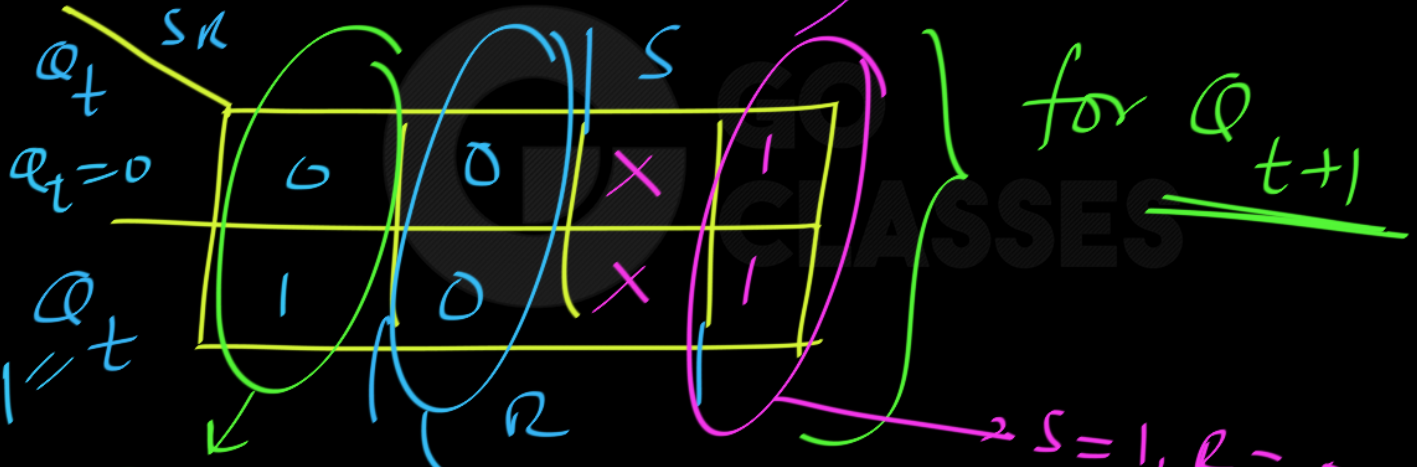
Clock	R	S	Q_t	Q_{t+1}
1	0	0	0	0
1	0	0	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	x
1	1	1	1	x

Coming back to SR ff :

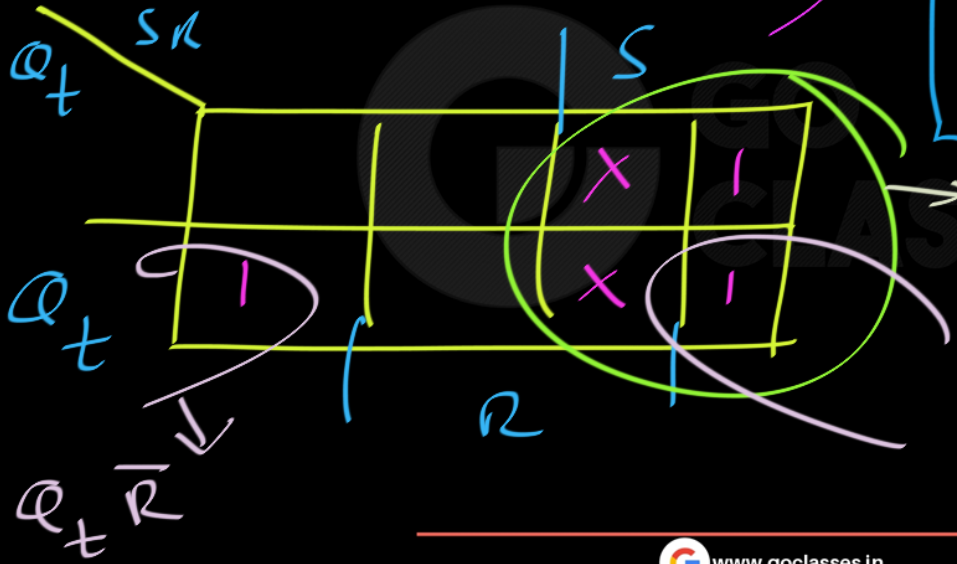
Clock	S	R	Q_{t+1}	
0	X	X	Q_t	Unchanged
1	0	0	Q_t	Retain
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	X	

⇒ Characteristic table (behaviour)

$Q_{t+1} = f(S, R, Q_t)$



$Q_{t+1} = f(S, R, Q_t)$



minimum SOP

$$Q_{t+1} = S + Q_t \bar{R}$$

$$Q_{t+1} = S + Q_t \overline{R}$$

$$Q_n = S + Q \overline{R}$$

Q_n → next State

Q → Present State

$Q_{t+1} = S + Q_t \overline{R}$

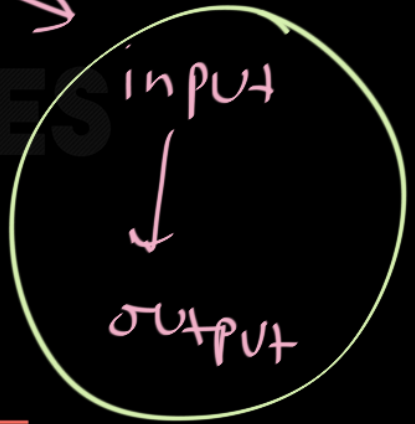
Prev. o/p feeding back

3rd implementation of SR ff:

Q: What is this circuit?

Characteristic table : (from input point of view)

S	R	Q_{t+1}
1	0	1
0	1	0
0	0	Q_t



Excitation table : (from output Point of View)

Present Q_t	Next Q_{t+1}	S	R
0	0	Reset / Retain	
0	1	Set	
1	0	Reset	
1	1	Set / Retain	

Excitation table : (from output Point of View)

Present Q_t	Next Q_{t+1}	S	R
0	0	0	1
		0	0
0	1	1	0
1	0	0	1
1	1	0	0
		1	0

In SR ff

Excitation table : (from output Point of View)

$Q_t \rightarrow Q_{t+1}$	S	R
$0 \rightarrow 0$	0	X
$0 \rightarrow 1$	1	0
$1 \rightarrow 0$	0	1
$1 \rightarrow 1$	X	0

Excitation Table

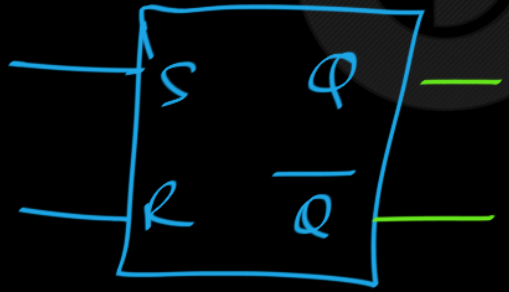
Excitation Table:

If o/p changes $Q_t \longrightarrow Q_{t+1}$ then what input combination can cause this o/p change?

State Diagram:

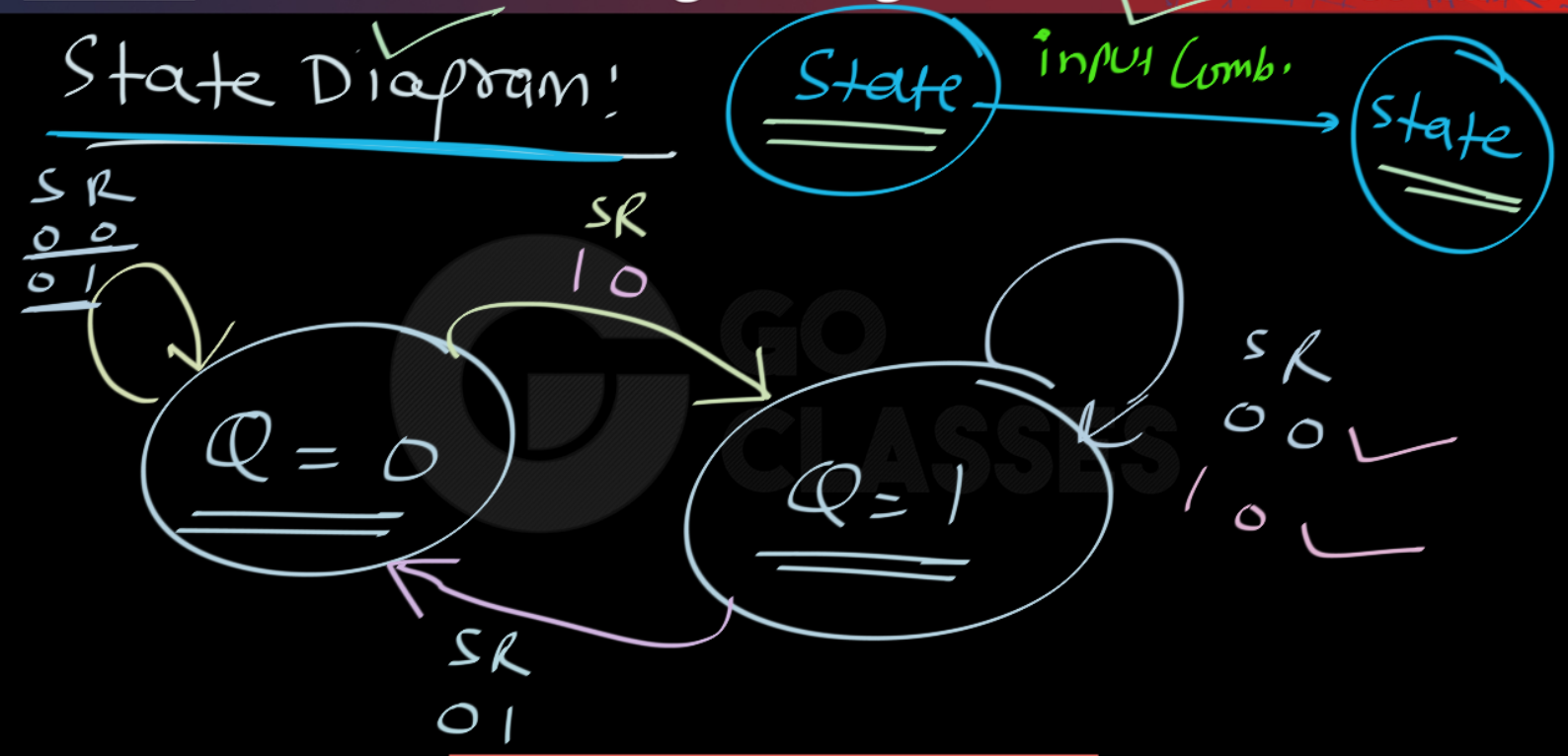
SR ff : ⇒ 2 states

o/p

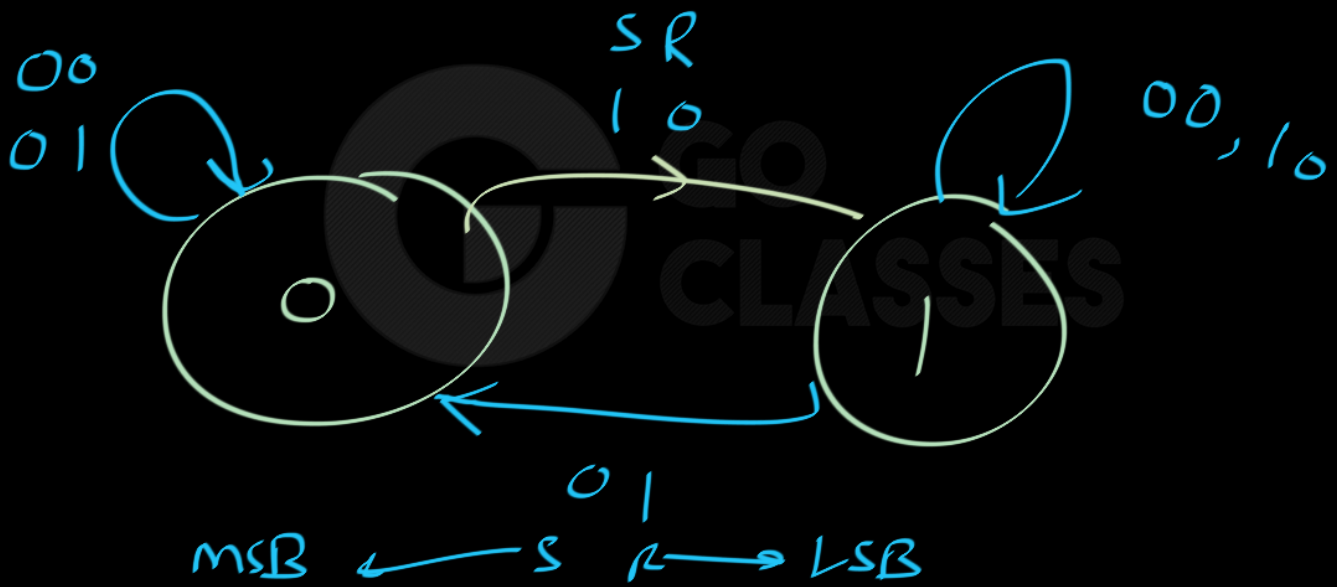


Q = 0 ✓

Q = 1 ✓


State Diagram:
State
Input Comb.
State
S R
0 0
0 1
S R
1 0
Q = 0
Q = 1
S R
0 0
1 0
S R
0 1

SR ff State Diagram!



SR ff:

Definition / behaviour

1. Char table
2. Timing Diagram, State Diagram
3. 3 implementations (standard)
4. Truth Table, Excitation Table,
5. $Q_{t+1} = S + \bar{R} Q_t$ (o/p equation)

$Q_{t+1} = S + \bar{R} Q_t$ → State equation

SR ff: → AND, OR implementation

more implementation:

- mux
- Decoder
- some other ff
- - - - - - -

$Q_{t+1} = S + \bar{R}Q_t$ → State equation

System Clock ⟶ fixed for all components

2.4 GHz

FF ✓

Register ✓

Counter ✓

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	X

Set

Reset

K-map for $Q_{t+1} = f(S, R, Q_t)$

Next State

Present State

SR

Q_t

$Q_t = 0$

$Q_t = 1$

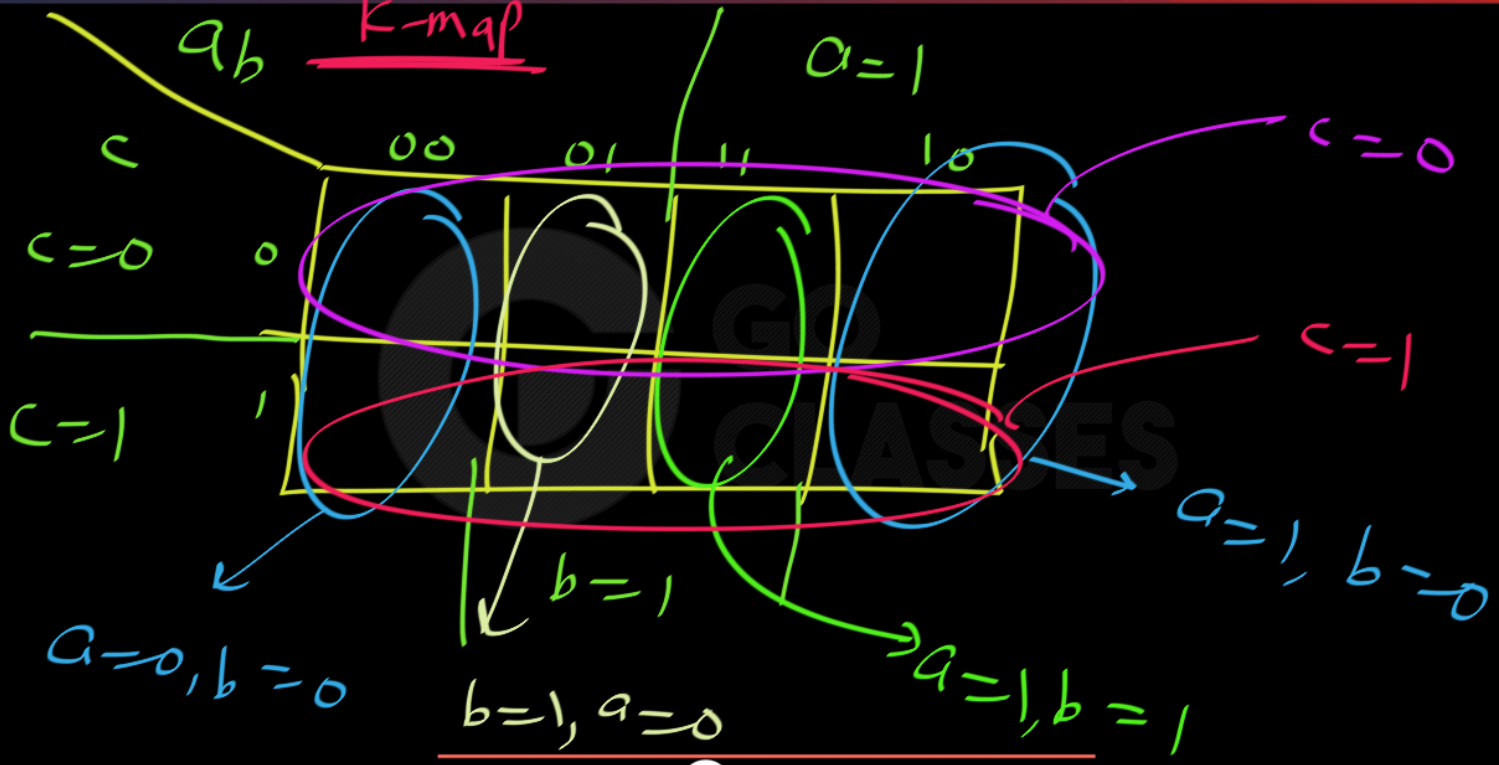
	00	01	11	10
0	0	0		
1	1	0		

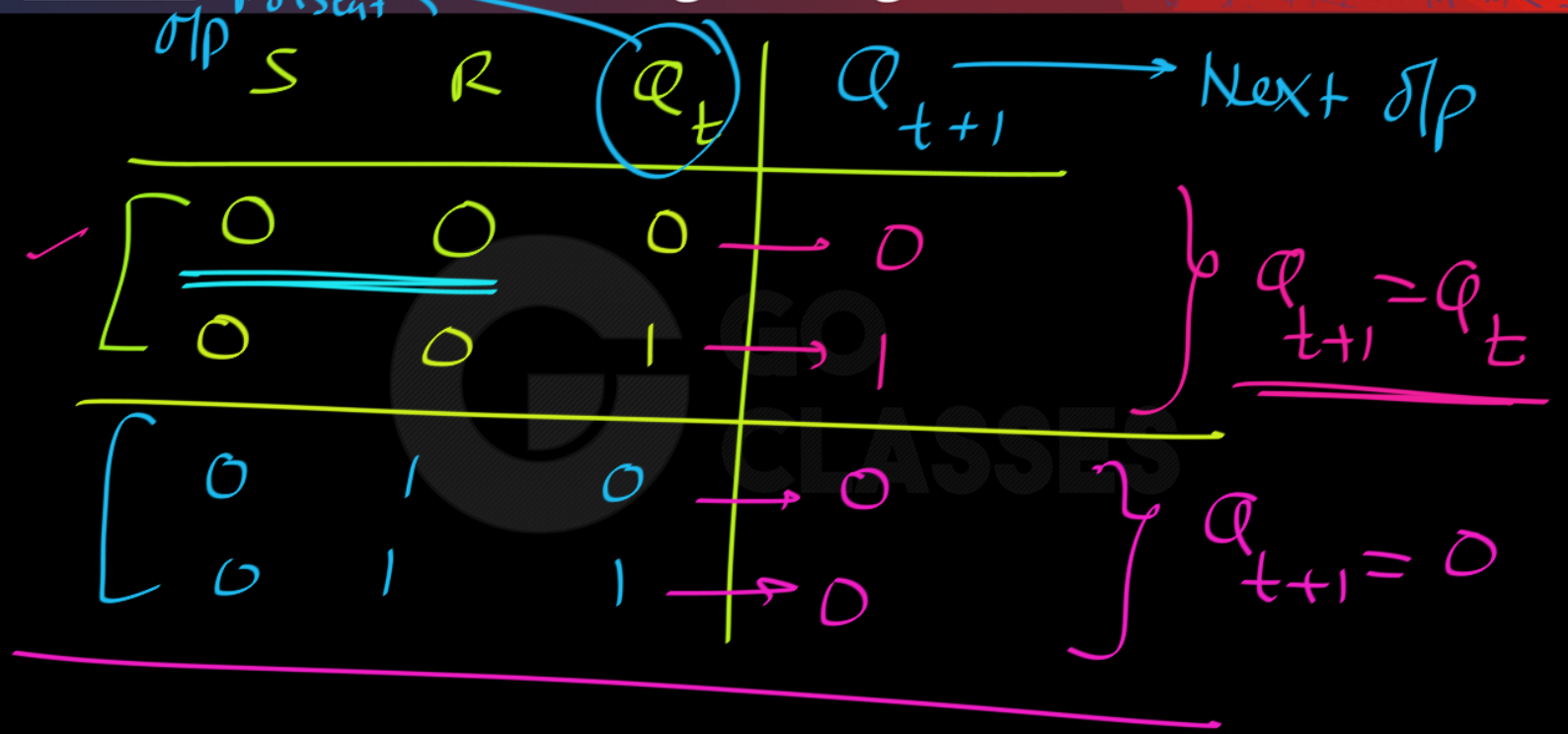
$Q_{t+1} = Q_t$

S=0, R=0

R=1

R=1, S=0

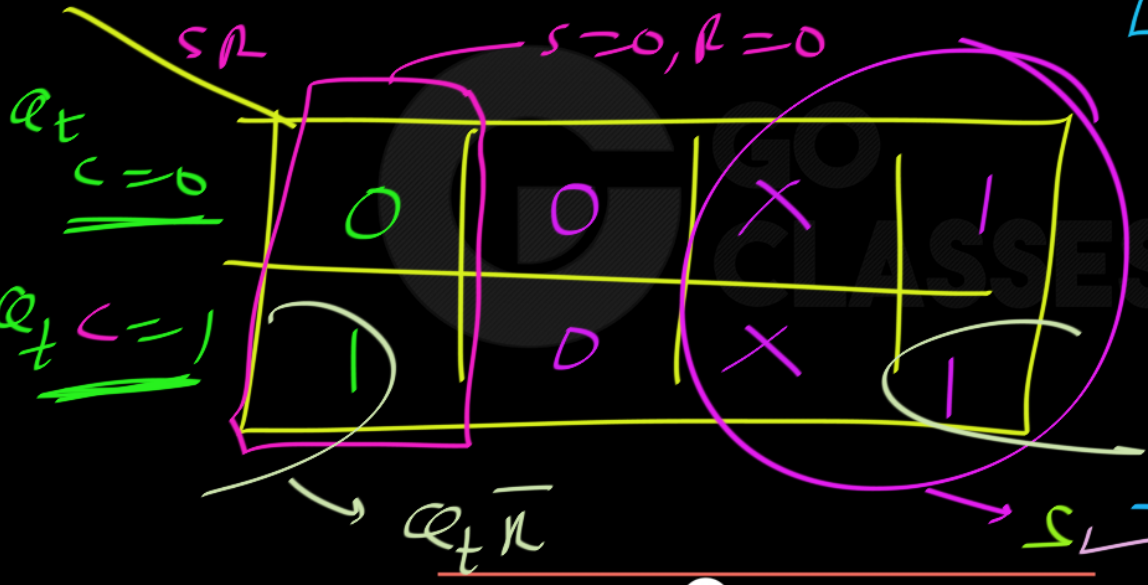

ab
K-map
a=1
c
00
01
11
10
c=0
0
c=0
c=1
1
c=1
a=1, b=0
b=1
a=0, b=0
a=1, b=1
b=1, a=0


Present
o/p
S
R
Q_t
Q_{t+1}
Next o/p
0 0 0 → 0
0 0 1 → 1
$Q_{t+1} = Q_t$
0 1 0 → 0
0 1 1 → 0
$Q_{t+1} = 0$

$Q_{t+1} = f$; $Q_t = c$;

$S=0, R=0$

$Q_{t+1} = f = Q_t = c$



k-map for f

$Q_{t+1} = f$

State Equation / o/p Equation:

$$Q_{t+1} = S + Q_t \overline{R}$$

$$Q_n = S + Q\overline{R}$$

} Same