CS 221

DIGITAL DESIGN

LECTURE : 20

DATE : 01/10/2013

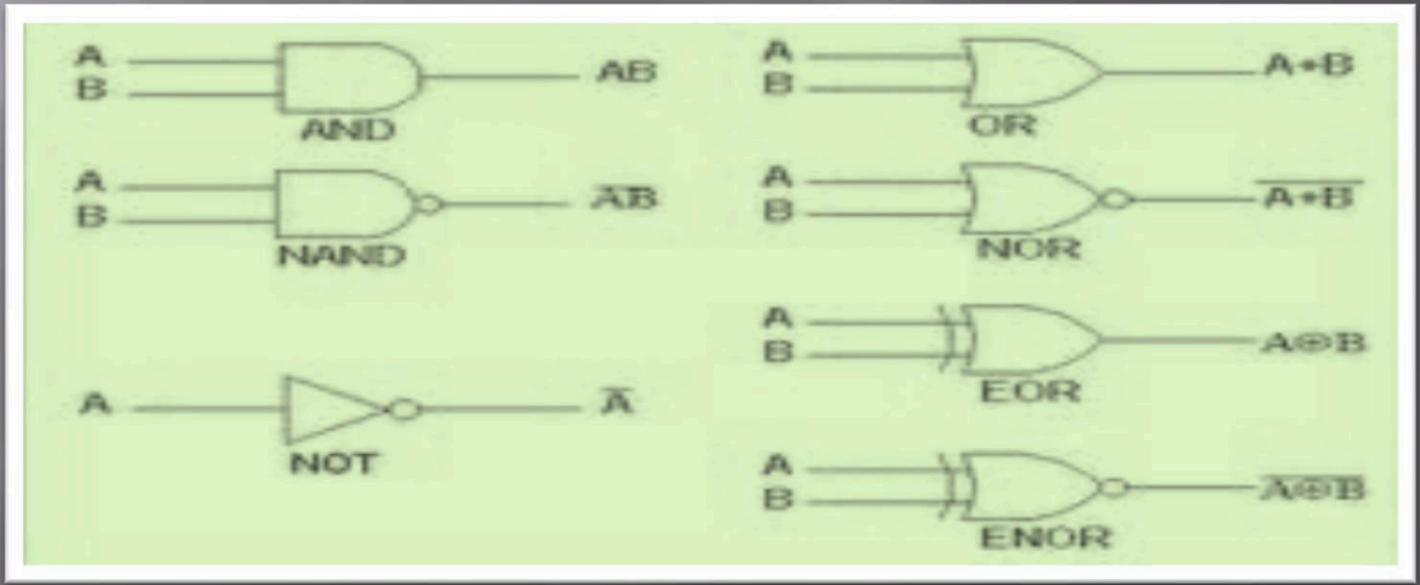
SUBMITTED BY:

ABHILASH KUMAR JHA	120123001
PRANVENDRA CHATURVEDI	120123028
SHUBHAM PANDEY	120123037

SEQUENTIAL LOGIC DESIGN

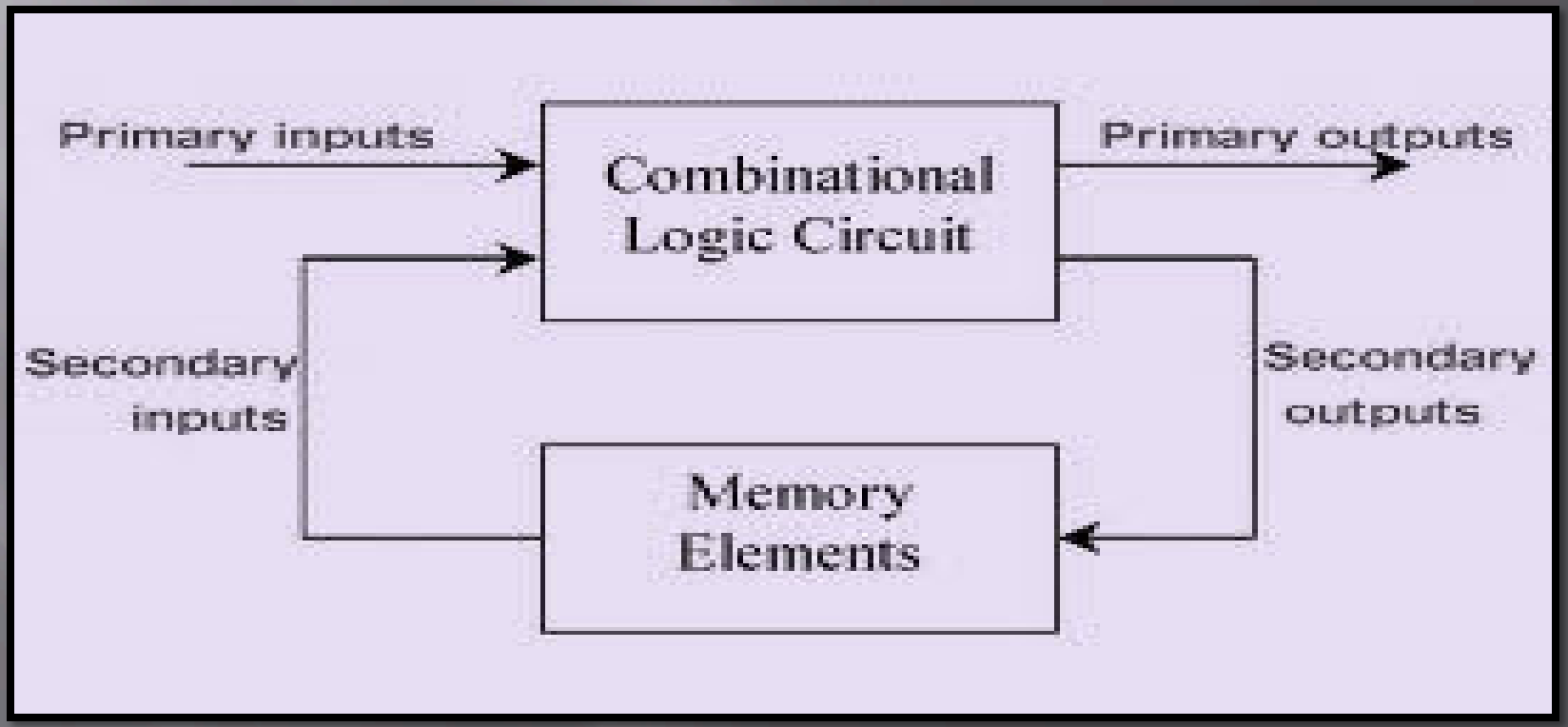
WHY ?

Combinatorial Circuits, discussed earlier, do not have memory – their functionality is decided purely by the current values (1/0) applied to the circuit.



INTRODUCTION

In digital circuit theory, **sequential logic** is a type of logic circuit whose output depends not only on the present value of its input signals but on the past history of its inputs.

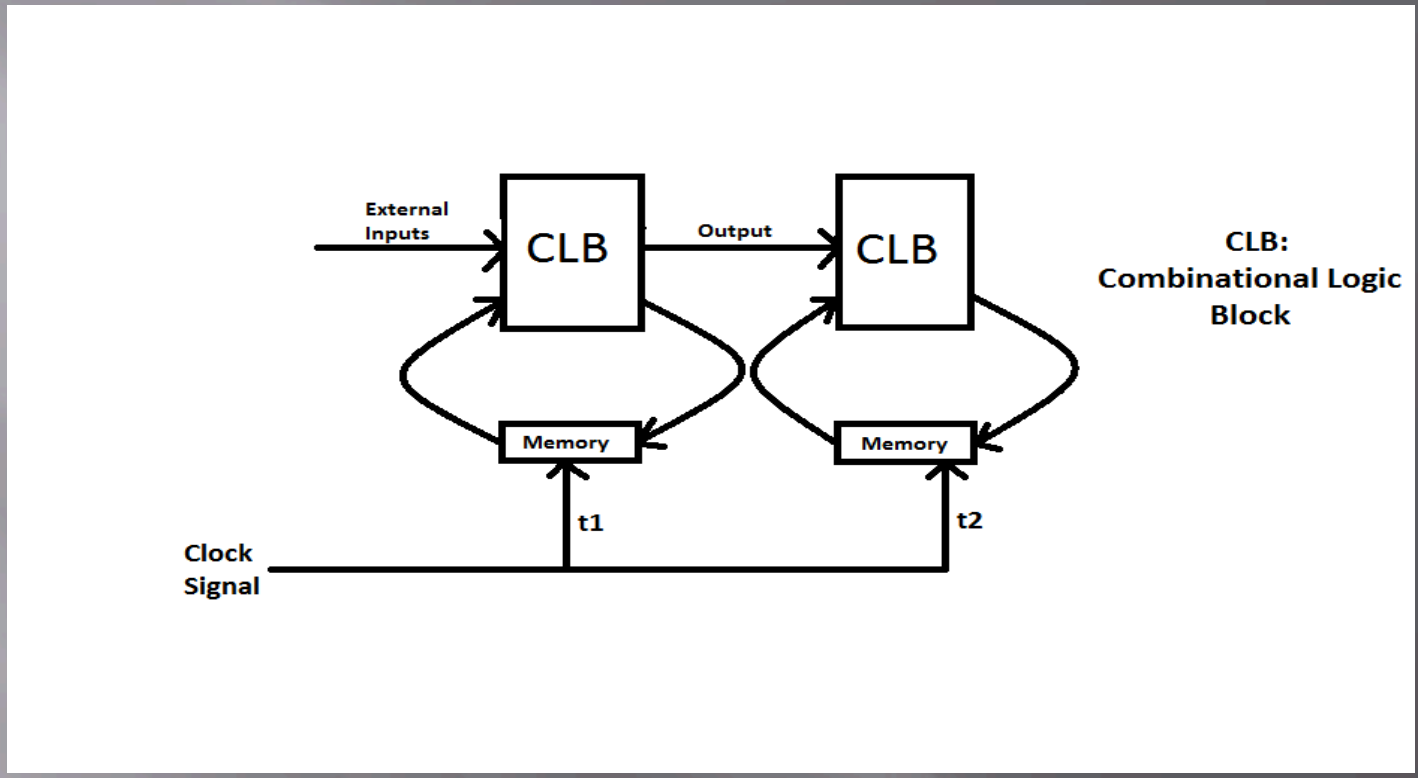


TYPES

There are two types of sequential logic circuits. Synchronous & Asynchronous

SYNCHRONOUS LOGIC CIRCUITS

A synchronous sequential circuit employs signals that affect the storage elements at only discrete instants of time. In circuit there are parts that are synchronized by a clock signal. In an ideal synchronous circuit, every change in logical level of its storage components is simultaneous. These transitions follow the level of change of a special signal called as the *clock*. Ideally, the input to each storage element has reached its final value before the next clock occurs, so the behavior of the whole circuit can be predicted exactly. Practically, some delay is required for each logical operation, resulting in a maximum speed at which each synchronous system can run.



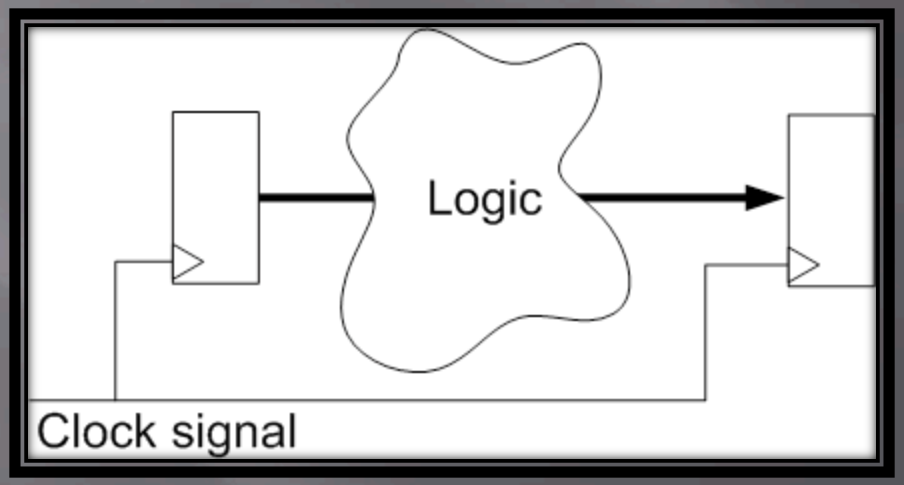


Fig: Synchronous Logic Circuits

ASYNCHRONOUS LOGIC CIRCUITS

An **asynchronous circuit**, or **self-timed** circuit, is a sequential digital logic circuit which is not governed by a clock circuit or global clock signal. Instead they often use signals that indicate completion of instructions and operations, specified by simple data transfer protocols. This type is contrasted with a synchronous circuit in which changes to the signal values in the circuit are triggered by repetitive pulses called a clock signal. Most digital devices today use synchronous circuits. However asynchronous circuits have the potential to be faster, and may also have advantages in lower power consumption, lower electromagnetic interference, and better modularity in large systems.

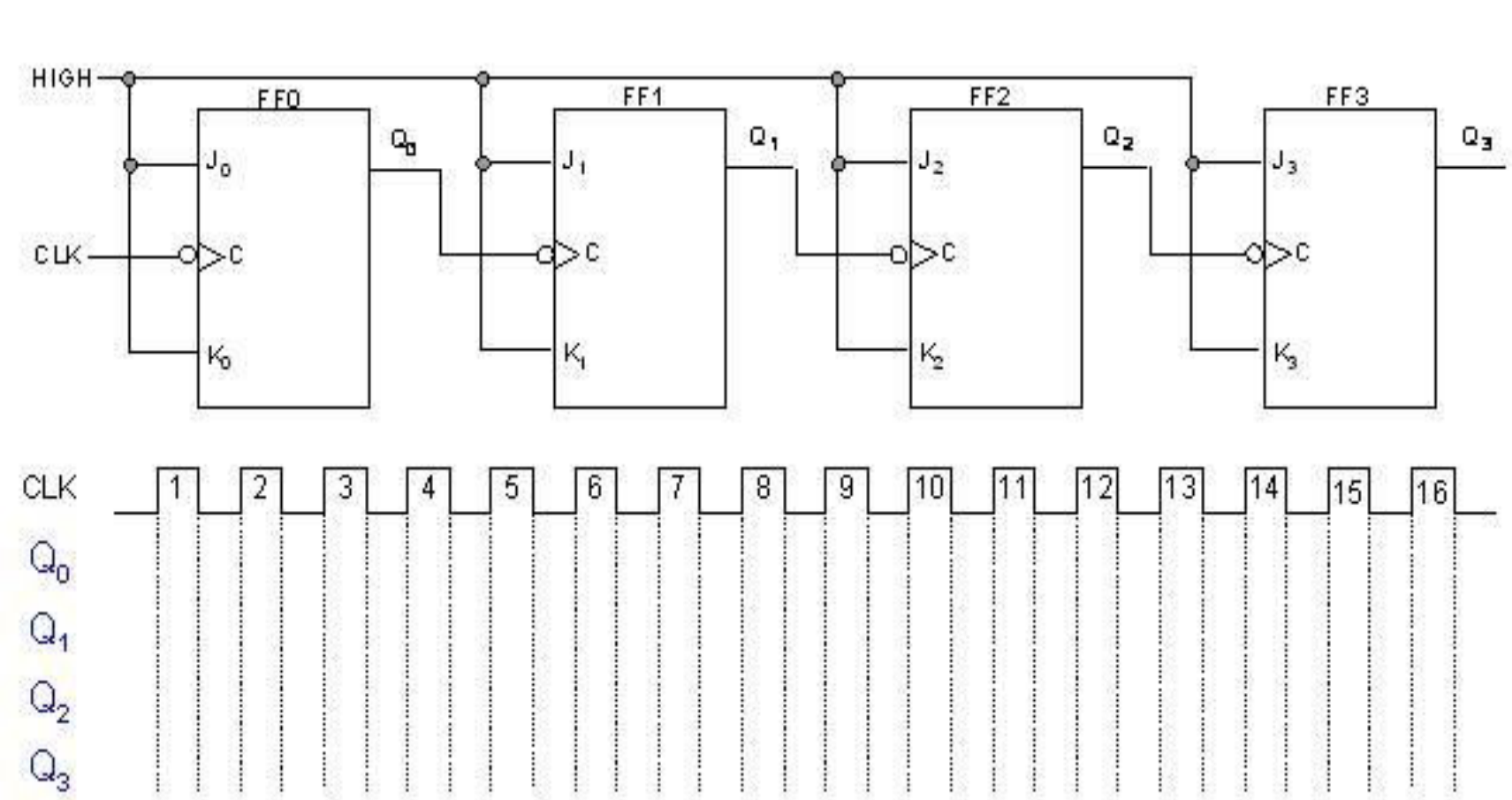


Fig . 4-bit asynchronous binary counter

A SIMPLE MEMORY ELEMENT

1) CASCADED INVERTER

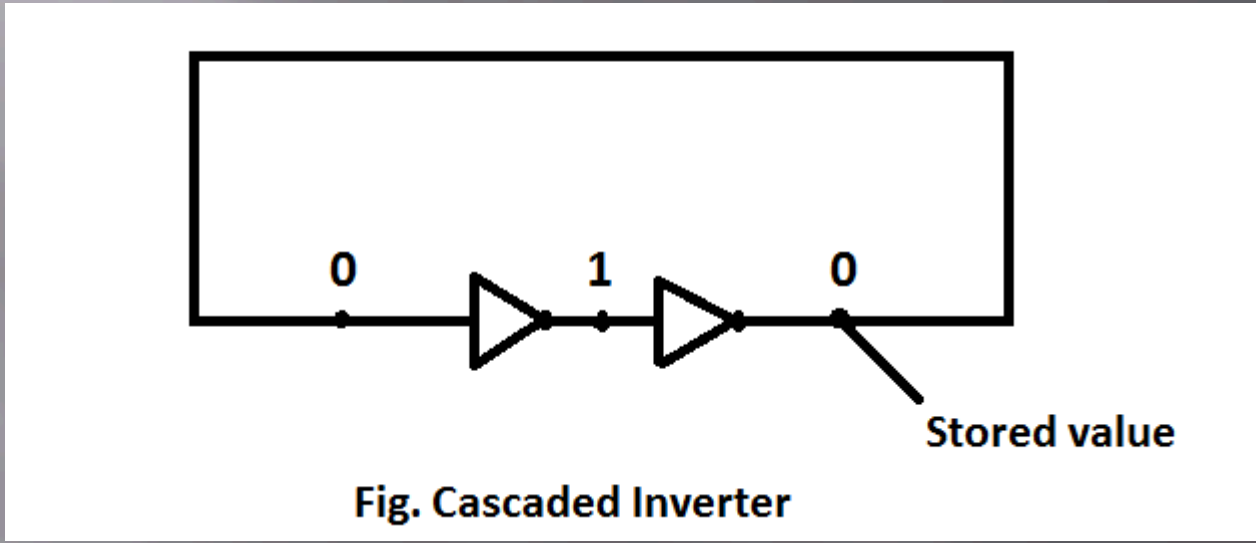
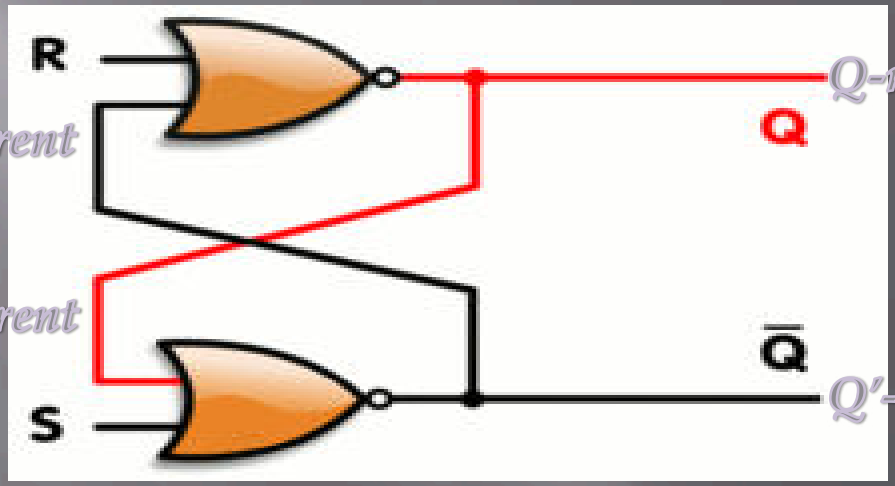


Fig. Cascaded Inverter

It stores 0 for 0 as input & 1 for 1 as input.
Drawback : We can't decide what value to store.

2) CROSS COUPLED NOR GATE (SET RESET LATCH)



		OBSERVATION
a)	R=0, S= 1	Q=1 (Set)
b)	R=1, S= 0	Q=0 (Reset)
c)	R=0, S= 0	Q=Q (Remember)
d)	R=1, S= 1	Invalid input

INPUT		OUTPUT	
R	S	Q	Q′
0	1	1	0
1	0	0	1
0	0	No Change	No Change
1	1	Invalid Input	Invalid Input

Output Equations for SR Latch

- To figure out how Q and Q′ change, we have to look at not only the inputs S and R, but also the current values of Q and Q′:

Q next = (R + Q′ current)′

Q′ next = (S + Q current)′

Input combination: SR = 00

- What if S = 0 and R = 0? The equations on the right reduce to:

Q next= (0 + Q′ current)′ = Q current . Q′ next= (0 + Q current)′ = Q′ current

- So when SR = 00, then Q next= Q current
- Whatever value Q has, it keeps
- So we have unit that can retain values

Input combination: SR = 10

- What if S = 1 and R = 0?
- Since S = 1, Q′ next is 0, regardless of Q current

Q′ next= (1 + Q current)′ = 0

- This new value of Q′ goes into the bottom NOR gate, along with R = 0 Q next= (0 + 0)′ = 1
- So when SR = 10, then Q′ next= 0 and Q next= 1
- This is how you set the flip-flop to 1. The S input stands for "set"
- In a physical circuit, there will usually be a delay from the time S becomes 1 to the time Q next becomes 1
- But once Q next becomes 1, the outputs will stop changing .This is a "stable state"

Input Combination: SR = 01

- What if S = 0 and R = 1?
- Since R = 1, Q next is 0, regardless of Q current:

Q next= (1 + Q′ current)′ = 0

- This new value of Q goes into the top NOR gate, where S= 0 , then Q′ next = (0 + 0)′ = 1
- So when SR = 01, then Q next = 0 and Q′ next= 1
- This is how you reset, or clear, the flip-flop to 0. The R input stands for "reset"
- Again, there will be delays before a change in R propagates to the output Q′ next, but eventually it will become stable

Input Combination: SR = 11

- Both Q next and Q′ next will become 0

  Q = Q′ = 0, which is logically inconsistent . It violates our assumption
- What happens if we then make S = 0 and R = 0 together

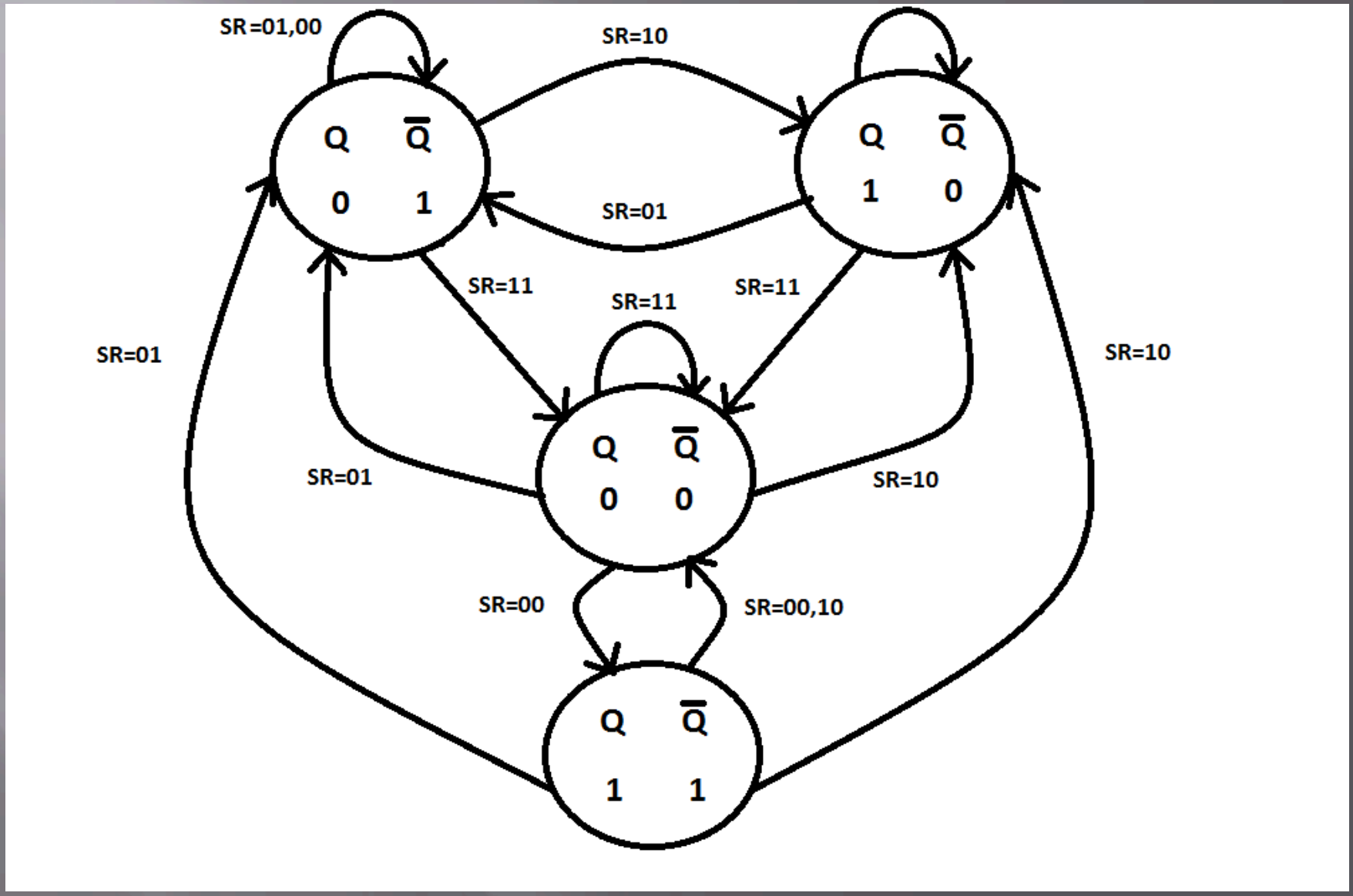
Q next= (0 + 0)′ = 1

Q′ next = (0 + 0)′ = 1
- But these new values go back into the NOR gates, and in the next step we get:

Q next = (0 + 1)′ = 0

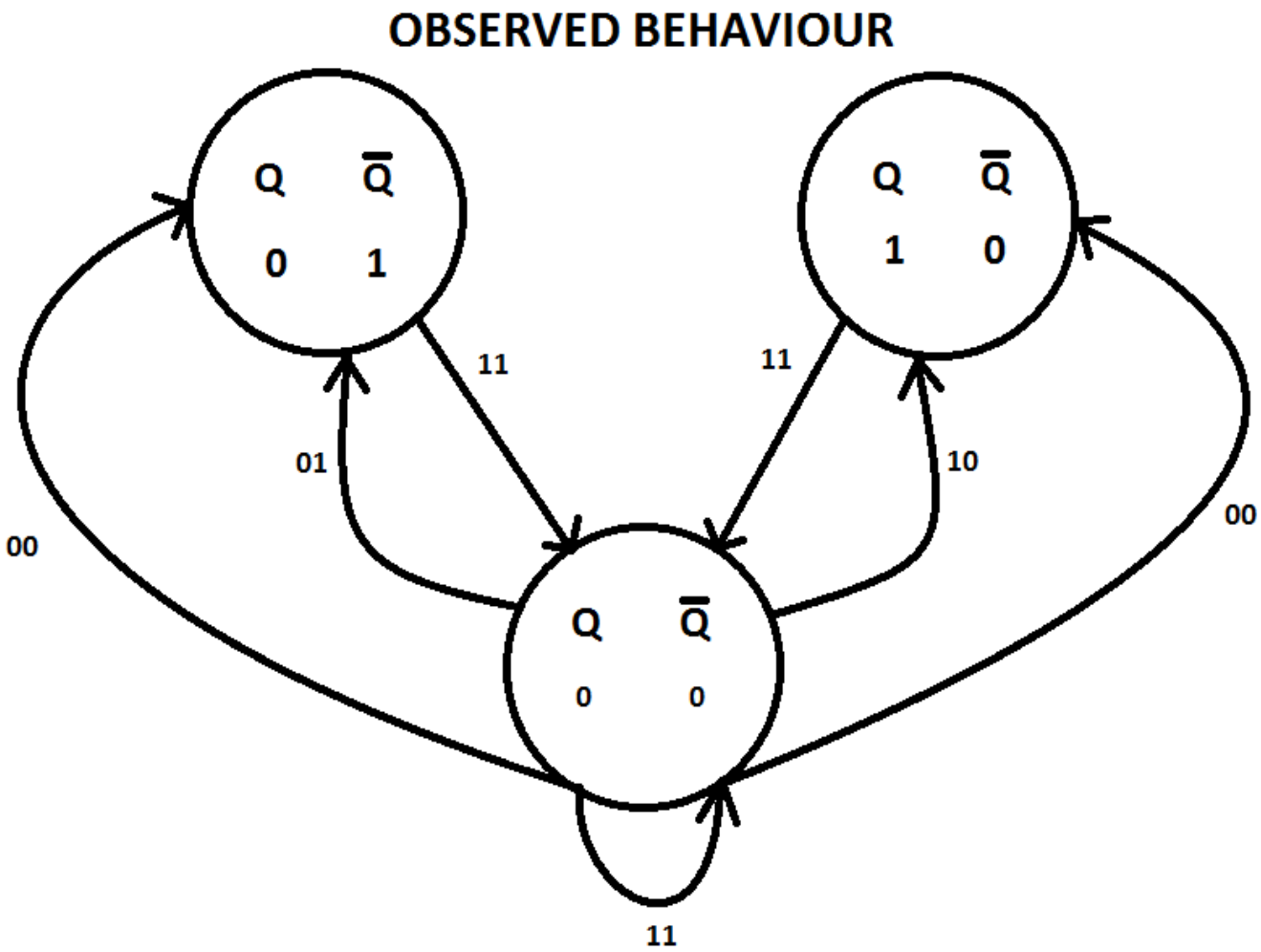
Q′ next = (0 + 1)′ = 0
- The circuit enters an infinite loop, where Q and Q′ cycle between 0 and 1 forever. This is called Race Condition.
- This is actually the worst case, but the moral is don't ever set SR=11!

STATE DIAGRAM



Theoretical

OBSERVED BEHAVIOUR



TIMING DIAGRAM