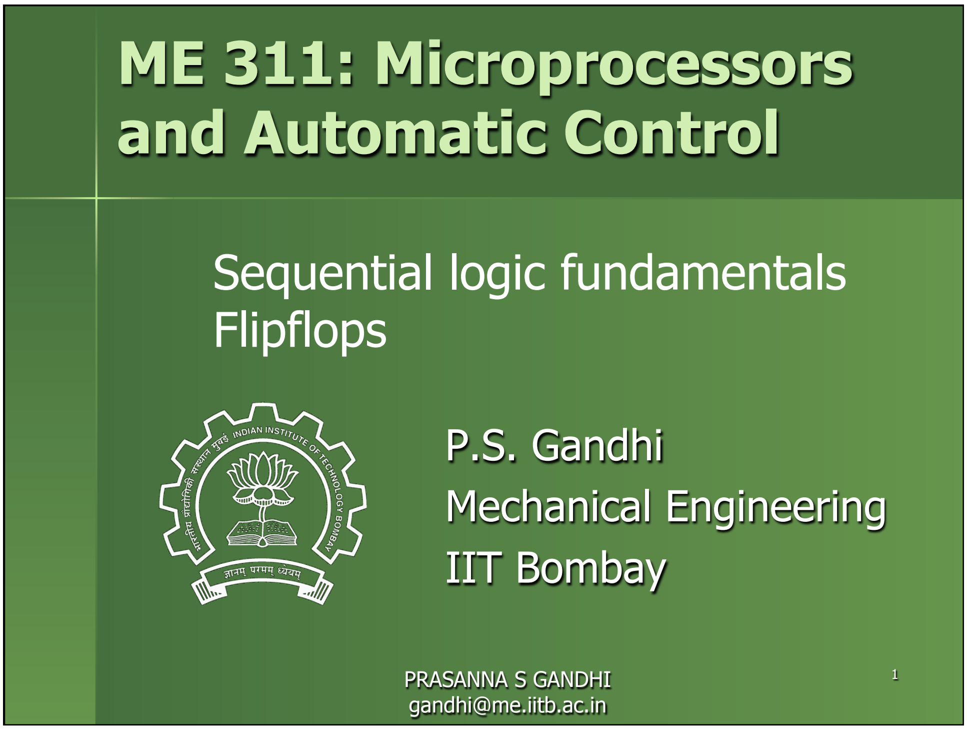
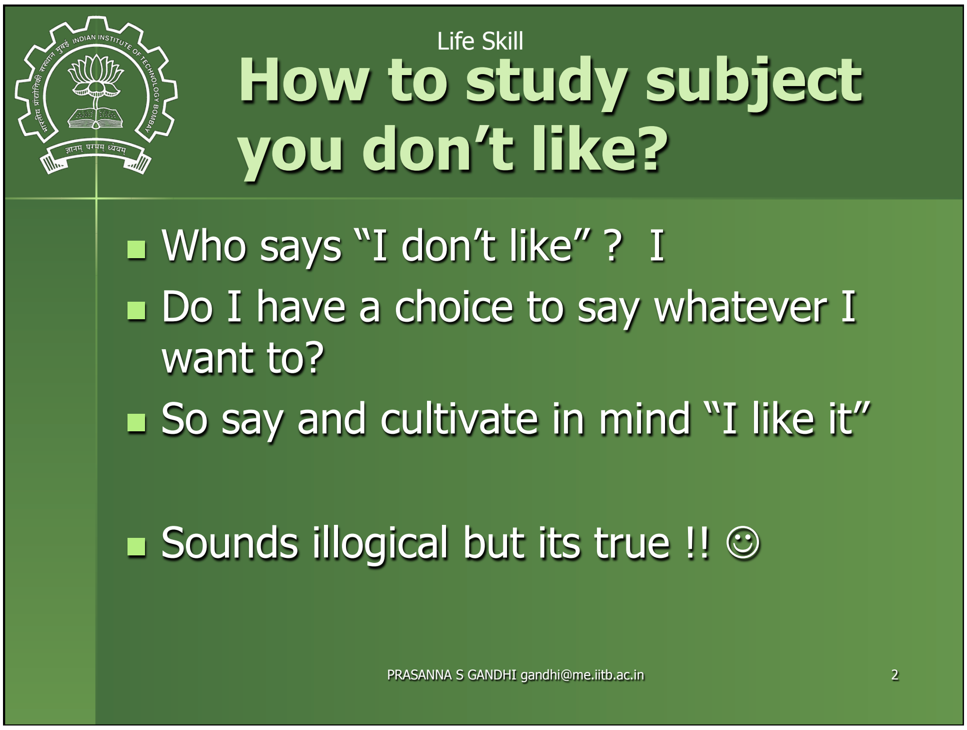


ME 311: Microprocessors and Automatic Control

Sequential logic fundamentals
Flipflops

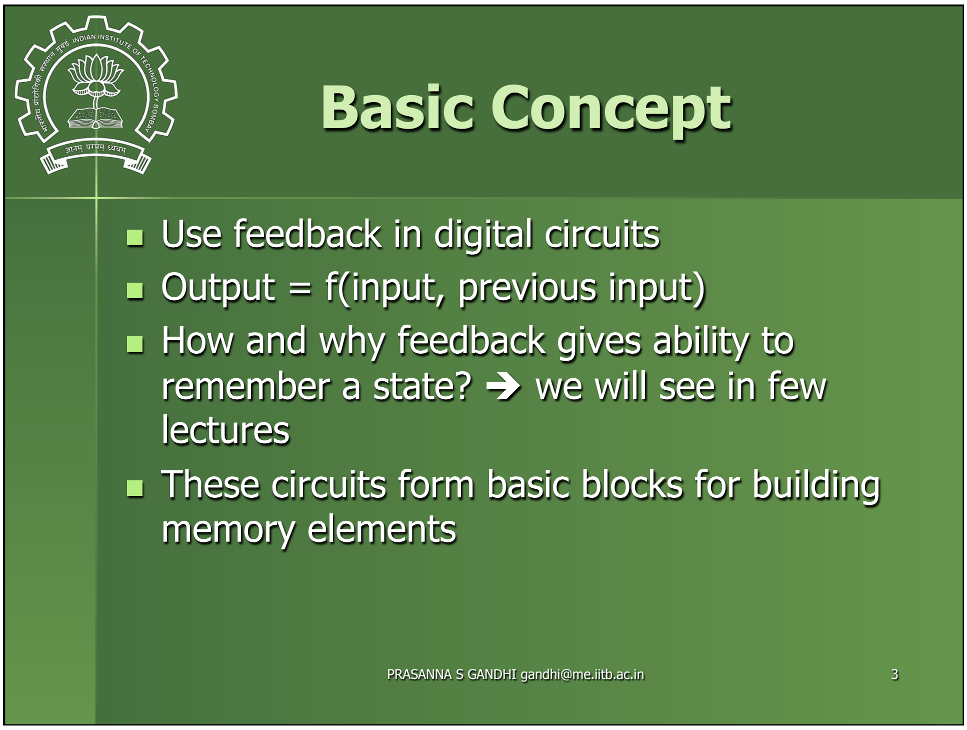
P.S. Gandhi
Mechanical Engineering
IIT Bombay



Life Skill

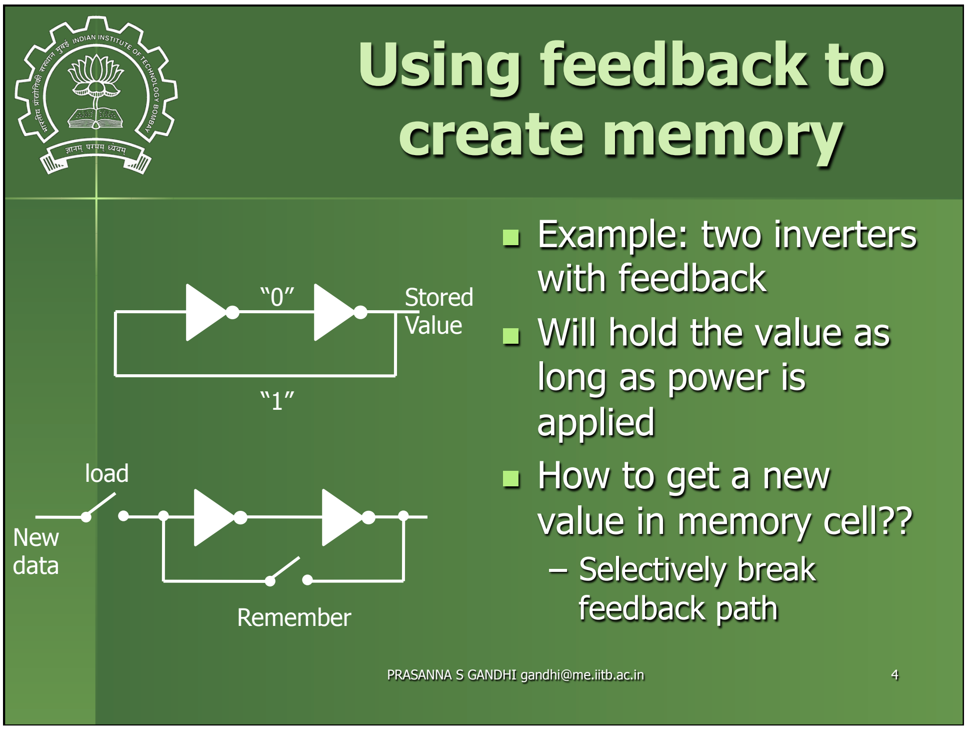
How to study subject you don't like?

- Who says "I don't like" ? I
- Do I have a choice to say whatever I want to?
- So say and cultivate in mind "I like it"
- Sounds illogical but its true !! ☺



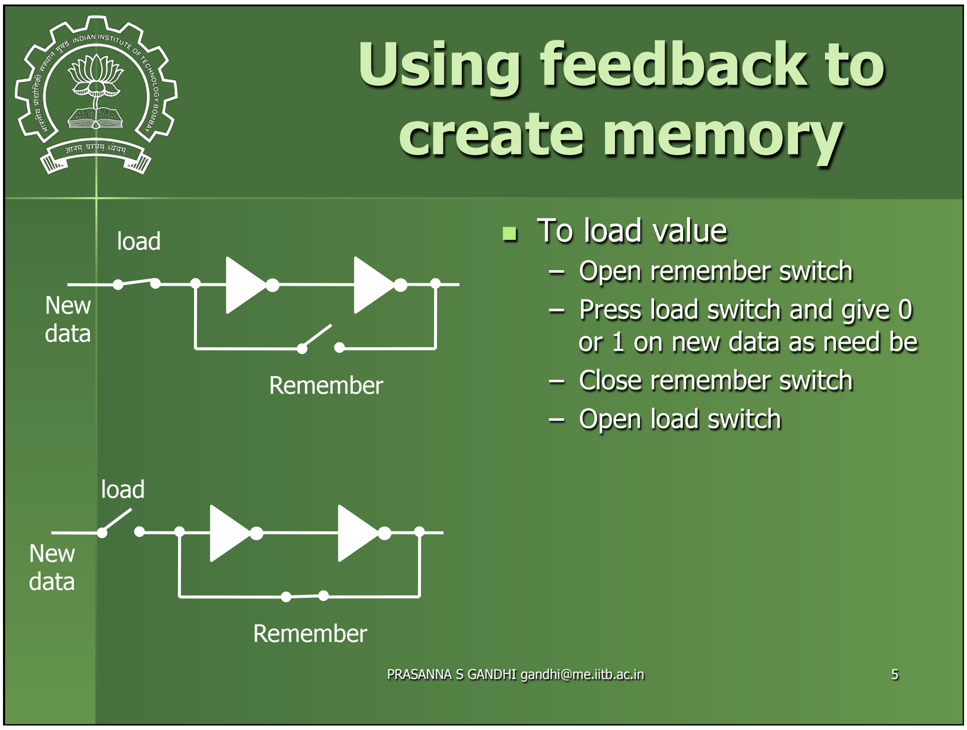
Basic Concept

- Use feedback in digital circuits
- Output = f(input, previous input)
- How and why feedback gives ability to remember a state? ➔ we will see in few lectures
- These circuits form basic blocks for building memory elements



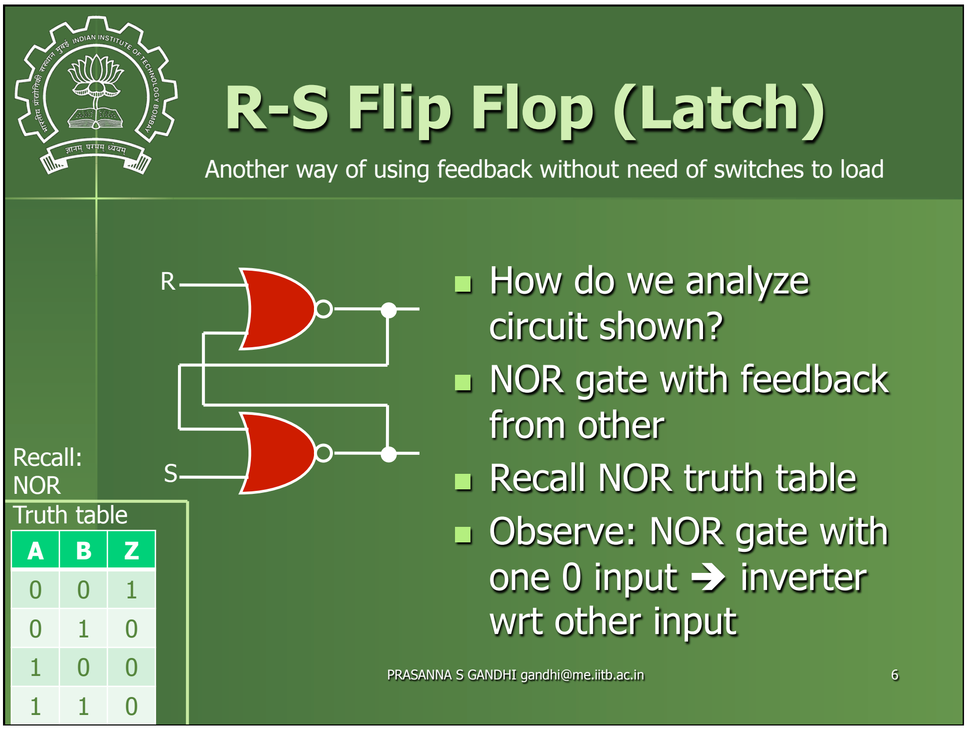
Using feedback to create memory

- Example: two inverters with feedback
- Will hold the value as long as power is applied
- How to get a new value in memory cell??
  - Selectively break feedback path



Using feedback to create memory

- To load value
  - Open remember switch
  - Press load switch and give 0 or 1 on new data as need be
  - Close remember switch
  - Open load switch



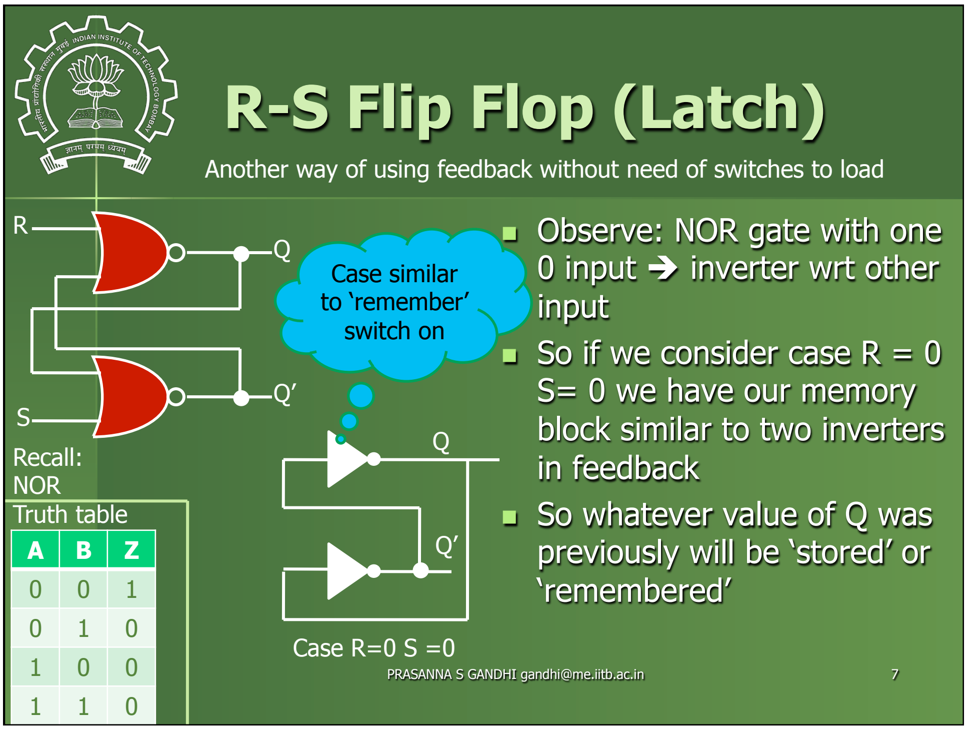
R-S Flip Flop (Latch)

Another way of using feedback without need of switches to load

- How do we analyze circuit shown?
- NOR gate with feedback from other
- Recall NOR truth table
- Observe: NOR gate with one 0 input ➔ inverter wrt other input

Recall: NOR Truth table

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0



R-S Flip Flop (Latch)

Another way of using feedback without need of switches to load

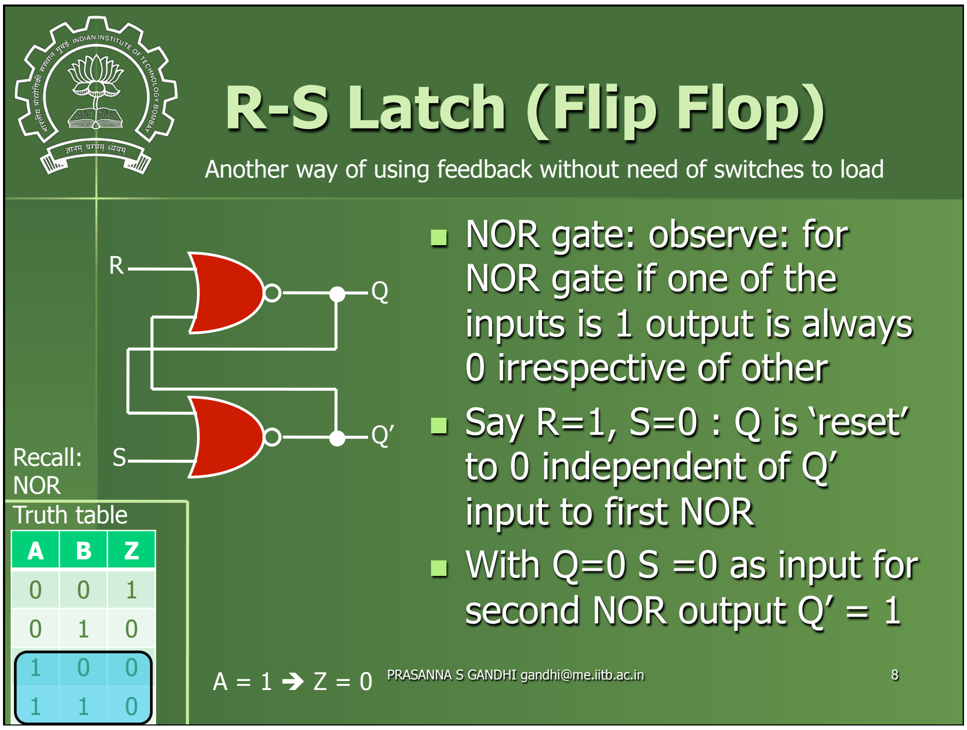
R
Q
S
Q'
Case similar to 'remember' switch on
Q
Q'

Case R=0 S =0

- Observe: NOR gate with one 0 input ➔ inverter wrt other input
- So if we consider case R = 0 S= 0 we have our memory block similar to two inverters in feedback
- So whatever value of Q was previously will be 'stored' or 'remembered'

Recall: NOR Truth table

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0



R-S Latch (Flip Flop)

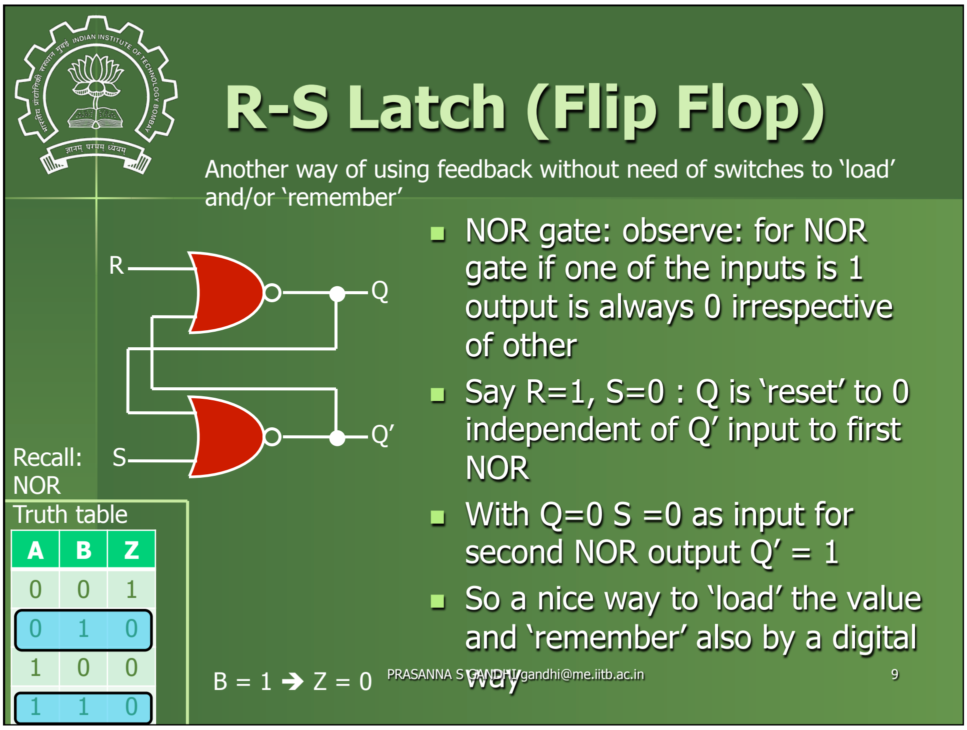
Another way of using feedback without need of switches to load

- NOR gate: observe: for NOR gate if one of the inputs is 1 output is always 0 irrespective of other
- Say R=1, S=0 : Q is 'reset' to 0 independent of Q' input to first NOR
- With Q=0 S =0 as input for second NOR output Q' = 1

Recall: NOR Truth table

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

A = 1 ➔ Z = 0



R-S Latch (Flip Flop)

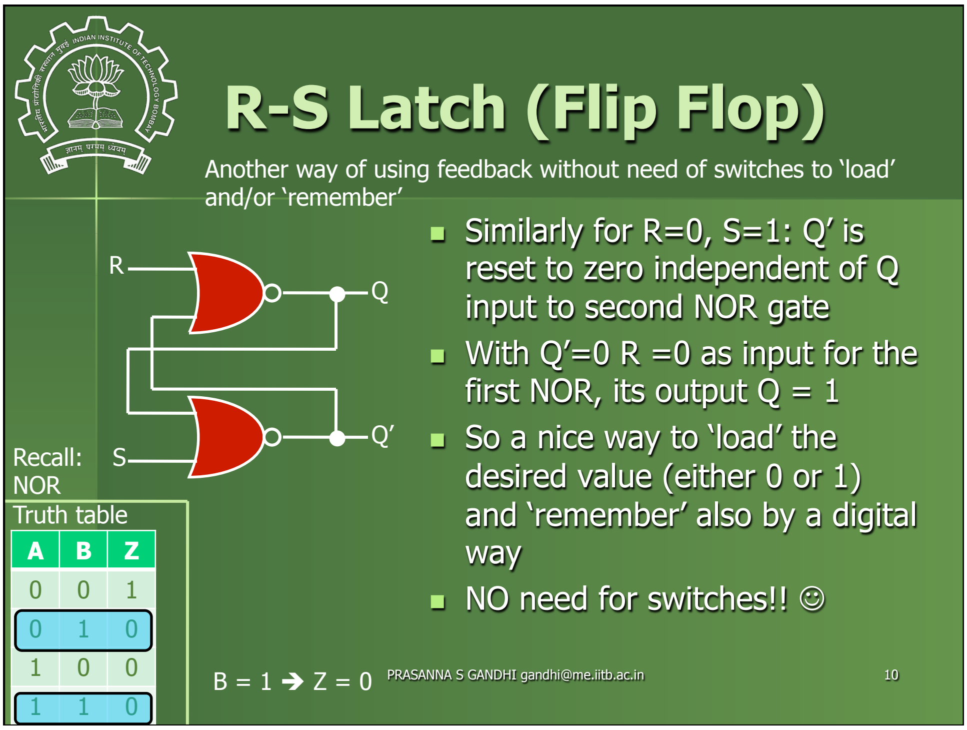
Another way of using feedback without need of switches to 'load' and/or 'remember'

- NOR gate: observe: for NOR gate if one of the inputs is 1 output is always 0 irrespective of other
- Say R=1, S=0 : Q is 'reset' to 0 independent of Q′ input to first NOR
- With Q=0 S =0 as input for second NOR output Q′ = 1
- So a nice way to 'load' the value and 'remember' also by a digital way

Recall: NOR Truth table

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

B = 1 → Z = 0



R-S Latch (Flip Flop)

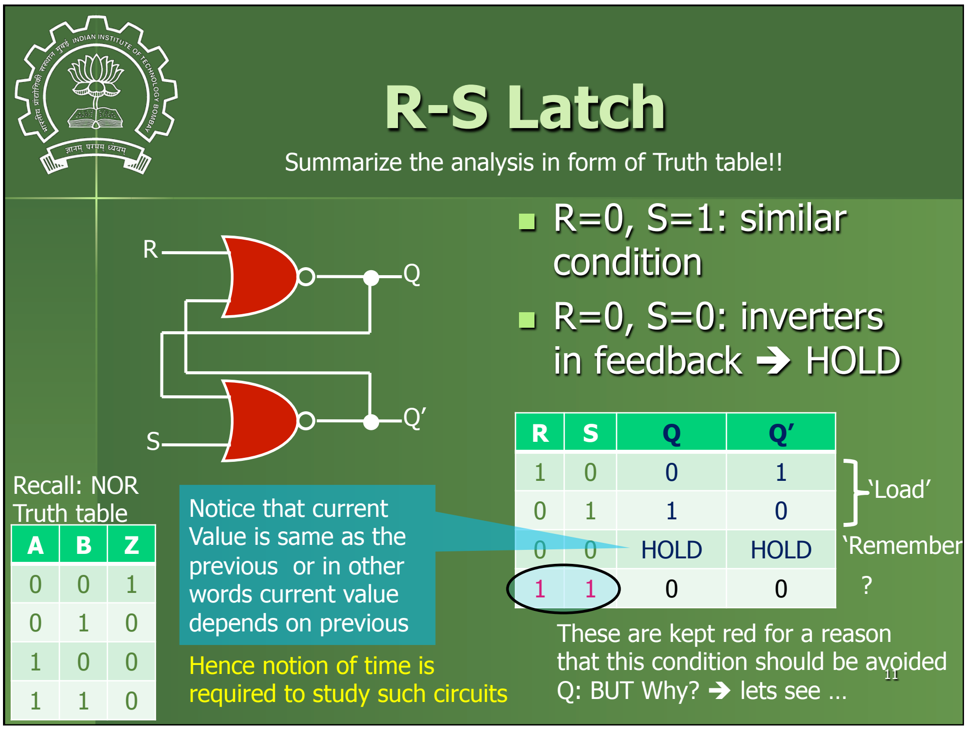
Another way of using feedback without need of switches to 'load' and/or 'remember'

- Similarly for R=0, S=1: Q′ is reset to zero independent of Q input to second NOR gate
- With Q′=0 R =0 as input for the first NOR, its output Q = 1
- So a nice way to 'load' the desired value (either 0 or 1) and 'remember' also by a digital way
- NO need for switches!! ☺

Recall: NOR Truth table

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

B = 1 → Z = 0



R-S Latch

Summarize the analysis in form of Truth table!!

- R=0, S=1: similar condition
- R=0, S=0: inverters in feedback ➔ HOLD

R	S	Q	Q′	
1	0	0	1	‘Load’
0	1	1	0	
0	0	HOLD	HOLD	‘Remember
1	1	0	0	?

Recall: NOR Truth table

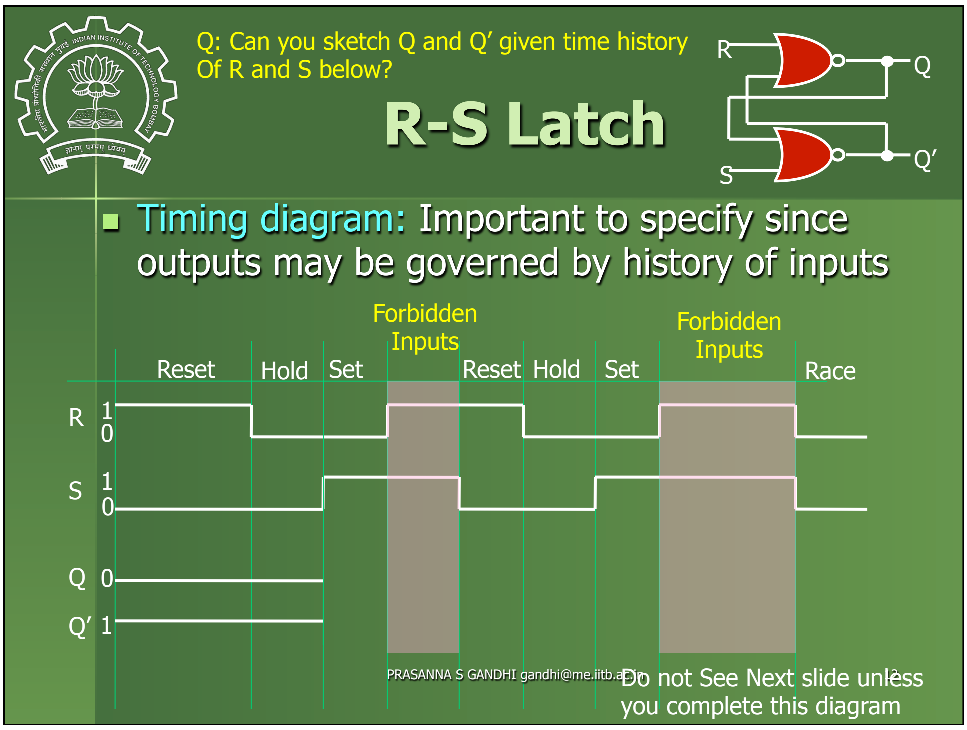
A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

Notice that current Value is same as the previous or in other words current value depends on previous

Hence notion of time is required to study such circuits

These are kept red for a reason that this condition should be avoided Q: BUT Why? ➔ lets see …

11



Q: Can you sketch Q and Q′ given time history Of R and S below?

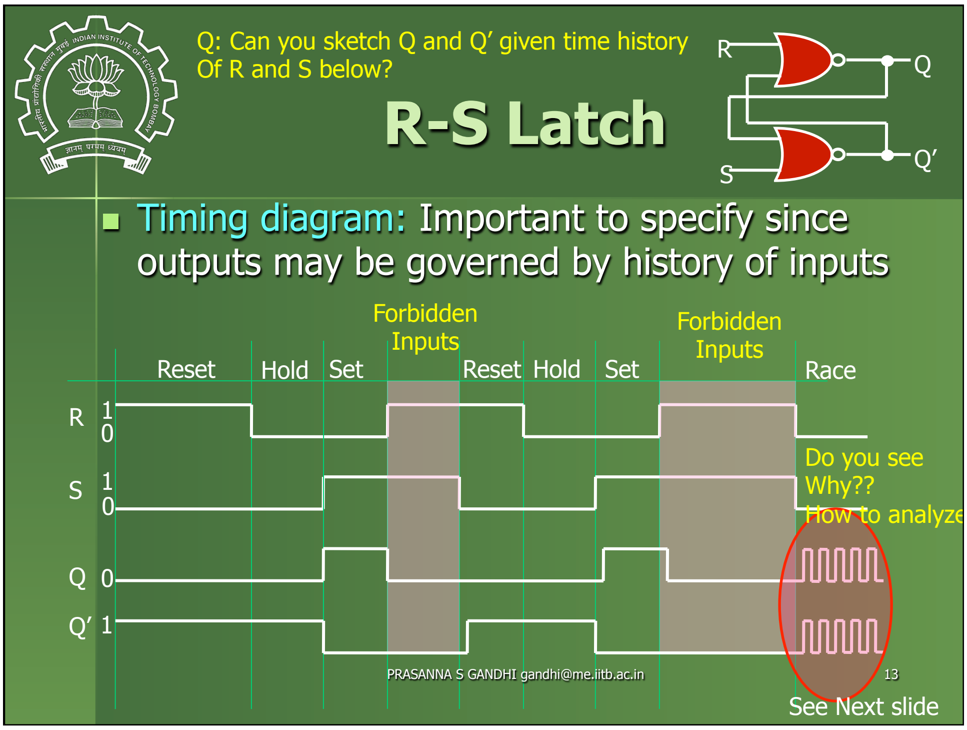
R-S Latch

- Timing diagram: Important to specify since outputs may be governed by history of inputs

PRASANNA S GANDHI gandhi@me.iitb.ac.in

12

Do not See Next slide unless you complete this diagram



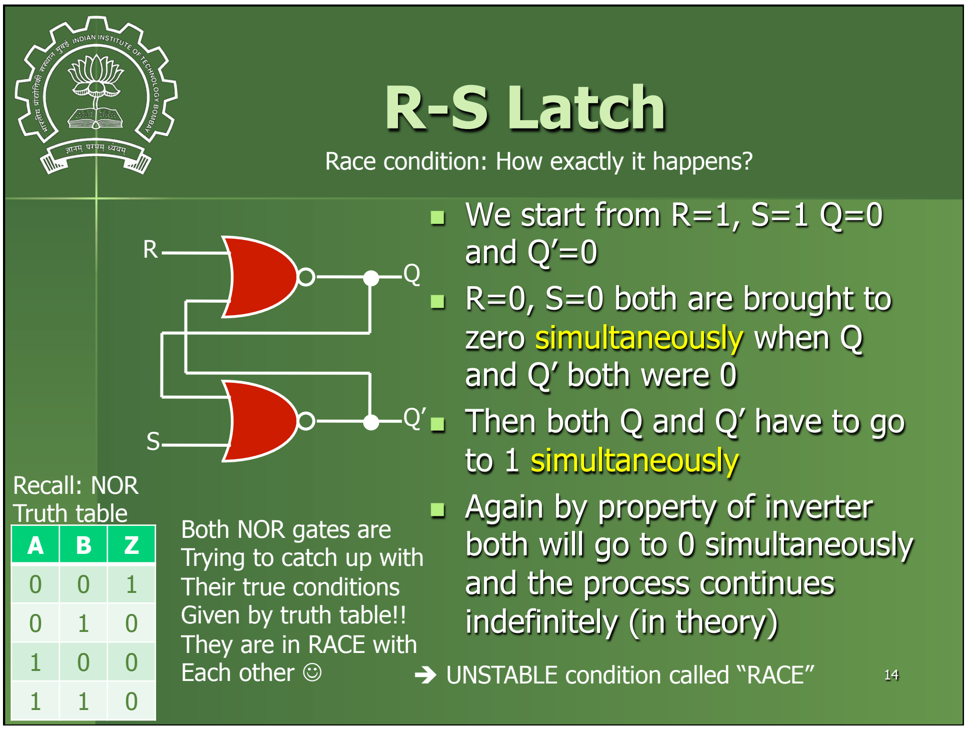
Q: Can you sketch Q and Q' given time history Of R and S below?

R-S Latch

- Timing diagram: Important to specify since outputs may be governed by history of inputs

Do you see Why??
How to analyze

See Next slide



R-S Latch

Race condition: How exactly it happens?

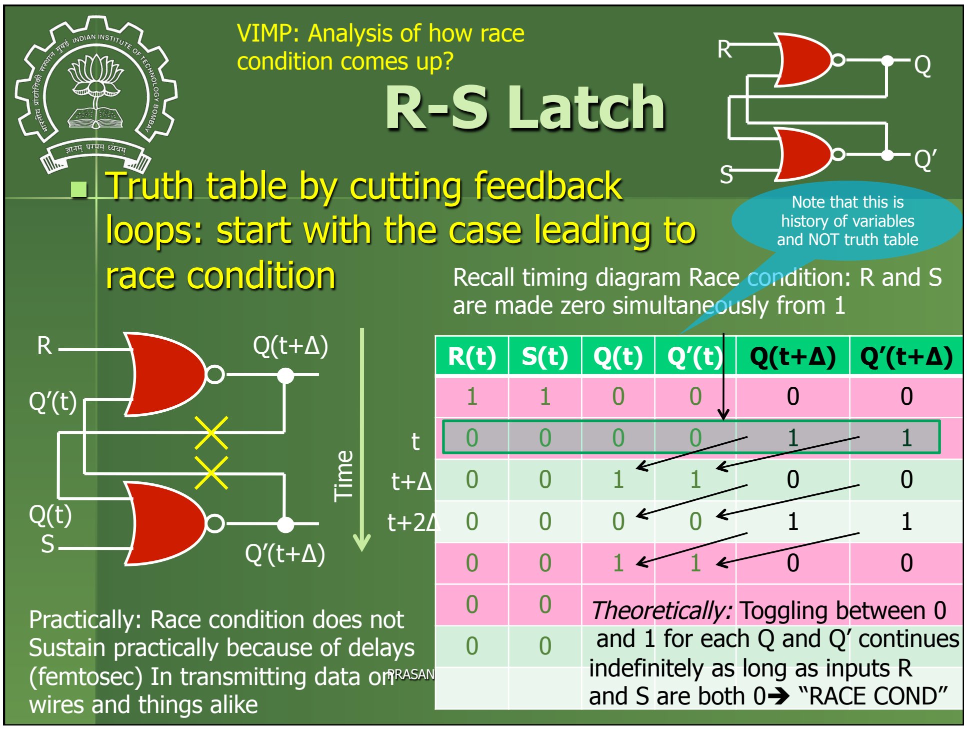
- We start from R=1, S=1 Q=0 and Q'=0
- R=0, S=0 both are brought to zero simultaneously when Q and Q' both were 0
- Then both Q and Q' have to go to 1 simultaneously
- Again by property of inverter both will go to 0 simultaneously and the process continues indefinitely (in theory)

➔ UNSTABLE condition called "RACE"

Recall: NOR Truth table

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

Both NOR gates are Trying to catch up with Their true conditions Given by truth table!! They are in RACE with Each other ☺



VIMP: Analysis of how race condition comes up?

R-S Latch

- Truth table by cutting feedback loops: start with the case leading to race condition

Note that this is history of variables and NOT truth table

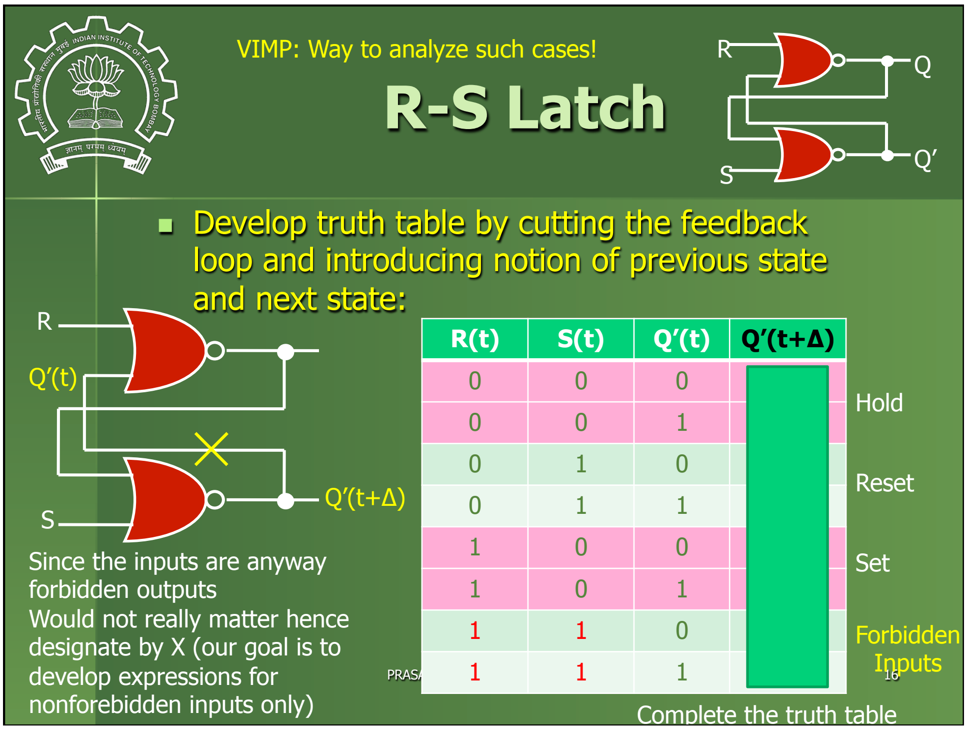
Recall timing diagram Race condition: R and S are made zero simultaneously from 1

	R(t)	S(t)	Q(t)	Q′(t)	Q(t+Δ)	Q′(t+Δ)
	1	1	0	0	0	0
t	0	0	0	0	1	1
t+Δ	0	0	1	1	0	0
t+2Δ	0	0	0	0	1	1
	0	0	1	1	0	0
	0	0				
	0	0				

Theoretically: Toggling between 0 and 1 for each Q and Q′ continues indefinitely as long as inputs R and S are both 0➔ "RACE COND"

Practically: Race condition does not Sustain practically because of delays (femtosec) In transmitting data on wires and things alike

PRASAN



VIMP: Way to analyze such cases!

R-S Latch

- Develop truth table by cutting the feedback loop and introducing notion of previous state and next state:

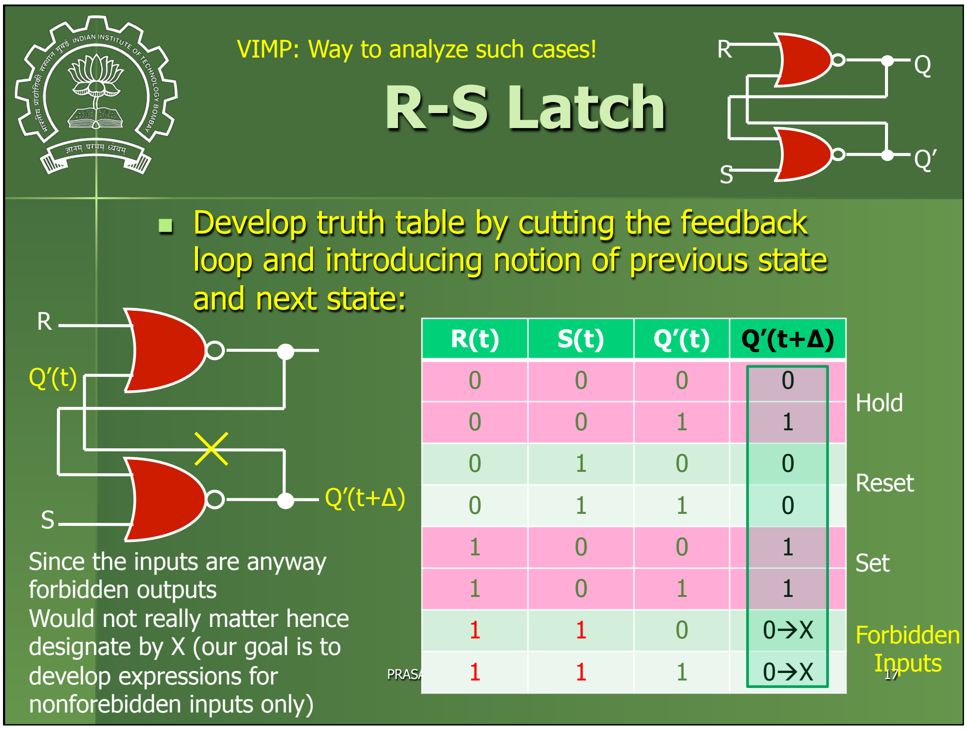
R(t)	S(t)	Q′(t)	Q′(t+Δ)	
0	0	0		Hold
0	0	1		
0	1	0		Reset
0	1	1		
1	0	0		Set
1	0	1		
1	1	0		Forbidden Inputs
1	1	1		

Since the inputs are anyway forbidden outputs Would not really matter hence designate by X (our goal is to develop expressions for nonforebidden inputs only)

PRAS

16

Complete the truth table



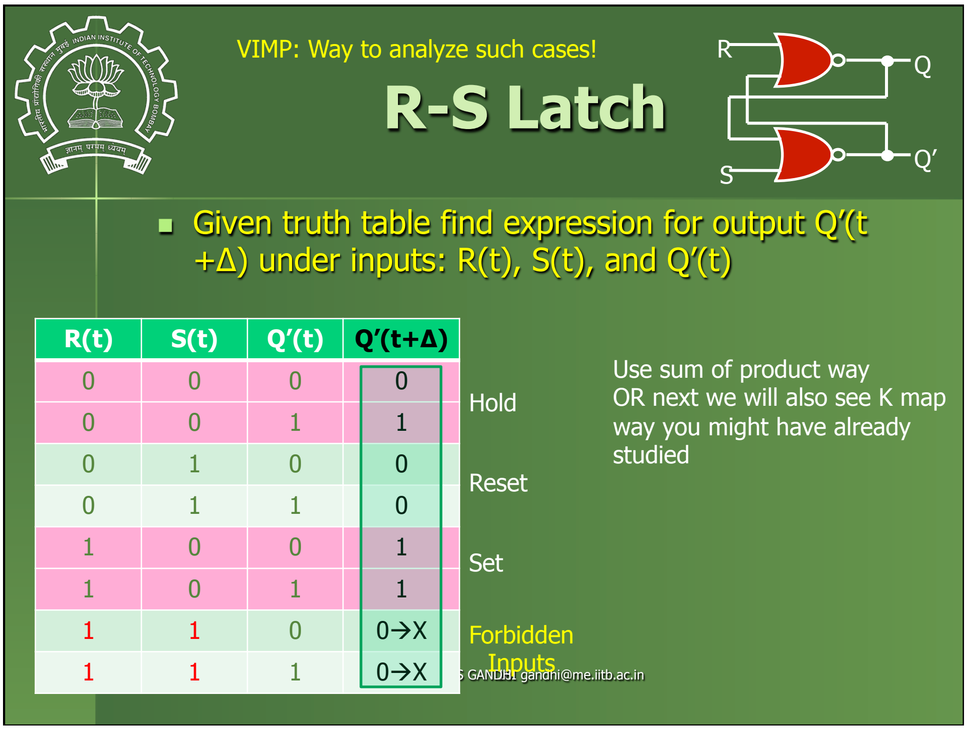
VIMP: Way to analyze such cases!

R-S Latch

- Develop truth table by cutting the feedback loop and introducing notion of previous state and next state:

R(t)	S(t)	Q′(t)	Q′(t+Δ)	
0	0	0	0	Hold
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	0→X	Forbidden Inputs
1	1	1	0→X	

Since the inputs are anyway forbidden outputs
Would not really matter hence designate by X (our goal is to develop expressions for nonforebidden inputs only)



VIMP: Way to analyze such cases!

R-S Latch

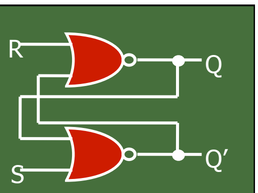
- Given truth table find expression for output Q′(t +Δ) under inputs: R(t), S(t), and Q′(t)

R(t)	S(t)	Q′(t)	Q′(t+Δ)	
0	0	0	0	Hold
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	0→X	Forbidden Inputs
1	1	1	0→X	

Use sum of product way
OR next we will also see K map way you might have already studied

VIMP: Way to analyze such cases!

R-S Latch



- Given truth table find expression for output Q′(t +Δ) under inputs: R(t), S(t), and Q′(t) using K map

R(t)	S(t)	Q′(t)	Q′(t+Δ)	
0	0	0	0	Hold
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	0→X	Forbidden Inputs
1	1	1	0→X	

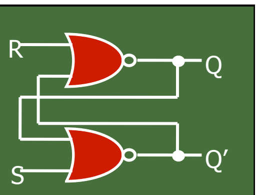
Step 1: Use standard template of 3 input K map and start filling Outputs (additional templates are Given in revision slides posted)

K Map

			R	
Q′ \ RS	00	01	11	10
0				
Q′ 1				
		S		

VIMP: Way to analyze such cases!

R-S Latch



- Given truth table find expression for output Q′(t +Δ) under inputs: R(t), S(t), and Q′(t) using K map

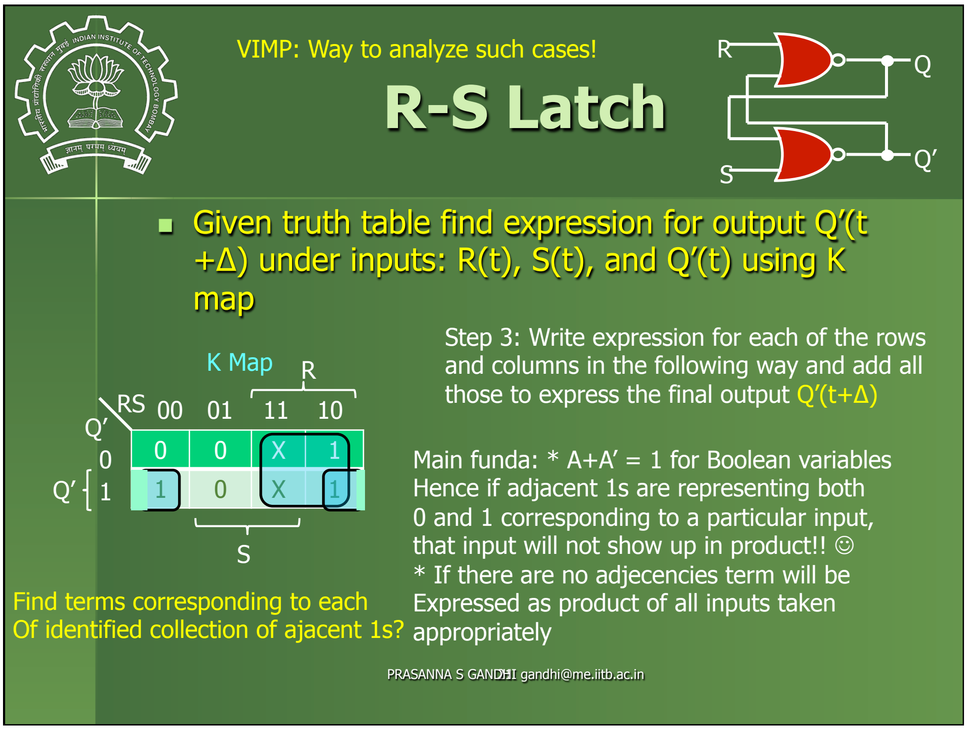
R(t)	S(t)	Q′(t)	Q′(t +Δ)	
0	0	0	0	
0	0	1	1	
0	1	0	0	
0	1	1	0	
1	0	0	1	
1	0	1	1	
1	1	0	0→X	Forbidden Inputs
1	1	1	0→X	

Step 2: Identify Rows, columns or Matrices of all adjacent 1s (See below) Even ajacencies at the end of row and Column are to be considered

K Map

			R	
Q′ \ RS	00	01	11	10
0	0	0	X	1
Q′ 1	1	0	X	1
		S		

In this case outputs X are considered 1 since anyway it does not matter Because corrsponding inputs are forebidden



VIMP: Way to analyze such cases!

R-S Latch

- Given truth table find expression for output Q′(t +Δ) under inputs: R(t), S(t), and Q′(t) using K map

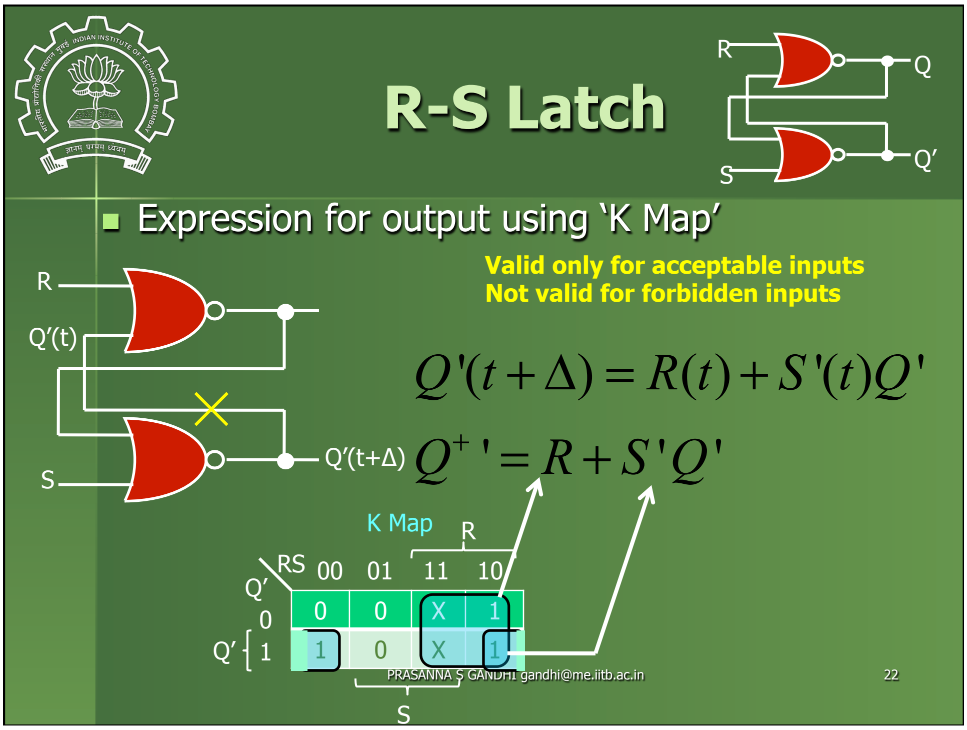
K Map

Q′ \ RS	00	01	11	10
0	0	0	X	1
1	1	0	X	1

Step 3: Write expression for each of the rows and columns in the following way and add all those to express the final output Q′(t+Δ)

Main funda: * A+A′ = 1 for Boolean variables Hence if adjacent 1s are representing both 0 and 1 corresponding to a particular input, that input will not show up in product!! ☺
* If there are no adjecencies term will be Expressed as product of all inputs taken appropriately

Find terms corresponding to each Of identified collection of ajacent 1s?



R-S Latch

- Expression for output using 'K Map'

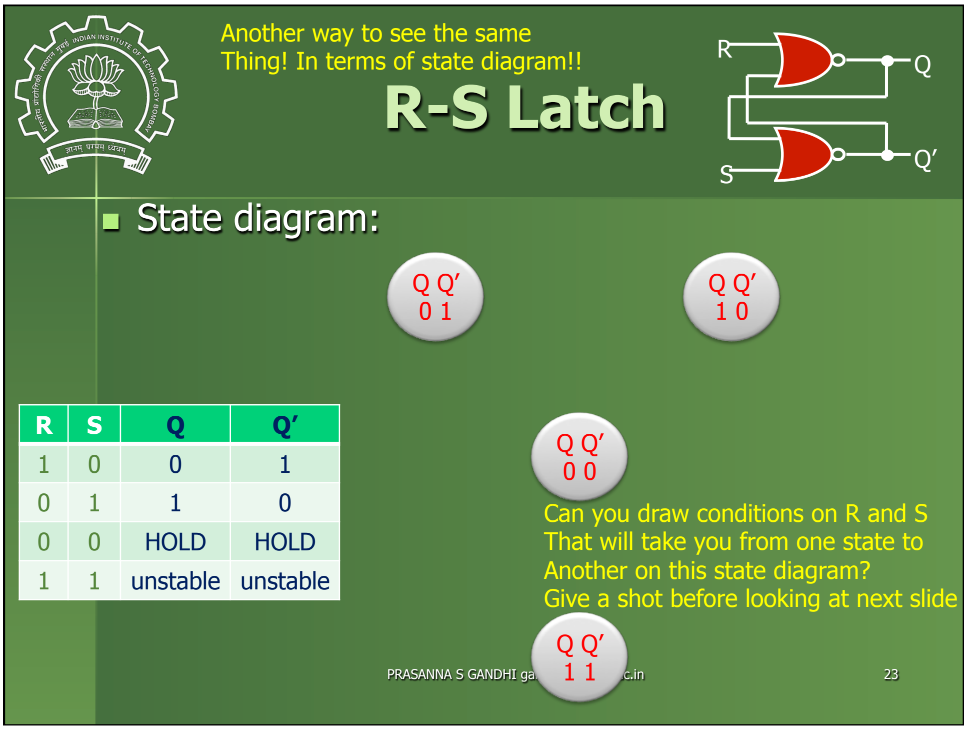
Valid only for acceptable inputs
Not valid for forbidden inputs

$$Q'(t+\Delta) = R(t) + S'(t)Q'$$

$$Q^{+}{}' = R + S'Q'$$

K Map

Q′ \ RS	00	01	11	10
0	0	0	X	1
1	1	0	X	1



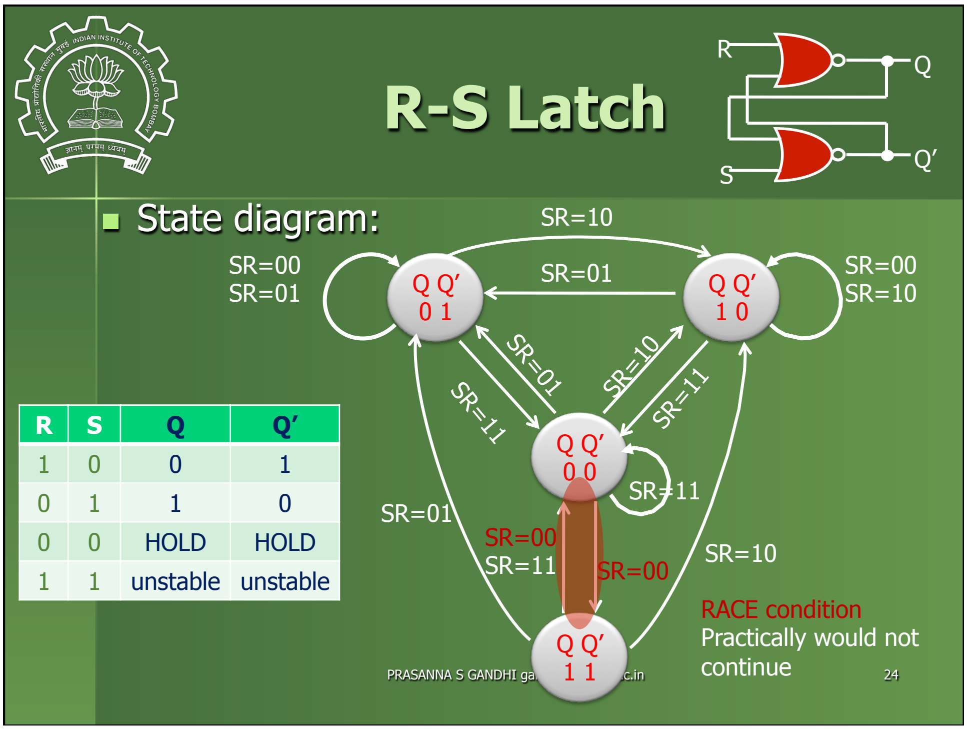
Another way to see the same
Thing! In terms of state diagram!!

R-S Latch

- State diagram:

R	S	Q	Q′
1	0	0	1
0	1	1	0
0	0	HOLD	HOLD
1	1	unstable	unstable

Can you draw conditions on R and S
That will take you from one state to
Another on this state diagram?
Give a shot before looking at next slide

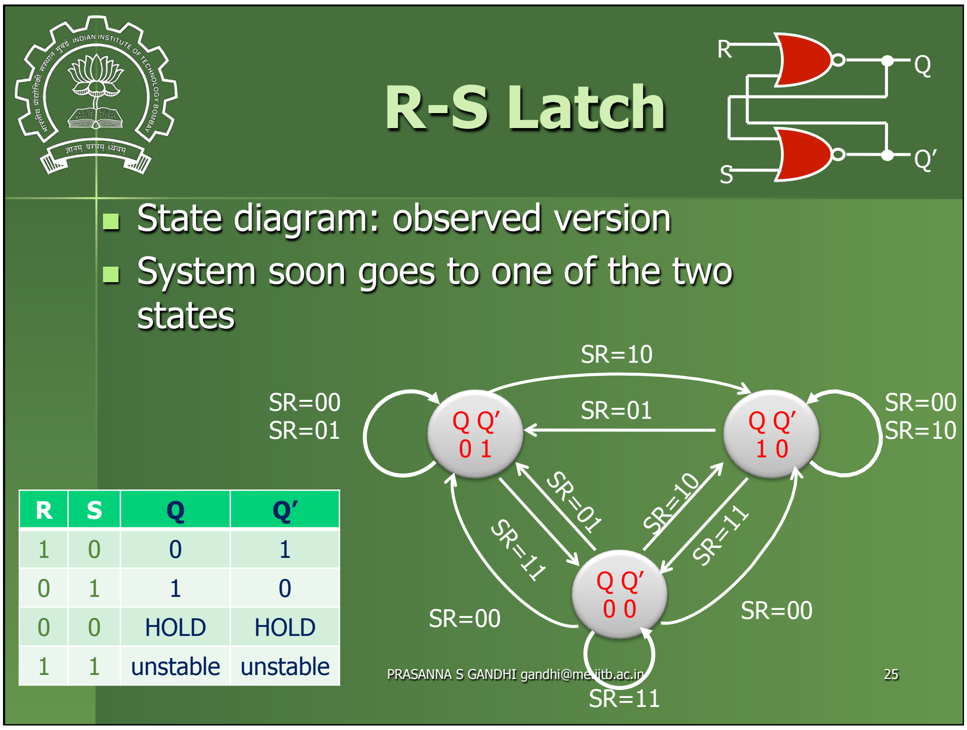


R-S Latch

- State diagram:

R	S	Q	Q′
1	0	0	1
0	1	1	0
0	0	HOLD	HOLD
1	1	unstable	unstable

RACE condition
Practically would not
continue

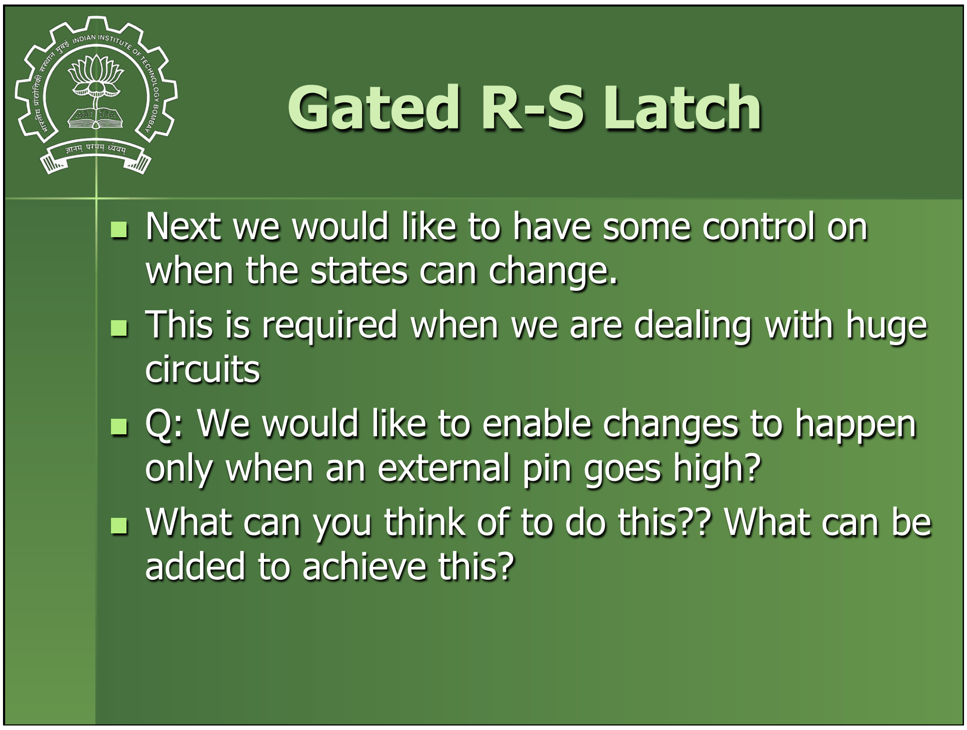


R-S Latch

- State diagram: observed version
- System soon goes to one of the two states

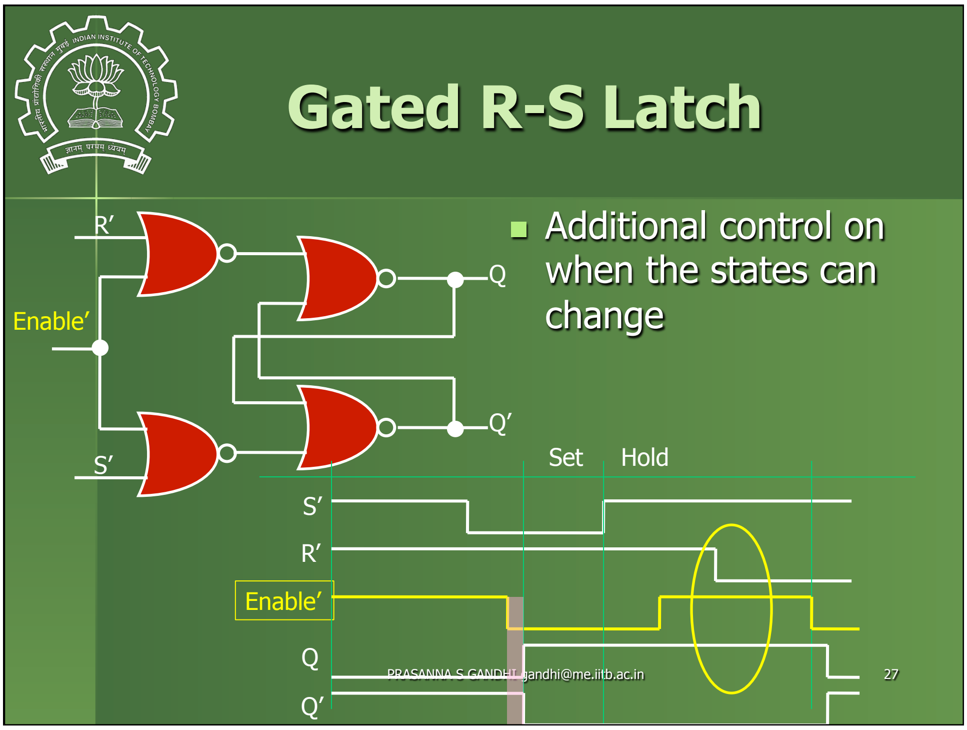
R	S	Q	Q′
1	0	0	1
0	1	1	0
0	0	HOLD	HOLD
1	1	unstable	unstable

PRASANNA S GANDHI gandhi@me.iitb.ac.in 25



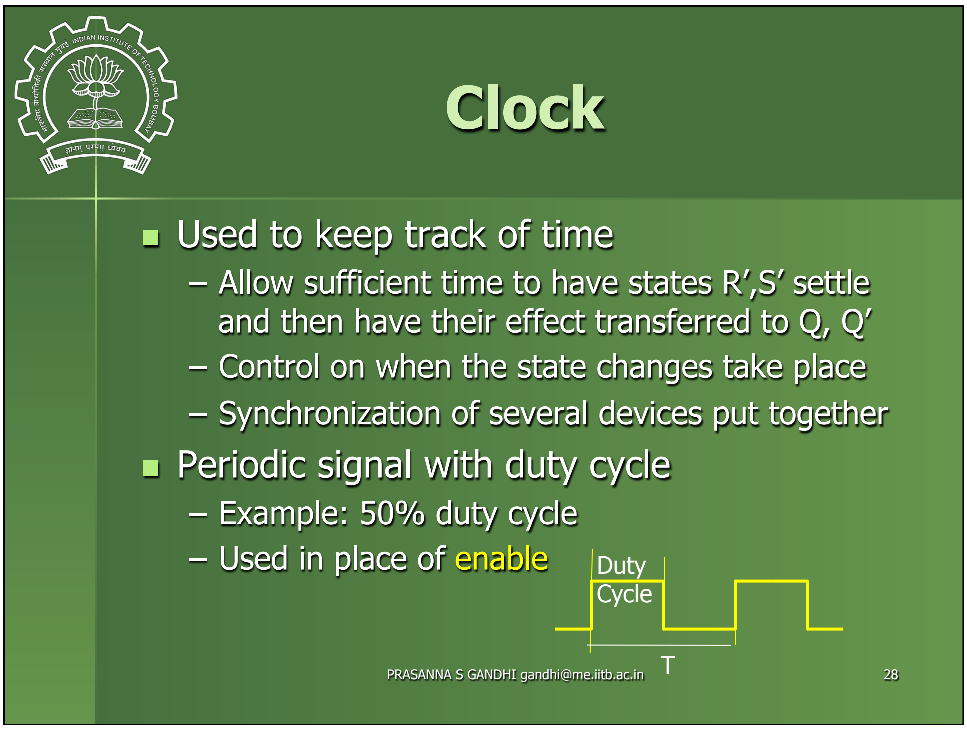
Gated R-S Latch

- Next we would like to have some control on when the states can change.
- This is required when we are dealing with huge circuits
- Q: We would like to enable changes to happen only when an external pin goes high?
- What can you think of to do this?? What can be added to achieve this?



Gated R-S Latch

- Additional control on when the states can change



Clock

- Used to keep track of time
  - Allow sufficient time to have states R’,S’ settle and then have their effect transferred to Q, Q’
  - Control on when the state changes take place
  - Synchronization of several devices put together
- Periodic signal with duty cycle
  - Example: 50% duty cycle
  - Used in place of enable

THANK YOU

PRASANNA S GANDHI gandhi@me.iitb.ac.in 29