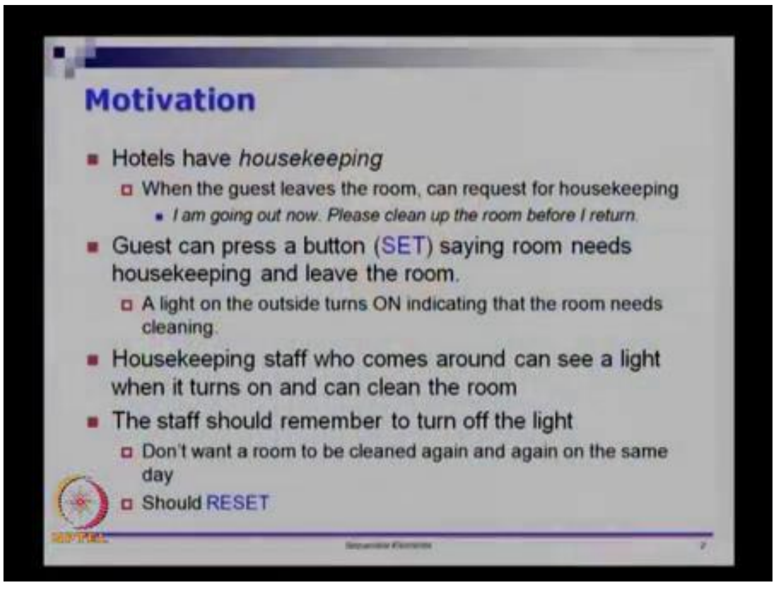
Digital Circuits and Systems
Prof. Shankar Balachandran
Department of Electrical Engineering
Indian Institute of Technology, Bombay
And
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Module – 18
Sequential Elements

Good morning everyone, I hope you have enjoyed the week 3 lectures. Particularly, we started programming in Verilog and we will see Verilog programming every week from now on and this is probably new to many of you. So, what I suggest is you start writing small Verilog modules time and again, so that you get use to the fact that you are not just designed circuits, so you are also writing program which models the circuit that you want which is unknowable to evaluation.

So, if you did the circuit in a piece of paper, I would have no way to check it, but if you write Verilog code which describes the circuit, I have ways to check it even from here. So, we are now in week 4 and in this week, we are going to look at basic sequential elements. So, let me give you a small motivation for this problem.

(Refer Slide Time: 01:10)



Motivation

- Hotels have *housekeeping*
  - When the guest leaves the room, can request for housekeeping
    - *I am going out now. Please clean up the room before I return.*
- Guest can press a button (SET) saying room needs housekeeping and leave the room.
  - A light on the outside turns ON indicating that the room needs cleaning.
- Housekeeping staff who comes around can see a light when it turns on and can clean the room
- The staff should remember to turn off the light
  - Don't want a room to be cleaned again and again on the same day
  - Should RESET

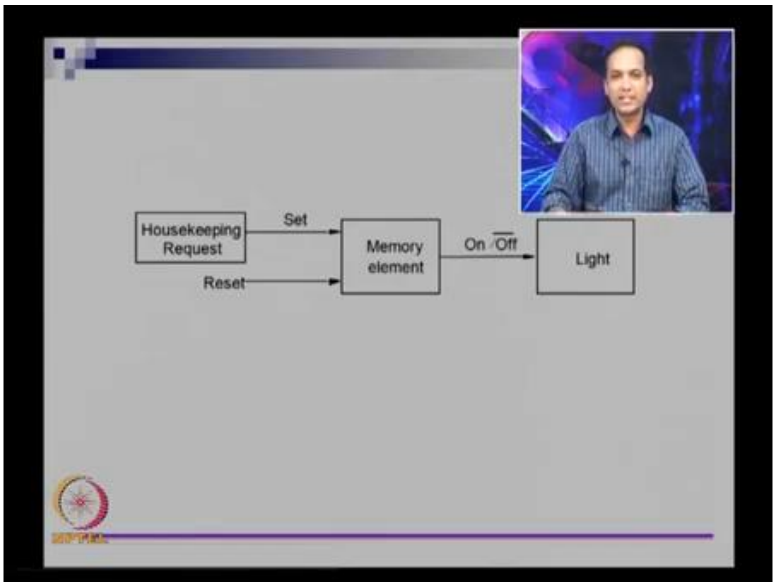
So, in hotels, they have what is called housekeeping. So, let us say a guest who is living in a room wants to leave the room, but wants to get it clean when the guest is away. So, what the guest wants to say, I am going out now, please clean up the room before I come back. And one way in which this can be done is, the guest can press a button, let us called it is SET, saying that the room needs cleaning, and let us say that this turns on a light outside and indicating that the room needs cleaning.

So, the house keeping staff, when they move around, they can see that the light is turned on. So, if the light is turned on, it is an indication to them that the room actually needs cleaning and the guest as left a message in some sense for the room to be clean. So, some housekeeping staff can go clean up the room and then press another button which resets the light.

So, one thing that you have to do here is, the staff member should go and turn off the light later. Otherwise, some other housekeeping staff, when they come around, they may still see the light turned on and they may go and try and clean the room once more and so on. So, what we want is, we want a mechanism by which there is some kind of a signal left by the guest, when the guest leaves and when the guest comes back, if he sees the light is off, he will know that the room was actually cleaned.

If the light was still on, he probably know that the room was not cleaned so far, so this is a set up that we want to take care of, so this is the common thing in hotels. So, I am going, go to a city for some conference and I see that so I leave in the morning, but I want the room to be cleaned, so that when I come back in the evening from the conference, I have a cleaned room. So, this kind of set up is very common in hotel housekeeping and this is a small motivational problem.

(Refer Slide Time: 03:14)



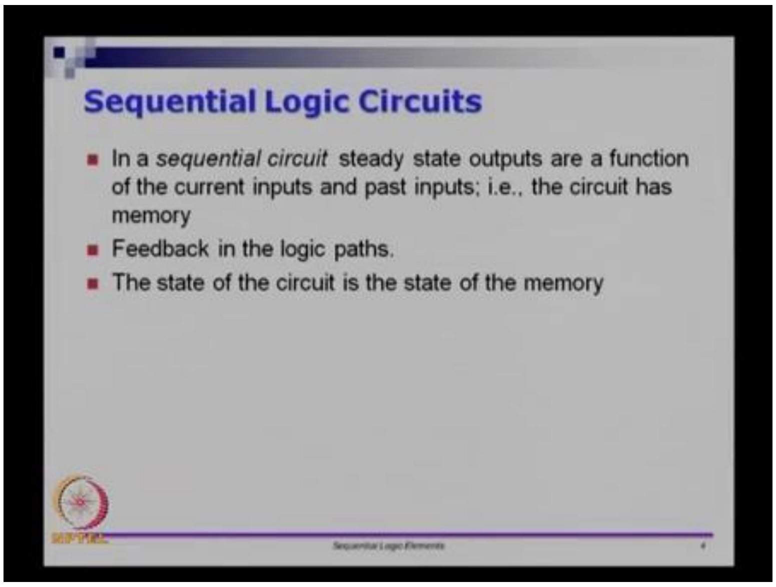
So, as a circuit, what we want is, in some sense, I as a guest must be able to set a housekeeping request and it should remain 1. Because, I press 1, let us say I set a request and then I leave, you do not expect me to sit there and keep continuously making request. So, once it is set, it should remain at that till the housekeeping staff comes cleans and then resets it.

So, this requires some kind of a memory element, it is not an instantaneous thing. So, remember, when we did combinational circuits, I said the moment something changes the input, if the input changes, the output changes and as soon as the input is removed, the output may or may not be change. In this case, what we want is, even if the input is removed, so I press a button saying, I need a request for this, it sense a signal called set to a circuit block and I may leave it. That signal is not going to be on any more, I may not have the signal any time.

But, this should be kept in the memory, which means the light should keep turned on, till the housekeepers come and reset it. Again, the same thing, once the housekeepers reset it, it should remain, because they are not going to keep pressing till I return back to the hotel room. So, they press a reset button and they leave and the memory system should keep it, the system should keep this in their memory and when I return back of see that the light is off. So, I would know that the reset button was pressed which in fact means the housekeeping staff came and clean the room.

So, this requires a memory element, we need to remember what was the state, even if the input was removed. So, this is the curial difference between what is called a combinational circuit block versus a sequential circuit block.

(Refer Slide Time: 05:14)



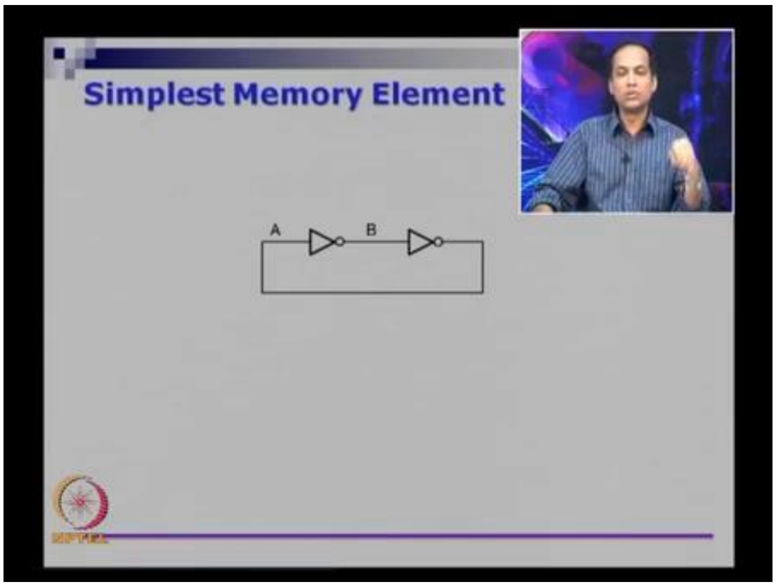
Sequential Logic Circuits

- In a *sequential circuit* steady state outputs are a function of the current inputs and past inputs; i.e., the circuit has memory
- Feedback in the logic paths.
- The state of the circuit is the state of the memory

So, in a sequential circuit steady state outputs are a function of the current inputs and the past inputs also, so the circuit does have some memory. So, when I say current and pass, it also means current and future. So, if I store something now, it is available for the future, it will be remembered for the future also, until some other condition happens. So, what this means is in terms of the circuit, there is actually some feedback path.

In the circuits of you seen so far, there is you start with the inputs, then these feeds some gates and these feeds some other gates in turn and so on till you have the output. At no point, we had a wire coming back from output of a gate to somewhere in the input of the circuit, that never happen so far. But, for memory, you need this feedback path and the state of the circuit is actually registered as the state of the memory.

(Refer Slide Time: 06:08)



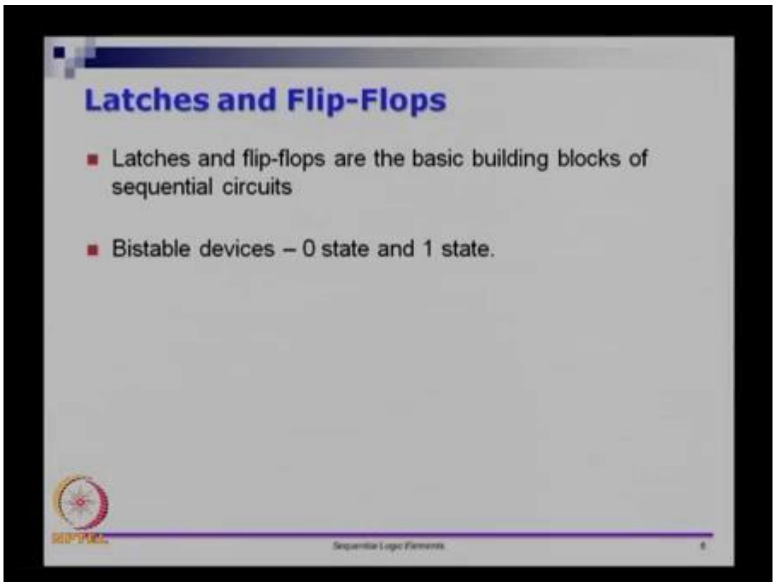
Simplest Memory Element

Let us see a very simple memory element. So, let us take two inverters and connect them back to back, so in this case, there is no extended input. For a moment, let us assume that this B is actually 1, if B is 1, you would expect that this inverter will drive is 0 and this being is 0 will again keep B at 1. So, B will remain in the steady state of being 1. However, if B is 0, then 0 compliment will be 1, that goes as A and then one compliment would come back as B. So, B would still remain at a steady state, where at this point the steady state would be 0.

So, what the circuit does is, if B started at 1 it remain at 1, if it start at 0, it will remain at 0 and it is stay there forever, so provider there is no disturbance and so on. So, the key thing is this circuit does not have any input at all. Since, that it is very uninteresting circuit; you cannot really do anything with this. Even though, it remembers the value, no matter what the values, the values remembered; however, it does not have any external input.

So, we need a memory element which can remember the value, in some sense, so go back to the problem I gave, we need to remember the set as well as the reset given to the circuit. So, we need something like that and it has to remember, so these are the three things.

(Refer Slide Time: 07:47)



Latches and Flip-Flops

- Latches and flip-flops are the basic building blocks of sequential circuits
- Bistable devices – 0 state and 1 state.

Sequential Logic Elements

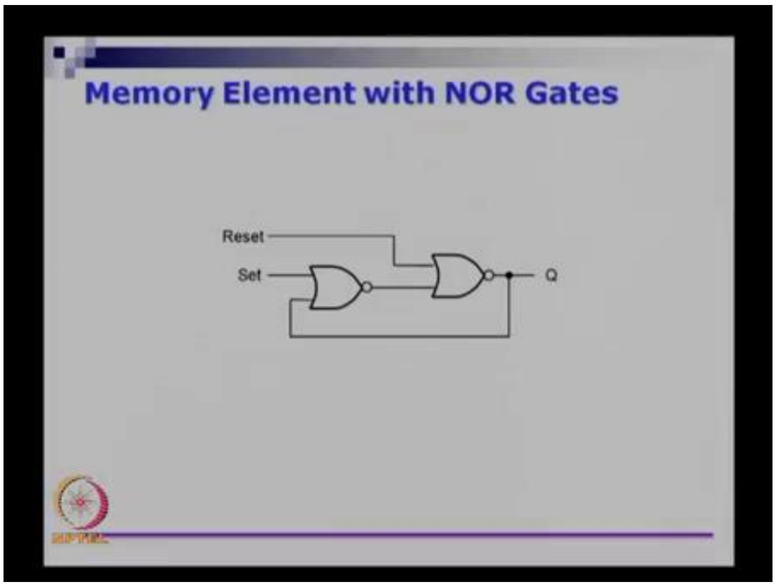
So, to do this, we have different kinds of basic logic elements. Latches and flip flops are the basic building blocks of sequential circuits. The first thing is these devices both the latches and flip flops are what are called Bistable devices. So, a Bistable device is one in which if you are at logical level 0 it will remain at 0 and it is a stable state. And if you are at a logical level 1, it will stay at 1 and that is also a stable state, unless there is a condition which moves it to the other state.

So, this is unlike a circuit like Schmitt trigger for instant, which is a mono stable multi vibrator. So, it is in one state, so let us say it is in 0 and you trigger a Schmitt trigger, it may go to 1 momentarily and then come back to 0. So, it is a mono stable multi vibrator, because there is only one stable state, but it can be in two different states. So, the one state is transient instrument trigger whereas the zero state is a stable state.

Whereas, in a flip flop, both the states are stable or in a large, a flip flop are a large. In both these different logic elements, both the zero state and one state are stable. So, many times, when we talk to people let them actually go back and forth between latches and flip flops and they may not make a specific distinction between those. So, however in this course we will be very careful about what a latches, what a flip flops is.

In fact, there are curial differences and an experience person may actually laps in say one for the other. But, usually from the contexts that they are talking about, it is usually clear whether they are talking about the latch or a flip flop. But, for this course, we will be very careful about, what a latches versus what a flip flop is and in the real world also, it is actually important.

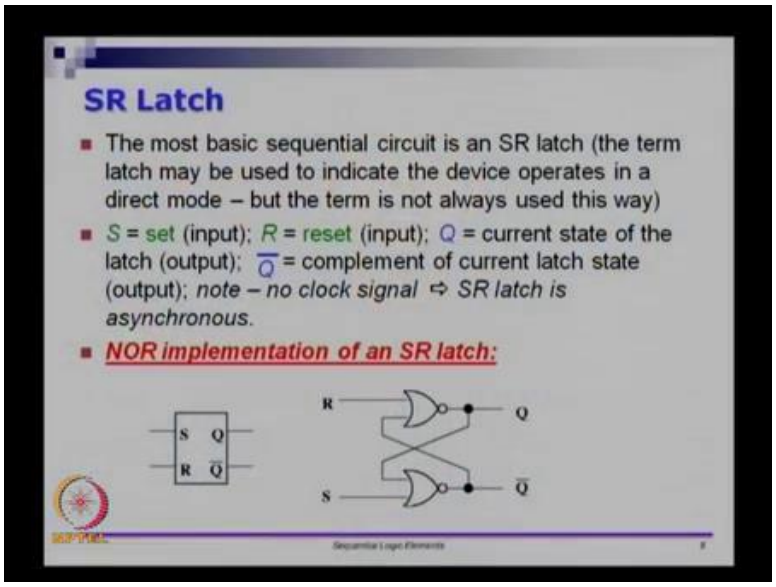
(Refer Slide Time: 09:40)


Memory Element with NOR Gates

So, let us do this memory element slightly differently, earlier I had two inverters, which were connected back to back. Instead, I am going to put 2 NOR gates, which are connected back to back. So, that is a NOR gate, whose output feeds into another NOR gate input, whose output in turn feeds into the first once input. So, there is feedback path, this would not happen in a combinational circuit.

In a sequential circuit, we will have feedback paths and we have two different signals namely reset and set. So, this is a memory element with NOR gates, we will see, why the acts as memory and what kind of circuit it is in and so on in more detail now. But, because there is feedback, there is some notion of memory, so will get that into a little while.

(Refer Slide Time: 10:34)



SR Latch

- The most basic sequential circuit is an SR latch (the term latch may be used to indicate the device operates in a direct mode – but the term is not always used this way)
- *S* = set (input); *R* = reset (input); *Q* = current state of the latch (output); $\overline{Q}$ = complement of current latch state (output); *note – no clock signal ⇨ SR latch is asynchronous.*
- ***NOR implementation of an SR latch:***

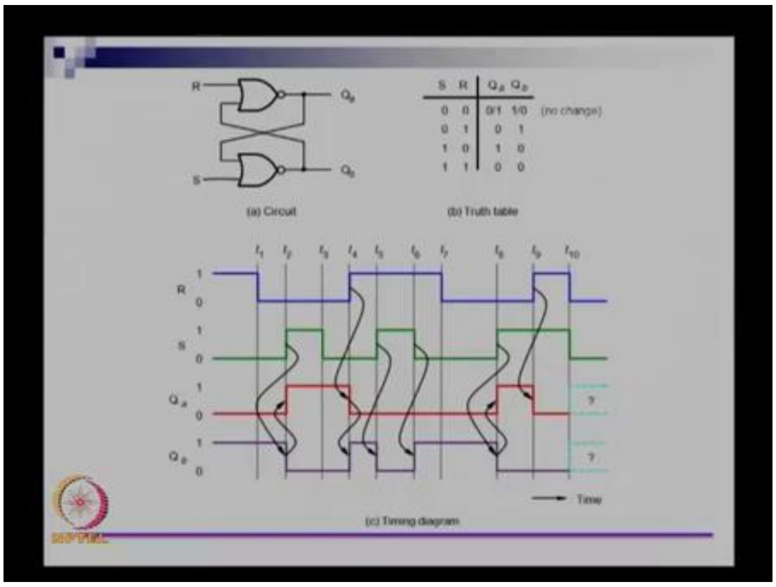
S
Q
R
$\overline{Q}$
R
Q
S
$\overline{Q}$

So, the first kind of logic element, I am going to talk to you about is something called SR latch, it is the most basic sequential circuit. So, it is very fundamental and we will use this SR latch to build other latches and so on. So, we will use this as a basic block to explain other things. So, what we see as a circuit here in the bottom is actually the same circuit that I showed earlier, except that, it is drawn in a different form.

So, you take a look at this circuit, so there is some signal called a set, there is some signal called a reset and there is an output called Q. So, there are 2 inputs and 1 input in this circuit. The same circuit is redrawn here, so there is a signal called S, there is a signal called R and there is an output called Q. On top of that, there is another output called Q bar, which is actually this line.

So, if I topped out this one and gave it as output will call that Q bar. So, what it actually has is two signals namely R and S and let us say R is going to stand for reset and S is going to stands for set will get to that in the little while. And will have two outputs Q and Q bar, where Q is the compliment of Q bar and the other way now. So, this is the basic definition of SR latch, but I first want to you be convinced that, this is going to behave, this is going to remember the values will have to convinced about that. And later, we have to convince about, whether or actually reset set, S actually set and so on, we will see that in detail from a next slide onwards.

(Refer Slide Time: 12:09)


R
S
Q_a
Q_b
(a) Circuit
(b) Truth table
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10}
?
Time
(c) Timing diagram

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	

So, the same circuit is given above. So, the sub figure a is the actual circuit and the way it behaves is actually given as a table here. So, let us assume that, this there are two outputs Q a and Q b for a moment instead of Q and Q bar, I am going to assume that the two different signals Q a and Q b. And as of now, we do not know the relationship between Q a and Q b, let us starts with them and let us assume that, there are 2 inputs R and S.

So, what we see here on the right side is at table which describes, what is going to happen, because of the circuit. So, let us starts with the case, where both S and R are 0. So, if both S and R are 0, so what would happen, because R is 0, if R is 0, Q b are 0 would be Q b compliment. So, let us assume in fact that, so R is 0, S is 0 and let us assume that Q a is 0 and Q b is 1 for a while.

Let us assume that, if Q a is 0, then S being 0 will take Q a OR S, which is 0, compliment of that, Q b will become 1. And because R is 0, you have 1 OR 0; that is a 1 and the compliment of that will go to 0. So, Q a will remain at 0 and Q b will remain at 1. Let us assume the same thing, where Q a was 1 instead, this Q a was 1, then 1 OR 1 is a 1, but there is a compliment here that will try Q to be 0 and 0 OR 0 is a 0, NOR of that is actually a 1, so Q a will be at 1.

So, you can see that if Q a is 0, Q b will be 1 and if Q a is 1, Q b will be 0 and when you apply 0 0 as inputs, they actually remain the same. So, if u start with the 0 here, this will be a 1, but under the conditions given for R and S, Q a will remain at 0 itself, it will not change, so this is 1 case. Let us go and see, what happens when you have the other cases.

So, what you see here in the bottom is what is called a timing diagram, we saw something like this in the verilog code last time. So, we had signals which were in the y axis and in the x axis, we had time. So, let us starts with each of these timelines, there is timeline t 1, t 2 so on up to t 10. And I am going to show you, what is happening in the circuit and will also convince also is that this table is correct.

So, let us for a while assume that R is 1 and S is 0 and let us also assume for a while that R means reset, so reset will drive Q a to 0, let us assume that for a while and we will also assume that S drives Q a to 1. So, R being 1 is suppose to drive Q a to 0 and S being 1 is suppose to drive Q a to 1. And I also claim that Q a and Q b are compliments of each other. So, let us assume that for a while.

So, let us see his wave form without making any of these assumption, let say that R is 1 and S is 0 for while now and Q a is 0 and Q b is 1, because if R is 1, then Q a is 0, it is quiet apparent. If R is 1, 1 NOR 1 is 0, Q a is 0, so 0 or 0 is 0, but and NOR will give you 1, so Q b will be 1. So, till t 1, I think it is straight forward, these signal are consistent with each other, you do not have to assume anything, just making R equal to 1 and S equal to 0 will imply, what Q a and Q b are.

Let us assume now at t 1, R actually goes down to 0, so this is a case, where both of them of 0, so both the inputs are R and S are 0. So, I said that, if both R and S are 0, Q a if it is 0, it will remain at 0 and Q b, if it is 0, it will remain at 0. So, in this case, Q a is at 0, it remains at 0 and Q b was at 1, it remain at 1, so this is up till time t 2. Now, let us assume that a t 2, this green one is S, let us assume that s goes to 1.

So, I said that semantic attach S is that is supposed to be setting the value of Q a to 1. So, let us see what happens, when S goes to 1, you can see that the effect of S going to 1 is not direct on Q a. In fact, it goes in changes Q b to 0; that is the direction of the arrow here. So, when you see a timing diagram and if you see an arrow coming from one signal into another signal, you can see the arrow here.

So, it is starts from here, but the direction of the arrow is from the green line to the purple line. So, the change in S is effecting a change in purple line, which is Q b. So, change of S from 0 to 1 will change Q b from 1 to 0. So, it is shown as they instantaneous here as in time t 2 itself magically this changing from 0 to 1, immediately changes or simultaneously, it is look like it changing Q b from 1 to 0. Usually, it is not instantaneous, it requires a little bit of a delay, let us not worry about that right now.

So, let us assume that is instantaneous for now, it is actually changing Q b to 0, S changing to 1 changes Q b to 0 and when Q b changes to 0, you see what happens, when Q b is changing to 0. So, Q b was earlier at 1, it is now changing to 0 and you see the condition for R, R at t 2 is at 0, so you have 0 NOR 0 which is 1, so Q a becomes 1. So, you can see the direction of arrow here. So, S becoming 1 changes Q b to 0 and Q b changing to 0 changes Q a to 1.

Now, at this point, let us say S, went back to 0 at t 3, so if you look at time between t 3 and t 4, both S and R are at 0. So, Q a will continue to remain at 1 and Q b will continue to remain at 0. So, we have reach time step t 4 so far. Now, let us look at the next change at this point, let us assume that R goes to 1 and S remains at 0. So, if R goes to 1, what is the effect of that, when R goes to 1, it will have an effect on Q a, because Q a is R NOR Q b. So, if R goes to 1, it does not matter what Q b is Q a will go to 0, because of NOR gate there.

So, you can see the cause and effect here, the cause is the tail of the line and the effect is the head of the arrow. So, the tail of the arrow is here and the head of the arrow is here. So, we can see that t 4 is changing the red line to go from 1 to 0. So, Q a is changing from 1 to 0, this is going to change the purple line or Q b from 0 to 1. So, it is clear from this 1, that when R is 1 and when S is 0, Q a is reset.

So, the word reset is use to indicate control changing the value to 0, it could be 1, if it is 1 earlier, it is actually changing to 0, if it is 0 earlier also right will remain at 0. So, reset means no matter, what the input condition was it will go to 0. So, now, we are at time t 4, let say the time passes and let say at this point t 5 at t 5 let say S comes on, at this point, both R and S are 1.

So, when both R and S are 1, it is looks like, so I said the semantic for R is reset and the semantic for the S is set, we want Q a to be reset, when R is 1 and Q a to be set, when S is 1. But, now both S and R are 1, let see what happens both S and R are 1, so when both S and R are 1 what happens is, so this S changing to 1 will change Q b to 0, Q b is 0, R is

1, so 1 OR 0 is 0, Q a also remains it is 0. So, between t 5 and t 6, Q a is 0 and Q b is also 0.

So, it is look like both being 1 is driving Q a and Q b to both 0's, we can see that here also, it is remarked here, both are 1, it is remaining at 0. Now, let us say at t 6, S again changes back to 0, if S changes to 0, then that changes Q b to 1, because the conditions of that Q a is already 0, S is changing from 1 to 0. So, Q b will change to 1 and Q b being 1 and you can see that R is actually 1, both are 1, Q a will remain at 0.

So, there is no change in Q a, there is no arrow like earlier in these two places, we saw that one of them change in change the other one also. Here, only one of them change, again here only one of them changes, not both Q a and Q b. Finally, now, let us move to step t 7, so at t 7, R is pull down to 0, so both R and S are 0. So, from t 7 to t 8, you will see that Q a and Q b will keep their old values Q a at 0 and Q b at 1.

Let us now assume that at time t equal to 9, S became 1, if S comes on when R is 0, then Q a will go on, so when S comes on Q b goes 0, because of which Q a goes on, so this is the sequence. Because, of the green line going 1, the purple line will go to 0 and red line will go to 1. So, for everything is alright, let us look at time step 9. So, at time step 9, R also comes on, if R comes on that will bring Q a to 0, you can see that here, if R comes on then Q a should go to 0.

Now, this is a peculiar time step, what does happen now is R and S are both at 1 and Q a and Q b are both at 0. Now, for some reason, let us assume that a t 10, both R and S are pull down to 0. So, the both are changing from 1 1 to 0 0, so if R is changing from 1 to 0, S is changing from 1 to 0 and the other 2 inputs, the Q a and Q b are already at 0. Now, let us see, what suppose to happen to Q a first. So, R is changing from 1 to 0 and Q b was already at 0. So, if you look at what will happen to Q a, this is 0, this is changing from 1 to 0, Q b at 0, so Q a able try to change to 1.

Let say at the same time, I was evaluating S, before this Q a changes to 1, let us assume that Q a is already 0, so if it is a 0, let us now evaluate S. This was at 0 and S also went to 0, 0 or 0 goes 0, the NOR will make it 1. So, if both R and S simultaneously go to 0, Q a and Q b both will go simultaneously to 1. So, but that itself is not the problem, if both of them goes simultaneously at 1, so let say both these gates have the same amount of delay, that is the amount of time, that change in input takes to propagate the output.

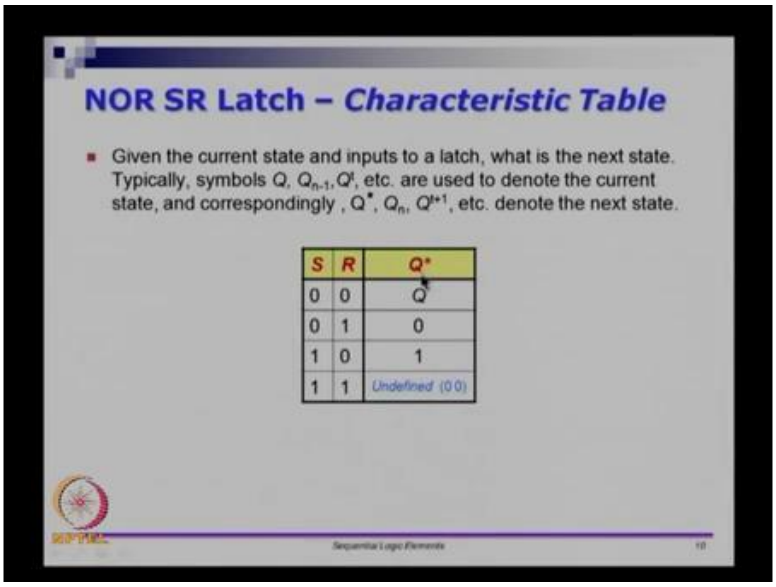
Let us say the both of them have the same amount of delay, then Q a and Q b both would have turn to 1 in this time point. If both of them turns to 1, now this is a 1 and S is 0, so Q b will try to change to 0. Similarly, Q b is a 1, R is 0, Q a will try to change to 0, so Q a and Q b both will again try to go to 0. So, both of them will try to go to 1 simultaneously, and then they both will try to go to 0 simultaneously. And then again they will go back to 1 and in go back to 0 and so on and it is not going to remains table at 0 or 1.

So, this is not the definition of a stable multi vibrator, both the outputs Q a and Q b will start oscillating indefinitely. So, this happens at the peculiar time, so this is a very careful condition. So, in English, they call it the perfect stam, the correct set of conditions which results in something like this. The correct set of conditions is, if both Q a and Q b are at 0 and R and S were at 1 and the both changes to 0 simultaneously.

If that happens, this is a perfect stam that makes the outputs toggle and no matter, what you do, the outputs are going to toggle continuously, this is not a stable system. So, if the inputs stay stabilized on at some point the output should stabilize, that is the nature of the circuits. So, if the inputs are not changing, if R and S come to a stable state, let us say both R and S remain indefinitely at 0, the outputs will still keep toggling from 0 to 1 to 1 to 0 and so on, this is a undesirable condition.

So, even though the circuits implementation is when both are 0, the previous values are retain, if R is 1, you can see that Q a goes to 0 and Q b goes to 1. If S is 1, Q a goes to 1 and Q b goes to 0 and if both are 1, they both go to 0. So, this is what the circuit does, but this is actually causing something undesirable, because if both R and S went from 1 1 to 0 0 at the same time, then the outputs being 0 0 will both try to switch to 1 1, and then they will again switch back to 0 0, and then so on indefinitely.

(Refer Slide Time: 27:54)



NOR SR Latch – *Characteristic Table*

- Given the current state and inputs to a latch, what is the next state. Typically, symbols Q, Q_{n-1}, Q^t, etc. are used to denote the current state, and correspondingly , Q^*, Q_n, Q^{t+1}, etc. denote the next state.

S	R	Q*
0	0	Q
0	1	0
1	0	1
1	1	*Undefined* (0 0)

Sequential Logic Elements 10

So, this is a bad thing, so what we are going to do is, we are going to characterize our S R latch. So, we have what is called a S R latch here, the name is there because of two signals S and R. So, we have two signals S and R and we are going to define this in a characteristic table. A characteristic table is actually simple, it has the 2 inputs S and R and we capture only the one of the outputs.

So, this state of a latch is usually just 1. So, the other one is just a compliment, let us not worry about the other one right away. So, what we are going to use is, we are going to state that the current time point, this is the value and if the inputs change, what will be the next value. Remember, the next value should depend on the current value as well as the inputs that have come in.

So, you will use the symbol Q, Q n minus 1 or Q of t to denote, what is the value at current time point and we will use Q star Q n are Q t plus 1 for what is suppose to be at the future. So, it is from time t to t plus 1 is going from current to the future, and then t plus 1 to t plus 2 is again from that current to it is future and so on, will use these notation. Either will use Q and Q star, Q n minus 1 and Q n or Q t and Q t plus 1 as much of possible, I will try and use Q t and Q t plus 1 to represent this.

So, let us see what a characteristic table is, if both are 0, then Q star or Q in the future is same as Q in the current value. So, if current value is Q, it to be 0 or 1, Q star will go to 0 or 1, depending on Q. Now, if R is 1 and S is 0, Q star are the next value will always be

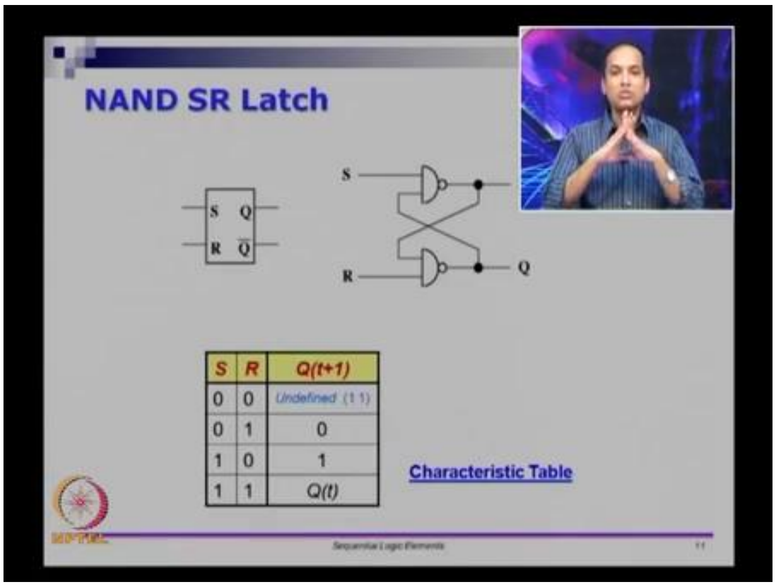
0, no matter, what in current value of Q is. Similarly, if S is 1 and R is 0, the output Q will always be 1, no matter what previous value is.

So, if the current value is 0, it will still go to 1, if current value is 1, it will remain at 1. So, that is the meaning of S R being 1 and 0. And finally, if both S and R are 1, this is undesirable, we do not want 1 1, because this 1 1, S and R being 1 1 and if for whatever reason, we saw that Q a and Q b are both 0, there is a problem. So, we do not want that condition. So, we will say that, that condition is undesirable or it is undefined, we will use this undesirable or undefined.

So, the key thing is, if you notice in the previous case, if Q a is 0, Q b is 1 and if Q b is 1, Q a 0, so in fact, instead of calling then Q a and Q b, we could call them Q and Q compliment, which is what I did in the previous slide actually. So, instead of calling them Q a, Q b assuming that these are two different things, if we did not have this condition 1 1 leading to 0 0. Let say 1 1 never happen, you can never give 1 1 as an input to the circuit.

Let us say some have you guaranty that S and R can never become one at the same time, then the output Q a and Q b will be compliment of each other. So, it is to give only one of them and not specify the other. We assume that in all these cases, you are talking about Q a of t and Q a of t plus 1 versus Q a of t will just call that Q from.

(Refer Slide Time: 31:24)



NAND SR Latch

S	R	Q(t+1)
0	0	Undefined (1 1)
0	1	0
1	0	1
1	1	Q(t)

Characteristic Table

Sequential Logic Elements 11

You can also design the same thing using the NAND based circuit. So, we have done the same thing as before except that. So, this is also like 2 NAND gates, which are couple to each other. So, this kind of alignment is called a cross couple alignment. So, we have two gates which are feeding, which are fed by R and S and there is some cross coupling between the outputs. So, this circuits is called a cross couple NAND circuit, the previous one we saw was a cross couple nor circuit.

So, in a cross couple NAND circuit, the behavior is as it was before. If S is 1 and if R is 0, the output Q of t plus 1 will be a 1, if S is 0 and R is 1, output Q of t plus 1 will be a 0, if both are 1, you will get Q of t, if both are 0, it is undesirable. So, this is the difference between a S R latch based on NAND versus and S R latch based on NAND. So, 0 0 is undesirable for a NAND based latch; however, 1 1 is undesirable for a NOR based latch.

And in a NAND based latch 1 1 will keep the previous value, it will retain a previous value of Q and Q bar, where as 0 0 will retain the previous value in the NOR based S R latch. So, these are basic conditions, S R latch could be either NOR based or NAND based. In a NOR based latch 1 1 is undesirable and in a NAND based latch 0 0 condition is undesirable.

So, we will see, why this is called a latch and all this undesirability in other things, how to get rid of it and so on from the next module onwards. So, as of now, just realize that S R latch and S R, S R latch based on NAND or NOR have some condition which is undesirable. Somehow, you should make that condition go away. So, what we do in the next module is, we will how to make the condition go away.

So, this brings me to the end of module 18 and I will see you in the next module, where we will talk about, how to get rid of condition, where S and R are taking the same value. We will see that in the next module.

Thank you and see you soon.