


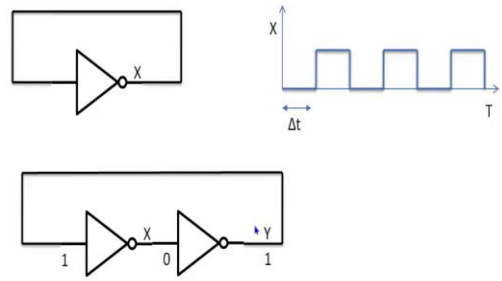
**Digital System Design**  
**Professor Neeraj Goel**  
**Department of Computer Science Engineering**  
**Lecture 41**  
**Indian Institute of Technology Ropar**  
**Latches**

(Refer Slide Time: 0:15)


Design of basic storage element



- Digital circuit with feedback



Digital Logic Design: Sequential Circuits



So, let us start with latches, latches means we want to design some sort of a basic storage element or a basic memory element. Now, because we want to design a memory element the basic technique for designing that would be using a feedback tool. So, in sequence, in combinational circuits in our previous module or so far whatever we have studied, we never seen a circuit where output one of the output or the output is connected back to the input.

What would happen if those outputs are connected back to the input? It is interesting, it is a different problem and it is a challenging problem as well. So, let us try to understand with the help of a inverter. So, if my inverter output let me consider that this input inverter output is X and this X is given back to my as input to this inverter what would be the output yes, most of you will be able to guess it correctly the output is going to be our oscillator.

So, as soon as this X will become 1 this X1 would be given as an input to this inverter then this 1 will become 0. So, we can put it like this. So, let us see initially my X was 0 then after giving it a you know input to inverter then after some time it will become 1, as soon as it will become 1 after sometime it will become 0, so it would keep on rotating between 1 and 0, 1 and 0 so, it is essentially an oscillator.

So, does it mean that I can use this circuit to design an oscillator design a something regular structure regular output signal like this? Yes, of course, we can design but the question would be that what is the time period of this this kind of oscillating signal. Yes, here also you are able to guess correctly then the time period would be equal to the delay of the inverter.

So, since inverter is a very basic fundamental unit and it would be used in so many different designs, so, this  $\Delta T$  in current technologies would be very, very small, infinitely small. So, we can so let us see if your processor or your system is working on gigahertz, so if you buy in a market and you buy and see what processes are there.

So, most of these processors are most of these edges they run at a couple of gigahertz frequencies, now so that means if they are working at couple of gigahertz frequency the delay of inverter you can imagine that it would be on the order of Pico seconds maybe couple of Pico seconds or 10s of Pico seconds.

So, if Pico seconds is the frequency, so that means this particular signal would be a frequency somewhere around terahertz. So, that could be a reason that particular circuit is not usually used as a oscillator design, because first of all we cannot control the frequencies and also we cannot vary different frequencies and the time period is also, this time period will also vary as the temperature or ambient characteristics will change.

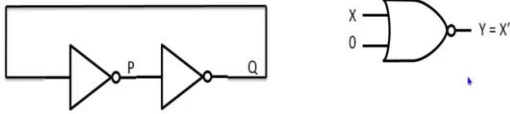
So, yes but the question here we wanted to see was can we design a basic memory element? So, instead of one inverter, if we keep two inverters, then? Yes, so this would be the circuit. So, let us say if there is an input of Y here, and that Y would be inverted as a X here and this X would be noted again is a Y. So, essentially, if somehow initially in this circuit, 0 was taught, so, 0 was the output here, it will always remain 0 as soon as power is available to this particular circuit. Because the 0 would be given as a input to this inverter output of this inverter X would be 1 and because X is 1 here, Y would be 0.

So, this particular circuit pair of two inverters are able to store this 0 as long as power is there. Similarly, if somehow, we are initially able to say that the circuit to be given, the circuit need to be stored one here as the output or in this particular wire one should be stored, then this circuit would do good, the same circuit would do good but somehow we have to initialize or start that the output has to be fun.

If it is 1, then circuit has capability to remember it as 1. So, in this way, if I have two pair of inverters, this sort of work like a memory element, they are able to store one bit of information, they can store both 1 or 0.

(Refer Slide Time: 6:03)

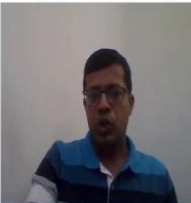
**Set Reset Latch**



How to set a particular output?

NPTEL

Digital Logic Design: Sequential Circuits.



But the unfortunate thing is, we are not able to set these values to 1 or 0. So, now instead of X and Y, I am writing it as P and Q. That is more like more standard terminology. So, if I have these two outputs, one is P another is Q the question which I am trying to say here that how can I set a particular output?

So, one thing is clear that we are able to set so, we are able to store this particular output, but the question is that how do we initialize, how do we say that this particular output would be 1 or 0. So, if it is 1, it will store 1 if it is 0, it will keep on storing 0, but we do not know how to initialize or how to set that particular output.

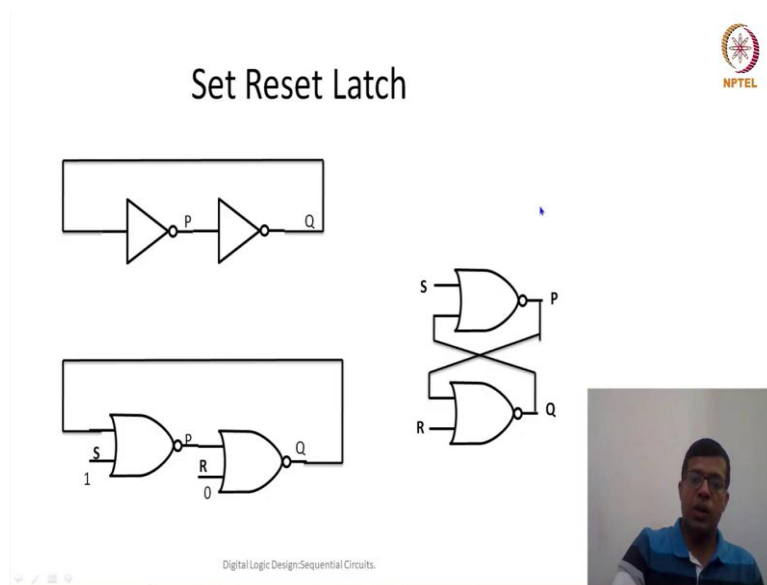
So, to set a particular output, we can remember that our NOR gate, our NAND gate, both of them can work like an inverter if both the inputs are connected. Or they can also let us take an example of a NOR gate. So, if one of the inputs is 0, and the other input is let us say X another input is variable it is it is connected to my variable input but the one of the input is 0, then my output is the inverted of inverter over like or basically inverse of whatever is the input X.

So, let us say try to recall what was our truth table for NOR gate. So, when it is 0, it depends on the X what is the output, if it is 0, so let us say if X is 0 the output would be 1 if X is 1 output would be 0. Now, on the other hand, if the other input is tied to 1, it will make sure the

output is going to be 0. So, the overall functionality would be if one of the input is tied to 0 then this NOR gate is effectively working like an inverter, while if this particular input is 1, the other input other end of the input is 1 then it will make sure the output is always 0 irrespective of X.

So, let us assume that we are always going to have this 0 here the other input is tied to 0 and then we can replace this inverter both of these inverters with our X sorry, our NOR gate. So, now we are replacing our inverters with the with the NOR gates.

(Refer Slide Time: 9:04)



Yes, so, we not only we can replace our inverter with an NOR gate, we can also replace them with the NAND gate, the overall mechanism is going to be seen or at least similar the only workout would be that what would be so here instead of 0 in case of 1, it is going to work like an inverter.

So, otherwise, more or less, whatever we are, whatever analysis we are going to do here, that else is equally valid for NAND based design also. So, as we said before, that we are considering that this other end of the input is connected to 0 0. So, it is effectively the storage element, whatever is Q, it will always remain the same value. It is not going to be changed.

And the other question which was before us that how can we set this Q, now we have the mechanism for setting this Q. So, what we can do is, let us say we can, if you want to wake, let us keep the other input at 0, but one of the input, let us say this input, I will make 1, if we make this input one, it will make sure that my Q would be 0. And because Q would be 0, P would be 1.

So now, in addition to storage element part, I also have a mechanism to set my output or set my Q to 0. So, to set my Q to 0 I for this particular NOR gate, I have to make input as 1. So, because I am making my output as 0, so I can call this particular input as a reset, because if it is 0, then it remains whatever is the previous one, but if reset is 1, then my output is 0.

Similarly, for the other NOR gate if I for the other NOR gate, if I make this as 1 because it is 1, it will make sure that the output at P would be 0, and if the other input is tied to 0, so that means my 0 and 0 1 output, the output at Q is going to be 1. So, in other words, if I am putting 1 here at the input of this NOR gate then my output is 1.

So, I can call this particular input as set. So, the overall behaviour would be if set is 1, my output would be 1, if set is if reset is 1, then my output is going to be 0. So, that is why this particular latch is called set reset latch and when both R and S are 0 0 that means the output is not going to be modified then this particular circuit is working like a memory element.

So, it is going to store whatever was the previous input given to the latch. But if I want to modify, I want to initialize to it, initialize my input as 1 or 0, so I can correspondingly set my value of R and S accordingly. So, if I want to set my stored element as 1, then again make S equal to 1, R equal to 0 if I want to make my stored element as 0 then you can make S as 0 and the reset S as 1. This whole circuit which is our SR latch can also be represented or can be shown as a cross coupled NOR gates.

So, you can see that this NOR gate S is the input here the other input is coming from the second NOR gate. So, the other input is coming from the Q and the output of this particular NOR gate is P. Now, for this particular NOR gate, the input one of the input is coming from P the other input is coming from R and the output is Q. So, just remember that my R is connected to a NOR gate whose output is Q my S is connected to a NOR gate whose output is P, so this cross coupled circuit you will see more popular wherever you will try to see in books or maybe other resources.

(Refer Slide Time: 14:21)

### Set-Reset Latch

- What if R and S are both 1
  - Both output would be 0
  - Both R and S change to '0' simultaneously
    - Unstable output
- For valid inputs both the outputs are complementary to each

Digital Logic Design: Sequential Circuits

So now, this is also, this can also be represented like a block diagram where R and S are the input and there are two outputs, typically, it is called Q and Q bar because, if this is 1 this is going to be 0 if it is this is the 0 then this is going to be 1 so this is called Q and Q bar complementary to each other.

Now, we have seen 3 cases where R and S both were 0, we have seen a case when R and R was equal to 1 and S for 0 in case R is 1 S is 0, then my Q is going to be 0 if my S is 1 R is 0 then my Q is going to be 1 and Q dash is going to be 0. So, this behaviour is but what is the what would be the output if both R and S both are equal to 1.

Now, when both R and S are going to be 1, then there are two interesting observations, one that when R and S both are equal to 1, when R and S both are equal to 1 because it is 1 and it is because it is a NOR gate it will make sure that Q as well as Q bar both will be equal to 0. So, this goes this goes beyond or basically be, is not going along with the definition of Q and Q bar.

So, we are saying that one of the output is complementary with the other input, other output. So, if both of the outputs are zeros that means we cannot represent it like Q and Q dash. That can still be beared. So, we can we can still bear this fact that output is not complementary to each other. But there are other challenges also.

So, the other challenge is that suddenly, when both R and S were 1, and suddenly both of them will become 0, at the same time, simultaneously both of them will become 0. Now, if

you remember that your R and S both are 0 previously, they were both 1 because previously they were both 1. So, Q and Q dash, both were 0.

Now, R and S is 0 Q and Q dash is 0 what would be output, what would be the new Q and Q dash, both of them would be 1 because the 0 is both, the input to both of the inputs of this XOR sorry NOR gate is 0 both input of this NOR gate is also 0. So, the both of the output are also 1. Now, since the circuit is identical, we have to assume the delays are also identical, if delays are identical to both of them, so at the same time, because this one will come to input at the same time for both of these NOR gates.

So, because now input is 1 to both of them, the output will become 0 again after some Delta time, now this delta time, after this delta time, both of this after that delta time the 0 would be presented as input to both of these NOR gates. So, then output will again become 1. So, this 0 1 0 1 will going to give us an unstable output.

But remember, this unstable output would be there when R and S were initially one and both of them change to 0 at same time, this is also called race condition. Why it is called race condition? because now both of my R this both of these may NOR gates they are racing ahead racing against each other.


Now, let us see one of them is little faster than the other then it will converge to either one or 0 we do not know whether it will converge to 1 or whether it will converge to 0 it depends on the delay of individual gate. Now, delay of individual gate may depends on 1000s of factors. It could be your fabrication method, it could be take 2 Watt wires, each Q and Q bar connected to what wires my R and S is connected to. So, what are the (( )) (19:05) capacitance what is the temperature and what is the variation of my silicon here and there.

So, because of so many factors, we do not know which of the output would be correct. So, because the output is unstable, because we do not know what is going to be the final output after this whole process. So, that is why simply in case of set and reset latch, we say the input combination 1 1 is invalid. So, that means we are going to say that whenever the combination is invalid, the output is going to be unknown. It is undeterministic, that is way better we will never ever give 1 1 as the input to set reset latch.

Let us try to recollect there are two reasons one reason is this unstable team the output the other reason is that by design we have seen there are certain advantages, if we make this Q, one of the output is Q another is complement of Q, so the P is equal to Q bar, so that that

helps us in internal design internal design of my RS latch. So, with this we can also confirm that my output because we were saying that both R and S can never be 1. So, this also gives us an corollary that your my output is always going to be complementary if we consider that as a invalid case.

(Refer Slide Time: 20:46)



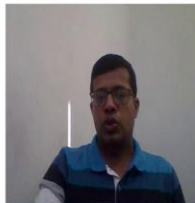
### State-table SR Latch

R	S	Q	Q'
0	0	Q	Q
0	1	X	1
1	0	X	0
1	1	X	Invalid

Next state equation:  
 $Q^* = S + R'Q$

SR Latch can be created using NAND gates as well

Applications:  
 -Storage  
 -Debouncing



Digital Logic Design: Sequential Circuits.

So, in other words, we can write this whole state table you are not calling it truth table we are calling it state table because, this value is state which we are trying to remember. So, when R and S both are 0, whatever was the Q whether it was 0, my next state is also going to be Q. So, if Q was 0 Q plus or basically next state of Q will also be 0, if Q was 1 the next state is also going to be 1.

But if reset was 0 set was 1 my irrespective of previous information in Q output is going to be 1 similarly, if reset is 1 and set is 0, then irrespective of previous state previous output, my next output is going to be 0 1 1 we are saying that whatever is the previous output, the output is going to be invalid. So, this this combination is invalid we are not going to use it.

So, we can also represent if we solve this as a K map problem or a combination problem, then we can see that we can represent Q plus or the next state of the output as S plus R dash Q. So, this is called next state equation, because it is telling us what would be the next output and next output depends on you see the two things previous input and current input, current input is R and S the previous input is stored in form of Q.

So, Q is the state information or the previous state information or the information which you have classified that based on all the 0s and 1s we have stored my current output SQ. So, my




next output depend on the current output as well as the current inputs and current output is the reflection of previous inputs of, previous input to this particular latch.

So, as I have said before, so this RS latch we can create using NAND gate also, but in case of NAND gate, this state table would change a little bit. So, because of the inherent nature of NAND gate you can take it as an exercise you can create SR latch using NAND gate. And yeah, so if we have created such kind of SR latches they can be used in several applications like we can use them to store some values and the other industry emulation is debouncing.

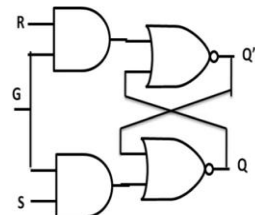
So, debouncing means that let us say we have certain place where there input keep on fluctuating. So, if input keep on fluctuating then we can design a switch like this that if it is connected to R although it can keep on fluctuating, but it will always give the output to 0 but if it gets connected to S, then it will always give output 2 output as one Q as 1. So, whenever there is a fluctuation in the input, we call it debouncing. So, over debouncing we can use this this SR latches.

(Refer Slide Time: 24:30)




### Gated Latch

R	S	G	Q	Q'
X	x	0	Q	Q
0	0	1	Q	Q
0	1	1	X	1
1	0	1	X	0
1	1	1	X	Invalid



Next state equation

$$Q^+ = SG + Q(R' + G')$$



Digital Logic Design: Sequential Circuits.

So, we will quickly cover two more topics now. So, now let us say here my output depends on S and R. Now, whenever S and R both will change or any of them will change that will change my output. Now, we are talking about clock signal sometime in initial part of this lecture. So, this clock signal is a common control signal which is found in most of the synchronous designs, so, this kind of a common control signal, which will make sure that changes would happen only with that control signal.

So, that control signal will define when to store into my latch, when I can avoid storing or when, when this input would be given to the output or when it would be gated. So, that is why these called gated latch. So, G works like a gate if G is open, then only R and S will work if G is closed, then R and S will have no impact on the output.

So, in other words, so, to implement that, what we have done is we have put up AND gate before R and S. So, if G is 1, then only R value will go here, if G is 0, then the output this particular input to my both of these NOR gate is going to be 0 0 because input is 0 0 whatever was there in Q and Q R they will remain same, they will not get modified.

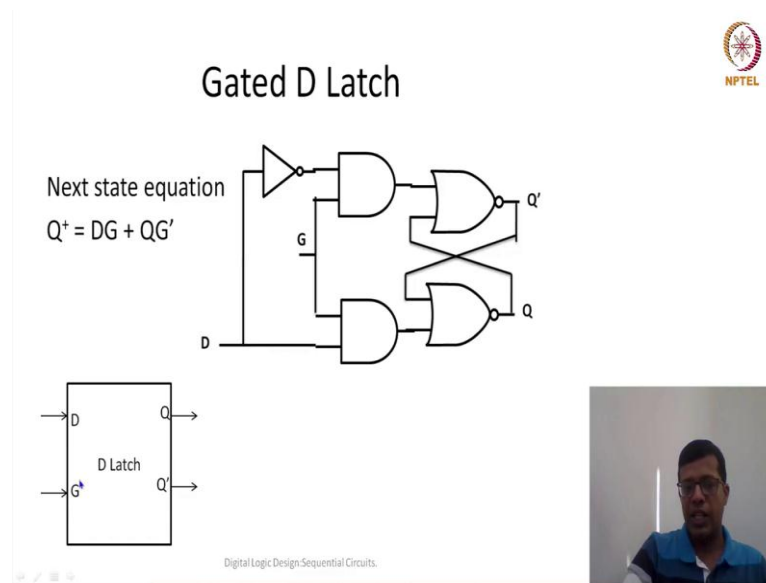
However, if my G is 1, then R and S can be given to these NOR gates as R and S over there in the previous RS latch. So, in other words, I can write this state table like this, that if my G is 0, it does not matter what is the value of R and S, my output my next state is going to be same as the previous state, my output will not get modified. Similarly, if G is 1 then it is effectively working like a standard RS latch. So, if G is 1 my output if an R and S both are 0, then next state is equal to the previous state. If G is 1 set is 1 and reset is 0 my next state is going to be 1 irrespective of the previous state.

Similarly, gate is 1, G is 1 and reset is 1, set is 0, then output is going to be 0 irrespective of the previous output. 1 1 is still an invalid combination here. So, but you can notice that 1 1 could be a possible combination here, if G is 0, but is G is 1, then it is saying invalid combination, usually the design should be such that R and S should never be 1 1.

So, although it is possible here because it is don't care condition, but otherwise, for the better circuit design it should not be, it should never be 1 1. Now, if using this truth table, if I would like to design my next state equation, so next state equation means the output Q plus output, Q plus is the next state or next output.

So, this I can decide using the gate it would be there. So, if G is 1 then S effect of S would be there, then Q plus would be 1 otherwise, it depends on whether what was the previous input, whether reset is 1, if reset is 1, then my output is going to be 0 and it at the time G also has to be 0. So, this can be created simply by you solving the K map for this these four variables.

(Refer Slide Time: 28:43)



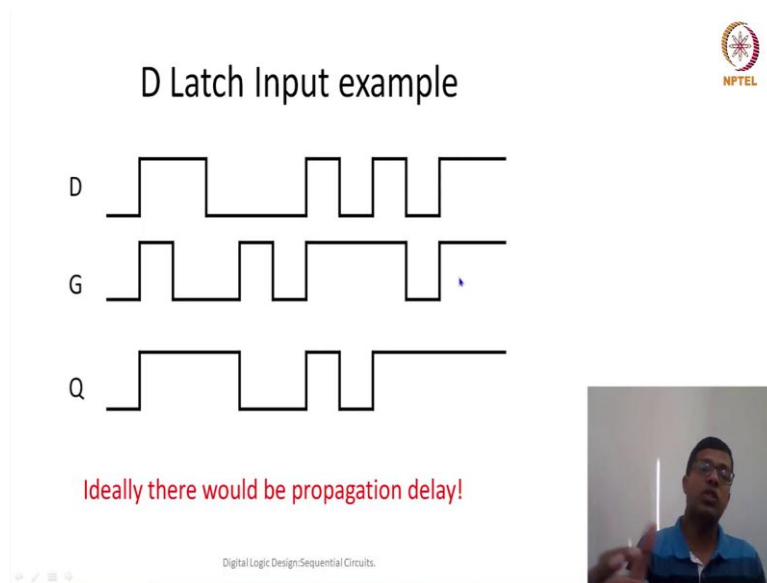
Now, there could be one more interesting variation of this gated flip flop, gated latch. So, in that case, what we do is instead of two inputs, let us create only one input which we will also make sure that we never arise this particular condition will never arise that both of them is going to be both of them are going to be 1 1. So, here whenever D is 1 then this is going to be D dash, so that means whenever 0 is given the input to the other latch, other NOR gate is 1, whenever input here is one it will become 0.

So, this particular D latch has removed along with the removing input SR equal to 1 1 condition it has also removed 0 0 condition which means my output if D is 1, my output is 1, if D is 0, my output is 0. So, that is why this particular D latch is also called an transparent latch, which means whenever G is 1, D value would be reflected in the output, if G is 1, then whatever is the value of D that should come to Q and Q bar.

So, whatever is the value of D, it will come to Q and whatever is the value of D dash will come to Q dash, if G is 0 that would help us in getting this S equal to 0 R equal to 0 condition that means, whenever G is 0 whatever was stored that would be retained in the output or this would work like a storage element whenever G is 0.

So, the next state equation we can simply write Q dash equal to Q plus, next state Q is equal to DG. So, that means, whenever whatever is the value of if G is 1 then whatever is the value of D is the next state if G dash, so, basically if gate value is 0 then whatever was the Q that would be the next state. Symbolically we can represent this as a D latch here G is a input, D is a input, Q and Q bar are the output. To understand this D latch better, let us try to quickly see through one example.

(Refer Slide Time: 31:13)



So, let us say this is a waveform for input waveform for D and this is input waveform for G. So, what should be the output, so you can see that this is the time axis and both if it is high, then you can consider it as 1 if it is low we can consider it as 0. So, let us look at the output. So, initially all of them are 0 and we consider that my initial state is also 0.

So, we are assuming about our initial state as soon as gate is 1 G is 1 my output would be equal to whatever is a D. So, if D was this my output is also 1. Now, after this one cycle, after this time, now G has become 0, now because G has become 0 D is 1, but you know when G is 0, then we my D latch will not even look at the value of D, but whatever is the previous output, previous output was 1 so, it will remain as 1 till my gate is 0.

So, gate is 0 till here, my output will remain one till this point. Now, at this particular moment of time my G has become 1. So, because G has become 1 it will now try to see what is the value of D, D value is 0. So, because D value is 0 my output will become 0, now further after some time gate has become 0, but because it is 0 it is still not looking at what is the value of D but till gate is 0 whatever was the previous state 0 that will be extended till G will become 1.

So, whenever G will become 1, then we will start looking at our D value. Now, G has become 1, at this time whatever was the D value it is transparently given as the output. So, now G is 1 for this long duration. So, this long duration whatever was happening in D that is happening at Q also, that means initially it is 1, then it has went to 0, then it has become 1. Now, at this moment of time it is a little tricky.

So, now G has become 0, because G is 0, we do not see what is the value of D, we simply copy whatever was the previous state, previous state was 1, so it is still 1. After some time, G became 1, so as G became 1 my output will be same as whatever is a D value so the output is 1. So, similarly, we can work out a different kind of inputs and for different inputs we can see what could we need various outputs.

This should be taken as a homework exercise we should see various combination of inputs and see that what would be the output corresponding to those input combinations. So, with this I would like to close my lecture and yeah, so with one more comment that although this particular whatever clock or whatever waveforms we have created.

So, these waveforms considered that the delay of this D flip flop is almost 0, but with propagation delay, for example, in this critical scenarios, this propagation delay would be important that based on the propagation delay, it would determine or it would decide what would be the final output.

(Refer Slide Time: 35:05)

Summary

- Latches

Digital Logic Design: Sequential Circuits.

So, in summary, we have seen we have understood what sequential circuits are, what are synchronous sequential circuits, how much amount of memory is required to design such a circuit. So, we have seen that how a feedback circuit can be used to design a simple memory element or simple storage element and then we have seen various combination that how beautifully people, we can write a 1 or 0 how we can use a simple inverter to design an SR latch or latch that would be able to store some values. So, with this I close today, and we will see tomorrow, more about these latches and how do they get connected to have more sophisticated memory elements. Thank you very much.