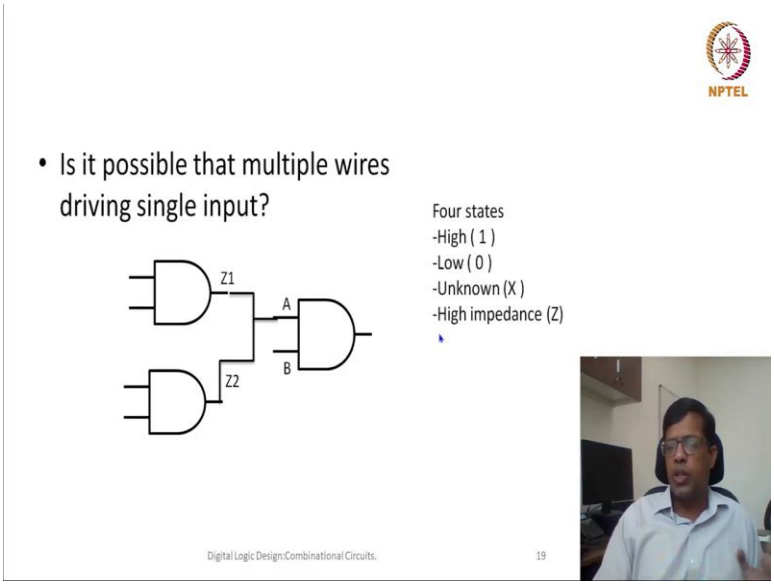
**Digital System Design**
**Professor Neeraj Goel**
**Department of Computer Science Engineering**
**Indian Institute of Technology, Ropar**
**Four State Logic**

Hello everyone. We have been studying about how combinational circuits can be designed, how we can formulate Boolean equations from English statements. And now we also know that how different gates can be combined to make a larger circuit. We have seen that a fan in of a gate would impact delay and fan out will also impact delay. So, in other words, we have seen that a particular gate can have multiple inputs and each of the output can go to multiple other gates as input.

I would like to raise one question here that have you ever thought what would happen if multiple of the outputs will go to the same input of the gate? In another words input of the gate is coming from multiple different wires, so different signals. What would happen in that case?

(Refer Slide Time: 01:27)




So let us take this example. Let us say I have two AND gates. These two AND gates, one of the AND gate has output Z1. Other has output Z2. Now both of these outputs are going to one input A. So what would happen in that case? We have to understand that these are all wires. This wire is connected to this wire. And when, let us say, consider that when there was only one, each input was received from one output.

So let us assume that this Z2 was not connected to A. What that means was that whatever was the voltage at Z1 would be transferred as the voltage at A ignoring the potential difference or loss of voltage due to these wires. Now we are assuming here that Z1 means that, 1 would mean voltage, high voltage and 0 means low voltage, ground.

So if both Z1 as well as Z2 both are connected to same input what would happen? So let us say Z1 is high, or Z1 is 1 and Z2 is also 1. In that case what would be A? A will also be considered as 1. Similarly if Z1 as well as Z2 both are 0 then also A would be 0. But what would happen if one of them is 1? Z1 is 1 and Z2 is 0. What would be the A here? It cannot be determined.

So what would happen in that case? This particular AND gate may not function correctly. Not only this particular AND gate functionality would be different word but it can also impact the functionality of these AND gates. You may ask why. So you see; let us say the voltage here was high. And voltage here was low.

So that means voltage here at A would be somewhere in-between. Or it may develop some short of short circuit from high voltage to low voltage. So we do not know what would happen here. So because of this uncertainty this gate certainly will not function as a correct functionality. So we call all of these, this particular input because it is not deterministic that what would be the input A. So we can call that, this particular input is undeterministic or unknown. We do not know whether it is 0, whether it is 1. It depends upon a lot of factors. Plus there is a lot of things that would, that can go wrong if both of these things are short.

Now, so this is called unknown. Similarly let us say, if there is a, this particular, other input of this particular AND gate, is not connected at all. That means it is left unconnected. What does that mean? Leaving unconnected means that there is no connection.

And what would be implied for the circuit? How circuit should behave if a particular input is not connected? So from the circuit perspective we can call that if the input is not connected then we can call it as a open circuit. And from the circuit perspective this open circuit can also be represented as very, very high resistance or infinite resistance. So if, in-between two points, let us say there is a point B and there is a ground here somewhere. Ground means zero voltage. So there is a infinite resistance between this point and the ground. That is why no current can pass. And it can be treated as a open circuit.

So that means in real circuits, when such kinds of situations occur, there would be more than this 0 and 1, two values which are possible. So in realistic scenario we can call that these digital circuits can have more than two values. We can call it four logic system. So there could be four different logic states which are possible. 1 is high which we can say 1. Or the other state is 0. This 1 and 0 we are familiar with, and that is what we were calling binary numbers, or binary representation, Boolean algebra. So all of them have only two states, 0 and 1.

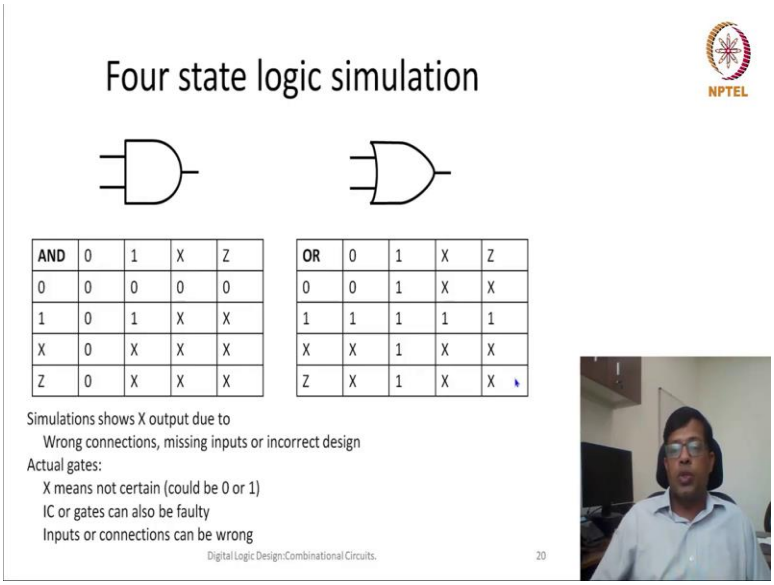
But in real circuits for a real hardware there could be these two additional states, one we have to call unknown. We do not know whether it is 0 or 1. Certainly it is between 0 and 1. Certainly it is between very high, so whatever this high voltage represents. Let us say it represent 5 Volt and low represent 0 Volt. So it could be anything between 5 Volt and 0 Volt. But we do not know what is the exact state.

So then there is other fourth state which is also called high impedance. High impedance means high resistance. And impedance, if you know the circuit theory then impedance means it could be resistance plus capacitance plus inductance. So it is a sum of all of them, effective sum of all of them. So high impedance means there is no connection between this particular point and the ground. Or in other words, if the connection is there resistance is so high that current cannot pass. So effectively whatever is the voltage, at wherever the voltage is there, so that voltage cannot be transmitted here.

So if real circuits will have these four states then; that means in our digital circuit also we have to take into account these four states. We have to see what would happen when any of these states will trigger. Why we are focusing only on 0 and 1? And why should we call this circuit as binary circuit or digital circuit?

So there are two things. One, that even though there are four states still the number of states are finite. Second, that in correct circuits we will try to avoid the last two states, unknown as well as high impedance as much as possible. So whenever we see unknown that means it shows some sort of an error. And high impedance is a very specific scenario which also should be avoided.

(Refer Slide Time: 08:56)




Now, so given let us say, assume that we have to work with these four states. So how our typical AND, ORs are, AND, OR gate would work like? So let us say this is an AND gate. And this is first input. This is the second input. So given two inputs how it would behave? That if 0 is ANDed with 0 then it would be 0. 0 is ANDed with 1 it would be 0. 0 is ANDed with X then also it would be 0. What does it mean? That it does not matter what the other input is. If one of the input is 0 output has to be 0. Even one of the input is 0, other input is Z then also my output has to be 0; because if one of the input is 0 it does not matter what the other input is.

Similarly if one of the input is 1 then output 1 would be possible only if the other input is 1. Otherwise if the other input is X then output would be X. Other input is Z or high impedance then still the output is unknown. We do not know what the output is. Similarly if one of the input is X and the other output is also X output is unknown. If one of the input is X and the other input is Z then also output is unknown.

So you can see that even with this four logic we can create this kind of a truth table and we can know that what would be the output. So you see that if one of the input is either X or Z then there is a good probability that the output would also be unknown. So which means that my circuit functionality has gone bad. It may not work like the intended circuit.

Similarly we can construct this truth table for OR gate also, in OR gate the logic is that if one of the input is 1 then there is a certainty that output is going to be 1 irrespective of the other input. If

the other input is 0 then also output would be 1. If other input is X then also output will be 1. If other input is Z then also output would be 1. But in other cases, let us say one of the input is X. Other input is X. Then output is unknown. Similarly one of the input is X and the other input is Z then also output is unknown. So what it essentially means that if your input is unknown then output will also go unknown.

So if we try to see it in full perspective, so that means whenever we are trying to design, it is completely prohibited that we connect two outputs to the same input. It will create lot of uncertainties. So other implication is, the reverse implication would be, let us say if you are trying to design something in Verilog. If you are trying to design something in Verilog then you see in your output waveform there is some X which essentially means that either there is some wrong connection or your input was X so the output is X. Or input was Z then also output could be X. Or your design could be incorrect.

So why we are talking about Verilog here? Because in Verilog whenever we declare any particular signal as wire or we declare any particular signal as register R E G REG, so then implicitly Verilog assumes that these are the four states of the signal which are possible in any of this, yeah. So these four states are implicit in Verilog so this is trying to represent what a realistic scenario would be.

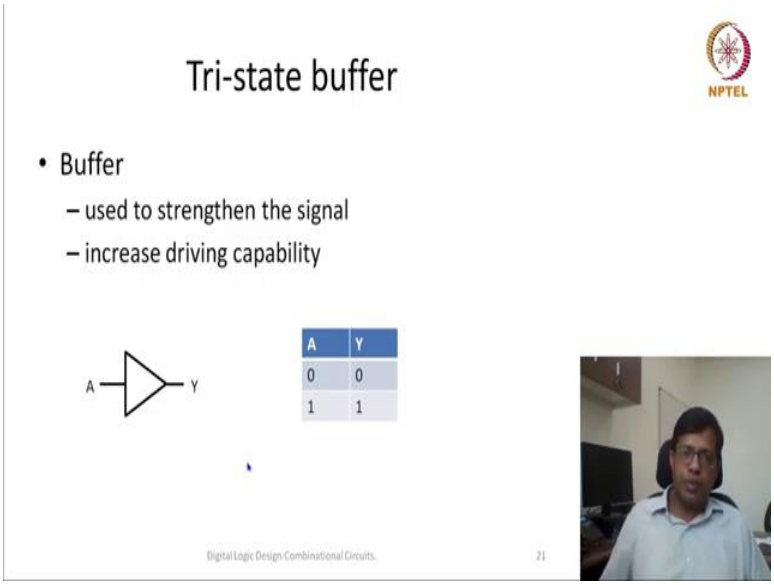
How or what would happen in the real circuit? Let us say you are trying to design something on PCB. You are trying to design something on breadboard and you are putting such kind of wires. What would happen? So wherever we expect that, wherever your Verilog simulation is showing X, in those cases in real circuit there could be a short circuit. Your circuit could fail. Circuit could burn. Because we do not know whether it is 0 or 1 and because of that large current is being drawn from these two wires what are getting connected, what are getting short-circuited. It can lead to faults. It can lead to failure of whole design.

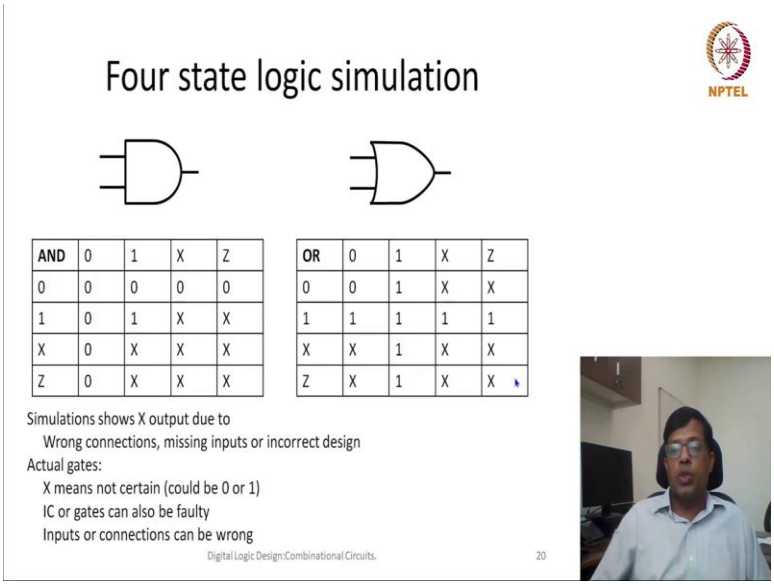
The other thing, if your output is not showing correctly or like this, there is also a possibility that internal, internally also there could be fault. So here IC could also be faulty. So, otherwise if input output connections are not connected correctly then also it can show different kind of errors. But the overall message here is, yes, overall message here is that yes, if two wires get

connected to the same input then there is a good amount of possibility of failure at larger level also.

So then one can ask here that, but what if I have to drive? I need to drive my same input from multiple sources. So, in other words, what you want to do is you want it to; you want it to drive the same input from multiple sources. We have given some example in the one of the previous lecture that let us say you wanted to have one speaker. So the speaker could take either input from the computer or it could take input from your mobile phone as well as it could take input from some Bluetooth. So there has to be some method, systematic method. So one of the method which we discussed in our previous class is multiplexer. So in multiplexing some control will decide which one to take.

(Refer Slide Time: 15:48)



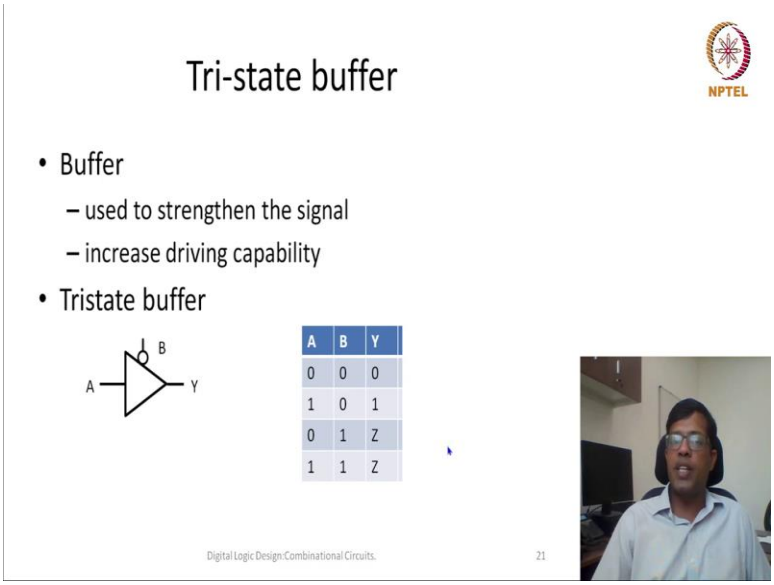




But there is other method also which is possible, which is called tri-state buffer. So if somehow, if you see our NAND or AND state, so if you take Z as one of the input while the other input is, other input is clear that it is 0. If it is 0, in case of AND it is going to be 0. If Z and you are doing an OR gate the output is going to be 1. So that means if one of the state is Z then it can help you in driving the circuit correctly. So, in other words, what we are doing is that one of the circuit, one of the wires is getting disconnected. Only one of them is active at a time.

To understand what tri-state buffer is let us first understand what is a buffer. So buffer is a simple circuit where the output is same as input. Here Y is the output. A is the input. And truth

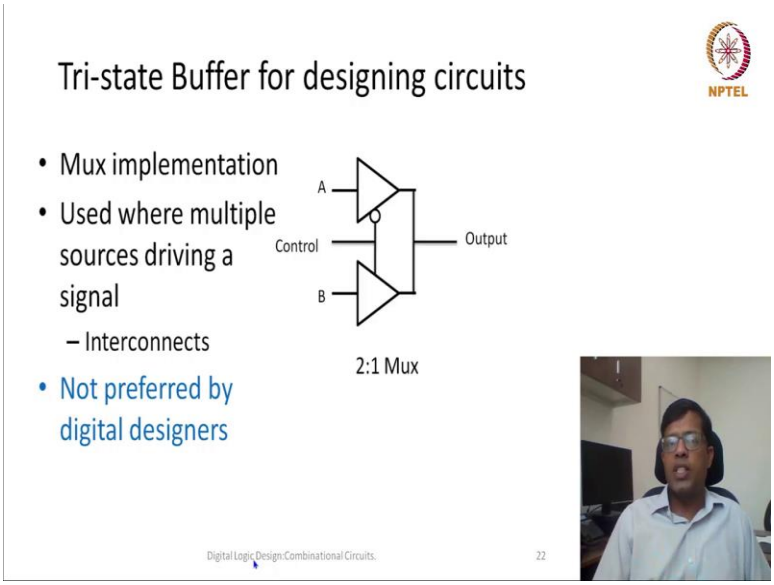
table is that if A is 0 Y is also 0. If A is 1 Y is also 1. Why do we use it? Buffers are essentially used whenever one particular output is driving multiple inputs. So because one output is driving multiple inputs so your strength of the signal has to be much higher. In other words, driving capability has to be much higher. So, sometimes designer will introduce this buffer. This buffer will increase the strength of the signal. Now you can, you can have multiple; you can, one output can drive multiple of the inputs.

(Refer Slide Time: 17:41)




Now in tri-state buffer, what would happen is you have additional signal here, B. So if B is 1 then this will work like a normal buffer. But if B is 0 then the output is going to be high impedance. So when B is 1 then it is working like a buffer. But when B is 0 then it is high impedance. Or sometime the functionality could be reversed if there is a bubble here. Bubble means that B is inverted. So my control input here is inverted, which means that if the control input is 0 then it will work like a buffer. If control input is 1 then the output would be high impedance.

(Refer Slide Time: 18:35)




So now these tri-state buffers can be efficiently utilized to design multiplexers. So how? Let us say I am driving, designing a simple circuit where this A is input, this is control. Now if control is 0, if control is 0 that means A would be passed to output. But because it is 0 it is going to be high impedance. That means it is going to be open circuited. It is not connected. Because it is not connected, so we can say finally output is equal to whatever is the output, output here is whatever is the output at the end of this particular gate.

Similarly when B is, control is 1. If control is 1 this is going to be in the state of high impedance or this particular circuit is not connected. But this one is working like; here the output would be B. So effectively this design is a 2 is to 1 multiplexer. So and such kind of implementation can be used whenever we want it to drive multiple sources. So we want it, one particular input to be driven from multiple sources. Interconnect or buses could be a good example. So what would happen? So let us say, your CPU is connected to memory. Your memory is also driven by CPU. Your memory is also driven by GPU. And similarly there are other processors also which are connected to memory.

But finally what do you want? Only one of them to be driven, driving this memory. So, in that case, I will have a intercorrect where we can use these tri-state buffers to effectively say that only one of them should be giving connected. So although on physically it looks like all of them are connected, but because of these tri-state buffers all other has to be in high impedance. So one of

them is connected to the output. All of other because they are in high impedance state, there they will not be connected or they are essentially open-circuited. So these tri-state buffers can be used this way to effectively design multiplexers and some other circuits also this is being used.

Earlier, like some twenty years back when we were in nascent stage, when we were started, when we started designing these VLSI chips or digital circuits, so at that time these tri-state buffers were very, very popular. But these days when we are automating most of the design, when we are using very low to design the final hardware, and while most of the designing is done by the EDA software or Synthesis software; there these tri-state buffers are usually not preferred.

We are teaching this to understand couple of these concepts. So one concept we have understood using tri-state buffer is that it we are not supposed to connect two of the output to the same input. The other thing is yes, we can connect if we can have, use these tri-state buffers in-between.

But designers do not use it because, because they cause a big hurdle in verification. So usually whenever we are doing verification we see in the simulation, whenever we see X somewhere that means unknown values. Or we see Z somewhere that means it is a trigger point that there is some error. And those errors can also be, can be recognized easily and then we can start fixing them. But if design has intentionally put these tri- state buffers then Z as a input is a valid input.

So because verification become more challenging or checking design rules automatically also become challenging. So that is why mostly it is not preferred by digital designers. So, but we have to understand the concept. We have to see that how this works and let us also try to minimize the use of these tri-state buffers in your designing also. So this is the finishing of one particular topic in this lecture. Now let us look at some other design examples or design statements.