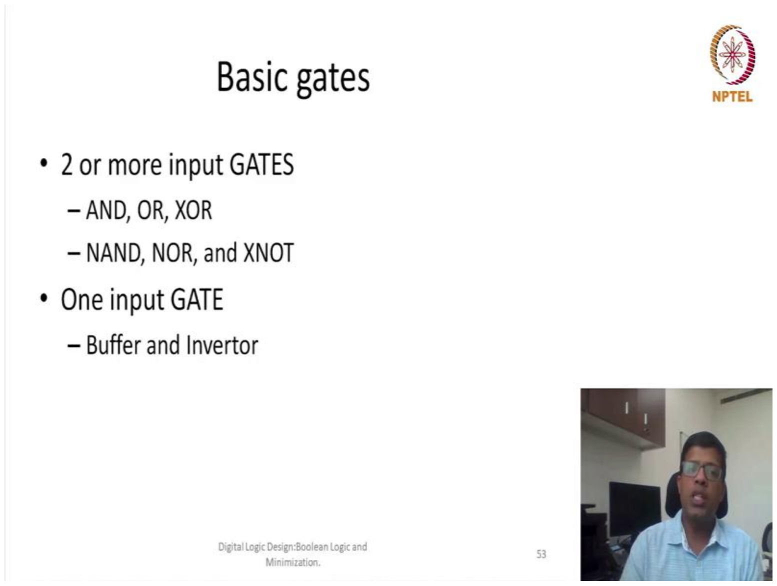
Digital System Design
Professor Neeraj Goel
Department of Computer Science Engineering,
Indian Institute of Technology Ropar
More Logic Gates

Hello everybody. Today we will discuss about more basic gates. So far we have seen three basic gates or basic operations in Boolean logic. One is AND gate, OR gate and inverter or a NOT gate. So, we will see that what are the other basic gates which are also considered basic or primitive gates.

(Refer Slide Time: 00:43)



Basic gates

- 2 or more input GATES
  - AND, OR, XOR
  - NAND, NOR, and XNOT
- One input GATE
  - Buffer and Invertor

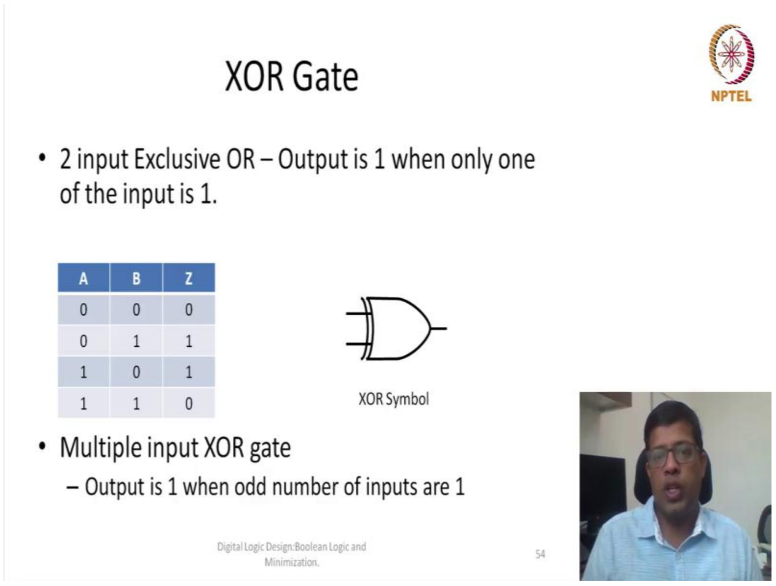
Digital Logic Design:Boolean Logic and Minimization.

53

So using this primitive or basic gates we construct all the building blocks or bigger digital logic. So, the number of gates can be enumerized as like AND, OR gate and there is also XOR gate or exclusive OR gate. Along with that 3 more combination which are inverter of these gates is called NAND gate, NOR gate and exclusive NOR gate. So, these are the gates the 6 gates AND gate, OR gate, XOR gate, NAND, NOR and XNOR. So, these 6 gates are possible with two or more inputs. Gates which are possible with the single input is inverter or a NOT gate.

Along with the inverter or NOT gate there is one more gate which is popularly used in some particular cases is called Buffer. So, Buffer does not do any logic so whatever is the input that is given as the output. So, it is just sometime it is required to increase the strength of the signal. So, we will not discuss this Buffer gate here. So, what are the new gates we are we have discussed we will be discussing today?

(Refer Slide Time: 02:07)



XOR Gate

- 2 input Exclusive OR – Output is 1 when only one of the input is 1.

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

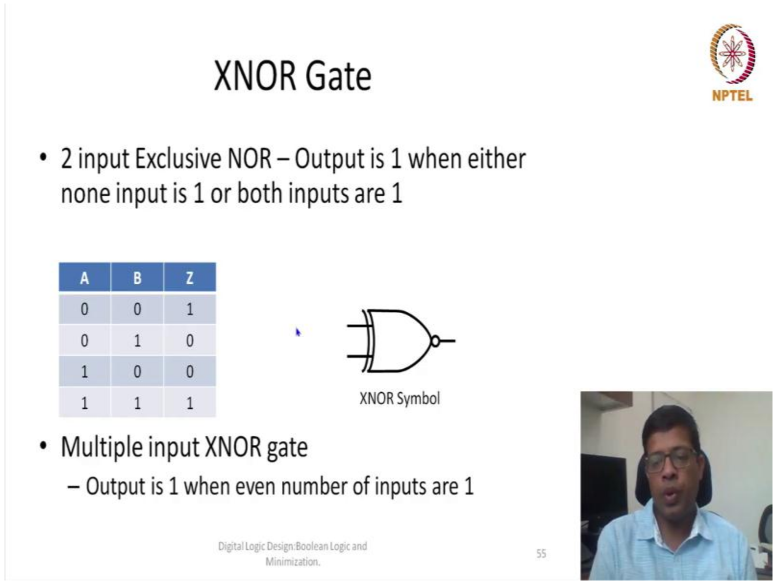
- Multiple input XOR gate
  - Output is 1 when odd number of inputs are 1

So, one of them is XOR gate. XOR is also called as exclusive OR gate. So, basic idea here is let us say there are 2 inputs if only one of the input is 1 then the output is 1. So, this is the basic behavior or basic functionality of an XOR gate. So, I want to write the truth table for this that means that only one of the output one of the input should be 1 then only output is 1. So, I can write it like when both the inputs are 0 the output is 0, when one of the input means B is 1, A is 0 then output is 1 or A is 1, B is 0 then output is 1.

If both A and B are 1 then they should be again 0. So this is the basic behavior of XOR gate. And the symbol for this gate is like you have a OR and along with a OR there is a one more line, one more line which is parallel to OR. This signifies the symbol of XOR gate. So, whenever we are writing this XOR we would write it like this. Algebraic symbol is that we write a plus and a circle over plus that symbol is in algebra wherever we are writing so that symbol we are using so along with plus there would be an circle over it.

So, this is how we define XOR gate. And if I want to have a XOR gate which have more than 2 inputs then the behavior or the functionality of XOR gate can be defined as the number of inputs should be odd. So, a number of 1s in the input should be odd. So, let us say if there are 3 inputs A, B and C if total number of 1s in any input patterned is either 1 or 3 then the output is 1 otherwise output would be 0. So, this is how we define XOR gate. Now, let us see the other one XNOR or exclusive NOR gate.

(Refer Slide Time: 04:21)



XNOR Gate

- 2 input Exclusive NOR – Output is 1 when either none input is 1 or both inputs are 1

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

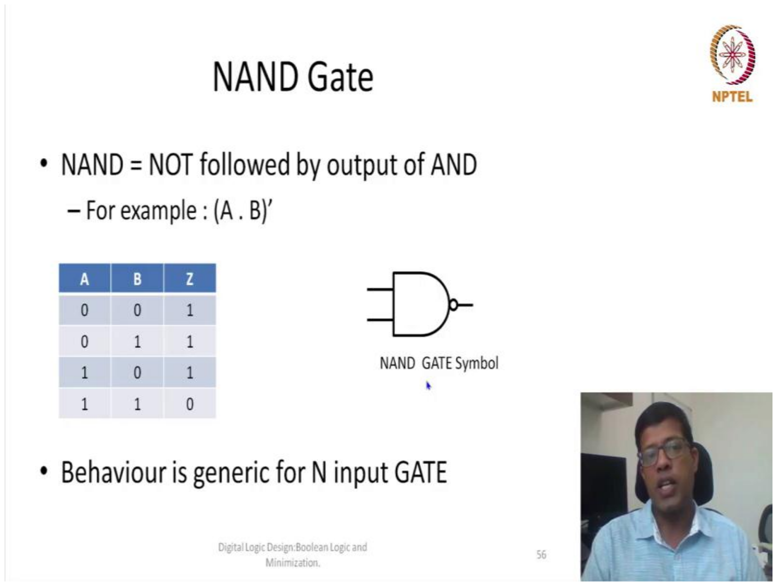
- Multiple input XNOR gate
  - Output is 1 when even number of inputs are 1

So, in exclusive NOR gate it is effectively the, if the output of XOR gate is inverted that would essentially mean XNOR gate. So, in in a mathematical term or in a new English language we can say that the output is only 1 if neither of the input is 1 or both the inputs are 1. So, the truth table would go like this. So, that means when both the inputs are 0 then output is 1, if both the inputs are 1 then also output is 1 otherwise the output is 0.

So, you can see the behavior or functionality or truth table output is exactly inverse of inverted the XOR gate output. So, that is why the symbol of the XNOR gate is also very similar to XOR gate but in inverter associated with the output. So, you remember that whenever we have this small circle at the output or at the input this represent the inverter or NOT gate functionality. So, the essentially NOR gate is the XOR gate, XNOR gate is the XOR gate plus an inverter in the output would be XNOR gate.

So, how can I define generically? So let us say if the XNOR gate will have more than 2 inputs then the output of XNOR gate would be 1 only if the number of inputs, number of inputs is number of 1s in the input is given. 0 is also considered even here. So, let us say this also include if this the generic definition also include 2 input XNOR gate, so if the number of 1s in any input pattern is 0, 2, 4, 6 then the output of XNOR gate would be 1. So, that means at any point of time if the number of inputs, number of 1s in the input is even then the output is 1. Otherwise, output is 0. So, these are the 2 new gates which we have introduced.

(Refer Slide Time: 06:45)



NAND Gate

- NAND = NOT followed by output of AND
  - For example : (A . B)'

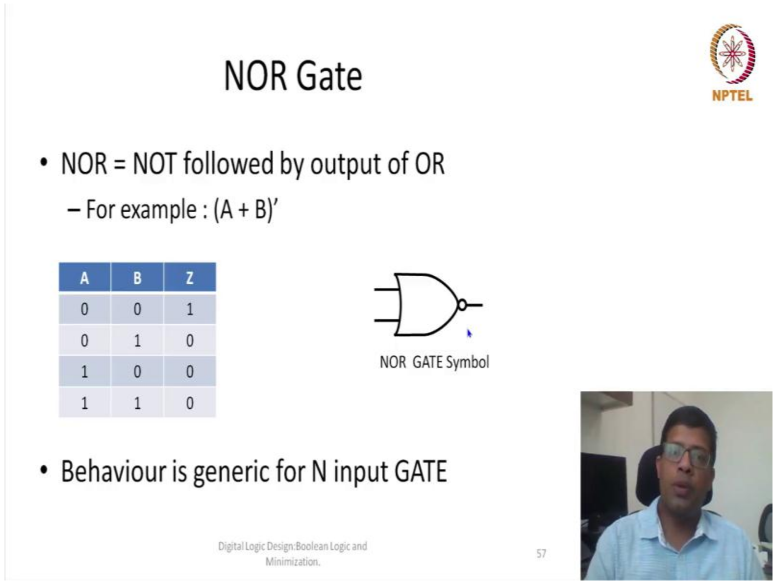
A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

- Behaviour is generic for N input GATE

The other gates are NAND gate. NAND gate means a NAND would be followed by the output of an AND gate. So, let us say if A and B are the 2 input variables they are ANDed, if their output is inverted then that would mean a NAND gate. So, that means the truth table would be whatever is the truth table of AND gate then inverting of that or basically wherever 0 is there it would be replaced by 1 that would be the output of the NAND gate. So, in a typical AND gate the output truth table is 0 ,0, 0, 1 and so the truth table of NAND gate would be 0 NAND 0 is 1, 0, 1 0 NAND 1 is 1, 1 NAND 0 is again 1 and NAND of 1 and 1 is 0.

The symbol would be used very similar to AND gate but at the output of AND gate there is small circle is there and circle signifies the inverted output. So, whatever is the output of my AND gate would be inverted that means my NAND gate. So, if inputs of NAND gate are more than 2 then also we can generically define. So, essentially when all the inputs are 1 then only output is 0 otherwise output is going to be 1. So, the behavior is generic only in the last row of my truth table would give output as 0 that means all the inputs are 1 then only output is 0 otherwise output is 1 in case of a NAND gate.

(Refer Slide Time: 08:29)



NOR Gate

- NOR = NOT followed by output of OR
  - For example : (A + B)'

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

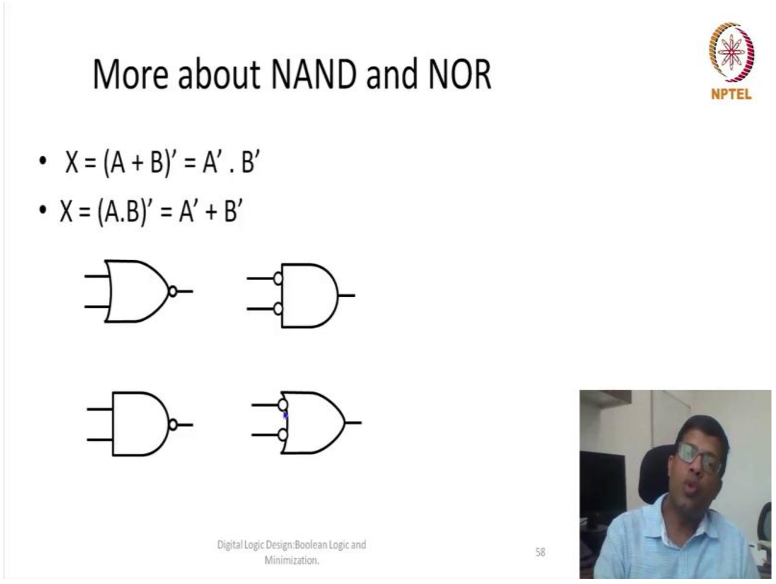
- Behaviour is generic for N input GATE

Digital Logic Design:Boolean Logic and Minimization. 57

Another similar gate is NOR gate. So, NOR gate a functionality is NOT followed by the output of OR gate. So we can also say it like that. This that a my A and B represent a OR gate if the output of both of them is inverted then it would be considered as NOR gate. Let us see the truth table of NOR gate. Truth table of NOR gate is exactly the opposite of whatever is the output of my OR gate. So, in OR gate whenever 0 and 0 is OR the output is 0, here the output is 1. In all other cases if any one of the input is 1 then the output is to be 1. In case of OR gate in the NOR gate when any of the input is 1 then output is 0.

So, generically we can say that if the input are more than 2, if the inputs are more than 2 if all the inputs are 0 then the output is 1 otherwise if at least one of the input is 1 then the output is 0. The symbol is there is a circle after a regular OR gate. So, this is how these few new gates are like the functionality of this new gates, new gates. So why we are studying about these new gates, we will see that they have certain interesting characteristics, they have some interesting properties and some certain interesting applications. So, let us quickly look at these characteristics and applications.

(Refer Slide Time: 10:12)



More about NAND and NOR

- $X = (A + B)' = A' . B'$
- $X = (A.B)' = A' + B'$

Digital Logic Design:Boolean Logic and Minimization.

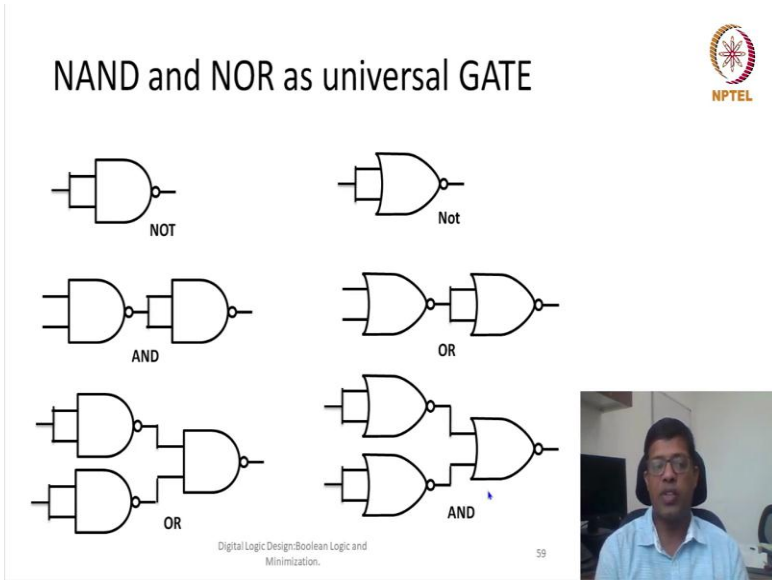
58

So let us see my this is my NOR gate so that means A OR B is inverted that means this represent the functionality of a NOR gate. Now, if I apply De Morgan's law what would happen that this dash would come inside like for each variable plus whatever is the AND this will become whatever is the OR that will become AND so, this expression applying De Morgan's law will become A dash means inverted A and inverted B.

So, this whole thing I can also write it like this that this NOR gate, NOR gate is essentially equivalent to an AND gate with both inputs inverted. So, this way this NAND gate, NOR gate is now related to a AND gate with my input inverted. So, similarly if I see a NAND gate, in a NAND gate A dot B the output is inverted. Now, if I apply De Morgan's law it become A dash plus B dash. So, symbolically these two are equivalent so that means a NAND gate is equivalent to an OR gate with the inverted input. So, this NAND and NOR gates now become a bridge between AND and OR gates.

So, essentially if I want to convert an AND gate to OR gate then I can use these bubbles or inverters to convert from AND gate to OR gate or NOR gate. So, similarly OR to NAND gate also it can be converted. So, this is, so this also give us some intuition that possibly this AND gates, NAND gates and NOR gates can also be can also be used to design an AND gate, OR gate or NOT gate. So, that is why these NAND and NOR gates are also called Universal gates.

(Refer Slide Time: 12:34)



NAND and NOR as universal GATE

What is the meaning of Universal? So, let us say we have only NAND gate which is there with us. So, using a single or using like only NAND type gates so NAND gates we can construct any number of digital logic. Whatever digital logic is possible so all of that can be designed, it can be constructed using only NAND gates or only NOR gates. So, this looks little confusing initially when we started our course we said that the Boolean algebra would be operated upon by 3 basic operators, that means OR, AND and NOT.

So, now we are saying that only one operator is sufficient that is NAND so why it is possible? Because NAND can essentially be used to represent NOT gate as well as OR gate as well as AND gate. Why? Because of the property which we have discussed in the previous slide. So, let us say first of all if I tie both the inputs if my NAND gate then if we see the truth table it would behave like a NOT gate. Similarly, so this is the basic property. So, now you see if you remember if my AND gate if both the inputs are tied for the AND gate it simply X dot X is equals to XOR so basically it does not carry any meaning if both the inputs are tied.

But now because of this inverter, because of this inverted output it would act like a inverter or a NOT gate. So, now because I can create an inverter out of my AND gate so I can create a AND gate also out of a NAND gate. So, NAND gate is essentially a inverted output of my AND gate so if I put another inverter after it so it would essentially means to me like a AND gate. So, although I required 2 NAND gates to represent one AND gate but it is possible so we can construct an AND gate out of NAND gates.

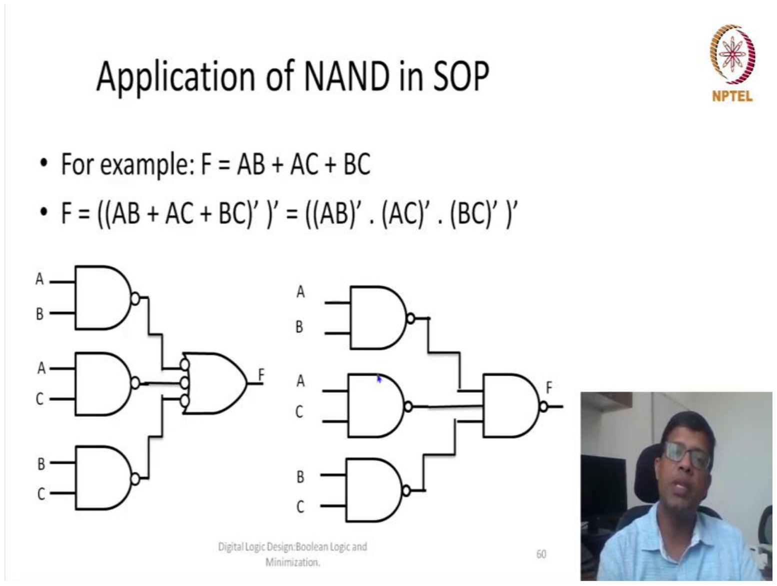
So, like in our previous slide we have seen a relation between OR and NAND gates so using the same principle we can also construct an OR gate out of three NAND gates. So, what we have seen that if we take the inverted inputs, if we make the input inverted for a NAND gate then it would essentially behave like an OR gate. So, very similarly the similar things would have happened in the NOR gate. So, in NOR gate also like you know A plus A is equal to A so that means this also does not have any value but because of this inverted output and if I tie both the inputs of my NOR gate it will also effectively work like a NOT gate.

So, similarly if I take 2 NOR gates and output of one of the NOR gate it is connected to both the inputs of my NOR gate so that means it would effectively work like a inverter and 2 inverters nullify each other output is effectively like a OR gate. Because of the relation between OR and AND, so I can invert both the inputs of my NOR gate and effectively it will work a NAND gate. So, this way these NAND and NOR gates act like a Universal gates and can be used almost to construct any Boolean logic or any digital logic. So, does it have any application?

So, yes. Initially when people have figured out that these NAND gates can become like or can be used like a Universal gate. So, what they did they created a some sort of a programmable hardware by designing a sea of NAND gates so sea of NAND gate means they created a chip which have thousands of NAND gates because fabrication of chip was always a costlier affair and complicated affair.

So, once they have fabricated these NAND gates then they use these NAND gates to construct any sort of design whatever the they wanted to configure. So, that is more like a significance or why we should see that if we have only one Universal gate how we can use that effectively to create any sort of other logic or other gates, so this is the one application.

(Refer Slide Time: 18:20)



Application of NAND in SOP

- For example: F = AB + AC + BC
- F = ((AB + AC + BC)')' = ((AB)' . (AC)' . (BC)')'

The other thing we can see is that the as a, we have seen that the all the Boolean equations we can write in a standard form which is sum of product or product of sum. So, these sum of product or product of sum can also be implemented using a NAND gate or NOR gate respectively. So, let us say let us consider this sum of product so I am taking an example let us say sum of product let us say F is equal to AB plus AC plus BC. So, all of these three are product terms and we can now we are adding these product terms.

So, do not see the NAND feature here, so we do not see a inverted output etcetera. So, how NAND gate can be utilized? So for that what we can do is we can take F double dash. So, this is how it would be otherwise implemented so you will have a 3 AND gates where A and B would be the input and the AB, AC, BC would be the input and the output of all of them would be given to a OR gate the output we can see as the F. So, now if I want to use a NAND gate what I will do is I will do a double F double dash.

So, that means I will take the inverter of, inverted output of AB plus AC plus BC and then whatever is the output I will take another inverted output. So, these 2 dash, 2 like F double dash is equal to F so that is how this two should be equivalent. Now, again if I apply De Morgan's law so which means that all the terms should be inverted and plus should be converted into dot. So, effectively what I am saying is AB dash is the inverted output of A AND B and inverted output of A AND C, inverted output of B AND C and then I will product all of them and then we will take another inverter, inverted output.

So, that would be equivalent to F. So, essentially what I am doing is I am creating all these NAND gates this is AB dash this is A and C inverted output of product of A and C and inverted output of AND of B and C. Now, all of these three things are ANDed and final output is also inverted, so this is also a NAND gate. So, essentially like similar to this any sum of product expression can be represented as, can be represented as AND, NAND, NAND implementation of can be can be implemented using a NAND, NAND gates.

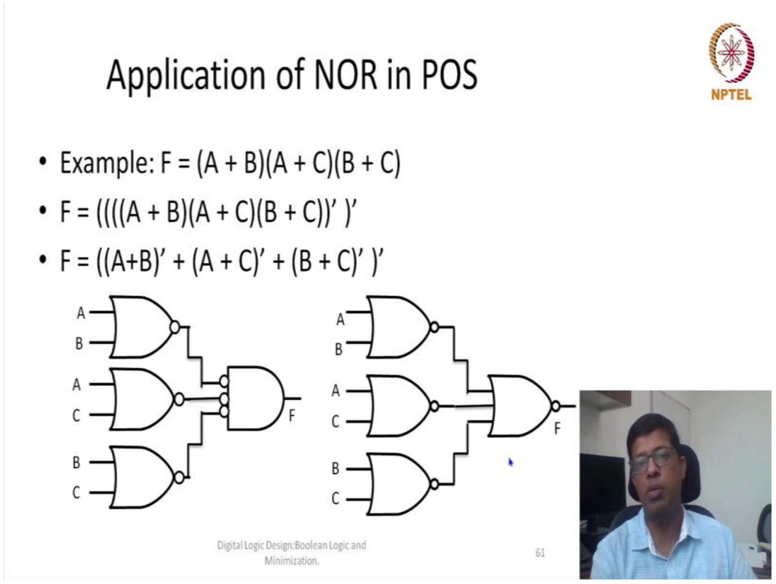
So, yes correct. Here you have only the un-inverted inputs what if the inputs are not un-inverted but there are some inverted inputs also, let us say AB dash or AC dash or A dash, B dash. So, here what we assume that the inputs literals, so, as our previous definition was the cost of we assume that the input literals are always available in both original form as well as in inverted form. So, given this assumption this simplification. Then we will always assume that let us say A dash is also available B dash is also available, so they can all be connected to our NAND gates and the output that means that we can see them as a F dash F.

Another way of looking at it is let us say if I if I put 2 inverters or 2 inverters in each of this intermediate output. So, if I put two inverters in this intermediate output then let us see how does it look like. So, let us say I have these 2 inverters for this wire, these two inverters for this wire, these two inverters for this wire. So, each of the wire has two inverters so effectively two inverters would act like a Buffer, so that means my whatever is the input would be the output.

However the functionality changes, because of this, this inverter this AND become NAND and all these ANDs become NAND. And because of these inverters here, now this OR gate with the inverted input is essentially equivalent to a NAND gate. So, this is how we can see this sum of product expression and its implementation using NAND gate. So, here also the same logic applies, so the idea, the basic idea of implementing all the sum of product expressions using NAND gate is that we can have a generic implementation.

So, that is the reason that we can go for always sum of the product expressions, so that if we have a configurable chip there we have two arrays of NAND gates. Then this two arrays of NAND gates can be utilized to design any kind of a configurable circuit. So, we will discuss about this details that how that circuit would be designed etcetera., would be sometime later in some other modules.

(Refer Slide Time: 23:48)



Application of NOR in POS

- Example: $F = (A + B)(A + C)(B + C)$
- $F = ((((A + B)(A + C)(B + C))')'$
- $F = ((A+B)' + (A + C)' + (B + C)')'$

Similarly, these NOR gates can be used for implementation of product of sum expression. So, again let us take a very similar example so, let us say F is a product of some expression. The 3 sum terms A plus B, A plus C and B plus C and all the three sum terms are in product. So, that means they are all ANDed. If we see the regular implementation, the regular implementation goes like this that you will have 3 OR gates and the output of these 3 OR gates would be ANDed to get the output F.

Now, similar to our NAND gate implementation, so we again double invert this F function F double dash. So, that means whatever this expression is A plus B, A plus C, B plus C this whole is inverted and the output is again inverted. So, because of De Morgan's law this first inverter can be converted into A plus B dash plus now this product is converted into plus AND is converted into OR, so, essentially A plus C dash plus B plus C dash and whatever is the output we will again invert it.

So I can I can put it like this A plus B dash is essentially a NOR of A plus B. So, the output of OR is inverted that means this expression similarly A plus C OR is inverted this expression and OR of B plus C is inverted that means this expression. And OR of all these 3 outputs are again ORed and output is inverted so these 2, these 2 implementations are essentially equivalent. The first implementation is using OR gate and AND gate, while the second implementation is using only NOR gates. So, similar to NAND gate this can also be understood like if we make these wires, if we put two inverters here, then the implementation is similar.

So let us say we put these inverters so this output essentially should not change because of these 2 inverters and similarly this output should not change because of 2 inverters and this output should not change because of this inverter. So now, because of these inverters this has become a NOR gate, this is also become a NOR gate and AND gate with a inverted inputs is called a NOR gate. So, this is how these NOR gates can be used for effective implementation of product of sum expressions and NAND gate can be used for effective implementation of product of sum, sum of products.

So, in summary of in this part of lecture we have seen that few new logic gates we have seen. We have seen XOR gate, we have seen XNOR gate and we also have seen the functionality of the NAND and NOR gates. A few interesting characteristics about NOR and NAND gate that how they can be converted into a sort of AND gate and OR gate and how they works like a Universal gates and how they can be used for effective implementation of product of sum or sum of product expressions.