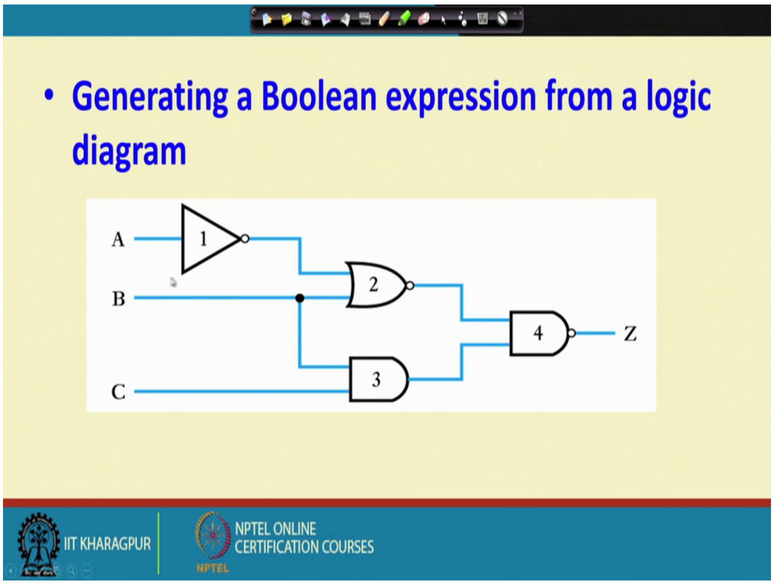
Digital Circuits
Prof. Santanu Chattopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

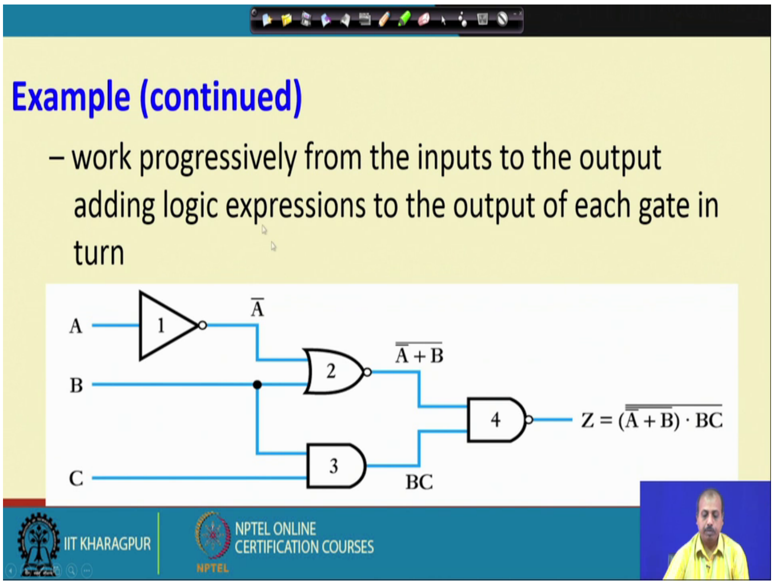
Lecture - 15
Logic Gates (Contd.)

So, for generating the Boolean expression from a logic diagram so, we have to proceed from the primary input site and go towards the primary output.

(Refer Slide Time: 00:18)



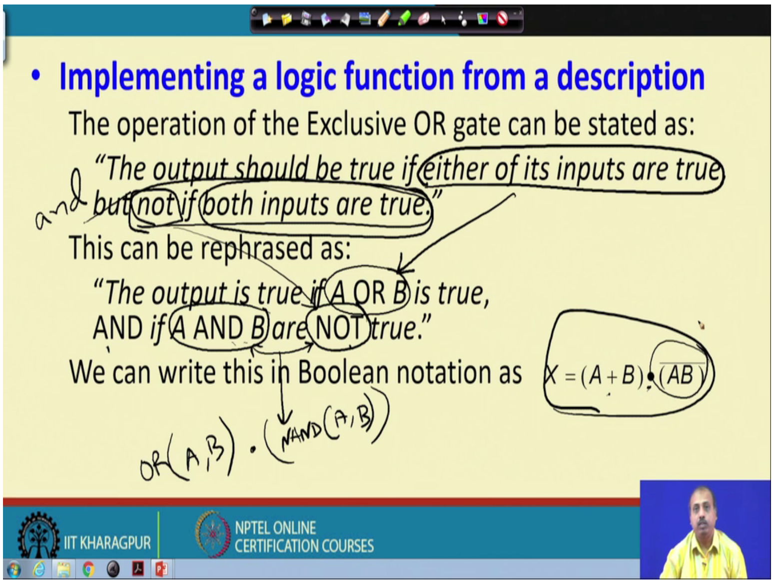
(Refer Slide Time: 00:22)



So, this is the procedure that work progressively from the inputs to the output adding logic expressions to the output of each gate in turn. So, this is my input A B C now, I find that at the first level there is an inverter here. So, inverter will transform A to A bar; at the next level if you see there are 2 gates gate number 2 and gate number 2 where gate number 2 is a NOR gate. So, in the NOR gate one input is A bar another input is B so, at this point I am getting the expression A bar NOR B ok. So, A or A bar or B whole bar whatever way you read it and then this gate number 3 it has got B and C add has two inputs and this is an AND operation. So, you get a BC as the output.

Then finally, at gate number 4 this is a NAND gate so, you gate the NAND of these two inputs. So, A bar plus B whole bar and BC so, you get the NAND of them so, Z equal to this expressions. So, of course this can be simplified, but it is not done here. So, you can break down this expression and see what it is it is turning out to be, but it can be see that later.

(Refer Slide Time: 01:33)



Now, sometimes it is necessary that we are given some logical statement and from there we need to get the corresponding Boolean circuitry. So, this can be done like this a typical example has been taken here. So, this exclusive OR gate so, we can write it like this that output should be true if either of its inputs are true, but not if both inputs are true; as you know that is the functionality of the XOR gate. So, its output is a one only when only one of the input is equal to 1.

So, you can say we can say the way we have put this statement so, it is true if either of its inputs are true. So, this is the first part of the statement so, either of its either of its inputs are true. So, this is the first part of the statement, but so, but we can did it like this; so if I replace this, but by and so, we can say and not if both inputs are true ok.

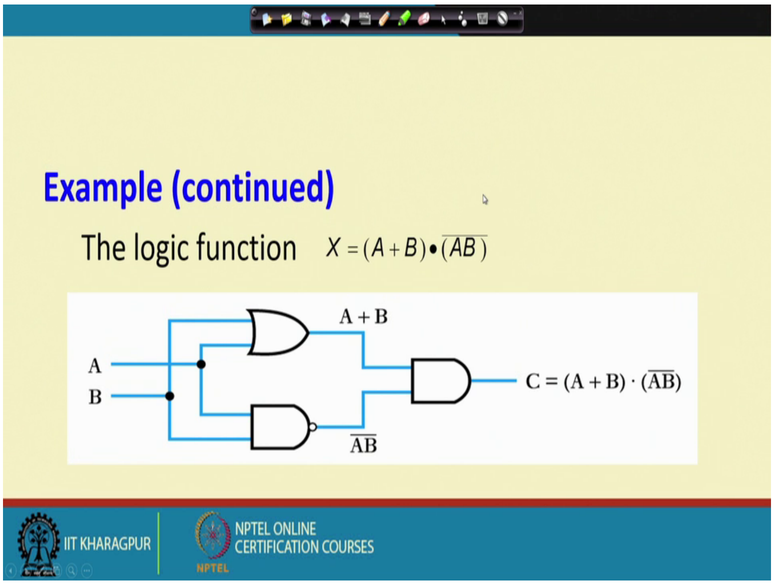
So, so this can be represented like this so you now just put the statement like this that this either of its inputs are true; so this boils down to A or B ok. So, this A this either of its inputs are true so, this boils down to A or B is true and not if both inputs are true. So, so, both inputs are true so, this part so, this part is basically A and B ok. So, this is A and B and this not so, this NOT is the it is not true so, this NOT.

So, this has to be say A or B should be there and A and B NOT should be there so, A and N NOT is basically the NAND. So, I can say this if I if I join this two I gate a NAND. So, from this I get a NAND of I get a NAND of A and B I get a NAND of A and B. So, that should be there and then it should be ended with it should be ended with A or B. So, this is OR of AB. So, this is the whole expression that I am expecting. So, you can write it like this so, here we have what I have written. So, this is first part is A or B then this AND so, AND is this and now I have got A and B NOT so, A and B NOT is basically this one. So, this is the NAND gate A and B NOT.

So, this way we can from Boolean expression Boolean from logic statements so, we can try to come to the corresponding Boolean expression so that may be done. And if it is difficult, we can try to draw the corresponding truth table from the Boolean from the logic expression from the logic statement. And from there we try to come to the Boolean expression.

So, we will see those techniques slowly has you proceed through the course.

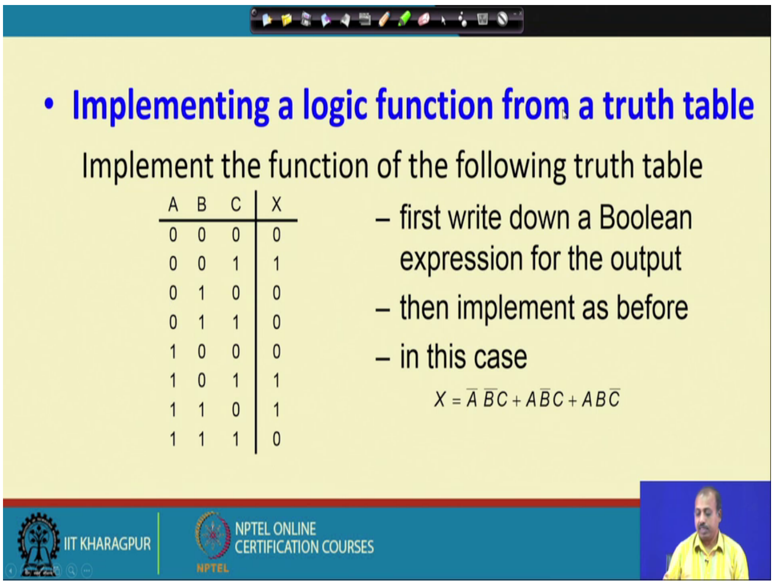
(Refer Slide Time: 04:48)



So, now once we have got the logic function so, we can representative in the form of gates like say so, this first of all I need and A or B. So, this OR gate it has got A and B has input so I gate A or B then I need a NAND gate. So, to get AB bar so, this is the NAND gate getting AB bar from A and B and then finally, these to are to be anded so, this is anded. So, you get C equal to A plus B dot AB bar.

So, this way from the logic statements, so you start with the logics logic statement and from there we finally come to the corresponding logic circuit. So, that can be done.

(Refer Slide Time: 05:33)

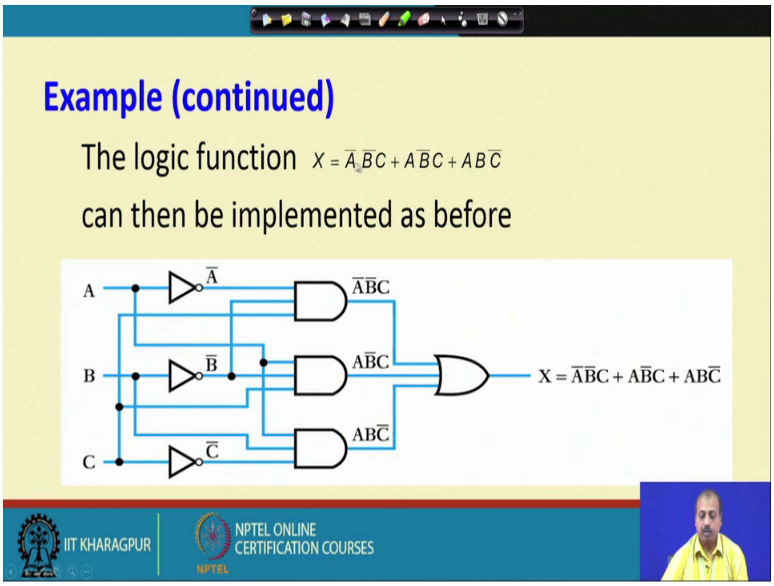


So, as I was telling that sometimes it is easier to get the logic of logic function through some truth table. So, so we start with the logic statement from there we try to draw the truth table and then the truth table gives us the functionality and then we try to implement it.

So, suppose we have got some logical statement and from where we get this as the truth table that is whenever the C being equal to 1 this is they so, I we write down all possible combinations of values combination of values of A B and C. So, 0 0 0 to 1 1 1 out of that we see that in these three cases the output is equal to 1 ok. So, when this A A bar B bar C A and B and 0 and C is 1 then output is 1 then this A B bar C in that case it is equal to 1 and A B C bar this case this is equal to 1.

So, if you are trying to get the corresponding logic circuit then what we should do from the truth table we should write down the Boolean expression first and then we try to we realize this Boolean expression by means of logic gate. So, corresponding to this we can say that X equal to A bar B bar C plus AB bar C plus ABC bar of course, I have not done any simplification of the Boolean expression here that may be done, but I am not going into that I will come to that later.

(Refer Slide Time: 07:03)

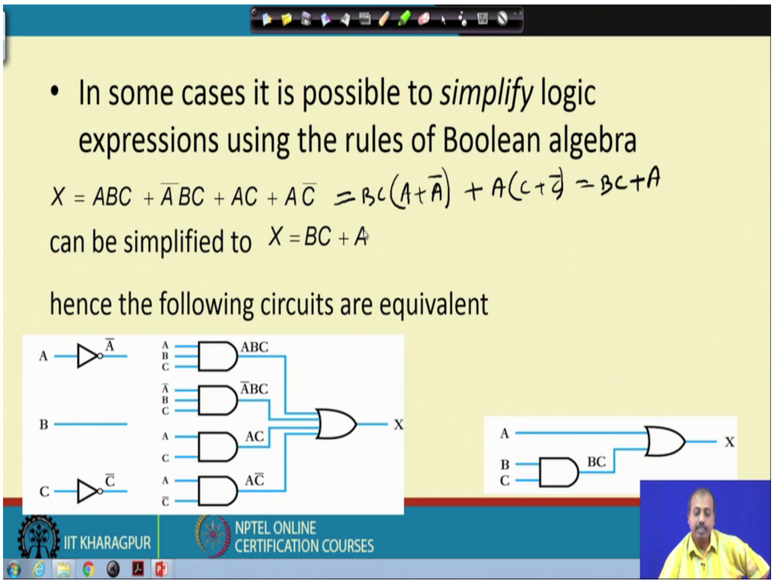


So, once we have got this Boolean expression now the process is straight forward. So, first of all at the top level I will need an OR gate and since there are three such terms in this in sum of product expression. So, I will need a three input OR gate so, these OR gate is three input OR gate. So, it will be the first input should be corresponding to A bar B bar C, second input to AB bar C and third input to ABC bar. Now, how to get this A bar B bar C? For getting A bar B bar C I need a three input AND gate and where A bar should be the first input, B bar should be the second input and C should be the third input. And for getting A bar so, we compliment A so, getting A bar here similarly, we compliment B we get B bar here. So, this A bar B bar and C there connected to the first gate first AND gate.

Similarly, second and gate I will realize AB bar C so, this A is connected this B inverted output B bar is connected and this C is connected. So, so, this way we get AB bar C and the third gate third gate, third AND gate is AB C bar. So, the similarly we get ABC bar and finally, when they are odd you get this Boolean expression fine.

So, in this way so, you can start with the truth table and from the truth table you can realize the corresponding function.

(Refer Slide Time: 08:25)



So, it may be possible to simplify the logic at that we have seen in our discussion in on Boolean algebra that some time it is possible to simplify this sum of product expression or product of sum expression, to have lesser number of literals and lesser number of operands. Like it may be that suppose we have got a Boolean expression like this so, X equal to ABC plus A A bar BC plus AC plus AC bar.

Now, if you try to realize it directly then what will happen is that you will need a three input or sorry four input OR gate like we have it here and then it should this four inputs correspond to this sum this product ABC A bar BC and AC and AC bar. Now for so, you since your needing both A and A bar so, you will be needing and inversion of A to get A bar.

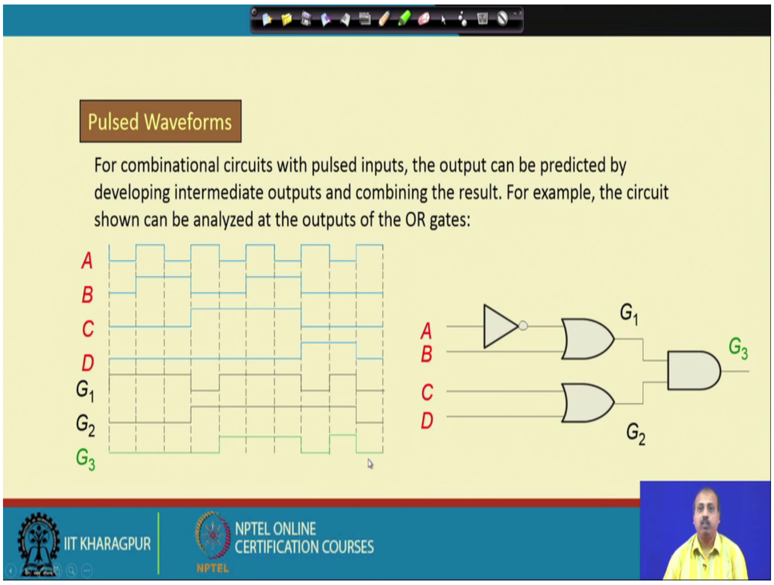
So, this line A bar line can be connect should be connected to this point whereas, the points where your getting A directly say can be can taken from this point and connected to gate. So, you can the so, this for the sake of simplicity the connections are not shown, but what we mean is this A this line is connected here. Similarly, this line is connected to this A, then this A ok, then this A it is connected to all this.

So, it is not shown a explicitly because that will complicated the diagram so, it is they are not shown. So, but this whole expression can be simplify to this BC plus A so, you can just do this simplification. So, you can just from the first two terms so, you can take BC common and this is A plus A bar. Similarly from the next two terms you can take A common so, it is C plus C bar. So, this BC A A plus A bar is equal to 1 and C plus C bar equal to 1 so, this simplified form is BC plus A.

So, once we have done this simplification so, you can realize the circuit with much less number of gates. So, you see that I needed two input OR gate for or in this A and BC and a two input AND gate for getting this product BC from the input B and C. So, that way this Boolean expressions can be simplified and then the simplified Boolean expression may have much less number of logic gate. So, that is one of the motivation why you should do this logic minimization because in terms of logic gates so, it will require less number of gates and less number of connections.

And complexity of the circuit becomes less and naturally the since the number of gates needed is less. So, the area then power requirement everything will go down for the resulting circuit.

(Refer Slide Time: 11:17)



So, next we will be looking into another important concept, like many times what happens is that if say this is suppose this is an example circuit where we have got a few gates G 1 G 2 G 3 and then inverter and is A B C D these are the inputs. So, many a times what happens is we this A B C D so, they do not have fixed value over the lifetime ok. So, they have got different values. So, we just draw some sort timing diagram so, they are also known has pulsed waveform.

So, this we just see try to see when this outputs inputs are changing like initially A was equal to 1, at time 0 it has become equal to 0. It continue still this time then it becomes 1 and it continue to be high for this much time, then it again comes down to 0, then it goes like this. So, this way suppose this is the wave form for A that is every alternate time is 10 so, it just toggles its state. Similarly, suppose the input B so, it was initially 0 initially low then here it goes 1 so, it remains heifer this much time then it goes low, then again it remains low for this time.

So, it toggles every 2 time units you can say so, it is toggling like that. Similarly, C is toggling every 3 times 4 times units so, this is this was initially 0, it becomes 1 here for 4 times units it remain equal 1 then it comes down and it goes like this. Similarly, D is on D is 0 for this much time then it goes high and remains high for 2 times units then comes down and then goes to 0. So, these are the inputs A B C D these are the inputs so, these way forms are given to us.

So, based down that we try to predict what will be the output of the circuit at different time instants. So, or to so, do that what do we do is we first see how G 1 will look like. So, for getting G 1 so, you see whenever A is equal to 0 so, A bar so, this is A bar this line is A bar so, this is equal to 1. So, or B so, you can understand since or of these two whenever B equal to 1 or A equal to 0; the G 1 will be equal to 1.

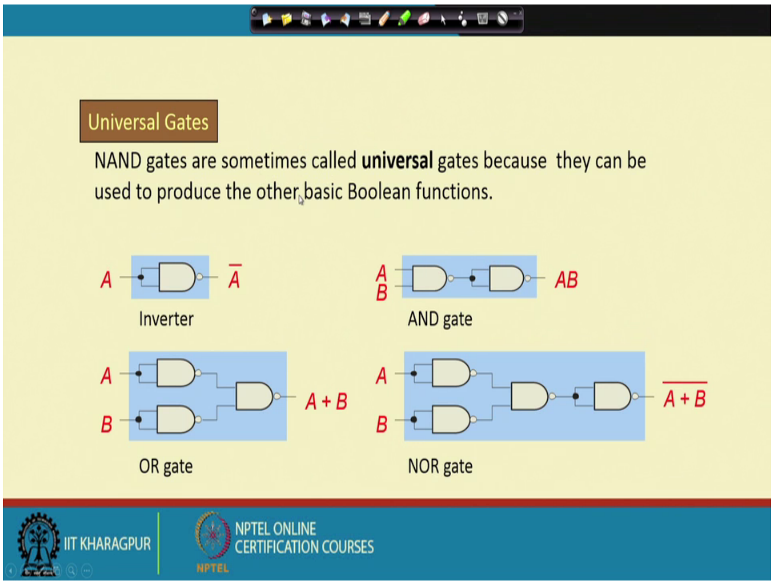
So, you see that whenever B is equal to 1 so, this G 1 equal to 1. Similarly, whenever B equal to 1 G 1 is equal to 1 and also whenever A is equal to 0 G 1 is equal to 1. So, A equal to 0 G 1 equal to 1 A equal to 0 G 1 equal to 1; similarly A equal to 0 G 1 equal to 1. So, whenever A is equal to 0 like here A equal to 0 G 1 equal to 1; so in this way from this logic circuits so, we can draw the diagram for G 1.

Similarly, for G 2 so, G 2 is basically or of C and D, so whenever C or D so, either of then is equal to 1 G 2 output will be equal to 1, otherwise it is 0. So, C is 1 for this much time so, accordingly G 2 is 1 for this much time and then after that say D has become 1. So, from this point onwards so, this is equal to again 1, but till D is equal to 1 and then at in this part so, both C and D are equal to 0. So, it comes down to 0.

Similarly, at the beginning both C and D are equal to 0 so, G 2 remains at 0. So, this way we can find out the status for G 2; how it proceeds with time and finally, the G 3 G 3 is the end of G 1 G 1 and G 2. So, whenever G 1 G 2 both are equal to 1 G 3 will be equal to 1 so, G 1 G 2 both are equal to 1 for this much time. So, this G 3 is equal to 1 again for at this time so, G 1 and G 2 both are equal to 1. So, this is equal to 1 otherwise it remains at 0.

So, this way very often so, will be drawing this pulse wave forms for combinational circuits. So, given the input so we will proceed step by step drawing this pulse is pulsed output for it of the gates. And finally, will reach the primary output and there we get the overall output for the system.

(Refer Slide Time: 15:25)



Next we will be discussing something called universal gate. So, as the name suggest so, universal means you can do you can realize anything using those gates ok, you can realize any digital circuits using those gates. So, so, like NAND gates is one of the one such example that can be used for realizing any circuitry and that is why it is called universal gate. So, they can be used to produce other basic Boolean function.

So, basics Boolean functions are if you look into this sum of product or product of some expression. So, the basic functions are AND OR and invert. So, if I have got an gate by which we can generate these three basic functions AND OR and INVERT so, that gate can be you used to realize any other function. So, that is why so, we will be calling this AND OR NOT as the basic gates and the basic Boolean functions.

So, this NAND gate can be used for do a realizing this basic Boolean function. How?. So, if you want get an inverter so, what you can do take a two input NAND gate and then just short both the inputs and connect the input A the inverter the corresponding so, we are interested to get this inverter. So, our objective is to get an inverter like this. So, this A will be input and A bar will be output ok.

So, what you do is that you take a NAND gate and then short the two inputs and then apply A at both the or you can say that you apply A at both the inputs so, and you get the A bar has the output. So, you can realize inverter using NAND gate then you can you can also you can also realize this and gate by using NAND gate, because for the realizing AND gate first of all you do a NAND of A and B.

So, here the objective is to get the AND function so, I will have this AND gate and this A and B are the two inputs ok. So, A and B are two inputs and you just do this; first of all you do a NAND so, you get AB bar here and after that use another NAND gate as inverter so, you get AB. So, you can get this AND function from the NAND. And for getting OR gate what we do is that we this OR gate this is slightly tricky. So, we use this De Morgan's theorem actually. So, this A or B it can be written has A or B whole double bar.

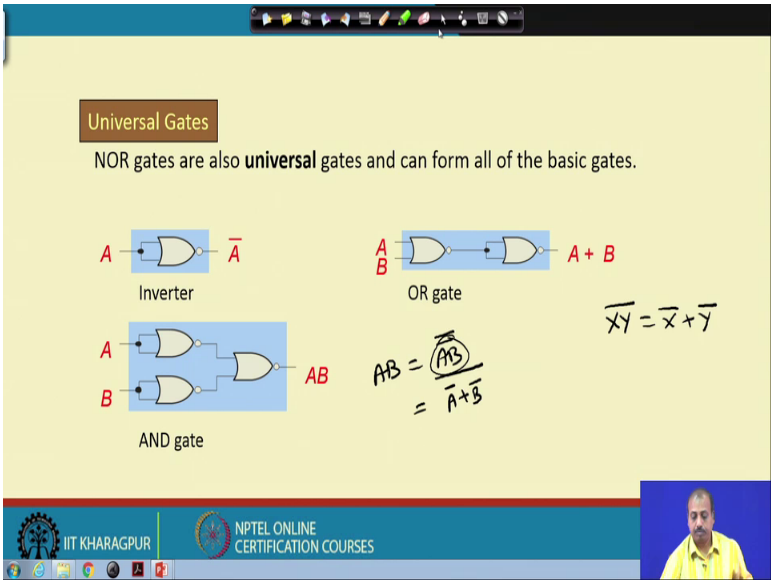
So, that way I am taking to a compliment twice so, has a result this will be complimenting this this will be giving me back this A plus B. Now, if I just apply De Morgan's law on this part ok, if I apply De Morgan's law on this part so, this will give me A bar dot B bar and this whole bar remains as it is. So, what is happening is that so, this is nothing, but NAND of A bar and B bar. So, you take the NAND of A bar and B bar and for getting A bar from A, you use the one NAND gate has an inverter and for getting B bar from B is another NAND gate has an inverter.

So, that way this whole circuit gives us A plus B ok. So, using only NAND gate of course, number of NAND gates needed is more, but that that is not a problem. So, if a if a assuming that I have got an infinite supply of NAND gates so, I can realize any function by means of this NAND gates only. So, I do not need any other gate for realizing the functions.

Now, similarly you can also have this you can also realize this NOR gate using this similar technique. Like say so, the up to this much this is this is OR gate and if you need a NOR gate so, you can just put another NAND gate is as an inverter at the end and that will give you the NOR gate. So, this NOR gate can also be obtained from NAND gate. So, all the gates can be obtained.

Similarly, so, we have got NOR gate as another universal gate. So, apart from NAND gate we have got NOR gate as another universal gate. How?

(Refer Slide Time: 19:56)

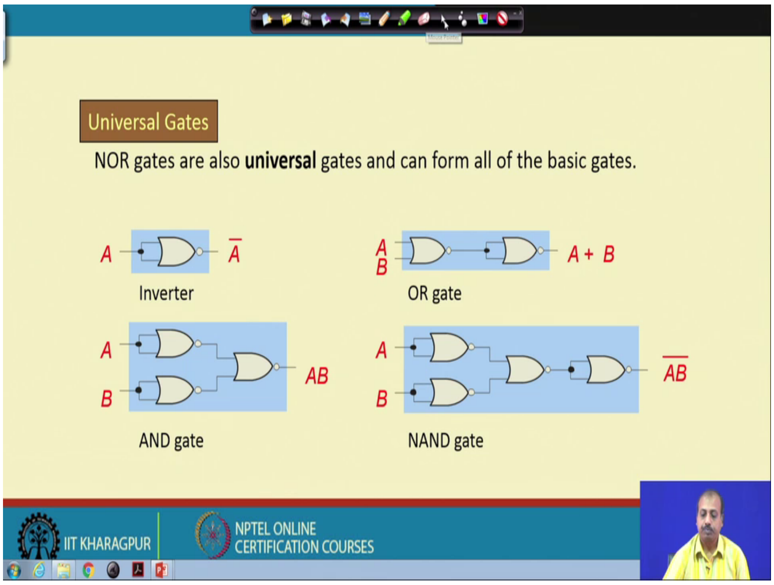


So, here also the same thing like if your if your trying to get this inverter A bar from A. So, what you do apply the same input A to both the inputs of the NOR gate so, you get A bar as the inverter. If you are trying to get a OR gate, it is quiet simple first of all you do a NOR of A and B so, you get A nor B and then use another NOR gate has an inverter. So, you get A OR B.

If you are trying to get an AND gate so, then it is straightly tricky. So, you have to apply that De Morgan's theorem. So, basically this AB can be written has AB double bar. So, this is A bar or B bar whole bar fine. De Morgan's law was this XY bar equal to X bar plus Y bar so, I am just using that formula. So, this AB bar and so, this AB bar I am writing as A bar plus B bar; the upper bar remains as it is.

So, what you have is that so, here you have got that this is the A bar or this is the A bar or B bar that an NOR gate A bar NOR B bar, that NOR operation and for getting A bar from A. So, use another nor gate as an inverter and for getting B bar from B use another NOR gate as inverter. So, that way we can do this thing.

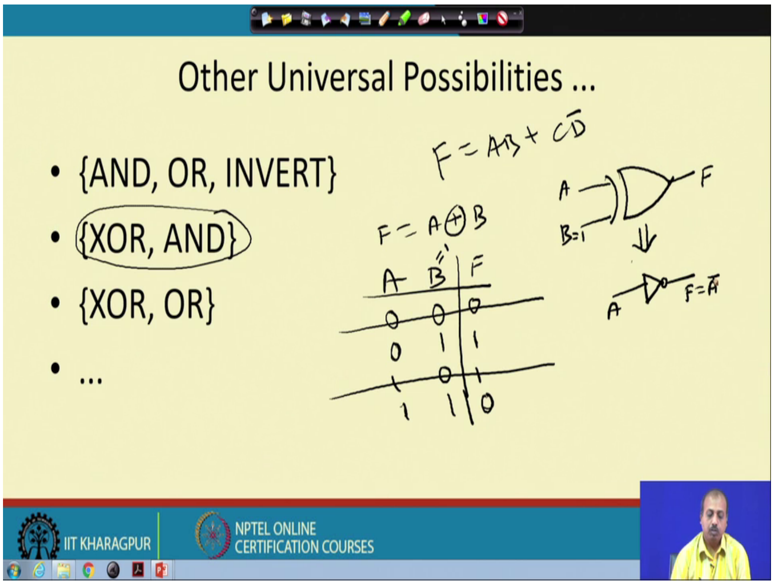
(Refer Slide Time: 21:28)



So, if you are trying to get a NAND gate then it is a simple so you get this AND gate up to this much and after that you put an inverter ok. So, another gate has an inverter so, you get the AB bar.

So, this way we can have this universal NOR gate it can realize any other gates ok, AND OR NOT. Actually, this NAND is not necessary because if you have to prove that a particular gate can act has universal gate, what you need to show is that it can be realize an inverter, it can realize the OR function, it can realize the AND functions. So, these are the three things to be shown so, this is shown has an extra ok. So, this is not mandatory to show.

(Refer Slide Time: 22:15)



So, some other combinations; so AND, OR, INVERT so, other universal gate combination. So, apart from this NAND and NOR so, we do not have any other gate which alone can act has a universal gate, but if you take a combination then it may be possible. Like say this one say AND, OR, INVERT is definitely AND OR INVERTER is definitely one possibility because this here it is directs. So, AND gate can be realize for realizing AND part, OR gate for the OR part and the INVERTER part for the NOT operation.

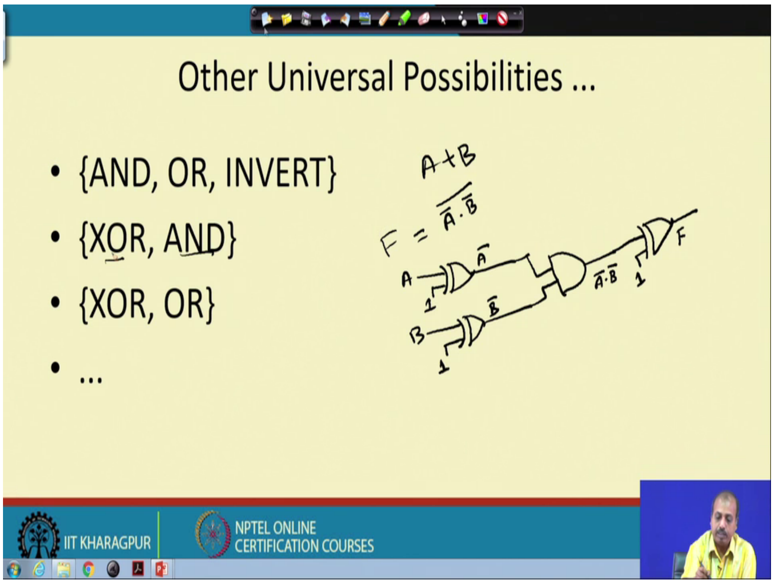
So, any from any Boolean function that you write say F equal to sorry say F equal to say AB plus so, AB plus CD bar something like that. So, we can very easily realizing using AND OR INVERT gate that we have already seen and interesting combination is this XOR and AND ok.

So, what is the this XOR so, you see suppose I have got XOR gate. So, F equal to A XOR B. So, what is the truth table for this so, this AB so, when both are 0 0 1 1 0 and 1 1. So, this is equal to 0 and this is equal to 0 and these two are equal to 1 fine. Now, you see that if I make this B input forcefully equal to 1 so, this B equal to 0 combination do not come. So, B is made forcefully equal to 1 then what happens is that if you give A equal to 0, you get A 1 at the output, if you give 1 as an input you give 0 as the output.

So, this XOR gate which is like this AB as two inputs and F has the output. Now, if I said this B to be equal to 1 then this is nothing, but I have got has if this is an INVERTER.

So, this when A is equal to 1 it behaves as an INVERTER, as if I have given A here and I have got F which is equal to A bar. So, in XOR gate so, if you tie one of the inputs to high so, you get an INVERTER. So, from the XOR gate I can get INVERTER.

(Refer Slide Time: 24:46)

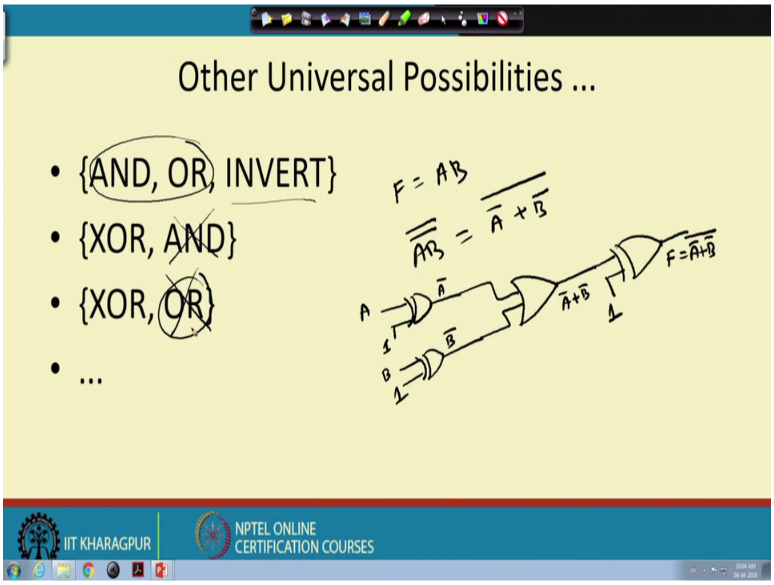


So, now if I if I have got an infinite supply of XOR and AND gate so, I can realize OR gate also for example, so, so, I have realize say A or B ok. For getting A or B so, I can I can write it like this so, A bar dot B bar whole bar. Now, I can for getting this A bar I can use one XOR gate ok. So, I can use an XOR gate with one of the input equal to 1. So, A is applied here so, this is your A bar and similarly I take another XOR gate with one input as B, other input tied to 1.

So, this gives me B bar and once I have got this A bar and B bar so, I can take an AND of these two I take AND of these two. So, I gate here A bar dot B bar and after that I have to take an I can take an inversion. So, I can take to another XOR gate ok. So, I can do it like this with one of the other input of the XOR gate tied to 1 ok. So, here you get the F. So, you see ultimately you are getting F equal to this one A bar B bar whole bar and that is nothing, but A or B. So, this way if I am given an infinite supply of XOR gate and AND gate I can realize OR function also and once. So, I have shown you the realization of OR; AND is already there in the set and from XOR I can get the inverter also. So, this AND, OR, INVERTER all the three basic gets can be obtain from this set XOR and AND. So, naturally XOR AND is an universal set.

Similarly, this XOR OR so, this is also an universal set because in this case OR is already there. So, what I what is not there is the AND gates. So, F equal to AB ok.

(Refer Slide Time: 26:40)



So, AB if you want realize so, you can take two bars and then you can simplify it has A bar plus B bar whole bar. And now, the same strategy I will follow I will take one XOR gate to get the INVERT of A ok, this is tied 1. So, you get A bar here and then you take another XOR gate so, this is your B and 1. So, you get B bar at this point and then you take OR of these two.

Since I have got an infinite supply of OR so, I can I can use one OR gate here. So, these gives me A bar OR B bar and then I do another inversion by connecting it to another XOR gate with the other input tied to 1; other input tied to 1. So, here you get F equal to A bar or B bar whole bar fine. So, this way we can realize this XOR OR it is also an universal combination. So, you can find any many other universal combination so, which does not include NAND and NOR gate ok. But the fundament thing like if you are given a only AND OR. So, only these two then it is not an universal set because you cannot do inversion.

Similarly, in the XOR family, so with XOR if AND is not given, OR neither of AND OR OR neither of them are given, then only XOR will not act has an universal set, because in that case you cannot realize all possible basic gates from there.

So whenever we have got a combination of gates that can give us all possible basics function. So, we will call it has an universal set, and then NAND and NOR gates their individually universal gates and we can have other combination like AND, OR, INVERT XOR, AND XOR, OR like that.