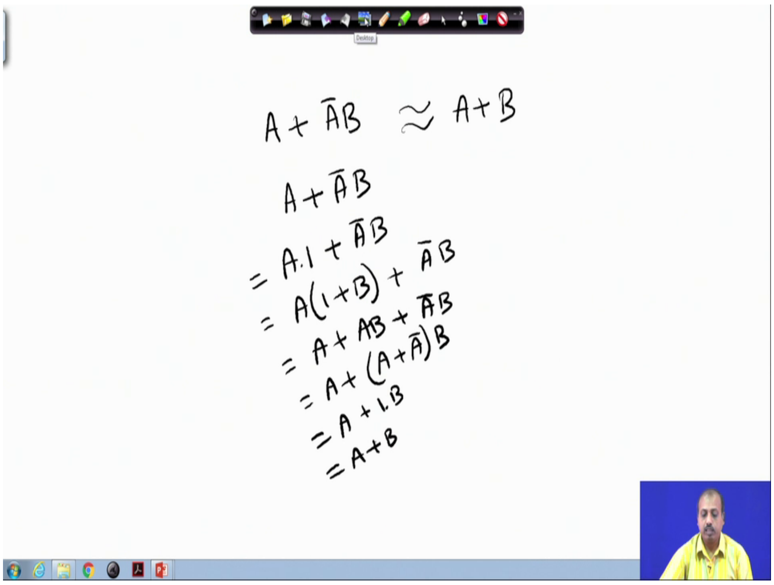
Digital Circuits
Prof. Santanu Chattopadhyay
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur

Lecture – 14
Logic Gates

So, next we shall be starting with Logic Gates. So, gates are used for realizing digital circuits. Now, before we do that we will just clear one confusion, which was there regarding this.

(Refer Slide Time: 00:38)

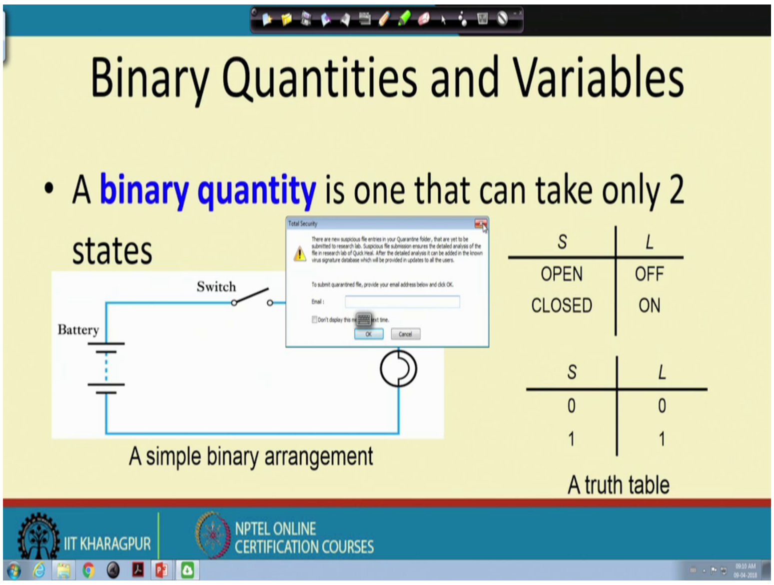


So, there was a confusion in Boolean algebra, where we had an expression like A plus A bar B and which we wanted to equate to A plus B so, that simplification process there are some problems.

So, let we will do the simplification like this like A plus A bar B it can be written as A dot A dot 1 plus A bar B, where this one can be written as A into 1 plus B plus A bar B. So, if you multiply you get A plus A B plus A bar B and so, it gives A plus A plus A bar in and B. So, this A plus A bar is again 1 so, this gives A plus B.

Since in a 1 plus there are some confusion about it. So, this is the derivation So, with this we will be starting with this logic gates. So, as I said that logic gates are circuit elements digital circuit elements by which we can realize the digital circuits. So, far we have seen how to minimize the Boolean function and discuss onwards we will see how to realize a Boolean function in terms of components.

(Refer Slide Time: 02:03)



So, so, if you the binary quantities and variables so, if you look into a binary quantity is one that can take only two states, so either it is high or low if you can think of it, as if there is a battery and, there is a switch and a lamp. And then when this switch is closed then, only the lamp will glow.

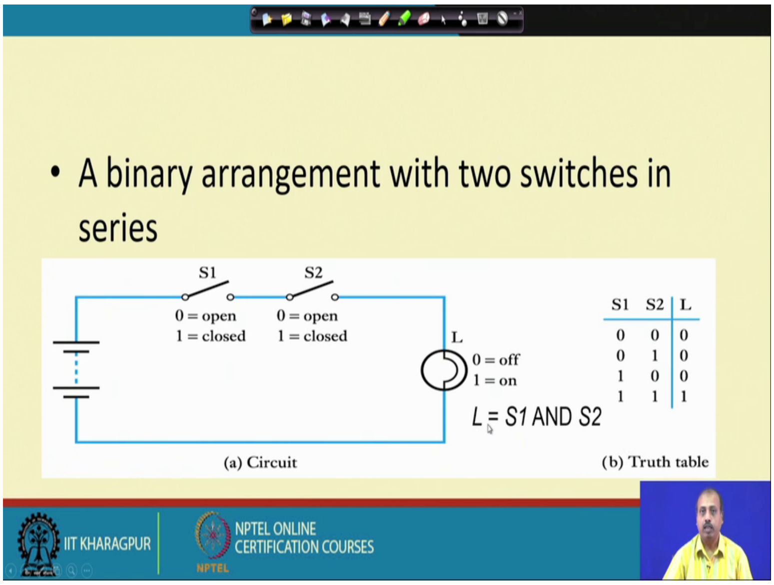
So, the switch was got two positions open position, when it is now like this at the closed position, when this bar connects to this point. So, that way we have got two positions for the switch open position and closed positions. So, when this is open then the lamp does not get the current flow. So, lamp is off and when this is closed the lamp is on.

So, that we can think of it as a variable the switch can be considered as a variable that has got two states in it open and closed and, accordingly the length has got two states off and on. So, this S and L that this switch and lamp both are binary quantities, or binary variables, you can just represent in the form of truth table, where we say that this open is represented by 0 and closed is represented by 1 for the switch.

So, switch can be either at state 0, or at state 1, on the other hand the lamp it can be off which is represented by say 0 and on which is say represented by 1.

You see that this open represented by 0, or this on represented by 1 so, these are arbitrary. So, this is done by the designer ok. So, for the design the simplicity so, you can follow any of the logic however, one thing is true that whatever convention we follow so, that has to be followed uniformly. So, you cannot change from one part of implementation to another part while doing these thing.

(Refer Slide Time: 03:51)

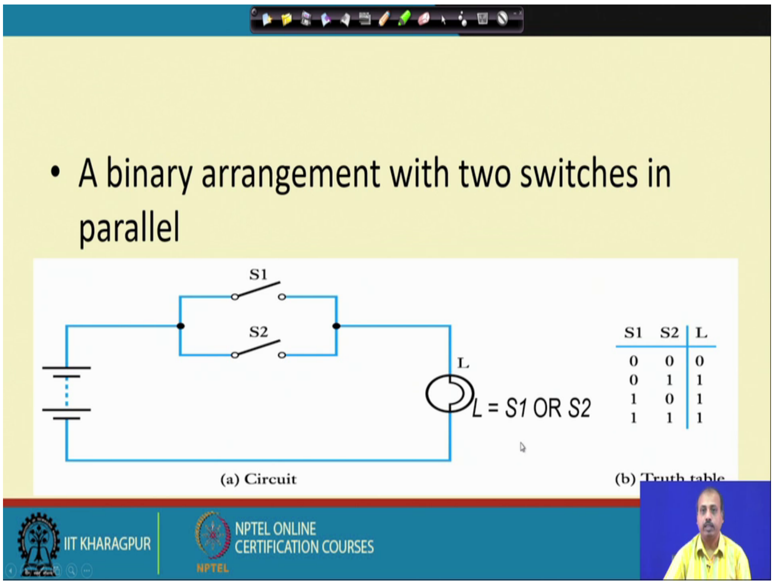


So, if we consider two switches like the suppose I have got S 1 and S 2 2 switch switches they are connected in series. So, the S 1 can be either open, or closed and S to can also be open or closed. Now, this lamp it get is the current flow only when both S 1 and S 2 are in closed state ok.

So, if I represent 0 open by 0 and closed by 1. So, only when both the switches S 1 and S 2 are closed. So, they are getting the value 1 and 1, then only the lamp is on so, this L equal to 1 otherwise the lamp is 0.

So, this is actually the condition that the lamp gets on only when both S 1 and S 2 are on both are closed. So, this S 1 and S 2 so, please mind this term and so, this AND gate is coming into picture. So, here I say that L the logic expression for L S is given by S 1 and S 2. So, when S 1 and S 2 both this variables get the value 1, then only L will get the value 1, otherwise L will get the value 0. So, this is the AND operation of S 1 and S 2.

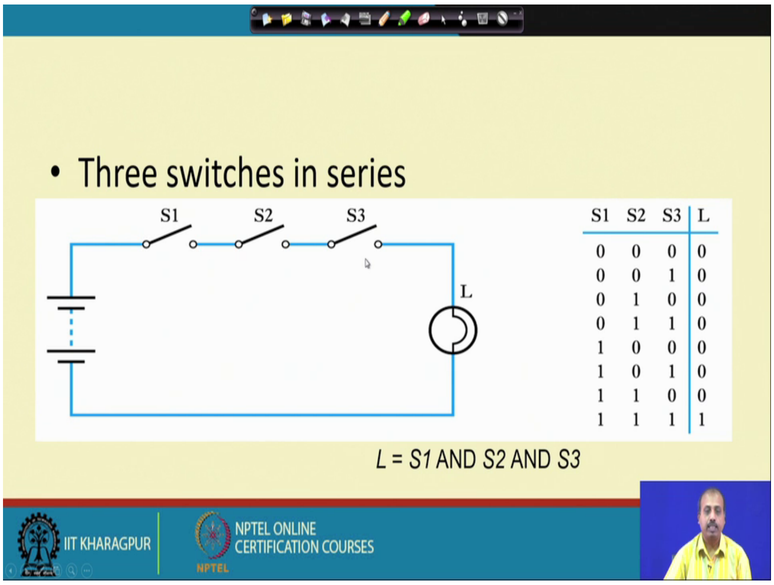
(Refer Slide Time: 05:00)



Similarly we can connect the two switches in parallel. So, apparently it seems why should we connect two switches in parallel, but for the sake of examples. So, let us take it like this that we have got two switches connected in parallel S 1 and S 2. And either of the switches being closed the light gets the lamp L gets current as a result it turns on. So, we can say that if again the same thing, if 0 for switches represent they are open and one represent they are closed.

So, whenever either S 1 or S 2 is 1, then L value becomes equal to 1. So, this is the truth table and, here the operation that we are doing say L is on if S 1 is 1, or S 2 is one again mind this term or, because this is the or operation of 2 binary quantity. So, this L binary quantity, or binary variable L, it get is the value of S 1 and S 2, S 1 or S 2 where S 1 and S 2 are the again 2 binary variables.

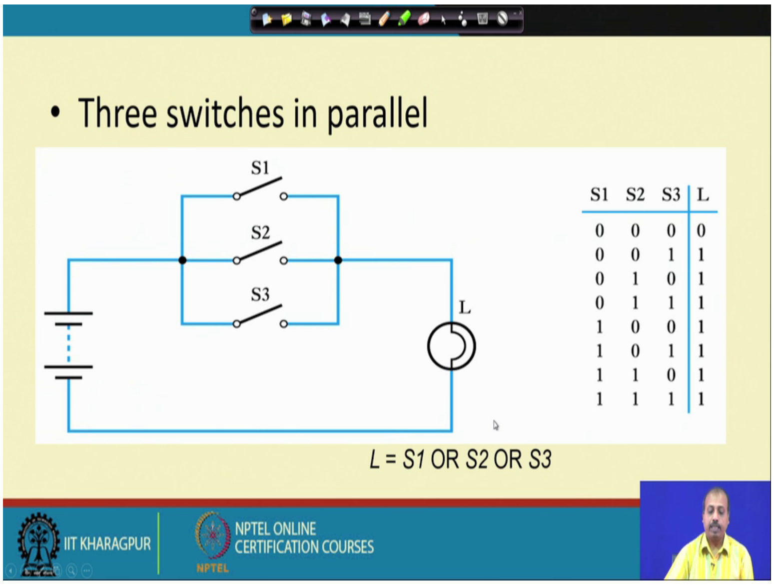
(Refer Slide Time: 06:00)



We can connect three switches in series like S 1, S 2, S 3. So, what happens is that here this lamp L get is current, only when all the three switches are closed. So, this can be represented in the form of a truth table, where this S 1, S 2, S 3 is so, if I just write down all the possibility. So, S 1, S 2, S 3 these are three variables so, it can start from 0 0 0 0 1 and go up to 1 1 1, where 0 means the switch is open and one means the switch is closed.

So, as I said that since this is a series switch S 1, S 2, S 3 so, this L will be equal to 1, only when S 1, S 2, S 3 all of them are equal to 1, So, in the truth table also you see that this L equal to 1 only when S 1, S 2, S 3 all are equal to 1.

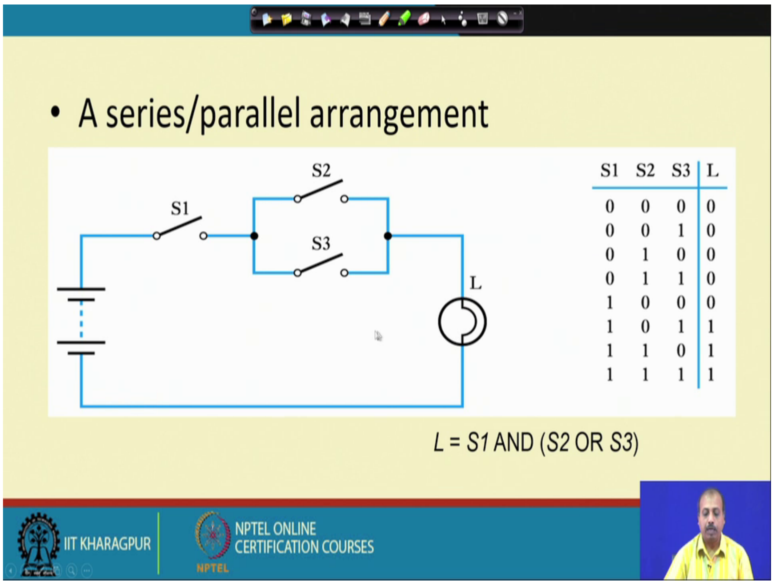
(Refer Slide Time: 06:48)



Similarly, we can have OR operation. So, we can have in parallel the switches may be connected in parallel S 1, S 2, S 3, they are connected in parallel so, whenever any of these switches are closed the lamp is turned on and, when all the switches are open and then the lamp is off. So, whenever this lamp L equal to 0 only when S 1, S 2, S 3 all of them are equal to 0. Otherwise whenever the any of them is equal to 1, the L value is equal to 1.

So, we can say that as if this lamp L is equal to S 1 OR S 2 OR S 3. So, it is the OR of 3 variables. So, 3 all the S 1, S 2, S 3 variables are OR to get the value of L, but OR so, for whatever we have seen OR is a 2 variable operand 2 opera it is a 2 variable operation OR 2 operand operation. So, we have to write 2 OR's in between so, S 1 OR S 2 OR S 3 ok.

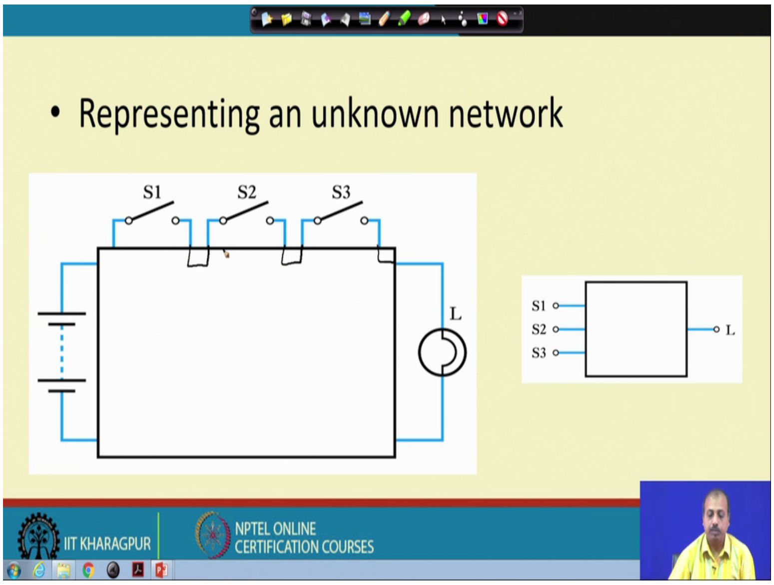
(Refer Slide Time: 06:50)



So, this way we can represent 3 switches in parallel slightly more complicated one. So, let us say that we have got the switch S 1 connected in series with a parallel combination of S 2 and S 3. So, for this lamp to glow S 1 must be closed and at least one of the S 2 and S 3 should be closed. So, this is basically S 2 OR S 3 AND S 1 ok. So, that is the logic for L so, here also if you look into the truth table, this L is on for L to be on S 1 must be equal to 1 and at least one of S 2 and S 3 should be equal to 1 so, then only L is equal to 1.

So, this over all operation of this a lamp can be written as L equal to S 1 AND S 2 OR S 3. So, that captures the status of all the three switches S 1, S 2, S 3 and we get the combination ok. So, this way we can represent this operation on Boolean quantities by means of switches and some output.

(Refer Slide Time: 08:50)



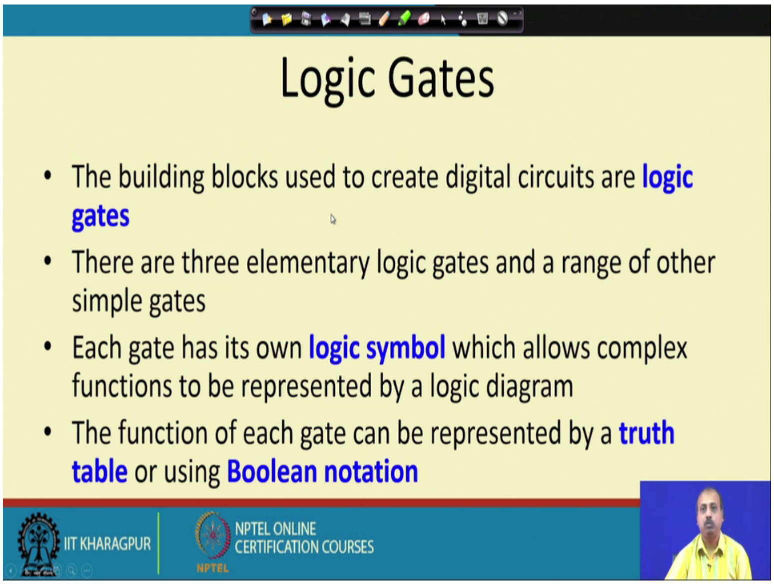
So, now, if we suppose we have got an unknown network ok. So, this is a black box so, this is a black box I do not know what type of connection we have between these points. So, this say from this output of S 1, how is it connected to S 2 so, S 2 S 3 maybe connected in series maybe connected in parallel maybe all of them are in parallel or maybe all of them are in series.

So, they are may be many possibility. So, it maybe the connection pattern is the like this, that from this point sorry from this point we have got so, from this point, we have got a connection that takes it to this as well as this. And then from this and this they are connected to the output point.

So, what happens is that so, we are getting S 2 OR S 3, because S 1 has to be closed and then at least S 2 OR S 3 has to be closed so, that we can get that so, that so that we can reach this point ok. So, this way or it may be that we have got a connection like this that S 1 is connected like this and S 2 is S 2, S 3 it is connected like this and, this is the connection.

So, that way what we have is that S 1, S 2, S 3, they are connected in series ok. So, this is AND operation so, if this is an unknown network if this is an unknown network, then we can we may try to represent. So, in a black box form so, we can say that as if there are three input S 1, S 2, S 3 for this black box and there is one output L Now, how do you will find out the operation so, exact operation.

(Refer Slide Time: 10:28)



So, for that what we need to do is that we need to we need to apply different combinations and see whether the light glows they are all not. Suppose only when we connect S 1, S 2, S 3 all of them in closed position, then only the light glows; that means, only when S 1, S 2, S 3 all the variables are 1 L will be equal to 1.

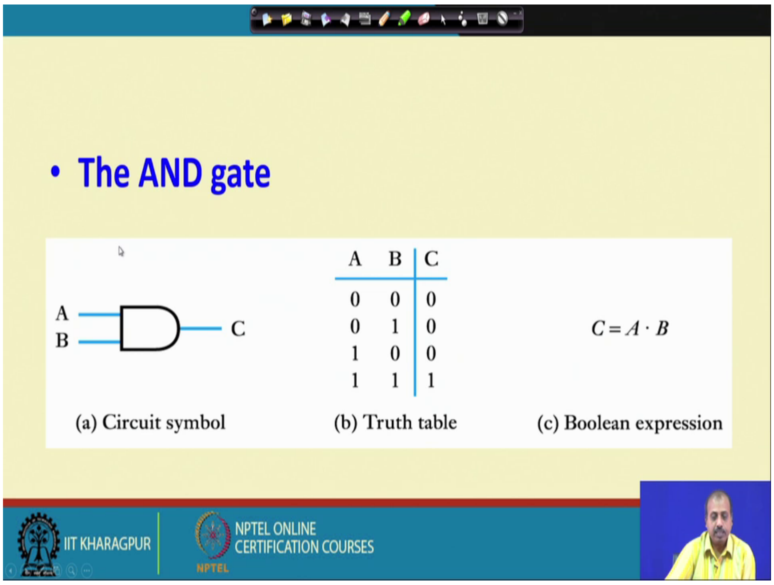
So, it is a and of three variables so, that way you can identify the logic function that is implemented by this box by applying different switch combinations here, and checking the status of this light.

So, this logic gates so, they are building blocks for the creating digital circuits and, there are three elementary logic gates and the range of other simple gates. So, we will see that there are three basic gates and some and there from and from there, there are many other derived gates, each gate has it is own logic symbol which is universally accepted. So, it is expected that we use those symbols to represent the close logic gates.

And so, we can represent ah. So, since each of them is a symbol so, you can use those symbol for AND operation OR operation etcetera and, then connect between the lines by means of some lines connect between the inputs and outputs by means of lines, to get the overall logic diagram.

So, that we will call a logic diagrams so, the complex function can be represented by logic diagram and, the function of each gate can be represented by a truth table, or using the Boolean notation. So, you can either use a truth table, or a Boolean notation in terms of some Boolean function you can represented.

(Refer Slide Time: 12:00)



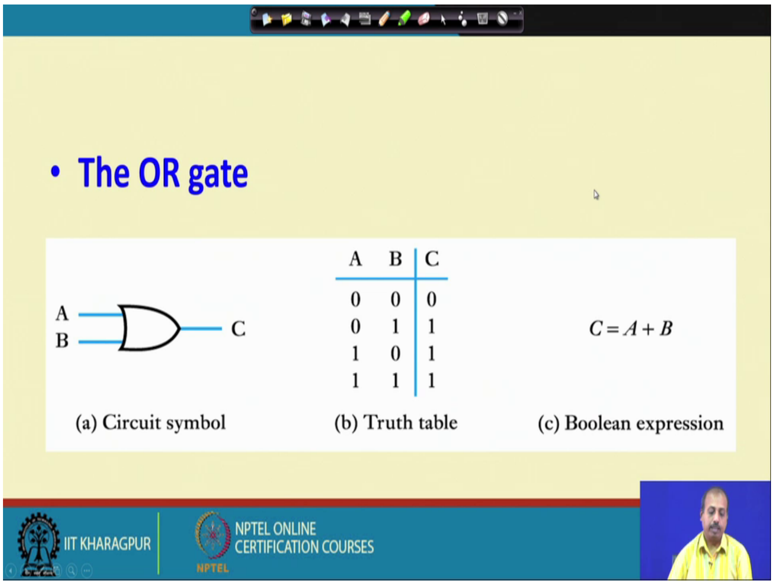
So, we will start with the first basic logic gate which is the AND gate. So, this is the symbol of the AND gate ok. So, this is universally accepted so, it is expected that whenever we are drawing AND gate we use this symbol. Now, why this symbol there is no reason like that and, in some special cases you may use some other notations, but it is expected that in general use this notation. So, this is the symbol so, whenever in the digital circuit you see this symbol we understand that it is an AND gate.

Now, there are 2 inputs coming to it and 1 output going out of it. So, AND gate always has got 1 output, but the number of inputs maybe more so, in this particular examples so, you have got 2 input AND gate. So, if there are more number of input. So, more number of lines will be connected to input side. So, this is the input side this is the output side, so, if there are more number of inputs. So, there will be connected to the more number of lines in the input side.

So, this same circuit symbol so, it is it can be represented in the form of truth table. So, truth table actually tells what is the functionality implemented by the corresponding logical element. So, as the name suggest this is an AND gate so, only when both the inputs A and B are equal to 1 will be having output C equal to 1. So, so, 0 0 when the inputs are 0 and 0 output is 0 0 1, then also 0 1 0, then also 0 and 1 1 is 1.

So, this is the truth table and the in terms of Boolean expression. So, this is an and of A and B 2 variable. So, C equal to A and B so, in different situation so, we will be using different notation like when you are trying to represent the circuit, we will draw this type of gates this type of circuit symbols, when we are interested about the functionality in the form of some truth table. So, we can write it like this detail functionality sometimes, we need a compact notation. So, there will be writing in terms of the Boolean expression. So, depending upon our requirement we will do that.

(Refer Slide Time: 14:00)



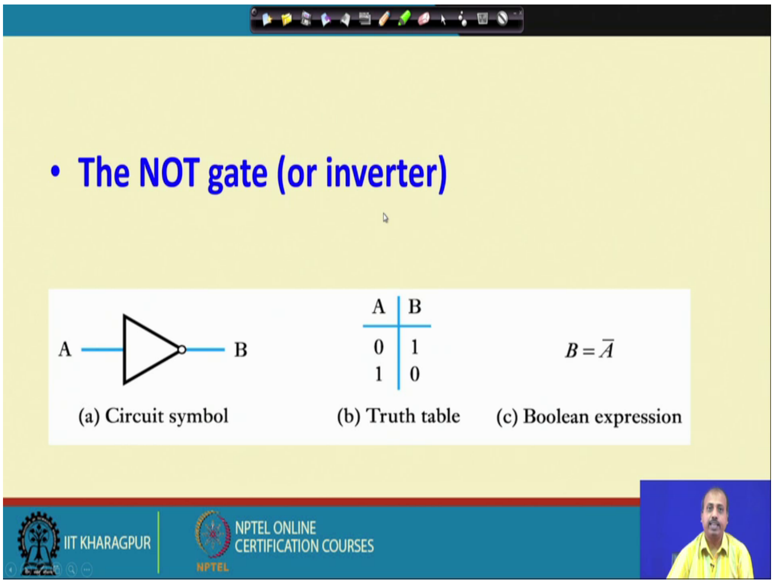
The next fundamental gate that we have digital circuit is the OR gate and, again thus this symbol is the universally accepted symbol and, it is again expected that for OR gate we will represent it like this. So, this is the circuit symbol, this is the truth tables. So, or means whenever at least one of the inputs is equal to 1 output will be equal to 1.

So, this is happening here so, you see whenever A or B at least 1 of them is equal to 1 output C is equal to 1. So, we get C equal to A plus B so, this is the Boolean expression and, this is the truth table.

Again the same thing that this is a 2 input OR gate. So, if you want to more number of input so, they can be added here. So, theoretically there is no limit on the number of inputs that you can have to a gate, but of course, ultimately this is the these will be represented by digital IC's. So, they will have their own restrictions. So, and you will find different number of inputs where different IC chips.

So, you can have say 2 input OR gate or 4 input OR gate like that. So, normally you do not have do not have odd number of inputs the gate and 2 and 4 these are the common ones. So, we do not have more because that will make the gate more complex.

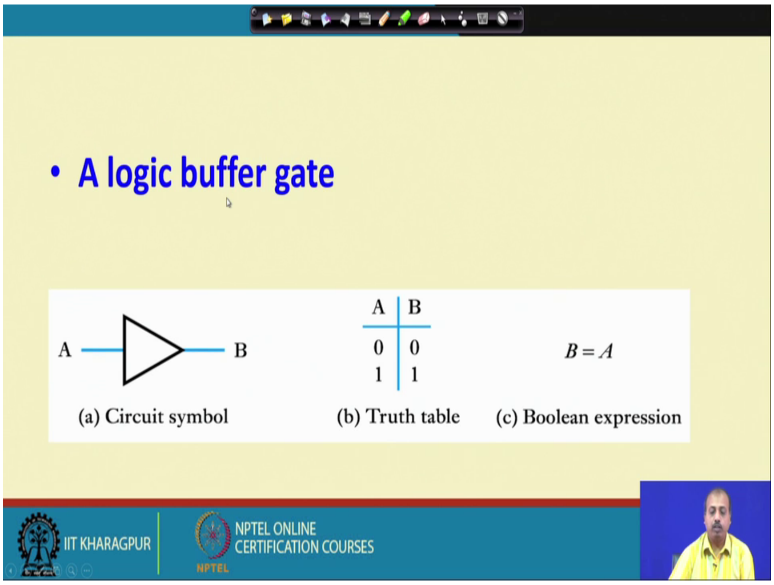
(Refer Slide Time: 15:17)



Another fundamental gate that is there is the NOT gate, or the invertor. So, it is symbol is like this a triangle and on top of the there is a bubble is ok. So, this is a single input so, OR NOT gate cannot have more than 1 input. So, it has got a single input and it has got a single output.

So, as the name suggest so, this is the invert of the input output is the invert of the input. So, in terms of truth table whenever A is 0, B is equal to 1 and, whenever A is 1 B is equal to 0 and, in terms of Boolean expression it is written as B equal to A bar ok. So, this is A bar so, this is the complement of A so, that is the NOT gate ok.

(Refer Slide Time: 16:00)

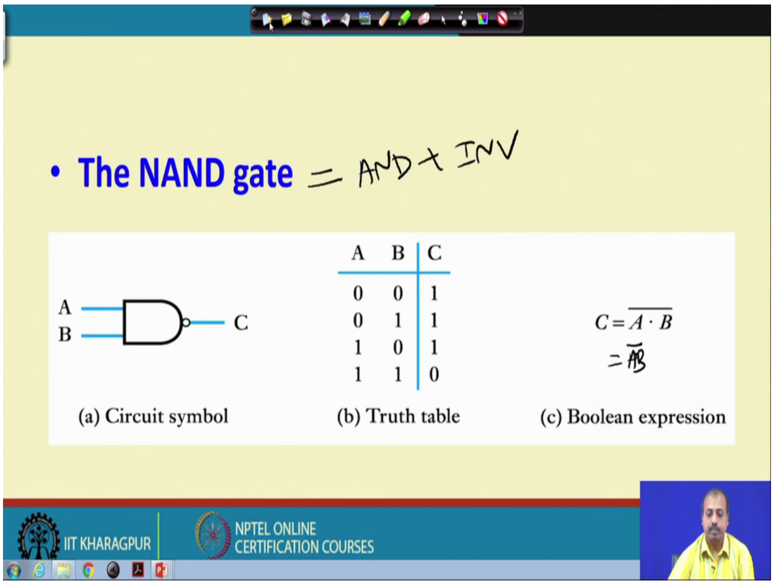


So, sometimes we use a logic or buffer gate. So, this is buffer means that it just copies the input to the output. So, a whenever A is 0 B is also 0, whenever A is 1 B is also 1 logic A Boolean expression is B equal to A, apparently it is seems that why do we need such buffer gate so, ok.

So, sometimes what happens is that we need to drive some large load, in terms of the in terms of circuit elements like it may be it may so, happen that one gate output it drives a number of gate out gate inputs in the next stage. So, that way it has to drive a large current and these buffers, they have the capacity to drive large current ok.

So, they can be connected. So, that way we can use this type of buffers so, this otherwise as for as logic is concerned, it does not add anything to the circuits. So, the it just copies the input to the output.

(Refer Slide Time: 16:53)

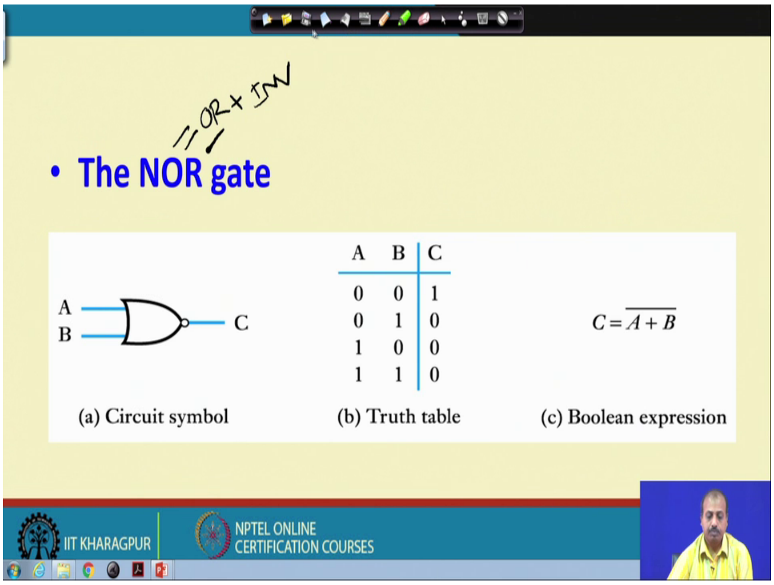


Next we come to some derived gates like the NAND gate first one is a NAND gate. So, this is actually you can say that NAND is equal to a NAND is equal to AND plus inverter ok. So, as so, this is a AND plus inverter you see that here the symbol is also like that. So, this is the AND gate and after that there is an inverter bubble ok. So, this A B so, as the name suggest. So, this will be output will be equal to 1 ok.

Whenever the AND gate output will be 0. So, AND gate output is 0 whenever any of these input is equal to 0. So, in terms of NAND gate we can say that whenever any of the inputs is 0 output will be equal to 1 and, only when both the inputs are equal to 1 the output will be equal to 0. So, this is the truth table. So, you see that whenever when both the inputs are equal to 1 output is equal to 0, otherwise output is equal to 1.

So, symbolically so, in a Boolean expression. So, it is represented like this C equal to A B bar so, this is also written as this dot is often ignored so we write it like this so A B bar so this is also there. So, this NAND gate so, you see that NAND gate is called a derived gate, because it can its can be derived from the AND gate and inverters and OR inverter, they are fundamental gates and, from them we can derived this gate. So, NAND gate is one such derivation.

(Refer Slide Time: 18:30)

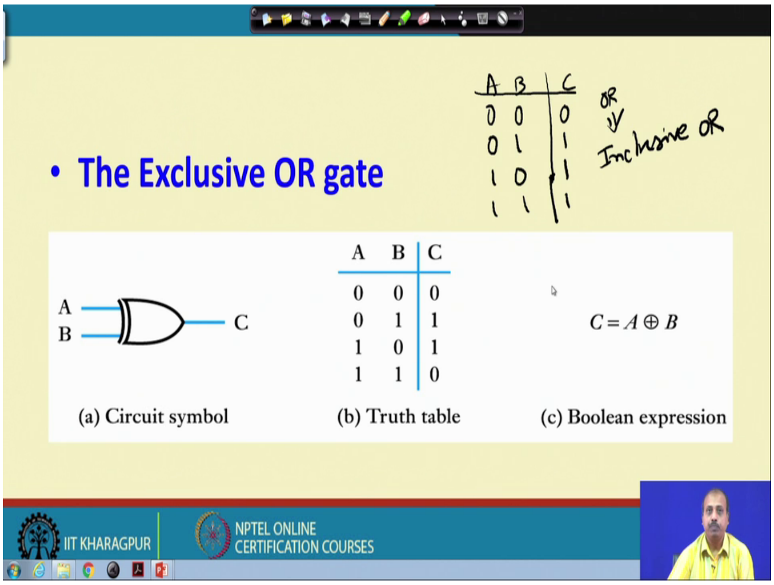


Next we will look into NOR gate so, just like NAND gate was equal to AND plus NOT so, similarly if you do OR plus NOT what you get is a NOR gate. So, what we have got this NOR equal to this NOR is equal to so, OR plus inverter. So, OR plus inverter and the symbol is also like that. So, we have got here the OR and then AND there is an inverting bubble. So, OR plus inverting bubble. So, that is the circuit symbol for NOR.

Of course the other thing remain same like I can have more number of inputs here and all and then the truth table is simple like OR gate was whenever, any of the inputs is equal to 1 output was equal to 1, so since this is NOR so, whenever any of the inputs is equal to 1 output will be equal to 0 and only when both the inputs are equal to 0 this NOR gate output is equal to 1.

So, this is the C equal to A plus B bar so, that is A plus B whole bar. So, that is the Boolean expression for NOR ok. So, of course, you can apply De Morgan's on this to simply to simplify the circuit that we will see later.

(Refer Slide Time: 19:52)



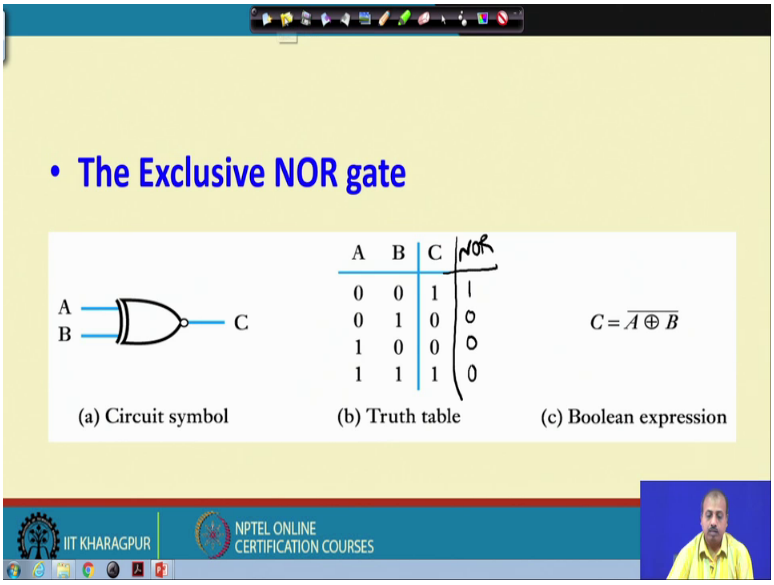
So, next we will look into the another derived gate which is known as exclusive OR gate. So, if you remember the OR gates. So, it was like this so, the truth table for OR gate was the A B and C, whenever the whenever any of the inputs was equal to 1. So, this was equal to 1 so, 1 0 is the also 1 and 1 1 was also equal to 1. So, this was the truth table for OR gate.

Now, you see that in case of X OR gate we first compare at the truth table levels. So, truth table wise you see that when both the inputs are equal to 1 X OR gates gives output 0 whereas, OR gate gives the output 1 so, the so, it so as the name suggest this exclusive OR gate it requires that exclusively only one of the inputs will be equal to 1. So, if both the inputs are equal to 1, then the output will be equal to 0. And as a result this OR function. So, this is also sometimes called inclusive OR so, this is also called inclusive OR, just to separate it out from exclusive OR. So, this is also known as inclusive OR.

So, sometimes we will see that exclusive OR gate detail later, it has got many fundamental usage in digital circuit design and, you see that here what we are having we are having 2 inputs A B and the C is output and C can be written as this is the symbol for X OR.

So, so this plus is the symbol for OR and, if you put a circle n around it. So, this is will be an X OR operation. So, this is an X OR operation. So, this is if both the inputs are equal to 1 in the out will be 0 output is 1 only when exactly one of the inputs is equal to 1.

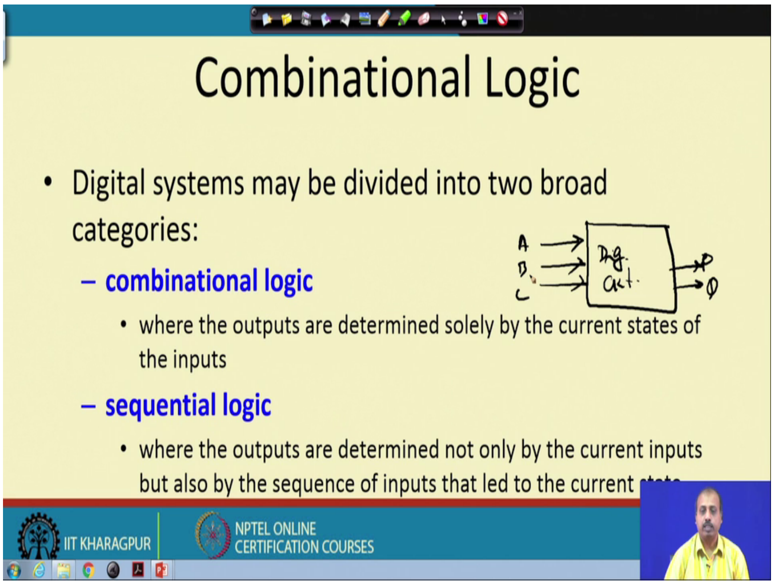
(Refer Slide Time: 21:45)



So, next we will look into the inverted version of X OR which is known as X NOR gate exclusive NOR gate. So, this is NOR but exclusive. So, NOR gate was that whenever we have got any of the inputs equal to 1 output was equal to 0. This is the difference in X NOR gate when both the inputs are equal to 1. So, output is equal to 1 so, this is not equal to 0.

So, in a NOR gate this output is equal to 0, but in an exclusive NOR gate so, these output is equal to 1. So, for your X for NOR operation. So, the truth table is was like this. So, this was the NOR operation where when both the inputs are equal to 1 output was equal to 0, now it will be equal to 1. So, symbolically we represent it as C equal to A X or B whole bar, or it is written as it is read as A X NOR B so, it is read as A X NOR B.

(Refer Slide Time: 22:50)



Next we will look into some design of combinational logic. So, we are using this basic gates that we have discussed so, far. So, we can realize some combinational function, any digital circuit can be represented by in terms of these gates, they will slowly go towards that.

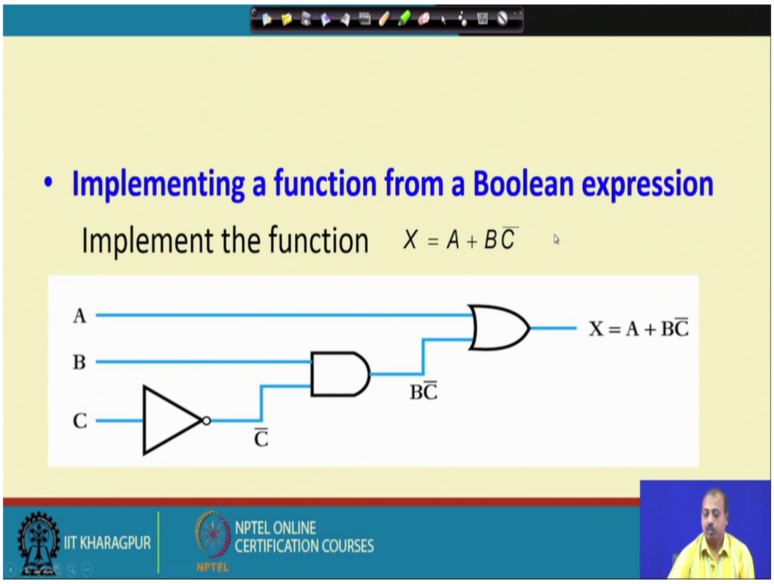
Now, if you look into all digital systems. So, they can broadly be classified into two categories one is known as combinational logic, another is known as sequential logic combinational logic is like this. So, output is determined fully by the current states of input. So, if I if I represents the if I represents this is suppose is digital circuit, suppose this is a digital circuit.

Now, it has got a number of inputs say A B C and, it has got say 2 outputs P and Q. Now, if it happens like this that the values of at any point of time, the values of P and Q it is dependent on the values of A B C at current instant only ok. So, it is does not depend on the previous values of A B C. So, if that is the situation then will tell that this is combinational circuit, because it does not depend on the on a it does not depend on the history of the inputs in previous times.

on the other hand, there will be an another class of circuits where you will see that this output P and Q it will depend not only on the current value of A B C, but on the previous values of A B C also like. If you observe the system from time t equal to 0 and at present you are at times a t equal to 10, then the output P Q at time ten will be dependent not only on the values of A B C at time 10, but also on the values of A B C at time instants nine 8 7 6 up till 0.

So, at all the previous values of A B C so, it need not be exactly at this bound that is like 9 8 7. So, you can say that it depends on the total history of this inputs at pre from the beginning of the system. So, that type of logic will be known as sequential logic. So, in our course we will be dealing with the both the types of logics. So, both combinational and sequential, but initially we will be discussing on combinational logic because, that that is the simpler to understand and then we will proceed towards the sequential logic.

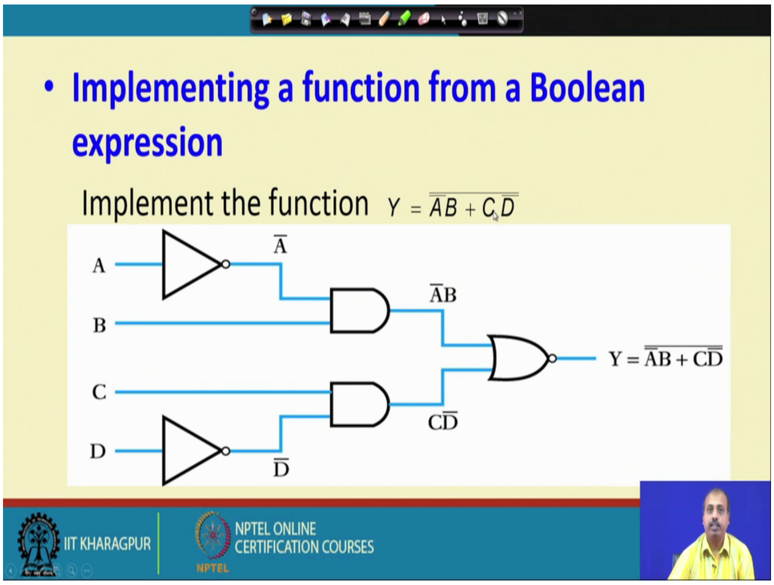
(Refer Slide Time: 25:22).



So, suppose we have got Boolean expression X equal to A plus B C bar ok. So, to how can we realize in terms of logic gates. So, you see that this OR so, fine at the highest level we have got this OR operation ok. So, we so, at that at the output at the top most level. So, we will have this OR gate so, OR gate will have 2 inputs. So, this A input and this B C bar inputs. So, input is connected to first input of OR gates and this B C bar will has to be connected to the second input of OR gate.

Now, how do you get B C bar for getting B C bar? So, you have to do B and C bar. So, you need an AND gate so, where 1 input is B another input has to be C bar. So, this here you get B C bar now how do you so, B we have already got because, B is a primary input to the circuit and, then the C bar to get to get C bar what we do is we start with C put an inverter on here, and then from the C we get a C bar so, that way this the circuit. So, it represents the it can implement the Boolean expression A plus B C bar.

(Refer Slide Time: 26:37)

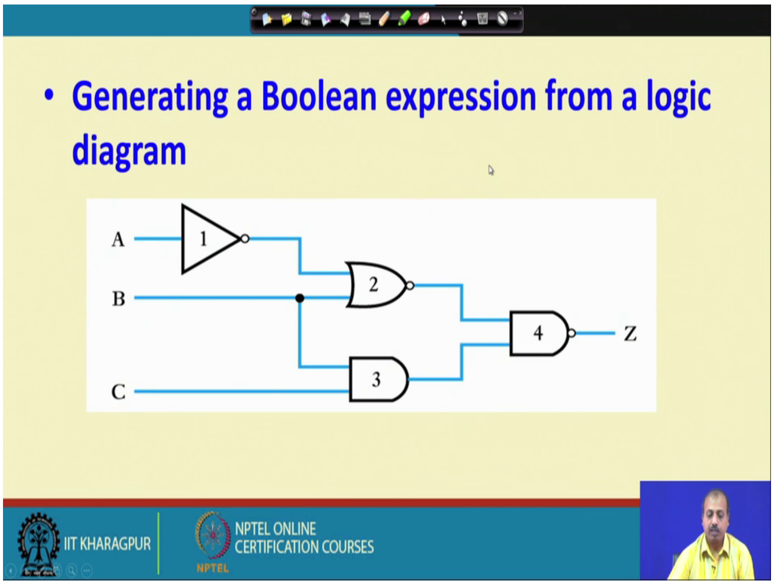


So, this way these Boolean expressions are implemented like let us say another take another examples slightly more complex Y equal to A bar B plus C D bar whole bar ok. So, you see that this if you look into the highest level. So, this is a NOR operations so, this plus so this is OR and there is a inversion at the top so; that means, this is a NOR operation. So, I take a NOR gate and, then to the NOR gate I should have the inputs A bar B and C D bar. So, this NOR gate it is getting two lines. So, in 1 line I have to realize A bar B another line I have to realize C D bar.

Now, how do I realize A bar B for getting A bar B I have to do and of A bar and B ok. So, this is a AND gate and we have got A bar and B here. So, B is coming directly from the primary input, but to get A bar I have to take and inverter and, then this a will be applied here and A bar will be obtained from there.

Similarly, for getting C D bar so, I need another AND gate and in this AND gate C will be one of the input and B bar will be the other input and for getting D bar from D I have to have an inverter in between. So, this way starting from the top level of your expression so, you can just go back, realizing till the simplest element that is till you reach the primary input. So, until the expression refines to the primary input we have to go on doing this thing.

(Refer Slide Time: 28:00)



So, we can also do something for generating the Boolean expression from a logic diagram. So, suppose this is a Boolean this is a logic diagram. So, we can trace through this logic diagram to see how the corresponding Boolean expression can be derived.