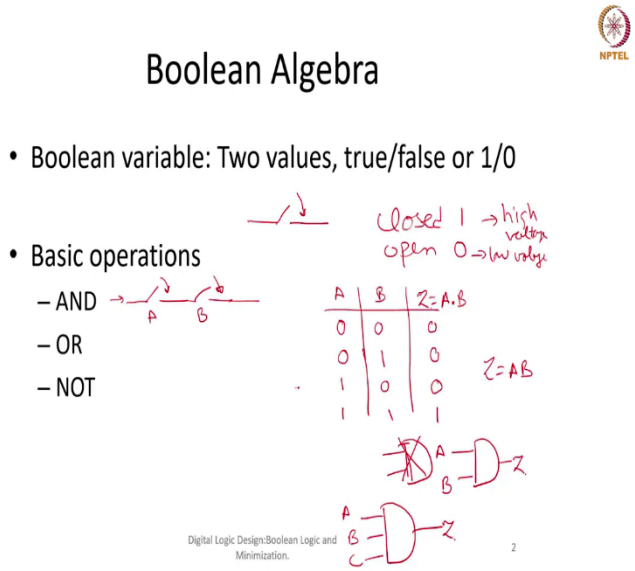
**Digital System Design**
**Professor Neeraj Goel**
**Department of Computer Science Engineering**
**Indian Institute of Technology Ropar**
**Boolean Functions**

Hello friends. Today we are starting this new module Boolean Algebra Logic Gates and Minimization. So, in our previous module, we have understood how various form of information like integer, float or characters, strings, how all of these information can be represented, represented in form of 0s and 1s. Now, towards designing a full fledge digital system, the first step is to understand how this, the information which is represented in form of 0 and 1 can be processed or how we can operate on those 0s and 1s.

So, to understand how that processing will work, how any mathematics which involves only two symbol 0 and 1, how that mathematics will work. So, the basic foundation of this mathematics is called Boolean Algebra. So, we will start in this module with the understanding of what Boolean Algebra is and what kind of operations we can perform on these Boolean expressions or Boolean variables.

(Refer Slide Time: 1:30)




Boolean Algebra was essentially an algebra who was invented by George Bool. So, at that time when he invented Boolean Algebra, he considered two values; 0, true and false. Now this algebra which was invented one and half century ago, it has various advantages in different field of mathematics as well as computer science like set theory, logic, and here digital logic.

So, when these two values, true and false are considered 1 and 0 or it is called two valued Boolean algebra. It is also called switching algebra. So, basically when you have a switch either it would be 1 or it would be 0. So, that is where it was, the term was invented as switching algebra. So, in a Boolean algebra there would be each variable will have only two values, either it would be 0 or it would be 1 and there are 3 basic operations that can happen on these any of the Boolean variable.

So, one of the operation is AND, another is OR and the third one is NOT. So, when we are saying AND operation, so first of all let us see that how a switch would look like. In a switch we can say that this is a switch, so whenever it is closed, then this would represent closed means 1 and when it is open, it is called 0. Similarly, similar to the switches which were used like 70-80 years before invention of logic gates, so in logic gates we will have 1 means actually high voltage or high voltage or high volt, high current or 0 means low voltage or low current.

Now, with the meaning of this 0 and 1, let us see what kind of basic operations we can perform. So, there are 3 basic operations, one is called AND, then OR and then NOT. So, AND means that there are two Boolean variables, both of them when let us say, there are two Boolean variables, when both of them are On, then only my output would be On. So, let us see this way, so let us say there are two switches, this switch is called A and there is a another switch which is called B.

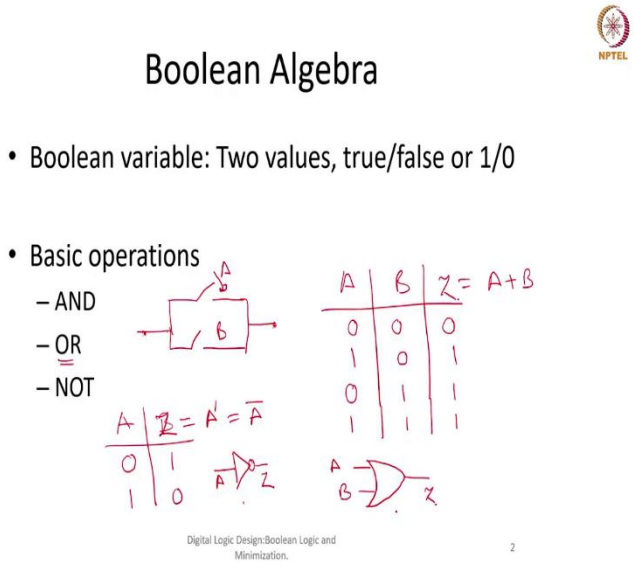
So, right now both of them are open, so because both of them are open, my output here is going to be combined output is going to be open because my current, if there is a current here, it cannot pass to this place. So, my let us say if I say this A value and this is B value, if both of them are open, that means both of them are 0, my output Z or a combined output Z would be 0. So, let us say one of them is open and one of them is closed, what is the effective output of this this whole circuit, it is still open because let us say B, A is open but B is closed, but because A is open, so current cannot transfer from this point to this point, so Z is going to be 0.

Similarly, if A is 1 B is 0, then also my output is going to be 0. When both of them are closed, both of them are closed, then only my output will be 1. So, this whole thing like these two variables can be extended to n variables, so either n variables in this AND gate and all of them, when all of them are 1, then only my output would be 1, so this is the basic characteristic of this basic operation of AND. Now when we are writing algebraically or

whenever we are writing any expression, so either we will write this dot symbol between two variables that signify that there is a AND operation or something we do not even specify.

So, we also say Z equal to AB, then also because there is something in between there is no operator written in between, so then also we say that it is a AND operation. And rather than describing using switches, this AND symbol is described using this particular symbol. So, I will write it clearly. So, this is the symbol for AND gate. There could be any number of inputs, so let us say the number of inputs I want to have more than 2, then also it is possible. A, B, or C so any number of inputs are possible if we are creating this AND gate. Now, let us say now let us see how this OR gate will work.

(Refer Slide Time: 7:16)




So, in case of a OR gate, the switches we can put like this, this is my input and this is let us say another switch, so this switch I am calling A, this switch I am calling B and my output would be measured across these two points. So, let us see all these scenarios again. Now, when the switches are in parallel, so let us say A is 0 and B is 0, so what is the effective output between these two points? So, A is 0, B is 0 that means Z is also equal to 0 because both of them are not connected, so my output is also not connected.

Now, let us say one of them, let us say A is 1 and B is still 0, what is the output between these two points? Now, because there is a path from A, even though B is open, then also we will say that the Z is 1. Similarly, if A is 0 but B is 1, then also output is 1 and if both of them are 1, then also output is 1. So, this is the basic operation of OR gate. In other words, so what is the logical symbol of this, we say A plus B.

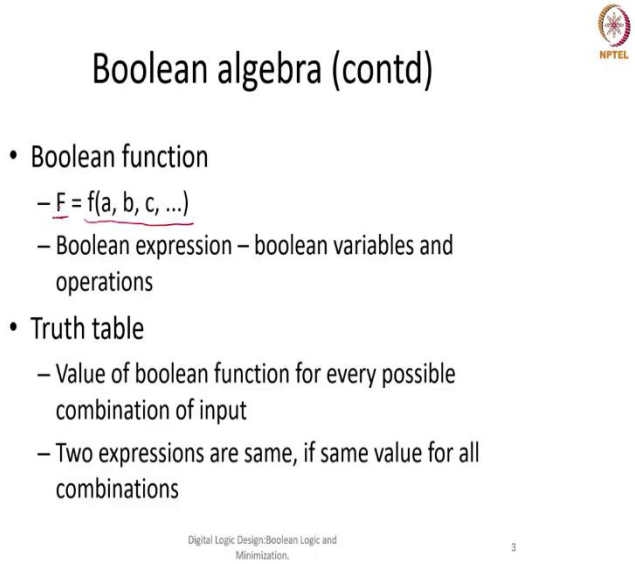
So, all though it is written plus, but because A and B both of them are Boolean variables, Boolean means they can have only two values, 0 and 1, and because both of them are Boolean variable, whenever we say plus sign, this plus sign essentially represent OR. How do we represent them symbolically? So, a symbolic symbol for the OR gate is like this where there could be any number of inputs and this is the output.

Now let us see this in a logical sense also. When we say that there is a variable A and variable B, so when we say A and B both of them should occur, so that means only when A is 1 B is 1, then only that expression would be 1. When we say A or B that means any of the event or any of the variable is 1, then we should be able to say our logic is correct. So, this is how AND and OR these basic operations will work.

The other one is called NOT. NOT is a inverter, so let us say if my A, so this will have only two, one input and one output. So, if A is 0, input is 0, the output is going to be 1. If input is 1, output is going to be 0. So, that is how a NOT gate will work or a NOT operation will work and how to represent it symbolically? Symbolically it is represented like a triangle and there is a small circle over there. So, A and Z. Algebraically we write Z if it is a inverter, most of the time we write, we put an dash over it and sometime we also say that there is a big bar or a bar over that variable that also means that that variable is operated by NOT operator.

So, this word operator and gates can be used interchangeably, we can say that this is a NAN, this is a NOT gate, this is a OR gate and similarly we have described the AND gate couple of seconds before. So, these are the basic gates. Now using these basic gates, we can think of even more complex operations and we can also think of some basic property of these variables and how when they would be operated, les say they are not operated against variable, but a constants. So, let us see some of these things.

(Refer Slide Time: 11:59)

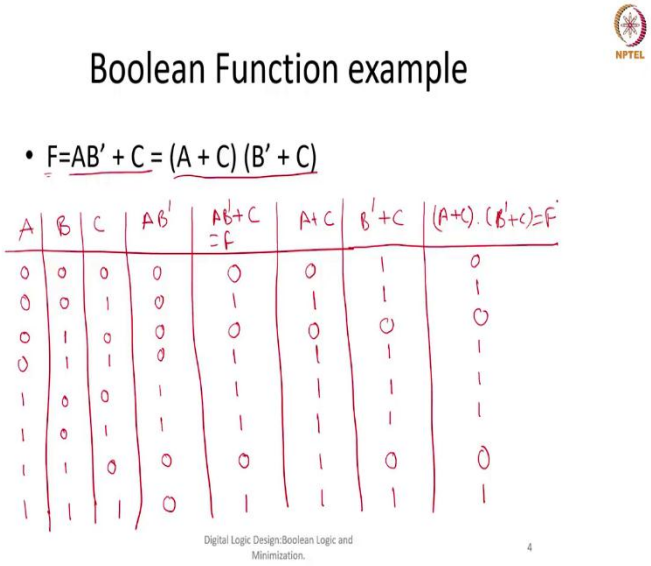



So, now couple of more definitions. A Boolean function is a function F so this F is also a Boolean variable, now a Boolean function will have n number of or a many number of Boolean variables and it would work only on the Boolean operators. Boolean operators, the basic operators are AND, OR, NOT, there could be even more complex operations which can be done using the combination of this AND and OR gates, AND, OR and NOT gates.

So, any function which has only Boolean variables and only Boolean operations would be called a Boolean function. And similarly, the expression which will have only Boolean variables and Boolean operations would be called expression. So, in this term, this part can also be called as Boolean expression, when you are saying that the output of this Boolean expression is equal to another variable, then I will call it a Boolean function. So, Boolean function would return 1 Boolean variable as a output.

So, another definition, truth table so when we want to represent a Boolean function, so if we are able to drive, if we are driving all possible values of A Boolean function for all possible inputs combination, then it is called truth table. So, if you want to say that two expressions are same, if we want to say two expressions are same, how to prove that the two expression, two Boolean expressions are same, if truth table for both the expression is same that means, for all the combinations, it is going to give same results so that means they both the expressions are actually same. So, let us understand this with an example.

(Refer Slide Time: 13:57)




So, let us say we have this expression. So, let us consider it independently, first expression we are saying is this, AB dash C, the other expression we consider is this A plus C B dash plus C. So, let us see how to design or how to write this Boolean expression in form of a truth table. So, for truth table, I need to write, so how many Boolean variables are there? So, there are 3 Boolean variables, one is A, another is B and the third one is C. And this Boolean expression is equal to F. So, that means this F is another Boolean variable which is actually the result of this function.

So, what I can do is, I can write a table in which I enumerate all possible values of A, B, and C. And then, because I want to evaluate this function F, so what I would like to do, I would say that what is the value of let us say A B dash and I will also then calculate A B dash plus C. Let me calculate in, so let me draw all the vertical lines and horizontal lines so that we can enumerate all possible values of A, B and C and this is also equal to F. So, how to enumerate all the possible values of 3 variables?

Let us say first keep the values of first two variables as 0 and then say that the third variable would be 0 or it could be 1, so now this enumeration has completed, now we will come to the second variable B, so we will keep our first variable as 0 and we will keep our second variable as 1 and then we will again make both the possibilities of C as 0 as well as C as 1. Now, these 4 possibilities would repeat again with the value of A equal to 1. So, A 0 0 0 1 1 0 1 1 1.

So, we are doing some sort of a shortcut because we are calculating A B dash, so we are combining two operations, so NOT of B we are calculating and then we are ANDing it with

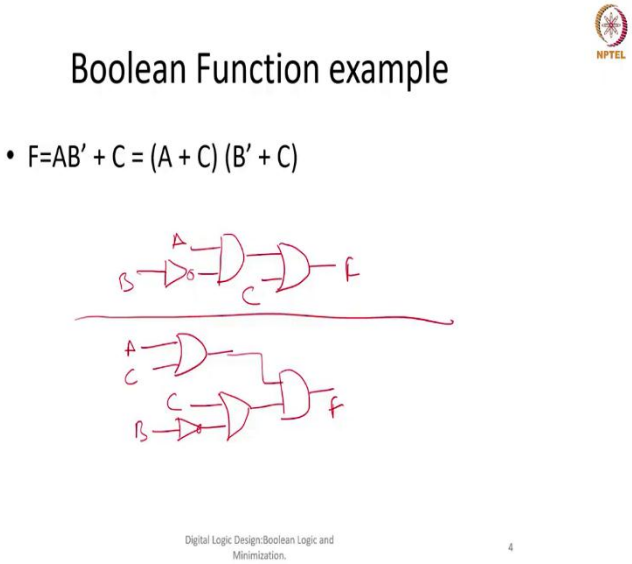
A. So, NOT of B is 1 and 1 AND with 0 is 0. Now, again NOT of B is 1 and 1 ANDed with 0 is 0. So, here NOT of B is 0 and 0 ANDed with 0 is going to be 0, so they also going to be same. Now, here, NOT of B is 1 and A is 1 so, 1 AND 1 is 1, similarly AND of 1 AND 1 is 1. Here, NOT of 1 is 0, 0 ANDed with 1 will become 0, here also 0 ANDed with 1 will become 0.

Now, similarly, now we will do a OR operation of AB dash and C. 0 and 0, OR of 0 and 0 is 0, OR of 1 and 0 is 1, OR of 0 and 0 is 0. OR of 1 and 0 is 1. OR of 0 and 1 is 1. OR of 1 and 1 is 1. And OR of 0 and 0 is 0, OR of 1 and 0 is 1. Good. So, let us extend this for the other part. So, if I want to compute this, how will I do? I will first do A plus C and then I will do B dash plus C.

A plus C so, I need to see this column and this column, A plus C, 0 OR 0 is 0, 0 OR 1 is 1, 0 OR 0 is 0, 0 OR 1 is 1. Now again, 1 and 0 is 1, 1 and 1 is 1, 1 and 1, sorry 1, yeah 1 and 1. Let us see B dash plus C. Now, B dash is 1 and C is 0, so this will become 1. B dash is 1, C is 1, this is also 1. B dash is 0 and C is 0 so this will become 0. B dash is 0 and C is 1, so output is 1. B dash is 1 and C dash, C is 0, so this is again 1, B dash is 1, C is 1, this is also 1. B dash is 0, C is also 0, this will become 0 and now B dash is 0, but C is 1, so this is 1.

Now we are going to AND this and this, so that means this is A plus C dot B dash plus C. Now, we will AND these two columns. So, 0 AND 1 is 0, 1 AND 1 is 1. This is 0, 1, 1, 1, 0, 1. So, if you see for all combination, this value is same as this, 0 is equal to 0, 1 is 1; 0 is 0. There are 3 1s, there are 3 1s, this is 0, this is 0, this is 1, this is 1. So, I can say that A plus C dot B dash plus C is also equal to F. So, this is how we can represent all of these and we can also calculate Boolean functions, Boolean.

(Refer Slide Time: 21:36)




Now, if I want to represent, so if I want to show this whole circuit using gate, so whole expression using gate, so I would write, I will first of all will make inverter of B and then I will AND it with A and then I need to OR it with C and then the output would be F. In the similar way if I want to do this, then I will say first of all I need to do OR of A and C and then B need to be inverted and then I need to do an OR with C and then both of the output of these OR gates has to be ANDed with and the result would be F. So, essentially these two circuits with this logic gate and this logic gate we have found that both of them are actually equivalent.

So, with the summary of this, so basically using basic expressions, we can represent different kind of functions, different kind of Boolean functions or Boolean expressions and truth table is also one of the good way to represent any kind of a Boolean expressions or a Boolean function because it is a you can put all the combinations, so it is more like a it is a complete expression of that particular function.

Now, why are we doing all of these things? Why we are talking about Boolean expressions or Boolean functions and Boolean gates? So, and also we have seen here that there could be many expressions it could be same, like these two expressions, both of these two expressions are same. So, what is the purpose of this or what do infer from this? So, one inference we can clearly take out is that same function could be represented in a multiple ways.

Now what would be the, what would be a good way or what would be the best or easiest or maybe the smallest, optimized way of representing an expression would be the best suited for designing a circuit? So, this would be our motivation for this lecture and maybe next couple

of lectures and actually this whole module which would be focusing on Boolean logic and minimization.

So we will also see that how to minimize a Boolean expression. So, how to remove those redundancies. So, in this lecture and the next lecture, we will see how we can do it using the expressions itself. So, using applying the basic Boolean algebra properties. So, to understand those properties, let us go through those basic properties of Boolean algebra.