

Digital Circuits and Systems
Prof. Shankar Balachandran
Department of Electrical Engineering
Indian Institute of Technology, Bombay
And
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Module - 3
Boolean Theorems, Binary Numbers

Welcome to module 3 of this course. In this module, we will learn basic Boolean theorems. So, from module 2, we will do a few more theorems and I will also introduce you to the concept of binary numbers.

(Refer Slide Time: 00:34)

DeMorgan's Theorem

$$\overline{(X_1 + X_2 + \dots + X_n)} = \overline{X_1} \cdot \overline{X_2} \cdot \dots \cdot \overline{X_n}$$
$$\overline{(X_1 \cdot X_2 \cdot \dots \cdot X_n)} = \overline{X_1} + \overline{X_2} + \dots + \overline{X_n}$$

Remember: $\overline{X \cdot Y} = X + \overline{Y}$
 $\overline{X + Y} = \overline{X} \cdot \overline{Y}$

Digital Logic Fundamentals

So, let us start with DeMorgan's theorem. This is something that you must have seen even in set theory. And this is a very useful theorem to know and you will see it used in several places. So, DeMorgan's theorem basically says that, if you have n terms of the form let us say X 1 or X 2 or X 3, so on up to X n. And if you take the complement of that; so that is what you see in the topmost line here. The complement of X 1 or X 2, so on up to X n. It is a same as X 1 complement and X 2 complement and so on up to X n complement. Similarly, the dual of that is if you take AND of X 1, X 2, so on up to X n and if you take the complement; it is a same as the logical or of X 1 bar plus X 2 bar plus so on up to X n bar. So, these two... You can imagine this one; for instance, if you substitute plus with union and bar with complement and AND with intersection; you

have the set theory. In set theory, you must have learnt DeMorgan's theorem. So, remember – you have keep to this in mind – if I have two terms of the form X and Y and if I have X and Y; if I take the complement of that; it is not the same as the AND of the complements. So, complement of the AND is not the same as AND of the complements. Similarly, complement of the OR is not the same as logical OR of the complements. So, keep this in mind, because this is a common and usual mistake that people do. So, keep this in mind however.

(Refer Slide Time: 02:21)

Proof Techniques (Example 1)

■ Prove $x + (y \cdot z) = (x + y) \cdot (x + z)$

RHS = $x \cdot (x+z) + y \cdot (x+z)$
 $= x \cdot x + x \cdot z + y \cdot x + y \cdot z$
 $= x \cdot (1+z+y) + y \cdot z$
 $= x \cdot (1) + y \cdot z$
 $= x + y \cdot z$

Fairly Straightforward

So, let us look at a few basic proof techniques. This will come in quite handy. So, we have seen Boolean algebra; we used bit of manipulations in the last module. We will see a few more of these things, because these things come in handy when you design circuits later. So, we will deal with algebra for some more time before we jump into circuits. So, let us start with the first one here. So, I have several such examples that follow. So, let us start with this – x plus y and z is x plus y and x plus z. So, this is actually the distributivity property. But, let us see how to prove something like this. So, in the last module, I showed how to the prove some of these things using truth tables explicitly. But, I want to be able to do this just using algebra, because truth table is laborious and you have to be very careful with all the 0's and 1's and so on.

Let us see if we can do it with basic algebraic techniques and the rules that we know already. So, let us start with this x plus y and x plus z. So, I start with the right-hand side; you can rewrite it has x and x plus x or z or y and x or z, because you have these two terms and you have these two terms. In some sense, it is like expanding the expression,

which you do in regular algebra. So, now, you have x and x plus z ; you can write it as x and x or x and z . So, essentially, you are taking the product into the terms here. So, that gives you four terms here. And if you look at these three terms here; there is x , which is common in all of these. So, you can rewrite it as x and... So, this gives you 1; this gives you z ; this gives you y . So, 1 or z or y . And you have y and z , which we keep as it is. So, we know that, 1 plus any variable or any expression for that matter is actually 1. So, in this case, z plus y is an expression. So, 1 or z or y is actually 1. So, we have that here. And finally, x and 1 is x itself. So, this is the same as what we have in the left-hand side. So, hence, this is correct.

So, the key thing that we did was here – this is fairly like what we do with regular algebra. So, if you have sum of two variables and sum of two variables and so on; expanding up to here is probably straightforward. And taking a common term out for AND is probably also straightforward. So, the only thing is we have 1 plus z plus y , which simplifies to 1, because this is Boolean logic. So, 1 or z or y is 1. So, this is a fairly straightforward example; where, this is much more easier than all the tables than other things that I did earlier. Let us look at a slightly different examples to see all the different Boolean techniques that we need to know.

(Refer Slide Time: 05:24)

Proof Techniques (Example 2)

■ Prove $x + x \cdot y = x$

$$\begin{aligned} x + x \cdot y &= x \cdot 1 + x \cdot y \\ &= x \cdot (1 + y) \\ &= x \cdot 1 \\ &= x \end{aligned}$$

Introduce a constant

So, I want to be able to show x or x and y is x . So, this is actually called absorption rule – x or x and y is x . Let us see how to prove this. Again we start from this case the left-hand side. So, one usual technique is try and introduce a term; in this case, x is ANDed with 1;

sometimes you OR it with 0. So, here we want to simplify x or x and y . So, if you introduce a term and 1; it does not change the value of the expression. Now, once you do this; then you have x , which is common to both the terms. So, you can take x out. So, this expression here is the same as x and 1 plus y or 1 or y . So, 1 or y we know is 1. So, x and 1 is x . So, in this proof, what we have used is we have introduced a constant here. So, we know that, x and 1 is x . So, if I give you this one; you would easily write it as x ; but, sometimes it is useful you go and take something like x here and write it as x and 1. So, in this case, we can see that, it brings x as a common term between these two.

(Refer Slide Time: 06:48)

Proof Techniques (Example 3)

■ Prove $x.y + y.z + x'.z = x.y + x'.z$

LHS

$$\begin{aligned}
 &= x.y + y.z.1 + x'.z \\
 &= x.y + y.z.(x+x') + x'.z \\
 &= x.y + y.z.x + y.z.x' + x'.z \\
 &= x.y(1+z) + (y+z').x'.z \\
 &= x.y + x'.z
 \end{aligned}$$

Introduce a constant and introduce a function

NPTEL

Let us see another example here; x and y plus y and z plus x complement and z is a same as x and y plus x and z . So, what we have is we have these two terms, which appear as it is on the right side. And magically, one term seems to vanish. So, this is what we are going to show. So, this is actually... So, this is a slightly tricky thing and this is called the consensus theorem by the way. So, x and y plus y and z plus x bar and z is the same as x and y or x bar and z . So, let us see how to start with this. So, you have this expression here. So, the first thing I am doing is I am taking out this term and ANDing 1 with that. So, that is correct because 1 and any term is the same as the term itself. So, here we have 1 and yz is the same as yz itself. Then, I take this 1 and use the previous technique. I introduce a constant; but, I am replacing the constant with the expression here. So, earlier I used the constant AND; I took something out in common. Here I am replacing 1 and I am introducing a function. So, we know that, x plus x complement is 1. So, I did that here. And once you have this; I can separate this term and this term. So, I

have a product here and I have a sum x or x complement. What I am going to do is I am going to take this out. So, y and z and x . And similarly, y and z and x bar here. So, this and that gives this term here.

And now, we have these two terms, which can group; and these terms, which we can group. So, there are common things here. You can see that there is x and y here; there is x and y here. So, 1 plus z is the expression, which comes from these two. Similarly, we see that, there is x bar z , which is common to this term and this term. So, we can take that out and we have y or 1 . We know that 1 or z is 1 and y or 1 is 1 . So, it simplifies to xy and xy or x bar and z . So, what happens is this term vanishes and you have xy or x bar and z . So, in this example, what we have done is we have introduced a constant here and we have introduced a function in terms of the constant. Usually we do the other way round; if I give you the expression x or x complement; I would expect you to write 1 . But, sometimes it is useful to take 1 and expand it to a function. So, this is a bit of... This takes a bit of work to understand and appreciate; it is not easy to come up with something like this in the first go; I do not expect you to do it right from the beginning. So, this case, I know that, I want to get rid of yz . So, I have to do some manipulation in that term. So, I could have started here; I could have started here; it could be any of those; but, I started with the middle term, because I wanted to get rid of that term. And when I wanted to get rid of that; so it is a y and z .

So, it is natural to put a 1 there and expand it in terms of something that is there on both the left side and right side. So, if you look at the left side, there is x ; and right side, it has x . Left side has y ; right side has z . So, the reason why I wrote x plus x complement here is it looks like there is some function of x which is here and some function of x which is here. So, there is x bar here and x here. So, I rewrote it as x plus x bar. And then the rest of these steps follow. So, this is not something that I expect you to do right in the beginning; but, it is good to know some of these tricks, so that you can employ them later.


(Refer Slide Time: 10:36)

Boolean Techniques (Example 4)

■ Simplify $x \cdot y + x \cdot y \cdot z + y \cdot z$

$$\begin{aligned} \text{LHS} &= x \cdot y + x \cdot y \cdot z + y \cdot z \\ &= x \cdot y + x \cdot y \cdot z + x \cdot y \cdot z + y \cdot z \\ &= x \cdot y (1 + z) + y \cdot z (x + 1) \\ &= x \cdot y + y \cdot z \\ &= y \cdot (x+z) \end{aligned}$$

Introduced a copy of a term

 A simpler derivation is there. Think about it!

Introduction 6

Let us look at another proof. So, we have x and y or x and y and z or y and z . So, the LHS is this expression. This is another common trick that can come in very handy at times. So, I am going to take this term here – x and y and z and make a copy of that. Why can I do that? I can do that because I know that, from the single variable theorems, x plus x is x ; which means even if I have n copies of x all or together, is the same as the term x itself. So, if you take the term x and y and z ; even if I make multiple copies of those; it is the same as one term itself. In this case, we are doing the other way round. We take one term and then we take multiple copies of that. In this case, two copies of it. And once we do that; we can see that, x and y is common to these two terms; and y and z is common to these two terms; and this will ((Refer Slide Time: 11:33)) simplify things. So, x and y and 1 or z – that is...

So, this is 1 or z goes away; x or 1 goes away. So, it gives us x and y plus y and z ; and y is common between these two terms; I can write it as y and x plus z . So, in this example, we introduced a copy of this term x and y and z and we got to y and x plus z . So, there is a much simpler way to do this; you go and think about that. But, I wanted to show these examples, so that you know this particular technique that, in Boolean algebra, it is okay to copy terms like this. So, if you do this in regular algebra; if I give you x plus a ; you cannot go and write it as x plus a plus a . So, this is not true for all values of a . However, in Boolean algebra, if I give you x plus a ; it is okay to go and write it as x plus a plus a , because you have a plus a is a ; we know that already.

(Refer Slide Time: 12:31)

Exercise :
Prove the Two Variable Theorems Below

- Absorption
 - $x + (x \cdot y) = x$
 - $x \cdot (x + y) = x$
- Combining
 - $x \cdot y + x \cdot y' = x$
 - $(x + y) \cdot (x + y') = x$
- DeMorgan's Laws (You probably know how to use Venn Diagram to prove this)
 - $(x + y)' = x' \cdot y'$
 - $(x \cdot y)' = x' + y'$

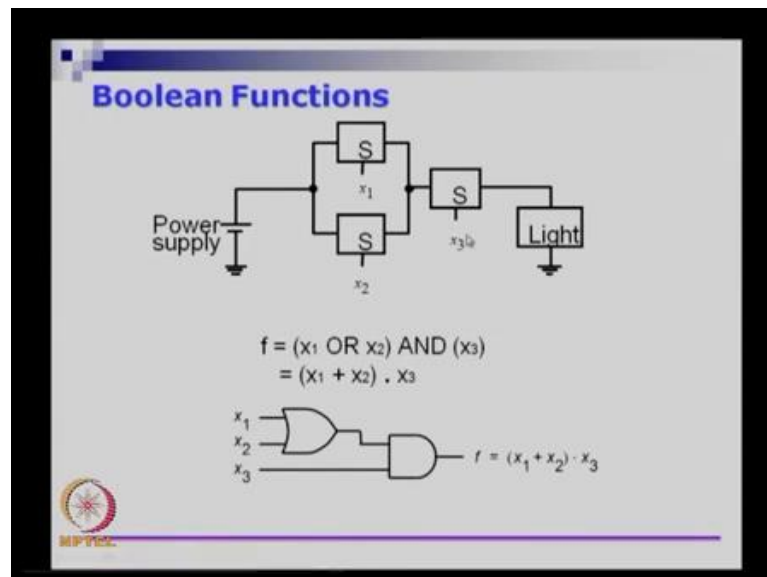
$x + (x' \cdot y) = x + y$
 $x \cdot (x' + y) = x \cdot y$

MPPTC 7

So, let us look at a few exercises; this is something that I want you to do later. So, I am

not going to teach this; but, I suggest that, you go and prove this later. So, we have the absorption law, the combining law, and DeMorgan's laws. So, I suggest that, you prove it either using Boolean algebraic techniques or at least using truth tables. Once you get comfortable with truth tables, maybe you can move to algebraic techniques; but, it gets easy as you go along. DeMorgan's law of course, you can prove it with Venn diagrams also; but, typically, we avoid proving with diagrams when we do Boolean algebra, because doing it with the few variables is okay if it goes beyond a few variables and gets complicated to do Venn diagrams. But, I give this as an exercise to you. Go and do this. So, let us go back to Boolean functions.

(Refer Slide Time: 13:25)



So, we looked at this circuit yesterday; where we have a power supply and a light and we have two switches: x_1 , x_2 in parallel and x_3 , which is in series. And if I give you such as switch; we argued that, either x_1 or x_2 should be on; and x_3 must be on definitely for the light to glow. If you want to write this as an expression; so you have let us say f is the expression for whether light is on or off. So, f is 1 if the light is on; f is 0 if the light is off. If you want to be able to write expressions for these; so let us go back to what I said in English. Either x_1 or x_2 must be on. So, we know that, the expression – the way to write that is x_1 or x_2 . And because of the series connection between this circuit and this circuit here; we need AND x_3 on. So, if x_3 is off, we know that, light cannot turn on. So, we have AND x_3 . So, you can see the parenthesis also, so that it shows how things are grouped.

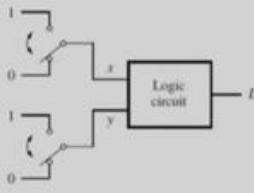
So, now I am going to write it as a Boolean expression; instead of using the word OR

and AND, I am going to use the symbols that I introduced in the previous module. So, x_1 or x_2 is x_1 plus x_2 ; and AND is replaced by a dot. So, x_1 plus x_2 and x_3 . So, if you want a circuit out of this; we know that, x_1 or x_2 – you have an OR gate; you put x_1 and x_2 as the two inputs; the output will be x_1 or x_2 . And x_3 here is ANDed with this term here. So, we put an AND gate and add these two terms here. So, this is supposed to give us the expression that we need. So, if I... We can mentally simulate it also. Let us say these are both 0; and let us say this is 1; which means both these switches are off and this is turned on. So, if I put a 0 here and a 0 here; this will be a 0 here because 0 or 0 is 0; and 0 with a 1 here; 0 and 1 will be a 0. So, the lamp will not turn on. However, if I have one of them to be a 1 here; let us say I have one of them to be a 1; then the output of OR will be a 1. I still need x_3 to be 1, so that the light turns on. So, you can also mentally simulate and check – given a circuit, whether it is trying to do what you want to do or not.

(Refer Slide Time: 15:51)

Design Problem 1

- Design a circuit for staircase light control using a two-way switch
- If both switches are OFF or ON, light is off
- If one of them is ON and another is OFF, the light is ON



x	y	L
0	0	0
0	1	1
1	0	1
1	1	0

(a) Two switches that control a light (b) Truth table

So, let us look at a slightly different problem now. This is a design problem. So, this is a common problem if you have a stairway or a staircase and you have two switches. This is something that you probably have in your homes. So, you have a switch in the – let us say in the ground floor and you have a light that you want to turn on for the staircase; either you can turn it on from the ground floor or from the first floor. So, this is a two-way switch. This is something that you probably have in your homes. So, we want to be able to design a circuit, which takes care of that. So, what we want to do is we have two switches that are there; we want that to be connected to a logic circuit in such a way that,

if both the switches are off or on, the light is off; if one of them is on and the other one is off, the light is on. So, this is fairly straightforward. So, you might see this in staircases; you also see that in reading lamps and so on. So, sometimes you have light by the bed side and you want to turn on or off the tube light by the switch that you have in the bed side. So, this is a common thing that we have at houses. So, let us see how to think about this. If both the switches are off or on, the light is off; if one of them is on and one of them is off, the light is on. So, we want to be able to think in terms of expressions for these or... So, we will start with the notion of a truth table for this. So, what we have is we have this expression – if both the switches are off.

(Refer Slide Time: 17:32)

	x	y	L
→	0	0	0
✓	0	1	1
✓	1	0	1
→	1	1	0

So, I am going to assume that, there are two switches x and y. And I want to be able to design the logic for L. Let us see how to go about writing a table for this; if both the switches are off or on. So, let us see all the possibilities that x and y can take 0 0, 0 1, 1 0 and 1 1 are the four possibilities for x and y. And what we want is when both of them are off, the light should be off. So, that is this row. When both the switches are on, the light must be off. So, this takes care of the first bullet. Let us look at the second constraint; if one of them is on, the other one is off; that is true for both of these rows. One of them is on, the other one is off. In that case, I want the output to be 1 or the lamp to glow. So, this is the truth table that we want. Sometimes you are given a problem in... So, the real world problems are always in English descriptions; you want to be able to take that and move it to tables. So, that is what we have done now. We have a truth table for this function. So, this is the same table

as what I did on paper. And this is the first time. Let us say I want to realize a circuit, which does that. So, I have a truth table; this does not give you a circuit.

(Refer Slide Time: 18:54)

Boolean Expression

- Lamp is ON when
 - Either x is ON and y is OFF
 - Or x is OFF and y is ON
- $L = x \cdot y' + x' \cdot y$

MPTEL

So, I want to be able to get a circuit, which does that. So, the first thing I want to be able to do is I want to write a Boolean expression for this. So, let us see how to write a Boolean expression for something like this.

(Refer Slide Time: 19:07)

Mod 3
Slide 9

x	y	L
0	0	0
✓	0	1
✓	1	1
→	1	0

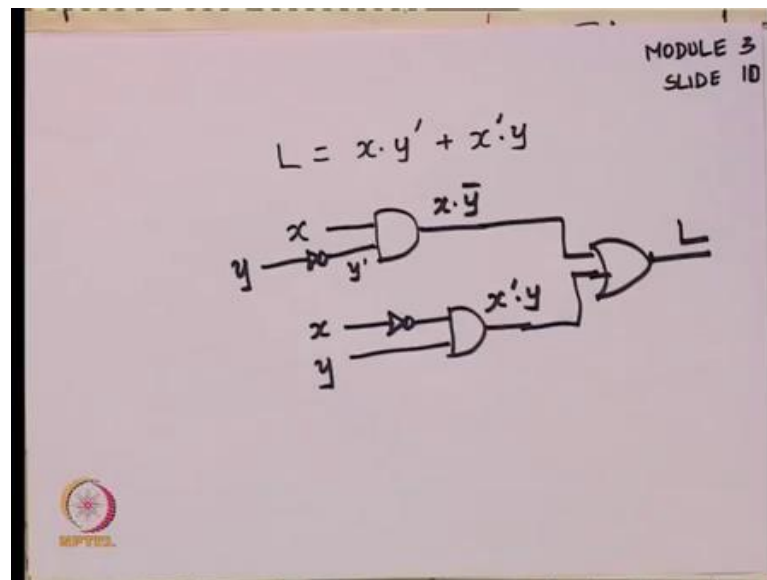
$x \cdot y'$
 $+ x' \cdot y$

MPTEL

So, what we do is we start from the right side. So, we start... We look at the output; we look at all the places, where we have a 1. So, there are two rows: this row and this row here, which have a 1. And what we are going to do is wherever there is a 1, we go and

looked at the left side. So, the left side is either we have 0 1 or 1 0. So, if you want to read it in English, lamp is on when either x is on and y is off or when x is off and y is on. So, let us look at the first condition – either x is on; x is on can be checked by checking the switch on x; and y is off; either that condition must be true or x must be off and y must be on. So, this term comes from this row here. So, x is on and y is off. And this term comes from this row here; x is off and y is on. So, now, from a truth table, we have an expression. And once we have an expression, we can do several things. So, the first thing is once I have an expression, I may want to realize a circuit out of this. So, let us go and try and realize the circuit out of this.

(Refer Slide Time: 20:42)



We have the expression x and y complement or x complement and y . I want to realize a circuit for this. Let us see how to realize a circuit for x and y complement first. x and y complement. So, if I give you y as input, I cannot use y directly; I want y complement; I know that, y complement can come if I have a y and if I stick an inverter there. So, this will be y complement. So, this is x and y complement. Similarly, if I take x , take its complement and AND it with y ; this gives me x complement and y . So, this takes care of the two terms; but, we still have this logical OR in between; I can do that by putting an OR gate here. So, this is the circuit, which can realize the functionality that we need it. So, what we have is the logic network that we need.

(Refer Slide Time: 21:44)

Boolean Expression

- Lamp is ON when
 - Either x is ON and y is OFF
 - Or x is OFF and y is ON
- $L = x \cdot y' + x' \cdot y$

(c) Logic network

(d) XOR gate symbol

Or, the circuit that we need is essentially slightly redrawn version of this. So, in the paper, I had two different x inputs and two different y inputs; here I am showing that, it is actually coming from the same inputs. So, x... This line goes as x and this comes as complement. And if you notice, there is a filled circle here. In electrical circuits, we use that to denote branches. So, you probably know that, from the basic electric circuits that, when you have a dot on a line, it means that you are branching. So, x is branching here and branching here. This line comes as x complement; this line goes as x. Similarly, for y here, this line goes as y; this line goes as y complement.

So, this one is for x and y bar; this is for y and x bar; and that is 1. And this is such a common thing that we have a symbol for it by itself. This is called the XOR gate. So, the XOR stands for exclusive OR – EXOR or XOR – stands for exclusive OR. Exclusive OR means if I have two inputs; exactly one of them must be 1, the other one must be 0. So, previously, we looked at OR – the logical OR, which is also called inclusive OR. So, it allowed the case where if both the inputs are 1, the output is 1. Whereas, in XOR, if both the inputs are 1, the output is 0. So, exclusive OR essentially says exclusively one of the inputs must be 1, which automatically means the other input is 0. So, what you want for this two-way staircase or this bedside lamp – bedside switch for a tube light, is essentially an XOR gate; it is as simple as that.

(Refer Slide Time: 23:28)

Design Problem 2

- Design a circuit which realizes the following truth table

x_1	x_2	$f(x_1, x_2)$
0	0	1
0	1	1
1	0	0
1	1	1

$f = ?$

MPPVSL

Let us look at another design problem – design a circuit, which realizes the following truth table. So, we have two inputs: x_1 and x_2 ; and we have one output: f of $x_1 \times x_2$. There are four combinations that x_1 and x_2 can take. The first thing I want you to think about is what is the way in which we can write this expression down. So, I have written the same thing on paper here.

(Refer Slide Time: 23:56)

MODULE 3
SLIDE 11

x_1	x_2	f
0	0	1
0	1	1
1	0	0
1	1	1

$$f = (\bar{x}_1 \cdot \bar{x}_2) + (\bar{x}_1 \cdot x_2) + (x_1 \cdot x_2)$$

MPPVSL

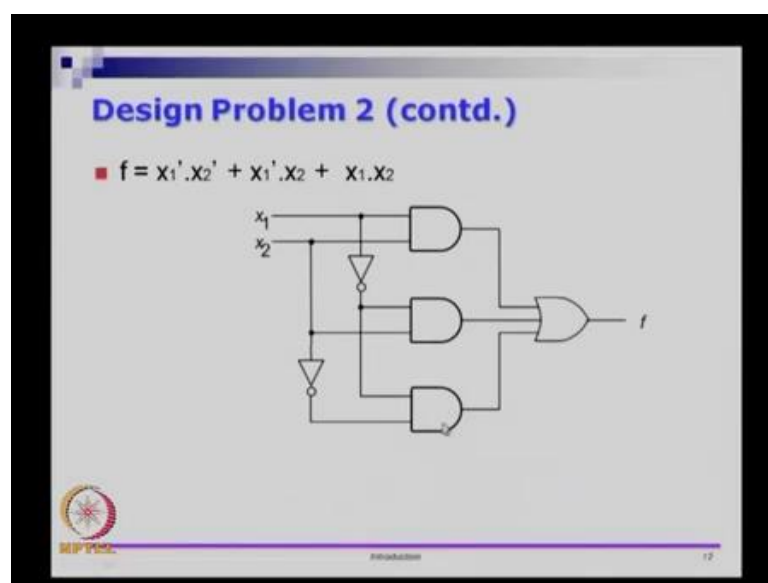
So, we have four terms: 0 0, 0 1, 1 0, 1 1; there are four combinations of x_1 , x_2 and I have f . So, just like what I said earlier, if I want to write a Boolean expression for this; we want to be able to look at all the 1's here and look at the corresponding terms on the left side and combine them properly. So, let us start with the row here. In this row, there

is a 1. And what is the combination of x_1 x_2 that gives a 1 there; x_1 must be off and x_2 must be off; correct? So, that gives you 0 0 here; then let us get to this row. So, this gives you the case, where x_1 must be off and x_2 must be on. And finally, this row says x_1 must be on and x_2 must be on.

So, we have three 1's here; these three 1's correspond to these three terms here. So, one thing you have to do is we know that, either if this input is given or if this input is given or if this input is given, the output must be 1. So, we stick plus in between these terms. That is the Boolean expression that we need. So, the Boolean expression for f is x_1 bar and x_2 bar; or, x_1 bar and x_2 or x_1 x_2 . So, eye-balling the truth table and being able to write Boolean expression like this is very useful.

So, sometimes, you will be given the Boolean expression; you have to write the truth table; but, more often they are not coming up of the truth table is fairly easy to write and then there are the techniques to go and write it as Boolean expressions and do several things with it. So, this is a very useful skill. Start from a row, which has... So, look at all the rows, which have a 1; and wherever there is a 1, go and look at the corresponding left side and come up with a term like this, which looks at the correct conditions, which will make them a 1. So, if x_1 is 0 and x_2 is 0, the output must be 1. That is what this one captures and so on. So, this is the same thing that I used in the previous example. For XOR also I used the same trick. This is a slightly different example; that is all. So, to implement this circuit; so it is a...

(Refer Slide Time: 26:20)



We have the Boolean expression here and we have three terms here. We need three AND gates for that. And we need to be able to combine three of these inputs using OR. So, instead of using a two-input OR, I have used a three-input OR gate here. A three-input OR gate can take three inputs and the output will be 1 even if one of the inputs is 1. So, even if one of these wires – if they have a 1, then the output will be a 1. The rest with these two inverters are just used to get these corresponding terms and so on. So, if you look at the circuit carefully, you will realize that, this term is capturing x_1 and x_2 . So, that is this; then this one is capturing x_1 complement and x_2 ; and this one is capturing x_1 complement and x_2 complement. And these three are logically OR together to get the final output.

(Refer Slide Time: 27:12)

Power of Simplification

$$\begin{aligned}
 f &= x_1'x_2' + x_1'x_2 + x_1x_2 \\
 &= x_1'x_2' + x_1'x_2 + x_1'x_2 + x_1x_2 \\
 &= x_1'(x_2' + x_2) + (x_1' + x_1)x_2 \\
 &= x_1' \cdot 1 + 1 \cdot x_2 \\
 &= x_1' + x_2
 \end{aligned}$$

Minimal-cost realization

The diagram shows a logic circuit with two inputs, x_1 and x_2 . Input x_1 passes through an inverter. The output of the inverter and input x_2 are connected to an OR gate. The output of the OR gate is labeled f .

So, sometimes it is useful to go and simplify this. We will take the expression that we had earlier. x_1 complement AND x_2 complement plus x_1 complement AND x_2 plus x_1 x_2 . So, this is the expression that we had earlier. So, now, I am going to try and simplify this; it looks like it is a big expression; I am going to try and see if I can simplify it. So, this is x_1 complement and x_2 ; I am going to take this term, which is in the middle and make two copies of it. So, I showed this earlier to you. I made two copies of this. And for this part, x_1 complement seems to be common; and for this part, x_2 seems to be common. So, once you have that, then we have x_2 complement plus x_2 ; that is the same as 1; x_1 complement plus x_1 is same as 1.

And we know x_1 complement and 1 is x_1 complement; 1 and x_2 is x_2 . So, this is a much simpler expression than what you have here. So, the way to interpret this

expression is... So, we want to be able to sometimes read in English and I understand whether what you have done is correct or not. Yes, I did start with this table here in slide 11; I started with this table here; but, I want to be able to take this expression here and understand whether it is correct or not. So, what does this expression say? Either x_1 must be false or x_2 must be true. So, that is what this one says; either x_1 must be false or x_2 must be true. Let us go back and check the table. Wherever there is a 1, either x_1 must be false or x_2 must be true.

So, if I take this row, x_1 is false. So, this is a 1. In this row also, x_1 is false. So, this is a 1. So, this is okay. In these two rows, x_1 is false is not correct. So, x_1 is actually true in these two rows. However, x_2 is true only here. So, that row gets a 1 and the other row gets a 0. So, again, x_1 must be false; this is true for these two rows. So, automatically, they became a 1. x_1 is false is not correct for these two rows. In this case, we expect x_2 to be 1 or true. And therefore, only this row gets a 1; the other row gets a 0. So, this is a fairly quick way to check whatever expression you have is it what you want from there. So, sometimes it is useful to think in English and go and justify that, the Boolean expressions are correct or not.

And once you have this, it is not just a mental exercise; it actually does something to the circuit. So, what it did is if we take x_1 complement or x_2 ; now, x_1 complement requires one inverter and we have a logical OR here that requires one OR gate. So, it looks like we used one OR gate and one inverter as supposed to the complicated circuit that we had earlier. So, if I charge you; let us say 1 rupee for each of these gates. Then, you will pay me 2 rupees: one for the inverter and one for the OR gate here. Whereas, if you did not minimize, you have two inverters, three AND gates and one OR gate; you would have paid me 6 rupees for it. So, in engineering, it is always about cost. Sometimes its cost is directly related to money. In this case, if I go and ask you to buy these components from a shop, if you did not minimize, you would have spent 6 rupees to do this circuit. If you minimized it, you would have spent only 2 rupees on the circuit. So, it is good to think about whenever you want to minimize, you are actually cutting down the cost in some sense. In this case, it is the count of the number of gates.

(Refer Slide Time: 30:51)

Positional Number System

■ Decimal: Uses base 10

Note: base 10 → $\times 10^2$ $\times 10^1$ $\times 10^0$ First column starts at "1"

7	5	6	=	$7 \times 10 \times 10 +$
				$5 \times 10 +$
				6×1

Hundreds Tens Ones

■ Binary Number: Uses base 2

Note: base 2 → $\times 2^2$ $\times 2^1$ $\times 2^0$

1	1	1	=	$1 \times 2 \times 2 +$
				$1 \times 2 +$
				1×1

2^2 2^1 2^0

MPPCS

Introduction

14

So, I want to do a slight switch from the Boolean expressions and give a very simple description of what number system means, because this is useful in the later modules. So, let us look at a decimal number like 756. So, what do we do? We start with hundreds, tens and ones. We usually start from the... If I give you a bigger number, you do not start from the left side; if I give you 756, you can read it a seven hundred and fifty six. If I give you 7564, you may read it as seven thousand five sixty four; but, if I give you 10 digits, you do not usually start from the left side. Usually start from the right side; you say that, right-most digit is for ones; the next digit is for tens; then it is hundreds, thousands ten thousands and so on.

So, if I give you this, in the decimal system, we have the positions; and the positions come with certain weight; the position – the rightmost position has a weight of 10 power 0 or 1. The position slightly to the left of it has a weight of 10 power 1 or 10; the position to the left of that has a weight of 10 squared or hundred and so on. So, if I actually want the value 756; so it is equivalent to 7 into 10 into 10 plus 5 into 10 plus 6 into 1. So, that is what gives me 756. So, this is a system, where we have used 10 as the base. So, the base is 10; which means the weights that we are using are all increasing powers of 10 – starting with 10 power 0, going to 10 power 1 and 10 power 2 and so on.

If you have the decimal let us say 756.4; then when you cross the dot, it actually goes as 10 power minus 1, 10 power minus 2 and so on down. The same thing can be done with so-called binary numbers. If I give you a number 1 1 1; first of all, a binary number can have only 0 or 1; it cannot have anything else. In this example, I have a binary number,

which has three 1's – 1 1 1. And I want to be able to say what this binary number is equivalent in decimal. If I want to do that; if I want to find out the weight of the actual value of the binary number 1 1 1; just like what we had here; we assign a weight of 2 power 0 for the last bit; assign a weight for 2 power 1 here and assign a weight for 2 power 2 here. So, the binary equivalent – the decimal equivalent of this is 2 squared into 1 plus 2 power 1 into 1 plus 2 power 0 into 1. That is what we have here. It is a same as 4 plus 2 plus 1, which is 7. So, 1 1 1 written in binary is the same as decimal 7. So, this is something that is very useful; we will need it in the next few modules and so on.

(Refer Slide Time: 33:41)

Binary to Decimal

- Because humans work well with decimal, it is useful to know how to convert between binary and decimal:

0000 ₂ = 0	1000 ₂ = 8
0001 ₂ = 1	1001 ₂ = 9
0010 ₂ = 2	1010 ₂ = 10
0011 ₂ = 3	1011 ₂ = 11
0100 ₂ = 4	1100 ₂ = 12
0101 ₂ = 5	1101 ₂ = 13
0110 ₂ = 6	1110 ₂ = 14
0111 ₂ = 7	1111 ₂ = 15

So, I have given a whole list of things. If you have 4... So, just like saying a four digit numbers and five digit numbers and so on; we say 4-bit numbers here. So, in this left side, we are 4-bit numbers. And this list all the combinations of these 4-bits can have. So, 0 0 0 0 0 0 0 1 all the way up to 1 1 1 1. And the notation here – 2 is actually written as a subscript – 0 0 0 0 subscript 2; which means I want to interpret this number as a binary number. So, 0 0 0 0 base 2 is 0 into 2 power 3 plus 0 into 2 power 2 plus 0 into 2 power 1 plus 0 into 2 power 0. That is the same as 0. Let us look at this row here – 1 0 1 1. This is 1 into 2 power 3, which is 8 plus 0 into 2 squared, which is 0 plus 1 into 2 power 1, which is 2 plus 1 into 2 power 0, which is the same as 8 plus 2 plus 1, which is 11. So, we use the subscript to say that, this is in a particular base. So, for example, if I gave you something like this without the base; let us say I gave you 1 1 1 without the base; you may interpret it as number 111. Whereas, I really want to say it is the binary number 0 1 1 1. So, that is why we have this base explicitly here. 0 1 1 1 base 2 is not the same as 1 1

1, because 1 1 1 is decimal 111. So, you want to be able to distinguish the source. Same thing here for example. If I gave you 1 1 here; if I did not give the base, usually it is base 10. So, this is number 11; but, the same thing if I give 1 1 base 2; that will be 1 into 2 plus 1 into 1, which should be number 3. So, this is something that is good to keep in mind.

(Refer Slide Time: 35:35)

Binary to Decimal (contd.)

8	4	2	1		
1	1	0	1	=	$1 \times 8 + 1 \times 4 + 1 \times 1 =$ $8 + 4 + 1 =$ 13

To convert to decimal, represent the column values in decimal

128	64	32	16	8	4	2	1		
1	0	0	1	1	1	0	1	=	$1 \times 128 + 1 \times 16 + 1 \times 8 + 1 \times 4 + 1 \times 1 =$ $128 + 16 + 8 + 4 + 1 =$ 157

And if you want to go from binary to decimal, all you have to do is start with 1 and go in multiples of 2. That gives you the weights for... So, in this case, we have four numbers; you put four columns: 1, 2, 4 and 8. So, it starts with 1 and it goes in powers of 2 towards the left and take the number, take the bit, multiply, take the number; multiply with the bit and so on; and add all of that up; that gives you in this case 13. Similarly, here you have a 128 plus 16 plus 8 plus 4 plus 1. That seems to be 157. So, 1 0 0 0 1 1 1 0 1 is actually the binary representation of number 157. In a later class, we will see how to go from a decimal representation to a binary representation. This is something that you might have done in school; but, I will give you a quick refresher in a separate module. So, this brings me to the end of module 3. In this module, we saw some basic Boolean theorems. We also saw various ways of explaining or simplifying expressions by using little tricks. I showed you how to take a description of a problem and come up with the circuit for that. And finally, I showed you the number representation. So, many of these skills are useful later. So, what I would suggest is for you to go and review the material a few times, so that you know exactly how to solve some of these problems, which come up later.

Thank you very much.