

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

**Lecture - 12**  
**Boolean Algebra (Contd.)**

(Refer Slide Time: 00:18)

Karnaugh Maps - Four Variable K-Map

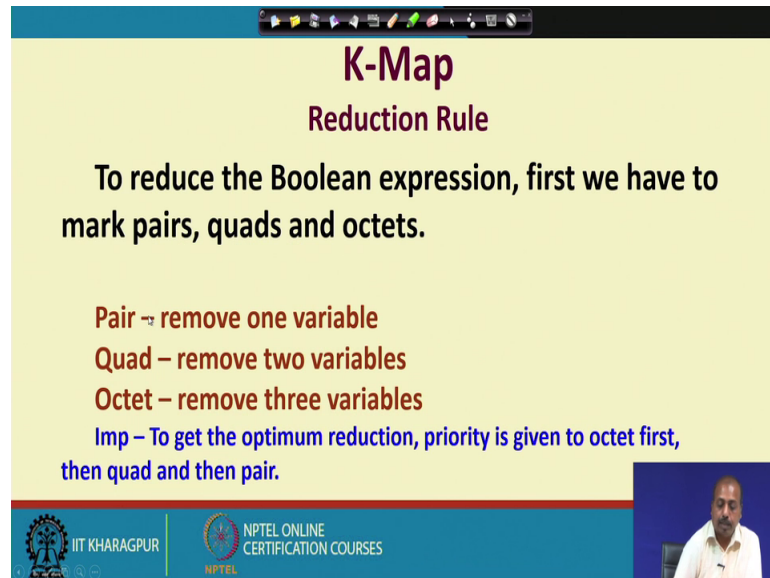
AB \ CD	00	01	11	10
00	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}C\bar{D}$	$\bar{A}\bar{B}CD$
01	$\bar{A}B\bar{C}\bar{D}$	$\bar{A}B\bar{C}D$	$\bar{A}BC\bar{D}$	$\bar{A}BCD$
11	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}\bar{C}D$	$A\bar{B}C\bar{D}$	$A\bar{B}CD$
10	$AB\bar{C}\bar{D}$	$AB\bar{C}D$	$ABC\bar{D}$	$ABCD$

C'D

D

So, Karnaugh map for 4 variables. So, you can understand what are we going to do. So, this is a in the row side and the column side we have got 4 rows and 4 columns, unlike 3 variable map where we have 4 rows 4 columns, but only 2 rows. So, since here we have to represent for 4 variables. So, we have got 4 rows and 4 columns in the row. We have got say we write it like this, so this diagonal line.

(Refer Slide Time: 00:51)



The slide is titled "K-Map Reduction Rule" and contains the following text:

**To reduce the Boolean expression, first we have to mark pairs, quads and octets.**

**Pair** → remove one variable  
**Quad** – remove two variables  
**Octet** – remove three variables

**Imp** – To get the optimum reduction, priority is given to octet first, then quad and then pair.

The slide also features logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES, and a small video inset of a speaker in the bottom right corner.

So, below we write the variables that will be represented by the variable that will be represented by the individual values of this rows. So, we have got this AB equal to AB vary. So, AB equal to 0 0 corresponds to the first row, AB equal to 01 corresponds to the second row like that. Or similarly on the column side so, CD equal to 0 corresponds to the first column CD equal to 01 corresponds to the second column so, like that.

So, we can understand the mean terms of the 4 variable functions, they are represented by different boxes in this 4 variable map. So, once we have represented the Boolean function by means of the Karnaugh map. So, we can follow some reduction rule some of that the rules, I have already said like we have to make grouping of this one's, ok. So, to reduce Boolean expression first we have to mark pairs quads and octets octet means the group of 8 ones quad means group of 4 ones, and pair means group of 2 ones.

So, if we make pair so, we can remove only a single variable, because between a pair only one variable can change it is a polarity. So, if the original function was a 4 variable function then if you are taking a pair. So, you will getting a you will be getting a 3 variable term out of that. Whereas, if you can make a quad. So, 2 variables will change it is polarity one in the row direction another in the column direction so, that way you can remove 2 variables. And if you are if you can make octet, ok. So, you can you can remove 3 variables because in a octet if you it is a 4 variable function then in the octet

the 3 variables will change their polarity only one variable will remain unaltered. So, that way you can do the grouping.

So, if you look into this one like if I can make an octet for example. So, this octet may be like this; so, if all these values are one then I so, all these values are one in my function, then I can make an octet. Now if you look into this octet what is happening is in the row direction, this a is changing it is polarity. So, it is going from A bar to A B is also changing it is polarity going from B bar to B and all. In the column direction so, C bar C is changing it is polarity it is going from 0 to 1. But only D is not changing it is polarity so, that way this entire octet will be represented by a single literal D if you can make quad say this one.

So, if I make a quad like this as I was telling that this a and B, both of them are changing their polarity. So, they will get eliminated what will remain is this CD bar. So, CD bar will remain as the term. So, if I can make so, it is beneficial so, if we can make quads we can make octets, for 4 variable function. So, if the octet is not possible you try to make this type of quads, and quads are also not possible then you try to make pairs. And the pairs are also not possible in that case we have to take as simple single variable.

Like it may so, happen that your ones are distributed like this. So, your ones are distributed like this. So now, you cannot do any pairing also, because you do not get one variable change over between the terms. So, you have to group like these individual terms only. And then there is no minimization that is possible, ok.

So, another important thing that we have is that while getting this octet or this quad or pair like that. So, you have to go in a particular row or a particular column only. So, you cannot make something like this. So, you cannot make a quad like this so, that is not allowed, ok. So, you can do you can make a quad in this format you can make a quad like this or you can make a quad like this ok, but they cannot be like say 2 ones here.

So, as I was showing previously. So, that type of grouping is not allowed. So, you have to group where only variables are changing; so in a row and column fashion. So, if you can, if you cover one you can cover one full row or a full column or multiple rows and multiple columns, but not part of if row and part of another part of one row and part of another row, or part of one column and part of another column to make say quads and all. So that is allowed because that does not lead to minimization.

(Refer Slide Time: 05:38)



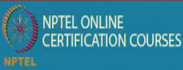

## K-Map

### Reduction Rule

**To reduce the Boolean expression, first we have to mark pairs, quads and octets.**

- Pair** – remove one variable
- Quad** – remove two variables
- Octet** – remove three variables

**Imp** – To get the optimum reduction, priority is given to octet first, then quad and then pair.

So, it is important that to get the optimum reduction priority is given to the octet first then quad and then pair. So, if you follow this rules so, it can be proved that you will definitely get a minimum some of product form, ok. Of course, there can be many groupings possible many alternative groupings possible, but all of them leading to some minimal form. So, you can be you can be assured that you will get a minimum size representation of the function.

(Refer Slide Time: 06:11)





## Karnaugh Maps - Four Variable K-Map

### Octet Reduction

AB \ CD	C'D'[00]	C'D'[01]	CD[11]	CD'[10]
A'B'[00]	1	1	1	1
A'B'[01]	1	1	1	1
AB[11]	1	1	1	1
AB'[10]	1	1	1	1

$f=1$   
 $f=1$

A'B'[00]	1	1		
A'B'[01]	1	1		
AB[11]	1	1		
AB'[10]	1	1		

So, this is the octet reduction so, this is one possibility this is another possibility, ok. So, we can make groups like this or groups like this. So, of course, what will happen if all this all the values are one like, if I have the Boolean function  $f$  equal to true. So,  $f$  equal to 1. That means, so, in that case all these are 1's. So, you can make this whole thing is a as a group. And now you see that all the variables are changing their polarities. So, naturally the minimized form is  $f$  equal to 1, because now we no variable remains which does not change it is polarity. So, that is there, but that is obviously, not a not a case to be minimized we normally do not come across such functions.

(Refer Slide Time: 07:07)

**Karnaugh Maps - Four Variable K-Map**

**Octet Reduction**

AB \ CD	C'D'[00]	C'D[01]	CD[11]	CD'[10]
A'B'[00]	1	1	1	1
A'B'[01]				
AB[11]				
AB'[10]	1	1	1	1

A'B'[00]	1			1
A'B'[01]	1			1
AB[11]	1			1
AB'[10]	1			1

Now, for another possibility of octet reduction as I was telling that the maps are considered to be folded maps. So, you can take this row adjacent to this row, because between these 2 rows only one bit is changing. So, only the a value A variable is changing it is polarity from A bar to A.

Similarly, between the columns also so, this first column and last column; so they are close to each other they are adjacent to each other so, you can fold your table, that way also. So, you can create this type of octet. So, what will be the minimized form? For this octet so, this is here you see that between these 2. So, A is changing it is polarity from A bar to A so, B bar is not changing it is polarity. But C and D they are changing it is polarity. So, this whole expression for this one is B bar.

Similarly, here when we are taking these 2; so you see that this A is changing it is polarity from A, A to one to 0 B is also changing it is polarity, between C and D C is changing it is polarity because for this for this column. So, this is C bar and for this column this is C, so, what is not changing it is polarity is the D bar. So, D remains complemented, here and D remains complemented there also.

So, this one so, this one the term that will get is D bar so, this is a we can do the minimization. So, we should try to maximize the number of octets and when we fail so, no more octet can be form. So, we try to make a quads, and A quads are not also not possible, then we will try to go to pairs that should be the rule that we should follow so, this is the quad.

(Refer Slide Time: 08:54)

**Karnaugh Maps - Four Variable K-Map**

**Quad Reduction**

AB \ CD	C'D'[00]	C'D'[01]	CD[11]	CD'[10]
A'B'[00]	1	1		1
A'B'[01]	1	1		1
AB[11]				
AB'[10]				

→ A'CD'

A'B'[00]	1			
A'B'[01]	1			
AB[11]	1			
AB'[10]	1			

So, this is one possibility so, this is another possibility. So, quad reduction and of course, we can also do this type of grouping like in the previous example you see that these one. So, if you take it in an isolated fashion then you so, not get. So, you will get a pair out of that. So, pair when we do so, you are you are actually taking more number of variables. Like this pair if you represent then this is basically A bar is not changing it is polarity. So, A bar C D bar so, that way only a only B is changing it is polarity. So, you can do it like this.

But the point is that you need not be doing it like that. So, you can you can make quads like this also. So, this one and this one because these 2 rows these 2 columns are adjacent

to each other. So, you can take them together, and you can do it in this way. So, you can make a quad out of that.

So, we will take one example.

(Refer Slide Time: 10:03)

**Karnaugh Maps - Four Variable K-Map**

**Quad Reduction**

AB \ CD	C'D'[00]	C'D[01]	CD[11]	CD'[10]
A'B'[00]	1			1
A'B[01]	1			1
AB[11]				
AB'[10]				

$A'D'$

A'B'[00]	1			1
A'B[01]				
AB[11]				
AB'[10]	1			1

$B'D'$

So, here you see that we have got this one this is A bar. So, this is A C bar so, this is a CD bar and C bar D bar. So, these between these 2 so, you see that your A bar is a is not changing it is polarity, B is changing it is polarity, and between these 2 C, C is changing it is polarity, but D is not changing it is polarity. So, you get so, if you take this a red one red quad that we are talking about. So, this quad boils down to A bar A bar and then D bar; so A bar D bar.

So, you can also make a quad like this. So, you take so, this one this one this one and this one. So, these 4 because these 2 rows are adjacent these 2 rows are adjacent to each other so, I can fold the table that way. Similarly these 2 columns are adjacent to each other so, I can also fold the table this way. So, ultimately after these 2 times folding this 4 ones come close to each other.

So, they are within they are adjacent rows and columns. So, I can do the; I can say that the minimization will be done and this is going to be. So, between these a is changing, it is polarity B bar is not changing it is polarity. So, this is B bar and between these 2 your C bar C is changing it is polarity, but D is not changing it is polarity so, it is B bar D bar.

So, if you group like this if you grouping like this fashion 4 ones like this. So, you can get a quad represented by B bar D bar.

(Refer Slide Time: 11:48)

### Redundant Term

$x \backslash yz$	0	1
00		
01	1	1
11	1	1
10		1


$xy + x'z + yz$  (consensus term)

$xy + x'z + yz = xy + x'z$


$x \backslash yz$	0	1
00		
01	1	
11	1	1
10		1

Term  $yz$  is redundant


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



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES



So, this is possible. So, sometimes we introduce some redundant some terms become redundant like see here what is happening is that you see that ok. So, this is a 3 variable Karnaugh map so, where we have got ones like this at this position so, we have got ones. So, this is this one this one and this one.

Now, you may group it somebody while doing since I cannot from quads. So, what I do I form pairs so, I form a pair like this I form a pair like this, somebody also makes a pair like this. So, and in another case somebody says that, I have to cover the ones. So, I have cover the ones like this so, I do not take this pair ok. So, what is the difference between these 2 cases?

Now, if you do it like this, if you take this third one also this third one also. So, between so, the first term that you get is this one this pair. So, it gives me the term  $xy$ , then this pair gives me the term  $x \bar{z}$ . So, this one this vertical one; so this gives me  $x \bar{z}$  and this one gives me  $yz$ . So, by taking that I can get then the expression like this.

Now, this is not the minimum form, because between this you can you can show that this is equivalent to  $xy$  plus  $x \bar{z}$ . So, how to do? How to do this? So, this  $yz$ ; so we can expand it as  $x$  plus  $x \bar{z}$ , and then do multiplication and then just redo the grouping



ultimately it turns out that these 2 are same. Whereas, the person who has not taken this consensus term  $yz$  will make grouping like this. So, which one is correct? So, you see that after when we do this so, we are not we are not getting the minimum form. So, we are getting some extra terms. So, the naturally we should not do this type of grouping.

So, here what is happening is that this ones that that are that are being covered by this particular group are already covered by other groups ok. So, they are already covered by other pairs so, this is a redundant term. So, at if one of them was not covered then of course, it is fine. So, this is. So, I can.

(Refer Slide Time: 14:39)

**More than one solution**

$F = \sum m(0,1,2,5,6,7)$

	$a$	0	1
$bc$	00	1	
01	1	1	
11			1
10	1	1	

$F = a'b' + bc' + ac$

	$a$	0	1
$bc$	00	1	
01	1	1	
11			1
10	1	1	

$F = a'c' + b'c + ab$

So, that way I can say that, if one of them was not covered. So, if I remove this so, if one of them gets uncovered then of course, you have to cover, but in this particular case this ones covered by the term  $yz$  are already covered by the other 2 terms. So, that brings redundancy. So, that should be so, you to get the minimum forms so, that should be avoided.

So, this is so, I can have more than one solution also like as I have said that the Karnaugh map solution is not unique. So, like here you see that for the same function consisting of the mean term 01 2 5 6 7 3 variable Karnaugh map. So, this is the 1 placement of ones that I have.

So, now, I can do grouping in different format like one person does the; so, no quad formation is possible so, only pairs are possible. So, while pairing so, I will do it like this. So, this is made one pair this is made one pair this is made another pair. So, that way I get an expression like this so,  $A\bar{B} + BC + AC$ .

Somebody does a grouping in this format; so, instead of covering like this covers say these, these 2 ones into one group into one pair, these 2 ones into one pair and these 2 ones in another pair. So, that way if we will do the covering, then you will get a different type of cover. So,  $A\bar{C} + B\bar{C} + AB$ .

Now, which one is better; so as for as the implementation is concern so, both of them are equally costly. So, here also the number of literals that we have is one 2 3 4 5 6, here also number of literals that we have got is one 2 3 4 5 6 numbers of terms in both the expressions are also same. So, this is there in 3 terms here and here also there are 3 terms; so these 2 expressions there, perfectly similar ok, and if you take it to their equivalent also because they are if you take into the canonical some of product form. So, they will boil down to all these ones that we have in the Karnaugh map that is 0 1 2 5 6 7 so, they are definitely same.

So, that is why we say that the Karnaugh map it gives us the minimal form. So, there can be multiple such minimum results, ok. So, it does there is no discrepancy between them. So, all of them are correct so, you can you can produce any of these grouping and say that the result is correct, ok. So, we will get the correct result?

(Refer Slide Time: 16:40)

AB \ CD	00	01	11	10
00	0	0	1	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$F = (A + B')(A + C)(A + D)$

Now, how do you do, for product of some expression. So, in case of product of some what we do we make groups of 0's ok; so here we in case of to get some of product representation. So, we were grouping the 1's in case of product of some will be grouping the 0's. Like here after putting the Karnaugh map, the ones and 0's on to the table. So, we are grouping this 4 0's into one quad this 4 0's into one quad, and this 4 0 that is this 2, and these 2 into one quad. You see if you look into it carefully you see that say this quad, and this quad, they are covering this 4 0's twice. But in a both the cases so, there is something new that is happening like if you if you remove this quad. So, this 0 will become uncovered. Similarly if you remove this quad, this 0 will become uncovered. So, at least 1 0 is there which is which will be uncovered, if we remove the quad. So, it does not.

So, it ensures that we are not introducing redundancy in the process like the example that I so, to some time back where the consensus term. So, that 1's even if we remove the grouping of that consensus term. So, ones will remain covered by some other pairs, but here if we remove any of this quads. So, they will be the at least 1 1 of 1 0 will become uncovered. So, we have got this type of grouping, and then while writing the expression. So, I we know that between these 2 in case of a say this particular quad, the symbol C is not changing, it is polarity the variable C is not changing it is polarity and variable a is not changing it is polarity.

So, for pos I will be writing in the complemented form so, 0's will be written as true and ones will be written as false as complemented one. So, this one gives me a plus C, so, this quad is a plus C. Similarly this quad is a plus B bar so, this is a plus B bar and this is a plus D ok. So, this way we can write down the consensus we can write down the product of some expression from the Karnaugh map. So, that is a good technique by which to get the product of some expression for Boolean functions.

(Refer Slide Time: 19:10)

### Karnaugh maps: Don't cares

- In some cases, outputs are undefined
- We "don't care" if the logic produces a 0 or a 1
- This knowledge can be used to simplify functions.

		A				
		00	01	11	10	
C	D	00	0	0	X	0
	01	01	1	1	X	1
	11	11	1	1	0	0
	10	10	0	X	0	0
		B				

- Treat X's like either 1's or 0's  
- Very useful  
- OK to leave some X's uncovered

$$\begin{array}{c|c} \text{Sw}_1 & \text{Sw}_2 \\ \hline \text{L}_1 & \text{L}_2 \end{array}$$

$$\begin{array}{c|c} \text{Sw}_1 & \text{Sw}_2 \\ \hline \text{L}_1 & \text{L}_2 \end{array}$$

Sometimes we have got a Boolean function that has some of the terms as do not care. So, like so, it may be the case that outputs are undefined. So, we do not care whether the logic value produced is 0 or 1. So, it may so happen that we may be interested in a say. So, we may say it like this that if say switch n one is on the switch one is on, then light 1 one is on, if switch 2 is on. Then light 1 2 is on,. So, suppose we tell only these 2 terms. So, if you are drawing a truth table then it is switch 1 switch 2, light 1 light 2. From this expression it says that if switch one is on whatever be the condition of switch 2 light one should be on. So, I am not bothered about the condition of switch 2. Similarly, if switch 2 is on whatever be the condition of switch one the light 2 should be on.

So, that way I say that so, this is the do not care condition. So, you do not care the condition of the switch 2 for controlling light one, similarly we do not consider the status of switch one for getting the status of light 2. So, this way we can have some of the terms as do not care. So, outputs are undefined. So, in those cases; so in some cases; so we can

take it as any value. So, we can take so this value so whenever drawing the truth table. So, for 1 0 so, we can say this is light 2 value is not known. So, light 2 may be on light 2 may be off. So, like that so, that way. So this, do not care so, this is actually for those combinations. So, they I can treated treat them as 0 or 1 arbitrarily. So, this is the function where we have said this AB and CD so, this AB. So, this AB is 0 0 01 1 1 0, CD is 0 0 01 1 1 1 0, and the function value. So, this so, if the AB is 0 0 and CD is 01, then it is one AB 01 CD 01 is also one. But if AB equal to 1 1 and CD equal to 0 0, then we are not bothered about what is the value of the function. So, that is represented as z in the truth table, ok.

Similarly, for AB 1 1 CD 01 we are not bothered about the output of the function. So, this is also x while considering while doing the minimization. So, we can treat this xs either as ones or as 0's; like, say if this x so, between these 2 xs. So, if I take these x as 0 and this this x has 1, then I can I not be bothered about realizing this 0. So, I can just I can make a quad out of that, but if I take this one as 0 I cannot make a quad. So, in that case I have to if I take this as 0, then I have to do a pairing like this or say this 2. So, I can do a I have to so, this is 0. So, I have to do a pairing in this fashion.

But if this is one, if I take this x as one then I can do make a quad formation like this. Similarly, for this x so, if I take a 0 I do not need to do anything for this row, but if I take this as one then of course, this has to be covered in some sense I have to introduce some term them. So, that way this xs to the facility of this to facilitate this minimization process, this xs can be taken as 1's or 0 it depending upon the situation.

(Refer Slide Time: 23:29)

### Karnaugh maps: Don't cares

$f(A,B,C,D) = \sum m(1,3,5,7,9) + d(6,12,13)$   
 • without don't cares  
 -  $f =$

$A'D + C'D$

A	B	C	D	f
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	X
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	X
1	1	0	1	X
1	1	1	0	0
1	1	1	1	0

So, you will take some examples and see, say this function ABCD so, these are the mean terms 1 3 5 7 9, and this, this 6 12 and 13 so, they are do not cares. So, on the output column we have got this 6 12 and 13. So, this values has do not cares, and then we can say that the corresponding truth table. So, it will be looking something like this.

Now, while making the groups what we do so, this x is treated as 1. So, I make a quad out of that similarly this 4 are ones. So, this quad will remain, but I can take this x as one and make a quad out of that. Similarly, this x is take it as 0 so, that I do not have to do a covering. So, I can get the minimization form like this quad gives me like  $A' D + A' D$  the first one and this is the blue square. So, this is the blue quad it gives me  $C' D$ . So, this way I can realize the functions by fruitfully exploiting the do not cares to be either equal to 0 or 1.

(Refer Slide Time: 24:37)

## Don't Care Conditions

- In some situations, we don't care about the value of a function for certain combinations of the variables.
  - these combinations may be impossible in certain contexts
  - or the value of the function may not matter in when the combinations occur

hexadecimal value  $\Rightarrow 0 \rightarrow F < 10$

10-12  
↓

0000

L  
0

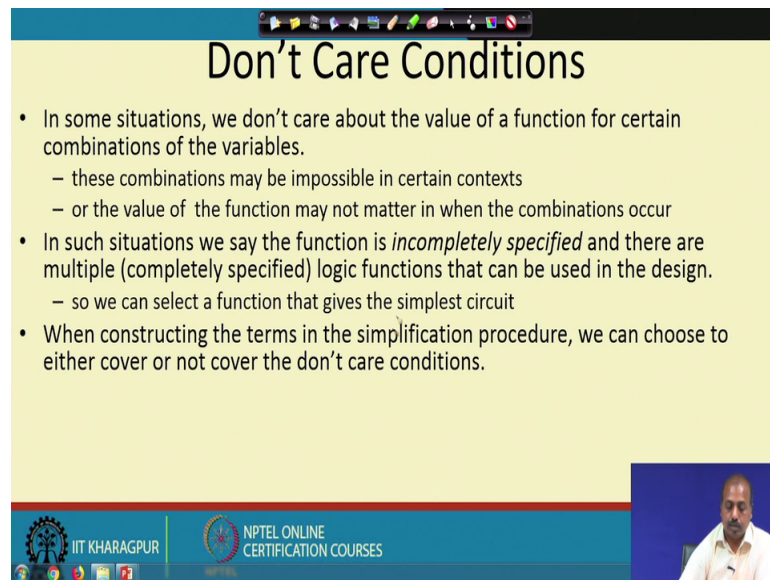
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, do not care condition in some situation we do not care about the value of a function at certain combinations of the variables, this combinations may be impossible in certain contexts, or the value of the function may not matter in when we combinations occur when the value the value comes is does not matter. For example, if we are say, if we are say doing a say conversion, from say one code to another code, ok. So, say I have I am doing a conversion like if I have if I. So, this hexadecimal number system so, hexadecimal value; so this hexadecimal value can go from 0 to f.

So, to represent this 0 to f, I need 4 bit value. So, this input is 4 bit. So, it is so, D one D 2 D 3 and D 4. And I say that whenever the value is less than 10 whenever the value is less than 10 one light will glow. And whenever the value is more than 10 so, whenever the value is more than 10, then the light will not glow light will be off. So, for the combinational like 0 0 0 0 to 10 so that is. So, 1 0 1 0, up to this combination the light value is equal to 1. So, all these are equal to 1, then for from 1 0 1 1. So, the light value is 0 to 1 1 1 1 the light value will be 0. So, it is less than 10.

So, may this so, this in this case it is completely specified, but suppose it says that for light value less than 10 the light. So, if the light value is less than 10 then the light should be on if the input value is less than 10 the light value should be on. And also if the input value is input value is a between 10 to 12 between 10 to 12; then also then the light value should be off.

(Refer Slide Time: 27:00)



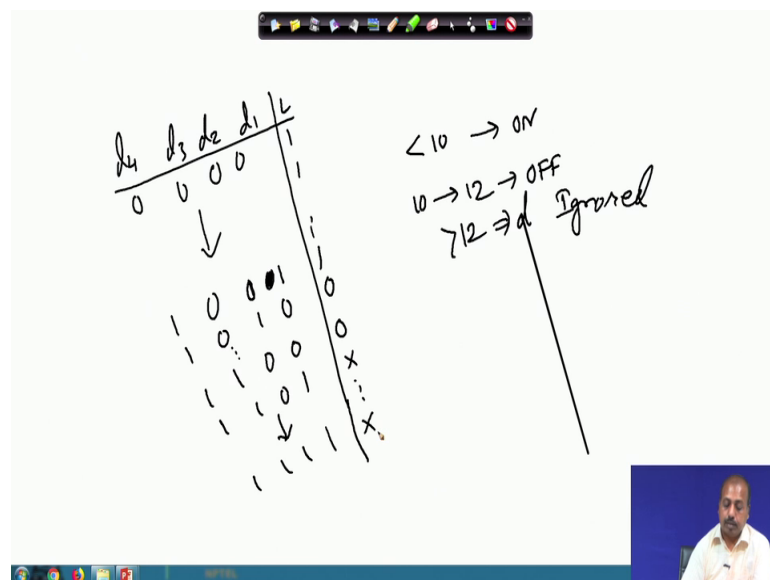
### Don't Care Conditions

- In some situations, we don't care about the value of a function for certain combinations of the variables.
  - these combinations may be impossible in certain contexts
  - or the value of the function may not matter in when the combinations occur
- In such situations we say the function is *incompletely specified* and there are multiple (completely specified) logic functions that can be used in the design.
  - so we can select a function that gives the simplest circuit
- When constructing the terms in the simplification procedure, we can choose to either cover or not cover the don't care conditions.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, between 10 to 12 the value should be off and so, if the light value is less than 10, then it is it must be on.

(Refer Slide Time: 27:07)



$d_4$	$d_3$	$d_2$	$d_1$	$L$
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1
X	X	X	X	X

$< 10 \rightarrow ON$   
 $10 \rightarrow 12 \rightarrow OFF$   
 $> 12 \rightarrow Ignored$

And if light value is between 10 to 12, then the light value then the light should be off if the decimal value is between 10 to 12, the light should be off. However, for the value greater than 12, it does not matter. So, it is it can be ignored it does not matter then the truth table that we were drawing previously.



So, it is  $d_1 d_2 d_3 d_4$ , and this is the light. So, for  $0000$  to  $0010$  so, up to this much the light value is one then for  $10$  to  $12$  for less than  $10$ . So, it is not so, it is not this is  $0$  up to  $9$  it should be one, from  $1010$  the light value should be off to  $12$ . So, that is  $1100$ , from  $1010$  to  $1100$  the light value should be off, but from  $1101$  to  $1111$  so, in this region the light value is undefined so, they are all do not cares.

So, this way we can introduce my problem statement be such that there are do not cares, and if do not cares are there. So, we can exploit it in the minimization process.