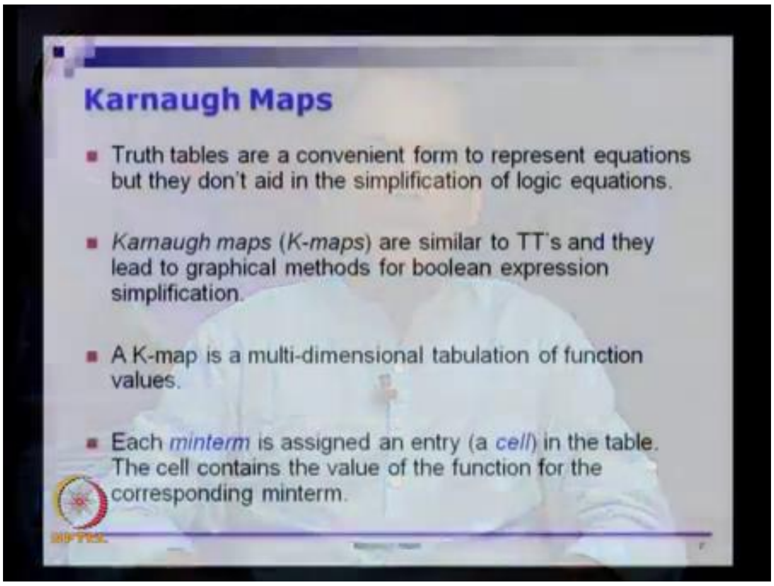
Digital Circuits and Systems

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 - 08
Karnaugh Maps

Welcome to module 8, this module we look at the basics of Karnaugh Maps.

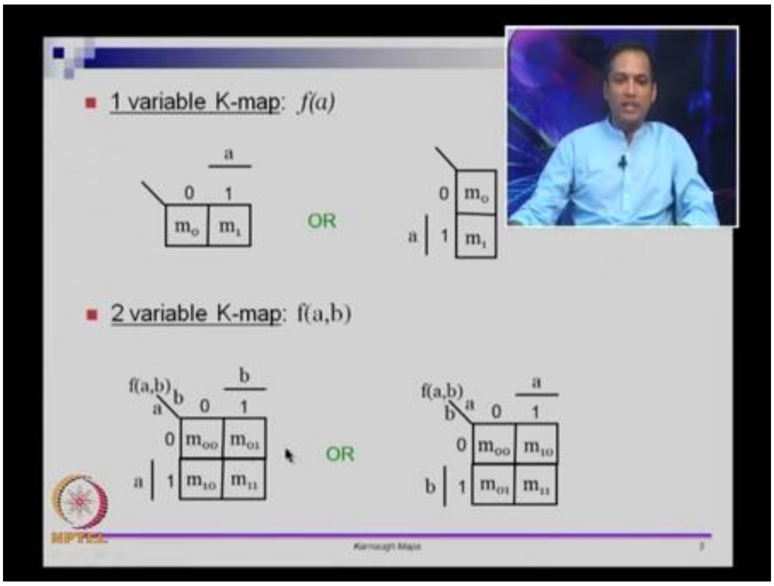
(Refer Slide Time: 00:25)



Karnaugh maps essentially truth tables, but it is slightly more convenient way of writing the truth tables. So, they are similar to truth tables, but they lead to what are called graphical methods for Boolean expressions simplification. So far we have use Boolean algebraic expressions directly, but Karnaugh map is a very interesting way of graphically or visually combining various terms together and it will give you something which is with which you can minimize the Boolean expression.

So, the way Karnaugh maps works is for each min term, we are going to assign an entry or a cell in the table. So, we are going to build something called a Karnaugh maps table, and for each min term we will assign one location in the table, and so if you have 3 inputs, then you know that the truth table has 8 different combinations. So, the Karnaugh maps for it will also have 8 different cells, and the cells contain the information of the function for the corresponding min term. So, if a corresponding min term turns on and then the function will turn on. So, we will see how to capture this information in the Karnaugh map.

(Refer Slide Time: 01:27)

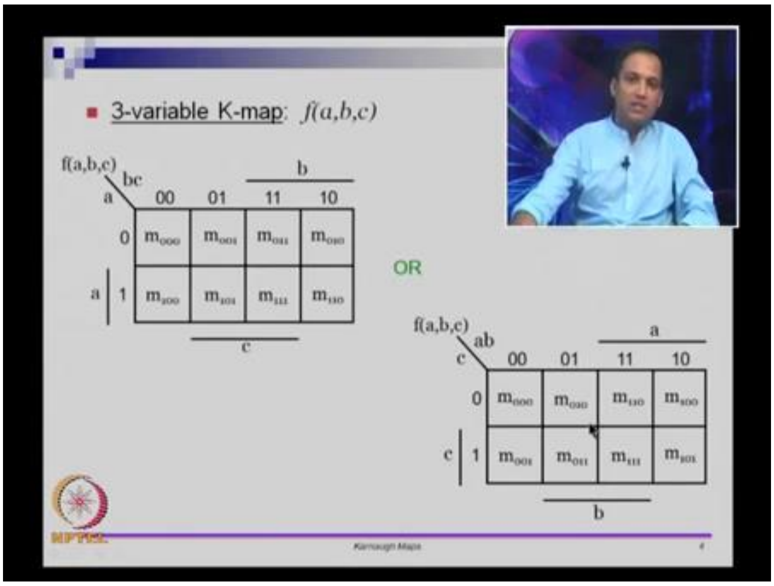


So, Karnaugh map for one variable essentially looks like this, so if I have only 1 input a, then a can take two combinations, either a can be 0 or a can be 1, I can write it in the row form, so a can be 0. So, corresponding to this you will put the entry m naught and corresponding to when a equal to 1, you will put the entry m 1. So, you can either do this or you can write the Karnaugh map like this, so m naught corresponds to row 0 and m 1 corresponds to row 1 and this part captures whether a is 1 or 0.

So, when a is 1 you go and look at what is the value for m 1, similarly in this 1 when a is 1 the value of m 1 tells you what the value of the function is. If you want 2-variable k map, we have f of a comma b, so in this case a, the possible values of a are listed in the rows and possible values of b are listed in the columns. So, in this case we go in the order 0 1, similarly 0 1 here. So, 0 0 corresponds to min term 0, 0 1 corresponds to min term 1, 1 and 0 corresponds to min term 2 and 1 and 1 corresponds to min term 3.

If your function is a b, you can also use the combinations of a in the columns and combination of b in the rows. So, in this case a is listed along the columns, so as you move from one column to the other a takes different values. So, in this column a is supposed to be 0, in this column a is supposed to be 1, in this row b is supposed to be 0 and this row b is supposed to be 1. So, if you notice the min terms here verses the min terms here they are only shuffled slightly. So, m 0 0 and m 1 1 are in the same place, 0 1 and 1 0 have move to a different location, because a and b are swapped in the directions.

(Refer Slide Time: 03:27)



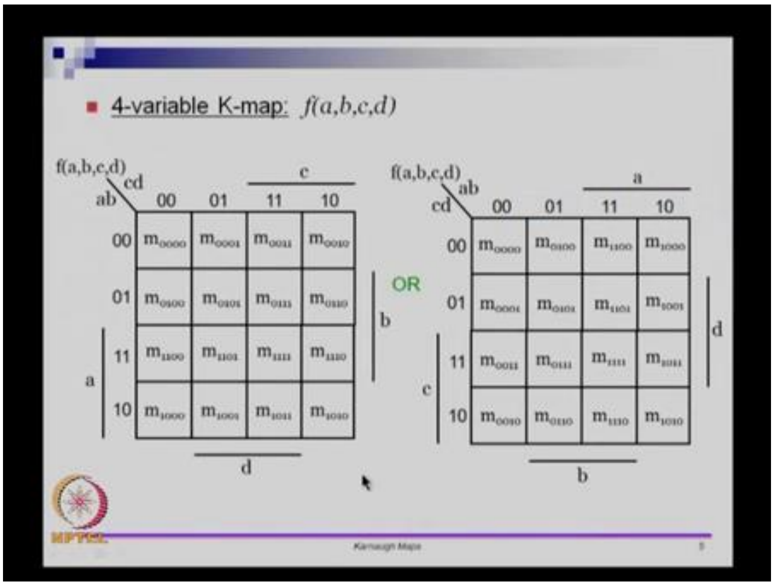
And a 3-variable k map let us say, it is a function of a, b, c, so we have taken one of these variables namely a and we list all the possible values that a can take, namely 0 and 1 and for each row you list all the possible values that b and c can take. So, b c can take 0 0 0 1, 1 0 or 1 1, but as I said earlier we will in the previous module I mention this, we will instead of having 0 0 0 1, 1 0 and 1 1 we will keep it as 0 0 0 1, 1 1 and 1 0 we will see why this is so in a little while and the corresponding min terms are here.

If you notice this row corresponds to a equals 1, so these 4 min terms m 1 0 0, m 1 0 1, m 1 1 1 and m 1 1 0, you can see that you are looking at a in the uncomplimented version in each of these 4 terms and b and c are coming in all possible combinations. So, this row corresponds to a equals 1 and this row corresponds to a equals 0. If you go and look at these 4 entries here, so you go and read it m 0 1 1, 0 1 0, 1 1 1 and 1 1 0 if you notice the second bit is turned on, which means for b equals 1 these are the 4 min terms that capture the fact that b is 1 or it is going to be in the uncomplimented form.

And similarly, if you take these 4 this captures information that c should be 1, c is in the uncomplimented form, you can see that 0 0 1, 1 0 1, 0 1 1 and 1 1 1 has the last bit c which is 1. So, this is a 3-variable k map you can... So, here I have chosen a to b along the rows here and b and c along the columns, you can also do a slight different rearrangement, in this case c is picked to be along the rows and combination of a, b are along the columns. So, correspondingly the min term will also get shuffled.

So, 0 0 with 0 gets 0 0 0, 0 1 with 0 gets 0 1 0 and so on, so you should not read from rows going into columns, you should read from columns going into rows, because we have fix the order to be a comma b comma c. So, we should read the bits for a first, then b then for c. So, this cell is corresponding to a equals 1, b equals and c equals 0 or m 6.

(Refer Slide Time: 06:00)

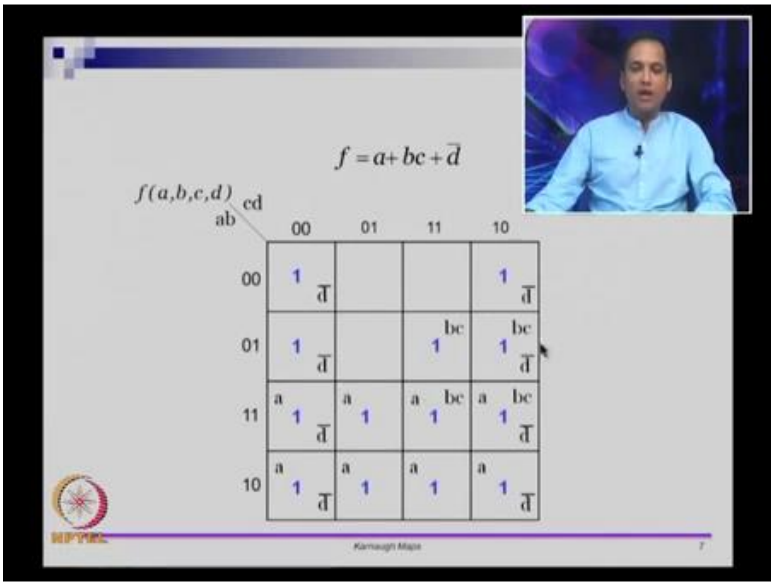


And 4-variable k map is essentially list all the combinations of 2 inputs along the rows and another 2 inputs along the columns and the only difference is you see 0 0 0 1, 1 1 and 1 0 as we did in the previous example. So, in the 3 input case you had 0 0, 0 1, 1 1 and 1 0 we list in the same order here 0 0, 0 1, 1 1 and 1 0 the same thing for c and d which are the 2 other inputs 0 0, 0 1, 1 1 and 1 0.

So, in this table the last two rows correspond to the fact that a should be 1 or it corresponds to a being uncomplimented. The middle 2 rows correspond to the fact that b is uncomplimented, the last 2 columns corresponds to the fact that c should be uncomplimented and these 2 columns for d to be uncomplimented. So, this picture here is essentially just picking c, d for the rows and a, b for the columns. So, you can see that if I pick this cell, the function says it is in order a, b, c, d.

So, to say what min term should go here I should look at a, b here which is 0 1 and c d here is 0 1. So, this is m 0 1, 0 1 if I look at this cell here, this corresponds to a equals 1 1 and a b equals 1 1 and c d is 0 1. So, this is m 1 1 0 1, so that is what we have here, so these 2 tables are actually equivalent only that the order in which a, b, c and d are pick or different here.

(Refer Slide Time: 07:35)



Let us see why the whole thing is useful, so all this thing that I showed so far does not really help you in doing anything better. So, it is only just the table rewritten in a different form; however, let us see how we can actually use k maps to simplify things. So, let us start with this one f is a plus b c plus d bar, let us see how this things can be represented in a table.

So, remember this function is having 4 variables namely a, b, c and d and if I do not mention explicitly we will keep the order always in increasing alphabetical order a, b, c, d, x, y, z and so on. In this case, we will assume that the function is written as a, b, c, d I am going to take a b along the rows and c d along the columns. So, you have a b in the 4 possible values, c d in the 4 possible values, with this little twist. We go 0 0 0 1, 1 1 and 1 0 similarly 0 0 0 1, 1 1 and 1 0 here.

Now, let us see these three terms, so like we did with the truth table in previous module, I am going to go and fill up this K map table with 1's and 0's. So, the first terms is a, so if a is 1 no matter what b, c and d are f will become 1 and to capture that so this is the place where a is 1, you can see that a b you put 1 1 here and 1 0 here. So, this corresponds to a equals 1, because that is a row where a is 1 and so are all these entries.

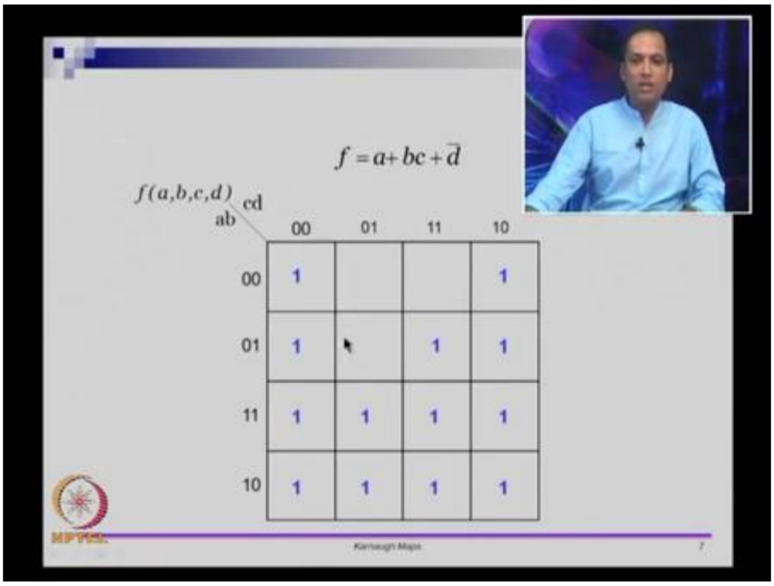
If I look at all the entries here, they all corresponds to a being 1, so b is of course, taking 1 or 0 we do not care about that. So, if a is 1 all these terms correspond to the min term where a is 1, so all these are the min terms where a is 1. Finally, let us look at b c, let us find out all the min terms which correspond to b equal to 1 and c equal to 1. So, if you go and look at this column here, we look at this column here, so b is 1 and c is 1, so this is a min term where which corresponds to b equal to 1 and c equal to 1.

But, that is not the only place all these 4 cells correspond to b equal to 1 and c equal to 1. So, you go and look at these, this is the row in which, so these are the rows in which b is 1, these are the columns in which c is 1, so the intersection of that is these 4 cells. So, these 4 cells correspond to the fact that b and c are both 1, finally let us look at d bar. So, if d is 0, f should be 1, so wherever d is 0 we are going to mark, so we need a 1 there, so let us see...

So, if I look at this column here, we know that the c d or c corresponds to 0 and d corresponds to 0. So, this must have a d bar and so will this, this and this entry, because all of that corresponds to d b 0. Similarly, if I go and look at the last column this also corresponds to d equal to 0. So, this corresponds to c equal to 1 and d equal to 0, you do not care about the value of c when we look at this term here. So, all those entries also become, they are all corresponding to the fact that d should be 0.

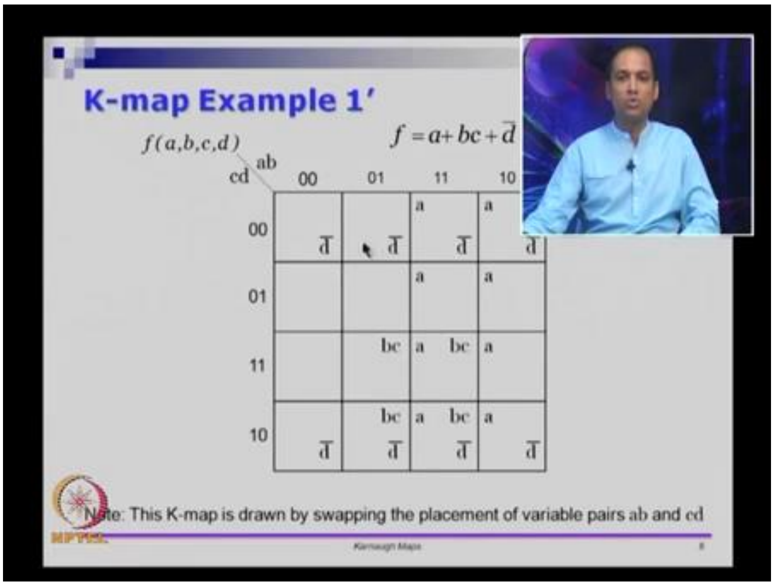
Now, wherever you see something like this, so this is true when a is true, this is true when either a is true or b c is true, this is true only when b c is true, this is true only when d is 0 and so on. So, wherever you have some entry, you make all of them 1's, because that is what the truth table tells us, you can see that. So, for example in this cell a is 0, so this is 0, b and c are both 0. So, this term is also 0; however, if d is 0 this cell should get a 1. So, this is having only the information that d bar should be 1 for the set to become 1 and in this specific case d bar is actually 1.

(Refer Slide Time: 11:46)



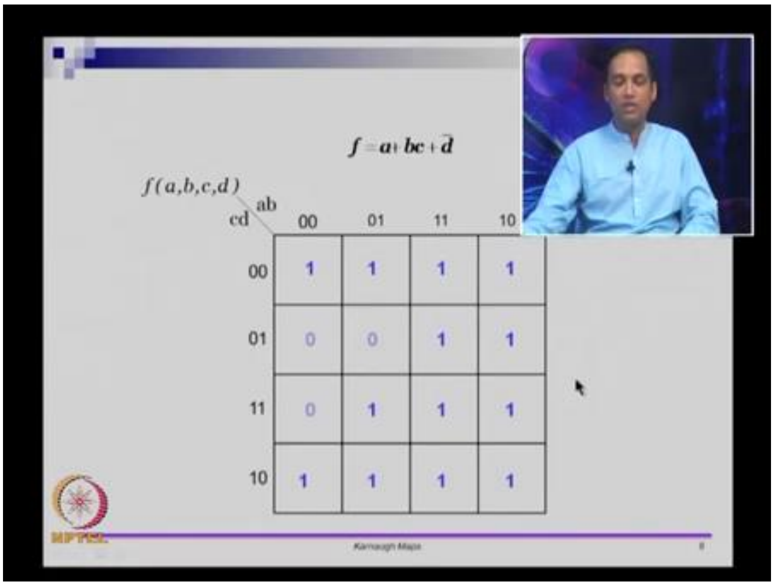
So, wherever we have some entry we make all of them 1's, we do not need the sum terms any, we do not need the product terms any more we can drop it. All the terms that are not there, they become 0's, so these 3 entries are 0's. So, I if go and write the truth table for this function, it will have 2 power 4 or 16 rows, among the 16 rows, 13 rows will have a 1 and the rows corresponding to these 3 min terms will have a 0, you try it on. So, I put it in the K maps table, you go and put it in the regular truth table, where you have 16 different rows with 4 combinations and you will notice that min term 1, min term 3 and min term 5 should, will be 0's if we expand this function, so you go and try that out yourself.

(Refer Slide Time: 12:32)



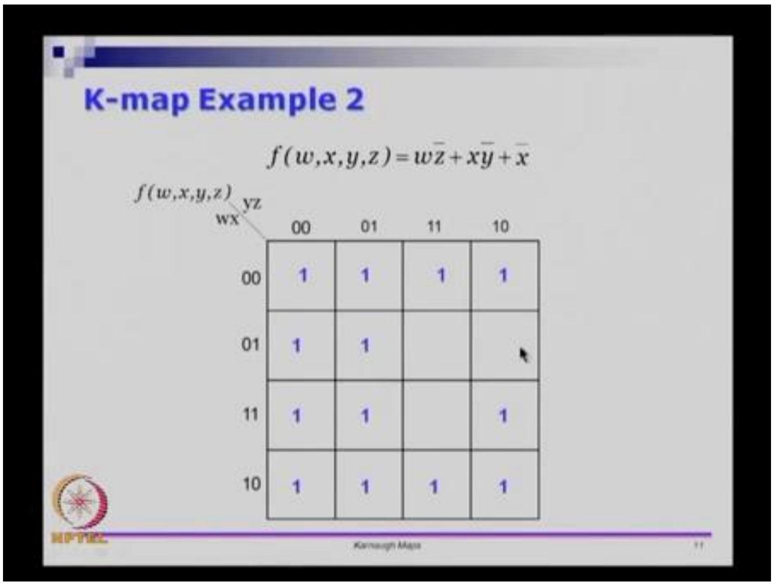
So, let us see how K map is drawn, it is start with in this example a b and c d are switched around, the same function that a b and c d are switched around. So, first a is true for these 2 columns, then b c is true for these 4 cells and d bar is true for these 2 rows and we can do the same thing that we did before, wherever you have an entry you put a 1 and wherever you have no entries you put a 0, this is also a valid K map.

(Refer Slide Time: 13:00)



So, this is an essentially different way of folding the same truth table. So, so far what we have is a truth table and K maps are not different things, they capture the same functionality except that the truth table will have a single vector for the column. So, you will have 1 output vector which is the last column. However, in a K map the 1's and 0's are actually embedded inside a either a square or rectangle or something like this. So, in this case we have to pick the ordering of the inputs and based on that we put the min terms corresponding.

(Refer Slide Time: 13:36)

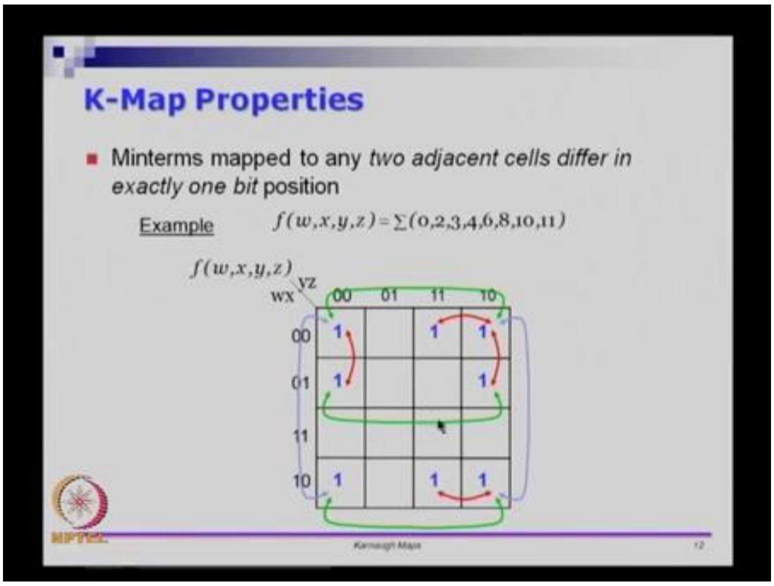


Let us look another example here w z bar plus x y bar plus x bar, let us say that is the thing that we want to capture. So, now let us go and see where w z bar, what are the cells that corresponds to w z bar. We want w to be on and z to be off, which means we have to look at the rows where w is on. So, which are the rows where w is on? These are the 2 rows where w is on, but we also want z to be 0 or z to be off. So, these are the 2 columns where z is off.

So, the w z bar will correspond to the intersection of the condition where w is on which are these 2 rows and z is off which are this column and the last column. So, those are the 4 terms that correspond to w z bar. Now, let us look at the middle term x y bar, for x y bar we want x to be on and y bar to be on or x to be on and y to be off. Where is x on? X is on in these 2 rows in the middle.

So, and we take the intersection with wherever y is off that is this column and this column. So, intersection of these 2 rows and these 2 columns give me these 4 cells, these cells here that is my x y bar. Finally, x bar is these 2 rows. So, wherever you have any such term, you started with the blank 1, wherever you have such a term you put 1's and you can get rid of the actual terms that gave you that and wherever you do not have them you can put 0's. So, in the previous slide these blank cells must get a 0.

(Refer Slide Time: 15:28)



Now, let us look at some important properties of K maps, so one crucial property and the reason why we rearrange the whole thing is because the adjacent terms in the min term differ exactly by 1 bit position. So, if you go and look at any 2 cells which are adjacent to each other, so you go and look at this cell and this cell, this cell corresponds to w, x, y, z being 0 0 0 1 this cell corresponds to w, x, y, z being 0 1, 0 1.

So, if I take these to positions 0 0 0 1 and 0 1 0 1 it differs only in the value of x, w, y and z are all the same. If I take let us say these 2 cells, this corresponds the min term m 1 0 0 1 or m 9 and this corresponds to the term 1 0 1 1 or m 7. So, m 9 and m 11 if you go and think about it they differ in exactly one bit position, namely the third bit.

So, y is the only term that differs from the, this bit to this bit, in this place y is 0 and this column y is 1. Now, if I go and look at this table here this is w x, y z is given as a sigma of 0, 2, 3, 4 up to 11 let say how this can be filled up. So, this is a canonical sum of product, if you want to fill it up this is 0, 1, 2, 3. So, 0, 2, 3 and they are on, then the next row is for 4, 5, 6 and 7. So, among 4, 5, 6, 7 only 4 and 6 are on.

So, let us do this once more 4 we can see that 4 is here, so we put a 1, 5 is this term, but 5 is missing, 6 is the last column and then 7 you come back. So, you go from left to right then skip then come back. So, let us do it for this row, so this is for 1 0 0 0 that is 8, 9 skip come back. So, 8, 9, 10 and the 11 and this row would be 12, 13, 14 and 15 we can see that non of 12, 13, 14, 15 are there. So, all those cells are 0 and among 8, 9, 10, 11 we see 8, 10 and 11. So, 8, 10, 11 or 1, 9 is 0 and so on.

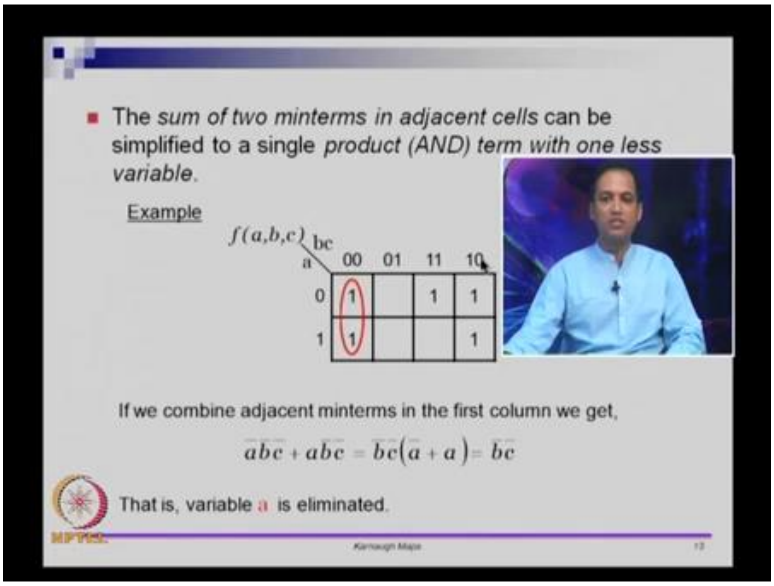
One thing you can notices is, even among the rows we have 0 0 0 1 1 1 and 1 0, the reason why we have this is because we want the same think that the adjacent cells should differ exact in 1 bit position. If I go and look at this cell and this cell, only w is changing x, y and z retain the same values. So, this whole think is interesting, because we can then do what is call combining. So, these two calls differ exactly one position, so what these two cells, these two cells and so on.

So, other interesting thing is this cell and this cell also differs only by one position, even though technically they are not adjacent to each other in the picture, what you can see is this term is corresponding to 0 0 0 0, this corresponds term to 0 0 1 0. So, the left most cell and the right most cell here are actually adjacent of each other, because the differ in only one bit position. So, one way to think about this is you take a paper like this.

If you put up entries here, so you know the adjacency right away, if you write it in the order 0 0 0 1 1 1 1 0 and so on, you know what adjacency is. What else is adjacent is if you are able to role the paper like this, so what it did I had a table like this, if I role a paper like this whatever cells are adjacent we call them adjacent also or I have a table like this and I role a paper like this 1 row folding into another whatever cells are adjacent they are adjacent in K map sense.

So, what we have is this row it is adjacent to this row and this column is adjacent to this column right and otherwise graphically whatever is adjacent is also adjacent. So, this is a interesting property of K map, this is possible only if you write it in this order 0 0 0 1, 1 1 and 1 0 not if you write it as 0 0 0 1, 1 0 1 1. So, these are showing other thinks which are adjacent of each other. So, the purple one and the green one are also show in adjacencies.

(Refer Slide Time: 20:11)

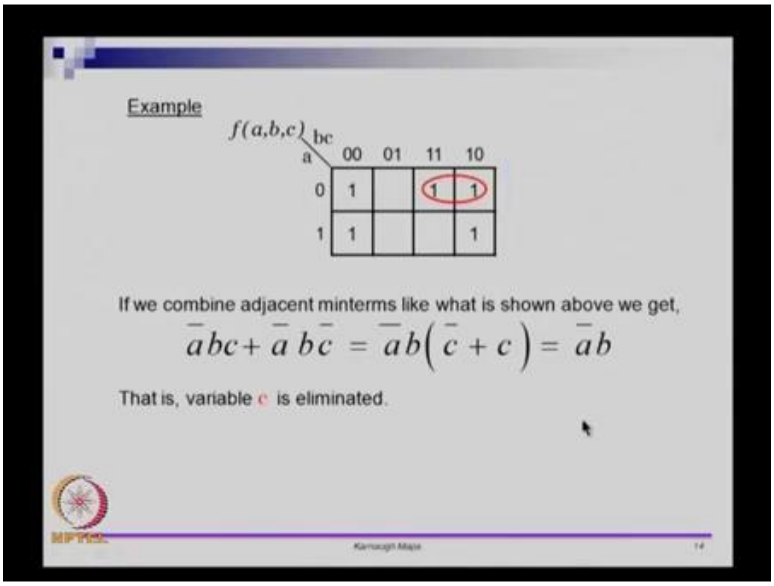


So, the reason why this is interesting is, if you take 2 min terms that are adjacent and if we combine them, the product terms simplifies by 1 less variables I will explain what; that means,. So, let us take adjacent terms now let us take this column, this take this column graphically we can see that these are adjacent each other, this is adjacent to this. Now, let us go and write the expression corresponding to this, what is the expression for this, this is a bar b bar c bar and the expression corresponding to this min term is a b bar c bar.

So, these are product terms the expression which capture these two 1s is a bar b bar c bar plus a b bar c bar, you can see that b bar c bar is common. So, it is a bar plus a which is same as b bar c bar, if I take two 1s that are adjacent to each other and if combine them it knocks out one of the variables. In this case, we know that b c is constant we are not changing the values of b c we have still in the same column, but we are going across multiple rows, whatever rows we are going across we can eliminate them.

So, if you do the same thing here for instants, so this term will corresponds to a bar b c bar, this one is corresponds to a b c bar, if combine them it is b c bar. So, if combine these two you can just look at for combine these two I do not have to look at a at all, what is the term corresponding to this one. So, it says b should be on and c should be off, so this last column corresponds to b c bar.

(Refer Slide Time: 21:52)



Let us do this one more time, let us look at this two 1s which are adjacent to each other, if I group them together, this term corresponds to a bar b c and this term corresponds to a bar b c bar. So, we have that as expression here, a bar b is common, so a bar b you can take it out and these to 1s corresponds to a bar b. So, from the picture if I want to inform that I can say that the oval is across these two 1s, in these two 1s as a if you notices where the columns are the column for c is changing, the column for b is remaining is same at 1, so I b is remaining at 1.

And this is in the row 0 we have a move to row 1, so row 0 which corresponds to a bar we have a bar, so this is corresponding to a bar b. So, the way I read this is I looked at which columns that is crossing and that variable will not appear in the product term, wherever you see the labels or constant. So, in this case the label for b is constant it is for 1 and 1, s b is constant across these 2 columns. So, I put b here and it is in this row where a is corresponding to 0, so I take the logical and of those two, so it is a bar and b.

So, you can as a exercise try and see, what is the expression corresponding to grouping this min term and this min term. This min term and this min term if you group it together, there adjacent because I talked about this, you can fold and the paper will touch each other, you can take that table and fold it and the 1 here will touch the 1 here. So, you go and find out what is the term corresponding to taking these two 1s and combining them.

And you will see that it is actually corresponding to the fact that a should be 1 and c should be 0. So, you will see that the term corresponding to this is a c bar, I do not need to do Boolean minimization like this, I can straight away tell you that the 1s that are here and here will combine and give you a c bar. And 1s that are here and here will combine and give you a bar c, it is because a bar and it is corresponds to c bar you can try and verify that yourself.

So, this brings me to the end of this module, so what we saw on this module was, we were able to take Boolean expressions and we are able to write it in terms of K maps, we will see how this k maps is graphically even though in some sense similar to the truth table being rearranged, it gives you something really powerful that the adjacent rows and adjacent columns if we combine them, they are going to knock out one variable and so on.

We will use this fact in the lot more detail in the next module, so as of now just try this little exercise and try figure out the terms by doing the Boolean expression yourself, and minimizing as well as just by looking at the labels of the rows and columns and guessing what the entry must be directly. So, try and do this yourself.

Thank you.