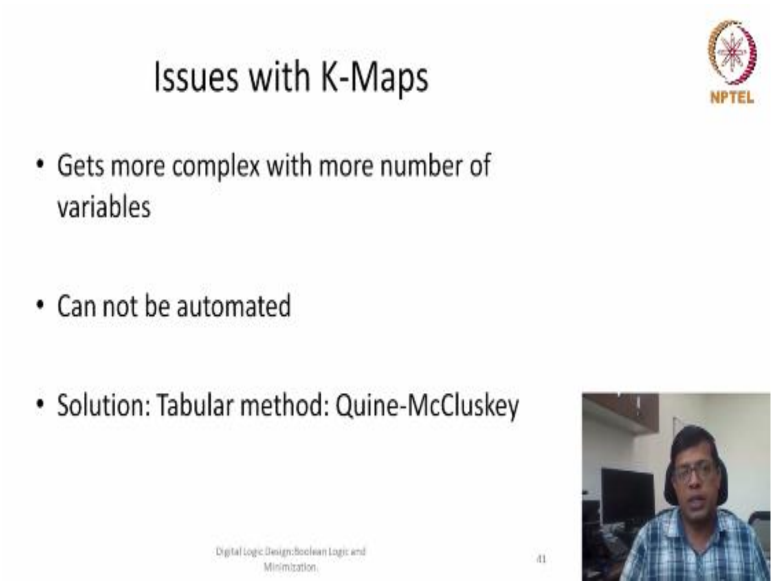
Digital System Design
Professor Neeraj Goel
Department of Computer Science Engineering
Indian Institute of Technology Ropar
QM-Method

Hello in our previous lecture we have seen how K maps can be used to optimize Boolean expressions, sum of minterms or product of maxterms.

(Refer Slide Time: 00:32)



Issues with K-Maps

- Gets more complex with more number of variables
- Can not be automated
- Solution: Tabular method: Quine-McCluskey

Digital Logic Design:Boolean Logic and Minimization. 41

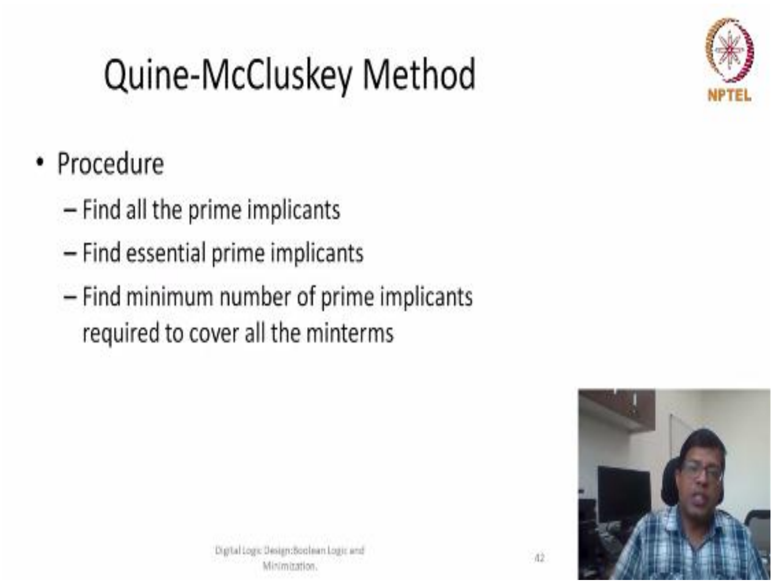
We have also seen that in K maps if the number of variable increases from 3 to 4, complexity increases, if 4 to 5, number of variable increases from 4 to 5 then complexity even increases. If number of variables are more then complexity is beyond control, the best advantage of K map that it is visually appealing and we can visually look for implicants and minimize the terms that will be lost as the number of variable increases.

Further, there is also when you will solve the problem with K maps then you will observe that sometime you will miss a particular optimization or particular combination which could otherwise have been there. So, because we rely on our instincts, we rely on our visual capability that whether these two terms can be combined or not. So this visual capability sometimes is limiting and it does not give us a full proof method.

Other issue is that when you are using K maps and let us say you would like to create a computer program you would like to create a computer program which can help you in optimizing, it would be difficult it would take little more time, it would be more number of lines of code which will help you in combining the terms and identifying which one is important and which one is let us say prime implicant.

So that is why there has to be some other method which can help us in automating and which can also help us in whatever number of variables are there it could be effective. So with that purpose one another method has been proposed by a researcher which is quite popular is called Quine McCluskey or in short it is also called QM method.

(Refer Slide Time: 02:38)



Quine-McCluskey Method

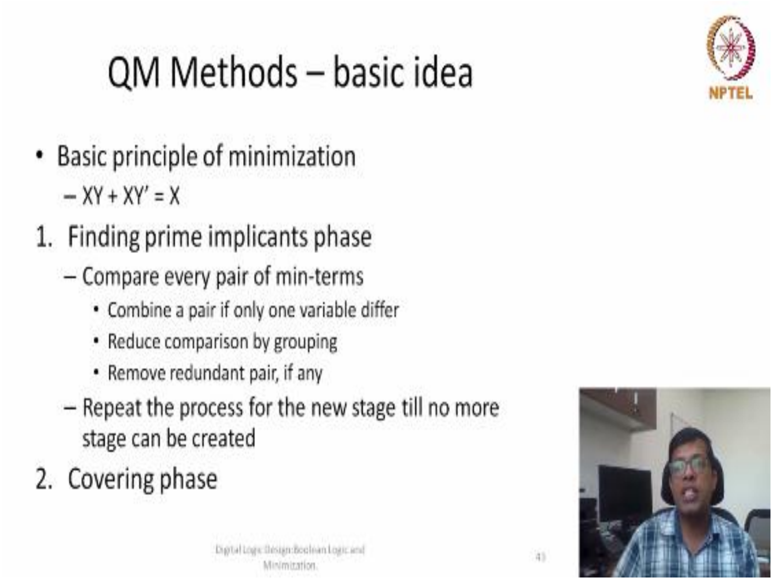
- Procedure
  - Find all the prime implicants
  - Find essential prime implicants
  - Find minimum number of prime implicants required to cover all the minterms

Digital Logic Design:Boolean Logic and Minimization. 42

So in Quine McCluskey method, the procedure is essentially same whatever we were doing in K maps we are doing the similar things, but in a more tabular method or in a text based method rather than a graphical method. So, if we summarize what were the steps in K map in optimization, Boolean optimization using K map the first thing we were doing we were filling the K map with all the 1s or with all the 0s and the second one was we were grouping them to form what could be the prime implicants.

And then we were trying to find essential prime implicants means those prime implicants which are very essential because a particular minterm is covered by only one particular prime implicant and then finally giving an optimization term. So, the same method can be utilized rather than in graphical but in a tabular method. So this method called QM or Quine McCluskey method.

(Refer Slide Time: 03:49)



QM Methods – basic idea

- Basic principle of minimization
  - XY + XY' = X
1. Finding prime implicants phase
   - Compare every pair of min-terms
     - Combine a pair if only one variable differ
     - Reduce comparison by grouping
     - Remove redundant pair, if any
   - Repeat the process for the new stage till no more stage can be created
2. Covering phase

Digital Logic Design: Boolean Logic and Minimization

41

So in Quine McCluskey, the basic principle is still same that two terms can be minimized or combined if only one variable is different so let us say XY plus XY dash equal to X. So, because Y and Y dash are present and we add both of them then Y can be eliminated. So, this X could be only one variable it could be a set of variables, but essentially in two terms only one particular variable is different or one particular variable is present in complement as well as the original form than that particular variable could be eliminated.

So this is the basic idea. Now in this QM method again the method is similar so we have two phases. One phase we find all the prime implicants. So in this phase we are exploring what all implicants are possible so we enumerate all of them and in the second phase we cover all the minterms using these prime implicants we find that which prime implicants should be included in our final solution.

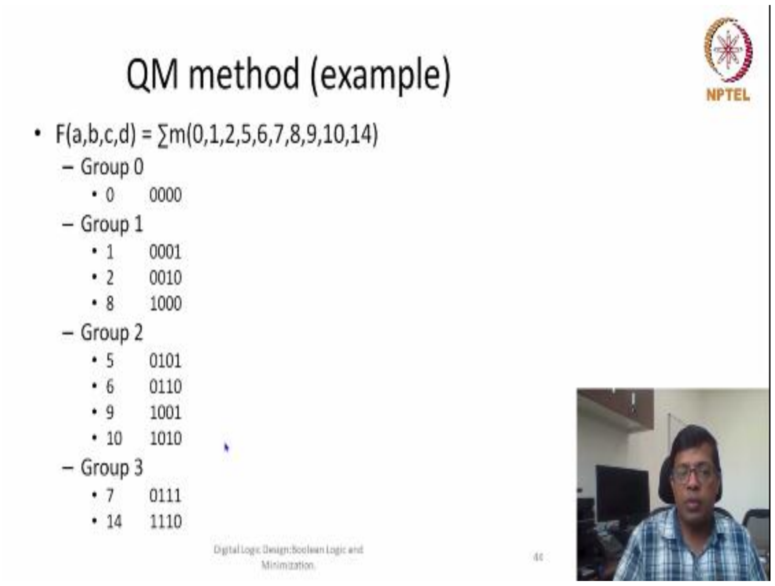
So to find what all prime implicants are there as a systematic approach what do we do? We compare each minterm with the other minterm. So, let us say there are n minterms which are present so there would be n into n minus 1 comparisons. So, we are comparing each minterm with the other minterm and then see whether this minimization is possible or not. So, to make it a little easier so n into n minus 1 become a very large number than a grouping is done.

So in this grouping number of comparison are reduced. By grouping, you see that two minterms could be combined only if variable is different. So, if two particular minterms has number of 1 different than more than 1 then it may not be able to group them. So, what is done in Quine McCluskey method that we arrange all the minterms in order of number of 1s present in those minterms. So, that help us in reducing the number of comparison.

So, for example, we compare only with the group of minterms which has only 0, only number of 1s and with the other group of minterms which has two number of 1s in them. So, we will see all these things with example and then after doing all the comparison sometime if some particular pair is redundant then we can remove here at this stage. So, this process would be repeated like after comparing we will get the second stage where one variable has been eliminated.

And then we can have another stage where two variables has been eliminated and there could be third stage where three variables has been eliminated. So, the total number of stages would be equal to maximum number of variables, so that way till all the possibilities that all the variables has been eliminated if that finishes, so we also finish this finding of our prime implicant phase. Once all the prime implicants has been computed, found and then we start the covering phase. In the covering phase we see that each minterm should be at least covered by one of the prime implicant. So let us try to understand this whole process with the help of an example.

(Refer Slide Time: 07:49)



QM method (example)

- F(a,b,c,d) = ∑m(0,1,2,5,6,7,8,9,10,14)
  - Group 0
    - 0 0000
  - Group 1
    - 1 0001
    - 2 0010
    - 8 1000
  - Group 2
    - 5 0101
    - 6 0110
    - 9 1001
    - 10 1010
  - Group 3
    - 7 0111
    - 14 1110

Digital Logic Design: Boolean Logic and Minimization

44

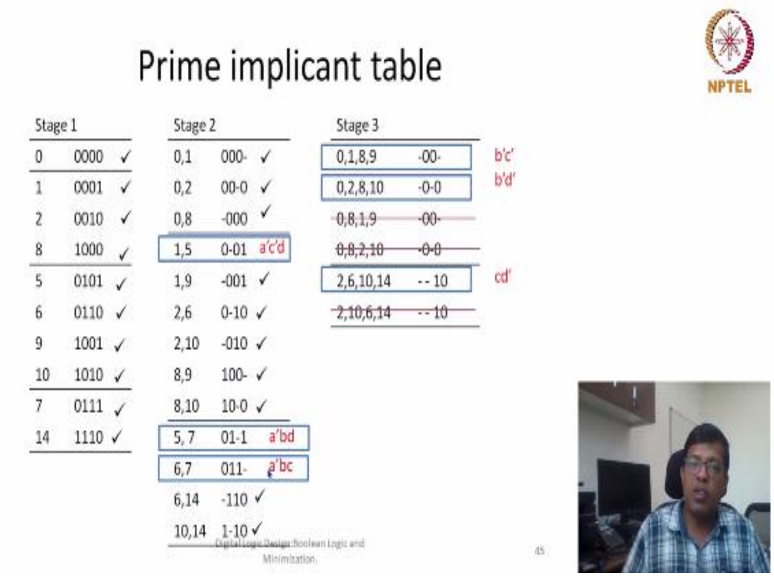
So let us take this function I have function F which has two variables a, b, c, d where a is of more significant and d is the least significant and the set of minterms are 0, 1, 2, 5, 6, 7, 8, 9, 10, 14. So as I said the ideal way would have been that we compare each of the minterm with the other minterm and see if there is a difference of only one variable. So, these comparisons would lead here so total number of variables here are 1, 2, 3, 4, 5, 6, 7, 8, 9, 10. So, 10 into 9 90 comparisons would be there in first stage itself.

So, to reduce the number of comparison we arrange them. So, we arrange in two ways; one we will arrange in based on the number of 1s, for example 0 will have no 1s so it group 0 and all these minterms 1, 2 and 8 has total number of 1s present in each minterm is 1. So 1, 2 and 8 so similarly 5, 6, 9 and 10 all of these minterms has total number of 1 in them is 2. Similarly, 7 and 14 total number of 1s which are present in these minterms are 3.

So, we are grouping them in the order of number of 1s present in each minterm and also it is sort of an ascending order. So first, would be the minimum number and after that in each group also we can arrange them in the ascending order. So, let us try to again I will repeat that why what is the logistic here. So see a group 0 can never be combined can never pair with a group 2 because there are already two 1s here there is no 1 here.

So, the condition for combining, condition for pairing is that only one variable should be different or one variable should be present in complimented as well as the original form. So, that means there is no possibility of grouping between known adjacent groups in this method. So group 0 can only combine with group 1 can only combine with group 2 or group 0, group 0 has already been done so we can discard that. So each group can compare with the other set of, the next set of group. So this is how we can combine and each time we are reducing one particular variable. So, let us see it with like more detail in this particular example.

(Refer Slide Time: 10:33)



Prime implicant table

Stage 1		
0	0000	✓
1	0001	✓
2	0010	✓
8	1000	✓
5	0101	✓
6	0110	✓
9	1001	✓
10	1010	✓
7	0111	✓
14	1110	✓

Stage 2		
0,1	000-	✓
0,2	00-0	✓
0,8	-000	✓
1,5	0-01	a'c'd
1,9	-001	✓
2,6	0-10	✓
2,10	-010	✓
8,9	100-	✓
8,10	10-0	✓
5, 7	01-1	a'bd
6,7	011-	a'bc
6,14	-110	✓
10,14	1-10	✓

Stage 3		
0,1,8,9	-00-	b'c'
0,2,8,10	-0-0	b'd'
~~0,8,1,9~~	~~-00-~~	
~~0,8,2,10~~	~~-0-0~~	
2,6,10,14	- - 10	cd'
~~2,10,6,14~~	~~- - 10~~	

Digital Logic Design: Boolean Logic and Minimization.

45

So first in the stage we will write all the minterms and also we will write that this is group 0, group 1, group 2 and group 3. So, we are not explicitly writing group 0, 1, 2, 3 it is there in our mind and these horizontal lines can help us in separating that this is the boundary between group 0 and group 1 and this is the boundary between group 1 and group 2. Group 2 means two number of 1s, group 3 means 3 number of 1s in each minterm.

Now to generate the pairs for the second stage we will compare group 0 with group 1 and the terms which can be optimized, the terms which can be reduced will replace a dash with them. So for example 0 and 1 could be combined so we will write pair 0 and 1 and 000 and because 0 and 1 both are present so this can be removed and we are replacing it with dash. So, similarly 0 and 2 can be combined and 0 and 8 can also be combined.

So, with this because all the minterms of group 0 has been compared with all the minterm of group 1 we are done with the first group of stage 2. For the next group of stage 2, we have to compare all the minterms which are present in group 1 with all the minterms which are present in group 2. So, 1 would be compared with 5, 6, 9 and 10. So because 1 is compared with 5 we can see that 1 and 5 can be combined because this particular variable is present as 0 as well as 1 so we can eliminate and we can write it as 1, 5 means 0 dash 0 1.

Similarly, 1 but 1 and 6 cannot be combined because 1 and 6 we see there are three variables which are different. However, 1 and 9 can be combined. So, we write 1 and 9 here and the difference in 1 and 9 is only at the topmost bit otherwise 0, 0, 1 is same for both of them. So, whatever the variable which is different we are replacing that variable with the dash. Now in the similar way 2 is compared with 5, 6, 9 and 10.

And wherever it find only one variable difference that particular pair would be written in this particular group. So, similarly, now this whole group would be completed by comparison of this that means these three variables has to be compared with this 4 minterms so that means 3 into 4 12 comparisons would take place and out of 12 comparisons, we see that we can have 6 pairs, we got 6 pairs out of these comparisons.

Similar to this so you see here also the total number of 1s in this whole group is 1 so we can also call it this group 1. Now similar to creating this group 1 we can go ahead and create group 2 by comparing group 2 with group 3. So, group 2 and 3 of stage 1 can be compared with group 3 of stage 1 to eventually lead to group 2 of stage 2. So you see 5 and 7 can be compared so if I compare 5 and 7 0101, 0111.

So you see only this particular variable is different here otherwise all other variables are same. So 5 and 7 could be grouped and we can write it as 01 dash 1 and similarly 6 and 7 could be combined, 6 and 14 could be combined and 10 and 12 could be combined. So one more observation here if we see this observation then you see that the difference between two minterms, so that they can be combined is only 1 or 2 or 4 or 8 so we can recheck from all the pairs.

So the difference between two minterm is 1, 2, 8 or 4. So this is true for all of them. So this gives us another hint while doing comparison whether the number of if we do comparison among these groups. So this idea can also be taken that if the difference of the minterm, if the difference of minterm, the lower minterm and the higher minterm, so basically yes lower means smaller in value and larger in value.

So, smaller minterm and larger minterm the difference is the power of 2 then that means they can be combined, but the condition for this combination is that they should be created in form of these groups and only the larger number can be subtracted from the smaller number and then the difference should be power of 2. So then they can always be combined. So with this, yeah, now after this stage 2 we can also keep on doing this process for stage 3 or stage 4.

So in stage 3, we again compare whatever are the pair of minterms or whatever implicants we get in this minterm in this group with the whatever implicants we have in the second group, zeroth group and first group. Now we are comparing 01, all the pairs of this group with all the pairs of this group. So if we compare 0, 1 with 1 and 5 because here dash appears at different place so we cannot combine it.

So, they had the condition is dash should appear at the same place and there has to be one variable difference. Now 0, 1 and 1, 9 can also not be combined because dash is appearing at different place. 0, 1, 2, 6 cannot be combined. 0, 1, 2, 10 also cannot be combined because of same reason, but 0, 1 and 8, 9 can be combined because first of all dash is appearing at the same place and there is only one variable difference, 0 and 1.

So 0, 1, 8, 9 we create a new combination here or new pair here and the term we will write it as because 0 and 1 is here so we replace this particular variable with dash dash 0, 0 dash. So, now 0, 1 and 8, 10 cannot be combined because dash is appearing at different place. Similarly, 0, 2 would be compared with all of them. Now, if you see 0, 2 can be combined only with 8, 10.

So because dash is appearing at same place and the value then will be 0, this variable is present as 0 and 1 so this is more significant one. So, we can write it as dash 0 dash 0. So, similarly now 0, 8 would be compared with all of these variables, sorry, all of these groups; 0, 8 would be compared with all the pairs present in this group. So, now if we see 0, 8 is compared to make things faster so see that wherever dash would appear at same place. So for example 0, 8 and 1, 9 dash is appearing at same place so we can combine them it will become dash 00 dash.

Similarly, 0, 8 would be combinationable with 2, 10 because dash is appearing at same place so it will become dash 0 dash 0; dash 0 dash 0. So, this is how when all the options would be exhausted then we will move on the comparison with the second group. So, we will compare all the pairs present in this group with all the pairs present in this group. So, from comparison of this pair to this pair we have lead, we got these two combinations which are there.

So basically 2, 6 could be combined with 10 and 12 because you see dash is appearing at same place and there is a difference of this variable more significant variable so it become dash dash 10. So, similarly 2, 10 and 6, 14 can also be combined. So during this combination we can also see whether the same principle hold or not. So, you see that the difference between this pair and this pair is again power of 2.

So 0, 1 and let us say 8, 9 the difference is power of 2 that means 8 and similarly if we see the difference between 0, 2 and 8, 10 is again a power of 2. So, first of all the difference between each pair it has to be same and it has to be power of 2 so that means they can be combined. So, why I am telling the difference part because if we use this difference part, so that means we can write a systematic algorithm to find out all these implicants at stage 2 or all the implicants at stage 3 or stage 4.

So the process could be algorithmically sound and you need not to, computer need not to write all these tables in the way we are writing. So computer can simply create these pairs by looking at whether the difference is more than, difference is equal to power of 2 or not. So after stage 3, we again look for if there is a possibility of stage 4 or not. So for that we compare all of these implicants, all of these pairs with all of these pairs, so which is now we do not see any possibility of combination because dash dash is appearing at this place and here in none of the implicant dash dash appears at this place.

So stage 4 possibility is not there. So that is why we stop it here and now we will try to remove the redundant terms. So if we look at the redundant terms you see this particular set 0, 1, 8, 9 is also equivalent to 0, 8, 1, 9 so we can simply remove this and similarly we can remove 0, 8, 2, 10 because it is equivalent to 0, 2, 8, 10. Similarly 2, 6, 10, 14 is equivalent to 2, 10, 6, 14 so one of them can be removed.

So, this way we have removed all the redundant terms. Now, let us see the list of our prime implicants. To find the list of prime implicants what we also do is we will go from the higher stage to lower stage and we see that if all the implicants there are covered by the larger set then we can simply ignore them. So to see that first we will try to see whether 0, 1 has been covered by something there in the stage 3 or not yes 0, 1 has been covered here. So, similarly 0, 2 has been covered by this particular implicant.

So this also has been covered 0, 8 has been covered by first implicant of stage 3. 1, 5 has not been covered so 1, 5 we will not tick, then we will see for 1, 9 is covered by a first implicant of stage 3. 2, 6 is covered by first implicant of group 1 of stage 3. Similarly, now we can see 2 and 10 has been also covered, it has been covered here as well as it has been covered here as well as it has been covered here. 8 and 9 is also covered by the first implicant. 8 and 10 is covered by this implicant, 5 and 7 we do not see here so it is not covered, so that means this also become a prime implicant.

Similarly, 6 and 7 is not covered by any of these implicant they also become prime implicant, 6 and 14 is covered by this implicant so we can tick this also and 10 and 12 has been covered by this implicant, 10 and 14 has been covered by this implicant so we can tick this also. Whatever is ticked so that means that they are not prime implicant because of larger set exist where these implicants are only subset.

So prime implicant are the superset which cannot be combined further. So because 0, 1, 0, 2 all of these could be combined further, so they are not prime implicant. However, 1, 5, 5, 7 and 6, 7 are prime implicant because they could not merge further or group further into a larger implicant. So, from the stage 2 similarly we had to go to stage 1 and we have to see whether all the minterms has been covered by at least one of the prime implicant or not, one of the implicant or not.

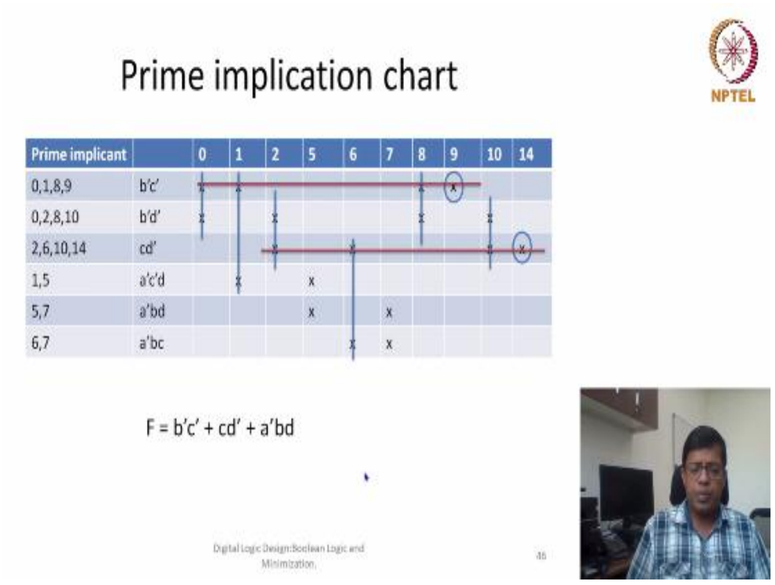
So 0, 1, 2, 8, 5, 6, 9, 10, 7, 14 all of them has been covered, so we can simply tick all of them so that means these minterms are not prime implicant or there is a larger implicant which is present, which can, for which these minterms are subset of. So after looking at these now we can clearly mark what are our prime implicant. So our prime implicants are 0, 1, 8, 9; 0, 2, 8, 10 and 2, 6, 10, 14; 1 and 5, 7 and 6 and 7.

So after finding the prime implicants we can also see what would be their minterm what would be the product expressions for each of these groups. So, let us consider the first variable is d and the second is c and then b and then a. So, we can write this particular product term as b dash c dash and this one could be written as b dash d dash and this one can be written as cd dash. This one can be a dash c dash and d.

And this one can be written as a dash b and d and this one can be written as a dash b c. So this is how we can find the product term for each of these groups. So with this our first stage is over. So, the summary of this first stage and if we compare ourselves with the K map method. So in this method we have exhaustively searched all the implicants and then we have exhaustively search for what are our prime implicants?

So because the method is systematic we are comparing each and every term. There is no chance that we are going to miss any implicant. The second thing is because the comparison can be written nicely in terms of an algorithm. So, this could also form a nice optimization. So this optimization can be written using a systematic algorithm in any of the high level language C, C plus plus, Java etcetera. So with this finish, like after finishing this prime implicant table now we find that which prime implicant should be included in our final solution.

(Refer Slide Time: 27:47)



Prime implication chart

Prime implicant		0	1	2	5	6	7	8	9	10	14
0,1,8,9	b'c'	x	x					x	x		
0,2,8,10	b'd'	x		x				x		x	
2,6,10,14	cd'			x		x				x	x
1,5	a'c'd		x		x						
5,7	a'bd				x		x				
6,7	a'bc					x	x				

F = b'c' + cd' + a'bd

Digital Logic Design:Boolean Logic and Minimization. 46

To do that, we create a prime implication chart. Prime implication chart means which prime implicant should be added. Here, in all the rows we write all the prime implicants which are there we are also writing their corresponding product terms and then for each column we write all the minterms. To make things easier for each minterm which are present in that particular prime implicant we mark a cross sign.

So for example 0, 1, 8, 9 is a prime implicant so we can mark cross sign for minterm 0, 1, 8 and 9. Similarly, 0, 2, 8 and 10 so we can mark corresponding to minterm 0 we can mark cross and then again for 2, for 8 and for 10. Similarly, we mark the presence of all the minterms which are present in a particular implicant.

So after creating this prime implicant chart the purpose of this prime implicant chart is that all the minterms should be covered by minimum set of our prime implicants. I will repeat what do we want? We want a minimum solution, in the minimum solution the total number of prime implicant should be minimized. The other condition is all the minterm should have been covered by at least one prime implicant.

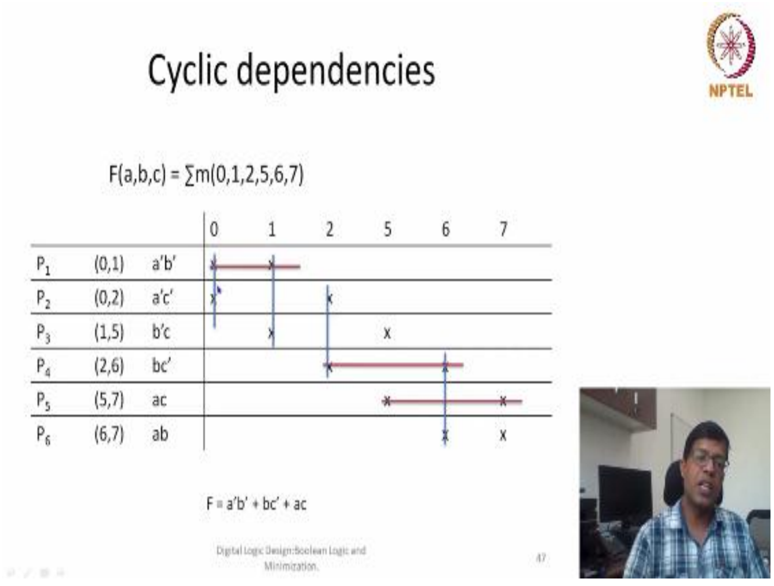
So this at least one prime implicant gives us a hint that those minterms which are covered by only prime implicant they are the most important prime implicant, so they become essential prime implicant. So by looking at this table we see that in which minterm there is only one tick. So 0 is covered by two prime implicant, 1 is covered by two prime implicant, similarly so 2, 5, 6, 7, 8, 9 is covered by only one implicant.

10 is also covered by two implicants and 14 is covered by one implicant. So essentially the implicant which are covered, minterms which are covered by only one implicant we circle them so any implicant which is covering only this particular minterm is essential, essential means we have to take them in our final solution. So, because we have to take them in our final solution, so what we can do is we can put a cross here.

So like this we put a cross here for 2, 6, 10, 14 and similarly we put a cross for 0, 1, 8, 9. So putting a cross means that anything which has been covered, so any minterm which has been taken care by this particular prime implicant we need not to worry about them. So, we can put a horizontal line also that 10 we had taken care of, 8 we had taken care of, 6 we had taken care of, 2 we have taken care of, 1 and 0 also we had taken care of. So what crosses are left? So that means those minterms we have to find that they also need to be covered.

So, we see 5 and 7 these are the two minterms which are left and what is the best possibility just by looking at it we can say that 5 and 7 this particular implicant is a better possibility because by including only one implicant, one prime implicant both these terms could be covered. So our final solution could be we can say that these three are the terms which we will say that b dash c dash plus cd dash and a dash bd, so this could be our final solution. So this is how the covering phase would finish. And the method is generic it could be applied to all other possibilities. Now sometime whenever we are covering some special challenges can come. So for example we do not see any term which creates two essential prime implicants.

(Refer Slide Time: 32:04)



Cyclic dependencies

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

			0	1	2	5	6	7
P_1	(0,1)	a'b'	x	x				
P_2	(0,2)	a'c'	x		x			
P_3	(1,5)	b'c		x		x		
P_4	(2,6)	bc'			x		x	
P_5	(5,7)	ac				x		x
P_6	(6,7)	ab					x	x

F = a'b' + bc' + ac

Digital Logic Design:Boolean Logic and Minimization.

47

So let us say for this particular example my function is abc, it has only three variables, a is the more significant one and total number of minterms 0, 1, 2, 5, 6, 7. So I have already created what are the possible implicants, prime implicants and I have also created this prime implication chart. So, in this prime implication chart these are the terms, they are the prime implicants P1, P2, P3, P4, P5, P6.

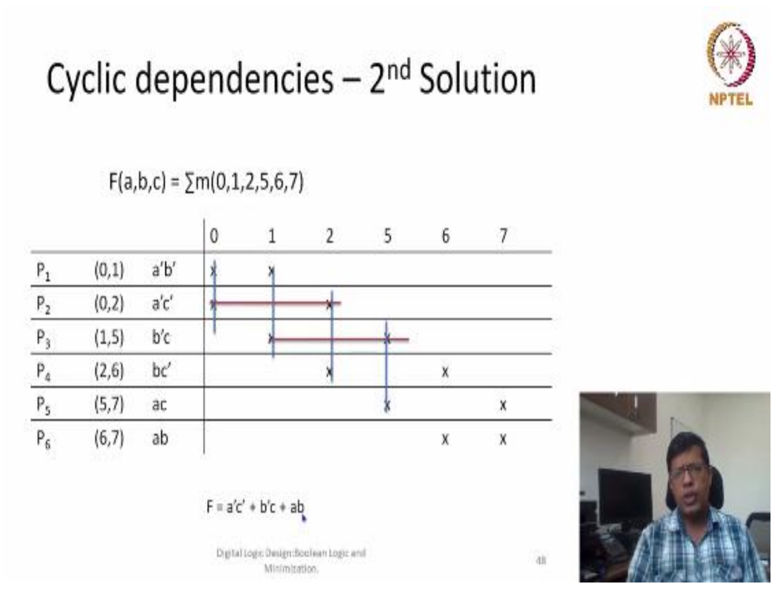
And if I look at this prime implication chart, I see that all the minterms are covered by at least two of the prime implicants. So, because all the terms are covered by two prime implicants, so we do not know which one is essential and how do we go forward. So, in such cases one idea could be that we do a coin toss and we randomly select the first one. So, let us say I had to make a choice for the 0th minterm.

So for 0 minterm I have two choices either I select 0, 1 or basically a dash b dash or I select a dash c dash. Now let me say that I would like to select the first one a dash b dash. So, because I am selecting a dash b dash, so because I am selecting a dash b dash so the process would be that I will take care of 0 as well as this particular implicant will take care of minterm 0 as well as minterm 1.

So because both 0 and 1 minterm has been taken care, so I can remove I can put a horizontal line corresponding to this and I can put a cross here so that means 0 and 1 has been taken care of. So out of the remaining minterms 2, 5, 6 and 7 just by looking visually I can see that the next possible way would be if I select 2 and 6 so basically b c dash. So, because b c dash would be able to cover two minterms otherwise both of this P2 or P3 would cover only one minterm.

So the only choice we will have is P4. So, we will take P4. Now correspondingly minterm 2 and minterm 6 can also be removed and after removing minterm 2 and 6 we are left with minterm 5 and 7 and we see that can be covered by prime implicant 5 or 5, 7 or ac. So, the total solution will be, sorry, so the total solution would be a dash b dash bc dash and ac. So this is the way we can, so again for the same particular, same problem let us look at the other solution let us say for 0th minterm instead of selecting P1 if I select P2 what would happen? So let us see.

(Refer Slide Time: 35:27)



Cyclic dependencies – 2nd Solution

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

			0	1	2	5	6	7
P_1	(0,1)	a'b'	x	x				
P_2	(0,2)	a'c'	x		x			
P_3	(1,5)	b'c		x		x		
P_4	(2,6)	bc'			x		x	
P_5	(5,7)	ac				x		x
P_6	(6,7)	ab					x	x

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

Digital Logic Design: Boolean Logic and Minimization.

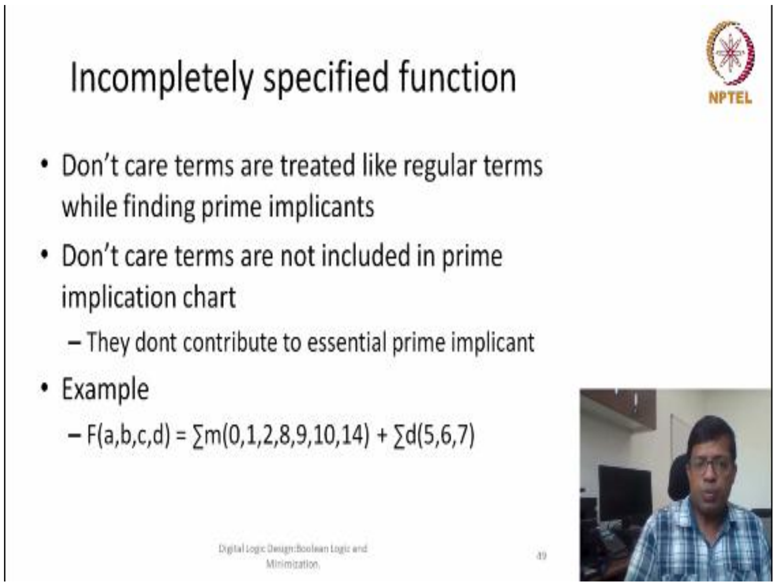
48

So in this case I select P2 for covering my minterm 0. So if I select P2 that means I have to remove these two minterm 0 and 2 from the table. So, if I remove 0 and 2 the remaining minterms are 1, 5, 6 and 7. So out of that I see 1 and 5 is the better possibility because it is able to cover two different minterms so then I will select 1 and 5 and after selecting 1 and 5 I can remove minterm 1 and minterm 5 from the chart.

And out of the remaining minterm 6 and 7 I see I can use P6 that means 6 and 7 are ab as in my solution. So, in this particular case I came up with two different solutions which are entirely different, but both of them are equivalent because in both the cases total number of prime implicants in my final solution is 3 and in both the cases total number of literals is equal to 6.

So, they are equal in weight, they are, both of them are equally optimal solutions. So, this is how sometime we have to do with a cyclic dependency or so sometimes this guess based solution or trial and error based solutions they could lead us to slightly suboptimal solutions. So for that also mathematically we can have some sort of, some other elaborate algorithms which can give us optimal solutions. So, we are not covering those details in this particular lecture. So, this is how we do for this kind of a cyclic dependency cases.

(Refer Slide Time: 37:40)



Incompletely specified function

- Don't care terms are treated like regular terms while finding prime implicants
- Don't care terms are not included in prime implication chart
  - They dont contribute to essential prime implicant
- Example
  - F(a,b,c,d) = ∑m(0,1,2,8,9,10,14) + ∑d(5,6,7)

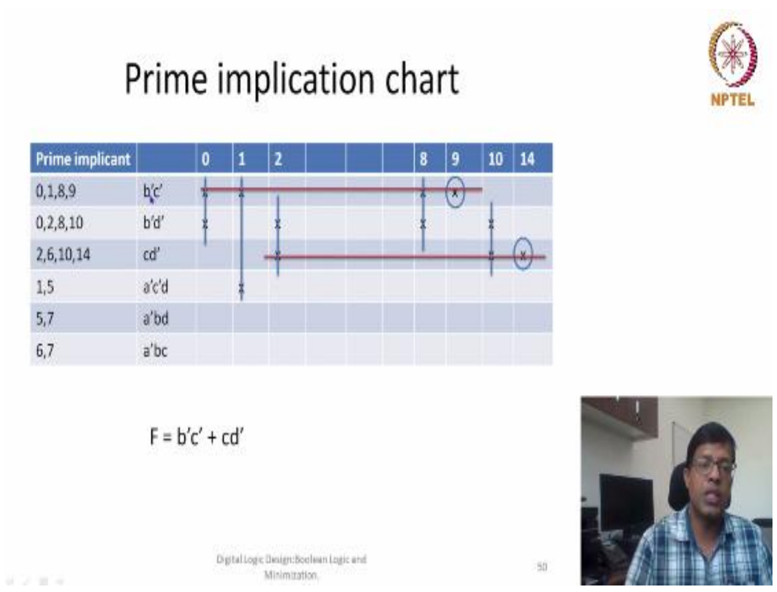
Digital Logic Design:Boolean Logic and Minimization. 49

Now let us see another case also when our problem or function is not completely specified. What does that mean? So there are some don't care conditions. So, what would we do if there are some don't care conditions? In case of don't care conditions, so you see that in QM method there are two phases. In first phase, we try to find all the prime implicants, in the second phase we are covering all the minterms.

So, in case of incompletely specified solution the first phase there is no difference we treat all the don't care conditions as our regular conditions or regular minterms, but in the second phase while creating the covering chart or while creating prime implication chart we do not list these don't care terms. So because don't care terms are not listed so they do not contribute to essential prime implicants.

So, what I will is, I will take the same example which we have consider for explaining our QM method, but out of this method, so what I have done is I have taken couple of terms which I created as don't care terms. So, let us say the total number of terms otherwise were 10 and out of 10 I created 5, 6, 7 these three minterms as a don't care terms. So if I do so because 5, 6, 7 are treated as a regular minterm for first exercise of creating all the implicants.

(Refer Slide Time: 39:29)

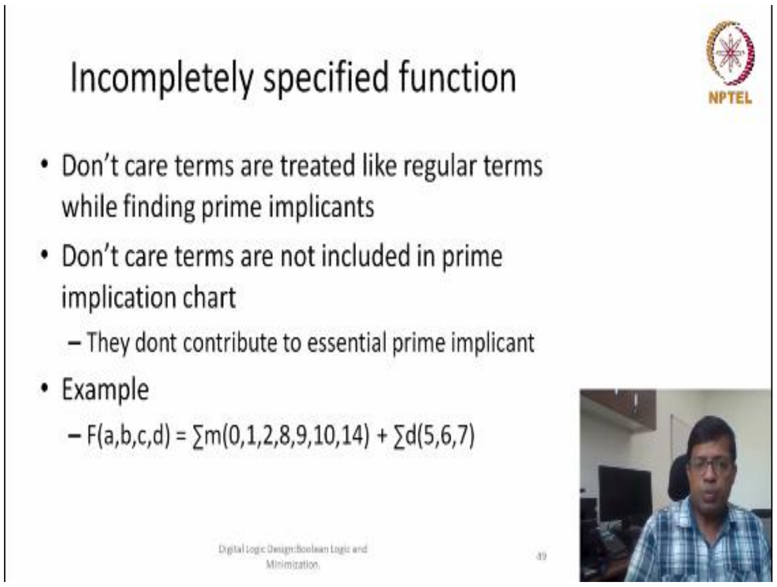


Prime implication chart

Prime implicant		0	1	2				8	9	10	14
0,1,8,9	b'c'	x	x					x	ⓧ		
0,2,8,10	b'd'	x		x				x		x	
2,6,10,14	cd'			x						x	ⓧ
1,5	a'c'd		x								
5,7	a'bd										
6,7	a'bc										

F = b'c' + cd'

Digital Logic Design:Boolean Logic and Minimization. 50



Incompletely specified function

- Don't care terms are treated like regular terms while finding prime implicants
- Don't care terms are not included in prime implication chart
  - They dont contribute to essential prime implicant
- Example
  - $F(a,b,c,d) = \sum m(0,1,2,8,9,10,14) + \sum d(5,6,7)$

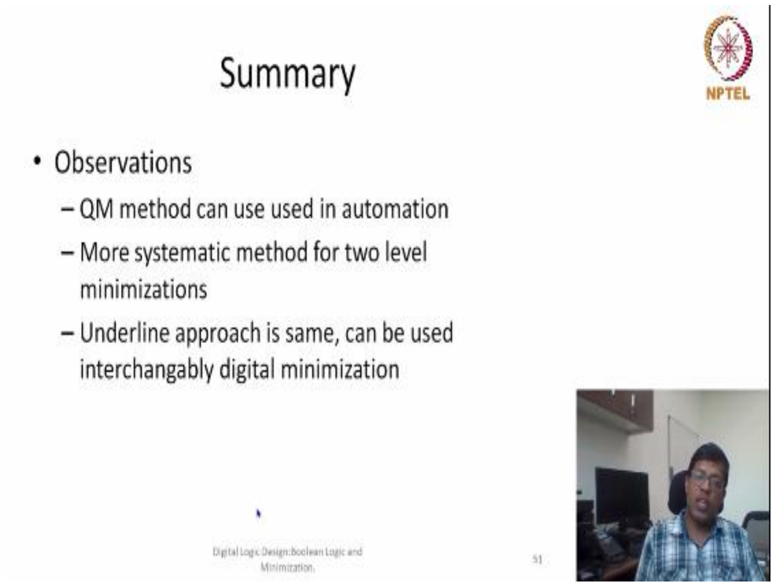
Digital Logic Design:Boolean Logic and Minimization. 49

So I can create the prime implication chart in the similar way as the previous examples. So these are the list of all of my prime implicants, but this three minterms 5, 6 and 7 are not listed in this prime implication chart because they are not listed in prime implication chart. So, they will not contribute to finding the maximum or the best cover. So, the process would still be same that looking at the chart, we will first look out that which particular minterm is covered by only one prime implicant.

So 0 is covered by two prime implicants, 1 is also covered by two prime implicants. If we see then we see 9 and 14 these are the two minterms which are covered by only one prime implicant. So, we can choose those prime implicants as essential prime implicants. So after choosing those implicants as essential prime implicants, now we can remove the minterms which has been covered by these two prime implicants.

So after removing all these implicants now we see that essentially all the minterms has been already covered. So, this will become the final solution, final solution would be these two essential prime implicants and the product term corresponding to these prime implicant that means b dash c dash plus c d dash. So, this is the final solution. So with this, we finish with this QM method of logic minimization.

(Refer Slide Time: 41:21)



Summary

- Observations
  - QM method can use used in automation
  - More systematic method for two level minimizations
  - Underline approach is same, can be used interchangably digital minimization

Digital Logic Design: Boolean Logic and Minimization.

51

And we can summarize that this QM method is more systematic because it is exhaustive in nature you do not miss out any particular prime implicant and it is also systematic in finding essential prime implicants and doing the optimizations around that. So, this is with respect to K map we can see that it is more systematic approach. The better part or another bigger advantage of this QM method is that we can write algorithms for it.

So we can write C, C plus plus code or we can write in terms of a high level language so that we can minimize whatever given number of minterms are there. So, another summary so far what we have seen whatever from the previous two, three lectures, so, another summary is that the underlying method for minimization is essentially same. We were using same methods when we were, similar combination theorems were used when we were using Boolean algebra.

And similar methods or similar theorems were used when we were doing combination or finding implicant or minimization using K maps, similar methods are used here. So, this knowledge, this understanding how two minterms or how two variables get combined or can how one particular variable will get optimized out of two product terms. So this particular method, this particular understanding can be used interchangeably also.

So whenever you are looking at two algebraic terms then also you can keep your K maps in your mind and then can help you, this can help you in doing optimizations, similarly when you are doing QM or K map based method then also those algebraic, that algebra should be there in your mind so that you can, you can do optimization in a better way, you can cross check your optimizations. So this is how we can do two level minimization.

So we call it two level minimization because the end result is always either sum of product or product of sum. So there would be two level of logic, AND gate and then OR gate, AND gate followed by OR gate or NAND gate followed by NAND gate because the total number of gates in all the solutions would be two, so it is called two level minimization. So with this, I will close this today’s lecture. Thank you very much.