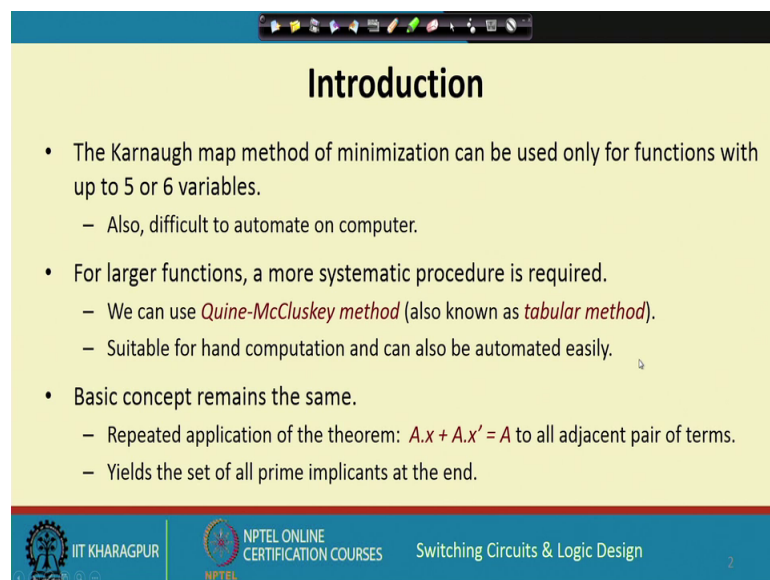


Switching Circuits and Logic Design
Prof. Indranil Sengupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture – 19
Minimization Using Tabular Method (Part -1)



So, we have seen; how we can use the Karnaugh map method to minimize switching functions. We had looked at 3 and 4 variable functions as examples and we have also mentioned that we can go up to maximum 6 not more than that. So, today, we shall be discussing some method which is sometimes call the tabular method which is more systematic in this respect.

(Refer Slide Time: 00:42)



Introduction

- The Karnaugh map method of minimization can be used only for functions with up to 5 or 6 variables.
 - Also, difficult to automate on computer.
- For larger functions, a more systematic procedure is required.
 - We can use *Quine-McCluskey method* (also known as *tabular method*).
 - Suitable for hand computation and can also be automated easily.
- Basic concept remains the same.
 - Repeated application of the theorem: $A.x + A.x' = A$ to all adjacent pair of terms.
 - Yields the set of all prime implicants at the end.

 IIT KHARAGPUR |  NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

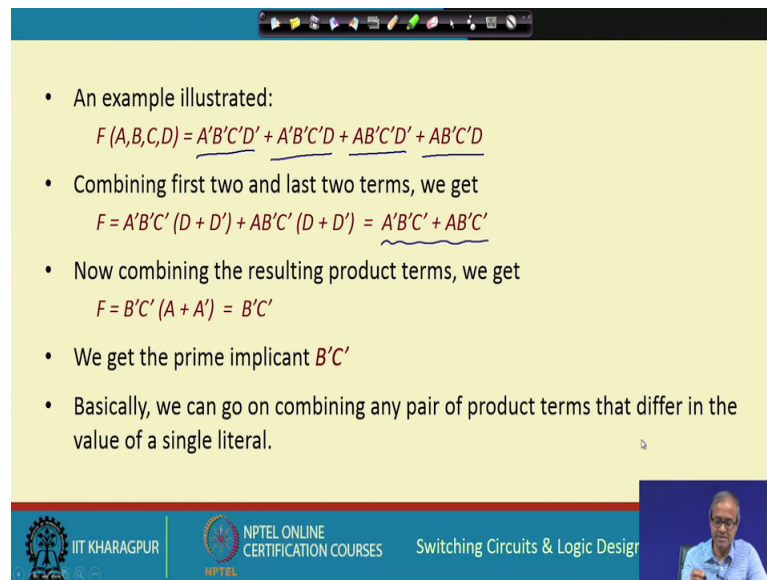
Let us first look at the motivation why we are trying to go for this? We have already mentioned that that in the Karnaugh map method; we can go up to maximum 5 or 6 variables. More importantly, it is not easy to automate on a computer, you see whatever you do, whatever design minimization we do today, today everything is done on computer, right, this Karnaugh map which you are doing with hand very nicely, we can visualize, we can write down the expression, but you think, if I have to write a computer program to do this, this will not be so easy, quite complicated finding out the visually the maximum size the cubes try to make them largest not so easy.

So, Karnaugh map is not so easy to program and automate ok, this is one problem. So, to remove this drawback, say for larger functions and also from the point of view of automation we need a more systematic procedure there is a systematic procedure which was proposed by Quine and McCluskey which is sometimes known as Quine McCluskey method because it uses some kind of a tabular structure.

It is also refer to as Tabular method sometimes, this method has an advantage that is can this can be automated easily and also you can also do it for hand computation as we shall illustrate with some examples, but automation is you can write a computer program for this in a much easier way, but what is the basic idea behind this approach? The idea is very simple. We already know an identity like this that any sub function multiplied by a variable x plus that sub function multiplied by \bar{x} , if you take a common x and \bar{x} goes, you get only A ; that means, one variable gets less becomes less.

So, if you repeatedly apply this simple rule to all adjacent adjacent pair means which differ in a single variable. So, if we repeatedly apply this the concept is whatever you get at the end, they will only with this setup prime implicants. So, you start with all your true minterms go and applying this rule repeatedly on wherever you can at the end where you see that you cannot apply this rule anymore, you will find that whatever you are left with they are the prime implicants and prime implicants are something which will make your final minimized expression some set of prime implicants, you have to select after that ok, this is basically two step process first generate all the prime implicants, second select the minimum set of prime implicants that can cover the whole function.

(Refer Slide Time: 04:13)



- An example illustrated:
$$F(A,B,C,D) = \underline{A'B'C'D'} + \underline{A'B'C'D} + \underline{AB'C'D'} + \underline{AB'C'D}$$
- Combining first two and last two terms, we get
$$F = A'B'C'(D + D') + AB'C'(D + D') = \underline{A'B'C'} + \underline{AB'C'}$$
- Now combining the resulting product terms, we get
$$F = B'C'(A + A') = B'C'$$
- We get the prime implicant $B'C'$
- Basically, we can go on combining any pair of product terms that differ in the value of a single literal.

Let us see how it works. So, a very simple example; let us first illustrate with the help of an algebraic expression consider a 4 variable function like this, there are 4 product terms or 4 minterms, you see there are some minterms which differ in one variable like let say $A\bar{B}\bar{C}\bar{D}$ and $A\bar{B}\bar{C}D$, they are adjacent and similarly $A\bar{B}B\bar{C}\bar{D}$, and $A\bar{B}B\bar{C}D$; they are also adjacent.

So, you can combine the first two and the last two terms you can take them common; so, D disappears same also D disappears, you get $A\bar{B}\bar{C}$, plus $AB\bar{C}$. Now you can go and applying the repeatedly, again, you see here again, you can apply this same thing, they are again adjacent they differ only in A .

So, you can again apply this same rule and you get finally, $B\bar{C}$, now what is $B\bar{C}$? $B\bar{C}$ is a prime implicant. So, the idea is as follows whenever you have this minterms, go on applying this rule whenever possible, how many times possible; ultimately you land up with something those will be the prime implicants, we go on combining any pair of product terms that differ in a single literal value this is the basic idea behind the tabular method which we shall be illustrating with some examples.

(Refer Slide Time: 06:10)

The Basic Concept

- Two k -variable terms can be combined into a single $(k-1)$ -variable term, if and only if they differ in only one literal.
 - We use the binary representation of the minterms for convenience.
 - Two minterms can be combined if their binary representations differ in only one position.
 - We use the symbol '-' to indicate the absence of a literal.
- An example:
 - $A'B'C'D$ denoted by 0001, $AB'C'D$ denoted by 1001.
 - After combining, we get -001.

Handwritten annotations on the slide include a circled dash symbol, a vertical subtraction of binary numbers (0001, 1001, -001), and a circled expression $B'C'D$.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

Let us now look at the basic concept; what we do in the tabular method, two k variable terms can be combined by an application of that rule that I mentioned to form a single k minus 1 variable term, here an example is given $A\bar{B}\bar{C}\bar{D}$, 4 variable term; there is another 4 variable term, you combine them what you get this A disappears you get $\bar{B}\bar{C}\bar{D}$, 3 variable term here k equal to 4, k was 4, after combining you get single k minus 1 variable term.

This is possible if and only if 2 terms differ in a single literal here, they were differing only in A and $A\bar{B}\bar{C}\bar{D}$ was common, right, for convenience, we use the binary representation $A\bar{B}\bar{C}\bar{D}$, we write as 0001 means $\bar{1}$ means uncomplimented. Similarly $AB\bar{C}\bar{D}$, we write as 1001. So, we have 0001 and 1001. So, wherever there is common, you keep them unchanged and wherever that is different, you put a dash ok.

So, you get dash 001 which means $\bar{B}\bar{C}\bar{D}$, A has vanished; this is the idea. So, in this binary representation, we are using this symbol dash to indicate the absence of a literal. So, in the Tabular method, we use such binary representation this dash to manipulate such minterms and apply these minimization rules.

(Refer Slide Time: 08:23)

The Tabular Method :: Identify all Prime Implicants

1. Arrange all the minterms in groups, based on the number of 1's.
 - The number of 1's in a term is called the index.
2. Compare every term of index i with each term in the group with index $(i+1)$.
 - Where possible, merge them using the rule: $A.x + A.x' = A$.
 - Place a check mark next to every term that has been combined with at least one term.

Diagram illustrating the process: Two groups of minterms are shown. The first group, labeled i and $i=1$, contains the minterms 1000 and 0100. The second group, labeled $i+1$, contains the minterms 1010 and 0110. A curved arrow points from the first group to the second, indicating comparison. Checkmarks are placed next to the minterms 1000 and 1010, indicating they have been combined.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

So, this steps of the tabular method broadly the first have that means, identification of the prime implicants that part of it, there are three steps. First is that we identify all the minterm minterm in true minterms, of course, mint the true minterms in groups based on the number of ones in the minterms and the number of ones in the minterms is called the index and what we do suppose I have two groups, let us say, I have made two groups; one with index i , I have made another group with index i plus 1 well just repeating what does index mean index means how many number of ones are there let us let us take an example let us say i equal to 1.




So, i equal to 1 means in this group of i I will be having minterms like 1000 0100 like this, there is a single one and i plus 1, I will be having 1010 0110 like this where there are two, two ones and what we do after that is that compare every term of the index i group, this group with each term every term with the group i plus 1, you compare means every element from here, with every element of here and check whether they differ in a single position or not and wherever possible, you apply this rule ok.

Now if you are able to apply this rule, for example, here there was 1000 here was 1010, it was differing in one position, you could have applied, if you are applying, you put a check mark against the two elements tick; that means, you have applied rule here right this is something you do.

(Refer Slide Time: 10:40)

The Tabular Method :: Identify all Prime Implicants

1. Arrange all the minterms in groups, based on the number of 1's.
 - The number of 1's in a term is called the *index*.
2. Compare every term of *index i* with each term in the group with *index (i+1)*.
 - Where possible, merge them using the rule: $A.x + A.x' = A$.
 - Place a *check mark* next to every term that has been combined with at least one term.
3. Now compare the terms generated in Step-2 in the same fashion; generate a new term by combining two terms that differ by only a single 1 and whose dashes are in the same position.
 - The process continues until no further combinations are possible.
 - The remaining unchecked terms are the *prime implicants* of the function.



Switching Circuits & Logic Design


And after you do this step to you. compare the terms which have generated after the step in the same fashion generate some new term by combining two terms that differ in a single one, again dash in the same position, I shall show an example that explains; what this mean and you go and repeating this combination process combining process until no further application of this rule is possible at that point all the terms which are not checked or not checked, they will be the prime implicants of the function; the point it notice that at every step you will be comparing the terms of the index i group with the index i plus 1 group.




(Refer Slide Time: 11:32)

Example 1: $F(w,x,y,z) = \sum (0, 1, 2, 5, 7, 8, 9, 10, 13, 15)$

	Step (i)				Step (ii)				Step (iii)			
	w	x	y	z	w	x	y	z	w	x	y	z
0	0	0	0	0 ✓	0,1	0	0	0 -- ✓	0,1,8,9	--	0	0 -- A
1	0	0	0	1 ✓	0,2	0	0	-- 0 ✓	0,2,8,10	--	0	-- 0 B
2	0	0	1	0 ✓	0,8	--	0	0 0 ✓	1,5,9,13	--	--	0 1 C
8	1	0	0	0 ✓	1,5	0	--	0 1 ✓	5,7,13,15	--	1	-- 1 D
5	0	1	0	1 ✓	1,9	--	0	0 1 ✓				
9	1	0	0	1 ✓	2,10	--	0	1 0 ✓				
10	1	0	1	0 ✓	8,9	1	0	0 -- ✓				
7	0	1	1	1 ✓	8,10	1	0	-- 0 ✓				
13	1	1	0	1 ✓	5,7	0	1	-- 1 ✓				
15	1	1	1	1 ✓	5,13	--	1	0 1 ✓				
					9,13	1	--	0 1 ✓				
					7,15	--	1	1 1 ✓				
					13,15	1	1	-- 1 ✓				

$\left. \begin{matrix} 0 \\ 1 \\ 2 \\ 8 \\ 5 \\ 9 \\ 10 \\ 7 \\ 13 \\ 15 \end{matrix} \right\} \text{no merging}$

The set of prime implicants are:
 $\{x'y', x'z', y'z, xz\}$



Switching Circuits & Logic Design


Let us take this example worked out and after that we shall workout an example by hand.

the font size is little small, but let us see here we have a function like this where these are the true minterms 0, 1, 2, 5, 7, 8, 9, 10, 13, 15, there are 4, 5, 6, 7, 8, 9, 10, you see first here in the first column, what we show here is we write down this 10 minterms, in the order of the number of one ones in their binary representation that is the index the first group consist of index 00 is there, 0, 0, 0, 0; second group is index 1. So, 1 index 0 is done, 0, 2 and 8, they have single ones 1, 2 and 8, then group with index 2, we have 5, 9 and 10, they are with index 2, then index 3, three ones we have 7 and 13 and finally, 15 which is index 4.

So, for an n variable function, let us say for an n variable function, the index can go from 0 up to n 0 up to n like here for 4 variable function, you can go from index 0 up to index 4, right, this is what we do first, then let us see what you have done we compare this group 0 and group 1 index 0 and index 1, this is 0 0 0 0, this is 0 0 0 0 and 0 0 0 1 that differ in one position you combine them, you get 0 0 0 dash and you put a tick here put a tick here you have applied them and here you write 0 comma 1 saying that 0 and 1, you have combined you go and comparing 0 0 0 0 you compare the next one, also 0 and 2, also you can compare differs in one position. So, it will be 0 0 dash 0 this. So, this also is ticked 0 and 8 can also be combined one position dash 0 0 0 tick. So, this comparison is done now you compare this with this.

So, all element of this you compare it all element of this see that which can be compared combine 0 0 0 one you can combine with 0 1 0 1, there is one gap ok. So, 1 and 5 can be combined 0 dash 0 0, this will become dash 0 and 1 0 1. So, you also put a tick here, similarly this 1 and 9 can become 0 0 0 1 and 1 0 0 1 dash 0 0 1 put a tick here similarly 2 and 10; 0 0 1 0 and 1 0 1 0 can be compared first one is different dash 0 1 0 tick 8 and 9 can be compared 1 0 0 0 and 1 0 0 1, it will be 1 0 0 dash all unit ticked 8 and 10 also can be compared already ticked, then they send this group same way 5 and 7 can be compared 0 1 0 1 and 0 1 1 1 position difference 0 1 dash 1 put a tick 5 and 13 can be combined put a tick 9 and 13 can be combined already ticked, similarly, 7 and 15 can be combined 13 and 15 can be combined.

So, from the first column, it generate the second column, repeat this process as long as you can, now start with this second column, same concept; now you have this 1, 2, 3, 4,

groups, compare first and second group, now here same thing 0 0 0 dash, you see where you can compare; you can compare.



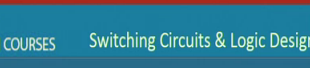

(Refer Slide Time: 16:13)

Example 1: $F(w,x,y,z) = \sum(0, 1, 2, 5, 7, 8, 9, 10, 13, 15)$

Step (i)				Step (ii)				Step (iii)				
w	x	y	z	w	x	y	z	w	x	y	z	
0	0	0	0	0,1	0	0	0	0,1,8,9	--	0	--	A
1	0	0	1	0,2	0	0	--	0,2,8,10	--	0	--	B
2	0	0	1	0,8	--	0	0	1,5,9,13	--	--	0	C
8	1	0	0	1,5	0	--	0	5,7,13,15	--	1	--	D
5	0	1	0	1,9	--	0	0					
9	1	0	0	2,10	--	0	1					
10	1	0	1	8,9	1	0	--					
7	0	1	1	8,10	1	0	--					
13	1	1	0	5,7	0	1	--					
15	1	1	1	5,13	--	1	0					
				9,13	1	--	0					
				7,15	--	1	1					
				13,15	1	1	--					

$\begin{array}{r} 000 - \\ 100 - \\ \hline -00 - \end{array}$

The set of prime implicants are:
 $\{x'y', x'z', y'z, xz\}$

This with here 1 0 0 dash you see 0 0 0 dash and 1 0 0 dash dash must be in the same place and 0 on different. So, if you combine, it is become dash 0 0 dash you see you will combine becomes dash 0 0 dash 0 1 and 8-9 are being compared 0 1 8 9. So, you tick this, you tick this 0 1, you cannot combine anything else 0 2 0 2 is here, 0 2 you can compare with 8 10. So, this also that is ticked 0 2 also get ticked 0 8, you can combine with 1 0 8 and 1 9. So, this will get ticked 0 8 1 9 and 0 8 also 2 10 0 8 2 10, this will also get ticked. So, this is done. Now similarly this and this with all of this you combine you compare with this, this is 0 dash 0 1 1 dash 0 1 you can combine this.

So, now this will also get ticked this. So, in this way you go and taking I am not showing everything you go and take in I shall workout in example later. So, you get this and after the here you see that after this step, you cannot combine anything else here, you see this and this you cannot combine anything more. So, whatever remains which are not ticked which you call A, B, C, D, these are the 4 prime prime implicants dash 0 0 dash means x bar y bar dash 0 dash 0 means x bar z bar dash dash 0 1 means y bar z dash one dash one means x z, these are all prime implicants. So, let me workout an example by hand, it will be clear to you.

(Refer Slide Time: 18:21)


Example 2: $F(A,B,C,D) = \sum(1, 2, 4, 8, 10, 14, 15)$

ABCD	
1	0001
2	0010 ✓
4	0100
8	1000 ✓
10	1010 ✓
14	1110 ✓
15	1111 ✓


2,10	-010
8,10	10-0
10,14	1-10
14,15	111-

PI

- A'B'C'D
- A'BC'D'
- B'CD'
- AB'D'
- ACD'
- ABC




IIT KHARAGPUR



NPTEL ONLINE
CERTIFICATION COURSES

Switching Circuits & Logic Design



Let us take an example like this it is simpler; there are 7 minterms 1, 2, 4, 8, 10, 14, 15. So, let me form the table like that.

So, these correspond to the 4 variables A B C D. So, I there is no index 0 term. So, index 0 is not there. So, index 0 is blank. So, index one; you have 1 2 4 and 8 1 is 0 0 0, 1 2 is 0 0 1, 0 4 is 0 1 0, 0 8 is 1 0 0 0, then you have index to index is a 1 10 only 10 10 is 1 0 1 0, index 3 is only 1 14, 14 is 1 1 1 0 and index 4 lastly 15 1 1 1 1, these are the initial groups. Now from here, let us form the next level groups, let us see let us compare these with this 0 0 0 1, 1 0 1 0, we cannot compare 0 0 1 0 and 1 0 1 0 you can compare they differ in one position. So, I put a tick here, I put a tick here and if I combine; what I get I combine 2 and 10 right 2 comma 10 what I get dash 0 1 0 dash 0 1 0.

Then you continue comparing, you cannot this 4 and 10, you cannot compare, but 8 and 10, you can compare, 8 and 10 is differ in 1 position again. So, you put a tick write 8 comma 10. So, it becomes 8 and 10 1 0 dash 0 1 0 dash 0 to this group is done. Now compare this and this 1 0 1 0 and 1 1 1 0, they can be merged differ in one position 10 comma 14, it becomes 1 dash 1 0, similarly 14 and 15; you can combined 14 and 15 1 1 1 dash these are all ticked, right, but these two are not ticked 1 and 4 and not ticked. So, you continue with this try to merge this and this you cannot merge anything because the dashes are not in the same position similarly here and here, you can you cannot merge

anything more ok. So, there are 1, 2, 3, 4, 5, 6, 6 prime implicants which are unticked or unchecked.

So, your prime implicants are in this case first one is 0 0 0 1 which is A bar B bar C bar D, second one is 0 1 0 0 A bar B C bar D bar, third one is dash 0 1 0. So, A is not there, B bar C D bar 1 0 dash 0 A B bar D bar 1 dash 1 0 A C D bar and lastly 1 1 1 dash A B C; these are the 6 prime implicants for this function and using this tabular method I have identified all the prime implicants, right.

Let us workout another example for your convenience; the head will clear for you.

(Refer Slide Time: 22:44)

Example 3: $F(A,B,C,D) = \sum(0, 3, 5, 8, 10, 13) + \sum_{\phi}(2, 7, 11)$

0 0000 ✓	0,2 00-0 ✓	0,2,8,10 -0-0 ✓	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <p>A'BC A'CD A'BD BC'D B'D' B'C</p> </div>
2 0010 ✓	0,8 -000 ✓	2,3,10,11	
8 1000 ✓	2,3 001- ✓	2,3,10,11 -01- ✓	
3 0011 ✓	2,10 -010 ✓		
5 0101 ✓	8,10 10-0 ✓		
10 1010 ✓	3,7 0-11 ✓		
7 0111 ✓	3,11 -011 ✓		
11 1011 ✓	5,7 01-1 ✓		
13 1101 ✓	5,13 -101 ✓		
		<u>Six PI</u>	

IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

Switching Circuits & Logic Design

Let us take this example in addition, I am showing some do not cares. So, this do not cares also I am including in my table to form the prime implicants. So, in the first step do not cares also will be included. So, I am showing the table again this A B C D, I am not writing same way. So, there is a 0 index 2 minterm here 0 0 0 0 and 1 index there is who is one index.

It is here and 2 is here 2 and 8 2 is 0 0 1 0 single 1 8 is also single 1 1 0 0 0, then 2 2 index 2 and with 3 5 and 10 3 is 0 0 1 1 5 is 0 1 0 1 10 is 1 0 1 0, then index 3; we have 7, 11 and 13, remaining 3 7, then 11 and 13 that is it. So, you continue with the same process go and combining try to combine this group with this group 0 and 2 gets

combined, see you tick here, tick here, it become 0 0 dash 0 0 and 8 gets combined dash 0 0 0, this is the first group, then do with this and this are ticked.

So, two you try to merge 2 and 3 can be combined 2 and 3 becomes 0 0 1 dash, then two 2 10 can be combined dash 0 1 0 and then 8 and 10 can also be combined, but it does not add any more tick 8 and 10 to be 1 0 dash 0. Now in the next 1 3 and 7 can be combined 3 and 7 3 7 is 0 dash 1 1, then 3 and 11 can be combined; 3 and 11 dash 0 1 1, then 5 and 7 can be combined, 0 1 dash 1, then 5 and 13 can be combined dash 1 0 1.

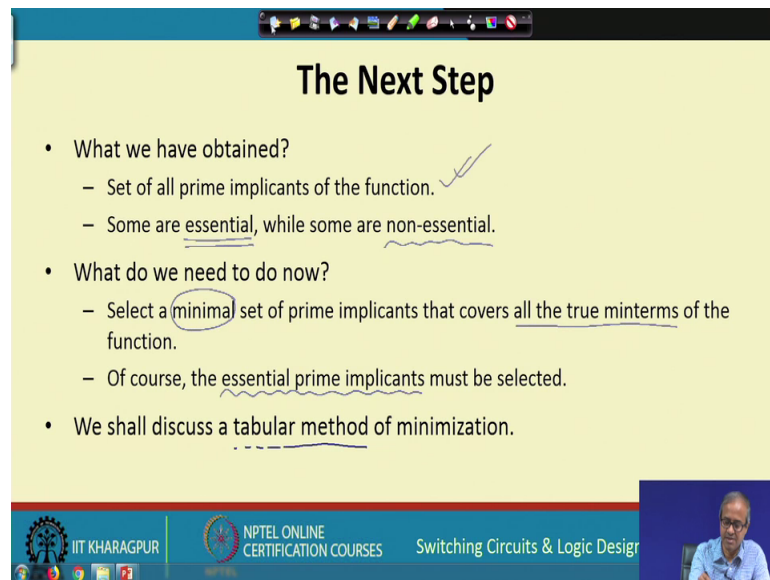
So, you have this step, now you continue with that in the next step, you see that where you can combine this group and this group here you can combine 0 2 and 8 10 0 2 and 8; 10. So, you get 0 2 8 10, you get dash 0 dash 0 dash 0 dash 0, similarly 0 8 and 2; 10, you can combine you will get this same thing, they will also get ticked and 2-3 means and similarly, if you combine this group with this group, they will be another one generated 2 3 10 11; 2 3 10 and 11; 2 3 with 10 11 0 dash dash 0 1 dash dash 0 1 dash yeah.

So, you see whatever there is untick now ah. So, is it correct I think this is not correct let check it ones 0 2 8 10, it is done, then here whatever 2 10 and 3 2 10 and 3 11 ok, 2 10 and 3 11 can be combined. So, it be 2 3 10 11, that is fine, it was right 2, 3, 10, 11; 2 3 10; 2 3 10 11; it is dash 0 1 dash 0 1 yeah, this is right fine. So, how many remaining untick this is 1 2 3 4 5 and 6. So, there are 6 prime implicants here also.

So, you can less them, this one for example, will be $A\bar{B}\bar{C}A\bar{B}\bar{C}$; this one will be $A\bar{C}D\bar{A}\bar{C}D$, this one will be $A\bar{B}D$, this will be $BC\bar{D}$, this will be $\bar{B}D\bar{C}$ and this will be $\bar{B}\bar{C}$. So, in this way you see you can follow this systematically, you can do it by hand, you can generate all prime implicants of the function.

There is the first step of the tabular method, this method can also be automatic uses systematically, you can do it, you can also write a program to do the same thing ok.

(Refer Slide Time: 29:43)



The Next Step

- What we have obtained?
 - Set of all prime implicants of the function.
 - Some are essential, while some are non-essential.
- What do we need to do now?
 - Select a minimal set of prime implicants that covers all the true minterms of the function.
 - Of course, the essential prime implicants must be selected.
- We shall discuss a tabular method of minimization.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | Switching Circuits & Logic Design

So, the next step; which we shall be discussing in on next lecture so, once you have obtain the prime implicants of the function, when we know earlier that among the prime implicants; some of them may be essential well some may not be essential.

So, now our task is to select the minimal set of prime implicants that covers all the true minterms of the function of quotes we must include the essential prime implicants in addition sum of the non essential prime implicants. So, in the next lecture, we shall be discussing a systematic method for selecting the set of prime implicants to cover the function that will complete this method ok. So, with this, we come to the end of the present lecture, we shall be continuing with this discussion in the next lecture.

Thank you.