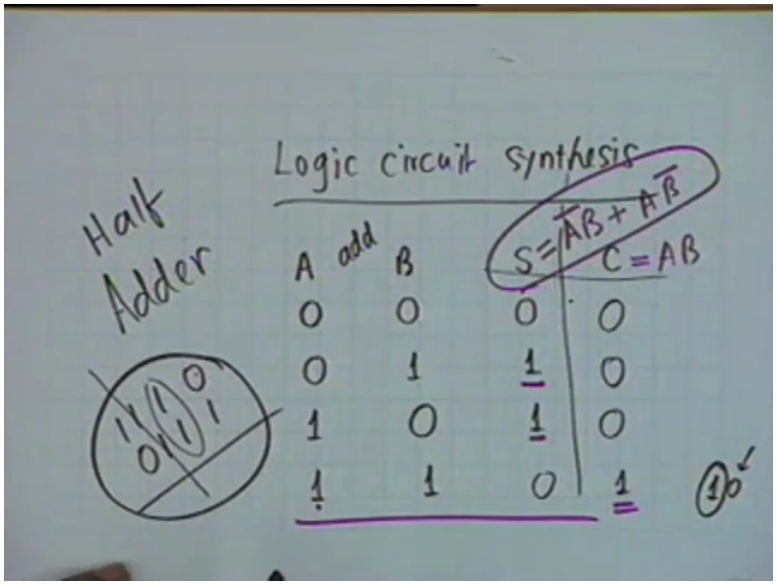
Introduction to Electronic Circuit
Prof. S.C Dutta Roy
Department of Electrical Engineering
Indian Institute of Technology Delhi
Lecture 40
Digital Circuits (Contd.)

(Refer Slide Time: 1:13)



40th lecture, we continue our discussion on digital circuits. In contrast the analysis logic circuit synthesis means devising a circuit to perform a given logic function. For example when you want to add 2 binary numbers, 2 bits let say A and B, well, you know if 0, now it is addition you know if we add 0 and 0 it is not an OR operation, okay. When we add, how do I represent this?

A added to B 0 and 0 the result is 0, A add 0 and 1 the result is 1, when you add 1 and 0 once again the result is one but when you add 1 and 1 10, so this sum is 0 and this 1 is a carrier and therefore in general I would require a sum and a carrier, when we add 2 bits, 2 binary digits then we require a sum and we cannot express the results by 1 bit, we require 2 bits one of them is the sum and the other is the carrier.

The carry is 1 only in the 4th case in all other cases the carry 0. We want to make a digital circuit for this purpose and such a digital circuit is obviously for obvious reasons it is called an Adder. It is also given an adjective of half adder because it does not consider the carry of the previous bit addition, this addition maybe at a certain place in the addition for example 1

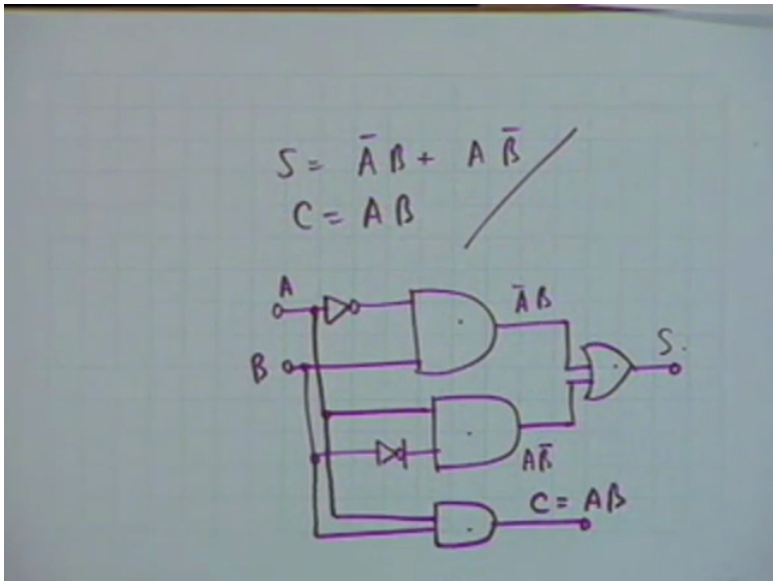
1 1 0, 0 1 1 1 now if I add these 2 I get a sum of 0 and the carry of 1 but the carry of the previous edition is not taken care of, so this is called half adder.

And you can appreciate that at one half adder will be needed in the addition of any 2 binary number. The first digit the rightmost digit addition shall require a half adder because you do not have a carry from the previous edition because there is no previous edition. On the other hand if you want to add any 2 bits in between you shall require a full adder. An adder in which the carry of the previous edition is also accounted for is called a full adder.

First let us look at half adder, in half adder this is the logic there are 2 outputs a sum and a carry and there are 2 inputs A and B, how do you represent S and C that is the question? And we can verify that in such situation you have to consider only the output which are 1 that is if consider only this output and this output and then write S as in terms of A and B, A is 0 here, so A is not true, so you write A bar and the B is 1, so B or either this or AB bar this is the sum, is the point clear?

It can be easily verified that this truth table SA and B is represented by the factor and the simplified method it is true for all, it is valid in all the situations, simplified method is we will look at those rows in which the output is 1 and take the AND operation of the variable, wherever the variable is 1 take the variable as it is, wherever the variable is 0 take the variable in a complemented form, this is always true. For example, for writing C we have only to consider this row and in this row you see A and B both are 1 therefore C is simply equal to AB.

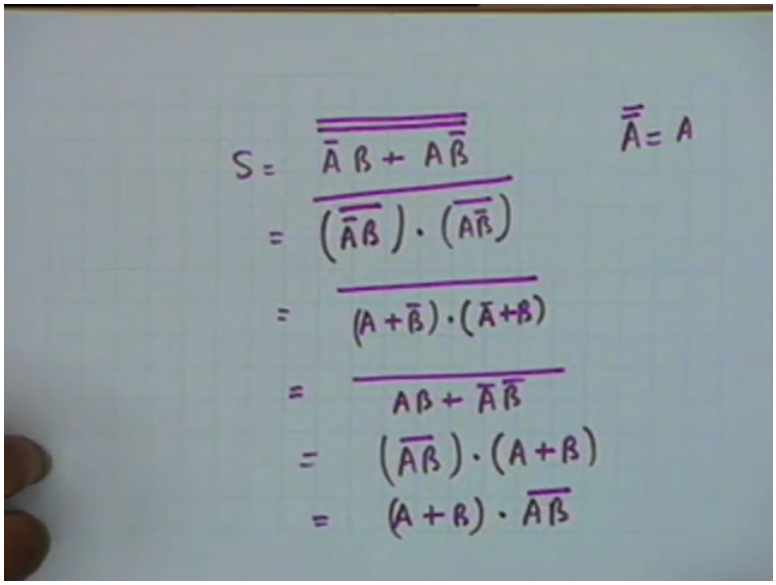
(Refer Slide Time: 6:29)



Now if I represent this half adder logic I divide the half adder logic A bar B plus AB bar I should say or A bar and B or A and B bar or I will simplify this in terms of ordinary language C is simply AB then all that I have to do is, I have to use a NOT gate for A and then I had B, had an AND gate this is A bar B. Similarly I have to have an A and B with a NOT gate another AND gate and these 2 outputs are to be ORED therefore I require another OR gate and this is S.

C as you see is simply AB and therefore what I require is another AND gate and this is C equal to AB, how many gates are (()) (7:43) 1, 2, 3, 4, 5, 6, alright. This is the simple synthesis starting from the simple (()) (7:52) synthesis. Well, in synthesis also one asks the question, can you get a simpler form?

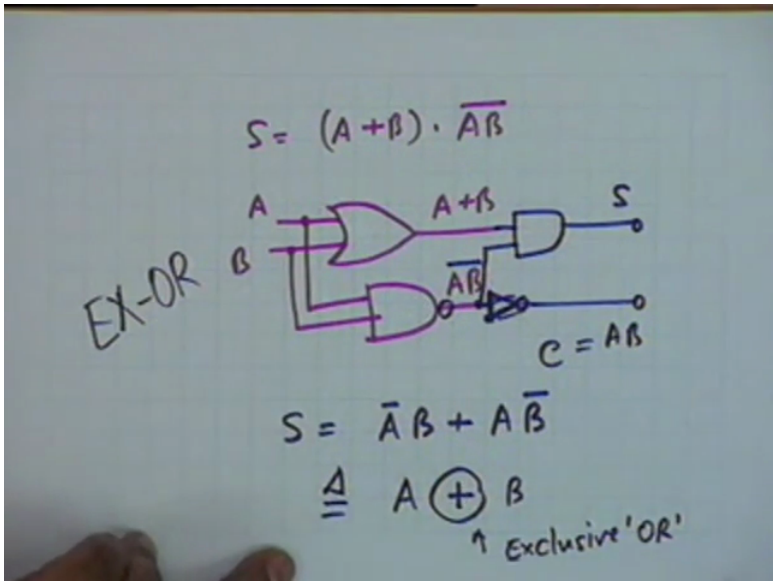
(Refer Slide Time: 8:14)



Well, a simpler form is possible as you shall see now. Our S is A bar B plus AB bar or I can write this with 2 inversions, (()) (8:26)? Every variable with 2 inversions is the same and therefore S I have written has 2 inversions, alright. Now if 1 inversion I will apply them De Morgan's theorem that is A bar B inversion and AB bar inversion and the whole thing inverted again, alright.

A bar B inversion this OR becomes AND and AB bar inversion, is this step clear? Okay, so conversion remain, now expand this I get A OR B bar by the Morgan's theorem AND operation is converted OR with inversion and what we did here? A bar plus B, now if I simplify this AA bar is 0 and BB bar is 0, so what I get is AB OR A bar B bar. I can also write this as, now I take this inversion out AB bar and A OR B, can I write like this? Okay AB bar and A bar B bar whole bar, can I write like this A OR B and AB bar, yes? Okay.

(Refer Slide Time: 10:26)



Now let us see how this expression can be represented by another logic circuit S equals A or B and AB inverted, you see I have A and B, I have an OR gate where this is A or B then I have A and B then a NOT or I can combine into, alright just take a NAND then this is AB bar then...

“Professor -Student conversation starts”

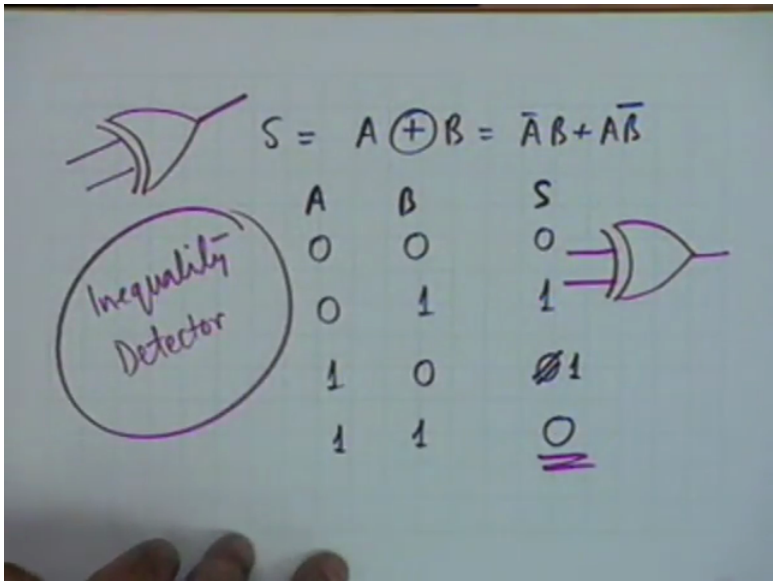
Student: (()) (11:05)

Professor: And therefore, anyway it is okay. We have to use another NAND to be able to get AB which is equal to C, alright.

“Professor-Student conversation ends”

Now I take this and this and then NAND (()) (11:26), how many gates are used? 1, 2, 3, 4 instead of how many we have used earlier?7, alright. So this is less costly than the previous circuit. This expression S equal to A bar B or AB bar which is the logical expression of the sum digit for the sum bit is also considered as an operation and this operation is defined as, definition an exclusive OR operation that is A or if it is exclusive OR then you put a circle around this B and this operation is called exclusive OR operation. Sometimes in the world of abbreviations it is also called EX-OR operation, EX-OR is exclusive OR operation, alright. And there are manufacturers make gate which are exclusive OR gates.

(Refer Slide Time: 12:58)

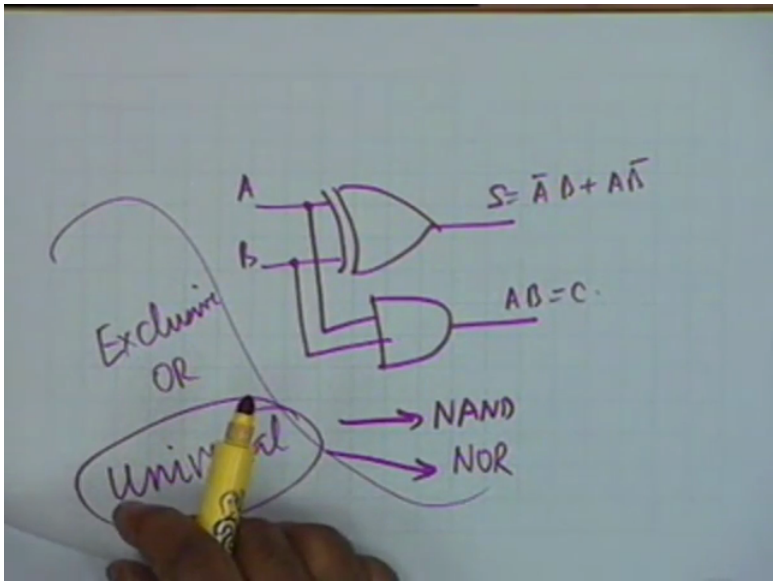


To look at this further since this is an important operation exclusive OR is A exclusive OR B is A bar B plus AB bar, alright. If I write the truth table A B and S, further through table it is same as that of the some bit 0 0 is 0, 0 1 is 1and 1 0 is 1 and 1 1 is 0 this is the only thing where it differs from the OR gate, is not it? This is the only row in which the output differs from the OR gate this is why it is called an exclusive OR.

Now follow this carefully, you can also see that the output of this exclusive OR gate is 1 only when the 2 inputs are dissimilar and therefore an exclusive OR is also called inequality detector that is when the 2 inputs are unequal than the output shall be 1 this is called an inequality detector and exclusive OR has many applications, the simple gate has many applications.

Circuit symbol for an exclusive OR is exactly like that of an OR, except that another curve is included here, another similar curve, okay. What you do is, 2 input, well instead of drawing a simple OR you make this double line and that is the mark, why this symbol why not something else? This is a universally adopted consumption now, originated basically with the inventors of digital circuits exclusive OR.

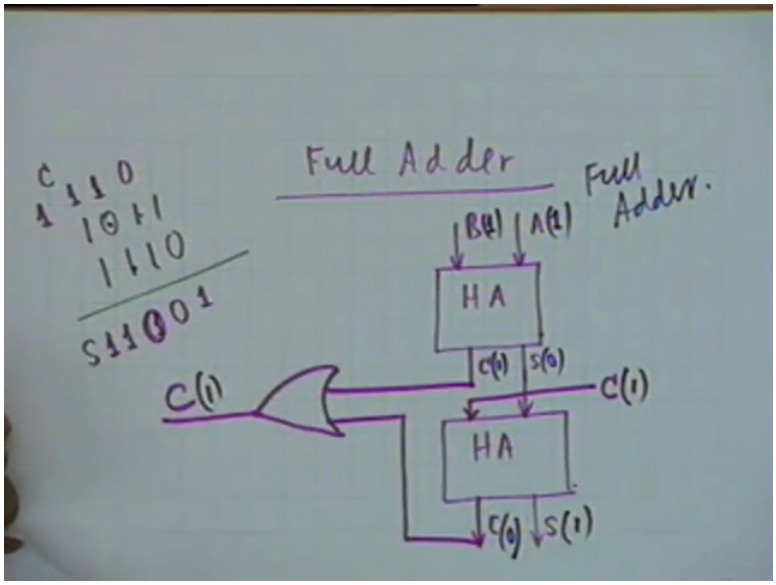
(Refer Slide Time: 15:01)



So in terms of an exclusive OR you see the half adder circuit is simply A b you can do it only 2 gates, one is the exclusive OR which gives you the sum and the other is NAND, AND, okay. This is C, this is A bar B plus AB bar. Question, can a number NAND gates become a peculated exclusive OR, yes. Can a number of NOR gates be connected to make exclusive OR?

If a manufacturer only exclusive OR, can you get all other gates? Why not? If we manufacture only exclusive OR gate, a company decides to manufacturer only exclusive OR and then here are circuit requires AND gates, you require OR gate, you require NOR, that is the labelling. So exclusive OR is therefore not a universal gate, now I am introducing another term universal, universal gates are only these 2 NAND and NOR. Universal gates are those with the help of which all other gates can be simplified, exclusive OR does not qualify in that category.

(Refer Slide Time: 16:46)



Now let us look at full adder, a full adder as I have said required to add 3 things given 2 digits, given 2 bits and the carry of the previous and therefore we shall require 2 half adder because one half adder can add only 2 bits and therefore we shall require 2 half adder, let say half adder 1 and half adder-2. Suppose I want to add let say A and B, 2 bits A and B the result would be sum and carry, it is half adder.

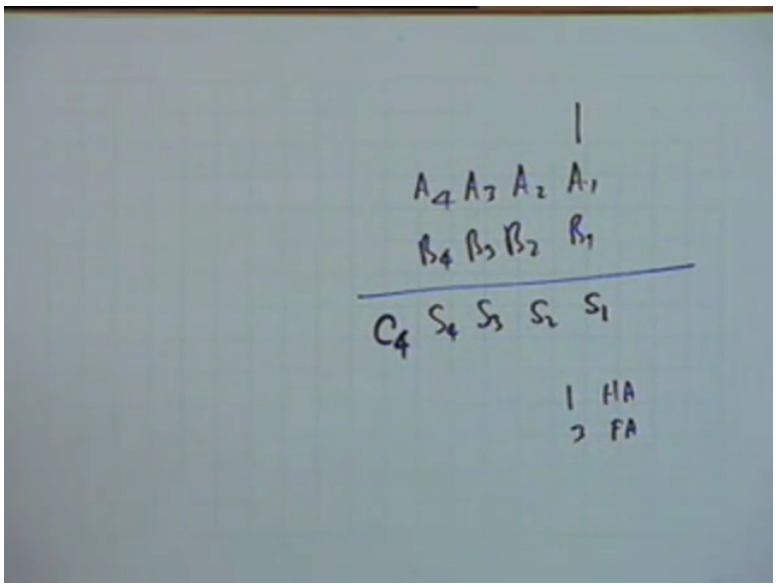
Now the sum comes to the next half adder and add to the carry of the previous addition the result would be, I will tell you what to do with this, where would this carry go? To the next left addition point, alright. So this carry will go to next addition point and then this would be a sum and this would be carry, well, what do I do with this? This would go to an OR operation, this is the carry that goes to the left to the next addition.

Let me illustrate this, the best way to show that it is correct is to illustrate. Let 2 numbers 101 1, 1 1 1 0 we represent the sum here and then carry here, okay. 1 and 0, the sum is 1 then the carry is 0, this carry goes to the next, alright. Next bit, alright. 1 and 1 is 0 and the carry is 1 it goes here, this carry had no effect because it is 0. Now we have 0 and 1 is 1 this carry then goes to the next one.

Or another way of looking at this is 1 plus 0 is 1, 1 plus 1 is 1 0, I beg your pardon this is the 0 and 1 goes here. Now we have been given a situation in which the 2 bits are 1 1 and the carry is also 1, okay. What would be the result here? This is 101 1, so 1 and another carry 1, so this is the one, agreed? Now let me present this here A is 1 B is 1, okay, so the sum is and the carry is 1.

The previous carry was 1, so this sum and carry this is as you know 1 and C is now 0, 0 or 1 this is 1 (()) (20:30). We can show that this is a full adder circuit for any situation and this is the most clear situation where all the 3 are 1, other way you will get trivial but an OR operation is needed here not AND. Suppose this carry was 1 this carry was 1 then this carry would have been, if it was not OR it was an AND this would have been? So we can do it AND also because if this is 1 and 0 this AND this is 0, so we require an OR you can show that this is the circuit.

(Refer Slide Time: 21:27)



Therefore if I have let say 4 bit number to be added to another 4 bit number let say I have A4, A3, A2, A1, B1, B2, B3, B4, alright. How many half adder shall we need? Just one, half adder is required only for this, only for the first column for the others I require, how many? Obviously 3 full adder, so I require one half leather and 3 full adder, alright. The result sum 1, sum 2, sum 3, sum 4 and the carry, carry after 4.

“Professor -Student conversation starts”

Student: (()) (22:19)

Professor: Yes.

Student: (()) (22:22)

Professor: Yes. 10 full adder, oh! That will be very, it would not know when to stop, no, such things are dangerous in digital circuits. In fact sometimes they occurred without even wanting

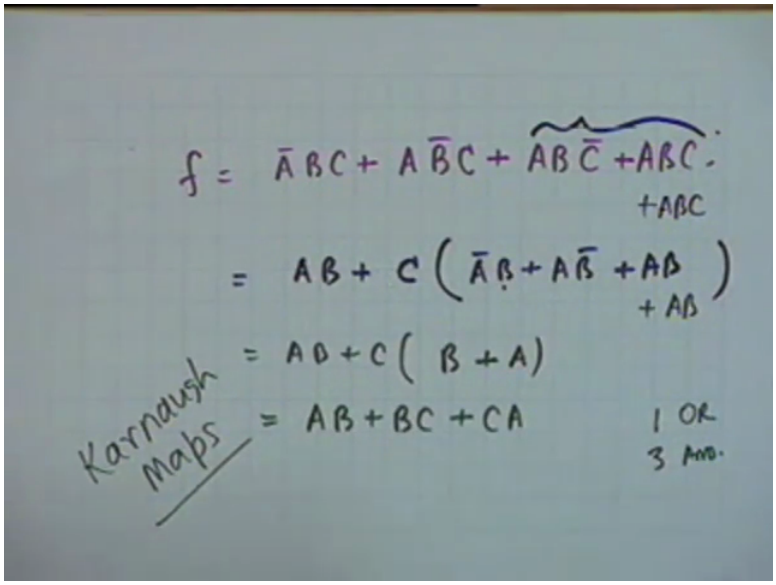
it because there are delays in the circuit and then you have terrible problems, a very serious problem. There are sums like (()) (22:49) and things like that.

“Professor-Student conversation ends”

Okay, so we know what is a full adder? Finally we look at a specific design example and that is how we can conclude this class and this design example is there are 3 people in let say a board of a company, there are 3 people and a motion will be passed when majority of them say yes, majority means 2, alright. Majority is 2 or it can also be 3, this is a very simple situation we want to make a digital circuit to represent this, such a situation is called a board in situation, a both taking situation.

Obviously it is a board taking situation, let this 3 people in the board be A B and C and let the decision be alright. Therefore if AB and C are always be 0 the decision is 0, okay. If 0 0 1 then the decision is 0. If it is 0 1 0 BC you see that I am increasing by 1 it is 0, 0 1 1 it is 1, 1 0 it is 0, 101 it is 1, 1 1 0 it is one and 1 1 1 that is 1, how many? 1, 2, 3, 4, 5, 6, 7, 8, okay I am done.

(Refer Slide Time: 25:21)



Now what I want to do is, I want to write an expression for f, a logical expression and as I have said all you have to do is to ones in which the results are 1 and all of them nobody will read the logical expressions corresponding to this, for example this row represents A bar, B C wherever there is 0 you think you compliment wherever there is one you take the variable and therefore my f shall be equal to A bar BC plus which means an OR, the next is AB bar C, the next is ABC bar and the 4th term is AB.

Now I want to make a digital circuit to realize this, you see if I make a straightforward realisation when you require 3 inverters A bar, B bar and C bar 3 NOT gate 1, 2, 3, 4 AND gate, 4 AND gate and how many OR? What you mean 3? Are you sure you require 3? There are OR gates available with more than 2 inputs, so 1 OR gate should do, will 1 engage here? Because AND is coming before OR operation, alright.

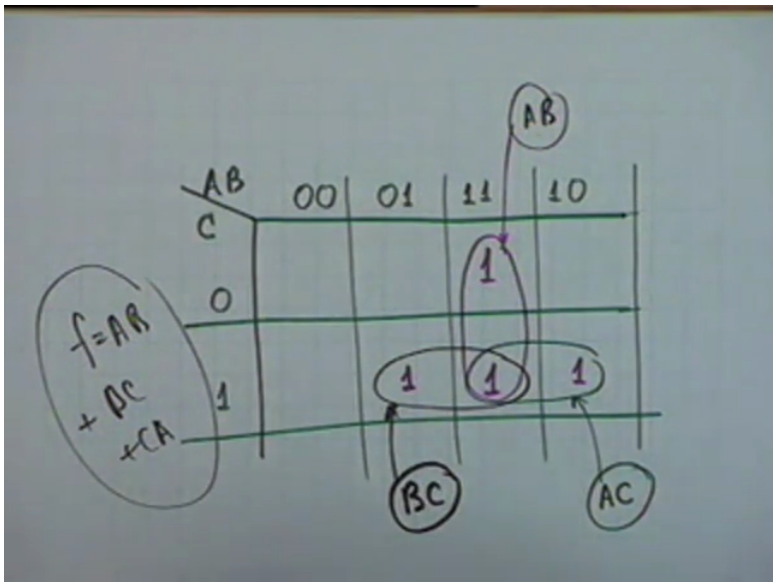
And it is an expensive circuit therefore you ask can it be simplified? Well, what you do is? since there is an ABC, we add another ABC you see how it simplifies, we are not going to the truth table, from the truth table we found out this, now let see if we can simplify this. We add an ABC and then what we do is we combine these 2 then you will get AB C or C bar 1 therefore it is simply AB and from the rest we take C common.

Then we get A bar B plus AB bar plus AB, alright. And to be able to simplify this now we add one more AB. This requires a little bit of experience we had one more AB then it says AB plus C you see A bar B plus AB is simply B and AB bar plus AB is simply A and therefore

this is simply AB plus BC plus CA which means that you require. Now one OR gate no inversion and 3 AND gate, okay.

This is a simplified call for 3 input OR gate, if you do not have 3 input OR gate, what do you do? To input OR gate, okay. Now there is a mechanisation of this simplification and this second (()) (28:30) was done by a gentleman by the name Karnaugh and we do it with the help of table which are called Karnaugh maps and I shall simply illustrate for this particular example, we are not going to Karnaugh maps in general but just to give you a flavour of what a Karnaugh map is we go into this example.

(Refer Slide Time: 29:06)



Karnaugh map we once again consider this truth table and what is done is that possible combinations of AB are written on the horizontal line, what are the possible combinations?

00, 01, 11, 10 very specific briefing (()) (29:33) Karnaugh map must be written with 2 adjacent numbers deferring in only 1 bit not more than 1 bit. You see we did not write 10 after 01 because when that (()) (29:47) change in both the bits there is a discipline that one has to exercise, alright. Consecutive numbers must not have more than 1 way to (()) (30:00) is different.

00, 01 only the second bit is different, 01 and 11 only the first bit is different, 11 and 10 only the second bit is different, okay. These are the possible combination and C can be either 0 or 1 then you draw squares like this, okay. And you put here the combination which make the

output equal to 1, you see one of the combinations is AB01, A0B1 and C1, so this is1, alright. I am talking from the truth table.

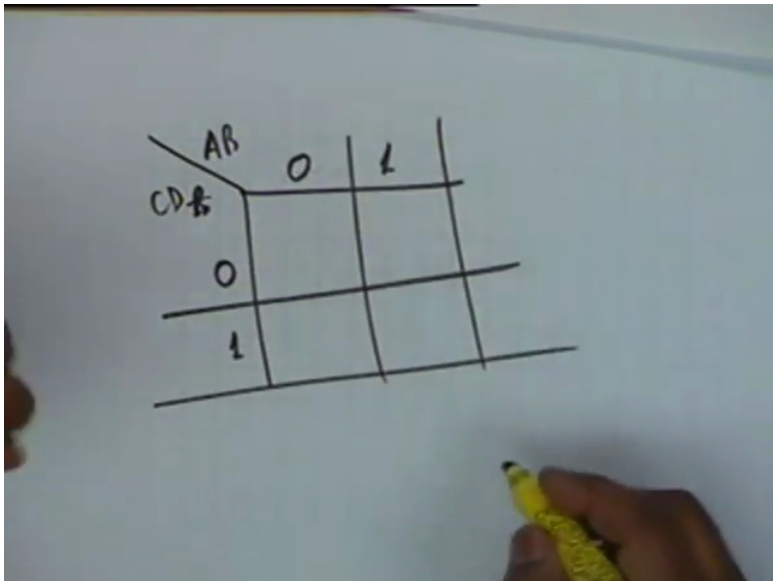
Truth table says 011 gives 1, so AB is 01 and C is 1, this is 1, the next one is 101, so 10 and 1, so this is 1, alright. 10 and 1 the next 1 is 110, so 110 this is 1, okay and the 4th one is 111 that is this 1, all other leave empty, alright. Then what one does is, one take 2 adjacent 1, alright. 2 adjacent ones for example I can take these 2 and here there can be sharing that is 1 here can be shared by this as I left this.

Therefore I can have this combination and finally I can have this combination, this 1 is shared by this 1, this 1 and this 1, it is something like coherent bonding even semiconductor physics, okay. It is permitted but what (()) (32:11) is whatever you did with the help of simplification, it can be done just by looking at it. You see, let's take this circle, this pair, this pink circle pair which is for this C is a bit trivial whether C is 0 or 1 it doesn't matter, alright.

And therefore this pair is represented by simply AB, alright. These 2 1 are represented by AB because it is 11 A is 1 B is 1, alright. If it was 10 then we would have AB bar but since A is 1 B is 1 it is simply AB of (()) (32:55) Pink bar, let me, okay. The green one, yes, for this you see 11 10, so it is independent of B, B has changed from 1 to 0 but both are 1 and therefore this is simply AC, alright.

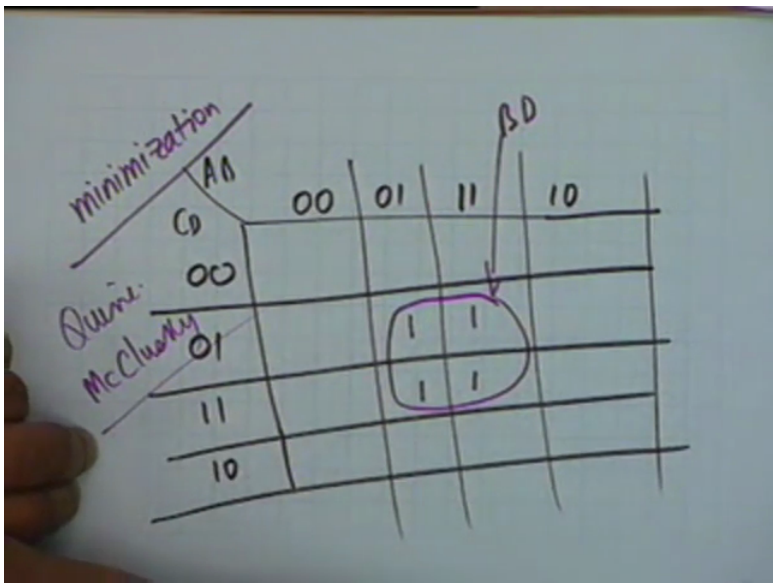
And for the black one finally, it is independent of what, A and therefore this represent BC and that is all that you have to take care of, you have to care and then you get f the function as the OR operation of all the 3 expression that is AB plus BC plus CA and you can show by truth table that is indeed the majority machine or the board taking machine.

(Refer Slide Time: 34:10)



One ask the question suppose there are 2 variables, if there is 2 variables then it is simpler, okay. If it is 2 variables then you simply write A on this line 01 and B and the biotical 0 1, alright, so you shall have 4 circles like this and did you find out which ones are 1try to pair at the (()) (34:25), so there are no pair, so there is only one 1 and of course you know what the function is, the simpler.

(Refer Slide Time: 34:44)



Suppose we have 4 variables that is you have, let say AB and CD, alright. Then I must do it separately then you see the AB possibilities are 00, 01 then 11, 10 in a similar manner 00, 01, 11, 10 and you shall have how many position? 16 position, okay. And from this from the truth

table you find out which ones are 1, which combination? For 01 01, is it 1? If it is 1 then write 1 here.

Now in such a situation which is so happened that there are 4 adjacent ones, for example if we have let say pair can of course we utilized but if there are 4 adjacent ones, well then A enclose within a curve and look at what it is independent of? The expression corresponding to this, you see 01, 11 A has changed therefore it should be independent of A, it should have B, alright.

It is independent of A, here 01, 11 what has changed? C has changed therefore this is simply probability and it is a tremendous simplification. Unfortunately only up to 4 variables if it is more than 4 then this square, this map becomes very complicated and one is not able to minimize. The simplification that we have been doing is technically known as minimization. For more than 4 variables Karnaugh map is not used there are alternative method for minimization.

One of them is for example Quine McCluskey, just remember the name, Quine was the supervisor of a professor named McCluskey who is now in Arizona State University about 80 years old but Quine is dead, Quine was the supervisor but this method is still very popular. There are many other variations of minimization method and that is for, we shall close this class.