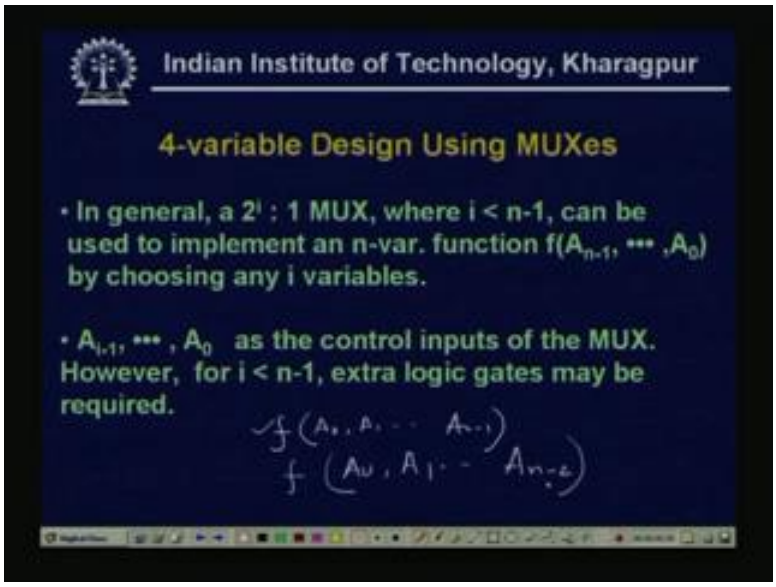
Digital Systems Design
Prof. D. Roychoudhury
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 18
Combinational Logic Problem Design

We are discussing, how we can design the combinational problems or complex combinational problems using fundamental circuits like decoder multiplexer adders etcetera. Last day we have discussed, how we have we can design combinational circuits using multiplexer some the generalization of that design. Today, we will see some large problem design, but before that we can we continue some the four variable design that we are discussing in the last class.

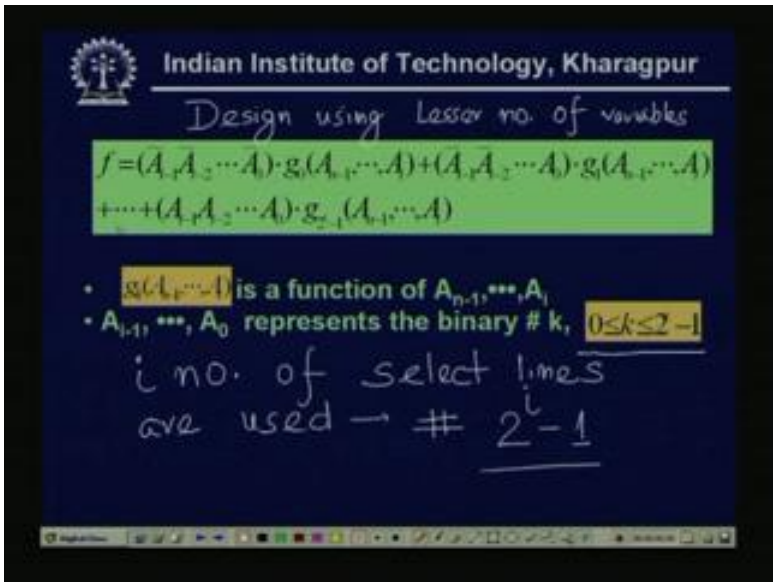
(Refer Slide Time: 01:34)



So, four variable design using MUXes, last day what we have seen that in general a 2 to the power i 2 1 MUX where i less than n minus 1 can be used to implement and n variable function. So, that n variable is 0 to n minus 1 by choosing any i variables, means that if I have a function f n say n number of variables A 0 A 1 up to A n minus 1. Then, first what we have discussed that a multiplexer where that n number of select lines can be used to realize this function. Then, always it is possible to design this function using multiplexer.

Now, if I want to optimize or minimize the circuit design, then even the lesser number of select lines can also realize the same function. That means, f of A 0 A 1 up to say n minus 2 instead of n minus 1. That means, that n minus 1 number of select lines can also be used. Now in this case, that some extra logic gates may be required so, today we want to discuss that. How we can design using lesser number of select lines with the extra gates at the primary input?

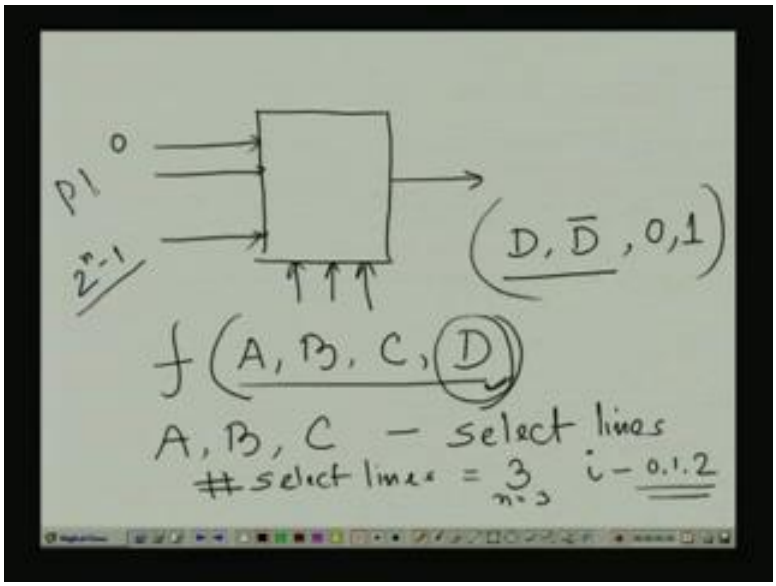
(Refer Slide Time: 03:39)



Now, first we define the problem. So, this is my design using lesser number of variables or these variables means we can tell that select lines. So, first say A 0 to Ai minus 1 means that i my i number of lines select lines are used which determines these I minus number of which determine the some number k. That means which select line I want to choose. So, if it is i number we know that maximum number can be represented by 2 to the power i minus 1. So, that is why my k bound or k is 0 less than k less that 2 to the power i minus 1.

Now, the function that I want to realize that f should be some function of this A i or function of A i. That means, that f equal to some combinations of A's. That means, the select lines it is AND ed with the some g function that is applied to the primary inputs; that means g k is a function of A i to A minus 1.

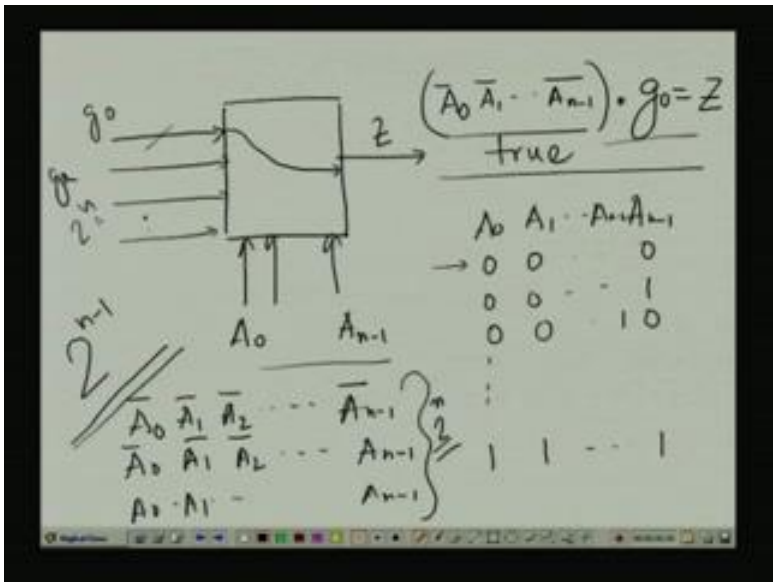
(Refer Slide Time: 06:03)



If I take a small example, what I can tell, say I am taking a four variable, I want to realize say a four variable function A B C D. Now, what I can do among these four variables I will take three variables that A B C as the select lines. So, my number of select lines select lines are 3, so number of select lines are 3. That means i can be 0 1 2, 0 to n minus 1. So, this is here n equal to 3 and this is 0 to n minus 1 means 0 1 2, so these are my select lines.

Now, the D is left, because I want to realize a function f, so this D and as this is a 3 input multiplexer. So, there can be 2 to the power 3; that means, if it is n minus 1. That means, 0 to say this is 2 to the power n minus 1. These many number of primary inputs can be there and these inputs can take the value the D D bar, because D variable is left and 0 and 1. So, any combination of this D D bar 0 1 that should be applied to the primary input and then it will be pass to that output Z and then it is realized. So, this g k is a function of n minus 1 to A I, why this is ended? See this is that from any combination of A 0 to Ai minus 1 and it is ended.

(Refer Slide Time: 08:50)



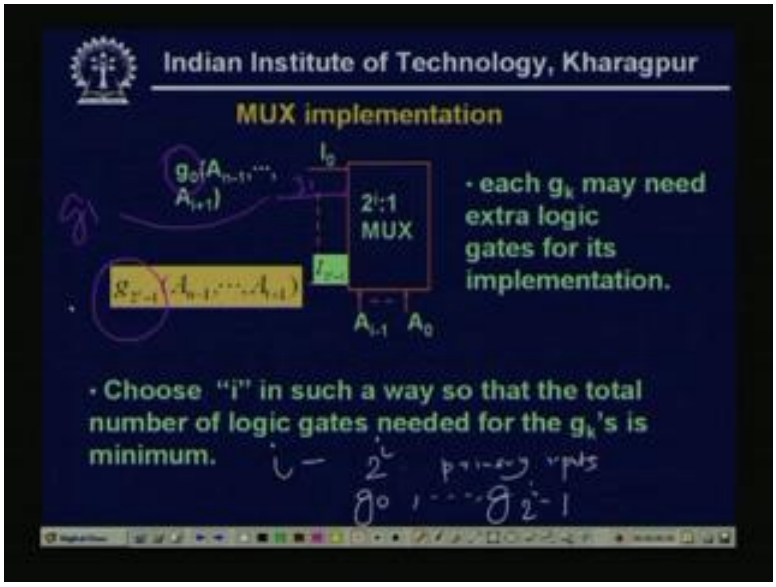
See that, what we have done that, if I take say A 0 now say I am telling A 0 to A n minus 1. Now, it should take all possible combination; that means, A 0 A 1 up to A n minus 1. So, these will take all 0, then 0 0 0 1 0 0 0 1 0 say this A n minus 2, so this type of all combinations it will be taking. Now, what do you mean by this combination, when it is when A 0 to n minus 1, they are taking all zero's. That means these represents the first combination is actually the complemented of all the variables A 0 bar A 1 bar A 2 bar dot dot up to A n minus 1.

Similarly, then all complemented 1 means all complemented except the last bit n minus 1. Or reverse also we can take if we think that A 0 is my LSB, then it will the reverse thing. Similarly, it would it should take A 0 A 1 A 2 up to n minus 1 means this is these are all 2 to the power this is n, so 2 to the power n my 2 to the power n such combinations. If it is 0 to n minus 1, if I take these are n minus 1 variables, then this should be 2 to the power n minus 1 combinations. So, these combinations are applied to the, or these will be treated as the select lines.

Now, see here I am taking a function say some g k values I am taking. So, that for 1 fixed combination of A 0 to n minus 1 that 1 particular input is pass to the output say this is my g 0. Then, for say for the first combination when all the variables are complimented A 0 A 1 bar up to A n minus 1 bar, then for this combination say this is true and then g 0 is also true, this will be my output.

Now, what is g 0, again the g is the combination of these variables that applied or function that applied in the select lines. So, it should be ANDed; that means my output should be treated as one particular line or the function applied to one particular primary input ANDed with the that combination applied to the select lines or the according select lines. So, this is a always this will be a AND functions. And, there are if it is n number of select lines then 2 to the power n such primary inputs or functions apply available at the primary input lines.

(Refer Slide Time: 13:04)

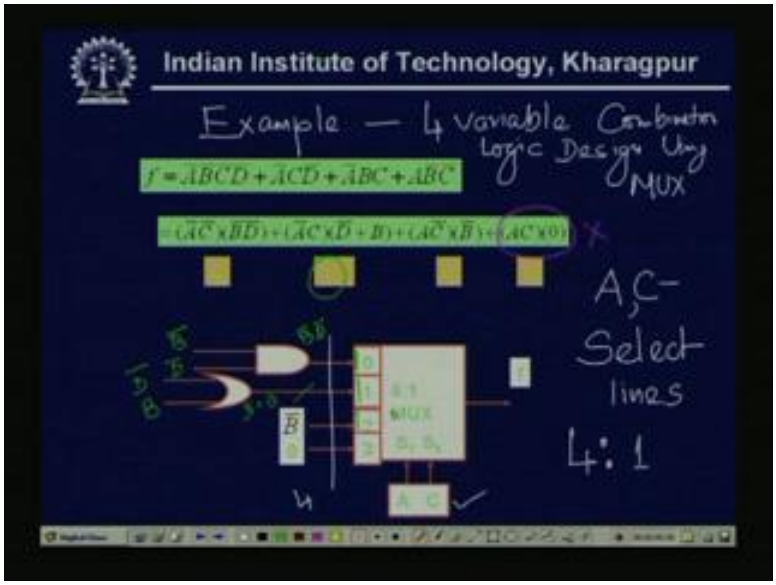


So far, what we have discussed; that means, if i summarize that A 0 to Ai minus 1; that means i number of select lines are chosen. Now, each primary input, say primary input function is represent as g, then a each g or each primary input line can be represented as the function of one particular combination applied in the select lines. That means say A n minus 1 to A i plus 1.

I am representing the primary input functions at each line by g. So, there will be as there are i select lines and this is a MUX. So, there are i for i select line 2 to the power i number of primary inputs. So, 2 to the power i means this will be 0 this will written as g 0 to g 2 to the power I minus 1. This should be i number of primary inputs, so this is these we have written as this is g 0. Say the next line would be i 1 that will be g 1, then this is last line is g 2 to the power i minus 1.

Now, choose i in such a way, so that the total number of logic gates needed for the g k's is minimum. So, today we want to concentrate that, how we can reduce this number of select lines, so that will minimize my design. Last lecture we have discuss the design that if I take that all the variables as the select lines, then always it is possible to implement or to realize the combination circuit using multiplexer.

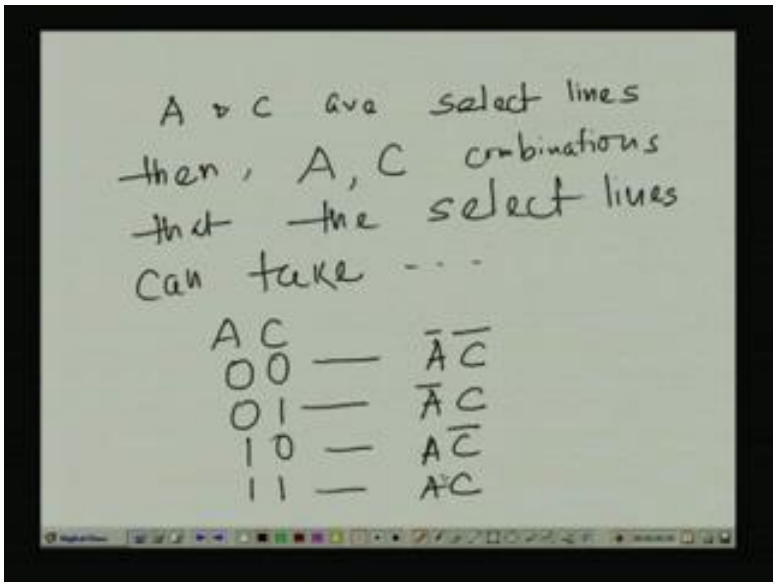
(Refer Slide Time: 15:43)



Now, today we want to discuss that part, now first we take a combinational circuit we take on example. So, this is one example of four variable combinational logic design using multiplexer, using MUX. Now, that we take the function as A bar B bar C bar D bar A bar CD bar plus A bar BC plus AB bar C bar. Now, I can rewrite the function as see this is the four variable functions, first I am taking AC the two variables as the A and C are the select lines. So, these are my A and C are my select lines.

So, if AC is the select line, then this should be a 4 to 1 MUX, because if there are 2 select lines; that means, 2 square or 4 number of primary inputs will be there. So, this is the four numbers of primary inputs and 1 primary 1 output. So, the variable left are B and D, I have taken AC as the select lines.

(Refer Slide Time: 17:51)



AC should take that say, if it is AC are the select lines. So, if A and C are select lines. Then the combinations AC can take, AC combinations that the select lines can take or we can think this as the control inputs are 0 0 0 1 1 0 1 1. See 0 0 means A complement C complement this is A complement C 1 0 means A C complement 1 1 means AC.

So, now I want to represent this function with respect to this A bar C bar A bar C AC bar AC. The first min term A bar B bar C bar D bar, so if I take A bar C bar common then B bar D bar is left. So, the first min term is written as it A bar C bar ANDed with B bar D bar, plus here the second term is A bar C A bar C and D bar. See another the fourth min term is also SA or third min term is A bar C, so this is D bar plus B if I take common then A bar C ANDed with D bar plus B

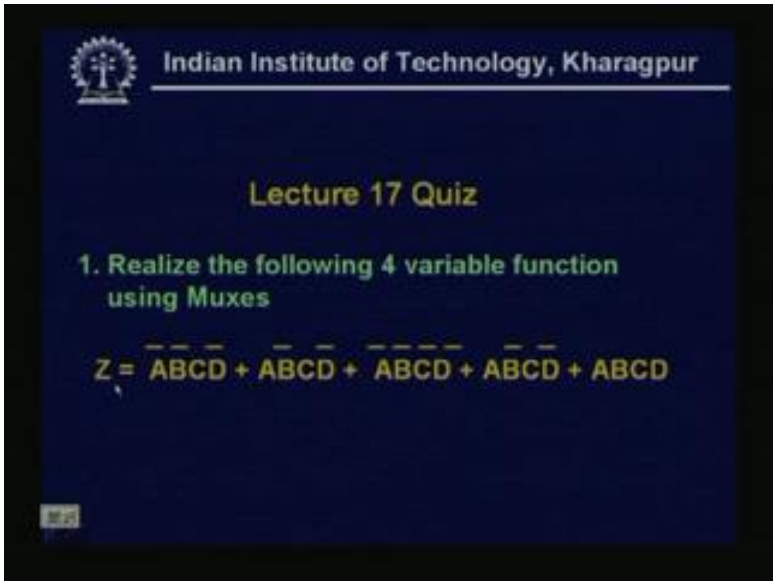
This AC bar only B bar is there and see there is no term consisting of AC. So, that is why I have kept AC ANDed with 0. That means this term does not exist in the this term does not exist in the function. So, now if I consider a multiplexer design whose AC value can take these variables. That means my i 0 input can this is my i 0.

So, this i 0 can take, see i 0 means A bar C bar, so this should be B bar D bar. See B bar ANDed with D bar; that means my the first primary input is B bar D bar. I am assuming that the all complemented variables are available. So, this output of AND is the B bar D bar. Now, if I consider the second primary inputs means i 1, now second input means the select line should be 0 1 mean A bar C. So, the select line should A bar C and then i 1

should be pass to f, then what is my i 1, from the function we see i 1 is D bar plus B. So, this is my A bar C and this is my D bar plus B, so this is my i 1 term. So, second input, so again this is D bar and this B and this should be OR. So, this OR output is D bar plus B. Now, we see that i 2 length, i 2 primary input is the AC bar, now see AC bar means that it is my B bar only. It should be AND ed with B bar. So, this it is AND ed with B bar.

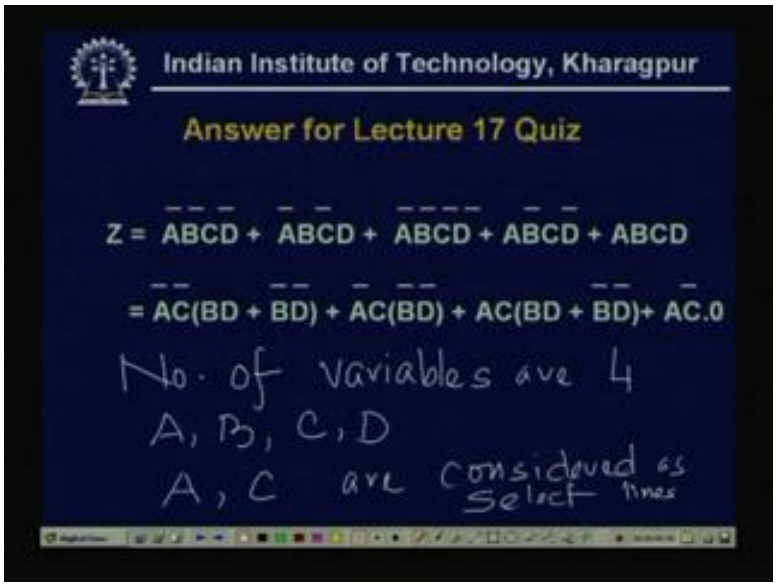
Now, i 3 is that 0, because this term was not there in the function, this is i 3, i 3 is 0. So, this is 1 multiplexer realization of this function where we have taken two variables as the select line instead of four variables. But some extra logics we have applied in the primary input. So, this is one optimal design of this particular function.

(Refer Slide Time: 23:38)



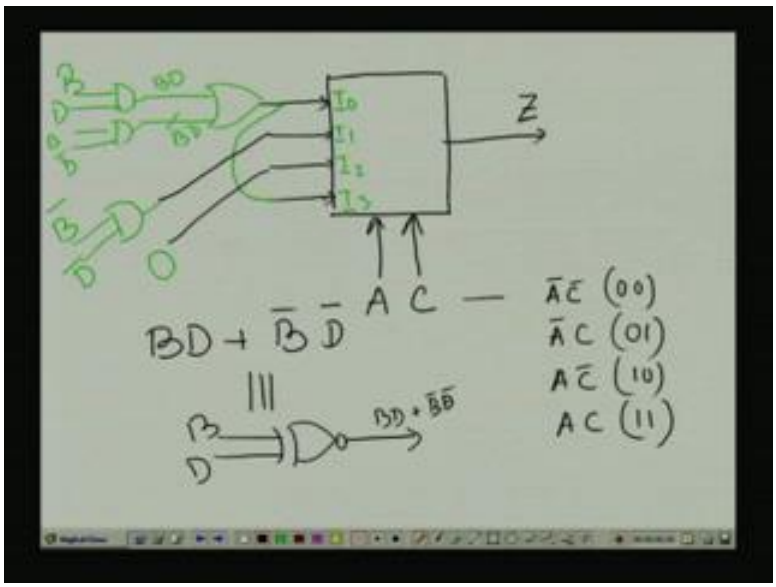
Now, we see that we discuss the last days quiz question, so quiz question was that realize the following four variable function using multiplexers. The function was Z equal to A bar B bar CD bar AB bar CD bar A bar B bar C bar D bar AB bar CD bar plus ABCD. So, just now what problem we have discussed, it is a similar type of function.

(Refer Slide Time: 24:05)



See, what will be the answer, so again what will be doing first we will rewrite the function in terms of the select lines. So, first we have to select that, what will be the, what variable we will be taking as the select lines, see here the number of variables are the number of variables are 4. They are A B C D. Say among these four variables say AC just like the previous example, AC are considered as select lines.

(Refer Slide Time: 25:25)



So, what I want, I want a multiplexer design whose select lines are AC, as there are two select lines. So, obviously, there will be four primary inputs they are I 0 I 1 I 2 I 3, there

will be one primary output Z. Now, we see that AC can take again my select lines can be select lines can be A bar C bar means 0 0, A bar C means 0 1, AC bar 1 0, AC 1 1. So, my next step is that I want to make or I want to design that function as the end function of this combination of select lines and that particular primary input. So, that it realizes each minterm.

So, we see the min terms first, so min term is AB bar C bar D bar among these four variables AC are the select lines. So, we AC here the these min term, AC exists in the form of A bar C. So, this is actually it should come to the I 1. So, I am giving, this is my I 1 term where it is A bar C into B bar D bar that means A bar C ANDed with B bar D bar, this gives my I 1 .

Next term is next term consists of A bar C bar and the third min term; that means, the second min term and the third min term both consists of A bar C bar. So, my first primary input I 0 should be A bar C bar BD ANDed with BD plus B bar D bar, so this is my I 0.

Similarly, the AC terms there are 2 min terms the fourth and fifth min term consist AC. So, from here and here if I take AC then again it should be also BD plus B bar D bar this is my actually I 4 term I 3, I 3 term. This is my I 3 and this is AC bar, there is no AC bar existing in the function; that means there is no min term where AC bar is there. So, this is my I 2.

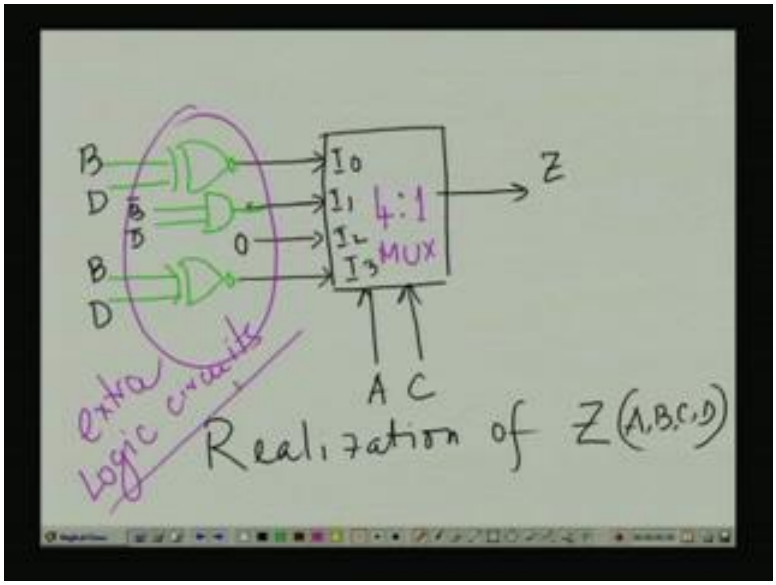
So, now the first I 0 term consist BD plus B bar D bar, if I first it should be a or this should be an output of OR whose inputs are BD and B bar D bar. Again what I can do, this can be 2 1 AND function whose two input are BD and B complement D, D complement.

Now, this BD plus B bar D bar again that is the function of the I 3 term, so directly what I can tell that, the same output I can give to I 3 this is my I 3. And, the I 1 and I 2 these are different, so I 1 is A bar C into B bar D bar, I 1 is B simple B bar D bar. That means 1 AND function two input this is B bar this is D bar. And I 3 is I 2 component is 0, this I 2 is 0, so it will realize the same function, what is given.

Now, there can be another realization see these actually BD plus B bar D bar this is nothing but a realization of the X NOR, this is a XNOR 2 input XNOR whose inputs are

B and D. Then, this is actually BD plus B bar D bar, so what we can do that input of I 0 and I 3 can be replaced by a two input XNOR.

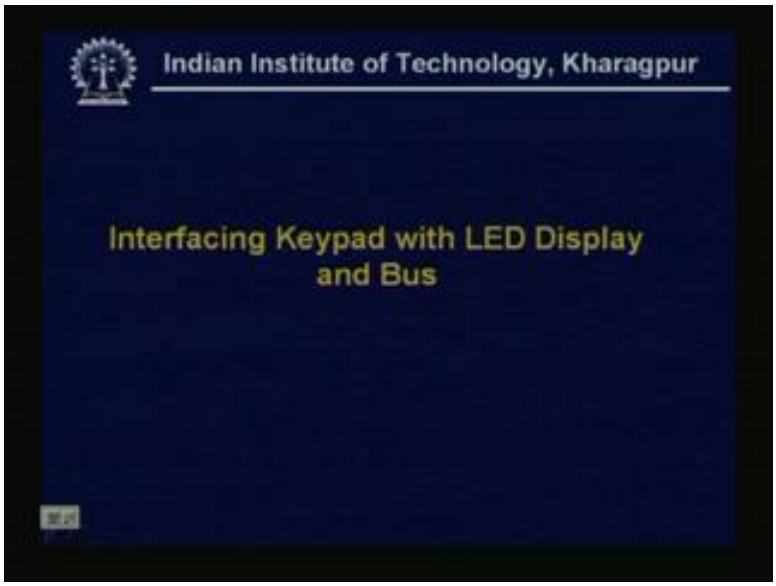
(Refer Slide Time: 00)



So, if I again draw that this will be my two select lines AC, that I 0 and I 3 both are coming from some XOR XNOR output. The outputs are both the cases outputs are BD, so both the cases that two input it is BD and this is a BD plus BD bar and I 2 was 0 and I 1 was B bar D bar. So, it will as it is I 1 and I 2 is 0. So, I 2 is simple made 0, I 1 is again 1, two input AND whose inputs are inputs are B bar D bar, these inputs are. So, again this can be another realization where this is my is a 4 to 1 MUX. And actually, this realizes the function given, realization of the function Z ABCD. So, in this way we can get a optimum MUX realization optimum design using multiplexers.

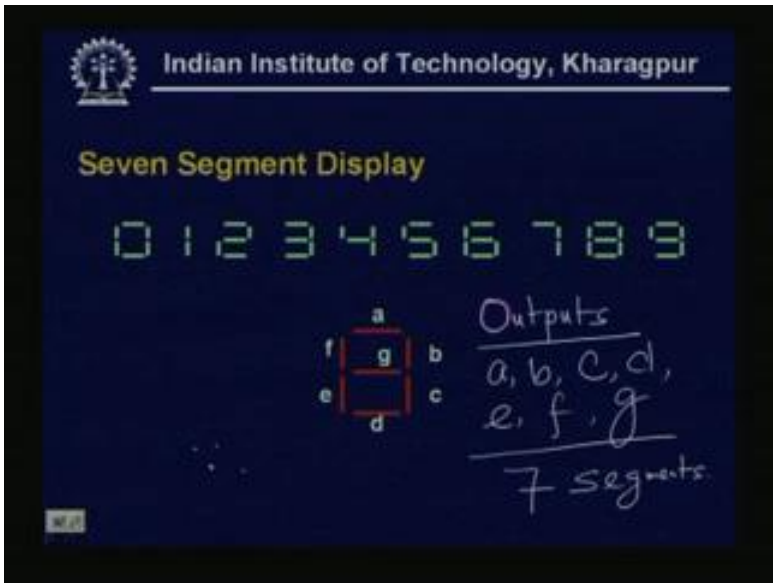
Now, another thing is that even we have to select a appropriate number of or minimum number of select and based on that the inputs lines to be designed. So, some extra logic circuits, so these are my extra logic circuits are needed or we can tell the extra gates that are needed to realize the function.

(Refer Slide Time: 35:39)



So, now we shift to a different type of design problem again that is a combinational problems combination logic design. This is an interfacing keypad with LED display and bus.

(Refer Slide Time: 36:05)



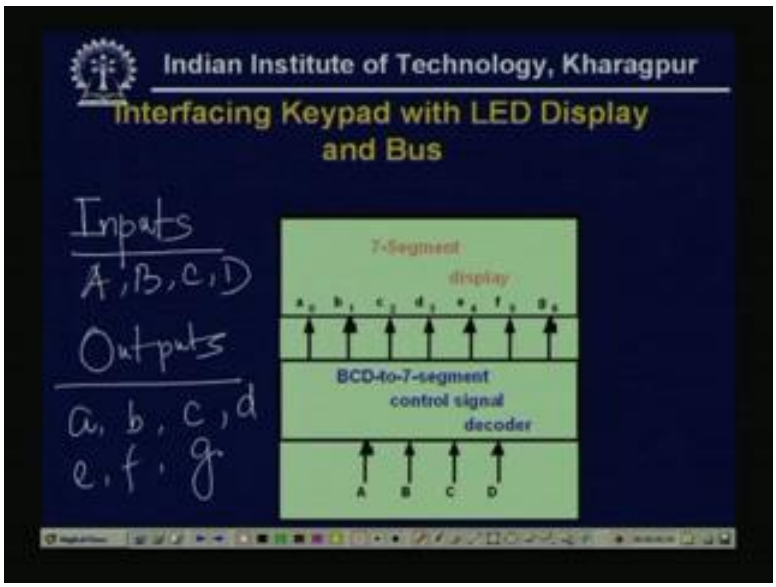
So, first we see that, what is the problem, we want to design an interface this is the interface design problem. And see there are, and I want this is a LED that seven segment display. Here, there are seven components say a, b, c, d, e, f, g and we want that every segment to be every segments of the output of one design. That means the outputs are…

if I take the outputs, these are a b c d. These are the seven segments and parallely I want that, these should be taking these should go to the bus also.

Now, these are the… if I want all the digits, we know that here see that if I 1 2 draw that 0 or 0 to display. Then I have to that seven six segments among the seven segments that except g all the six segments should be accepted. Similarly, that for one that either f e or b c that two segment should be selected.

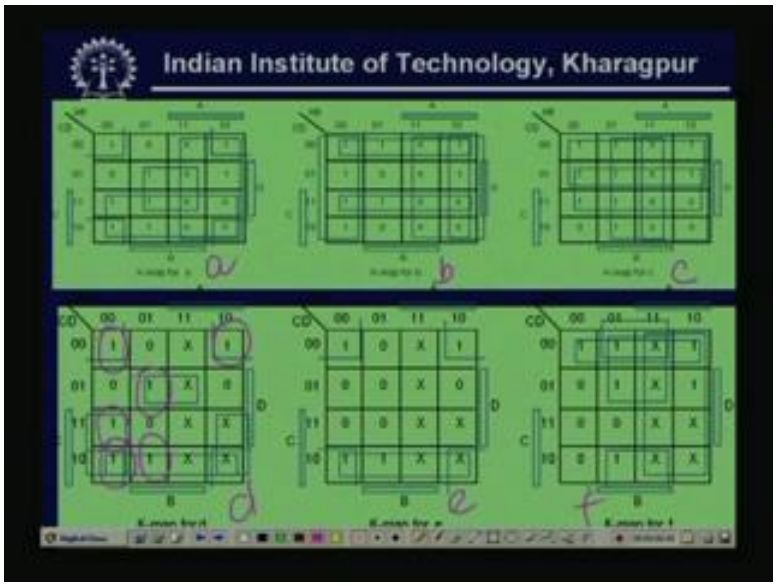
Similarly, all these segments should be o this is my problem design, that I want some inputs and which will give the select the control lines. So, that it can again it can select the particular segments for a digit, so this is the problem.

(Refer Slide Time: 39:16)



So, overall thing this is a BCD to seven segment control signal decoder. Say here, I want that this ABCD are my inputs. So this is a problem or this is combinational circuit design where the inputs are A B C D and outputs are that the seven segments a b c d, a b c d e f g the seven segments. So, this is a BCD to seven segment control signal decoder, we can represent this problem like that

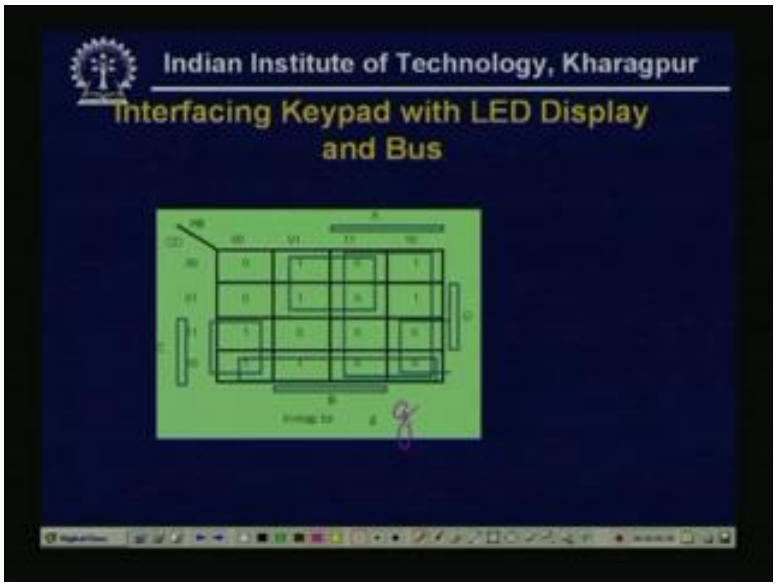
(Refer Slide Time: 40:12)



Now, first thing is there are seven output lines, even if we ignore the other control signals that for each segment will be treated as the output lines. So, there are the problem becomes that as if there are four input lines A B C D and that seven output lines that all output lines are as if that all segments a b c d e f g. So, if we consider these are the Karnaugh map that for that each segment, so if we consider that Karnaugh map for d any one, we can take this is a Karnaugh map for d.

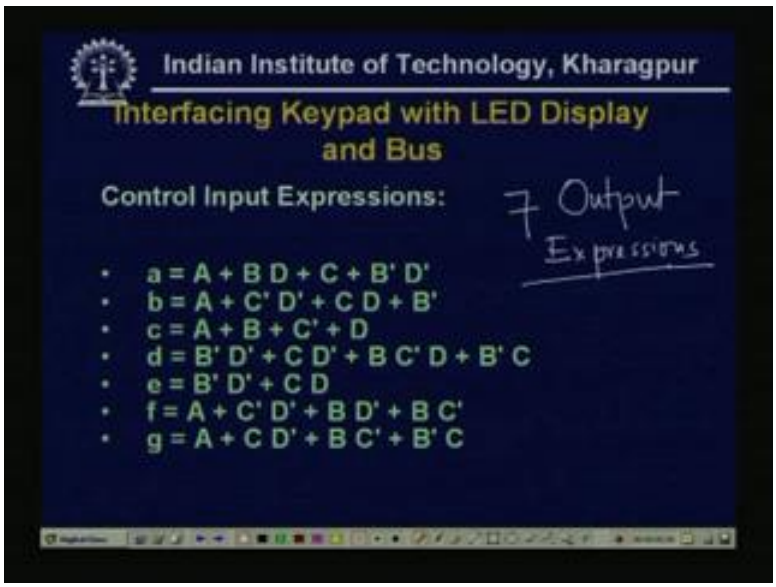
I have four variables, this is A B C D, so see that A bar B bar C bar D bar this is this should be true or this minterm will be there. Similarly, that here A bar B C bar D that is also there, so this 1 this is 1 this 1 there will be. So, now from that Karnaugh map concept, what we can do, we can reduce the, or we can minimize the function. First, we will select that couple as already we discussed. And similarly, for each segment that this is for my a this is for my Karnaugh map for b c d e f and that g.

(Refer Slide Time: 42:20)



This is for g, we can first we will draw the Karnaugh map for each segment. As if this segment is the output for that design.

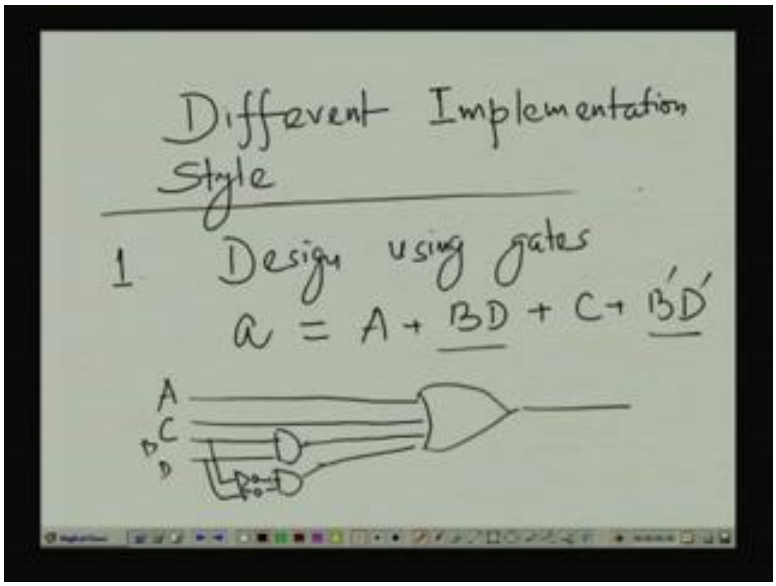
(Refer Slide Time: 42:40)



Then, the in control input expression because these are nothing but the control segment, that which segment should be selecting to give a particular digit. So for, a is that if I minimize the function using Karnaugh map, then a is A plus B D plus C plus B complement D complement. Similarly, b is A plus C complement D complement plus CD plus B complement, C is similarly A plus B plus C complement plus D.

So mainly, what we got, we have say seven we can tell that output expressions. Now, how we design this thing these are my expressions, now how will design now, there can be many designs because these are nothing but some seven combinational logics seven. Now, as this is a large set of expressions, what we can do.

(Refer Slide Time: 44:11)

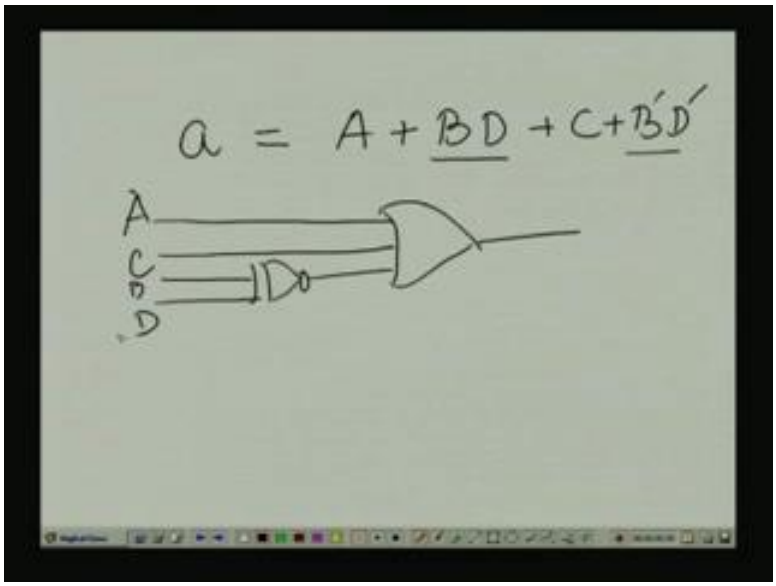


There can be different type of realization or what we can tell that different implementation style. One first, what we can do, the simplest is the design implementation using gates. (Refer Slide Time: 42:40) See if we again notice that for a, this is nothing but AND OR expression. So, this is A plus BD plus C plus B complement D complement.

So, I can write a equal to A plus BD plus C plus C plus B complement D complement, how I can do that thing, again there are four inputs. So, simple this thing I can do that this is a four input OR A, then C then BD and B complement D complement which is nothing but I can if it is a four input XOR, then this is one AND whose two inputs are B and D

The fourth one again it is a AND whose this is a B compliment and this is a D complement. So, the is one simple realization I can do, what I can do this another realization can be, another realization can be again.

(Refer Slide Time: 46:31)



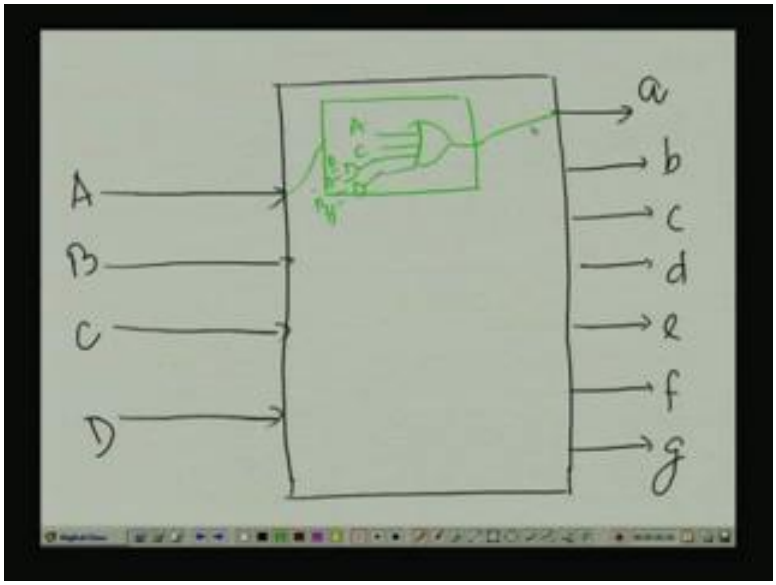
If I consider a equal to A plus BD plus C plus B complement D complement, see if it is not a AND OR gate any using any other logic gates, if I want to minimize the gates. And say, I have all are different type of gates are available I have that option that AND OR XOR XNOR whatever that all possible gates that are available, it is not a AND OR realization.

Then, what I can do, again I can take that this is a four input OR and say simple A and D are A and C are feed as it is. Then C BD and B complement D complement is nothing but a XNOR, this is a XNOR output whose inputs are B and D. So, this can be another realization of a.

So similarly, if we see that other expressions, (Refer Slide Time: 42:40) See this is A plus C complement D complement CD plus B complement. See that these are very similar that CD plus C complement D complement means this is CD plus C complement D complement means again this is nothing but a XNOR realization whose inputs are C.

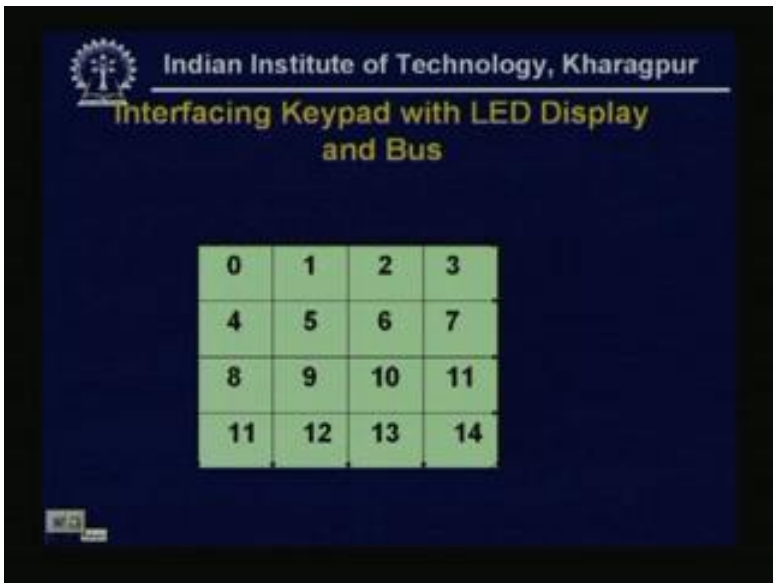
And that A and B complement as it is so this is similar type of structure. So, in this way I can design, this eight expression for or that that seven segments seven expressions using that simple combinational logic gate or simple logic gates. These are nothing but the combinational designs.

(Refer Slide Time: 48:52)



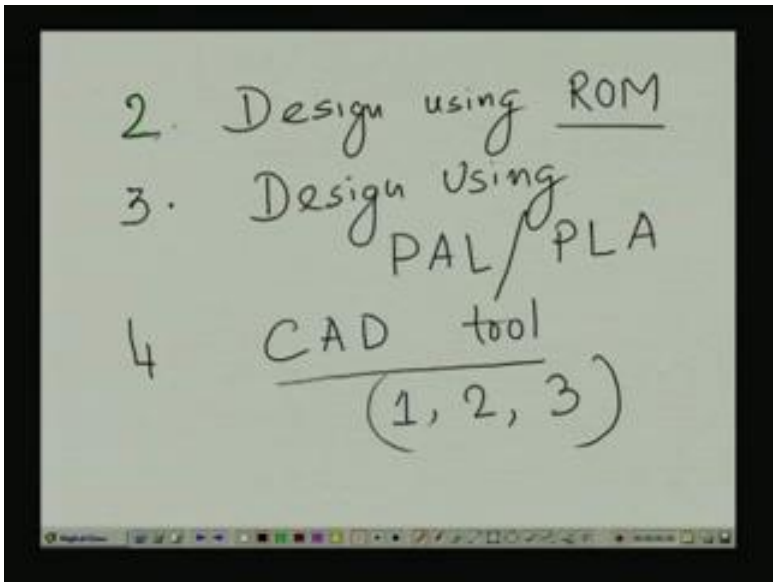
Now, only thing is that that this should be a A B C D, if I take these as this design as the black box. Then, there would be four input that A B C D the segments are a b c d e f g and just now what we have discussed that. Say this is my logic design of a say 1, this is 1 four input OR gate where this A is they are C is there and the XNOR if we consider or BD and BD bar. See if it simple, this is this is B and BD bar, this is B and B bar D bar. So, these similarly for all segments there can be the different type of circuitry and we can put in this way.

(Refer Slide Time: 50:59)



Now say, I can tell that there can be many others design structure, so this is one type of thing this is design using gates.

(Refer Slide Time: 51:20)

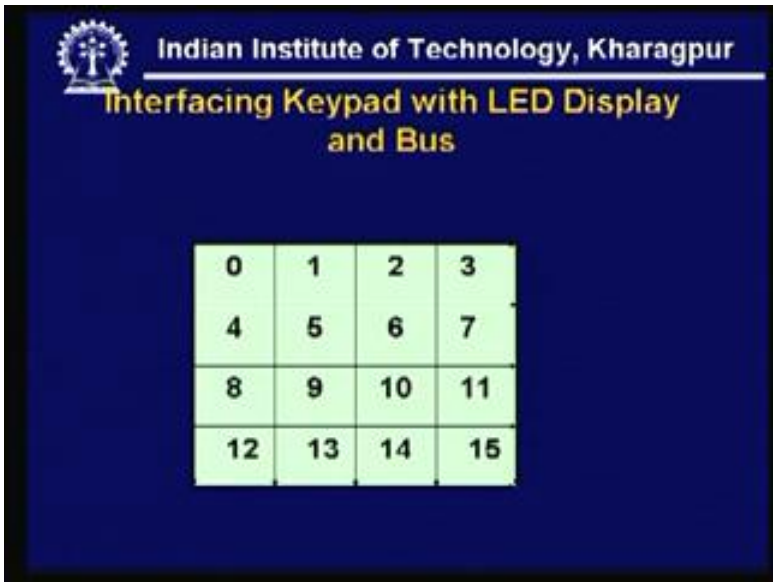


Now, the second can style can be that this can be a design using the ROM, so far we have not read the memory design using the read only memory we can also design this thing. This can be design using some programmable array logic it is called PAL or PLA programmable logic array or some CAD tool if it is a big design even some CAD tool can also synthesize or realize this function in different way.

But, normally the CAD tool we will be using any one of the 3; that mean CAD tool will use either 1 2 or 3 these options, so these are the different styles. So far, we have read only the design style or the implementation style of one; that means, the combination logic gates.

Now, another thing is that combinational logic design that just now we discussed using simple AND OR logic OR the XNOR, that also if it is a big design, that also we can do using multiplexers. Using decoder or if it is a different type actually depends on application it can be using adders half adders or full adders. So, when we will be reading that the memory read only memory and the programmable logic arrays. Then will discuss that how the combination logic can be designed or there will complex combination logic can be design using these ROM or PAL and PLA.

(Refer Slide Time: 53:31)



Now, our problem was our design problem was the interfacing keypad with LED display and bus. So, see if this is my keypad say I have 15 such keyboard I want that when I press 1 key day that number 5 key then that binary should be that should be or in that display board that five will be activated. And for that, I have to for that is that particular segments of the seven segment display must be selected.

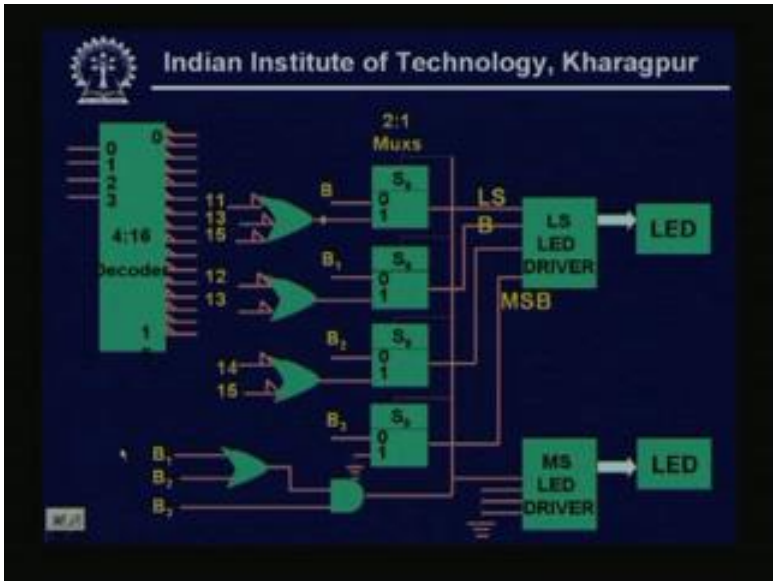
(Refer Slide Time: 54:18)

So, you see that what will be the, if I take that this RC means these are rows and columns. So, these are the there are 4 row and 4 columns, this should be here this should

for 5 it should be the second row and second column 2 to similarly for 10 it should be third row third column like that. So, these are RC gives the all the elements or the all the keys. The position of all the keys, I should be 1 1 1 2 1 3 1 4 means this is my first row, similarly if it is this can be second row third row like that.

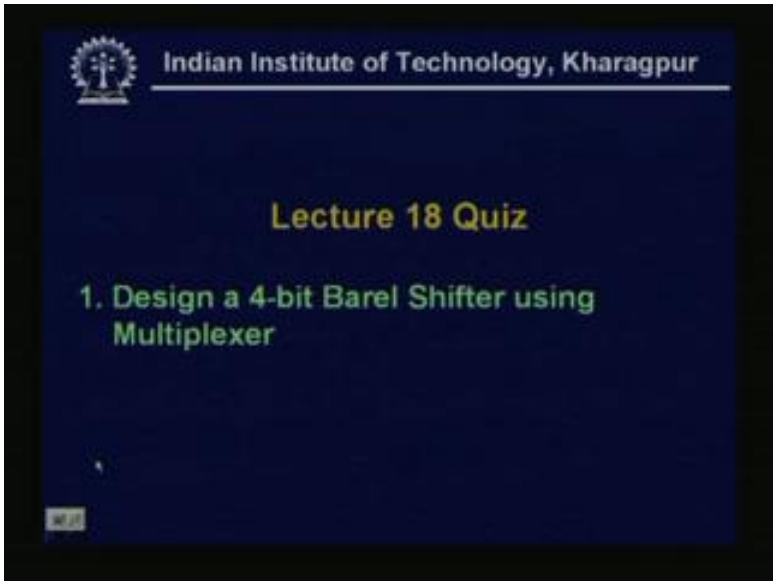
Now, the bus lines this is my bus lines; that means when these 1 1 should be pressed or 1 2 that, then these bus lines should be 0 0 0 1. This is some 4 bit lines we have given, because I have 15 such keys, so I need 4 bit to be. Similarly, this is from my most significant driver lines and this is a least significant drive lines. Now, if I similarly if I design the Karnaugh maps for every line then I will be getting the circuit in similar manner.

(Refer Slide Time: 55:50)



Now, using decoder and multiplexer I can design the circuits, this is my overall design say I have used a 4 to 16 decoder. So, these are my a b c d lines these are my four inputs and I have used 4 2 to 1 multiplexer which are actually driving the least significant LED driver and the most significant LED driver is a 4 bits and accordingly the LED are displayed. That means that particular segments of the LED are selected and they are displaced. So, this is the overall circuitry

(Refer Slide Time: 56:32)



Now, that lecture 18 quiz, design a 4 bit Barel shifter using multiplexer, this is an extra thing.

Thank you.

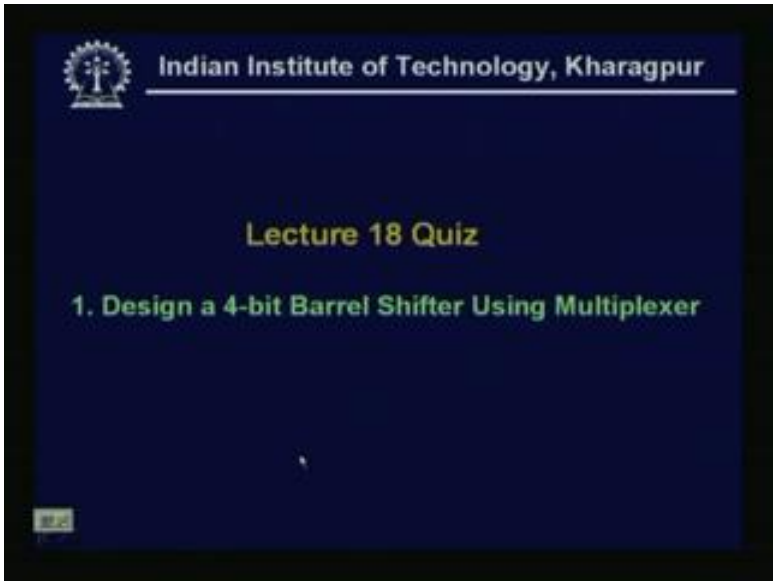
In the last few lectures we learned, how the combinational circuits can be designed using functional blocks or the fundamental circuits like decoder multiplexer adders etcetera.

(Refer Slide Time: 57:05)

Last time we have seen the some generalized structure of realizing combinational circuits using multiplexer. Now, today we will read another type of combination logic design

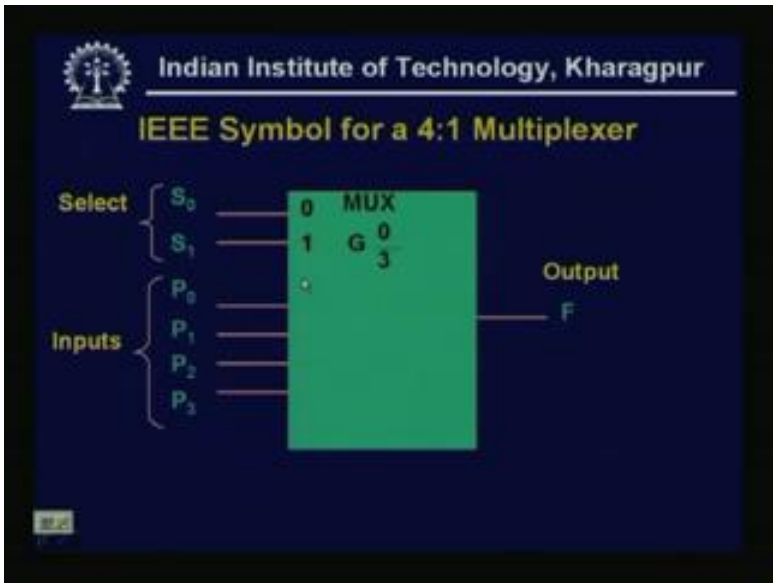
using programmable arrays of logic gates, before that before we start that programmable arrays quickly we solve the last days quiz question.

(Refer Slide Time: 57:50)



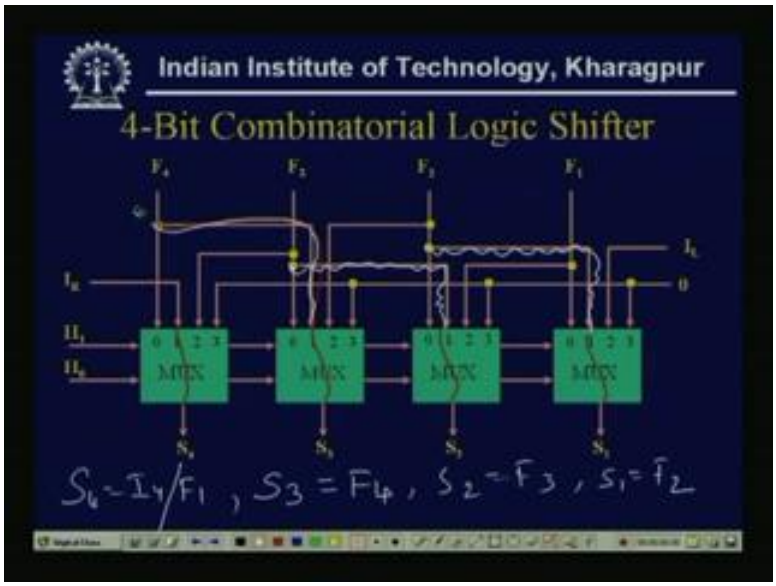
The question was that a design it is a 4 bit Barrel shifter using multiplexer.

(Refer Slide Time: 58:01)



So, answer is, see that already we have discussed the 4 2 1 multiplexer design it has 2 select lines. And, 4 input lines 1 output lines and this is the algebraic symbol for 4 to 1 multiplexer, we will use this multiplexer to design a shifter 4 bit shifter.

(Refer Slide Time: 58:27)



Now, first we see the normal combinational logic shifter see this is 4 4 to 1 multiplexer we have used. This is my MUX 1 MUX 2 MUX 3 MUX 4 or reversely we can take that outputs are S 1 is to S 3 S 4, and the functions that F 1 F 2 F 3 F 4 that we want to implement or we want pass that the as the output.

Now, there are the shifter.