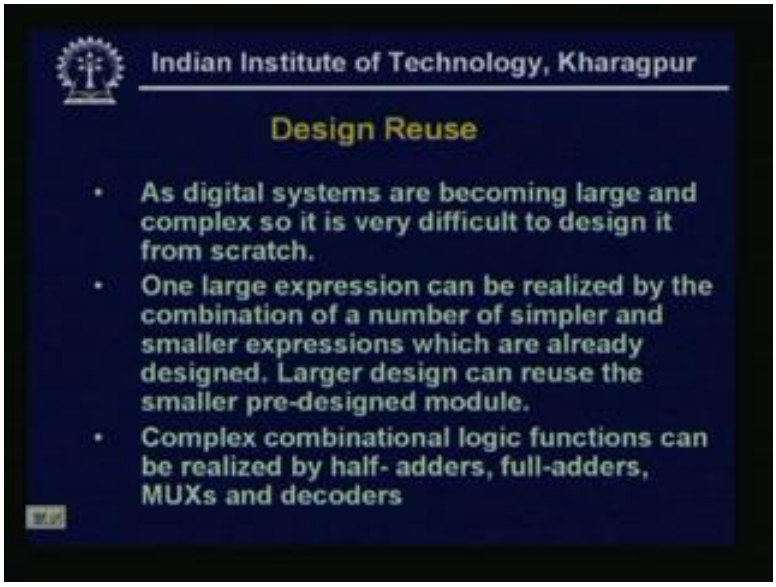
Digital Systems Design
Prof. D. Roychoudhury
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 16
Design of Complex Combinational Circuits
Using Fundamental Circuits

In that last lecture we have learnt how to design the fundamental circuits, means the common digital elements like, decoder, the multiplexer, one half adder, the full adder. Now, today we will see how the large complex combinational circuits can be designed using this fundamental circuits, means this half adder, full adder, decoder, multiplexer or even it can be some smaller modules.

(Refer Slide Time: 01:27)



Now, the basic concept behind this large combinational circuit design is the design reuse, now what do you mean by design reuse. As digital systems are becoming large and large and; obviously, they are becoming complex due to different type of applications particularly the communication applications. Now, it becomes almost impossible to design the whole system from scratch. Means the design fundamentals we have read starting from the truth table or the Boolean equations, then the reducing that thing it is almost impossible.

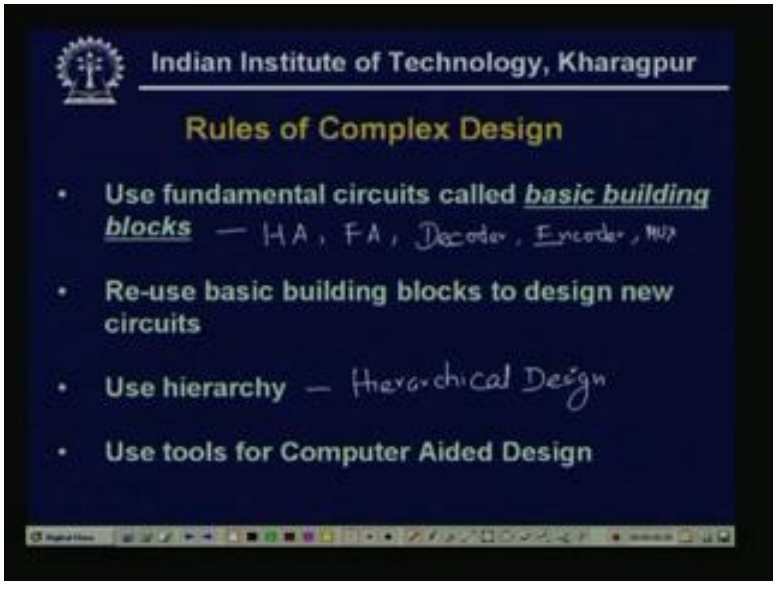
Now, we are very fortunate that starting this behavior of this large circuits, what we have observed, that this large expression, which represents the behavior of the complex

circuits that also can be realized by a combination of smaller expressions, which are all ready designed. Now, larger design can reuse the smaller predesigned module, means that one large equation as if that are partitioned into a smaller set a number of smaller equations.

Now, this equation can be realized by the rule we have read or the design rules we have already learnt. So, there are all ready some modules are all ready designed, these modules we are telling as if these are some fundamental circuits that all ready we have learnt. Now, using this modules these are the predesigned modules; that means, we can now design a big circuits.

Now, for this large complex designs as if the components are not the gate, not the AND, OR, NAND this types of gates rather they are replaced by half adders, full adders, MUX, decoders. This we are calling as if the reusing of the circuits or the fundamental circuits or the digital common digital elements.

(Refer Slide Time: 04:09)

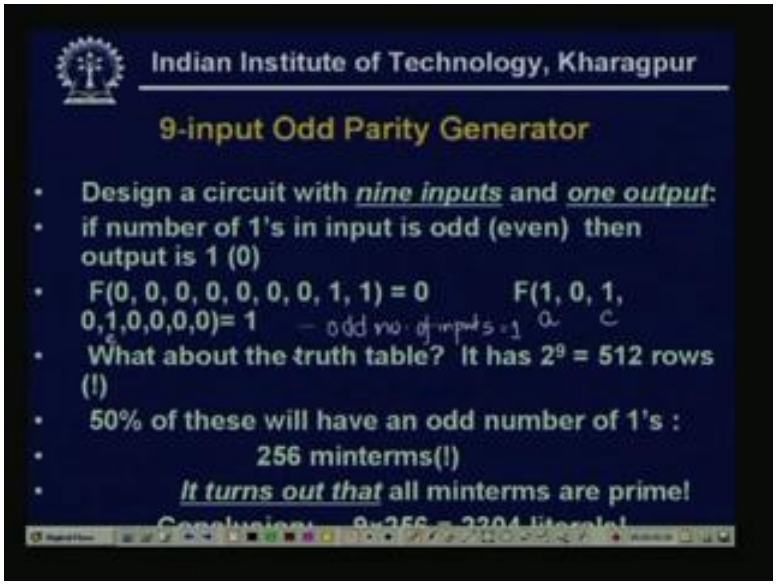


Now, what will be rules for this complex design, all ready we know that the simpler design how we can develop. Now, use fundamental circuits called the basic building block, as if this will be treated as the basic gates. So, the basic gates are now replaced by basic building blocks where this building blocks are the already, what we have read. That our half adder, our full adder, the decoder, the encoder and particularly the MUX and they can be any other smaller modules.

Now, this reuse of basic building blocks to design new circuits, so as if this smaller circuits or what we have defined that basic building blocks, they are predesigned, pre-verified that they are working correctly. They are realizing the small expressions or the equations correctly and then we can use this building block to design the complex circuits, now use a hierarchy.

That means, here we can use or the concept of hierarchical design one of such hierarchical design, last day we have seen when a larger MUX multiplexer, we have designed from the one smaller multiplexers we call that this is a multiplexer tree all ready we have read that thing. Now, as the circuits are large or that systems are becoming complex, now almost it is impossible to do this design or to develop the system manually. Now, for this purposes we use the CAD tools the Computer Aided Design tools, now first we take one very simple example, that one 9 input odd parity generator.

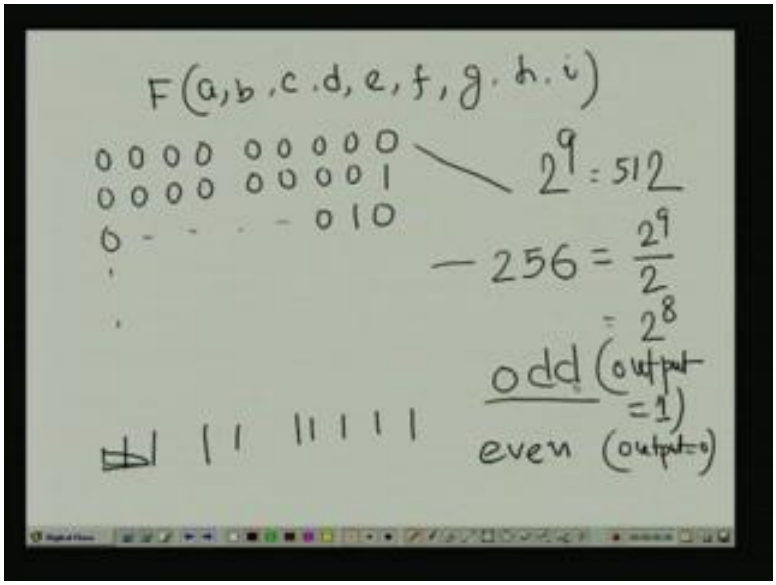
(Refer Slide Time: 06:55)



Now, design a circuit with 9 inputs and 1 output, how we have defining the parity generator, if number of 1's input is odd, then we call this is a odd parity or the then outputs is 1. And if the number of 1's are even then the output will be 0 and then it will call the even parity. So, we are defining the parity as if the total number depending on the total number of 1's in it is input and accordingly the output will be defined.

So, if I take one 9 variable function, say F and see here if we give a, b, c, d, e, f, g, h, I these are the 9 variables, then h and I are 1 1, means there are 2 inputs which are true means 1. So, if it is this equal to if it is even then the output will be 0, now if the number of inputs are 1 see here a, c, e there are only 3 inputs which are true; that means, here if it is a, c and e these are 3 inputs. So, these are odd number of inputs are 1, so the output will be 1, this we are defining as a odd parity. Now, what about the truth table, it has as it is 9 variable function, so it has 2 to the power 9 equal to 512 such rows and 50 percent of these will have an odd number of 1's.

(Refer Slide Time: 09:22)

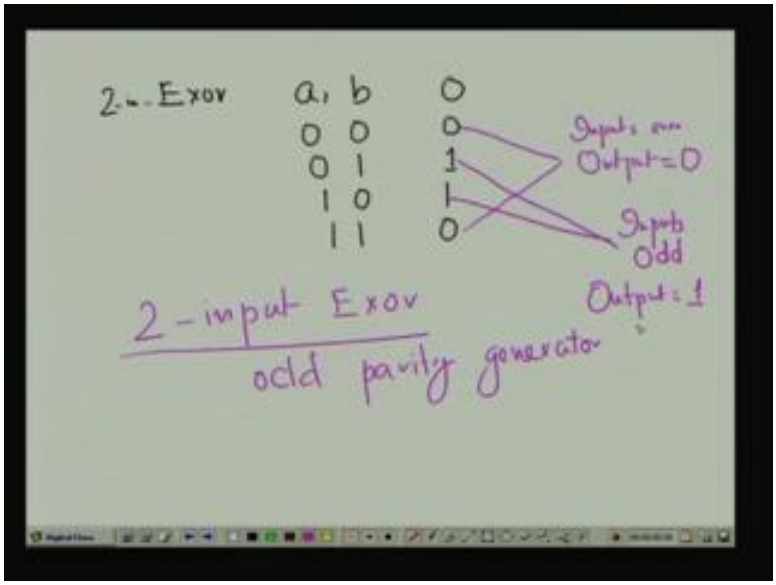


So, 256 min terms there will be 256 min terms, means if again if we explain see there are we have taken a function F of 9 variables a, b, c, d, e, f, g, h, i. Now, if I draw the truth table, then this 9 variable they will be 0 0 0 to all 1 all possible combinations I have to take. Now, if we see that here it will be 512, 2 to the power 9 equal to 512 combinations. Now, this 512 combinations there will be 256 such cases, where half of the cases 256 means actually 2 to the power 9 by 2 half of the cases. It will be odd input, odd means output is 1 as we have defined the parity odd and it will be even parity output is 0.

So, if I now I draw the Karnaugh map, then in the map, it is a 9 variable function map. So, there the number of min terms for the odd parity will be 256 because, we will be getting 256 one and so 256 min terms will exist ((Refer Time: 11:37)). Now, it turns out

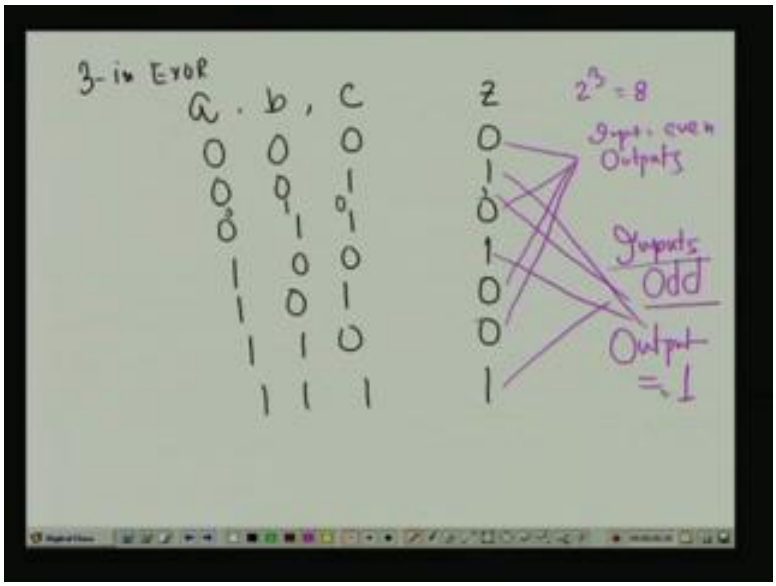
that all min terms are prime, now conclusion is that 9 into 256 is 2304 that number of literals will be will exist.

(Refer Slide Time: 12:20)



Now, again if we remember our EXOR's we see that what was our EXOR function. See our 2 input EXOR function was or if we write the truth table of EXOR, this is a 2 input EXOR say a, b output is o. Then if it is 0 0 output is 0, if it is 0 1 output is 1, 1 0 output is 1, 1 1 output is 0. See, that this two cases where the inputs are even, then output becomes output is 0, but here these two cases, see the number of inputs are odd only 1 inputs are odd, so output equal to 1. So, a 2 inputs EXOR can be treated as a 2 variable or parity function, this can be treated as a odd parity generator for 2 inputs or 2 variables.

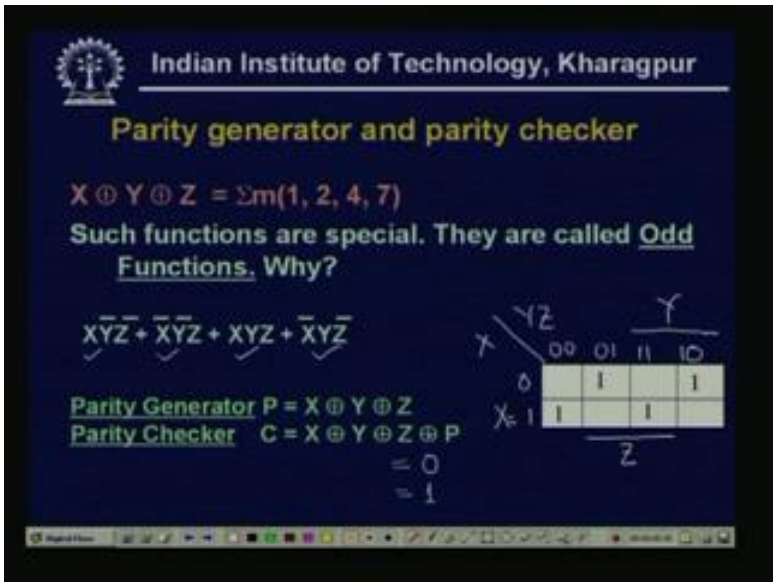
(Refer Slide Time: 14:15)



Now, if we observe the 3 input EXOR's what we will be seeing that if we change, if it is 3 input EXOR. Then, if we take the all the combinations say output is z, then it will be 0 then 1 0 0 if I take it will be 1 1 0 1 it will be 0 1 1 0 it will be 0 1 1 1 it will be 1 1 we have missed that is 0 1 0 2 then also it will be 1. Now, here there at 2 to the power 3 there are 8 combinations, 2 to the power 3 equal to 8 combinations from there that there are 4 cases, means when the inputs are even then output equal to 0.

And there are 4 cases, then it is giving the output odd or the; that means, if inputs are odd means total number of 1's in the input combinations are 1. Then output equal to 1, which is our definition of the odd parity generator, so 1 3 input EXOR that is a 3 variable odd parity generator.

(Refer Slide Time: 16:01)

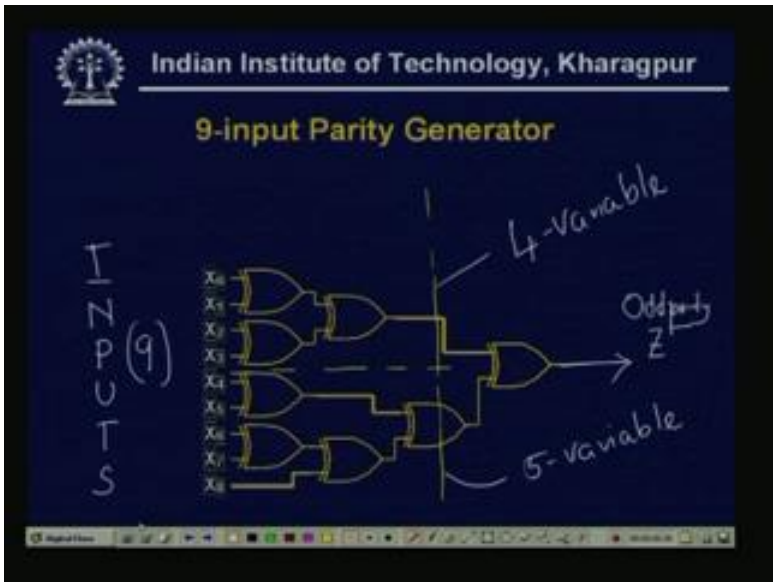


So, all ready we have read that this is nothing but, a that min terms where the if we draw the Karnaugh map that 1, 2, 4, 7 this 4 min terms will exist. Now, such functions are special they are called the odd functions. Because, they are generating as just now we have discussed, they are generating that the output 1 for input combinations odd, means odd weight means the number of 1's in the input combinations are odd.

Now, if we see the Karnaugh map if we draw that it is a 3 variable function X, Y, Z. We draw this is X and this our Y, Z then this will be 0 0 0 1 1 0 1 1 now this is 1 1 1 0 1 1 1 0. See here for this two terms Z is true and this two term Y is true, here X is true means X equal to 1. Here it is 0, now see this min term is nothing but, X, Y bar Z bar this is my the first min term, this is the X bar, Y bar, Z. This is my second min term, this is X, Y, Z the my third min term and this is X bar, Y Z bar my fourth min term.

So, this realizes our EXOR functions, now this is the parity generator that P equal X EXOR, Y EXOR, Z. Now, what will be the parity checker, the parity checker will be that if with the parity generator, if I compute the EXOR functions and we see that if it is equal to 0 or equal to 1. That means, if it is same parity then it will be giving 0, it will be a different parity, then it will be giving 1, so this same circuit or the EXOR circuit can be used as a parity generator as well that can be used as parity checker.

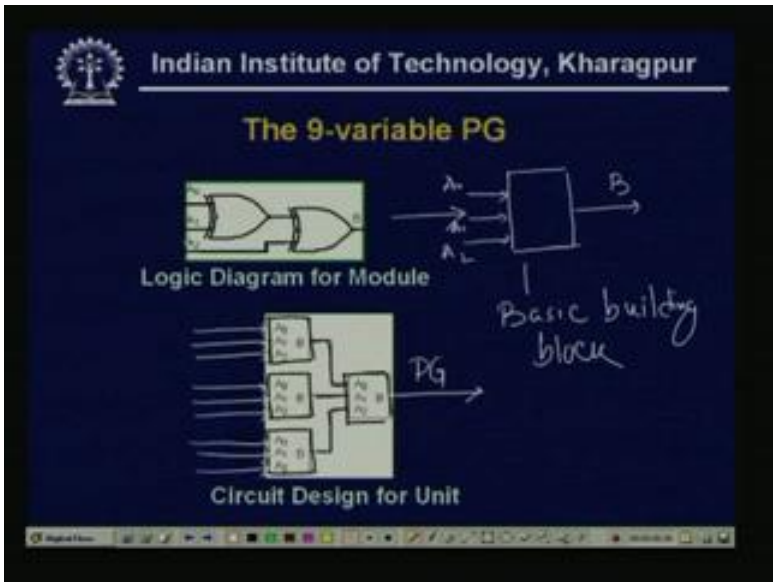
(Refer Slide Time: 19:14)



Now, if we see that the circuit that for the 9 input parity generator circuit, so as all ready we have seen, that it is a for the three variable it is a 3 input EXOR's. So, if it is a 9 input parity generator. So, it will be a 9 input EXOR functions, so if I draw the circuits see there are X 0 to X 8, these are my input variables, so these are 9 variable. And everywhere I am using a 2 input EXOR's, see here this is designing a 4 variable EXOR's, so if we partition the circuit this is a 4 variable and the lower portion this is a 5 variable.

So, this is a 4 variable and this is a 5 variable and then again another 2 input EXOR has been used which is producing the output Z or the odd parity. This is generating my odd parity.

(Refer Slide Time: 21:11)

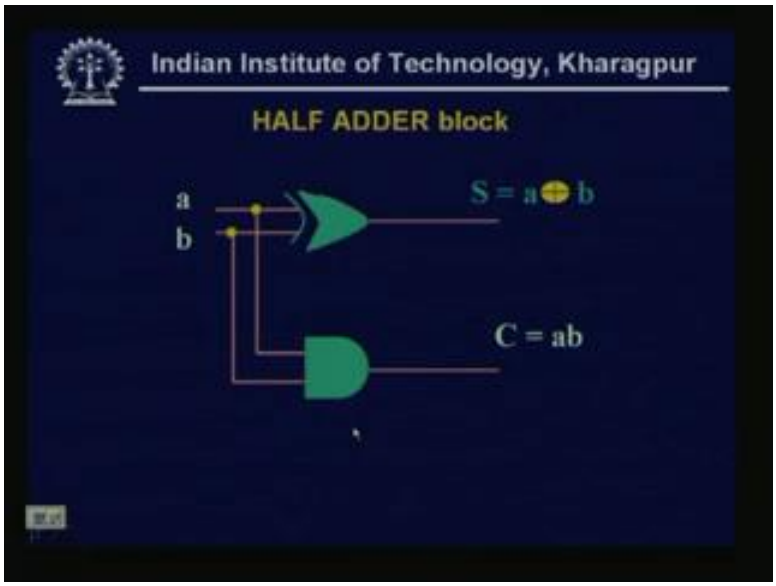


Now, see we have taken this 9 variable as if this is a 3 variable EXOR functions, so if we redesign that thing. See, if we redesign see here this is the logic diagram for the 3 variable odd parity A 0, A 1, A 2 are the 3 inputs and B are the output. So, if now these we have just redraw that thing, as if this is one fundamental circuits I am defining or this is one module, this module have 3 inputs A 0, A 1, A 2 and 1 output say B.

Now, for to design a larger circuitry or to design a larger variable parity generator, I can reuse this module as a basic building block. So, this is my basic building block for designing the larger variable parity generator, so this is my basic building block. Now, in the this second figure we have been redrawn this circuits using this thing. So, if we say this is my basic building block, block 1, my block 2 this is my block 3. And another 3 input building block we have used the output is the parity generator and this inputs or the 9 inputs of the parity generator, which I want to compute the parity.

So, this is the simple concept of the design reuse; that means one 3 input EXOR's or that 3 variable parity generator first I have design. And then using the smaller modules as if this is my basic building block, again I am redesigning the larger variable parity generator. Now, this concept we want to utilize further. Again another design example we have taken that is very common as well as very important in real life's design, that is called the multiplier design.

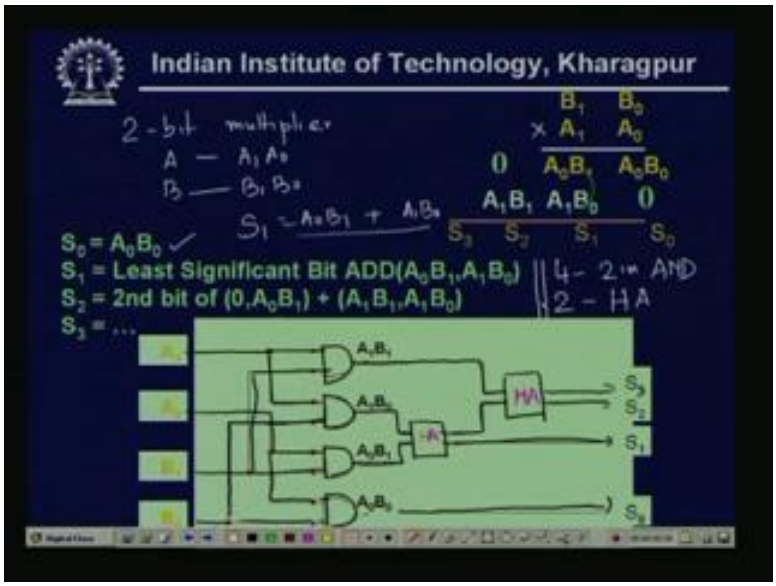
(Refer Slide Time: 24:58)



Now, before we start the multiplier design again we recapitulate quickly the our half adder full adder circuits. We remember that our half adder circuits are that there are two operands and we neglect the carry input. And sum is defined as the a EXOR b and the carry generated will be the AND of a and b, so C equal a b and C equal to a EXOR b.

Now, this can be treated as a half adder module, see this is my half adder module or basic building block whose inputs are 2 inputs only a and b and there are 2 outputs the sum and the carry. Now, we want to utilize or use this basic building block of half adder for larger designs see how it works.

(Refer Slide Time: 26:15)



Now, consider 2 bit multiplier, see that we want 2 bit multiplier and the 2 operands are A and B, see this is A is A 0, A 1, B is B 0, B 1. Now, we are multiplying these two, so from our simple multiplier rule, we will multiply in this way this will be A 0, B 0 it will be added. Then it is added with or the next term will be better we first we compute the terms, this will be A 0, B 0 second term will be A 0, B 1 then it will be shifted left in one place 1 bit shift and the next term will be A 1, B 0 and then A 1, B 1.

Now, we take the sum of the terms, so the first term will be A 0, B 0 means our S 0 is A 0, B 0, S 1 will be A 0, B 1 plus A 1 B 0, S 2 is as it is A 1, B 1 and as there is no carry for this particular multiplication, so S 3 is 0. So, we can write now that the terms or expression for S 0, S 1, S 2, S 3, so as all ready we have defined S 0 is A 0 B 0, S 1 is the least significant bit of add A 0 B 1, A 1 B 0.

See that this is the two terms this is one summation S 1 is the or S 1 is the A 0 B 1 sum of A 1 B 0. So, the sum of the two terms will be actually S 1, then what will be my S 2, S 2 is the second bit of A 0 B 1 plus A 1 B 1 and A 1 B 0, see the this is that S 2, S 2 means my the sum of the third term and the carry of the second term. So, that is written here and for this case there is no S 3, but for sum multiplication S 3 can be there.

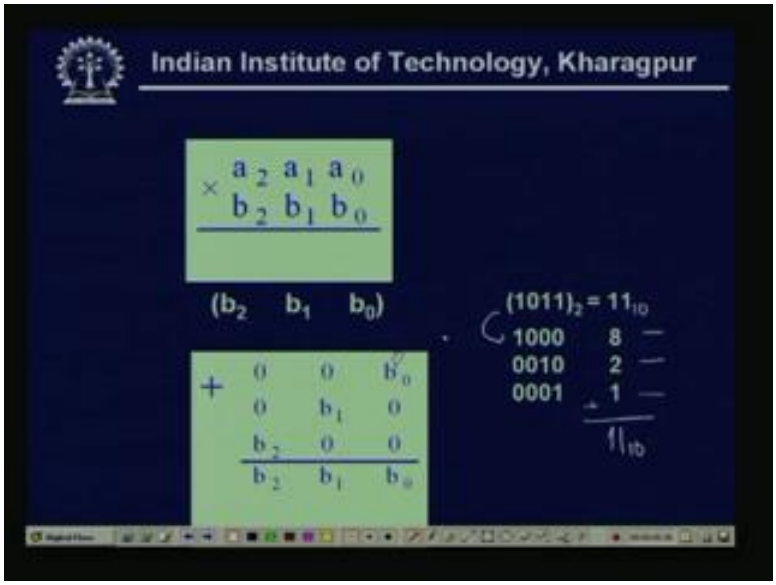
So, if we now see that what will be our S 0 see our S 0 is simple A 0 B 0, so A 0 B 0 means one simple AND gate. So, this is my AND gate and see that A 0 comes this is my A 0, A 0 is 1 input and B 0 is 1 input, so this is A 0 B 0 is S 0, what is my S 1, S 1 is the

two terms will be added and the sum of that, see first we have to add A 0 B 1. So, this is the AND gate, which is whose inputs are A 0 and B 1 and the output is fed to a half adder or this is one half adder, we need half adder module whose 2 inputs are 1 is A 0 B 1 other is A 1 B 0.

So, this is another A 1 B 0, so this is A 1 B 0 is the output of 1 AND gate, whose inputs are A 1 and B 0, B 0 itself, so this is 1 half adder module. Now, another half adder I need whose that the carry of the previous half adder, the carry of the second terms that is 1 input and other one is the A 1 B 1. So, A 1 B 1 is coming from a 2 inputs AND gate whose inputs are; obviously, A 1 and B 1 and this is sum is S 2 and if there is one carry the that is my S 3, so the product is S 0, S 1, S 2, S 3 we are getting from here.

Now, see here there are 4 AND gates, 4 2 input AND gates we need 4 2 input AND gates and 2 half adder. So, these half adders are used as the basic building blocks for a 2 bit multiplier, now we see a larger multiplier and see whether the half adder is will be sufficient or we need some other basic building block.

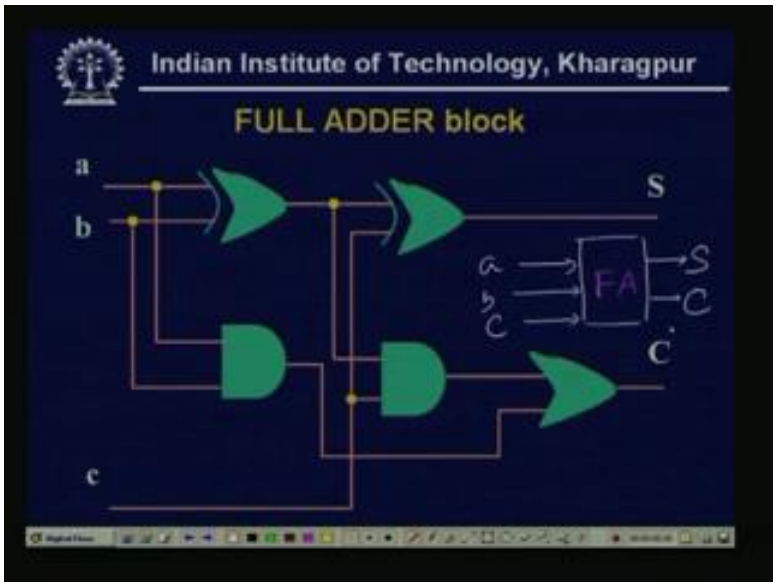
(Refer Slide Time: 33:42)



See here, if we multiply the 3 bit; that means, a 0, a 1, a 2 to be b 0, b 1, b 2 then actually b 0, b 1, b 2 we can write as if these are the combination of three. So, 0 0 b 0, 0 b 1 0, b 2 0 0 because, these are the LSB, LSB is b 0 the second bit is b 1 and the MSB is b 2. So, as if we can partition or we can break the b 2, b 1, b 0 as the three different combinations and then if we are it will be b 2, b 1, b 0.

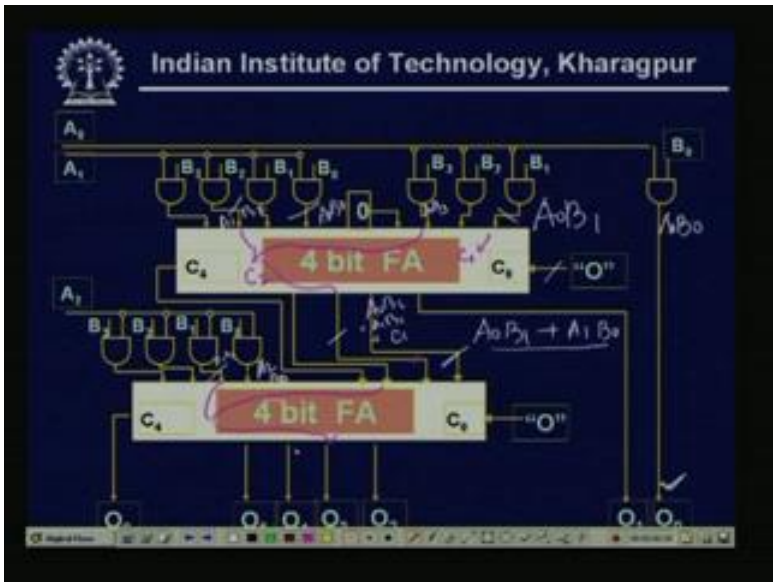
Just if you take an example, say if it is 1 0 1 1 then see our I can break that 1 0 0 0, 0 0 1 0, 0 0 0 1. So, 0 0 0 1 means this is the decimal 1, 1 0 means 2 this is 8, so 821 his is actually if we this will be 11 and 1 0 1 1 that is also my 11 and decimal, so this two are same. Now, this concept is being utilized in the larger multipliers.

(Refer Slide Time: 35:20)



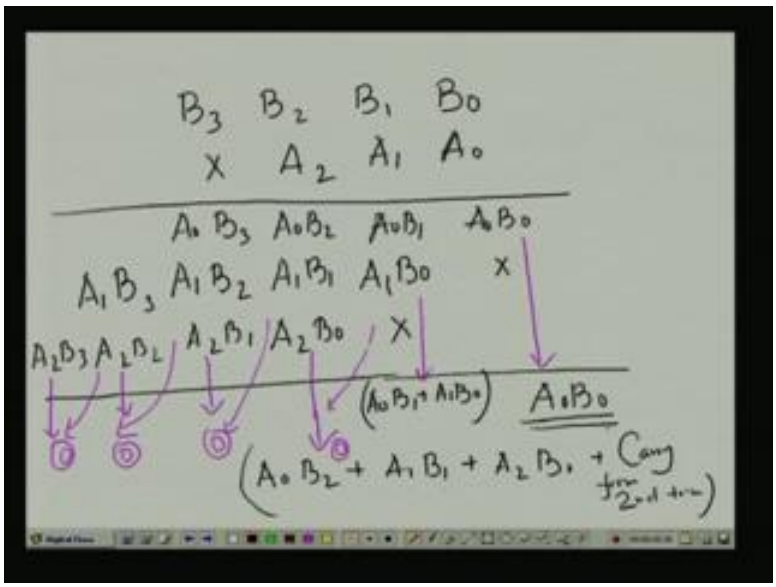
Now, before again we see that larger design of the multiplier, again we recapitulate the full adder circuit. Now, the full adder as we remember here with the two operands another operand is there, which is the carry in. Because, that as it is say it again the 2 bit adder if we consider, previous bit carry is also considered because, it is doing that sum, so S is a EXOR b EXOR c, C is a b plus b c plus c. Again we can design this thing as if this is a full adder module whose inputs are 3 a, b, c and outputs are sum and carry out. So, now this full adder can be utilized as the basic building block of the complex circuits.

(Refer Slide Time: 36:45)



Now, we take a of 4 bit multiplier say some B 3, B 2, B 0 and the multiplier is a 3 bit multiplier say A 0, A 1, A 2. So, we are taking a multiplier that B into A, where B is B 3 B 2, B 1, B 0, 4 bit whereas, A is 3 bit A 2, A 1, A 0 now if we multiplier. See, that the similar way that we have all ready we have done that multiplication if we see, if we just if we take that example that multiplier.

(Refer Slide Time: 38:10)



Say what we want B 3, B 2, B 1, B 0 and we want A 2, A 1, A 0 to be multiplied with this. Again there will be some terms it is A 0 B 0, then this term will be A 0 B 1, then A

0 B 2 then A 0 B 3, now it will be multiplied A 1 it will be shifted by 1 plus, say it will be A 1 B 0, A 1 B 1, A 1 B 2, A 1 B 3. Again it will be shifted when it will be multiplied by A 2 B 0, A 2 B 1, A 2 B 2 and A 2 B 3, now we have to add the whole thing.

So, A 0 B 0 will be as it is, so this will be this thing, now this two terms they will be added as A 0 B 1 plus A 1 B 0, see there will be no for the second term there will be no carry. Now, see for the third term it will be A 0 B 2 plus A 1 B 1 plus A 2 B 0 plus the carry coming from the second term. Similar thing will happen when we consider the fourth sum, this will be A 0 B 3 plus A 1 B 2 plus A 2 B 1 plus the carry coming from the third term.

So, we can just as if this is the carry coming from here, here the carry coming from here, similarly here the carry coming from here and the sum and here the carry coming from here and the sum. So, this way we can write that thing, as if these are the these are my sums, so now the we have to develop a circuit for which we will be doing this AND means multiplication, as well as the sum of the terms. ((Refer Time: 41:42)) Now, see this is my A 0, A 1, A 2 are the 3 inputs the multiplier and these are the B inputs the multiplicand, so this is my B multiplicand the 4 bits.

Now, see that this output of this AND gate this is A 1 B 3, so output of this AND gate is A 1 B 3, similarly this will be A 1 B 2 this is A 1 B 1 this is my A 1 B 0. Now, this term is A 0 B 1, A 0 B 2, A 0 B 3 and A 0 another AND gate is there, which is simple we computing A 0 B 0 my first term. We remember, ((Refer Time: 43:36)) the first term is this was my first term that A 0 B 0 it will be as it is, so this is my A 0 B 0 term that it will be as it is giving the LSB of the output, so this is my O 0.

Now, what was the O 1, see O 1 was A 0 B 1 A 1 B 0 which see, so O 1 is coming where that 1 input is A 0 B 1, so this is my A 0 B 1 and this is my A 1 B 0. So, it is a 4 bit full adder whose the 2 LSB's are added and there is no carry in the carry in is 0. So, it is simply giving that A 0 B 1 plus A 1 B 0, so this is my the second output bit O 1. Now, the see now as it is a full adder; that means, that carry of this sum it is propagated to the second bit.

So, now there is 1 carry say C 1, ((Refer Time: 45:28)) now if we see the second term, see here the A 0 B 2, A 1 B 1, A 2 B 0 all are added with the carry from the second term and these are A 0 B 2, A 1 B 1. See, these are A 0 B 2 the second term and this is the A 1

B 1, so A 0 B 2 and A 1 B 1 see this is my A 0 B 2, A 1 B 1 plus my carry. Now, these are added another term is there, what was the another term that is my see still this value is left A 2 B 0. So, I have to at A 2 B 0 also.

So, this where is my A 2 B 0 because, this is one term and see the LSB of the this is a 4 bit adder, then another is this is my A 2 B 0. So, this is my A 2 B 0. So, A 2 B 0 and this it will give my O 3, so 2 full adder, 4 bit full adder we are using where the LSB of the first adder with the carry from the previous term is fed as the LSB to the next full adder and the LSB of the multiplicand is the other term left that is the A 2 B 0 ((Refer Time: 47:29)).

Now, similarly for the fourth term we see that carry coming from the third term plus A 0 B 3, A 1 B 2, A 2 B 1, we see that again this is my A 0 B 3 and second term is this is my the third 1 is A 1 B 2, so this is A 1 B 2. Now, A 0 B 3 and A 1 B 2, see this is A 0 B 3 and A 1 B 2 these are added with the carry C 2, now they are feeding as the third one means there they are going here. Now, this is my another term is left another, term is A 0 B 3 and A 1 B 2 is all ready consider another term is A 2 B 1 carry is also consider.

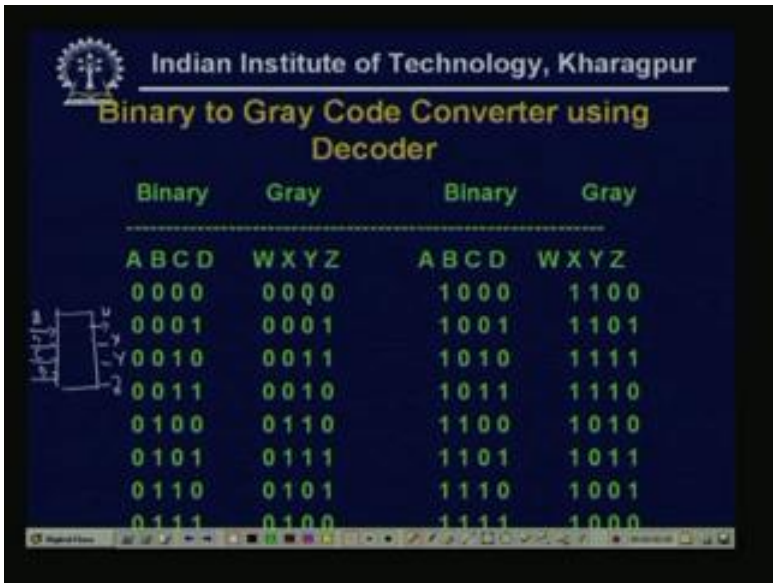
So, here see now that A 2 B 1 the second term, here this is my second one this is my second bit and here A 2 B 1 that is also second. So, for the next 4 bit full adder the second bit it takes the summation of the second bit, so this is one and this is another. So, it will be giving the O 3 and similarly the O 4 and O 5 they are also being generated.

Now, mainly what we want to show or we study from this circuit, that we are actually designing a multiplier, a 4 bit multiplicand a 3 bit multiplier. Now, just do that thing we are using some AND gates, some basic gates and 2 4 bit full adder, so this 4 bit full adder we are defining as if these are my 2 basic building blocks or 2 predesigned module that I am reusing here. So, these are some fundamental circuits full adder is a fundamental circuit, I am using that thing in my larger designs or complex designs.

Now, so far we have seen that two examples, where we have used one simple smaller circuits say 3 input EXOR, a 4 input EXOR that can be treated as a module and they are being used as the basic building block of the larger circuits. Then, we have studied multiplier, if it is a smaller 2 bit multiplier I the my half adder can be a basic building block, if it is a 4 bit multiplier my full adder circuit can be a basic building block.

Now, there are many other fundamental circuits, decoder, multiplexer, encoder, de multiplexer. Now, one such example we see where the decoder can be used as a basic building block.

(Refer Slide Time: 50:42)



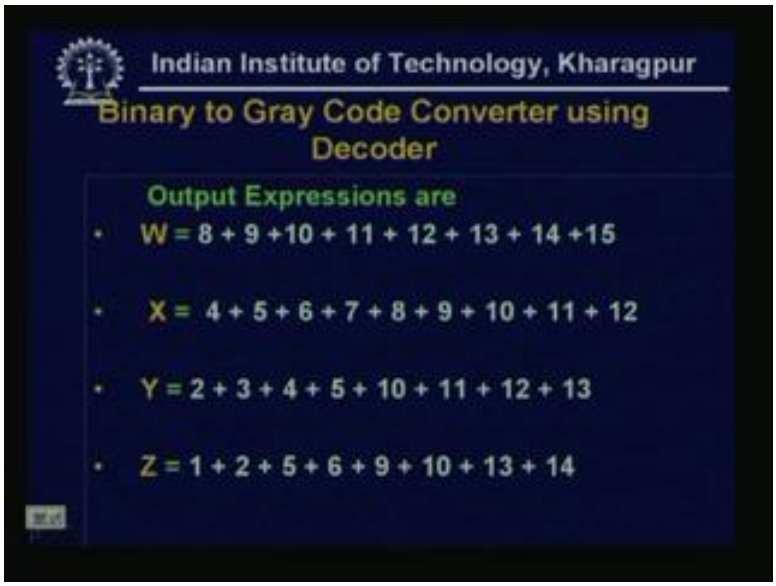
And the example is or the design is the binary to gray code converter, so the problem is we want to circuit, where the input is binary and it will generate the output which is actually the gray code, means the it gray code there is one unique property is there that it generates a unique distance it maintains a unique distance from to it is previous one. So, just we see here there is; that means, I want better I draw I want a design say this is a black box whose inputs are, A, B, C, D, outputs will be W, X, Y, Z.

So, as there are 4 inputs I am giving 0 0 0 0 to 1 1 1 1 that 16 such combinations, then for 0 0 0 0 it is a again the 4 bit output w, x, y, z as 0, then that 3 0 1 0 0 0 1 again output is 0 0 0 0 1. Now, see that here that distance is one, now distance we define that number of places the it varies, see here that it the previous one is all 0 and the next one varies in the in one place only, so here the distance is become distance is 1. Similarly, that when it is 0 0 1 0 this becomes see that now the output changes, the output becomes 0 0 1 1. So, that again it maintains a unique distance from the previous one.

When it is 3 again this becomes 2, so that it continues the property is maintained, so in this way we can generate that see this is a 4 bit for gray code generator. So, and if we observe the outputs W, X, Y, Z the patterns of W, X, Y, Z we will be seeing that always

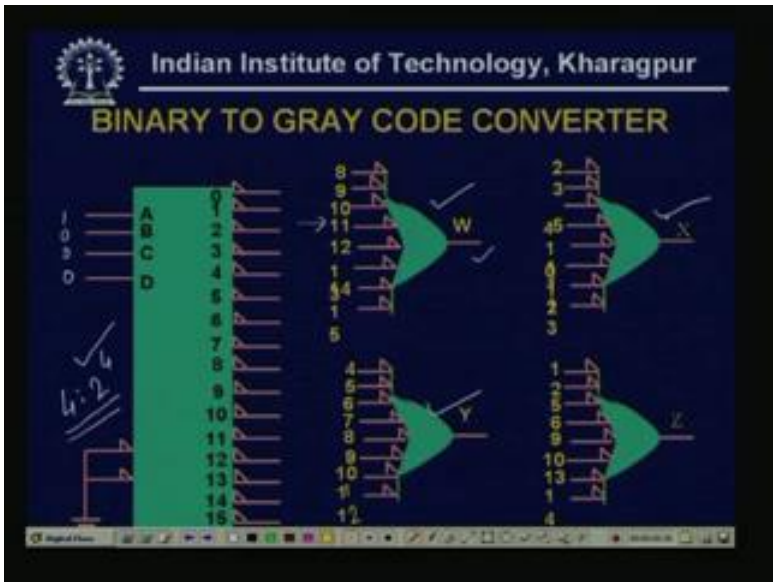
that it maintains that property that always unique distance is maintained from the previous one. So, between two consecutive patterns of w, X, Y, Z always the distance becomes one; that means, the number of places it varies is one, so now, I want to design this type of circuit.

(Refer Slide Time: 54:53)



Now, if we there are 4 output lines W, X, Y, Z, so the output expressions if we write it is we will be seeing that ((Refer Time: 55:05)) these are see if we just see that W, then W will be 1 for when A, B, C, D values 8, 9, 10, 11 up to 15 similarly X, Y, Z are there. So, we can write the output expressions that W becomes 1 when this min terms exist 8 to 15. Similarly, X will be there 4, 5, 6, 7, 8, 9, 10, 11, 12 Y will be 2, 3, 4, 5 and again 10, 11, 12, 13 Z is 1, 2, 5, 6, 9, 10, 13, 14. Now, if we draw the Karnaugh map and as we know that we can reduce the expressions for the 4 outputs.

(Refer Slide Time: 56:00)



So, the circuits will be like that, see these are my W, X, Y, Z these are my 4 outputs W, X, Y, Z, if you observe the inputs are 8 to 15 these are the decimal value I have written. And the A, B, C, D inputs are inputs given then accordingly that this outputs will be generated, which is fed to the that W, X, Y, Z generator. See, here this is a decoder, this is a 4 to 16; that means, 4 to 2 to the power 4 decoder that we are utilizing to design this type of circuits.

So, when this A, B, C, D values will be 8 to 15 means 1 0 0 0 to all 1 then my W will be generated that it will be fed to the input of the W. Similarly, when A, B, C, D value will be anyone of this means 4, 5, 6, 7 and 8, 9, 10, 11, 12, then it will be Y will be generator and similarly X and Z will be generated. So, here decoder is used as the basic building block, so in this way that larger complex circuits can be designed from the smaller modules.

We end this lecture here, but we will continue this for another different other variety of examples, thank you. In the last lecture we have seen, how the complex combinational circuits can be designed using fundamental circuits, this fundamental circuits mean the we have considered a half adder, full adder, the decoder. Now, today we will read how the combinational circuits can be designed using another fundamental circuits called multiplexer, now already we have read the functions of multiplexer, but again quickly…