

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

Lecture - 05
Digital Logic – III

We will continue the, Digital Logic lecture 3, but before that, we quickly go through the answers of the last days, lecture quiz.

(Refer Slide Time: 00:49)

Indian Institute of Technology, Kharagpur

Lecture 4-Quiz

1. Convert the decimal number 0.252_{10} to binary with an error less than 1%
2. Convert the following decimal numbers to hex and binary coded hex
(a) 584 (b) 639.45
3. Convert the following binary number to octal and binary coded octal
(a) 10101.1101
4. Write the ASCII code for the message
The email is digital@iit.ac.in

So, the last days, quiz there are four questions given, and we see the answers.

(Refer Slide Time: 01:03)

Indian Institute of Technology, Kharagpur

Answer for Lecture 4-Quiz

1. Convert the decimal number 0.252_{10} to binary with an error less than 1%
Ans. - Error allowable is $0.01 \times 0.252 = 0.00252$

$2^{-n} < 0.00252 \text{ or } 2^n > 397$

$n = \frac{\log 397}{\log 2} = 8.63$

$n = 8.63 = 9 \text{ (nearest integer)}$

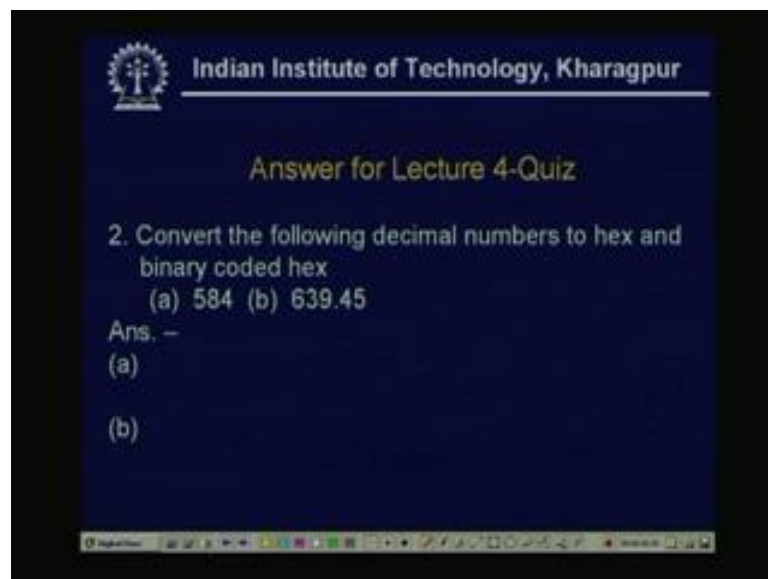
$0.252_{10} = 0.010000001_2$

The first question was the convert the decimal number 0.25 to decimal to binary with an error less than 1 percent. Now, as the error, is 1 percent, so the, what will be the error on 0.252, so will be multiplying this thing, by 0.01 into the number 0.252, so this is equal to 0.00252. Now, the truncation error; that means, when we convert the decimal number to the binary, up to what digit, we will convert it. So, if it is up to n bits that means, after decimal point, we if we take the value up to n bits, then we know the error is 2 to the power minus n, last lecture we discuss that thing.

So, this error should be less than, this 1 percent error means 0.00252, so 2 to the power minus n is less than point 0.00252, that can be written as, 2 to the power n greater than 397. So, from this expression, we can get the n, if I take the log on both side, then it will be log of 397 divided by log 2, and this is value equal to 8.63. Now, as n is the number of bits, so this should be some integer value, and that is why, we are taking, the nearest integer is 9, then that is my nearest integer.

So, when we convert 252 as binary, then we have to if we take, the number of bits, after decimal point, is not n means, n equal to 9 that means, if we take 9 bits, then it will be, the error will be less than 1 percent. So, that means, 0.252 in decimal is equal to, 0.0100 00001 in binary, that means, 9 bits after, this decimal point, now you see, the next question.

(Refer Slide Time: 03:33)



Next one is, the convert the following decimal numbers to hex, and binary coded hex, the first one, we see that 584 is the number.

(Refer Slide Time: 03:45)

Handwritten slide showing the conversion of 584 from decimal to hexadecimal using repeated division by 16.

$$584_{10} = x_{16}$$

$r = 16$

$16 \overline{) 584}$	
$16 \overline{) 36}$	8
$16 \overline{) 2}$	4
0	2

Arrows indicate the sequence of remainders: 8, 4, 2.

$$584_{10} = 248_{16}$$

Now 584, if I, this is in binary, so if I want to, convert this thing as, some hexadecimal numbers, we know we have to, divide that thing by, the radix, and here the radix is 16. So, we divide 584 by 16, then it will be, quotient will be 36 and the remainder is 8, again we divide, quotient is 2 remainder is 4, again we divide, quotient is 0 remainder is 2, as the quotient is zero, so we stop division. And, if I take, in this order, then I am getting the 584 10 is 248 in 16, so this is my hexadecimal conversion.

Now ((Refer Time: 05:06)), another part is there, that binary coded hex.

(Refer Slide Time: 05:25)

Handwritten slide showing the conversion of 584 from decimal to binary and then to hexadecimal.

$$584_{10} = 1001001000_2$$

$r = 2$

$$512 + 64 + 8 = 584$$
$$= 1 \times 2^9 + 1 \times 2^6 + 1 \times 2^3$$
$$= 512 + 64 + 8$$
$$= 584_{10}$$
$$584_{10} = 001001001000_2$$
$$= 248_{16}$$

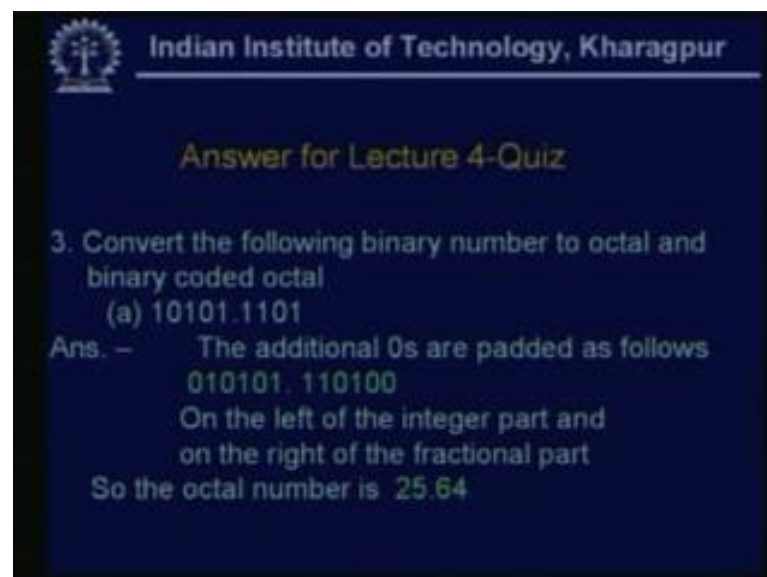
Now, if I want the binary coded hex, then what we have to do, then we will, we will be taking, say again 584 in decimal, now we will, we are converting it to binary. Now, we have to divide, here radix is 2, here radix is 2, so we have to divide by 2. Now, if we get divide it by 2, we will be getting, that it will be, that as it is see, my number becomes that, it is simply it is 584, means this is 512 plus 64 plus 8, so this will be my 584.

So, how we can represent then, then my 0th position means, one is 1 value is 0, 2 is 0, 4 is 0, 1 8 is there, then 16 is 0, 32 is 0, 64 is 1, then 120 it is 0, 256 is 0, then 512 bit is 1. So, this is my binary conversion, just if I check, this is for checking, then this is my 0th position 1 2 3 4 5 6 7 8 9, that means, 1 into 2 to the power 9 plus 1 into 2 to the power 6 plus 1 into 2 to the power 3, so this is, becomes 512 plus 64 plus 8, which is nothing but, our 584 in decimal.

So, I got my binary 584 10 is 1001001000 in binary, now I want the binary coded hexadecimal. So, for hexadecimal what, last time we discussed, that four bits, we have to take together, and that should be, from my LSB side. So, if I now mark this four, then again this four, and see two bits are left so, I will put, two zero's there, and now I take that, this is my binary numbers. Now, if it is 0010, this is 0010 means, this is my 2, then it is 0100, means this is my, 4 and 1000 that is my 8.

So, this is 248 in hex, which is matching, to our previous result, that see, when I was converting from decimal to hexadecimal, it was converting to that. So, similarly, we can do the b part also, now we, see the our third question.

(Refer Slide Time: 09:07)



Indian Institute of Technology, Kharagpur

Answer for Lecture 4-Quiz

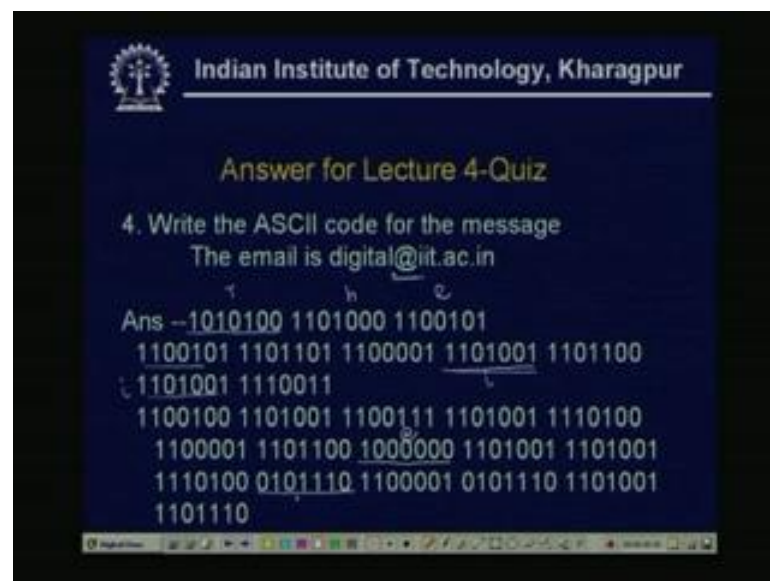
3. Convert the following binary number to octal and binary coded octal
(a) 10101.1101

Ans. – The additional 0s are padded as follows
010101.110100
On the left of the integer part and
on the right of the fractional part
So the octal number is 25.64

The convert the following binary number, time octal and binary coded octal, so this is totally similar, we will not continue this thing, because in the hexadecimal, just now we have done, there we have taken four bit together. See, that what we have done, here that we have taken, if it is, all ready it is a binary number given, 10101 and this is 1101. So, here, if it is binary to octal we have to do that means, three bits together, and here two bits are left, so put 1 0 here, then it will be three bit, so this part will become, 010 means 2, and 101 means five, so this is my 25.

And in this part again, it will be, only 114 bits are there, again if I do, I will put 2 and then, it will be 1 and again it is 5, so it will be my 8. A simple, that binary we will take three bits together, and then will, put that thing, the ((Refer Time::10:27)) octal. Now, the next question was, the fourth question was, right the ASCII code for the message, the email is digital, at IIT dot ac dot in.

(Refer Slide Time: 10:33)



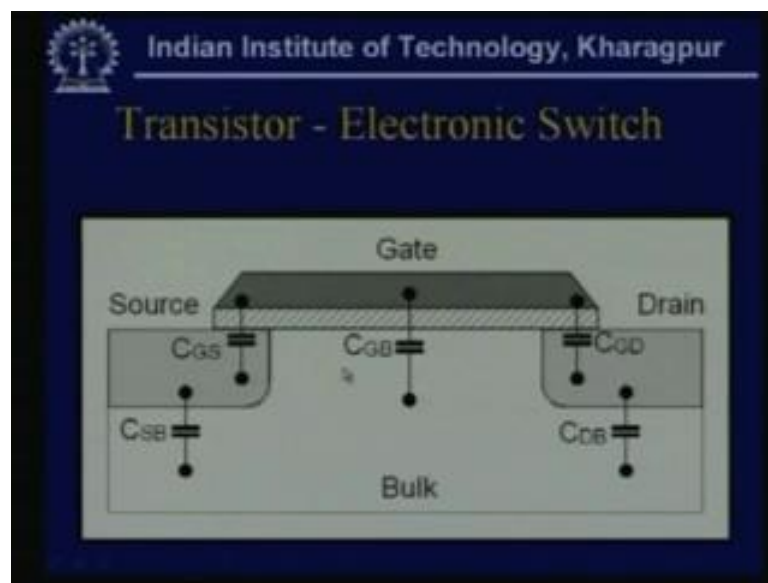
So, ASCII means, that for every character, there is some 7 bit number, now see, it is very simple thing, and what we have done, mainly for each character, I have converted the 7 bit ASCII. As, here the T is a capital one, so this is my, the ASCII of capital T, so this is my 1010100 is capital T, then this is h, and this is, so this is my, capital T this is small h, and this is small e. You see, again the email, so this e again this should be, the same e, see that 1100101, here also this is 11100101, so this is my e.

Now, this is e m a, so this is my, i, this is my, the ASCII code of small letter i, and this is l, see, this the again this is e's, so here, it should be i, and see the 1101001, so this is my

same ASCII code of i. Now, is digital, similarly, I have written all the, 7 bit code for each characters, now see, this is at the rate, so this is d i g i t a l, so this should be my, at the rate, this is my at the rate. See, this is 1 and six 0, this 7 bit code, or this 7 bit number, is the actually the ASCII code of at the rate, similarly, again this is IIT, and this is my dot, so this is my dot.

See again, that ac, so this should be, again my dot, so this is my d i g i t a l at the rate IIT, dot ac dot in, in that way I will be doing that thing. So, this is, this should be my ASCII code of IIT dot ac dot in, now we start the today's lecture, that digital logic 3.

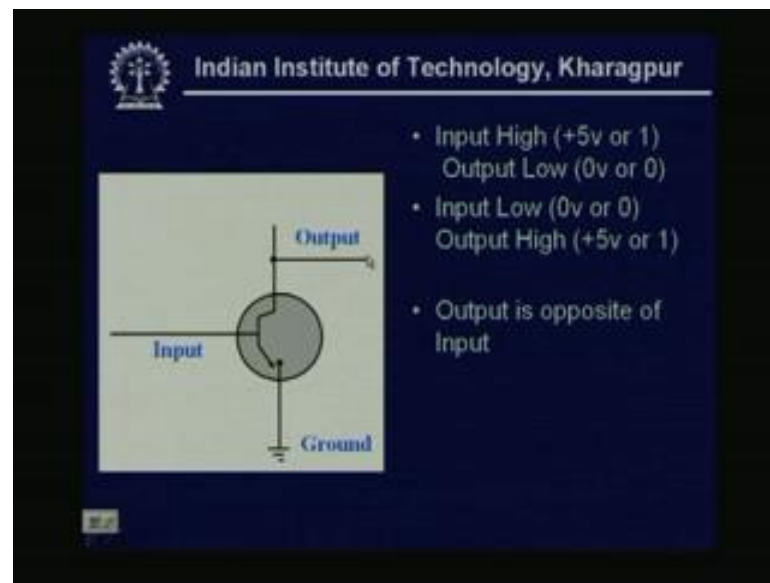
(Refer Slide Time: 13:14)



See, first, today first we see, that one basic element of digital logic, and that is nothing, but a switch, all ready the introductory class, we have mentioned. Now, how one switch is implemented, now we know, that a transistor, can be used as a switch. Now again we recapitulate, from our knowledge, the previous knowledge, that it has, the three things the base, collector and emitter, so this can be a switch, how this can be, acted as a switch.

That, if the base is high means, that one 5 volt supply is applied at the base, now this makes the connection. So, if it, it makes the connection that means, the path is on, that means, current will flow, or the base become, if the current flows means the, base becomes low that means, 0 volt. So, this will disconnects so simple, that if base is high, that it will be disconnected.

(Refer Slide Time: 14:44)



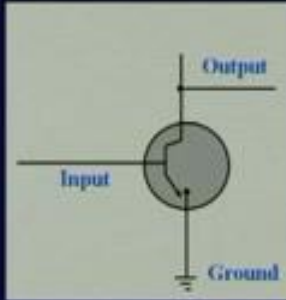
Now, this concept, we can apply here, so input and output, see, now the base is replaced by input, and the emitter, and the collector is replaced by the output, and this is the emitter is ground. So, now the if input is high, means the plus 5 voltage, and then the output will be low, similarly, the if the input is low, then obviously the switch, actually there is no connection that means, that output is will be the high plus 5 volt. So, what we are getting, that see, if the input is a high means, the plus 5 volt power supply, then we are getting, output as a 0 volt.

And, input low, is it 0 volt, output is a 5 volt that means, this is the totally opposite thing, we are getting. Now, this is the, basic concept of, that of our, that logic family, now exploiting this idea, we will developed, we will develop, that our logic families. Now, we see that, this is a Not gate, the same thing, we have taken.

(Refer Slide Time: 16:16)

Indian Institute of Technology, Kharagpur

NOT Gate



- Input High (+5v or 1)
Output Low (0v or 0)
- Input Low (0v or 0)
Output High (+5v or 1)
- Output is opposite of Input

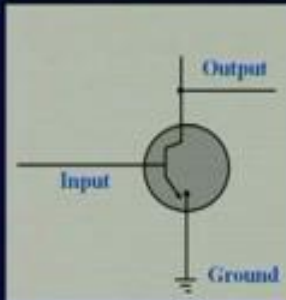
Input	Output
1	0
0	1

Because, NOT means what, that I want the what I am applying to the input, I want that, this should be inverted, that means, if my input is A, say I am getting, that my input is, now some and using some notation A. So, input high, this is a high voltage plus 5, and that I am representing as, say 1, then output is 0, say my output notation is 0, so if it is, 1 means 5 volt, I am getting output as 0, so this is my input, this is my output. Now, reverse 1 if A is 0, then output becomes 1 that means, output is opposite we are telling, this as the NOT gate.

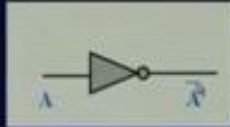
(Refer Slide Time: 17:58)

Indian Institute of Technology, Kharagpur

NOT Gate

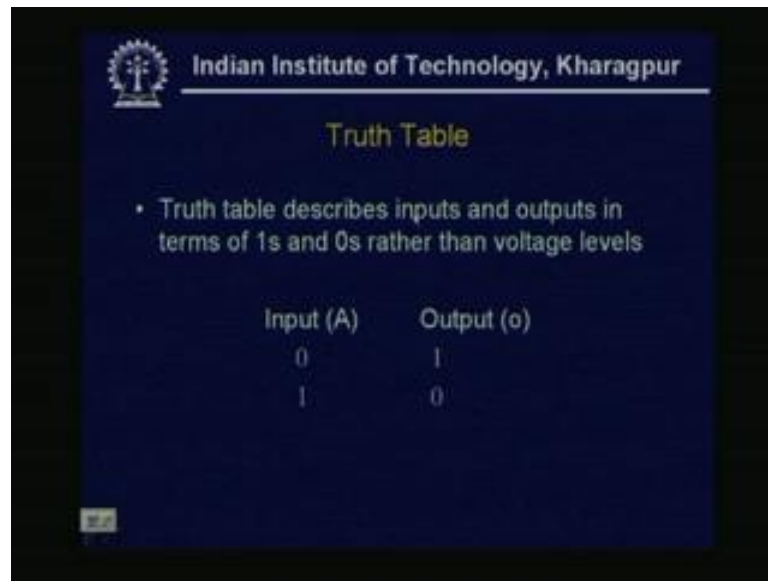


- Input High (+5v or 1)
Output Low (0v or 0)
- Input Low (0v or 0)
Output High (+5v or 1)
- Output is opposite of Input



Now, we see, now this is the, when we represent, these switching concept, we are telling this is my NOT gate, NOT gate symbol, that means my input A. And, that output is the compliment of this thing, or it is inverted, that means, when it is 0 this becomes 1. When it is 1 this become 0, and this symbol we use, for the NOT gate, we called this is my NOT gate.

(Refer Slide Time: 18:36)



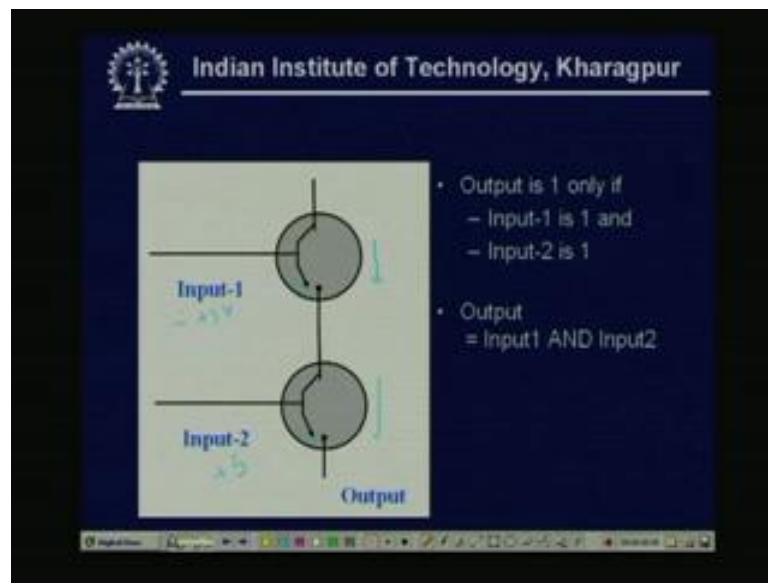
The slide is titled "Indian Institute of Technology, Kharagpur" and "Truth Table". It contains a bullet point stating: "Truth table describes inputs and outputs in terms of 1s and 0s rather than voltage levels". Below this, a table shows the relationship between Input (A) and Output (o) for a NOT gate.

Input (A)	Output (o)
0	1
1	0

Now, another thing is the truth table, so truth table describes inputs and outputs, in terms of 1's and 0's rather than voltage levels. So, just now, what we have seen ((Refer Time: 18:51)) see the input, I have given a 5 volt power supply. So, this is my voltage level, either my voltage level, is plus 5 volt or 0 volt, now if I represent, this plus 5 volt as a 1, and the 0 volt as the 0. Then the, the function the way, we are representing this function, that we are calling as the truth table.

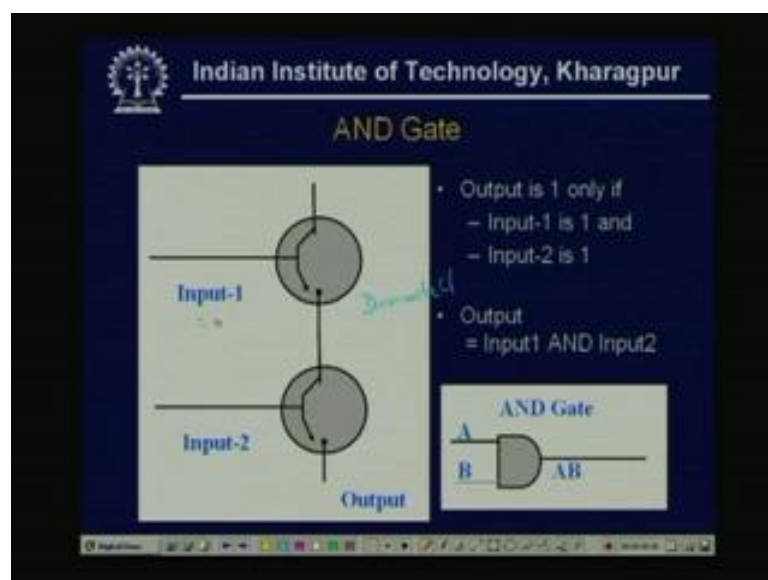
See, so this describes, input output terms, in 1's and 0's, rather than voltage levels, so just now, what we have seen, that input A, that becomes as it is only, single input line. So, either it can be 0, or it can be 1, and output will be, if it 0, it will be 1, if it is 1, it will be 0, so this, is this is the truth table concept.

(Refer Slide Time: 19:47)



Now, now we see, that a slightly complex things, see the, if the output is 1 here, when output will be 1, if both the switch are connected, that means, see here, here this switch will also be connected, here this switch also be connected, this is also connected, this should be also connected. So, what I want for that, I want, then my this, I want this input 1 that should be equal to the plus 5 volt, I want my input 2 should also be plus 5 volt, then, this should be my output should be 1. So, output is input 1 and, and 2, input 2, that means, there should, if it for this, this is at there will be two inputs and one output.

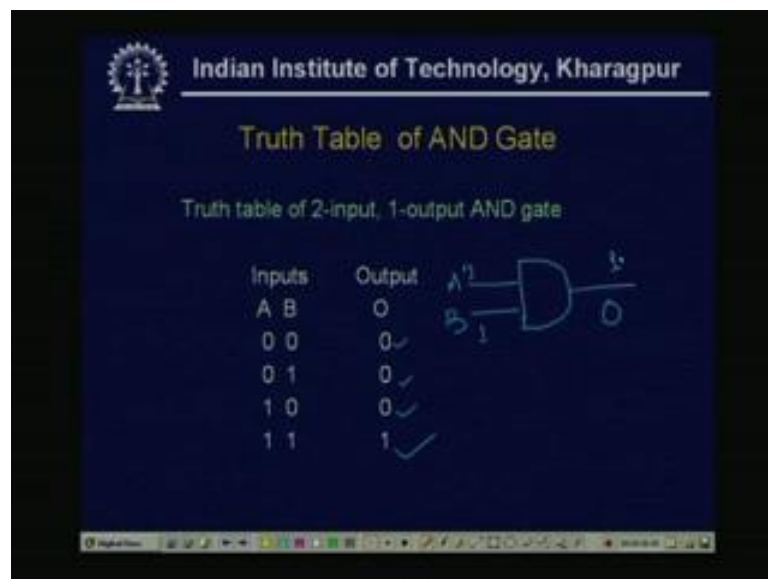
(Refer Slide Time: 21:20)



So, how I represent that thing, see, I can represent, by this circuit, by this notation, so this is normally called the AND gate, means, the if it is two input, that means, if both the inputs are 1, see then, both the switches are connected, and that means, output is 1. Now, if any one of the switch is disconnect, then actually output my is 0, so if any one of the switch is disconnected means, my input is 0 or 0 volt. If I consider voltage level, that means, my if I put, or we write in this way, that say my input, this one is, either this one is, disconnected, this is disconnected.

Disconnected means, that input is 0, and this is connected means, output is 1, but whatever, be the case then output is 0, even if it is 1, and if it is 0, then also it is 0, output is 0. And if both are disconnected, then obviously, the output is 0, because there is no path for current flow. So, now if we, represent this thing, by a truth table fashion means, we want, we describe the input output relation, by using the notation of 0's and 1's where the same thing that 1 means that some high voltage, 0 means some low voltage. But instead of voltage levels, I am using the 0 and 1, and that we have defined as the, truth table.

(Refer Slide Time: 23:26)



The slide is titled "Indian Institute of Technology, Kharagpur" and "Truth Table of AND Gate". It includes a subtitle "Truth table of 2-input, 1-output AND gate". The table shows the relationship between inputs A and B and the output O. To the right of the table is a circuit diagram of an AND gate with inputs A and B, and output O.

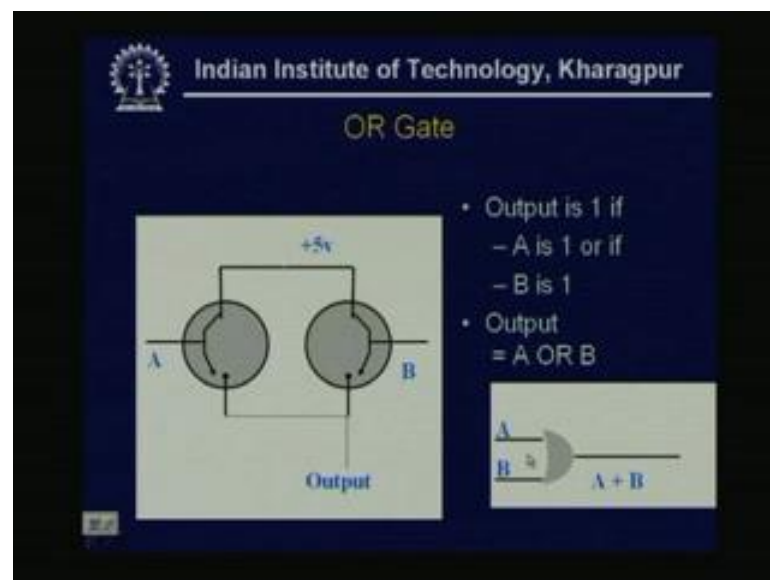
Inputs		Output
A	B	O
0	0	0
0	1	0
1	0	0
1	1	1

So, if we see that, see the truth table of AND gate, so this is, if it is a 2 input AND gate, and 2 input 1 output. So, just now, what we have drawn, that this was the, this is another logic gate, we call this is my, 2 input and 1 output, so this is input is A, another input is B, and this is my output say Z, or say I am taking, this is as O. So, this is my, this is my

output 0, now what we can see, that if A and B, the first value is, that A and both the inputs are 0, 0.

That means, if A equal to 0 and B equal to 0, now what will be the value, that means, both the switches are disconnected, so the output will be output will be 0. So, this is the first value, now, if A is 0, but B is 1 that means, remember that, one switch is disconnected, and in this case, the again the output will be 0. Similar, thing happens for the third case, and only the output will be 1, when both the inputs are 1 means, both the switch are connected, so when the input is only 1, 1 then only I am getting a 1 at the output. So, here only if, both the inputs are high that means, plus 5 volt, then only I am getting a, plus 5 or 1 at the output, so this is the truth table of AND gate.

(Refer Slide Time: 26:02)



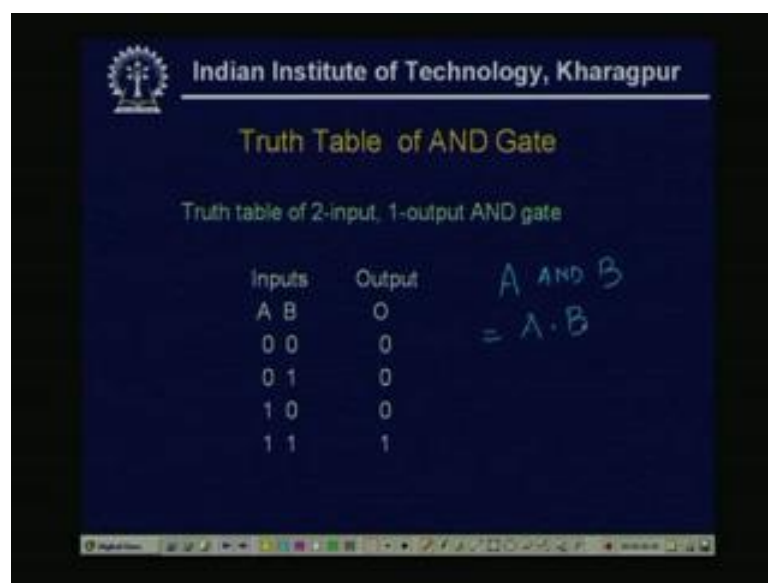
Now, we see, the another basic gate, and call the OR gate, so see, here the switches are connected in a slightly different way, see that, in previous cases, the it was connected, we can tell these are in series connection, the two switches as if they are connected in this way. Now, here, they are connected in parallel, now see that, say that these line, is a some plus 5 volt power supply is attached, that is to the it is connected here, now at the base, we are applying the, some input, and here also some different input B.

Now, if I change my voltage, at this two lines, what will happen, how, it will behave, so the output is 1 means, now 1 means, this is a plus 5 voltage, output is 1, if A is 1 or if B is 1. Means, if either one of these switch is connected, then actually it is getting a path, so it will be, flowing ,also the output is 1, that means, this 5 volt I am getting here, and if

both are connected, then obviously, it will be 1. But, if both are disconnected, only that is the situation, when the output will be the 0.

Because, if both are disconnected, then there does not exist, any path, this there does not exist any path, to get this high voltage, in this line. So, if we summarize, what will happen, output is 1, if A is 1 or if B is 1, and output is written as A or B, so this is my concept of OR gate. How the gate is denoted? So the gate is denoted like this notation. So, the two inputs are A and B, these are the two inputs, and only one output, again this is a two input one output, OR gate and normally the notation is A plus B.

(Refer Slide Time: 28:53)



Indian Institute of Technology, Kharagpur

Truth Table of AND Gate

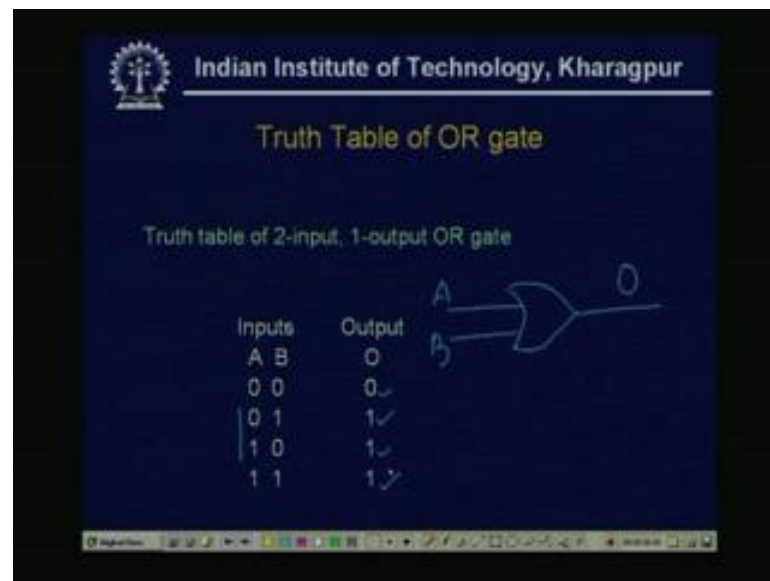
Truth table of 2-input, 1-output AND gate

Inputs		Output
A	B	O
0	0	0
0	1	0
1	0	0
1	1	1

$A \text{ AND } B = A \cdot B$

And, the AND gate normally, it is written as A dot B, so for AND Gate, this is written as, A and B is A dot B, it is the notation, and for the OR gate, this is A plus B ((Refer Time: 29:23)) then, what will be the truth table.

(Refer Slide Time: 29:45)




Indian Institute of Technology, Kharagpur

Truth Table of OR gate

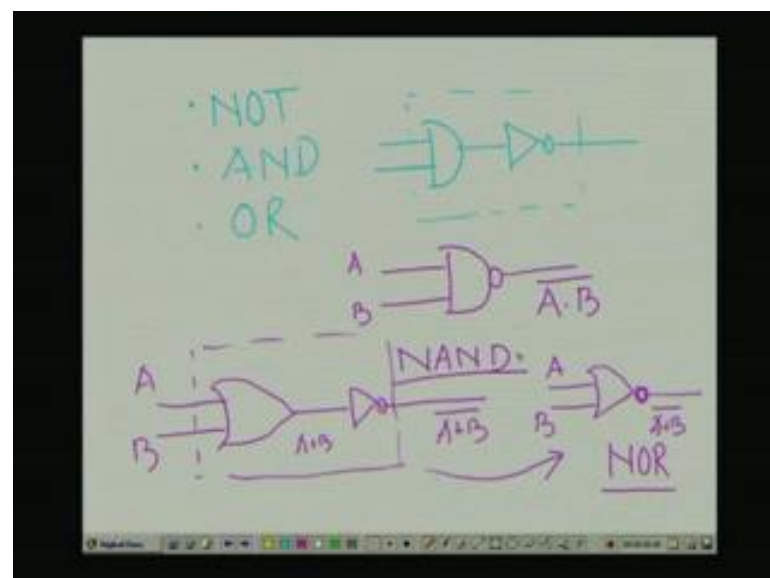
Truth table of 2-input, 1-output OR gate

Inputs		Output
A	B	O
0	0	0
0	1	1
1	0	1
1	1	1



See, the similar way, this is a 2 input A B and output O, so again if we draw, so if we draw, the OR gate, this is a 2 input OR, now input lines I have given, A and B and output is O. so if AB is 0, 0 then, both are disconnected both the switch, so output will be 0. If A is 0, B is 1 or A is 1 B is 0, so if any one of the switch is connected that means, it is getting a path, so output becomes 1. If both are connected, then obviously, it is getting, in both the direction, it is getting the path. So, this becomes 1, so this is my, truth table of the OR gate.

(Refer Slide Time: 30:51)



Now, if I put a AND gate, so mainly these three are the, we so far we get, three basic gates, that NOT, then AND gate, and then OR gate. Now, from these three basic gates, again I can generate, two other very important gates, again they will be treated as basic, so that means, if say this is one 2 input AND gate, and if I put, a NOT gate here, then I am telling, these I am converting as a, as the 2 input NAND. That means, AND these notation that means the, these notation always means, that it is inverted, that this is my A dot B complement, so this is my NAND gate, and it is normally this is represented as NAND.

Similarly, if I take instead of the AND, if I take a 2 input OR, then what will happen, I take a 2 input OR, and I put a NOT gate here, so this is my A, B, so this, this is A plus B and this will be A plus B complement. And, this whole things, again similar way, if I replace this, to it will be, a NOR just OR, and with a circle at the end, this will be A B and it will be, A plus B complement, so this is my NOR. Again this NAND and NOR are two other basic gates.

(Refer Slide Time: 33:56)

2-input NAND = 2-input AND

Inputs		Outputs	
A	B	O _{AND}	O _{NAND}
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Now, what will be the truth table of, of this NAND and NOR, see what, we have seen, that if it is a 2 input NAND, then it is actually, equal to what we can write, that 2 input AND, that is complemented. So, what will be the truth table, if I write together, say this is my inputs the A and B, these are my outputs, say this is the I am giving O, output for AND, and this is the output O for NAND. Now, see what will be the thing, say A B it can take 2 values, because 2 binary inputs means 2 square four.

So, if it is 0, 0, that what will be the AND, remember, it will be obviously, it will be 0, no path, and it is the compliment, so 0 NAND is, all ready we have written, the 2 input NAND, is a 2 input AND compliment. So, 0 compliment means, this becomes 1, similarly, if it is a 0 and 1, again it will be 0, so this will be 1, it will be 1 0, then also it is 0, then compliment of this is 1. If it is 1 1, then only I am getting, a 1 output, at that AND gate, and obviously, this will be 0. So, see, the output or the truth table output, of NAND is the complement of the truth table output, for the AND.

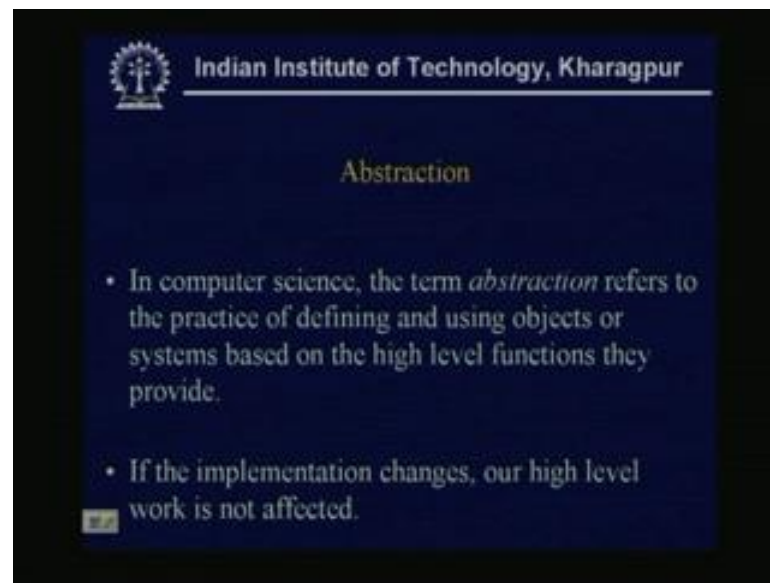
(Refer Slide Time: 36:16)

$2\text{-in NOR} = \overline{2\text{-in OR}}$

Inputs		Outputs	
A	B	O _{OR}	O _{NOR}
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

So, similarly I can get the NOR, because my 2 input NOR, is nothing but, the 2 input OR compliment, complimented. So, similar way, if I can draw the truth table, then the, my inputs are similarly A and B outputs, are output for OR, and output, for NOR, again you take that, 2 values if it is 0 0, output OR, output for OR is 0. If it is, now it is complimented, or inverted. So, it will be 1, now if any one of the input is 1, then actually, the output will be 1, it will be 0, if it is 1 0, then it will be 1, it will be 0, if it is 1 1 then OR is 1, and output is 0, so this is my truth table of the OR gate.

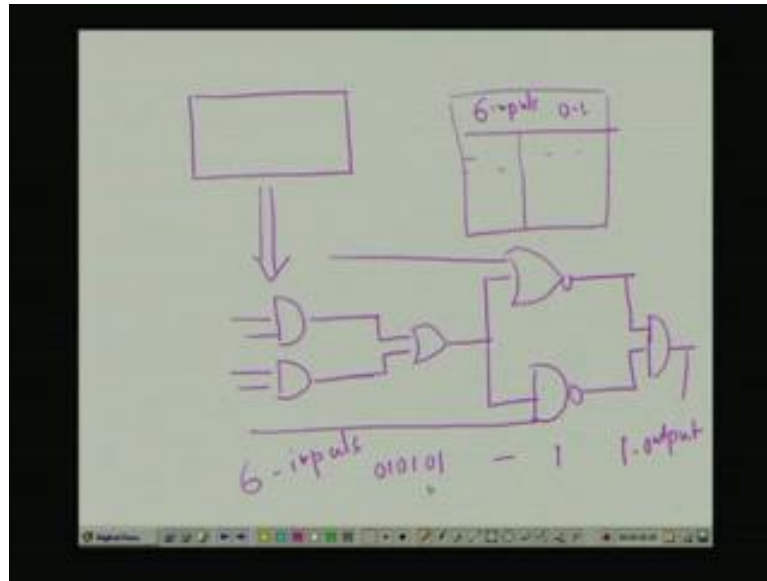
(Refer Slide Time: 38:00)



Now, we see that, what do you mean by some abstraction, now in computer science, the term abstraction, refers to the practice of defining, and using objects or systems, based on the high level functions, they provide. So, just now, what we have seen, the AND gate, the OR gate ((Refer time: 38:28)) these are nothing but, some functions, now a big system I can represent, by with the, with using this, a number of, this AND, and OR gates.

Now, abstraction is that, how we are defining a, object or a big system, high level on a on a high level functions, and this is the abstraction, in computer science. So, if the implementation changes, our high level work is not affected, that means, if your, that AND gate OR gate that means say, I am taking one example.

(Refer Slide Time: 39:24)

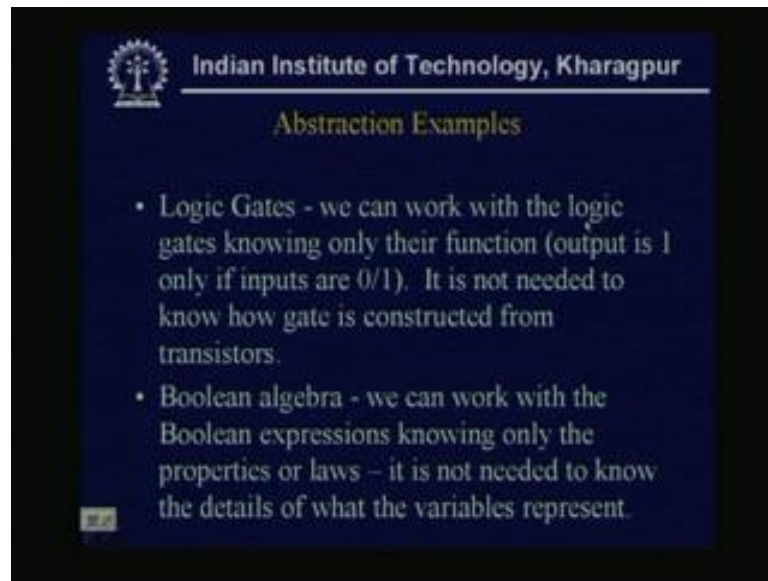


Say I am, I am taking one high level system, say one high level system, I am just giving a black box. And, it is represented, say if this is, represented as, say one sub system is represented like, some 2 input AND, number of 2 inputs AND'S, then some OR, then say some NAND, or, or NOR gate, I am giving NOR, say it is some, say it has another input, it has also another input, and this way. Now, see this is one circuit, now if I change, because that is, it will give some, say it has, 1 2 3 4 5 6 inputs, 6 inputs and 1 output.

Now, if I change, these some of the AND or OR gates, so still I am getting the same functions, that means the truth table, it is, it is giving that for 6 inputs, there will be, some truth table, some value 0101, here also I am getting some output, these are for my inputs, and these are my outputs. Now, if I change, some of this thing, still the behavior is same that means, for the same input, I am getting the same output, say if it is my, input is say 010101, so I am getting 1 as output.

Now, even if I change, some of the gates, still I am getting a 1, then abstraction means, that is how it is implemented, whether this, AND gate is OR gate, or the previous AND, OR gate is replaced by a NOR gate, it does not effect, good. So, this is we are calling that abstraction, of a digital circuits.

(Refer Slide Time: 41:50)

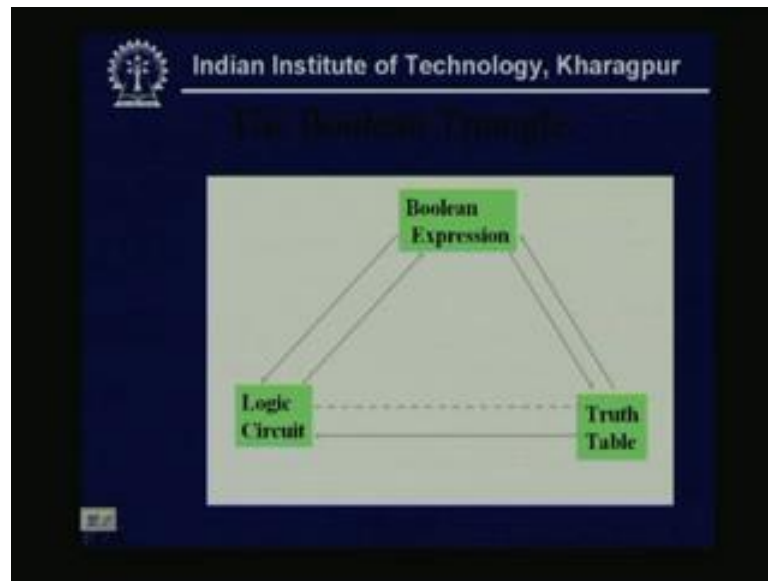


Now, we give the abstraction as an example, just now, we have I have taken, that one big system, I have represented by, some logic gates, AND, OR, NOT, NAND, NOR, these are the some basic gates, we just now read, and using that I have given. Now, this logic gates, we can work with the logic gates, knowing only there function, means that, just now we have seen, that if it is a 2 input AND gate, that it will be, output will be, only 1, when the 2 inputs are 1.

So, these are, we are calling the function of the AND gate, that or, that means, output is one only, if inputs are either 0 or 1, it is not needed to know, how gate is constructed from transistors. So, similarly, that I am getting the behavior of a large system, I do not know, that or it is not needed, It is not needed to know, that how the gates are connected or, which gates are connected. So, similarly, another example is the Boolean algebra, so we can work with the Boolean expressions, only the properties or laws, it is not needed to know, the details of what the variables represent.

So, all ready, we have reads the logic gates, and now later, we will be seeing, that Boolean algebra, again that can be or Boolean expressions. And, some of the operations, defined on the Boolean expressions or Boolean variables, that we can tell the Boolean algebra, that also be, the can be used for the abstraction, for digital systems.

(Refer Slide Time: 43:45)



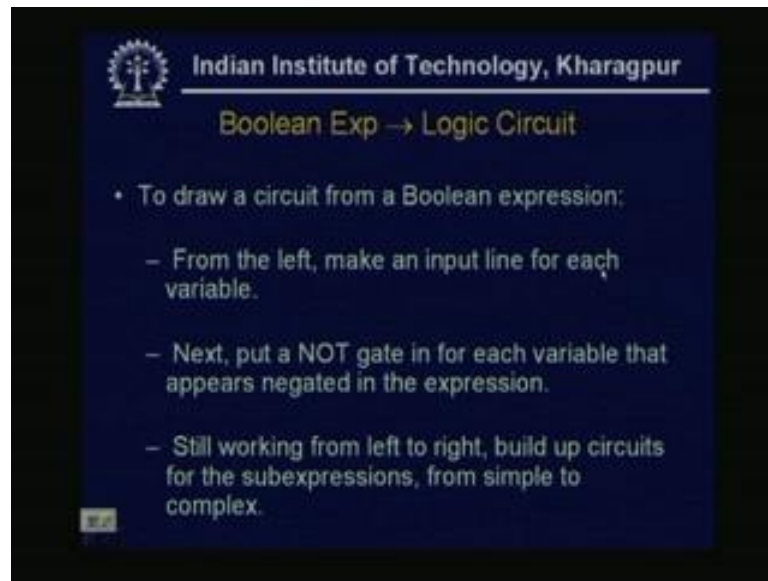
Now, so far we have read the logic circuit, and truth table, so now, again we will read the, how this can be used as a Boolean expression. Now, just we see, that some expressions, using some Boolean variables, and what are Boolean variables, Boolean variables means, it can take one variable, can take only 2 values, either 0 or 1 that means, all binary variable. So, these are Boolean and using that, some expression is written, so this is Boolean expression.

Now, how they are related, so these, that from if a Boolean expression is given, this arrow means, if one expression is given, from there I can draw a logic circuit. Logic circuit means, one circuit consists of the, a number of logic gates, and the gates are of type, just now, we have read the OR, AND, NOT, NAND, NOR etcetera. Now, for if the Boolean expression is given, I can draw a truth table, what is truth table, truth table is the description, of inputs and outputs with respect to 0's and 1's only, to describe the, it is overall behavior.

Now, see from the logic circuit to Boolean expression, if a Boolean expression is given, I can draw the logic circuit. In reverse direction, if the logic circuit is given, from the, from there I can also, compute the Boolean expression, say if the logic circuit is given, I can draw the truth table, and if the truth table is given, the logic circuit can easily be drawn, from the truth table. If the truth table is given, from the Boolean we can, or compute the Boolean expression, we can write the Boolean expression, for the circuits, so this is called that, the triangle or the abstraction triangle.

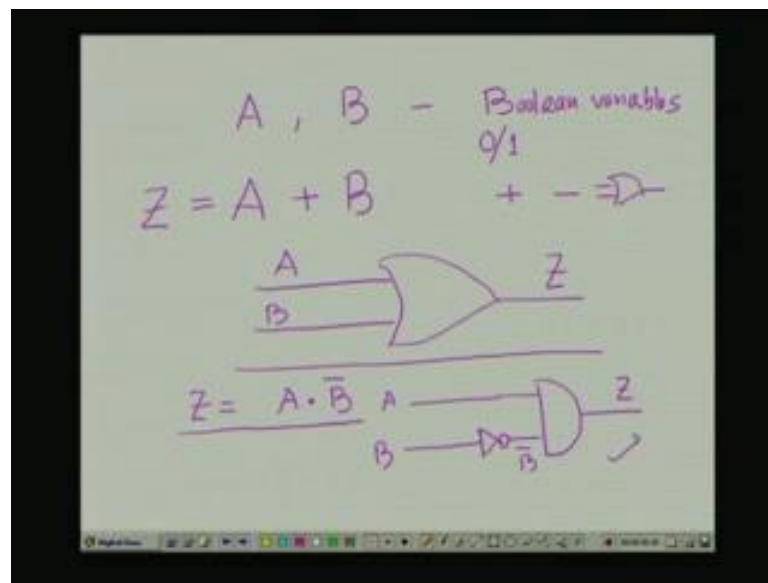
Now, one by one, we will see first, that how, if the logic circuit is given, how we can, write the Boolean expression, how we can write the truth table. Then we will be seeing that, if the expression is given, how we can draw the logic circuit, how we can the truth table. If the truth table is given, how the Boolean expression can be written, how the logic circuit can be written.

(Refer Slide Time: 46:42)



First, we see Boolean expression to logic circuit, now to draw a circuit, from a Boolean expression. So, from the left make an input line, for each variable, say I have some variable, first we see then, we will be giving example, first we read the rules of variation, from the left, make an input line, for each variable, next put a NOT gate in for, each variable, that appears negated in the expression. Still, working from left to right build up circuits, for the sub expressions, from simple to complex, see, first we take one example.

(Refer Slide Time: 47:45)

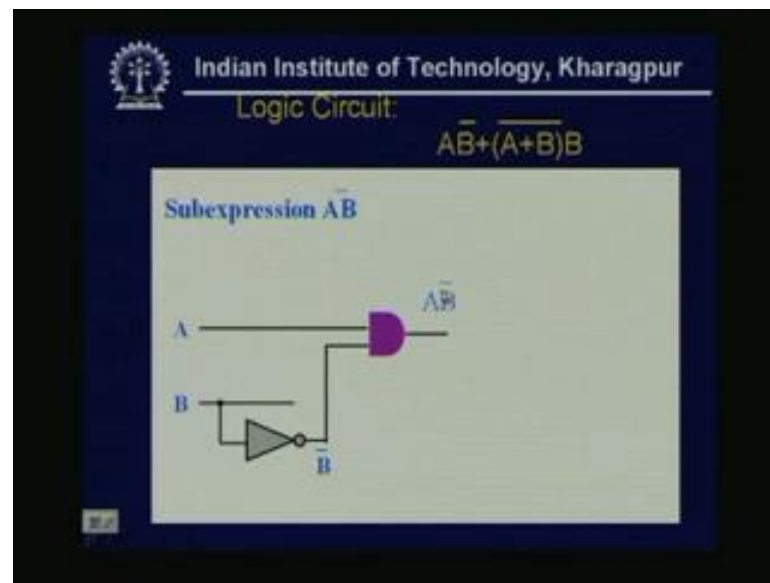


So, one I am taking, A and B are 2 Boolean variables, Boolean variables, so that means, A and B can take values, 0 or 1. Now say, one expression is given, say A simple expression I am taking A plus B, that means, these are the 2 Boolean variables, this is one operator, what operator it is, our OR operator. So, we know that for this, one logic gate exist, for one plus I know, that my logic gate is OR, so from left, what was the rule the left, just we give a notation for variable, or the say, this was one line.

So A is, A will be replaced by a line, similarly, B will be, replaced by a line, then the, it will be replaced by my, logic operator, by my logic gate. So, this is nothing but, your Z equal to, this is the circuit, so if the expression Z equal to A plus B, then this is my logic circuit for that, now say, Z equal to say, A dot B complement is given. Then for A one line, now for B another line, and, as it is complemented or inverted, so I will put a NOT gate there, so this is my actually B complemented.

Then it is dot means, it will be replaced by a AND gate, and this is my Z, so this will be the Z equal to A, B bar. So, so simple OR gate and the AND gate, can be extracted, if this Boolean, 2 Boolean expressions are given.

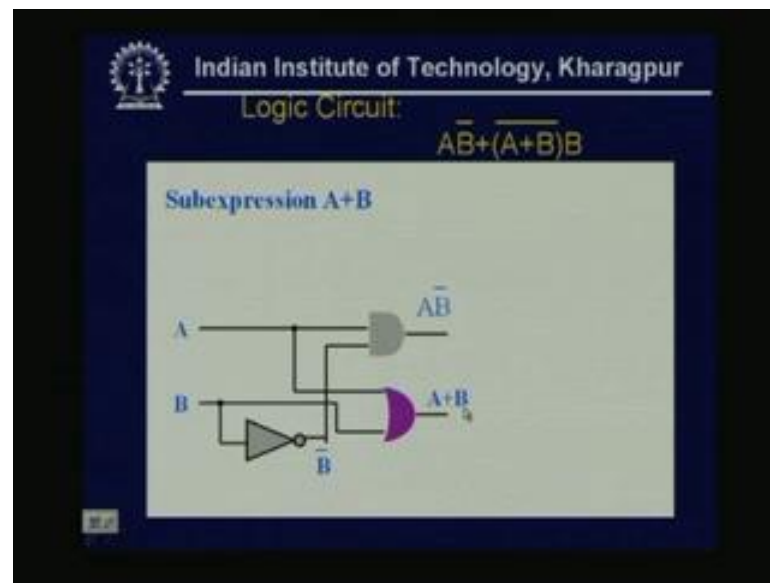
(Refer Slide Time: 50:29)



Now, we see, a slightly bigger example, see one logic circuit is given, that is or, or a expression is given, that $\overline{A}B$ complement, plus $\overline{A+B}$ dot B. Actually here, we can tell, that AND is there, this is dot, this is also dot, now for see, there are two variables only, A and B, so I give two lines A and B. Now, see, I have another, this will be B complement, so I have another variable needed, that is complement of B, so this is B complemented line, this is for that, one NOT gate is, put on the path of B.

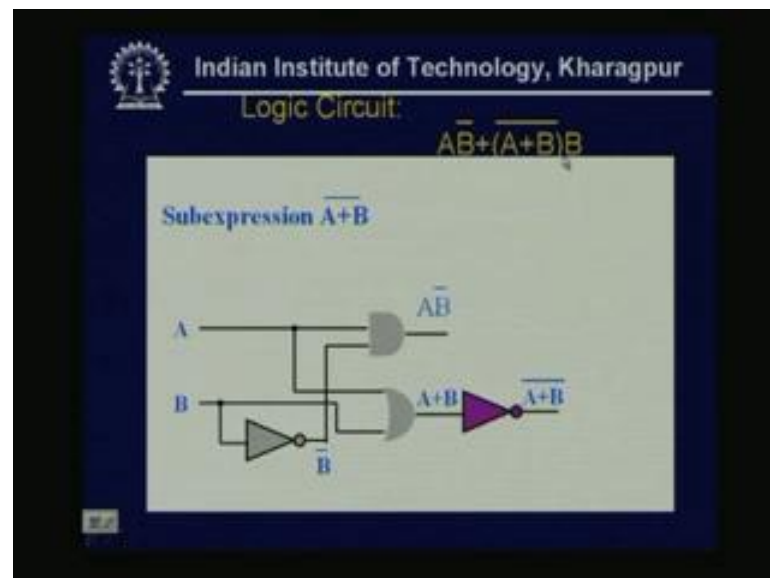
Now, I want $\overline{A}B$ means, \overline{A} dot B, so I have A line, I have B line, so I will put, one 2 input 1 output AND gate, that whose output, is that $\overline{A}B$. So, these sub-expressions, sub-expressions for A, B, that is, this is the circuit for the, sub-expressions A, B.

(Refer Slide Time: 52:44)



Now, still I have another sub-expression, and that is A plus B bar dot B, so first we see, how I can draw A plus B bar. Now, see A is available, B is also available, so A, and B, I give a, OR plus means my OR operation, so one 2 input OR gate, and the output will be A plus B. So, this is the simple sub-expression of A plus B the extra that is added with the previous picture.

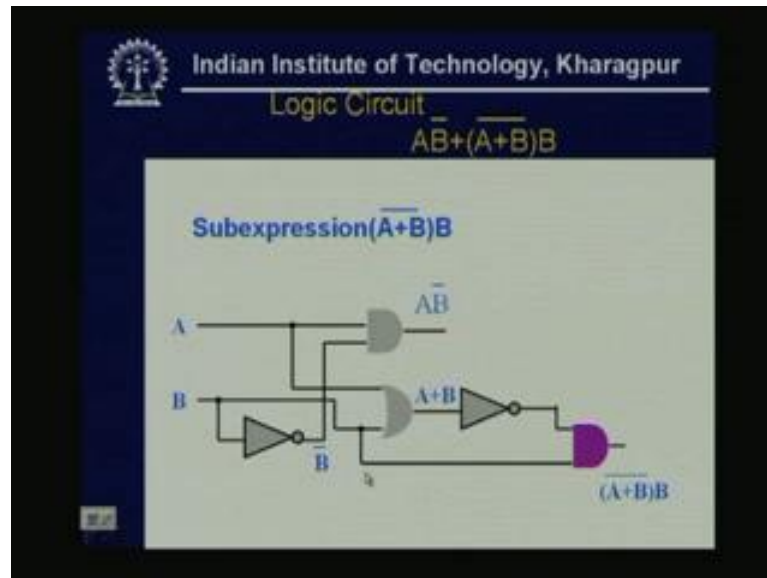
(Refer Slide Time: 53:17)



Now, A plus B complement, so simple, I will put, one NOT gate at the, as, A plus B as the input. Or, other way I can tell, A plus B is directly applied, as the input, of an, of a NOT gate or a inverter, and then output will be A plus B complement. Now again, in the

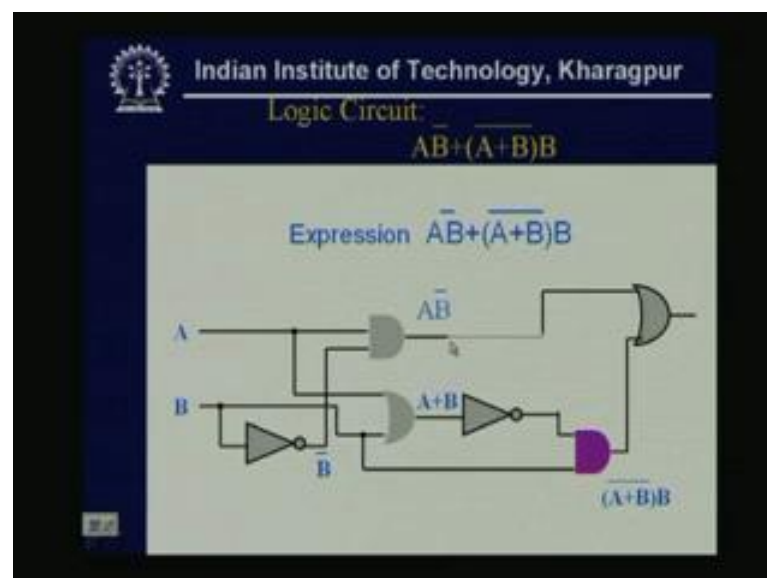
expression I need another, it should be A plus B complement to B, A plus B complement B.

(Refer Slide Time: 54:02)



So, now see, this is my A plus B complement, and this B is available, so actually, this B is going here, and this is my B, and this is a 2 input AND gate. So, A plus B complement B, so output of this, AND gate is A plus B complement B.

(Refer Slide Time: 54:26)



Now, the full expression we see, this is my A, B bar, the first part A, B bar, and the right hand, the next portion is A plus B complement B, that is the another input, these two input is are fed to 1 or, because it is a plus, plus means my OR, two input OR gate. So,

this is my, output Z, so I get my output Z, as the A, B bar, plus A plus B complement B, so in this way, we can, what we have done, if a Boolean expression is given, then we can draw a logic circuit.

That means, the a number of logic gates, with the interconnections, so that it realizes, that expressions, means that, if I give the input values or in terms of 0's and 1's, what will be the output of that expression. That means, that Boolean, what value will be evaluated, or the output value, of the expression evaluated, the logic circuit, should evaluate the same value, for the same input set. And then, we can tell, that the Boolean expression is correctly, converted to the logic circuits.

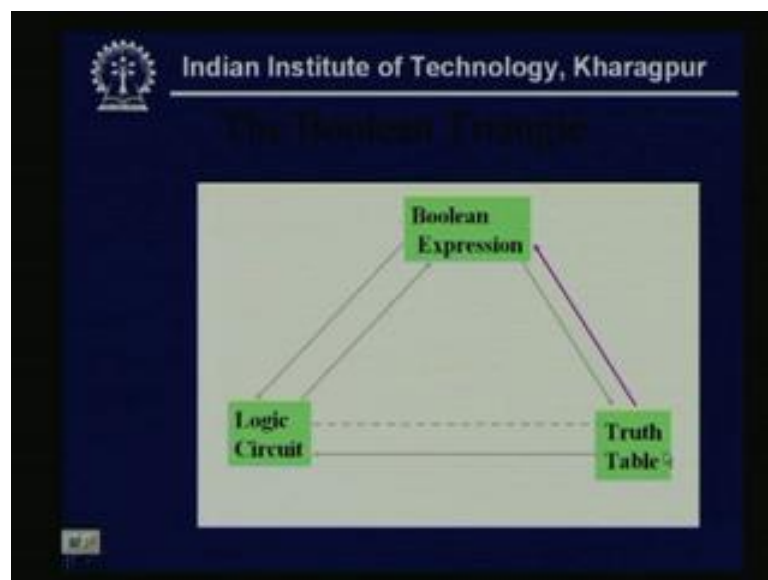
So, only we have seen, the, this portions, the, our in the triangle, the Boolean expression, to the logic circuits part two. So, next day, will be seeing the other conversions, other conversions of the circuit, Today we will end here.

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

Lecture – 06
Boolean Algebra

Today, we will start discussion on Boolean algebra.

(Refer Slide Time: 57:30)



So, last day we have discussed, that the design of digital circuits, I am just, to design that digital circuit, what are the basic needs, that we have discussed in the introductory class. So, if we recapitulate quickly, the summary of the last day's lecture, that we found that, three things, that logic circuits, truth table or Boolean expressions, that any one this thing, is necessary. Now, last day we have seen some Boolean expressions that means, given, a digital design to be built up, first we have identified, as some of the expressions.

Or the, if the truth tables, we have seen that, how the truth table, can be built up from Boolean expressions, and again from the Boolean expression the logical circuits, can also be built up. Now, from logic circuit, if the logic circuit is given, from there the Boolean expression can also be evaluated, similarly, if the logic circuit is given, the truth table

can be constructed. If the truth table is given, then the logic circuit can be, designed from the truth table, so today we will see, that how this Boolean expression, can be built up.