

Computer Organization and Architecture: A Pedagogical Aspect

Prof. Jatindra Kr. Deka

Dr. Santhosh Biswas

Dr. Arnab Sarkar

Department of Computer Science & Engineering

Indian Institute of Technology, Guwahati

Fundamental of Digital Computer

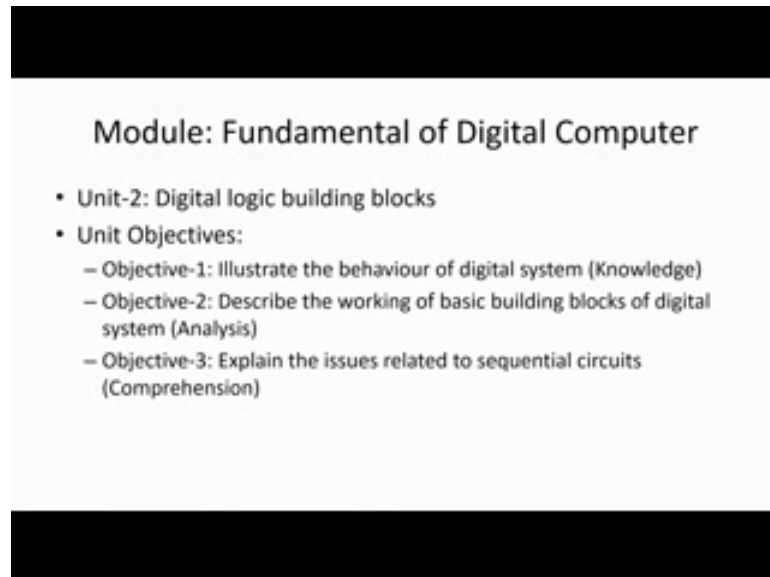
Lecture - 02

Digital Logic Building Blocks

Hello everybody welcome back to the online course on Computer Organisation and Architecture. So, we are in the first module, first module is based on your fundamental of digital computer and we are in the unit 2, Digital Logic Building Blocks. Now, we are going to see what are the things that we are going to cover in this particular unit.

So, as we mention that we are going to first mention about the objective. Let us see what are the objective that we have today.

(Refer Slide Time: 01:00)



Module: Fundamental of Digital Computer

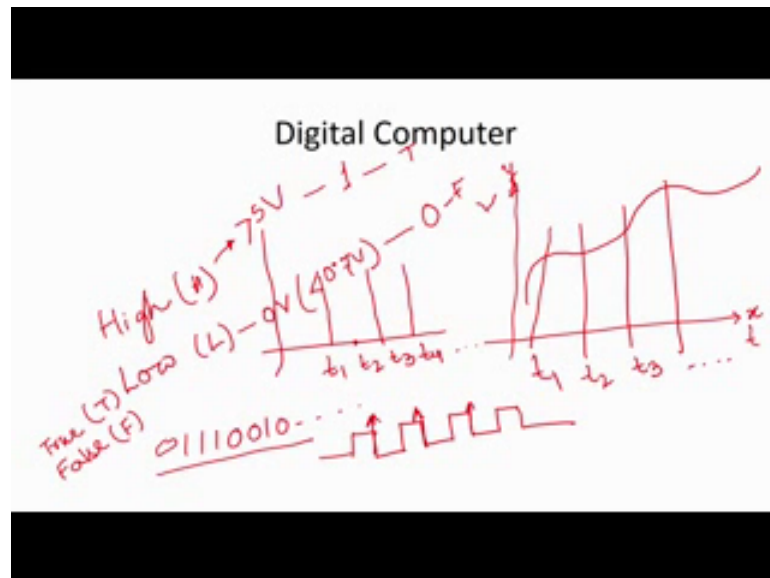
- Unit-2: Digital logic building blocks
- Unit Objectives:
 - Objective-1: Illustrate the behaviour of digital system (Knowledge)
 - Objective-2: Describe the working of basic building blocks of digital system (Analysis)
 - Objective-3: Explain the issues related to sequential circuits (Comprehension)

So, in this particular case we have mentioned 3 objective, objective 1 illustrate the behaviour of digital system. Again we are going to cover this things in knowledge level because these are the things that will be needing while going to discuss about computer organisation and architecture. But if you look into the objective. So, if you are going to look for the behaviour of digital system in another (Refer Time: 01:25), the digital logic

design in that case these objective may go into either analysis level or maybe in the design level. But here we are just going to torch up on it and we are going to give the information in knowledge level and will be using this knowledge while discussing about the fundamentals of computer organisation and architecture. Objective 2, describe the working of basic building blocks of digital system. So, here we will slightly give ideas about the analysis of the digital system. And objective 3 explain issues related to sequential circuit we will come back to this point what is sequential circuit.

Now, we are talking about digital computer or this modules is fundamental of digital computer. When we talk about digital computers necessarily another question will come to our mind that I used the summed a computers are there. So, in that case you can look into this issue and you can said that yes you can categorise the computer in 2 disc 2 broad categories, one is a digital computer and second one is analog computer here in this particular course we have going we have mainly going to emphasis on digital computer what is a digital computer and what is analog computer. So, this computers are electronic devices. So, it works on electrical signals that signals may be either voltage or current. So, in case of analog computer we work with continuous signal.

(Refer Slide Time: 03:01)



So, said it if I am going to draw a behaviour say some time and in this particular case say you are having the voltage speed in x axis we are having time and y axis say we are having voltage. So, this electrical signals first continuously then we said this is your

analog signal. So, in case of analog computer we are going to work with continuously varying signals, but in case of digital computer we are going to sample these things that signals in particular instance of time and we are going to look for those signal values only say. As per example I can have some timing instance I can say what is the signal value of time t_1 , what is the signal value of time t_2 , t_3 etcetera. So, in this way we are going to look at it; that means, we are going to work with this discrete signal. So, digital means discrete signals.

So, when you are going to talk about digital computer you are mainly you are going to concentrate on 2 level of signals either at some timing instance the signal is high or in some instances signal is low something like that. So, in t_1 signal is high, t_2 signal is low, t_3 signal is high, again t_4 signal is high like that. So, for sampling this thing basically you work with a clock signal. So, basically we have continuous running clock and we are going to sample the signals at some instance of time. So, maybe somewhere here.

And secondly, I am saying that you are going to work with these discrete signals and you have going to see the voltage values at some instance of time. So, this voltage level may be either high or low. So, in some cases we represent it by H or low. And generally when talk about high voltage basically it is depending on the technology it may be something voltage more than 5 volt in some cases it may be 8 volt in some cases it may be 12 volts and for low means it is basically 0 volt or maybe a very low voltage which is less than some 0.7 volt or something like that.

So, if voltage level falls below this particular level generally we said it this is your low signals and 1 voltage level goes above some shortage well we said it this is high level. So, generally we represent this 2 step as high and low. For our convenient that high may be represented by 1 and low will be represented by 0. So, basically it is a combinations of 0s and 1s, on there at some instance of time the signal value is high that is represented by 1 and in some cases the signal value is low in that cases it is represented by 0. So, eventually we are getting some signals value which is you are say 0 1 1 1 0 0 1 0 something like that.

So, finally, what will happen? We are coming down to some number system which is your binary number system. So, in our discussion mainly we have going to concentrate

or going to discuss with respect to it is binary number system. So, digital computer works on binary number system. In top level we can say like that, but in low level you said it will work on either some high voltage or some low voltage. So, this is the way we can look upon the digital computer.

Again this digital computer is basically related to digital logic. So, in case of logic generally we are having 2 tool values, one is called true and second one is called false. So, true is represented by t and false is represented by false. So, in some places what will happen 1 will be represented by your true and 0 will be represented by false. So, eventually all are going to have the same meaning. So, in case of digital computers we are going to work with discrete signal values of distinct signal may be high or low high values will be represented by 1 or 2 and low values will be represented by 0 or false.

Now, what is the digital circuit? So, in case of digital circuit we can say, it any digital logic function is represented by Boolean expression. So, if we write some Boolean expression that Boolean expression is going to evaluate some function and that function can be implemented with the help of your digital logic circuit way.

(Refer Slide Time: 07:41)

The slide is titled "Digital Circuits" and contains the following text:

- Boolean expressions
 - Any digital logic function is expressed by Boolean expression
- Mainly two categories of digital circuits
 - Combinational Circuits
 - Sequential Circuits

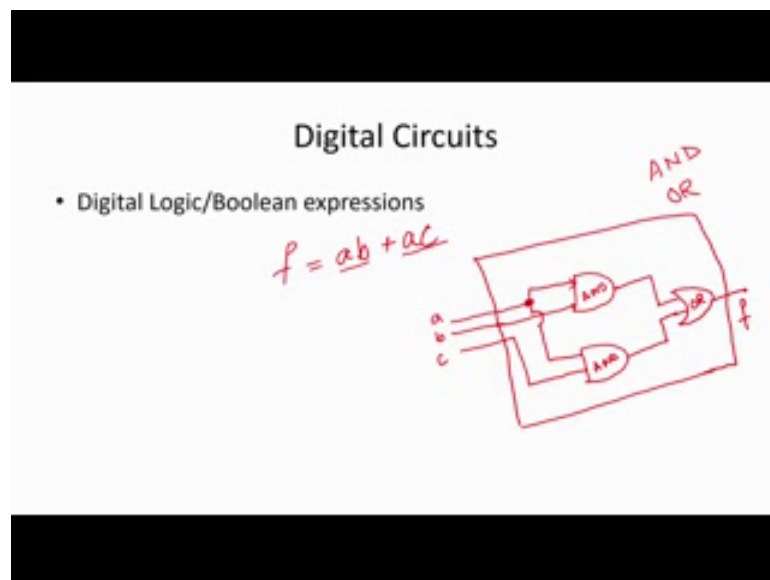
To the right of the text is a hand-drawn diagram in red ink. It shows a rectangular box labeled "ckt" (circuit). On the left side, there are three input lines labeled I_1 , I_2 , and I_n . On the right side, there are two output lines labeled "output". Below the box, there is a larger, empty rectangular box, possibly representing a component or a larger system.

Again the digital logic circuit can be categorized into 2 different categories one is your combinational circuit and second one is your sequential circuit.

So, in the particular case I will just simply said it I am going to take a digital circuit I am going to just think it as a black box I do not know or your (Refer Time: 08:11) what is inside that particular box. So, here we are going to give some input signal say I 1, I 2 to say I n and we are going to get some output signals. This output signals depends on the input signals and depending on the behaviour of the circuit we are going to get the values of those particular output signals. So, such type of circuit is known as your combinational circuit. So that means the output entirely depends on my input this is a combinational circuit.

But when we talk about the sequential circuit then what will happen some of the output will be given as a feedback and (Refer Time: 08:57) input to the circuit; that means, the current output of the circuit depends on my previous output that previous output is coming as an input to my circuit. So, in that particular case we are going to said this is the sequential circuit and output depends on the previous output also. So, we have to have some mechanism to retain this particular previous output and we will see that particular things when you are going to talk about sequential circuit.

(Refer Slide Time: 09:32)



Now, these correlation between Boolean expression and digital logic. So, say that I am simply talking about simple functions say ab plus ac. So, this is a Boolean function and a b and C are Boolean variable; that means, it can take values either true or false, on other hand I can said it true can be represented by 1 and false will be represented by 0. So, if

we are having such type of Boolean expression for every Boolean expression we are having a digital logic circuit. So, how we can implement? I can say it this is a and b it says that these values of these particular component is 1 provided both a and b are true, both a and b are 1 then the value is 1. Again this function is having 2 components a and b and a and c this is the all combination of plus combination it says that the function value will be 1 either a and b is 1 or a and c is 1, so that means we are going to talk about 2 operations 1 is your AND, and second one is your OR. For such type of operation we are having digital logic gates.








So, in this particular case what I can say that, I can say that I am having an AND combination 2 and combination and I am having 3 inputs a , b , c . So, it is basically a and b is ending together and a and c is ending here. So, 1 volt a and b and a and c are true when what will happen, the output of these 2 is going to be high or true. Now, main final function is the OR of these 2 combinations, so we are going to connect 1 OR gate over here and finally, this is the functional value f .



So, these are the AND gates, 2 AND gates and 1 OR gate. So, this is a circuit and these circuit implements this particular Boolean expression. Now, we can say that now, I already mention that I can feel these particulars internal details as a black box and whenever I am going to give the input over here depending on the input we are going to get the output. So, this is the (Refer Time: 11:57) see what are the different types of logic gates we have.

(Refer Slide Time: 12:01)

Combinational Circuits

Logic Gates

Name	NOT	AND	NAND	OR	NOR	XOR	XNOR																																																																																																
Alg. Expr.	\bar{a}	$a \cdot b$	$\overline{a \cdot b}$	$a + b$	$\overline{a + b}$	$a \oplus b$	$\overline{a \oplus b}$																																																																																																
Symbol																																																																																																							
Truth Table	<table border="1"><tr><td>a</td><td>x</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	a	x	0	1	1	0	<table border="1"><tr><td>a</td><td>b</td><td>x</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	x	0	0	0	0	1	0	1	0	0	1	1	1	<table border="1"><tr><td>a</td><td>b</td><td>x</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	x	0	0	1	0	1	1	1	0	1	1	1	0	<table border="1"><tr><td>a</td><td>b</td><td>x</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	x	0	0	1	0	1	0	1	0	0	1	1	1	<table border="1"><tr><td>a</td><td>b</td><td>x</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	x	0	0	1	0	1	1	1	0	0	1	1	0	<table border="1"><tr><td>a</td><td>b</td><td>x</td></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	x	0	0	0	0	1	1	1	0	1	1	1	0	<table border="1"><tr><td>a</td><td>b</td><td>x</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	x	0	0	1	0	1	0	1	0	0	1	1	1
a	x																																																																																																						
0	1																																																																																																						
1	0																																																																																																						
a	b	x																																																																																																					
0	0	0																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					
a	b	x																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
a	b	x																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					
a	b	x																																																																																																					
0	0	1																																																																																																					
0	1	1																																																																																																					
1	0	0																																																																																																					
1	1	0																																																																																																					
a	b	x																																																																																																					
0	0	0																																																																																																					
0	1	1																																																																																																					
1	0	1																																																																																																					
1	1	0																																																																																																					
a	b	x																																																																																																					
0	0	1																																																																																																					
0	1	0																																																																																																					
1	0	0																																																																																																					
1	1	1																																																																																																					

  $(a \cdot b) \cdot c = a \cdot (b \cdot c)$

So, logic gate basically we are having some standard gates and here in this particular pose we are going to see in the gate level on, how those particular gates are implemented you are not going to look for it. That implementation issues will be discussed in some other courses may be in your digital circuit design. So, here first gate is your NOT gate. What is that NOT? It says that if my input is high then output is low and output is input is low output is high. So, a is your input and x is your output. So, when a is 0 x is 1 when a is 1 x is 0.

Second gate is your AND gate. So, in AND gate already I mentioned that if the both the inputs are high then output is high. So, this behaviour is represented with the help of a table operation and these table is known as your truth table.

So, basically the behaviour of digital circuit can be represented with the help of truth table. So, here I am in 2 input a and b. So, when both the inputs are high then output is high otherwise output is a 0. So, you just see this combination when both are high then output is high in other cases output is 0.

Similarly, we are having a NAND gate. NAND gate is nothing, but and an invert. So, first we have going to ending the operation this is AND then these bubble is going talk about the invert it. So, you are going to say this is the NAND gate. So, since NAND is the invert of your AND. So, you just see the truth table the output is exactly inverse of this particular output of AND gate, when it is 0 than output is 1 and when output of AND

gate is 1 than output of NAND gate is 0. Like that we are having OR gate, in case of OR gate either a or b if any one of the signal is high than output is high otherwise it is 0. So, you just see that if both the input has 0 than output is 0, but if any one of this input is high than output is high. Similarly we have going to get an NOR gate, NOR is nothing but OR and invert. So, first we having the OR combination than this bubble again represent the invert, you just see the or is inverted over here when output of or is 1 than output of NOR is 0. So, nor is invert of these.

Like that we are going to have another function called XOR it is called exclusive OR. So, in case of or if anyone is high then we are going to said that output is high. But in case of XOR we said at output is high provided either or than is high. So, in that particular case if both are high in that particular case output is 0. So, either or than a and b is high than output is high. So, similarly XNOR is nothing, but the invert of XOR. So, exclusive or an exclusive nor. So, you just see the functional value just invert of this particular XOR gate.

Now, you just see that here I am just mentioning some of the gates like that we are having other gates also. So, in that particular case that not it is an unary gate we are having only 1 input and depending on that we are having the output, but other gates are you binary gates because we are having 2 inputs depending on this input we are going to have output. So, now, you just see that we are having 2 inputs.

Now, here I have such some of their example and how many such type of gates you can have if you are going to take 2 inputs. So, it is basically you can have several combination and if you look into all those combinations say these are the 4 different possibilities that we are having either both are 0 0 0 1 1 0 and 1 1 these are the only possible combination for 2 input gates. So, we are having 4 possible input combination and for these 4 possible in input combination what you can said that either all outputs are 0 all outputs are 1 or maybe some of them are 1 and some of them are 0. So, in that particular case you can said that we are going to get total 16 different combination; that means, 16 different function or binary function can be implemented if we are having 2 inputs.

Now secondly, this gates can be extended 4 more inputs you can say if I am going to look for a gate where I am having 3 input over here set 3 input AND gate. So, these gate

can be physiology like that first I am going to have and combination of first 2 and secondly, the whatever output you are getting that will be combined with the third input. So, like that we are going to get the effect of 3 input AND gates or may be 3 input AND gates can be implemented with the help of electronic circuit. So, this is possible because this particular operations are associated; that means, $a \cdot b \cdot c$ is equals to $a \cdot b \cdot c$. So, since it is associating, so subset of expansion is always possible.

(Refer Slide Time: 17:25)

Logic Gates

Name	NOT	AND	NAND	OR	NOR	XOR	XNOR																																																												
Alg. Exps.	\bar{A}	$A \cdot B$	$\overline{A \cdot B}$	$A + B$	$\overline{A + B}$	$A \oplus B$	$\overline{A \oplus B}$																																																												
Symbol																																																																			
Truth Table	<table border="1"> <tr><td>A</td><td>0</td><td>1</td></tr> <tr><td>\bar{A}</td><td>1</td><td>0</td></tr> </table>	A	0	1	\bar{A}	1	0	<table border="1"> <tr><td>A</td><td>0</td><td>1</td></tr> <tr><td>B</td><td>0</td><td>1</td></tr> <tr><td>$A \cdot B$</td><td>0</td><td>1</td></tr> </table>	A	0	1	B	0	1	$A \cdot B$	0	1	<table border="1"> <tr><td>A</td><td>0</td><td>1</td></tr> <tr><td>B</td><td>0</td><td>1</td></tr> <tr><td>$\overline{A \cdot B}$</td><td>1</td><td>0</td></tr> </table>	A	0	1	B	0	1	$\overline{A \cdot B}$	1	0	<table border="1"> <tr><td>A</td><td>0</td><td>1</td></tr> <tr><td>B</td><td>0</td><td>1</td></tr> <tr><td>$A + B$</td><td>0</td><td>1</td></tr> </table>	A	0	1	B	0	1	$A + B$	0	1	<table border="1"> <tr><td>A</td><td>0</td><td>1</td></tr> <tr><td>B</td><td>0</td><td>1</td></tr> <tr><td>$\overline{A + B}$</td><td>1</td><td>0</td></tr> </table>	A	0	1	B	0	1	$\overline{A + B}$	1	0	<table border="1"> <tr><td>A</td><td>0</td><td>1</td></tr> <tr><td>B</td><td>0</td><td>1</td></tr> <tr><td>$A \oplus B$</td><td>0</td><td>1</td></tr> </table>	A	0	1	B	0	1	$A \oplus B$	0	1	<table border="1"> <tr><td>A</td><td>0</td><td>1</td></tr> <tr><td>B</td><td>0</td><td>1</td></tr> <tr><td>$\overline{A \oplus B}$</td><td>1</td><td>0</td></tr> </table>	A	0	1	B	0	1	$\overline{A \oplus B}$	1	0
A	0	1																																																																	
\bar{A}	1	0																																																																	
A	0	1																																																																	
B	0	1																																																																	
$A \cdot B$	0	1																																																																	
A	0	1																																																																	
B	0	1																																																																	
$\overline{A \cdot B}$	1	0																																																																	
A	0	1																																																																	
B	0	1																																																																	
$A + B$	0	1																																																																	
A	0	1																																																																	
B	0	1																																																																	
$\overline{A + B}$	1	0																																																																	
A	0	1																																																																	
B	0	1																																																																	
$A \oplus B$	0	1																																																																	
A	0	1																																																																	
B	0	1																																																																	
$\overline{A \oplus B}$	1	0																																																																	

Complete Set of Operators: NOT, AND, OR

Universal Gates: NAND/NOR

$\bar{B} \cdot A + B \cdot \bar{A}$

Now, we are talking about this particular logic gate here we are having 2 terms call complete set of operators and I think you may be knowing about this things so that complete set of operators audio nothing, but AND, OR and NOT gate. If we have this 3 gates than, all others can be all other gates can be implemented with the help of these 3 gates. That is a simple example you just see that if I am going to talk about this particular XOR gate. What is the behaviour? If 0 and 1 output is 1 and 1 and 0 output is 1 so that means what I can say that it is basically nothing, but b inverse a . So, this is b is 0, a is 1 at that particular point the output value is 1 similarly either b or a inverse. So, this is nothing, but this expression is going to give may this particular XOR combination. Now, just see if we do not have this particular gate than what will happen with the help of NOT, AND, and OR gate I can have the effect of this particular gates.

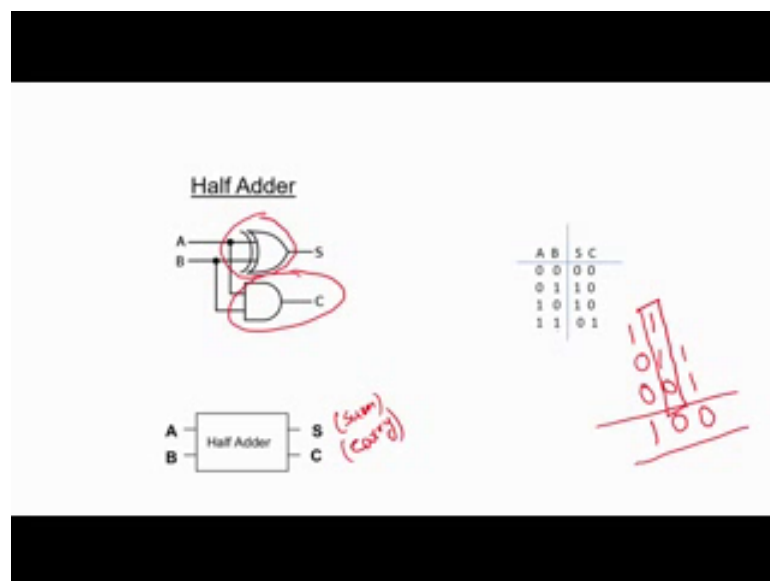
So, this is b inverse, a inverse we are using this particular NOT gate then b inverse not a ; that means, you are using this particular AND gate we having 2 and combination and

there. Now, combined with this particular OR gate. So, you just see with NOT, AND, and OR all circuits can be implemented so that is why you are going to say that this is the complete set of operators.

Along with that we are having another times call universal gates. So, NAND and nor as treated as universal gates Why you said these are the universal gates? Any digital logic circuit can be implemented with the help of only NAND gate or only NOR gate. So, this is the things that we are having that is why you said these are the universal gate. Now, as an assignment you just think how to be implement these particular function with the help of only AND gate, NOR gate or NAND gate just take as a assignment and just see how the circuit can be implemented with the help of only NAND gate or with the help of NOR gate.

Now, we are going to see some of the building blocks that will be used while constructing our computers. So, one thing is first circuit I am going to talk about here is your half adder this is basically adding of 2 numbers and these are the addition of binary numbers because we can represent only 0 and 1 that is all, high voltage means 1 and low voltage means 0. In that particular case that behaviour of half adder can be represented with the help of this particular truth table, we having 2 input a and b, and 2 output sum bit and carry bit as represent for sum and C represents for carry.

(Refer Slide Time: 20:14)



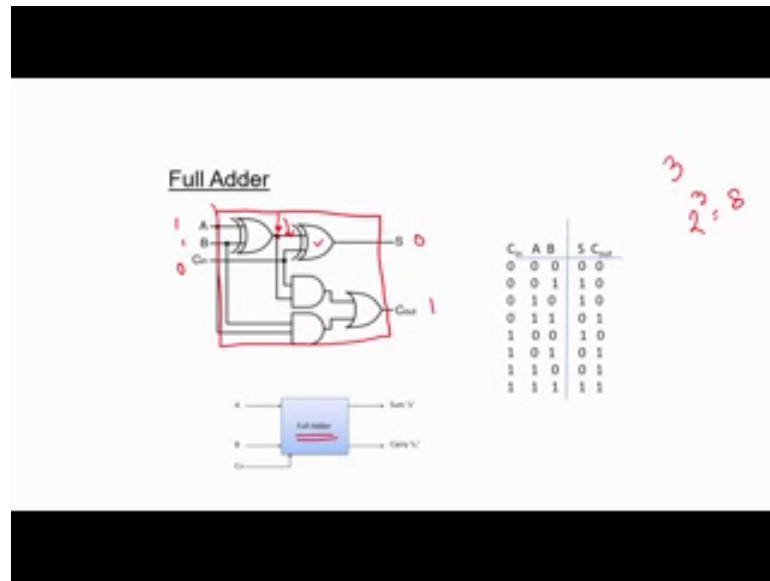
So, this is basically when both a and b are 0 then my sum is 0, when a and b is 1 then sum is 1, when a is 1 b is 0 then sum is 1 and when a is 1 and b is 1 I am going to get sum is 0, but it is going to generate carry call 1. So, this behaviour for S we need 1 circuit for C we need another circuit.

Just look into the behaviour of S, here we are going to set up this is nothing, but the exclusive or so that means this exclusive OR gate is going to give me the behaviour of some function and if you see that C it is nothing but AND combination and both are 1 then output is 1. So, in that particular case this AND gate is going to give me this particular carry function.

Now, here I am talking about a half adder when we talk about half adder necessarily that something will come to your mind; that means, there may be some other adder also. Let us C this is a adder circuits which can add a single bit numbers know just consider about some 3 bit numbers then what will happen, we are going to add it 1 and 1 is 0 and it is going to generate carry 1, then 1 and 1 we are going add together then output is 0 and it is going to generate 1 carry. Now, 1 0 and 0 is going to give you 1, so this is basically the result is your 1 0 0.

Now, possibilities fines you just say that I am going to give a and b I can get the sum bit along with that I am going to get that carry bit. But when I am coming to the next bit now, what happened you just say that I have to take decision the functional value output of this depends on 3 inputs; that means, you have to take care of both the input a and b along with that the carry that is coming out from my previous bit. So, for that we are having another circuit we call this is the full adder.

(Refer Slide Time: 22:28)



In case of full adder, now you just said it we are having 3 input a b and carry in the carry in is coming from the previous bit position. So, it is coming from here and we are going to get this carry in and when we get them together it will generate a career like this one. So, we are having this particular carry out. So, the behaviour of this circuit can be represented with the help of this particular truth table.

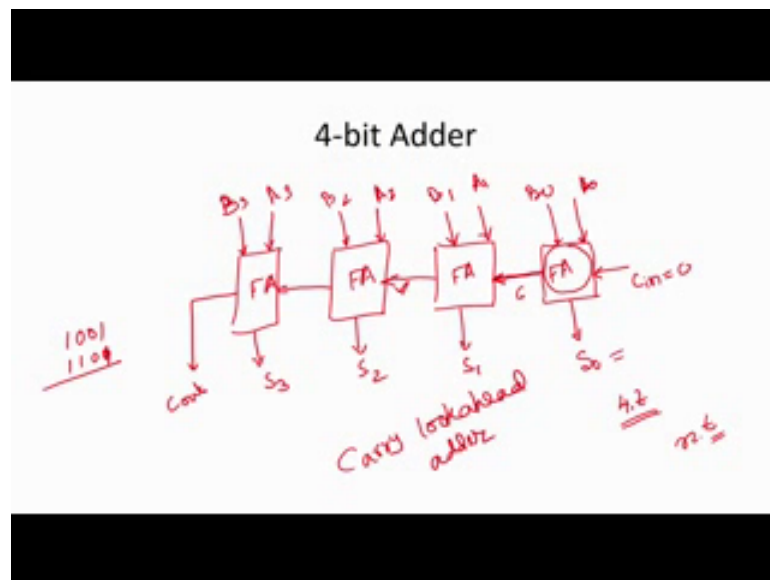
Now, since we are having 3 inputs over here. So, total input combination will be your 2 to the power of 3 which is your 8. So, this is the maximum number of combination that we have. So, we have started all this possible combination over here and depending on the behaviour of addition we are going to find out what will be the my output function. Now, once we have this particular output function then we can implement with the help of this particular circuit. Now, you just said it, now this complete circuit will be treated as my full adder and finally, once you give the signal over here say 1 1 0 finally, we are going to get the output ratio 1 and 1 sum is 0 carry is 1.

Now, we have to see one more things. Now, said these are some logic gates or electronic component when you put signal 1 and 1 immediately we are not going to get the result over here because this electronic components are having some delays, it need some times and that time whatever time it is required we say this is the propagation delay of that particular block, here we are simply using an exclusive OR gate. So, this XOR gate will take some time to give me the finals table output over here. So, once you get a stable

output then only we can work with this particular second gate because 0 is coming scene is coming immediately as soon as I am giving the input, but the second input to this particular XOR will come after some unit of time only which depends on the propagation delay of this particular gate and this second gate will also have some propagation delay so finally, we are going to get a correct result. So, this time we have to consider and we are going to say that the final output we are going to get after some amount of time once you give the stable input and this time depends on the propagation delay of the components of f that gate so that means to get the result of a full adder is going to take some time we have to consider that particular time also.

So, this is the internal circuit. So, as a block represent you can considered this is full adder I am having 2 input a b along with that we having that carry and it is going to give me 2 output sum bit and carry bit. So, this is the block diagram, but what is their inside these particular full adder if look into we are going to have this particular circuit only. Now, we know or we have seen the behaviour of half adder and full adder. Now, if you are going to construct the 4 bit adder how we are going to construct it; that means, we need to at 4 bits so that means we need 4 adder or here we can use the full adder events.

(Refer Slide Time: 25:51)



So, what I can say that this is my 4 full adder, so here I am having input A 0 B 0, A 1 B 1, A 2 B 2, A 3 B 3. Now, what will happen this first full adder is going to give me the some bit and along with that it is going to be give me 1 carry out also that carry out will go as

an out an carry into the next full adder. So, this is your S 1 and it will go that carry out will go as an carry in this is your S 2 this is your S 3 and along with that I am going to get 1 carry out also. Since least significance which not having any carry inputs so that carry input with means set to 0 on the other hand this full adder can be replace by half adder also. Now, this is the 4 bit adder. Like that if I need an 8 bit adders I can cascade 8 full adder I am going to get the circuit.

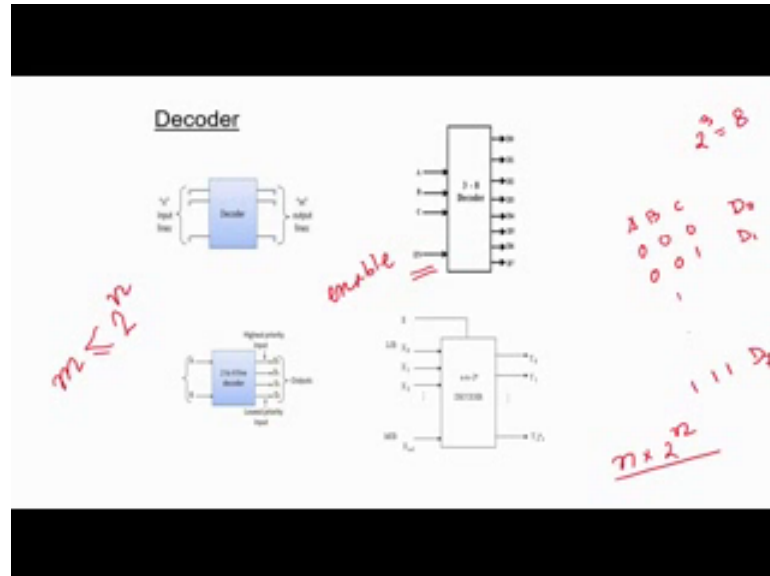
Now, you just see I am talking about the propagation delay. Now, these full adder is having some propagation delays. Once I am giving this A 0 B 0 C in it will check some time and you are going to get the output over here and along with that these C B 2 is going together. Now, I am giving this 4 bit number say 1 0 0 1 and 1 1 0 0 in that particular case I can say that second full adder cannot perform a job immediately as soon as I am giving A 1 B 1 it is a it needs to a wait for the stable output in this particular carry out of the first full adder.

Once you have going to get the stable output than only this particular second full adder can perform itself, like that second full adder is going to give me some output the third full adder need to wait for that particular stable output. So, in that particular cases you just see that if the propagation delay of the first adder is some time t than total time that required to get correct output of this full adder maybe your 4 times of t because second full adder is going to give me the correct result after tasks after t unit of time. Second full adder is going to give me a stable output after t^2 type. So, like that it t is taking the $4t$ time. Now, we like that if you are going for n b data; that means, the profile propagation time will be your nt , where t is the propagation delay of 1 full adder.

So, you just said it. Now, it is we are talking about say 32 bit computers 64 bit computers; that means, it will take lot of time to give me the output. So, for that the simple circuit we are not going to use for that we will use another circuit modification of this particular adder circuit which is known as your carry look ahead adder. So, here we are not going to discuss about this things as an information I am giving you that we are having another circuit that where it is known as your carry look ahead adder and carries will be generated at the same time and simultaneously it will be propagated carry generation and carry propagation. So, carry look ahead adders. So, with the help of these carry look ahead adders we can reduce this particular propagation delay of the adders. So, this is some bit of information I am giving you so that are if you are interested then

you can just look for some any book on digital logic where adder has been discussed thoroughly.

(Refer Slide Time: 30:10)

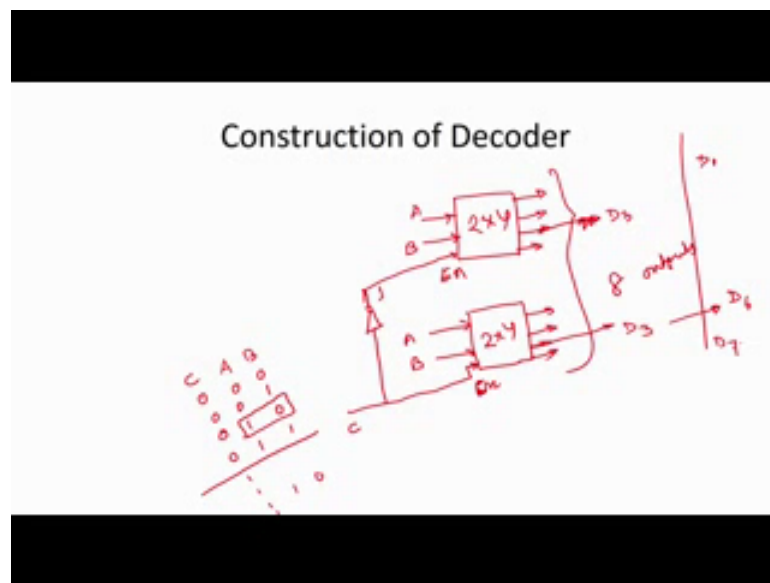


Now, the second circuit is your decoder you are having another building block called decoder. Here we are having n input lines and m input output lines and the m whatever, m output lines we are having this is basically it is less than equal to 2 to the power n. So, if we are having an input lines maximum output lines we are going to get as your 2 to the power n. And what is this circuit behaviour? Depending on the input combination only one of the output line is high. So, here you just said it we are having 3 by 8 decoders we having 3 input lines and we are having 8 output lines D 0 to D 7. So, this 3 input lines A B C. Now, what will happen since we are having 3 input lines. So, what will be the total number of input combination this is your 2 to the power 3 which is equal to 8. Depending on one combination generally one of the output line is high. So, basically what will happen when both are all are 0 then D 0 is high when that one signal is high that C is high in that particular case I can say that D 1 is high like that when all combination is 1 than you said it this is your D 7 is high

So, depending on the input combination only one of the output signal is high. So, this decoder we are going to use for selecting some of our requirements or some of our function. Say if we are having some sizes that that particular instance of time we have to select only 1 in that particular case you are going to use this particular decoder. So, this is

a 3 by 8 decoders, like that we are going to get n by 2 to the power n decoder where n is the input line and 2 to the power n is the output lines when this particular decoder. In most of the cases we are having n signal called enable signals n is nothing, but enable; that means, when this enable signal is high then only it is going to behave like a decoder otherwise it is not going to work may be all output will be 0. So, this is with enable like we are going to have a decoder.

(Refer Slide Time: 32:39)



Now, this enabled and help us to construct some of the decoder say if you are having a decoder say I am having (Refer Time: 32:43) 2 by 4 decoders and I say that I am having it is a line 2 input lines A and B, but I need set 3 by 8 decoders then what I can do, I can use to 2 by 4 decoders. So, these are the output lines 4 output lines we are having and these are the 4 output lines that we are having for the second decoder. So, total 8 output lines will get.

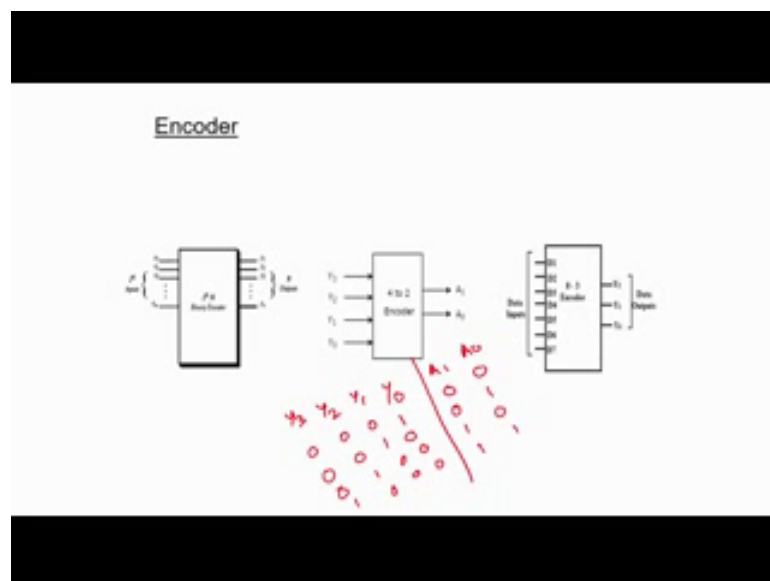
Now, here in both the cases I am going to give A 0, A and B, and A and B. Now, depending on this particular A and B say. Now, A and B having 2 combination only 4 combination 0 0 0 1 1 0 and 1 1 when input same I am giving same 1 and 0 than what will happen this particular line of say D 3 of this particular decoder and D 3 of this particular decoder will be selected because this is the combination. Now, we have to select 1 or these 2 outputs. So, for that what will happen? We need another input, so I am having this particular C input than what we are going to do we use an inverted and to this

things above here this is the enable line and this is the enable line over here. Now, we just see that when I am having C when C equal to 0 then what will happen we will see that we are going to this is C equal to 0. So, enable line is 0 over here. So, you are not selecting this one since C equal to 0. So, here I am going to get 1. So, in the particular case we are selecting this particular decoder; that means, you are selecting this particular decoder output line D 7 and all this things will be 0.

Now, similarly when C is equal to 1 again I am going to get 4 combination. So, C 1 0 then what will happen in that particular case since C is equal to 1 here I am going to get 0. So, you are not selecting this particular of 4 lines and selecting only that particular D 3 and it is nothing, but. Now, we can say that these D 3 is going to act as my D 6 because we are having from D 0 to D 7. So, like that we can use a decoder or say you can use multiple decoder to construct a desired decoder and it is possible due to this particular enabled lines.

Now, we are having another circuit call encoder. So, encoder is the reverse of decoder in case of decoder we are going to decode from n line to 2 to the power n line in case of encoder we are coming from 2 to the power n lines to n lines. So, total we are having 2 to n input lines and we are having n output lines. So, this is a 4 by 2 encoder.

(Refer Slide Time: 35:38)



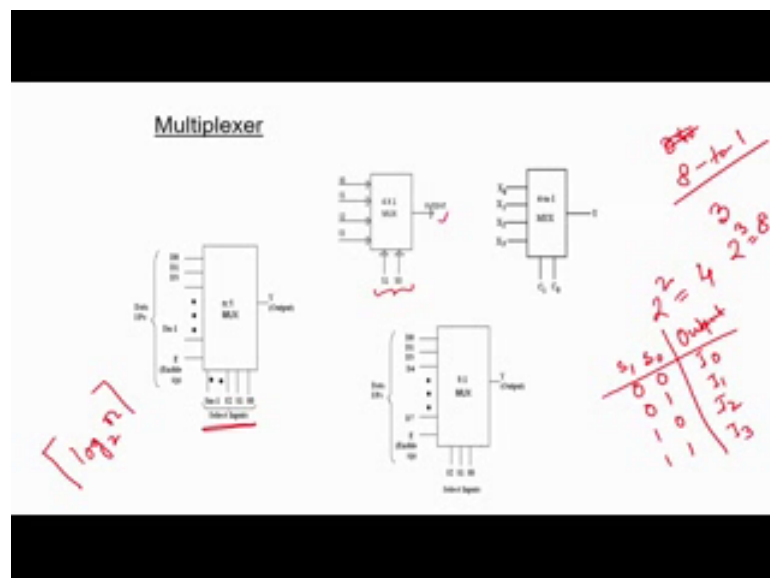
So, depending on this scenario, now you are going say either A 0 or A 1 will be high. Now, what we can do say when I am going to set this Y 0, Y 1, Y 2, Y 3. Now, at any

point of time say any one of the input line is high. So, this is basically say I am having this particular pattern or maybe this is.

So, these are the 4 inputs then what basically I can said it this is your. So, these input says that I am having 4 inputs. Now, depending on that I am going to encode it and in that particular case I can say that this one Y 0 is 0 will said it is going to represent 0. So, both A 1 and A 0 will be 0 when Y 1 line is high than in that particular case we will said it is going to represent 1. So, this will be your 0 1 when Y 2 line is 1 then it may be represented by 2 and when y 3 line is high it is represented by 3 binary 3. So, this is the way we can encode it. So, encode is reverse of decoding circuitry. So, here we are having 2^n to the power n input lines and we can get n output lines.

So, here what happened was just discussing those things in block level how to implement it these are very simple combinational logic circuit. You can take any book on digital logic and you can see the how it is done and this is very simple circuit I am telling you with the help of AND gate it can be done. So, this is encoder. Like that another building block we have which is known as your multiplexers. So, it is basically going to multiplex or masking some of the input signals. So, it is basically like that we are having several signal at any point of time you are going to take only 1 signal out of this particular and possible signals. So, here it is something like that if we are having n input signal we are going to put it to 1 output signal. So, it is called n by 1 mux.

(Refer Slide Time: 37:52)



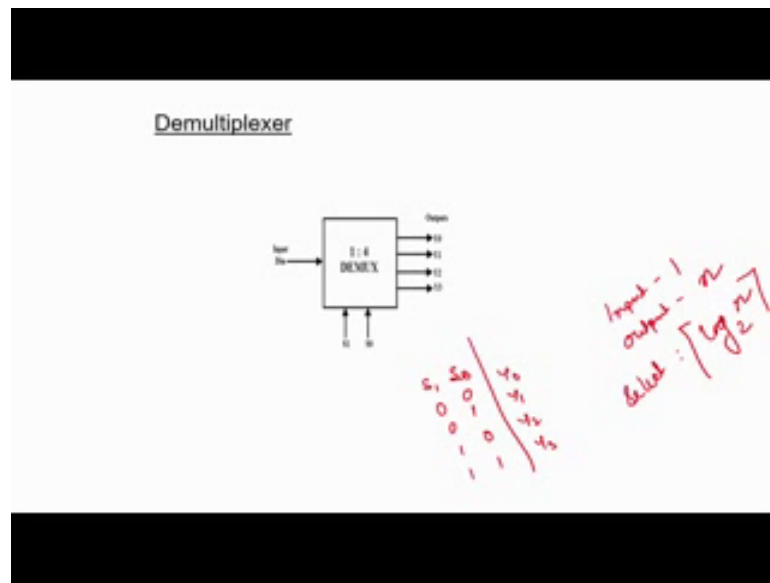
Now, how you are going to select it for that we need some select line or control lines that control lines depends on this particular number of input lines. If we are having n input lines then number of select line will be your $\log_2 n$ scaling outlay. Now, will see this, this things will small example. So, I am talking about a 4 by 1 multiplexer. Now, we are having 4 input lines I_0, I_1, I_2 and I_3 depending on my requirement we are going to take only one of this particular input signals and put it into the output signals which is known as your masking the other signal and I am going to put the in 1 particular input lines to the output line. To select this particular 4 input what happened we need some select line and control signals. So, here we are having 2 select line S_0 and S_1 .

So, you know that 2^2 is equal to 4; that means, depending on the combination of this particular 2 select line we are going to select one of this particular input lines. So, basically I can said it this is a S_0 and S_1 . So, what is the combination of those particular select line? That maybe 0 0 0 1 1 0 and 1 1. Now, what will happen depending on this particular select line what will be the my output we are going to select one of this particular input signal.

So, say that when S_0 and S_1 is 0 then I can say that we are going to select that input line I_0 and it will be transfer to the output side when it is your 0 1 then I_1 then 1 0 I_2 and 1 1 is your I_3 . So, this is the behaviour of multiplexer. From n possible input signal we are going to select only 1 out of those n signals so that is why what is the input output and select line relationship we are having n input lines, we are having 1 output lines to transfer any one of this particular input lines to the output lines we need select signal of control signal what is the number of control signal it is your $\log_2 n$ signal. If we are having 8 2 1 marks then what will happen you need 3 select line because 2^3 is equal to 8. So, this 3 select line will have combination form all 0s to all 1 8 possible combination.

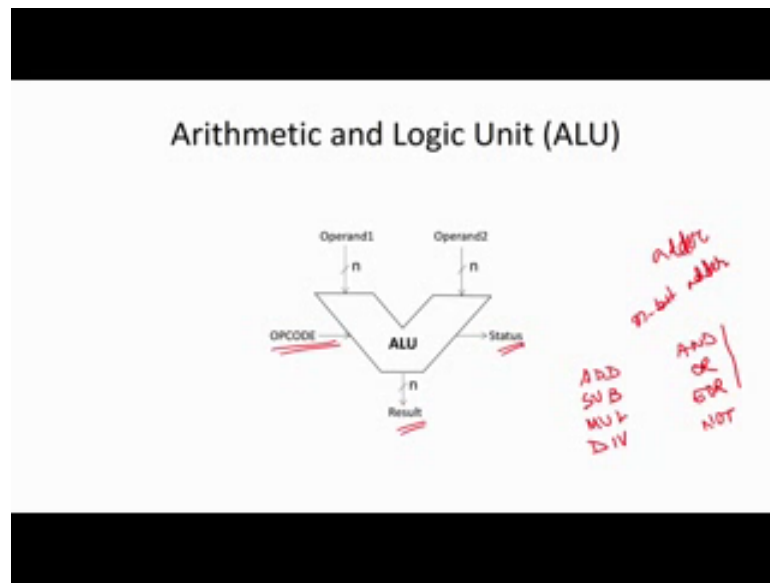
So, we are going to select any one of this particular input lines and I am going to put in to the output. So, this is a multiplexer circuit again it is a combinational circuit and very simple circuit.

(Refer Slide Time: 40:41)



Another one we are having the demultiplexer which is the reverse of your multiplexer. So, here we are having 1 input line and we are going to transfer it to any one of those particular output line. So, if we are again I can set up input line is 1 output line and then what will how many select line we have again this is your $\log_2 \log n$ to the base 2 signals. So, we are having 1 input lines. Now, we are having to select lines depending on those particular select line we are on going to transfer this input line to any one of those particular output lines. So, again I can say that if it is your S 0 and S 1 and that we are having a 4 combination and depending on that what is the output line. So, here input will transfer to Y 0, Y 1, Y 2 and Y 3. So, we are having 1 input signals and. Now, that will be transferred to any one of this particular output lines. So, this is demultiplexer. So, in computer we are extensively going to use those particular building blocks to construct of computer.

(Refer Slide Time: 41:53)



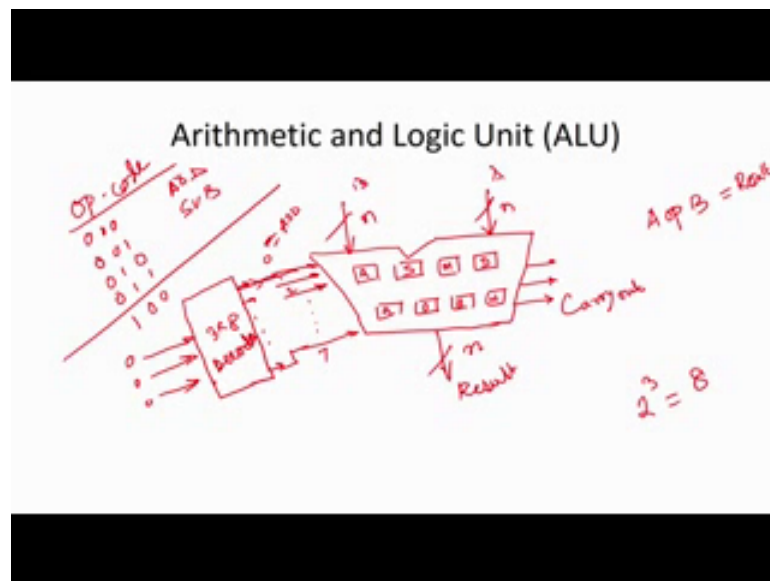
Another unit we are having called arithmetic and logic unit ALU. This is the basic processing element inside of computer which can perform some arithmetic operation and logic operation. So, this is your block diagram, we are not going to see what is there inside this particular ALU just say that we are having some circuit like that we having a adder circuit. If it is an n bit ALU; that means, both the inputs are of size n bits and result is also n bits; that means, we are having a n bit header. Already we have seen how to construct an n bit data.

Similarly what will happen we may have several operation over here. So, I consider that I am having say addition operation, I can have subtraction operation, I can have multiplication operation and say I am having division operation. So, I am having 4 processing element which can popular operation addition, subtraction, multiplication and division. So, these are the say 4 arithmetic operation. Along with that we may have some logical operation also logic operation also, I can say that I may have that AND operation, OR operation or maybe say XOR operation or maybe I can say another one say NOT operation. So, in this particular logic operation AND, OR, XOR are your binary operation, but not is an unary operation because it is going to inverse one of the input and going to get the output as the input of this particular input.

Now, you just see that like that this is a processing element and I am having 4, addition 4 arithmetic processing element and 4 logic operation at any point of time you are going to

perform one of these particular operation. Now, what operation we are going to perform that will be given by this particular signals. So, depending on the signals we are going to get the result and along with that we are going to get some status also, some of the status with will be given or some more additional counter signals we are going to get and that will be set or reset according to the behaviour of this particular circuit. So, this ALU is the basic building blocks or basic processing element that we are going to use in our computer to perform some of arithmetic operation as well as some logic operation.

(Refer Slide Time: 44:30)



So, now, just say that how we are going to construct it. Now, if say that this is the ALU I am having these 2 input and input n bit input and, this is basically I can say that this is A, this is B and here I am going to get n bit result and say that I am having those particular processing element these are the 4 addition, subtraction, multiplication and division similarly 4 logic operation that say n, OR, XOR and say NOT.

Now, that any point of time we are going to give 2 inputs over here A and B, and we are going to perform 1 operation and depending on the operation we are going to get our result. Now, how we are going to select this particular operation? Now, we are talking about the previous set we have seen that we are going to have the output and here we are having some status line one of the status line maybe I can say about carry out like that we can have several status line. Now, select any one of this particular input operation we need the appropriate signals, for that what will happen I can use 8 different signal. So,

this is your 0 1 2 3 like that up to 7. So, you can say it when I am giving 0 signal as high than you can say that this is going to perform the operation addition. So, in that particular case what will happen? These addition circuit will be selected and both the input will be diverted the adder circuit and we are going to get the result; that means, we need a control signal to select any one of the (Refer Time: 46:30).

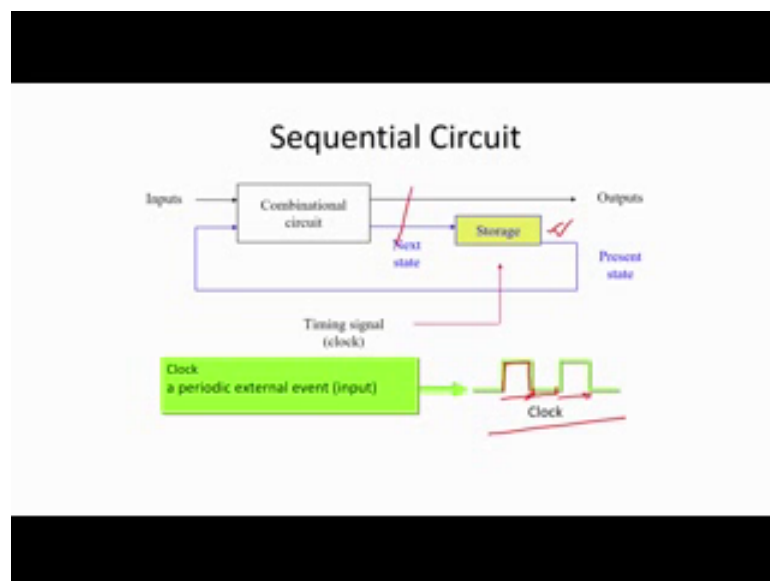
But you can reduce the in signal over here instruct 8 signal what will happen you can control it with the help 3 signals only because you know that 2 to the power of 3 equals to 8, if I am going to use 3 signals than we can generate this 8 possible combination because this 3 comb signals are going to have 8 different possible combination. So, in that particular case what we can do you are going to use at 3 by 8 decoder; that means, we are having 3 inputs and we are having 8 output lines and those output lines connected to those particular select signals of this ALU.

Now, when I am giving say all 0 then what will happen? This is A 0 then this signal will be high; that means, you are going to see the adder circuit. So, this is the way I have going to have 3 bit code and this 3 bit codes is known as my half code which is your operation code if it is your 0 I can say it this is your add when it is your 0 0 1 I can say this is your subtract like that, so 0 1 0 and 0 1 one. So, these are the arithmetic operation. So, when this third bit I am going to put as 1, then I can say it I am going to those particular logic operation. So, now, say since with the help of 3 bit I can having 8 different combinations you can use an 3 by 8 decoder to get select the appropriate operation. So, this is the way you are going use our arithmetic and logic unit in our computer and what operation we are going to perform that will given by the half code which is a binary code depending on the combination of the input signal we are going to see the one of the operation and this ALU is going to perform that particular operation. So, we are going to use and ALU in our computer. So, you can have 8 bit ALU which can perform operation of 8 bit numbers we may have 16 bit ALU which can perform operational 16 bit numbers.

Now, I have already mention that we are having 2 type of digital logic circuit, one is your combinational circuit another one is sequential circuit I have several briefly give idea about the combinational circuit which you are going to use while constructing the digital computer. But we may have a many more other circuitry also, but in this particular case we are not going to discuss, but just giving some idea. Now, whatever circuit you are

going to have I think you can analyze it you are going to feel get an feeling how that particular circuit will be constructed or implemented. Now, second class of circuit is your sequential circuit. So, in case of sequential circuit already I have mention that the current output depends on some previous output also so that is why it is a combination of your combinational circuit as well as some storage. So, this is a combinational part of its sequential circuit depending on input you have going to get the output and some of the output is going to act as an input for the next time.

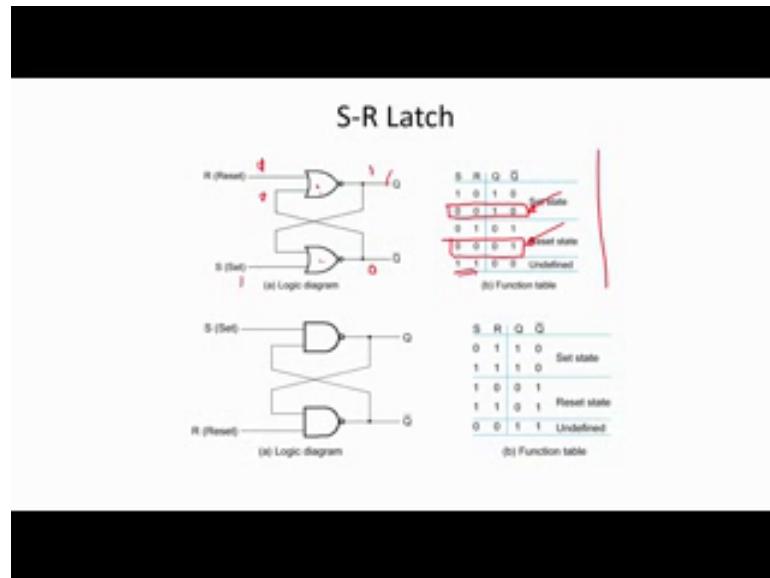
(Refer Slide Time: 49:34)



So, those output we need to retain it and we have going to say this is the some sort of storage element we are having. Now, this storage will come whatever you have store or retain that will come as an input to the circuit and the next output will depends on the current input as well as the previous the output. Now, when you are talking about the distinct say current output previous output; that means, it is related to time. So, these time will be maintain by a timing signal which is be call this is the clock signal. So, in every clock signal, whenever this clock coming then at that particular point this circuit is going to perform its operation. We are having the input signal, we have the previous output, now this circuit is going to works when this particular clock is high and according to give me output and when this clock is low reading this time this circuit is not going to be high.

So, we can just think in that particular way. So, when this is the 1 step this is the second step. Now, this output of this particular present step dependence on the output of the previous step. So, it will be control by a clock. So for that we need storage. So, when we are going to look for sequential circuit basically you have to see how you are going to retain the information; that means, to the storage element.

(Refer Slide Time: 51:24)



So, for that we are having a storage element called S-R latch, S stands for set, R stands for reset set and reset. So, it will be implemented with the help of your NAND gate this is the NAND implementation and this is a NOR implementation.

Now, how it is going to behave just I am going to explain one of this particular table. Say 1 S equals to 0 and R equals to 0 then I am having 2 input Q and Q bar, one is the compliment of the other. So, S means said when S equals to 1 it is going to said the output to 1 Q to 1 and then Q bar equals to 0 when 0 1 combination is here than it is going to research this particular circuitry and we have going to have this is A 0. And now, when 0 and 0 then what will happen what is the output over here. Now, here you just see that I am having 2 combination 0 0 is your 1 0 and 0 0 is your 0 1. So, see output varies when I am giving both input is 0 0 at some situation I am going to have 1 0 and in some situation we are going to have 0 1. So, its dependence on my previous computer if after 1 0 combination if I am giving I am giving 0 0 then what will happen it is going to retain

the previous output 1 0 will be retain over here; that means, I set it and after the I keep the signal 0 0 then I am going to retain this particular output forever.

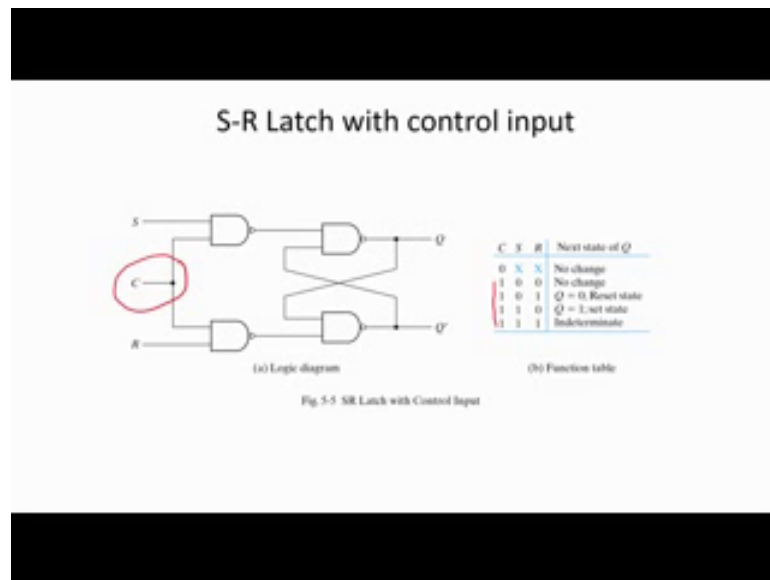
Again when I am going to give 0 1 I am resetting it. So, you are getting 0 1 output after that I am giving 0 1 then what will happen we are going to retain this particular information. So, this is basically presence output depends on my previous output so that is the way we are going to retain our previous information. So, this is the single bit S R latch. So, we can go for more number of bits. So, you are going to represent these think. Now, we just see how it is going to work. When it is your 1 0 say I am giving 1 over here. So, S R I will be giving one of the input is your high then what will happen OR gate is giving to me output is high and NOR gate is going to giving to me output is 0. So, it will be 0. Since their 0 is set back to this particular point and reset I am going to put a 0. So, when both are 0 then Q will be one. So, this is your 1 0.

Similarly you can analyze what will happen in case of your 0 1 also and in case of 0 0 it is going to retain the previous input you can see when input both 0 0 it depends on my this output. So, this is S-R latch, but here we are having 1 problem when we give 1 1 combination.

So, it is we are saying that one is the complement of other because if we give both as 1 1 then what will happen, it is going to get 0 0. So, it is undefined you are saying, but we do not have any problem. But after 1 1 if you give 1 0 then you are going to get 1 0 if you gives 0 1 then we are going to get 0 one, but what will happen after 1 1 if you are going to get 0 and 0 0 because in that particular case 0 0 may turn up to be a 1 0 or 0 0 may turn up to be 0 1, so what we are going to get?

So, this is a problem and we set this is the rest condition because why will say rest condition we are using 2 NOR gate over here and abrigator having some propagation delay, but it is not possible to fabricate to get with the same propagation delay here will be an fraction of differences, one will be slightly faster in the order. So, after 1 1 whether we are going to get 1 0 or 0 1 with dependence on the propagation delay of these 2 gates whichever is faster it is going to be x pause and accordingly it is going to set the output so that is why this is the rest condition and we should apart this particular 1 1 combination otherwise we do not have any problem.

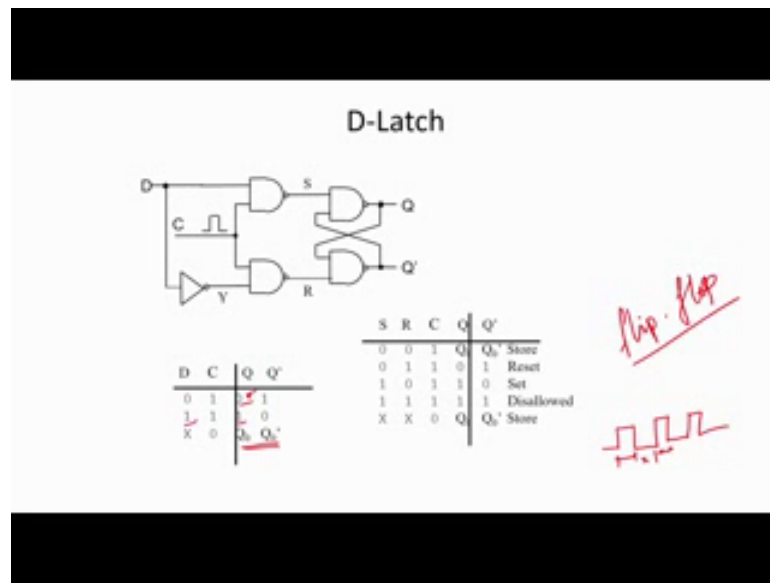
(Refer Slide Time: 55:55)



So, to apart this things what will happen? We are going to have a S-R latch with control input, here we are going to put an control input and this circuit is going to work when this control input is 1, when it is 0 that whole circuit is not going to perform there will be no sense; that means, whatever information we have it is going to retain over here if Q is 0 it will remain as 0.

Now, what will happen? Now, in that particular cases whenever you want to set it then you are going to give 0 1 over here whenever you want to reset it I am going to give 1 0 over here and if I am going to 0 0 then it is going to retain it. But when I am going to apply clock at that particular point we will try to applied to giving 1 and 1 combination at S and R, that is the way you can control it with the help of this control signal.

(Refer Slide Time: 56:50)

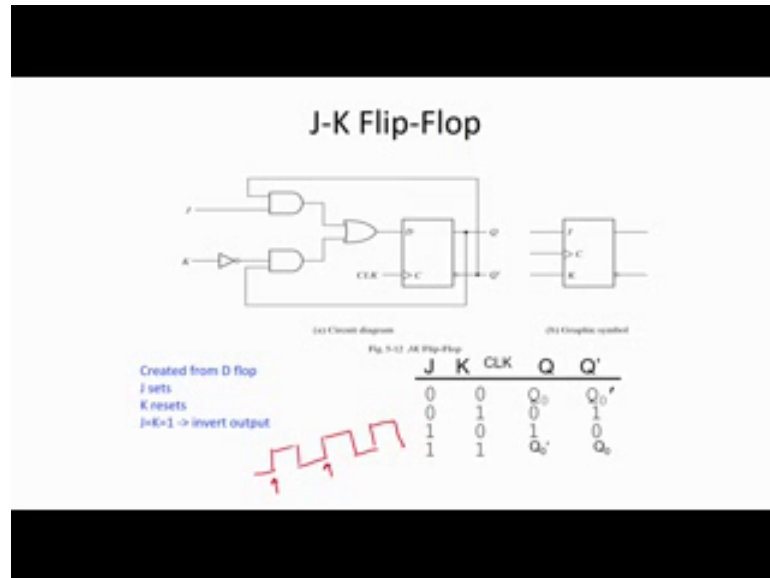


So, this is the basic building blocks about latch S R latch and with the help this things you can construct some of the other lectures or other flip flops. So, when we talk about it is clock then we use the term flip flop also. So, when we talk about latch then at the particular time that control clock signal is not there, but when it is a clock then we say these are flip flop also. So, we can construct those particular flip flop, with the help of these particular basic S R latch is control input.

Now, in this particular cases what will happen you just see that here we are having 2 input S R. So, in that particulars what happen what we have doing 1 is the compliment of the others. So, if it is d is 1 the other is your 0 and if it is 0 and other is 1. So that combination 1 1 is totally avoided. Now, it own go to an rest condition. And if you look into the behaviour then what will happen? When control input is not then whatever may be the D value then it is going to return my previous input. So, when D value is 0 then output is 0 when D is 1 output is 1, basically it is D is 1 output is 1 D is 0 output is 0 you can analyse it with the help of this particular table when in that particular case we say this is the D flip flop. Why you are going to say D flip flop you just see the behaviour whatever input we are giving it is coming as an output in the next test. So, after some delay it is appearing is my output; that means, when clock signal is arriving. So, we are giving a continuous clock signal. So, when clock signal is present at that time whatever input we have it will be (Refer Time: 58:42) to the output, but Q and during this time we are doing nothing again this point we are doing it the informer.

So, whatever input giving it is reflected in output to some delays this delays depend on the propagation delays of the components that is why we say this is a D latch or D flip flop or dilate flip flop, this is one.

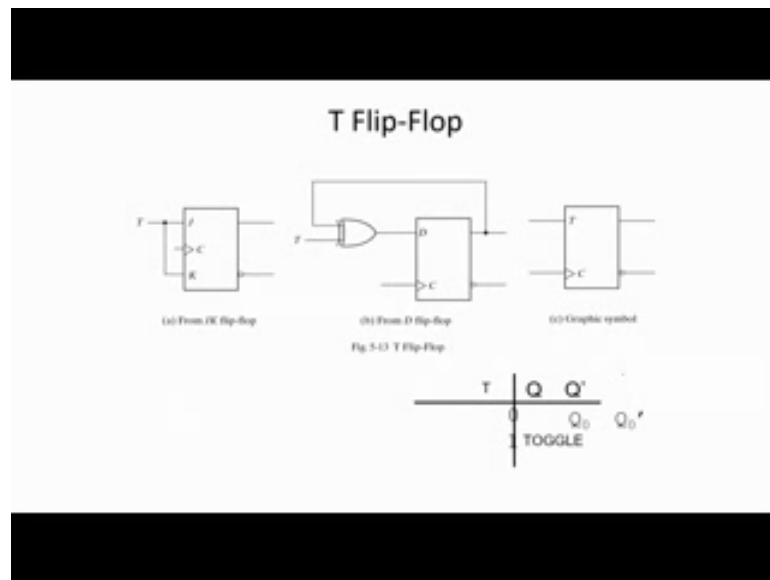
(Refer Slide Time: 59:02)



So, another 1 we having j k flip flop so again it is constructed you can constructed with the help of D flip flop here we can provide this J and K. So, just see the behaviour if says that if J and K are 0 0 there is no sense of the output if it is yours J is 0 and K is 1, basically K represent for reset we are resetting it. So, output is 0 and when it is your 1 0 Js then for yours said; that means, you are setting it output is 1 and when it is 1 1 at the particular point the output (Refer Time: 59:39), if it is Q then it will become Q bar.

So, if output is 0 then it will become 1 if output is 1 then it will become 0. So, this is another building blocks we call J K flip flop. So, you can control it with the help of this input signal J and K and along according to the values of J and K we are going to have the output over here. So, 1 is your no sense, 1 is your toggling the output and other to combination 1 is your set, other 1 is your reset. So, we can return the information at the output till the next clock arrives. So, we are having a continuous block over here.

(Refer Slide Time: 60:15)

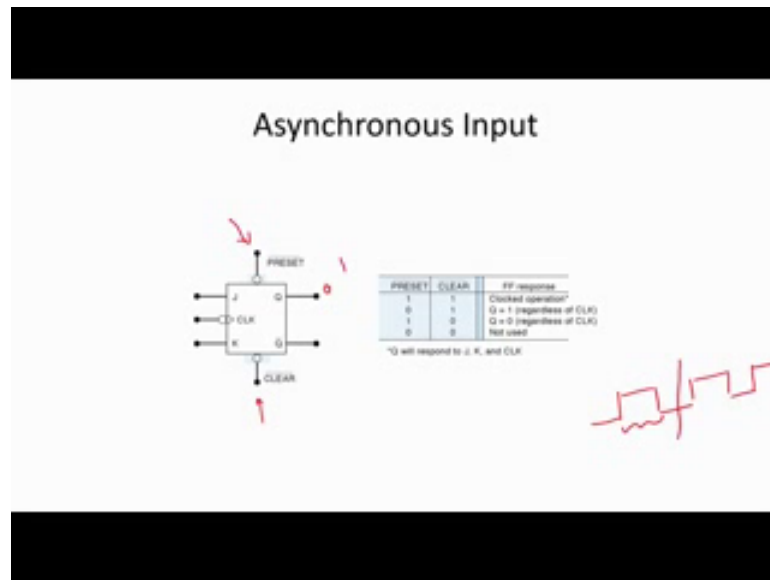


So, another one we are having T flip flop which is your toggle. So, this is very simple form constructing from J K flip you just see that when both the input is 1 then what will happen it toggles basically if output is 0 then it becomes 1 1 8 1 than it becomes 0. So, we tied this particular J and K to 1 symbol input. So, if T is 0 we do not have any sense in the output when T is 1 than it basically toggles. So, this is another building block. So, these are the storage, element flip flop. So, basic 1 is R S latch that from latch we can construct our flip flop and basically we are going have 3 basic flip flop 1 is your D flip flop, second one is your J K flip flop and third one is a T flip flop. With the help of this flip flop we can construct some other flip flops also. So, we are not going to discuss about it, but with the help of this flip flop we can now, construct our storage element.

Now, along with that we are having 2 more signals call one is pressure and one is your players. So, these are basically asynchronous input when you are coming about asynchronous input; that means, we are having another type of input also which is known as your asynchronous input. So, in case of asynchronous input what will happens, say here when I am giving sending my input J and K when output is going to get sense it will be synchronised with the help of an clock signal, when there is a clock signal arrives then during that time output senses. So, this is related to this particular control clock and you say this is the synchronised behaviour. Now, if you are using 2 flip flop to keep 2 bit of information if it is a synchronous circuits than what will happen output is going to get sense simultaneously when that clock signal is T and in case of asynchronous signals

what will happen as soon as my signal is coming irrespective of the clock signal it is going to sense the behaviour of the output. So, said it I am going to consider about a longer clock period.

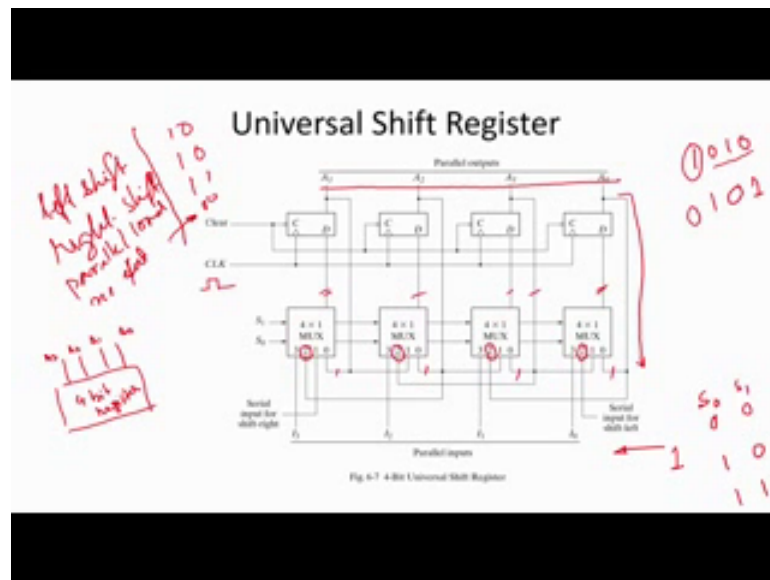
(Refer Slide Time: 62:14)



So, in that particular place set in case of synchronous if this clock is there then only output is going to get sense but if some input is coming over here there will not any sending output. But in case of this 2 asynchronous input be set and clear irrespective of the clock signals it is going to take immediate effect and accordingly it is going to sense the output. So, these are use to either set or clear this particular flip flop.

So, you are having storing some information, but at some point of time I want to clear it than as soon as I give the clear signal over here than what will happen the output will 0 it is clearing the information. And as soon as I am giving the p such signals basically you are going to set the value to 1, so you say this is the p setting a flip flop. So, it is irrespective of the top signal whenever this p setting signal will come what happened that cube it will be set to 1, so that is why you are saying asynchronous flip flop because while constructing our circuit sometimes we will be needing this particular clearing the information or setting the information to 1. Now, with the help of this particular flip flop we now, you can construct some of the basic building block. One of the basic building blocks is your registers. What is a register basically? Register is an device electronics circuit where we can store information.

(Refer Slide Time: 63:56)



So, in that particular case if I am going to say that I am having a 4 bit register, then what will happen we are storing whole bit of information and whenever required you are going to take it to the output line say A₀, A₁, A₂, A₃. So, you have storing some information. Now, what is store in this particular register? Basically we are having 4 flip flop according to our requirement we have storing the information in the flip flop and after that whenever is required you are going to take this particular information. So, here we are going to talk about it universal shift register. So, basically this is the register with the help of this register we can store our information and we can perform some operation and you can take out our information whenever it is required.

So, it is an give me a diagram of 4 bit universal shift register. So, what happens basically? It is having 4 operation 1 is your shift register we can shift the information either towards right or towards left. So, one is your left shift second one is your right shift and another information we are having parallel load we can both the information parallel. So, 4 bit of information we can load it parallel and whatever information we are having these are available always in this particular output lines. So, these are basically D flip flop these are the cube point of this T flip flops these are always available.

Now, you just see what is there basically I am saying that we are having 3 operation left shift right shift parallel load and along with that we can say that no effect. So, whatever information we have always these are retail. So, for that now, you are having 4 operation

you need to perform. So, for that what happen you are using a 4 by 1 max depending on the input scenario we are performing this particular, giving one of the input as an input to this particular dip flip flops. So, say this 4 by 1 max have been 4 input selection and we are going to select one of this particular input and putting in are the output of the match. And how you are going to select it? We are going to select it with the select line S 0 and S 1. Now, just I am going to consider 1 particular input you just see this particular signal this output is directly connected to 0 this output is connected to 0. So, all the outputs are connected to 0; that means, whenever I am giving that select line as 0 0 this input lines will be transferred to gives output lines of the max and that will go as an input to the D flip flop and whenever clocks arrives then what will happen that will be transferred to this thing. That means, you see that we are retaining that information over here; that means, that no effect will be given by this particular control signal S 0 and S 1

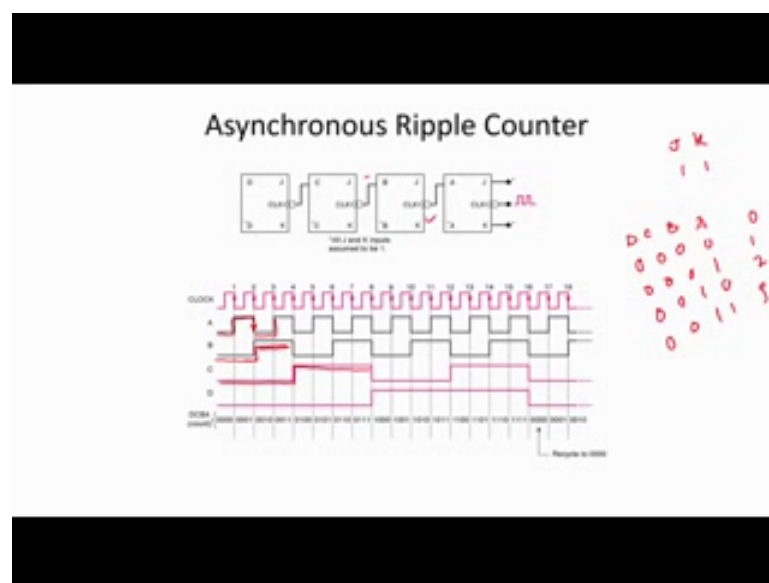
Now, just consider one particular thing. You just see about this particular input 2. Now, what is happening in this particular things we are giving a new input and whatever information we have over here this is coming as an input to the next T flip flop. The output of this one is coming to as an input of this things. Now, what will happening say initially if I am storing say 1 0 1 0 say we are retaining it. Now, whenever I am giving the input as 1 0; that means, 1 0 I am selecting this particular 2 signal then what will happen whatever input I am giving it is coming to this particular input signal and it is available over here. Now, from A 0 what about output is here it is coming as an input for the next flip flop. So, now, this is basically transferring the information from A 0 to A 1, A 1 to A 2, A 2 to A 3.

So, basically 0 will come over here, 1 is come over here and 0 will come over here and this one we are not keeping anything. So, this is 0 1 0; that means, these 3 and whatever (Refer Time: 68:47) I am giving input 1 then it will become 1. So, in that particular case say whatever we are having we are shifting this information what is left. So, it is now, new information will come this one we are not storing anyone. So, this shift at 1 bits towards left and the new input is coming to this particular serial input. So, this one 0 combination is going to give me your left shift operation.

Similarly we can have the right shift also just see the behaviour and you can find out the control signal and whenever I am going to give 1 1, so, basically that left shift we are going to get that one 0 so here right shift will be 1 0. Now, when you give 1 1 then what

will happen you just see that this third input line of max will be transferred to the output of the max and when clocks arrives then that will be stored in this particular D flip flop so that means we have parallel loading this particular input to this particular register. So that is why we say this is a universal shift register we can perform the shift operation as well as parallel load or we can retain the information also as long as we required it. So, this is a 4 bit. So, if you need an 8 bit then what will happen? Now, I can use 8 flip flop and 8 max. So, to store our information in our computer in our processor we use such type of register.

(Refer Slide Time: 70:26)



Another one it is called counter. So, this is the ripple counter we are having. Now, what is basically, it is having you just see that we can say that this is a continuous running clock and as soon as the clock is coming then what will happen the output is going to get sense. Now, what basically (Refer Time: 70:39) here we are going to put 1 1 as J and K for all the flip flop; that means, that J K flip flop is going to act as an toggle flip flop T flip flop. Now, what will happen say A B C D, initially say we are resetting it say all are 0.

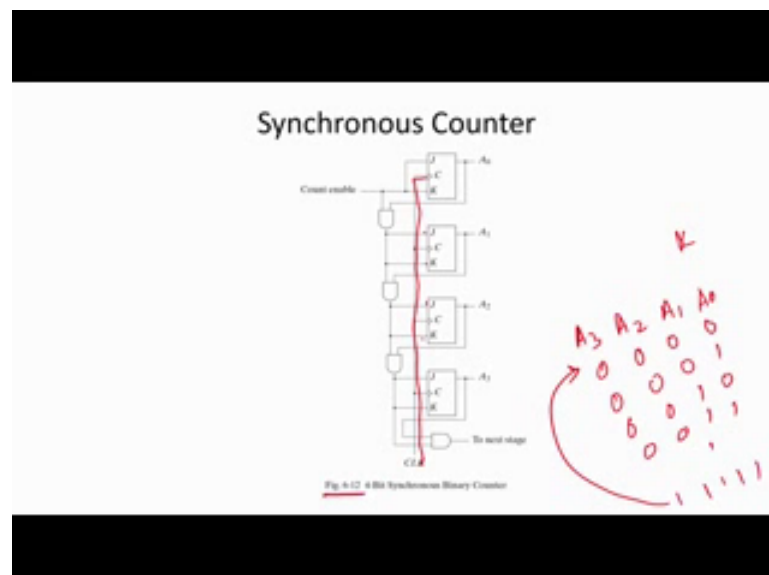
Now, what will happen when this clocks arrives then what will happened input is going to get sense over here depending on this particular behaviour. So, as soon as this clock and arrives then what will happen that A is going to cut the call, so it will become 1. Since this is remain as 0. So, all will remain 0. So, this is the behaviour you just see that

when clocks arrived that A is going to get it from 0 to 1, but it will remain 0. Now, whenever its next clock is coming you just see that again it is going to get sense, but not since this cloth is available now, now, this b is going to get sense. So, now, when this signal is coming over here then I am going to get 0 0 1 0.

Similarly when this is coming. Now, 1 it will remain as 1. So, there is no sense in C block. So, it will remain as 0 and next block coming it is 1 1. So, we are going to have such type of (Refer Time: 72:00). So, this is the number if I am going to look for the these things this is 0 1 2 3. So, it is going to have that going to give as counting affect you can count our numbers. So, this is another circuit. So, in details you are not going to discuss it; and why you are going to set this is asynchronous because all are not controlled by a sand clock signal they are going to control by different clock signal. So, this clock is coming out from the first flip flop.

And when this signal is going to get it sense from say 1 to 0 then this flip flop is going to get a effected it will go with from 0 to 1. So, all are not synchronise by the same input call they are synchronised by different or they are a parted by different signals clock signals so that is why I know this is asynchronous counter.

(Refer Slide Time: 72:50)



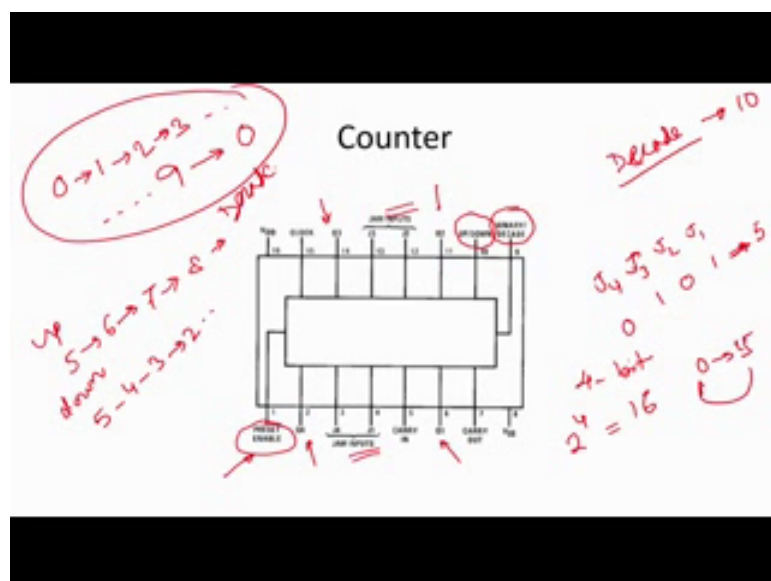
Similarly we are having that synchronous counter you just (Refer Time: 72:56). Now, here we are giving the same clock to all the flip flop. So, again it is a 4 bit counter I can say that A 0, A 1, A 2, A 3. Now, you just see how you are setting J and K over here

depending on that you are going to get the behaviour initially if all are 0 then when first clock pulse comes then it will become 0 0 1, next clock pulse it will become 0 0 1 0 then 0 0 1 1 like that it will go and finally, all are will be 1 and again from 1 it will go to 0.

You can analyze this particular circuit and we have saying this is your synchronous counter because all the flip flops are control by 1 common clock and I think here you can see this things that some figure 6.12 is coming basically I am taking this particular diagram for my book of digital design by M Morris Mano and this counter circuits are the explain in the chapter 6. So, you can go for this particular chapter if you want to have some details of analysis of this particular counter.

But here we are simply talking about the information purposes only in knowledge level we are going to use counter when you are going to be build our computers. So, this is the some blocks of we have counters.

(Refer Slide Time: 74:15)



Now, what will happen you just see that you are I am having some of the input signal 1 is your appt. So, it is at are counting up or counting down. along with that we are having some input also that means with the help of this input we can preset this particular counter with the help of this particular presets enable. Now, just said it if I am going to give set these are J 1, J 2, J 3, J 4. Now, in that particular case if I am giving 1 0 0 1 and I am enabling this particular preset then what will happen. We are going to set this

particular counter to 0 1 0 0 and this outputs are basically these are Q bit these are outputs. So, in this counter we are going to get output as 0 1 0 0.

At that particular point I think you are having slight idea about binary number system and decimal number system. So, in that particular case I think this is your the single fight. In next class we are going to see about more details about this things.

Now, when I said that this is an up counter than what will happen it is going to come from 5 6 7 8 liked that in case of up counter when it set it as down counter than it is going to good a come down basically going to 5 4 3 2 like that. This is the down counter. So that means we can reset the counter to some initial value from that we can count up or we can count down also. Along with that we are having another one call counter signal call binary and decade. I think you know what is decade, decade stand for a particular decade; that means, it is 10.

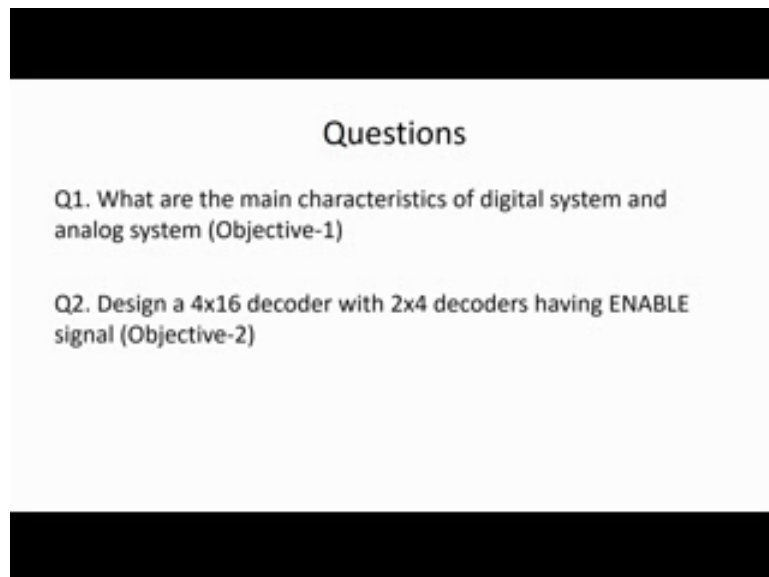
So, if it is a safe binary counter it is a 4 bit counter what is the different combination we are having in 4 bit this is your 2 the power of 4 which is equal to 16. That means, it will count from 0 to 15 and come back 0; that means, we can have the counter for 16, 0 to 15 and coming back to 0 when we are going to use as a binary counter. But in case of decade counter what will happen we are going to restrict the count to 10 only, 0 to 9 when as soon as your 10 will be coming then again will be reset to 0; that means, we can counter decimal number 0 to 9. So, what will happen it will come from 0 1 2 3 like that it will count up to 9. Then what will happen? When after 9 in case of binary counter it should go to 10, but now, here again it will be 0 1 2 3 like that, when you set to as a decade counter we are going to count 0 to 9 again these set to 0. But when it is a binary counter then we have going to count from 0 to 15 then again 0. So, this is a counter that we are going to use in outside.

So, here what will happen, we have discuss some of the building blocks that will be used in our computer and what is those building blocks we are having to type of building blocks 1 is your combinational circuit second one is a sequential circuit; in case of sequential circuit flip flop is the main building block with the help of 1 flip flop we can store 1 bit of information. So, if we need to return more bit of information than you are going to use more flip flop. And 2 basic building blocks we have discussed over here 1 is your register, registers are nothing, but the storage element we can store our

information with the help of and bit register we can store and bit of information. Another one counter it is basically going to counter sequence from 0 to and, if you are going to look for an and bit counter than what will happen if can count from 0 to 2 the power of and if it is in binary counter, but that counter can be configured for other counter also like that we have discuss about the decade counter.

So these are the basic building blocks that you are going to use while going to construct your computers. Now, just sees some of the test item of questions. So, first question I am saying that what are the main characteristics of digital system and analog system.

(Refer Slide Time: 78:37)



Questions

Q1. What are the main characteristics of digital system and analog system (Objective-1)

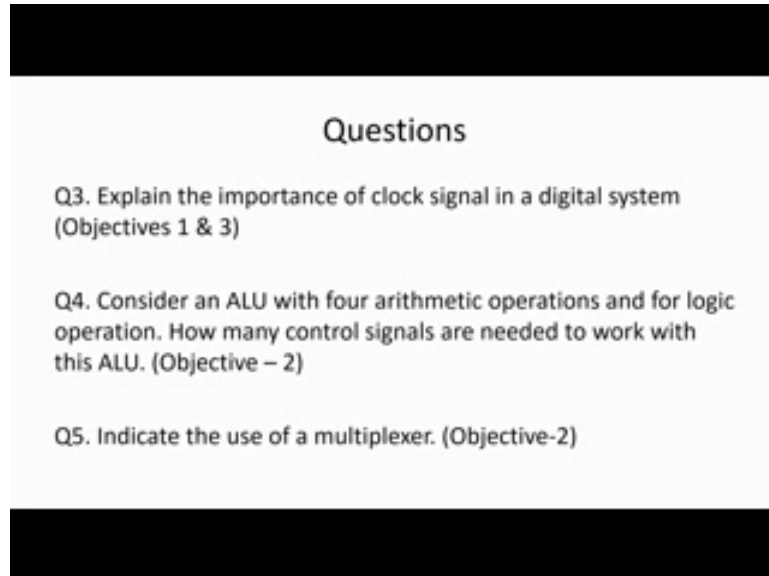
Q2. Design a 4x16 decoder with 2x4 decoders having ENABLE signal (Objective-2)

So, already I have explain about it digital mean discrete it is going to work form discrete signal. So, we have mention some of the objective. So, this question is related to the objective one of this particular mount unit.

Second question I am talking about design a 4 by 16 decoder with 2 by 4 decoder having enable signals. This is meeting the objective 2. So, construction of some digital building blocks. So, already I have explains 1 example I think I am constructing a 3 by 8 decoder with the help of 2 by 4 decoder. Now, you are asking to construct 2 by 4 by 16 decoder with the help of 2 by 4 decoder. Just first you think how many decoder will be needed to construct this one, you have to be correct 1 if you have going to get the right number then drawing diagram is a very simple one. So, you have to find out how

many 2 by 4 decoder will be needed to 4 by 16. So, 4 4 za 16, but I can tell you that it is not 4, 2 by 4 decoder, this is not the correct answer.

(Refer Slide Time: 79:53)



The slide is titled "Questions" and contains three numbered questions:

- Q3. Explain the importance of clock signal in a digital system (Objectives 1 & 3)
- Q4. Consider an ALU with four arithmetic operations and for logic operation. How many control signals are needed to work with this ALU. (Objective – 2)
- Q5. Indicate the use of a multiplexer. (Objective-2)

Third one I am talking about explain the importance of the clock signal in digital system. So, this is objective 1 and 3 basically, in case of your sequential circuit it may be synchronous on asynchronous we need the clock signal. So, I just look for importance of this particular this clock signal when you are going to design any digital system.

Question 4, I am giving like that consider and ALU 4 arithmetic operation and 4 logic operation. How many control signals are needed to work with this ALU? I think I have already explain this things in my lectures we can go back to that particular portion and you can get it over they are.

Question 5, indicate the use of a multiplexer. So, you know what is multiplexer we have several signals, now you want to select one of them only that can be done with the help of multiplexer. Like say in question 4 this is a use of math decoder. With the help of decoder we can select one of those particular operation. Now, you just think where you are going to get the use of multiplexer that we are having several input and we have going to select one of these things.

I think already in our lecture we have used in somehow in multiplexer, maybe you remember it while going to look for the designing of our universal register. I think you have used multiplexer over here. Now, you look for some other example where the multiplexer will be used. So, with that I am going to wind up this particular unit.

Thank you very much.