Digital Circuits and Systems

Prof. S. Srinivasan
Department of Electrical Engineering
Indian Institute of Technology, Madras
Lecture - 30
Encoders and Decoders

(Refer Slide Time: 1:45)

Digital Circuits and Systems

* Encoders and Decoders

* Combinational Design using Decoders

* High - Impedance State and Tri - State Logic

Lecture #29 Prof. S. Srinivasan Department of Electrical Engineering

So in the last lecture two lectures we talked about multiplexers in detail and how to use them as building blocks or functional units in designing a combinational logic. Today we will see another circuit which is MSI which is a decoder so first we will see what an encoder is and then a decoder, encoders and decoders.

As the name suggests it one codes the various combinations into a smaller number of bits and the other expands it back. So an encoder for example suppose I had eight inputs 8 to 3 encoder assuming only one of the inputs is active, it could be active high or active low at a given time all we have to know is which of these eight inputs is active so you need only three bits for that. By combination of these three bits you can determine which is the active input and all others are inactive at that given time. So this is the process of encoding. Encoding reduces the possibilities or the number of inputs required from 8 to 3 for example or sixteen to four. So compressing the data from different possibilities into a smaller number of bits is the process of encoding. In this case we will have eight inputs let us call this I_0 I_1 and I_2 etc I_7 and you will have three outputs we will call them O_0 O1 O2 this is MSB and this is LSB.

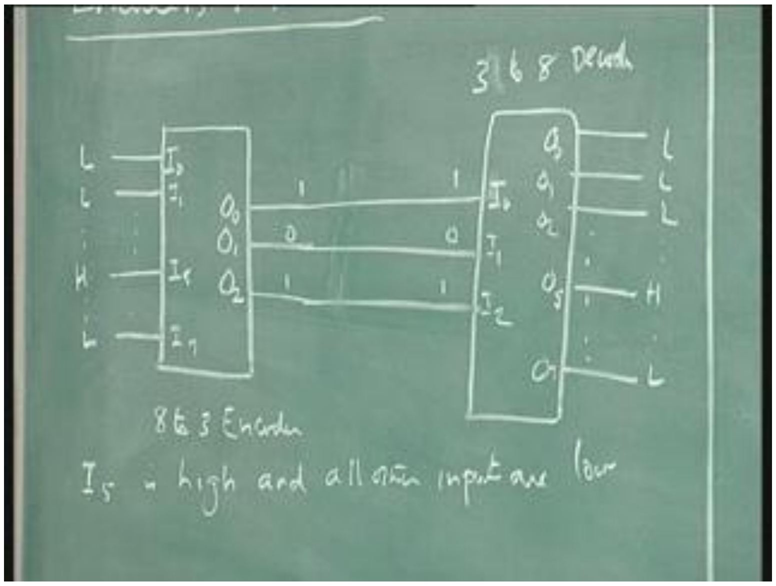
Of course if more than one input can be high at the same time then what happens in which case we talk of priority?

If more than one input is active at a given time then we don't know how to determine the output but there can be some rules we can build some rules into that. We will not talk about that, I am just going to tell you about the type of MSI which are available.

So, for example if I_5 is at high and all other inputs are low the output would be 1 0 1. This process is called encoding. My question is as I said if more than one input is high what will happen to the output? We can have a procedure or a rule for this. One scheme is called priority encoding. I can say the input with the highest binary value could be recognized or the input of the lowest binary value is a recognizer. You can say priority goes from I_0 to I_7 that means if I_0 and I_1 are both active I_0 will be coded and I_1 would be ignored. This is one scheme. You make the rule and then build the circuit accordingly.

Or I can say lowest order. I can say higher inputs have priority that means supposing I_5 and I_6, if along with I_5 I_6 is also active high I can say I_6 should be recognized and not I_5, then the output will not be 1 0 1 but it will be 1 1 0. Anyway this is called encoding process of reducing the number of bits into smaller number of bits without any loss of data. In the reverse of this the complimentary operation is called the decoding operation. So this is an 8 to 3 encoder (Refer Slide Time: 6:11) so we will have a 3 to 8 decoder so I can put this here straight away so three possibilities are there and any combination of this can happen because there will be eight different outputs so the corresponding output will be high and all others should be low. So I can have again I_0 I_1 I_2 lowest order and these are the highest order bits and then I can have O_0 O_1 O_2 O_7. Now this 1 0 1 goes there as 1 0 1 making O_5 high and all others low. This is the decoding operation. This is the encoding operation and the decoding operation.

(Refer Slide Time: 7:12)



Why do we need this?

I have eight possibilities as the inputs and eight possibilities are required as the output and I don't want to run eight lines from inputs to outputs because one of these lines will be active high at any given time and others will not be. So instead of running eight lines and finding out which is active high I can send three lines with a code a coded information into that and then at the output will decode it at the receiving end we will decode that three bits to find out which input exactly was high here at the sending side. This is compression data, data is compressed. Eight lines instead of running because of the length requirement we can do it with three bits, three lines will do the job. And if there is a priority scheme here it doesn’t really affect this here because here it is an expansion scheme so three is expanded into eight so whatever is the combination will be expanded accordingly.

Here of course there may be problem with more than one input being high so in that case we will have a priority scheme by then. In such a case it is called a priority encoder, either a normal encoder or a priority encoder. The priority encoder will have a priority scheme built in saying when more than one input is high which of those inputs will be recognized and coded at the output.

Now how is it going to help us in our combinational logic design? It is just as multiplexers helped us to design combinational logic without going to the gate design. Because after all inside you can build these gates, we can draw a truth table, can you not? Can I not make a truth table out of this?

ABC are the three inputs or I_0 I_1 I_2 are the three inputs, O_0 to O_7 are eight outputs so I can draw a truth table and 0 0 0 combination will make O_0 high and all others low so I can have a gate realization for this, likewise I can have a gate realization for this so all are gates finally. Basically they are all AND OR inverter. I told you long ago that AND OR inverter is a sort of universal combination by which any circuit can be recognized and can be designed so the same thing applies to the encoder and decoder.

Or if we don't want AND OR combination we can have NAND gates universal gates or any other combination of gates that you want to work with. So what is the advantage of using this? Now we have this scheme of so many equivalent gate functions may be about ten to fifteen or twenty depending on the number of inputs number of outputs and priority schemes built or not somewhere between 10 to 20 or 10 to 30 gates will be there so when you compress it and make it as a single IC and make it available to you putting in one IC chip then it becomes a Medium Scale Integrated circuit MSI.

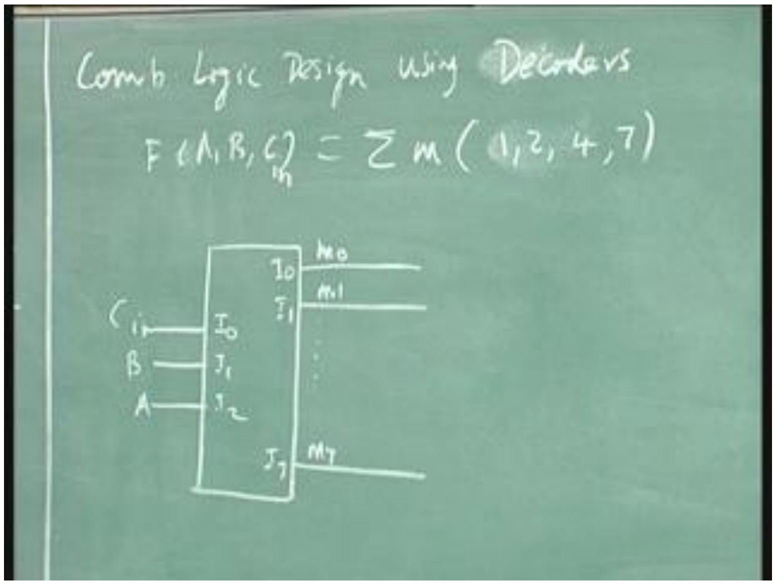
Our aim in the last few lectures has been to use these Medium Scale integrated circuit components in the design of combinational circuits. So we have to see how this hardware encoder and decoder will be used in combinational logic design. Let us say design a combinational logic using Decoder. This is a very straightforward procedure here because even in multiplexer of course the mapping procedure was very clear but it is not even that here it is a straightforward case. Because when you have three inputs and eight possibilities this is similar to the truth table is it not. That means all possible min terms are generated. The question is one of them will be high and all others will be low.

but in general though when you have three inputs and eight outputs it's equivalent to a truth table with three inputs and eight outputs and each of this output being a min term and if you know which are the min terms that are high for that particular combination of the hardware then I can combine them all together and make a circuit out of it. It's as simple as that.

Suppose I want to realize a function f sigma F(A, B, C) A being A, B and C you will have to be always clear with which is MSB and which is LSB. Here I am considering this as the MSB and this as LSB (Refer Slide Time: 12:10). So, when you map it I should connect A to this, b to this and c to this so that's why I am writing in this fashion F(A, B, C) it will be equal to sigma let us say arbitrary or whatever you want or use the full adder (1, 2, 4, 7) is the sum of the full adder. You have seen full adder so many times 1-bit full adder min terms which are high for the full adder sums are min terms (1, 2, 4, 7) so we will realize a full adder and that's what we did with multiplexer to start with, we will do the same thing with decoder.

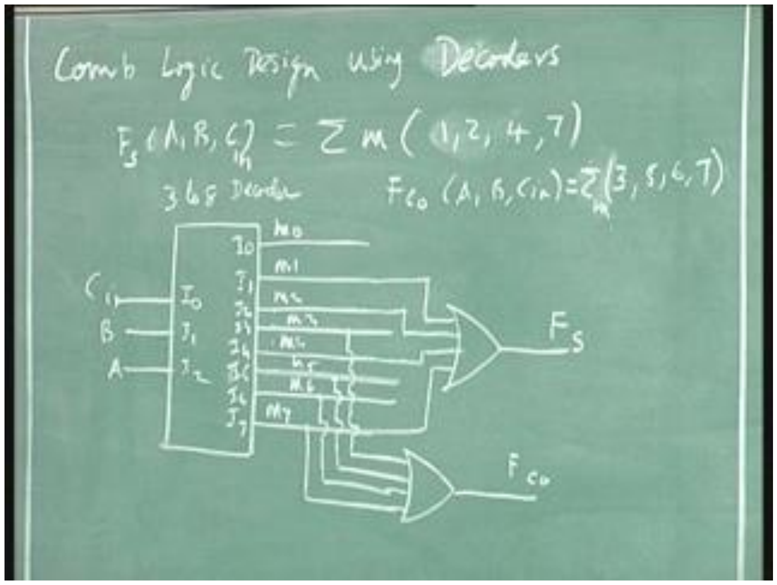
So all you have to do is to get my three inputs this is C_{in}, here this is A, this is B, this will be C_{in} in that order I am writing so the same order should be mapped and output will be m min terms and I will call this I_0, I will call this m_0 which is same as I_0, here these are Is equivalent to the min terms so I will put I_0 inside and m's outside, this is m_0, this is m_1 etc and this is m_7.

(Refer Slide Time: 13:50)


Comb Logic Design using Decoders
F(A, B, C_in) = Σ m (1, 2, 4, 7)
C_in
B
A
I_0
I_1
I_2
I_0
I_1
I_7
m_0
m_1
m_7

Now in order to get a full adder I need to combine m_1 m_2 m_4 m_7 into an OR gate. So I take m_1 m_2 m_4 m_7 this is my f this is the sum of the full adder, a simple scheme of realizing a combinational logic using a 3 to 8 decoder. This is a 3 to 8 decoder. This will the only function I can realize if I have one more function using the same min terms I can also get it. Suppose I want a carry also in this can I use the same decoder for this? Of course I can. So my carry would be I will call this F1 or Fsum, F carry out of ABC in or if I remember it was 3, 5, 6, 7 sigma that is F_{co} (A, B, C_{in}) is equal to sigma m(3, 5, 6, 7) these are the min terms for which the output is 1 for the carry so I can combine them this now requires m_3 m_5 m_6 m_7 so m_3 would be here I_3 m_5 I_6, m_3 m_5 m_6 m_7. So we will have to take m_3 m_5 m_6 and m_7 put it in OR gate and this is my F_{co} carry out f sum F_s.

(Refer Slide Time: 16:28)


Comb Logic Design using Decoders
$F_s(A, B, C_{in}) = \Sigma m(1, 2, 4, 7)$
3 to 8 Decoder
$F_{co}(A, B, C_{in}) = \Sigma_m(3, 5, 6, 7)$
C_{in}
B
A
I_0
I_1
I_2
m_0
m_1
m_2
m_3
m_4
m_5
m_6
m_7
F_s
F_{co}

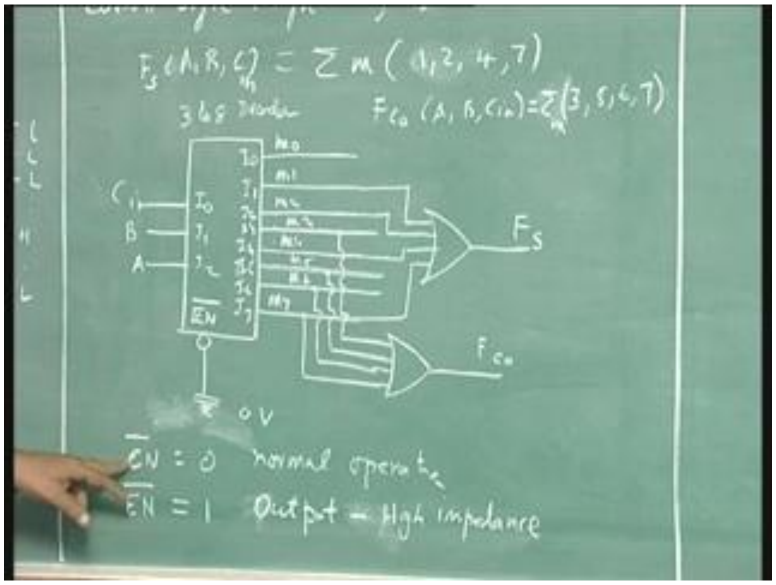
So there is one 3 to 8 decoder and two gates and we are able to realize a full adder so that is the MSI concept Medium Scale Integrated circuit replacing the bunch of gates. It may not be most efficient way as I said, it’s a mapping process. In the mapping process you have two problems. One is that mapping itself may not be efficient because the hardware availability is limited and you have to get the best possible way of doing it. The second limitation is that not always there will be a straight cut method for doing it. As I said the other day it’s a heuristic method sort of an intuitive method. We have to identify the hardware availability and the input requirements and try to match it so there may be an error in that.

Even if you do software for this software may not always do it the optional way. So we may use some extra hardware in that process because of that or there may be also be extra hardware required because there is no one to one correspondence or everything that I want to realize I may not have an equivalent MSI. But still it is worth it because the number of gates is drastically reduced in terms of, the number of ICs is drastically reduced saving space or whatever power, cost, size and so on. The same is repeated there, the parameters we were stressing on right from beginning in this course.

Of course I am giving you a bare bone sort of a approach in the problem. There are some practical issues here. We are building most of the things conceptually, we are not getting into circuit details. Of course as I said that is not required if you are going to use a logic design or you know what the input output specification is. But even in a logic design you should know a few things for example you should know how much voltage you should apply, should you apply 250V or 10Kv or 5V or 3.3V. Of course you will get different data sheets but you should know what order of voltage you are looking at, what are the current levels and so on. Suppose you buy an LED to indicate your output, what type of LED, what type of current is available and whether an LED can go with that so these are the things you should know of course, a little bit of the gate level electrical specifications as we call them.

Even though it's all logical and we are not getting to the physical design which is the next level where you replace this logic by components which are available we are not getting into that because of the background is not adequate in terms of the electronics but there are certain things. One of the things is there is always an enabled class in all of these things, whether it is a decoder, encoder or a multiplexer it will always be a signal called enable signal, I will introduce it here because it's all same in many ICs but I thought sometimes I should tell all these things to you before you get out of this course like this preset and clear there are extra signals in flip-flops. For a logical design or a simple analysis understanding of the flip-flop you don't need to know it but then I introduced it.

(Refer Slide Time: 19:45)


$F_S(A,B,C_{in}) = \Sigma m(1,2,4,7)$
3×8 Decoder
$F_{Co}(A,B,C_{in}) = \Sigma_m(3,5,6,7)$
C_i
B
A
I_0
I_1
I_2
EN
F_S
F_{Co}
0V
$\overline{EN} = 0$ normal operation
$\overline{EN} = 1$ Output – High impedance

Likewise here I want to introduce a signal called the enable signal. 'Enable' usually is a negative signal the active low. Whenever you put a bubble or a signal it means the signal is active when it is low and when it is high it is not active. Whenever you don't put a bubble signal is active when it is high and not active when it is low. There is an enable signal here and this has to be tied to ground if this circuit will work (Refer Slide Time: 20:19) and this represents ground in electronic circuits or this is 0V. That means if enable is low whatever we said about this decoder encoder is true. If enable is high if I do not connect the enable to ground properly then whatever you give here will not reflect in the output, the output will not behave as expected in the truth table, this is true of many ICs multiplexer as an enable signal, encoders have enable signals, decoders have enable signals and sometimes more than one enable signal. These are signals that are required for controlling.

I may have a circuit and a board and a system, I may have this IC and a circuit but I may not want that to be active at a given time, I want it to be active at some other time so how do I control that, I cannot just pull it out whenever I want to put it in and whenever I do not want I pull it out, it's not possible to physically remove the IC and put it back. Suppose you want that IC not to affect the rest of your operation you don't want an IC there at that time you have to pull it out and you want it there you put it in but it's the electrical equivalence of removing and putting it back.

When enable is low it's equivalent to keeping it there and when enable is high it is equivalent to removing it so that it is not there. So electrically it is not there physically you may have inserted the IC in a socket but electrically it's not there electrically you can ignore it. Such things are required very much because many designs are complex designs where several ICs have to co-exist and certain operations requires certain paths and at that time the other circuits should not come in the way so how do you control it is by these things.

And, there may be more than one enable. When you say enable if you want to be very clear and if you don't know all these and since you don't have the circuit diagram so every time you are talking about a decoder you cannot draw a circuit and show enable as a bubble so what you say is simply say enable bar so people understand. When you say a circuit as an enable bar which means I should make it 0 for it to be normally active, a normal operation so enable 0 is a normal operation and enable 1 is abnormal operation it's an electrical open circuit.

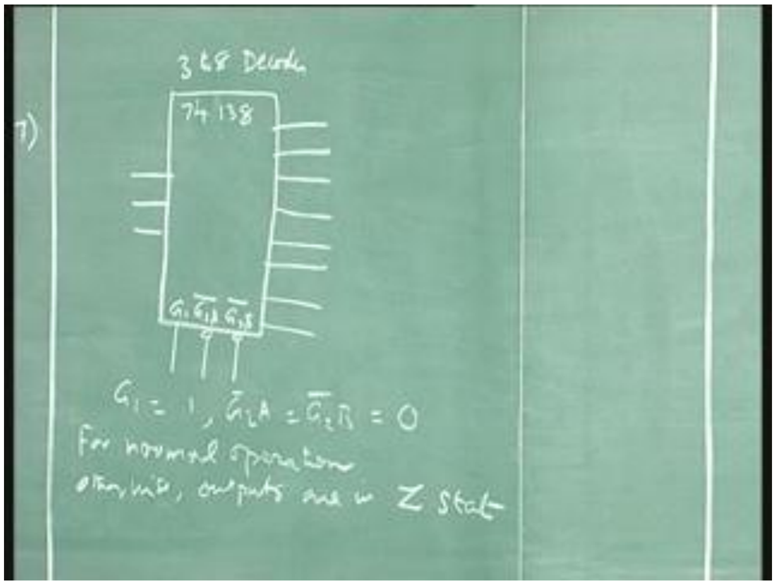
I have to define a term here and since I have not defined a term here I am not able to write it it's called tri-state operation output tri-states. Again these are some compulsions. As you go and I have to introduce a new term I can't keep postponing these things. The output is tri-stated. Here I can say that the output impedance is very high, output high impedance, output impedance high, output high impedance state. High impedance means impedance such that it is as high that you can ignore it. All of you have gone through electric magnetic circuits course so you know what it is. High impedance means physically if we put hundred mega ohms it is just as if it's equivalent to open circuit for practical reasons. of course mathematicians may argue that it's not 0 it's not open but we are engineers not mathematicians so we have to be practical. When you do something you will have to do the practical implementations.

Therefore high impedance means the output is as if it's disconnected from the rest of the set. Like you sit in the class and disconnect yourself from what is happening thinking about something else so that is exactly what high impedance state is. You are physically present mentally off many times it happens right? I do it when I go to some meetings which is boring but I have to be there because I am required to be there, it happens to everybody. The only thing is I wish that it doesn't happen too often in this course. That is exactly high impedance.

So we have a function called tri-state gate function which is what will be put in output for these operations. But there can be more than one enable so this is only one enable. In a practical decoder like 3 to 8 decoder generally there are three enables G_1, G_2A and G_2B where G_2A and G_2B are active low and G_1 is active high. So, if this circuit works normally, this is a practical 3 to 8 decoder, I can give you the IC number if you want to this is 74138, it is not necessary to remember numbers. In fact I am not even sure it's 128 or 138 but I think it's 138.

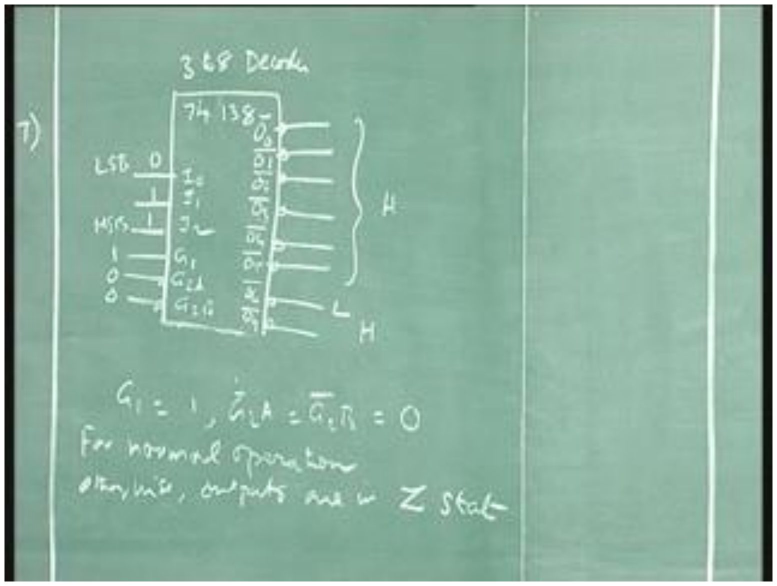
Now this particular IC which is three inputs eight outputs has three enables that means for a circuit normal operation G_1 should be high, G_2A and G_2B should be low only then the circuit is a normal operation. Why do you need three? As I said sometimes the control signals I want to control it I may have to control it from different times of view from different ways so I can use these three combinations. But only when all the three occur as required only then there is normal operation but the rest of the time otherwise outputs are tri-stated, otherwise outputs are high impedance states, outputs are in Z state. Instead of writing every time high impedance state capitalize Z and put it as Z state. Z state means high impedance state for digital circuits, this is an accepted nomenclature.

(Refer Slide Time: 27:41)


3 to 8 Decoder
74 138
G_1 $\overline{G_2A}$ $\overline{G_2B}$
$G_1 = 1$, $\overline{G_2A} = \overline{G_2B} = 0$
For normal operation
otherwise, outputs are in Z state

When you say Z state it means a high impedance state, accepted nomenclature will be digital. I wish I remembered whether it's 128, 138 but it's ok let us assume 138 for now. Please check up may be 128. The outputs are not active high. One other practical note on this IC is that the outputs are active low, so instead of getting O_0 O_1 O_2 how many are there? 1 2 3 4 5 6 7 I will probably put the enables here it doesn't matter, enable does not have to be at the bottom because it's all on in what way it is, it's only a representation of circuitry in the book, you can write it here also; G_1 G_2A G_2B, this is I_0 I_1 I_2 (Refer Slide Time: 28:57) and this is LSB, MSB. O_2, O_3, O_4, O_5, O_6, O_7 and these are active low outputs as well.

(Refer Slide Time: 31:00)



That means if a particular combination occurs the combination occurs after enable is properly connected of course. So we are assuming this is 1, this is 0 and this is 0 and I have properly connected the enables. If this is my only IC and it's not required to be controlled by any other event in the circuitry I can directly connect it to ground and power supply. These are control signals but it does not mean that it always has to be controlled. Whenever we need to control they are available to you. If you do not want to control and you always permanently want this normal operation then there is no problem make 1 0 0 connected to high, connected to low, the power supply connected to low it will work always normally so that is possible, it's not an excluded operation.

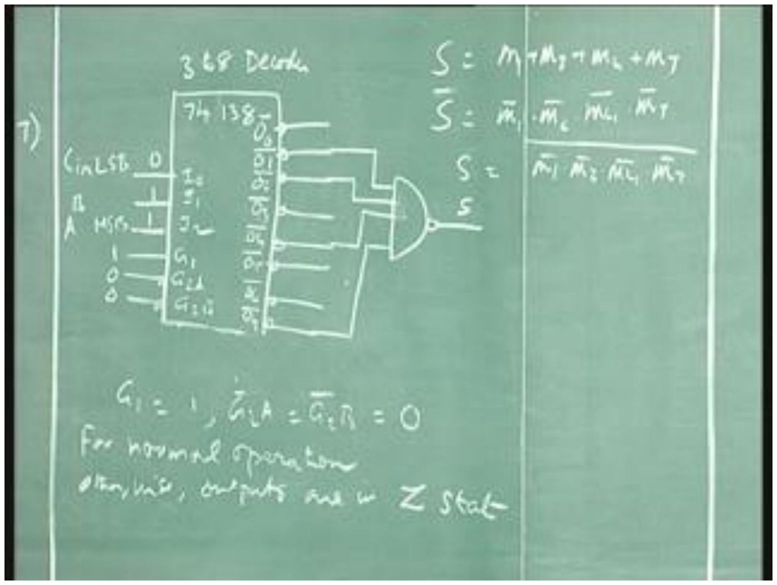
Anyway having done that suppose I give 1 0 1 or 1 1 0 O_6 should be active high and all others should be active low. But because we have bars on the outputs if you want you can even put a bubble on the output, the bar and bubble are redundant, bubble is only when you draw whereas when you are writing and when you are looking at the data sheet you will see these signals I_0 I_1 I_2 and O_0 O_1 O_2 so you need a bar for that. So O_6 will be low and all others will be high. Normally it will be the other way. When I say active high outputs O_6 should be high and all others should be low so this is a practical decoder which is available, there are three enables, three inputs, eight outputs are active low. So if I want now to use this practical decoder in my adder circuit can I do that?

I have connected these four into OR gate to get sum, connected these into another OR gate to get carry (Refer Slide Time: 31:14) assuming these were active high but now I have learnt from the data sheet that these outputs are not active high but they are active low so they are active low outputs.

How do you change this design? I use NAND gates. What we want is, sum is m_1 m_2 m_4 m_7 but what is available is only m_1 bar m_2 bar m_6 bar and m_7 bar so we will see what S bar is. So S bar would be m_1 bar m_2 bar m_4 bar m_7 bar but you want inverse of that you want sum so sum is NAND. So connect this (Refer Slide Time: 32:40) 1, 2, 4 and 7 into a NAND gate we get sum for full adder if I connect my A B and C_{in} of the full adder of the inputs. Likewise I can connect this through NAND gate m_5 m_6 m_7 another NAND gate m_3 m_5 m_6 and carry out so carry output is m_3 m_5 m_6 etc.

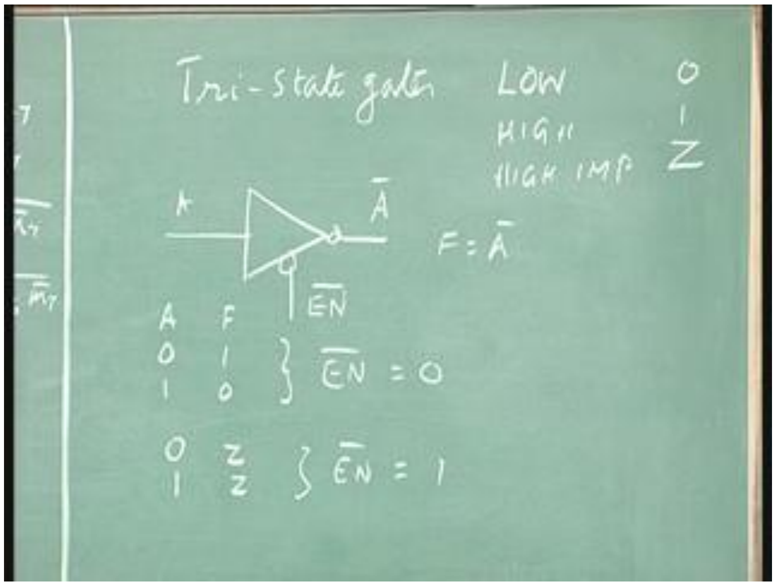
Some of these practical things you should know that's why I thought I will introduce this. when we look at a circuit logically you may be right conceptually you may be right but then it will not work because you are assuming the outputs to be active high but they are really active low or you did not know there was an enable which has not been properly connected or you did not get the proper power supply voltage in ground, you should know which pin has to be given power supply and which pin has to be given ground. So all these things you should know.

(Refer Slide Time: 34:18)


3 to 8 Decoder
74 138
C_{in} LSB
B
A MSB
$S = m_1 + m_2 + m_4 + m_7$
$\bar{S} = \bar{m}_1 \cdot \bar{m}_2 \cdot \bar{m}_4 \cdot \bar{m}_7$
$G_1 = 1, \bar{G}_{2A} = \bar{G}_{2B} = 0$
For normal operation
otherwise, outputs are in Z state

Of course you can look through the data sheet but you should know there are a few things you should look for. You don't have to remember the details but you should know to look for something. Without knowing that there is something to look for then how will you look for them so that is why you look for this? Coming back to the tri-stating what do you mean by tri-stating and how it will be active high. Suddenly we say it's active because we know only two things an input or output can be low or high where low means 0 high means 1 for positive logic, 0V and 5V, 0V and 3.3V, 0V and 4.2V or whatever but now suddenly I am introducing a third concept called the output can be 0 or 1 or a high impedance state. So what is a high impedance state? These are called tri-state gate the tri-state meaning there are three states tri-state gates.

(Refer Slide Time: 38:36)



The tri-state gates have three states. The signals of tri-state gates are three states; low, high and high impedance. So this is 0, this is 1, this is Z. There is a third state and that is why it is called a tri-state. Actually this tri-state was long ago coined by National Semiconductors a major semiconductor manufacturer and they patented it and the term was patented so other combinations should not use this term tri-state but they have to say three state buffers. I think now it must be sufficiently long and the patent must have expired so probably can use it. So you will find in text books freely tri-states whereas when you look at company literature and some other company other than National you will always be careful to write three state buffers because somebody can go to court and file a patent violation case against the user. Tri-state is as if the tri-status because the word was coined by National Semiconductors long long ago and it is history now.

So what you mean by this? I have an input and output so let us take a simple inverter, input is A and the output is A bar so this is the two state gate where A is 0 A bar is 1 so you know the truth table of this inverter F is equal to A bar, all of you know this. Now we are introducing a third concept where the output can be either 1 0 or Z. How do you know or how do you control this? When you want output to be of high impendence how you make it high impendence so I need some other extra input for that. So I have a third input called enable input that is where the enable concept came.

So this is true if enable is 0 so now this truth table becomes true or normal operation only when enable is 0 and if enable is 1 which is high like as I said enable is usually an active high it is not a rule, you may find a circuit in which enable is high so don't always assume, you look at the data sheet always. Before you use an IC practically you have to look at the data sheet. today everything is available in the web, you can go to the website of that particular manufacturer search that IC you are looking for that fellow give you the data sheet in fact there will be so much details which will confuse you so you have to look for what you want.

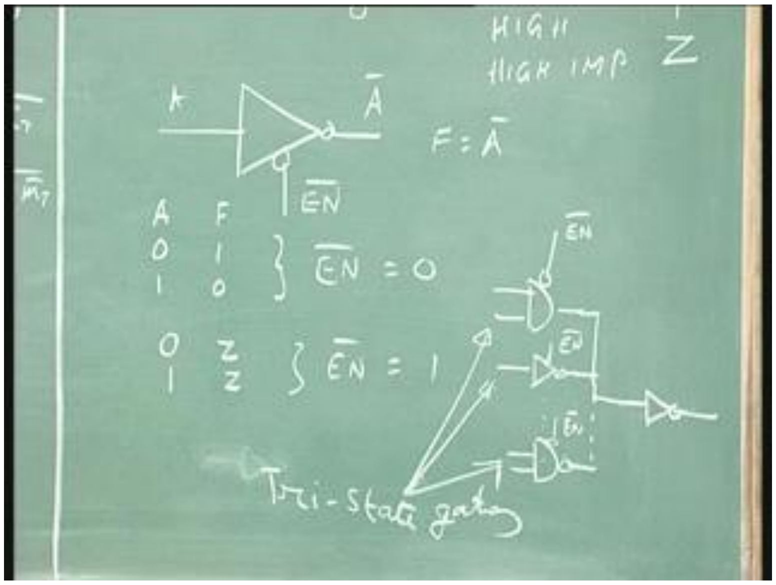
On the other hand, if enable is high so the opposite of it is disable now if A is 0 or 1 the output is……, doesn't matter what the input is the output is always high state. Now this is a very important development in the digital hardware. Conceptually there is nothing here but hardware there is a third feature so now if I put all these inverters or any other gates this is only an inverter I gave as an example, the same thing can apply to any gate. You can have an AND gate with an enable in principle of course, it may or may not be there as a component so if that enable is low AND gate works as AND gate and if enable is high whatever inputs you give the AND gate does not give the AND output but it gives you a high impedance output. That is what I said just now in decoder. It can also be there in encoder, it can be in multiplexer, it can be anywhere.

What is the idea?

Now in all these ICs there is something present like that, some other gate and I am connecting them all to a destination. These are different sources of data or control functions or whatever and this will now go to another circuit here. I want this combination but at one time there will be only be one source of data. So, at a given time I would like the output here to be determining this gate only and this should not interfere. That means this output will be inverse of this and this should be this (Refer Slide Time: 40:25). At some other time I want this output to be inverse of this and this should be this. So I may have different sources of data or different connections in a circuit all feeding to a same input, input of a same gate and I want only one of those inputs to be effective at that time so that the output will be determined accordingly in my circuit description. I can do it when I want and when I don't want it I can remove this ICs. As I said that's not a good solution, not a practical solution.

If I don't do that what will happen is even though this is high or low, all of them have to be high or low, if this is high and all of them are low where low is 0V that means practically ground. So I am trying to push this high voltage into this so at that time this is low and then this gets grounded, this also gets grounded. So whenever I try to pull this up using the output of this because this is high these two gates will put it down to 0 these are the problems. So if I have the Z feature built into all of these, all are tri-state gates, now all these gates are tri-state gates (Refer Slide Time: 41:58) then I don't have any problem. So I will control the circuit such that when the output of this has to be the input of this I will make sure these two outputs are of high impedance which has no effect practically on the operation of this circuit.

(Refer Slide Time: 42:36)



When I want this output to be the input of this circuit I will control these two outputs to be high impedance so that these two circuits will not have any effect on the output of that circuit which means electrically you are disconnecting it but physically you are not. So this is also built in many circuits. The enable inputs is a very common feature in multiplexers all ICs especially MSI and LSIs. Because these MSI LSI will sit on what is called a bus, this is called a bus (Refer Slide Time: 43:06). We generally refer to a bus as a vehicle that transports lot of things at the same time, many people can join in the bus and then go likewise here a bus refers to something where many signals can join and go. So the common bus is there. All these outputs feed into a common bus but the input will be used in only one of those common signals which are useful to the input so I have to disconnect the other inputs from the bus but physically disconnecting is not possible so I electrically disconnect it. The other inputs are not required at a given time.

Therefore when you have bus operations of these ICs especially MSI and LSI as they go more and more complex in design we will use all of these in our design because it is very complex and all of them coexist. At the same time only a few of them will have functions at a given time their roles and the rest of them will be, it is like somebody asking somebody else to shut up, lot of people are there and only one person is allowed to talk so what happens here is something similar to that.

Now we have practically covered lot of grounds in the case of MSI LSI the basic concept of MSI based design as against gate based design. The advantage is that there is a simple hardware which can replace the tons of gates with all the attendant advantages in terms of the cost, size, power consumption and everything, the speed of operations and performance. The best match may not be possible either because of the non-availability of the hardware feature or because of the dubitation of the design procedure but still it is worth it but it’s better than the other suggestions.

Then we talked about MUX and decoder which are two major components in MSI in design of these things and we talked about how to map a given hardware, a given Karnaugh Map into this hardware. We talked about how active low and active high outputs are possible and we talked about enable feature and how enable feature controls the operation. What happens when the circuit is disabled? It goes to the high impedance state the high Z state and this is the idea behind that.

The next step we will take is towards LSIs Large Scale Integrated circuits and they will have to be programmable as I said because when you have larger and larger circuits it should be more and more useful. I cannot make a simple circuit which is mappable or non-mappable so I just throw it and get another circuit. When I make a Large Scale Integrated circuit it should be available for many operations so they become programmable.

We will talk about it but before that I want to mention that MSI is not restricted to only combinational logic circuits, these are combinational logic building blocks, multiplexers and decoders. But you think of a counter a 4-bit full adder MSI, if I can make one 4-bit full adder one IC in each full adder you have about how many gates? You have two Exclusive OR gates for add and four gates for the carry and all that and if you add this is what will happen is you will get something like 24 or 25 gates it qualifies for an MSI.

So functionally small scale integrated circuits can do but then put them together it becomes an MSI. The same way a counter can be or a register can be a MSI. Even though a flip-flop will be a Small Scale Integrated circuit because it only has couple of gates or at the most four to five gates including clock. When you put flip-flops together to make a 4-bit register or 4-bit counter they become an MSI.

So we have now used MSIs without our knowledge, 4-bit full adders we have used in our design, 4-bit full adders are adder subtracters and all that type of things, we talked counters and registers, shift registers these are all MSI circuits they are also MSI based design. Of course we can also use them in some other ways we will see it later on.first let me finish the programmable aspect of combinational logics Large Scale Integrated circuits the programmable devices called PLDs where PLD stands for Programmable Logic Device. So in the next lecture we will start talking about PLDs Programmable Logic Devices. But later on we will come back to revisit the MSI sequential circuits, how they can be used in roles which we are not familiar with. Now we know how to use it in as a counter register but you can also use it for implementing the sequential logic, we will see that later.