

**Digital System Design**  
**Professor Neeraj Goel**  
**Department of Computer Science Engineering**  
**Indian Institute of Technology, Ropar**  
**Decoders-1**

(Refer Slide Time: 00:14)

**Demultiplexer (DEMUX)**

One input, multiple outputs. Input value is forwarded to selected output line based on control

X and Y could be bus

Digital Logic Design: Combinational Circuits. 23

So now we would like to design a circuit where one input and multiple outputs. Input value would be forwarded to the selected output line based on the control. How can we design this particular circuit? So let us read it again. We have one input and multiple outputs. Now the input value, whatever is the input value 0 or 1, that would be copied or that would be forwarded or that will be connected to the selected output. So there could be n number of outputs. And it has to be connected to only one of the output depending on some of the control bits.

It looks like it is a inverse circuit of multiplexer. In multiplexer we had multiple inputs, one output. And this one output was selected based on the control lines. Here it is reverse. We have multiple of these all outputs but only one input. And control is selecting that input should go where. So because it is sort of reverse of multiplexer it is called, this particular thing is called demultiplexer. In short, we also called it demux.

So if 1 is to 8 demux is to be created then this particular input is X and I have, I would require 3 control lines. How many control lines would be required? So let us say the number of outputs are N. Then the number of control lines has is to be log of N, or actually ceiling of log of N base 2.

So there are 4 outputs. Then number of control inputs would be 2. If number of outputs are 8 here then the control inputs would be 3. Similarly for more than 8, let us say 10 then number of control signals has to be 4.

So if a, b, c is 0 then this Y has to be copied to this particular, first output. So similarly second output, third output, it all depends on what is the value of a, b, c. I can also keep it like this that the  $Y_0$  would, if I need to implement this as a Boolean logic then I will say that  $Y_0$  is essentially  $X \cdot \bar{a} \cdot \bar{b} \cdot \bar{c}$ . So essentially when a, b, c is 0, 0, 0 then whatever is the X that has to be passed to  $Y_0$ . Now because a, b, c is 0, 0, 0 so this particular control signal will become 1. So whatever is the value of X, if it is 0  $Y_0$  would be 0. If X is 1 then  $Y_0$  will also be 1. Similarly  $Y_1, Y_2, Y_3, Y_4$ .

So you can see these are all minterms. So the minterm 0, minterm 1, minterm 2, minterm 3, 4, 5, 6, 7 and all of them are ANDed with X. So in this way we can see the cost of my demultiplexer is actually equal to, if the number of outputs are N then the cost is equal to N AND gates. And how many inputs to each AND gate? It depends on the number of control lines and input. So the total number of gates would be total number of control lines plus 1, that many number of inputs to of each of these AND gates. So this is the cause of my demultiplexer.

Similar to multiplexer this is 1 line multiplexer. So that means it is 1 to 8 demux, demultiplexer but the input size is a single bit. There is a good possibility that I can have a similar to multiplexer there could be multiple, so these, all these inputs and outputs could be buses. So that means there could be multiple bits present here. So again remember, so when we are saying multiple bits are present they essentially means, in the terms of implementation, this is multiple copies of this 1 to 8 demultiplexer. It is a simple copy of all of them.

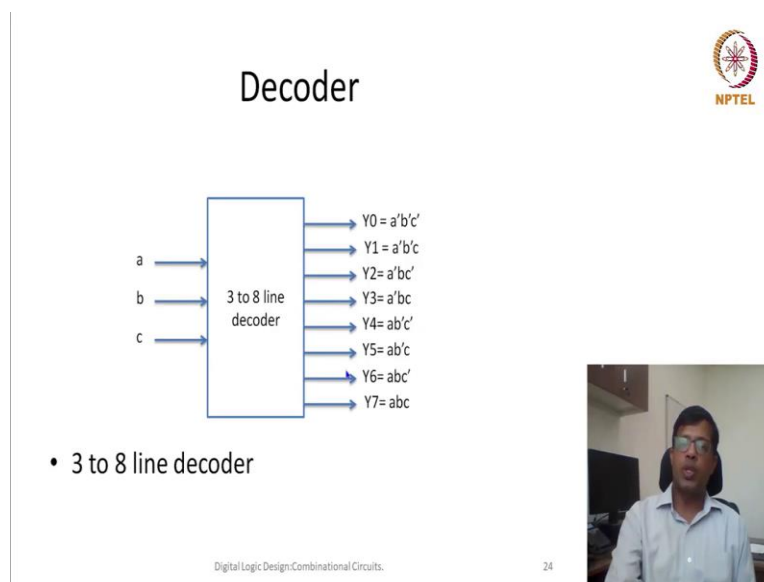
Let us say this X is 4 bit input. So in one of these demux,  $X_0$  is given as input. In other one,  $X_1$  is given as input. The other one  $X_2$  and  $X_3$  is given as input. So whenever we say vectored or bus-based multiplexer. In circuit it would be implemented as replicated copies with different inputs. One of the each bit would be given as a different 1 input to one of these demuxes. So this is how these demultiplexers can be designed and can be utilized.

Now if I have such a standard circuit, I can call from now onwards whenever we are designing, we can think of using this particular circuit if required. But where it will get utilized? Usually

wherever your multiplexer is used demultiplexer will also be used on similar places. So where it is used? So, for example your multiplexer is used in routers to connect one of the input to multiple outputs.

Now when the packet which is going forward it will be use multiplexer; whenever packet is coming back then it is to use demultiplexer. So this way these demultiplexers are used in complementary to the multiplexer. Or wherever we see that this 1 output has to be forwarded to multiple of the selected inputs, so we have to see that in which scenario this will fit in.

(Refer Slide Time: 06:52)




Similar to this demultiplexer there is also another popular circuit, commonly used circuit which we call decoder. So in this decoder, the difference between decoder and demultiplexer in way is that if this particular line X is 1, if this X is 1 then based on the value of a, b, c, one of the output is going to be 1.

So, in other words, let us say if a, b, c the input is 0 0 0 then Y0 would be 1. If the input is 0 0 1 then Y1 would be 1. So, in other words, this decoder will have only one of the output as 1. All others as 0. And it depends on what is the input pattern. Now there could be N output and the number of input would be  $\log N$  base 2. So if there are 8 outputs that would be 3 inputs. And based on the 3 input, one of the lines will be selected. One of the line would be, would be high. All others would be 0. So this is also quite a popular circuit and you can say it is very similar to demultiplexer. But the cost is less.

Now cost of decoder, I can clearly say that it will have  $N$  AND gates. Each AND gate would have number of inputs equal to the number of control inputs. So this is called 3 to 8 line decoder. And you can see the other purpose, you see you are decoding, so because now you have  $N$  numbers. You wanted to select which particular number it is. You have  $N$  LEDs, let us say. Which LED to lit? Then only let us say if there are 16 LEDs and only 4 bits are sufficient to know that which LED should lit. So this way, this decoder can help us in identifying which number it is. Each number is represented as a different wire.

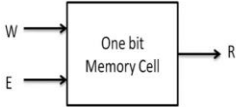
Now to see the, to see the utilization and how these encoders, how these decoders, multiplexers, demultiplexers could be utilized; let us take one, one design example. So with this design example let us try to understand which one to be used, how to be used.

(Refer Slide Time: 09:42)




## Design exercise

- We have a hardware that can store one bit data – we call it memory cell
  - Memory cell has two inputs, E and W. If E is 1, W gets written into memory cell
  - Memory cell has one output R, which reads the content of cell



```
graph LR; W --> MC[One bit Memory Cell]; E --> MC; MC --> R;
```

Digital Logic Design: Combinational Circuits. 25



So in this design exercise let us try to understand what this question is. So we are assuming that we have a special kind of a hardware. The hardware can store one bit of data. And let us call this hardware as a memory cell.

Now you can see this memory cell could be an electronic circuit; could not be an electronic circuit. It is saving one bit of data. You see whatever you call, memory, so some of these memories, they store data in form of magnetic, magnetic polarity, some optical disc store in terms of light, whether there is cavity or there is no cavity. And if we are storing this value as a


charge, if there is no charge then 0; if there is a charge then it is 1. Then you can call it as memory cell.

So let us not going into the implementation internal of this memory cell but we say that there is a hardware which can store one bit of data. How to read it? So this particular memory cell will have one output R. So this R, if we see it always reflect whatever is the, whatever inside value is there. If inside value is 1 then R will always be 1. If inside storage is 0 then R would be 0. And then, as a input, so we should be able to write this memory cell. And let us say we have these two inputs which help us in writing. If E is 0, so if E is 1 whatever is the content of W that will get written to memory cell. But if E is 0 then nothing happens.

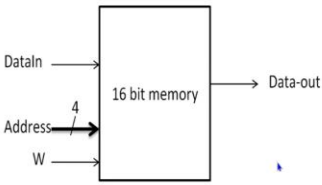
So that means whatever is the content of W will be written into this memory cell only and only if E is 1. Otherwise W does not, will not be written to this memory cell. So essentially it is a control. E is a control signal or enable signal which says that whether this memory cell has to be written or should not be written. So this way we should be able to control what to be written to these memory cells, what not to be written. So this is the basic entity we are assuming that this is given as a input.

(Refer Slide Time: 12:16)


**Design exercise**



- Design a memory which has 16 cells, which can be accessed using 4 bit address
- If W is 1, write DataIn. Data-out always reads



```
graph LR; DataIn --> Mem[16 bit memory]; Address[4] --> Mem; W --> Mem; Mem --> DataOut[Data-out];
```



Digital Logic Design: Combinational Circuits. 26

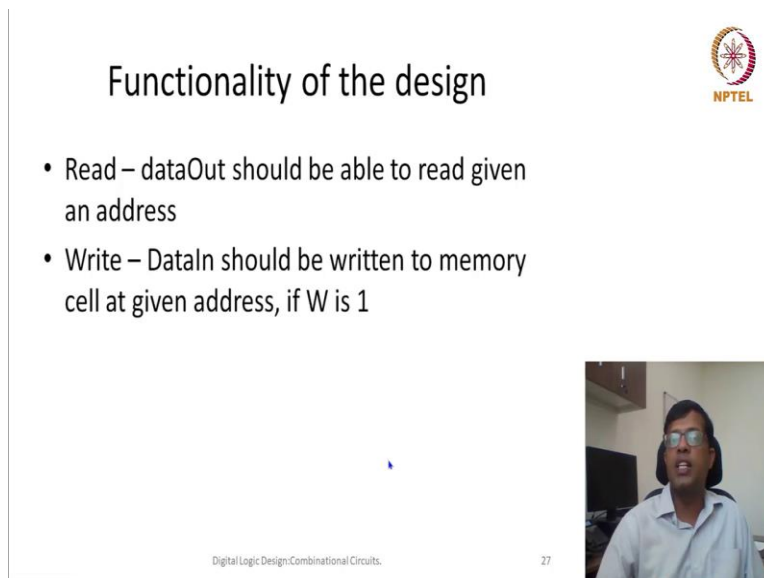
Now the design exercises, we want to design a memory which has 16 such memory cells. Now because there are 16 cells we can access them using 4 bits. Let us say the bit address is 0 0 0 and I can access cell number 0. Similarly bit address 0 0 1, then 0 0 1, then I can access cell

number 1. So along with this address we also have, if W is 1 then Data In would be written to that particular memory cell to which this address is mentioning. And otherwise, this address will always give out whatever is written in that particular address, in that particular memory cell.

So, in other words, if my W is 1 then Data In will go to the particular address. If W is 0 then whatever address is given that Data Out will read out from that particular memory cell to whatever this address is pointing to.

Now I want to design such a circuit which will do this functionality. So whenever we are designing a circuit it is always better that first we clearly draw such kind of a diagram which clearly segregate what is input, what is output. Here we know that Data In is 1 bit input, Data Out is 1 bit output, and address is a 4 bit input. W is also 1 bit input. And inside this there are 16 memory cells which has the description like this, that which has two, which has two inputs, each cell has two inputs W and E, and one output R. Now how to design this? So for designing the first step is we have to see what would be the input and output, what is going to be their parameters like what are the sizes of input, output. The second is to describe the functionality clearly.

(Refer Slide Time: 14:29)



The slide is titled "Functionality of the design" and features the NPTEL logo in the top right corner. It contains two bullet points: "Read – dataOut should be able to read given an address" and "Write – DataIn should be written to memory cell at given address, if W is 1". A small video inset in the bottom right shows a man speaking. At the bottom left, it says "Digital Logic Design: Combinational Circuits" and at the bottom center, the number "27".

- Read – dataOut should be able to read given an address
- Write – DataIn should be written to memory cell at given address, if W is 1

So here we see there are two functions this particular circuit is performing. One function is Read. So that means Data Out will read given an address. So given an address it will go to a particular memory cell and whatever is the value there it will read out, it will go into Data Out. It should be

connected to Data Out. While the other functionality is Write. So that means Data In should be written to a memory cell. Which particular memory cell? Whichever is given by the address and Data In would be written if W is 1.

Now given this functionality let us think out that if I need to design how should I go about it? Since we have studied this multiplexer, demultiplexer, decoder then mind is forcing us to think that possibly we have to use any of them. Maybe we were supposed to use tri-state buffers also. So, now let us first consider write. So in case of write what do I want? I have one input Data In and that particular input Data In; I have one particular input Data In in and that particular input Data In has to go to any of the one of the 16 memory cells based on address.

(Refer Slide Time: 16:11)

**Writing to the memory**

Using Demux                      Generate E

DataIn will be connected to all W0 to W15

Digital Logic Design: Combinational Circuits                      28

So it looks like that I, it is good to use. The first thought is coming that I should use demultiplexer. Now in case of demultiplexer what would happen that my Data In would be given as input and my control would effectively be the address, it is a 4 bit address. And whatever are the outputs, 16 output of my demultiplexer, so all of them I should be connecting to W0, W1, W15 up to W15.

What does it mean? It means that if my address is, let us say, 0 0 0 0; 0 0 0 0, yes, and then whatever is Data In that should go W0. So if Data In is 0 then W0 will become 0 and Data In is 1, W0 will become 1. What would happen to all others? W1, W2, W15, all of them will also be 0 if the address is not selecting that particular W. So, or that that particular memory cell. So it that

particular memory cell is not selected then those inputs would be 0. And Data In would go to W0 or that particular cell if that particular address is 1.

So there are two issues here; that we still have to generate E signal or enable signal of each memory cell. The second is that because of this demultiplexer, using of demultiplexer, all of these W values are either 0 or 1. It depends on the address. So it looks like a duplicate effort. So because I also have to first of all generate my enable signal, the second thing that whatever is my Data In is going to a particular W it becomes sort of unnecessary. So rather than that, what if I do, so then we have to discard this and what we have to do is we focus on generating enable signal rather than W signal.

So what we will do is we will connect this W, W of the final interface, memory interface, 16 cell memory interface. The W, we remember is, if it is 1 then it would be written. If it is 0 it will not be written. Now this W be used to generate the enable signals, and address is used to a demultiplexing. So address is working as a control and W of the input is going to generate the enable signal. And Data In is given to, connected to all W0 to W15 of our memory cells.


So this way, what we are able to do that this control signal or enable signal is generated using the write, using this 1 to 16 demultiplexer. Now let us think that can we use decoder here? Decoder perhaps, it can be used if we use this address. If we use address as a input to decoder then there would be 16 output of course. And now those 16 output has to be ANDed with the write so that if write is 1 then only that particular enable signal will be 1 otherwise it has to be 0.

So that is how we have to think rationally. We have to see that which particular circuit can be used so that my end requirement can be met. Now similarly this enable signal would be generated using demultiplexer where input is W. And Data In is connected to all W0 to W15 signals.

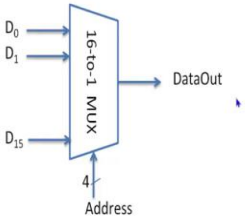


(Refer Slide Time: 20:40)


**Reading from the memory**



- Using Mux



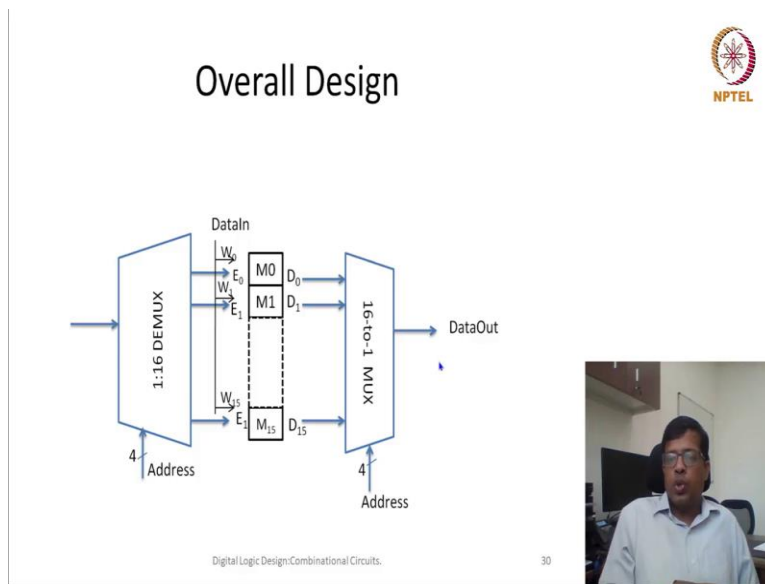
Digital Logic Design: Combinational Circuits. 29



What about read functionality? Read, we can see that, in read what we are doing that data D, output from memory cell is always available. So we simply have to read based on the address specified, and we can always read it. So simply we can have 16 to 1 mux where address can work as a control and D0 to D15 could be connected to input of my multiplexer. And then I can say my output is Data Out.

Here the specification was more or less clear. There was less confusion because it was clear that I have to get the output from one of those memory cells. So there are 16 inputs. And out of that I have to select one. So I can use multiplexer here.

(Refer Slide Time: 21:36)



Overall the complete design would look something like this, that there has to be 16 memory cells. Overall design would be there would be 16 memory cells, let us say, starting from M0, M1 to M15. Now first of all, all the Data In signals, like there would be, if we remember that my memory cell has three inputs; W, Enable and D. So all the W signals would be connected to Data In. Now the question is, if I am writing Data In to all the W signals then to whom this particular data signal would be read, would be written?

Now this W would be written to only that particular cell where Enable is 1. So Enable would be 1 so there would be Enable signal. These Enable signals are essentially coming from, from my demultiplexers. So this demultiplexer address is given and the write of the input is given; write, the control signal is given as the demultiplexer. So all the Enables would be determining that which particular write will be; which particular Data In would be written to memory cell.


Now D here, all of them are connected to my multiplexer. And then address is determining that which particular one would be given as Data Out. Let us say there is a further condition that I have to read only if my control signal is 0. In that case I can have this multiplexer which would have either Enable signal as a input. So, but we can always discard. We can say that this Data Out we are not going to read. So we can have another AND gate here that this Data Out would be read only if my data, my read signal is 0.


So this is how like, whenever a question, whenever a design question is given we have to think over it. We can break it into smaller functionality. And we can work on all those functionality independently to come up with the complete design.

(Refer Slide Time: 24:11)

**Summary**

- Learnt about tri-state buffers
- De-mux
- Decoder





Digital Logic Design: Combinational Circuits. 31

So the summary of this lecture is that we have learnt about how the signals are connected, what could be the solution if we have to connect multiple of the outputs to the same input. So that was suggested as a tri-state buffer. And then we have seen that how demultiplexers can be utilized, or how decoder can be utilized.

So, one more thing which we have learnt that in our digital circuits, our final output should be either 0 or 1. If it is X then it is undesirable. If it is Z or high impedance then also it is undesirable. So we have to design a circuit any, that all my all outputs, all my intermediate signals should be clearly 0 or 1. We have to avoid the value of unknown or high impedance as much as possible. Thank you very much.