

Digital System Design
Professor Neeraj Goel
Department of Computer Science Engineering
Indian Institute of Technology Ropar
Lecture 49
Memory

Hello, everyone. Today is the last lecture of this module and today we are going to discuss about memories.

(Refer Slide Time 00:23)



Where to store large data?

- Flip-flops?
- Array of flip-flops?
- Data is the key – we need to store huge number of bits
- => Memory
 - Semiconductor memory
 - Magnetic memory
 - Optical memory



Digital Logic Design: Sequential Circuits.

So, the first question which we are going to answer in this today's lecture is, if we need to store large amount of data, huge data, then where do we store? Should we store it in flip-flops? So, you can see that one flip-flop, should we store it in latches or flip-flops? So, one latch is a cross coupled, can be designed using NAND gates or NOR gates plus, it should have set and reset capabilities. And then flip-flop is an edge triggered device. So, where should we store this data?

Flip-flop could be a good option if we also require this data to be accessed synchronously and data to be stored synchronously, then flip-flop could be a very good choice. However, if we need to have, need to store large amount of data or a huge data, then flip-flop may not be sufficient because number of flip-flops we would require is would be huge.

Then one option could be we can have an array of flip-flops. But let us say the data is even larger. So, you have, let us say one Megabyte of data. So, that means 10^6 bytes. So, then array will become so, large that the delay of accessing flip-flop would itself be very,

very large. So, then we would need to have a memory cell structures which could be simple, where the access time to read the data or write the data, could be very less.

So, if the number of bits which we would like to store is huge, and it is actually huge in many cases. You see, when we would like to store, when you would like to store files, when you would like to store songs, movies, the data, in terms of number of bits, not only it would reach in Megabytes, sometimes it is Gigabytes or sometimes it is even more. So, the amount of data which we need to store, can become very, very huge, very, very large.

So, the flip-flop cannot be very efficient structure if we are storing the data for a, we are storing large amount of data. So, wherever we are storing this data, those structures, those units can be called memory. Now, if that memory where we are storing the data, it could be flip-flop, it could be made of flip-flop also, it could be made of some other entities.


Now, if those memory structures are made up of silicon, any form of silicon, then we call is semiconductor memory because they are made up of some transistors or combination of transistors or some different type of, could be different type of transistors, may not be conventional ones. And all of them could be categorized as semiconductor memories.

There could be other options also, because you see, what we are going to store is 0 and 1 and to store 0 and 1, we just require two states of the matter. So, that is why we can use not only semiconductor but characteristics like voltage or current. There could be other characteristics of matter which can be utilized to store the, store the data or store bits, 0 and 1.

So, Magnetic memories are also, quite popular so, you take two magnetic, either you say it is magnetic or you see what is the direction of magnetism. Based on that you say whether it is 0 or 1. So, there, we just have to say two properties. Whether it is non magnetized or whether it is magnetized. So, you will see, or either you have seen or you will see in other courses that in magnetic, there is a Hysteresis loop. So, because of that Hysteresis loop many of the material can be categorized in two of the states.

So, magnetic properties are used quite popularly to store data. So, we have magnetic tapes or floppy drives and even the hard disk, most of us, which we use the hard disk which we use, HDD, Hard Disk Drives, so, they are all magnetic in nature. The other properties also people have utilized like optical properties, whether it is transparent or non-transparent. So, our CD drives or DVDs, they are all using this optical property to store 0 and 1. They are all memories. But we are focusing our discussion today on the semiconductor memory.


(Refer Slide Time 05:39)



Related technology terms

- RAM – Random access memory
 - Contrast: Sequential access memory
- ROM – Read only memory
 - PROM
 - EPROM
 - EEPROM
- NVRAM

Digital Logic Design: Sequential Circuits.



In semiconductor memory, there are further, couple of other terms which you will see. One term which you would have popularly seen is RAM, Random Access Memory. What does this RAM mean? That means that your memory will have a lot of data, you can consider it like an array of elements. Now, if you can access any of the element at randomly, that means it is a Random Access Memory.

In other words, so, let us say, you have address x and to access address x , you need not to access address $x - 1$ or $x + 1$, then you can call it Random Access Memory. So, that means any of the address can be independently addressed. This random access memory does not say that all the element locations, all the locations will have, will require same amount of time to access but it just says that any of the addresses can be accessed randomly. In contrast, there could be sequential access memories.

So, historically, when people have invented memories or they have started working on memories, initially, there used to be magnetic tapes. So, these magnetic tapes are very similar to, if you would have seen tape recorders, so, there is a, there is a brown color tape that used to rotate around. So, now, if you want to access anything in between, you have to roll all the tape to that particular point, then only you can access it. So, those, when your data is stored on such kind of a sequential device, it would be sequential access memory, it will not be random access memory. You cannot access randomly any particular point. You have to go through a sequence.

So, today, most of the memories are random access. So, either be it hard disk or be it your storage device or your semiconductor memories. Most of the semiconductor memories, they

are anyway, random access but optical drives or hard disk drives, there is a, there is some kind of a sequentiality which we have to maintain.

So, another term which is also, quite popular is ROM, Read Only Memory. So, Read Only Memory is a type of memory where you can only read. So, this, we have already studied in one of the previous lectures so, there could be various types of Read Only memories which is, programmable ROM, electrically programmable ROM or electrically erasable programmable ROM.

So, that means, if ROM cannot be written, that will also, make it useless. So, that is why these ROMs, although mostly, they would be used for read purpose, but using some special procedures or using some programming methods or maybe some different, special equipments, they can still be written. So, that is why it is called programmable ROM. So, that they can be programmed, maybe using a different equipment but when they are plugged into device, mostly they would be read, they will not be written.

Slowly, we are going into an era where these ROMs are also, being programmable or being written by these devices and these devices also become a commodity hardware. So, you would have seen that couple of years back, CD or compact disk or the optical memories used to be only, read only memory. That is why they used to be called CD-ROM. But then you also, get read, write capability that you have a CD drive, which can write onto these ROMs as well.

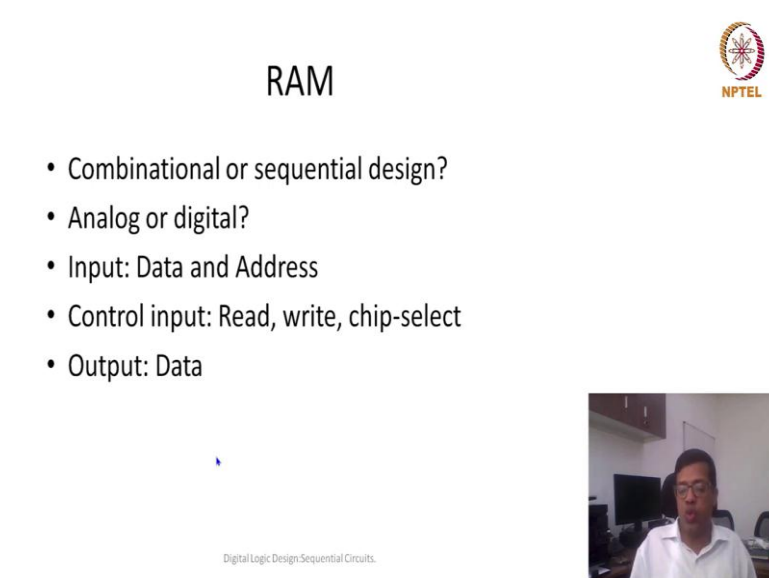
So, another characteristics of this Read Only Memory is that when you plug off the power, there is no current, there is no power, then also, content would remain onto this. So, that is why, these read only memories, like CD-ROM or DVD-ROM, they are also, used for storage. So, for example, if you see your computer, you have one memory which is storage, which, where all your operating system, all your files are stored. On the other hand, there is something which is read, RAM or your DDR RAM. So, that RAM will only work when power is there but otherwise data would be stored onto your storage.

So, in that way, usually these random access memories, it is again, usually, I am using this word usually because most of the times read only memories, RAMs the way they are designed, they would, they will have no data when power is off. So, that is why there is another special category of RAM which is called Non-Volatile RAM.

Non-volatile RAM, NVRAM is a RAM which is non-volatile. So, that means when power will be plugged off, then also, content would remain in that RAM. So, it could be randomly

accessed, plus it is non-volatile in nature. So, you can also, treat NVRAM as a some sort of a read only memory that you can read through it, there would be some special procedures that would be employed, so that we can write onto it.

(Refer Slide Time 11:21)



The slide is titled "RAM" in a large, bold, black font. Below the title is a bulleted list of characteristics:

- Combinational or sequential design?
- Analog or digital?
- Input: Data and Address
- Control input: Read, write, chip-select
- Output: Data

In the bottom right corner of the slide, there is a small video inset showing a man in a white shirt speaking. The NPTEL logo is in the top right corner, and the text "Digital Logic Design: Sequential Circuits." is at the bottom center.

So, now, we would be focusing further our discussion only on the RAM. Now, in the RAM one of the, yeah, the first question which I would like to ask is that whether your RAM is a combinational design or a sequential design? So, let us look at the fundamentals. Combinational design means given an input, your output is going to be same.

Here, in case of a RAM or a memory, what is an input? Input may be the address, input would be the data input and the output would be the data output. Now, in case of combinational design, given the set of inputs, your output is always same. But here, your output depends on what is stored inside the RAM. So, that means, it cannot be combinational.

Now, is it a sequential design? Yes, it is a sequential because it depends on what you have stored earlier in previous inputs. So, it depends on previous input, what you have stored in the previous input would determine what would be the output at current address or, yeah.

So, it is a sequential design, but it is not a synchronous, it may not be synchronous sequential design, it may not have a clock. It may not be synchronized, the output may not be synchronized with clock or input may not be synchronized with clock. So, it may not be synchronous sequential design, but definitely, it is a sequential design.

The other question which looks quite obvious but still, I would like to ask, whether it is an analog design or a digital design? Because we are always storing 0 and 1, so, it appears that it


is a digital design. But when we look at the implementation of most of these RAMs, semiconductor RAMs, then you will see, because we would like to have highest speed, because we would like to have highest performance, so, internally, they may be designed as analog units.

So, there may be amplifiers, there may be continuous signal which is being monitored. So, although interfaces are digital, but internal implementation could be analog for these RAMs. So, that is why for this course or for digital design, we consider a cell or one unit where the memory, where the data, one bit of data is being stored. We can abstract it as one unit. Implementation on that unit could be analog or could be digital in nature.

So, what is a typical input? As we have discussed, the input would be using a data and address and there would be additional control inputs where we will say read, whether we would like to perform a read operation or write operation. The other control input could be chip select. Whether we are, we actually want to perform read or write or not. So, if chip select is 0, that means we do not want to perform read or write, but if chip select is 1, then based on read and write values, we would either perform read or perform write. Now, output would be data.


Now, when would, you see, data is both, input as well as output. So, when data would be input, when we are doing a write operation, when we are doing a read operation, then data would be output. So, it is a very specific example where data could be a in output where you can write as well as you can read from the same wires. So, now, these are the two operations which you perform, either you perform read or you perform write operations.

(Refer Slide Time 15:02)



Memory size terminology

- Bytes: 8 bits
- Size : in terms of bytes
 - Kilo Bytes = 2^{10} Bytes = ~1000 bytes = 1KB
 - Mega Bytes = 2^{20} Bytes = ~ 10^6 Bytes = 1MB
 - Giga Byte (GB) = ~ 10^9 Bytes
 - TeraBytes (10^{12} B), PetaBytes (10^{15} B), ExaBytes (10^{18} B)
- Memory addressing
 - Word addressable
 - Byte addressable



Digital Logic Design: Sequential Circuits.

RAM



- Combinational or sequential design?
- Analog or digital?
- Input: Data and Address
- Control input: Read, write, chip-select
- Output: Data

Digital Logic Design: Sequential Circuits.



The other characteristics about the memory is the, yes, size. So, memory is characterized by your input and output. Additionally, it is characterized by what is the size, how much content, it can store. So, when we talk about content which it can store, it is typically characterized by how many bytes it can store.

So, bytes is usually written by a capital B whenever it is, whenever written in a short form and if you are writing bit, then it is written in small b. So, you can distinguish, if for example, somebody has written 1 K and small b that means one 1 kilo bit but if it is B is capital then it is, it means that it is 1 kilo byte. And one byte, we know has 8 bits. So, the typical units for size could be Kilobytes. That means 10^3 is to power 10 bytes. And 10^6 is to power 10 is 1024 so, we can approximate it to 10^3 or 1000 bytes or can also, write it as 1KB.

Similarly, the other units are 1 Megabytes, 10^6 is to power 20 bytes or 10^6 is to power 6, sorry 2^{20} is to power 20 bytes or 10^6 is to power 6 bytes. You can also, write it as 1MB, capital M and capital B. Further scale could be Gigabyte means 10^9 bytes. Terabytes, 10^{12} is to power 12 bytes, Petabytes, 10^{15} bytes, Exabytes 10^{18} bytes.

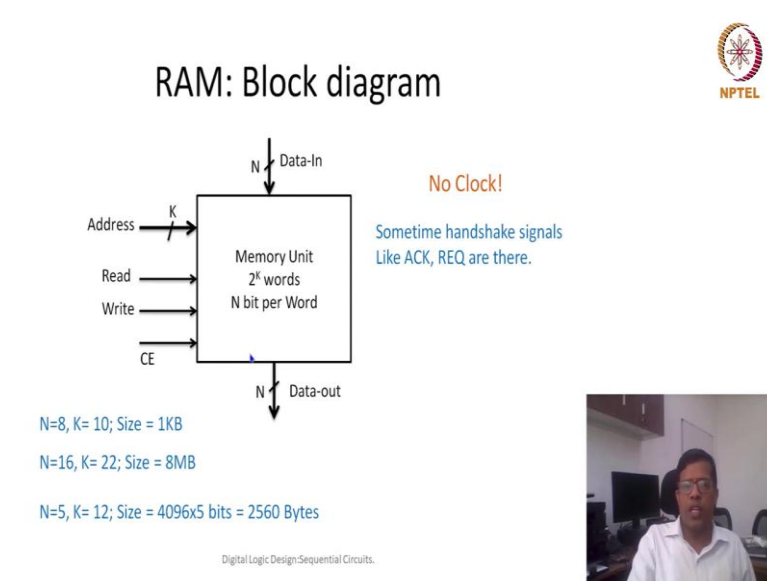
So, some of these we have to remember, at least you, when you are buying a computer then you would have seen Kilobyte, Megabyte, Gigabyte, as well as 1TB is used quite frequently. But slowly as we are moving toward cloud data centers, 1 Petabytes or 1 Exabytes will also become a normal amount of storage that we would require in coming years or coming days.

The other point is also, important that how do we say, whether memory addressing, whether it is word addressable or byte addressable. Both of these are possibilities. For that we also, have to understand what is word in terms of memory. So, word is how many bits are, how many bits are stored in one address or one row. So, is, or how many, sorry, how many bits

can be accessed whenever we are saying, you say that data is an output, data is an input. So, how many bits can be accessed when data is out or data is in, that is called Word.

Now, this word could be, Word size could be 4, 8, 16, 64, et cetera. Now, although Word size could be more than a byte, but sometimes memories could be byte addressable. If Word size is less than byte, then it is definitely Word addressable but if memory, Word size is more than a byte then also, you can address it using a byte addressable. So, what does that mean? Let us see with one example.

(Refer Slide Time 18:36)



Now, by looking at input and output this could be a block diagram where you can have a data input, you can have a data output as, and the other inputs are address read and write. Now, when we are saying data input, we are saying N bits of data can be written and N bit of data could be read out. So, that means the Word size is N-bits.

So, if address is K then 2 is to power K words could be stored in this memory and they can be accessed. So, that means we are assuming here, it is word addressable. Now, if 2 is to power K words could be stored and each of the word can have N bits, what is the total size of the memory? 2 is to power K into N, that many number of bits and if we divide it by 8, that many number of bytes.

Now, in this module, there could be an additional, option input which is chip enable or circuit enable or you can call it whatever, it is the enable circuit. If this enable is 1, then only it would be performing its functionality. If it is 0, then even data out should be high impedance. There should not be any data out.

So, now, let us look at the sizes. So, one more thing that here, you see that there is no clock. So, we are not considering this random access memory as a synchronous sequential design, it is only sequential design. The other thing is that because if it is a larger in size, let us say Gigabytes or maybe even Megabytes, the latency, the access time, could be quite larger than the normal clock cycle, the clock cycle of your combinational design.

So, then it may not be, so, when you are saying reading, or writing, so, you have passed on the address at one clock cycle and maybe it can take 10 to 20 clock cycles to generate the data out. So, because, further this data out could be generated at variable amount of time. So, if it is variable amount of time, then we have to have some sort of a handshake signals which we will say that now data out is available, please read it. When it has been read then only a next address could be given.

So, this kind of a handshake signals could be given. And if these handshake signals are given then, then we can call this particular kind of a memory as asynchronous memory, they are not synchronous, but they are opposite, they are asynchronous. They are always out of sync with our clock.

So, now, let us talk about sizes. So, if you say, let us say N equal to 8 and if K equal to 10, then what would be the size? K equal to 10 means 2^{10} words and N equal to 8 means byte, 1 byte, one word equal to 1 byte so, that means, so, if N is equal to 8 and K equal to 10, then the total size is 1 Kilobyte.


So, let us say N equal to 16 and K equal to 22. If K is equal to 22, total number of words is going to be 2^{22} . That means 4 Mega Words. And now, what is the size of one word, it is equal to 16. So, that means 2 bytes. So, total size would be 8 Megabytes.

Now, here one point which we can stress upon, that if this memory is not word addressable but byte addressable, then the total number of bits required for the address would be, this K value will become 23, sorry, 23, not 22. So, that we can address each byte, although data out could be 16. So, although, still, word size is going to be 16 but because if it is byte addressable then we can address each and individual bytes also.

Now, let us take one more example, let us say N equal to 5 and K equal to 12. If K is equal to 12, total number of words are going to be 4096 and 5 bits are there. Now, total size would be 4096 into 5 bits and if I convert into byte then I have to divide it by 8 so, then it comes out to be somewhere around 2560. So, this is how we can calculate the sizes based on what is the


address and how many, this is the maximum size, given an address, this is the maximum size, it could be sometimes smaller than that. So, this is how, we can calculate addresses.

(Refer Slide Time 23:50)



RAM Implementation

- Memory cells
 - Flip-flops
 - SRAM: Custom design using 4-8 transistors
 - DRAM: Custom design using 1 transistor
- Memory cell input/output
 - Data in, Data out, select line, read-write
 - Select lines, Data lines



Digital Logic Design: Sequential Circuits.

Now, how do we implement these random access memories? Random access memories as we discussed earlier, it could be implemented using flip-flops but the only issue with the flip-flop is that the number of transistors required, the area required size is going to be huge. So, one flip-flop would have at least 4 AND gates. So, because of 4 AND gates, the total amount of circuitry involved, it could range from 8 to 10 transistors.

Now, because memories are huge, memories are huge means they are storing huge amount of data and based on the amount of data, sometimes we have to reduce the cell size. That is why static RAM, SRAM is a very popular type of RAM implementation. Here, it is designed using, usually using 6 transistors. So, these 6 transistors are implemented like a SR latch where you have a back to back connected your invertors and then you have set and reset transistors which can help you to set any value or reset any value.

Now, this SRAM is much faster than flip-flop because it does not involve any kind of a redundant or unnecessary transistors and it is, it also requires lesser area than flip-flops. So, it is lesser costly than flip-flops, array of flip-flops but still faster than flip-flops. So, the only challenge with SRAM is because it requires some sort of analog components like, it requires amplifiers or pre-charge circuitry which is analog in nature so, that is why when you had to design, it is called custom design, so, then you have to, you cannot design it using direct gates. So, the process, implementation, fabrication is also, different.

The other possibility then your requirement even increases further, then you can design, one memory cell using a single transistor. That single transistor work like a capacitor. So, if you charge it, it will remain 1 for certain amount of time, let us say 1 millisecond or couple of milliseconds. So, you have to recharge it regularly. But the density of this memory is the highest.

It would be slower in nature because every time you have to keep on refreshing, keep on recharging, also, whenever you read any data from this, then again you have to recharge. So, because of all those things, DRAM, dynamic RAM is slower but has the highest density. So, also, cheaper in price.

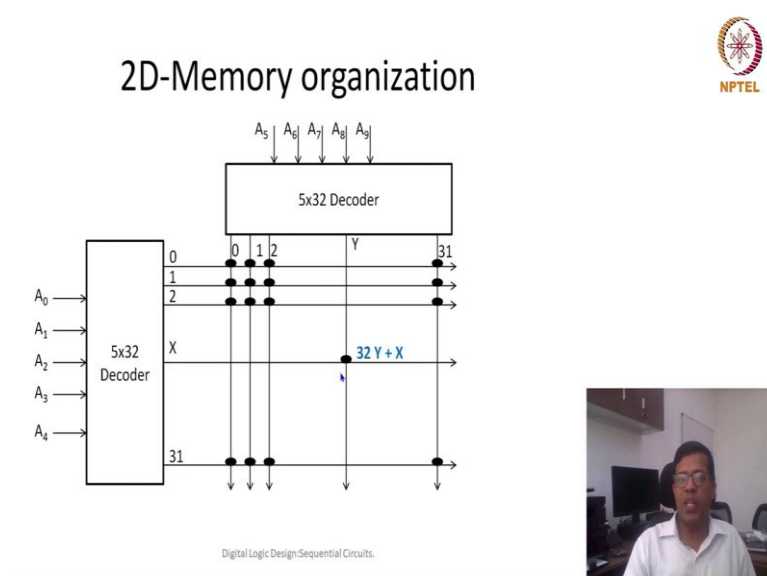
That is why, in our computers, the fastest memory is going to be flip-flop, slower than that there would be SRAM and the memory which is largest in nature, that means your main memory, which is in terms of like, 1 Gigabyte or 8 GB or 16 GB, that is usually DRAM or dynamic RAM.

As we have seen earlier, so, each, this memory would have an input like, total memory would have an input of, you will give address, you will give data input, data output and then you will have read and write, etcetera. But each memory cell will have only 4 inputs. Data input, data output, select line and read write. If select line is 1 then that cell would be active otherwise that cell will not be active.

Further, based on this SRAM cell and DRAM cell, we further refine our input output only to two lines. One is select line, that means if select line is 1 then cell would be active otherwise cell output is going to be high impedance. Another is data line. So, this data line could be, data could be used for input as well as for the output.

Whenever we are using it for data input, then we require some sort of a buffers, so, that those write would have a good amount of charge, good amount of voltage and whenever we are reading, then we will try to sense what voltage is there in that particular cell. So, these data lines work as both input as well as output lines.

(Refer Slide Time 28:45)



Now, let us see that how we would implement that. So, this implementation we have already discussed in one of our lectures when we were doing combinational modules. So, we implemented our memory as one dimensional array and there were, we were using D multiplexer to read the data. Or in other sense, we were using Decoder and then we were trying to access all the data.

Now, that single dimensional, one dimensional memory organization could become less effective if the size will grow. So, for example, if you would like to implement one megabyte of memory, then you would require a decoder which has 20 to 2 is to power 20 lines. So, this huge complexity could be made simpler by making a 2 dimensional memory organization.

So, in case of 2 dimensional memory organization, what we do is, we do two things. One, we will divide the address into two parts. One is high part, high end addresses, address bits and the low end address bit. And there we have two different decoders. For example, here, what we are taking example is a 1 Kilobyte of memory.

So, in 1 Kilobyte of memory we have divided the addresses into 5 and 5 so, these are the lower addresses, these are the high, higher significant addresses. So, this high addresses, when we are, we would use decoder here as well as here. So, now, when we are using decoder here, what you will see, there would be 32 outputs, here also, we will have 32 outputs.

And then, each of these dot, would actually correspond to an AND gate. So, of this is 1 and this is 1, then only select line would be 1 for this particular cell. And if cell, if select line is 1 then only that cell would give an output otherwise it would be in high impedance state.

So, by having this two dimensional structure, what we are doing is, we are reducing the complexity of our decoder. So, you see, for example, if I was taking example of 1 Megabyte of memory, so, in 1 Megabyte, there would be 10 to 1024 decoder at one side and 10 to 1024 decoder at the other side which has much lower complexity than 20 into 10 is to power 6 decoder. So, our complexity has reduced.

The other benefit of this kind of a 2 dimensional structure is that, your now memory cells are also organized in two dimensional fashion. Because memory cells are organized in 2 dimensional fashion, now, they form, when we will physically implement them, they will form a square structure rather than a long structure.

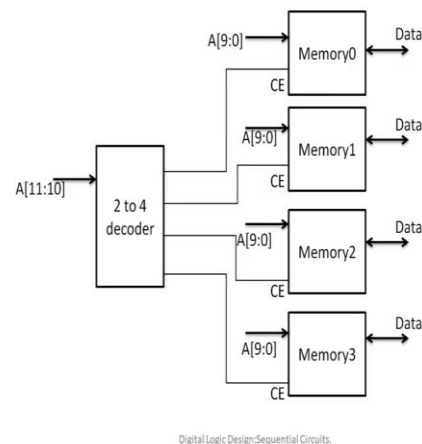
So, this square structure is also, beneficial because now, the excess time would be equal, more or less equal for all the cells and also you will see that when you would like to buy a chip so, it more the rectangular in nature, better it would fit onto your final PCB boards. So, here, for example, here, these are the lower end ones. I can call these as row decoders. This one, I can call as a column decoder.

So, when row and column of a particular cell would be 1, when both of them would be 1, for example, x is 1 and y is also, 1, then only that address would be active. So, here, that particular address I can say, it is $32 \times y + x$. So, because these are the higher end addresses, and these are the lower end addresses. So, being lower, lower end addresses, they would always correspond to modulo 32 of the address and this would be, whatever is the address, address divide by 32 would be the A5, A6, A7, A8, A9.

So, now, this is how 2 dimensional memory organizations can be there and which can help us in designing more regular structure as well as lesser cost in terms of area and delay. The other advantage, you can also, see that because both of these decoders are working in parallel, so, they would be faster. Now, the critical part, delay is going to be this decoder plus 1 AND gate and then accessing this particular set. So, this is one way to mitigate the complexity by designing 2 dimensional memory organizations or memory structures. The other way to mitigate the complexity is designing it in a hierarchical manner.

(Refer Slide Time 33:58)

Hierarchical Memory Organization




So, when you are saying designing it in a hierarchical manner, then what you do, you can have a two level decoders. So, for example, I can have four different memory modules. So, all of these 4 different memory modules, can take lower bit of the addresses and higher bit of the addresses would be taken by decoder. This decoder will say that which particular memory has to be enabled.

Now, you see, if this address range is from 0 to 1 Kilobyte, then the, this top, top two bits would be 0 0, so, that means this would be, Memory 0 would be selected. When the address range is 1 Kilobyte to 2 Kilobyte then the input here is going to be 0 1, so, the first memory 1 module would be selected because this is going to be the output. Similar is the case with the Memory 2 and Memory 3 module. Now, address, lower bits of the address would be given would be given as an input to all of them, similarly data would also be given at all of them.

Now, what would happen if this particular memory module is not selected? Because this memory module is not selected, so, my data would be in high impedance. So, because this data is high impedance, this is also high impedance, this is also high impedance and only one of the data is 1, so, I can simply connect all these wires to give the data.

There is no OR gate required, there is no other, any other circuit which is required. So, this is, this is the fine grain implementation. So, we are not going into further detail of the implementation which is based on cells or which is based on recharge circuitry. If you are interested, I can point you to the links where you can study about those details.

(Refer Slide Time 35:58)



Verilog Code

```
//Memory size is 64 words of 4 bits each
module Memory(Enable, ReadWrite, Address, DataIn, DataOut)
input Enable, ReadWrite;
input [5:0] Address;
input [3:0] DataIn;
output [3:0] DataOut;
reg [3:0] DataOut;
reg [3:0] Mem[63:0];
always @(Enable or ReadWrite or DataIn or Address)
    if(Enable) begin
        if(ReadWrite) DataOut = Mem[Address];
        else Mem[Address] = DataIn;
        else DataOut = 4'bz;
    end
endmodule
```

Digital Logic Design-Sequential Circuits.



Now, with this, mostly we are done with the design or basically organization of Memory. Now, let us say if we have to model the same memory in a Verilog code. How do we write? So, we will write a module, memory and then we have these inputs, Enable, ReadWrite Address, Data input and Data output. And then we have to specify that my input, so, basically, Enable and ReadWrite are single bit input because we are going to design 64 word of memory with 4 bits in each word.

So, the total number of addresses are 64, so, that means 6 bits required. Those 6 bits we can write as 5 colon 0 Address. And now, each word size is 4 bits so, we can say the data input DataIn has, is an input with 4 bits, 3 colon 0. Now, the output is also 4 bit, because my word size is 4 bit so, output is also 3 colon 0.

Now, we will talk about this reg, definition of DataOut. Now, internally, this memory has to be stored in some sort of an array so, this array is going to be 63 colon 0 because there are 64 elements, each element is of 4 bits so, we are writing, we are declaring that register 3 colon 0. So, that means this is the word size and this is the array which we are going to, where we are going to store our memory. So, a single dimensional array.


Now, we can write a behavior code for this Verilog memory. Now, we can say whenever Enable is there, ReadWrite is there, DataIn is there or Address is there, that means any of them will change, then we are going to see what is the, what would be the, there would be change in output. Because, it is, from the interface it can be treated like a combinational circuit.

So, when any of these inputs will change, my output will change. If Enable is there, then only I can check whether ReadWrite is there or not. When Enable is 1, then only we can check about ReadWrite. If ReadWrite is 1, so, that means we are going to read the data. So, where we are reading, we are reading into DataOut and the address, whatever is the given address, that we can take from the, from this memory type and then that can be given as DataOut.

Now, because this DataOut is written in an Always block, because DataOut is written in Always block so, I need to declare this DataOut as a, as a reg data type. So, if ReadWrite is 1, if ReadWrite is 0, that means I would like to write. Write which particular thing? I would like to write into this address DataIn value so, I am saying, memory address and then DataIn, whatever is the, in DataIn, that would be written into this particular address.


However if Enable is 0 that means my output has to be in high impedance so, I can specify my output as high impedance if my Enable is 0. So, this is how we can write Verilog code for a memory module and with this, I would finish the lecture, close the lecture.

(Refer Slide Time 39:35)



Summary

- Memories are integral part of digital system
- Used to store data, control, configuration



Digital Logic Design: Sequential Circuits.

And I can summarize this lecture as two takeaway points. One, that these memories because they are integral part of the digital system so, they are implemented in various ways. Different implementations are there, we in generically we call them Random Access Memory, then specific implementation could be SRAM or DRAM or it could be an array of flip-flops.

Now, we have seen some of these implementations and we have also, seen some way, because these memories are going to be huge, large. So, there could be some, we studied at

least two different techniques to mitigate the complexity if the size is large. One is by organizing the memory as a two dimensional structure. The other is by hierarchically organizing the memory sets.

The other thing which we will see in, in our coming lectures is that these memories are also integral part of our digital system because they would be used to store the data, used to store the control and can be used to store the configurations and can be stored to, store the input. So, with this, I would like to close. Thank you very much.