

(Refer Slide Time: 22:03)

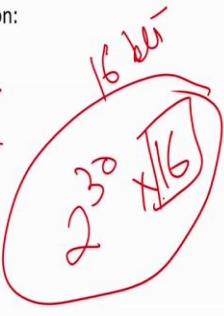
**Semiconductor memory configuration**

Total size of the data is  $2\text{GB} = 2^{31}$  Bytes. (as 1 G Byte =  $2^{30}$  Bytes)

So,  $2\text{GB} = 8 * 2^{31}$  Bits =  $2^{34}$  Bits.

c) Double Byte (16 bit) in each memory location:  
Size of data bus=16 bits  
No. of addresses= $(2^{34})/16=2^{30}$   
Size of address bus=30 bits.

d) 32 bits in each memory location:  
Size of data bus=32 bits  
No. of addresses= $(2^{34})/32=2^{29}$   
Size of address bus=29 bits



Again the same thing we have taken now it is a double byte. So, why do we actually have different type of a memory organization? the idea is that sometimes if you make the memory size too wide then what it may happen that you may wasting your size that means, say a single instruction takes about a 16 bits or 8 bits. But you can never implement a single instruction or explain the meaning in one or two bits. So, if you have a two bit organized memory then to find out the meaning of a meaning to find out what is the meaning of a valid word or what do you mean means valid instruction you have to read 8 or 10 memory locations. Then you have to assemble them and then you have to find out the meaning out of it that is not a very good idea.

So, generally say we are taking a double byte that is 16 bit. So, may say maybe you are going to fit the whole instruction in that. So, just read one word and your job is done. But for example, if I have a 64 bit word then what will happen then one big word will have one or two or three instructions then again if you read you will be reading three instruction at a time and then again partitioning it, so that is not a very good idea basically.

So, what we will try to do people try to keep the memory organized in such a fashion that it at least holds a single instruction or may be half of the instruction something like that. So, at least by reading two words you can make up the meaning out of it; nobody wants to do it something like that they want to put two or three instructions in one word that does not look good in a nutshell basically. All these are very broad ideas exceptions are there, but in general

terminology I would like that all my instruction and data be meaningful in a single word, but sometimes it is not possible. Because you all know that to represent a character you require less number of bits and if you have a value long-long int or long-long float, it takes more number of bytes. But generally it is never like that that you have to read ten memory locations to find out the meaning of a single variable may be two memory locations are enough to understand what is the meaning of that whole word or the whole data?

So, in this case they are saying that double bite so that means, each word is having 16 bits. So, what will be the number of addresses  $2^{34}$  byte, 16 that is  $2^{30}$ . So, the address bus size is 30 bits. Data bus size will be 16 bits because your 16 bits you can do together. Similarly we can discuss for 32 bits also. So, in this case there will be 29 bits. So, what will be the byte organization or the byte organization of the data or word organization of the data that size will be your data bus. And total memory size divided by that will give you the number of addresses it's very simple idea.

So, either if you represent memory as  $2^{34}$  bits, then immediately you have to tell what is the byte or what is the organization of the memory like 2 byte double byte is the size. Or you can write  $2^{30} \times 16$  that is also that will tell you I have  $2^{30}$  memory locations and each memory location size is 16 bit that also will be sufficing. So, anyway is ok for us.

(Refer Slide Time: 24:56)

**Memory Operation**

Now, when instruction LOAD Acc, 0003<sub>H</sub> is executed the values in the system buses connecting the CPU to the memory are as follows:

- Address Bus: 0000 0000 0000 0011 (i.e., 0003<sub>H</sub>) → MAR
- Control line: 1 (i.e., read from memory)
- Data Bus : 0001 0010 (i.e., contents of memory location 0003<sub>H</sub>) → MBR

The figure given below illustrates the values in the system buses when instruction LOAD Acc, 0003<sub>H</sub> is executed

Now, with this basic configuration of the memory now we will say that for example now will as the main goal of this module will to understand what is an instruction how it executes

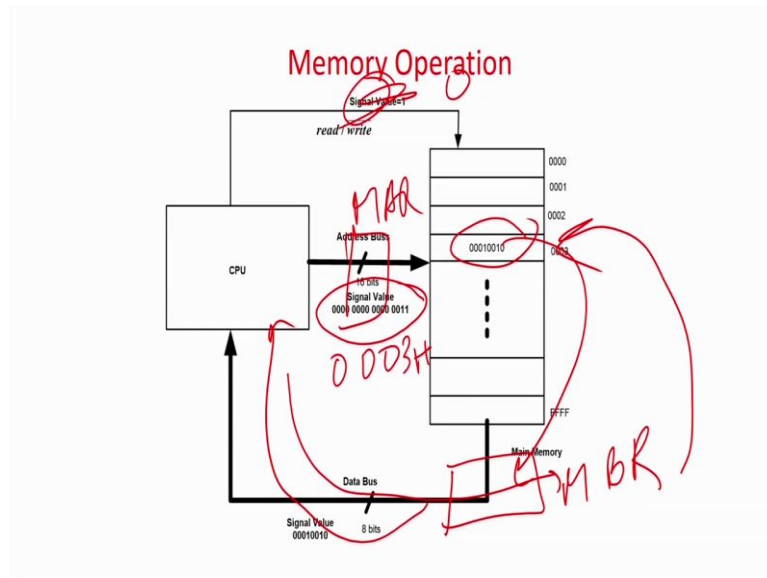
etcetera. So, now, we will see basically how a memory read or write is an instruction a simple instruction load accumulator 0003. So, what is an accumulator for the time being in the last module also Prof. Deka has given a basic idea of what is a register etcetera. So, for us you can understand that accumulator is one of the most primary register because more or less all the combinations are done on a register on a register which is the accumulator for timing being just understand that it is the core, one of the core registers.

So, load accumulator-3 that means what some data from the main memory whose location is 0003 has to be loaded into the accumulator that is the job. Again one thing has to be emphasized that whatever memory location we write in an instruction are all for the main memory, we do not write generally the address for which is the hard disk of the external memory. If some data is not available in the main memory then it has to be loaded from the hard disk to the main memory, but that is the story we are going to look at later, but not emphasis of this course.

So, load accumulator 0003 that is how it will happen. So what it happens is that the CPU will generate this is in the address bus that is this address I want read. So, this will be fed to the address bus again the control line has to be made 1 because we have to read from the memory, then what happens is basically then well while executing the instruction the memory data the data in the memory.

Let us assume that the content of main memory is 0001 0010, it is an 8 bit word or a byte addressable memory. So, this value the 8 bit value will be loaded into the data bus, but it will come through the register which is called the memory buffer register. The address bus this is value will be written in the memory address register. So, memory address register will have the value 0003. So, now, the memory will know that this data whatever available in my memory location number four that is 0003 has to be dumped to something which is called the memory buffer register ok. And then why it has to write why because why it has to write to the MBR because this is a read operation that is somebody the CPU is reading this ok. And then this memory buffer register input the value into the memory to the accumulator register.

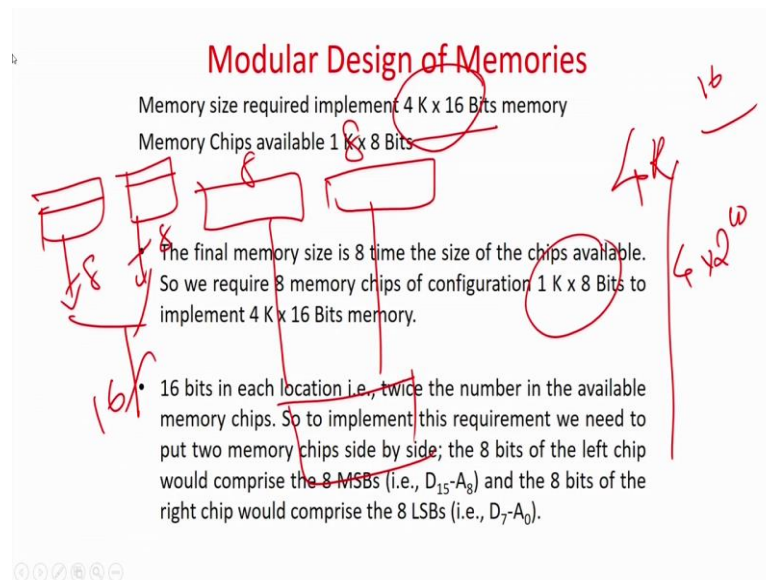
(Refer Slide Time: 27:18)



So, now, let us look at it in the figure. In fact, we have not drawn over here that is the memory buffer register, and there will be a memory address register, memory address register will be there. So, say this is the content. So, you are giving the value of the signal value that is equal to 0003 hex to memory buffer register. So, this one will be address, this is one. So, it will be read the value will read to the memory buffer register from memory bus will come to the CPU that is it has to be written to something called the accumulator that is what is the memory operation cycle.

So, again I can do the reverse. So, if you say for example, in this case if you want to say that I want to store accumulator value 3, so in that case everything will be same except in this case the value will be 0. So, whatever value of whatever value the accumulator will write to the memory buffer register will be returned to the memory location, just a reverse by this one. So, this is in a nutshell again the read and write operation of the memory.

(Refer Slide Time: 28:08)



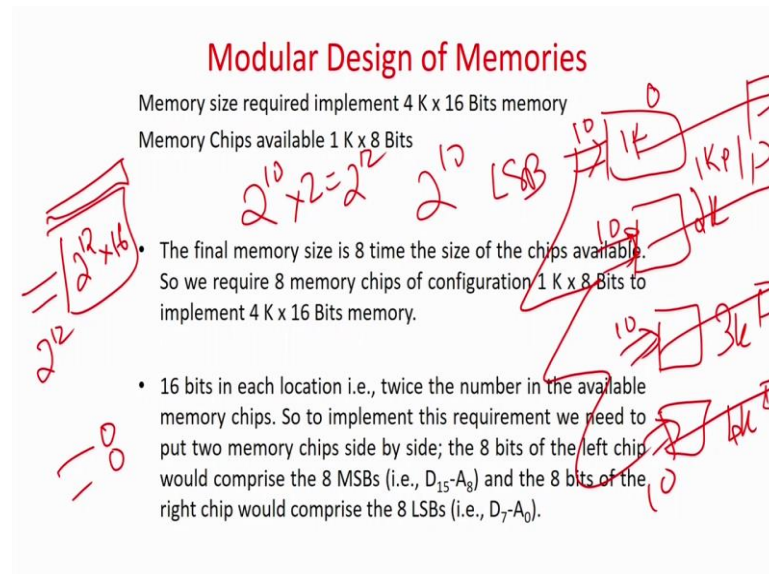
So, now, let us think that we have a RAM maybe we all know nowadays we know that we are purchasing RAM in terms of slot. So, we purchase 1 GB RAM slot four then we put four slots together, maybe we have 2 GB RAM cards and put in this slot; that means, memories are modular. So, nobody actually purchases maybe a 16 GB RAM in one chip generally they are maybe brought down into multiple levels like 1 GB - 8 cards, 4 GB - 2 cards. So, we require modularity of the memories. So, in fact why it is required because then only you will be able to have flexibility otherwise for each design you may have to make tailor made memory which is not actually a very good idea because I require generality and then I want to plug in and plug out to do that.

So, for example, say that I want to take a  $4\text{ k} \times 16\text{ bits}$  memory that is I require a 4 k that this has to be 4 that is  $4 \times 2^{10}$  and the size is 16 bits. Fine I can make a chip like this for you, but then again I have to design fabricate everything for you that is not a very good idea, but say for example, what are the chips I am having in the market we say  $1\text{ k} \times 8\text{ bits}$ . So, that is a byte addressable memory is available and only 1 k memory is available. So, using this basically I have to design my required memory, so that is you have modularly plug in and do slight changes and the design should be ready all memories basically follow a modular design.

So, let us see basically if this configuration is given to you, so how do you do it? So, in fact, you see 16 bits and this is 8 bits. So, naturally you have to put 2 together there is no other way we can do it this 8 bits and this is 8 bits. So, the data bus will read parallel, very simple idea

that we will have two 8, 8 bits here one 8 bit here and then both of them will be reading, there will two data buses. And finally, we will merge the data which will be a 16 bit this is 8 bit and this is 8 bit, this part is very simple and it can be easily done. Ok now, the next part is that, but we have only 1k memory size, but we require a 4k memory size so that is what is the trick so obviously, 4 and 1 very simple we will say that I will put four in a row.

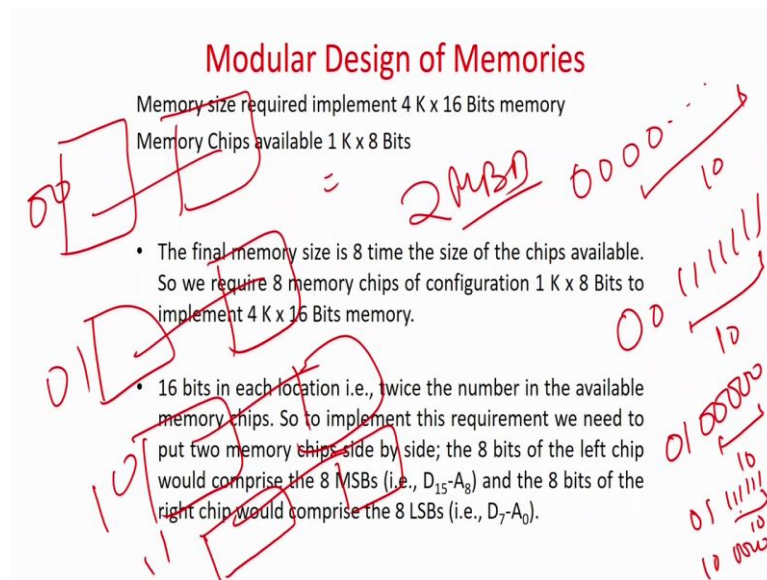
(Refer Slide Time: 30:22)



That is I will put 1, 2, 3 and 4 in a row. So, this one will be 1 k this one will be 1 to 2k this one be 3k and this one will be 4k this 0 to 1k then 1k + 1 to 2k and so forth that will be done, but now 1k means what  $2^{10}$ . So, the address bus this one is 10 bits, the address bus this one is 10 bits, this is 10 bits and this is 10 bits. So, 10 bit address bus is there. And you will have 4k. So, it is a 4k memory means  $2^{10} \times 2$  that is equal to  $2^{12}$ . So, in this case the address bus size is 12 for the overall memory, because overall memory is  $2^{12}$  that is  $2^{12} \times 16$  that is the overall configuration. Ok so, this one is 16 because we have now put two in series and in fact there are two other we need this to draw because the that will happening these two are happening in parallel.

But now I have to access these four in a serial manner, because here the number of addresses is  $2^{12}$  because that is 4k, but each individual block is size of 10, 10, 10. So, what I will do this 12 bit memory bus the LSB 10, I will connect to all, that all the memories I will be accessing with 10 LSBs of the address bus will be connected to the four memory, then 2 LSBs are available. So, if I say that if I now I have put a logic that when I want to select this row when I

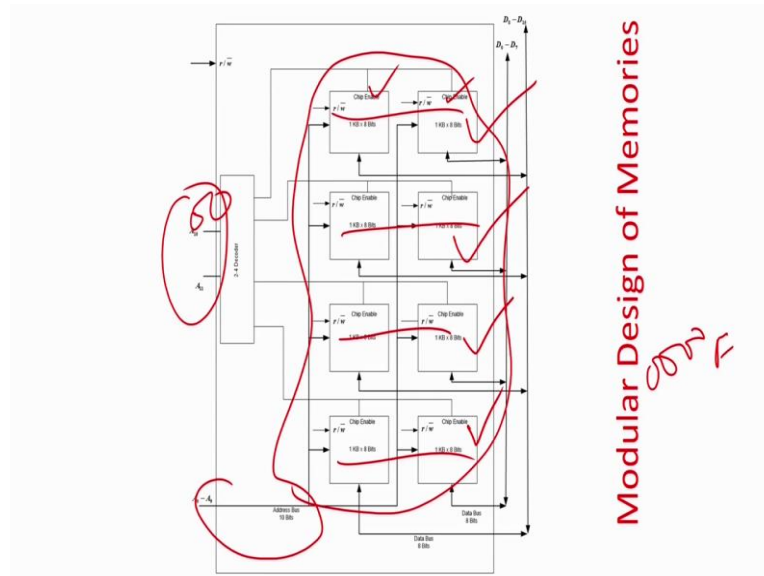
have to select this row when I have to select this row and when I have to select this row the idea is very simple. So, if as I told you the first memory cell of the first memory block will be accessed from 0000 that is from 0 to 1k, the next will be from  $1k + 1$  to  $2k$ .



So, in a very similar manner, what we do is that we connect the ten LSBs of the memory block of the address bus to all these two cells. And the last two bits MSB of the address bus which is now 12 bits will be connected to this four memory cells in a very nice manner that is there is something I have not told you as of now there is something called a memory select.



(Refer Slide Time: 33:55)



I will just show you the configuration is something called chip enable. So, if you look at it, it is something called the chip enable over here. So, what do you mean by chip enable that is it actually tells you what is if the chip can be switched on and off like for example, let me this is the basic configuration. So, again I will tell you what is the configuration is. So, in this case if you look at it as I told you these are 16 bits. So, this one will be one, this is the two, two chips are in sequential. So, this is 8 bits, 8 bits will be accessed.

Now, as I told you, if you look at it, this is the one address bus which is the LSB ten bits. So, A0 to A9 of the address bus for the whole chip. So the individual size is  $1k \times 8$ . So, the address bus size is ten, but for the overall the address bus size is 12. So, the last two bits A11 and A12 are going to control which chip, which row has to be selected as we have seen if the MSB two bits are 00 so this one has to be selected. And if it is 01 this row 10 and 11 and all the other bits row has to be all the other row must not be selected at that time.

So, what the how can we implement? A very simple implementation is a 2 : 4 decoder because each chip has something called a chip enable. So, if the chip enable is 1 that chip is operating otherwise it not giving any output. So, I put a 2 : 4 decoder and I give the MSB two bits as inputs. So, now, if it is 00 that means, only this line will be one and all other lines will be basically 0, because a decoder acts in that fashion. So, 00 means the first line 01 second line and so forth will be only 1s and all others will be 0.



So, if the address is say all 0s; that means, I have to access the first row of the first set of chips that is the first row of the or I should say the first word of the first two rows of chips, so all 0s. So, of course, last ten bits are 0, LSB ten bits are 0. So, this two are accessed and the MSB two bits are also 0; that means, this line is basically a 1. So, all others are 0. So, this chip enable and this chip enable are 1 making these two chips on and all others will be all three other rows will be 0. So, they will not give any output. So, definitely you are going to access this row.

Again let us take the example that all 1s, that is the last row of the chips has to be accessed. So, all 1 means every bit is a 1 of the address bus. So, in fact all one means last ten bit. So, this two bits will be last row will be accessed last row will be accessed last row will be accessed and last row will be accessed why because ten bits are all ones, but now which among these four row will be selected. So, again as all 1 is the address. So, this MSB 2 bits are also 11. So, MSB bit 11 imply that, this row is 1 and all other rows are 0. So, this row this row and this row will be the module off manner; and only this two rows will be in a module on there is chip enable, and these two will be giving the data.

So, in fact if you look at it, so this is what is the configuration that this ten bits bold line if you look at it, this bold line will access all the chips at a time because the chips are of size 1k. So, if we have to access any row say zeroth row, second row, first row, so all the corresponding or same rows for each of this chip will be access simultaneously. So, if you say that 0000f. So, all the first rows of all the elements will be selected. But now you have to check correspondingly if the last MSB that whether I want to take from this row or this row or this row or this row. So, if this is 00. So, only this chip will enable on and this row will be accessed and so forth.

So, this is actually what is something called a modular design. That is you take different modules and arrange them using a decoder. So, if the if you are giving a byte addressable memory and you have blocks of size 8, one is enough if you have byte addressable memory blocks and you want to have 32 bits as the address as the word size here, so  $8 \times 4$ , so 4 memory blocks has to be put in series. And then depending on the size row you have to multiply by that number.

(Refer Slide Time: 37:59)

## Modular Design of Memories

- 4 K of memory locations i.e., four times the number in the available memory chips.
- Need to put four memory chips arranged in a columnar fashion. For locations 0 – 1023 we use the first chip, 1024-2047 we use the second chip and so on.
- Use the chip enable line of the memory chips. Only when the chip enable line is 1 the corresponding memory chip is functional (i.e., ON), else it is OFF and do not send/receive data through the data-Bus.
- 4 K locations imply that the address bus size is 12 bits. Among these 12 bits, the LSB 10 Bits (i.e.,  $A_0$ - $A_9$ ) are connected to the data Bus of the memory chips. The MSB 2 Bits (i.e.,  $A_{10}$ - $A_{11}$ ) are connected to 2-4 Decoder, whose outputs are connected to the chip enable lines of the memory chips.

So, if I write in a formal manner say for example, you have a block size is 1 byte and this size is also say 1k, 1 kB memory or 1 MB or whatever you want to. Then very easily understand that if this is 1 byte, so if you have a 8 byte or a 8 byte or 16 byte or whatever is the size then you have to increase in that row in the row size. That means, if you have the say the unit size is  $x$ ,  $x$  is the word size of the memory blocks available, and if you want to implement a memory whose size overall size is say  $3x$  in the row then you have to put three together. Of course, if the size is  $y$  and if you require a memory whose size is  $3y$  then you have to put 3 in the column size very straight forward. Only thing is that we have to appropriately put a what I told you have to appropriate the 2 : 4 whatever is required as the decoder which will select the appropriate row, this row or this row or this row depending on the memory accessed. So, all the LSBs are generally fed to all the memory chips and the MSBs are kept to the decoder so that is what is the idea.

So, if I go over this, so in this case all the things I will detailed out that 4k is the size of memory. So, we need to put 4 such in the columns in the rows, so we require four. So, one is two four. So, four such will be there in the number of rows and the column size is double. So, you require 2 blocks in the column and then the LSB ten bits will be connected to the data bus of all the memories and the MSB two bits are connected to 2 : 4 decoder and this is what is the memory organization. So, whatever I told you can read it form the theory which is given in the PPT, and it will be module will be clear.

(Refer Slide Time: 39:36)

## Modular Design of Memories

- Need to read memory location  $FFF_H$  (i.e., 4095<sup>th</sup> Location)
  - The address bus has the value 1111 1111 1111. The 10 Bits LSB of the data bus selects the last locations of all the memory chips. But, MSB 2 Bits selects the 4<sup>th</sup> line of the decoder (because  $A_{10}=1$  and  $A_{11}=1$ ) thereby enabling ONLY the memory chips of the last row.
  - The data bus outputs 8 MSBs (i.e.,  $D_{15}-A_8$ ) from the left chip and 8 LSBs (i.e.,  $D_7-A_0$ ) from the right chip of the last row.

Now, they have given that they have given an example that how do you want to read the memory location  $FFF_H$  that is forty ninth row. So, in this case if you look at it, so all 1s will be the memory and then the LSB 8 bits will be all once. So, last row of all memories will be accessed and then if you look at it the last two bits MSB also 11; that means, the fourth row will be selected and that is it. And in this case two memories are in series, so 16 bit will be the data bus.

(Refer Slide Time: 40:07)

## Questions and Objectives

- Q1: What are some of the main semiconductor memories which were used for designing the main memory? Also, discuss their pros and cons.
- Q2: Assume a hypothetical CPU connected to the main memory whose size is 64 K x 8 Bits. Now an instruction LOAD Acc, 0003H is executed which loads the value present in memory location 0003H to the accumulator in the CPU. Explain how this operation is executed by illustrating the values in the system buses connecting the CPU to the memory.
- **Application: Demonstrate:**—Demonstrate the use of semiconductor memories which are used for designing the main memory of a computer.
- **Comprehension: Describe:**—Describe how to address a particular memory location of the main memory.
- **Comprehension: Explain:**—Explain the connection of the main memory to the processor through system bus.
- **Analysis: Determine:**—Determine the size of a memory module. Explain the size of a memory location - byte addressable, word addressable, etc.
- **Comprehension: Explain:**—Explain the memory read and memory write operation.

So, basically with this we come to the end of this unit on basic black box accessing of a memory. In this case, we have not gone into the details of what how the memory is implemented etcetera. We have tried to show that on this from the perspective of the CPU if you have to give the address, you have to read, you have to write, what is the basic data basic infrastructure required, how do you give the address, what are the registers involved, how it is read and write only in the this prospective we have seen. Because for this unit this module and the next module where we will be mainly understanding how a code executes, this much black box view of the memory enough for us to understand.

And next to next module, module will be a fully dedicated on memory where you will learn the inner details of how a memory is implemented, how it is interfaced, how it is synchronized etcetera; so towards the end basically if we look at the questions and objectives. So, there are four questions or you can have the four questions are put over here. And we can just see that it will obviously, meet the objectives, like first question is what are the what are some of the main semiconductor memories used and discuss the pros and cons. Like we have RAM we have registers we have DDR RAM. So, of course, it will cover the objectives where we are able to demonstrate the use of semiconductor memories which are used in the main computer that is the first question, we satisfy the objective that is we are able to answer the question which objective will be satisfied.

Similarly, we are given a hypothetical memory configuration and then we have a asked how it can be interfaced etcetera with the system bus. So, in fact this will be covering of the objective on how to address a particular memory in the memory, how explain how the connection of the main memory to the CPU or system bus etcetera.

(Refer Slide Time: 41:49)

### Questions and Objectives

Q3: The capacity of a main memory is 2GB (Giga Byte). Provide the size of Data Bus and Address Bus with proper explanation if the main memory is arranged as:

- a) Bit Organized
- b) Byte Organized
- c) 16 bit in each memory location:
- d) 32 bit in each memory location:

Q4 We need to design a memory of configuration  $4\text{ K} \times 16\text{ Bits}$ . However we have memory chips of configuration  $1\text{ K} \times 8\text{ Bits}$  and standard digital circuit blocks like decoder, multiplexes etc. Explain the design and also show the values of the system buses involved for a given memory write operation.

- **Application: Demonstrate:**--Demonstrate the use of semiconductor memories which are used for designing the main memory of a computer.
- **Comprehension: Describe:**--Describe how to address a particular memory location of the main memory.
- **Comprehension: Explain:**--Explain the connection of the main memory to the processor through system bus.
- **Analysis: Determine:**--Determine the size of a memory module. Explain the size of a memory location - byte addressable, word addressable, etc.
- **Comprehension: Explain:**--Explain the memory read and memory write operation.

And this is the question on if there is a memory of such a size if it is byte organized, bit organized, how it can be accessed and what is the basic architecture; what is the data bus size, what is the address bus size. So, I think it can be easily satisfied by this to what do I say the objectives and so forth that is the basic modular design of a memory which we have seen. So, this is the part of it. So, in fact we are seen that by covering this unit and basic very few basic questions basically we are able to satisfy all our objectives. So, in fact this brings us to the end of this unit that is a black box view of a memory which will be mainly required to understand how basically the codes can be executed.

Thank you and in the next unit we will be mainly again looking at how instructions are designed, how they are accessed and so forth.

Thank you very much.