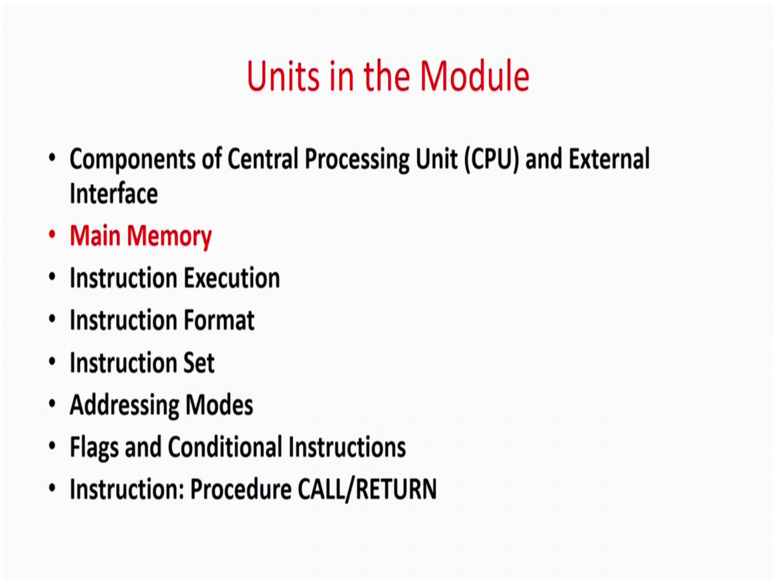
Computer Organization and Architecture: A Pedagogical Aspect
Prof. Jatindra Kr. Deka
Dr. Santosh Biswas
Dr. Arnab Sarkar
Department of Computer Science & Engineering
Indian Institute of Technology, Guwahati

Lecture – 08
Main Memory

(Refer Slide Time: 00:35)



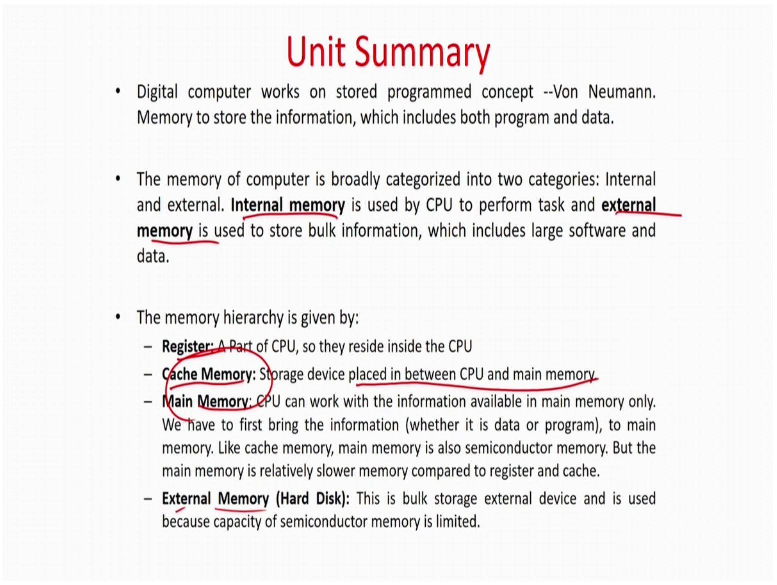
Units in the Module

- Components of Central Processing Unit (CPU) and External Interface
- Main Memory
- Instruction Execution
- Instruction Format
- Instruction Set
- Addressing Modes
- Flags and Conditional Instructions
- Instruction: Procedure CALL/RETURN

So, welcome to the second unit of the module on addressing mode instruction set and instruction execution flow. So, in the last unit, we have seen; what are the basic components of the CPU, and the external interfaces, and the basic memory structure, and how they are all integrated. Now, we will go to the in this you as this what you as we all said that will covered the basic idea of how a instruction it is execute, what are the instruction set format etcetera. So, we are now going to the second part that is required to understand the execution of the memory, execution of the instructions and what are the different instruction modes that is the main memory. Because as you know that we have all studying about von-Neumann architecture, so in that case all the data and the instructions are present in the memory.

So, even if you is want to understand about instruction execution the format the sets and different addressing modes and so forth, you have to know the idea how they are all organised in a memory. So, the second unit of this is the main memory which you are going to look today.

(Refer Slide Time: 01:26)



Unit Summary

- Digital computer works on stored programmed concept --Von Neumann. Memory to store the information, which includes both program and data.
- The memory of computer is broadly categorized into two categories: Internal and external. **Internal memory** is used by CPU to perform task and **external memory** is used to store bulk information, which includes large software and data.
- The memory hierarchy is given by:
  - **Register:** A Part of CPU, so they reside inside the CPU
  - **Cache Memory:** Storage device placed in between CPU and main memory.
  - **Main Memory:** CPU can work with the information available in main memory only. We have to first bring the information (whether it is data or program), to main memory. Like cache memory, main memory is also semiconductor memory. But the main memory is relatively slower memory compared to register and cache.
  - **External Memory (Hard Disk):** This is bulk storage external device and is used because capacity of semiconductor memory is limited.

So, now as you have to told that the course is delivered in a pedagogical manner or pedagogical paradigm, so we will see what we are going to see in this unit that is the unit summary. So, as I told you we are all looking at the von Neumann architecture concept which will consist of both the data and the program and the data.

So, you have to know about the idea of what is a memory what is the main memory, and what is a secondary memory etcetera, and how it is addressed, and what is in a nutshell how it the or in a black bus how the CPU interfaces with the memory. There will be in fact a detailed whole module on the memory design in which will know the more internals of the how the memory works, how it made up of how the codes and data are organized there, so that is the dedicated module for that. But in this module we look only the black box level view of a memory which will required to understand how work would executes.

So, basically if you look so memories are divided into mainly two types internal memory and external memory. So, internal memory basically is the semiconductor kind of a memory in which case you have a register. So, register is a part of the CPU itself. So, as we discussed in the last units, so this something like if you want to add two numbers so basically they have stored in a memory in a memory which is called the register; that means, something called a cache memory and a main memory. So, actually main memory is the what we have already heard the word called RAM. So, in a lay man language RAM, they are lot of technical has been come into, but in a lay man language what is known as a RAM is basically your main memory.

So, basically your CPU or your arithmetic logic unit of the main which is the computing unit of the CPU, basically it can talk only to the main memory that is it can generate the address and then it can read and write data from the main memory. But actually in between the main memory and the register, so you can think that the CPU of the computing unit is mostly closely attached to something which is called the register. And then all the operations mainly they compute at the register level. Because if you add two numbers generally one the temporary data or the result to be first stored in the registers and then they will go to the main memory. So, basically it is the most near path of the CPU, but actually the cost of having too many large number of registers is very high. So, therefore, the bulk code of the data executes mainly in the main memory which can be addressed by the CPU.

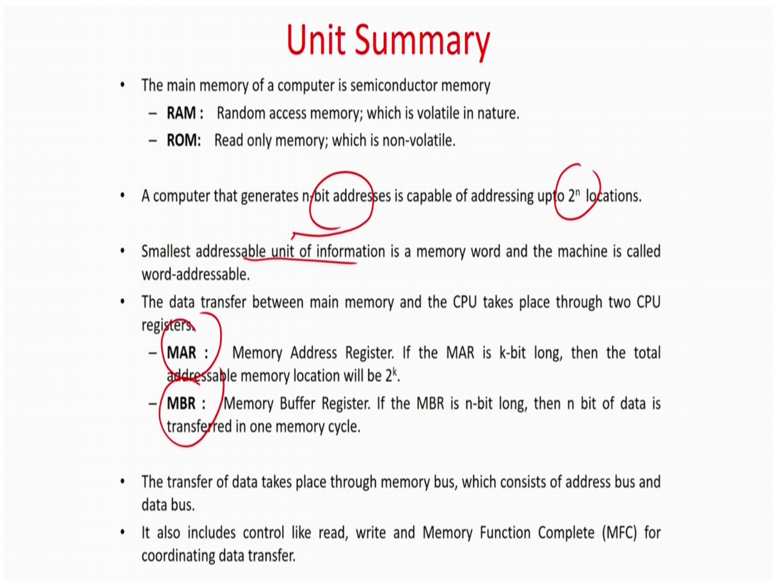
But there is another memory which lies in between the CPU and the main memory is called the cache memory. So, we will learn in more details about cache memory when we will going to into the full module on memory design. But the basic idea is that whenever you want to refer to some data, generally the address is generated for the main memory and as main memory is much slower compared to a register these something in between which is the cache. So, the cache will get some of the memory, some of the data from the main memory put it into the cache and they will be used.

And again when the data which is in the cache has been exhausted or you require any more new data, it will coming from the main memory. So, in fact as of now if you understand the concept of a register and the concept of a main memory, it will be a surprising to understand the concept of this module and also the unit here. So, in fact, you have to understand that the CPU will generate some memory address which corresponds to the main memory. And the data will be coming from the main memory to the register, for time being cache is transparent for our case and so forth.

And there is something called the last or the most top level in the memory which is called the hard disk that is your external memory. So, external memory can be further more if you go down the line there can be tape drives etcetera and there can be bulk SSD hard drives etcetera so, but for our case we are calling in a nutshell that is the hard disk or the external memory where the whole data will be stored. So, basically this is not a semiconductor type of a memory generally a magnetic memory if you are in a broad way.

So, basically what to happens is that whenever you want to execute a code we execute it from the memory locations on the main memory only. And whenever you want have something to be loaded which is not in the main memory then they are all copied from the main memory to the sorry from the hard disk or the external memory to the main memory and then the code executes. So, idea is that if the CPU wants to generate some address, the address is are generated mainly from the main memory. So, if data is not available in the main memory or some code or some instructions is not available over there, it is brought from the external memory, but external memory addresses are not generated as a part of the execution of the code. So, for us most important here are to understand the register and the main memory; others are actually in fact transparent to understand to this module.

(Refer Slide Time: 05:52)



Unit Summary

- The main memory of a computer is semiconductor memory
  - **RAM :** Random access memory; which is volatile in nature.
  - **ROM:** Read only memory; which is non-volatile.
- A computer that generates n-bit addresses is capable of addressing upto 2^n locations.
- Smallest addressable unit of information is a memory word and the machine is called word-addressable.
- The data transfer between main memory and the CPU takes place through two CPU registers.
  - **MAR :** Memory Address Register. If the MAR is k-bit long, then the total addressable memory location will be 2^k.
  - **MBR :** Memory Buffer Register. If the MBR is n-bit long, then n bit of data is transferred in one memory cycle.
- The transfer of data takes place through memory bus, which consists of address bus and data bus.
- It also includes control like read, write and Memory Function Complete (MFC) for coordinating data transfer.

So, now let us going to what will be next in the unit summary, which will more important to us are basically the main memory. So, main memory as a semiconductor memory as I told you that there are two types basically one is RAM and one is ROM. So, the RAM is the random access memory, basically it is volatile; and ROM is the read only memory, but both RAM and ROM are basically random access only that is that is not a sequential access, you can access any unit or any word of that memory. But it is from the I means the name has been carried over as for ROM also can be access random manner, but read only memory means you cannot write anything over there and it is non volatile.

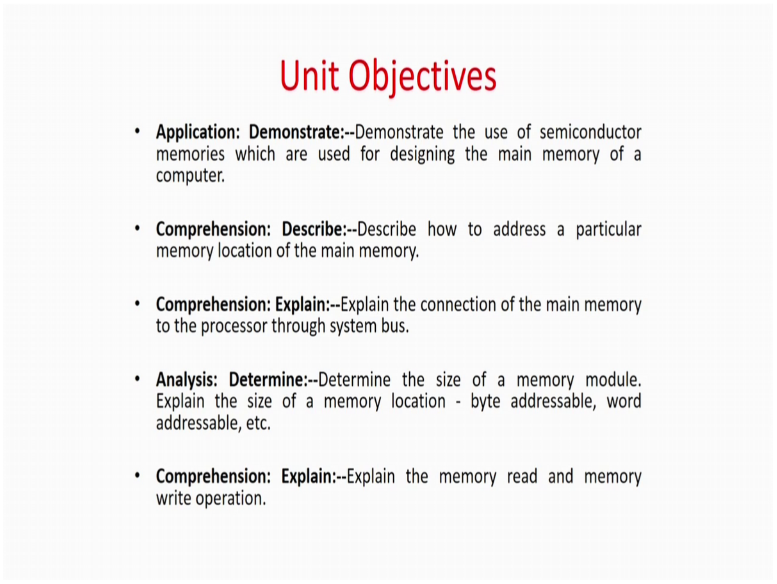
So, generally the bios or some important paths of your code or which is require for the machine is boot up which cannot be changed or in fact you should say that even if the machine is in shutdown mode, when it comes back to power or we require those course to be executed are all the read only memory. And mainly the random access memory is exactly what we understand by the term that which is used and reused of loading your code loading the data, changing the data and again writing it back is the random access memory.

So, now if you look at how basically it works. So, generally CPU will generate n bit address, because the memory will have some locations which can be done by the address. So, generally there are 2 to the power n locations, where n is the number of address lines. So, in that means, if there is a n lines address bus, so the number of locations will 2 to the power n; and by giving an address you can access any one of the data in that location.

And again the smallest unit of information is called a memory word that is a single bit may not be able to be accessed over memory, it memory can organized say for example, 2GB into 8, there is 2 giga bytes. So, into 8 means the memory is organized with each word is having 8 bits. So, at a time you can only read 8 bits out of the memory in one cycle. You cannot go for 4 or you cannot go for 16 so that is actually called a minimum smallest addressable information of a memory. It is not that in a word in a memory, which is organise as byte, you can access a single bit so that is the way how can you access it.

And in fact there are two very important registers one is the memory address register and one is the memory buffer register which will be used to access the memory. So, in fact whichever word you want to access that address will be given in the memory address register and whichever data you want to read or put it into the memory that is written in the or read into the memory buffer register. And in fact data transfer happens to the data bus and address is sent by the address bus. And there is some control signals like you want to read the memory you want write the memory and then some synchronization that is read is over write is over all those things. So, there are some control lines.

(Refer Slide Time: 08:35)



Unit Objectives

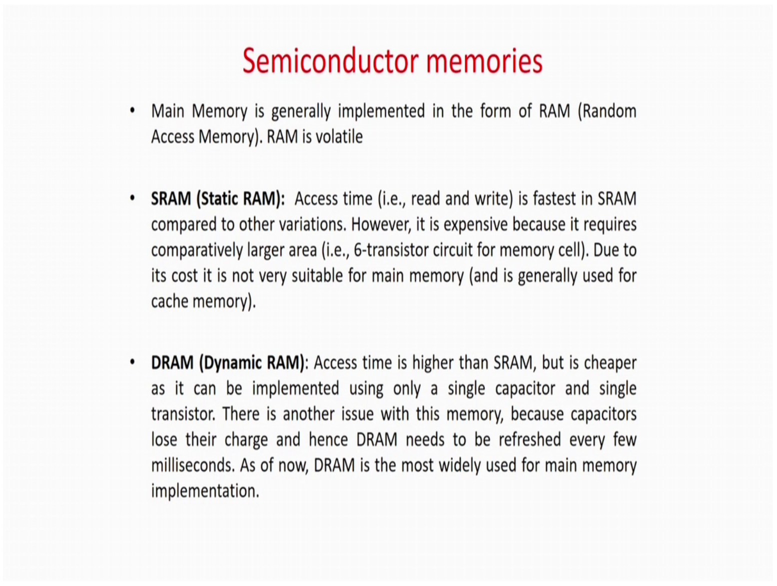
- **Application: Demonstrate:**--Demonstrate the use of semiconductor memories which are used for designing the main memory of a computer.
- **Comprehension: Describe:**--Describe how to address a particular memory location of the main memory.
- **Comprehension: Explain:**--Explain the connection of 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, in the nutshell this is how was the basic summarized notion of a memory which you are going to see in this unit. In fact the details of how a memory is constructed, how exactly data is organized. If there is some data which is absent in the main memory has been brought from the hard disk, if some memory is not available in the cache it has to be brought from the main memory all those details for that the dedicated module given on a memory. We will study it later.

So, what is the object is of this unit, after doing this unit or studying the unit you will be able to demonstrate the use of a semiconductor memory which is used in designing the main memory of a computer that is why it is require the basic idea able to demonstrate. Demonstrate in the sense that if there is a code how it is stored in the memory, how it is access and in what manner the CPU can interface with the memory. Then next you will be a is a comprehension objective you can able to describe how a particular memory address is accessed. Then you will able to explain the connection of the main memory through the process through the system bus that is data bus, address bus and control bus, then analyse you able to determine the size of a memory given its configuration that is byte addressable word addressable etcetera that if I tell you the memories 1 GB into 8.

So, what does it mean, so that configurations you will able to de demonstrate. And finally, you will able to explain the read and write memory operation that how it is done. This unit is basically giving a very broad idea of a memory main memory that is use to understand how a code or a instruction is executed. More details will be in a separate module.

(Refer Slide Time: 10:01)



Semiconductor memories

- Main Memory is generally implemented in the form of RAM (Random Access Memory). RAM is volatile
- **SRAM (Static RAM):** Access time (i.e., read and write) is fastest in SRAM compared to other variations. However, it is expensive because it requires comparatively larger area (i.e., 6-transistor circuit for memory cell). Due to its cost it is not very suitable for main memory (and is generally used for cache memory).
- **DRAM (Dynamic RAM):** Access time is higher than SRAM, but is cheaper as it can be implemented using only a single capacitor and single transistor. There is another issue with this memory, because capacitors lose their charge and hence DRAM needs to be refreshed every few milliseconds. As of now, DRAM is the most widely used for main memory implementation.

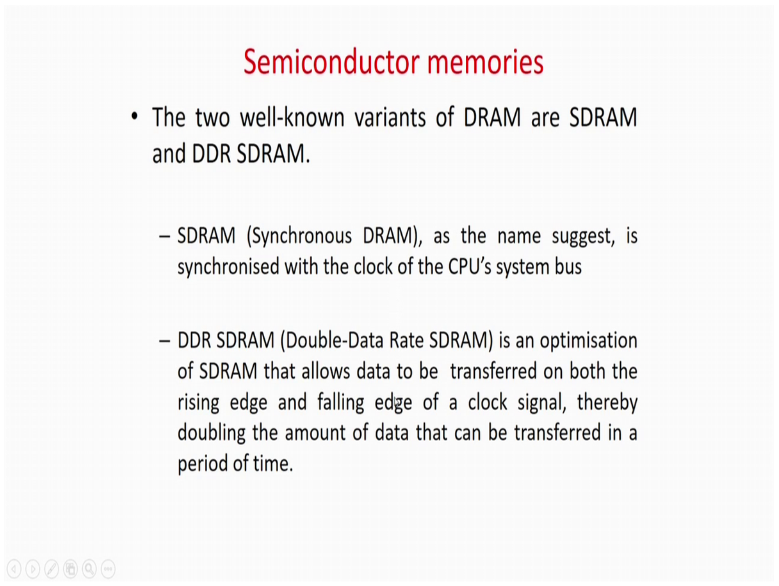
So, basically we start there are two types of memory, one is a semiconductor memory and one is a basically non semiconductor memory that may be a tape drive that can be a magnetic memory, which is generally consists of the external memory. But more important to us is something called the semiconductor memory because semiconductor memories are faster, you can build them on the cheap, but again they are expensive. So, you cannot have a 1 tera byte RAM that is giving not possible by today's fabrication technology, but it will be extremely faster.

So, what we will do is that they actually gradually if you go down the memory hierarchy that is if you go from the hard disk to the RAM to the main memory then to the RAM and then to the register, this size will come down, speed will increase. And of course, the cost will increase that is what is the idea. So, basically the main memory which is mainly we are concerned about and the register that is mainly made up of flip flops, but main memory which you are mainly concerned about at present are basically of semiconductor.

So, basically there are two types of memory one is the static memory, and one is the dynamic RAM. So, static RAM the access time is faster in SRAM once you have to any other variation. Actually it is a extremely fast generally it may be used for means register design and, but the size will be slightly larger because they are actually taking six transistor should design. As I told you at present in this unit we are not going to depth of how the sixth transistors are organized, you will be learning in details in later modules of the course.

There is something called the DRAM which is dynamic RAM. It is a much access time is higher in fact that is more cheaper to implement in static RAM, but the idea is that is actually implemented by the single capacitor and some register size will be smaller than the static RAM. So, you therefore, you can be larger size of DRAM in the same price compare to various SRAM. But time access time will be slightly high, but still it is doable. So, most of the modern days, you can understand DRAM is the most wide technology for main memory implementation. Static memory is size is bit high, so you can think that it can you can use it for designing of some high-speed memory is to be registered so something like that.

(Refer Slide Time: 12:07)

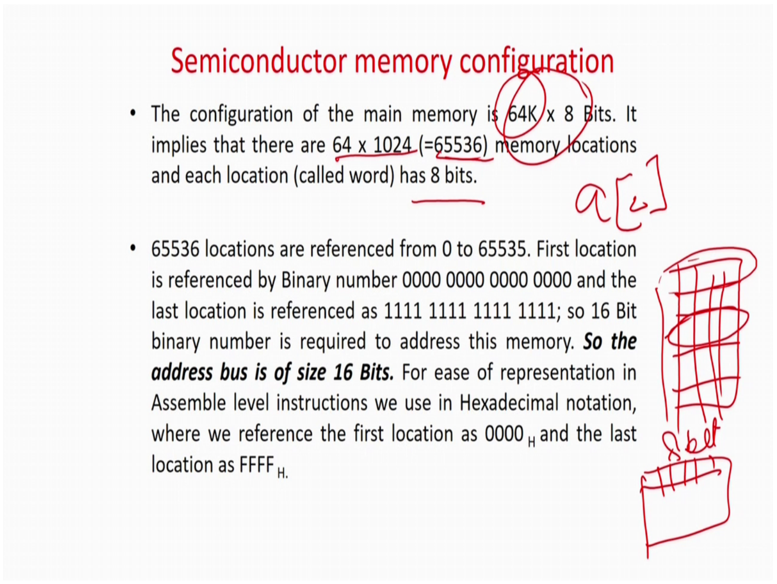


Semiconductor memories

- The two well-known variants of DRAM are SDRAM and DDR SDRAM.
  - SDRAM (Synchronous DRAM), as the name suggest, is synchronised with the clock of the CPU's system bus
  - DDR SDRAM (Double-Data Rate SDRAM) is an optimisation of SDRAM that allows data to be transferred on both the rising edge and falling edge of a clock signal, thereby doubling the amount of data that can be transferred in a period of time.

As I told you our most of the RAMs are including dynamic ram. So, there are two type of dynamic RAMs, one is synchronized DRAM and one is DDR RAM. So, synchronized DDRAM is as the name suggests, you can read the memory location at every edge of the clock, it may be positive edge it may be negative edge in any edge you can read or write it. And that is actually it say that it is synchronous RAM which is synchronized with the operation of the clock. Double SDRAM is also synchronize operation, but it more faster or doubly faster than the SDRAM because in this case you can read and write the memory at both the edges of the clock that is you can also go for the rising edge as well as you can do the falling edge. So, the speed of this RAM will be higher.

(Refer Slide Time: 12:47)

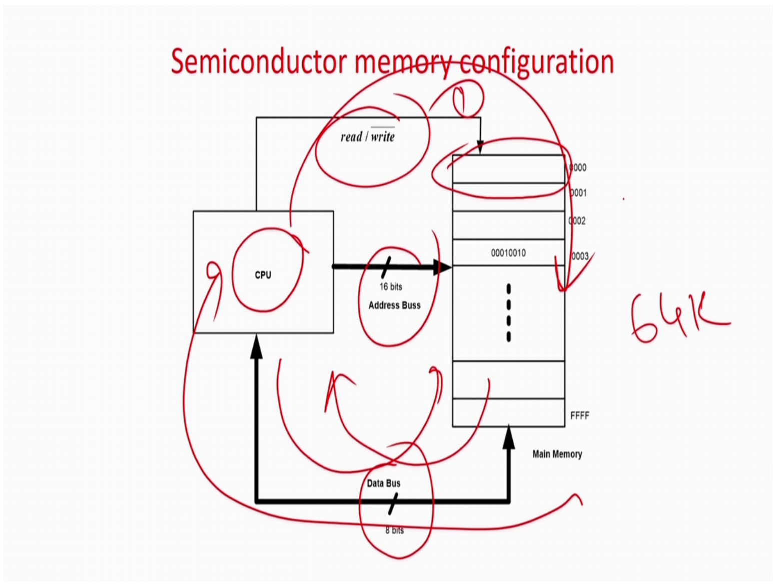


Semiconductor memory configuration

- The configuration of the main memory is 64K x 8 Bits. It implies that there are 64 x 1024 (=65536) memory locations and each location (called word) has 8 bits.
- 65536 locations are referenced from 0 to 65535. First location is referenced by Binary number 0000 0000 0000 0000 and the last location is referenced as 1111 1111 1111 1111; so 16 Bit binary number is required to address this memory. ***So the address bus is of size 16 Bits.*** For ease of representation in Assemble level instructions we use in Hexadecimal notation, where we reference the first location as 0000_H and the last location as $FFFF_H$.

So, now we are going to look at a what is a memory configuration, is somebody tells you that I have a generally memory configuration are written in a product type of manner. So, for example, they have written that 4 k into 8 bits that means, what in this case I will show you the figure of a memory first.

(Refer Slide Time: 13:04)



So, the memory basically looks like this, this is a an array, but you can just think that memory is basically is an array. So, let me first go back memory actually looks like an array. For the very simple notion you can think it is an array, but this is only one difference is its two-dimensional array I should say that it is a basically a two-dimensional an array. So, for example, we can access a 2, 3. So, you can this you can easily do in this c program, but here is slightly interesting that you cannot do this bitwise access. Because I told you in a memory the difference is that you have to access one word at time or in fact you can think that is 2d binary or read and write you have to do one complete row at a time. So, the row is called the one in terms of memory technology.

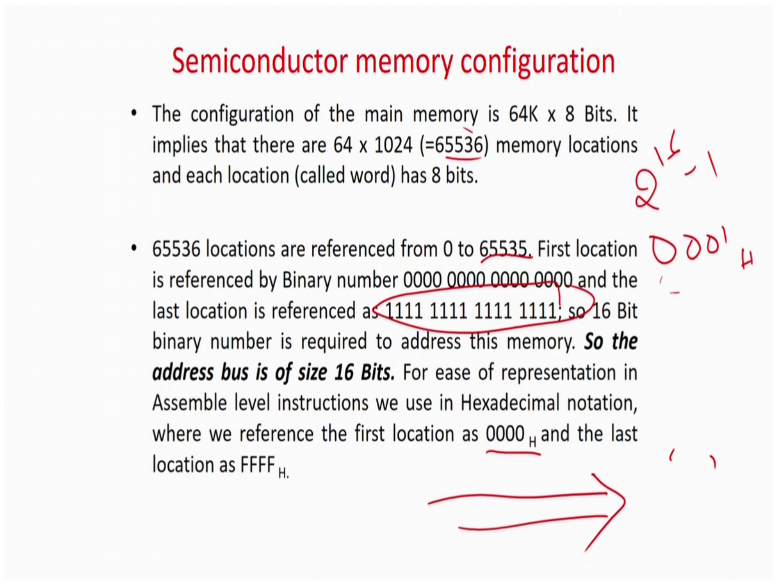
So, in fact if I say in that way you have to actually drop this through so that means, you can access location number to 0 1, 2 then this whole row will be read in the memory or you can write it in the memory, individual bit access is not possible. So, you can remember whenever in this manner that is a 2d array is no problem about it, but only thing is that you cannot read a single bit out of it, you have to the read whole row at a time or write at a time.

So, here it is saying that it is 64 k into 8 bit memory. So, what does it mean it means that the 2 d array looks like this is 8 bit that is row size is 8 bit or 8 units are there in which each unit you will be able to write either a 0 or a 1. So, there are 8 bits over there and that is actually called the one word.

And then what is the number of rows in the memory, so that is actually given by the first. So, it is saying it is 64 k so that means, it is actually 64 into 2 to the power 10 that is 64 into 1024 that is something like that is right. So; that means, we have 64 into 2 to the power 10 that is 64 k that many memory location that means that many rows will be available. And each location has 8 bits that is actually called the word, but in fact the very similar we can write it as an array of size 655368, but only thing is that this 8 individual bits cannot be accessed you can address the whole 8 bit at a time. So, in fact this will be called as a byte accessible memory because one word or one byte can be accessed here.

And how do you read out the locations. So, how to you access the memory. So, in fact 0 to memory locations start from 0 to 65535, and the first location is all this and the last location is this. So, that is the if you say and you have so they anyway we say that whenever this is the number that is the number of rows is this one. So, it is actually 2 to the power 16.

(Refer Slide Time: 15:41)



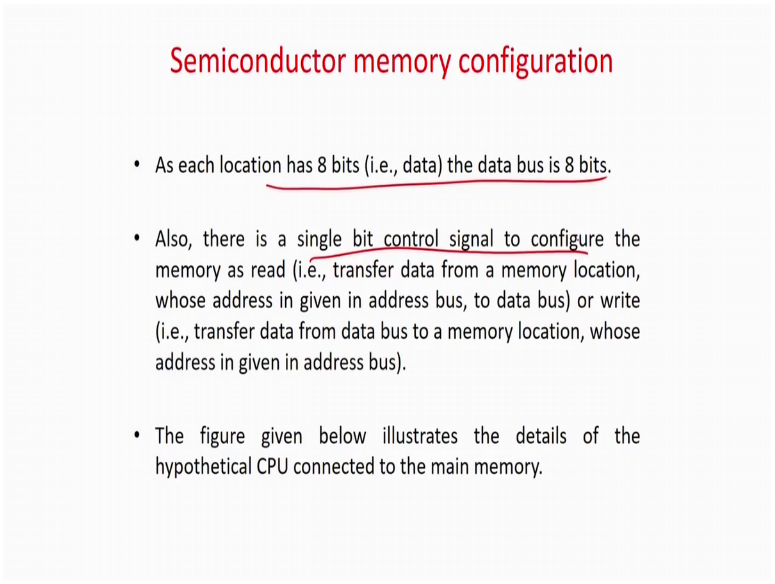
Semiconductor memory configuration

- The configuration of the main memory is 64K x 8 Bits. It implies that there are 64 x 1024 (=65536) memory locations and each location (called word) has 8 bits.
- 65536 locations are referenced from 0 to 65535. First location is referenced by Binary number 0000 0000 0000 0000 and the last location is referenced as 1111 1111 1111 1111, so 16 Bit binary number is required to address this memory. ***So the address bus is of size 16 Bits.*** For ease of representation in Assemble level instructions we use in Hexadecimal notation, where we reference the first location as 0000_H and the last location as $FFFF_H$.

So, if you go for 2 to the power 16 then you are going to get this number minus 1 of course, then actually your bits I mean bus address bus size is 16 bits that means, you have to access words from 0 to 65535. So, you need a binary numbers which can access numbers from all 0s to 35535 that is the value. So, is a 16 bit number all zeros means the first memory word and all ones means the last memory word. And therefore, the address bus from where the address we will be given will be a 16 bit bus which actually comes from this number.

And again, but generally it is very cumbersome to write these values. So, we write in a hexadecimal notion that is 0 0 0 0 hex that is the first memory location and f f f f is the last memory location. So, any value if you give, so if I give 0 0 0 1 hex that means, I am accessing the first memory location z 0 is this one first memory location I am going to access. So, if I write 0 0 0 1 hex as the first memory location is address, and depending on the construal value of read or write, you will be able to either read all the 8 bits that is very important to note compare to 2 d array, all the 8 bits will be either read and written at that memory location.

(Refer Slide Time: 16:55)



Semiconductor memory configuration

- As each location has 8 bits (i.e., data) the data bus is 8 bits.
- Also, there is a single bit control signal to configure the memory as read (i.e., transfer data from a memory location, whose address in given in address bus, to data bus) or write (i.e., transfer data from data bus to a memory location, whose address in given in address bus).
- The figure given below illustrates the details of the hypothetical CPU connected to the main memory.

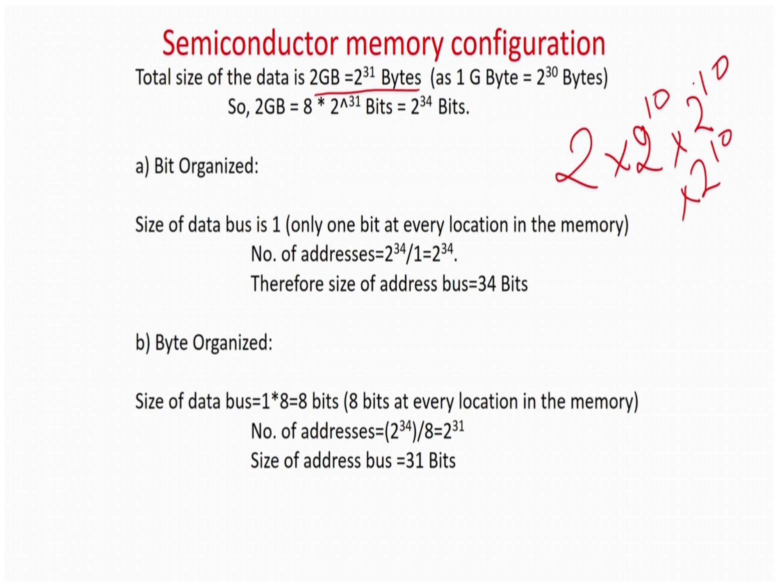
As I told you, each location has 8 bits. So, the data bus will be 8 bit bus. So, when you address. So, you referring one memory location now you have to read or write data from that. So, you require another bus which is actually called the data bus. Now, what is the length of that bus or the width of that bus sorry the width of that bus, it will be the second term in this case you can access word whose size is 8 bits. So, the word size will be 8 bits. So, if the memory is 64 k into 32 that means, it will be four byte addressable memory. So, the address bus size is 32 bytes bits.

And again very important basically we are doing two staff at a time either we are reading the memory or we are writing the memory reading the memory means we are taking whatever value is in the corresponding memory location the address to the CPU and write is the reverse direction. So, there should be a control line to tell whether it is a read or a write memory. So, there will be a single control signal which will tell you whether to read and write.

And the figure below actually this one is a memory, I will come to that again I am saying that how the memory cells are organize. So, that a single word you can single control line will make it read or a single control line biology will make it write depends on the how the memory is implemented it. So, all the details of the internal memory structure will be looking at the future, but for time being the knowledge we are giving for the memory in a very nutshell will be enough to understand how a code executes, so that I told you this is the CPU it will generate 16 bit address.

Now, why a 16 bit address because the memory size is 64 k which is nothing but 0 0 0 0 to f f f f; and we know that this is actually 8 bits or byte addressable to the data bus which is bidirectional bus will be 8 bits writing will be from this renew from this side and is a control signal. So, if it is read-write bar that means, if these signal value is 1, then the memory will be in a read mode, this way; and if you make this as 0 then it will be a write mode so data from here will be coming over there. So, in a nutshell a memory will looks like this.

(Refer Slide Time: 18:50)



Semiconductor memory configuration

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

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

$2 \times 2^{10} \times 2^{10} \times 2^{10}$

a) Bit Organized:

Size of data bus is 1 (only one bit at every location in the memory)

No. of addresses=2^{34}/1=2^{34}.

Therefore size of address bus=34 Bits

b) Byte Organized:

Size of data bus=1*8=8 bits (8 bits at every location in the memory)

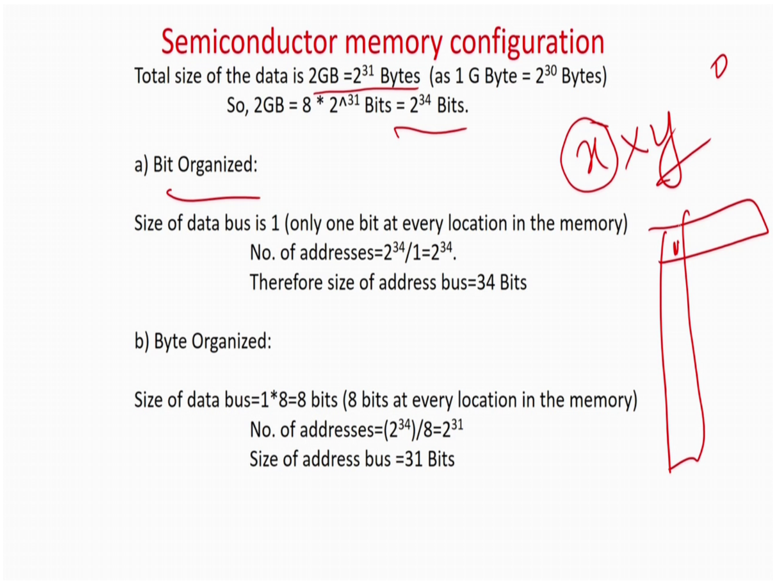
No. of addresses=(2^{34})/8=2^{31}

Size of address bus =31 Bits

So, now we will study some of the basic semiconductor memory configuration by examples. So, sometimes let us take the examples of 2GB into that is 2GB. So, what is 2 GB that is 2 into GB means mega bytes sorry it will be kilobytes, mega bytes, giga bytes and tera bytes something like that right. So, 2 to the power 10 that is your kilobytes megabytes and giga bytes so 2 to the power 30, so that is 2 to the power 30 so that is nothing but 1GB is 2 to the power 32 GB is 2 to the power 31. So, as you can see. So, byte bit. So, it is you can see that is 2 GB nothing but 2 to the power 31 bytes that is 2 to that is one because 1 byte as I told you 2 to the power 30 bytes and 2 GB is 8 into the 2 to the power 31 bits because 1 byte is 8 bits. So, the whole memory you can tell is as 2 to the power 34 bits.

But again I told you that is generally we write a memory in terms of x cross y that is x is the number locations and this is the bit.

(Refer Slide Time: 19:52)



Semiconductor memory configuration

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

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

a) Bit Organized:

Size of data bus is 1 (only one bit at every location in the memory)

No. of addresses$=2^{34}/1=2^{34}$.

Therefore size of address bus=34 Bits

b) Byte Organized:

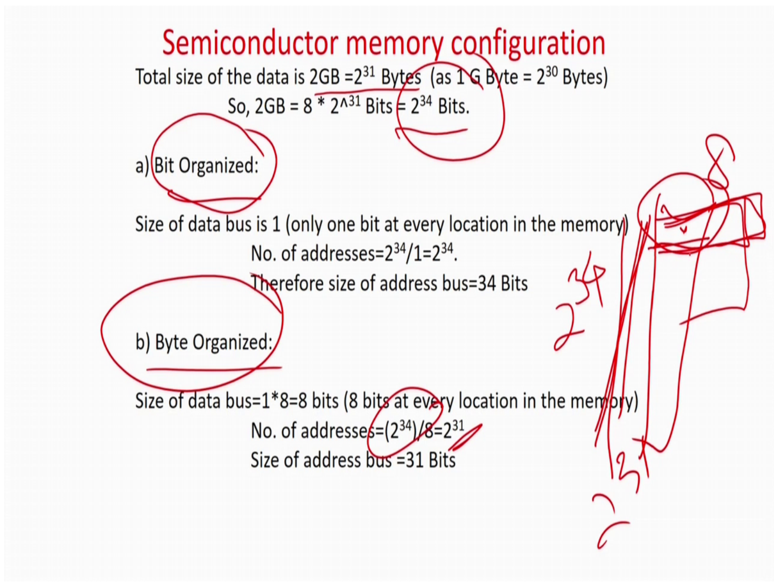
Size of data bus=1*8=8 bits (8 bits at every location in the memory)

No. of addresses$=(2^{34})/8=2^{31}$

Size of address bus =31 Bits

So, if you write the 2 to the power 34 bits; that means, I am not telling you; what is the length of this one and what is the width of the memory. So, you have to tell in which way. So, in this case it is saying the whole size is 2 to the power 34 bits that is again another way of describing a memory, but immediately you have to tell that it is bit organized. So, what do you mean by bit organized; that means, there is only one bit in the row. So, it is a very non practical kind of a memory, just to show the example either you write x y or you can tell the total size of the memory that is in this case 2 to the power 34 bits. How 1 byte is equal to 2 to the power 30 bytes 2 bytes 2 GB is equal to 2 into 2 to the power 31 bits bytes sorry and then 1 byte is equal to 8 bits. So, the whole size is 2 to the power 34 bits, but as I know this what is the size and what is the size over here. So, in this case it is said it bit organized bit organized means.

(Refer Slide Time: 20:49)



Semiconductor memory configuration

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

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

a) Bit Organized:

Size of data bus is 1 (only one bit at every location in the memory)

No. of addresses$=2^{34}/1=2^{34}$.

Therefore size of address bus=34 Bits

b) Byte Organized:

Size of data bus=1*8=8 bits (8 bits at every location in the memory)

No. of addresses$=(2^{34})/8=2^{31}$

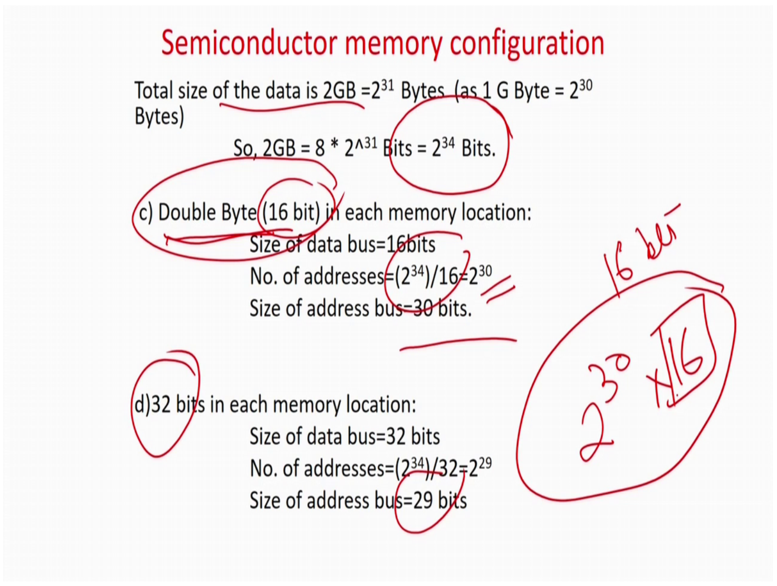
Size of address bus =31 Bits

There is only one bit at a time. So, in this case; obviously, the whole length is 2 to the power 34 that is 2 to the power 34 memory locations will be there because only one bit at a time very impractical kind of a memory only the idea is given you by the organise this is more or less practical. So, it is says that this length size is 8 bits that is an in a byte in bite organized individual bits can be accessed, but as I told you this is a non very not of practical way of implement because generally means we represent a data in terms of 1 byte, 2 bytes because that is more meaningful.

So, you want to access one meaningful word at a time in this case you want to add may be 8 bits one at a time you have to read eight locations and then merge it and then the understanding the meaning of it. So, it is not a very good way of doing it that is more better that you put 8 bits or 16 bits in one row, get the value and make a meaning directly out of it there are then going in a sequential manner. So, this byte organized.

So, what do you mean by byte organized, there are 8 bits over here. So, what will be the length, it will be 2 to the power 34 divided by 8 that is the 2 to the power 31. So, this size is 2 to the power 31 bits that is 2 to the power 31 locations are there. So, in this case what will be the size of the address bus 31 bits, because you need to access all the 2 to the power 31 bits very simple.

(Refer Slide Time: 22:03)



Semiconductor memory configuration

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

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

c) Double Byte (16 bit) in each memory location:

Size of data bus=16bits

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

16 bits
$2^{30} \times \boxed{16}$

Again the same thing we have taken now it is a double byte. So, why do you actually have different type of a memory organization the idea is that sometime if you make the memory size we too wide then what it may happen you may wasting your size that means, say single instruction takes about the 16 bits or 8 bits. But oh you can never implement a single instruction or explain the meaning in one or two bits. So, if you have a two bite organized memory then to find out the meaning of a meaning to find out what is the meaning of a valued word or what do you mean means valued 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.

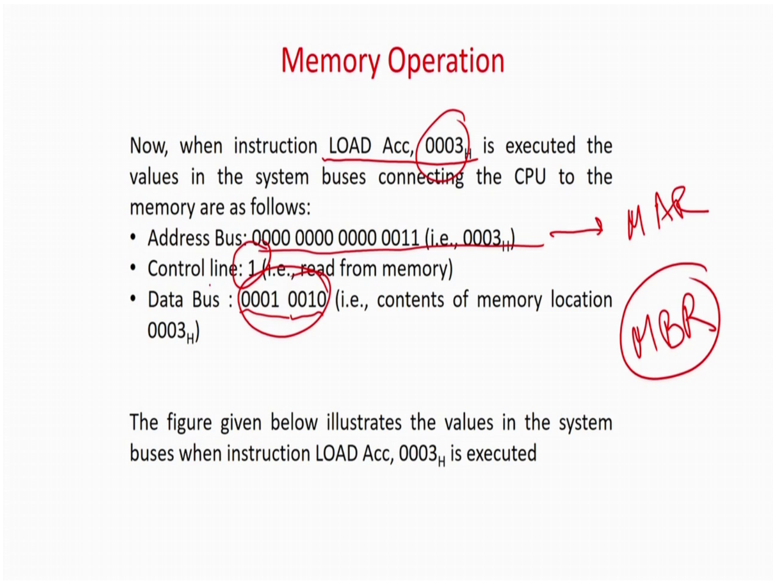
So, generally say we are taking double byte that is 16 bite. So, ma say may be you are going to feed the whole instruction in that. So, just read one word and your job is done. For example, if I have 64 bit what then what will happen then one bit what will have one or two or three instructions then again 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 will try to do people try to keep the memory organize 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 idea the 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 address is 2 to the power 34 byte, 16 that is 2 to the power 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 will be 29 bits. So, what will be the byte organization of 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 the number of address bus with the very simple idea.

So, either if you represent as 2 to the power 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 to the power 30 into 16 that is also that will tell you I have 2 to the power 30 memory locations and each memory location size is 16 that also will be the surprising. So, anyway is ok for us.

(Refer Slide Time: 24:56)



Memory Operation

Now, when instruction LOAD Acc, 0003_H 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_H)
- Control line: 1 (i.e., read from memory)
- Data Bus : 0001 0010 (i.e., contents of memory location 0003_H)

The figure given below illustrates the values in the system buses when instruction LOAD Acc, 0003_H is executed

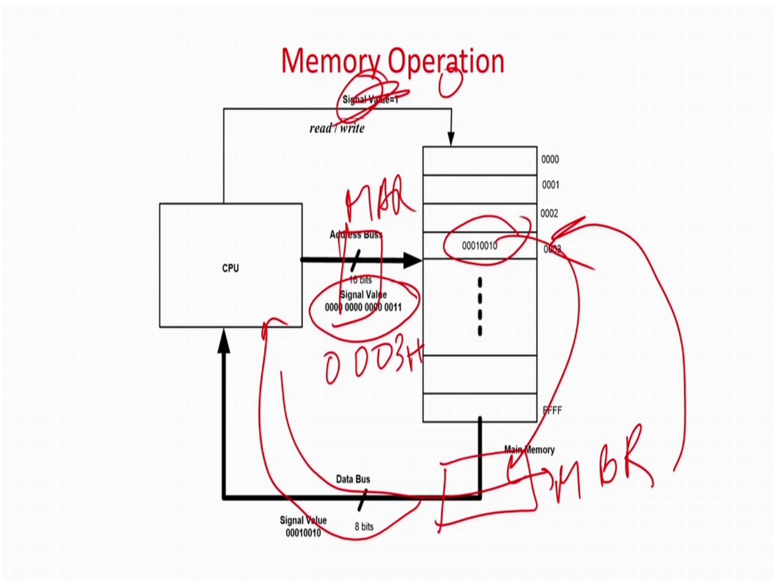
Now, this basic configuration of the memory now will say that for example now will as the main goal of this module will to understand what is an instruction how it execute etcetera. So, now, we will see basically how a memory read or write is an instruction a simple instruction load accumulator 0003. So, what is an accumulator for the time being in the last module also Prof. Deka has given a basic idea of what is 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 emphasize that what is the memory location we write in an instructions 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 just 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 emphasize of this course.

So, load accumulator 0003 that is how it will happen. So what did happen 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 one because you want 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 a 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. 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. And then this memory buffer register input the value into the memory to the accumulator register.

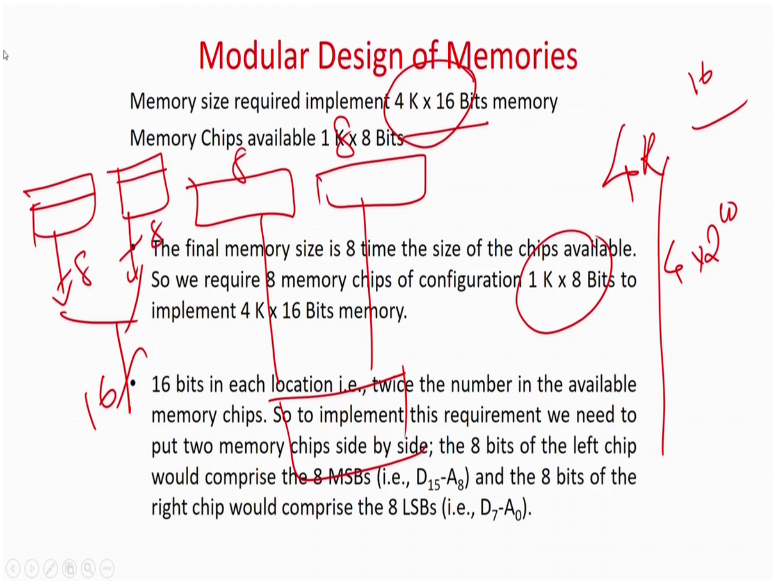
(Refer Slide Time: 27:18)



So, now, let us look at in the figure. In fact, we have not drawn over her 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 (Refer Time: 27:45) to be 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 return to the memory location, just a reverse by this one. So, this is in a nutshell again we read and write operation of the memory.

(Refer Slide Time: 28:08)



Modular Design of Memories

Memory size required implement 4 K x 16 Bits memory

Memory Chips available 1 K x 8 Bits

- The final memory size is 8 time the size of the chips available. So we require 8 memory chips of configuration 1 K x 8 Bits to implement 4 K x 16 Bits memory.
- 16 bits in each location i.e., twice the number in the available memory chips. So to implement this requirement we need to put two memory chips side by side; the 8 bits of the left chip would comprise the 8 MSBs (i.e., D_{15}-A_8) and the 8 bits of the right chip would comprise the 8 LSBs (i.e., D_7-A_0).

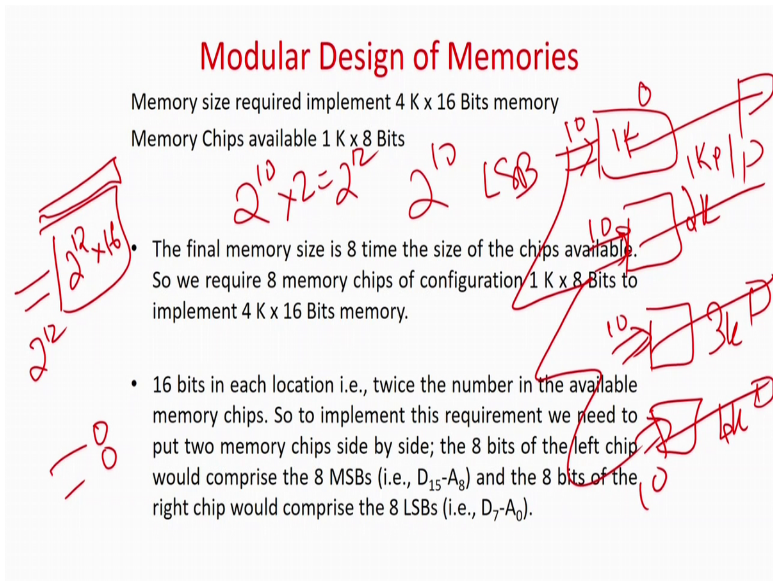
So, now, let us think that we have a RAM may be we all know now a that is we are purchasing RAM in terms of slot. So, we purchase 1 GB RAM slot four then we put four slot together, may be you have 2 GB RAM cards and put in this slot; that means, memories are modular. So, nobody actually purchase may be a 16 GB RAM in one chip generally they are may be broad 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 able to have flexibility otherwise for each design you may have to make tailor made memory which is not actually very good idea because I require generosity and then I want to plug in and plug out to do that.

So, for example, say that I want to take a 4 k into 16 bits memory that is I require a 4 k that this has will be 4 that is 4 into 2 to the power 10. And this 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 k into 8 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 a my required memory, so that is you modularly plug in and do slide 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 is how do you do it. So, in fact, you see 16 bits and this is a 8 bits. So, naturally you have to put 2, 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 parallely very simple idea that will have two 8 8 bits here one 8 bit here and then both of them will be reading that will two data buses. And finally, 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. Now, the next part is that, but we have only 1 k memory size, but we require a 4 k 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)



Modular Design of Memories

Memory size required implement 4 K x 16 Bits memory

Memory Chips available 1 K x 8 Bits

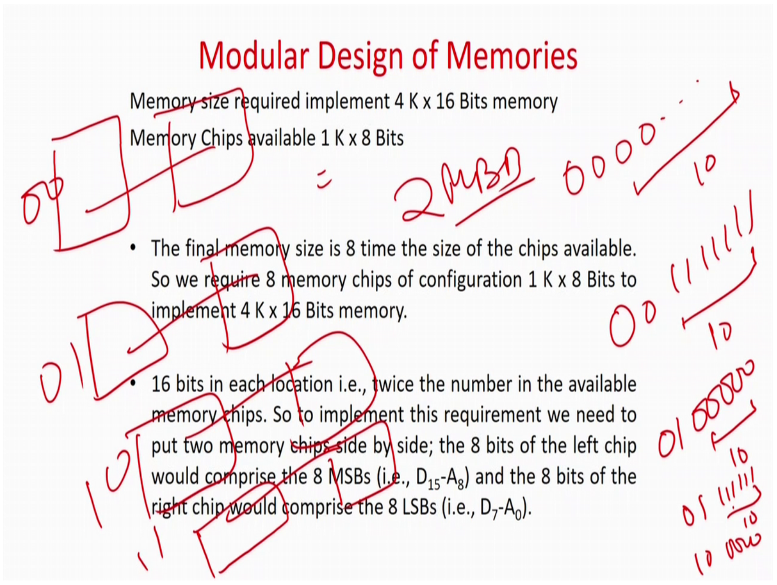
- The final memory size is 8 time the size of the chips available. So we require 8 memory chips of configuration 1 K x 8 Bits to implement 4 K x 16 Bits memory.
- 16 bits in each location i.e., twice the number in the available memory chips. So to implement this requirement we need to put two memory chips side by side; the 8 bits of the left chip would comprise the 8 MSBs (i.e., D_{15}-A_8) and the 8 bits of the right chip would comprise the 8 LSBs (i.e., D_7-A_0).

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 2 k this one be 3 k and this one will be 4 k this 0 to 1 k then 1 k plus 1 to 2 k and so forth that will be done, but now 1 k means what 2 to the power 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 4 k. So, it is a 4 k memory means 2 to the power 10 into 2 that is equal to 2 to the power 12. So, in this case the address bus size is 12 for the overall memory, because overall memory is 2 to the power 4 k that is 2 to the power

12 into 16 that is overall configuration. So, this one is 16 because we have now two, two in series and in fact there are two other we need this to draw because the that will happening this two are happening parallely.

But now as the access this four in a serial manner, because here the number of (Refer Time: 31:29) is 2 to the power 12 because that is 4 k, 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 in the 10 LSBs of the end of process will connect 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 1 k, the next repeat from 1 k plus 1 to 2 k.

(Refer Slide Time: 32:13)



Modular Design of Memories

Memory size required implement 4 K x 16 Bits memory

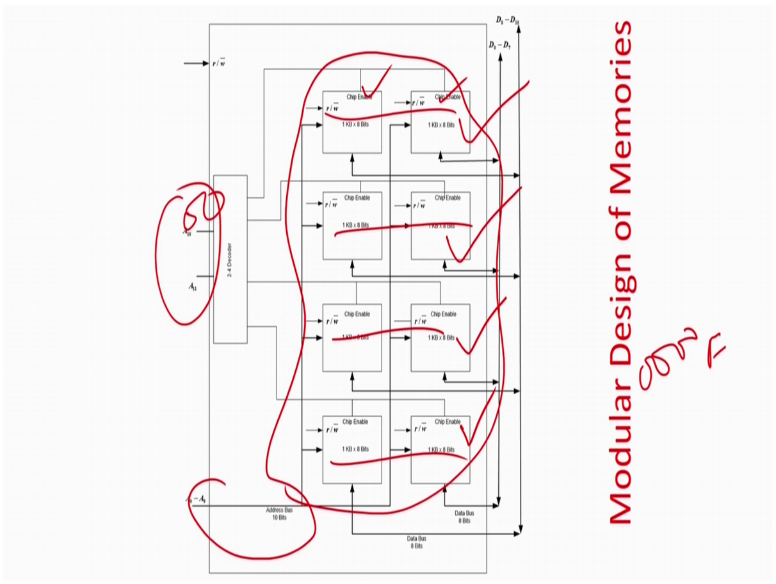
Memory Chips available 1 K x 8 Bits

- The final memory size is 8 time the size of the chips available. So we require 8 memory chips of configuration 1 K x 8 Bits to implement 4 K x 16 Bits memory.
- 16 bits in each location i.e., twice the number in the available memory chips. So to implement this requirement we need to put two memory ~~chips~~ side by side; the 8 bits of the left chip would comprise the 8 MSBs (i.e., D_{15}-A_8) and the 8 bits of the right chip would comprise the 8 LSBs (i.e., D_7-A_0).

So, that means, what if the memory is 0000 may be all 000 is a 10 bits all 000 then the first memory block will be selected. Again if it is 0 1 then all 000000 that is the next memory block; that means, when all this is this for corresponds to the first memory block till how much little bit a 00111111. So, this is the again 10 bits. So, this corresponds to the first memory first two basically I should say the first two memory cells or the memory chips in the row. Then these 10 this is 1 k is done. So, next it will be what next will be 0100000 this is 10 bits that is the next number for this. So, immediately in this case this two blocks has to be selected till what till it will be 0111111, this is again 10 bits this way selected next it will be 1 0 and 0000. So, if the LSB two is 0 0 then this one is selected the if the sorry MSB if the MSB is 0 1 this word selected. If the MSB is 1 0 this word selected and if the MSB is 1 1 MSB of the I am talking about the MSB of the address bus to now the address bus is 12 bits. So, this two will be selected.

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 the something I have not told you as of now the 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 actually tells you what is if chip can be switch 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 as I told you this a 16 bit. 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 10 bits. So, A 0 to A 9 of the address bus for the whole chip. So, individual size is 1 k into 8. So, the address bus size is 10, but for the overall the address bus size is 12. So, the last two bits A 11 and A 12 are going to control which chip which row has been selected as you seen if LSB two bits are 00 so this one has to be selected. And if it is 0 1 this row 1 0 and 1 1 and all the other bits load has to be all the other row must not be selected at that time.

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

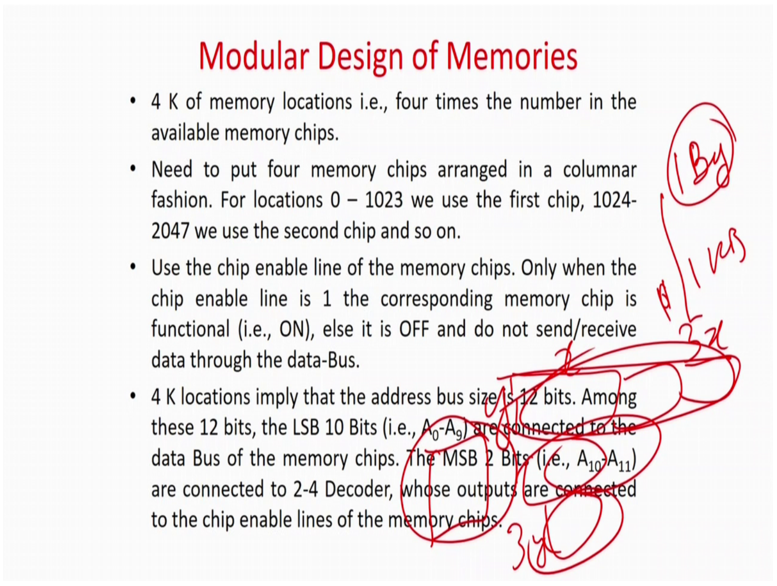
So, if the address is say all 0s; that means, I have two access the first row of the first set of chips that is the first row of the that is the first word I should say the first word of the first two rows of chips, so all zeros. So, of course, last 10 bits are 0, LSB 10 bits are 0. So, this two access and the MSB two bits are also 0; that means, this line is basically a one. So, an all others are 0. So, this chip enable and this chip enable are one making these two chips on and all others will be all three other rows will be 0. So, that is not giving any output. So, definitely you are going to access this row.

Again let us take the example that all ones that is the last row of the chips has to be accessed. So, all one means every d has one of the address bus. So, in fact all one means last 10 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 access accessed why because 10 bits are all ones, but now which among the four row will be selected. So, again as a all one is the address. So, this MSB 2 bits are also 1 1. So, MSB bit 1 1 imply that, this row is 1 and all other rows are 0. So, this row this row and this row will be mean a module of manner; and only this two rows will be in a module on there is chip enable, and these two giving the data.

So, in fact if you look at it, so this is; what is the configuration that this 10 bits bold line if you look at it, this bold line will access all the chips at a time because chips are of size 1 k. So, if you want to access any row say zeroth row, second row, first row, so all the corresponding or same rows for each of the 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 0 0. So, only this chip will enable on and this row will access and so forth.

So, this is actually what is a what is something called a modular design. So, you take different modules and arrange them using a decoder. So, if the if you given 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 into 4, so 4 memory blocks is put in series. And then depending on the size row you have to multiply by that manner.

(Refer Slide Time: 37:59)



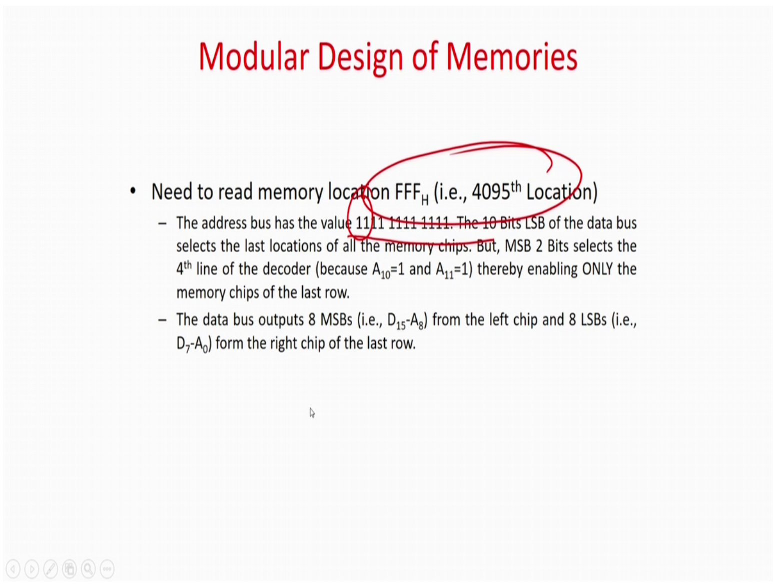
Modular Design of Memories

- 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, a you have a block size is 1 byte and this size is also say 1 k 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 said 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 3 x 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 3 y 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 two is to four whatever is require the decoder which will select the appropriate row, this row or this row or this row depending on the memory access. 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 4 k 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 10 bits will be connected to the data bus of all the memories and the MSB two bits are connected to two is to four decoder and this is what is the memory organization. So, whatever I told you can theory which is given in the PPT, and it will module will be clear.

(Refer Slide Time: 39:36)

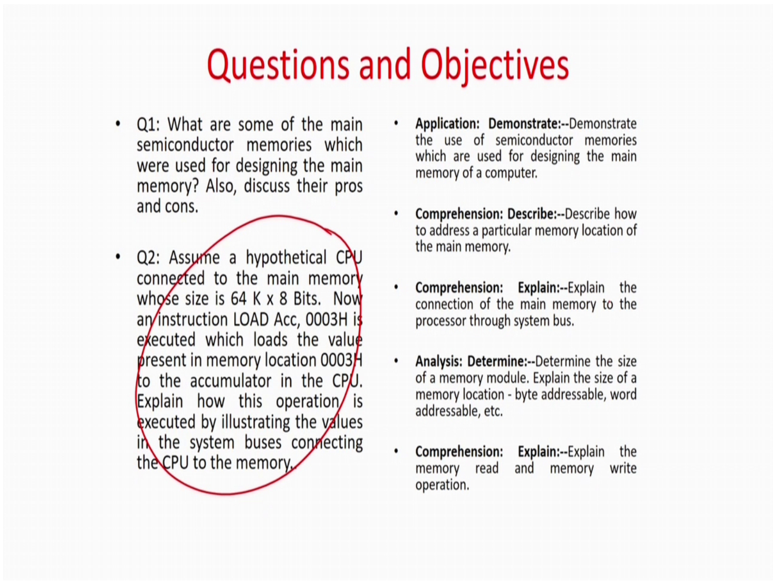


Modular Design of Memories

- Need to read memory location FFF_H (i.e., 4095th 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 4th 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) form the right chip of the last row.

Now, they have given that they have given an example that how to 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 once 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 1 1; 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.

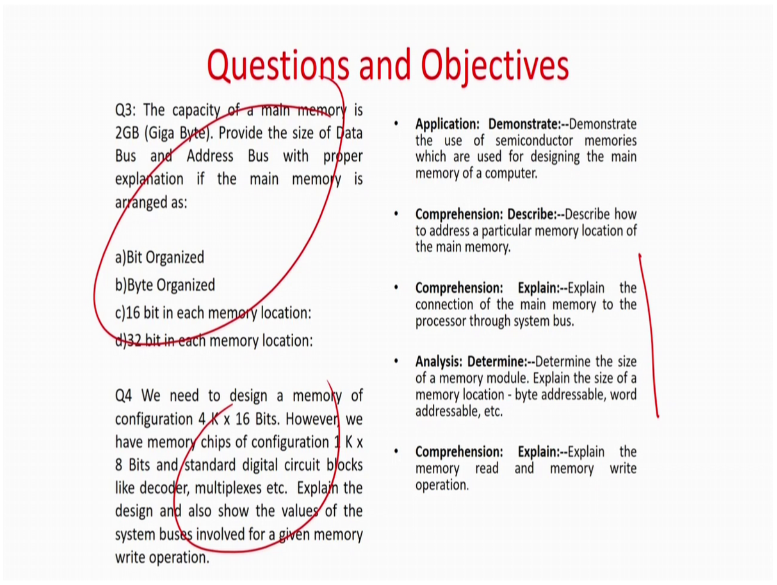
- **Application: Demonstrate:**--Demonstrate the use of semiconductor memories which are used for designing the main memory of a computer.
- **Comprehension: Describe:**--Describe how to address a particular memory location of the main memory.
- **Comprehension: Explain:**--Explain the connection of 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 you 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 memories implemented etcetera. We have try 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 require, how do you give the address, what are the register involves about how an it is read and write only in the this prospective we have played. Because for this unit the for this module and the next module where will be the many understanding how work 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 we 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 you 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, need 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 object is where we are able to demonstrate the use of semiconductor memory which are used in the main computer that is the first question is of this is objective that is your able to answer the question which object will satisfied.

Similarly, we are give a hypothetical memory configuration and then we have a asked how it can be interface etcetera with the system bus. So, in fact this will be covering of the objective on how to address of 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 K x 16 Bits. However we have memory chips of configuration 1 K x 8 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 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 you 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 require to understand how basically the codes can be execute it.

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

Thank you very much.