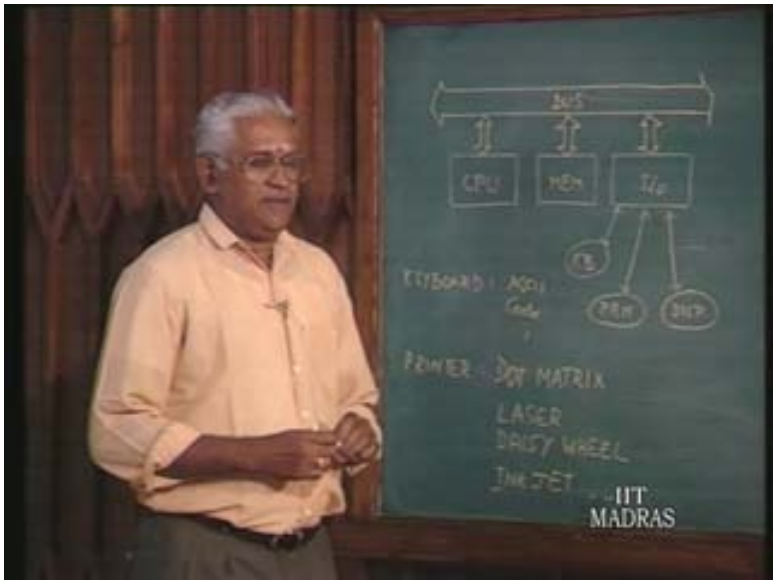
Computer Organization
Part – I
Prof. S. Raman
Department of Computer Science & Engineering
Indian Institute of Technology
Lecture – 3
Introduction to System: Hardware

In the previous lecture I said that I will help you get an overview of software and hardware in general. By and large, we covered most of the software: different aspects of software and families of software, and then we started with the hardware which we will try and complete today and possibly extend. Whatever that the user sees are listed here. If he opens a computer system then he would find a CPU board, some memory boards and some I by O boards.

(Refer Slide Time 00:01:09)

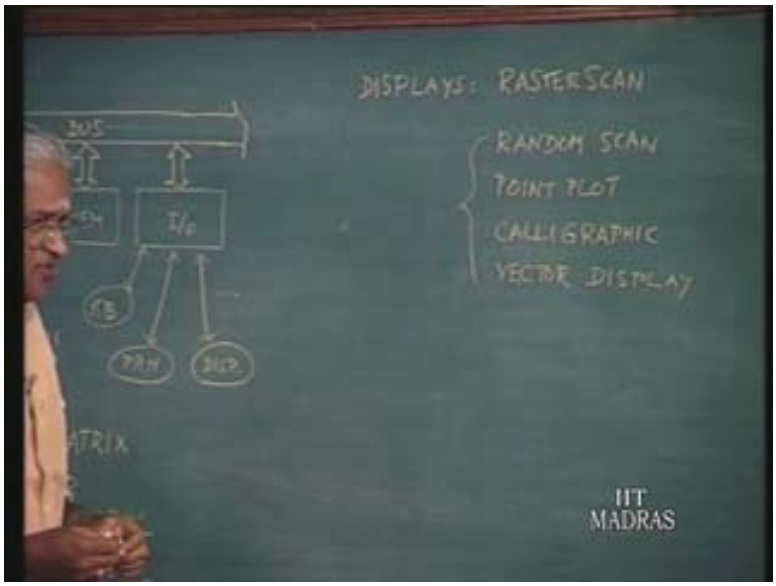


Essentially the I by O boards are the ones which help interface the man with the machine through the devices: a keyboard, through which you would be inputting; printer, through which he will be getting an output; and the same thing about the displays. There are different types of keyboard; for instance the CPU will be looking for some code from the keyboard, then that particular keyboard will be called an ASCII coded keyboard. There can be different types of codes; similarly there are different ways in which the information can be sent through the keyboard, and accordingly the keyboard types will vary. If you take the second category, say, printer, again there are different types – there will be a dot matrix printer; there will be a laser printer; there may be a daisy wheel printer; there may be an inkjet printer. Obviously you can see you have different devices here, and given a particular device, there are different types.

So what exactly wills the CPU is looking at, at this end, the bus end? That must be standardized; otherwise we cannot have a set of standard signals and set of signal lines here and a specific protocol. So this I by O is essentially an interface, and that particular interface will have to act like a buffer between the specific device, whatever may be the type, and the standard bus. This is very important.

For instance, if you take a dot matrix printer, the information that will be sent can be as slow as one dot or one point; even a particular letter or a character in English can be formed with a 5 into 7 matrix; that is why it is called a dot; whereas when you take a daisy wheel, the particular character will be printed directly; that is, we will be dealing with the information about a character at the character level here, whereas here we will be dealing with the information related to your character at a lower level. So these variations will have to be taken care of, and that will be done at the interface level. Now there is a laser printer here, there is an inkjet printer here – so all the information on this side will have to be buffered and on this side there must be some standardization. If you take this place, the next one – the most popular one is the raster scan; that is what we have even in TVs.

(Refer Slide Time 00:04:33)



What is a raster scan? If you take the screen or a display, this forms scanning across the screen in this particular manner; that is, from one end to another, and the time taken for this particular beam to move from this end to the other end will be about 52 micro seconds; and then for it to go back, it takes 12 microseconds. Now within these 52 microseconds, one line of information may correspond to one dot. That particular thing will be scanned; now in contrast with raster scan, you have other types, and in fact all these belong to the same family. Random scan is one where you are not forming a pattern like this: from left to right across the screen; randomly you can create a point here; another point here; another point here; another point here; and so on.

(Refer Slide Time 00:05:40)

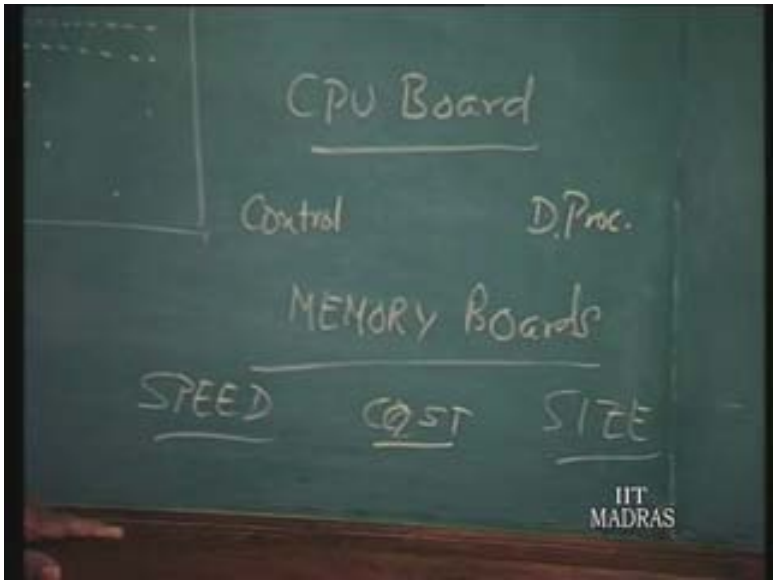
This is random; in fact the word random is used because it can be done in a random manner, and then it is in contrast with the raster. In a point plot also there is just one particular point at a particular time. The way the information must be fed to the display in these will be different. So these things will have to be taken care of; CPU may be sending just one character, let us say A, and that particular A will have to be displayed in different ways, depending upon the types of display. On this side we should not bother about that; it must be something uniform. We will talk more about this later.

On this side you see that there are variations but it is the same thing – random scan or point plot or calligraphic or vector display – they all are almost similar. For the present, note that there are lot of variations here; the same thing is true about this also. If you take a CPU board, depending on the application, the characteristics of the CPU board may vary. Earlier we said that they have a wide range; that is, at one end we have control applications and at the other end we have data processing application. The system that is meant for control applications will have one type of CPU board; for instance, in control, it may not always be just processing the data or carrying out some computation with reference to numbers. The input may also be coming in the form of analog signals. So the CPU board will also have to deal with analog circuits; the analog signals may also be coming from different places in a process control application, in which case the signals must be conditioned, sent, digitized, and subsequently, only then will they be processed. So the type of CPU board you have for control applications will be different from pure data processing applications.

We can go on talking about this, but not at this stage. Now if you just check memory, for instance memory boards; these memory boards again are of different types. The main thing in this is the speed with which a particular memory responds to a request from the CPU, the speed with which a particular memory responds to a request from the CPU. Essentially, CPU is the master; that is, usually the CPU will be the master of this bus.

As a master, it will request memory and the memory will respond with the data – may be instruction, or whatever it is. The speed with which it responds is important. The higher the speed of memory, costlier it is going to be; obviously starting with speed you can see the cost part of it also changes; so they are all interrelated.

(Refer Slide Time 00:09:45)



It all starts with the speed. Normally CPU is very fast and memory is also made up of electronic circuits; so we may say that CPU and memory interact at electronic speeds; and that is not the case with I by O. I by O usually is slow; but not always. For instance, if you take this display it is going to be very fast, but most of the I by O devices will be slow. The CPU requires memory and here we may have memory of different categories. So we may talk about memory hierarchy here – go back to the original when we were saying that the core of a computer system is a hardware. The core of the hardware will be the CPU; the CPU deals with some memory which can respond as fast as CPU.

The CPU does whatever it wants in that particular way so the fastest memory will be here; and then you have the slowest memory away from it; and then further slower ones outside – like this. That means basically the fastest speed or the highest speed memory will be here; that one generally is called a cache memory. One that is not so fast but fast enough is generally called drams; later on I will give you some information about this – how it came about because of the technology. And this is because of the position in which it is; this is how the name had come about. This dram or dynamic ram is generally called main memory. Earlier, it was called core memory because that was the main core. The outer layer is generally called secondary; it used to be called secondary memory. These days generally, we have the term storage resource for this particular one. Secondary storage usually consists of some magnetic devices, say discs – discs may be tape or whatever it is.

(Refer Slide Time 00:12:49)

If you work out the cost part of it, usually it is indicated as cost per bit of storage; bit binary digit is one bit zero or one cost per bit. The cost per bit will be the highest here and the lowest here. How about outside of this? Outside of this also we have what actually I may call the real world. The data may be coming from the real world through appropriate transducers and, for instance, in control applications. Analog signals come through transducers – which information also can come from here.

There are many ways in which you can organize the data; you can collect data; put them here, or here, or here, and let the CPU access. This is one way of doing it. The other way is that the CPU may access the data directly from the real world also due to various reasons. Essentially what the user is doing is that he is going to make use of the devices and provide the input/output; so he is also providing the data that can also come from the memory. So essentially the memory you have is also a secondary memory or secondary storage. Of late, main and secondary are dropped; memory is essentially this particular layer; storage means this particular layer.

Memory also stores; what is stored is also available as memory – that's the term. Let us put it this way: the cache memory is going to respond to the CPU at the same speed as the CPU. The response of the dram is going to be slower than that and of this particular one further slower and then, in the real world, it is really dead slow – quite often that is what it is. So we can think in terms of this: when we go out what we have is decreasing speed; and also decreasing cost. And because of decreasing cost what you have is increasing size; so you would find that the size of the costliest cache will be less and the size of the cheapest or the inexpensive storage will be more. We can think in terms of this, that is the cost part of it; the size part of it – I will not discuss cost because it will keep varying – and then the speed part of it. A typical figure can have cache and the dram on the main memory, and then the storage. It is only typical; it varies from time to time.

The speed of this is around 10 nanoseconds; that is, in 10 nanoseconds, the information or the data, one unit of data that the CPU wants, can be made available; whereas, this particular one may be about 100 nanoseconds and that usually is in milliseconds, some 2 to 10 milliseconds. The cost of it varies so I am not going to bother about that. The size of this is usually in kilobytes; say 1 kilobyte; 4 kilobytes; sometimes even up to 64 kilobytes. Dram will be in order of megabytes; one talks about 8 megabytes, 16 or 32 megabytes; and storage will be hundreds of megabytes; say gigabytes even of the order of hundreds of megabytes. So the memory hierarchy part of it is such that the CPU essentially will be looking for data in cache because it is very fast and so the data will be available as soon as CPU wants. Otherwise the CPU will have to wait. Whenever the CPU waits and does not proceed, obviously it becomes inefficient; that means we say the processor is not being well utilized.

In case the required data is not available, this will be loaded from dram assuming it is available there. In case that is not available, it will be loaded from this particular one. Now you may ask what does the processor or CPU do in case some information is getting loaded from here to here? CPU may be waiting, or it may be taking up some other program and carrying it out. The same thing is true about this – if data is not at all available anywhere here, obviously the required data is not available in CPU and so it will have to be loaded subsequently from the real world. That is about the memory hierarchy part of it.

There are other things also we can talk about the memory – in fact what I will call is based on technology, or technology-driven classification. For instance, I have already talked about this – in the dram, d actually stands for dynamic; it is a dynamic ram. As you may have guessed, it is another type of static ram. Instantly, a static ram will be faster than dynamic so invariably you will find cache memory consisting of static ram chips; that is the technology part of it. And dynamic ram is slow; that is what it is called dram, though essentially that is the memory part of it. There are other things also one can talk about here – I will not go too much into the technology as of now.

We are talking about the memory; at the same time let us not forget about certain aspects of the storage also. Storage as I said essentially is in fact the most inexpensive one and it was the outer layer. We are all familiar with the floppy diskette, that is, the floppy subsystem in which the media will be the floppy diskette and the hard disk subsystem, which is the hard disk – you are all familiar with that mostly. We will not talk more about it, except noting one important thing here. Though we generally call it storage, do not forget that they not only store data, the same thing can also be used for input/output. The files can be stored there for providing the input and the same way the output of the system can be stored in the form of the file and kept here. So storage is also for I/O; like in the case of the other devices, in the case of storage also we would need the appropriate interface. For the floppy subsystem, there will be one type of interface; for the hard disk subsystem, there is another type of interface.

We will now discuss a few points about I by O. You have already noticed that the devices that are there, whether it is for storage or I by O, they have different characteristics. So obviously, the interface circuitry is needed here – what will it consist of? Normally here we have the bus; I will just redraw this again – how the bus here and then the I by O part is essentially providing the interface and what you have for a specific device. And then, we have the device.

Let us say for instance if you have a disk subsystem, I will just call it system; generally the computing system that will be the subsystem, what exactly does this interface consist of? Many of you may have noticed that in the case of hard disk subsystem or floppy diskette subsystem, you always have some moving element; that is, the media itself will be rotating around. And then there may be an arm going over it; that means essentially these are electromechanical devices. That is, the interface circuitry part of it will consist of these two things in the interface circuitry part of it. One will be what you may call the device electronics part. This in fact has nothing much to do with the data that is going to be transferred over at this end. The device electronics part of it is essentially for taking care of the specific requirements of this device. Then you have the other one, which is generally called a device controller or just the controller, the controller part of it. Whenever we discuss a computing system and then interaction, it is this particular one which we will be concerned with, not this particular one.

Now for instance, the device electronics will include the appropriate circuitry for rotating the motor; for moving the arms; for positioning the reading head over a specific place on the media. When we discuss the bus point, we will not bother very much about it we will worry more about this controller part. This controller is the one which is going to collect the data from here from the input/output and pass on to the bus side and it will be passing it on.

This side must be uniform, because we are stating that if you do not have a uniform set of signals here we cannot have a standard bus. So the diskette controller or a keyboard controller or a printer controller has to see a uniform thing on this side and then take care of the specific characteristics of the respective devices on this side. I was telling that as a part of operating system there will be I by O routines. I also mentioned that these I/O routines are standard and part of the system; that is, one user need not keep developing. Given a particular device, the routine will be there and any user can make use of it. As part of that I by O routine, you would also find the software aspect; generally that particular thing is called a device driver, a device driver, which essentially is the software associated with it; of course it will be running as part of the operating system. There is one device driver for each disk subsystem or any of these devices. That means as far as the I by O interface is concerned, we have the device electronics to take care of the movement of heads etc. for that particular device.

Then we have the device controller, which will take care of the bus standards and pass on the information, and as part of the operating system, there will also be device driver. Driver is not hardware; it is pure software part of it – now that is one important thing.

In fact we may say that this particular device driver is the highest level of the software in the operating system for that particular device; that means one device driver for each device. Here there are again a set of commands and the actual user need know only about those commands while dealing with that device. That in fact summarizes the hardware part of it. Now we will just take a look at two other things – whenever we talk about computer system in general, of late we talk about two things: one is the stand-alone computers. By that what we mean a computer which is there and that is a computing system, let us say a desktop computing system, which has its CPU memory and I/O chords and all those necessary for a particular user. So generally what we have here is a single user in mind. In contrast to this, because of the technology advances, now we also talk about network computers. Here again there is networking and we also have these independent systems, but then the independent users can communicate over these. There are other aspects also; I will presently tell you what they are.

So having discussed stand-alone computers and networking, we now talk about the over all system, the network computing system. What are the various purposes of networking? As anyone can guess, first it is to establish communication among the users. There may be different reasons for this. Let us say in a typical office, the officer may be having one system on his desk, the secretary may be having one system on the desk, and then there may be other people and we essentially establish communication among these. In a typical design office, a technical manager will be having this and then there may be some three or four different design teams, may be hardware design teams or software design teams. Given a hardware design team, there may be circuit designers; there will be PCB designers, and so on. So having a network system, you can see that communication among these batch mates in a given batch can be established. The other need for network computers is basically to share resources. What are the resources we are talking about here? It is possible that some of these say devices may be very costly.

(Refer Slide Time 00:31:04)

Let us just say supposing we have a printer and it is a dot matrix printer. These are fairly inexpensive; suppose you go for very costly printers; let us say laser printers. Suppose there is one laser printer; you don't need a laser printer on each desk, like you have a computer on each desk. Among the group of people, you can have this particular thing shared. Now we also talk about other things like for instance file servers. That is, all the files are available in a particular system and that can be accessed by other users.

Generally these resources are shared for two reasons: one is the data is available in common and can be used by others and the second thing is that you may be having very expensive things and you cannot afford to have one for each user. The third is basically for remote access. Suppose my system, with its local information in the hard disk is available elsewhere; and through all the network I can remotely log in to my system and then access the data – that's the third aspect of it. Here actually it is very much like having an independent system of your own, but physically you are not in the place where the system is. So making use of the network part of it, you remotely access your system so that, in essence, these are the set of reasons for networking of computers.

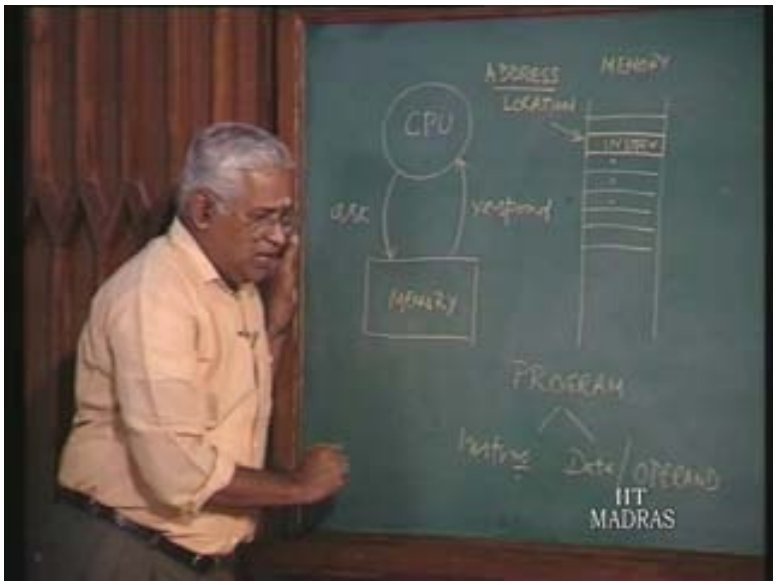
I was mentioning about multimedia computers earlier – one typical use of a multimedia computer is for video conferencing. That is, the conference sides are graphically distributed in many places and then, let us say there is a conference or a meeting NAND let us say some four people are participating from four different cities. A person can raise a question, which is the audio part of it; it goes to the different places and in response to this, from another place another person is showing a video clip. And that again goes to all the places. While watching a video, for instance, simultaneously from two different places, two different users may raise some question, which means two audio files are being generated in two different places. We need very high speed network to support such things; but I am just talking about a typical application and of course, soon such applications will be found in use.

I hope by now you have an overview of a computing system and then specifically some aspects of software and hardware. In fact we can go on talking about it – as you may have noticed I was just mentioning the term bus; I have not gone into the details of it; in fact there is more to it. We will possibly take up this later on. Now essentially you would have observed by now that there are CPU, memory and I/O. If I leave I/O as extended memory we have only CPU and memory. In a computing system what we have is a CPU and the memory. So these are the two things and as I said I/O is extended memory so I will just check it as part of that. The CPU as master is going to demand something or ask memory for something, and memory will respond to that. Precisely that is what we have – that is why we said CPU is usually the master and memory is usually the slave. So for us to get an idea about what in fact goes into the CPU and then later on also take what goes into the memory system, we should really know how exactly the interactions will be going on. What it asks and how it asks and how the memory responds – we will see all these in detail. For the present, I will say that the CPU will ask memory for some specific procedure; let us say some slots are there in a memory.

Then into these slots, the data or instructions will be entered; these are the instructions of CPU. In fact what we are what we are talking about is that the CPU is ultimately going to execute a program and the program essentially consists of instructions and data. We may also use the term operand for data sometimes; so this is what the CPU will be asking. We are assuming that for every organization there is a program; internally the CPU will be either asking for the instruction, the memory will be responding by passing on the instruction, or later CPU may be asking the memory for the data or the operands, in which case the memory will be responding with that.

Here it is arranged in such a way that we may take one of these and then each part of this memory, such as this location can be assigned a specific address, which means this address is unique for this location. This is a very common sense approach. In fact I always hold the view that CS not only stands for computer science but it also stands for common sense, for everything that you find in computer science you can always find a common sense thing to that. Now this is something like a street consisting of houses and each house has a specific address. So we may now say that the CPU when it asks, the memory is going to pass on an address and in that particular specific location, there is some instruction and that instruction will be passed on as a part of response to the CPU by the memory.

(Refer Slide Time 00:39:11)

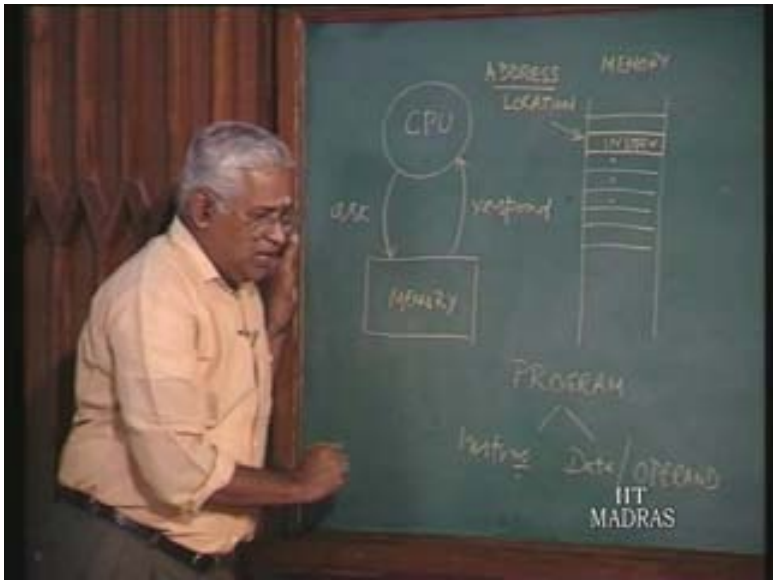


So essentially this is what is going on because a program will consist of instructions which the CPU is capable of executing and it will also include the data which the user wants to be processed. In other words essentially, you have two major phases of operation by the CPU. The two major phases are: fetch the instruction and the other phase of it is: execute that instruction. In this fetch and execute phases, one particular instruction will be fetched and it will be executed. The execution really depends on what the instruction was.

For instance, we may have an arithmetic instruction, let us say, add. Suppose the instruction is for addition, then a minimum of two data will have to be passed on, A and B. During the execute phase, the data A and B will also be fetched from the memory because we assume that it would have been stored earlier in the memory and these will be also fetched to the processor. Remember it is the processor, which is going to execute; memory will only be responding with either instruction or data. So when we talk about fetch, essentially what we are talking about is the instruction fetch; and fetching an instruction is always the first thing to do.

After it fetches the instruction, as part of execute phase, the data will be fetched and then the actual operation will be also carried out. So instructions fetch and execute and for part of the execution of the instruction there may be data fetch or operand fetch also. This whole thing is called an instruction cycle. Precisely that is what the CPU always keeps doing; that is, the instruction cycle consisting of a fetch cycle and execute cycle.

(Refer Slide Time 00:41:59)



How does it go about? Let us say the CPU will be placing the address of instruction on the bus and the memory will respond by putting the contents of that particular address, which in this case is instruction on the bus; CPU will get that. That is part of the instruction fetch phase. Then the particular instruction, which has gone into the CPU, will have to be interpreted. So usually the interpretation will also be done before the execution comes; so we may say fetch; interpret; sometimes you have a different name for it: that is decode; and then execute. This is what you have essentially in any instruction cycle.

We are not bothered very much about this interpretation or decoding for the simple reason that it is not going to take much time; whereas the fetching of instruction and the execution will take more of CPU time. The decoding of it will be done almost instantaneously or it takes much less time but nevertheless it is very much part of it. So an instruction is fetched from the memory; it is interpreted or decoded.

We say decode because the instruction that comes is in a coded form. So that code will have to be understood and that particular code will be unique to that particular processor. That is why it is called decoding, and interpretation is basically, understanding what that instruction says. So fetch the instruction, understand what that instruction is or decode that particular code, and then execute. Now as part of the execution, I have already mentioned operand may have to be fetched because essentially program consists of instructions and operands or data. So as part of the execute, again data fetch may be there and after the data is fetched from the memory – because may be the data is in another place – the CPU will have to place the address of that location and the memory will pass on this data to the processor. After the processor gets both the instruction and the required data, it depends on the instruction. Then it really will be executing or we may say it is going to process the data. That is one instruction cycle.

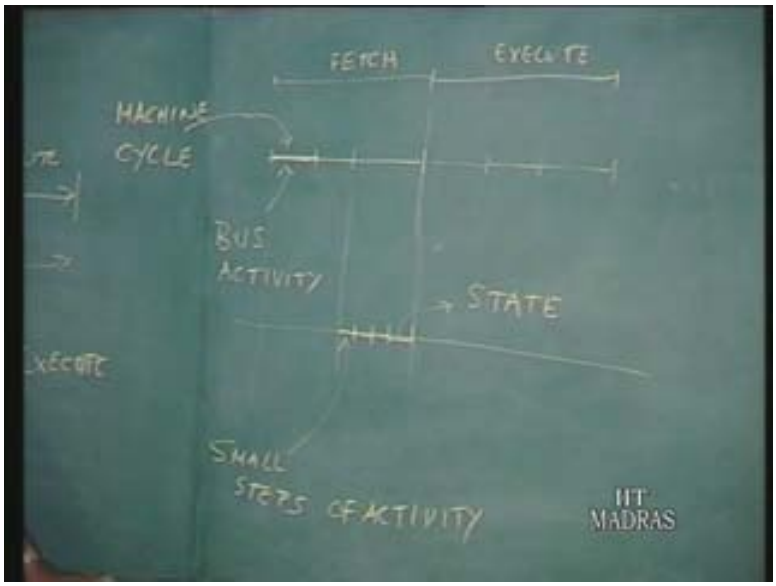
Understanding of this is very basic and very important because again and again in this particular series of lectures, I will be talking about this instruction cycle. Now having understood that we have an instruction cycle, which consists of fetch and execute, we can see that the instruction fetch may consist of fetching possibly one or two or three units of information or data. That is, for instance, though marked like this, it is possible that the instruction is occupying more than one location, in which case if this is the unit of data that memory can always put on the bus. Or if the CPU is capable of getting, then assuming the instruction is put in three locations, the CPU will have to place this address and then this address and then this address. That is what I have indicated here. It so happened that I have just assumed the instruction is fetched in three phases.

What does it do every time? The CPU addresses, and for addressing and getting the response or data – remember one thing: whenever I say data it is not always only the data it may also mean the operand. For instance, instructions are also referred to as data. So be careful whenever we use the term it is so in almost all the books also for instance. The CPU asks for some data and it is passed on. The first thing it will be asking for will be the instruction that is one form of data. Now we find that instruction has come in three steps and then similarly, possibly the particular instruction may again need to access the memory thrice. So each time what is happening here is it is fetching one unit of information from the memory.

This particular one takes place over the bus and so we can say that in this particular one, there is a bus activity that has been initiated and it will be concluded. So we may say that in this particular duration, the CPU initiates data transfer by addressing memory and the memory passes on one unit of information or data whatever you call it in this particular case, one part of instruction. When bus activity takes place this particular one is called a machine cycle. That is, during one machine cycle, which is part of the overall instruction cycle, the CPU has placed and got some information and then each of these again can be split. I will just take any one of these, which is true for many of these. Also some minor activities will be going on.

Now I am just showing as if that there are four activities, minor activities going on in this. These are some small steps in the entire machine cycle. Given a machine cycle some small steps of activity are going on.

(Refer Slide Time 00:49:51)



To understand that particular one, we really have to go into the details of the processor, which is what we will be shortly doing. Now we can summarize that essentially the program is executed by taking instruction after instruction. The instruction cycle consists of machine cycles and in each machine cycle, there is some bus activity and each machine cycle consists of some small steps of activity. Specifically, each one is called a state of the processor.