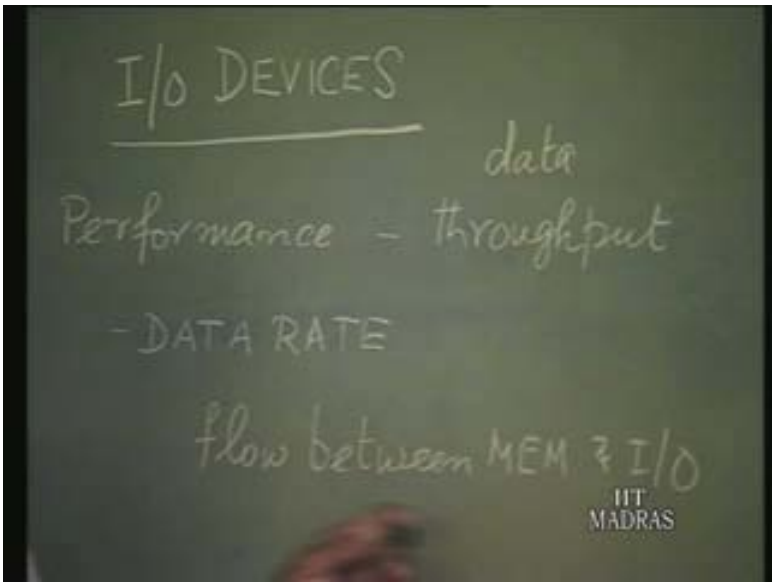
Computer Organization
Part – III
Prof. S. Raman
Department of Computer Science & Engineering
Indian Institute of Technology, Madras
Input/output
Lecture – 29
I/O Devices

In the previous lecture we looked at the evolution of I/O. We saw actually how, from the processor itself, the device interface had evolved so that we have a main processor and a similar processor for the I/O, its own memory. The I/O processor is actually taking care of the I/O. We also surmised that possibly the evolution of I/O would be in such a way that for each type of I/O there can be one processor. Now let us take a look today at the I/O devices, a few devices. It is not a full coverage on all types of devices; I will only mention just a few of them just to get some idea about the issues involved, so that we may have a better understanding for the study of these or preparing ourselves for the study of the bus that is interconnecting the CPU memory and the I/O. Normally while talking about the processor I was always talking about the performance of the processor and how exactly, for improving the performance of the processor, we had also looked at the hierarchy of the memory.

(Refer Slide Time: 04:15)



For instance the cache came merely to see that the CPU throughput will be more. So essentially it is that performance we are talking about. To improve the performance of it we will keep doing anything at the system level. Now when we talk about the I/O devices how exactly would we characterize the performance?

This is also something we will take a look at. The moment we talk about I/O, essentially the rate at which the data flows is one specific thing that comes to our mind because performance in the case of I/O will translate in terms of what we may call a throughput. In the case of I/O we have to specifically talk about the data throughput, which is nothing but the rate at which the data can flow between memory and I/O. The data rate is something concerning the flow between memory either directly or under the control of the processor. We are not talking about the direct memory access here – flow between memory and I/O. What exactly is the rate at which the data flow is there? The I/O device's performance will be given in terms of that. So we talk about high speed I/O or a low speed I/O or a medium speed and the range of it – that is one thing. Another one related to this I/O is invariably that the disk system is involved.

How fast can you access a given file because that file may be an input file or an output file, which you stored in the disk? When we talk about disk, remember that it not only serves the purpose of storage, but also can serve as input and output device; it has a dual role. In terms of these disk systems, we would talk about the rate at which the disk accesses take place – this is another thing. We will talk more about this particular one when we take up the disk sub-system. So the I/O performance will be measured in terms of the data rate, essentially which will give an indication about the flow rate of data between memory and the I/O. As I said, when we say memory and I/O, it is not directly accessing the memory. That would be under the control of the CPU. The second one is the disk accesses. Now when we talk about the disk access what we have in mind is how quickly an input file can be accessed; how quickly an output file can be generated – so these are the things. We have two things to talk about. We will talk about this later because specifically we have to know something more about the disk before we can talk about this.

I thought I would mention this performance issue because ultimately the main aim of a computer system design is to see that with the least cost what is the maximum performance that you can get from that overall system, say, from CPU; from memory; or from I/O. Finally it will get linked to that, and invariably you will find that the rate at which the data flows, the rate at which you can access a file – all these things will play a role. In fact with the limitations of one processor, another issue will come. If that is the bottleneck, how about having more than one processor? Does it contribute? For instance in place of one processor if we put two processors, does it help? Does the performance get just doubled or not? We will invariably find it out. You will find a figure of improvement only by 1.6 to 1.8; it will never be 2 because of having two processors – we will talk about all these things later.

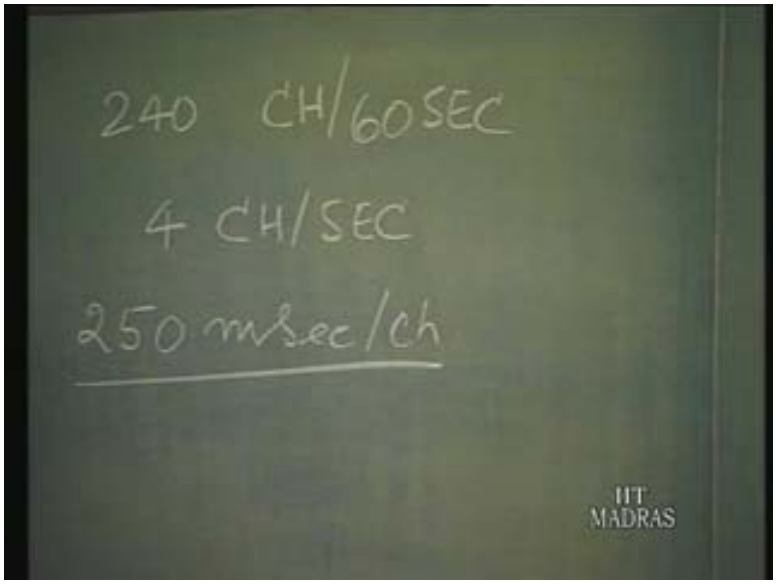
Now specifically we will go into some aspects of the I/O devices per se, meaning we are going to talk about some details which really we need not be very much concerned about. That is we talk about the lowest level and then I will say these things are buffered in the I/O interface part and the system will not really look into these low level details. Now the simplest one would be to start with, say, keyboard. The first thing we can think of; so let us just take a set of keys forming a keyboard.

(Refer Slide Time: 11:32)

Now this is typically an input device because whenever we talk about an I/O device these are the things that would matter. What exactly is the unit and what type of transfer is it? Is it an input or output? Who exactly uses it? Certainly a man, some human being, is going to use this – the third aspect is, what is the rate at which the data flows? So essentially these are the three dimensions along which any of these devices will be categorized. That is, the transfer is an input or output and who exactly is the user, meaning some of them may be controlled by the system itself; the man may not be at the other end. Now in the case of keyboard, a man is going to use it and the rate at which it flows. We take the keyboard – the keyboard after all is not very much different from the typewriter. So we it operates at a rate of about 50 words per minute; this is on the high side – it could also be 60 or 70 – so if one can type at the rate of about 50 words per minute, that is the highest rate. Of course the lowest rate can be anything – I can type 1 character today and I can type another character next day.

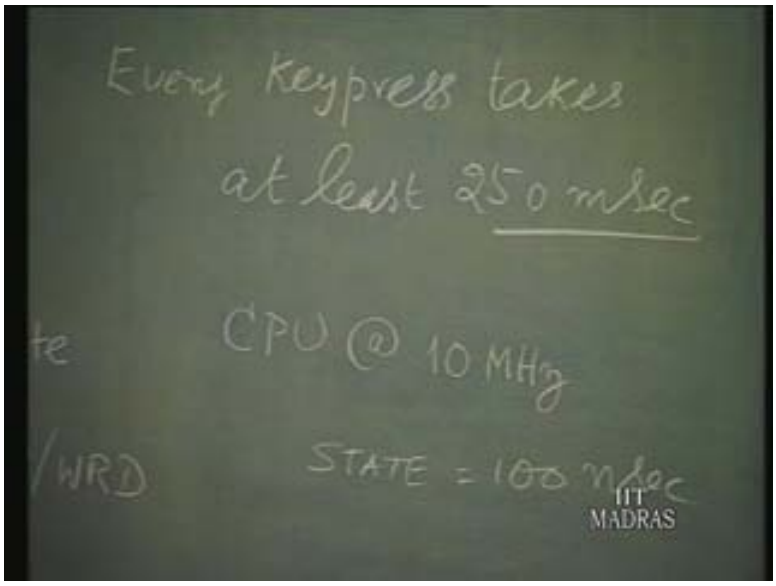
So this is about it. If you want to see the average may be about 40 or even 30; so here it is 50 words per minute. We have to see the number of letters or characters when this is translated, and then we will see in terms of seconds. On an average, a word consists of about 5 letters or 5 characters. So given this particular one, this would mean there are actually 250 characters per minute. I will just round this off to 240; you will know why it is approximately 240 because a minute consists of 60 seconds. So approximately, we have 240 characters per minute, that is, 240 characters per 60 seconds.

(Refer Slide Time: 12:20)



So approximately it is 4 characters per second; a second consists of 1000 milliseconds. So in 1000 milliseconds, we get 4 characters – what does it mean? It means essentially you take about 250 milliseconds per character; so for each character, just one key is pressed. There is a delay of as much as 250 milliseconds before the subsequent key will be pressed; that is, in 1000 milliseconds, there is 1 second. There are 4 characters, so 250 milliseconds between two key presses; we can say every key pressed on the key board takes at least 250 milliseconds.

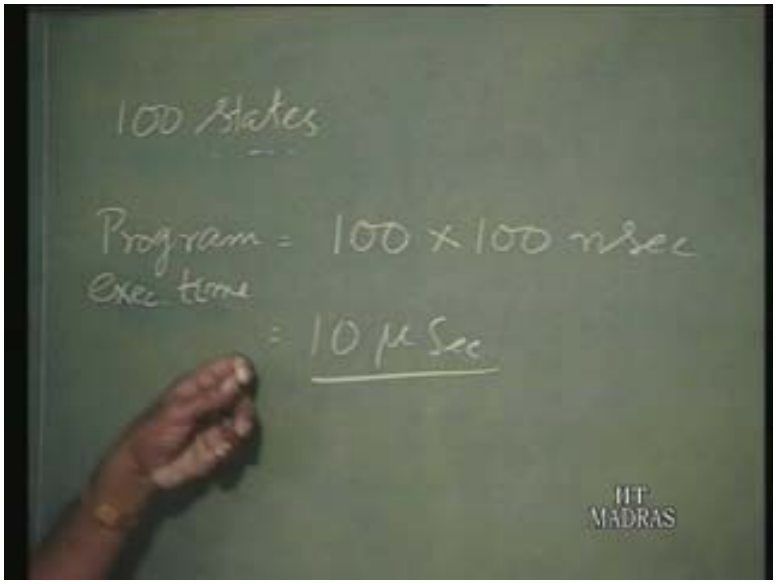
(Refer Slide Time: 15:01)



This is the rate at which the data flows; so that is how you arrive at it. You just imagine – 250 milliseconds is a very long delay. It is a very slow one; there is plenty of time for the processor to process. If you assume that you have a CPU which runs at 10 MHz, 10 MHz itself is a very low frequency.

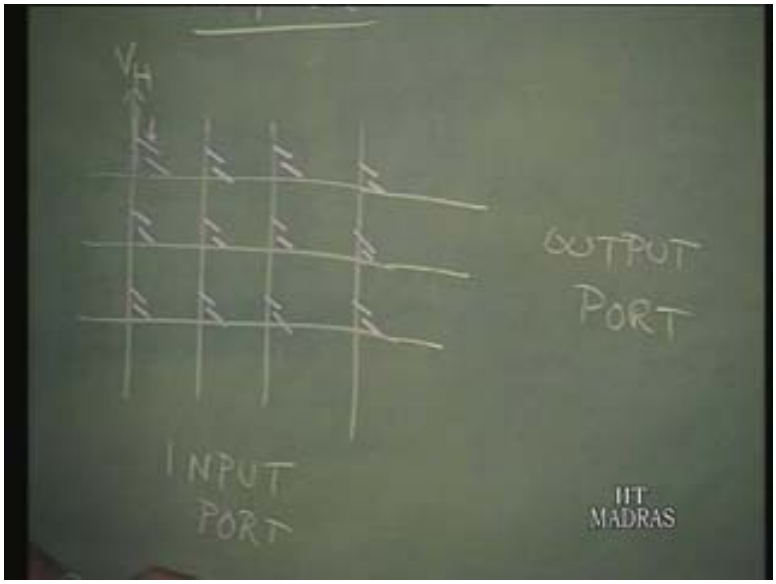
So 1 MHz would be 1 microsecond, 10 MHz would be 0.1 microsecond, that is, 100 nanoseconds. That is the state in 100 nanoseconds. So suppose you have a program which processes the input because every key press must be linked finally with some character; some signal must be generated. So we need a program, which will have to understand which key has been pressed on the keyboard. This would mean that for this state duration is 100 nanoseconds. Suppose we have a program which makes use typically of some 10 instructions; 10 instructions are there in the keyboard or what is called a key input processing program or keyboard routine. Now if each instruction needs about another 10 states, the whole thing would have minus 10 instructions into 10 states minus 100 states.

(Refer Slide Time: 16:28)



The program goes through 100 states and each state takes 100 nanoseconds. For the entire program to analyze which key it is, we have 100 states into 100 nanoseconds. That will be 10 microseconds. It means the key that has been pressed can be identified in a matter of about 10 microseconds, and the time we have between two keys is 250 milliseconds. So there is plenty of time; this is a very low speed device, input device. Let us see about the keyboard mechanical arrangement; we just looked at a data transfer rate. Now typically, we talk about what is known as a matrix. I will just assume a matrix of 4 into 3 keys and at the intersection of each of these, I will put one key.

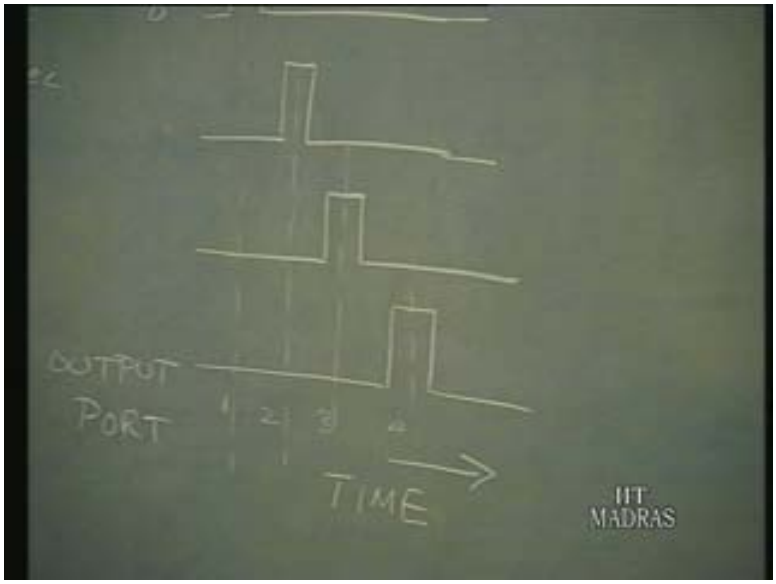
(Refer Slide Time: 19:10)



So if I put keys in this manner everywhere, what do the two things represent actually? It is something like a relay contact. So when I press a key in this direction, the contact is going to be made and if I return it to some voltage, V high, then that particular thing is available on this line. That is, I am energizing this line. So when the key is pressed, this particular one is passed through the key to this. Now this would indicate that I have to energize these lines, these four lines, and I am going to monitor the signal on these three lines. Generally these four lines are called input port lines and these three will be called output port lines. So at the lowest level, the program essentially consists of sending some signal here and then seeing what signal comes on which of these output port lines. There is a small problem as we can see. When this line has been energized and key has been pressed, then this high voltage is made available here. We will assume that only one key has been pressed.

Of course, if this key also has been pressed, then this voltage is available not only on this but also on this. What we think is simultaneous pressing of two keys; but from computer point of view it may not be simultaneous because there is going to be plenty of delay. Now there is another confusion here. What is that? When this key has been pressed, we said the signal is available on this line but it is not very clear whether it is because of this key or this key or this or this – which key pressed has really caused that. So that will have to be resolved. What is done is first energize this line and then, after some time, energize this line; and after some time, energize this line; after some time, energize this line. In other words this V_H that has been marked can really be done like this – that is this level is V_H.

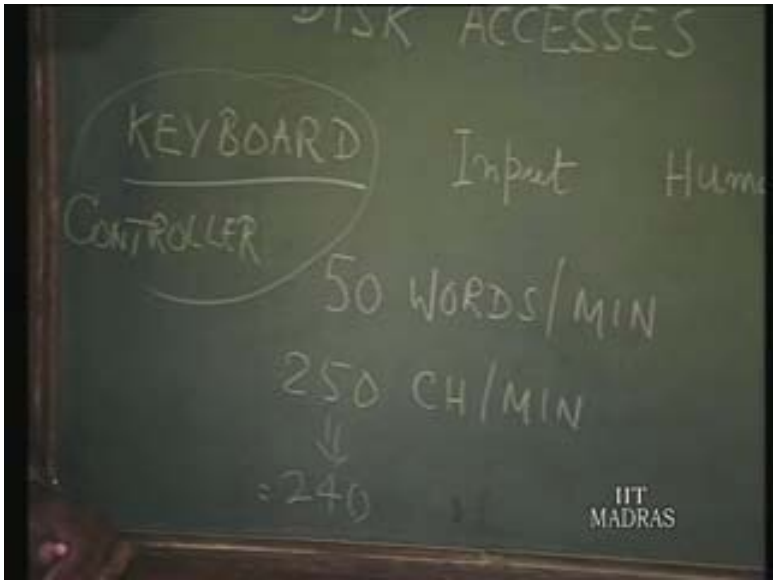
(Refer Slide Time: 21:41)



What we have here is the time axis; actually here is the time axis. This particular signal at this instant of time will be sent to line 1. This would correspond to line 2, this will correspond to line 3, this to line 4. Now there is no confusion as long as the system can just check. It should not miss the signal period; that is, this particular thing should not be missed. The output port must be monitored here, here, here, and here; then it will be very clear. There will not be any problem. So this is the way you resolve which of these four keys pressed. There is no problem about these three because they will be coming on three different lines. So which of the four keys is pressed is the confusion mainly because you have a common line. Essentially there is no confusion about generating a set of signals on this and putting it on this and looking for the exact instant the output signal is generated because this time instant is going to tell. In other words the resolution among these four is resolved temporarily or you make use of the time and resolving is done spatially because in space we have three different outputs.

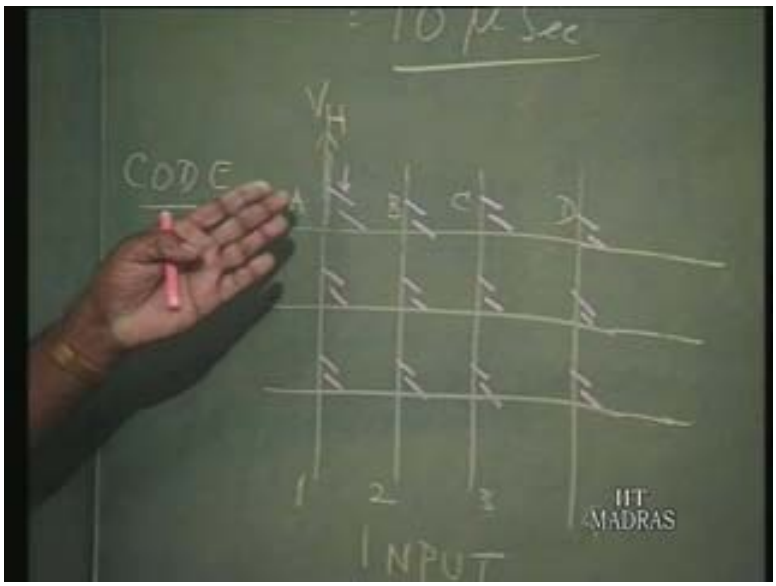
In other words, by generating an input word and by looking at the output word, we would be able to identify which key has been pressed. May be this key corresponds to the character a; and this to b; and this to c; this to d; or whatever it is, or may be it is numeric. But what I have shown here is only for 4 × 3; in other words only for 12 keys. Now as you can see, here the computer is not going to take – the program part of it is not going to take – more then 10 microseconds to see what this is or which key in fact has been pressed. And nobody can press the key faster than 250 milliseconds so there is plenty of time. So now this is the way it is done at the low end, but really speaking, the kind of thing that you are familiar with is an I/O interface. If the keyboard is the device, what is the interface for that? The interface for that would be a keyboard controller. The keyboard controller is interfaced, which is the one that is going to interface between the keyboard and the bus.

(Refer Slide Time: 24:49)



I have worked out the details – all these things will be taken care of by the keyboard controller. That is, there will be hardware as well as the software part associated with that. Really speaking, the user need not be concerned with all these details but I am showing these to indicate to you the data rate and how exactly the whole thing can be organized. From the keyboard controller point of view, the keyboard controller is directly going to identify which key and then give the unique code associated with that key.

(Refer Slide Time: 25:35)



The unique code associated with the key is what is wanted by the processor so that it can identify what key was pressed; it does not really need all these details.

So all this I have worked out is purely from the point of view of the controller design. Now this is how the keyboard part of it is handled typically by one specific arrangement. There are many arrangements by which one can identify which key has been pressed and so on. Again I have not gone into the details of the code that is generated here. There can be different codes that can be generated; that is all about the keyboard. We were talking about the key board interface details.

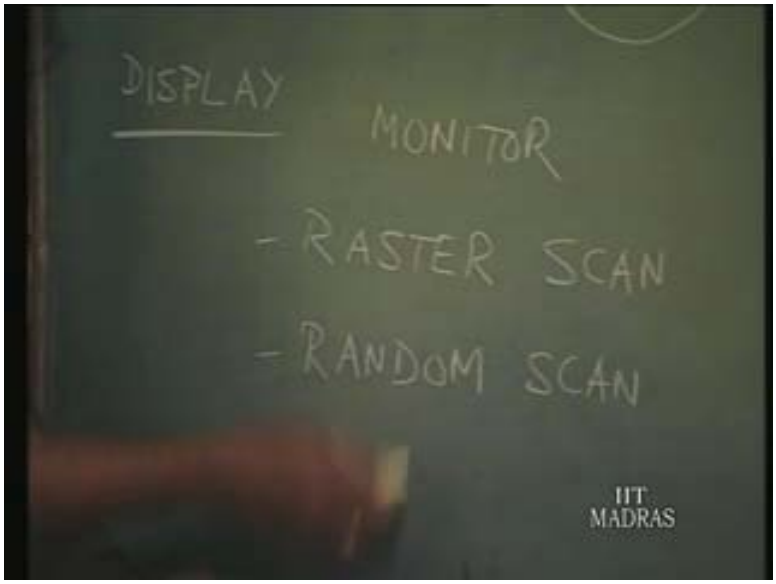
Now let us go back to the general picture where we have the CPU, memory and the I/O interface, which interfaces the device with the bus of the system. If the device is a keyboard, for instance, as we have seen, then we are talking about this I/O interface as the keyboard controller. In fact what I was working out as the input and the output port is the hardware part of the controller. From the system point of view or from the processor point of view as we had seen earlier, it is only the CPU that will check whether this device is ready, specifically, whether this I/O interface shows that the device was ready and had generated the data and what is the data it would have generated? It would have generated a code corresponding to the key that had been pressed.

So the CPU will only look at the status of this and see whether the data is ready, that is, the code for the key pressed and then it will read the code. Other details, that is, the interface, the input/output port and all those things we were just seeing a few minutes back are all part of the interface.

So the CPU will scan and then get that code may be by a programmed I/O technique or by interrupt driven technique – the code for the key that had been pressed. All those things are taken care of by the controller. Certainly I talked about only two things – programmed I/O and interrupt. I did not mention DMA because DMA is essentially for very fast device and we had seen how slow the keyboard device is. If this device were a fast one, then it may be by DMA also. If you have a printer, you will have a printer controller.

If you have a display you need a display controller, and if you have a disk then you have the disk drive, disk controller, and all those things associated with that. But from the bus point of view or from the system point of view what we need is some code for the data which will have to be transferred. As the second example, let us take display itself. There are different types of displays, starting from the simplest – LCD or LED display; seven segment display is too simple – I will not talk about that. Let us just take the monitor, that is, most of the computer users will be using the monitor. What do you have on that? Essentially the principle is not different from what you have on the TV display.

(Refer Slide Time: 30:38)



If that is so then, as we had mentioned in the beginning of the series, it will be a RASTER display or a RASTER scan display or just a RASTER display. But there may be other mechanisms also. In fact we have also mentioned about a RANDOM scan display or many other terms also were introduced, meaning in the RASTER scan, if you just take a look at the screen, then you talk about a line of display.

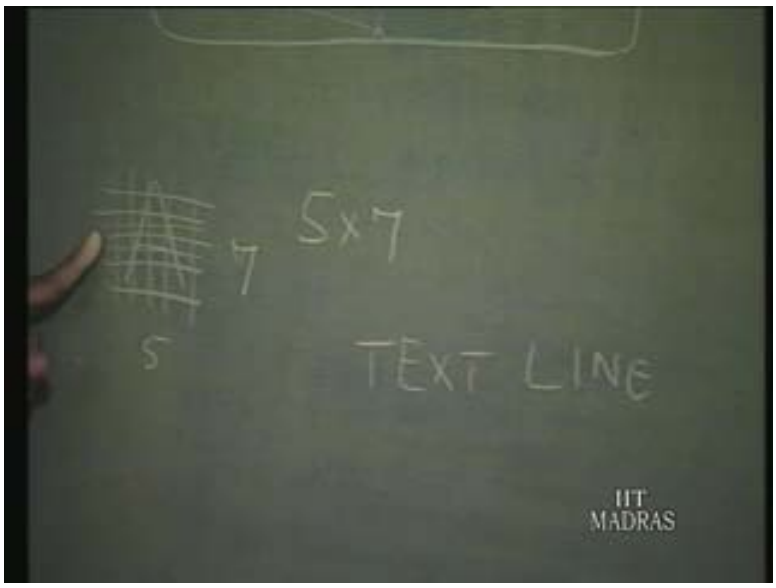
(Refer Slide Time: 33:17)

That is, just corresponding to a point and series of points on that, and I am going to just show here with some space in-between but there may not be any space. Certainly there may not be much of a space as shown here.

And then suppose on this display you have a line of characters, then we would need quite a few of these RASTER lines. Generally, when you take a character, for instance say a or b, can be expressed in terms of matrix say 5 into 7 and so on. And then you can take those specific points. I have just shown 5 into 5 here, but a better resolution will be 5 into 7. If I do this particular one with 5 columns and 7 rows, then this would mean that I need 7 such RASTER scan lines on this. Let us now just take one line – this is how it is going and that scan line will come to the end and then it will go back, not directly but something more is involved in this. In the case of a RASTER scan there are three points; display is an output unit like keyboard. Invariably human beings use that. The rate at which something can be shown and at which the data must come is the output. So CPU must generate the data, may be CPU or the display controller.

Finally this is where we have the display and here we have the display controller. If you have the RASTER scan display here, what we are talking about is the appropriate controller for that. Generally it is called CRT controller itself because it is cathode ray tube. For this particular arrangement in which the character can be shown, each line of display is the text line; for each text line of display, I would need a minimum of seven such display lines plus at least one in between two text lines.

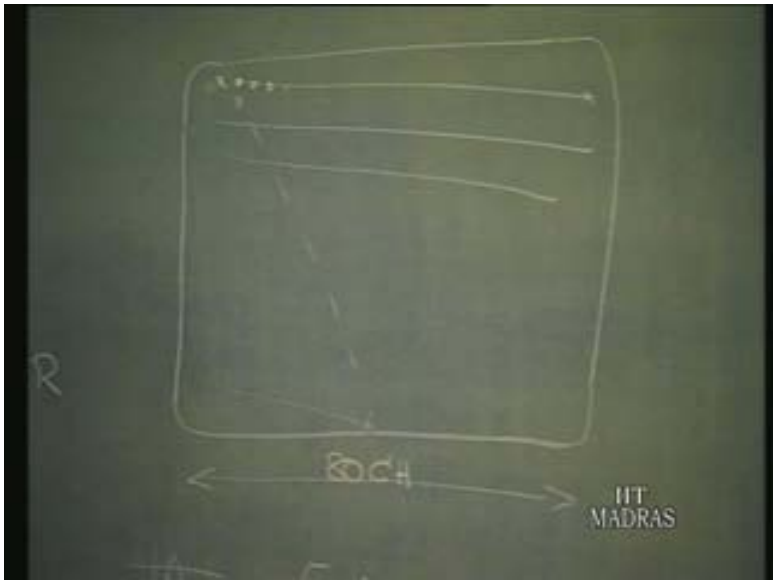
(Refer Slide Time: 34:17)



On each particular display line, it is going to be either an illumination of that point or non-illumination. In other words, by putting somewhat like this, just imagine that these points will be closer. This is how I generate the display. In the RASTER scan for normal electronic beam to go from this end to that end, we can just say approximately it will take about say 52 micro seconds. For this to go from here to here, it is going to take another 10 or 12 microseconds or something like that. So we say that this particular one swing back display will be inhibited there.

In other words the blank part is called a horizontal blank. I did not explain horizontal blank earlier; when it goes from here, there will be a vertical blank. I did not go into those details; nevertheless we can say that in a total of about 64 microseconds we have to keep generating these points and then displaying. Otherwise the persistence of vision will come and then you will not be able to see that character anymore. Well, assuming that across the width of this screen I am going to have say 80 characters and already I have said there are 5 points I might need for each character in the horizontal direction. I would need one extra for space between two characters. So there are 6 dots into 80 or 480 dots overall. These 480 dots will have to be generated in the time of 64 microseconds.

(Refer Slide Time: 36:49)



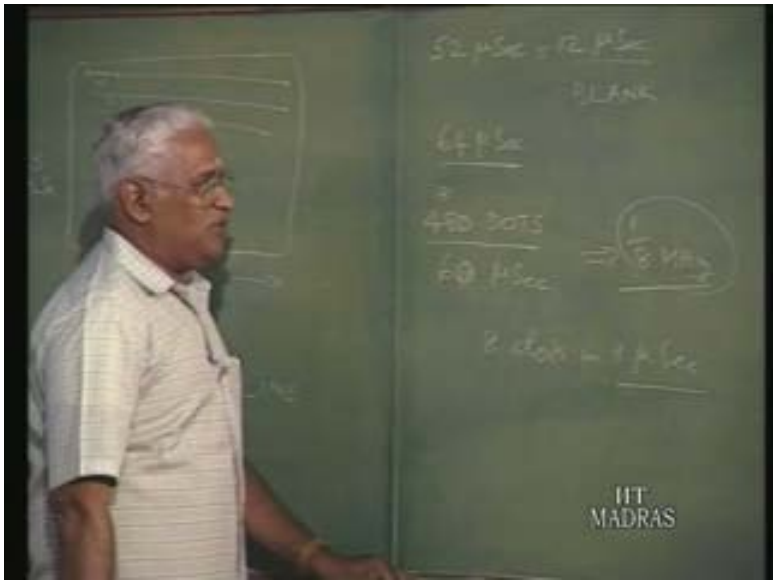
(Refer Slide Time: 37:44)

Really speaking, it is 52 microseconds, but then you also have this. So now you can see what we are talking about is a microsecond. I have just taken one line that is some what like this either this or this. It will have to be generated for about 25 text lines.

(Refer Slide Time: 37:55)

Each text line will consist of these 7; 7 plus 1 is equal to 8; and so on. We have to take that into account while working out the speed. We have to work out everything – it is not 480 dots; they will have to come in 64 microseconds. But then, for the electron gun to come back to display, it is going to take the whole screen so we have to take that into account also and work out. Nevertheless in 64 microseconds, we have 480 and then we have 25 text lines and so on and so forth. This whole thing will have to be taken into account. Now you can see microseconds would mean so many mega hertz so it is approximately 60 microseconds. We have 8 microseconds and this will correspond to 8 MHz This is the frequency at which these things will be coming. This will lead to eight dots in 1 microsecond. So in 1 microsecond, there is one dot, which would mean 1 MHz; 8 dots will be 1/8 MHz what did we have earlier? We had something like the rate 250 milliseconds; so you can just work out the time.

(Refer Slide Time: 40:48)



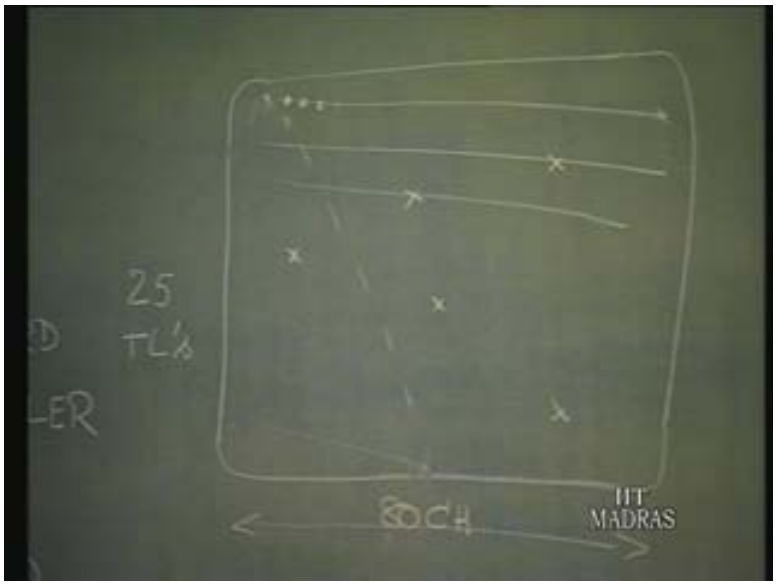
This is one of the fastest devices and I have not said about all the points in this display – that is one thing. This entire thing will have to be stored somewhere and then there are other calculations we have to work out; that is, 30 frames per second will have to be generated. All these cannot be controlled by the processor so that the data can be moved to this. Invariably, this being very fast, the display controller or in short CRT controller, will need a data buffer for the entire thing. In other words we need memory just for that, display memory; in fact it is a buffer with image memory.

(Refer Slide Time: 42:34)

This storage, in fact very fast storage, will be needed as part of the display controller and here the CRT controller and the DMA will also be used; because it is a very fast one the processor will not be generally be able to handle. What we have is bit by bit, so how many bits are there for the entire thing? In other words whatever is going to be displayed each bit is one RAM cell for each bit that is displayed. It is also called bit storage. These are all the things that would go into the controller part of it. From the processor point of view, because it is not going to deal with the data directly, sometimes if this data will have to be changed, then the buffer will have to be updated appropriately and, being very fast, the processor will not directly go in. So at some point when the display is ready to display something, CPU will be immediately cut off; the DMA and the CRT controller will take over; and then whatever is necessary will be displayed right from that bit image storage on to this monitor.

Now regarding the random scan, what is the difference between these? In this case of RASTER scan, only in a specific way – I have shown here line by line – the scanning will be done and there are also other details. So any book on TV technology will talk about interlacing and what not, whereas in the case of RANDOM scan, a point can be displayed here and next point can be displayed any where here and another point here; another point here on a different line; then the next point here and so on and so forth.

(Refer Slide Time: 44:48)



That is it is done randomly as the name itself indicates. What does it really mean? It really means that at any instant, one point is displayed and that is the reason it is also called a point plot display. That point is displayed just by giving the x and y axes of this: x axis part of it and the y axis part of it. In other words, for the controller for the point display, it is enough if it has two data buffers, one for x and one for y; but there will some problem because we may not be able to refresh that buffer very fast. If we have only one for x coordinate, call it x data register and another as y data register – suppose we have only these two, then only one point can be displayed.

But then before another point can be displayed this will have to be over written. So it may not be really possible but that is the minimum that you need. By giving the x and y and the third one and asking for a display, which is in fact a command, it would go into what is called the intensity; a command for making the particular thing intense, more intense or less intense, which is, display or no display kind of a thing. Depending upon the kind of the display – whether it is a RASTER scan or a RANDOM scan, the controller will appropriately vary. Then, by giving the bit image storage like we had in the RASTER scan, here you can have series of x and y registers or an array. Really speaking we have to talk about array of registers, array of x and y register in a specific sequence so that the subsequent ones can all be stored. So we talk about arrays, array of registers, x y registers.

(Refer Slide Time: 47:08)

The minimum is just one pair but then we can have multiple pairs of these registers and that is what we will be having in the controller, this particular display part of it. In fact I have skipped a lot of details; I am just giving you some idea about the difference between these various units. Now, for instance, let us go to the third one the printer.

(Refer Slide Time: 50:36)

Similar to what we have seen here you can print dot by dot or you can print at the character level. Now if you go into print it dot by dot, then similar to the display we can have a dot matrix, because what you have here is a matrix; in fact what you have here is the character represented by a set of dots, specifically by a set of matrix of dots. You can go on this way, so the dot matrix printer will print only one dot at a time. So really the head will keep moving over this; it will just make an impact on the paper, wherever it has to create a dot. Now another one is at the character level; for instance, say a line printer or some printers, the Daisy wheel printer or any other printer – these will print at the character level.

That means a code will go to the printer and in the case of a line printer there would be actually a chain, the chain on which you have these various characters marked so that right under the print head, the appropriate character will be positioned. Similarly in the daisy wheel, it will be rotating. There will be many characters and it will be rotated, so that right under the print head whatever comes, it will be printing. Now as you can see, dot matrix, line printer, Daisy wheel printer are all impact printers; that is, finally a hammer will come and then makes an impact with the medium, that is, the paper, on which printing takes place.

So you have other types of printers also. Now here we can see that this is going on at the character level, whereas here it is at the lower level, and that is at the dot level. In general what are the characteristics of the printer? The printer is also an output device; obviously the print out is going to be used by the human, but the output is going to be generated by the system. What is the rate at which it is done? Essentially, it would involve some electromechanical thing. So we talk about different speeds, say 50 characters or 100 characters – but certainly not more than 200 or 300; that is about the limit. How close can the dot be – we talk about the resolution. Depending on that speed also will keep varying.

Now we can see that in the case of Daisy wheel or a line printer, it is going to be a whole character itself available; it is very much like what you find on the typewriter. So the resolution will be very high; here unless the dots are very close, we are going to have either a rough print or a fine print or a what we may call as the rough print. So it depends on the spacing between the dots. Now depending on the speed or the rate at which printing must take place, the data must be sent. If it is very slow speed, then the data can come directly – it can be controlled by the CPU. Otherwise you need a buffer. Generally in the case of printer, you will only talk about a line buffer; say for one line not many more because it is not all that fast. We will see something more about these devices in the next lecture.