

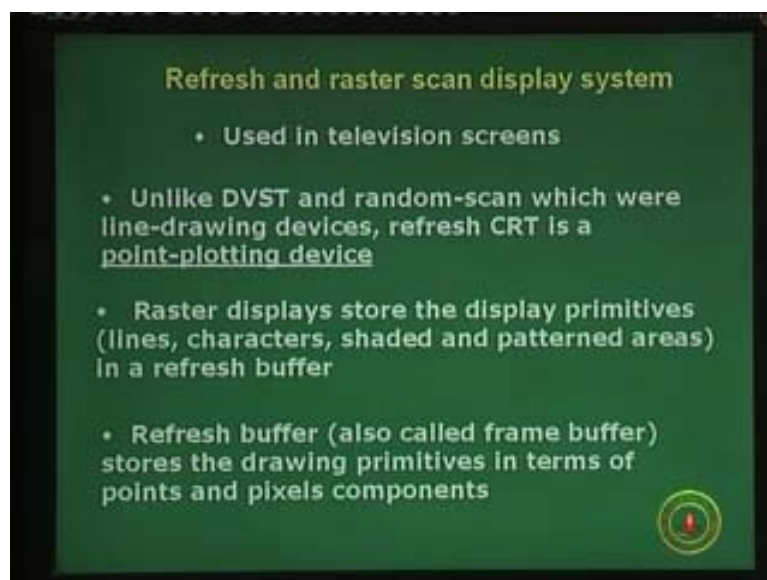
Computer Graphics
Prof. Sukhendu Das
Dept. of Computer Science and Engineering
Indian Institute of Technology, Madras
Lecture - 3
Section II
CRT Display Devices

Hello and welcome back to the lecture on CRT display devices. In the last lecture we covered two of three different types of CRT display devices. First was the DVST or Direct View Storage Tube and second one was Calligraphic or Random Scan display systems. The difference between the two types of display systems which we discussed in the last lecture was that the DVST has a long persistence phosphor coating on the screen and the Random Scan display system had a short persistence phosphor which used to decay in about 10 to 100 microseconds so it needed a refresh.

We talked about a refresh rate of certain 30 frames per second to be done for Random Scan display systems whereas in the case of DVST it was a long persistence phosphor so it could stay for several seconds. The other difference between the two was that in DVST we also called it a vectored or line or a stroke based system in which we were essentially drawing lines on the screen. In fact characters were made up of short segments or lines and if you want to draw any picture you also had to draw lines on the screen. So that was essentially the random scan display system and of course in the DVST also we were drawing lines, the beam had to be switched on and off in both the cases.

Now we come back to today's lecture which is the third category of the display system called refresh or raster scan display system. You can call it a refresh raster or refresh scan or a raster scan; it is different from the Calligraphic or Random Scan display system. We will find out in what way it differs. Let us look at the screen.

(Refer Slide Time: 5: 46)



The first point is that refresh and raster scan display system typically is the most common standard used in modern display systems as well as in modern television screens. How does it differ from the random scan display system? Unlike the DVST and the random scan display systems which were essentially line drawing devices, the refresh CRT or a raster CRT is essentially a point plotting device. So we are essentially putting the points on the screen now rather than drawing lines.

We will go through this lecture on how this is implemented or how this is basically performed in the screen. The next point is the raster displays store the display primitives in a refresh buffer. We talked about the refresh buffer in the random scan display systems which had a display list or program which had to be cycled at certain rate. In that case it essentially contained lines, in this case we will say that it entirely stores display primitives like lines, characters, shaded and pattern areas. We talked about these display output primitives in the introduction part of the lecture where these are the primitives used for any graphic standards or graphic software and basically these are stored in refresh buffer and still all these are implemented by point plotting device.

Therefore, essentially we plot points corresponding to these display primitives. The refresh buffer is also called the frame buffer. It stores the drawing primitives in terms of its points and pixel components. So this is an interesting point which I was just mentioning that the frame buffer or refresh buffer stores the drawing primitives in terms of points and pixels. So, in the Random Scan basically we were storing line commands to draw pictures, characters whatever it is or even curved objects made up of lines.

In the case of refresh buffer or frame buffer they essentially store the point coordinates and the pixel intensity values for each drawing primitives. Typical examples of these drawing primitives, as you can see, in the screen, in the second paragraph or the second line says that: the primitives are lines, characters or shaded and patterned areas. So this is essential and the only difference between the DVST and Random Scan on one side and the refresh or raster on the other side.

Now, to implement this point based method of drawing there are various facets, concepts and principles in the refresh and raster scan display system which is not present in the other two types of display systems the DVST and Random Scan. We will talk of concepts such as the amount of memory needed in the frame buffer. The pixel access time, the bandwidth is all the concepts which will come here and they are much much more important in this lecture or when we talk about refresh raster scan then when we talk about DVST or Calligraphic or Random Scan display systems. So before we talk about these concepts let us look into the architecture of the simple raster graphics system.

I think in the previous lecture we also discussed about simple architecture for a Random Scan system. So this is almost similar to that I will say in this case a typical computer system will have its own CPU, system memory, system bus, I/O devices and video controller which drives the computer monitor. We have seen the internals of a computer monitor in terms of its electronic gun and of course the horizon deflection plates, the control grid, the focusing system and the phosphor coated on the screen. We also discussed about the frame buffer being a part of the system memory.

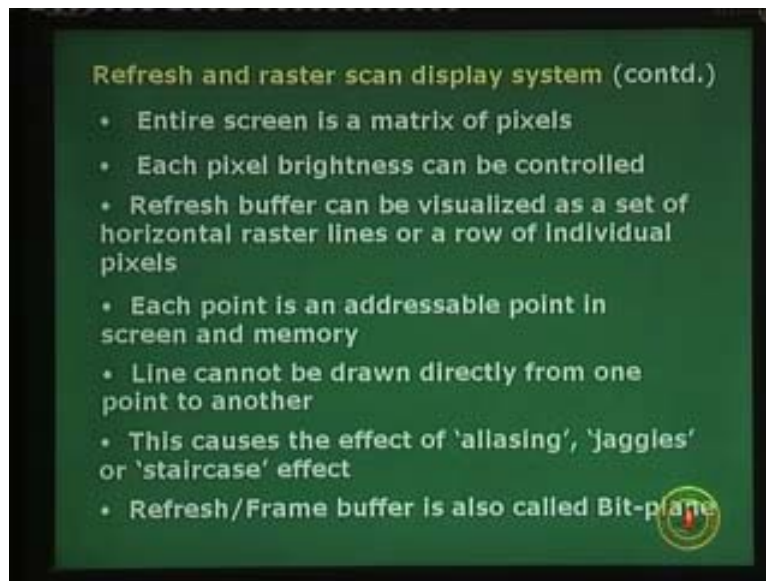
We will come to that. But this is a simple system where we say that the CPU puts the frame buffer instructions somewhere in the system memory and the video controller gets the output primitives the point and the pixel commands from the system memory and takes it out and displays into the monitor. So the video controller has a particular task, we see the block diagrams inside the video controller of what it is doing in due course of time.

As I was talking about slightly elaborated architecture of the raster system with a fixed portion of the system memory reserved for the frame buffer as I was talking about that it can say that the system memory and the frame buffer together can be conceived or visualized as a single memory module of the system or you can have a part of the system memory or RAM as you call it identified as the frame buffer where the CPU or the application program puts the instructions in terms of the display list or the display program we talked about in the case of a random scan that is put inside the frame buffer. And the video controller will access the frame buffer through the system bus and display contents in the display monitor.

Well a few other points. Another important point which is a distinguishing fact between the refresh and raster scan display systems on one side and the line based display system, I will call it as the line based, the Random Scan, Calligraphic or the DVST essentially or line plotting commands, stroke, vector or line is point based in this case. So when you are talking of points we have to now visualize that the screen in the case of a refresh raster scan consists of a matrix of points or what are called as integer pixels or simply pixels on the screen. The screen which you are seeing on the television monitor or the modern display monitors, computer monitors or even LCD panels essentially consists of an array.

We discussed about an array in mathematical algebra. It is an array or matrix of pixels and the entire screen instead of having a uniform coating in the case of a DVST or even Random Scan which draw lines. But now since we have to draw points the points are at specific locations on the screen. So when I want to draw a point at the coordinate x, y inside the screen I must access only the corresponding pixel which is a small dot on the screen which illuminates when an electron gun fires an electron beam into the part on the screen the points starts glowing. So essentially there are a few million points on the screen which we will call as matrix of pixels. That is the difference between the line based and the point based.

(Refer Slide Time: 15:53)



Each pixel's brightness can be controlled that is the property of a monitor which we used to do in the case of random scan or DVST. The brightness can be controlled by the strength of the electron beam which is firing inside into the point and the strength of electron beam controls the brightness of individual pixel. The next point, refresh buffer can be visualized as a set of horizontal raster lines or a row of individual pixels. The concept is like a matrix or an array where we have horizontal rows and vertical columns. So obviously we talk of these as horizontal raster lines.

We do not bother right now about the vertical columns for the time being but assume that in an array or a matrix pixels we have horizontal or raster lines or row of individual pixels and that is very important. We will see why we have to consider that rather than vertical columns. Each point on the screen is an addressable point in screen as well as in the memory, this is interesting. I was just saying that if we talk of any arbitrary point with coordinates x, y on the screen and you want to access that by firing an electron beam onto the point on the screen it is equivalent to trying to access similarly a memory location without an array or a matrix.

In memory you can also define a chunk of memory which stores the values of matrix of pixels and you have to access that particular addressable point in the memory and the contents are then transferred to the point on the screen. Basically what is done is, the contents of the memory location which hold the color information of that point which will in turn control the strength of the electron beam which will be fired on to that particular corresponding point on the screen and will actually dictate how much of intensity you want at that particular point or what color you want at that particular point. So we will say there is a one to one correspondence between a point on the screen and the corresponding point in the memory area.

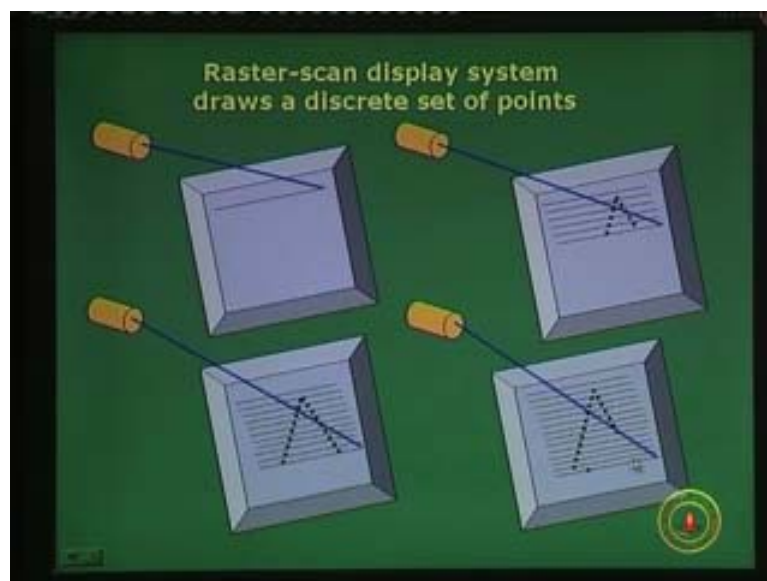
Line cannot be drawn directly from one point to another, this is an interesting fact. We were not worried about point in the case of DVST or in the case of random scan it was our basic unit. We used these lines and short strokes in vectored mode, drawing in the case of Random Scan or DVST to draw lines or characters or whatever complex

pictures you want. But in this case since you are only addressing and drawing points, now it means that a line a short or long one has to be drawn with the help of points. That means you are basically not drawing a simple line by joining one point to another as you draw on the screen but you basically have to enlighten or switch on pixels or points which lie between the starting point and the end point of a line. So, basically you have to switch on points or draw points which lie between one point to another. A line cannot be directly drawn it is not simple like **petting** a scale and drawing a line in a vector mode. Just switch the beam from one point to another, after switching it on the line is drawn on the screen that is not possible. In this case you have to switch on the points which lie between the starting point and the end point in the screen and this is how you can draw a line.

We will see an example of how this is drawn and later on when we discuss about graphical output primitive drawing algorithms, scanline drawing algorithms we will see how this is implemented quite fast in a computer but they have their own problems. We will see again with an example which is unavoidable in the case of refresh and raster scan display systems in the sense that this causes the effect of aliasing, jaggies or what we call as staircase effect. So we have the effect of aliasing, jaggies or staircase effect when we try to draw a line.

In the case of a refresh raster scan display system this concept or this drawback does not exist in the case of DVST or Random Scan display systems, we will see with an example. The last point is a terminology which says that the refresh buffer or the frame buffer as these terms are again used interchangeably. It is also called a bit-plane. In the special case it is called a bit-plane but you can generally call it as a bit-plane or refresh buffer or frame buffer or refresh frame buffer. So, we know that raster scan display system draws a discrete set of points and this is done with the help of an example like this.

(Refer Slide Time: 20:15)



This picture shows that there is a yellow cylinder on the top left which signifies as an electron gun. Then there is a blue line coming out of the gun which indicates the electron beam and when the electronic beam is basically moving horizontally across the screen. Assume this rectangle which you see with a border to be your television screen or the monitor in which you are visualizing this particular lecture or watching this lecture basically and the electron beam moves from left right on this screen.

Remember, it does not draw a line because the raster scan display system essentially was drawing a line. But that would have been the case for the Random Scan but in this case it has to put a point. So only at points where the beam is switched on you will find a point glowing on the screen otherwise not. So what we are assuming now is that when you see these horizontal black lines on the screen you assume that the beam is switched off. It will be switched on at only those points which you want to mark on the screen. So, let us say the beam with the help of horizontal and vertical deflection plates or electromagnetic coil will be deflected from left to right on the screen, it will go down again and keep scanning what we call as horizontal row of pixels or horizontal scanline. So, the first line is drawn but it is not displayed because the beam is switched off.

The beam has to move from left to right but you do not see anything on the screen. That line what you see is basically the trajectory over which the beam has moved but since the beam is off none of the pixels or points are on. So let us see what it does. After a few such scan lines you will see an effect like this. What you will basically see is those points on the screen where the beam would have switched on. So if you take a particular horizontal line, let us say the top one here, if the beam would have switched on only at one point and switched off again making that pixel to glow at some point beneath the beam then it would have switched on twice, one for the left pixel and one of the right pixel on the same horizontal scanline. So wherever you want these pixels to glow to give the effect of a line you have to make the beam turn on and turn off very quickly because the beam is traveling very fast from left to right on horizontal scan lines or horizontal rows of pixels and it keeps doing this from line one line, second line and third line and so on. So, at some intermediate stage you will probably find out that half of the scan lines have been covered and at few positions the beam has been turned on and it is at only those points where the pixels are visible.

More on that, the electron beam has now been traversed more than three quarters of the frame or the screen and it has switched on and off twice for each scan line giving the effect of two inclined lines. It appears as an inverted beam for the time being. But since the picture is not drawn I will not tell what we are going to draw actually overall on the screen but now it gives the effect of two lines meeting to form a corner or something like that. But except the first horizontal or second scanline the beam would have switched on only once. At each other scan line at two integer pixel positions the beam would have turned on and it is off for the rest of the period. So you basically do not see the horizontal scanlines, these are just the trace on the screen to give you the visualization.

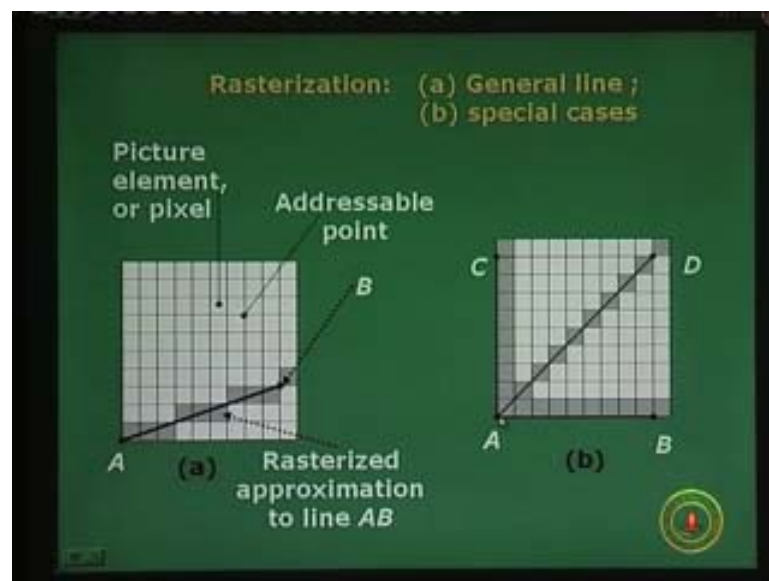
What you essentially see is these pixels, very dense in general but I have made it sparse to give you visualization. So you basically see two continuous lines which meet at the top corner and that is what you see. At the end after the entire picture is complete it will look like something like that. Probably I am trying to draw, in this

case let us say the effect of a triangle. That means I will probably draw one more line and these two lines will probably converge at some other point when the bottom of the screen. So whenever you want the beam to switch on to draw or make a pixel glow or draw a point as I was talking about in the case of a raster scan the beam is switched on.

You see the simple picture which consists of two or three lines the beam is essentially switched off. For the majority part of the scan for the entire frame it is only switched on at one or two occasions or one or two points within a line. So that is an important part of point based drawing for a raster scan display system. It basically, as the title says it draws a discrete set of points on the entire screen or even if you take a horizontal line it may not draw a line it may just draw one, two or few points for each horizontal row of pixel or what is technically termed as a horizontal scanline.

We discussed about problems in line drawing in terms of what is called a stair case effect or jaggies or aliasing problems are several other terms used interchangeably and basically means the same thing. We will look into the screen and see two special cases of the line; general and a special case; this is what I mean by a general line. Let us talk about a horizontal area of pixels or matrix of pixels basically. There are two ways by which pixels are addressed either in memory or the screen.

(Refer Slide Time: 25: 40)



We assume that the pixel addressable points are integer values x , y horizontal and vertical, ordinate and **absis** are integer values starting from 0 or 1 up to n depending upon the resolution of the screen. And the addressable point is the intersection of any two such grids and the picture elements or pixels which basically grow on the screen can be visualized to light either on the intersection point as shown by the addressable point on the screen or it could be inside of that small square element of the grid.

The picture element as it is called the pixel, in fact the word pixel is coined by the combination of two words called picture element. Now in this case we want to draw a line from point a to point b. So point a is at the bottom of the screen as it is given here

and the point b will be somewhere here. So unlike in the case of random scan where the beam was moving from point a to point b what we do in this case is we have to switch on a set of pixels or points which lie close to the actual line joining point a and point b. There are many sophisticated algorithms.

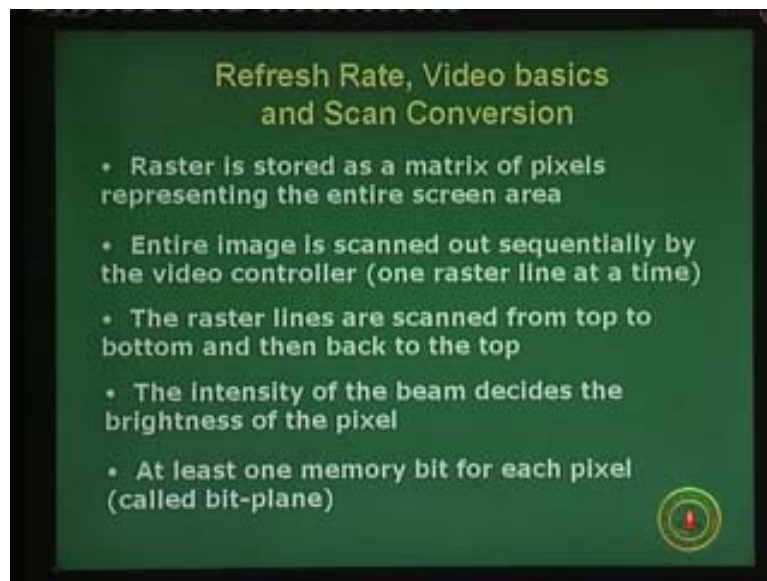
We will talk about one such line drawing algorithm to find out which pixels are the closest and give the visibility of a continuous line say line ab. So which are the pixels to be switched on? This is a particular example, the rasterized approximation to line ab means that we will not see this line, the dark line joining points a and b here as discussed about. We basically see these grey shaded pixels, square blocks they will not appear square because they are so small and finite that they will appear to be a continuous line. But if you observe very closely with the help of a lens or sometimes with a naked eye in case of a poor or a low resolution display system, TV screens or even monitors you will probably see that the pixels will form an arrangement like these and these are the pixels which the algorithm has found out or the programmers have found out that lie very close to the line a and b. You see an effect of the staircasing here. It shows that it is an effect of a stair casing, generally you will have this. That is why we will call this picture A as a general line in which you have this of effect of staircasing or jaggies or aliasing problem of a line. This is true with the digitalized world or discretized world even in other areas of computer science and digital geometry or digitization or whatever you name it this is bound to be there. You cannot avoid it in fact and this is the case where how sure the general line means. Well, the special case of a line, this effect of stair casing, jaggies or aliasing will not occur in special cases of a line where the line is purely horizontal or purely vertical or inclined at exactly 45 degree with respect to the horizontal line. These are three cases of lines.

If you see, if I am taking four points point A here on the figure B I am taking about figure B point A is on the left hand side bottom, point B is a right bottom, D is on the top right and C is on the top left. So if we see the pixels which are drawn for the line A to C and A to B and A to B being a perfect horizontal line, A to C being a perfect vertical line and the line A to B being a perfect diagonal line inclined at 45 degree or $\pi/4$ radians. In these cases we can almost assume that the pixels exactly lie on the line or adjacent to the line, in this case it does not matter. But it forms a perfect arrangement as desired for the line and you do not have the effect of stair casing or aliasing as you see on the left hand side of the figure for a general line. But if you take all possible lines which can be drawn on the screen you will see that the special cases do occur but they occur rarely. So, general line occurs more frequently and so you will have the effect of jaggies or aliasing or stair casing. I hope this concept is clear. We will discuss more about this when we talk about algorithms for scanline or line drawing this will be more evident. We will talk about that later on as we go.

At the beginning of the lecture I did say there are certain concepts to do with the refresh scan displays, we may not be concerned with some concepts like DVST and we were discussing about DVST or random scan display systems. There are some concepts which we are not bothered, which we have to worry about in the case of raster refresh displays. There are two parts one is to do with points and line drawing which we already discussed about but there are more to it in terms of the memory, the amount memory the scan rate, the scan memory, video basics etc. So, we will move into those concepts for refresh raster for the rest of the lecture today or even may be in

the rest part of the lecture. Well, we already talked about that the raster is stored as a matrix of pixels representing the entire screen area we already know that I **retreat** that point. And then we also see with an example the entire image is scanned out sequentially. We know that how it is done by the electron beam and usually it is switched off and is only switched on when you have to make a pixel glow so it is kind of essentially done by the video controller which has control of the electron beam. So it scans one raster line at a time and in each raster line it knows which pixel to switch on or off essentially the beam is off. Wherever it wants to switch on it will switch on and go off. The raster lines are scanned from top to bottom and then back again it has to go to the top, why? This is obvious because we have to refresh this screen.

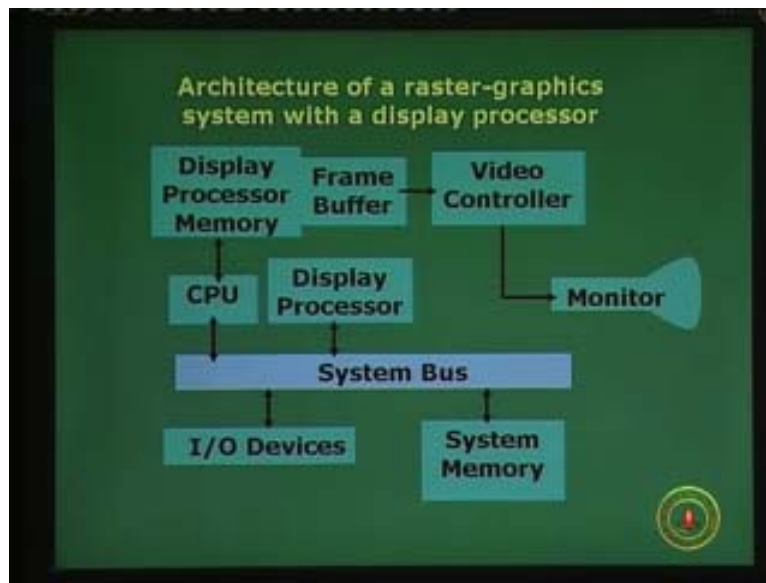
(Refer Slide Time: 30:22)



Unlike the DVST, both the Random Scan and the raster scan display systems have one thing in common, the persistence of phosphor is too low. It will vanish in no time. It takes about 10 to maximum 100 microseconds, it has to be scanned. The pixel has to be visited again otherwise it will switch on and off too early for you to have a consistent view. Flicker-free display is what we want and to do that we know that we have to refresh. So the refresh word is still there but the raster has come in. So essentially we are drawing points on each raster line and they are scanned from top to bottom and then we have to again go back to the top to start the scan once again. The intensity of the electron beam decides the brightness of the pixel.

In the case of a Random Scan the intensity was dictating the brightness of a line, in this case it controls the brightness of a pixel. And of course the brightness of the line will be dictated by the brightness of all the pixels which lie within the line. Well we at least have one memory bit inside the frame buffer which could be a part of the system memory or it could be a part of the video controller, we will see that. But essentially when we talk of the adapter cards typically they have the video RAM in-built in the card along with the video controller which drives the monitor. But in some parts the frame buffer could be a part of the system RAM or it could be in the video controller. So, at least one memory bit for each pixel. We discussed about this point earlier, we call it as a bit-plane when we have one memory bit plane for each pixel.

(Refer Slide Time: 30: 38)



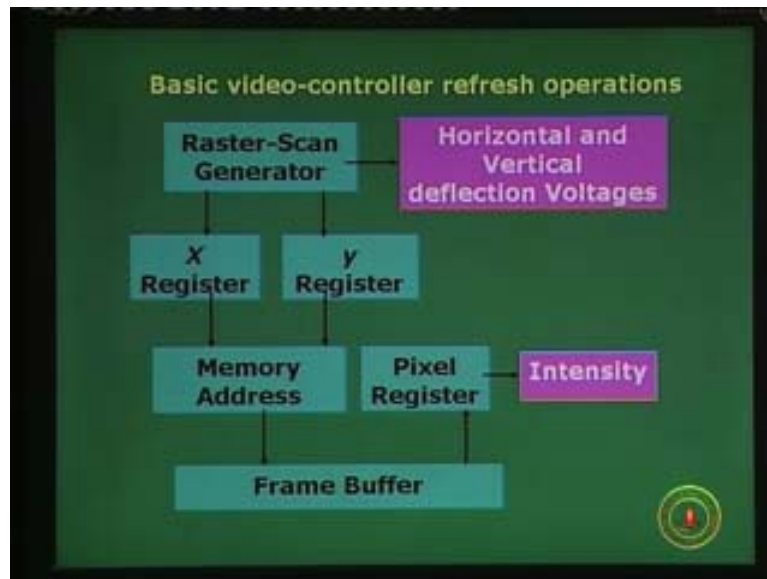
The architecture of a raster graphics system with a display processor is a slightly more elaborate diagram. We discussed about this figure earlier on several occasions in today's lecture as well as the earlier one. The difference in this particular diagram or figure compared to those which were there in the previous diagram was the display processor. So that is the additional part the rest of the CPU which accesses the display memory directly instead of the memory sitting through the system bus that is the difference here. The configuration could differ but the frame buffer is a part of display memory in this case and the video controller takes the input from the frame buffer, drives the monitor all that is fine. The extra part is the display processor.

We need an extra display processor to do some jobs of computer graphics such that it helps us to avoid overloading the CPU or Central Processing Unit. The central processing unit has to do lot of tasks in terms of managing the system, running the operating system, handling the inputs and outputs and all that. We have a system memory down below as we can see from the system bus but we have a display processor memory and a separate display processor in memory which the CPU can directly access.

The frame buffer is a part of display processor memory and the display processor basically does all the computer graphics picture drawing, manipulation command and it freezes the CPU from getting overloaded of the burden of doing these jobs connected to scan conversion, refresh rate and all that because this CPU gets involved in doing all these graphical image drawing, manipulation commands which requires a very fast bandwidth, a very fast operation then the CPU will be busy doing that and the system will be devoid of resource of the CPU to run other programs and the system will be very sluggish. The system will not be able to respond to other requests of other users for non-graphical work applications which do not require much of graphics that this CPU will not be able to respond. So you have a separate display and it is almost a must in most parts of the systems to have a separate display processor which runs the graphics application programs and packages and accesses the display processor memory or the frame buffer to make the display list, the display program

which in turn will be driving the video controller. So, that is the job of a display processor.

(Refer Slide Time: 33: 01)



The job of a video controller is very important. If you remember, from the previous diagram, if necessary, we will revisit it again. It basically gets the input from the frame buffer and drives the monitor. What do you need to drive to the monitor? You need to give input to the monitor two things, where the beam should be and what should be the intensity of the beam. I repeat again the beam is been scanned from left to right each scanline from top to bottom back again to the top, that is fine. So at each point you must know where the beam should go, automatically the video controller will do that. So at any point of contact with the memory screen the video controller must know whether the electron beam must be switched on or it should be remained off as it is or if it is switched on with how much intensity, what should be the strength of the electron beam such that the desired intensity level which is in the frame buffer for the corresponding pixel.

Remember, there was a one to one correspondence between the two arrays or the matrix of pixels on the screen and also in the frame buffer. Each point addressable memory there was a corresponding point on the screen which should be off or on, on with a certain brightness or color. So, when you are addressing that point in system memory or frame buffer the corresponding beam is also trying to fire that pixel on the screen. So, you should know what should be the intensity or the color of that point and hence the strength of the intensity. The strength of the beam must be controlled to give it the desired intensity at that particular point.

So, if you look back into the figure essentially the video controller must give two outputs. First one if you see within the pink shaded diagram here with what characters, horizontal and vertical deflection voltages. That dictates where the electron beam is currently and what pixel on the screen it is trying to access. So at that particular point the deflection voltages of the horizontal and vertical deflection plates or it could be the solenoid or the magnetic coil which controls where the beam

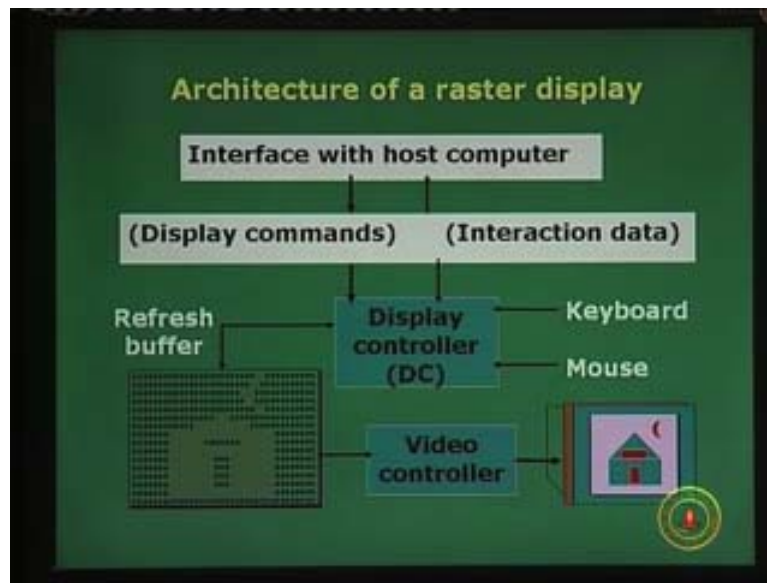
position currently is that is dictated by what memory you are accessing at the frame buffer, that is number one. And what should be the strength of the electron beam, whether it should be switched off or whether it should be switched on the intensity of the beam. So these are the two outputs of the video controller which goes inside the monitor or the monitor basically needs these two inputs from the video controller to guide the beam and switch on and off at the particular intensity. What does it do? Well, you have a block called the raster scan generator which will automatically keep doing this refreshing. We have not discussed about the video basics and refreshing, we will do that.

But basically the raster scan generator keeps automatically generating these horizontal and vertical deflection voltages corresponding to the addressable point in the memory from the frame buffer and it will load the corresponding values of xy coordinates of that integer position of the screen to two registers x and y which will in turn address memory through a memory address register. Well, this is something to do with concepts of computer organization or operating system. Those who have that background will easily understand that the memory address is built from the contents of the x and y register and it will help you to access a particular location inside the frame buffer corresponding to the point you are addressing on the screen.

I repeat it again; the point on the screen addressed and the point in the memory at the frame buffer must be addressed simultaneously. It is done with the help of this block diagram but the raster scan generator generates the analog voltages for deflection of the beam. It also generates the corresponding digital values of the x and y coordinates which in turn will generate the memory address and hence accesses the point on the frame buffer. Once you address a particular point in memory the frame buffer will return the contents. So it will return the contents to a pixel register which will hold the value of the intensity of that particular point which you want on that screen for a particular point.

Two corresponding points are addressed simultaneously; one in the frame buffer, and one in the screen. So you address them by the video controller, deflection voltages are generated, beam is already there, what should be the intensity? The contents of the frame buffer are taken out and the pixel register drives again in a digital to analog converter to generate intensity, basically it is a voltage to convert the strength of the electron beam in which it will turn the intensity or color of the pixel or point on the screen. I hope this concept is more or less clear in fact it should be clear to those students who have some background at least of concepts of computer organization and even operating system little bit. The concepts of computer organization architecture are probably more important to understand this block diagram where you access a particular point in the frame buffer, get the intensity through the pixel register put that intensity voltage on the electron beam to dictate the strength of the beam or the intensity of the pixel screen and the horizontal deflection voltages dictate where the beam is. So we will get out of this and that is another way of visualizing the architecture of raster display system.

(Refer Slide Time: 38:36)

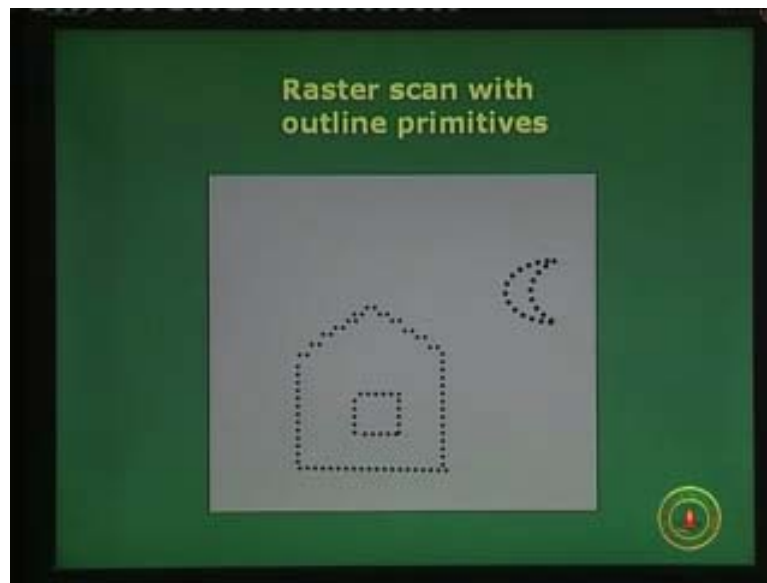


Now we do not worry much about the internals of a video controller or a system. We typically look at the two bottom parts of the screen, I say that it is a one to one correspondence between the refresh buffer or frame buffer and the screen. So that is the left hand part. We discussed about the refresh buffer being a horizontal array of memory locations. The screen is also a horizontal array of pixels. So, the host computer will particularly have the display commands and input output to the keyboard and the mouse, the display controller will now load the refresh buffer corresponding to what the display commands have corresponding to the computer graphics application package which is trying to draw a picture on the screen. The refresh buffer will have the contents.

Typically what I have done is tried to put the refresh buffer assuming it to be a single bit display system so I have loaded 0 or 1. If you see on the bottom left the refresh buffer will have 0 drawn with black and one is drawn with orangish color. So I am trying to draw a structure of something like a house with a moon on the top, the refresh buffer will have the contents as shown on the left. I repeat again, black 0s and the 1s are in color, so just distinguish between two values and that refresh buffer drives the video controller, the video controller will take the inputs of all the contents one by one and put it on the screen.

The screen will probably have a very nice picture as given on the right hand side, a house with a window and a door and a nice moon on the top of the sky. So that is the typical example of a correspondence of contents of the screen and the refresh buffer. This diagram probably shows the 1:1 correspondence if not explicitly very implicitly within the picture; well this is a very good example. When we talk about the raster scan with outline primitives we are talking about points being drawn on the screen. So a line essentially consists of points and these points help to draw a line and this is again a very simplified picture of a house with a window, a simple example of just the outline primitives.

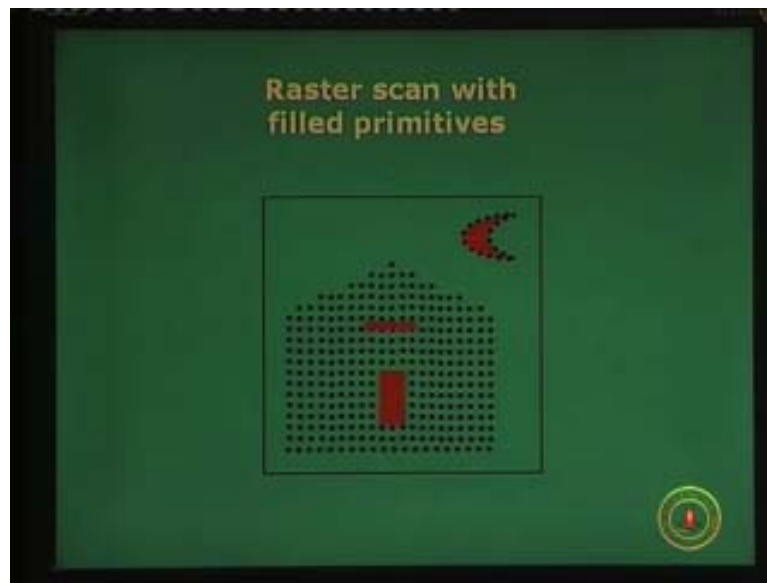
(Refer Slide Time: 40:25)



We are drawing lines and curves with the help of points and this will also be the contents of the frame buffer as what you will see on the screen. The background could be black and the outline primitives could be white or vice versa. It depends upon the way you want to represent this picture. So, essentially again this is a good example to show that you are drawing points; you will never draw a line. And with the help of points itself you have to draw lines, curves or anything else within the picture.

Raster scan will fill primitives, the same diagram now I want to fill the area. I have filled some parts with color to give it a continuous shade but essentially filling is done by drawing lines with the help of points again. Remember, everywhere it is a point either you talk of a shaded area will have points with a particular color or shade and the inside of this house basically is filled with horizontal lines. You can fill with vertical lines also not a problem. You can draw lines from anywhere on the screen to anywhere on the screen but essentially you are only putting points. So this is a filled primitive where the region is filled with a certain grey shade or color.

(Refer Slide Time: 41:13)



You can have an outline as drawn here (Refer Slide Time: 40:25) or you can have filled primitives as shown here (Refer Slide Time: 41:13). We continue with the discussion of refresh rates, video basics and scan conversion. We are discussing refresh raster display systems and now we know how to draw pictures in refresh scan but we have not moved towards the demands of system memory and bandwidth and speed with respect to scan conversion. Let us take a typical example of memory usage.

We will say if one uses a 512 into 512 element raster display. What do I mean? Well, we talked of matrix or an array of pixels, so there are 512 rows and 512 columns, perfect square matrix, aspect ratio 1. And, if we have so many pixels on the screen we should also have corresponding points in the frame buffer to draw a picture and for this if we even have a single bit-plane we need 2 to the power 18 bits in a single bit plane and that we are talking of a memory size of the frame buffer of 32 kilobytes. That is 2 to the power of 18 bits or 32 kilobytes, 2 to the power of 18 comes because 512 is 2 to the power 9 two of them so we get 2 to the power 18 single bit plane that means it is 0 or 1 is a value and so we have a black and white picture on the screen. So for such a display system I hope you followed the calculation that we require 32 kilobytes of memory.

(Refer Slide Time: 44:35)


Refresh Rate, Video basics and Scan Conversion (contd.)
A typical example:

If one uses a 512x512 element raster display, then 2^{18} bits are necessary in a single bit plane. Memory size required: **32 KB**

A DAC (digital-to-analog converter) is used to convert the bit value (0, 1) to analog signals for refreshing the screen

Memory size required for N-bit plane gray level frame buffers:

N	Size in KB
3	96
8	256
24	768



A Digital to Analog Converter or popularly called a DAC typically is used to convert the bit values 0 or 0 to analog signals for refreshing the screen. The analog signal will drive the intensity of the screen; it will drive the electron beam. The strength of the electron beam in turn will dictate the intensity in the screen and of course it will be refreshed, that is fine. Memory size required for an n-bit plane, grey level frame buffers we talked about in the previous example on the slide where capital N was equal to 1 so it is a single bit plane, but now in this 512 into 512 element raster display we keep on increasing the value of N so what is the memory required, this table shows typical examples of three necessary memory requirements for different values of N.

The calculations are very simple because when N was equal to 1. we talked about 32 kilobytes of memory, when N is equal to 3 we talked about 36 kilobytes which is nothing but 32 into 3 and you can keep doing that, 32 into 8 when n is equal to 8 gives 256 and for 24 we talk about 768 kilobytes or about 0.75 megabytes of memory for N equal to 24 and we are talking of these with the 512 into 512 element raster display. If we reduce or increase the raster display size or resolution as it is talked about, the memory requirement of the frame buffer will also go up and down. Continuing with this we discussed about a minimum refresh rate of about 30 hertz for a Random Scan display system. We need not worry about the DVST. If we remember much earlier coming back to refresh rate for raster display system it is recommended to run it at about 60 hertz to have no flickering or to avoid flickering.

(Refer Slide Time: 48:46)


Refresh Rate, Video basics and Scan Conversion (contd.)

Refresh rate to avoid flickering - 60 Hz

If one uses a 1024x1024 high resolution CRT:

N	Display Color	Memory Size
1	Black & White	128 KB
8	256 colors	1 MB
24	16 million colors	3 MB
32	16 million colors	4 MB

Even 32 bits per pixel with 1280x1024 pixels raster are available.



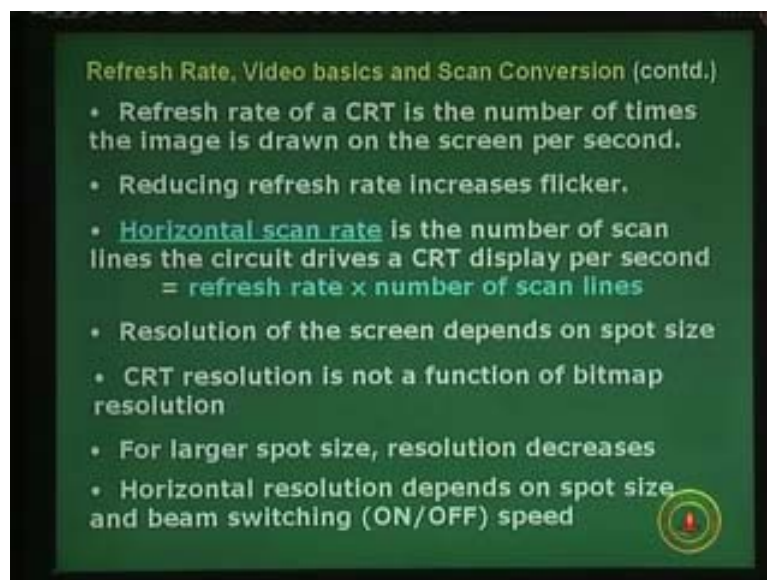
If you want a complete flicker free display well you should have it more than 30 hertz scientifically but if you have it at 60 hertz absolutely there is no chance of any flicker which is visible to the naked eye. Coming back to memory requirements I jump up from the resolution which we discussed in the previous slide of 512 into 512 resolution, CRT to a 1024/1024 high resolution CRT and these are for the different values for n on the left hand side. This table shows what the memory requirements are and displays itself. Well, when capital N is equal to 1 I did say we are talking of a monochrome black and white pure display system and we require in this case 128 kilobyte, why? A 1024 is 2 to the power 10 so we have 2 to the power 20 bits and that gives you about 128 kilobytes of memory when capital N is equal to 1.

Remember, it is very easy to calculate that. In fact 512 into 512 we were talking of 32 kilobytes and that will be four times. Now, 32 into 4 will give you 128 kilobytes when capital N is equal to 1. remember that previous slide N is equal to 1 and 512 into 512 was requiring 32 kilobytes and now we require 128 and capital N becomes 8 the display, we can talk of display colors or display grey levels, we can have grey shades or colors 256 of them instead of having just 0 or 1 and we require a memory size of 1 mega byte, of about 1 mega byte which is eight times 128 so that is 1megabyte. And as we keep going on with increasing values of n 24 and 32 we can have 16 million colors possible to display.

We require about 3 to 4 mega bytes of memory in the frame buffer to display a system. Now if you look at the last two rows of the table the memory requirement goes up to 3 to 4 megabytes for a 1024 high resolution CRT because capital N has gone up from 24 to 32 and I am talking of last two rows because of the increase in the capital N the memory requirement is more. But we are not talking about increasing the colors, 16 million colors are the same in fact the extra 8 bits which are all available here are used for helping some other operations of hidden surface removal and shading to do with z buffer algorithm.

We will talk about that when we talk about 3D shading models later on. So we have utmost 16 million colors possible. Of course if you still have at the maximum may be a little bit more 32 bit pixels now-a-days is possible. We talk about very high bit frame buffer with 1280 into 1024 pixels or rasters that are available. We even talk of more than that where one of the rows or columns can go to about 1600 pixels which is a very very large high resolution monitor and 32 bits pixel is typically the highest standard. Well, almost the last slide for today before we move on to a few questionnaires is the refresh rate of a CRT. We define this as the number of times the image is drawn on the screen per second. That is a very simple definition. We discussed about the refresh rate 60 hertz recommended 30 hertz is the minimum. So the number of times the image is drawn in the screen per second. And of course the more the refresh rates the better. But if you reduce the refresh rate you have the advantage of reducing the frame buffer size advantages but it increases the flicker.

(Refer Slide Time: 52:39)



The viewer may not like it if it is not flicker-free. If the screen flickers, the viewer may like the picture on the screen or the display itself and then will move over to a slightly better display with a higher refresh rate. Well, we talk or introduce a term called horizontal scan rate. Horizontal scan rate is the number of scanlines the circuit drives a CRT display per second. And we talk of this as refresh rate multiplied by number of scan lines. Refresh rate multiplied by number of scan lines gives the horizontal scan rate and the resolution of the screen depends upon the spot size, this is very very important.

People always confuse that the resolution of the screen depends on graphics adapter card or how much memory you have on the screen, it is nothing to do with it. The display device comes, that is, the monitor comes with an inbuilt resolution and it is not dependent on the computer graphics system which you are going to use. You can connect almost any display device to almost any computer graphics system. They must handshake in the sense that the computer graphics system must know the resolution in the screen, whether it is too high or whether it is too low, of course if it is too high some graphic systems which are low may not be able to drive the display

system and that is possible. But when we are talking of the resolution of the screen it is dependent on the display system and it is basically dependent on the spot size.

What do you mean by the spot size? Physically this very small dimension of the pixel screen which grows on the screen, that spot on the screen it is small if you are drawing points on the screen. So physically it has a finite small size and that spot size dictates the resolution because you cannot pack more than a certain amount of spots in a certain dimension of the physical screen. Let us see, it could be 1 foot by 1 foot and each of them could be as small as 1/10 of an inch or something. So if you are talking about 12 inches by 12 inches you can pack say 120 of those spots in each of the horizontal row or vertical column. So the spot size is the resolution not the graphic system which you are using, it is very very important.

CRT resolution is not a function of the bitmap resolution, I repeat it is similar that the spot size dictates the resolution and the resolution is not a function of the bitmap resolution on the graphic system but it is basically a function of spot size. For larger spot size you can have it but the resolution will be very poor. You can have a very poor resolution with a larger spot size. It is better to have a small spot size but there are manufacturing limits of manufacturing technology which cannot make the spot size than a particular limit.

The last point in this screen and in the lecture today is the horizontal resolution of the screen. The screen's spot size dictates the resolution and the resolution again also depends on the spot size and the speed at which the beam can be switched on and off. Why this is necessary? The beam is moving from left to right at a very fast speed and we are drawing point. To draw that point we make the beam on at a particular pixel and then go off, on at a particular point and go off. So we have to keep doing this for as many pixels on the line which you need to switch it off. It could be 1, 2 or may be a few more, may be a few hundred points have to be turned on and off or may be alternate ones. You make it on and go off, make it on and go off that dictates. So we stop here with this lecture.