

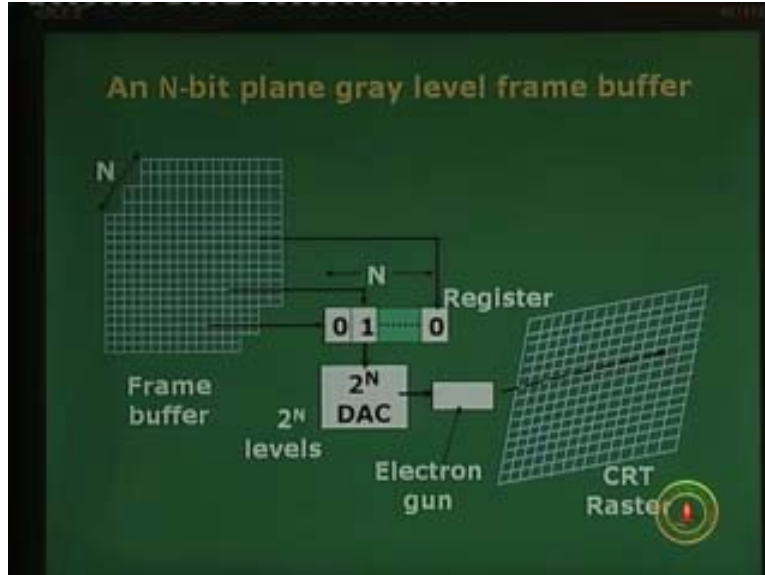
Computer Graphics
Prof. Sukhendu Das
Dept. of Computer Science and Engineering
Indian Institute of Technology, Madras
Lecture - 5
CRT Display Devices

Hello everybody, welcome back to the lecture on Computer Graphics. In the last lecture we had been discussing about CRT displays and CRT monitors and specifically the refresh raster displays. Today we will continue on that and try to wind up the discussion on CRT displays and the three main concepts with respect to the refresh raster displays which is a point plotting device unlike the line drawing or stroked displays. Vector displays of random scan or DVST is, we basically draw points and we worry about video rate refresh, and the scan rate number of lines.

Time regarding vertical retrace or a horizontal retrace, we talk of interlaced fields, we also talked about the bandwidth about accessing a pixel, the time for accessing a pixel to switch on and switch off the beam by controlling the beam, we talked about architectures and finally we also discussed about the direct relationship between the frame buffer, memory, array of pixels and the spots on the screen.

So if we look back in the last slide which we discussed last class; we were talking about N-bit plane grey level frame buffer so we visualize that a bit plane is a single bit so there are N such bits tact and for corresponding each pixel there are N-bits or typically it could be one byte if N is equal to 8. The corresponding contents of the bits from those N-bit planes are stacked and loaded in parallel way on to an N-bit register which fires 8 to the power N DAC Digital to Analog Converter which generates the corresponding voltage proportional to the digital value at the register and the electronic gun gets that input and controls the strength of the electron beam which will go on and strike the corresponding dot on the screen.

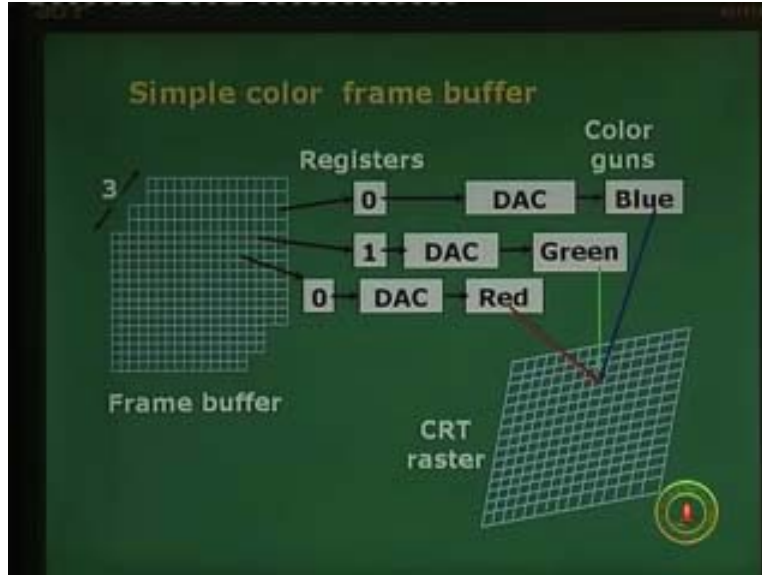
(Refer Slide Time: 2:55)



Of course, we do not have to worry about the deflection voltages generated by the video controller automatically as the beam is being scanned left to right and keeps going down with horizontal retrace and then of course even or odd fields, vertical retrace, diagonal retrace and all that is done automatically and at the same time at a given instant of time when the deflection voltages the horizontal and vertical deflection voltages allow the beam to strike a particular point corresponding to that the intensity of that pixel is being taken out from the frame buffer and loaded on to a register which in turns goes to the DAC and fires the electron beam.

Now we have been talking about N-bit N equal to 3, 8 various configuration bit plane when N is equal to 1 where we have black and white. What about the color, so when you talk about color, we go to the next slide from here.

(Refer Slide Time: 4:00)



A simple color frame buffer can be done with N equal to 3. We discussed about that in the last class that to implement a color we cannot do this with a single gun. We need three separate guns for the three fundamental color. We are talking of an RGB representation of the color image. There of course the other representation which we will talk later in the lecture is about color models and color representation but here we just have N equal to 3 that means we are just assigning a single bit for an individual color. The red color has a single bit plane and the R, the G and the B also for the green and blue. Henceforth we will talk of RGB as red, green, blue respectively. They are the single bit plane for each individual color.

So if you look into the monitor now, the screen shows that there are three such bit planes and we assume here that the corresponding bit values 0 or 1 only in this case is been taken out from the frame buffer, loaded on to a 1-bit register which in fact fires the DAC. The Digital to Analog Converter will convert that bit value into a corresponding voltage 0 or some non-zero values and the corresponding color guns. Now we have the three color guns. We have been talking about one color gun so far, now for the first time we are going to have three color guns separately for the three colors or three channels of visibility.

You can visualize red, green and blue and what is done basically is, all the beams correspondingly from all the three guns are simultaneously fired on to one pixel point and this pixel should also have three dots respectively for R G and B and they will receive the corresponding intensities of the guns and illuminate pixels which will appear as color. So you have the raster, you will have the same resolution as the frame buffer but now each pixel has three dots and that is the significant difference. We will look into that as we go long. But before that we just use N equal to 3 then N -bit plane available frame buffer in case of just one bit for each color frame buffer where we get 8 colors.

(Refer Slide Time: 5:50)

N-bit plane gray level Frame buffer (Contd.)

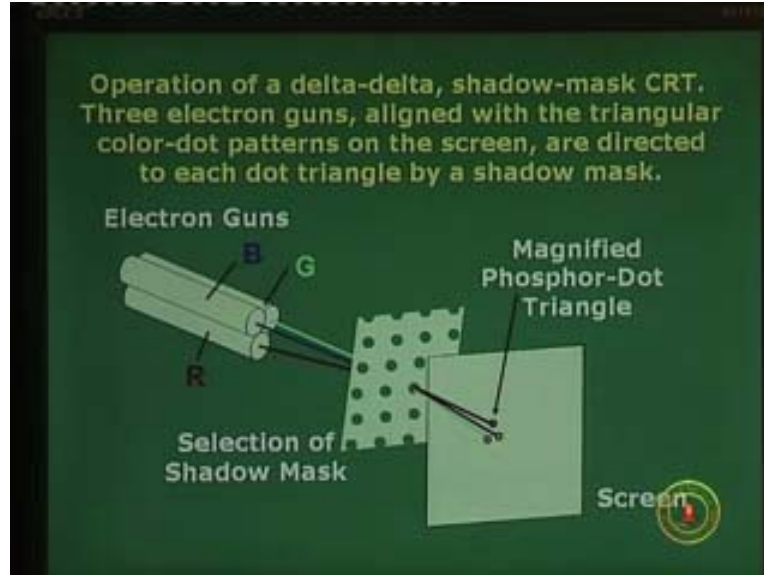
- In case of one-bit for each color frame buffer, we get 8 colors as:

COLOR	RED	GREEN	BLUE
BLACK	0	0	0
BLUE	0	0	1
GREEN	0	1	0
CYAN	0	1	1
RED	1	0	0
MAGENTA	1	0	1
YELLOW	1	1	0
WHITE	1	1	1

If you are used to truth tables for N equal to 3 there are 8 possible values which one can hold in the frame buffer at each pixel position. So we correspondingly have 8 different colors and this table shows the corresponding values of red, green and blue. The first row talks about red, green and blue all three 0s and we have the black color which is of course very obvious because none of the electronics guns are firing with intensity with the beam is switched off and the pixels also down grow so we have black.

At the bottom of the row we have all of them burn simultaneously and hits the corresponding three pixels we have the white color. And in between you have other variations of course the pure blue and the green and the red corresponding to 0 0 1, 0 1 0 and 1 0 0 are the truth table for RGB respectively. And the other mixtures of 0 1 1 will create a cyan and 1 0 1 will create a magenta, label the corresponding color with the color value itself that it gives you an idea. It is a very nice illustration and red, green combination together 1 1 gives you the yellow color. So it typically can have eight colors using just N equal to 3 bit planes and this sort of getting an eight color display was additionally called as EG or Extended Graphics adapter card which is used to be provided with EG display. Well, we are talking about three dots on the screen which is to be fired by three different guns and this shows how it is implemented.

(Refer Slide Time: 7:21)

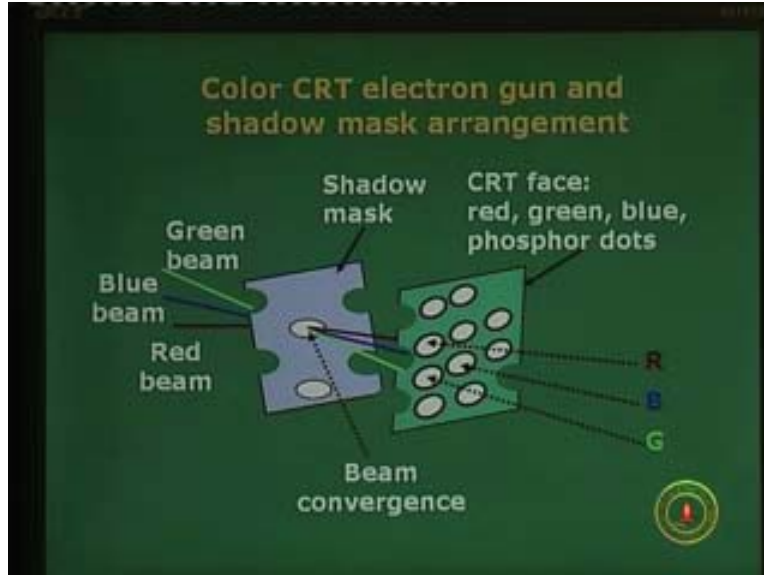


So we talking of an operation of what is called a delta, delta shadow mask CRT and three electronic guns aligned with the triangular color dot patterns on the screen are directed to each dot triangle by a shadow mask. So the beam from a tree, electron guns which are also arranged in some triangular arrangement as like the color dot patterns on the screen and the beams come out, the respective guns as we can see in the left hand side of the screen R, G and B will emit correspondingly, beams which are sensitive to R, G and B respectively and they actually pass through a shadow mask which allows the beam to pass through only that dot and hit the screen.

The shadow mask arrangement does not allow beams to go out at any arbitrary angle and hit the screen. It is specifically aligned with the corresponding arrangement of what I will call as the magnified phosphor dot triangle which is magnified for you to see it otherwise it is almost not useable to the naked eye. You might use a lens and come very close to a CRT monitor to see individual dots that may be possible but not feasible to naked eye so it has been zoomed and shown separately. That is a typical arrangement to a triangular arrangement of the three phosphor dot triangles and they respectively are sensitive to the color of electron beam which is now fired by respective guns. That is very very important, RGB, and of course the screen is full of such dots. So selection of shadow mask is very important and the beams actually converge at the center of the shadow mask and then cross over and hide the screen.

This is probably a better arrangement to show what actually was meant in the previous diagram in terms of we have enlarged the shadow mask and also the CRT face. We are talking of a colored CRT electron gun and a shadow mask arrangement for a colored CRT screen. So the beam comes from the left hand side, those guns are assumed to be there but it is not shown from the left hand side, we have the green beam, the blue beam and the red beam coming out from the respective electron guns. They intersect the beams and converge at the perforation.

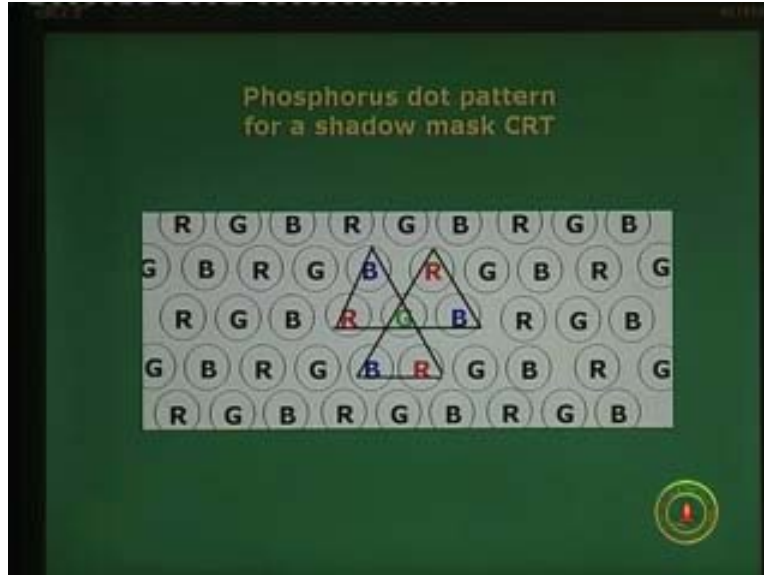
(Refer Slide Time: 9:04)



This is basically a perforated sheet which is like a filter as what you can call but it is metallic. The shadow mask is typically metallic, a perforated sheet type of thing is what you can imagine which has perforations or holes through which the beams are allowed to pass not arbitrarily anywhere and the beams converge there, of course nothing happens it just emanates out in a triangular arrangement once again and they will go and heat the corresponding red, green and blue phosphors dots on the CRT face as mart on the right hand side of the figure, the dashed arrow shows the corresponding dots on the screen R, G and B.

Do not mistake yourself to visualize that the beam passes through the CRT face and goes towards the letters R, G and B. In this case the top one is the R and the right is the blue and the bottom one is the green respectively as the beams converge out of the shadow mask and hit the corresponding phosphor dots on the screen. I think this is an arrangement which shows you how it is done.

(Refer Slide Time: 10:38)



But if you look at this CRT face itself and see how the triangular arrangements keep going, basically what it means is that we have a phosphor dot pattern of a shadow mask CRT and this is a CRT and of course the shadow mask is in front of you, do not worry about that but the corresponding beams the R, G and B come out of the shadow mask and hit those corresponding pixels which are shown by the color, then we have this as it is shown by the circle. So one such circle is a phosphor dot basically and three of them form a triangle so that three triangles shown the top to corresponding 2D scanline.

Number one; let us say are the top scanline and similarly the bottom we have another scanline which is also going and the RGB alternates basically. So the same G could be used for the next one and so on. So the beams have to be changed as you go along. The BRG arrangement becomes basically in the next pixel RGB and so on. This is how you probably save space otherwise you need to have separate triangles, non overlapping dots, the CRT have to be very large and you cannot make it very compact high resolution within a particular dimension so that now you have a continuous picture otherwise if you have large dots separated you do not have a very continuous feeling about the picture.

Therefore, for a very high resolution CRT this is the typical arrangement of the phosphor dot pattern on this screen. Continuing with N-bit plane grey level frame buffer we will come into more illustrations and pictures but there are just a few points regarding the bit planes and colors, typically 8 bit planes per color is used. So if you want to have full color for RG and B separately we talked about three colors, each individual bit plane and we also talked about N-bit plane grey level. So why not have the N equal to 8 bit plane for each individual color? For R we can put 8 bits, for G also 8 bits and for the blue B also 8 bits. So, together we basically need 24-bits and a 24-bit plane frame buffer with 8-bit planes per color is also used. That is a typical arrangement which is desired and each group of bit planes which forms corresponding to one particular color basically drives an 8-bit DAC. It is getting a bit complicated but we will try to see.

(Refer Slide Time: 13:56)



I will give an illustration of the entire figure somehow because we talked about 1-bit plane and then of course N equal to 8 and then three color guns with each bit plane for color but we are not talking about 8-bit planes per color. So, 24-bit planes have to be put inside the video controller as well as the monitor to be driven by the video controller. Each group now generates 256 shades of intensities of red, green and blue. That means you have 256 shades of red corresponding to dark and very red and similarly you can have 256 shades of green as well as 256 shades of blue and these are each group which could generate that.

Group means we are talking about the 8 bit planes corresponding to a group and then an 8 bit register and then a corresponding DAC which will be fired by the register to generate different beam intensities corresponding to different shades of colors. So typically if you look at this we talked of 2 to the power 3 is equal to 8, 2 to the power 8 is equal to 256. Now we are talking of 2 to the power 24 possible colors that is a very very large number of colors. We are talking of about 16 million almost and that is the engineering standard which is used, 16 million colors and this is called a full color frame buffer, 16 million colors are there at our disposal and we call it as a full color frame buffer.

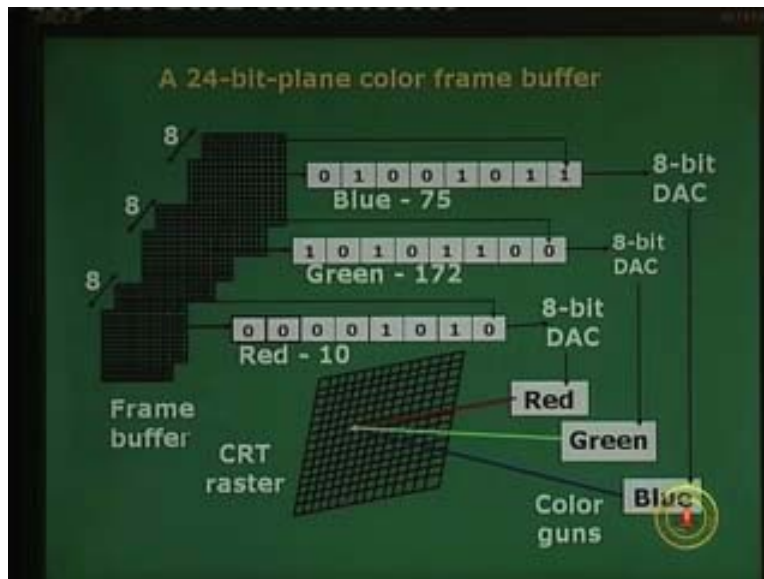
Now, before I go into the next slide I must put a point here that I am sure that whether an artist or a graphics phase designer or a web page designer in no way can visualize or even you can say need 16 million colors to draw any picture. However complex it is, how many ever sort of colors you may need to display for a screen, either you are talking about a map or a 3D diagram or simple lines and textures and things like that I do not think you need more than a few hundred. Few hundred colors are in fact quite sufficient for you to display any complicate picture.

Of course let us not go to the algorithm about the selection of colors. There are many such things existing. But what we are talking about is 16 million colors, it is impossible

to even visualize. To distinguish between two such colors if I give you a very adjacent ones just differing one bit with red, green and blue I do not think you can visualize. So it is often necessary that the graphics designer or the artist whomsoever, we talk about, when the artist paints also with a color he will make certain colors with again different shades on his canvas and then use it to draw and he probably does it with a few tens and different types of colors and draws it in layers and phases.

When we look at color shading we will see there is a corresponding layer. The Peters algorithm we talked about uses a concept of what a painter does in terms of shading but really you need more than few tens or couple of hundred different colors to do that. So why do you need. We talk about 24-bit full color frame buffer because it is also difficult to implement, you do not need that, so you select the set of colors out of these. Let us go to the next slide to see the complexity involved in a 24-bit full colored frame buffer and we will see how to do that as given here.

(Refer Slide Time: 16:06)



We have three groups of 8-bit planes each, each for individually separate colors on the left hand side and then the frame buffer. We will see the black colored frame buffer groups of 8, each group of 8-bit planes for red, green and blue respectively. So the corresponding bit values are loaded on to the 8-bit registers. This is a typical example where the 0s and 1s are shown in binary and the corresponding decimal value for blue equal to 75. You can check it out yourself: green equals to 172 and red is equal to 10 1010 that is, 10 is also displayed. That means we are going to display a particular point with color intensity where the red component which will be 10 out of 256, you get a fraction of that. Green is 172 also 256 and 75 out of 256. Basically the programmer does not need to load values like this. He typically loads in a normalized range from 0 to 1.

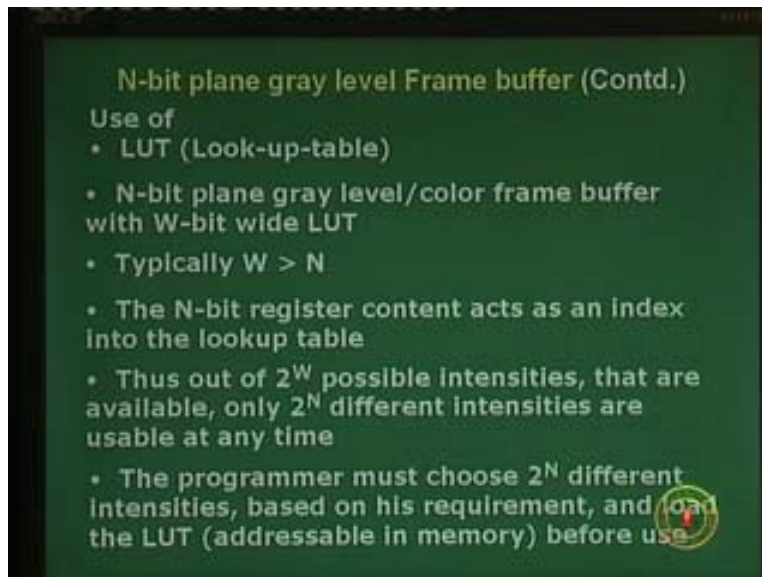
From 0 to 1 the values are put on to the bit plane and then it is automatically loaded on to the register which again drives an 8-bit DAC. Now there are three 8-bit DACs, three

color guns, three 8-bit registers, 24-bit plane. So that is a huge demand on a bandwidth and memory requirement and the color guns will of course go and ignite the phosphor dot triangle on the screen to eliminate this color. Now in this arrangement typically you will have 16 million colors possible. But typically it takes a raster screen which is of a maximum resolution of about few 100/100, even take to eliminate by 1024 for 1700.

We are talking of the order of about 10 to the power of 6 or different pixels and even if you put each like a random dot color pattern, each different color on the thing you will not probably need 16 million colors and that is stupid. You are not talking about trying to put random dot color arrangement and show something. So you need different set of colors, you definitely need fixed set of colors to draw any arbitrary picture and you do not need to worry about 2 to the power of 24 colors which you will split.

To make the life of the programmer easier he does not have to worry about 16 million colors which you choose from, we use the concept of what is called a look-up-table just before the frame buffer shortly. It basically sits between the frame buffer and the registers. We talk of N-bit plane grey level color frame buffer with W-bit wide LUT or Look-Up-Table or luters as it is called. There are different pronunciations but I will use the word LUT correspondingly as given here are a lot.

(Refer Slide Time: 21:18)

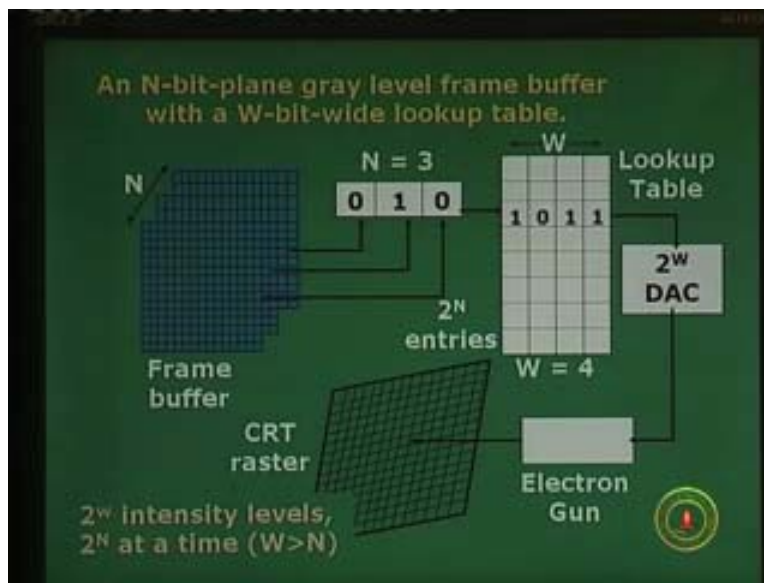


Typically the value of W is much much more than N. That does not mean we are choosing a 24-bit plane, now W was more than that. That is not the key, let us try to visualize what will happen if N is equal to 8. Try to visualize N equal to 8 and W could be more than 8, it could be 12, 16, or up to 24 but you do not need more than 24. So W could be 24 up to maximum and N less than W is what you have to visualize. The N-bit register will load the contents of the frame buffer on to the n-bits of the register, the content of that means the corresponding byte value which comes out of the register that acts as an index.

Index, I hope you understand, in terms of memory address; Index to the look-up-table. So we visualize look-up-table as a sort of sequence of memory positions. How to address the corresponding position in the memory address depends upon the index pointed out by the register or from the look-up-table and that index will be used to access the corresponding contents of the look-up-table. Thus if you have this arrangement, we will put the figure now. Thus out of 2^W possible intensities because the look-up-table is W -bit wide it can hold obviously 2^W possible values or 2^W possible intensities and that is what is available. And out of that only 2^N different intensities are usable at any point of time, why? Because that is what will be loaded inside the frame buffer. The frame buffer is N -bit wide so you will have 2^N possible values so the look-up-table W possible values. So at any given point of time you can choose 2^N usable values out of 2^W possible intensities.

Remember, W is more than N . So 2^W possible values is much larger than 2^N values which can be loaded into the frame buffer. But you can choose which of those 2^W intensities or values of color you would like to use and choose them and load the frame buffer. So this is how the look-up-table basically helps the programmer to choose these colors efficiently and without bothering about very large value. That is what I was saying, the last point, the programmer must choose 2^N different intensities based on his requirements and load the LUT or the look-up-table which is addressable position in memory like we discussed about frame buffer which is addressable, the LUT is also addressable and that has to be loaded before use. Well, we look at the diagram of N -bit plane grey level frame buffer with W -bit wide look-up-table, which is the overall element in the simplest situation.

(Refer Slide Time: 26:39)



I have taken a simple example so that first of all the scenario looks simple and of course fitted on a particular screen itself. We are talking of an N -bit frame buffer N is equal to 3

simple cases which we have chosen. You remember, we got the EGA 8 different colors possible. So now $2^3 = 8$ different possible intensities you can use or the programmer can use and the question is he may need only 8 but which 8 out of that many. I mean he can choose 8 out of $2^4 = 16$ million colors but he is happy with 8 he does not need more than 8 is what is to be assumed.

So, the contents of the frame buffer of loaded on to a 3-bit wide register and that register contents now act as a, you see a pointer at the look-up-table from the register, when $n = 3$ is written there is a pointer to the look-up-table so that in the contents now the value is in this case 2 so 0, 1, 2 so it is pointing to the third row starting from 0. The count 0, 1 and 2 is pointed to the third row of the look-up-table which has the value 1 0 1 1. What is that value? It is $8 + 3 = 11$. But what I mean is the contents of the register acting as an index or a pointer to the look-up-table entries and the corresponding third row or third content will be picked out from the look-up-table which is W bit wide.

Now this is interesting. Register was 3-bit wide eight possible values but W is 4-bit wide so 2^4 or in fact you may have, out of 2^4 possible values, the look-up-table number of entries will be only 2^3 because that is what the register can access. The register with 3-bit wide or N -bit wide cannot access more than 2^N entries or $2^N - 1$. Thus basically it is look-up-table with 8 entries because $N = 3$, $2^3 = 8$, $2^3 = 8$ and 0 to 7 so there are 8 entries in look-up-table but each is four bit wide. That is very interesting.

Each is 4-bit wide and this content is 11 which is coming out from the third location point it to the register and the look-up-table output goes to 2^W DAC, that is interesting. Now the DAC instead of being 2^N -bit wide is basically having capacity of 0 to 2^W will have different levels from 0 to 2^W . So it has more number of capacity then what the frame buffer can access and that is an advantage because now $2^4 = 16$.

So, out of 16 possible intensities we choose your 8 that the programmer chooses his 8 possible intensities whatever he needs for to draw this picture or color and then loads the corresponding 8 positions of the look up table with only those intensities. So the programmer prairie knows where the color which he requires is and in what position in the look-up-table his desired color is any, 8 desired colors out of 16 which is 8 out of 16. So he chooses 8 out of 16, loads the look-up-table with the corresponding 8 entries and accesses it by a register. So that picture I hope is very clear to you and the look-up-table output goes to 2^W DAC and the DAC gives the corresponding analog voltage intensity, analog voltage which is proportional to the input to the DAC. So out of 16 possible values he gets an 8 but it has the capacity to give any output from 0 to 15 is also the right possible value. So electron gun up to the CRT and the rest of this part is very simple.

This is a typical arrangement, hope you understand from the figure about an N -bit plane grey level frame buffer with W -bit wide look-up-table. If I did not use this look-up-table

what will happen without that look-up-table? The N-bit contents of the register would have directly gone to the 2^N DAC. Directly from the register it would have been driven to the DAC and would have a 2^N DAC and would you have only eight possible intensities but fixed eight intensities is at any point of time and that is what the electron gun is able to fire and what you can choose. Now you can choose any 8 out of 16 in fact increase W.

Take W equal to 8 if you take W equal to 8 what you are visualizing? Well, the size of the look-up-table in terms of the content entries could be still 8, it depends on the value N which is 3 but you can now choose those 8 colors from 256 colors because W is eight bit wide so there are 256 possible color values or grey level intensities and you can choose 8 out of 256 and load the look-up-table and address just the corresponding 8 and you will then use it to 2^W DAC which needs because the output will be a W bit wide from the look-up-table. So you need the 2^W DAC not 2^N to fire the electronic gun.

So we are talking of 2^W intensity levels; 2^N at a time when W is more than N. If you look at the bottom left of your screen that is just common which we are just discussing; now we talking of 2^W intensity levels, 2^N at any given point of time. Now, if we change the look-up table content those 2^N which is accessible, then those values could change. The 2^N number remains constant but the entries in the look-up table could get changed and at any given point you can basically have two pictures with two different look-up tables on a particular screen entirely having two different sets of colors chosen by the programmer.

Now, the most complex seen is that we are looking at a 24-bit plane color frame buffer with a 10-bit wide look up table. Well I cannot put all that here but what we are looking here is we are looking at N equal to 8 because 24-bit. So 8 into 3 is 24 N-bit wide. Assume N equal to 8 so we have a 24-bit plane, there are groups of 8-bit planes as you see on the left hand side, one for red, one for green, one for blue and each is N-bit wide. The contents are put to a horizontal register here, this is the register for red, that is the register for green and that is the register for blue, those registers are all N-bit wide. In this case again since it is a 24-bit plane we are talking about N is equal to 8 but you can even take a typical case of N equal to 4 so we are talking about may be a 12 bit plane so that does not matter.

What it basically means is you are talking of N-bit wide register in this case is 8. So we have 2^N entries for each color set for the frame buffer. In frame buffer for each group or for each individual color consider only R, only green or only blue or any one of the three colors. We have N-bit planes for all, we have an N-bit register for the red and the contents of the N-bit register are fed as an entry to the look-up-table. So you are looking at any one out of the 256 possible entries in the look-up-table because 2^N we have taken N to be 8 so 2^8 .

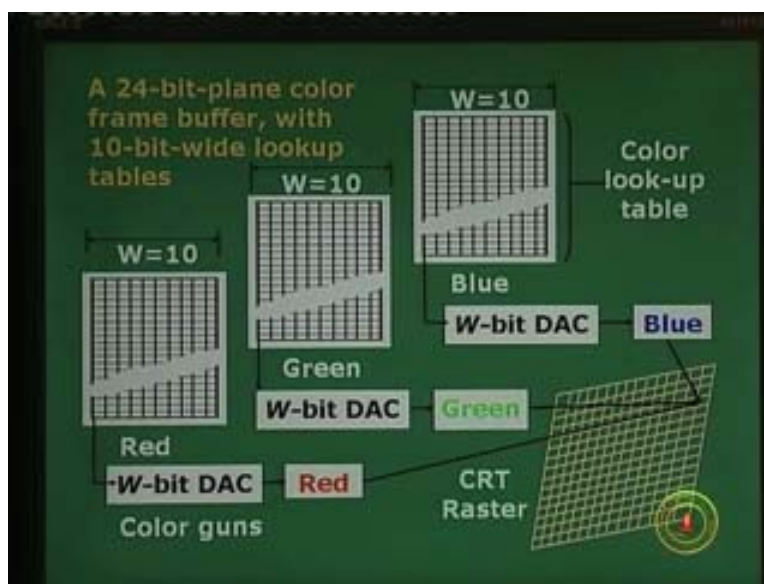
There are 256 entries in the look-up-table for each channel similarly for red similarly for green similarly for W. That is an example of a color look-up-table here. This is the look-

up-table for blue on the right hand side, middle we have look-up-table for green and left hand side bottom we have the look-up-table for red. It is not necessary that W should be equal to N we discussed about this earlier.

In this case we have assumed that the look-up-table is 10-bit wide, W equal to 10 in all these cases red, green and blue. That means we have been looking at intensities of 2 to the power N which is about 1024 one make, 1024 possible intensities out of 1024 possible intensities of red separately, green separately and blue separately. That means the darkest value of the red or any color up to the brightest value of the particular color. That entire range is now split into 1024 different quantum levels or discrete steps is what you can imagine from 0, 1, 2 1024 minus 1 or 1023. So, out of 1024 possible values, where is 1024 coming from? Well W equal to 10, 2 to the power 10 so that is the way it comes out, 1024 or 1k that is one make of 1k values. Out of those 1k possible values which you can assign on to the look-up-table the programmer prairie chooses 256 or 2 to the power 8 possible values and loads the look-up-table.

Now why you choose 256 again? It is 2 to the power N . As we look into the figure 256 possible entries but each entry is 10 bit wide. Each entry is 10 bit wide will be one color red, green and blue, one color obviously. But the shade of that color will be 1 out of 1024 possible values two to the power ten two to the power W as we talk of. So 1 out of 1024 values will be loaded in each entry of the look-up-table and there are only 256 such possible values in one look-up-table, shades of red, shades of green and shades of blue respectively. These are the three different shades which we are talking about for three different rates. And now you can imagine this look-up-table contents will now be passed on the left I could put it all in one slide. If you look back into the slide again it runs from the 24-bit plane, if you recollect the picture which was just shown then we are looking at 24-bit plane to the W width wide look-up-table. So, we move to the next slide.

(Refer Slide Time: 31:28)



When we move to the next slide it is a continuation of the previous picture so I hope you are able to see the next slide. We are looking at a 24-bit plane color frame buffer with a 10 bit wide look-up-tables and for these now you can see the picture. I will show the continuity of the picture I will go back to the previous slide. So we are looking from a 24-bit plane frame buffer to a look-up-table and from look-up-table now you are seeing the remaining part of the picture because if I had condensed these two into one picture things would have been small for you to visualize what is going on so this is the continuity.

So we stopped in the previous slide with the look-up-tables. Now the look-up-table contents are being addressed by the corresponding register one out of the 2^n or 256 positions. One such content will be dragged out like a memory address and then the memory contents are taken out and it is passed on to a W bit wide DAC, W-bit wide Digital to Analog Converter three DACs again for three separate colors R, G and B or red, green and blue and the DAC contents are again put to the corresponding red, green and blue colors and they are all converted to a single pixel to a fast triangular phosphor dot arrangement for RGB in a single raster frame. I hope this arrangement is getting clear. I will just role back one slide back and forth for you to visualize the continuity. These two slides are from one picture basically we are taking this from the book by Roger.

We are talking about a 24-bit plane color frame buffer and then we are talking about the N-bit wide. We have N-bit wide registers sitting in between three registers for each color, three look-up-tables for separate colors. Then from the three corresponding look-up-tables, the next picture is, we fill it to the corresponding three DACs each W-bit wide then again the three color guns, three color guns generate the corresponding intensities, they all fire on to the CRT raster.

This is probably the last slide which we look in terms of CRT display monitors to wind up the remaining part of the lecture on display devices. We may not come back to that again. I thought you have done extensively on CRT monitors specifically to do with the raster refresh video scan, basic refresh rate, memory access, architecture and so on. We looked at the arrangement of guns, look-up-tables and all that which is necessary. Those are the common display devices which are used, the monitored ones. Therefore the other types of display devices I gave at the beginning about three classes back and I think we must just touch upon a few points about the other modern days display devices which are coming out. Consistently as you see it was almost on the screen, the TV of course EC or monitor based but we started to have the LCD panels or the flat panel display monitors which are different. They have some advantages with respect to the CRT monitor. I thought we will wind up this discussion with a few points about LCD and flat panel displays.

So LCD is a Liquid Crystal Display device made up of basically six layers. The first of them is a vertical polarizer plane then the second layer is the layer of thin grid wires and the third is the layer of the liquid crystal which is in the heart or in the corner, the center of the displays and then again the layer of horizontal grid wires then of course again horizontal polarizer as the fifth layer and finally reflect. So that is made up of six layers.

The crucial part is the two polarizers and the layer of LCD is in between and that is what causes the LCD to function, it glows or it becomes dark. The LCD material is made up of long crystalline molecules. So when the crystals are in an electric field they all line up in the same direction. So it is more to do with devices and technology of materials. So we assume for the time being we do not have much time to go into details for this course. So touching upon it LCD material is made up of long crystalline molecules and the crystals line up in the same direction when the electric field is applied.

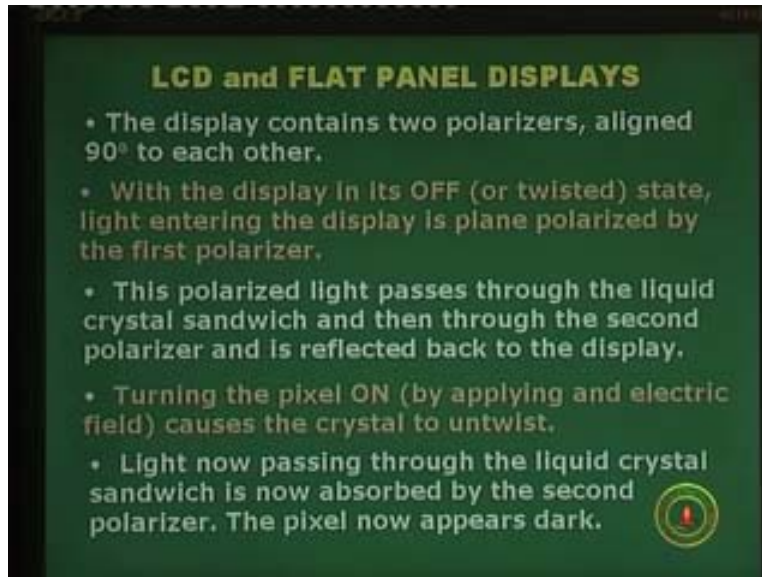
(Refer Slide Time: 36:06)



The active matrix panel is also a term used for all these basically flat panel displays and also LCDs occasionally. Active matrix panels have a transistor at each grid point x, y at each position and crystals are dyed up to provide the colors. So you can have LCD and color monitors as well. And the transistors basically act also as a memory because they also cause the crystals to change their state quickly and to hold on the voltage till the transistor stage is switched back again. The advantage of LCD displays is that you can almost imagine they are there in your calculators and palmtops, desktops and even on other controls sometimes on TV, air conditioners, handling devices like cell phone, mobile units and all that. So LCD displays are very low cost, are low in weight, they are small in size and they have a very low power consumption as well.

Continuing with that, the display contains the two polarizers. In fact we discussed about six layers and the task mainly is done by the two polarizers and the LCD crystals in between. The two polarizers align at 90 degree to each other. We talked of horizontal as well as vertical polarizers and with the display in its off condition or in its off state the light entering the display is plain polarized by the first polarizer.

(Refer Slide Time: 37:45)

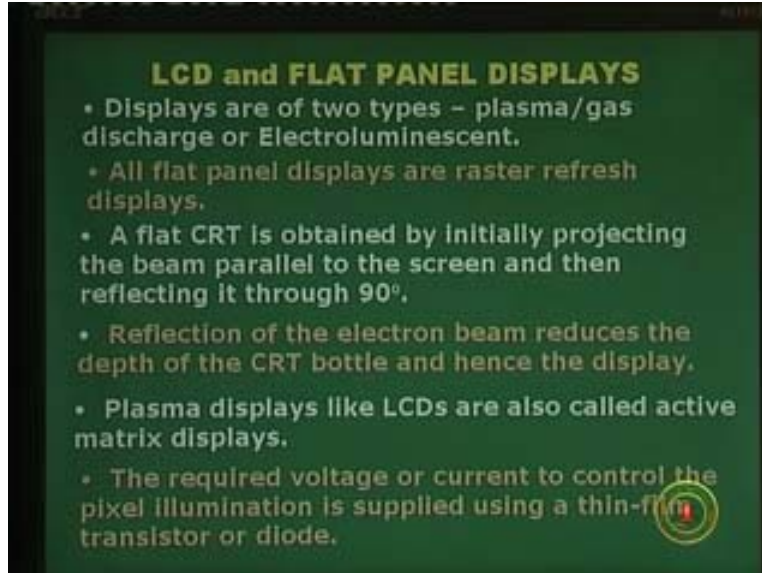


When the display is in off state the light entering the display is polarized by the first polarizer then what happens? This polarized light passes through the liquid crystal sandwich. So we are talking of layers and in between liquid crystal sandwich and then through the second polarizer and is reflected back to the display. So in fact when the display is off the light is allowed to pass through these layers, pass through the first polarizer the sandwich layer of crystals in between and also to the second polarizer which is if it is horizontal then you have the vertical one and the light is allowed to go for the display and we see the portion of the LCD illuminated, illuminated or it is lightened.

When turning the pixel on means it is basically done by applying an electric field causes the crystal to untwist. The crystal changes its position or state and that causes something else because the light now passing through the liquid crystal sandwich is observed by the second polarizer. In the previous case it was allowed to go through, in the second case it is observed by the second polarizer and the pixels starts to appearing dark. So dark and light is what is controlled by the corresponding transistor or the electric field by switching this state off the crystal in terms of twisting or not twisting so that is logic which we must just keep in mind.

One has to study extensively about LCD panels by opening manuals and books to understand the technology behind it. Well just again we are going into the introductions of these, basically displays are of two types; plasma gas discharge or electroluminescent. We are talking of flat panel displays they could be of two types. All flat panel displays are raster refresh displays.

(Refer Slide Time: 40:47)



We talked of raster displays at length in terms of the CRT monitors or all flat panel displays like the CRT monitors are the refresh raster. A flat CRT unlike the monitor is where you have a big arrangement of the electronic gun the anode heating element and then of course the horizontal deflection plates, control gates, focusing unit either electro magnetic coil to reflect the voltage or analog or deflection plates horizontal and vertical and then the beam passes through the shadow mask has to fall on the phosphor coated screen and dots and all those arrangements makes the CRT look very big. You can see the TVs or the monitors with the CRT. If it is inside the CRT it has to be big and voluminous because of this and they are rather pollen because it increases the weight of the system there is heat generation where ever it works and also it occupies space, volume and all that.

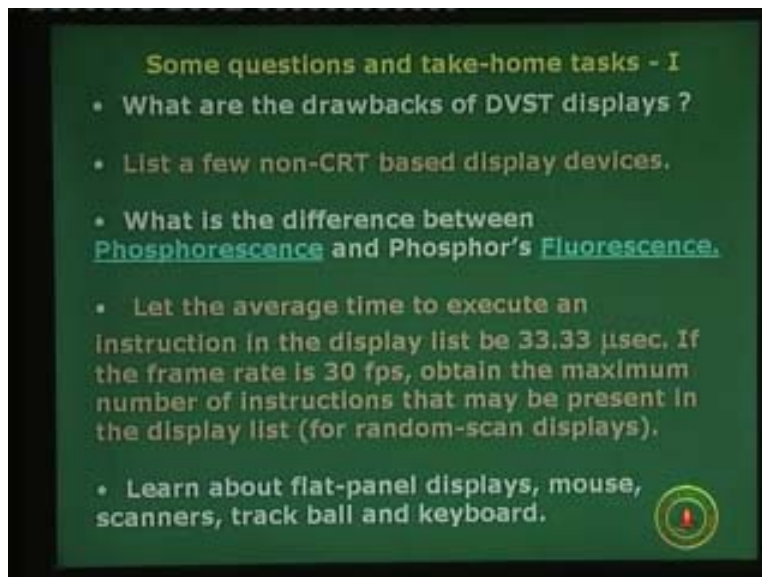
The flat CRT has a very big advantage that it is sleek because it is flat, the flat CRT is obtained by initially projecting the beam parallel to the screen and reflecting it to 90 degree. So typically when you are watching a screen, a flat panel unlike the electron beam for a CRT monitor which directly goes almost perpendicularly to the phosphor screen and then it is deviated and impinges on a phosphor dot on the screen it is other way round that the beam is basically running almost vertically along the screen and then diverted and hits the particular point on the screen. It is reflected by 90 degree and hits the existing plane so that is the advantage of a wider screen. Reflection of electronic beam is done by of course electro optical devices, reduces the depth of the CRT bottle and also the display so that is obvious.

And the plasma display like the LCDs is also active matrix displays. They are also called the active matrix displays here. And the required voltage or current to control the pixel illumination is supplied using a thin film transistor or diode. That is the technology which is used to control the pixel illumination because we need to control the intensity also

there along with the color so that is done by a thin film transistor on diode. This brings us to the end of the lectures on CRT display devices.

We had a sequence of four hours or four lectures and we have discussed extensively about CRT monitors because that is what is the basic target .we should know about the technology. And we touched upon a few points about a LCD panels and a flat panel display monitors. The other types of output devices of course which we may not touch, input devices like the keyboard and the mouse we talked about and of course output devices like plotters and printers and all that. But with respect to the time allotted I think I will move over to the algorithms and other concepts on graphics and you have to bear with me that I could not talk much at length about other interface and other input output devices used in computer graphics. And I spent most of the time on CRT display devices and touched upon flat panel devices. I have made at the end of each lecture a set of questions which you should try to solve it yourself, answer yourself. And even there are some problems given in each of these and they will be shown at the end of each slide. And please go back learn more from the books and references which have been coated in the introduction slide and try to sole these problems.

(Refer Slide Time: 42:17)



Some questions and take-home tasks - I

- What are the drawbacks of DVST displays ?
- List a few non-CRT based display devices.
- What is the difference between Phosphorescence and Phosphor's Fluorescence.
- Let the average time to execute an instruction in the display list be $33.33 \mu\text{sec}$. If the frame rate is 30 fps, obtain the maximum number of instructions that may be present in the display list (for random-scan displays).
- Learn about flat-panel displays, mouse, scanners, track ball and keyboard.

Although we said that you could solve all the problems at home we will discuss a few selective set of problems right now and we will choose one problem from each particular task. So kindly choose one such problem from the take home task one.

Can we discuss task one fourth problem?

Yes that is nice, that says that the average time to execute an instruction in the display list is about 33.3 microseconds and if the frame rate is 30 frames per second the problem says obtain the maximum number of instructions that may be present in the display list for random scan displays. If you remember, random scan displays was basically a line drawing display device and it had in the display list a set of instructions so it is a

sequence of instructions which are basically executed one after the other. And we want to find out what is the maximum number of instructions which could be placed. So solve the problem, let us come here.

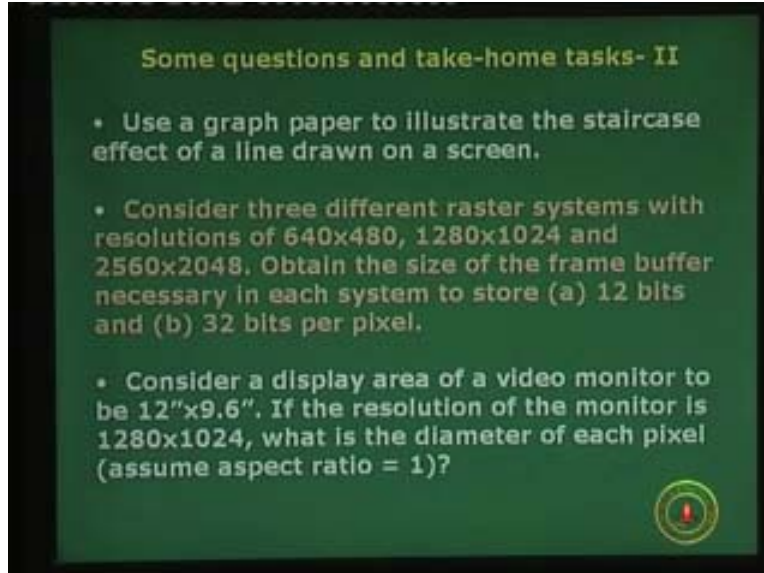
(Refer Slide Time: 44:29)

Handwritten calculations on a chalkboard:

$$\begin{aligned} & 33.33 \mu\text{sec} \\ \text{FRAME RATE: } & 30 \text{ Hz} \\ \text{TIME TO SCAN} & \\ \text{ONE FRAME: } & \frac{1}{30} \text{ sec} \\ & = 33.33 \text{ ms} \\ \text{Maximum} & \\ \text{number of Instructions} & \\ & = 1000 \end{aligned}$$

The problem says that we have a frame rate of about 30 hertz that is the frame rate and from that we can easily obtain time to scan one frame. The time to scan one frame multiplied by the frame rate is the number of frames per second fps or hertz gives one second. So time to scan is 1 by that value which is $1/30$ seconds or 33.3 milliseconds, $33.3/1000$ into 10 to the power minus 3 that is the engineering unit, milliseconds and hence since the problem states that the time to execute anything instruction is only 33.33 microseconds that is 10 to the power minus 6 unit here and time you have actually is 10 to the power minus 3 the unit here for one frame. So, total number of instructions should be this divided by this 33.3 and the answer is very easy that you can actually put 1000 frames per second. That is very easy to compute, the answer is 1000 microsecond. Let us go to the other question.

(Refer Slide Time: 44:37)



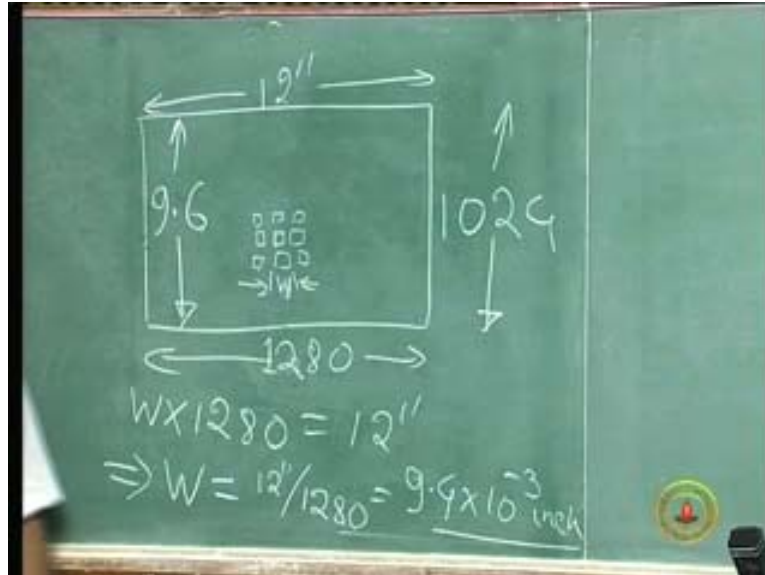
Some questions and take-home tasks- II

- Use a graph paper to illustrate the staircase effect of a line drawn on a screen.
- Consider three different raster systems with resolutions of 640x480, 1280x1024 and 2560x2048. Obtain the size of the frame buffer necessary in each system to store (a) 12 bits and (b) 32 bits per pixel.
- Consider a display area of a video monitor to be 12"x9.6". If the resolution of the monitor is 1280x1024, what is the diameter of each pixel (assume aspect ratio = 1)?

Sir, can we discuss task two third problem?

Task two third problem, that is good. Well, you see this is something to do with the resolution of the screen. If you remember, when we were talking about resolution of the screen there were concepts of frame buffer, bandwidth and scan rate and things like that. But the actual resolution which is there on the monitor, it depends on the pixel size, physical dimension of the pixel size. So let us look at this problem. That is where, in this case although this problem is very easy we said that if you have a video monitor where the size is 12 inch in width and 9.6 in height and we had said that along this we have 1280. So we basically have one to put 1280 pixels along each 1280 pixels along each row and each row will be 1280 and the total number basically will be 1280 columns and the number of rows is 1024. There are 1024 rows and 1280 columns of pixels. That means the row width is basically 1280 and the height of a column is 1024 so it is the other way we can visualize it. Or you can also say that the number of columns is 1280, number of rows is 1024.

(Refer Slide Time: 47:47)




That means each pixel, I have a small pixel here that is arranged. We assume that for the time being they are arranged in the form of a matrix and there are 1280 such that in the horizontal direction x direction let us say and in the y direction 1024. And the question is how many such pixels we can pack? It depends upon the physical dimension so the number of pixels multiplied by each unit gives this. So we can say that if each such unit is of width W then we will say W multiplied by 1280 gives you actually 12 inches and which in turn basically tells you the W that is the width will be actually 12 inches by 1280 that is very simple. And that the answer is about 9.4 into 10 to the power minus 3 inches. So we can actually get these values also by dividing $9.6/1024$. You will also actually get this value you can see and feel for yourself that it is almost close to 1000, 9.6 here. So the value will be little bit numerical less than 9.6, it will be 9.4 into 10 to the power minus 3 inches so that is the width and the height for the each individual pixel and that is the way you compute the size. Let us go to the next one.

(Refer Slide Time: 47:48)

Some questions and take-home tasks- II

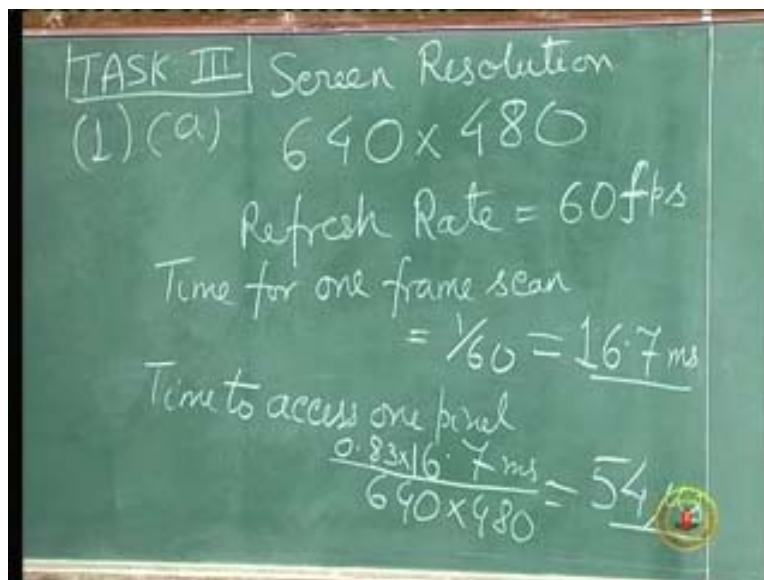
- Use a graph paper to illustrate the staircase effect of a line drawn on a screen.
- Consider three different raster systems with resolutions of 640x480, 1280x1024 and 2560x2048. Obtain the size of the frame buffer necessary in each system to store (a) 12 bits and (b) 32 bits per pixel.
- Consider a display area of a video monitor to be 12"x9.6". If the resolution of the monitor is 1280x1024, what is the diameter of each pixel (assume aspect ratio = 1)?




Sir, in task three can we discuss the first question?

Task three first question as said here, we are asked to compute the access time for each pixel for systems which have two different resolutions as given there. One is 640 into 480. The other is 60 and 1280/1024 is the other resolution and the refresh rate is 60 frames per second. Let us take task three and problem one and we are talking of a resolution a. So we are talking of a resolution which is 640 into 480 that is the image resolution or screen resolution also you can say. I will better put it as screen resolution.

(Refer Slide Time: 52:05)



TASK III | Screen Resolution
(1) (a) 640 x 480
Refresh Rate = 60 fps
Time for one frame scan
= $\frac{1}{60} = 16.7 \text{ ms}$
Time to access one pixel
 $\frac{0.83 \times 16.7 \text{ ms}}{640 \times 480} = 54$

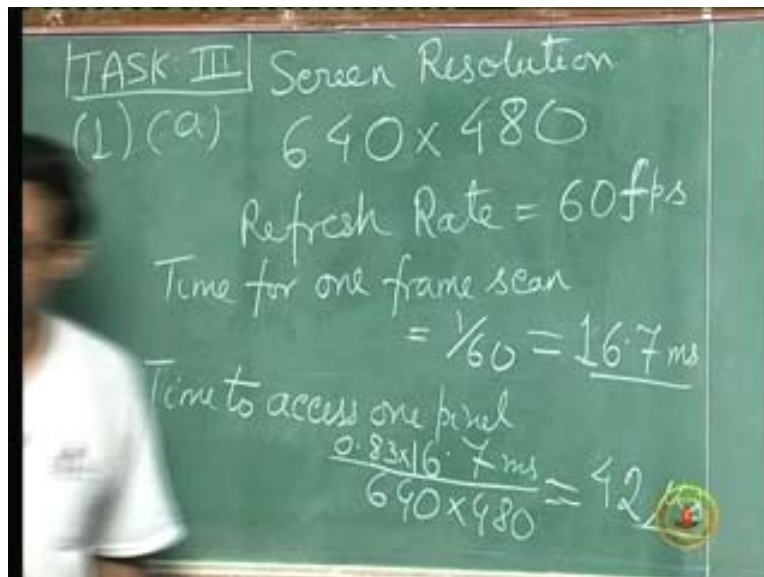


And we are talking of a refresh rate of 60 hertz or 60 frames per second. I did teach you that you can use these terms interchangeably hertz means per second or fps is frames per second. So if that is the frames per second time for one frame scan or basically meaning scanning one frame, time required to scan one frame I write it as time for one frame scan is basically $1/60$, how much it comes to $1/60$? It will come to about 16.7 milliseconds. You can check it out yourself it will come to about approximately 16.67 or 16.7 milliseconds. It is 16.67. So 16.67 so I have rounded it off to 16.7 that is good. So 16.7 milliseconds must access so many pixels 640 into 480 pixels must be accessed in 16.7 milliseconds.

Thus, time to access to one pixel is actually you can take the 16.7 divided by 640 into 480 so many milliseconds. If you divide this and compute yourself you can check the answer it will come to about 54 microseconds. It will come to about 54 microseconds. Now this value of 54 microseconds actually includes the time for vertical retrace. That means individually accessing all the pixels from left to right per row and then from second row, third row and all that. And then you need to have time for retrace, both horizontal and vertical retrace times.

Actually if you take into account that time which typically takes about 17%, about 17% of the time then you will be left over with about 0.83 of the time so this value 54 microsecond will actually you need to probably put a multiplying factor of about 0.83 assuming 17% of the time is gone for retrace per frame in out of 16.7 millisecond. So you will be left over with 0.83 of the time. If it is 15% for retrace then you will have 0.85. That value goes down by some amount, typically you can calculate for yourself 54 it will come to about 42 microseconds or so, you can check it out yourself how the value is, **can you check that out?**

(Refer Slide Time: 52:34)



You can check it out right now it comes to about 45 precisely about 45. You can round it off; the value comes to about 45 microseconds. That is a part of the problem and I will over write in a similar way. The part b says that the resolution is increased, the resolution is increased to 1280 into 1024 so we move into the second part of the problem of task three problem one where the resolution now is 1280 into 1024 keeping the refresh rate per same. Time to access for one second is also same. And we keep that factor of 0.83 here. So I will say .83 into 16.7 milliseconds divided by 1280 into 1024. You can have a calculator yourself and you can check it out. It comes to 10.5.

(Refer Slide Time: 54:51)

TASK III | Screen Resolution
(1) (b) 1280 x 1024
Refresh Rate = 60 fps
Time for one frame scan
= $\frac{1}{60} = 16.7 \text{ ms}$
Time to access one pixel
 $\frac{0.83 \times 16.7}{1280 \times 1024} = 10.5$

You can see that the bandwidth requirement is probably so large now and this is the time to access for pixel. We discussed about bandwidth requirement now, where within this time to access per pixel it has to go on, we must go and switch on and switch off. We are talking about almost 5 microseconds of access time, we are talking about almost the bandwidth requirement of 1 Mb, 1 mega hertz type of bandwidth is what we are requiring. But of course there are higher resolution monitors that are also available, higher frame rates and we can start to visualize the amount of time in which the beam has to switch on and switch off, access a particular pixel, will be still and going down further and further. Can we discuss some other problem? We just have time.

(Refer Slide Time: 54:55)

Some questions and take-home tasks-IV

- Assuming a transfer rate of 0.1 MB/sec, how much time would be necessary to load pixmap with resolutions: (a) 512x512x1, (b) 1024x1280x1.
- We require large refresh rate mainly due to the short persistence of the phosphor. Why not use a long persistence instead, to reduce the frame rate ?
- Obtain the percentage of time (per frame) when the beam does not trace any image, in these cases:

Visible Area	FPS	Inter lace	Retrace Time (VER)	Retrace Time (HOR)
640x485	30	No	1250 μ s	11 μ s
1280x1024	60	No	1250 μ s	7 μ s
1280x1024	60	Yes	600 μ s	4 μ s

Task four first problem.

Yes, that is a very interesting problem. Task four problem one, thank you. Assuming the transfer rate of 0.1 mega byte per second, so we are talking about it transfer rate of about 0.1 mega byte per second which is almost, I will take it approximately equal to 100 and 100 kilo byte per second. It is not true but it is an approximation 1 mega make is about 1024 so it should actually come to 102 kilo byte per second, you can actually take that. And then how much time would be necessary to load pixmap with resolution? So the first problem talks about 512 into 512 into 1 so many bits. It is in bits and that is in bytes so to transfer these into bytes I must divide it by 8 so there are so many bytes have to be transferred and the rate is known as 100 kilobytes per second.

Therefore, if you calculate this basically the time requirement taken so in 1 second or 100 kilobyte you can transfer in 1 second so this actually comes to, you can calculate yourself, it comes to about 32, this is actually about 32 kilobyte. You can check it out yourself because it is 512 divided by 8 will actually give you 2 to the power 6 so you will have 2 to the power 5 here and 1 term going here to take care of the kilobyte so 2 to the power 6 is what you get. Basically get 32 kilobyte that needs to be transferred, 100 kilobyte in 1 second so in 32 kilobyte the time required will be about 0.31 seconds so that is the answer for the first part.

I will quickly overwrite the answers. With the second part if I say the second part that is the part b) has 1024 and 1280 that is more or less the large monitor, 1024 into 1280 divided by 8 so many bytes have to be transferred and this typically goes to about 157 kilobytes, you can check it out for yourself, actually it will come to about 157 kilobytes. Basically you have to transfer the bytes into kilobytes so you can compare if in 100 kilobytes you have to transfer in 1 second so 157 kilobytes you need, the answer is very easy 1.5 seconds so that is the answer. So 1.5 seconds is the answer I have. You should try rest of the problems which are given in the take home task, thank you very much.