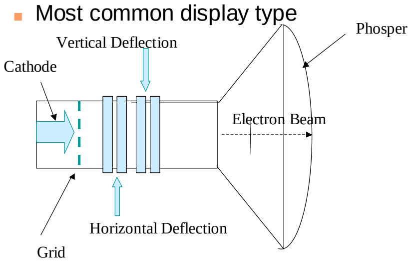
Display Technology

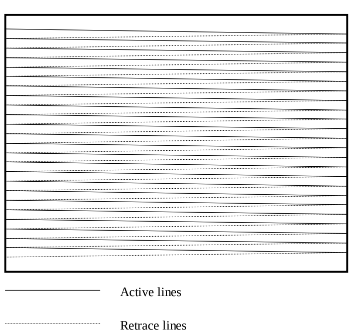
- Display Technology Tree:
  - CRT
  - Flat Panel
    - Active
      - Fluorescent
      - Gas Discharge (plasma)
      - Electroluminescent
      - LEDs
      - Incadescent
    - Passive
      - Liquid Crystal Displays (LCDs)
      - Electromechanical

Cathode Ray Tubes

- Most common display type



Raster Scan



Beam directed in lines (scan lines) across screen.

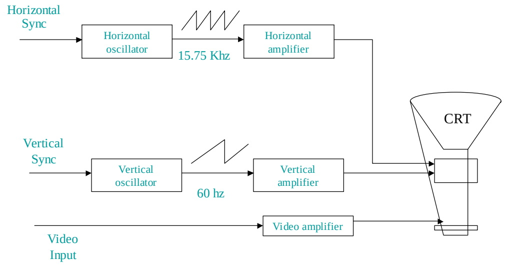
Dots are created by turning the beam on & off.

During *retrace*, the beam is off.

***Horizontal retrace* is when beam returns to left edge**

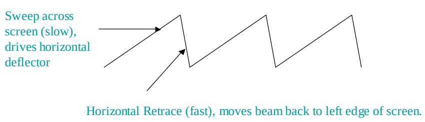
***Vertical retrace* is when beam returns to upper left corner.**

Horizontal, Vertical Syncs

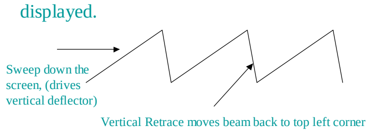


4/23/01 4

The horizontal sweep controls the number of lines on the screen



The vertical sweep controls how fast each screen is displayed.



4/23/01 5

How many lines on screen?

- Let Horizontal sweep = 15.75Khz
- Let Vertical sweep = 60 hz
  - Num lines = horizontal/vertical = 15750/60 = 262.5
- These numbers are for standard North America broadcast television
  - Half line allows next field (next 262.5 lines) to be offset by one half line (even & old fields). Two fields make one frame
  - Gives an interlaced display of 524 lines refreshed at 30 hz
    - Human eye can detect flicker at 45 Hz - don't notice flicker on TV because of image types

4/23/01 6

How many Dots on screen?

- Video input controls whether beam on or not
- How fast we can turn beam on/off during horizontal trace time
- Monitors use internal clock to sample video signal
  - Monochrome - only one line for video signal
  - Digital RGB - three digital lines (Red, Green, Blue)
    - **Gives 8 colors (a 4th line, an intensity signal, can be used to give 16 colors)**
  - Analog RGB - three analog lines, each driven by 8-bit DAC - gives 256 * 256 * 256 = 2^{24} colors

4/23/01 7

VGA Timing (640 dots x 480 lines)

Horizontal Sync = 31.5 Khz, Vertical Sync = 60 Hz

Internal Monitor clock (Dot Clock) for latching video signal is 25.175 Mhz

#max dots per line = Dot Clock Freq/ Horizontal Sync

= 25.175 Mhz / 31.5 Khz = 800 Dots

Only can use 640 dot times out of possible 800 for display because we need black areas on left/right edges and time for horizontal retrace.

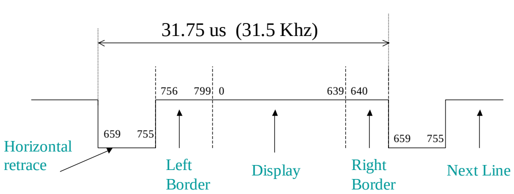
#max lines per screen = Horizontal Sync/Vertical Sync

= 31.5 Khz / 60 hz = 525 lines

Only 480 lines usable, need blank areas on top/bottom, time for vertical retrace

4/23/01 8

Horizontal Sync Timing

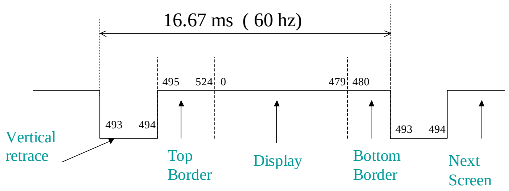


800 dot times per line

Counter can be used to keep track of horizontal screen position.

4/23/01 9

Vertical Sync Timing



525 line times per screen

Counter can be used to keep track of vertical screen position.

4/23/01 10

Other resolutions

- 800 x 600
  - Dot clock 36 Mhz
  - Horizontal Sync 35.15 Khz
  - Vertical Sync 56 hz
- 1024 x 768
  - Dot clock 64.142 Mhz
  - Horizontal Sync 48.3 Khz
  - Vertical Sync 60 hz
- Allow about 20% of horizontal trace time for borders, retrace
- 6% to 8% of vertical trace time for borders, retrace

4/23/01 11

What drives video signal?

- Every dot clock time for visible pixels, need to determine value of video signal
  - For monochrome, 1 bit per pixel (black or white, on or off)
  - For Digital RGB, 4 bits per pixel (R,G,B, Intensity)
  - For Analog RGB, N bits per R,G,B value where each RGB value can have 2^N distinct values
- The memory that defines the screen contents is called the display memory or frame buffer.

4/23/01 12

Memory Calculations

- 800x 600 x 24 bit color – how much memory?

  800 x 600 x 24 bits/pixel = 11520000 bits

  = 1440000 bytes = 1406 Kbytes = 1.4 Mbytes

2 Mb video card fine for this resolution/color

1280 x 1024 x 24 bits = 3.75 Mbytes

Need 4 Mbyte Video card

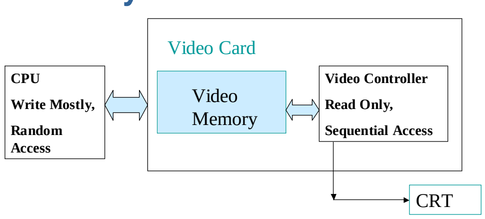
4/23/01 13

Display Memory Characteristics

- Accessed at high data rates
- Will need to accessed by two sources
  - CPU which will be doing read/writes to random locations
  - Video signal driver which will be reading memory locations in a fixed pattern (the scan pattern)
- Can use either SRAM, DRAM, SDRAM or specialty graphics memory to implement the graphics memory.
- Graphics memory usually on same board as rest of video logic

4/23/01 14

Memory Access



Video controller must arbitrate between its accesses and CPU accesses. CPU connection to card can either be via system bus (I.e. PCI bus) or dedicated connection (Advanced Graphics Port).

4/23/01 15

What type of Memory for Video?

- Early cards used SRAM because of simple interface (no refresh, no DRAM controller needed), and because of low memory size requirements (a few Kbytes)
- As colors/video memory size increased, began using DRAM for density/cost reasons
  - Some special DRAMs appeared (called Video DRAMs) that had dual port access – parallel input/output for CPU, serial output for CRT. Died out because of cost reasons.
- Memory choice today is Synchronous DRAM since 3-D graphics cards need 16 Mbytes and up
  - Some 3D cards performance is limited by speed of DRAM – even DDR-DRAM not fast enough.

4/23/01 16

2D Graphics

- 2D graphics operations are very simple
  - 2D graphics operations required by 'window' type Graphic User Interfaces
- Block moves – memory to memory copy of rectangular areas on screen
  - Implements window movement
- Block Fills (fill rectangular area on screen)
- Hardware support for cursor operation
  - As cursor moves over screen pixels under cursor are not damaged
- Hardware support for Sprite movement/collision detection
  - Sprite is rectangular group of pixels representing an object
  - Used by 2-D games

4/23/01 17

Animation

- To produce animation on a screen, successive 'frames' are displayed in which screen objects are in slightly different positions in each frame.
- The area of memory that the video controller is currently displaying on screen is the *active Frame Buffer.*
  - Most cards support two frame buffers. The CPU writes to the 2nd frame buffer to prepare the next frame.
  - When the next frame is ready, the 2nd frame buffer becomes the active Frame buffer and the 1st frame buffer is used to create the next frame.
- The rate at which new frames are produced is called the *Frame Rate* (not the same as Video refresh rate).

4/23/01 18

3D graphics

- 3D graphics has higher requirements than 2D graphics in terms of:
  - More memory bandwidth needed
  - More memory needed
  - More computations needed
  - More of everything needed

4/23/01 19

Creating a Frame for a 3-D Scene

A frame in a 3D world consists of 3D objects viewed from a particular camera angle and lit with one or more light sources.

The 3D objects are composed of multiple polygons; usually triangles.

A 3D object is usually covered with a *texture*, which is a 2D bitmap that is mapped onto the 3D object. An example of a texture is a 'wallpaper' mapped onto a 3D wall.

Lighting in the scene will affect how the pixels appear in the final scene.

4/23/01 20

Creating a Frame for a 3-D Scene (cont).

1. Transform (scale, rotate, translate) objects to viewing angle. Involves floating point calculations in terms of matrix multiplies. Done by CPU usually. Some lighting is also done here.
2. Clip objects to the 3D Viewport size – all polygon vertices now have 3 coordinates : X, Y, and Z (Z is depth of object from viewer or distance of object from viewer). Color information is also attached to each object.
3. *Rasterize* each object in scene. This coverts each object to a 2D flat representation of texels. There are many steps to rasterization.

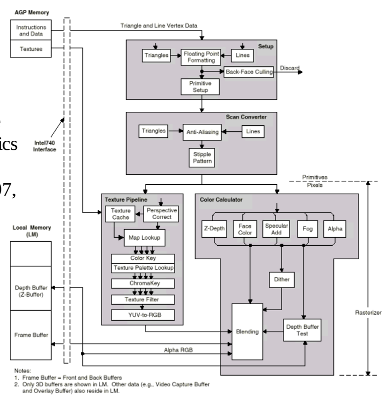
4/23/01 21

Rasterization (done by 3D card)

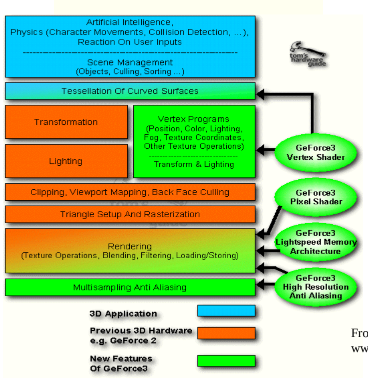
1. EACH object is rasterized, even if one object is in front or behind other objects. The object becomes a list of dots (texels), each dot has an X, Y, Z screen coordinate and R,G,B color values.
2. The 'Z' coordinate determines which texels obscure other texels.
3. Transparency effects may combine texels of one object with another interesting ways.
4. Textures are mapped onto the object and will effect the final R, G, B values of the texel.
5. Some lighting can also be done at this stage to effect the R,G,B value of each texel.
6. Once all objects in the scene are rasterized, the final frame buffer values are the 'pixels' seen on the screen.

4/23/01 22

Hardware 3D graphics pipeline (circa 1997, Intel 3D)



4/23/01 23



Hardware 3D graphics pipeline (2001)

From: www.tomshardware.com

4/23/01 24

Texture Mapping

Another texture

Object

Object with texture mapped on it

Multi Texturing means to apply more than one texture to the same object.

Newer graphics chips can do this in one pass, older chips require multiple passes.

4/23/01 25

3D Graphics Main Memory Access

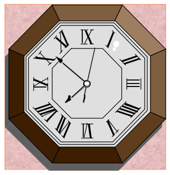
- The frame buffer and extra memory (for multiple frame buffers, texture bitmaps) resides on the 3D graphics card
- Not all texture information can be held at one time on Video Card, textures are swapped between 3D card and main memory
  - Needs to be done FAST so that pauses are not caused when creating scenes that require new textures
- Object lists are stored in main memory and sent to 3D card for rasterization.

4/23/01 26

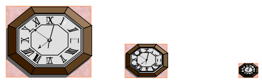
How big is a texture map?

- Texture maps vary in size, need a texture map for each object
- Limited to rectangular sizes, power of 2 (16 x 16, 32 x 32, 64 x 64, etc)
- A 64 x 64 map has 2^6 x 2^6 = 2^{12} (4096) texels, each texel has 8 bits for R,G,B. So a total of 3 x 4K = 12 K bytes
- There may exist multiple variations of a texture map for use at different viewing distances
  - More detailed, larger closer up
  - Less detailed, smaller further away
  - Known as mip-mapping

4/23/01 27



Mip Mapping – applying different versions of a texture map to an object based upon distance.



U x V, LOD = 0

U/2 x V/2, LOD = 1

U/4 x V/4, LOD = 2

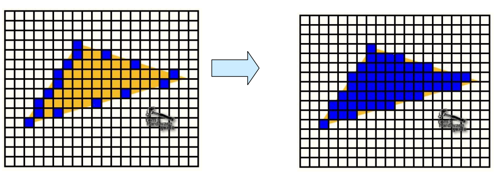
U/8 x V/8, LOD = 3

Width (u), Height (v) must be powers of 2. From first texture map (2^M x 2^N), OpenGL constructs other succeeding texture maps until a texture map is reached in which one dimension equals 2^0 (1)

Example 32 x 16, 16 x 8, 8 x 4, 4 x 2, 2 x 1

4/23/01 28

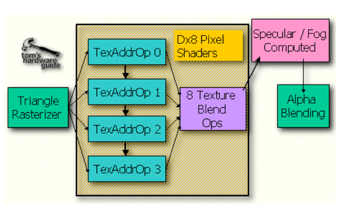
Rasterization – transforming a triangle to pixels



From: www.tomshardware.com

4/23/01 29

Geforce3 - Can apply 4 different textures per pass.



From: www.tomshardware.com

4/23/01 30

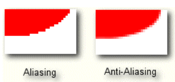
Bump Mapping – Advanced Texturing

Geforce3 supports reflective bump mapping in hardware pipeline.

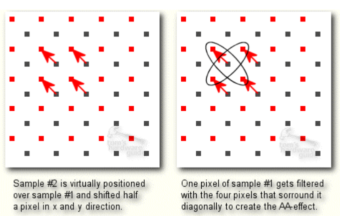
From: www.tomshardware.com

Anti-Aliasing – get rid of line jaggedness

Need to smooth an edge by averaging pixel values in a neighborhood.



Geforce3 averages 5 pixels to get one pixel.



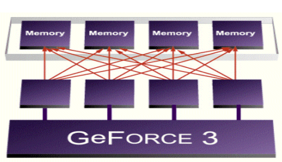
From: www.tomshardware.com

Memory Bandwidth

3D controllers have kept increasing the width of the memory interface to increase the bandwidth.

Geforce2: 256 bit wide memory path (32 bytes).
Problem: Need more random access for high performance (not all bytes in a 32-byte block needed).

Geforce3: Crossbar switch to four 64-bit memory blocks.



From: www.tomshardware.com