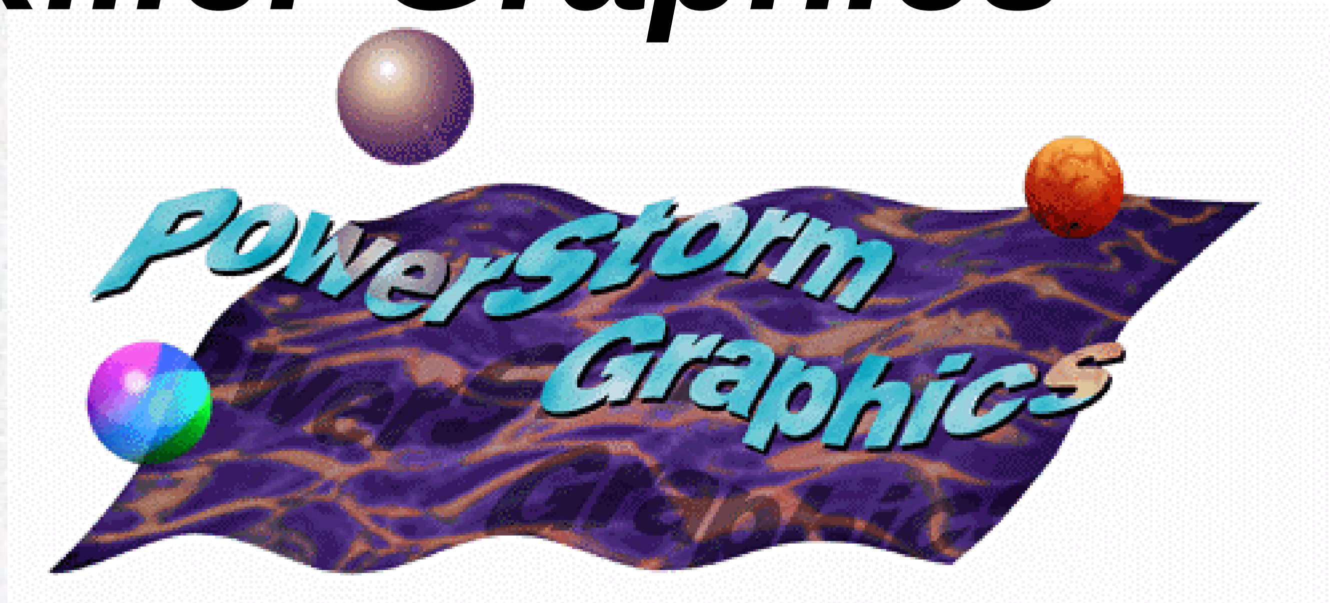
digital™

Killer Graphics



"Graphics 201 Continued"

PowerStorm Graphics

PowerStorm Advantage

- Leverage Alpha processor for highest performance and algorithmic flexibility on:
  - Compute Intensive operations
  - Data intensive operations
  - Complex algorithms
- Leverage Graphics hardware for:
  - Pixel level operations

digital™

Alpha Processor

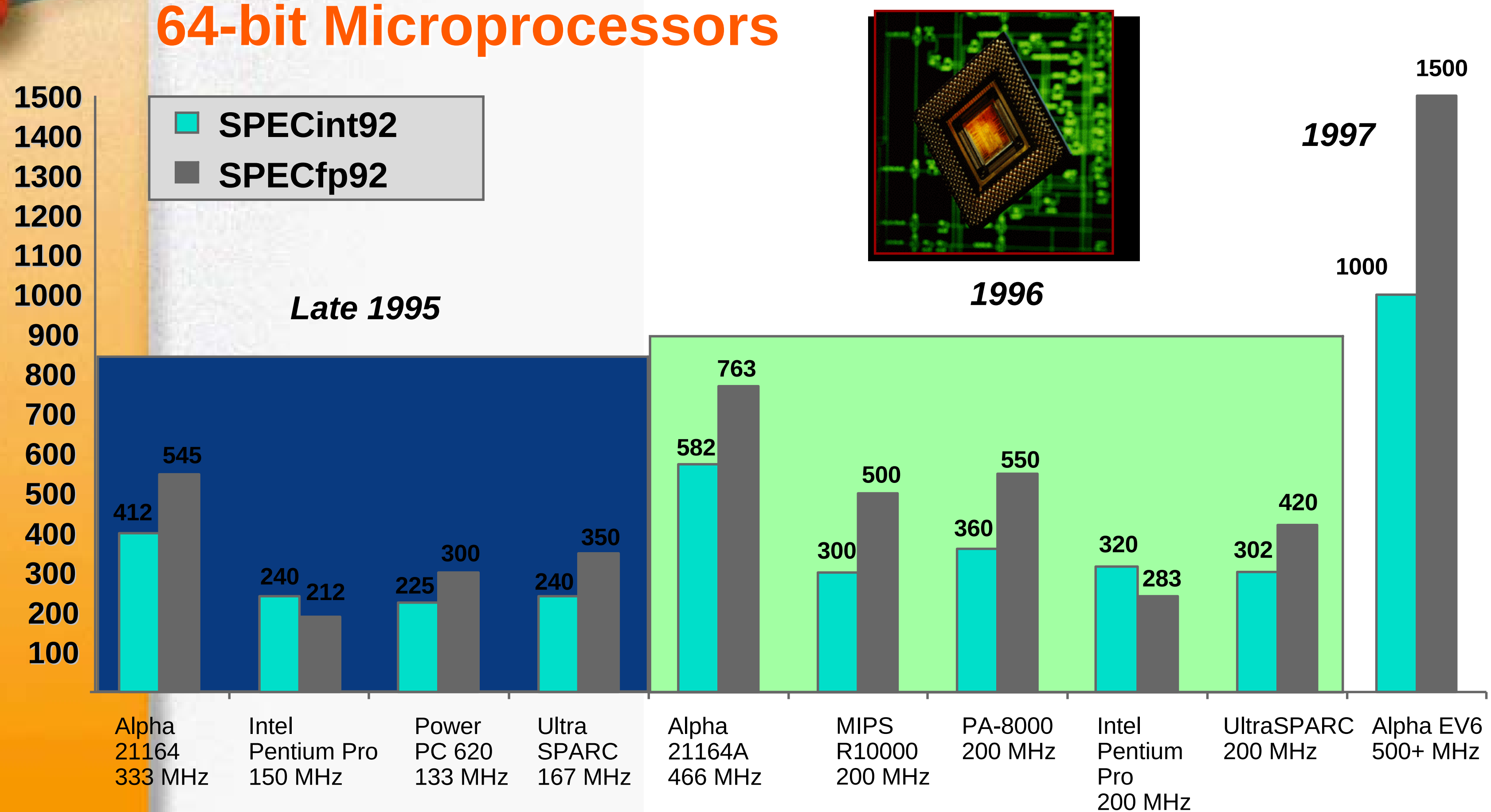
Designed for Performance - Tuned for Graphics and Multimedia

- **Highest Performance CPU**
  - Industry dominating floating point!
- **Alpha Instructions Tuned for Graphics and Multimedia**
  - Ultra fast L1 cache and large on-chip L2 cache contain a full graphics kernal and large graphics data sets -- complex graphics operations run at full speed
  - Data prefetching enables large graphics data structures to be manipulated with minimal latency
- **Balanced, Scalable systems**
  - Scalable CPU and graphics performance
  - Deliver maximum performance through applications, consistently

digital™

The Alpha Advantage

64-bit Microprocessors



Alpha Powered Graphics Transformation

- Quad issue
- Overlapped load/store
- Parallel multiply/ add
- Full update of transform matrix each cycle
- Full transform in 13 clock cycles!
  - 12 multiply
  - 12 add
  - 19 load/store
  - conditional branch
- Sustained transform rate of 38M vertices/sec

digital™

Loop:				
1	FM: zm02 = z * m02 Floating Point Multiply z by m02 and save as zm02	FA: x' = x' + ym01 Floating Point Add ym01 to x' and save as x'	I1: lds m00 load matrix value m00	I2: pIn++ increment input data structure pointer
2	FM: zm12 = z * m12 Floating Point Multiply z by m12 and save as zm12	FA: y' = y' + ym11 Floating Point Add ym11 to y' and save as y'	I1: lds m10 load matrix value m10	I2: lds x load input x coordinate vertex i
3	FM: zm22 = z * m22 Floating Point Multiply z by m22 and save as zm22	FA: z' = z' + ym21 Floating Point Add ym21 to z' and save as z'	I1: lds m20 load matrix value m20	I2: lds m21 load matrix value m21
4	FM: zm32 = z * m32 Floating Point Multiply z by m32 and save as zm32	FA: w' = w' + ym31 Floating Point Add ym31 to w' and save as w'	I1: lds m30 load matrix value m30	I2: lds y load input y coordinate vertex i
5	FM: xm00 = x * m00 begin next transformation Floating Point Multiply x by m00 and save as xm00	FA: x' = x' + zm02 Floating Point Add zm02 to x' and save as x'	I1: lds m01 load matrix value m01	I2: lds m02 load matrix value m02
6	FM: xm10 = x * m10 Floating Point Multiply x by m10 and save as xm10	FA: y' = y' + zm12 Floating Point Add zm12 to y' and save as y'	I1: lds m11 load matrix value m11	I2: lds z load input z coordinate vertex i
7	FM: xm20 = x * m20 Floating Point Multiply x by m20 and save as xm20	FA: z' = z' + zm22 Floating Point Add zm22 to y' and save as y'	I1: lds m12 load matrix value m12	I2: lds m22 load matrix value m22
8	FM: xm30 = x * m30 Floating Point Multiply x by m30 and save as xm30	FA: w' = w' + zm32 Floating Point Add zm32 to w' and save as w'	I1: lds m31 load matrix value m31	I2: lds m32 load matrix value m32
9	FM: ym01 = y * m01 Floating Point Multiply y by m01 and save as ym01	FA: x' = xm00 + m03 Floating Point Add xm00 to m03 and save as x'	I1: sts x' store transformed x coordinate vertex i-1	I2: n-- decrement loop counter
10	FM: ym11 = y * m11 Floating Point Multiply y by m11 and save as ym11	FA: y' = xm10 + m13 Floating Point Add xm10 to m13 and save as y'	I1: sts y' store transformed y coordinate vertex i-1	I2: nop
11	FM: ym21 = y * m21 Floating Point Multiply y by m21 and save as ym21	FA: z' = xm20 + m23 Floating Point Add xm20 to m23 and save as z'	I1: sts z' store transformed z coordinate vertex i-1	I2: nop
12	FM: ym31 = y * m31 Floating Point Multiply y by m31 and save as ym31	FA: w' = xm30 + m33 Floating Point Add xm30 to m33 and save as w'	I1: sts w' store transformed w coordinate vertex i-1	I2: nop
13	FM: nop	FM: nop	I1 pOut++ increment output data structure pointer	I2: bgt n, Loop conditional branch; go to Loop: or end loop

digital™

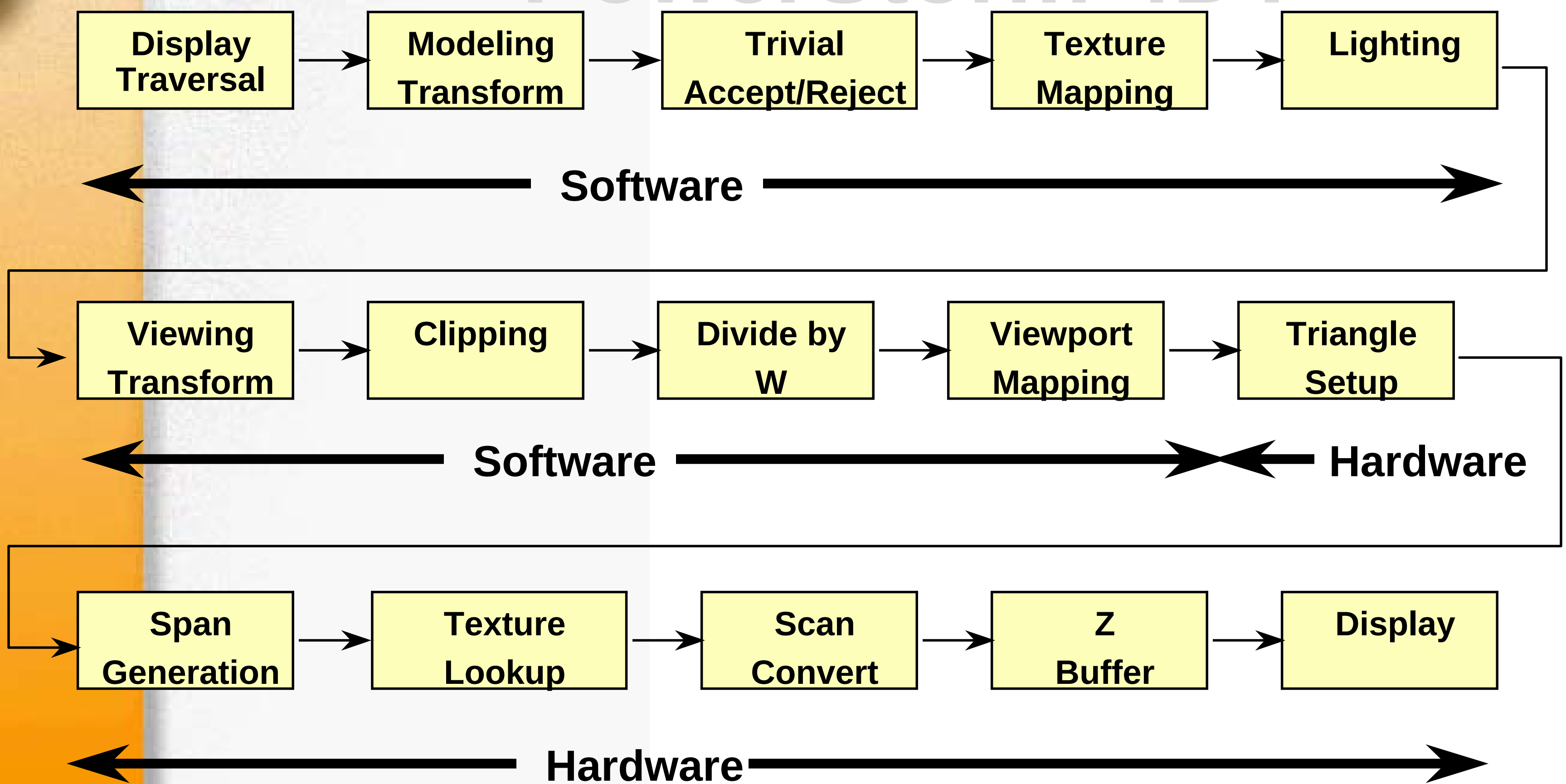
PowerStorm 4D40T, 4D50T, 4D60T

They Doesn't Just Run OpenGL... They Were Built for it!

- **Leadership price performance with advanced graphics**
- **Advanced high performance shading and texture mapping supporting large texture sets**
- **High performance 3D starting under $4,000!**
- **Lowest entry price texture mapping -- under $4,600!**

digital™

Graphics Pipeline on PowerStorm 4DT



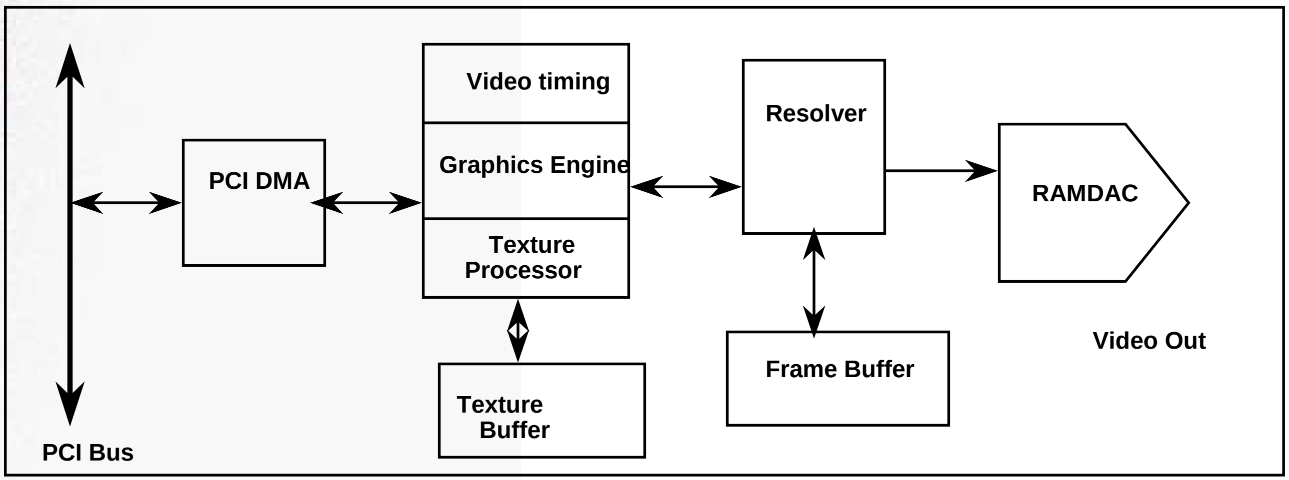
digital™

PowerStorm 4D40T, 4D50T, 4D60T

- Common architecture
- Common graphics engine and module design
- Advanced technology for advanced performance:
  - State-of-the-art graphics ASICS implemented in .5 micron CMOS
  - BGA and CGA packaging
  - SDRAM memory
- User installable texture memory options

PowerStorm 4DT Technology

- Single PCI board (PCI daughtercard for VGA and power)
- High density, ultra fast graphics ASICs implemented in .5 micron CMOS
- Frame buffer implemented with fast SDRAM
- Texture memory implemented with fast SDRAM



digital™

Graphics Engine

- Receives graphics requests from host via PCI bus
- Performs line and triangle setup
- Interpolates triangle edges and calculates span start/stop values (screen coordinate and RGBA)
- Performs texture operations (per pixel)
  - performs texture lookup, including MIPmap
  - performs bi-linear or tri-linear interpolation
  - sends RGBA texture values to Resolver
- Video timing function synchronizes double buffer swap with monitor vertical refresh

digital™

Texture Memory

- Texture mapping implemented in Graphics Engine
- Texture memory is optional
  - Two field installable memory modules
  - DIMM form factor (different design)
  - Available in 4MB, 16MB or 32MB
- Texture memory required for hardware texture mapping
- Supports OpenGL Texture Object Extension
- Automatic fallback to OpenGL software texture mapping
- SDRAM memory

Resolver

- System can hold either two or four Resolver Chips
- Each chip has three main modules
  - Two resolver modules
  - One Screen Refresh module
- Resolver module is responsible for
  - Translating span requests into manipulations of the frame buffer
  - Masking
  - Alpha tests
  - Z Buffering
  - Stencil test
  - Frame buffer merge operations
  - Double buffering

Frame Buffer

- Displayed Planes
  - Image — 24-bit RGB, double-buffered
  - Image Context — 4-bit, double-buffered
  - Fast Clear — 1-bit, double-buffered
  - Select Buffer Image — 1-bit, single-buffered
  - Overlay — 8 or 4-bit, double-buffered
  - Select Buffer Overlay — 1-bit, double-buffered
- Construction Planes
  - Alpha — 8-bit, single-buffered
  - Stencil — 6 or 8-bit, single-buffered
  - Z — 24 or 32-bit, single-buffered
  - Mask — 2 or 4-bit single-buffered
- Implemented in SDRAM

RAMDAC

- Drives monitor
- Converts digital RGB values to analog RGB plus horizontal and vertical sync
- Uses 10-bit RGB, providing for gamma correction
- Provides 4 color lookup tables (palettes) for 8-bit color
- Provides 64x64 hardware cursor

PowerStorm 4DT Family

- PowerStorm 4D40T
  - Entry level, 16 MB video memory, dual rendering ASICs
- PowerStorm 4D50T
  - Faster graphics clock, 16 MB video memory, dual rendering ASICs
- PowerStorm 4D60T
  - Fastest graphics clock, 32 MB video memory, quad rendering ASICs

Common Characteristics

- Balanced performance -- high 2D and 3D performance
- 24-bit double buffered true color display
- Z-buffer
- Alpha blending
- Anti-aliased vectors
- 1280x1024 resolution (1600x1200 on PowerStorm 4D60T)
  - 75HZ refresh rate
- Hardware shading (Gouraud shading)
- Hardware texture mapping
  - Requires optional texture memory (custom memory)
- 4 or 8 plane double-buffered overlay
- 16 MB or 32 MB Video memory (plus 0-32 MB of texture memory)

PowerStorm 4DT Specifications

Product	4D40T	4D50T	4D60T
Max. Resolution	1280x1024	1280x1024	1600x1200
Refresh rate	76 Hz	76 Hz	76 Hz
Video RAM	16 MB	16 MB	32 MB
Texture Memory	0,4,16,32 MB	0,4,16,32 MB	0,4,16,32 MB
Color Planes	24-bit DB	24-bit DB	24-bit DB
Overlay Planes	4-bit DB	4-bit DB	8-bit DB
Z-Buffer Planes	24-bit	24-bit	32-bit
Alpha planes	6-bit	6-bit	8-bit
Stencil planes	6-bit	6-bit	8-bit
Total bits per pixel	**100**	**100**	**128**
Performance			
10-pixel vectors	1.1M	2.1M	2.3M
25-pixel triangles	500K	1M	1M
Texture fill rate	14Mpixels	30Mpixels	30Mpixels

Performance Notes

- PowerStorm Graphics (including 4D40T, 4D50T, 4D60T) rely on Alpha processor for geometry processing
  - Performance scales with increasing CPU performance -- until you hit the limits of the graphics hardware
  - Previous Digital graphics hardware has been the limiting factor
  - With 4D50T and 4D60T, the fastest current Alpha processors -- 500 MHz -- *do not drive them at their limits!*
- For best graphics performance, use PowerStorm 4D50T or 4D60T and the *fastest processor possible*

digital™

4DT PerformanceCharacteristics

- Easy to achieve performance!
- Direct rendering
- Short render pipe
- Per-pixel Window ID
  - Doesn't care about partially occluded windows
- True color, double-buffered, with alpha
- Overlay planes (double buffered)
- Works with FX!32
- Optimized OpenGL
  - Applications analyzed
  - Critical OpenGL functions optimized (assembly code)

digital™

Performance Guidelines

- PowerStorm 4D40T
  - Alpha processors under 300 MHz and all EV4 processors; All Intel processors
- PowerStorm 4D50T
  - Alpha processors above 300 MHz
- PowerStorm 4D60T
  - Alpha processors above 300 MHz
  - Where high resolution is required
  - Where 8-bit overlays are required
  - Where the utmost graphics performance is required (especially with large polygons)

Generating High Visual Quality

- Antialiasing polygons requires computing pixel color based on information about where that polygon resides within the depth of the scene (its Z-value)
- There are four primary methods for doing this:

•Alpha Blending	Requires time-consuming scene sorting
•Multi-Sample Buffer	Single-pass but poor transparency
•Accumulation Buffer	Multi-pass
•DIGITAL Hi-FIVE	Single-pass, no sorting, good transparency

Accumulation Buffer Method

- Minimizes memory needs
- Maximizes rendering time
- Basic Algorithm:
  - Z Sort all polygons in the scene from largest (furthest) to smallest (closest) if using transparency
  - *Iteratively* render all polygons into the accumulation buffer
    - min. 2 - 7 times for low-fidelity realism (jaggies)
    - 8 - 16 times for high-fidelity realism
  - Divide each color value at each pixel by the number of iterations required
- Inconsistent image realism depending on the number of iterations
- Time consuming (16 passes) and high-fidelity realism *-or-* Fast and low-fidelity realism

RRQ

- Recap
- Review
- Questions

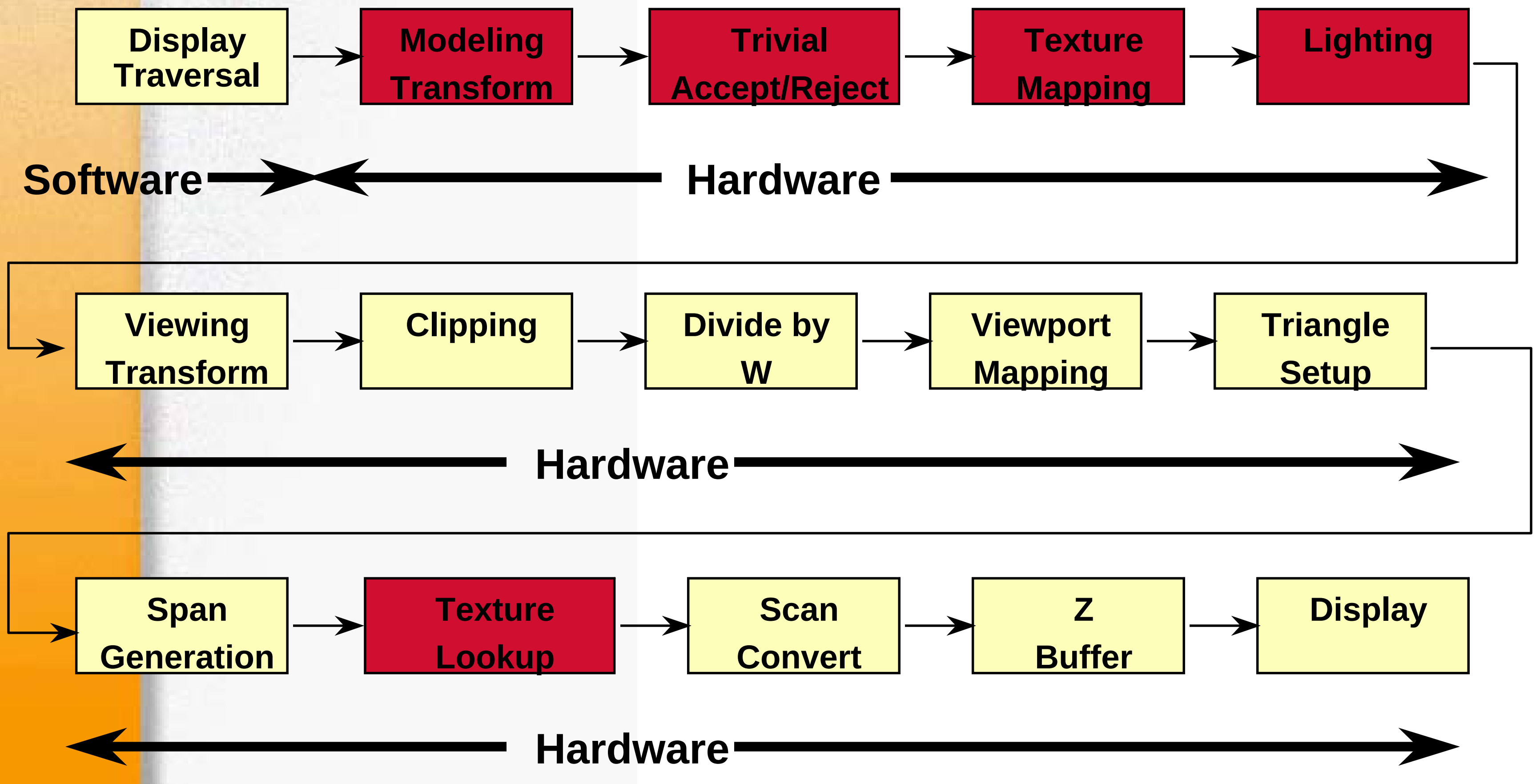
Graphics Architectures

Graphics Architectures

- Several different graphics hardware/software architectures possible
- All graphics architectures represent tradeoffs
- Digital's PowerStorm architecture deliberately designed
- Key elements are to understand the graphics pipeline, OpenGL, and hardware vs. software implementations
- Key issues:
  - Geometry Accelerators
  - Graphics interface bandwidth

digital™

Graphics Pipeline: Geometry and Rendering Acceleration



digital™

Geometry Accelerators

- Hardware geometry acceleration trades performance for flexibility
- Hardware geometry acceleration difficult to do effectively
  - Lighting models -- Application constructs
  - Data structure design -- Texture mapping
- Hardware geometry acceleration most effective with display list systems
- Hardware geometry acceleration easiest with simple shading
- Hardware geometry acceleration difficult to move to different graphics API (eg: Starbase to OpenGL)

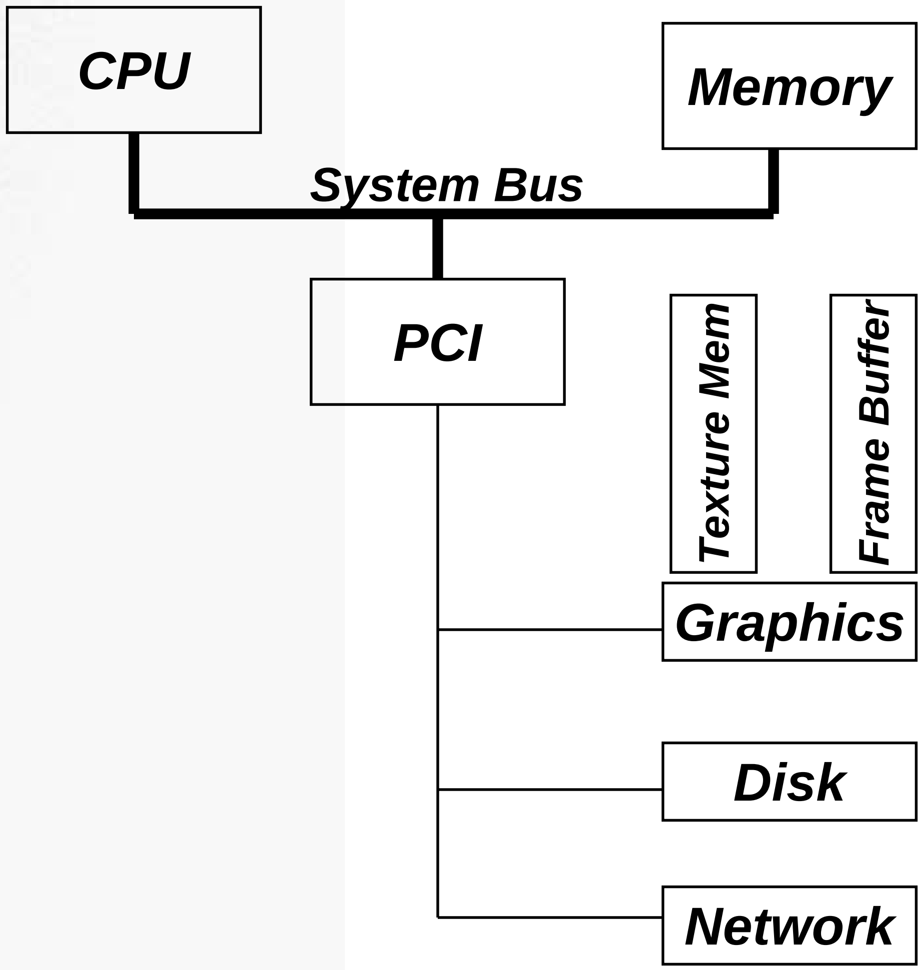
Graphics Performance

- Graphics performance determined by *slowest component*
  - Can host processor generate graphics commands quickly enough?
  - Can graphics interface bus transfer graphics commands quickly enough?
  - Can graphics hardware process graphics commands quickly enough?
- Balanced systems design is critical

System Architectures

- Digital PowerStorm
- SGI O2
- SGI Octane

Digital Architecture



digital™

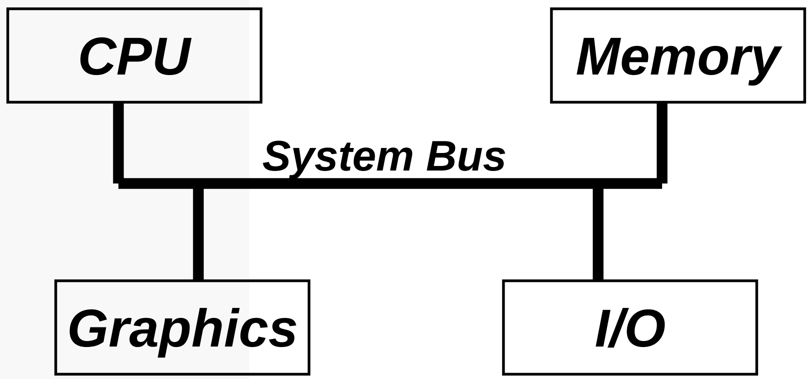
Digital Notes

- Texture Memory: 1.2 GB/sec sustained delivered bandwidth (short packets) (PowerStorm 4D60T)
- Frame Buffer: >500 MB/sec sustained (read/modify/write) (short packets) (PowerStorm 4D60T)
- System Memory/System Bus : >500 MB/sec (platform dependent)
- Concurrent!

digital™

SGI O2

"Unified Memory Architecture"



Main memory shared by:

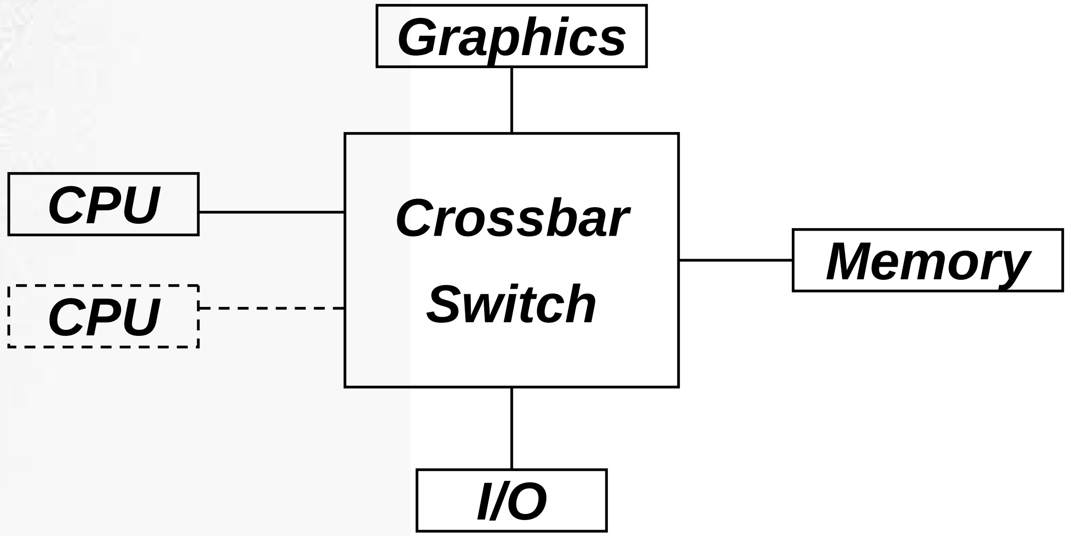
- ***CPU***
- ***Frame Buffer***
- ***Z-buffer***
- ***Texture Memory***

digital™

O2 Notes

- Memory conflicts between CPU, Graphics and Texture
- Base systems don't have enough memory
- System is under-powered (R5000)
- Can't take advantage of R10000 -- only 15% speedup
- System is ***slow***

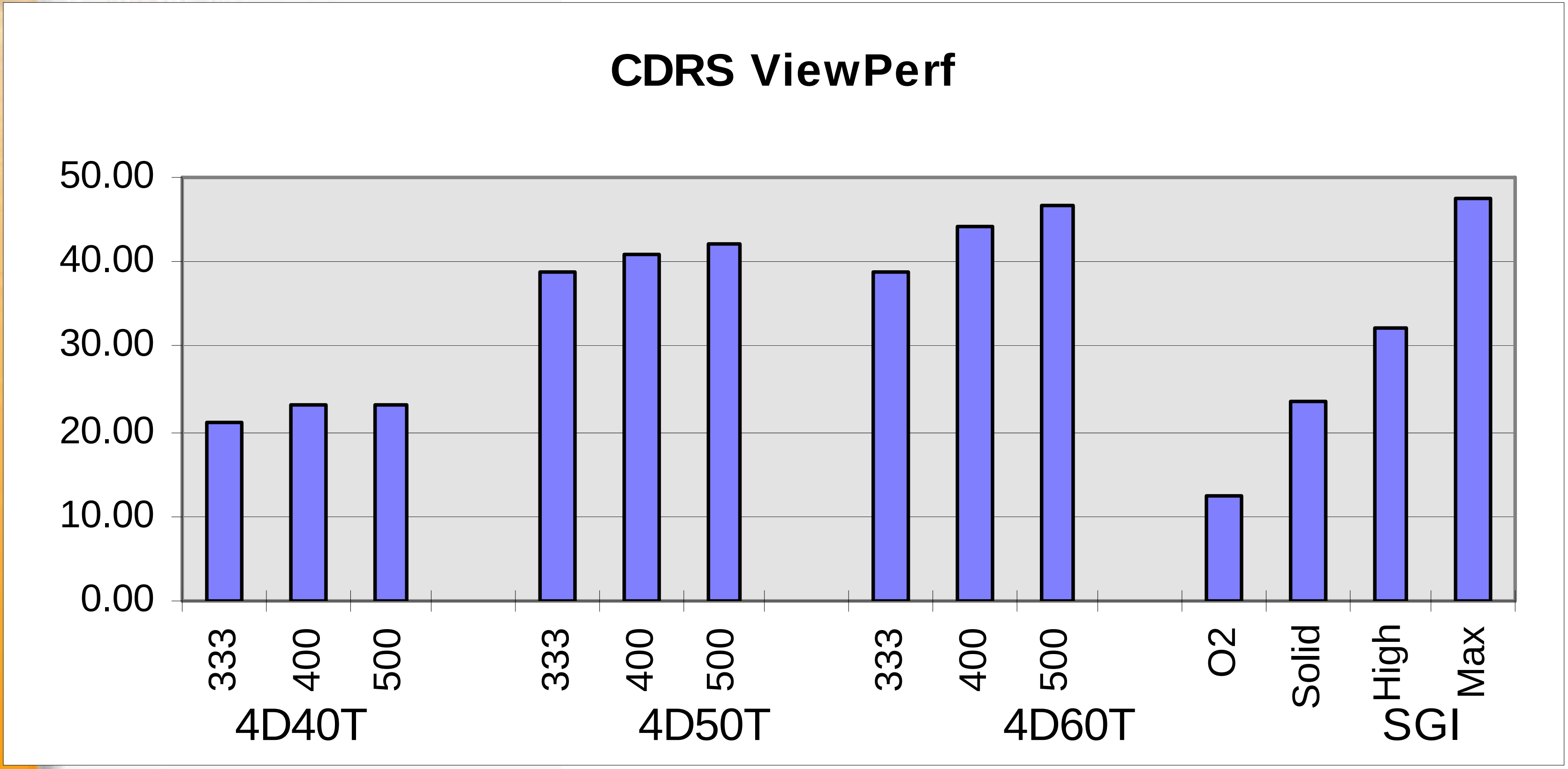
SGI Octane
“Crossbar Switch”



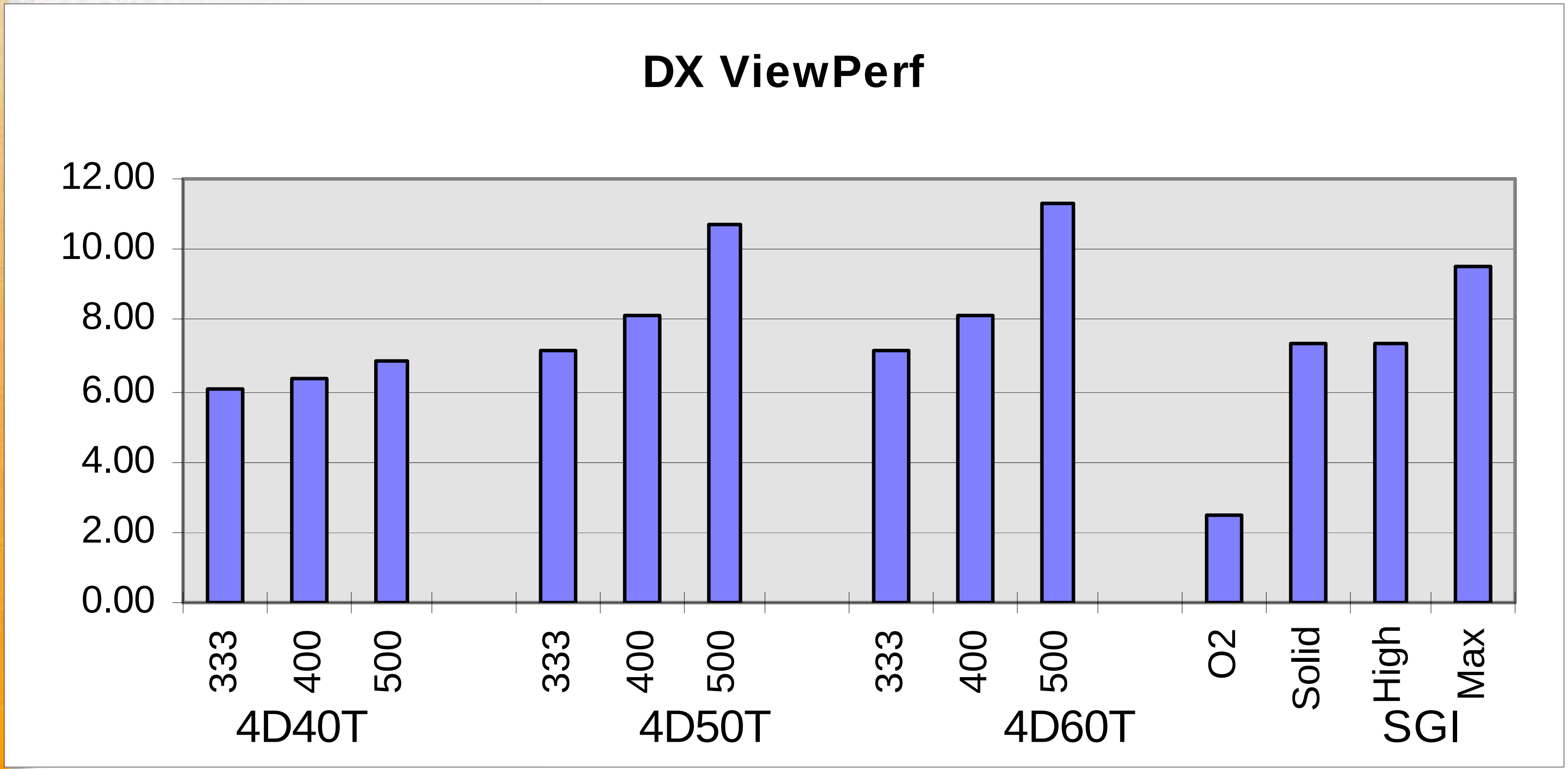
Octane Notes

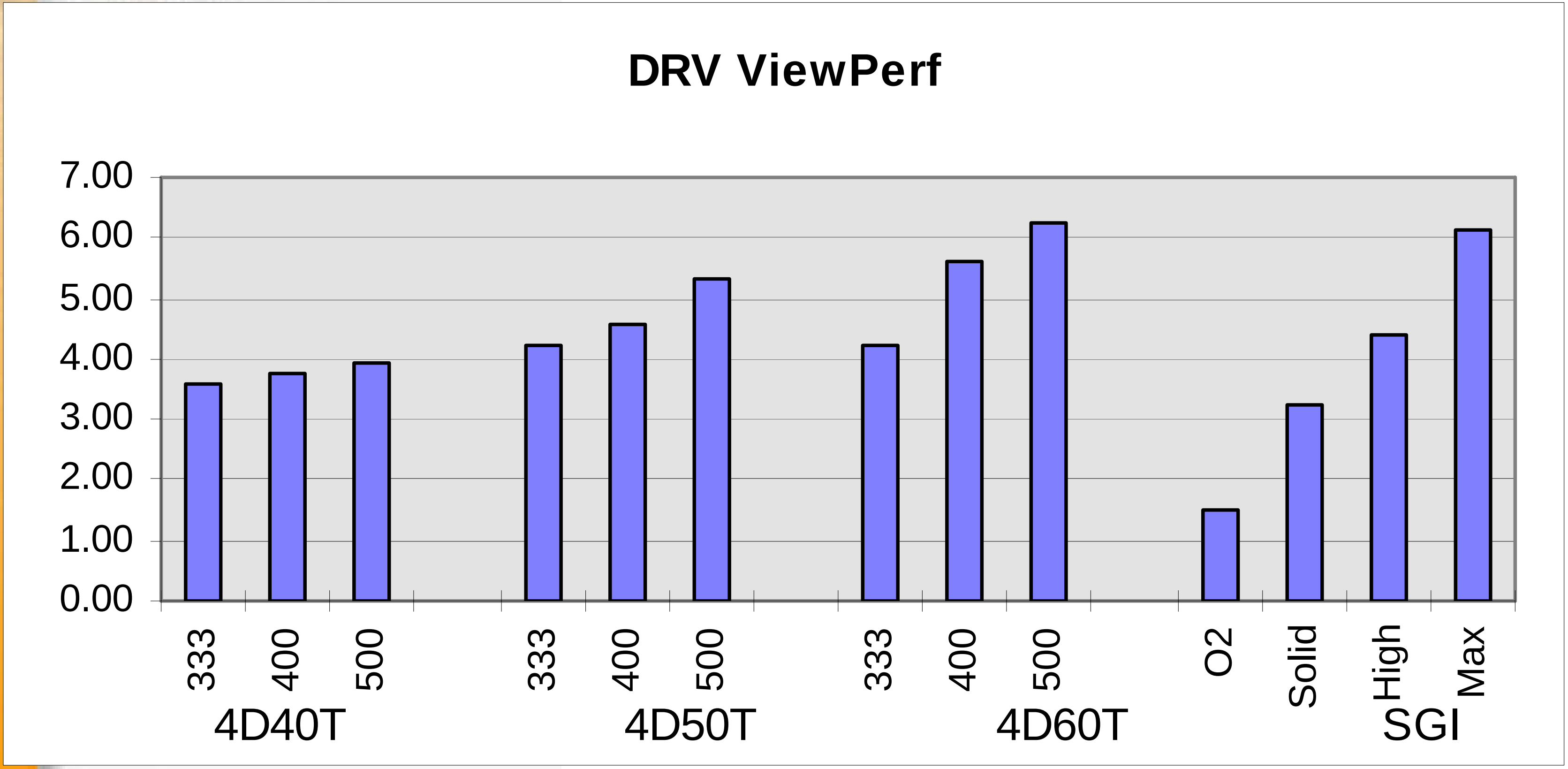
- Crossbar switches improve system throughput -- if you have enough resources *"things hanging off the switch"*
  - –Good architecture for large servers
  - –Octane doesn't really benefit
- Crossbar switches tend to increase latency
  - –SGI doesn't publish any latency info...

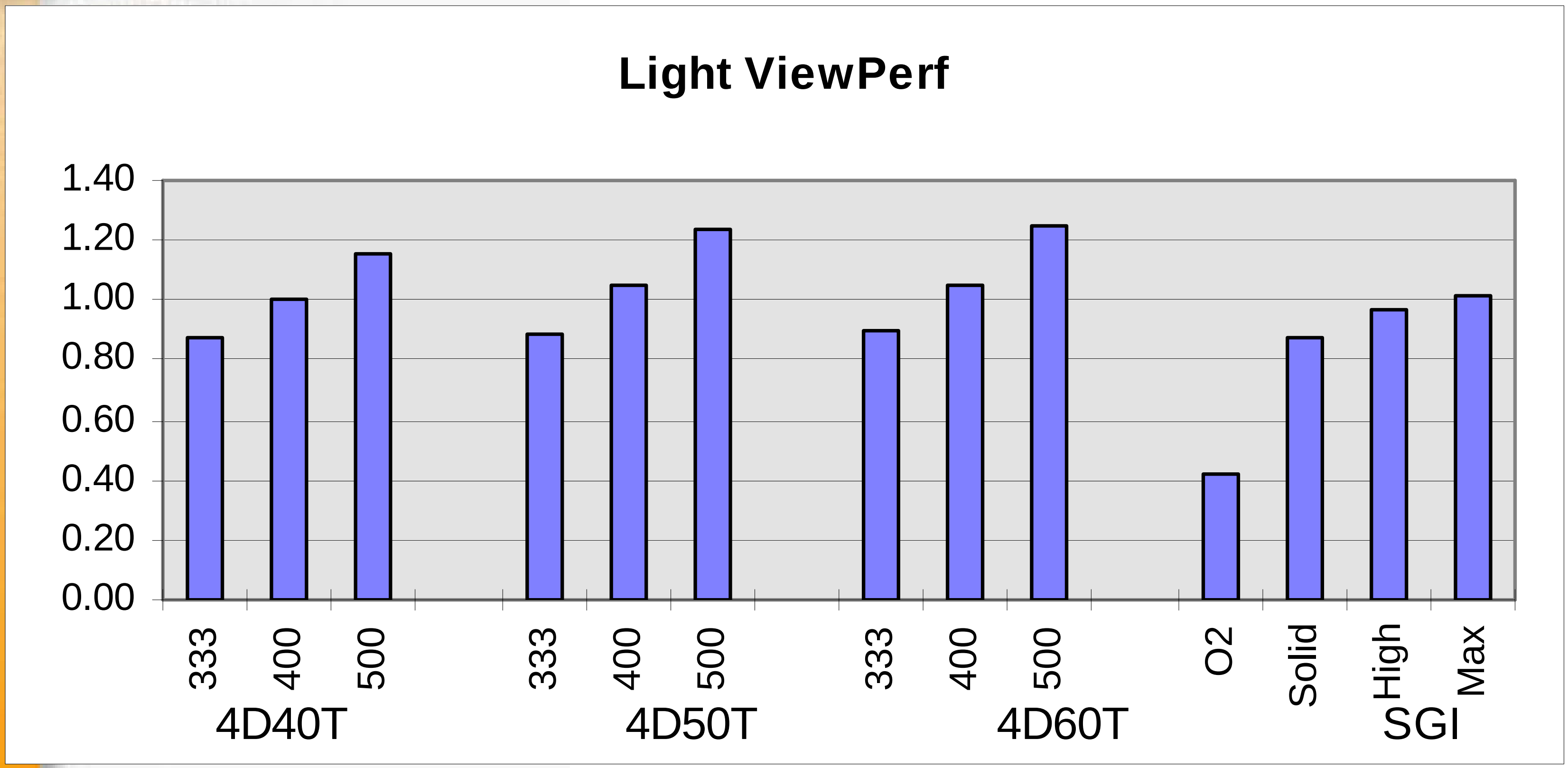
Graphics Performance Benchmarks



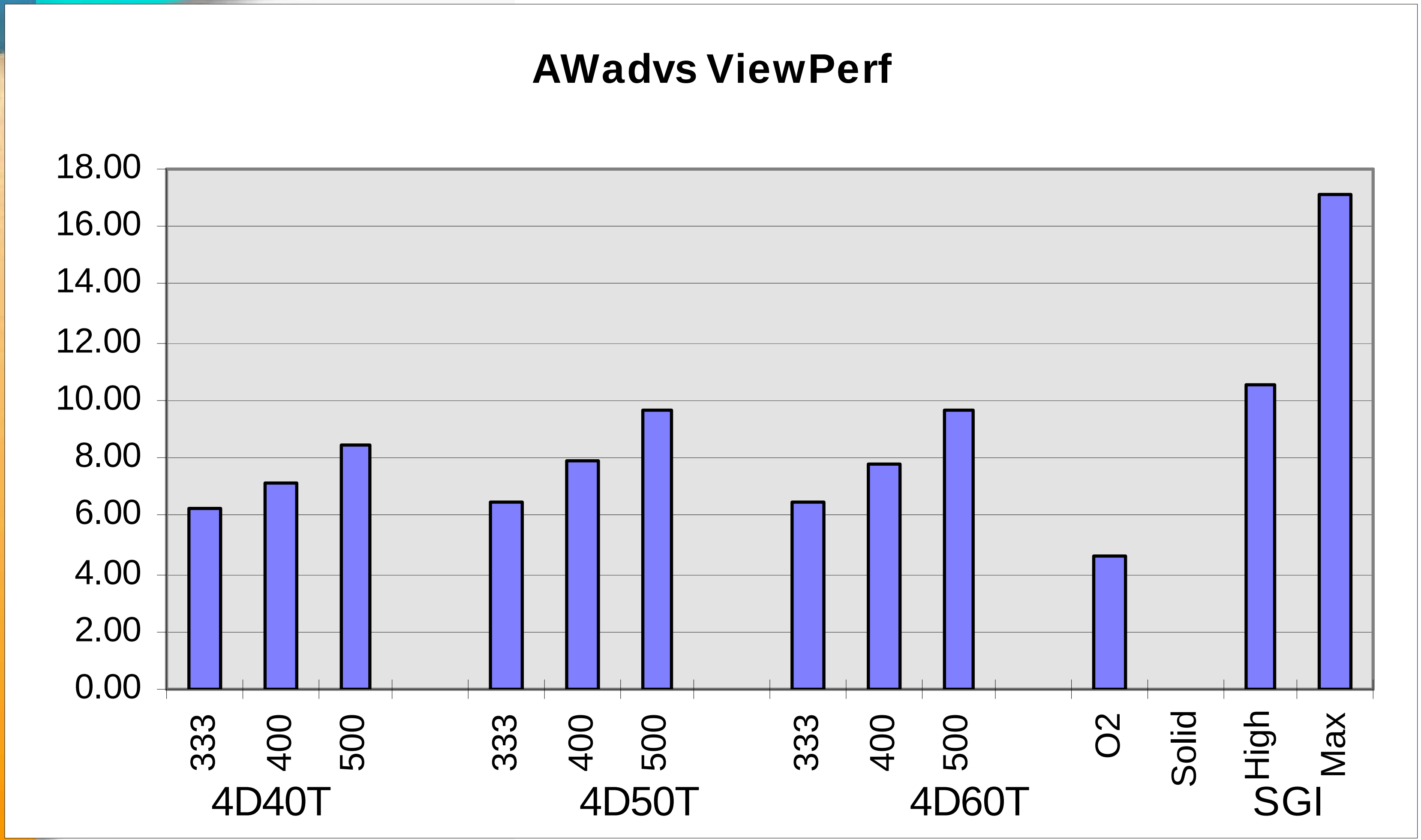
CDRS ViewPerf
50.00
40.00
30.00
20.00
10.00
0.00
333
400
500
4D40T
333
400
500
4D50T
333
400
500
4D60T
O2
Solid
High
Max
SGI


DX ViewPerf
12.00
10.00
8.00
6.00
4.00
2.00
0.00
333
400
500
333
400
500
333
400
500
O2
Solid
High
Max
4D40T
4D50T
4D60T
SGI


DRV ViewPerf
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
333
400
500
4D40T
333
400
500
4D50T
333
400
500
4D60T
O2
Solid
High
Max
SGI



Light ViewPerf
1.40
1.20
1.00
0.80
0.60
0.40
0.20
0.00
333
400
500
4D40T
333
400
500
4D50T
333
400
500
4D60T
O2
Solid
High
Max
SGI


AWadvs ViewPerf
18.00
16.00
14.00
12.00
10.00
8.00
6.00
4.00
2.00
0.00
333
400
500
4D40T
333
400
500
4D50T
333
400
500
4D60T
O2
Solid
High
Max
SGI

Short Subjects

digital™

Multi-Head

- Available today on UNIX
- Was Available on NT 3.51
- Microsoft broke on NT4.0
- We are working on Multihead for summer
  - –Homogeneous graphics hardware

Stereo Imaging

- True 3D display
  - –Separate images for left/right eyes
  - –Actual depth
- Full Screen Stereo
- OpenGL Stereo
  - –Stereo in a Window
  - –Quad-buffered (left/right/front/back)

Stereo on PowerStorm 4DT

- OpenGL Stereo, Quad-buffered
  - –True Color!
  - –Textures
- 4D40T/4D50T: 800x600
- 4D60T: 1280x1024

Stereo

- Available for UNIX in Open3D 4.2
- Coming for NT in GSSS 4.3 (summer)
- Verified with CrystalEyes glasses from StereoGraphics
  – StereoGraphics now ships all cables with Digital packages (different connectors)

Imaging

- 2D Raster Data
- Convulations and Filtering

digital™

Animation

- Creation of a series of frames
- Heavily used in TV and Movies
- Phases:
  - –Modeling
  - –Rendering
  - –Compositing
  - –Post Production & Video Editing

Modeling

- Creation of 3D models, textures, motion paths, deformations, lighting
  - –Artistic rather than technical focus
- Interactive and graphics intensive
  - –Requires CPU, graphics, memory, disk
- Applications such as SoftImage, LightWave 3D, 3D Studio Max, Alias
- Productivity vital factor

Motion

- *KEY* part of animation!
- Define:
  - –Start point/end point
  - –Start time/end time
  - –Motion path (including orientation)
  - –Velocity, acceleration, location, time, transitions along path
  - –Object deformations

Kinematics

- "Collection of rigid bodies connected by joints"
  - –Joints: pivot, ball, slider, etc.
- Possible motions constrained
- "Skeleton" or "Bones"
- Inverse Kinematics
  - –From final motion, derive motion of each component body/joint

Motion Capture

- **Lifelike motion extremely complex and difficult to produce**
- **Solution: Instrument a person and record motion!**
  - **Movement of joints**
- **Match *human joint* to *computer model joint* and apply motions**
- **Result: *Lifelike Motion!***

digital™

Particle Systems

- Used for smoke, bubbles, leaves, tornados, snow, etc.
- *Programmatic* and changes over time
- Define:
  - –Starting characteristics
  - –Environment
  - –Particle behavior and life
- Merged with other graphics through compositing

Rendering

- Takes the input data (models, lights, textures, motion paths)
- Separate from modeling package
- Calculates individual frames
  - –High resolution (2k x 2K -- 4K x 4K)
  - –Advanced algorithms (anti-aliasing, lighting, shadows, effects)
  - –Highly realistic!
- Server based; hours to days processing
  - –*RenderPlex*

Compositing

- Merges raster images from multiple sources into a single image
- Examples:
  - –CG rendering *plus*
  - –Particle system *plus*
  - –Live actors
- Done on a frame by frame basis

Post Production

- Video editing
  - CG animations
  - Live action
- Non-linear video editing
- Sound
- etc.

Compression

- File (and video) oriented
- Lossless: RLL, LZW
  - –Output identical to input
- Lossy: JPEG, MPEG, Fractel, Wavelet
  - –Output is not same as input
  - –Greater compression
  - –“Looks good”

RRQ

- **Recap**
- **Review**
- **Questions**

LEADING THE NEXT GENERATION OF COMPUTING !

digital™

Digital Equipment Corporation