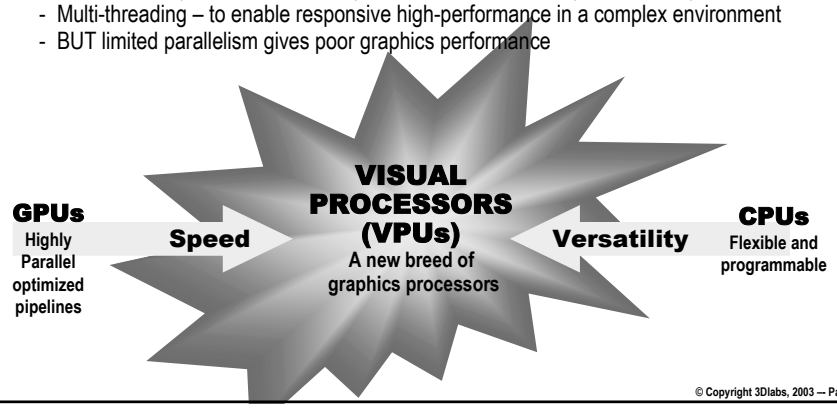
Visual Processing

The next graphics revolution

- **GPUs – Graphics Processors – have been engineered for extreme speed**
  - Highly parallel pipelines – exploits natural parallelism in pixel and vertex processing
  - BUT hardware-centric design limits algorithm complexity and adaptability
- **CPUs – Central Processors – have been engineered for flexibility**
  - High-level language programmability – using state-of the art compiler technology
  - Virtual memory – to cost-effectively break the barriers of physical memory
  - Multi-threading – to enable responsive high-performance in a complex environment
  - BUT limited parallelism gives poor graphics performance



3Dlabs

The Need for Programmability

Visual realism needs more algorithmic flexibility

- **Today's fast graphics chips are used for modeling and design…**
  - Limited, hardware-based vertex and fragment operations

- **But highly realistic images are still rendered on CPUs**
  - Large, sophisticated software renderers

- **Today's graphics hardware cannot accelerate large software renderers**
  - Traditional hardware has fixed functionality, limited flexibility and poor programmability

Visual Processing Goal #1

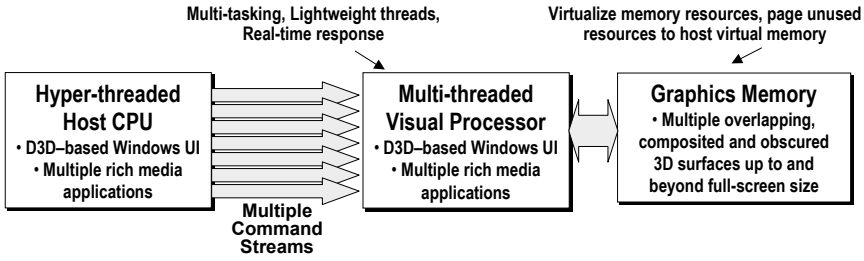
Accelerate shading algorithms expressed in high-level, hardware independent shading languages

3Dlabs

The Need for Flexibility

Learning from decades of CPU experience

- **Too many graphics chips are optimized to run a single game application**
  - Single task, single full screen rendering surface
- **There are now many competing demands for graphics resources**
  - Multiple threads and applications – all demanding seamless interactivity
  - 3D GUIs arriving – using an impossibly large amount of graphics memory



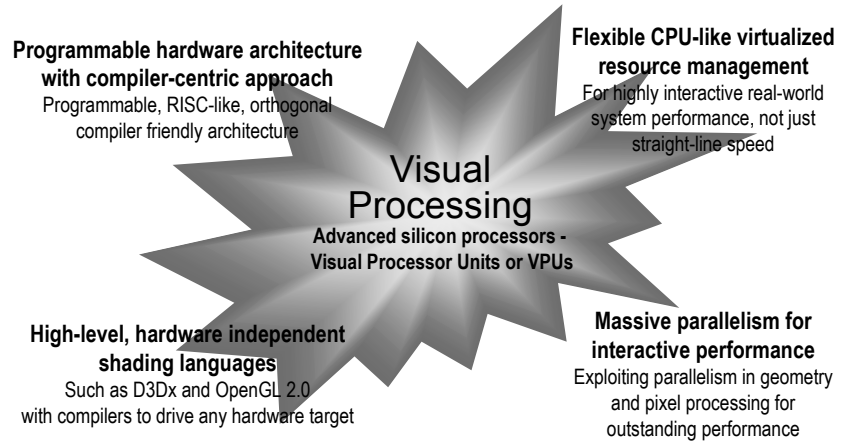
Visual Processing Goal #2

Virtualize processor & memory resources for optimal system performance and flexibility

The Visual Processing Vision

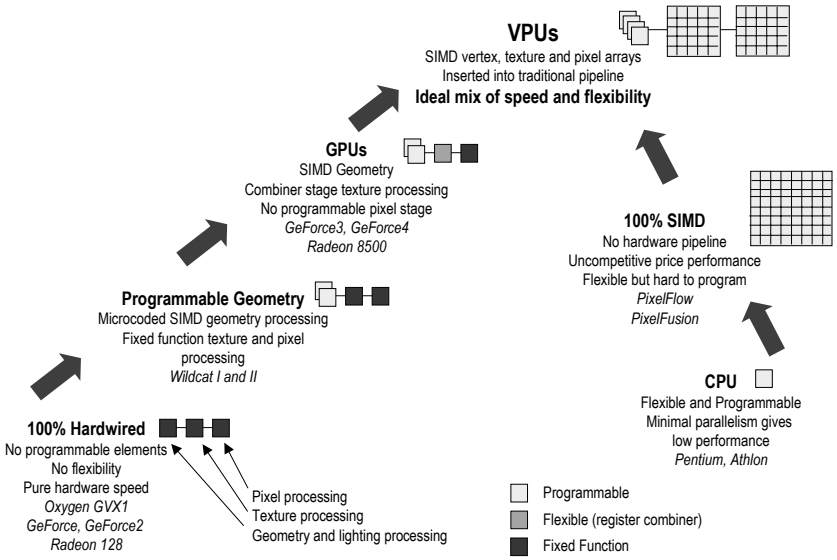
Combining state-of-the-art hardware and software

- **To create the next major step in graphics acceleration capability**



3Dlabs' Visual Processing Architecture

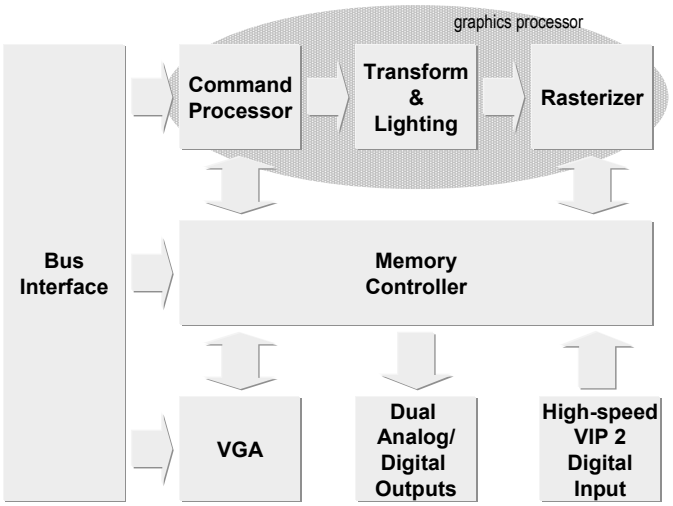
The ideal mix of flexibility and price/performance



P10 Block Diagram

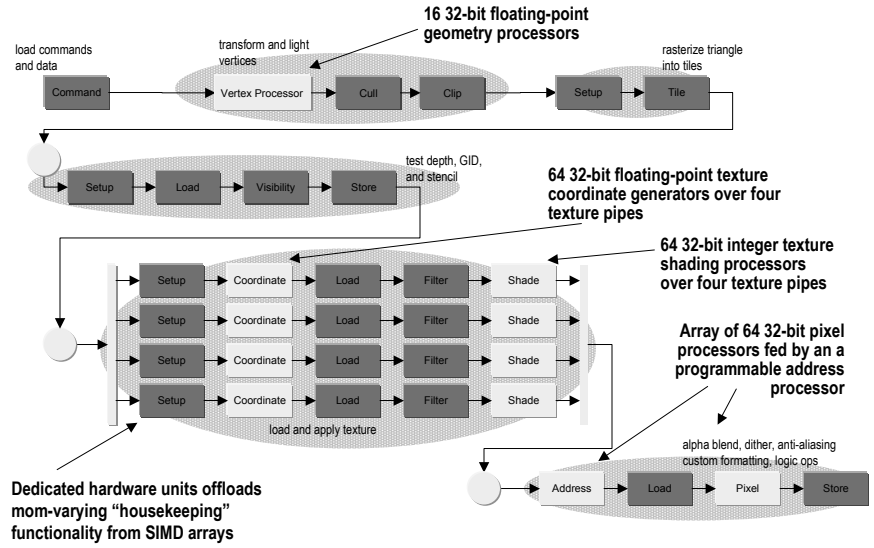
The industry's first VPU

- **At a high-level – P10 looks deceptively normal**



P10 Pipeline Architecture

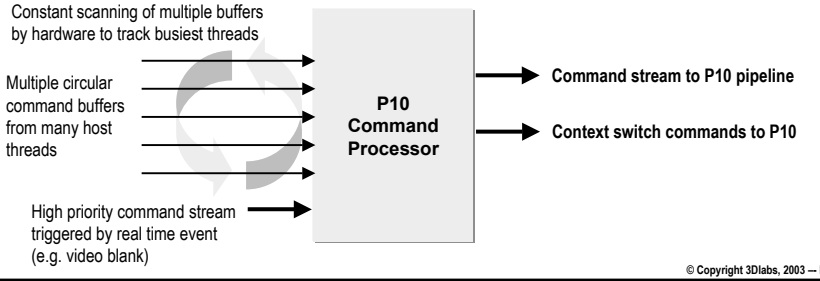
Traditional pipeline with five programmable stages



P10 Command Processor

Creating the first real-time, multi-threaded VPU

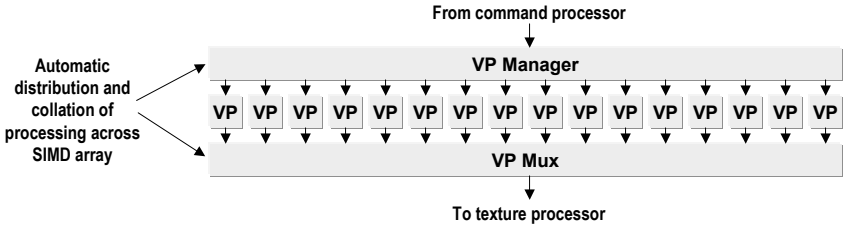
- **P10 can efficiently process multiple visual processing threads**
  - Use a "virtual VPU" context for each thread
- **Hardware scans command ring buffers from multiple host threads**
  - No CPU software intervention required – VPU context switches to track CPU workload
- **Efficient context switch to rapidly jump between VPU threads**
  - General-purpose state change in 15us – very low-overhead
- **Isochronous interrupt invokes hard real-time state change in 3us**
  - High priority processes e.g. no-drop video processing



P10 Vertex Processing

Flexible, powerful and efficient

- **SIMD array of 16 floating point scalar processors**
  - Powerful, RISC-like instruction set Compiler Friendly
  - Move, Add, Mul, MAdd, Min, Max, IntFloat, Fract, Trunc, Dot, Div, RSqrt, Log, Clipping
- **Flow control includes conditional jumps, subroutines, loops** Compiler Friendly
  - A superset of DX9 flow control
- **Parallelism is completely automatic** Compiler Friendly
  - Simple programming model - single processor



3Dlabs

Using P10 Vertex Processing

Substantial flexibility

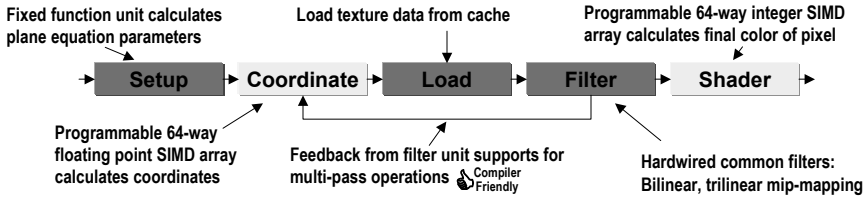
- **Loops add significant flexibility** Compiler Friendly
  - E.g. can programmatically handle up to 200 lights
- **Can write intermediate results into memory** Compiler Friendly
  - Enables multi-pass algorithms
- **Can process surfaces and higher-order geometry**
  - NURBS, N-Patches, tessellation, surface subdivision, vertex blending
- **Static and dynamic displacement mapping**
  - Offset position of vertex by contents of texture map
- **Wavelet-based geometry compression**
  - Useful for large geometry models
- **No set limits to what ANY of the SIMD arrays can used for**
  - General purpose processors reading and writing memory
  - Floating point image processing
  - Generate texture coordinates
  - Payroll …

3Dlabs

P10 Texture Processing

128 32-bit SIMD Processors

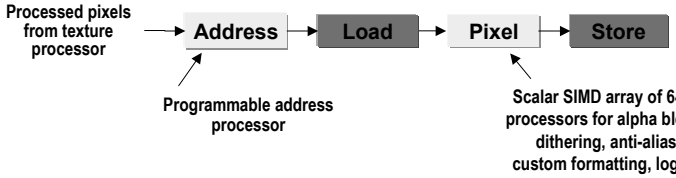
- **Programmable filter kernels**
  - Flexibility to go beyond hardwired filters - anisotropic, bi-cubic, mip-mapped 3D…
- **Eight simultaneous textures in a single pass (Nvidia has four, ATI six)**
  - Each map independently 1D, 2D, 3D or cube with independent size, wrapping, filtering
- **Up to 8Kx8K or 1Kx1Kx1K texels per texture**
  - Arbitrary size – no power of 2 restrictions (unless doing mip-mapping)
- **Programmable texture formats**
  - Compressed DXT1-5, YUV422 supported natively, others can be programmed
- **Programmability for advanced functionality**
  - Wavelets, ray casting into volumetric textures, advanced video operations…



Programmable Pixel Processing

Unprecedented back-end processing flexibility

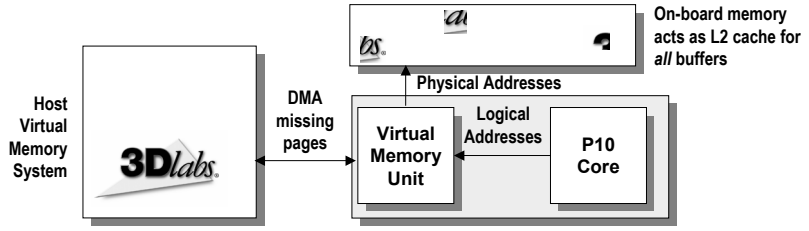
- **100% programmable back-end flexibility** Compiler Friendly
  - For anti-aliasing, image processing, compositing etc…
- **Highly flexible pixel formats** Compiler Friendly
  - Use byte planar memory formats for unlimited pixel depth – including 16 bit components
  - R10:G10:B10:A2 is also a natively supported format
- **Rich anti-aliasing choices**
  - OpenGL 'edge' anti-aliasing – 1-16 samples
  - 64-bit hardware accumulation buffer – unlimited samples
  - Super sampling – effectively unlimited samples
  - Multi sampling – 1-8 samples - anywhere on a 16x16 sub pixel grid, stochastic positioning
    - Preferred - same memory footprint as super sample - higher performance, slightly less quality



P10 Virtual Memory

Enabling virtual 16Gbyte graphics boards

- **Virtual addresses converted to physical addresses – just like a CPU**
  - Logical address space of 16Gbytes with a 4Kbyte page size
  - For ALL memory accesses – not just textures
- **On-board memory becomes a large L2-cache** Compiler Friendly
  - With hardware optimized demand paging to and from host virtual memory/disk
- **Solves multiple intractable texture management problems**
  - Massive textures can be transparently handled – even if they don't fit in physical memory
  - Eliminates memory fragmentation - texture pages do not need to be physically contiguous
  - Improves performance – pages only loaded as required
  - Obscured graphics memory automatically paged out to host

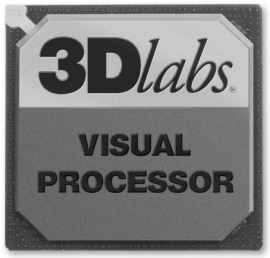


The P10 Processor

Leading performance, functionality and quality

- **170GFlops and 1 TeraOp Processing Power**
  - Over 200 floating point and integer SIMD processors
- **Over 20 GB/sec Memory Bandwidth**
  - Up to 256MB of 256-bit DDR memory
- **Fast graphics performance**
  - 60M drawn polygons/sec, Over 1Gpixels/sec fill-rate: drawn, trilinear mip-mapped
- **10 bit DACs**
  - Industry-leading quality
- **Dual head**
  - Dual analog and digital displays
- **High-speed VIP-2 video input port**
  - For TV and video applications
- **For embedded applications**
  - 14W worst case power scales linearly with frequency
  - Can interface directly to PCI-66 (with clamping)
    Can drive 128-bit memory for reduced power and space

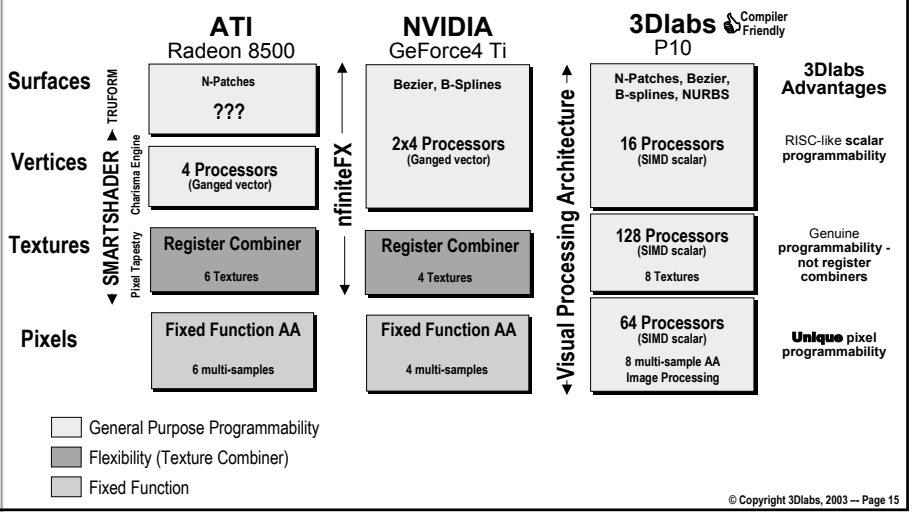
820-ball thermally-enhanced HSBGA package



Visual Processing Architecture

Flexible programmability throughout the pipeline

- **Compilers need RISC-like orthogonality**
  - NOT CISC-like specialized gadgets (and acronyms)



P10 Display Unit

The ultimate RAMDAC

- **Two units in P10 for independent dual displays**

16 entry GID LUT fed from 4 bit planes of any channel for per-pixel control of double buffering, LUT select etc.
Control field inserted back into alpha channel