AMD
Smarter Choice

Radeon™ HD 2900 and Geometry Generation

Michael Doggett

September 11, 2007

Overview

- Introduction to 3D Graphics
- Radeon 2900
  - Starting Point
  - Requirements
  - Top level
  - Pipeline Blocks from ‘top to bottom’
    - Command Processor
    - Shader Setup Engine
    - Ultra Threaded Dispatch Processor
    - Shader Core
    - Texture
    - Render Backend
    - Memory Controller
- Dirext3D 10 Geometry Shader
- Conclusion

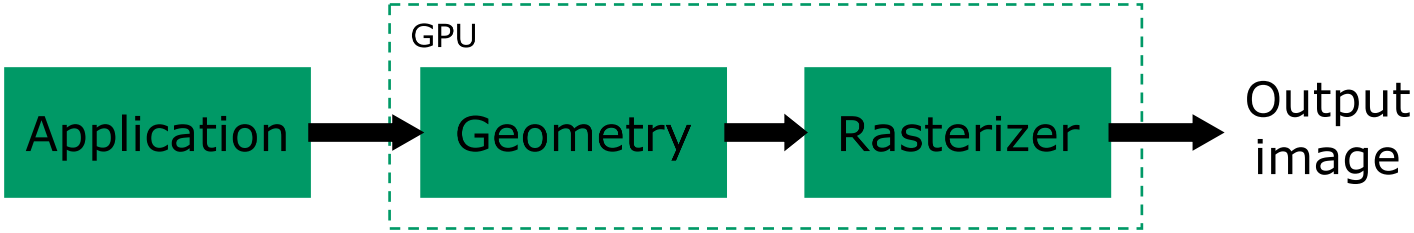
Introduction to 3D Graphics

Rendering Pipeline

Combination of devices and methods for creating image from 3D scene description

Common pipeline structure:

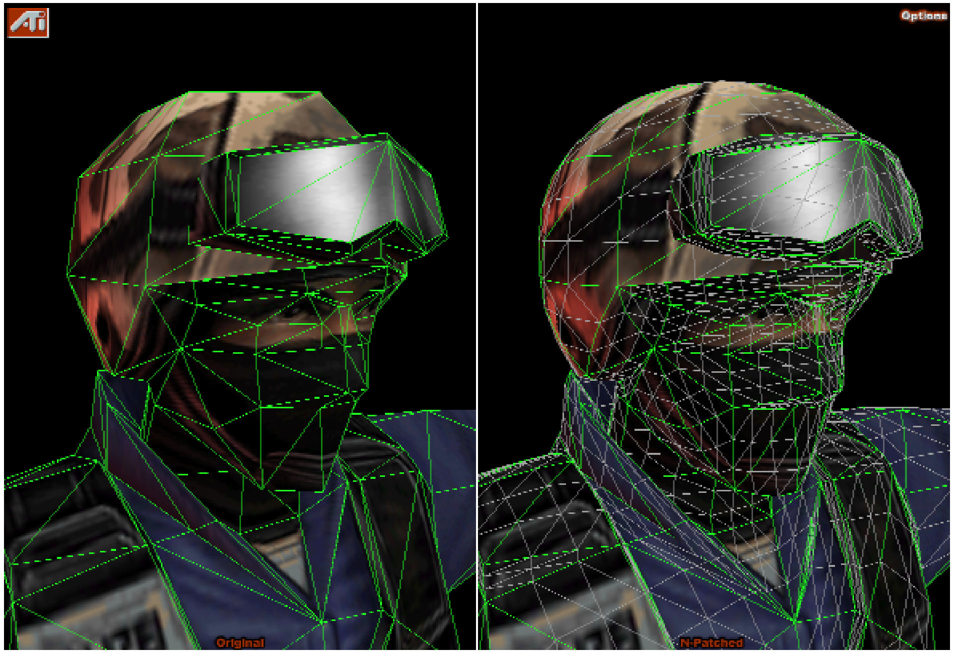
- Application (runs on CPU)
- 3D rendering – GPU
  - Geometry/vertex processor
  - Rasterizer



Polygons

The most common polygon type used in 3D graphics is a **triangle**

More polygons – better approximation of the surface curvature



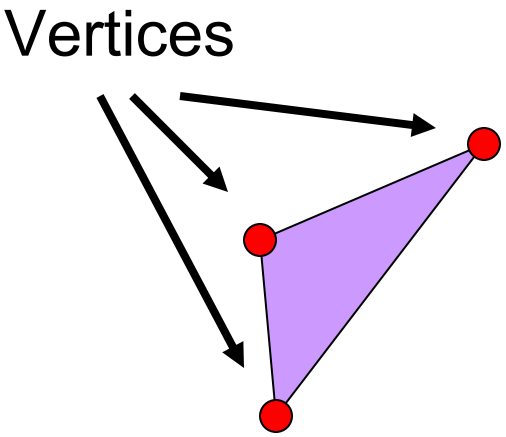
Geometry Definition

Triangles are defined by points in corners – **vertices**

- Operations on geometry is called **vertex processing**
- Vertex position is relative to some arbitrary point

Other descriptions of vertex attributes

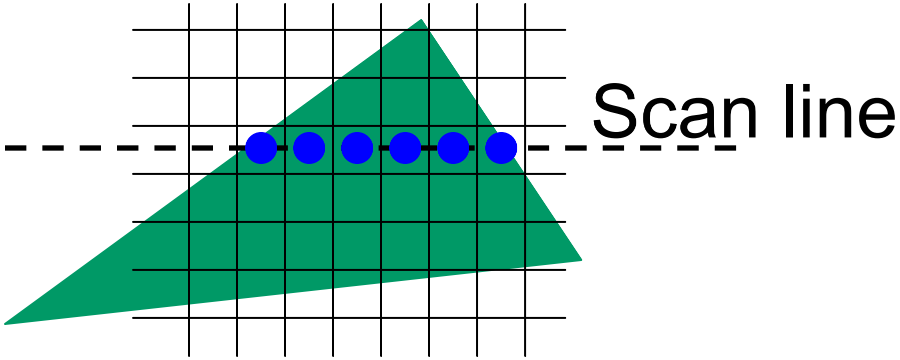
- Color
- Texture coordinates
- Normal
- Other custom attributes



Rasterization

Break up triangles into raster elements (pixels)

- Perform scan conversion

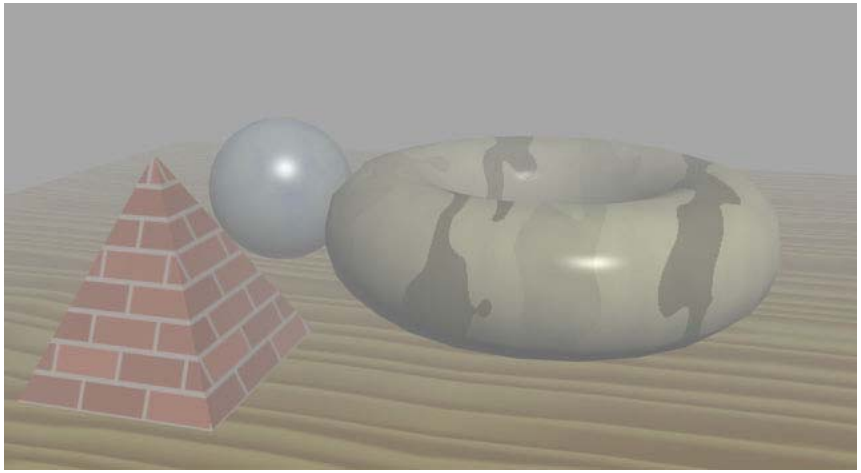


Later perform other per-pixel operations

Pixel Processing

Each screen pixel of the rendered 3D surface is evaluated

- Light can be added to control the lighting of the scene
- Texturing adds details
- Various effects can be applied to enhance the scene appearance



Texturing

One of the simplest ways to define material

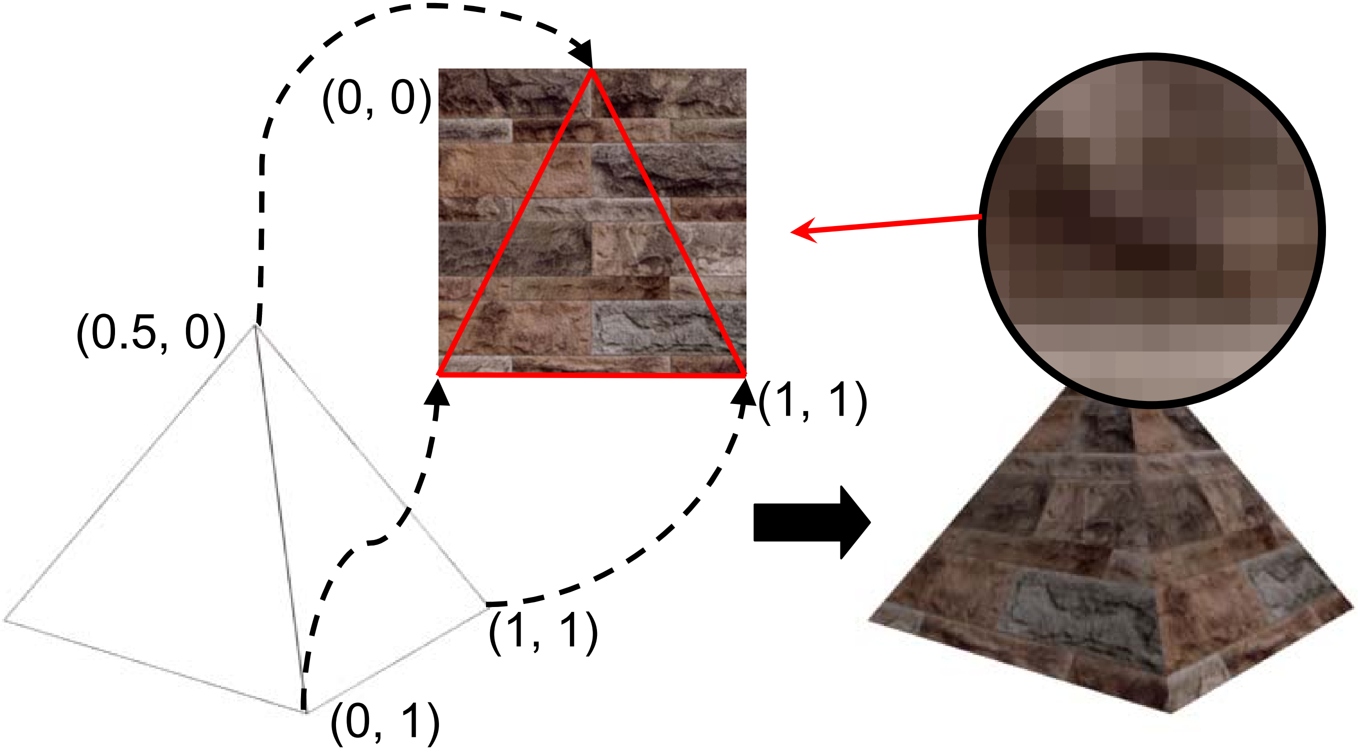
- Adds details to objects without extra geometry
- All details are in painted **textures**

Many different parameters control the texture appearance

Texturing

Think of textures as thin stretchy painted film used for shrink-wrapping

- **Texture coordinates** define texture application



Radeon 2900

Starting point

- Combine the best of existing technology
  - R5xx series
    - Heavily threaded shader cores
      - Hides latency of memory fetch
    - Vec4+1 Vertex and Vec3+1 Pixel shaders
    - Ringbus memory subsystem
  - XBOX 360 GPU
    - Unified shader architecture
      - Vertex and Pixel
      - Vec4+1
      - Stream Out
    - Unified L1 texture cache
    - Introduced Tessellator

Requirements

- DirectX10 compatible
- Support new driver model
  - Vista driver model
- Scalable family
  - "Number" of shader cores, texture units, render back-ends.
  - Shader scalable in number of pipes, SIMDs.
  - Target specific cost, feature set and performance levels for each part

Top Level

Red – Compute

Yellow – Cache

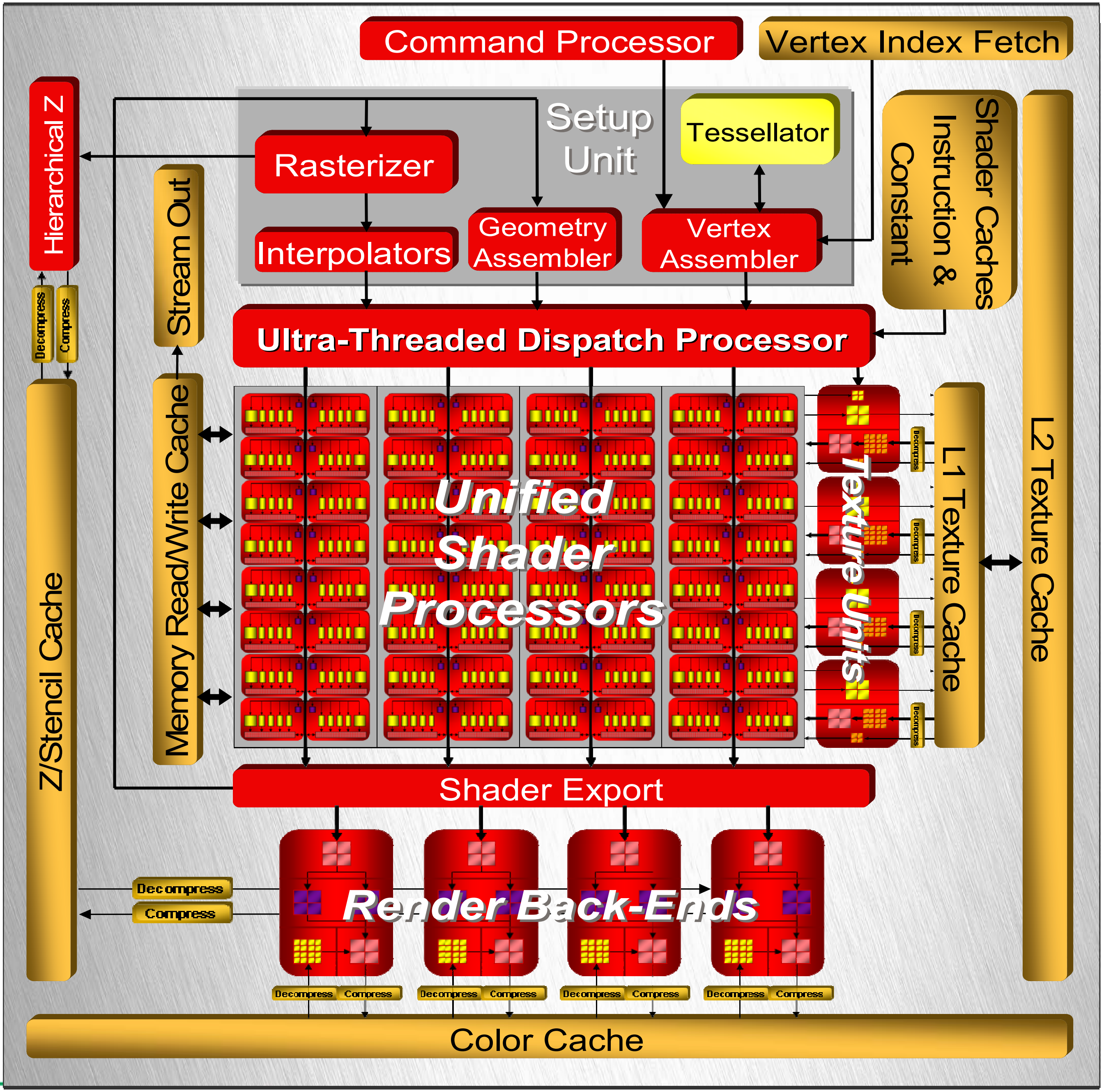
Unified shader

Shader R/W

Instr./Const. cache

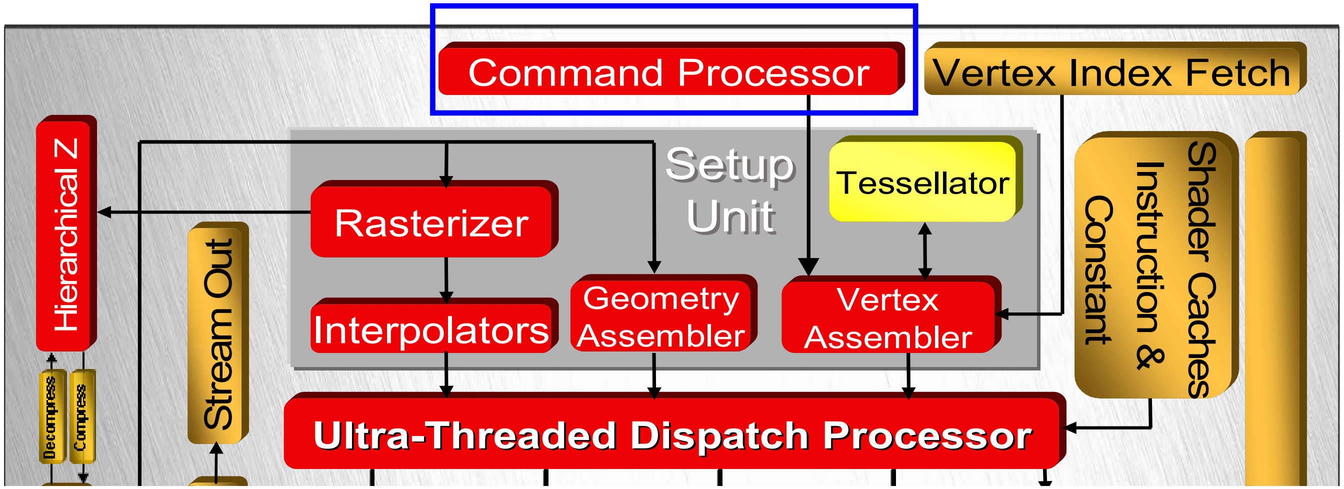
Unified texture cache

Compression

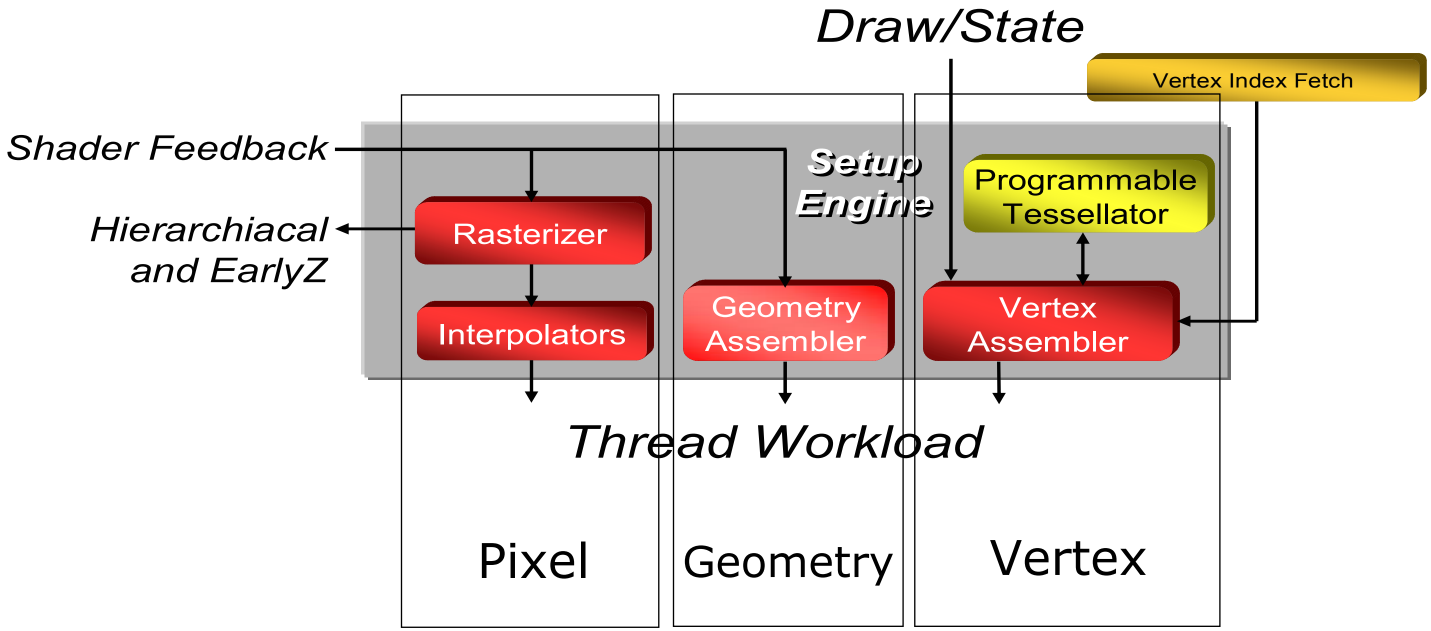


Command Processor

- GPU interface with host
- A custom RISC based Micro-Coded engine
- First class memory client with Read/Write access
- State management

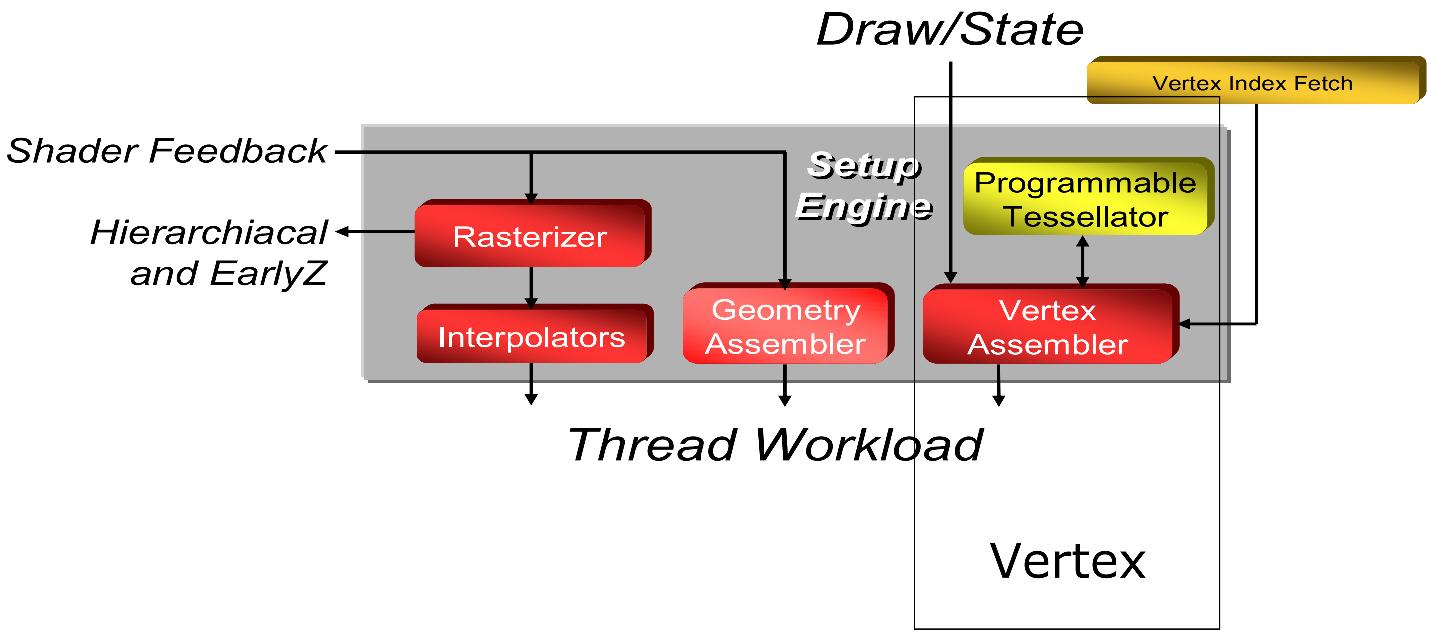


Shader 'Setup Engine'



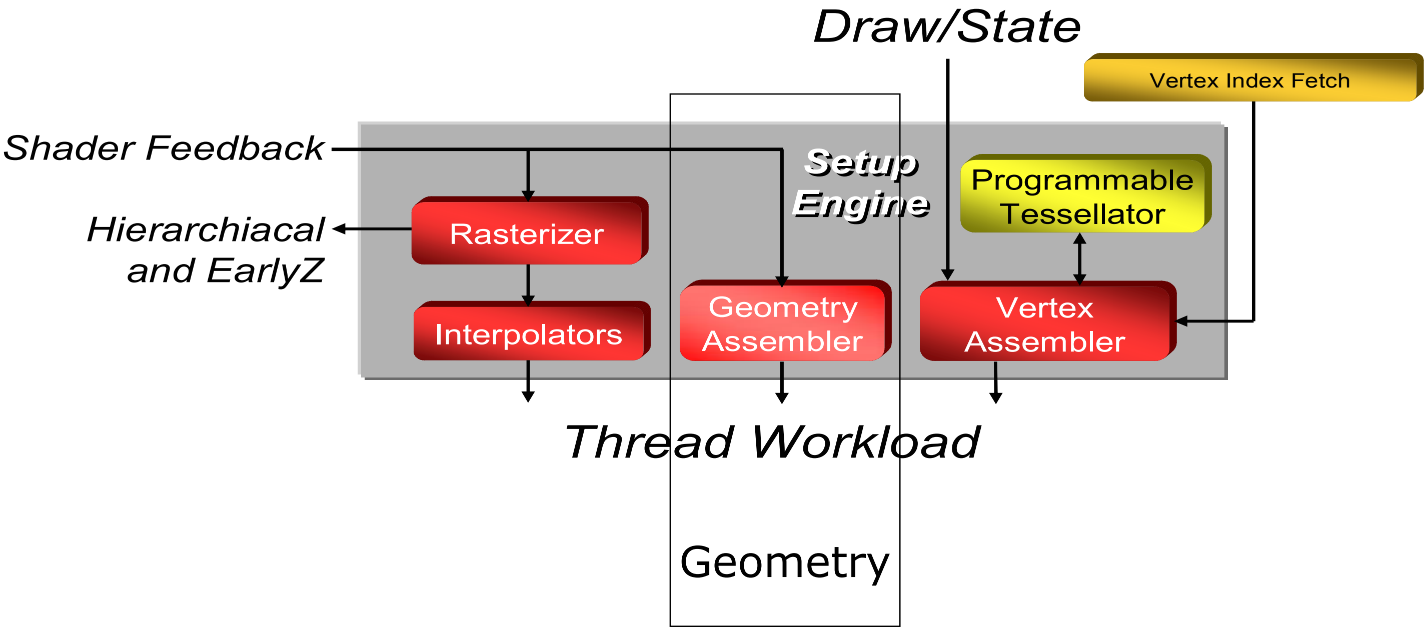
- 3 groups of blocks feeding 3 data streams
  - Each group feeding 16 elements (Vertices/Geometry/Pixels)/cycle

Shader 'Setup Engine'



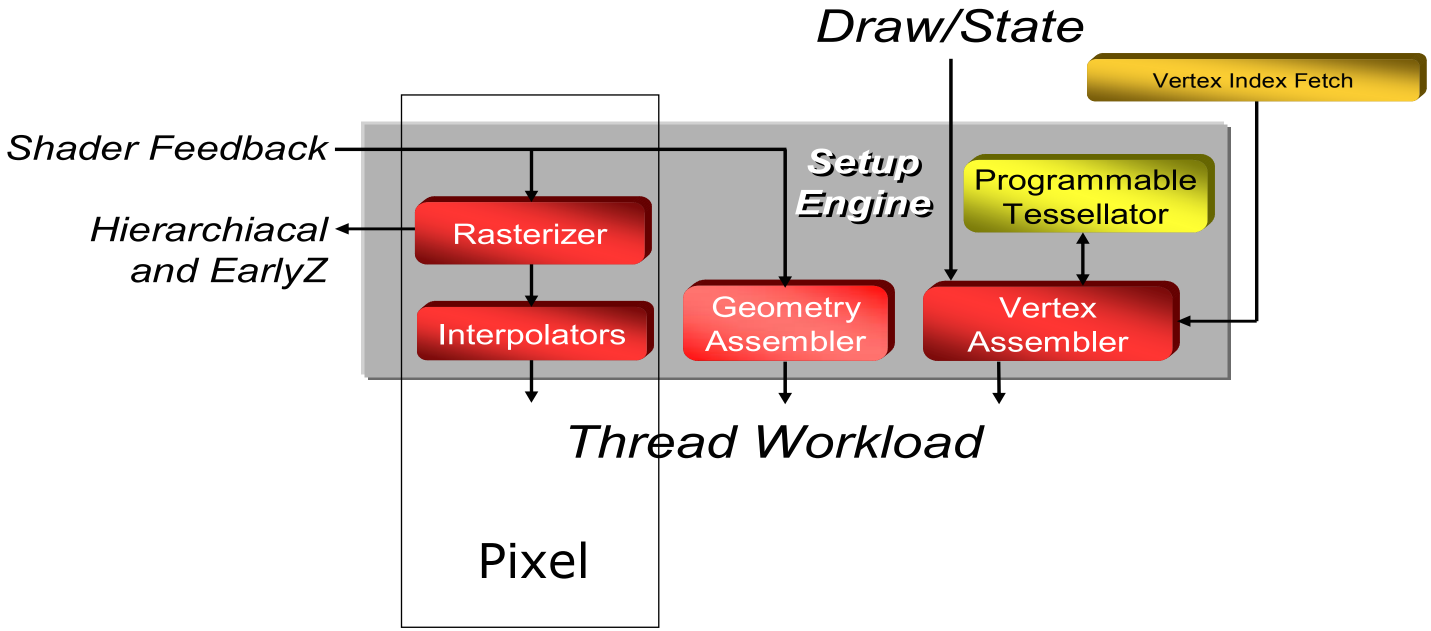
- Vertex blocks
  - Primitive Tessellation
  - Inputs – Index & instancing
  - Sends vertex addresses to shader core

Shader ‘Setup Engine’



- Geometry blocks
  - Uses on/off chip staging
  - Sends processed vertex addresses, near neighbor addresses and topological information to shader core

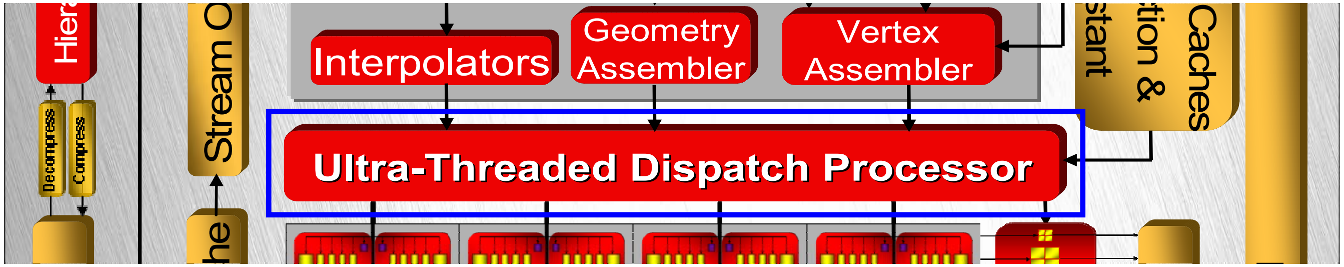
Shader 'Setup Engine'



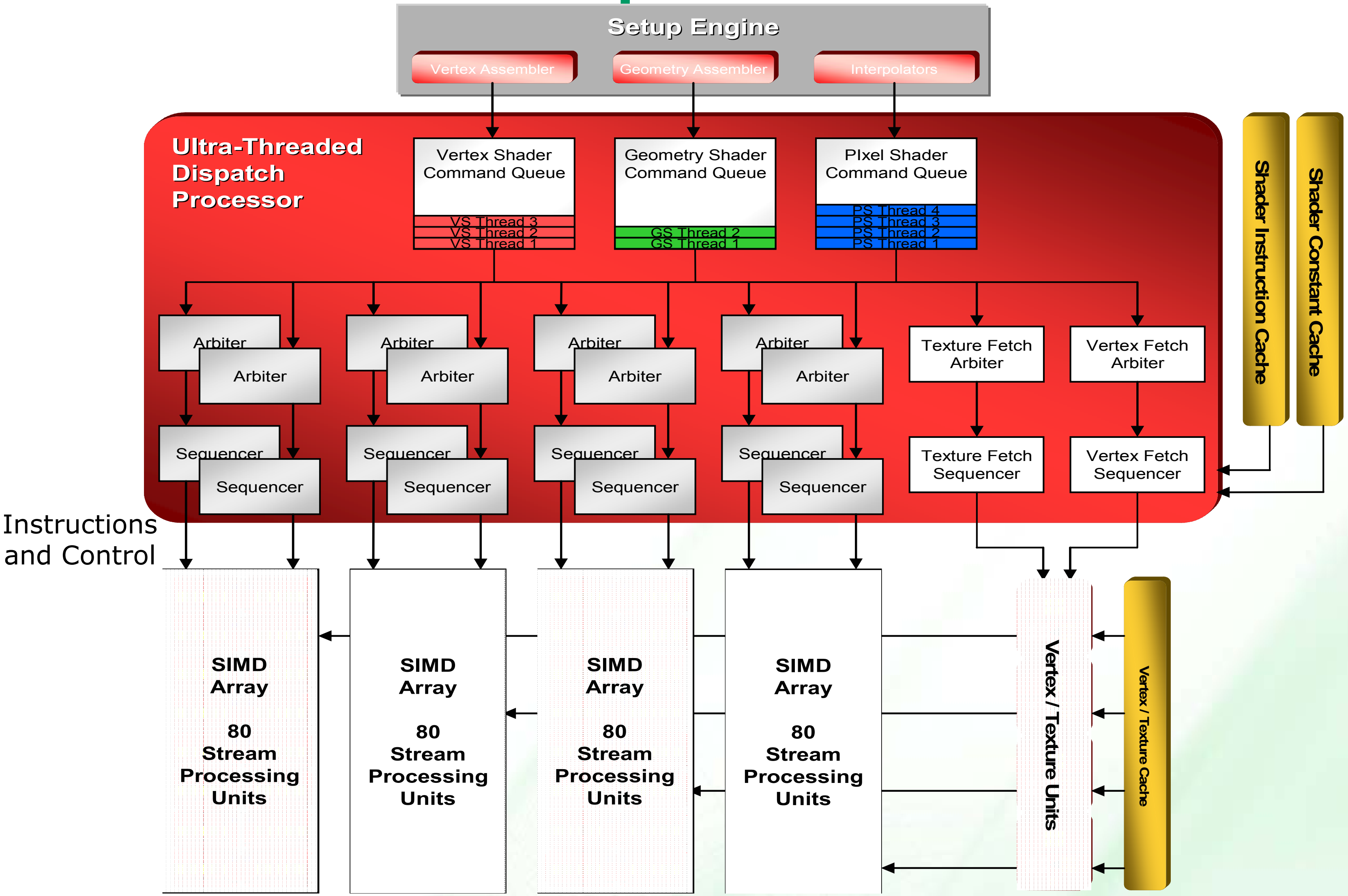
- Pixel blocks
  - Triangle setup, Rasterization and Interpolation
  - Interfaces to depth to perform HiZ/Early Z checks

Ultra-Threaded Dispatch Processor

- Main control for the shader core
  - All workloads have threads of 64 elements
  - 100’s of threads in flight
  - Threads are put to sleep when they request a slow responding resource
- Arbitration policy
  - Age/Need/Availability
  - When in doubt favor pixels
  - Programmable



Ultra-Threaded Dispatch Processor



Setup Engine

Vertex Assembler

Geometry Assembler

Interpolators

Ultra-Threaded Dispatch Processor

Vertex Shader Command Queue

VS Thread 3

VS Thread 2

VS Thread 1

Geometry Shader Command Queue

GS Thread 2

GS Thread 1

PIxel Shader Command Queue

PS Thread 4

PS Thread 3

PS Thread 2

PS Thread 1

Arbiter

Arbiter

Arbiter

Arbiter

Arbiter

Arbiter

Arbiter

Arbiter

Texture Fetch Arbiter

Vertex Fetch Arbiter

Sequencer

Sequencer

Sequencer

Sequencer

Sequencer

Sequencer

Sequencer

Sequencer

Texture Fetch Sequencer

Vertex Fetch Sequencer

Shader Instruction Cache

Shader Constant Cache

Instructions and Control

SIMD Array

80 Stream Processing Units

SIMD Array

80 Stream Processing Units

SIMD Array

80 Stream Processing Units

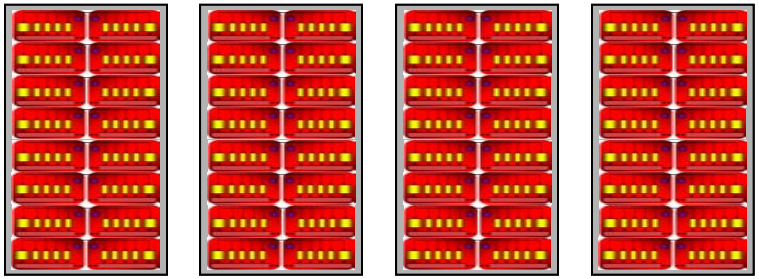
SIMD Array

80 Stream Processing Units

Vertex / Texture Units

Vertex / Texture Cache

Shader Core



- 4 parallel SIMD units
- Each unit receives independent ALU instruction
- Very Long Instruction Word (VLIW)
- ALU Instruction (1 to 7 64-bit words)
  - 5 scalar ops – 64 bits for src/dst/cntrls/op
  - 2 additional for literal constant(s)

Stream Processing Units

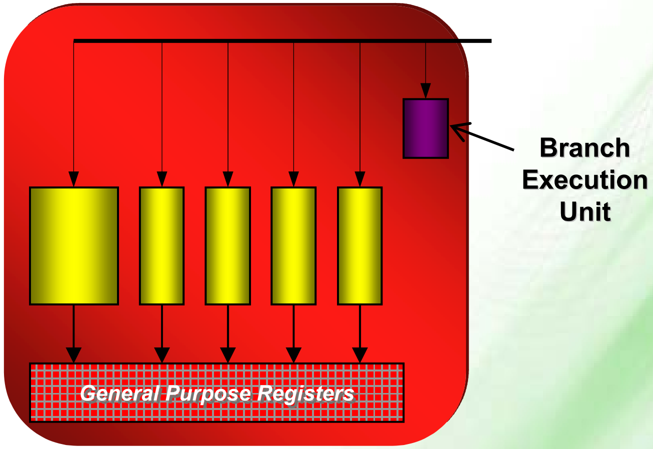
5 Scalar Units

- Each scalar unit does FP Multiply-Add (MAD) and integer operations
- One also handles transcendental instructions
- IEEE 32-bit floating point precision

Branch Execution Unit

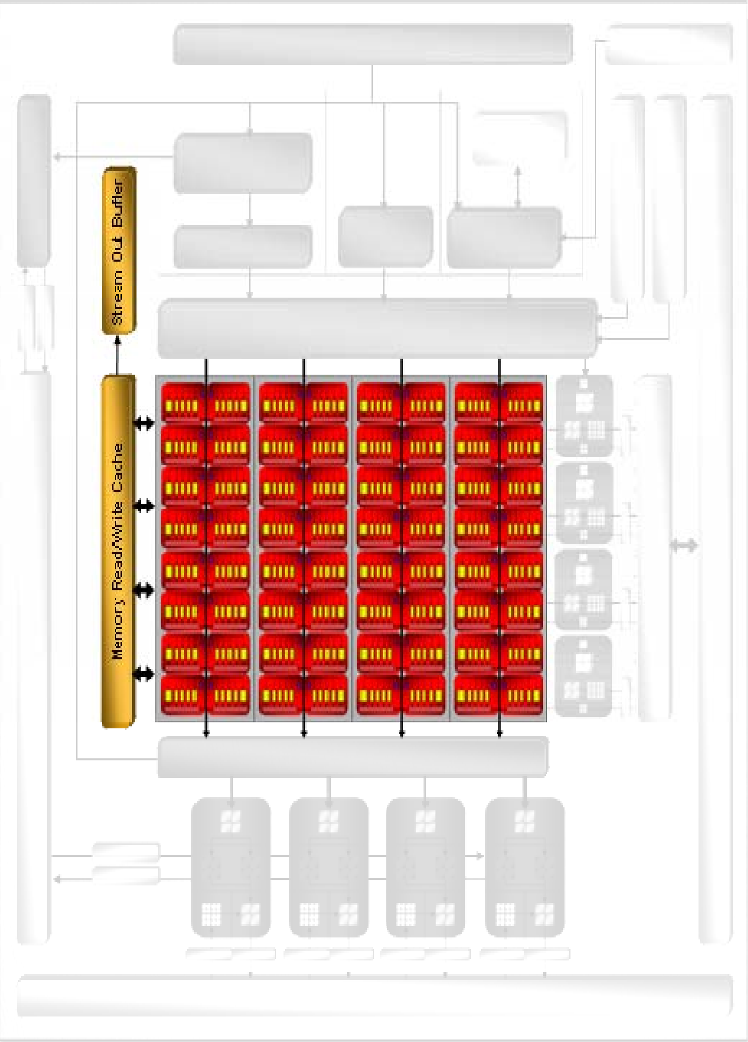
- Flow control instructions

Up to 6 operations co-issued

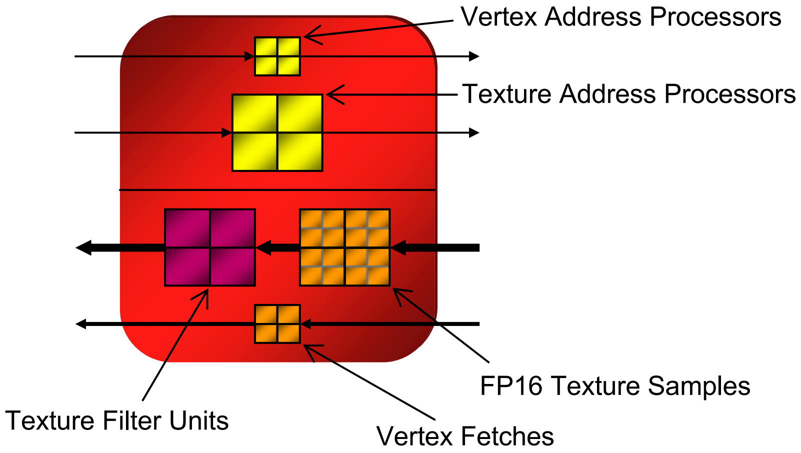


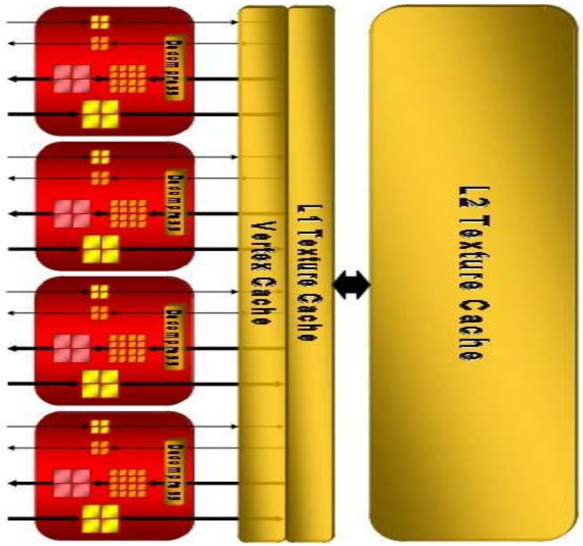
Memory Read/Write Cache

- Virtualizes register space
  - Allows overflow to graphics memory
  - Can be read from or written to by any SIMD (texture & vertex caches are read-only)
  - 8KB Fully associative cache, write combining
- Stream Out
  - Allows shader output to bypass render back-ends and color buffer
  - Outputs sequential stream of data instead of bitmaps
- Used for Inter-thread communication



Texture Units





Fetch Units

- 8 Fetch Address Processors each
  - 4 filtered and 4 unfiltered
- 20 Texture Samplers each
  - Can fetch a single data value per clock
- 4 filtered texels (with BW)
  - Bilinear filter one 64-bit FP color value per clock, 128b FP per 2 clocks for each pixel

Fetch Caches

- Unified caches across all SIMDs
- Vertex/Unfiltered cache
  - 4kb L1, 32Kb L2
- Texture cache
  - 32KB L1, 256KB L2
  - Texture

Render Back-Ends

Double rate depth/stencil test

- 32 pixels per clock for HD 2900
- New HiStencil

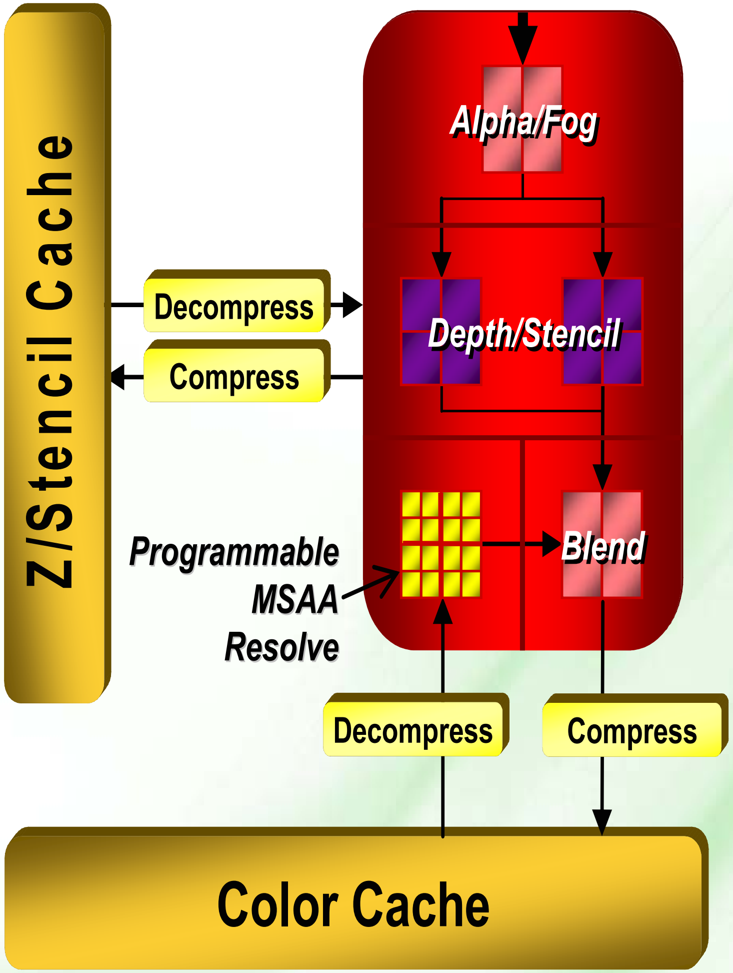
Programmable MSAA resolve

- Allows Custom AA Filters

New blend-able DX10 surface formats

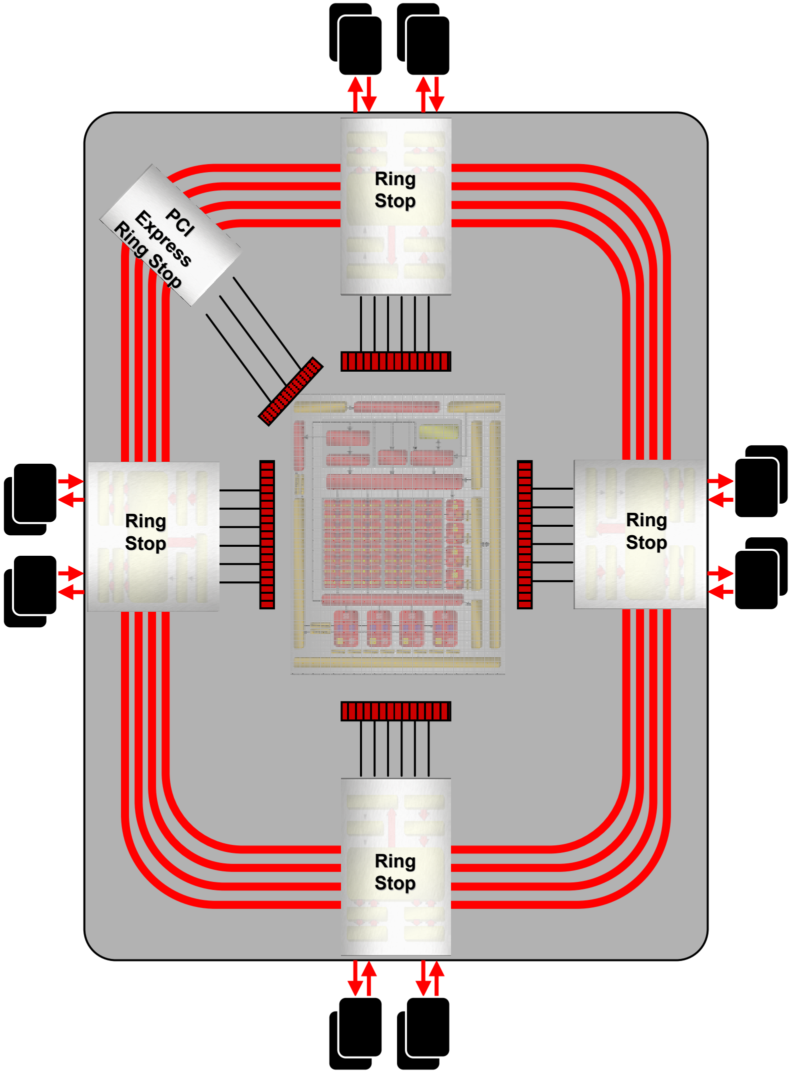
- 128-bit and 11:11:10 floating point format

Up to 8 Mulptiple Render Targets with MSAA support



Memory Interface and Controller

- 512-bit Interface
  - Compact, stacked I/O pad design
  - More bandwidth with existing memory technology
  - Improved cost:bandwidth ratio
  - 8 x 64 bit memory channels
- Double ringbus
  - 512 bit read and write

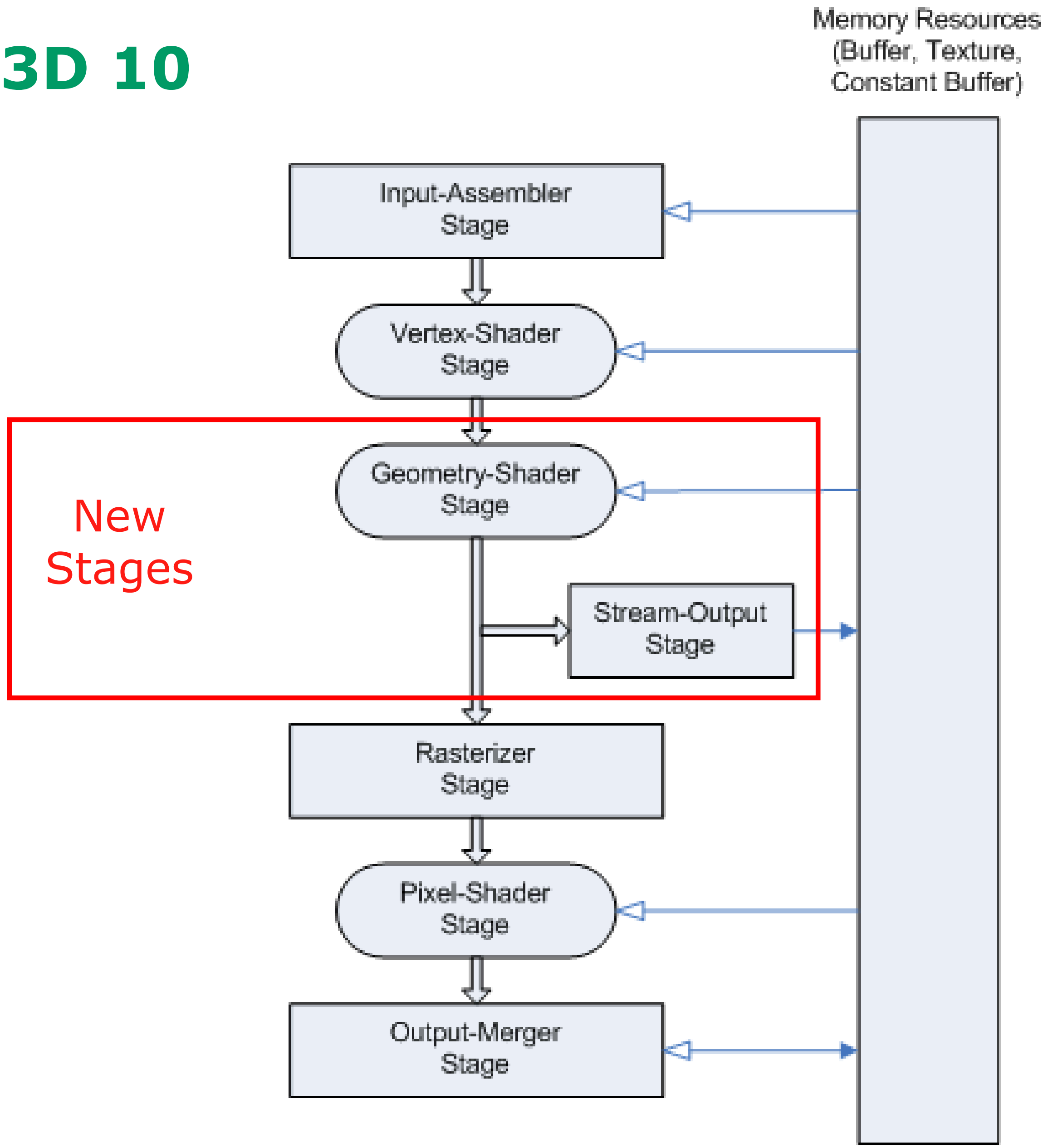


Radeon HD 2000 Series

Radeon	2900	2600	2400
Stream Processors	320	120	40
SIMDs	4	3	2
Pipelines	16	8	4
Texture Units	16	8	4
Render Backends	16	4	4
L2 texture cache (KB)	256	128	0
Technology (nm)	80	65	65
Area (mm2)	420	153	82
Transistors (Millions)	720	390	180
Memory Bandwidth	512	128	64

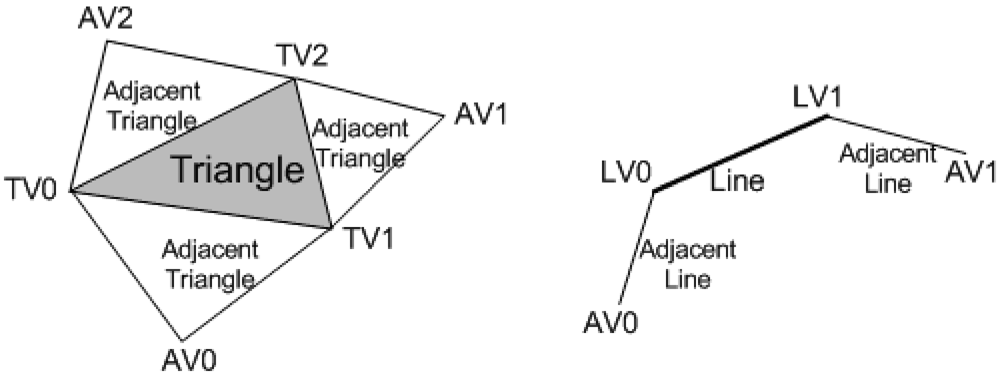
Direct3D 10 Geometry Shader

Direct3D 10


Memory Resources
(Buffer, Texture,
Constant Buffer)
Input-Assembler Stage
Vertex-Shader Stage
New Stages
Geometry-Shader Stage
Stream-Output Stage
Rasterizer Stage
Pixel-Shader Stage
Output-Merger Stage

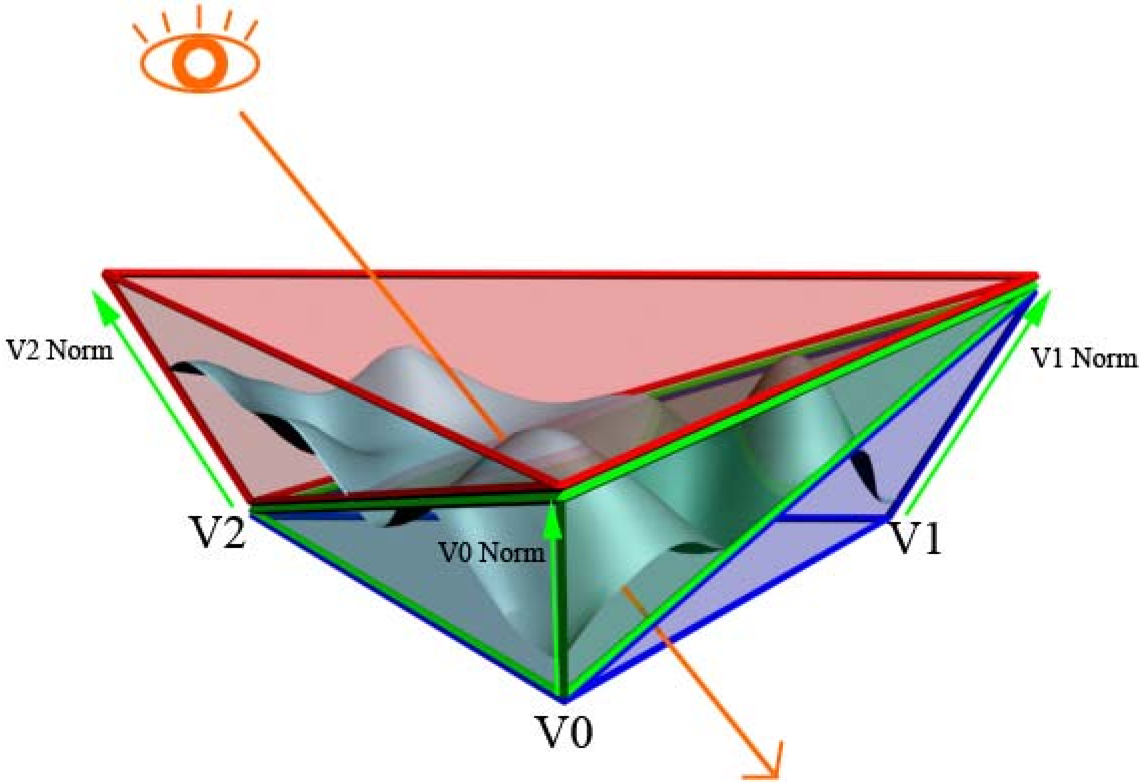
Direct3D 10 Geometry Shader

- Input
  - Primitive
    - Line, Point, Triangle
  - Neighbouring vertices
    - 2 for Line, 3 for Triangle
- Output
  - Rasterizer
  - Vertex buffer in memory (Stream Out)



Geometry Shader example

- Per-pixel Displacement Mapping
  - – Hirche, Ehlert, Guthe, Doggett, GI2004
- Similar to DirectX SDK Displacement Mapping



Where next ?

- Move fixed functions blocks to shader
  - Improve programmability, reduce area, improve reuse, maintain/target performance
- Enhancements for GPGPU
  - Improved precision and compliance
  - New APIs, new functions
- New technologies such as 65, 55, 45, 32…
- Graphics and gaming keeps on evolving
  - DX-next is already being discussed
  - We are well into next generation and next-next generations …

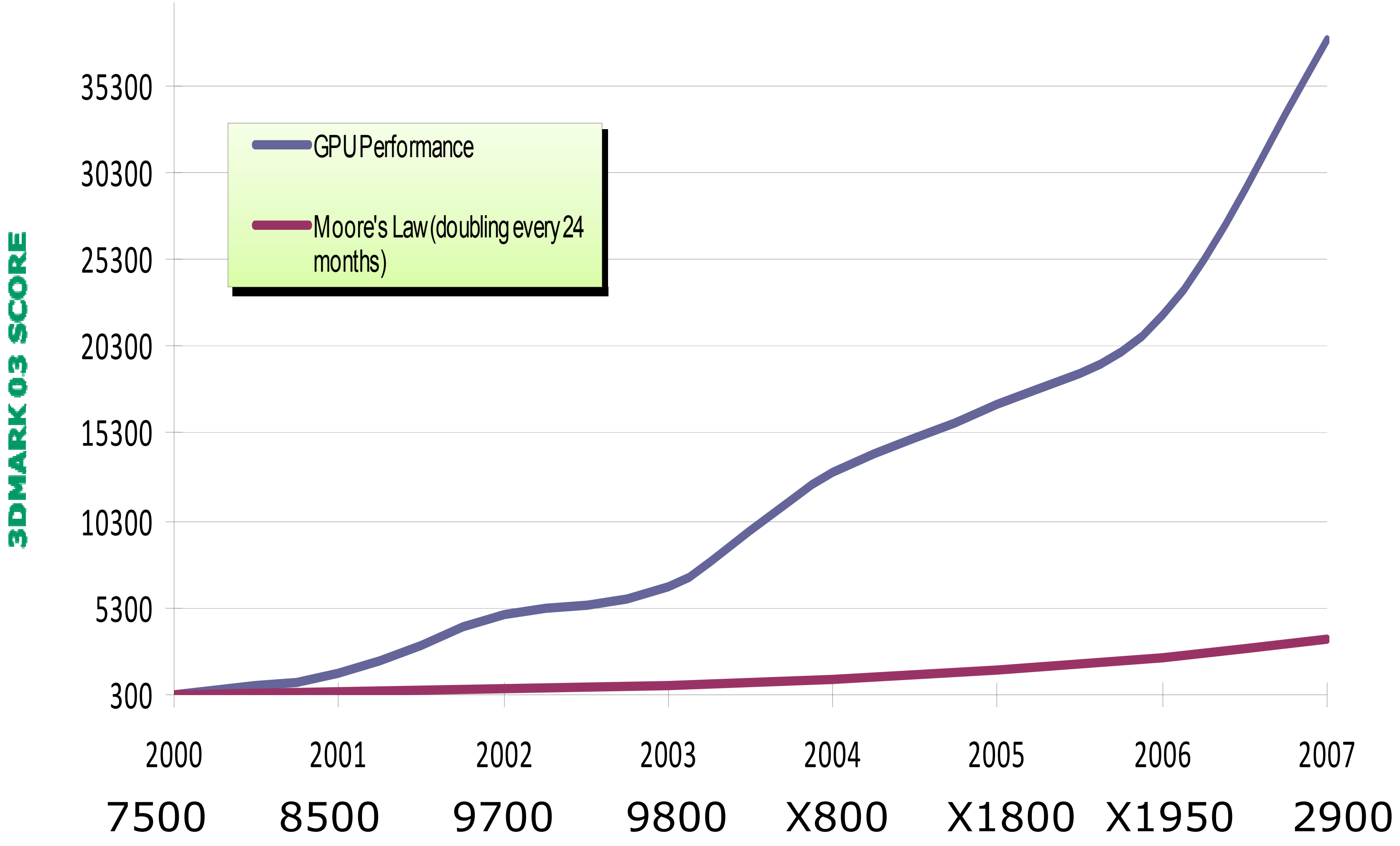
Radeon HD 2900

- Unified shader
  - Vertex, Geometry and Pixel
  - Multiple SIMD
  - 5-way scalar
- Shader cached memory read/write
- Geometry shader on/off chip storage
- 512 bit stacked I/O Memory Interface
- Full DX10 functionality

Questions and Demo

- Thanks to Eric Demers and Mike Mantor

Performance Improvements



	Rage Pro	Rage 128	Radeon	Radeon 8500	Radeon 9700 Pro	Radeon 9800 XT	Radeon X850 XT Platinum Edition	Radeon X1800 XT	Radeon X1950 XTX	Radeon HD 2900 XT	
Year	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	
Transistor Size	350 nm	250 nm	180 nm	150 nm	150 nm	150 nm	130 nm	90 nm	90nm	80nm	
Transistor Count	5 million	13 million	30 million	60 million	110 million	110 million	160 million	321 million	384 million	700 million	*1.7x / year*
Clock Speed	75 MHz	100 MHz	183 MHz	275 MHz	325 MHz	412 MHz	550 MHz	625 MHz	650 MHz	740 MHz	*1.3x / year*
Rendering Pipelines / Shader Processors	1	2	2	4	8	8	16	16	48	64	*1.6x / year*
Memory Bandwidth (GB/sec)	0.6	1.6	5.9	8.8	19.8	23.4	37.8	44.8	64.0	106.0	*1.8x / year*
Pixel Shading / Fill Rate (Mpixels/sec)	75	200	366	1100	2600	3300	8800	10000	31200	47360	*2.0x / year*
Vertex Processing (Mvertices/sec)	4	8	30	69	325	412	825	1250	1300	11840	*2.4x / year*
System Bus Bandwidth (GB/sec)	0.53	1.06	1.06	1.06	2.11	2.11	4	4	4	4	*1.3x / year*
DirectX Version and Shader Model Support	DirectX 5	DirectX 6	DirectX 7	DirectX 8.1 (SM1.4)	DirectX 9.0 (SM2.0)	DirectX 9.0 (SM2.0)	DirectX 9.0 (SM2.0b)	DirectX 9.0 (SM3.0)	DirectX 9.0 (SM3.0)	DirectX 10.0 (SM4.0)	
Features	Hardware Triangle Setup	128-bit memory interface, AGP 4X	Hardware transform & lighting, DDR memory support	Programmable Shaders, Higher Order Surfaces	Floating Point Processing , 256-bit memory interface, AGP 8X	Unlimited shader instructions , 256MB DDR2 memory support	PCI Express x16, GDDR3 memory support	Dynamic shader flow control, 128-bit precision, HDR output	GDDR4 memory support	Unified Shader Architecture, 512-bit memory interface	

The ATI Radeon HD 2900 developed by AMD is a Graphics Processing Unit (GPU) capable of massively parallel computation for high performance 3D graphics and general purpose algorithms. The unified shader architecture consists of a combination of MIMD and SIMD architectures of 5 way scalar arithmetic units running in parallel. The shader uses multi-threading to hide latency of memory access so that compute units are keep busy. The threads consist of vertex, geometry and pixel threads that represent different programmable stages of a traditional 3D graphics pipeline mapped onto a single scheduled shader unit. Varied distributed and unified caches are used for data, instructions, read only texture reads and vertex data. A ring based memory subsystem allows multiple clients to access multiple memory channels.

The Radeon HD 2900 includes a tessellator for generating highly detailed surfaces using the GPU in a fixed function pipeline. More complex surface operations can be performed using the Direct3D10 API's new geometry shader. This shader stage allows programmable operations on the geometry in a 3D scene, not just the individual vertices or pixels. This geometry shader gives access to neighbouring vertices which allows computation of surface properties. These surface properties can then be used to generate new vertices to represent a more complex geometry then was originally sent to the GPU enabling algorithms to be run that previously needed to be run on the CPU.