Quick glance agenda

• Discuss previous product families
• Introduction to HD2000 family
• Overview of major blocks and features of HD2000 series
  • Will cover from “top of pipe” to “bottom”
• Conclusion and questions

• NB: I’ve tried to leave the marketing-weasel out of this presentation ;-) 

• NB2: Will focus on products, not so much pie-in the sky...
How does one go about designing a “next architecture?”

- Mixture of art, science, politics and wishful thinking
- Evolving process, working with ISVs, MS and with internal feedback
  - Want to link new architecture with industry inflection points
- The balance between evolution and revolution is a constant consideration
  - A balance of resources vs. schedule vs. benefits vs. thought
- The first step forward is to look back...
Starting with the past

- R5xx went into production in fall 2005
- ATI’s “culmination” of the split shading PC architecture
  - Note worthy by having the first heavily threaded shader cores, capable of hiding most latencies and allowing for long shaders with branching and chained indirections
  - Note worthy by having full associative client caches, capable of hiding up to >256 cycles of memory latency
- Caches are mainly used to hide client memory requests
- Note worthy by having first ringbus memory subsystem (256bx2), with adaptive client arbitration including real time clients
R5xx architecture overview

- Classic Split Vertex / Pixel shaders
  - Vertices split round robin into VS shaders
- Screen aligned pipelines, including arbitration and load balancing
  - Pixels split along screen alignment into PS shaders
- Shaders designed for Vec4+1 or Vec3+1
- Distributed texture L1, 32k total
- Vertex cache at 4k~8k
Closer to the present

- Xenos (Xbox360) went into production in Summer 2005

- New architecture different than PC line
  - Unified shading architecture with “SM3+” cores
    - Vertex and Pixel
    - Thread based arbitration and load balancing
    - Unified L1 cache for textures
    - vertex fetch from shader
    - Vec4 + 1 shader architecture
    - Stream out
    - Split render clients from main core (using embedded dram)

- Introduced a generalized and flexible tessellator
What to build then?

• Considered multiple variations on architecture
  – “Folded” R5xx architecture
    • Good SW model and performance understood
    • Unclear how to achieve unified performance levels
  – Xenos extended to support new shaders and backend
    • Need new RBs and new MC
  – Other architectures

• Set our course for new horizons...
ATI Radeon HD 2000 Series: The Next Generation of GPUs

- A new architecture, leveraging the best of both worlds
  - More in tune with Xenos to leverage ISV experience
- With a new shader core
  - Including scalar based ALUs with improved precision
  - New deeply threaded sequencers and arbiters
  - Virtualization of main resources: Shaders, GPRs
  - Multi level and multiple shader I/O caches
- New backend
  - New Z architecture
  - New virtualized resources
  - Designed for FP16 (64b pixels)
  - Double the latency compensation in clients
- Fully distributed MC with 512b (8ch) memory organization
  - Read and write ring busses (512b each way, both R/W)
- New Video features as well
  - Introduction of AVP and UVD
- Technology: Highspeed 80nm & Low power 65nm
A scalable family

- Desktop will have price points from <$99 to ~$400

**Scalable family**
- We design with a “numbers” of for most elements, including shader cores, pipelines, texture units, raster backends
- Shader scalable in multiple dimensions as opposed to other products (#pipes, #SIMDs)
- Core functionality exists in all parts, since even 1-of elements have the basic feature sets
- A few features can be completely removed to save area
  - Or added, in the case of UVD/AVP (HD video at ultra-low power)
- Target specific cost, feature set and performance levels for each part
Block Overview: Command Processor

- First “layer” of GPU in design
- Handles command streams fetching, state management including register subsystems
- Functional highlights:
  - Core micro-coded engine with ucode RAM store
    - Controls state and decodes command streams
  - Full memory client with Read/Write access
  - Multiple command queues
  - Multiple register interfaces, including graphics and system
  - DMA engine for command stream fetching
  - Interrupt system
- New engine and new pipelined focused on small batch issues
**Shader “setup”**

- 3 main groups feeding 3 main data streams
- **Vertex block**
  - Performs tessellation
    - Flexible design, supporting different tessellation mode and extendable to support various tessellation algorithms
  - Fetches vertex index streams and sends addresses to shader core
  - Up to 16 Verts / cycle
- **Primitive block**
  - Sends post processed vertex addresses, near neighbor address and topological information to shader core
  - Up to 16 prims / cycle
- **Pixel generation block**
  - Contains the “classic” setup and scan converter
  - Sends data to interpolators before entering shader core
  - Also interfaces to depth to perform HiZ / Early Z checks
  - Up to 16 fragments / cycle
Flexibility in Tessellation Modes

Discrete: Integer subdivision level per primitive  
Continuous: floating point subdivision level per primitive

1-level Subdivision  
2-level Subdivision  
3-level Subdivision  
Subdivision level = 1.0  
1.0 < Subdivision level < 3.0  
3.0 < Subdivision level < 5.0

Adaptive: floating point subdivision level per edge of a primitive
Geometry Performance

Practical polygon throughput is determined by a number of factors:

- **Vertex fetch rate**
  - 1 vert/cycle on X1000

- **Vertex cache size & efficiency**
  - 4k/8k on X1000

- **Vertex shader performance**
  - Up to 8 vec4 on X1000

- **Geometry shader performance**

- **Geometry amplification capabilities**

- **Triangle setup rate**
  - 1 prim / cycle on X1000

HD 2000 series feature major improvements in all of these areas:

- Fetch up to 16 vertices per clock
- Up to 8x increase in vertex cache size vs. X1000 series
- Unified architecture can increase available vertex shader processing power by up to 10x X1000 series
- Programmable tessellation unit for accelerated geometry amplification
  - We sustain > 700Mprim/sec in demo
- Setup 1 triangle per clock cycle
Ultra-Threaded Dispatch Processor

- Main control for the shader core
  - All workloads are distributed into threads of 64 elements
  - We have 100’s of threads in flight, to hide latency of resources
    - This is the main way that GPUs and CPUs are different
    - When a thread requests a resource that takes indefinite time or a long time, the thread is slept until the request is returned

- A blend of arbitration and sequencing
  - Arbitration occurs for all resources
    - This occurs for input into the shader from the different streams of work coming in
      - We queue up each stream independantly, to control ratio of workloads
    - This occurs for ALU, all I/O (output, cache requests, export/import) and between parallel work loads
    - ALUs execute pairs of threads interleaved
  - Arbitration algorithms are programmable
    - Attempt to find the best one to minimize time spent in shader
    - Attempt to find the best one to maximize utilization
    - Current algorithms confidential
Ultra-Threaded Dispatch Processor

- Sequencing occurs for all shader instructions
  - All instructions execute in parallel and independently, including for each set of SIMDs
- Sequencing considerations
  - Multiple instruction streams in parallel
  - Virtualized instruction cache for all instructions
  - Multiple threads (100’s) executing shader clauses at the same time (each one locking down cache lines)
- Beyond shader instruction virtualization
  - Constant virtualization must also be done by this unit
- New UTDP design does all the arbitration, sequencing and virtualization requirements for R600
Ultra-Threaded Dispatch Processor

- Setup Engine
  - Vertex Assembler
  - Geometry Assembler
  - Interpolators

- Ultra-Threaded Dispatch Processor
  - Vertex Shader Command Queue
  - Geometry Shader Command Queue
  - Pixel Shader Command Queue

- Arbiter
- Sequencer

- SIMD Array
  - 80 Stream Processing Units

- Texture Fetch
  - Arbiter
  - Sequencer

- Vertex Fetch
  - Arbiter
  - Sequencer

- Shader Instruction Cache
- Shader Constant Cache
Stream Processing Units

4 SIMD Arrays of 16 SPUs

SPU arranged as 5-way scalar shader processor

- Co-issue up to 5 scalar FP MAD (Multiply-Add) instructions per clock
- One of the 5 stream processing units handles transcendental instructions as well (SIN, COS, LOG, EXP, etc.), on top of regular FPMAD
- 32-bit floating point precision
- Mul and Add are 1/2ulp IEEE RTNE
- Up to 5 integer operations also supported (cmp, add) and 1 integer multiplier

Branch execution units handle flow control and conditional operations

- Condition code generation for full branching
- Predication supported directly in ALU

General Purpose Registers

- Multi-ported but shared among the processors

Issuing

- Each SPU works on 4 separate elements, issuing 5 scalar instructions over 4 cycles
- BEU is issued to separately
Texture Unit Design

1, 2 or 4 texture units
- 8 Texture 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)
  - Can bilinear filter one 64-bit FP color value per clock, 128b FP per 2 clocks

Multi-level texture cache design
- Large, shared L2 cache stores data retrieved on L1 cache misses
  - Unified 16k or 32k L1 texel cache
  - 0 or 32k structure cache (unfiltered)
  - 0, 128KB or 256KB L2 per model
Texture Unit Features

- **Focus on floating point textures**
  - 64-bit HDR textures bilinear filtered at full speed (~7x faster than Radeon X1000 series)
  - 128-bit floating point textures filtered at half speed
  - Support for new HDR (RGBE) texture format
- **Large L2**
  - Improves performance for large textures and/or wide pixels
  - Performance focus for large kernel filters
  - Performance focus for large data sets, such as matrices, etc...
- **Depth Stencil Textures (DST) with Percentage Closer Filtering (PCF)**
  - High performance soft shadow rendering
- **High resolution texture support**
  - Up to 64 megatexels (8192 x 8192)
- **Full texturing capability accessible to all shader types, equally**
- **New memory tiling structures for various textures**
  - Improved volumetric texture performance
  - With advanced and programmable memory controller, performance options for texture surfaces are quite numerous
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)
  - Can export data to stream out buffer
  - 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

- Uses include:
  - Interthread communication
  - Render to vertex buffer
  - GPR overflow
  - Overflow storage for GS data (since it can generate widely variable amounts of output data)
Render Back-Ends

Double rate depth/stencil test
- 32 pixels per clock for HD 2900
- 8 pixels per clock for HD 2600 & HD 2400
- New features such as Re-Z and HiStencil

Fast Post-Processing Effects
- Render-to-texture much more efficient than previous chips

MSAA resolve functionality is programmable
- Makes fast Custom Filter AA possible

New blendable surface formats
- Allows new DX10 formats to be displayable
  - 128-bit floating point format
  - 11:11:10 floating point format

MRT (Multiple Render Target) support
- Up to 8 MRTs (double Radeon X1000 series) with MSAA support
New CFAA – Custom filter anti-aliasing

• Anti-aliasing is about higher sample rates and resolve
  • Both need to improve to give better results

• Resolve is a post-processing filter applied to each rendered frame
  • Low pass filter designed to help remove high frequency overlapping patterns
  • These appear as jagged edges and shimmering in image

• How we can do better
  • Programmable fragment positions
  • Non-uniform sample weights
  • Sampling from outside pixel boundaries
  • Filter kernels that adapt to the characteristics of each pixel

• Benefits and advantages
  • Software upgradeable
  • Can be used to enhance in-game AA settings for most DirectX 9 titles
  • Works together with all other ATI Radeon AA features
  • Works with HDR
  • Works with stencil shadows
  • More samples per pixel than MSAA without increasing memory footprint
Custom Filter Anti-Aliasing (CFAA)

12x CFAA Narrow Tent Filter
Custom Filter Anti-Aliasing (CFAA)

Adaptive Edge Detect Filter
- Performs edge detection pass on rendered image
- Edge pixels resolved using more samples along direction of edge with high quality filter
- Other pixels resolved using fewer samples and box filter

Benefits
- Provides excellent edge smoothing where it’s needed the most
- Reduces texture shimmering
- Avoids blurring of fine detail (e.g. small text)
- Provides better quality per sample than supersampling, with better performance
Memory Interface and controller

Packing 512b onto the die
- New, compact, stacked I/O pad design
- Double the I/O density of previous designs

- New double ringbus
  - Have 256b x 4 (1k) sets of bidirectional ring busses for read and write data packets
  - Distributed arbitration for channels
  - Full TLB translation layers per client
  - Done to improve routing, easier physical layout, minimal latency and great scalability

Benefits of a 512-bit interface
- More bandwidth with existing memory technology
- Lower memory clock required to achieve target bandwidth
- Improved cost:bandwidth ratio
<table>
<thead>
<tr>
<th></th>
<th>Rage Pro</th>
<th>Rage 128</th>
<th>Radeon 8500</th>
<th>Radeon 9700 Pro</th>
<th>Radeon 9800 XT</th>
<th>Radeon X850 XT Platinum Edition</th>
<th>Radeon X1800 XT</th>
<th>Radeon X1950 XTX</th>
<th>Radeon HD 2900 XT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Year</strong></td>
<td>1998</td>
<td>1999</td>
<td>2000</td>
<td>2001</td>
<td>2002</td>
<td>2003</td>
<td>2004</td>
<td>2005</td>
<td>2006</td>
</tr>
<tr>
<td><strong>Transistor Size</strong></td>
<td>350 nm</td>
<td>250 nm</td>
<td>180 nm</td>
<td>150 nm</td>
<td>150 nm</td>
<td>130 nm</td>
<td>90 nm</td>
<td>90 nm</td>
<td>80nm</td>
</tr>
<tr>
<td><strong>Transistor Count</strong></td>
<td>5 million</td>
<td>13 million</td>
<td>30 million</td>
<td>60 million</td>
<td>110 million</td>
<td>160 million</td>
<td>321 million</td>
<td>384 million</td>
<td>700 million</td>
</tr>
<tr>
<td><strong>Clock Speed</strong></td>
<td>75 MHz</td>
<td>100 MHz</td>
<td>183 MHz</td>
<td>275 MHz</td>
<td>325 MHz</td>
<td>412 MHz</td>
<td>550 MHz</td>
<td>625 MHz</td>
<td>650 MHz</td>
</tr>
<tr>
<td><strong>Rendering Pipelines / Shader Processors</strong></td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>16</td>
<td>48</td>
<td>64</td>
</tr>
<tr>
<td><strong>Memory Bandwidth (GB/sec)</strong></td>
<td>0.6</td>
<td>1.6</td>
<td>5.9</td>
<td>8.8</td>
<td>19.8</td>
<td>23.4</td>
<td>37.8</td>
<td>44.8</td>
<td>64.0</td>
</tr>
<tr>
<td><strong>Pixel Shading / Fill Rate (Mpixels/sec)</strong></td>
<td>75</td>
<td>200</td>
<td>366</td>
<td>1100</td>
<td>2600</td>
<td>3300</td>
<td>8800</td>
<td>10000</td>
<td>31200</td>
</tr>
<tr>
<td><strong>Vertex Processing (Mvertices/sec)</strong></td>
<td>4</td>
<td>8</td>
<td>30</td>
<td>69</td>
<td>325</td>
<td>412</td>
<td>825</td>
<td>1250</td>
<td>1300</td>
</tr>
<tr>
<td><strong>System Bus Bandwidth (GB/sec)</strong></td>
<td>0.53</td>
<td>1.06</td>
<td>1.06</td>
<td>1.06</td>
<td>2.11</td>
<td>2.11</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td><strong>DirectX Version and Shader Model Support</strong></td>
<td>DirectX 5</td>
<td>DirectX 6</td>
<td>DirectX 7</td>
<td>DirectX 8.1 (SM1.4)</td>
<td>DirectX 9.0 (SM2.0)</td>
<td>DirectX 9.0 (SM2.0b)</td>
<td>DirectX 9.0 (SM3.0)</td>
<td>DirectX 9.0 (SM3.0)</td>
<td>DirectX 10.0 (SM4.0)</td>
</tr>
<tr>
<td><strong>Features</strong></td>
<td>Hardware Triangle Setup</td>
<td>128-bit memory interface, AGP 4X</td>
<td>Hardware transform &amp; lighting, DDR memory support</td>
<td>Programmable Shaders, Higher Order Surfaces</td>
<td>Floating Point Processing, 256-bit memory interface, AGP 8X</td>
<td>Unlimited shader instructions, 256MB DDR2 memory support</td>
<td>PCI Express x16, GDDR3 memory support</td>
<td>Dynamic shader flow control, 128-bit precision, HDR output</td>
<td>GDDR4 memory support</td>
</tr>
</tbody>
</table>
What comes next?

• Lots and Lots...
  – (Tons of tuning for current architecture! – But I’m HW!)
  – DDR5 is around the corner
  – Enhancements for the emerging GPGPU fields
    • More precision
    • New APIs, new functions
  – New technologies such as 65, 55, 45, 32...
    • New technology introduction rate is still very high
  – Graphics and gaming keeps on evolving
    • DX-next is already being discussed
    • We are well into next generation and next-next generations...
ATI Radeon HD 2000 Series

- Brand new architecture, with a significant set of new features and abilities
- Focused on shader core with floating point data sets
- Full DX10 functionality as core to the design
- Available at multiple price points, from below $99 to ~$400
- A lot of work, by over 300 engineers, over 4 years...
Questions And demo!

- Note: Resumes for jobs or internships:
  
  eric.demers@amd.com (HW)
  
or raja.koduri@amd.com (Apps/3D Research)