GPU Architecture

Michael Doggett
ATI
GPU Architecture

- RADEON X1800/X1900
- Microsoft’s XBOX360
  - Xenos GPU
- GPU research areas
• Products from cell phones to super computers
RADEON X1800/X1900
• November ‘05
• DirectX 9.0 Shader Model 3.0
• Core clock 625MHz
• Memory clock 750MHz
• 512MB of graphics memory
• 321 Million transistors, 288mm², 90nm
• 8 Vertex Shaders
• 16 Pixel Shaders
  • 32bit IEEE Single Precision float
  • 4 quad based, threaded units
  • 512 threads of 16 pixels allow efficient dynamic branching
Vertex Shader
Pixel Shader

- **ALU 1**
  - 1 Vec3 ADD + Input Modifier
  - 1 Scalar ADD + Input Modifier

- **ALU 2**
  - 1 Vec3 ADD/MULL/MADD
  - 1 Scalar ADD/MULL/MADD

- **Branch Execution Unit**
  - 1 Flow Control Instruction
Thread Size and Dynamic Branch Efficiency

- Efficient
- Less Efficient
- Inefficient (Does not take advantage of Dynamic branching)

Sample Shader

```latex
if (shadow) {
    Process Shadow
    // X clocks
} else {
    Process light
    // Y clocks
}
```
RADEON X1900 XTX

- 48 shader pipes
- Core clock 650MHz
- Memory clock 775MHz
- 512MB of graphics memory
- 384 Million transistors, 352mm², 90nm
Xenos: XBOX360 GPU
System architecture

CPU

GPU
Northbridge

UNIFIED
MEMORY

Southbridge

2x PCIE
500MB/s

2x 10.8 GB/s

22.4GB/s
700MHz
128bit
GDDR3

32GB/s

DAUGHTER
DIE

2x 10.8 GB/s
Rendering performance

- GPU to Daughter Die interface
  - 8 pixels/clk
    - 32BPP color
    - 4 samples Z - Lossless compression
  - 16 pixels/clk – Double Z
    - 4 samples Z - Lossless compression
Rendering performance

- Alpha and Z logic to EDRAM interface
  - 256GB/s
  - Color and Z - 32 samples
    - 32bit color, 24bit Z, 8bit stencil
  - Double Z - 64 samples
    - 24bit Z, 8bit stencil

**DAUGHTER DIE**

- 8pix/clk, 4x MSAA, Stencil and Z test, Alpha blending
- 256GB/s
- 10MB EDRAM
GPU architecture

- **Unified Shader**
- **GPU architecture**
- **Vertex Pipeline**
- **Pixel Pipeline**
- **Display Pixels**
- **Primitive Setup**
- **Clipper**
- **Rasterizer**
- **Hierarchical Z/S**
- **Index Stream**
- **Generator**
- **Tessellator**
- **Output Buffer**
- **Texture/Vertex Fetch**
- **Memory Export**
- **UNIFIED MEMORY**
- **DAUGHTER DIE**
Unified Shader

- A revolutionary step in Graphics Hardware
- One hardware design that performs both Vertex and Pixel shaders
- Vertex processing power
Unified Shader

- GPU based vertex and pixel load balancing
  - Better vertex and pixel resource usage
- Union of features
  - E.g. Control flow, indexable constant, ...
- DX9 Shader Model 3.0+

3 SIMDs

48 ALUs

Vertices  Pixels

Vec4  Scalar
Memory Export

- Shader output to a computed address
- Virtualize shader resources - multipass
- Shader debug
- Randomly update data structures from Vertex or Pixel Shader
- Scatter write
• Shader fetch can be either:
  • Texture fetch (16 units)
    • LOD computation
    • Linear, Bi-linear, Tri-linear Filtering
    • Uses cache optimized for 2D, 3D texture data with varying pixel sizes
    • Unified texture cache
  • Vertex fetch (16 units)
    • Uses cache optimized for vertex-style data
GPU research areas
GPU research areas

- Graphics can always use increased processing power
  - Higher screen resolution
  - More complex shaders
  - Compute in shader and save on memory bandwidth
  - Product differentiator
- GPU measured on bang for buck
  - Performance/mm²
- Rapid feature set change
  - DX10 coming with Microsoft Windows Vista, end ’06
    - Less optional features
• How to use more replicated tiles with only local interconnect?
  • Saves on chip level interconnect
  • Improves floorplanning
  • A RAW-style architecture for graphics

• Multi-GPU
  • Current solutions split workload between multiple GPUs
    • By frame
    • Single frame tiling
    • Combinations
  • What alternative architectures can single GPUs scale up to?
<table>
<thead>
<tr>
<th>Cloth</th>
<th>Smoke</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Cloth Example" /></td>
<td><img src="image2" alt="Smoke Example" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rigid Body Collisions</th>
<th>Water</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="Rigid Body Example" /></td>
<td><img src="image4" alt="Water Example" /></td>
</tr>
</tbody>
</table>
• Which applications run well on the GPU?
• What modifications to GPU architecture enable more General Purpose computing?
  • Full IEEE754 compliance
  • 64bit floats
  • Scatter, Xenos Memory Export
Rendering + Physics
GPU research areas

- GPU/CPU convergence
- Higher Order Surfaces
  - DX10 geometry shader allows access to triangle’s neighboring vertices
  - Mesh modification on GPU