Today: Graphics Processors

Computer Graphics. A brief introduction to “the pipeline”.

Stream Processing. Casting the graphics pipeline into hardware.

Unified Pipelines. GeForce 8800, the new architecture from Nvidia.
AGP 4X: Hi-Speed Graphics Bus

Dedicated Graphics RAM

DDR SDRAM DIMM slot

167 MHz Memory bus

PowerPC G4 microprocessor (L2 cache: 512K 1:1)

167 MHz MaxBus

AGP 4X bus

Radeon 9200 graphics IC

32 MB DDR RAM

To Display

DVI/VGA/composite/S-video output port

FireWire 400 port

Ethernet port 10/100 Mbps

Average selling price (ASP) for GPUs: $30

ATI Radeon 9200: Graphics Processing Unit (GPU)

Recall: Mac Mini G4 System Diagram

CS 152 L20: Buses, Disks, and RAID

© 2005 Apple Computer, Inc. All Rights Reserved.
About 12 MB/frame (24-bit pixels)
24 frames/sec: 300 MB/second
A “dumb” graphics card ...

AGP 4X: 1.1 GB/s. Can handle 24 f/s (300 MB/s) for a 2560x1600 display.

Problem: CPU has to compute a new pixel every 10 ns. 10 clock cycles for a 1 GHz CPU clock.

Control logic sends “this frame” out of other buffer to display.

CPU writes “next frame” in one buffer.

Double Buffering:
Q. What kind of graphics are we accelerating?
A. In 2006, interactive entertainment (3-D games). In the 1990s, 2-D acceleration (fast windowing systems, games like Pac-Man).

Graphics Acceleration

Q. In a multi-core world, why should we use a special processor for graphics?
A. Programmers generally use a certain coding style for graphics. We can design a processor to fit the style.

Next: An intro to 3-D graphics.
The Triangle ...

Simplest closed shape that may be defined by straight edges.

With enough triangles, you can make anything.
A cube whose faces are made up of triangles. This is a 3-D model of a cube -- model includes faces we can’t see in this view.

A sphere whose faces are made up of triangles. With enough triangles, the curvature of the sphere can be made arbitrarily smooth.
A teapot (famous object in computer graphics history). A “wire-frame” of triangles can capture the 3-D shape of complex, man-made objects.
Triangle defined by 3 vertices

By transforming \((v' = f(v))\) all vertices in a 3-D object (like the teapot), you can move it in the 3-D world, change it’s size, rotate it, etc.

vertex \(v_0 = (x_0, y_0, z_0)\)

vertex \(v_1 = (x_1, y_1, z_1)\)

vertex \(v_2 = (x_2, y_2, z_2)\)

If a teapot has 10,000 triangles, need to transform 30,000 vertices to move it in a 3-D scene ... per frame!
Vertex can have color, lighting info ...

If vertices colors are different, this means that a smooth gradient of color washes across triangle.

vertex $v_0 = (r_o, g_o, b_o)$

vertex $v_1 = (r_1, g_1, b_1)$

vertex $v_2 = (r_2, g_2, b_2)$

More realistic graphics models include light sources in the scene. Per-vertex information can carry information about how light hits the vertex.
We see a 2-D window into the 3-D world.
From 3-d triangles to screen pixels

First, **project** each 3-D triangle that might “face” the “eye” onto the **image plane**.

Then, create “**pixel fragments**” on the **boundary** of the image plane triangle.

Then, create “**pixel fragments**” to **fill in** the triangle (rasterization).

**Why “pixel fragments”?** A screen pixel color might depend on many triangles (example: a glass teapot).
Process each fragment to “shade” it.

Algorithmic approach: Per-pixel computational model of metal and how light reflects off of it. Move teapot and what reflects off it changes.
Process each fragment to “shade” it.

Artistic approach: Artist paints surface of teapot in Photoshop. We “map” this “texture” onto each pixel fragment during shading.

Final step: Output Merge. Assemble pixel fragments to make final 2-d image pixels.
Applying texture maps: Quality matters

“Good” algorithm. B and C look blurry.

“Better” algorithm. B and C are detailed.
Putting it All Together ...

*Luxo, Jr*: Short movie made by Pixar, shown at SIGGRAPH in 1986. First Academy Award given to a computer graphics movie.
The graphics pipeline in hardware (2004)

Vertex “stream” sent by CPU

Process each vertex

Create pixels fragments

Process pixel fragments

Output Merge

Programmable CPU “Vertex Shader”

Programming Language/API? DirectX, OpenGL

Programmable CPU “Pixel Shader”

To display

 Algorithms are usually hardwired

PowerPC G4 microprocessor (L2 cache: 512K 1:1)

167 MHz MaxBus

AGP 4X bus

DVI/VGA/composite/S-video output port
Vertex Shader: A “stream processor”

Vertex “stream” from CPU

From CPU: changes slowly (per frame, per object)

Input Registers (Read Only)

Constant Registers (Read Only)

Working Registers (Read/Write)

Shader CPU

Shader creates one vertex out for each vertex in.

Output Registers (Write Only)

Vertex “stream” ready for 3-D to 2-D conversion

Shader Program Memory

Short (ex: 128 instr) straight-line code. Same code runs on every vertex.

Only one vertex at a time placed in input registers.
Optimized instructions and data formats

128-bit registers, holding four 32-bit floats.

Input Registers

\[
\begin{array}{cccc}
x & y & z & w \\
\end{array}
\]

Typical use: \((x, y, z, w)\) representation of a point in 3-D space.

Shader CPU

Typical instruction:

\[
\text{rsq } \text{dest } \text{src}
\]

\[
dest.\{x, y, z, w\} = 1.0/\sqrt{\text{abs}(src.w)}.
\]

If \(src.w=0\), \(dest \to \infty\).

Output Registers

\[
\begin{array}{cccc}
x & y & z & w \\
\end{array}
\]

To 3-D/2-D

The \(1/\sqrt{()}\) function is often used in graphics.
Easy to parallelize: Vertices independent

From CPU

Input Registers
\[ x \quad y \quad z \quad w \]

Shader CPU

Output Registers
\[ x \quad y \quad z \quad w \]

Why?
3-D to 2-D may expect triangle vertices in order in the stream.

Caveat:
Care might be needed when merging streams.

To 3-D/2-D
Pixel shader specializations ...

Texture maps (look-up tables) play a key role.

Pixel shader needs fast access to the map of Europe on teapot (via graphics card RAM).
Pixel Shader: Stream processor + Memory

Pixel fragment stream from rasterizer

Indices into texture maps.

Engine does interpolation.

Texture Registers

Indices into texture maps.

Shader CPU

Shader creates one fragment out for each fragment in.

Registers (Read/Write)

Register R0 is pixel fragment, ready for output merge.

From CPU: changes slowly (per frame, per object)

Input Registers (Read Only)

Only one fragment at a time placed in input registers.

Constant Registers (Read Only)

Texture Engine

Memory System
Recent Design: Nvidia GeForce 7900

278 Million Transistors, 650 MHz clock, 90 nm process

- **Vertex Shaders**: 8
- **Pixel Shaders**: 24
- **Output Merge Units**: 
- **Texture Cache**: 
- **3-D to 2-D**: 

CS 152 L22: Graphics Processors

UC Regents Fall 2006 © UCB
Basic idea: Replace specialized logic (vertex shader, pixel shader, hardwired algorithms) with many copies of one unified CPU design.

Unified Architectures

Consequence: You no longer “see” the graphics pipeline when you look at the architecture block diagram.

Designed for: DirectX 10 (Microsoft Vista), and new non-graphics markets for GPUs.
DirectX 10 (Vista): Towards Shader Unity

Earlier APIs: **Pixel and Vertex CPUs very different ...**

<table>
<thead>
<tr>
<th>Feature</th>
<th>1.1 2001</th>
<th>2.0 2002</th>
<th>3.0 2004*</th>
<th>4.0 2006</th>
</tr>
</thead>
<tbody>
<tr>
<td>instruction slots</td>
<td>128</td>
<td>256</td>
<td>≥512</td>
<td>≥64K</td>
</tr>
<tr>
<td></td>
<td>4+8‡</td>
<td>32+64‡</td>
<td>≥512</td>
<td></td>
</tr>
<tr>
<td>constant registers</td>
<td>&gt;96</td>
<td>&gt;256</td>
<td>&gt;256</td>
<td>16x4096</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>32</td>
<td>224</td>
<td></td>
</tr>
<tr>
<td>tmp registers</td>
<td>12</td>
<td>12</td>
<td>32</td>
<td>4096</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>input registers</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>4+2§</td>
<td>8+2§</td>
<td>10</td>
<td>32</td>
</tr>
<tr>
<td>render targets</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>samplers</td>
<td>8</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>textures</td>
<td>8</td>
<td>16</td>
<td>4</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2D tex size</td>
<td></td>
<td>2Kx2K</td>
<td>8Kx8K</td>
<td></td>
</tr>
<tr>
<td>integer ops</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>load op</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>sample offsets</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>transcendental ops</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>op</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>derivative op</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>flow control</td>
<td></td>
<td>static</td>
<td>stat/dyn</td>
<td>dynamic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>stat/dyn</td>
<td></td>
</tr>
</tbody>
</table>

**Table 1: Shader model feature comparison summary.**

---

UC Regents Fall 2006 © UCB
Geometry Shader: Lets a shader program create new triangles.

Also: Shader CPUs are more like RISC machines in many ways.

Stream Output: Lets vertex stream recirculate through shaders many times ... (and also, back to CPU)
Why? Particle systems ...
NVidia 8800: Unified GPU, Announced last week

Thread processor sets shader type of each CPU

128 Shader CPUs

Streams loop around...

1.35 GHz Shader CPU Clock, 575 MHz core clock
Graphics-centric functionality ...

Texture engine and memory system

3-D to 2-D (vertex to pixel)

Graphics-centric functionality ...

3-D to 2-D (vertex to pixel)
Can be reconfigured with graphics logic hidden ...


1000s of active threads

3 TeraFlops Peak Performance Ships with a C compiler.

Texture system set up to look like a conventional memory system (768MB GDDR3, 86 GB/s)
Chip Facts

90nm process
681M Transistors
80 die/wafer (pre-testing)

Design Facts

4 year design cycle
$400 Million design budget

600 person-years: 10 people at start, 300 at peak

A big die. Many chips will not work (low yield). Low profits.
Some products are “loss-leaders”

Breakthrough product creates “free” publicity you can’t buy.

(1) When chip is “shrunk” to 65nm fab process, die will be smaller, yields will improve, profits will rise.

(2) Simpler versions of the design will be made to create an entire product family, some very profitable.

“We tape out a chip a month”, NVIDIA CEO quote.
GeForce 8800 GTX Card: $599 List Price

PCI-Express 16X Card - 2 Aux Power Plugs!

185 Watts Thermal Design Point (TDP) -- TDP is a “real-world” maximum power spec.
Dustbuster-style fan to move 185 Watts
Face was “scanned” to create a vertex model. 8800 GTX was used to do skin, eye, lips and hair rendering.
Create standard model from common practice: Wire-frame geometry, triangle rasterization, pixel shading.

Put model in hardware: Block diagram of chip matches computer graphics math.

Evolve to be programmable: At some point, it becomes hard to see the math in the block diagram.

“Wheel of reincarnation” -- Hardwired graphics hardware evolves to look like general-purpose CPU. EECS visitor Ivan Sutherland co-wrote a paper on this topic in 1968!
Reminder: Final Checkoff this Friday!

Final report due following Monday, 11:59 PM

TAs will provide “secret” MIPS machine code tests.

Bonus points if these tests run by end of section. If not, TAs give you test code to use over weekend

Mid-term, group talks after Thanksgiving