Unified Shader Design

Lan-Da Van (范倫達), Ph. D.
Department of Computer Science
National Yang Ming Chiao Tung University
Hsinchu, Taiwan

Spring, 2021
Source from:

A 195 mW, 9.1 Mvertices/s Fully Programmable 3-D Graphics Processor for Low-Power Mobile Devices

Outline

- Introduction
- 3-D Graphics Processor
  - Bandwidth Reduction
  - Pixel-Vertex Multi-Threading
- Mobile Unified Shader
  - Internal Architecture
  - Low-Power Lighting Engine
  - Unified Shader Datapath
  - Micro-Level Power Management
- Results
- Conclusion
Introduction

- Low-power & small-area for mobile devices.
- Fully programmable 3-D graphics.
- Single unified shader architecture.
3-D Graphics Processor
3-D Graphics Processor

• ISSUE:
  – The mobile unified shader uses floating-point matrices, which take thousands of cycles to generate on an embedded RISC processor. The cycle overhead limits the 3-D graphics performance.
3-D Graphics Processor

- **SOLUTION:**
  - Employ the matrix generator.
  - It performs floating-point matrix operation
    - addition, multiplication, inversion, transposition, rotation, deterministic, and so on.
  - All those matrix computation is controlled by the host RISC processor using the dedicated matrix generator instructions.
  - The workload of 3-D graphics pipeline is well distributed to the RISC processor and 3-D graphics processor.
ISSUE:

– The 3-D graphics processor is located in a mobile multimedia SoC as an IP.
– The 3-D graphics processor shares the bus bandwidth with the other components.
– The whole system performance is dependent on the communication cycles for transferring the graphics data between the external memory and the graphics hardware.
• ISSUE:
  – The 3-D graphics processor fetches the required data through AMBA bus and it consumes 696 MB/s bus bandwidth.
    • 216 MB/s for vertex attributes
    • 480 MB/s for rendering data.
  – The AMBA bus which has 400 MB/s capacity can allow less than 120 MB/s for 3-D graphics processor.
  – This bandwidth mismatch causes graphics performance degradation.
Bandwidth Reduction (3/6)
• **SOLUTION:**
  – Pre/post vertex cache
    • The graphics data has high data-locality.
    • Due to the triangle strip and fan arrangement
  – 4:1 texture compression (skip here)
  – 2-D direct mapped cache (skip here)
    • Up to 98% hit ratio in real 3-D applications
  – Texture cache
• SOLUTION:
  – With those techniques, the rendering data transaction is reduced to 91.2 MB/s.
  – As a result, the 3-D graphics processor consumes less than 120 MB/s and it provides its peak-performance in the mobile multimedia SoC.
Bandwidth Reduction (6/6)
• ISSUE:
  – Since there is only one processing unit, the graphics data traverse the unified shader twice in a single graphics pipeline.
  – 3-D graphics performance is highly related to utilization of the unified shader.
  – How to utilize the unified shader effectively?
• **SOLUTION:**

  – Pixel-vertex multi-threading (PVMT)
    
    • When the texture cache miss occurs during per-pixel operations and vertex buffer contains vertices to be computed, PVMT issues next vertices and SIMD datapath and SFU perform per-vertex operations until the texture cache filling is finished.

  – The content characteristics such as texture cache miss rate or vertex/pixel ratio highly affect the effect of the PVMT.

  – By employing PVMT with 5 entries vertex buffer, 94% of the vertex operations are interleaved into the pixel operations with only 6% performance degradation.
Pixel-Vertex Multi-Threading (3/4)
Pixel-Vertex Multi-Threading (4/4)
Outline

• Introduction
• 3-D Graphics Processor
  – Bandwidth Reduction
  – Pixel-Vertex Multi-Threading
• Mobile Unified Shader
  – Internal Architecture
  – Low-Power Lighting Engine
  – Unified Shader Datapath
  – Micro-Level Power Management
• Results
• Conclusion
Internal Architecture
# Internal Architecture

<table>
<thead>
<tr>
<th>F</th>
<th>General SIMD INSTR Fetch</th>
<th>Graphics INSTR Fetch</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>Final INSTR Decoding</td>
<td>Const. Reg. (SRAM) Read</td>
</tr>
<tr>
<td>E1</td>
<td>SIMD ALU Pipeline</td>
<td>SIMD MUL Pipeline</td>
</tr>
<tr>
<td></td>
<td>4-way 32b Floating ALU</td>
<td>4-way 32x16 F-P MUL</td>
</tr>
<tr>
<td></td>
<td>4-way 32b Integer ALU</td>
<td>4-way 32x16 Integer MUL</td>
</tr>
<tr>
<td>E2</td>
<td>4-way 32b Floating ALU</td>
<td></td>
</tr>
<tr>
<td>E3</td>
<td>Pipeline Register (E3-E4)</td>
<td>4-way 32b Floating ADD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4-way 32b Integer ADD</td>
</tr>
<tr>
<td>E4</td>
<td>Pipeline Register (E4-W)</td>
<td>4-way 32b Floating ADD</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4-way 32b Integer ADD</td>
</tr>
<tr>
<td>W</td>
<td></td>
<td>All Register Files Writeback</td>
</tr>
</tbody>
</table>
Low-power Lighting Engine

\[ \text{Color}_{\text{ORD}} = C_{\text{amb}} + \{(N' \cdot L_{\text{dir}}) \times C_{\text{diff}}\} \\
\quad + \{(N' \cdot H)^{C_{\text{pow}}} \times C_{\text{spec}}\} \]

\[ \text{Color}_{\text{LNS}} = C_{\text{amb}} + \{(N' \cdot L_{\text{dir}}) \times C_{\text{diff}}\} \\
\quad + 2\left\{\log_2 (N' \cdot H) \times C_{\text{pow}} + \log_2 C_{\text{spec}}\right\} \]
Low-power Lighting Engine

(a)
Low-power Lighting Engine

Conventional
- DP3
  - Result1.x, Ldir, N'
  - Result1.y, H, N'
  - Result1.w, C_POW,
- MOV

LIT
- Result2, Result1;
- Compute Lighting Values

MAD
- Result3, Coff_DIFF, Result2.y, Coff_AMB;
- Diffuse Light + Ambient Light
- Output_Color.x, y, z, Coff_SPEC, Result2.z, Result3;
- + Specular Light
- Data Dependency

Proposed
- DP3
  - Result1.x, Ldir, N'
  - Result1.y, H, N'
- MOV
  - Result1.w, Coff_POW;
  - Result2.x, Coff_AMB;
  - Result2.y, Coff_DIFF;
- TLT
  - O[COL].xyz, Result1, C_SPEC, Result2;
- R1.x = Ldir \cdot N'
- R1.y = (H \cdot N')
- R1.w = Cpow
- R2.x = C_AMB
- R2.y = C_DIFF

(b)
Unified Adder

(a)
Unified Multiplier
Micro-level Power Management
Outline

• Introduction
• 3-D Graphics Processor
  – Bandwidth Reduction
  – Pixel-Vertex Multi-Threading
• Mobile Unified Shader
  – Internal Architecture
  – Low-Power Lighting Engine
  – Unified Shader Datapath
  – Micro-Level Power Management
• Experimental Results
• Conclusion
Chip and Demo Results
Comparison Results

**Vertex Fill Rate (Transform & One Specular Light)**
- Z-3D JSSC ’04 JSSC ’05 This Work
- 0.38 1.1 3.6 9.1

**Performance Index (Kvertices/s per mW)**
- Z-3D JSSC ’04 JSSC ’05 This Work
- 10 5.23 23.2 43.3

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation Frequency (MHz)</td>
<td>132/33</td>
<td>132/33</td>
<td>200</td>
<td>100</td>
</tr>
<tr>
<td>Vertex Fill Rate (Mvertices/s)</td>
<td>0.38</td>
<td>1.1</td>
<td>3.6</td>
<td>9.1</td>
</tr>
<tr>
<td>Power Consumption (mW)</td>
<td>38</td>
<td>210</td>
<td>155</td>
<td>195</td>
</tr>
<tr>
<td>Process Technology</td>
<td>0.18µm Logic</td>
<td>0.16µm DRAM</td>
<td>0.18µm Logic</td>
<td>0.13µm Logic</td>
</tr>
<tr>
<td>Programmability</td>
<td>N/A</td>
<td>N/A</td>
<td>Vertex Shading</td>
<td>Vertex Shading Pixel Shading</td>
</tr>
</tbody>
</table>
# Characteristic Results

## Feature Summary

<table>
<thead>
<tr>
<th>Implementation Results</th>
<th>Technology</th>
<th>0.13μm 7M CMOS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Power Supply</td>
<td>1.2V</td>
</tr>
<tr>
<td></td>
<td>Transistor Counts</td>
<td>1.3M logic gates</td>
</tr>
<tr>
<td></td>
<td>Die Size</td>
<td>3.3mm x 3.0mm</td>
</tr>
<tr>
<td></td>
<td>Frequency</td>
<td>100MHz</td>
</tr>
<tr>
<td></td>
<td>Power Consumption</td>
<td><strong>195mW @ Full 3D Graphics Processing</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ARM-9: 48mW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>System Peripherals: 22mW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mobile Unified Shader: 68mW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Vertex / Pixel / Fragment Generator: 57mW</td>
</tr>
<tr>
<td>Graphics Performance</td>
<td>Performance</td>
<td><strong>Vertex Fill Rate</strong>: 9.1Mvertices/s (w/ TnL)</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Raster Fill Rate</strong>: 100Mpixels/s</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Texel Fill Rate</strong>: 400Mtexels/s</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Pixel Fill Rate</strong>: 7.4Mpixels/s (with Vertex Shading, Pixel Shading)</td>
</tr>
<tr>
<td></td>
<td>Programmability</td>
<td>Programmable Vertex Shader</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Programmable Pixel Shader</td>
</tr>
<tr>
<td></td>
<td>Standard API</td>
<td>OpenGL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OpenGL</td>
</tr>
<tr>
<td></td>
<td>Screen Resolution</td>
<td>Up to 1024 x 1024 pixels</td>
</tr>
</tbody>
</table>
Conclusion

• Propose low-power fully programmable 3D graphics unified shader engine for the mobile device
  – 35% area reduction
  – 28% power reduction
  – Vertex fill rate: 9.1 Mvertices/s (w/ TnL)

• Propose the PVMT scheme
  – 94% of the vertex operations are interleaved into the pixel operations.