GPU-CPU Multi-Core For Real-Time Signal Processing

Saraju P. Mohanty
Department of Computer Science and Engineering
University of North Texas, Denton, TX 76203.
Email-ID: saraju.mohanty@unt.edu

Abstract

• Modern graphics cards are supported with powerful computational facilities for fast computation of vertex geometry and realistic rendering of 3D.
• Introduction of programmable pipeline in the graphics processing units (GPU) has enabled configurability.
• GPU which is available in every computer has a tremendous feat of highly parallel SIMD processing, but its capability is often under-utilized.
• We analyze computation of general algorithms on GPU.

Introduction and Motivation

• Modern GPU architectures are providing ways of configuring the graphics pipeline.
• We want to examine power of the GPUs for general purpose computing.
• GPUs are designed to perform a specific set of operations on large amounts of data.
• In last decade the computing power of the GPUs is increasing much faster than Moore’s law.
• The performance gap of GPU and CPU is widening.
• It is shown that a GeForceFX 5900 processor operating at 20 GigaFlops is equivalent to a 10GHz Pentium 4 processor.

Graphic Processing Unit (GPU) – A Broad View

• The architecture of GPU offers a large degree of parallelism at a relatively low cost through the well known vector processing model known as Single Instruction, Multiple Data (SIMD).
• GPU includes 2 types of processing units: vertex and pixel (or fragment) processors.
• The programmable vertex and fragment processors execute a user defined assembly level program having 4-way SIMD instructions.
• The vertex processor performs mathematical operations that transform a vertex into a screen position.
• This result is then pipelined to the pixel or fragment processor, which performs the texturing operations.

Conclusions

• GPU is an excellent candidate for performing data intensive signal processing algorithms.
• A direct application of this work is to perform DCT and IDCT on GPU in real-time signal processing to be used for broadcasting where real-time video compression is needed.
• The disadvantage in GPU is in the fact that the data transfer rate from the graphics hardware to the main memory is very slow.
• This bottleneck degrades the performance as it is needed to bring the results to the main memory in the general purpose computation.

An Application Scenario – Internet Protocol Television

• In IP-TV that broadcasts live events, a server containing GPU and CPU can perform tasks, like compression and copyright protection.
• The data, which arrives at shared memory, is directed by CPU and sent to GPU, then data is mapped into GPUs memory.
• The data, which arrives at shared memory, is directed by CPU and sent to GPU, then data is mapped into GPUs memory.

Experimental Results for Execution Time

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>CPU Time</th>
<th>GPU Time</th>
<th>GPU Release Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU only</td>
<td>4.37ms</td>
<td>Free</td>
<td>Fully Occupied</td>
</tr>
<tr>
<td>GPU only</td>
<td>44.57ms</td>
<td>Free</td>
<td>Fully Occupied</td>
</tr>
<tr>
<td>CPU + GPU</td>
<td>2.13ms</td>
<td>21.157ms</td>
<td>50.87% Free</td>
</tr>
</tbody>
</table>

Computations of Discrete Cosine Transformation (DCT) – A Case Study

• We used a 256×256 grayscale image to compute the DCT using OpenGL API. The input data are eight bit integers. The computation involves block-wise DCT of 8 × 8 pixels.
• The computation of DCT on GPU is performed using one or more rendering passes.
• The working of a rendering pass can be divided into 2 independent stages.
• 1st stage: A number of source textures, the associated vertex streams, render target, the vertex shader and pixel shader are specified.
• 2nd stage: It is the rendering stage which is triggered issuing the DrawPrimitives call.

IP-TV Broadcasting Showing a Shared Architecture for GPU-CPU Multi-Core Processing in a Video Server.

VLSI Design and CAD Laboratory (VDCL)