HPCG benchmark for characterizing the performance of SOC systems

Presenter: Rabi Javed Abbasi
Supervisor: Alistair Rendell
Contents

• Introduction
• Motivation and Objectives
• Methodology
• Challenges
• Questions
Introduction

• HPL: a **widely recognized** metric for ranking high performance computing systems.

• HPL performance is no longer strongly correlated to application performance.

• HPCG is an optimized implementation of PCG
Introduction (continued)

• HPCG contains essential computational and communication patterns that predict actual application performance.
• Type 1 + Type 2 patterns
• HPCG is a new benchmark with only limited implementations.
Motivation

• LINPACK Benchmark is 36 years old
• No longer so strongly correlated to real apps.
• Encourages poor choices in architectural features
• Benchmarking for days wastes a valuable resource
Motivation

- Ranking of computer systems using the new metric must correlate strongly to how our real applications would rank.
- Optimization of metric results for a particular platform, will lead to better performance in our real applications.
Objectives

• Develop robust implementations of HPCG that can be used to assess the performance and estimate the power usage of a variety of SOC devices.
Methodology

• Understanding the HPCG metric

1) Generate a synthetic symmetric positive definite (SPD) matrix
2) Set up data structures for the local symmetric Gauss-Seidel preconditioner.
3) Compute preconditions, post-conditions and invariants
4) Perform m iterations, n times, using the same initial guess each time
Methodology (continued)

- Implementation on Accelerator
  - OpenCL vs CUDA

<table>
<thead>
<tr>
<th>OpenCL</th>
<th>CUDA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Industry Wide</td>
<td>NVIDIA only</td>
</tr>
<tr>
<td>CPU Support</td>
<td>No CPU Support</td>
</tr>
<tr>
<td>Task parallel compute model</td>
<td>No native thread support</td>
</tr>
<tr>
<td>Requires greater programing effort</td>
<td>Relatively easier to program</td>
</tr>
</tbody>
</table>
Challenges

• Efficiently distributing the problem size onto a 3d domain.
• Managing data transfers from CPU to Device and back.
• Utilization of global vs local memory.
Questions ?