Overview

- Motivation
  - Computation intensive algorithms
  - Fine grained architectures
- Problem Definition
  - Resource to Performance Tradeoffs
    - Hardware/logic gates to performance tradeoffs
    - Memory to performance tradeoffs
Experimental Setup

- Computation intensive algorithms
  - Interpolation Kernels
- Fine Grained Architecture
  - FPGA
- Fine Grained Parallelism
  - Mitrion virtual processor
    - Extract fine grained parallelism
  - Mitrion-C high level language (HLL)
- Hardware Platform
  - Cray XD1 with Vertex-4

Mitrion Parallel Architecture

- Mitrion Virtual Processor (MVP)
  - Fine-Grained, Soft-Core Processor
  - Almost 60 IP blocks defined in HDL [1]
  - Non von-neuman architecture
- Mitrion-C
  - HLL for FPGA
  - Data dependence instead of order-of-execution
  - Parallelism Language Constructs [2]
  - Pipelining
Interpolation Kernels

- What is interpolation
  - Process of calculating new values within the range of available values [3]
- Cubic interpolation
- Bicubic interpolation
  - Applying cubic in 2D
  - 5 cubic kernels

Figure 1: 2D Interpolation

Main Idea

- Parallelism Levels
  - Bit Level Parallelism (BLP)
  - Kernel Level Parallelism (KLP)
  - Problem Level Parallelism (PLP)
- Mitrion SDK extracts maximum possible parallelism within kernel
  - Based on data dependence

Figure 2: Cubic interpolation kernel
Main idea (Conti.)

- Maximum parallelism at one level is not ultimate solution
  - Customized parallelism at different levels
  - Can better adjust Resource-performance tradeoffs
    - Gates-performance tradeoff
- Parallelization Levels
  - Single Kernel (SKZ)
  - Cross Kernel (CKZ)
  - Multi-SKZ
  - Multi/CKZ

![Parallelism and Parallelization Levels](image)

Parallelization Levels

- Single Kernel Parallelization (SKZ)
  - Only kernel level parallelism (KLP)
  - Manually define data independent block

![SKZ](image)
Parallelization Levels (Conti.)

- Cross Kernel Parallelization (CKZ)
  - Extend kernel by mixing more than one kernels
  - Replicate computation intensive data independent blocks
  - Resource computation balance

Figure 5: CKZ

Parallelization Levels (Conti.)

- Multi-SKZ
  - Replicate kernels which already have SKZ

Figure 6: Multi-SKZ
Parallelization Levels (Conti.)

- Multi-CKZ
  - Replicate kernels which already have CKZ

![Figure 6: Multi-CKZ]

Results

<table>
<thead>
<tr>
<th>Design Approach</th>
<th>Execution Time (ms)</th>
<th>Flip Flops (%)</th>
<th># Memory banks</th>
<th>p</th>
<th>r</th>
<th>k</th>
<th>m</th>
<th># Points computed in Parallel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKZ</td>
<td>10</td>
<td>20</td>
<td>1</td>
<td>6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>CKZ</td>
<td>2.5</td>
<td>68</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>4</td>
<td>-</td>
<td>4</td>
</tr>
<tr>
<td>Multi-SKZ</td>
<td>1.53</td>
<td>73</td>
<td>1</td>
<td>6</td>
<td>-</td>
<td>-</td>
<td>8</td>
<td>&lt; 8</td>
</tr>
<tr>
<td>Multi-CKZ</td>
<td>0.19</td>
<td>74</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>20</td>
<td>&lt; 80</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equidistant Bi-cubic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SKZ</td>
<td>10</td>
<td>4</td>
<td>1</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>CKZ</td>
<td>2.5</td>
<td>16</td>
<td>4</td>
<td>8</td>
<td>8</td>
<td>4</td>
<td>-</td>
<td>4</td>
</tr>
<tr>
<td>Multi-SKZ</td>
<td>0.33</td>
<td>43</td>
<td>1</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>47</td>
<td>&lt; 47</td>
</tr>
<tr>
<td>Multi-CKZ</td>
<td>0.17</td>
<td>55</td>
<td>4</td>
<td>8</td>
<td>6</td>
<td>4</td>
<td>24</td>
<td>&lt; 96</td>
</tr>
</tbody>
</table>

Table 1: Results
Conclusions

- Specific conclusions
  - For very limited resources, SKZ is better
  - CKZ is better for applications with high unbalanced computation distribution
  - SKZ and CKZ are better for large size applications
  - Multi-CKZ can provide high level of parallelism at cost of design complexity
  - Multi-SKZ and Multi-CKZ are attractive for small size Real-Time applications
- Using parallelization levels
  - Can adjust trade-offs
  - Can achieve highly custom parallelism
- Mix of parallelization levels can produce
  - Application-specific parallelism
  - Resource-specific parallelism

Future Work

- Automation of parallelization levels
- Parallelization levels to deal with other tradeoffs
- Generalized parallelization levels for all application
- Generalized parallelization levels for graphical processors to adjust tradeoffs
  - Floating point and accuracy
References


Tack
(Hope you enjoyed)