Multi-kernel Floorplanning for Enhanced CGRAs

Aaron Wood, Adam Knight, Benjamin Ylvisaker, Scott Hauck
Coarse-Grained Reconfigurable Architectures

- Lots of ALUs
- Word oriented interconnect
- Embedded memory
- Fine grained control
- FPGA style configurability
Time Multiplexing

- Small per cycle configurations
- Pipelined applications
- Scheduled interconnect
Multiple Kernels

- Physical regions
- Independent control domains
- Flow controlled inter-kernel links
- Composition of kernels
Resource Allocation

More resources = faster
Limited total resources
Maximize throughput
Resource Allocation
Digital camera pipeline example

Where do new resources go?

Cycles per sample

DC → INT → LPF → ED

10  12  10  6

Throughput
1 sample / 12 cycles
Resource Allocation
Digital camera pipeline example

Where do new resources go?

On the slowest kernel

Throughput
1 sample / 12 cycles
Resource Allocation
Digital camera pipeline example

Where do new resources go?

Throughput
1 sample / 12 cycles

Throughput
1 sample / 10 cycles

Cycles per sample
10 12 10 6
8 10 6
Resource Allocation
Digital camera pipeline example

Where do new resources go?

Throughput
1 sample / 12 cycles

Cycles per sample:
DC  INT  LPF  ED
10   12   10   6

Or set of slowest kernels

Throughput
1 sample / 10 cycles

Cycles per sample:
DC  INT  LPF  ED
10   8    10   6
Resource Allocation
Digital camera pipeline example

Where do new resources go?

Throughput
1 sample / 10 cycles

Cycles per sample
10 8 10 6

Throughput
1 sample / 12 cycles

Cycles per sample
10 12 10 6

Throughput
1 sample / 10 cycles

Cycles per sample
10 8 10 6

Throughput
1 sample / 8 cycles

Cycles per sample
7 8 6 6
Resource Allocation
Digital camera pipeline example

Where do new resources go?

Cycles per sample

DC → INT → LPF → ED
10

DC → INT → LPF → ED
10

DC → INT → LPF → ED
7

Throughput:
1 sample / 4 cycles
Resource Allocation
PET Event Detection Example

Where do new resources go?

Threshold → Math

Cycles per sample

Threshold: 8
Math: 19

Throughput:
1 sample / 8 cycles
What happened to the slowest kernel?

**Resource Allocation**

**PET Event Detection Example**

Cycles per sample

Port rate

Threshold

8

Port rate

Math

19

Throughput

1 sample / 8 cycles

1 sample / 8 cycles
Resource Allocation Algorithm

• Allocate minimal resources to each kernel
• Do
  – Find kernel(s) limiting performance
    • Translate performance through port rates
    – Increment resources of these kernels
• Until
  – Resource exhausted
  – Limiting kernel(s) at recurrence II

• Provably optimal algorithm
Allocation Progression

Total Resources Allocated

Iteration

DC | INT | LPF | ED

1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45
Initiation Interval Progression

![Graph showing Initiation Interval Progression with different iterations and resource levels. The graph has a y-axis labeled 'Resource II' ranging from 0 to 400, and an x-axis labeled 'Iteration' ranging from 1 to 45. The graph includes lines for DC, INT, LPF, and ED, each represented by different colors.](image-url)
Kernel Placement

Resource allocation: 

Simulated annealing to arrange allocated resources

Cost function

Maintain cohesion of individual kernels

Maintain abutment of communicating kernels
Cost Function

Kernel Cohesion

Perimeter 20

Perimeter 22
Cost Function

Extend metric to inter-kernel communication
### Placement Results

- Runtime dwarfed by other parts of the tool chain
  - Wavelet application completes in less than a minute
  - Many options for optimization – incremental cost function

<table>
<thead>
<tr>
<th>App</th>
<th>Min Cost</th>
<th>Generated Cost</th>
<th>Cost Ratio</th>
<th>Average Wirelength</th>
<th>Max Wirelength</th>
<th>Kernels</th>
</tr>
</thead>
<tbody>
<tr>
<td>DWT</td>
<td>100</td>
<td>108</td>
<td>1.08</td>
<td>1.0</td>
<td>1.0</td>
<td>3</td>
</tr>
<tr>
<td>PET</td>
<td>44</td>
<td>44</td>
<td>1.00</td>
<td>1.0</td>
<td>1.0</td>
<td>2</td>
</tr>
<tr>
<td>Bayer</td>
<td>176</td>
<td>182</td>
<td>1.03</td>
<td>1.0</td>
<td>1.0</td>
<td>5</td>
</tr>
<tr>
<td>IPL</td>
<td>156</td>
<td>164</td>
<td>1.05</td>
<td>1.0</td>
<td>1.0</td>
<td>4</td>
</tr>
<tr>
<td>Wavelet</td>
<td>476</td>
<td>522</td>
<td>1.10</td>
<td>1.2</td>
<td>5.0</td>
<td>18</td>
</tr>
</tbody>
</table>
CAD Tools for Enhanced CGRAs

• Builds on capabilities of previous work
• Reduced design effort
  – Provides greater flexibility for the programmer
  – Maximizing performance for a given device
• A path toward processor array tools
  – Applicable to floorplanning these devices as well
  – Leverage greater independence between device sections
Thanks!