A Spatial Computing Architecture for Implementing Computation Circuits

David Grant
Supervisor: Dr. Guy Lemieux

davidg,lemieux@ece.ubc.ca

Overview

- Introduction
- Architecture
- Tools
- Preliminary Results
- Conclusions
Introduction

- **Computational Circuits**
  - Software converted to hardware for performance
    - ex. molecular dynamics, rendering, encoding, ...
    - Word-oriented circuits

- **Circuits can be very large**
  - Synthesis and Simulation have long runtimes
  - Wastes designer productivity

- **Research Objective**
  - Quickly compile and run computational circuits
  - Build a spatial computer
Introduction

• Computational Circuits on FPGAs
  ➔ FPGAs bring simulation speed to ¼ ASIC
  ➔ Problem 1: Long Synthesis Time
    • Synthesis down to gate level
    • Hinders incremental debugging and development
  ➔ Problem 2: Hard Capacity Limit
    • Software accustomed to “infinite resources” model
      – Run a few more instructions
      – Add a few MB of memory
      – Add a few million gates ????
Introduction

- Architecture and Tools to implement computational circuits
  - Custom Architecture
  - Fast Tools

- Architecture+Tools address synthesis time and capacity problems
Introduction

- **Goals**
  - 10x faster than FPGA CAD tools
    - Minutes, not hours, for large design
    - Compile+Place+Route runtime like a software compiler
  - 10x capacity of an FPGA
    - Gracefully increase capacity at expense of speed
  - No less than 1/10th the speed of FPGA
    - Slower only at “full capacity”
    - As fast as FPGA at “low capacity”
Architecture

- **Architecture Summary**
  - 2D Array of Processing Elements (PEs)
    - PE is a small, fast sequential computer
    - 3 GHz
  - Mesochronous clock network
    - Known phase mismatch between neighbours is negligible
  - $n$ cycle fixed schedule
    - One user cycle
**Architecture**

- **PE Router**
  - 5 x 5 data crossbar
  - Fixed static schedule
  - Pipelined interconnect
  - PE accepts 1 write/cycle

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>S</th>
<th>E</th>
<th>W</th>
<th>PE</th>
<th>Raddr</th>
<th>Waddr</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>PE</td>
<td>PE</td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W</td>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

![Diagram of PE Router](image)
• PE Core
  - Fixed static program
  - 1 time-mux ALU
  - Direct connections to neighbours
  - 6 node memories
  - Data memory
  - PLA for bitops
Architecture

- **Architecture questions**
  - PE core
    - Node memory sizes?
    - Memory size?
    - Heterogeneous (e.g., some columns have more memory)?
    - ALU width? 2-way static superscalar?
    - PLA size? Use LUTs instead?
  - PE router
    - Bus widths?
    - Number of routers per PE?
Tools

- **Fast tools**
  - Behavioural compilation and word level synthesis
  - ~1000 PEs to place and route in this architecture
  - ~100,000 CLBs to place and route in FPGA

- **Tool flow**
  - Parallelize
  - Combine
  - Placement
  - Schedule and Route
  - Code Generation
Tools

- **Tool Flow Summary**

  - Circuit Description
  - Parse Parallelize Combine
  - Code for each PE
  - Placement
  - Compile and Assemble
  - Scheduling and Routing

To: Simulator Performance Results

Bitstream

- Code for each PE
- Placement
- Scheduling and Routing

Compile and Assemble
Preliminary Results

The diagram shows the relationship between ME Circuit Size (LAB equivalent) and ME Circuit Speed (Full Block Searches per Second). The x-axis represents the ME Circuit Size (LAB equivalent) ranging from 0 to 2500, while the y-axis represents the ME Circuit Speed ranging from 0 to 400000. Several data points are marked on the graph, including 16x16, SA-2D, TA-2D, 8x8, FPGA, Bit Parallel, 4x4, and 2x2. The graph also includes a trend line labeled "Proposed Architecture."
Thanks

- **Main Idea**
  - Take a computational circuit and compile+run it fast on a spatial architecture

- **Custom Architecture**
  - Time-multiplexed array-of-processors
  - High bandwidth, low latency communication

- **Fast Tools (Future Work)**
  - Behavioural compilation
  - Coarse grained architecture

- **Preliminary Results**
  - Shows performance vs. capacity tradeoff

- [davidg,lemieux]@ece.ubc.ca