WaveScalar
Your comments

- How power-efficient is WS?
- Why do you ignore overhead instructions?
- What about context switches?
- What about OS interactions?
- What about fault detection?
- Why do you need ripple numbers and unordered memory?
- How does the spiller allocate memory?
- What’s stopping DF now?
- Why do you focus on single-thread performance?
- How does flow control work?
- How much does hierarchy buy you?
WaveScalar’s solution

- Order memory operations
- Just enough ordering
- Preserve parallelism
Wave-ordered memory

- Compiler annotates memory operations
- Send memory requests in any order
- Hardware reconstructs the correct order
Wave-ordered memory

- Compiler annotates memory operations
  - Sequence 

- Send memory requests in any order

- Hardware reconstructs the correct order
Wave-ordered memory

- Compiler annotates memory operations
  - Sequence #
  - Successor
- Send memory requests in any order
- Hardware reconstructs the correct order
Wave-ordered memory

- Compiler annotates memory operations
  - ➡️ Sequence #
  - 🔴 Successor
  - 🔵 Predecessor
- Send memory requests in any order
- Hardware reconstructs the correct order
Wave-ordering Example

Load: 2 3 4
Store: 3 4 ?

Store: 4 5 6
Load: 5 6 8
Store: ? 8 9

Load: 4 7 8
Wave-ordering Example

Load 2 3 4
Store 3 4 ?
Load 4 7 8
Store ? 8 9

4 5 6
5 6 8

2 3 4

...
Wave-ordering Example
Wave-ordering Example

Load 2 3 4
Store 3 4 ?

Load 4 7 8
Store ? 8 9

...
Wave-ordering Example

Load 2 3 4
Store 3 4 ?

Load 4 7 8
Store ? 8 9

Load 2 3 4
Store 3 4 ?
Wave-ordering Example

Load 2 3 4
Store 3 4 ?

Load 4 7 8
Store ? 8 9

Load 2 3 4
Store 3 4 ?
Wave-ordering Example

Load
2 3 4
Store
3 4 ?

Load
4 7 8
Store
? 8 9

Load
2 3 4
Store
3 4 ?

Load
? 8 9
Store
Wave-ordering Example

Load 2 3 4
Store 3 4 ?

Store buffer:

Load 4 7 8
Store ? 8 9
Wave-ordered Memory

- Waves are loop-free sections of the control flow graph
- Each dynamic wave has a wave number
- Each value carries its wave number
- Total ordering
  - Ordering between waves
  - “linked list” ordering within waves

[MICRO’03]
Wave-ordered Memory

- Waves are loop-free sections of the control flow graph
- Each dynamic wave has a wave number
- Each value carries its wave number
- Total ordering
  - Ordering between waves
  - “linked list” ordering within waves

[MICRO’03]
Wave-ordered Parallelism

<table>
<thead>
<tr>
<th>Store</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Load</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Load</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>Store</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>
Wave-ordered Parallelism
Wave-ordered Memory

- Annotations summarize the CFG
- Expressing parallelism
  - Reorder consecutive operations
- Alternative solution: token passing [Beck, JPDC’91]
  - 1/2 the parallelism
WaveScalar’s execution model

- Dataflow execution
- Von Neumann-style memory
- Coarse-grain threads
- Light-weight synchronization
WaveScalar Outline

- Execution model
- Hardware design
  - Scalable
  - Low-complexity
  - Flexible
- Evaluation
- Exploiting dataflow features
- Beyond WaveScalar: Future work
Executing WaveScalar

- Ideally
  - One ALU per instruction
  - Direct communication

- Practically
  - Fewer ALUs
  - Reuse them
WaveScalar processor architecture

- Array of processing elements (PEs)
- Dynamic instruction placement/eviction
Processing Element

- Simple, small
  - 0.5M transistors
- 5-stage pipeline
- Holds 64 instructions
PEs in a Pod
Domain
Cluster
WaveScalar Processor
WaveScalar Processor

- Long distance communication
  - Dynamic routing
  - Grid-based network
- 32K instructions
- \(\sim 400\text{mm}^2\) 90nm
- 22FO4 (1Ghz)
WaveScalar processor architecture

- Low complexity
- Scalable
- Flexible parallelism
- Flexible allocation
WaveScalar processor architecture

- Low complexity
- Scalable
- Flexible parallelism
- Flexible allocation
WaveScalar processor architecture

- Low complexity
- Scalable
- Flexible parallelism
- Flexible allocation
WaveScalar processor architecture

- Low complexity
- Scalable
- Flexible parallelism
- Flexible allocation
WaveScalar processor architecture

- Low complexity
- Scalable
- Flexible parallelism
- Flexible allocation
WaveScalar processor architecture

- Low complexity
- Scalable
- Flexible parallelism
- Flexible allocation
WaveScalar processor architecture

- Low complexity
- Scalable
- Flexible parallelism
- Flexible allocation
WaveScalar processor architecture

- Low complexity
- Scalable
- Flexible parallelism
- Flexible allocation
WaveScalar Outline

- Execution model
- Hardware design
- Evaluation
  - Map WaveScalar’s design space
  - Scalability
  - CMP comparison
- Exploiting dataflow features
- Beyond WaveScalar: Future work
Performance Methodology

- Cycle-level simulator
- Workloads
  - SpecINT + SpecFP
  - Splash2
  - Mediabench
- Binary translator from Alpha -> WaveScalar
- Alpha Instructions per Cycle (AIPC)
- Synthesizable Verilog model
WaveScalar’s design space

- Many, many parameters
  - # of clusters, domains, PEs, instructions/PE, etc.
  - Very large design space
- No intuition about good designs
- How to find good designs?
  - Search by hand
  - Complete, systematic search
WaveScalar’s design space

- Constrain the design space
  - Synthesizable RTL model -> Area model
  - Fix cycle time (22FO4) and area budget (400mm$^2$)
  - Apply some “common sense” rules
  - Focus on area-critical parameters

- There are 201 reasonable WaveScalar designs
  - Simulate them all
WaveScalar’s design space

[ISCA’06]
Pareto Optimal Designs

[ISCA'06]
WaveScalar is Scalable
WaveScalar is Scalable

7x apart in area and performance
Area efficiency
Area efficiency

- Performance per silicon: IPC/mm$^2$
Area efficiency

- Performance per silicon: IPC/mm²
- WaveScalar
  - 1-4 clusters: 0.07
  - 16 clusters: 0.05
Area efficiency

- Performance per silicon: IPC/mm²
- WaveScalar
  - 1-4 clusters: 0.07
  - 16 clusters: 0.05
- Pentium 4: 0.001-0.013
Area efficiency

- Performance per silicon: IPC/mm$^2$
- WaveScalar
  - 1-4 clusters: 0.07
  - 16 clusters: 0.05
- Pentium 4: 0.001-0.013
- Alpha 21264: 0.008
Area efficiency

- Performance per silicon: IPC/mm$^2$
- WaveScalar
  - 1-4 clusters: 0.07
  - 16 clusters: 0.05
- Pentium 4: 0.001-0.013
- Alpha 21264: 0.008
- Niagara (8-way CMP): 0.01