KiloCore: A 32 nm 1000-Processor Array

Brent Bohnenstiehl, Aaron Stillmaker, Jon Pimentel, Timothy Andreas, Bin Liu, Anh Tran, Emmanuel Adeagbo, Bevan Baas

University of California, Davis
VLSI Computation Laboratory
August 23, 2016
• Number of processors on single die vs. year
  – Each processor capable of independent program execution
KiloCore Chip

<table>
<thead>
<tr>
<th>Technology</th>
<th>32nm IBM PDSOI CMOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. Procs.</td>
<td>1000</td>
</tr>
<tr>
<td>Num. Mems.</td>
<td>12</td>
</tr>
<tr>
<td>Die Area</td>
<td>64 mm²</td>
</tr>
<tr>
<td>Array Area</td>
<td>60 mm²</td>
</tr>
<tr>
<td>Transistors</td>
<td>621 Million</td>
</tr>
<tr>
<td>C4 Bumps</td>
<td>564 (162 I/O)</td>
</tr>
<tr>
<td>Package</td>
<td>676 Pad Flip-Chip BGA</td>
</tr>
</tbody>
</table>
## Single Processor Tile

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tile Area</td>
<td>0.055 mm²</td>
</tr>
<tr>
<td>Transistors</td>
<td>574,733</td>
</tr>
<tr>
<td>Instruction Memory</td>
<td>128 x 40-bit</td>
</tr>
<tr>
<td>Data Memory</td>
<td>256 x 16-bit</td>
</tr>
<tr>
<td>Input FIFO Size (x2)</td>
<td>32 x 16-bit</td>
</tr>
<tr>
<td>Instruction Types</td>
<td>72</td>
</tr>
</tbody>
</table>

- **Tile Area**: 0.055 mm²
- **Transistors**: 574,733
- **Instruction Memory**: 128 x 40-bit
- **Data Memory**: 256 x 16-bit
- **Input FIFO Size (x2)**: 32 x 16-bit
- **Instruction Types**: 72
Single Memory Tile

- **Tile Area**: 0.164 mm²
- **Transistors**: 3,813,095
- **SRAM Size**: 64 kB
- **Input FIFO Size (x2)**: 32 x 18-bit
- **Input FIFO Size (x1)**: 16 x 2-bit
- **Output FIFO Size (x2)**: 32 x 16-bit
Overview

• KiloCore is best suited for computationally-intensive applications and kernels

• Each processor holds up to 128 instructions
  – 40-bits per instruction
  – Modified during application programming
  – Typically static during the run time of an application
  – Larger programs are supported for processors neighboring a memory module

• Data is passed by messages between processors
  – A pair of processors neighboring a shared memory may transfer data through that memory
Programming

• Applications are implemented as a set of suitably small programs by:
  – Organizing the application into a group of tasks
  – Partitioning task code into serial blocks
  – Replicating parallelizable code blocks
• Partitioning techniques are suitable for tool automation

Example of an application mapped onto KiloCore
**GALS Clocking**

- Globally Asynchronous, Locally Synchronous Clocking
- 2012 oscillators
  - One per processor, packet router, and memory
- Oscillators may:
  - Independently change frequency
  - Halt within 1-5 clock periods when work is not available
  - Restart in less than 1 clock period
- Halted processors consume 1.1% of their typical active power
- Data is synchronized using dual clock buffers between domains

Note: Halted processor power measurement taken at 900 mV
Communication Network

- **Two layer circuit switched network**
  - Statically configured during programming
  - Source-synchronous
  - 16-bit data width per link
  - Up to 28 Gbps per link
  - 456 Gbps total tile I/O

- **Dynamic packet routing network**
  - Wormhole routing
  - Source-synchronous
  - 16-bit data width per flit
  - Up to 9.1 Gbps per link

Note: bandwidth measurements taken at 1.1 V
Processor Pipeline

- 7-stage pipeline
- 16-bit, fixed-point datapath
- 40-bit, memory-to-memory instructions
- Single-issue, in-order execution

<table>
<thead>
<tr>
<th>Instructions by Opcode Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add/Sub</td>
</tr>
<tr>
<td>Logic</td>
</tr>
<tr>
<td>Mac</td>
</tr>
<tr>
<td>Branch</td>
</tr>
<tr>
<td>Other</td>
</tr>
</tbody>
</table>
Processor Pipeline

• Signed and unsigned operations
• Multiplier is 16-bit in, 32-bit out, with 40 bit accumulator
  – Supports one multiply per two cycles
• Predication supported for all instructions
• Automated loop hardware accelerates innermost loops
• Static branch prediction
  – Controlled by opcode selected during compilation
  – 94% of branches predicted correctly in sampled applications
    • Many branches close loops or handle special cases
    • Difficult to predict branches are often replaced with predication
Processor Data Memory

• Two data memory banks
• Instruction operands sourced one from each bank
  – Each source is assigned a default bank; if either source reads the other bank, swap banks
• Instructions optionally write back to one or both banks
  – Software selects this by setting a Dual_Write flag
The compiler will:
- Find variables potentially read on the same cycle
- Construct read conflict lists
- Map variables to memory banks to avoid same-bank conflicts

A variable is mapped to both banks only when a conflict is otherwise unavoidable.

### Example of variable conflict analysis and mapping

<table>
<thead>
<tr>
<th>Var.</th>
<th>Conflicts with</th>
<th>Mapped to bank</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B, E, …</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>A, E, …</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>A, B, …</td>
<td>0 &amp; 1</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

### Rule Table

<table>
<thead>
<tr>
<th>Instr.</th>
<th>Src 0 bank</th>
<th>Src 1 bank</th>
<th>Swap read banks?</th>
<th>Dual write flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>C=A+B</td>
<td>0</td>
<td>1</td>
<td>No</td>
<td>0</td>
</tr>
<tr>
<td>E=D-C</td>
<td>1</td>
<td>0</td>
<td>Yes</td>
<td>1</td>
</tr>
<tr>
<td>X=E-A</td>
<td>0</td>
<td>0</td>
<td>Yes</td>
<td>0</td>
</tr>
<tr>
<td>Y=E-B</td>
<td>0</td>
<td>1</td>
<td>No</td>
<td>0</td>
</tr>
</tbody>
</table>
• Each independent memory module connects to two neighboring processors
• Offers 64 kB of storage – 780 kB total across 12 memories
• Supports random and burst access modes, with programmable addressing patterns
Shared Memory, Instruction Streaming

• Memory may stream instructions to one neighboring processor
• Extends program size from 128 up to 10,922 instructions
• Program control is handled in the memory module
  – 16-bit controller
  – 8-deep branch prediction and correction queue
• Used for complex administrative tasks and highly serial, low priority tasks
Physical Design Notes

• Tools used:
  – Design Compiler by Synopsys
  – SoC Encounter by Cadence

• 34 days between full access to design libraries and tapeout

• Chip functionality:
  – All processors, network, and shared memory are fully functional except hold time violations on some network paths

• Non-custom BGA flip-chip C4 package:
  – Indirect power delivery outside the center of the processor array leads to voltage droop in outer processors when operating at high voltage and activity
## Frequency Measurements

<table>
<thead>
<tr>
<th></th>
<th>Supply Voltage (V)</th>
<th>Maximum Frequency (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Processor</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 V</td>
<td>1.78 GHz</td>
<td></td>
</tr>
<tr>
<td>900 mV</td>
<td>1.24 GHz</td>
<td></td>
</tr>
<tr>
<td>560 mV</td>
<td>115 MHz</td>
<td></td>
</tr>
<tr>
<td><strong>Independent Memory</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 V</td>
<td>1.77 GHz</td>
<td></td>
</tr>
<tr>
<td>900 mV</td>
<td>1.27 GHz</td>
<td></td>
</tr>
<tr>
<td>760 mV</td>
<td>675 MHz</td>
<td></td>
</tr>
<tr>
<td><strong>Packet Router</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 V</td>
<td>1.49 GHz</td>
<td></td>
</tr>
<tr>
<td>900 mV</td>
<td>884 MHz</td>
<td></td>
</tr>
<tr>
<td>670 mV</td>
<td>262 MHz</td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**
Measurements made at 25°C; lowest measurements are at the respective minimum operable voltages
### Power Measurements

<table>
<thead>
<tr>
<th></th>
<th>1.1 V</th>
<th>900 mV</th>
<th>560 mV</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Processor</strong></td>
<td>38.8 mW</td>
<td>17.7 mW</td>
<td>0.7 mW</td>
</tr>
<tr>
<td><strong>Memory</strong></td>
<td>59.0 mW</td>
<td>26.5 mW</td>
<td>9.5 mW</td>
</tr>
<tr>
<td><strong>Packet Router</strong></td>
<td>5.5 mW</td>
<td>2.1 mW</td>
<td>0.4 mW</td>
</tr>
</tbody>
</table>

**Diagram:**

- **Process Power**
- **Memory Power**
- **Router Power**

**Power at 100% Activity (mW)** vs **Supply Voltage (V)**
Measurements

- KiloCore has a potential maximum of 1.78 trillion instructions per second using 40 Watts
  - Assumes a custom package design
- At minimum voltage, KiloCore performs up to 115 billion instructions per second using 0.7 Watts
- Processors achieve their optimal energy times time of 11.1 (pJ x ns / instruction) at a voltage of 0.9 V
- Chip minimum voltage is constrained by any active application’s usage of memories or routers
  - 760 mV if any independent memory is in use, 670 mV if the packet network is in use, 560 mV otherwise
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleepwalker [1]</td>
<td>1</td>
<td>65</td>
<td>0.42</td>
<td>25 23.6</td>
<td>0.4 0.375</td>
<td>2.6 2.2</td>
<td>104 93.2</td>
<td>N/A</td>
</tr>
<tr>
<td>IBM Cell [2]</td>
<td>9</td>
<td>90</td>
<td>14.5</td>
<td>5000</td>
<td>1.3</td>
<td>1100</td>
<td>220</td>
<td>2.46</td>
</tr>
<tr>
<td>Tilera/EZChip Gx72 [3]</td>
<td>72</td>
<td>40</td>
<td>-</td>
<td>1200</td>
<td>-</td>
<td>750</td>
<td>625</td>
<td>3.44</td>
</tr>
<tr>
<td>Intel TeraFlops [4]</td>
<td>80</td>
<td>65</td>
<td>3</td>
<td>4000 3130</td>
<td>1.2 1.0</td>
<td>70.6 49.1</td>
<td>17.7 15.7</td>
<td>2.65</td>
</tr>
<tr>
<td>Ambric Am2045 [6]</td>
<td>336</td>
<td>130</td>
<td>-</td>
<td>300</td>
<td>-</td>
<td>79.4</td>
<td>265</td>
<td>0.713</td>
</tr>
<tr>
<td>KiloCore [7]</td>
<td>1000</td>
<td>32</td>
<td>0.055</td>
<td>1782 1237 115</td>
<td>1.1 0.9 0.56</td>
<td>21.9 13.8 5.8</td>
<td>12.2 11.1 5.8</td>
<td>4.24</td>
</tr>
</tbody>
</table>

1. JSSC'13  2. MICRO'05  3. EZChip Product Brief 2016  4. ISSCC'07  5. JSSC'09  6. MICRO'07  7. VLSI Symp.'16
Applications

- Several applications have been implemented for KiloCore:
  - Fast Fourier Transform
    - 4096 length, 16-bit fixed-point data
    - Using 980 processors, 12 memories
    - 138 thousand FFTs/s at 4.0 Watts
  - Advanced Encryption Standard
    - 128-bit keys
    - Using 974 processors
    - 14.9 Gb/s at 9.1 Watts
  - Low Density Parity Check
    - 4095 code length
    - Using 944 processors, 12 memories
    - 111 Mb/s at 3.4 Watts
  - Record Sort
    - 100 Byte records with 10 Byte keys, 1850 records per sorted block
    - Using 1000 processors
    - 12.4 million records/s at 0.8 Watts

Notes:
Performance based on cycle-accurate simulations using fine-grain sub-instruction energy measurements at 900 mV. Implementations have not been optimized.
Application Comparison

- Application implementations compared against a desktop Intel i7-3770k processor.
  - 22 nm technology, 160 mm² die area
  - Using FFTW, C++ std::sort, open source AES C library, custom LDPC C++ implementation
    - FFT operating on single precision floating point data, not using AES specialized instructions, operating on pre-cached data, using 8 threads

Relative Throughput:

- AES: 14.6
- LDPC: 3.5
- FFT: 1.9
- Sort: 3.1

Relative Throughput per Watt:

- AES: 79
- LDPC: 53
- FFT: 8
- Sort: 23
Acknowledgments

• Funding and Support:
  – DoD and ARL/ARO Grant W911NF-13-1-0090
  – TAPO
  – NSF CAREER award 546907
    CCF Grant No. 430090
    CCF Grant No. 903549
    CCF Grant No. 1018972
    CCF Grant No. 1321163
  – SRC GRC Grant 1598
    CSR Grant 1659
    GRC Grant 1971
    GRC Grant 2321
  – ST Microelectronics
  – C2S2
  – Intel Corporation
  – UCD Faculty Research Grant
  – MOSIS
  – Artisan

– UCDFacultyResearchGrant
– MOSIS
– Artisan