Cymric
A Framework for Prototyping Near-Memory Architectures

Chad D. Kersey\textsuperscript{1}, Hyesoon Kim\textsuperscript{2}, Sudhakar Yalamanchili\textsuperscript{1}

The rest of the Cymric team:
Nathan Braswell, Jemmy Gazhenko, Prasun Gera, Meghana Gupta,
Hyojong Kim, Joo Hwan Lee, Troy O’Neal, Rahul Rajan

\textsuperscript{1}Georgia Tech School of ECE; \{cdkersey,sudha\}@gatech.edu
\textsuperscript{2}Georgia Tech College of Computing; hyesoon@cc.gatech.edu
Summary and Outline

Summary

We have built Cymric, which:

- Enables simulations and prototyping of novel heterogeneous architectures, such as those employing stacked DRAM.
- Uses C++ as a common, unified description language.
- Includes prototype logic layer model for stacked memory systems which runs on the Micron EX-800 HMC/FPGA board.

Outline

- Background
- The Cymric Infrastructure
- Target Design
- Implementation Platform (EX-800)
- Project Status
Vault-based stacked DRAM.

- HMC organization: multiple channels (vaults).
- One bank per layer per vault.

Hybrid Memory Cube (Micron):
- 16 or 32 vaults.
- 4 or 8 DRAM layers.
- 4 160Gb/s (each way) links

Also: HBM, WideI/O
To evaluate stacked DRAM architectures employing near-memory computation, we have developed Cymric, an open source environment including:

- **CHDL hardware design library.**
- **Caches, scratchpads, and ROMs using CHDL’s memory interface format.**
- **HARP SIMT instruction sets, Harmonica 2 implementation.**
- **HARP LLVM back-end and run-time library (running OpenCL-like kernels).**
- **Memory interfaces for Altera’s Avalon and Pico’s HMC controller.**

### Overview of Cymric.

1. Except pieces marked in gray.
CHDL output can be:

- Synthesized to netlist.
- Simulated at logic level.
- Run on FPGA w/ vendor tools.

- Designs in CHDL are netlist generators.
- Ultimately generating basic gates and memory arrays.
- All written in standard C++.

Parameterizability

HARP’s configurability is enabled by C++’s support for template metaprogramming, brought into the hardware domain by CHDL.
Implementation Environment: CHDL

CHDL provides:
- Structured signals.
- Loadable netlists.

The CHDL module library provides a set of pre-built IP-cores along with HTML documentation of the interfaces.

Modules with CHDL memory interfaces can be simulated in the Structural Simulation Toolkit\(^1\).
- Harmonica cores in FPGA clocked at 125MHz, with 10000 seconds of static overhead.
- Simulation environment runs at about 100Hz, with 100 seconds of static overhead.
- For under 1M cycles, simulation is faster.
- Simulations allow arbitrary configuration of the memory system.

\(^1\)Rodrigues, et al. 2011 ACM SIGMETRICS
SIMT architectures:
- Many threads in flight.
- Threads grouped into warps.
- SIMD parallelism within warps.
- Many warps in flight; few hazards, less forwarding.
- Pipeline parallelism between warps.
- Lanes enabled using *active mask*.
- When control flow paths diverge, serialize.
- Due to TLP and large number of warps: high MLP, mem. latency tolerant.

*4-lane SIMT pipeline.*

HARP (our SIMT implementation)
- Architecture for a Heterogeneous Architecture Research Prototype.
- GPGPU-like. Also provides dynamic thread/warp spawn instructions.

---

1 Based on image from nVidia.
Cymric processor options include Harmonica\(^1\), supporting the HARP family of SIMT instruction sets.

- To the assembler and linker, a short string identifies the ISA.
- The fields of the ArchID correspond to parameters of the Harmonica implementation.

**HARP Architecture ID String**

- 4 bytes per word
- "Word" instruction encoding
- 16 GP regs.
- 16 pred. regs.
- 8 SIMD lanes*  

*ignored by assembler/linker

**Parameters**

HARP ISAs are fully parameterized, including:

- Number of registers.
- Number of SIMT lanes.
- Available instructions and functional units.

---

\(^1\)Kersey et al. FCCM 2014 (poster)
Implementation Environment: Harmonica

Harmonica sizes with various parameter values.

<table>
<thead>
<tr>
<th>Warps</th>
<th>Lanes</th>
<th>Gates</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>8</td>
<td>285005</td>
</tr>
<tr>
<td>16</td>
<td>32</td>
<td>1078223</td>
</tr>
<tr>
<td>32</td>
<td>8</td>
<td>328092</td>
</tr>
<tr>
<td>32</td>
<td>32</td>
<td>1145118</td>
</tr>
</tbody>
</table>

- Custom SIMT design using HARP ISA.
- Warps, lanes, registers: parameters.
- Simple design (1600 lines of C++).
- Doesn’t extract ILP within threads.
- Optional floating point support.
Implementation Platform: EX-800

- Running prototype on Micron EX-800 HMC evaluation board.
- 4 Altera Stratix V FPGAs, each with a 160Gbps link, internally split into five 32Gbps links.
- We build our prototype by modifying existing FPGA firmware.

EX-800 HMC board. ¹

Relevant components annotated with unidirectional bandwidths.

¹Source: picocomputing.com
Running prototype on Micron EX-800 HMC evaluation board.

4 Altera Stratix V FPGAs, each with a 160Gbps link, internally split into five 32Gbps links.

We build our prototype by modifying existing FPGA firmware.

RELEVANT COMPONENTS ANNOTATED WITH UNIDIRECTIONAL BANDWIDTHS.

1Source: picocomputing.com
Prototype: Target Design

Our planned target design:

- **Current capability:** running single MIPS or Harmonica instance on the EX-800 at 125MHz.
- **Next:** fully populating the FPGA with cores and running applications; attempting to reach 250MHz.
Prototype: Memory Interface

The crucial modules for prototyping are the memory interface adaptors, converting between the following:

- **CHDL mem_port**:
  - Limited to 1 beat.
  - Masked writes.
  - Parameterized.

- **Pico bus. (EX800 config)**
  - Synchronous.
  - No flow control.

- **EX-800 HMC interface. (x5)**
  - 128 bits wide.
  - 6-bit tag.
  - 1 to 8 beats.
  - No masked writes.

Four clock domains:
- Configuration bus. (4MHz)
- HMC interface TX. (250MHz)
- HMC interface RX. (250MHz)
- Half-RX core clock. (125MHz)

**CHDL Asynchronous FIFO**
- Fully parameterized.
- 59 lines. (150 typ. for vl.)

---

OpenHMC is being considered as a fully open-source alternative.
Cymric: Status

Designs simulated:

- Harmonica cores in various configurations attached to simulated memory systems, running relational benchmarks (join, select, and project), key/value store, graph BFT, FFT, and matrix multiply.
- Heterogeneous Harmonica/MIPS systems with shared memory.
- Multi-core MIPS systems using SST’s simulated coherent caches.

Designs FPGA proven:

- RISC and Harmonica with scratchpad memories on Cyclone II.
- RISC/Harmonica together with external DDR 2 on Stratix 3.
- Harmonica multi-cores with external DDR 3 on Stratix V.

Designs and programs run on the EX-800 so far:

- Scratchpad memory attached to configuration bus.
- One RISC core attached to scratchpad memory attached to configuration bus.
- One small (4 lane, 4 warp) Harmonica core running simple single-threaded test program attached to single HMC interface.

Chad D. Kersey, Hyesoon Kim, Sudhakar Yalamanchili
## Cymric: Status

<table>
<thead>
<tr>
<th>Project Universe</th>
<th>Code</th>
<th>Test</th>
<th>Doc</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HARP Universe</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>harptool</td>
<td>✓</td>
<td>△</td>
<td>✓</td>
<td>Assembler/linker/emulator.</td>
</tr>
<tr>
<td>compiler</td>
<td>△</td>
<td>△</td>
<td>△</td>
<td>LLVM back-end.</td>
</tr>
<tr>
<td>benchmarks</td>
<td>✓</td>
<td>X</td>
<td>△</td>
<td>Assembly language apps.</td>
</tr>
<tr>
<td><strong>CHDL Universe</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>chdl</td>
<td>✓</td>
<td>△</td>
<td>△</td>
<td>Hardware design framework/library.</td>
</tr>
<tr>
<td>chdl-stl</td>
<td>✓</td>
<td>✓</td>
<td>△</td>
<td>Template library for CHDL.</td>
</tr>
<tr>
<td>chdl-module</td>
<td>✓</td>
<td>✓</td>
<td>△</td>
<td>Use netlists as sub-modules.</td>
</tr>
<tr>
<td>iqyax</td>
<td>✓</td>
<td>△</td>
<td>X</td>
<td>MIPS-like in-order core.</td>
</tr>
<tr>
<td>harmonica</td>
<td>✓</td>
<td>△</td>
<td>X</td>
<td>SIMT processor core.</td>
</tr>
<tr>
<td>harpsim</td>
<td>✓</td>
<td>△</td>
<td>X</td>
<td>Harpica simulator.</td>
</tr>
<tr>
<td>cache</td>
<td>✓</td>
<td>△</td>
<td>X</td>
<td>Set associative cache.</td>
</tr>
<tr>
<td>chdl-sst</td>
<td>✓</td>
<td>△</td>
<td>X</td>
<td>Use CHDL in SST framework.</td>
</tr>
<tr>
<td>mem-iface</td>
<td>△</td>
<td>△</td>
<td>X</td>
<td>Memory interface adaptors.</td>
</tr>
</tbody>
</table>

**✓** Not started.  **△** In progress.  **✓** Finished.
Thank You

Questions?