Heracles: Fully Synthesizable Parameterized MIPS-Based Multicore System

Michel Kinsy
Overview of the *Heracles System*

**Motivation**

- Multicore architectures have become mainstream computing platforms.

- Designing these architectures requires selecting many parameters including core computation power, memory hierarchy, topology, routers, routing algorithms, area, power, among others.

- If we take research in routing algorithms for Network-on-Chips (NoCs), for example, finding an appropriate platform to evaluate the routing scheme in fine detail is extremely difficult.

- Software systems can be too slow: DARSIM, HORNET, GRAPHITE, etc.

- Hardware systems can be too inflexible: RAMP, LEON, S-Scale, etc...
Overview of the **Heracles System**

*What about a hardware toolbox?*

Not just a system but a hardware toolbox for design Exploration of Multicore architectures:

- Micro-architecture trade-offs at core and router levels
- Memory hierarchy
- Routing Algorithms
- Fast testing environment with RTL level correctness

**Heracles:**

- an open-source complete multicore platform written in Verilog,
- with RISC MIPS Core, as processing unit,
- user-controlled distribution scheme of memory,
- realistic fully detailed buffered routers,
- supports both logic-based and table-based routing algorithms,
- fully parameterized and modular,
- can be reconfigured and synthesized into different topologies and sizes.
Overview of the Heracles System

Heracles in Mesh Configuration

- Core Starting PC and Routing Table Data
- Node
- MIPS-Based Processing Element
- Memory Subsystem and Router
It is a single-issue in-order MIPS core, with 7-stage pipeline, fully bypassed, no branch prediction or branch delay slot, running MIPS-III instruction set architecture (ISA) without floating point.
The target FPGA used for our synthesis is Virtex-5 LX330T.

<table>
<thead>
<tr>
<th></th>
<th>Used</th>
<th>Available</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registers</td>
<td>1,635</td>
<td>207,360</td>
<td>under 1%</td>
</tr>
<tr>
<td>Lookup Tables</td>
<td>2,529</td>
<td>207,360</td>
<td>1%</td>
</tr>
</tbody>
</table>

- The core is oblivious to the memory hierarchy.
- Its interface is kept simple.
- The MIPS core can be replaced by any other core with no change to the rest of the platform.
Memory System Organization

Two views of an node memory subsystem

(a) Expanded View of Memory Hierarchy

(b) Expanded View of Router Wrapper

Memory System Wrapper and Router Wrapper
Memory System Organization

Memory System Wrapper

- **Cache system**: data cache & instruction cache can be separate or unified. INDEX_BITS & OFFSET_BITS parameters control the number of cache blocks and their size.

- **Address resolution logic**: determines if a request can be served at the local memory, or if the request needs to be sent over the network.

- **Local block memory**: the LOCAL ADDR BITS parameter is used to set the size of the local memory.
Memory System Organization

Memory System Wrapper

- Packetizer Lite: is responsible for converting data traffic into packets that can be routed inside the Network on-chip (NoC).
Memory System Wrapper

**MIPS core with I-Cache and D-Cache, 2KB each, and 262KB of local memory.**

**Local memory size can be set on a per core-basis, and it can service a variable number of caches in a round-robin fashion.**

**This allows both Shared Memory (SM) and Distributed Shared Memory (DSM) implementations.**
Virtual-Channel Router Micro-Architecture

- VC_PER_PORT and VC_DEPTH parameters allow user controlled router size.
Router Architecture

On-chip Routing

- The routing operation has 4 phases:
  1. **Route computation** (RC), it determines the next hop for the packet.
  2. **Virtual-channel allocation** (VA), it allocates a virtual channel in the next hop.
  3. **Switch allocation** (SA), the flit competes for a switch port.
  4. **Switch traversal** (ST), it moves flit onto the output port link.

- In our design, each phase corresponds to a router pipeline stage.
- The switch allocation (SA) stage is the critical stage in our router design, due to the logical complexity of the arbiter.
- The arbiter is also responsible for adjusting priorities to promote fairness and avoid starvation.

- Supports both fixed logic (DOR) and table routing (BSOR) based oblivious routing algorithms.
Router Architecture

Virtual-Channel Router Micro-Architecture

- Can only support static, preconfigured routes, with no shared links.
Router Architecture

<table>
<thead>
<tr>
<th></th>
<th>Used</th>
<th>Available</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registers</td>
<td>2,806</td>
<td>207,360</td>
<td>1%</td>
</tr>
<tr>
<td>Lookup Tables</td>
<td>2,058</td>
<td>207,360</td>
<td>1%</td>
</tr>
</tbody>
</table>

The **RT_ALG** parameter is used to select the proper routing algorithm for a given application and topology.

- Supports for both static and dynamic virtual channel allocation.

Network Topology Configuration

- The parameterization of the number of input ports and output ports gives the platform the flexibility to metamorphose into different network topologies.

- For example, k-ary n-cube, 2D-mesh, 3D-mesh, hypercube, ring, or tree.

- A new topology is constructed by changing the **IN_PORTS**, **OUT_PORTS**, and **SWITCH_TO_SWITCH** parameters and reconnecting the routers.

- In the case of the 3D-mesh, **IN_PORTS** and **OUT_PORTS** parameters are set to 2 or 3, one to connect the router to the local core and the rest to the third dimension.
For 2-D mesh topology, we are able to fit up to 16 cores on the Virtex-5 LX330T FPGA board.

In the 2x2 configuration, the local shared memory is set to 260KB per core, in the 3x3 configuration the size is reduced to 64KB per core, and in the 4x4 configuration it is set to 32KB.
Router at different levels of the tree have different sizes, in terms of crossbar and arbitration logic.

The root node contains the largest router, and controls the clock frequency of the system.
The toolchain for this release supports a single-Thread C programming model. It is built around the GCC MIPS cross-compiler using GNU C version 3.2.1.

Currently, there is no direct high-level operating system support. Therefore, in the third compilation stage, a small kernel-like assembly code is added to the application assembly code for memory space management and workload distribution.

Users can modify the `linker.cpp` file provided in the toolchain to reconfigure the memory space and workload depending on system configuration.
Summary

- This project tries to provide the community of computer architects and students working on various aspect of multicore systems, an environment for fast design space exploration at RTL correctness.

- **Heracles** is a complete, realistic, fully parameterized, synthesizable, modular, multicore architecture platform. It uses a component-based design approach, where the processing element or core, the router and the network-on-chip, and the memory subsystem are independent building blocks, and can be used in other designs.

- Future versions will involve adding a small kernel binary code to each core on start up for handling exceptions and proper interrupts for peripheral communications. Multi-threading and dynamic runtime workload management among the cores, via thread migration, will be explored.