Kunpeng 920 (ARMv8) - Hardware-specific Support

Alex Margolin

UCX Hackathon, Dec. 2019
Outline

• Intro: Hi1620 Arm Processor (market name: Kunpeng 920)

• Special Instructions

• Special UD support

• Special Thresholds / Algorithms
Kunpeng 920 (Internally: Hi1620)

• Huawei’s Latest ARMv8-based SoC, targeting high-end systems
  • Up to 64 cores @3GHz,
  • Includes on-chip 100GbE RoCE support
# Huawei’s Hi16xx SoC Family

<table>
<thead>
<tr>
<th></th>
<th>Hi1610</th>
<th>Hi1612</th>
<th>Hi1616</th>
<th>Hi1620</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cores</strong></td>
<td>16</td>
<td>32</td>
<td>32</td>
<td>24 to 64</td>
</tr>
<tr>
<td><strong>Architecture</strong></td>
<td>Cortex-A57</td>
<td>Cortex-A57</td>
<td>Cortex-A72</td>
<td>Ares (ARMv8-based)</td>
</tr>
<tr>
<td><strong>Frequency (GHz)</strong></td>
<td>2.1 GHz</td>
<td>2.1 GHz</td>
<td>2.4 GHz</td>
<td>2.4 to 3.0</td>
</tr>
<tr>
<td><strong>Interconnect</strong></td>
<td></td>
<td></td>
<td>Up to 2S 96 Gbps/port</td>
<td>Up to 4S 240 Gbps/port</td>
</tr>
<tr>
<td><strong>IO</strong></td>
<td>16 PCIe 3.0</td>
<td>16 PCIe 3.0</td>
<td>46 PCIe 3.0, 8 x 10GE</td>
<td>40 PCIe 4.0, 2 x 100 GE</td>
</tr>
<tr>
<td><strong>Process</strong></td>
<td>TSMC 16nm</td>
<td>TSMC 16nm</td>
<td>TSMC 16nm</td>
<td>TSMC 7nm</td>
</tr>
</tbody>
</table>

[Diagram showing the timeline of Huawei's Hi series SoC development from 1991 to 2023.]

Huawei's Hi16xx SoC Family
Benchmarking Huawei ARM Server Processor for HPC Workloads

James Lin

ARM HPC User Group, Dallas, US November, 2018

Shanghai Jiao Tong University, Center for HPC
Roofline model for GTC-P code

- GTC-P code with 6 main functions including 2 main hotspots \textit{charge} (SCATTER) and \textit{push} (GATHER) kernels.

- It is a memory-bound code with irregular memory access.

- The code base is the MPI + OpenMP version of GTC-P code. We compile the code with GCC 8.2 compiler.
Vectorization on Hi1616 and Hi1620

- **Hi1616**
  - 128-bit SIMD
  - SP: 614.4 Gflops
  - DP: 307.2 Gflops

- **Hi1620**
  - 128-bit SIMD
  - SP: 1,536 Gflops
  - DP: 384 Glops

- The throughput of DP vectorization is decreased to only half on Hi1620

<table>
<thead>
<tr>
<th></th>
<th>SP Scalar</th>
<th>DP Scalar</th>
<th>SP Vector</th>
<th>DP Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hi1620</td>
<td>2ins/cycle 7.989GFlops</td>
<td>2ins/cycle 7.989GFlops</td>
<td>2ins/cycle 31.954GFlops</td>
<td>1ins/cycle 7.989GFlops</td>
</tr>
</tbody>
</table>
Special Instructions

• Nothing exciting just yet... probably vector operations in the future.

• Meanwhile, I added “CLWB” for x86 (no PR yet):

  ![Synchronous Ordering (SO)]

  • Based on x86 ISA changes for PM
  • CLWB <Addr>
    – Writes back cache block to Memory Controller (MC)
  • PCOMMIT
    – Persists all accepted stores at MC
Special UD support

• We’re actually working on 2 new UD transports:

1. ud_1620 (for hi1620-specifics)
   *UD 1620 has some limitations compared to upstream UD, and a custom WQE

2. offloading service, not yet public
Special Thresholds / Algorithms

• 64 cores is much more than Intel typically has – so we
  • Example: Shared-memory collectives

• Different thresholds for CPU/NIC offload

• Using the aforementioned special ISA / RoCE support