A design space exploration targeting next-generation HPC architectures

Constantino Gómez
Ph.D. Student
New opportunities for HW customization

• Blossoming of the IP block licensing model
  • **Reduced cost** and **time-to-market** of new custom designs.

• Evolution of the *pure-play* foundries
  • Everyone can get the **best CMOS technology at a ‘affordable’ price.**

• Moore’s law is dying
  • We have **problems adding** more simultaneously **ACTIVE transistors.**

• Result: **No more one shoe fits all**
  • Supercomputers targeting one field of applications reappearing
  • AI Bridging Cloud Infrastructure, Sierra and Summit, Astra...
Designing application-specific architectures

• Which configuration of arch. components is most efficient for this application?
  • Requires insight about the performance tradeoffs and interactions between those architectural components.

• Design space considerations
  • Hardware I want to cover: memory tech., accelerators, etc.
  • Applications and parallel programming models I would use.
  • Simulators and hardware models that are available.
Our Design Space Exploration parameters

SIMULATOR PARAMETERS/COMPONENTS

<table>
<thead>
<tr>
<th>#Cores</th>
<th>Issue width / ROB</th>
<th>Frequency</th>
<th>Cache Size L3 / L2</th>
<th>Memory BW</th>
<th>SIMD width</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 / 40 (lo-end)</td>
<td>1.5 GHz</td>
<td>96MB/1MB</td>
<td>4ch DDR4</td>
<td>128-bit</td>
</tr>
<tr>
<td>32</td>
<td>4 / 180 (med.)</td>
<td>2.0 GHz</td>
<td>64MB/512MB</td>
<td>8ch DDR4</td>
<td>256-bit</td>
</tr>
<tr>
<td>64</td>
<td>6 / 224 (high)</td>
<td>2.5 GHz</td>
<td>32MB/256K</td>
<td></td>
<td>512-bit</td>
</tr>
<tr>
<td>8</td>
<td>8 / 300 (aggr.)</td>
<td>3.0 GHz</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

864 Detailed Arch Simulations per Application

Applications (MPI+OMP)
- NAS BT-MZ
- NAS SP-MZ
- HYDRO
- LULESH
- Specfem3D
- HPCG

We perform a large experimental campaign:
- 6 Apps, 864 Simulations per Application
- 256 nodes, up to 64 cores per node
## Our Design Space Exploration parameters

### SIMULATOR PARAMETERS/COMPONENTS

<table>
<thead>
<tr>
<th>Cores</th>
<th>Issue width / ROB</th>
<th>Frequency</th>
<th>Cache Size L3 / L2</th>
<th>Memory BW</th>
<th>SIMD width</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 / 30 (lo-end)</td>
<td>1.5</td>
<td>96MB/1MB</td>
<td>4ch DDR4</td>
<td>128</td>
</tr>
<tr>
<td>32</td>
<td>4 / 180 (med.)</td>
<td>2.0</td>
<td>64MB/512MB</td>
<td>8ch DDR4</td>
<td>256</td>
</tr>
<tr>
<td>64</td>
<td>6 / 224 (high)</td>
<td>2.5</td>
<td>32MB/256K</td>
<td></td>
<td>512</td>
</tr>
<tr>
<td>8/300 (aggr.)</td>
<td>3.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

864 Detailed Arch Simulations per Application

### Applications (MPI+OMP)
- NAS BT-MZ
- NAS SP-MZ
- HYDRO
- LULESH
- Specfem3D
- HPCG

We perform a large experimental campaign:
- 6 Apps, 864 Simulations per Application
- 256 nodes, up to 64 cores per node
Our simulation infrastructure

MUSA (Arch. Simulation infrastructure)

- Application Instruction Trace
- Dimemas
- TaskSim

Ramulator

Power models
- DRAMPower
- McPAT

Performance Data

Energy consumption data

"Data science"
SIMD width: performance

• With respect to 128-bit @ 32- and 64-core configurations:
  • 512-bit FP units yield 20-40% performance speed-up on HYDRO, BTMZ, SPECFEM3D and HPCG.SPMV. Up to 75% avg. in SPMZ.
  • LULESH is auto-vectorization unfriendly (at least for GCC).
SIMD width: energy-to-solution

- With respect to 128-bit, @ 32-core and 64-core configurations
  - In average, a core with 512-bit width consumes +60% Power [Watts]
  - 512-bit and wider configurations would require efficient auto-vectorization or manual tuning to achieve higher energy efficiency.
  - 256-bit width configurations are able to reduce slightly energy-to-solution.
Memory bandwidth: performance and power

• Performance
  - Memory bound apps like LULESH, HPCG benefit greatly (60%) in 8 channel configurations w.r.t 4 channel.
  - In other apps adding more channels does not affect performance at all.

• In 64 core configurations
  - Doubling the number of DDR channels increases the total node power consumption by 10%-20%.
  - 8 channel configurations provide a better energy to solution for LULESH and HPCG.
Custom-Application Architecture Designs

• We explore 4 aggressive architecture designs
  • In SPMZ we test 1024- and 2048-bit wide SIMD units.
  • For LULESH we test 16ch. DDR4 (300GB/s) and HBM (2TB/s).

<table>
<thead>
<tr>
<th></th>
<th>SPMZ-targeted architecture</th>
<th>LULESH-targeted architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best-DSE</td>
<td>Performance</td>
<td>Power</td>
</tr>
<tr>
<td>Vector+</td>
<td>1.13x performance; similar increase in power</td>
<td>1.07x performance; 0.53x reduction in power</td>
</tr>
<tr>
<td>Vector++</td>
<td>1.43x performance but 3.14x additional power</td>
<td>MEM++: 1.30x performance</td>
</tr>
</tbody>
</table>

Vector+: 1.13x performance; similar increase in power
Vector++: 1.43x performance but 3.14x additional power
MEM+: 1.07x performance; 0.53x reduction in power
MEM++: 1.30x performance
Issues & Lessons learned
Missing simulators and models

• Power models for state-of-the-art technologies
  • McPAT supports down to 22nm (we are already at ~7nm)
  • HBM-1/2 industry datasheets not available.

• Simulators and configurations for new memory technologies
  • HBM2/3 and other 3D stacked mem.
  • Support for tight interconnections.
Throughput issue in large-scale benchmarks

- Some large application benchmarks do not fully use the available compute resources.
  - Low shared-memory OpenMP parallelism.
    - With 64 cores, runtime cannot produce enough parallel tasks for all cores.
  - Compilers have troubles auto-vectorizing the code
    - Codes do not include the appropriate structures, directives or pragmas.

- Learning: these software issues can prevent us from properly evaluating the strengths and weaknesses of your designs.
Speedup simulation times

• Sampling simulation techniques could help us to reduce simulation time in large scale simulations
  • Applications are large but usually have very repetitive exec. patterns.
  • We can simulate in detail a few representative parts of the code and extrapolate the rest.

• Important questions and challenges about sampling
  • How do we select automatically the most representative regions?
  • How do we warmup the simulator to avoid cold caches?
  • How are you going to estimate the power consumption?
Open question

• How do we develop and integrate all the necessary pieces to create an infrastructure that allows us to simulate next-generation systems at reasonable speed and accuracy?
  • Things to discuss: Applications, Simulation techniques, Missing/outdated models, Academia collaboration, Priorities, etc.

Find all the results in our Paper (accepted at IPDPS’19): <https://upcommons.upc.edu/handle/2117/131511>

Thank you! Feel free to contact me on:
constantino.gomez@bsc.es
Backup slides
L2-, L3-Cache Size: Performance

• Fitting app working sets has a huge impact in performance.
  • Cache blocking software optimizations should be always encouraged.
  • w.r.t 512 KB, in most cases we observed that increasing L2 size beyond 512KB does not noticeably reduce MPKI.

• Based on our experiments, the best design point is:
  • ~1 MB Shared L3 per core. And ~512KB private L2.
Core Out-of-Order: performance and ETS

• Performance-Energy tradeoffs with respect to aggressive
  • In most of our benchmarks low-end configurations have better energy consumption but at the cost of some performance degradation.
  • In most cases ‘moderate’ configurations have similar performance and consume 20-25% less energy.