Performance monitoring on Intel Haswell platforms

Thomas Röhl (RRZE, Uni Erlangen-Nürnberg)
ZKI AK Supercomputing
20. October 2015
HPM event classes (Core)

- 413 configurable events per core
- 3 fixed-purpose events per core
- Classes:
  - CPU cycles
  - Instruction pipeline (issued, executed, retired)
  - Branch prediction
  - Caching behavior (Hits, Misses, Prefetch, TLBs, …)
  - ...
- Almost no floating-point events!
HPM event classes (Uncore)

- Specific events for each Uncore box with PMUs
- Uncore events cover the whole socket

<table>
<thead>
<tr>
<th>Uncore component</th>
<th>#Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>L3 cache segments</td>
<td>Huge amount of events!</td>
</tr>
<tr>
<td>Memory channels</td>
<td></td>
</tr>
<tr>
<td>QPI interconnect</td>
<td>But many are dedicated to system engineers for validation</td>
</tr>
<tr>
<td>PCIe interface</td>
<td></td>
</tr>
<tr>
<td>Socket internal ring</td>
<td></td>
</tr>
<tr>
<td>Power control</td>
<td></td>
</tr>
<tr>
<td>Ring to QPI</td>
<td></td>
</tr>
</tbody>
</table>
LIKWID suite tool overview

- likwid-topology – *Read topology of current system*
- likwid-pin – *Pin processes/threads to CPU cores*
- likwid-perfctr – *Hardware performance monitoring (HPM)*
- likwid-agent – *HPM monitoring daemon*
- likwid-bench – *Microkernel benchmark tool*
- likwid-setFrequencies – *Manipulate CPU frequency*
- Likwid-mpirun – *MPI wrapper with pinning with HPM integration*
HPM tool likwid-perfctr

- Provides 439 events per core
- Supports all Uncore boxes, RAPL and all event options
- Uses native Linux kernel interfaces
- Code instrumentation API (C/C++, Fortran, Lua, Java, Python)
- Complete API (C, C++, Lua)
- No virtual counters!
- No interpolations!
- No instruction-based sampling (only time-based)

Webpage: [https://github.com/RRZE-HPC/likwid](https://github.com/RRZE-HPC/likwid)
Measure FLOPs on Haswell?

- No, only partly
- Intel released:
- LIKWID offers:
  - AVX_INSTS.ALL
  - AVX_INSTS.LOADS
  - AVX_INSTS.STORES
  - AVX_INSTS.CALC
- Problem: vinsertf128 counted as calculation
### Accuracy of LIKWID metrics on Haswell

*Tested with likwid-perfctr and likwid-bench*

<table>
<thead>
<tr>
<th>Type</th>
<th>Avg. Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Branch prediction</td>
<td>0.20 %</td>
</tr>
<tr>
<td>Retired instructions</td>
<td>0.00 %</td>
</tr>
<tr>
<td>Load/store ratio</td>
<td>0.08 %</td>
</tr>
<tr>
<td>AVX FLOPs</td>
<td>0.05 %</td>
</tr>
<tr>
<td>L1 - L2 bandwidth</td>
<td>1.51 % (with ddot 5.27 %)</td>
</tr>
<tr>
<td>L2 - L3 bandwidth</td>
<td>1.14 %</td>
</tr>
<tr>
<td>µOPs retired</td>
<td>0.07 %</td>
</tr>
<tr>
<td>Memory bandwidth (HA)</td>
<td>0.13 %</td>
</tr>
<tr>
<td>Memory bandwidth (iMC)</td>
<td>0.18 %</td>
</tr>
</tbody>
</table>

Tests: load, store, copy, stream(_avx), triad(_avx), daxpy(_avx), ddot(_avx)

Machine: Intel Xeon E5-2695 v3 (SMT enabled)
Shortcomings of HPM on Haswell

- No FLOPs events (AVX_FLOPS.CALC not official)
- Instruction decomposition not possible
- Instruction to UOP decoding only partly
- UOP pipeline usage per execution unit
- Cache line sharing at L2 cache level
- CPU dispatch port events only partly SMT aware
- Interrupt generation by Uncore events
- Amount of split cache line loads
Conclusion

- Huge amount of HPM events (core + uncore)
- But some meaningful events missing
  many not relevant for performance monitoring
- LIKWID provides comprehensive support
- PAPI/perf_event mostly only core-local support
- (Not official) support for AVX floating-point calculation ops
- Many events are validated and accurate by LIKWID team
  (PAPI starts validation in near future)
Thanks for your attention

Questions?

https://github.com/RRZE-HPC/likwid
Thomas.Roehl@fau.de