Erlangen Regional Computing Center

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

LIKWID

HPC Café 14.07.2020

Thomas Gruber

HPC Service, RRZE Erlangen

- „Like I Knew What I'm Doing"
- Tool suite for performance-oriented programmers
- Developed by HPC group of RRZE since 2009
- Open source and active development

LIKWID TOOLS

- Goals:
  - Offer powerful tools for daily work
  - Simple access to derived metrics for performance engineering

LIKWID tools overview

LIKWID tool	Description
likwid-topology	System topology information
likwid-pin	Control CPU affinity of processes/threads
likwid-perfctr	Measure hardware performance events
likwid-bench	Microbenchmarking suite
likwid-setFrequencies	Manipulate CPU/Uncore frequencies
likwid-features	Manipulate CPU settings (prefetchers)
likwid-memsweeper	Cleanup caches and NUMA domains

likwid-mpirun

For hardware performance measurements at RRZE systems:

- PBS (emmy): **`:likwid`** property
- SLURM (meggie): **`-C hwperf`**

LIKWID – likwid-topology

```
$ likwid-topology
--------------------------------------------------------------------------------
CPU name:   Intel(R) Xeon(R) Gold 6246 CPU @ 3.30GHz
CPU type:   Intel Cascadelake SP processor
CPU stepping:  7
********************************************************************************
Hardware Thread Topology
********************************************************************************
Sockets:    2
Cores per socket: 12
Threads per core: 2
--------------------------------------------------------------------------------
HWThread Thread      Core     Socket       Available
0        0           0        0            *
[…]
```

LIKWID – likwid-topology

```
HWThread Thread        Core      Socket       Available
0        0             0         0            *
[…]
47    1      23    1      *
--------------------------------------------------------------------------------
Socket 0:( 0 24 1 25 2 26 3 27 4 28 5 29 6 30 7 31 8 32 9 33 …)
Socket 1:( 12 36 13 37 14 38 15 39 16 40 17 41 18 42 19 43 20 …)
--------------------------------------------------------------------------------
********************************************************************************
Cache Topology
********************************************************************************
Level:          1
Size:       32 kB
Cache groups:  ( 0 24 ) ( 1 25 ) ( 2 26 ) ( 3 27 ) ( 4 28 ) ( 5 29 ) …
```

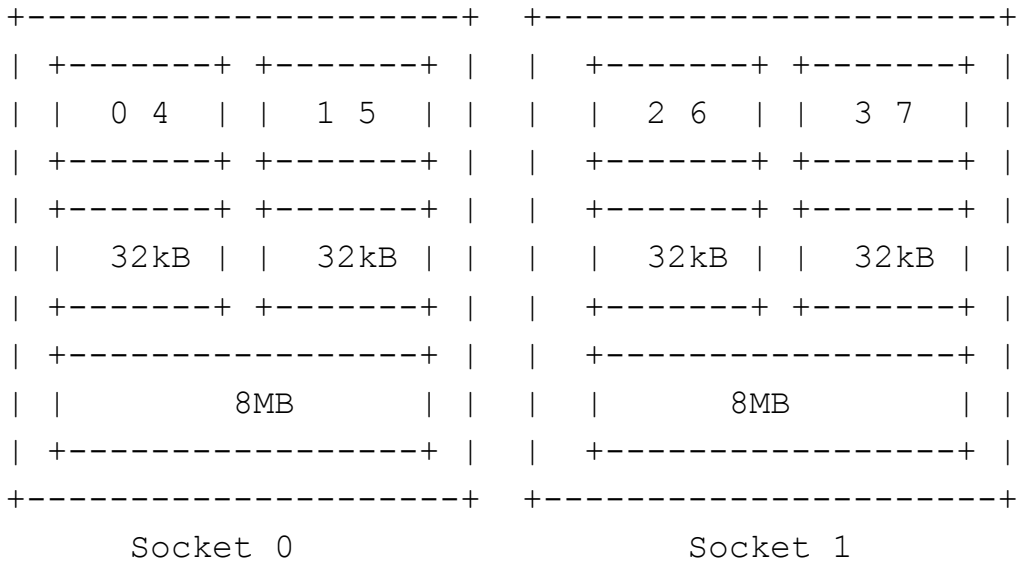
LIKWID – likwid-topology

```
**************************************************************
NUMA Topology
**************************************************************
NUMA domains:     2
--------------------------------------------------------------
Domain:         0
Processors: ( 0 1 2 3 4 5 6 7 8 9 10 11 24 25 26 27 28 29 30 31 32 33…)
Distances:      10 21
Free memory:        63556.7 MB
Total memory:       64072 MB
```

LIKWID 5.0 with Nvidia support: there is another section with GPU topology

LIKWID – likwid-pin

- Pin application threads/processes to distinct HW threads
- Supports most parallelization runtimes
  Pthreads, OpenMP, C++11 threads, …
- Supports affinity domains
  - N: whole node
  - Sx: CPU socket x
  - My: NUMA domain y
  - Cz: Last level cache z



- **`$ likwid-pin -c 0,1,2-3 ./a.out`** → {0,1,2,3}
- **`$ likwid-pin -c S1:1`** → {3}
- **`$ likwid-pin -c S0:0@S1:0`** → {0,2}

Logical indexing in affinity domain:
Physical HW threads first
{0,1,2,3,4,5,6,7}

- **`$ likwid-pin -c E:N:4:1:2`** → {0,1,2,3}
  4 threads, stride 1, chunksize 2 → jump over HT threads

LIKWID – likwid-perfctr

- Hardware performance monitoring
- Different measurement modes:
  - Start-to-end
  - Timeline (time-based sampling)
  - Stethoscope
  - MarkerAPI (code instrumentation)

All likwid tools use cpustr like `likwid-pin`

- `$ likwid-perfctr -c/-C <cpustr> -g <eventlist> (opts) ./app`
  - `-c` : Measure on cores in <cpustr>, `-C` : Measure & pin
  - `-g` : List of events or performance group (events + derived metrics)
- `$ likwid-perfctr -a`: List of supported performance groups
- `$ likwid-perfctr -e`: List of supported counters and events

LIKWID – likwid-perfctr

```
$ likwid-perfctr -C 0,1 -g L2 ./app
-------------------------------------------------------------
< output of ./app >
-------------------------------------------------------------
[…]
+-----------------------------+-----------+-----------+
|            Metric           |   Core 0  |   Core 1  |
+-----------------------------+-----------+-----------+
|      Runtime (RDTSC) [s]    |    2.6439 |    2.6439 |
|  L2D write bandwidth [MBytes/s] | 6744.8121 | 6743.6037 |
|  L2D write data volume [GBytes] |   17.8325 |   17.8293 |
|      <there is more>        |           |           |
+-----------------------------+-----------+-----------+
```

LIKWID – likwid-perfctr

- If interested in measurements of code regions, use MarkerAPI

```
#include <likwid.h>
LIKWID_MARKER_INIT; // in serial region
LIKWID_MARKER_REGISTER("Compute"); // in parallel region
LIKWID_MARKER_START("Compute"); // in parallel region
<code>
LIKWID_MARKER_STOP("Compute"); // in parallel region
LIKWID_MARKER_CLOSE;  // in serial region
```

Recommended: Reduces startup overhead

Multiple regions and nesting allowed

- Compile with **-DLIKWID_PERFMON**
- **$ likwid-perfctr -C 0,1 -g L2 -m ./app**

Activate MarkerAPI mode

```
$ likwid-perfctr -C 0,1 -g L2 -m ./app
Region Compute, Group 1: L2
+--------------------+-----------+-----------+
|    Region Info     |   Core 0  |   Core 1  |
+--------------------+-----------+-----------+
| RDTSC Runtime [s]  | 77.724980 | 77.725490 |
|     call count     |      1000 |      1000 |
+--------------------+-----------+-----------+
[…]
+-----------------------------+------------+------------+
|            Metric           |   Core 0   |   Core 1   |
+-----------------------------+------------+------------+
|      Runtime (RDTSC) [s]    |    77.7250 |    77.7255 |
|    L2 bandwidth [MBytes/s]  | 10275.9891 | 10272.4909 |
|    L2 data volume [GBytes]  |   798.7010 |   798.4344 |
+-----------------------------+------------+------------+
```

Region name and events

One column per core

Measured region time and region calls per thread

Metrics table (there are more)

- Wrapper for MPI for pinning and HPM

```
$ likwid-mpirun -nperdomain S:1 ./a.out
```

- One process per socket on all hosts

```
$ likwid-mpirun -pin S0:0-3_S1:0-3 ./a.out
```

- Two processes per node, each using 4 threads

```
$ likwid-mpirun -np <p> -g FLOPS_DP (-m) ./a.out
```

Measure DP Flops on with all procs (with MarkerAPI)

- Works with Intel MPI, OpenMP, Mvapich and SLURM

```
salloc -N X ; likwid-mpirun -np Y ./a.out
salloc -N X -C hwperf; likwid-mpirun -n Y -g L3 ./a.out
```

LIKWID- Wrap-up

- LIKWID provides easy-to-use interfaces to topology, affinity and hardware performance counting
- Can be integrated in C/C++ and Fortran code
- Big worldwide community

- If you want to add it to your projects, feel free to contact us

- Just interested in the numbers?
  Try our job monitoring infrastructure

- https://hpc.fau.de/research/tools/likwid/