Energy measurement on Intel architectures

Martin Golasowski

martin.golasowski@vsb.cz

IT4Innovations
national01$#&0
supercomputing
center@#01%101

2nd of March 2018

Agenda

The Hardware

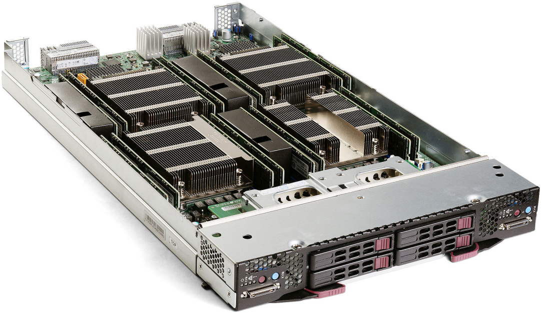
- Intel Turbo Boost and Enhanced SpeedStep
- Intel RAPL

The Tools

- The Linux way - powercap
- x86_adapt
- likwid
- PAPI
- perf
- tiptop
- Intel Xeon Phi

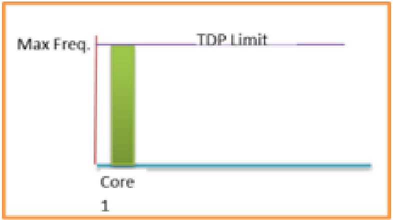
Interpreting results

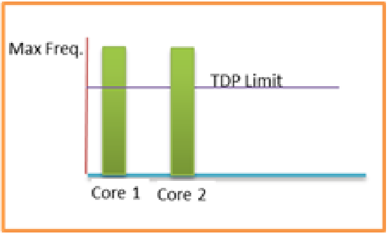
The Hardware

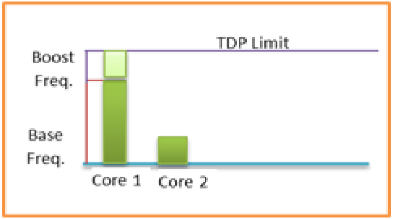


Intel Turbo Boost

Thermal Design Power (TDP) - maximum amount of power required to dissipate by the cooling system





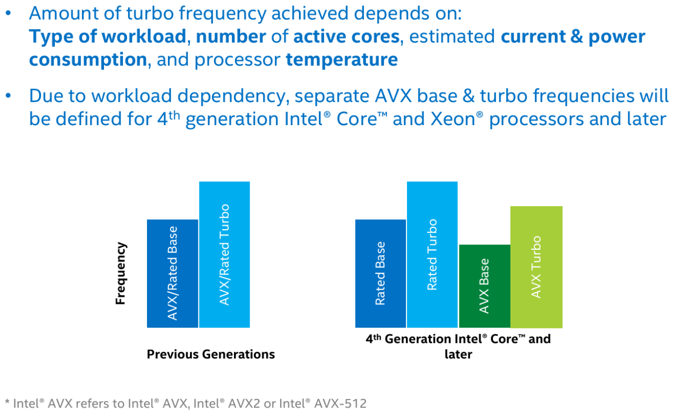


Intel® Xeon® Processor E5-2680 v3 - TDP: 120W

https://ark.intel.com/products/81908/Intel-Xeon-Processor-E5-2680-v3-30M-Cache-2_50-GHz

Turbo Boost & AVX Base Frequencies

With Haswell and later (also includes KNL):



- Amount of turbo frequency achieved depends on:
  Type of workload, **number** of **active cores**, estimated **current & power consumption**, and processor **temperature**
- Due to workload dependency, separate AVX base & turbo frequencies will be defined for 4th generation Intel® Core™ and Xeon® processors and later

* Intel® AVX refers to Intel® AVX, Intel® AVX2 or Intel® AVX-512

(Image: Intel)

Turbo Boost MAX 3.0 - available from Broadwell-EP
AVX base frequency is set per core - better granularity.

Intel Enhanced SpeedStep

Approximate CPU power consumption:

$$P = CV^2 f$$

where:
C is capacitance of the processor circuitry (fixed)
V input voltage
f frequency

- Exposed through ACPI since Pentium M - `intel_pstate` module
- Set of P-states: voltage/frequency pairs of operating points
- Switching between states has latency
- OS controlled policies - Linux `cpufreq` governors

Processor P and C states

C-states

- Idle modes - how deep processor sleeps
- C0 - Running ⇒ C6 - Maximal voltage reduction

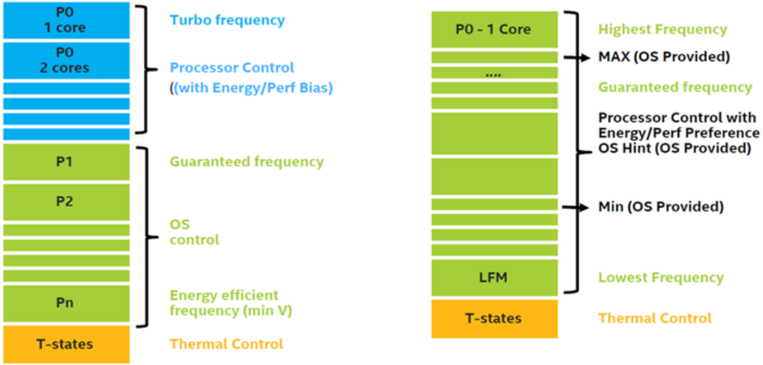
Transistion between states introduces latency - max. C-state can be forced

```
Kernel parameter: intel_idle.max_cstate=0

Verification: $ cat /sys/module/intel_idle/parameters/max_cstate
              9
```

P-states

- Operating modes - frequency/voltage pair
- Controlled by OS (ACPI)
- Skylake introduces autonomy

Intel Speed Shift



P0
1 core

P0
2 cores

Turbo frequency

Processor Control
((with Energy/Perf Bias)

P1

Guaranteed frequency

P2

OS
control

Pn

Energy efficient
frequency (min V)

T-states

Thermal Control

P0 - 1 Core

Highest Frequency

MAX (OS Provided)

....

Guaranteed frequency

Processor Control with
Energy/Perf Preference
OS Hint (OS Provided)

Min (OS Provided)

LFM

Lowest Frequency

T-states

Thermal Control

Intel® SpeedStep® Technology – Energy/Performance Bias effective with Turbo Range

Intel® Speed Shift Technology

cpupower - Linux cpufreq interface

```
 [root@cn11 ~]# cpupower frequency-info
analyzing CPU 0:
  driver: intel_pstate
  CPUs which run at the same hardware frequency: 0
  CPUs which need to have their frequency coordinated by software: 0
  maximum transition latency:  Cannot determine or is not supported.
  hardware limits: 1.20 GHz - 3.10 GHz
  available cpufreq governors: performance powersave
  current policy: frequency should be within 1.20 GHz and 3.10 GHz.
                  The governor "powersave" may decide which speed to use
                  within this range.
  current CPU frequency: 1.30 GHz (asserted by call to hardware)
  boost state support:
    Supported: no
    Active: no
    3000 MHz max turbo 4 active cores
    3000 MHz max turbo 3 active cores
    3100 MHz max turbo 2 active cores
    3100 MHz max turbo 1 active cores
```

Also available in: `/sys/devices/system/cpu ...`

More information:

https://www.kernel.org/doc/html/latest/admin-guide/pm/cpufreq.html

Running Average Power Limit - RAPL

- Software-based power meter in CPU
- Metrics available through MSRs
- Used by Intel Turbo Boost
- Introduced in Sandy Bridge microarchitecture

HUABANG
0 0 0 0 0 6
kWh
10000 1000 100 10 1 0.1
DMM862-4
SINGLE-PHASE WATT-HOUR METER
IEC62053-11 ② IP53 2008
230V 10(40)A 50Hz 360rev/kWh
№ 060257427
ZHEJIANG HUABANG ELECTRIC POWER CO.,LTD.

RAPL Domains Hierarchy

- **Package** - Socket
- **Power Plane 0 (PP0)** - Individual CPU cores
- **Power Plane 1 (PP1)** - Uncore devices
- **DRAM** - RAM memory

Grain of Salt

Always refer to CPU datasheet, available domains are model-specific. For example PP0/1 may not be available on some Haswell CPUs.

RAPL Interface - Machine Specific Registers

RDMSR/WRMSR - Privileged instructions for accessing MSRs

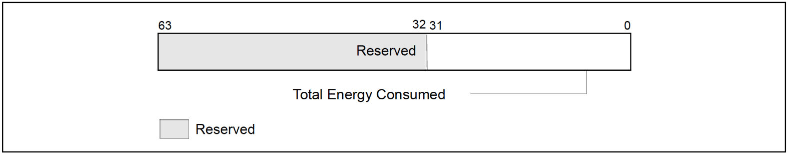


Figure 14-33. MSR_PKG_ENERGY_STATUS MSR

More info: System Programming Guide, Chapter 14.9.

Measurement units and increments

Units

Values from **MSR_XXX_ENERGY_STATUS** are not final.
Use following formula to obtain correct values:

$$x_{value} = c \cdot \frac{1}{2^m}$$

where:
c is value obtained from the STATUS register
m value of multiplier provided by **MSR_RAPL_POWER_UNIT**

Value overflow

The STATUS register is updated in 1̃ms interval and wraps in 6̃0s, earlier in case of heavy load.

The Tools

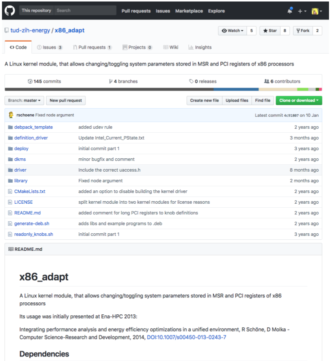
Linux Power Capping Framework

- Devices exposed through sysfs hierarchy
- Using intel_rapl kernel module

```
  [root@cn11 ~]# ls -R1 /sys/devices/virtual/powercap/inte
...
/sys/devices/virtual/powercap/intel-rapl/intel-rapl:0:
constraint_0_max_power_uw
constraint_0_name
constraint_0_power_limit_uw
constraint_0_time_window_us
constraint_1_max_power_uw
...
```

x86_adapt

- Linux kernel module and library
- Secure access to MSR and PCI registers from userspace
- Useful for building custom tools



tud-zih-energy / x86_adapt

A Linux kernel module, that allows changing/toggling system parameters stored in MSR and PCI registers of x86 processors

145 commits · 4 branches · 0 releases · 6 contributors

rschoene Fixed node argument — Latest commit 4c91807 on 10 Jan

debpack_template	added udev rule	2 years ago
definition_driver	Update Intel_Current_PState.txt	3 months ago
deploy	initial commit part 1	3 years ago
dkms	minor bugfix and comment	2 years ago
driver	Include the correct uaccess.h	8 months ago
library	Fixed node argument	2 months ago
CMakeLists.txt	added an option to disable building the kernel driver	2 years ago
LICENSE	split kernel module into two kernel modules for license reasons	2 years ago
README.md	added comment for long PCI registers to knob definitions	2 years ago
generate-deb.sh	adds libs and example programs to .deb	2 years ago
readonly_knobs.sh	initial commit part 1	3 years ago

README.md

x86_adapt

A Linux kernel module, that allows changing/toggling system parameters stored in MSR and PCI registers of x86 processors

Its usage was initially presented at Ena-HPC 2013:

Integrating performance analysis and energy efficiency optimizations in a unified environment, R Schöne, D Molka - Computer Science-Research and Development, 2014, DOI:10.1007/s00450-013-0243-7

Dependencies

More info: https://github.com/tud-zih-energy/x86_adapt

Using x86_adapt

Individual MSRs are available as r/w knobs defined in the library.

- API available through `libx86_adapt.so`
- Populates `/dev/x86_adapt/*`

Available knobs listing:

```
Item 0: RESET
----------------
Item 1: Intel_xd_bit_disable
----------------
Item 2: Intel_PERF_GLOBAL_STATUS
----------------
Item 3: Intel_RAPL_Pckg_Energy
...
...
```

x86_adapt - Using C API

```
#include <stdio.h>
#include <x86_adapt.h>
...
        // Lookup RAPL cpu item
        const char* item_name = "Intel_RAPL_Pckg_Energy";
        int item_id = x86_adapt_lookup_ci_name(devtype, item_name);
        if(item_id < 0)
        {
                fprintf(stderr,"Could␣not␣find␣%s\n",item_name);
        }

        uint64_t result;
        int ret;
        if ((ret = x86_adapt_get_setting(fd_cpu,item_id,&result)) != 8)
        {
                fprintf(stderr,"Could␣not␣read␣item␣%d␣for␣cpu/die␣%d\n",item_id,CPU);
                return -1;
        }

        printf("CPU:␣%d␣|␣MSR_PKG_ENERGY_STATUS_MSR␣%llu\n",CPU, result);
...
```

likwid

- Set of CLI tools
- C API
- Performance and energy measurement
- OpenMP and MPI support
- Benchmarking

Power related tools:

- **likwid-topology**
- **likwid-powermeter**
- **likwid-perfscope**

Developed by: Regionales RechenZentrum Erlangen (RRZE)

https://github.com/RRZE-HPC/likwid

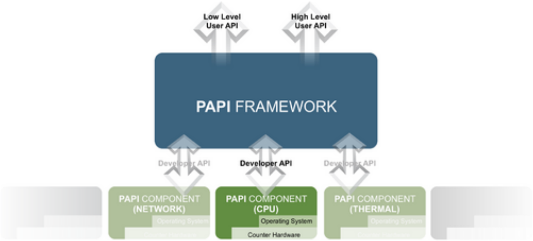
likwid-powermeter

Continuous measurement for selected CPU:

```
[root@cn11 ~]# likwid-powermeter -c 0 -s 2s
--------------------------------------------------------------------------------
CPU name: Intel(R) Xeon(R) CPU E5-2665 0 @ 2.40GHz
CPU type: Intel Xeon SandyBridge EN/EP processor
CPU clock: 2.40 GHz
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Runtime: 2.0007 s
Measure for socket 0 on CPU 0
Domain PKG:
Energy consumed: 32.4096 Joules
Power consumed: 16.1991 Watt
Domain PP0:
Energy consumed: 6.12715 Joules
Power consumed: 3.0625 Watt
Domain DRAM:
Energy consumed: 1.58621 Joules
Power consumed: 0.792828 Watt
--------------------------------------------------------------------------------
```

PAPI - Performance API

- Effort to provide portable API for accessing hardware performance counters
- Supports CPUs, network, accelerators, etc.
- Many features: custom events, timers, multiplexing, statistics



Developed by: University of Tennessee - Innovative Computing Laboratory

http://icl.cs.utk.edu/projects/papi/wiki/PAPIC:Overview

PAPI - Basic concepts

Components, Counters, Events

- High Level API
  - Easy to use (10 functions)
  - **Only preset events on the CPU**

- Low Level API
  - User-defined groups of Events
  - All PAPI components
  - Including native events

RAPL events available only as native!

PAPI - Components

Available components on a Sandy Bridge node (papi_component_avail):

```
...
Name:net         Linux network driver statistics
                 Native: 80, Preset: 0, Counters: 320

Name:rapl        Linux SandyBridge RAPL energy measurements
                 Native: 14, Preset: 0, Counters: 14

Name:stealtime   Stealtime filesystem statistics
                 Native: 33, Preset: 0, Counters: 33
...
```

PAPI - RAPL Events

Available **native** RAPL events on a Sandy Bridge node: (papi_native_avail):

```
Native Events in Component: rapl
rapl:::THERMAL_SPEC:PACKAGE0
rapl:::THERMAL_SPEC:PACKAGE1
rapl:::MINIMUM_POWER:PACKAGE0
rapl:::MINIMUM_POWER:PACKAGE1
rapl:::MAXIMUM_POWER:PACKAGE0
rapl:::MAXIMUM_POWER:PACKAGE1
rapl:::MAXIMUM_TIME_WINDOW:PACKAGE0
rapl:::MAXIMUM_TIME_WINDOW:PACKAGE1
rapl:::PACKAGE_ENERGY:PACKAGE0
rapl:::PACKAGE_ENERGY:PACKAGE1
rapl:::DRAM_ENERGY:PACKAGE0
rapl:::DRAM_ENERGY:PACKAGE1
rapl:::PP0_ENERGY:PACKAGE0
rapl:::PP0_ENERGY:PACKAGE1
```

PAPI - Low level API demo

```c
int event_code = -1;
PAPI_event_name_to_code("rapl:::PACKAGE_ENERGY:PACKAGE0", &event_code);

PAPI_event_info_t einfo;
PAPI_get_event_info(event_code, &einfo);


printf("Event␣symbol:␣%s\n", einfo.symbol);
printf("Description:␣%s\n\n", einfo.long_descr);

int event_set = PAPI_NULL;

// Create empty event set
if((ret = PAPI_create_eventset(&event_set)) != PAPI_OK) {
        handle_err(ret);
}

// Add RAPL event to event set
if((ret = PAPI_add_event(event_set, event_code)) != PAPI_OK) {
        handle_err(ret);
}

// Start collecting events
if((ret = PAPI_start(event_set)) != PAPI_OK) {
        handle_err(ret);
}

printf("Doing␣some␣FLOPs...\n");
```

PAPI - Low level API demo

```
...
FLOPS
....
// Stop collecting events
long long values[1]; // For one event
if((ret = PAPI_stop(event_set, values)) != PAPI_OK) {
        handle_err(ret);
}

printf("Energy␣consumed␣on␣PKG0:␣%lld␣%s␣\n", values[0], einfo.units);
```

perf - Linux profiling tool

- Common profiling tool in Linux
- Measuring, sampling, analysis
- Uses counters exposed by kernel

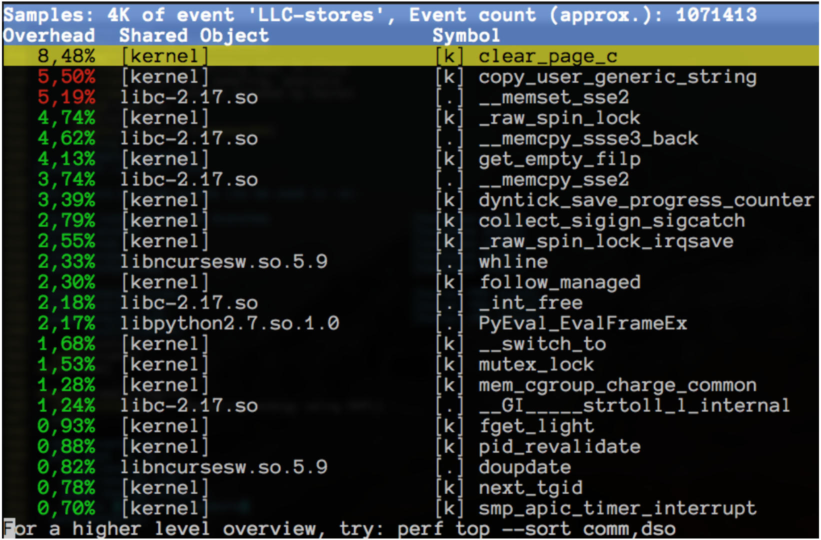
```
[root@cn11 ~]# perf list

List of pre-defined events (to be used in -e):

  branch-instructions OR branches [Hardware event]
  branch-misses                   [Hardware event]
  bus-cycles                      [Hardware event]
  cache-misses                    [Hardware event]
  cache-references                [Hardware event]
  ...
  power/energy-cores/             [Kernel PMU event]
  power/energy-pkg/               [Kernel PMU event]
  power/energy-ram/               [Kernel PMU event]
```

perf - Measuring energy using RAPL

```
perf stat -a -e \
  power/energy-pkg/,\
  power/energy-ram/,\
  power/energy-cores/,\
  cycles [binary-to-measure]
```

```
       time       counts   unit events
0.087152594         3,24 Joules power/energy-pkg/
0.087152594         0,11 Joules power/energy-ram/
0.087152594         0,92 Joules power/energy-cores/
0.087152594  137 374 362        cycles
```

perf - Real time monitoring

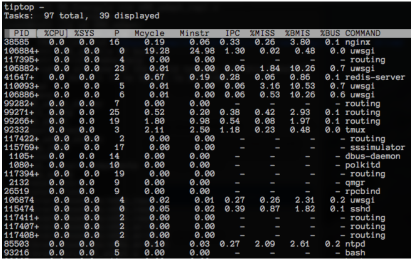


```
Samples: 4K of event 'LLC-stores', Event count (approx.): 1071413
Overhead  Shared Object           Symbol
   8,48%  [kernel]                [k] clear_page_c
   5,50%  [kernel]                [k] copy_user_generic_string
   5,19%  libc-2.17.so            [.] __memset_sse2
   4,74%  [kernel]                [k] _raw_spin_lock
   4,62%  libc-2.17.so            [.] __memcpy_ssse3_back
   4,13%  [kernel]                [k] get_empty_filp
   3,74%  libc-2.17.so            [.] __memcpy_sse2
   3,39%  [kernel]                [k] dyntick_save_progress_counter
   2,79%  [kernel]                [k] collect_sigign_sigcatch
   2,55%  [kernel]                [k] _raw_spin_lock_irqsave
   2,33%  libncursesw.so.5.9      [.] whline
   2,30%  [kernel]                [k] follow_managed
   2,18%  libc-2.17.so            [.] _int_free
   2,17%  libpython2.7.so.1.0     [.] PyEval_EvalFrameEx
   1,68%  [kernel]                [k] __switch_to
   1,53%  [kernel]                [k] mutex_lock
   1,28%  [kernel]                [k] mem_cgroup_charge_common
   1,24%  libc-2.17.so            [.] __GI_____strtoll_l_internal
   0,93%  [kernel]                [k] fget_light
   0,88%  [kernel]                [k] pid_revalidate
   0,82%  libncursesw.so.5.9      [.] doupdate
   0,78%  [kernel]                [k] next_tgid
   0,70%  [kernel]                [k] smp_apic_timer_interrupt
For a higher level overview, try: perf top --sort comm,dso
```

tiptop - Top for Hardware Performance counters

- Real-time diplay of IPC, cache misses, etc.
- ncurses base top-like

Developed by: Inria http://tiptop.gforge.inria.fr/



```
tiptop -
Tasks:  97 total,  39 displayed

  PID [ %CPU] %SYS    P   Mcycle   Minstr   IPC  %MISS  %BMIS  %BUS COMMAND
38585    0.0   0.0   16     0.19     0.06  0.33   0.26   3.80   0.1 nginx
106884+   0.0   0.0    0    19.28    24.98  1.30   0.02   0.48   0.0 uwsgi
117395+   0.0   0.0    4     0.00     0.00     -      -      -     - routing
106882+   0.0   0.0   23     0.01     0.00  0.06   1.84  10.26   0.7 uwsgi
41647+   0.0   0.0    2     0.67     0.19  0.28   0.06   0.86   0.1 redis-server
110093+   0.0   0.0    5     0.01     0.00  0.06   3.16  10.53   0.7 uwsgi
106886+   0.0   0.0    6     0.01     0.00  0.06   0.53  10.26   0.6 uwsgi
99282+   0.0   0.0    7     0.00     0.00     -      -      -     - routing
99271+   0.0   0.0   25     0.52     0.20  0.38   0.42   2.93   0.1 routing
99266+   0.0   0.0   19     1.80     0.98  0.54   0.08   1.97   0.1 routing
92332    0.0   0.0    3     2.11     2.50  1.18   0.23   0.48   0.0 tmux
117422+   0.0   0.0    2     0.00     0.00     -      -      -     - routing
115769+   0.0   0.0   17     0.00     0.00     -      -      -     - sssimulator
 1105+   0.0   0.0   14     0.00     0.00     -      -      -     - dbus-daemon
 1080+   0.0   0.0   10     0.00     0.00     -      -      -     - polkitd
117394+   0.0   0.0   19     0.00     0.00     -      -      -     - routing
 2132    0.0   0.0    9     0.00     0.00     -      -      -     - qmgr
26519    0.0   0.0    9     0.00     0.00     -      -      -     - rpcbind
106874    0.0   0.0    4     0.02     0.01  0.27   0.26   2.31   0.2 uwsgi
115474    0.0   0.0    0     0.05     0.02  0.39   0.87   1.82   0.1 sshd
117411+   0.0   0.0    2     0.00     0.00     -      -      -     - routing
117407+   0.0   0.0    2     0.00     0.00     -      -      -     - routing
117408+   0.0   0.0    2     0.00     0.00     -      -      -     - routing
85503    0.0   0.0    6     0.10     0.03  0.27   2.09   2.61   0.2 ntpd
93216    0.0   0.0    5     0.00     0.00     -      -      -     - bash
```

Power Measurement on Xeon Phi

- **Host:** ▸ PAPI[1] & ▸ RAPL (*Package*, *PowerPlane0*[2] & *DRAM*).
- **Coprocessor:**
  - On host: Use `micsmc -f` to get *Total Power*
  - On coprocessor:
    - `/sys/class/micras/power` (~50 msec updates), e.g.:

```
> cat /sys/class/micras/power
113000000
112000000
113000000            # Total instantaneous uWatt
221000000
16000000             # PCIe power uWatt
28000000             # 2x3 connector uWatt
69000000             # 2x4 connector uWatt
28000000 0 967000
32000000 0 1000000
31000000 0 1501000
```

    - Library *libmicmgmt* provides an API (see `man libmicmgmt`)

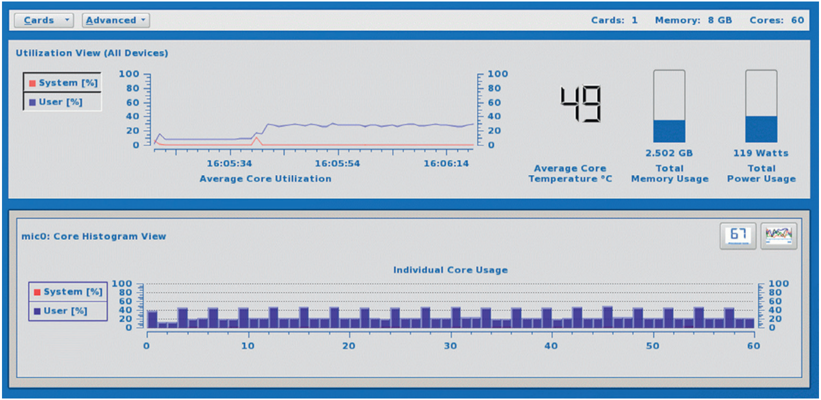
  More information ▸ here

Attention: Reading power values from KNC is "desctructive"(no idle power can be measured, but should be ~17 Watt)!

[1] For KNC modules *micpower* and *host_micpower* can be used

[2] All cores - except for HSW (see ▸ here)

micsmc - quite useful utility



Static measurement with **MERIC Tool**

- Multi-node energy measurement
- RAPL + x86_adapt
- Readex project

```
$ source meric/intel2017a/set_env
$ staticMERICtool/multiNodeStaticMeasureStart.sh --rapl
$ ./a.out
$ staticMERICtool/multiNodeStaticMeasureStop.sh --rapl

Runtime [s]: 6.59214
Overall energy consumption [J]: 1167
```

The tool needs a special permissions on the cluster.
For further info contact Ondřej Vysocký
`<ondrej.vysocky@vsb.cz>`.

Power vs. wall time tradeoff

- Reduce energy consumption via `cpufreq` or RAPL power capping
- Possible to find optimal tradeoff
  - Highly depends on load type and cluster utilization
- Support infrastructure (DRUPS, storage, monitoring,. . .) consumes a lot of energy

Thank you for your attention.