ECE 571 – Advanced Microprocessor-Based Design Lecture 16

Vince Weaver

http://www.eece.maine.edu/∼vweaver

vincent.weaver@maine.edu

21 March 2013

Project Reminder

- Topic Selection by Tuesday (March 26)
- Once you have selected a topic, start making a list of what machines, features, and benchmarks you need to ensure you have enough time to gather results.
  Some machine test setups will take longer to put together than others.

Project benchmarks

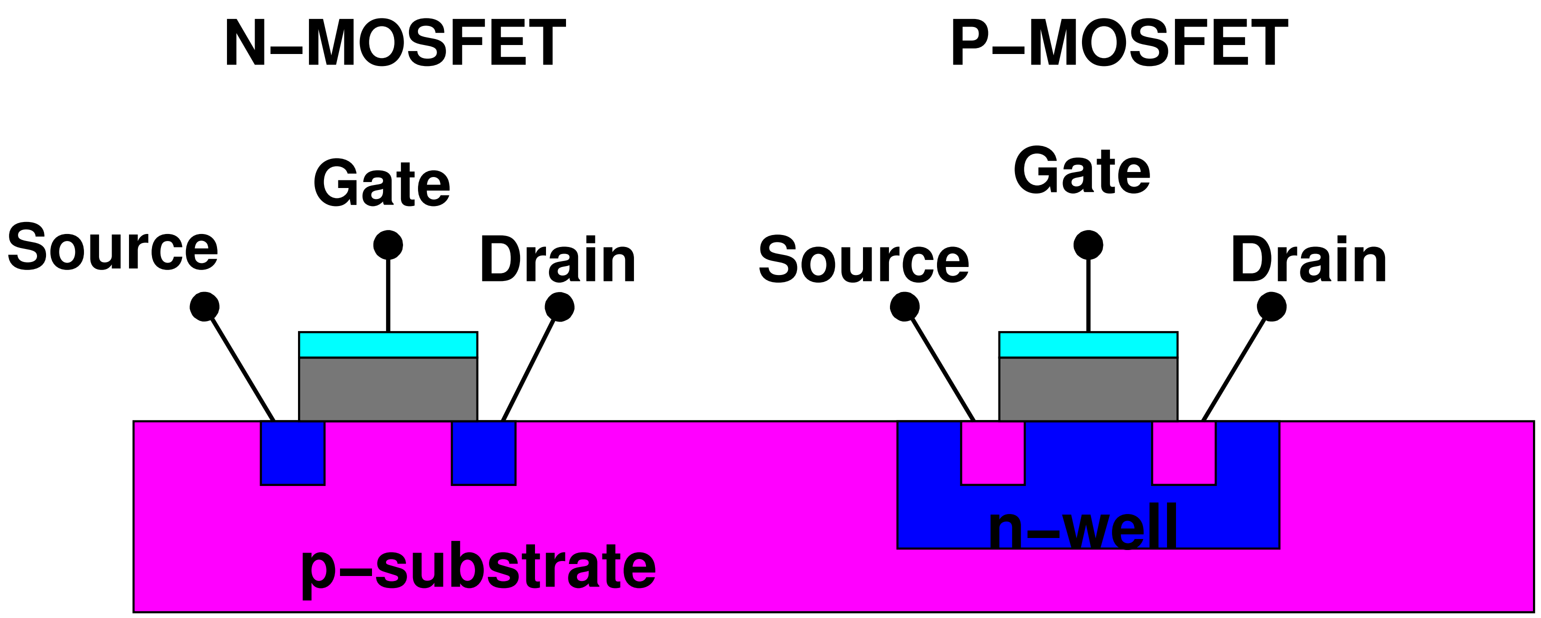
- Think of which benchmarks to use:
  SPEC 2006
  mibench
  HPC challenge
  Something custom?

CMOS Dynamic Power Review

- $P = C\Delta V V_{dd} \alpha f$
  Current is from charging/discharging capacitors ($C\Delta V V_{dd}$)
  So $P = IV = C\Delta V V_{dd}$
  The number of transitions is α (activity factor), times clock frequency (f).

- α is often approximated as 1/2, $\Delta V V_{dd}$ as V_{dd}^2
  $P = \frac{1}{2} C V_{dd}^2 f$

CMOS Transistors



CMOS Static Power

- Leakage Current – bigger issue as scaling smaller. Forecast at one point to be 20-50% of all chip power before mitigations were taken.
- Various kinds of leakage (Substrate, Gate, etc)
- Linear with Voltage: $P_{static} = I_{leakage}V_{dd}$

Leakage Mitigation

- SOI – Silicon on Insulator (AMD, IBM but not Intel)
- High-k dielectric – instead of SO2 use some other material for gate oxide (Hafnium)
- Transistor sizing – make only the critical transistors fast; non-critical ones can be made slower and less leakage prone
- Body-biasing

- Sleep transistors

Total Energy

- $E_{tot} = [P_{dyanmic} + P_{static}]t$
- $E_{tot} = [(C_{tot}V_{dd}^2\alpha f) + (N_{tot}I_{leakage}V_{dd})]t$

Delay

- $T_d = \frac{C_L V_{dd}}{\mu C_{ox}(\frac{W}{L})(V_{dd}-V_t)}$
- Simplifies to $f_{MAX} \sim \frac{(V_{dd}-V_t)^2}{V_{dd}}$
- If you lower f, you can lower V_{dd}

Thermal Issues

- Temperature and Heat Dissipation are closely related to Power
- If thermal issues, need heatsinks, fans, cooling

Energy Delay / Energy Delay Squared

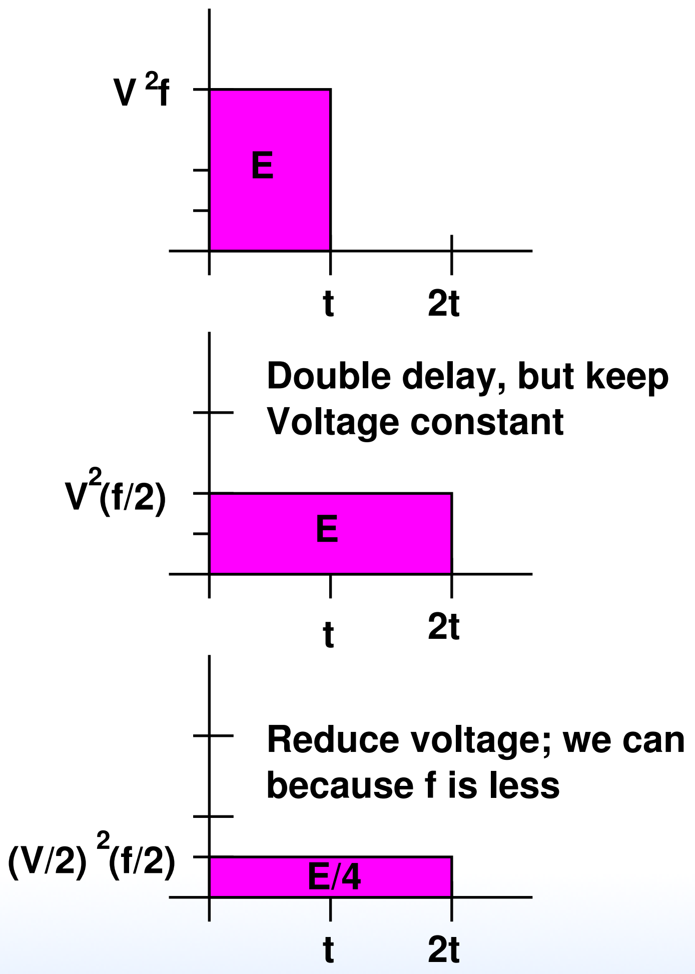
Lower is better.

Energy	Delay	ED	ED^2
5J	2s	$10Js$	$20Js^2$
5J	3s	$15Js$	$45Js^2$

Same ED, Different ED^2

Energy	Delay	ED	ED^2
5J	2s	$10Js$	$20Js^2$
2J	5s	$10Js$	$50Js^2$

Energy Example



Measuring Power and Energy

Why?

- New, massive, HPC machines use impressive amounts of power
- When you have 100k+ cores, saving a few Joules per core quickly adds up
- To improve power/energy draw, you need some way of measuring it

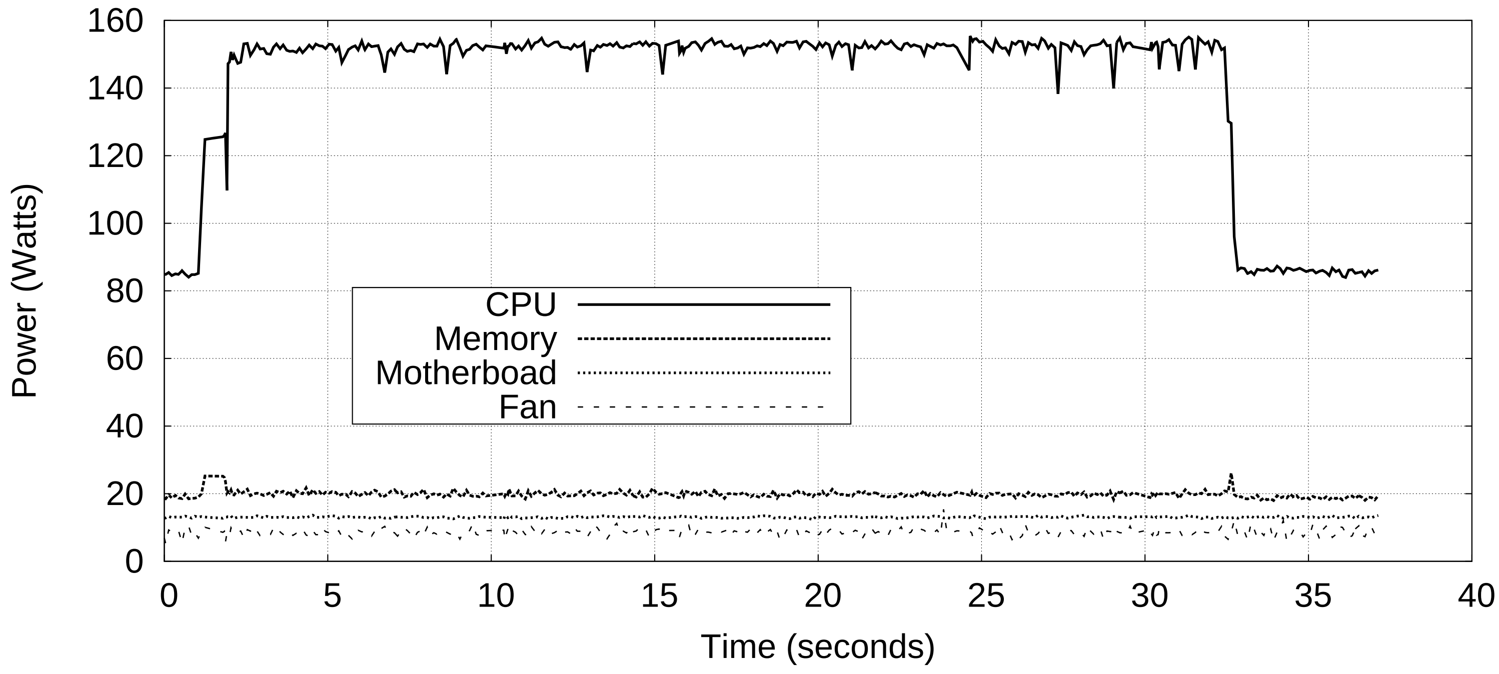
Energy/Power Measurement is Already Possible

Three common ways of doing this:

- Hand-instrumenting a system by tapping all power inputs to CPU, memory, disk, etc., and using a data logger
- Using a pass-through power meter that you plug your server into. Often these will log over USB
- Estimating power/energy with a software model based on system behavior

Existing Related Work

Plasma/dposv results with Virginia Tech's PowerPack



Powerpack

- Measure at Wall socket: WattsUp, ACPI-enabled power adapter, Data Acquisition System
- Measure all power pins to components (intercept ATX power connector?)
- CPU Power – CPU powered by four 12VDC pins.
- Disk power – measure 12 and 5VDC pins on disk power connecter

- Memory Power – DIMMs powered by four 5VDC pins
- Motherboard Power – 3.3V pins. Claim NIC contribution is minimal, checked by varying workload
- System fans

Shortcomings of current methods

- Each measurement platform has a different interface
- Typically data can only be recorded off-line, to a separate logging machine, and analysis is done after the fact
- Correlating energy/power with other performance metrics can be difficult

- PAPI (Performance API) is a cross-platform, open-source library for gathering performance-related data
- PAPI-C interface makes adding new power measuring

components straightforward

- PAPI can provide power/energy results in-line to running programs

More PAPI benefits

- One interface for all power measurement devices
- Existing PAPI code and instrumentation can easily be extended to measure power
- Existing high-level tools (Tau, VAMPIR, HPCToolkit, etc.) can be used with no changes
- Easy to measure other performance metrics at same time

Current PAPI Components

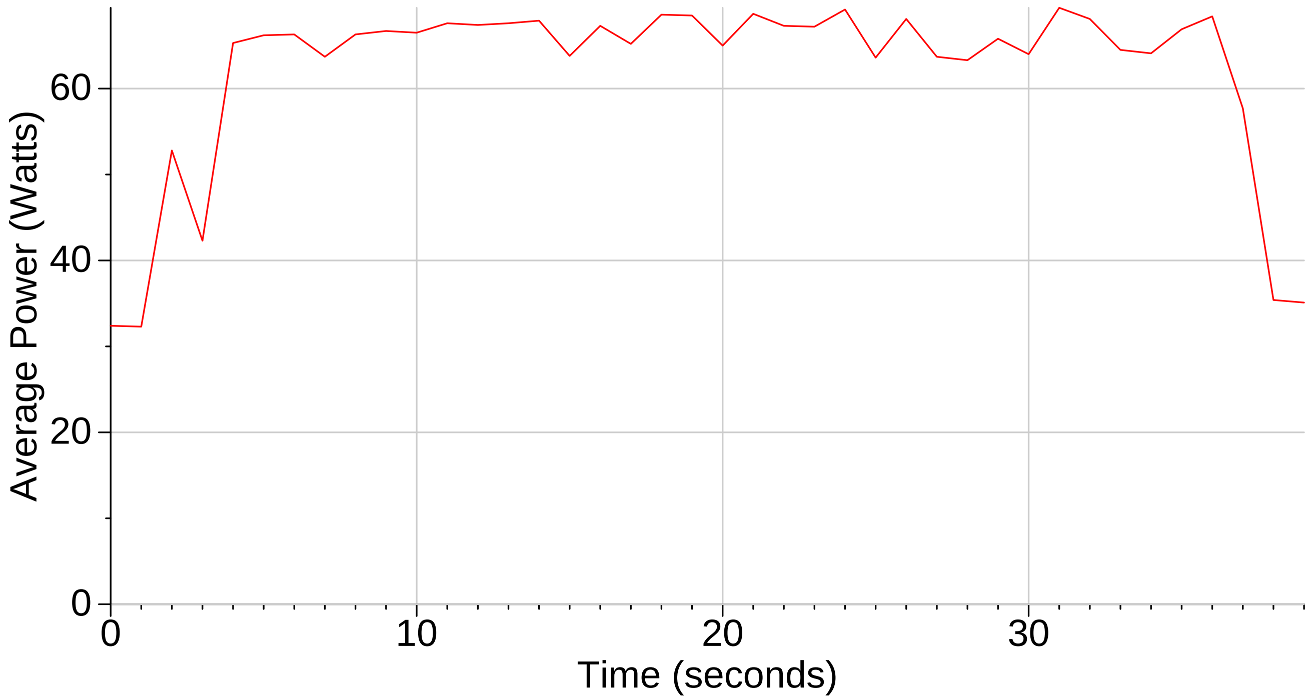
- Support for power components in recent PAPI 5.0 release
- Under development components found in `papi.git`

Watt's Up Pro Meter

Watt's Up Pro Features

- Can measure 18 different values with 1 second resolution (Watts, Volts, Amps, Watt-hours, etc.)
- Values read over USB
- Joules can be derived from power and time
- Can only measure system-wide

Watt's Up Pro Graph



PLASMA Cholesky Factorization N=10,000 threads=2

Measured on Core2 Laptop

RAPL

- **R**unning **A**verage **P**ower **L**imit
- Part of an infrastructure to allow setting custom per-package hardware enforced power limits
- User Accessible Energy/Power readings are a bonus feature of the interface

How RAPL Works

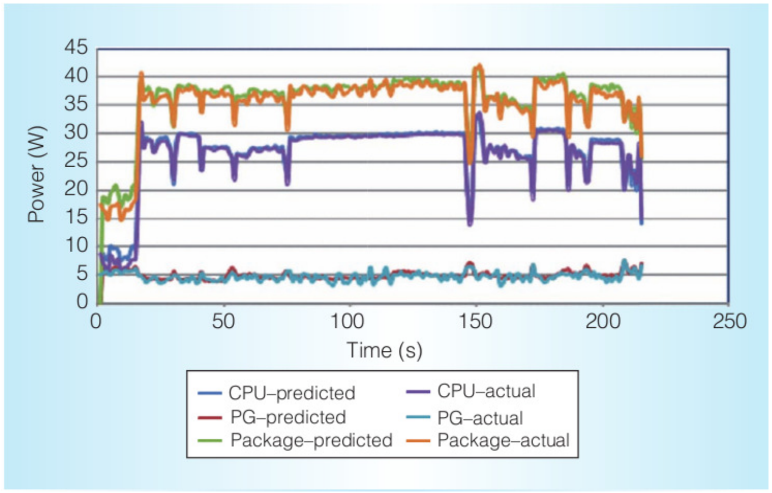
- RAPL is *not* an analog power meter
- RAPL uses a software power model, running on a helper controller on the main chip package
- Energy is estimated using various hardware performance counters, temperature, leakage models and I/O models
- The model is used for CPU throttling and turbo-boost, but the values are also exposed to users via a model-specific register (MSR)

Available RAPL Readings

- `PACKAGE_ENERGY`: total energy used by entire package
- `PP0_ENERGY`: energy used by "power plane 0" which includes all cores and caches
- `PP1_ENERGY`: on original Sandybridge this includes the on-chip Intel GPU
- `DRAM_ENERGY`: on Sandybridge EP this measures DRAM energy usage. It is unclear whether this is just the interface or if it includes all power used by all the DIMMs too

RAPL Measurement Accuracy

- Intel Documentation indicates Energy readings are updated roughly every millisecond (1kHz)
- Rotem at al. show results match actual hardware



Rotem et al. (IEEE Micro, Mar/Apr 2012)

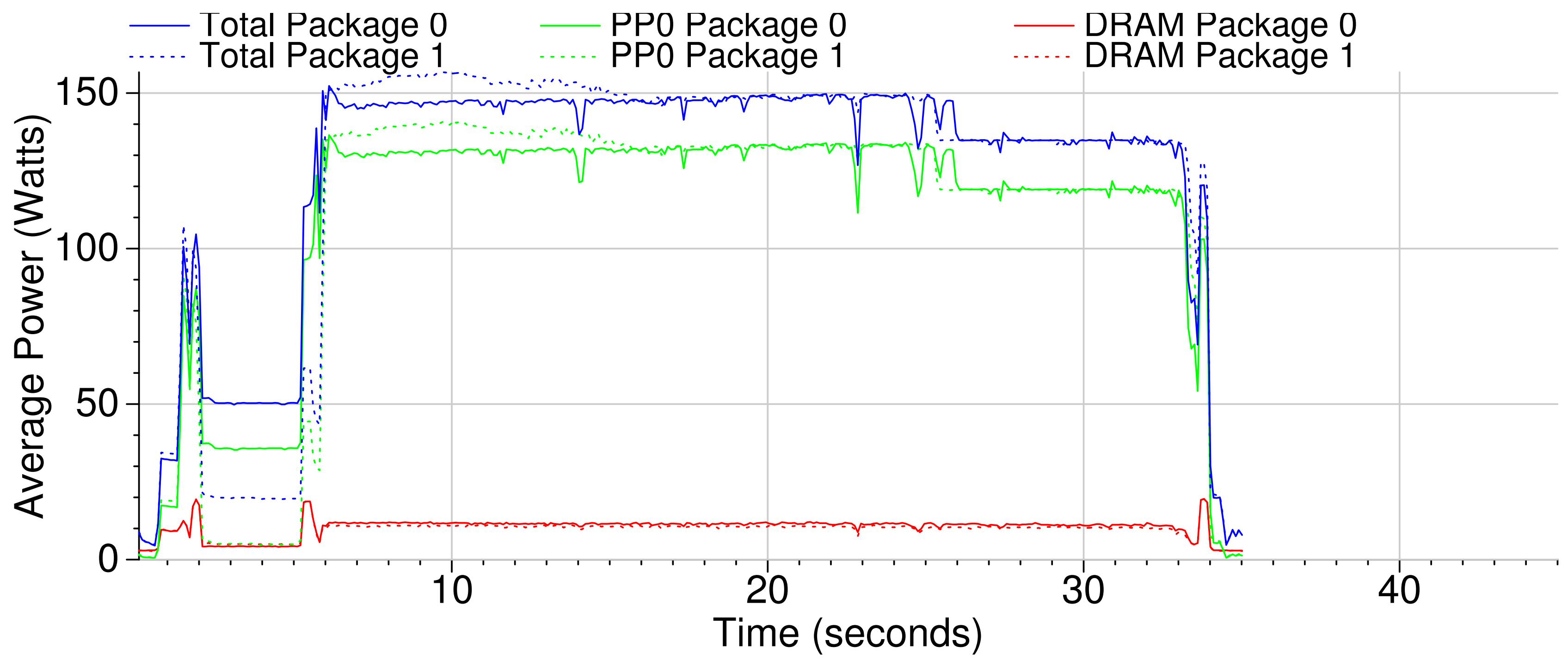
RAPL Accuracy, Continued

- The hardware also reports minimum measurement quanta. This can vary among processor releases. On our Sandybridge EP machine all Energy measurements are in multiples of 15.2nJ
- Power and Energy can vary between identical packages on a system, even when running identical workloads. It is unclear whether this is due to process variation during manufacturing or else a calibration issue.

RAPL PAPI Interface

- Access to RAPL data requires reading a CPU MSR register. This requires operating system support
- Linux currently has no driver and likely won't for the near future
- Linux does support an "MSR" driver. Given proper read permissions, MSRs can be accessed via `/dev/cpu/*/msr`
- PAPI uses the "MSR" driver to gather RAPL values

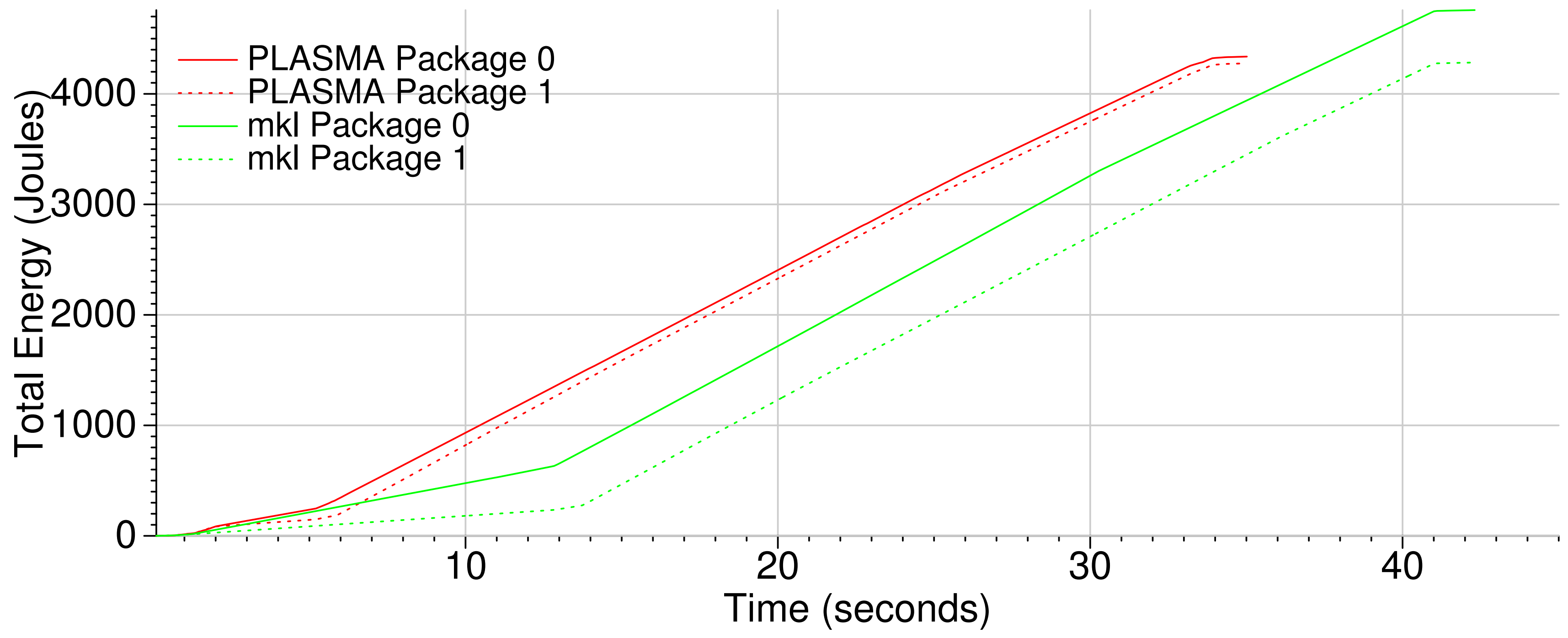
RAPL Power Plot



PLASMA Cholesky Factorization N=30,000 threads=16

Measured on SandyBridge EP

RAPL Energy Plot



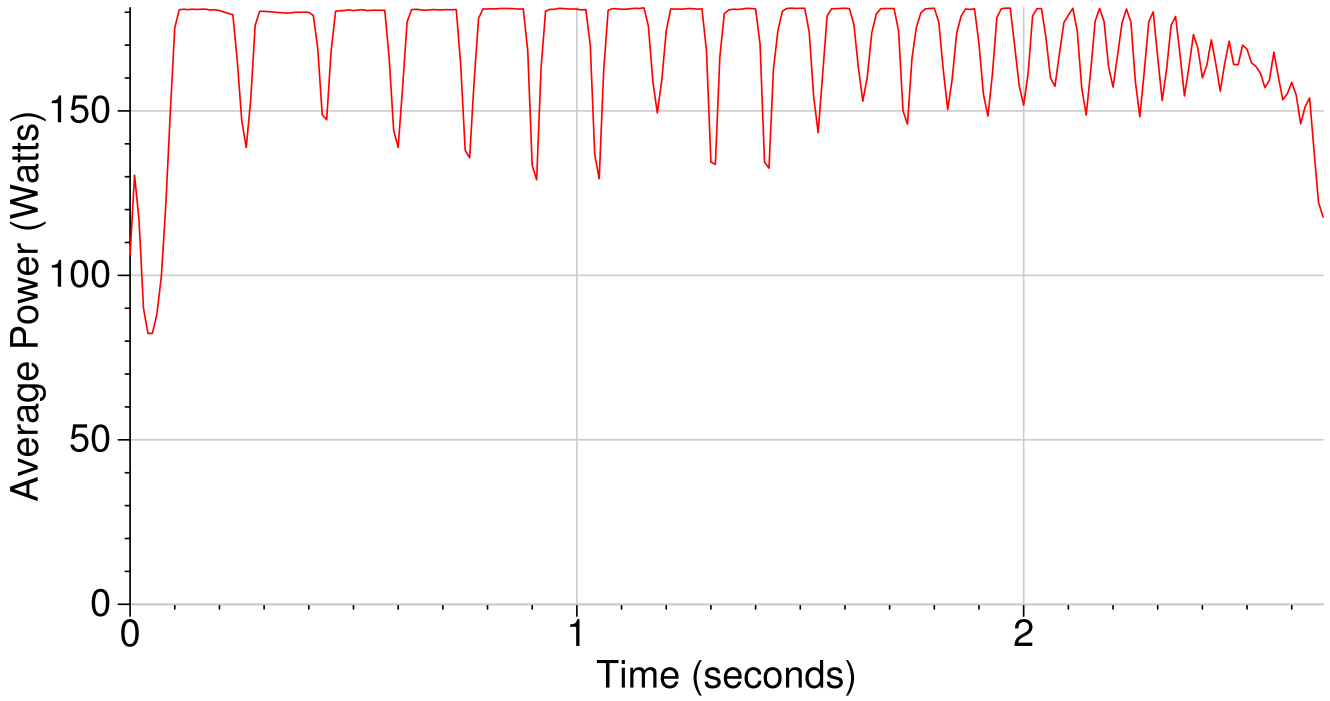
Cholesky Factorization N=30,000 threads=16

Measured on SandyBridge EP

NVML

- Recent NVIDIA GPUs support reading power via the NVIDIA Management Library (**NVML**)
- On Fermi C2075 GPUs it has milliwatt resolution within $\pm$5W and is updated at roughly 60Hz
- The power reported is that for the entire board, including GPU and memory

NVML Power Graph



MAGMA LU 10,000, Nvidia Fermi C2075

Near-future PAPI Components

These components do not exist yet, but support for them should be straightforward.

AMD Application Power Management

- Recent AMD Family 15h processors also can report “Current Power In Watts” via the Processor Power in the TDP MSR
- Support for this can be provided similar to RAPL
- We just need an Interlagos system where someone gives us the proper read permissions to `/dev/cpu/*/msr`

PowerMon 2

- PowerMon 2 is a custom board from RENCI
- Plugs in-line with ATX power supply.
- Reports results over USB
- 8 channels, 1kHz sample rate
- We have hardware; currently not working

PAPI-based Power Models

- There's a lot of related work on estimating energy/power using performance counters
- PAPI user-defined event infrastructure can be used to create power models using existing events
- Previous work (McKee et al.) shows accuracy to within 10%

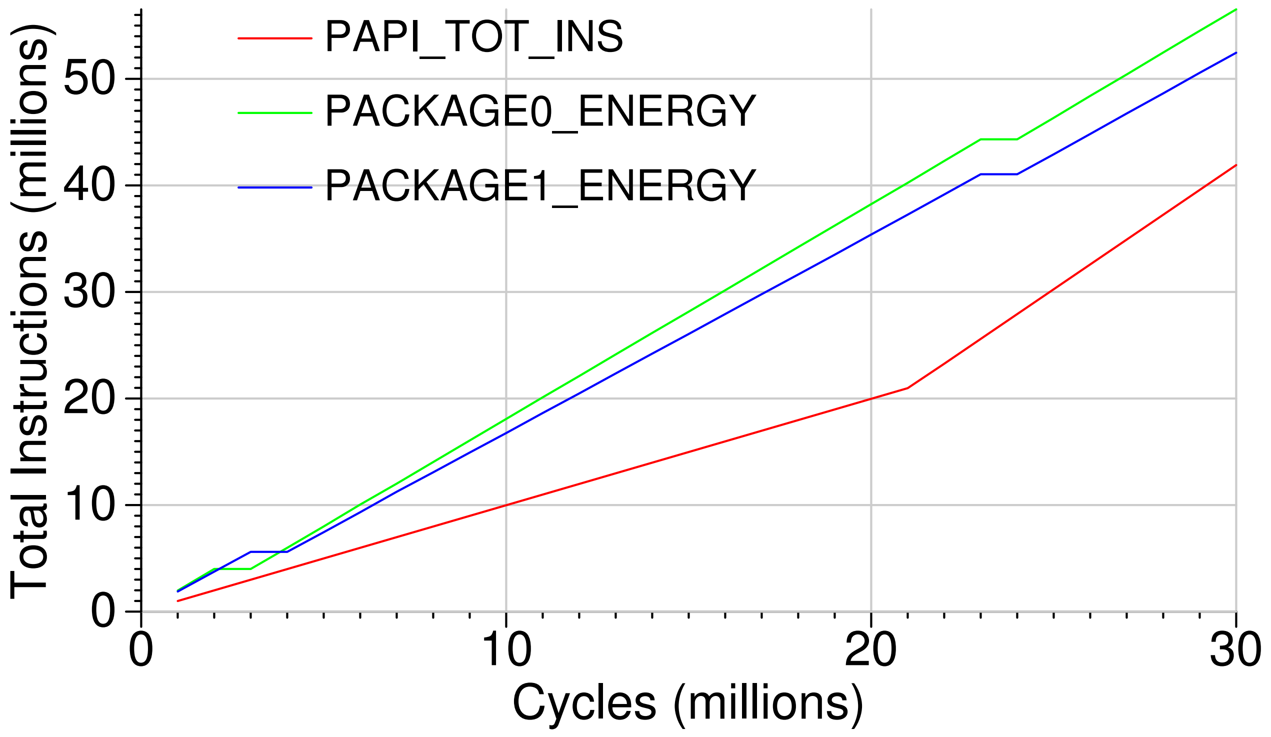
Measuring using PAPI

Measuring Energy/Power with PAPI is done the same as measuring any other event

Listing Events

```
>  papi_native_avail
====================================
 Events in Component: linux-rapl
====================================
------------------------------------
| PACKAGE_ENERGY:PACKAGE0
|   Energy used by chip package 0
------------------------------------
| PACKAGE_ENERGY:PACKAGE1
|   Energy used by chip package 1
------------------------------------
| DRAM_ENERGY:PACKAGE0
|   Energy used by DRAM on package 0
------------------------------------
```

Measuring Multiple Sources



INT/FP RAPL Test

Measured on SandyBridge EP

Beagle Board

- Has a 0.1 Ohm resistor you can measure voltage across to get current usage.
- Can read both sides of this using an on-chip ADC via i2c to calculate this with a complex set of equations.

PandaBoard

- Google this, people show which SMD pins to probe.

Raspberry Pi

- People using external power meters.