Reverse engineering power management on NVIDIA GPUs

A detailed overview

Martin Peres

Ph.D. student at LaBRI, Bordeaux

September 25, 2013

Summary

Introduction – Motivation

Power management in computers, why?

- To lower the power consumption of Data Centers;
- To increase the battery life of mobile computers;
- To have quieter and slimmer devices.

Reverse engineering power management, why?

Power management is:

- at least partially-assisted by software;
- almost entirely non-documented;
- often considered to be a manufacturer secret;
- thus poorly studied/implemented in open drivers;
- this is especially true in the GPU world.

Summary

Origin of the power consumption

Power consumption of a logic gate

$P = P_{static} + P_{dynamic}$

P_{static} : Small transistors leak current even when "blocked"

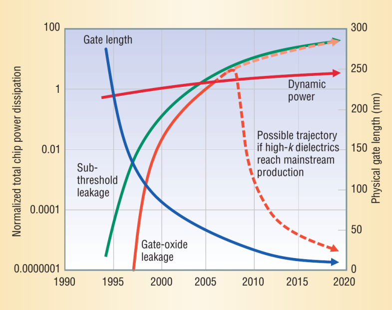
$P_{static} = V * I_{leak}$

I_{leak} depends on the voltage and the etching of the transistors.

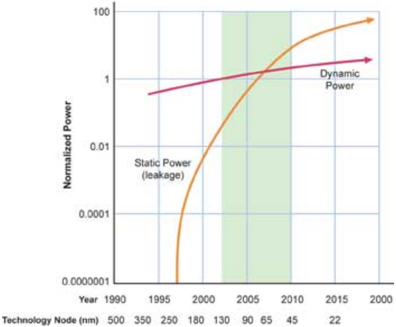
$P_{dynamic}$: Fighting the gate capacitance when switching

- $P_{dynamic} = CfV^2$;
- C: Capacitance of the gate (fixed);
- f: Frequency at which the gate is switched;
- V: Voltage at which the gate is powered.

The dynamic and static power cost



(a) Total chip dynamic and static power dissipation trends based on the International Technology Roadmap for Semiconductors (2003).



(b) Source: http://chipdesignmag.com/display.php?articleId=3310 (2009)

Usual ways of saving power

Usual ways of saving power

- Clock gating: Cuts the dynamic-power cost;
- Power gating: Cuts all the power cost;
- Reclocking: Adjusts the clock frequency and voltage.

Clock gating: Stopping the clock of un-used gates

- Update rate: Every clock cycle;
- Effectiveness: Cuts the dynamic-power cost entirely;
- Drawbacks: Increase of the complexity of the clock tree;
- Executed by: Hardware.

Usual ways of saving power

Power gating: Shutting down the power of un-used gates

- Update rate: Around a microsecond;
- Effectiveness: Cuts the power cost entirely;
- Drawbacks: May need to save the context before shutdown;
- Executed by: Hardware and/or software.

Reclocking: Dynamic Voltage/Frequency Scaling (DVFS)

- Update rate: Around a millisecond;
- Effectiveness: Impacts the static- and dynamic-power cost;
- Drawbacks: Affects performance;
- Executed by: Software.

Optimal DVFS policy

Optimal DVFS policy to stay in the power budget

- Find the bottleneck using performance counters;
- Lower the clocks of all the other clock domains;
- Lower the voltage of the power domains based on clocks;
- Increase the clock of the bottleneck clock domain;
- Repeat and learn about application patterns.

Constraints

- finding the bottleneck fast-enough;
- predicting the needed-voltage based on clocks' frequencies;
- calculating the memory timmings on-the-fly;
- supporting any combinaison of clocks.

A simple clock domain's clock tree

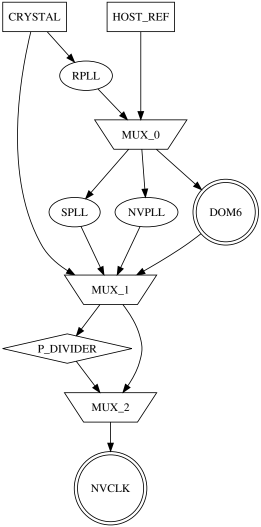


Figure : Clock tree for the core clock domain on nv84

Usual ways of saving power

Places to apply the proposed solutions

- card-level power gating (optimus);
- internal engines;
- VGA DACs;
- PCIe port (ASPM);
- anything using a clock and being part of a power domain.

PCIe ASPM impact

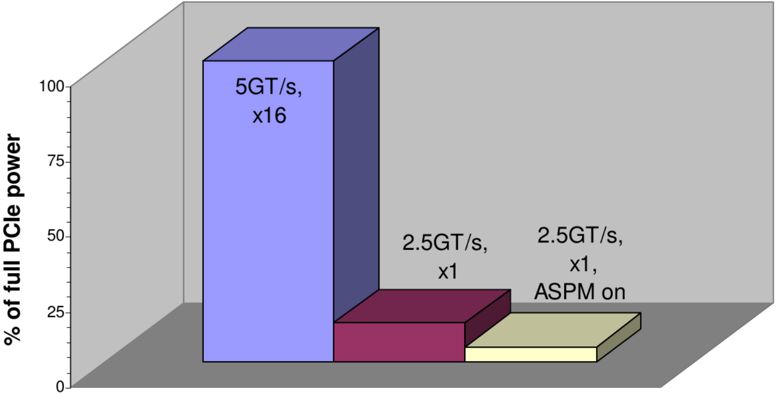


Figure : Maximum power consumption of the PCIe port at various link configurations.

Summary

PCOUNTER – Overview

Performance counters

- are blocks in modern processors that monitor their activity;
- count hardware events such as cache hit/misses;
- are tied to a clock domain;
- provide load information needed for DVFS's decision making.

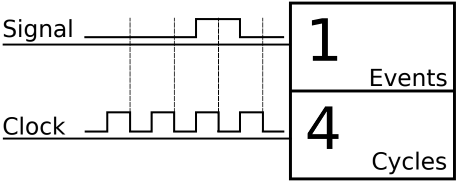


Figure : Example of a simple performance counter

PCOUNTER – Overview of a domain

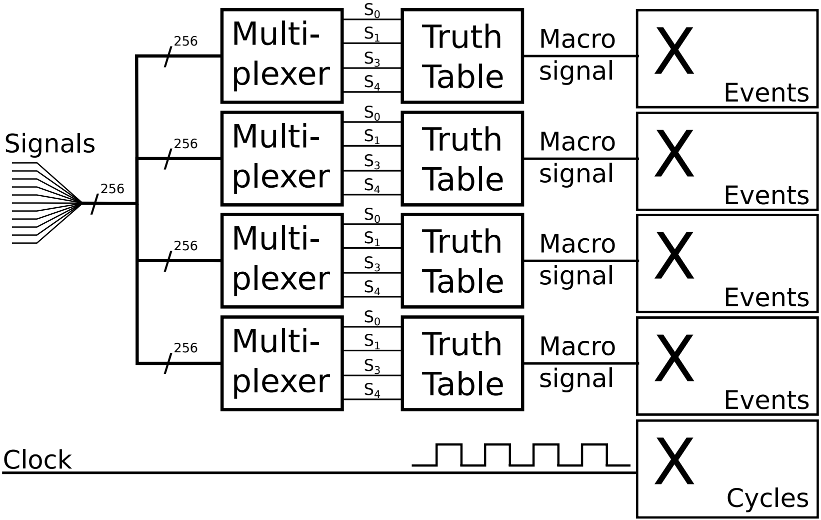


Figure : Schematic view of a domain from PCOUNTER

PCOUNTER – Other counters?

MP counters

- per-channel/process counters in PGRAPH;
- same logic as PCOUNTER;
- require running an opencl kernel to read them;
- share some in-engine multiplexers with PCOUNTER.

PDAEMON

- 4 global counters;
- very simplified logic;
- usually about the business of the other engines.

Counters – Which signals are known?

PCOUNTER signals

- very chipset-dependent;
- about 150 signals reverse engineered on nv50;
- thanks to Marcin (mwk) and Samuel Pitoiset (GSoC 2013).

MP counters signals

- all GPGPU signals exported by cupti on Fermi+ reversed;
- thanks to Christoph Bumiller (calim) and Samuel Pitoiset.

PDAEMON's signals

- 5 signals known;
- thanks to Marcin Kościelnicki (mwk).

Summary

PTHERM – Thermal management

PTHERM's thermal management

- sends IRQs to the host when reaching temperature thresholds;
- can cut the power of the card through a GPIO;
- can force the fan to the maximum speed;
- can lower the frequency of the main engine of the GPU (through FSRM).

PTHERM – Frequency-Switching Ratio Modulation

Frequency-Switching Ratio Modulation (FSRM)

- is used to lower the frequency of the main engine of the GPU;
- is useful to lower the temperature or the power consumption;
- is triggered automatically when reaching thresholds.

How can the FSRM lower power consumption?

- A divided clock is generated from the main engine's clock;
- The clock must be divided by a power-of-two (2 to 16);
- It can generate any clock frequency between these two clocks;
- With a lower clock, an engine consumes less power.

PTHERM – Frequency-Switching Ratio Modulation

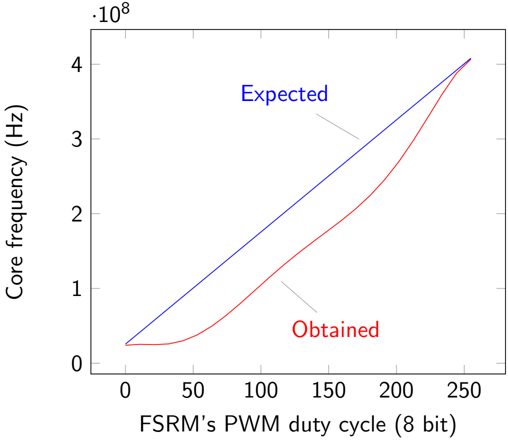


Figure : Frequency of the core clock (original @ 408MHz) when using a 16-divider and varying the FSRM

PTHERM – Power estimation

Calculating the power consumption

PTHERM estimates power consumption by:

- reading every block's activity (in use or not);
- summing the weighted activity blocks signals;
- applying a low pass filter.

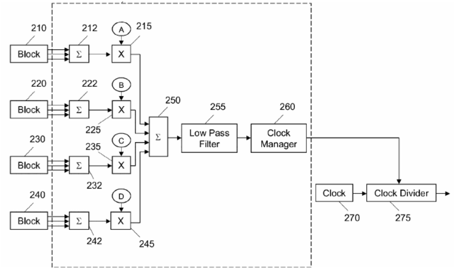


Figure : Extract of NVIDIA's patent on power estimation (US8060765)

PTHERM – Power limitation

PTHERM's power limitation can

- read the power consumption by counting the active blocks;
- update the FSRM ratio to stay in the power budget;
- use two hysteresis windows for altering the FSRM ratio;
- do all that automatically.

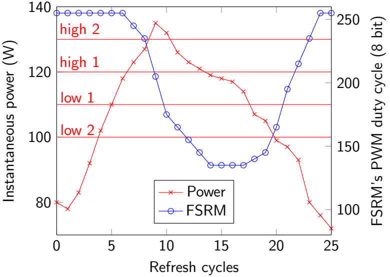


Figure : Example of the power limiter in the dual window mode

Power limitation – Actual implementation of NVIDIA

Power limitation – Actual implementation

- NVIDIA doesn't use PTHERM to implement power limitation;
- It may read power consumption from the voltage controller;
- and downclock the card when exceeding the budget.

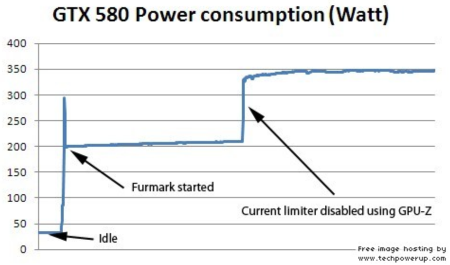


Figure : Effect of disabling the power limiter on the Geforce GTX 580. Copyrights to W1zzard from techpowerup.com.

Summary

PDAEMON – An embedded RTOS in your GPU

PDAEMON

- is an RTOS embedded in every new NVIDIA GPU (Fermi+);
- clocked at 200MHz and is programmed in the FμC ISA;
- has access to all the registers of the card;
- can catch all the interrupts from the GPU to the Host;
- features internal performance counters.

NVIDIA's usage of PDAEMON

- Fan management;
- Hardware scheduling (for memory reclocking);
- Power gating and power budget enforcement;
- Performance and system monitoring.

Summary

Conclusion

The GPU as an autonomic system

The GPU can:

- self-configure: thanks to PDAEMON that can act as a driver;
- self-optimise: using the performance counters;
- self-heal: recovering from over-temperature/current;
- self-protect: GPU users are isolated in separate VM.

Future works

- Implement stable reclocking across all GPUs;
- Write a test-bed for DVFS algorithms implementations;
- Document clock- and power-gating details;
- Reverse engineer more performance-counter signals.

Questions & Discussions

Questions & Discussions