Power Management of Datacenter Workloads Using Per-Core Power Gating

Jacob Leverich[1,2], Matteo Monchiero[1], Vanish Talwar[1], Partha Ranganathan[1], and Christos Kozyrakis[2]
[1]Hewlett-Packard Labs, [2]Stanford University

Abstract—While modern processors offer a wide spectrum of software-controlled power modes, most datacenters only rely on Dynamic Voltage and Frequency Scaling (DVFS, a.k.a. P-states) to achieve energy efficiency. This paper argues that, in the case of datacenter workloads, DVFS is not the only option for processor power management. We make the case for per-core power gating (PCPG) as an additional power management knob for multi-core processors. PCPG is the ability to cut the voltage supply to selected cores, thus reducing to almost zero the leakage power for the gated cores. Using a testbed based on a commercial 4-core chip and a set of real-world application traces from enterprise environments, we have evaluated the potential of PCPG. We show that PCPG can significantly reduce a processor's energy consumption (up to 40%) without significant performance overheads. When compared to DVFS, PCPG is highly effective saving up to 30% more energy than DVFS. When DVFS and PCPG operate together they can save up to almost 60%.

1 Introduction

Dynamic Voltage and Frequency Scaling (DVFS) is one of the most successful power management mechanisms provided by modern processors. Nevertheless, the efficacy of DVFS is limited by its dynamic range, ultimately fixed by the minimum voltage necessary to operate the transistors and the maximum voltage that can be thermally tolerated.

Voltage scaling is also less applicable to multi-core environments running heterogeneous workloads. Since there is typically a common voltage plane shared across all cores, a lower supply voltage cannot be employed unless all cores are simultaneously ready to use it. Alternatively, multiple voltage planes can be implemented at additional cost and design complexity. Thus, any heterogeneity in the characteristics of disparate workloads consolidated onto a multi-core chip forces a compromise in either performance, power consumption, or cost.

Additionally, we find that many deployments of multi-core chips exhibit only moderate utilization. For example, Barroso and Holzle [2] observe that processors in data centers operate mostly within a utilization range of 10% to 50%.

These issues motivate research on power management techniques that enable real differentiation in the power consumption of individual cores and very low-power modes. Power gating is a technique that allows one to shut off—i.e. *gate*—the power supply of a logic block by inserting a gate (or sleep transistor) in series with the power supply. Gating the power supply results in virtually no power consumption in the gated block. This technique can thus radically reduce a core's power consumption, providing a very effective per-core deep sleep state. In this paper, we specifically make the case for *per-core power gating (PCPG)*.

Per-core power gating complements existing voltage scaling techniques by providing an effective mechanism to reduce leakage power in a multi-core system when it operates at moderate utilization or when the aggregate workload exhibits high variability in resource usage.

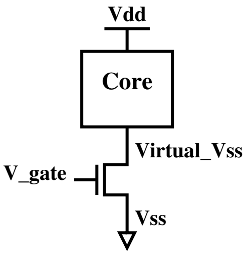
This work makes the following contributions. We present a per-core power gating architecture for multi-core processors that allows software to turn on and off individual cores using PCPG, as utilization and quality-of-service requirements vary. To quantify the potential of PCPG, we used a set of datacenter application traces with varying processor utilization. Datacenter applications exhibit load variability on a different time-scale than programs from conventional benchmark suites (e.g. SPEC CPU, MiBench) and thus present new opportunities for PCPG. Using a testbed based on a commercial quad-core chip, we demonstrate that PCPG leads to significant energy reduction (up to 40%) and power savings (up to 20%) without significant impact on performance. Moreover, we show that combining PCPG and DVFS leads to additional advantages (up to 60% energy savings and 25% power reduction) by providing a holistic approach to address both dynamic and leakage power in multi-core chips.

2 Architecture

PCPG has been widely adopted for embedded processor and Systems on Chip (see for example Nomura *et al.* 8-core Media Processor [10]), but for general purpose processors DVFS has been generally favored, likely because of the difficulties of implementing a power delivery network that enable power gating an entire core. High-end processors feature current densities that are an order of magnitude higher than embedded processors.

Manuscript submitted: 07-Jul-2009. Manuscript accepted: 06-Aug-2009. Final manuscript received: 13-Aug-2009.

Posted to the IEEE & CSDL on 8/19/2009
DOI 10.1109/L-CA.2009.46 1556-6056/09/$25.00 © 2009 Published by the IEEE Computer Society



Core and Package	
Technology node	65nm
Area of the core	25 mm^2
I_{max}	33 A
V_{dd}	1.06 – 1.35V
I_{leak}	4.2A (1.06V), 5.3A (1.35V)
T	333K
C_{core}	50nF
C_{dec_int}	100nF
C_{dec_ext}	500nF
L_{ext}	150pH
Power gate	
R_{on}	1.8m Ω (60mV)
T_{on}	100ns
T_{off}	190ns
E_{swon}/E_{swoff}	160nJ
P_{max}	59mW
P_{leak}	1mW

Fig. 1: Schematic of a power gate that employs an N-MOSFET as a sleep transistor, and circuit-level details for each core and power gate.

The Nehalem processor [1] implements its version of PCPG, using process-optimized sleep transistors to gate each core. Such devices have very low resistance when closed and individually isolate almost completely each core when open. A key feature of Nehalem is an integrated Power Controller Unit (PCU) that is able to turn off cores when they are idle according to some proprietary algorithm. In this work, we expose PCPG explicitly to a software-based controller. The goal is to maximize the duration of time that cores are gated, rather than to simply react to idle cores.

Circuits — We use Spice models to characterize the process of gating a core for performance and power. Figure 1 illustrates the basic circuit we assume, consisting of a NMOS transistor to gate the V_{ss}. We conservatively estimated the model parameters from published data [4], [6], [11] and measurements of an AMD 4-core Phenom X4 9850, which is the processor used in our testbed. We use NMOS devices because they are smaller than PMOS for the same channel resistance, but our results apply to PMOS devices as well. The companion table reports our estimates for the switching time and the energy consumption. Since we sized the sleep transistor quite aggressively to have very little resistance (1.8mΩ targeting 60mV of voltage drop at 33A), switching energy and time primarily depend on the charging/discharging process of the capacitance seen from the *Virtual_Vss* node, i.e. the core's capacitance and the internal decoupling capacitors (C_{dec_int}). The energy overhead of a transition is quite low, on the order of a hundred nanojoules. When a gate is off (power is gated), it introduces leakage power of roughly a milliwatt, while its active power when the gate is on is 6% of the core peak power consumption. Such low-resistance power gates do not cause significant thermal issues. We ensure that the power density developed within a power gate is always much lower than the power density of a core.

HW/SW Interface — We assume a physical implementation of PCPG very similar to Nehalem, but completely exposed to the software so that the control algorithm runs at the OS level. In our system, a transition to a gated state is thus performed by requesting a deep-sleep C-state for a specific core. Unlike Nehalem, where the PCU is responsible for deciding when to toggle a core's power supply, we chose to delegate this decision to a software entity.

The latency of power gating is determined by many factors other than the physical latency of driving the power gate. We found that the software and hardware procedures involved take much longer than the circuit time. We measured the time that it takes to enable or disable a core in Linux as roughly 100ms. This time accounts for all the steps needed to stop the core and prepare it to be shut off, including descheduling any running process, reprogramming the local APIC, servicing pending interrupts, and flushing the caches. Transitions across states for DVFS, for comparison, are much faster (microseconds), since the state of the processor does not need to be saved/restored.

Policies — PCPG performs best when controlled dynamically. This is particularly important for workloads with high variability or systems that spend a significant percentage of their time at low utilization [2]. There are several options in constructing a dynamic control scheme [8], [9]. Our evaluation employs a low/high-watermark algorithm. While this algorithm is rather simplistic, the goal is to evaluate the efficacy of PCPG rather than fully explore the control design-space.

Dynamic schemes for DVFS exploit memory bound workloads in order to reduce power without compromising performance. Using a lower frequency while the chip is waiting for memory to fulfill requests does not slow down applications. Similarly, PCPG can exploit I/O bound workloads, such as database servers, in order to reduce leakage power without compromising performance. Consolidating on to a single core multiple processes that mostly wait for I/O provides power saving without affecting system throughput. On the other hand, consolidating cache intensive workloads on a single core can lead to increased cache interference that increases the execution time of each application, even if the processor core is mostly idle.

3 Evaluation

We evaluated the performance and power impact of PCPG using a 2.5 GHz AMD Phenom X4 9850 system. This 65nm chip has 4 cores, each with a private 512KB L2 cache, a shared 2MB L3 cache, and an on-chip Northbridge. Our performance results are taken directly from this system, while our power results are a hybrid of real power measurements and an estimated reduction in leakage power for when we gate cores.

Load Generator — We developed a multi-threaded load generator to replay traces of CPU utilization measured from operational datacenters. The load generator takes the utilization samples and assigns work to several worker threads which alternate between executing dummy loops and sleeping. To promote fine-grained variability in the workload, the sampled utilization is used as a probability for whether or not to sleep at 10 millisecond time-steps so that the average utilization

Trace	Max Load	Min Load	Avg Load	Technique	Avg Power (W)	High Δ Power (W)	Low Δ Power (W)	Avg Δ Power (W)	Energy (J)	% Δ Energy	% delayed 1s	% delayed 2s	% delayed >2s	% 4 cores	% 3 cores	% 2 cores	% 1 core
SAP05	3.51	0.20	1.53	Base	59.66	-	-	-	7099	-	0.18	0.00	0.00	-	-	-	-
				DVFS	60.45	3.05	-1.50	0.79	7194	1.33	2.60	0.00	0.00	-	-	-	-
				PCPG	56.02	14.70	-16.68	-3.63	6667	-6.09	3.58	0.00	0.00	5.0	47.1	35.5	12.4
				Both	53.41	11.71	-21.60	-7.25	6356	-10.47	8.54	0.00	0.00	5.0	48.7	45.4	0.8
SAP11	2.71	0.55	1.26	Base	61.29	-	-	-	7294	-	0.07	0.00	0.00	-	-	-	-
				DVFS	61.83	4.37	-2.92	0.54	7358	0.87	2.08	0.00	0.00	-	-	-	-
				PCPG	49.13	-3.28	-20.28	-12.17	5846	-19.85	2.94	1.54	2.98	1.7	15.7	71.9	10.7
				Both	46.62	-4.90	-25.76	-14.89	5548	-23.94	4.60	2.28	1.71	1.7	15.0	80.0	3.3
SAP12	3.45	0.89	2.22	Base	82.84	-	-	-	9858	-	0.05	0.00	0.00	-	-	-	-
				DVFS	91.66	23.38	-7.10	8.82	10907	10.64	4.63	0.00	0.00	-	-	-	-
				PCPG	82.85	12.63	-11.27	0.01	9859	0.01	4.72	1.08	1.72	21.5	66.1	11.6	0.8
				Both	79.99	6.10	-12.30	-3.38	9519	-3.44	10.20	1.08	0.37	24.2	65.8	9.2	0.8
PHARMA04	2.84	0.00	0.42	Base	47.36	-	-	-	5636	-	0.22	0.00	0.00	-	-	-	-
				DVFS	37.42	-0.53	-17.97	-9.95	4452	-21.00	2.04	0.00	0.00	-	-	-	-
				PCPG	29.13	-9.20	-26.92	-18.23	3466	-38.50	9.93	3.39	0.00	0.0	9.1	7.4	83.5
				Both	25.49	-2.09	-31.20	-21.87	3034	-46.18	7.26	0.00	0.00	0.8	9.9	10.7	78.5
HCOM10	1.77	0.10	0.51	Base	45.30	-	-	-	5391	-	0.02	0.00	0.00	-	-	-	-
				DVFS	31.97	-0.07	-21.37	-13.33	3804	-29.43	0.92	0.00	0.00	-	-	-	-
				PCPG	25.90	-9.16	-27.77	-19.41	3082	-42.84	5.43	0.00	0.00	0.0	0.0	19.8	80.2
				Both	18.71	-13.80	-34.72	-26.81	2227	-58.70	5.69	0.00	0.00	0.0	0.8	25.0	74.2
HCOM19	2.79	0.00	1.45	Base	68.98	-	-	-	8208	-	0.06	0.00	0.00	-	-	-	-
				DVFS	58.40	-0.06	-23.44	-10.58	6949	-15.34	3.46	0.00	0.00	-	-	-	-
				PCPG	50.87	-4.02	-30.85	-18.10	6054	-26.25	5.66	2.00	1.79	5.0	43.0	2.5	49.6
				Both	49.61	0.82	-36.06	-20.04	5904	-28.07	8.26	2.13	1.94	15.0	32.5	5.0	47.5
ECOM3	1.78	0.01	0.84	Base	47.84	-	-	-	5693	-	0.03	0.00	0.00	-	-	-	-
				DVFS	48.30	18.17	-12.83	0.46	5748	0.97	1.27	0.00	0.00	-	-	-	-
				PCPG	32.09	-8.51	-25.99	-15.75	3818	-32.93	2.88	0.00	0.00	0.0	0.8	68.6	30.6
				Both	29.84	-10.13	-25.95	-18.00	3551	-37.62	3.05	0.00	0.00	0.0	0.8	78.5	20.7
DESKTOP	0.87	0.00	0.14	Base	40.62	-	-	-	4833	-	0.00	0.00	0.00	-	-	-	-
				DVFS	27.56	-4.49	-16.21	-13.05	3280	-32.14	0.19	0.00	0.00	-	-	-	-
				PCPG	19.12	-19.81	-23.13	-21.49	2276	-52.92	0.69	0.00	0.00	0.0	0.0	0.8	99.2
				Both	14.33	-21.72	-28.79	-26.29	1705	-64.73	1.20	0.00	0.00	0.0	0.0	1.7	98.3

TABLE 1: Results for a variety of utilization traces, including commercial application servers (SAP05 SAP11, SAP12, PHARMA04), web servers (HCOM10, HCOM19, ECOM3), and a desktop machine (DESKTOP). A few positive (lighter green) and negative (darker red) results are highlighted. The presented metrics can be divided into 4 categories: power, energy, delay, and PCPG state. For power, we present average power consumed during the trace (*Avg Power*), the largest positive power difference w.r.t the baseline configuration (90th percentile) (*High Δ Power*), the largest negative power difference w.r.t the baseline configuration (90th percentile) (*Low Δ Power*), and average power difference (*Avg Δ Power*). For energy, we present the absolute consumption in Joules (*Energy*) (which, since each trace runs for the same amount of time, is proportional to average power consumption), and the percent change in energy consumption (% Δ *Energy*). For delay, we show the percent of total work delayed for one second, for two seconds, and for more than two seconds. For the PCPG state, we present the percent of time spent with 4, 3, 2, or 1 core active.

approaches the sampled utilization over a one second period. The trace generator is *work conserving*. Every run of a trace will execute the same number of instructions, regardless of its variance in performance. A trace sample of x% utilization is interpreted as a finite quantity of work (i.e. # of instructions) rather than simply a utilization to match at a given point in time. If the load generator is unable to issue all of its work during a one-second time window, it records the left-over work as a deficit, which it will attempt to pay-down immediately in the next time window. Each time the generator retires a deficit, it records in a histogram the amount of time it took to retire that work. If a deficit remains following the end of a trace run, the load generator continues to run until the entire deficit is cleared. We validated our trace methodology by recording utilization traces of real workloads, replaying them, and then comparing the power consumption of the real workload to the power consumption of the load generator. We find mean error of -7 Watts and mean percent error of -3.4% for a run of SPECpower_ssj2008.

Traces — Our traces are comprised of samples of CPU utilization taken once per second on a variety of commercial settings, including highly utilized SAP servers, web servers with varied utilization, and enterprise desktop machines. We selected 8 traces, each 120 seconds in length, to use as loads to place on our test system through the load generator.

Performance Model — To study the performance impact of gated cores, we use Linux 2.6.24's built-in CPU "hotplug" support and measure actual performance of the system. Originally intended for systems with hardware support to install and remove CPU modules without interruption, the hotplug support mimics precisely the behavior necessary to model core gating.

Hybrid Power Model — Our power model is a hybrid of real power measurements, a model of the energy consumption of our power gates, and estimates of leakage power for the Phenom X4 9850 (including North Bridge and L3 cache). All together, we calculate the per-core leakage in the 2.5 GHz P-state as 7.13 Watts, and the leakage in the 1.25 GHz P-state as 4.45 Watts. Power consumption results for PCPG configurations are post-processed to account for the amount of leakage eliminated when cores are gated.

Dynamic Power Management Daemon — We implement a dynamic PCPG manager as a userspace daemon. The daemon polls processor utilization (`/proc/stat`) at the rate of 20 Hz by summing the time spent not idling or waiting for I/O. This utilization factor is averaged over a one second sliding window. The decision to enable or disable a core is based on a simple high/low watermark control algorithm. For comparisons with DVFS, we use Linux 2.6.24's "ondemand" DVFS governor.

Results — Table 1 shows the evaluation results for a variety of traces, including commercial application servers, web servers, and a desktop machine. See its caption for a description of each column. As none of these traces exhibited 100% load for their entire duration, opportunities were generally found to gate cores and reduce power consumption. These energy consumption savings range from 3.4% for SAP12, a trace with substantial utilization, to 64.73% for DESKTOP, a trace taken from a mostly idle desktop system. Overall, the energy saving is higher than 20% for 6 traces out of 8. While nominal power consumption of the PCPG configurations is well below the baseline system, there are some anomalous measurements. For example, the High Δ Power (the highest amount by which a configuration overshoots the baseline system at any correlated point during the trace) for SAP05 and SAP12 show that the PCPG configuration at some point consumes 14.7 Watts and 12.63 Watts, respectively, over the baseline. These readings correspond to periods immediately following low throughput, where the load generator is attempting to pay-down an accumulated work deficit.

In nearly all cases, the PCPG configurations come with some delay in work execution, ranging from 0.69% work delayed one second or more for the DESKTOP trace to 13.32% for the PHARMA04 trace. The amount of this delay correlates more strongly with the maximum load during a trace rather than the average load. For example, PHARMA04 exhibits significant work delay, yet the average load during that trace is only 0.42. Note that these work delays do not correspond to discrete "slow-downs" or increases in execution time. Indeed, all of the traces presented here had no work deficit upon completion. Rather, these work delays can best be interpreted as quality-of-service (QoS) metrics, and can mainly be used for comparison between different configurations. Nevertheless, they serve as a suitable lower-bound on performance.

4 Related Work

Significant effort has focused on algorithms for managing sleep states and DVFS [3], [5], [7]. Kim et al. studied per-core DVFS using on-chip switching regulators [6]; such devices incur higher overheads than simple power gates. More importantly, all of these studies target high-activity workloads (SPEC and HPC benchmarks) optimizing towards a different design point with respect to us. We evaluate PCPG in the context of variable and low-utilization workloads. As far as we know, our work is the first one at evaluating per-core deep sleep states and showing its advantages for datacenter applications.

5 Conclusions

This paper has made the case for per-core power gating (PCPG) as an effective technique to reduce leakage power in multi-core systems. PCPG exploits the fact that most server and client systems spend significant periods of time under moderate utilization, during which load can be aggregated on fewer cores. PCPG is orthogonal to existing power management techniques, such as DVFS, which targets dynamic power reduction at high utilization, and deep sleep states, which target leakage power consumption when a chip is idle. Our evaluation demonstrates that under dynamic control PCPG can lead to significant savings in static power and energy consumption, without significant performance reduction, for a wide range of workloads with periods of moderate utilization. We have also shown that PCPG is complementary to chip-wide DVFS techniques. Our results establish that PCPG is a practical and effective technique for static power management and that there is exciting future work on PCPG control algorithms, on techniques for combining PCPG with existing voltage scaling, and on evaluation of detailed implementations.

References

[1] Intel microarchitecture (Nehalem). Available at www.intel.com/technology/architecture-silicon/next-gen.

[2] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.

[3] J. Donald and M. Martonosi. Techniques for multicore thermal management: Classification and new exploration. In *Proc. of the 33rd annual Intl. symposium on Computer Architecture*, 2006.

[4] M. S. Gupta, J. L. Oatley, R. Joseph, G.-Y. Wei, and D. M. Brooks. Understanding voltage variations in chip multiprocessors using a distributed power-delivery network. In *DATE '07: Proc. of the conference on Design, automation and test in Europe*, 2007.

[5] C. Isci, A. Buyuktosunoglu, C.-Y. Cher, P. Bose, and M. Martonosi. An analysis of efficient multi-core global power management policies: Maximizing performance for a given power budget. In *Proc. of the 39th Annual Intl. Symposium on Microarchitecture*, 2006.

[6] W. Kim, M. Gupta, G.-Y. Wei, and D. Brooks. System level analysis of fast, per-core DVFS using on-chip switching regulators. In *Proc. of the 14th Intl. Symposium on High-Performance Computer Architecture*, 2008.

[7] J. Li and J. Martinez. Dynamic power-performance adaptation of parallel computation on chip multiprocessors. In *Proc. of the 12th Intl. Symposium on High-Performance Computer Architecture*, 2006.

[8] Q. Wu et al. Dynamic-compiler-driven control for microprocessor energy and performance. *IEEE Micro*, 26(1):119–129, 2006.

[9] L. Ramos and R. Bianchini. C-Oracle: Predictive thermal management for data centers. In *HPCA'08: The 14th Intl. Symposium on High-Performance Computer Architecture*, 2008.

[10] S. Nomura et al. A 9.7mw AAC-decoding, 620mW H.264 720p 60fps decoding, 8-core media processor with embedded forward-body-biasing and power-gating circuit in 65nm CMOS technology. In *Proc. of the IEEE Intl. Solid-State Circuits Conference*, 2008.

[11] J. Stinson and S. Rusu. A 1.5GHz third generation Itanium processor. In *Proc. of the Intl. Solid-State Circuits Conference*, 2003.