International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:10, No:1, 2016

Interplay of Power Management at Core and Server Level

Jörg Lenhardt, Wolfram Schiffmann, Jörg Keller

Abstract—While the feature sizes of recent Complementary Metal Oxid Semiconductor (CMOS) devices decrease the influence of static power prevails their energy consumption. Thus, power savings that benefit from Dynamic Frequency and Voltage Scaling (DVFS) are diminishing and temporal shutdown of cores or other microchip components become more worthwhile.

A consequence of powering off unused parts of a chip is that the relative difference between idle and fully loaded power consumption is increased. That means, future chips and whole server systems gain more power saving potential through power-aware load balancing, whereas in former times this power saving approach had only limited effect, and thus, was not widely adopted. While powering off complete servers was used to save energy, it will be superfluous in many cases when cores can be powered down. An important advantage that comes with that is a largely reduced time to respond to increased computational demand.

We include the above developments in a server power model and quantify the advantage. Our conclusion is that strategies from datacenters when to power off server systems might be used in the future on core level, while load balancing mechanisms previously used at core level might be used in the future at server level.

Keywords—Power efficiency, static power consumption, dynamic power consumption, CMOS.

I. Introduction

POWER consumption of computers has been a focus of research for many years, ranging from lower voltage transistors over frequency scaling and power gating to algorithmic approaches. Consequently, different methods for the reduction of power consumption in computers have been applied by different communities at various levels, e.g. reducing power consumption in single cores up to reducing power consumption of a data center by migrating load and switching off unused servers.

We present a model for power consumption that reflects recent technological developments and allows the conclusion that at core level, switching off some cores (and migrating workload to remaining cores) gets more important, and hence strategies from data centers might be useful to be applied in single systems. On the other hand, at system level, it seems that switching off complete server systems (and thus long restart times) can often be avoided because of low idle power, and thus load distribution strategies from multicore systems at the time when dynamic power consumption was dominating might be useful to be applied in datacenters.

While in CMOS devices dynamic power only occurs when switching takes place, static power resulting from leakage currents contributes permanently to the power consumption. In the past, the proportion of static power was low and was widely neglected. With decreasing feature sizes static power increases exponentially [1]. Despite the ongoing decrease in feature sizes of modern semiconductor devices the threshold voltage cannot be lowered at former rate. Smaller transistor dimensions lead to proportionally increasing leakage current. As a result, the energy consumption keeps almost unchanged while the chip area gets smaller.

Two results regarding power saving potential considering rising static power proportion are important: (1) Powering off parts of a chip is crucial to save energy in future CMOS devices and (2) cores are suitable units for on-demand powering off and on. The idle power consumption of the whole computer system decreases when powering off more parts. Consequently, the gap between idle and full power consumption of servers rises. Formerly, energy efficient load balancing had only marginal potential to reduce power consumption but with increasing full-to-idle power-ratio, it becomes more important.

We distinguish three levels in regard to power and energy: (1) A **single core** can be switched on or off. Above that, it can run at different frequency and voltage levels to reduce power. (2) In a **multicore processor** with n cores usually 1 to n cores are running, 0 if the system is off. (3) The whole **system** can be switched on or off and the system can be loaded between idle (booted but no actual work to do) and 100%. Switching cores on or off is orders of magnitude faster than shutting down or booting a whole system [2]. Due to different effects the full-to-idle power-ratio decreases for single cores. But in multicore environments the ratio increases because at low utilization the load is distributed among fewer cores while the others can be switched off. In this context, we examine to move from (a) *system on/off and DVFS for cores* to (b) *system on and cores on/off with or without DVFS*. The strategies are somewhat reversed. We examine several models for describing the power consumption using various strategies. This development has some advantageous effects regarding server farms in data centers. Usually, operators are not willing to shutdown servers, e.g. because of the risk that the system does not reboot. Above that, the time to boot a system is magnitudes higher (several minutes) than to power on cores. The response time to random changing conditions is much worse. Moving from powering off and on systems to load balancing to reduce power consumption is a more convenient solution.

In this work, static power consumption refers to the static part of the power consumption of a single CMOS device,

Jörg Lenhardt, Wolfram Schiffmann, and Jörg Keller are with the Faculty of Mathematics and Computer Science, Computer Architecture/Parallelism and VLSI, University of Hagen, P.O. Box 940, 58084 Hagen, Germany (e-mail: Joerg.Lenhardt, Wolfram.Schiffmann, Joerg.Keller@FernUni-Hagen.de).

e.g. a processor or multi-core processor. Dynamic power consumption represents the variable portion of the power consumption of a CMOS device when increasing or decreasing the usage. Note that idle power consumption represents the power consumption of an unloaded server system. Full power consumption represents the case that the server is fully loaded.

The rest of this paper is organized as follows: Section II presents preliminaries and related work. DVFS and power off in view of rising static power is discussed in Section III. The consequences of this development in regard to semiconductor devices (small) on the one hand and server farm (big) on the other is discussed in Section IV. We conclude and give ideas for future work in Section V.

II. Preliminaries

A. Shares of Power Consumption in Multi and Many Core Systems

In former years, the idle power consumption was the dominating part of the total power consumption. If we look at systems that became available up to late 2009 the idle portion was at 50% to 60% of the total power, leading to a full-to-idle power-ratio between 1.5 and about 2.0. New hardware generations leading to increased full power consumption (of course also to enhanced performance) and to a slightly decreasing idle power consumption. In Fig. 1, we compute and depict the full-to-idle power-ratio for a huge subset of SPECpower data [3] from 2007 to early 2014. Each dot represents the ratio for one system. The line is the linear fit of these data. The full-to-idle power-ratio increases steadily from 2009 to 2014, leading to values up to 4.2 in 2009 and 5 to 6 in 2012-2014. The ratio is increasing at about 0.5 per year. This means, the idle power consumption nowadays only claims 20% to 25% of the total power. This is due to improved power saving mechanisms in modern chips that reduces the idle power despite increasing static power consumption of the devices. So in a large scale (servers and server farms) load balancing in regard to energy efficiency becomes more and more interesting.

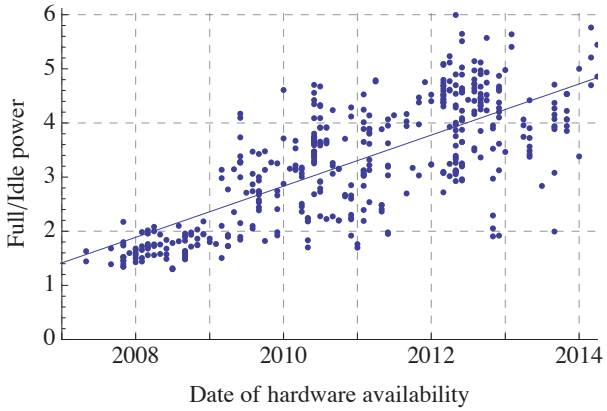

Fig. 1 Trend of relation between full and idle power

Besides processors and cores the rest of the system have a part in the power consumption of a server system. To get an idea about the shares (especially the share of the cores of a multicore CPU) we measured the power consumption of various standard multi-core PCs by using Linux sysbench benchmark for CPU performance. In several phases we measured the idle power consumption, and the power consumption using one core, two cores, and so on until all cores ran the benchmark. Each phase lasted ten minutes and between each phase was period of one minute idle. The whole process took about 55 minutes. Per second four power measurements were taken. A detailed result for one system (Intel i5 2500, 3.3 GHz) is shown in Fig. 2. On the X-axis is the time in minutes, on the Y-axis the power consumption in watts.

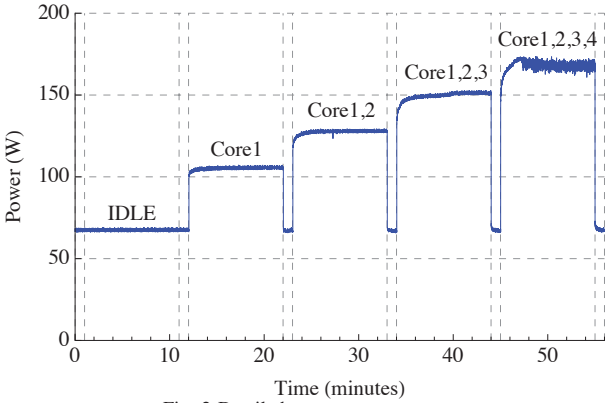

Fig. 2 Detailed power measurement

The power consumption was quite stable in all cases with heavier fluctuations when all four cores were used. Table I gives an overview over minimal (min), maximal (max), average (avg) power, and the increase (inc) to the average value of the previous phase. The power consumption increases by 36.8 watts from idle to using one core, then for each additional core the power increases by about 20 watts. It can be concluded that each core has a power consumption of 20 watts, while 15 watts is the basic power when using at least one core. The share of the full power consumption is about 12% for each core, plus 9% basic core power. 60% of the full power is consumed by the CPU cores, 40% are the remaining system and other CPU functions not directly related to core power consumption.

TABLE I
Minimal, Maximal, Average, and Additional Power (W)

	min	max	avg	inc
IDLE	66.4	69.2	67.0	-
Core 1	99.4	106.4	103.8	36.8
Cores 1,2	115.5	128.8	125.2	21.4
Cores 1,2,3	130.5	150.5	145.5	20.3
Cores 1,2,3,4	143.0	173.1	165.8	20.3

B. Workload Models

We assume a divisible workload that is expressed as a percentage of a multi-core CPU's maximum processing capability, and that can be re-distributed over cores. While this seems rather abstract, there are several examples from practice that match this model quite well. First, we consider web-servers, which are multi-threaded applications where each

request is served by a separate thread. The number of threads at a certain load level typically is much larger than the number of cores needed to process that load level. Hence, the load can be considered fine-grained enough to be modeled as a divisible load. A similar situation occurs with database applications where different queries are processed in different threads.

Although the heterogeneity of the threads' processing requirements can be larger than in a web-server, and although the threads might interact on the data stored in the database, the majority of queries typically contributes small loads and are unrelated, so that the model of a divisible load gives an approximation. Finally, we consider server systems that provide virtualization services by running virtual machines that look like different (physical) computers to customers. Often a server hosts more than one hundred virtual machines, at least many more than it has cores. Thus, each load entity, i.e. virtual machine, only contributes a small share to the total load, so that the workload is again fine-grained enough to be modeled as a divisible load.

The difference to the former applications is the duration of the load entities: while web requests and database queries are processed in fractions of a second, and there is a continuous stream of new requests, virtual machines are mostly run for days or even weeks. For short lived requests, load balancing occurs when the requests arrive, by assigning them to a core. For long lived virtual machines, load balancing occurs by migrating the virtual machine to a different core, which contributes an overhead which however is not very frequent and thus can be neglected: the duration of the virtual machines has the consequence that they terminate only seldom and thus load re-balancing is only necessary once in a while.

C. Related Work

In data centers, the servers' utilization typically lies just between 10 and 50%. Thus, if the servers provide the maximum performance while running at full power a lot of energy is wasted. The general objective is to adjust the power consumption proportional to the requested performance [4].

An overview over dynamic and static power consumption in CMOS devices is given in [5]. The most popular method to reduce power consumption consists of slowdown the chip's clock by means of DVFS. Unfortunately, the power savings of this technique are limited by the difference between maximum and static power of the chip. Due to the increasing leakage currents of chip level the efficiency of DVFS will be shortened unless the amount of static power cannot be reduced in the future [6].

In order to achieve energy-proportional computing a shutdown of components like functional units, cores or even complete servers would be inevitable and several other authors report about significant energy savings by means of these techniques [7]–[9]. Alternatively, reconfigurable application specific devices on the processor chip [10] and toggling between heterogeneous computing systems with different performance characteristics have been proposed [11].

Several authors examine the consequences of the breakdown of Dennardian scaling [12]. The portion of a chip which can be switched at full frequency is dropping exponentially with each process generation due to power constraints. Large fractions have to be dark or dim [1], [13]–[17]. Thus, both shutdown and slowdown techniques will be needed in the near future.

All the shutdown techniques suffer from the overhead to reactive the components when the performance requirements continue to increase. This results in delays that will depend on the complexity of the switched off components [18]. Moreover, the overhead can even increase the energy consumption. In [19] a strategy for avoiding those *negative* energy savings is presented.

In this paper we investigate the interrelationship between core level and server level power management in the face of recent developments in CMOS devices such as growing importance of static power consumption at core level and diminishing importance of idle power at system level.

III. DVFS and Power-Down

In this section we present several variants of two strategies to reduce power consumption of homogeneous many core systems. We have a look at DVFS strategies as well as the possibility to power off certain parts (in our case cores) of a CMOS device to reduce power and energy consumption. Our focus lies on the influence of the increasing static power in modern semiconductor devices.

In the following, *c* represents the number of cores, *l* the load of each core where $0 \leq l \leq 1$, s the static part of power consumption where $0 \leq s \leq 1$, *f* frequency of chip/core where $0 \leq f \leq 1$. *F* is a set of frequencies when using discrete frequency levels. We denote scaling, continuous, discrete, and power off by indices *sc*, *cont*, *disc*, and *po*, respectively.

A. Continuous DVFS and Power Off

The power consumption using continuous DVFS can be described with (1). The power consumption of a single core is the sum of the static part s plus the dynamic portion; the dynamic portion is the load divided by number of cores cubed times $(1 - s)$. We used a cubic exponent because changing frequency has linear and voltage scaling quadratic influence on the power consumption.

$$sc_{cont}(c, s, l) = c \cdot \left(s + (1 - s) \cdot \left(\frac{l}{c} \right)^3 \right) \tag{1}$$

The power consumption using the power off strategy can be described with (2). Because the maximum load is normalized to 1 for each core, the minimum number of cores that have to be used is the current load rounded to the next integer. For simplicity we decided to use a cubic approximation to describe the power consumption when using DVFS. We are aware that this approximation is not longer accurate because of the fail of Dennards Scaling. Nevertheless, it is an accurate enough description for the global trend in respect to the development of the increasing range between idle and full power consumption of whole server systems.

$$po_{plain}(l) = \lceil l \rceil \tag{2}$$

In Fig. 3 the resulting power consumption of both strategies for a 16 core system are plotted in regard to the load. Static portions of 0.1, 0.3 and 0.5 are considered. Powering off is plotted in red, and DVFS in blue.

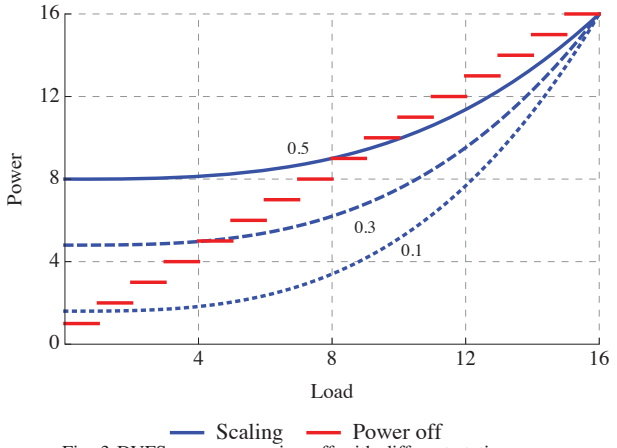

Fig. 3 DVFS versus powering off with different static power

The static part of power consumption has no influence when applying po_{plain}, so there is only one curve in the shape of a stair. In contrast, using DVFS static power influences the results. Having a low static fraction as found by older devices, DVFS leads to better results than power off except at very low utilization (see fine dashed blue line, $s = 0.1$). Increasing static fraction the power consumption increases as DVFS has only influence on the dynamic part. With $s = 0.3$ the results of power off are better till a load of 5 of 16 has been reached, at $s = 0.5$ powering off is better up to a load of 9 of 16. The differences at higher utilization decreases with increasing static fraction.

B. Discrete DVFS

In realistic clocked devices only a set of valid frequency levels are available. The device runs at one of these levels. Equation (3) models discrete DVFS. The only difference to the continuous version (1) is the dynamic part, where an auxiliary sub-function is used instead of the quotient of load and cores.

$$sc_{disc}(c, s, l) = c \cdot \left(s + (1 - s) \cdot mf^3(l, c)\right) \qquad (3)$$

The sub-function mf, see (4), calculates the minimum frequency at which the device has to be clocked to process the given load.

$$mf(l, c) = min\left\{f | (f \in F) \wedge (\frac{l}{f} \leq c)\right\} \qquad (4)$$

mf returns the minimum of a subset of valid frequencies F. The elements f of the subset have to be greater or equal the load l divided by the number of cores c to guarantee that the load is processed timely. In Fig. 4 the power consumption of a 16 core system using continuous (red) and discrete (blue) DVFS is shown for static portions s of 0.1 and 0.5. Valid frequencies F are 0.2, 0.4, 0.6, 0.8, 0.9, and 1.0.

Discrete DVFS produces a step function where the power consumption is generally higher than in continuous DVFS.

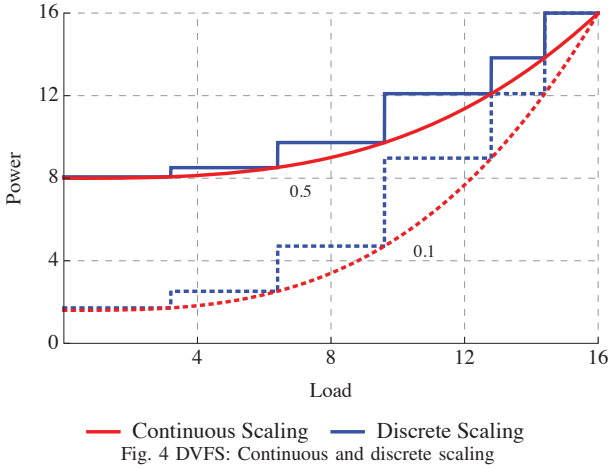

Fig. 4 DVFS: Continuous and discrete scaling

Only at points where l/c equals a valid frequency the power consumption of both variants are equal. As we consider a divisible load, our system would distribute this load as evenly as possible over all cores and thus all cores should be scaled by the system[1] to the same frequency, or onto two adjacent frequency levels as a consequence of the discretization.

C. DVFS in Combination with Power Off

To benefit from DVFS as well as from powering off cores a combination of both can be used. This can be modeled with (5).

$$po_{scale}(s, l) = \lceil l \rceil \cdot \left(s + (1 - s) \cdot mf^3(l, \lceil l \rceil)\right) \qquad (5)$$

The equation is similar to (3). The difference is that the number of cores is not a parameter any more. The minimum amount of cores is used to execute load l, which corresponds to the ceiling of l. The results in comparison to powering off cores without DVFS is displayed in Fig. 5 for a 16 core device. The static part s is at 0.3 in this example. Again the same frequency levels F as in the last subsection are used.

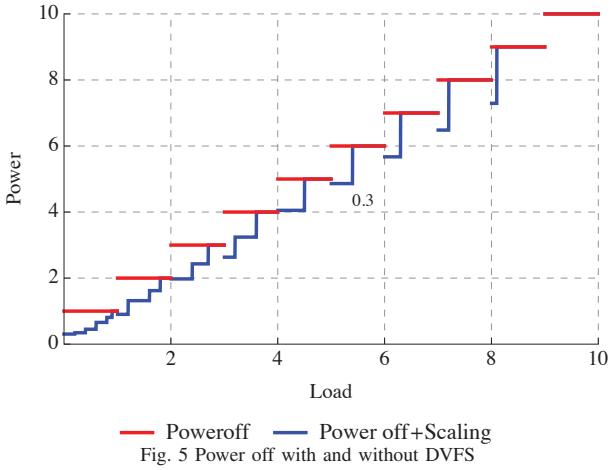

Fig. 5 Power off with and without DVFS

[1]If the frequency governor does not achieve this, the application or an adapted governor would have to enforce this.

In this case, the strategy leads to better results when l is lower 9, so in Fig. 5 the excerpt to load 10 is displayed, the rest above omitted. If the load l is low, more frequency levels can be used to reduce power consumption. Increasing the load less different frequency levels can be applied till all cores are running at $f = 1.0$. It is easy to understand when assuming a load of 9: Instead of using 10 cores and running at a frequency $f = 0.9$ the algorithm uses 9 cores.

In Fig. 6 the results of DVFS with and without powering off cores for a 16 core system are displayed. In this example, we have a static portion s of 0.5.

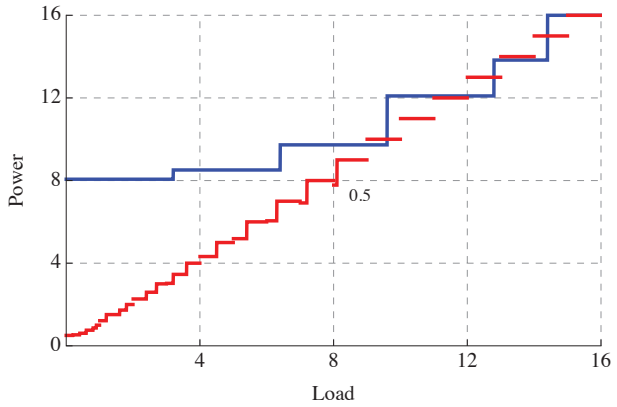

Fig. 6 DVFS with and without power off

Up to a load l of 9 the combination of powering off most of the cores and apply DVFS leads to less power consumption than DVFS alone. Above that, DVFS alone can achieve better results in some cases, e.g. from load 9 to about 9.5. This leads to the insight that in some cases it may be beneficial to use more cores than the minimum possible to achieve lower power consumption.

D. Decrease Power by Using More Cores

Although counter-intuitive, the previous subsection gave an example where it was advantageous to use more than the minimum number of cores. Equation 6 models this situation. sc_{disc} from (3) is used several times starting with load l rounded up to the next integer as input for number of cores. This is tried for all core counts up to c. The result with the minimum power consumption is used.

$$sc_{min}(c, s, l) = min\left\{sc_{disc}(m, s, l) | (m \in \mathbb{N}) \wedge (\lceil l \rceil \leq m \leq c)\right\} \quad (6)$$

The result for a 16 core system with a static part of $s = 0.5$ of the power consumption is shown in Fig. 7. DVFS only is shown in blue, DVFS with powering off cores in red and sc_{min} in dashed black.

For $0 \leq l \leq 4$, the results of sc_{min} are similar to powering off with scaling. After that, the results are better than or the same as DVFS (with or without power off). Especially for $8 \leq l \leq 12$, better results are achieved. Overall, sc_{min} is never worse than the other two strategies.

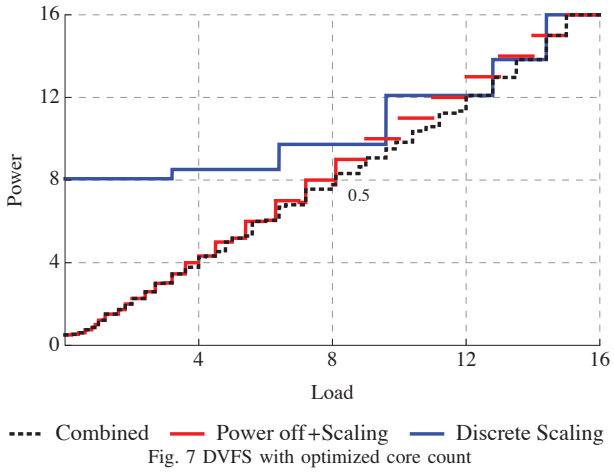

Fig. 7 DVFS with optimized core count

This model describes the theoretical optimum and it is pretty close to power proportionality. Unfortunately, in real systems there are usually components on chip level which lead to power consumption despite low loads. On the big scale, i.e. server systems, an idle power offset is present due to other components besides the main processor (e.g. main memory, hard drives etc.). Nevertheless, the main processor contributes significantly to the overall power consumption of a server system.

IV. Influence of Increasing Static Power

For a single core, the increasing fraction of static power, together with the restricted possibility for voltage scaling, leads to a diminishing influence of frequency scaling on power consumption. Hence, in case of multicores it might be advantageous to use fewer active cores at higher frequencies. In this respect, the operating system community could take a look at strategies used in datacenters at the level of complete server systems.

The good news for operating system researchers is that the capability to forecast workload changes can be restricted at core level, as the time to power up a core is much shorter (milliseconds) than the time to power up a complete system (multiple seconds). Thus, the algorithms to decide on core shutdown and power up can be simpler and thus more aggressive than at system level. For a complete system, the power consumption tends to better scale with the system workload, as unused cores within the processor can be switched off while not needed.

As the processor power consumption comprises a notable fraction of the system's total power consumption, and power saving features have been introduced in other system parts as well, such as turning off unused memory banks or hard disks, idle power gets low and it is seldom needed to switch off a complete system for power reasons. The good news for a data center provider is that the strategies when to power up a system can be simplified because a spare system as performance buffer does not hurt the energy budget anymore.

V. Conclusions

We introduced a simple but reasonable model for the power consumption of a computer system. Our model reduces relations between load (required performance) and power consumption to a small set of parameters. In this way, we are able to analyze the essential features and power management schemes of multicore-based server systems. A validation of our ideas using state of the art power and energy saving mechanism is planned for the near future including e.g. ACPI and Intel Energy MSRs using the RAPL interface. Our main concern was to present the influence of energy saving methods on the core/chip level to the power consumption of whole server systems: The range between idle and full power consumption of a server system tends to increase in comparison to the idle power consumption. Techniques like clock and power gating as well as other power saving technologies on core level not mentioned here lead to a further increase of the described effect.

While current developments in CMOS devices lead to shrinking differences between maximum and minimum power consumption at level of a single core, the possibility to switch off cores leads to growing differences between maximum and minimum power consumption (full and idle power) of complete server systems. As a consequence the strategies on the level of cores and complete systems can be reversed: Operating systems should give more importance to switch off cores if load is low while data center operators are mostly relieved from the question when or if to switch off complete server systems to minimize energy consumption. This has a tremendous positive side effect. While waking up a core can be done in several milliseconds, waking up a complete server might take a minute. Thus, avoiding to switch off complete server systems allows data center operators to react much faster to sudden increases of load without the previously high energy penalty of running idle systems. Moreover, decreasing the number of system starts per year has the tendency to increase system life time and thus brings an additional benefit.

References

[1] M. B. Taylor. A Landscape of the New Dark Silicon Design Regime. *IEEE Micro*, 33(5):8–19, 2013.

[2] R. Schöne, D. Molka, and M. Werner. Wake-up latencies for processor idle states on current x86 processors. *Computer Science - Research and Development*, pages 1–9, 2014.

[3] Standard Performance Evaluation Corporation. All Published SPECpower_ssj2008 Results, https://www.spec.org/power_ssj2008/results/power_ssj2008.html, 2015.

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

[5] J. Srinivasan. An overview of static power dissipation. Technical report, Internal report.

[6] E. L. Sueur and G. Heiser. Dynamic voltage and frequency scaling: the laws of diminishing returns. In *Int. Conf. on Power Aware Computing and Systems*, HotPower'10, pages 1–8. USENIX Association, 2010.

[7] A. Carroll and G. Heiser. Unifying DVFS and Offlining in Mobile Multicores. In *IEEE Real Time and Application Systems (RTAS) 2014*, Berlin, 2014.

[8] A. Dhingra and S. Paul. A Survey of Energy Efficient Data Centres in a Cloud Computing Environment. *IJARCCE*, 2(10):4033–4040, 2013.

[9] J. Leverich, M. Monchiero, V. Talwar, P. Ranganathan, and C. Kozyrakis. Power Management of Datacenter Workloads Using Per-Core Power Gating. *IEEE Computer Architecture Letters*, 8(2):48–51, February 2009.

[10] L. Wang and K. Skadron. Implications of the Power Wall: Dim Cores and Reconfigurable Logic. *IEEE Micro*, (September/October):40–48, 2013.

[11] G. Da Costa. Heterogeneity: The Key to Achieve Power-Proportional Computing. *13th Symp. on Cluster, Cloud, and Grid Computing*, pages 656–662, May 2013.

[12] R. H. Dennard, F. H. Gaensslen, Y. Hwa-Nien, V.L. Rideout, E. Bassous, and A. LeBlanc. Design of Ion-Implanted MOSFETs with Very Small Physical Dimensions. *Proc. of Solid-State Circuits*, 9(5):256–268, 1974.

[13] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam, and D. Burger. Power Limitations and Dark Silicon Challenge the Future of Multicore. *ACM Transactions on Computer Systems*, 30(3):1–27, 2012.

[14] N. Hardavellas, M. Ferdman, B. Falsafi, and A. Ailamaki. Toward dark silicon in servers. *IEEE Micro*, 31(4):6–15, 2011.

[15] W. Huang, K. Rajamani, M.R. Stan, and K. Skadron. Scaling with design constraints: Predicting the future of big chips. *IEEE Micro*, 31(4):16–29, 2011.

[16] M. B. Taylor. Is dark silicon useful? Harnessing the four horsemen of the coming dark silicon apocalypse. In *Proc. of Design Automation Conference*, pages 1131–1136. ACM, 2012.

[17] G. Venkatesh, J. Sampson, N. Goulding, S. Garcia, V. Bryksin, J. Lugo-Martinez, S. Swanson, and M. B. Taylor. Conservation cores: Reducing the energy of mature computations. In *Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, pages 205–218. ACM, 2010.

[18] A. Carroll and G. Heiser. Mobile Multicores : Use Them or Waste Them. *ACM SIGOPS Oper. Syst. Rev.*, 48(1):44–48, 2014.

[19] N. Madan, A. Buyuktosunoglu, P. Bose, and M. Annavaram. A case for guarded power gating for multi-core processors. *17th Int. Symp. on HPC Architecture*, pages 291–300, 2011.