# FirmLeak: A framework for efficient and accurate runtime estimation of leakage power by firmware

Arun Joseph, Anand Haridass, Charles Lefurgy[+], Spandana Rachamalla, Sreekanth Pai, Diyanesh Chinnakkonda, Vidushi Goyal[*]
IBM Systems & Technology Group, IBM Research[+], IIT Kharagpur[*]
arujosep@in.ibm.com

***Abstract*— Separating the dynamic power and leakage power components from total microprocessor power can enable new optimizations for cloud computing. To this end, we introduce FirmLeak, a new framework that enables accurate, real-time estimation of microprocessor leakage power by system software. FirmLeak accounts for power-gating regions, per-core voltage domains, and manufacturing variation. We present an experimental evaluation of FirmLeak on a POWER7+ microprocessor for a range of hardware parts, voltages and temperatures. We discuss how this can be used in two applications to manage power by 1) improving billing of energy for cloud computing and 2) optimizing fan power consumption.**

## I. INTRODUCTION

Power consumption continues to be major concern that impacts design constraints such as reliability, thermal margins and cost for data centers and enterprise servers [1]. The contribution of leakage power to total power consumption is quite significant [2, 3, 4]. Some power management optimizations focus primarily on the leakage power component of the total power [5]. Therefore, it is valuable to estimate the leakage power apart from the total power. Runtime power management in system firmware depends on accurate, real-time power consumption sensors. The leakage power estimation in modern high-performance many-core microprocessors, must account for significant manufacturing variation (core-core and chip-chip) [6] as well as workload-induced temperature variation across cores. The introduction of on-chip voltage regulators means that cores may operate at unique voltages which will cause leakage to vary widely from core to core. Additionally, the ability to power-gate entire cores is becoming common [7]. More recently, fine-grained runtime power gating (RTPG) has been effectively adopted even for the different functional units in a processor [8, 9]. Since firmware power management may operate on a longer time intervals than power-gating of cores or function units, the ability to derate the leakage estimate to account for the actual power-on time becomes important.

### A. Prior Work

There is some amount of prior academic and industrial work which provides solutions for runtime estimation of leakage power. Wei et al. [5] proposed an equation for estimating POWER7+ [10] VDD leakage power. Ibrahim et al. [11] proposed the use of leakage power tables, indexed by voltage and temperature, which are loaded at boot time. A driver loads the table depending on a leakage identifier and, the leakage power is estimated from the measured runtime voltage and temperature. Monferrer et al. [12] proposed the use of real-time voltage, temperature, and pre-determined constants to compute leakage power from a base reference leakage power value. Accurate runtime estimation of leakage power from [5, 11] requires a very intensive (for each unique hardware part or component, and across a wide voltage and temperature operation ranges) post-silicon leakage power characterization and data collection exercise. Considering the amount of variation in newer technology nodes [6], and the notable cost involved in tester time, this is a significant overhead on the post-silicon power characterization, power management and product teams, and often late in product cycle. Prior approaches [5, 12] do not accurately account for the notable real-time response variations and relative changes in the leakage power contributions of the different device types that were used in the design of the processor, under varying conditions of process, voltages and temperatures. Also, many prior techniques do not provide efficient techniques for real-time estimates of processor power gating domain leakage power.

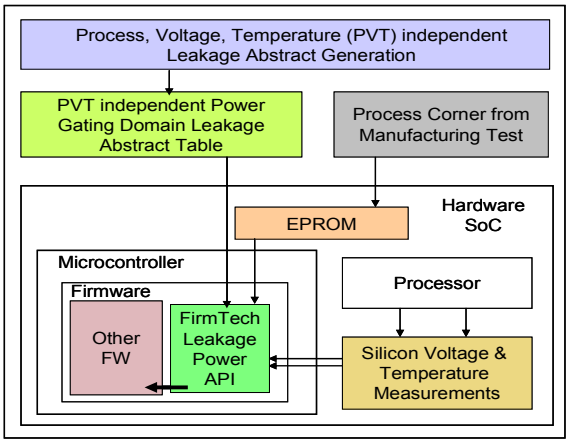


Fig. 1. FirmLeak Overview

### B. FirmLeak

In this paper, to overcome the above limitations of prior approaches, we introduce FirmLeak, a new framework for efficient runtime estimation of leakage power by firmware. Fig.1. shows a high-level overview of FirmLeak. FirmLeak introduces the use of Process, Voltage and Temperature (PVT) independent pre-silicon Power Gating domain Leakage Abstracts (PGLA) in firmware to accurately estimate per-device type contributions and total runtime leakage power. A Technology-specific Leakage power estimation Software (TLS) module, residing in firmware, reads the different PGLAs, along with real-time voltage and temperature information from on-chip hardware sensors, to enable efficient runtime estimation of leakage power. Process corner information learned from manufacturing test procedures, which

is directly or indirectly available in Vital Product Data (VPD), along with the PGLAs in firmware, enables efficient process variation aware runtime estimation of leakage power. In comparison with [5, 11], this approach enables a significant reduction in the post-silicon leakage power characterization effort (across the wide range of voltage and temperature conditions for each unique hardware part). Almost all the components in FirmLeak can be designed and developed during the pre-silicon design cycle of the processor. This capability significantly reduces the risk to overall product roadmaps. Further, the TLS module is designed such that it can be ported to the new technology nodes, and also be fine-tuned for accuracy refinements on a per device type basis, even very late in the production cycle, without the need for a significant re-characterization efforts in the hardware lab. Additionally, FirmLeak enables power-gating aware estimation by forming leakage abstracts for each power-gating domain in the chip, and by adding accompanying circuitry to track the time spent in power-gating states.

The rest of the paper is organized as follows. In Section II, we present two motivating use-cases and application of FirmLeak. Section III presents the motivation for PVT-independent LA for the power gating domains and the methodology adopted for its generation. Section IV presents leakage power estimation by firmware during runtime. Section V presents the experimental setup and results. Finally, we conclude in Section VI.

## II. MOTIVATING SCENARIOS

### A. *Cloud-based billing*

Energy consumption has become a large cost to cloud computing providers [13]. Cloud providers can differentiate their services by including energy costs in customer bills [14, 15]. This would reward customers with lower bills when they run workloads in an energy-efficient manner. While major components of a server are physically instrumented to meter power consumption, the resources assigned to Virtual Machines (VM) are often at a finer granularity. For example, some manufacturers provide measurement of microprocessor power, but none provide for physical power measurement of the cores in the microprocessor.

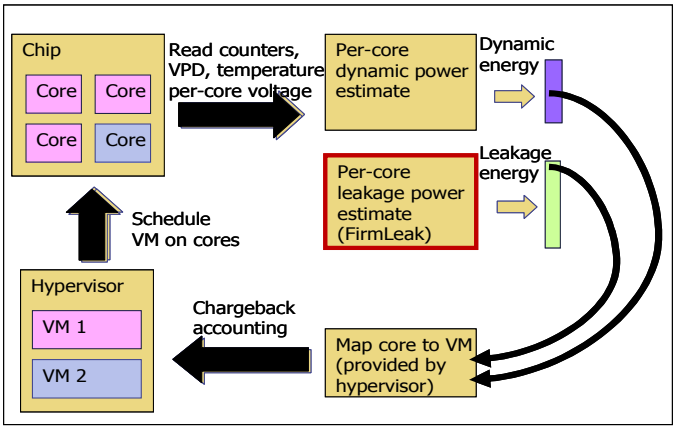


Fig. 2. Per core energy for VM energy chargeback

Since VMs are scheduled to cores, there must be a power consumption estimation of the cores for billing purposes. Power Proxies (PP) [16] have been proposed to estimate per core dynamic power. Across a broad range of workloads, they are accurate to within 10% of true power. Fig. 2 shows a possible future use of runtime dynamic and leakage power estimation for billing. As the hypervisor schedules VMs, it could use PP to estimate the dynamic power consumed by the VM during its scheduling interval. However the core event counters used by PP do not account for leakage power present even when no workload is running.

FirmLeak provides an estimate of leakage power that is complementary to PP. By using both FirmLeak and PP, as shown in Fig. 2, the complete core energy may be charged back to the VM. Billing the complete core energy consumption directly to the user provides a new incentive to use cloud resources efficiently to reduce costs. A customer who can tolerate a small reduction in performance could use per-core voltage-and-frequency scaling to substantially reduce leakage power. Recent microprocessors, such as POWER8 [17], allows per-core voltage scaling through the use of on-chip voltage. Additionally, the temperature of cores on a die will be different due to workload variation and the selected voltage level. Since voltage and temperature are likely to be different for every core, a mechanism like FirmLeak is required for per-VM billing.

### B. *Fan-based power optimization*

Fan power consumption can account for up to 23% of server power [18] and scales super-linearly with server utilization. Prior works [18, 19] shows that fan-based systems can reduce their total power by performing a joint optimization between the operating temperature of the microprocessor and the fan speed. Fig. 3 shows that as fan speed rises, the microprocessor is cooled, which reduces leakage power even as the fan power rises. This means there is an optimal temperature which reduces the sum of fan power and leakage power. A prior study presents an optimization that hunts for the optimum temperature set point [5]. A firmware feedback controller slowly adjusts fan speed while monitoring the total platform power. Since the system must thermally settle at each monitoring interval, the total settling time to arrive at the optimal fan speed is over 10 minutes.

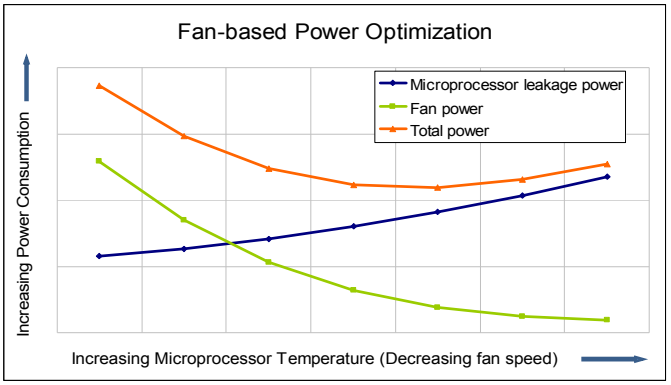


Fig. 3. Optimizing the sum of fan power and leakage power

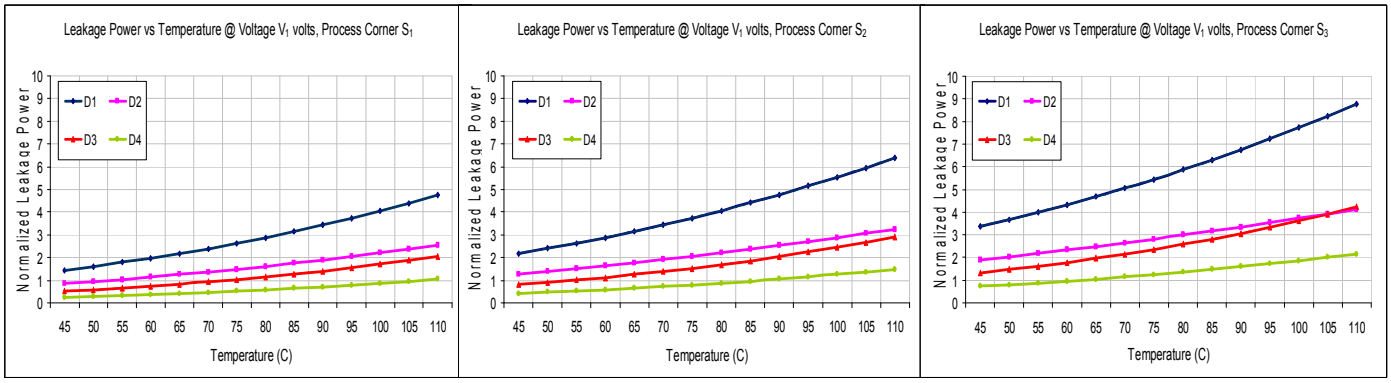


Fig. 4. Motivation for using per device type power gating abstracts in the runtime estimation of leakage power in firmware.

Therefore, the technique is not appropriate for short workloads or workloads that are not steady-state in power consumption or temperature for tens of minutes. An improved controller, proposed in [20], estimates the leakage power and fan power due to changes in the thermal setpoint to settle faster. However, it depends on a time-consuming characterization of leakage power at each voltage operating point of the microprocessor to provide a highly accurate estimate of leakage power. Additionally, it does not account for the use of power-gating. We improve the prior approach by using FirmLeak based power-domain aware leakage power estimation that vastly reduces the characterization time (since it is based on PVT-independent PGLA). As our estimates are more accurate due to the accounting of power gating effects and non-linear temperature response, the settling time of the thermal control is improved and the system arrives at an energy-efficient operating point sooner.

## III. PVT-INDEPENDENT LEAKAGE ABSTRACTS FOR POWER GATING DOMAINS

Leakage power consumption is strongly dependent on variations in process, voltage and temperature [4]. The notion of PVT-independent abstracts and its use for efficient hierarchical pre-silicon power analysis was introduced in [22]. Pre-silicon Leakage Abstracts (LA) store physical parameters, like per device type effective leaking device width that can be used to compute leakage power, and thereby are PVT-independent. The evaluation of LA for power consumption requires application of voltage, temperature, transistor threshold variations, and process sigma. In contrast, other traditional pre-silicon power models [23] are PVT-dependent as they directly reference power and do not enable efficient hierarchical abstraction.

In this paper, we introduce the creation of PGLAs, and its usage in firmware for the new application of efficient runtime estimation of leakage power. The LA based approach was preferred over other pre-silicon approaches for the following reasons. As the LA is PVT-independent it no longer needs to be unique for each unique hardware part, and also enables efficient accounting of fine-grained chip-chip, and core-core variations in process, and runtime voltage and temperature, without overheads involved in prior runtime methods [5, 12]. The fact that LA can be efficiently accumulated from lower levels of the design hierarchy to higher levels, for each voltage rail, and at each device type with minimal loss of accuracy [22], enables the efficient and accurate creation of PGLAs. LA enables the accounting of leakage power per device type for each voltage rail. As shown in Fig. 4, an experimental study of leakage power consumption of a device was done on four unique threshold voltage device types (D1, D2, D3, D4), at a fixed voltage (V1) for a wide range of temperatures (45C-110C) and three unique process corners (S1, S2, S3). From the leakage power trends for each device, it was observed that the response (slope) of each device type changes considerably with variations in process and temperature. Similar trends were observed while keeping temperature constant, but varying voltage. This motivates the need for per device type PGLAs in firmware, for enabling accurate runtime leakage power estimates for a wide range of hardware parts, voltages and temperatures. The PGLAs need not change for a change in technology characteristics, and can be fine tuned in the firmware layer, even late into the product cycle, based on the learning in the hardware lab, without having to regenerate the PGLAs. Also, the pre-silicon investment required to develop utilities to generate PGLAs is reasonable, considering the advantages it provides. The time required to generate the PVT-independent abstracts is approximately 5X less than traditional pre-silicon approaches, with a mean error of 0.073% over a range of 135 unique PVT points [21].

For each design unit that makes up the hardware block (example: microprocessor), we generate a LA, using techniques described in [22]. The LA for each power-gating domain is formed by accumulating the leakage abstracts of the physical units which are part of the same power-gating domain, on a per voltage rail, at a per device type basis. The same PGLA can be used for multiple instances of the same domain in a multi-core processor chip, as it is PVT-independent. Optionally, the LA of independent units can be stored in firmware for applications which require finer grained leakage power estimates. The different PGLA are stored in firmware for runtime estimation of leakage power. As an illustrative example we consider the POWER7+ microprocessor. The processor has 8 core chiplets, which consist of a core, L2 cache, and L3 cache. These units are partitioned into two power-gating domain types. The first domain type consists of the processor core and the L2 cache. The second domain is the

| Power-Gating Domain | Rail | Device type | | | | | |
|---|---|---|---|---|---|---|---|
| | | svt n | svt p | hvt n | hvt p | mvt n | mvt p |
| 1 | Vdd | $w_1,l_1,c_1$ | $w_2,l_2,c_2$ | $w_3,l_3,c_3$ | $w_4,l_4,c_4$ | $w_5,l_5,c_5$ | $w_6,l_6,c_6$ |
| 1 | Vcs | $w_7,l_7,c_7$ | $w_8,l_8,c_8$ | $w_9,l_9,c_9$ | $w_{10},l_{10},c_{10}$ | $w_{11},l_{11},c_{11}$ | $w_{12},l_{12},c_{12}$ |
| 2 | Vdd | $w_{13},l_{13},c_{13}$ | $w_{14},l_{14},c_{14}$ | $w_{15},l_{15},c_{15}$ | $w_{16},l_{16},c_{16}$ | $w_{17},l_{17},c_{17}$ | $w_{18},l_{18},c_{18}$ |
| 2 | Vcs | $w_{19},l_{19},c_{19}$ | $w_{20},l_{20},c_{20}$ | $w_{21},l_{21},c_{21}$ | $w_{22},l_{22},c_{22}$ | $w_{23},l_{23},c_{23}$ | $w_{24},l_{24},c_{24}$ |

Fig. 5. Power gating domain leakage abstract (PGLA) table

L3 cache associated with the chiplet. For a POWER7+ chip with 8 cores, there are 8 instances of each power-gating domain type. The leakage abstracts of the different processor core units, core level design objects, and L2 cache are accumulated to create the PGLA of the first power domain. Fig. 5 shows an illustrative example of the PGLA stored in firmware. Here w, l, c represent the effective leaking width, length and finger count respectively [21], which will be used for runtime estimation of leakage power. The columns in the table provide PGLA information for each device type in the processor. The device types shown are: super-high threshold voltage (svt), high-threshold voltage (hvt) and medium-threshold voltage (mvt) devices of both p and n transistor types.

## IV. RUN-TIME LEAKAGE POWER ESTIMATION

The leakage power of each power-gating domain in the chip is calculated as illustrated in Fig. 6. First, the PGLA information corresponding to the power gating domain and power rail is retrieved from the table in firmware. The information for each device type in the domain, along with process corner information available from manufacturing test, the present temperature (average of all thermal sensors in that power gating domain) and voltage conditions from hardware sensors, technology specific constants configured in firmware, is passed into a device level, technology specific leakage power computation model. This is repeated for all the desired voltage rails and device types of the power gating domain to compute the total leakage power for that power gating domain. In case there are no thermal sensors in a power gating domain, the average of the thermal sensors in the adjacent power gating domains can be used.

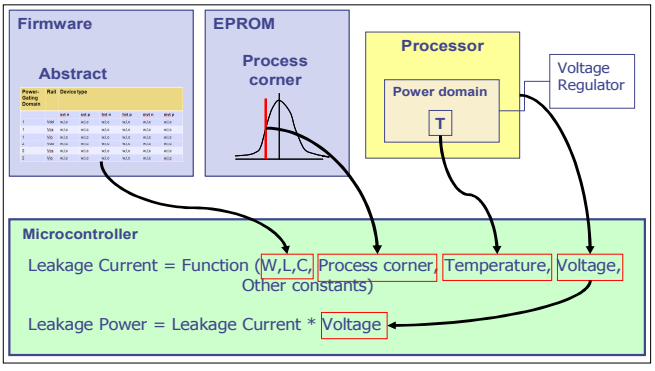


Fig. 6. Leakage power estimation via firmware

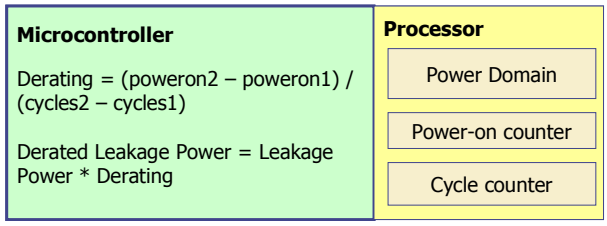


Fig. 7. Derated leakage power estimation

Power management algorithms are typically event driven or operate periodically in a real-time manner. In either case, obtaining the average power for a time interval of interest is a typical requirement. However, power-gating may happen autonomously due to other policies in the system software. Therefore, when the leakage power estimation corresponds to a long interval of time, it is possible that the power gating domain was only powered on during a fraction of the interval. Therefore, the leakage power should be derated according to the fraction of time the power domain was powered on. Fig. 7 provides an overview of the hardware, and the firmware required to perform the derating. First, there are counters operating outside the power-gating domain which count the number of fixed-rate machine cycles that have passed and the other set of counters count the number of fixed-rate machine cycles passed where the power domain was powered on. Note that fixed-rate counters increment at a fixed rate regardless of the core chiplet clock frequency and voltage setting. The firmware reads these two counters at the beginning and end of the measurement interval. The derating factor is the ratio of the incremented count in the power-on counter to the incremented count of the cycle counter during the interval, and will be a value in the range of 0% to 100%. This derating factor is multiplied by the leakage power previously determined to arrive at the final derated leakage power.

## V. EXPERIMENTAL RESULTS

For the experimental evaluation of FirmLeak, we used Power 730/740 class server [24] that uses 32nm POWER7+ processors, as shown in Fig. 8. This 2 socket server is positioned as an entry-level SMP server with up to 16 processor cores. The system planar supports two POWER7+ modules, with the memory controller of each processor module being connected to four memory buffer modules which then drive 16 DDR3 DIMM slots.

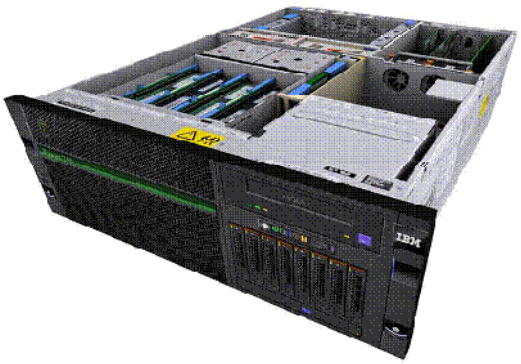

Fig. 8. POWER7+ server

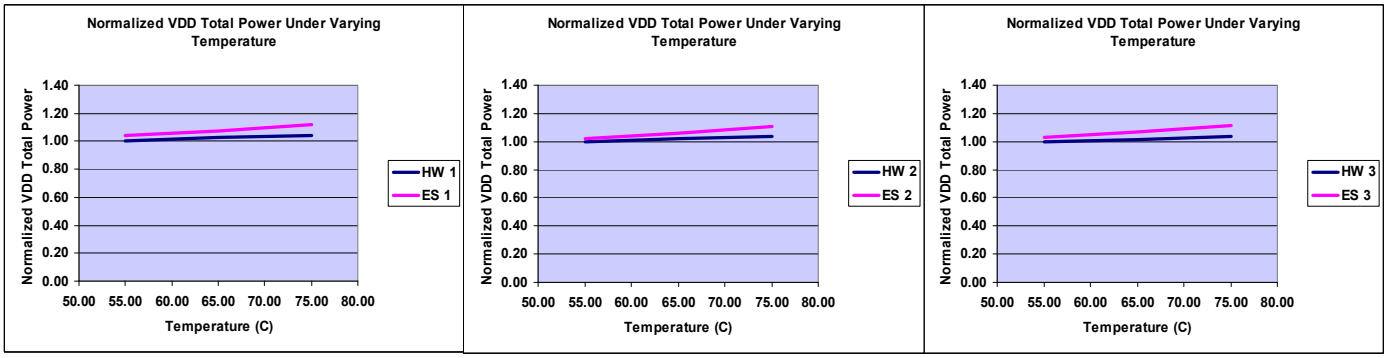


Fig. 9. VDD total power under varying conditions of temperature

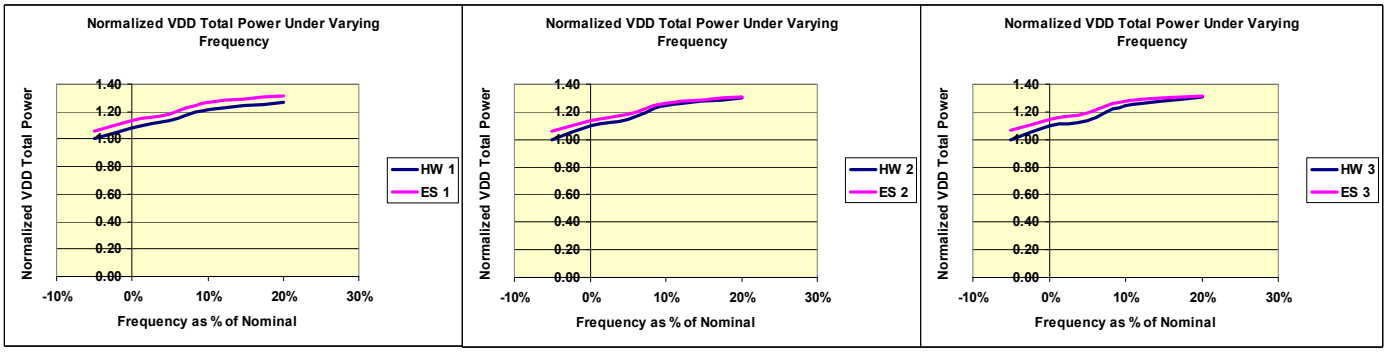


Fig. 10. VDD total power under varying conditions of frequency

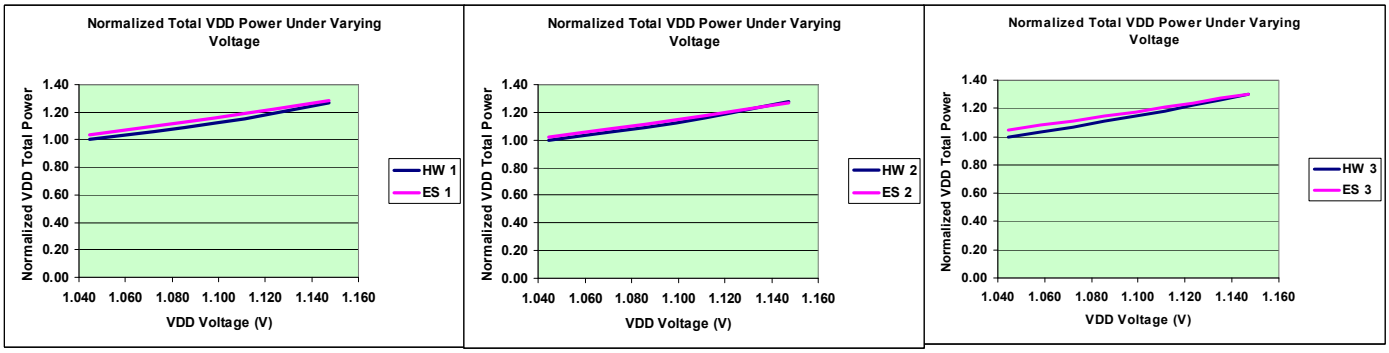


Fig. 11. VDD total power under varying conditions of voltage

Breakdown of total runtime hardware power into runtime dynamic and leakage power is non-trivial, especially under varying and heterogeneous workload activity conditions. For the experimental evaluation of FirmLeak, a constant high-utilization workload is run in an infinite loop on all cores in the processor, without power-gating any of domains in the processor. The total hardware power (HW) was compared with the summation of the leakage power estimate from FirmLeak and an accurate pre-silicon model [22] based estimate of dynamic power, to get a total runtime power estimate (ES). In terms of switching activity, the workload was defined to sustain 80% utilization of the execution resources, run the fetch engine at 100% utilization, 80% utilization of the load-store unit, and 80% utilization of the floating point resources and close to maximum utilization of the issue unit. The workload has no cache misses, address translation misses and mispredicted branches. Also, all pre-fetch engines will be running, resulting in the maximum traffic through L3 cache and the memory controllers. Additionally, to enable a more extensive evaluation of FirmLeak across a range of varying voltage, temperature and frequency conditions, the system controls were used to deterministically control the processor voltage and frequency, and the system fan speed. To get the dynamic power estimate, the workload was executed on a RTL model of the processor chip, to generate switching activity trace information, which was then fed into a dynamic power model, along with runtime information of voltage and frequency. Due to the challenges in conclusive analysis when conditions like voltage, frequency and temperature vary at the same time, a set of three different experiments were performed. The first experimental evaluation was performed under varying conditions of average on-chip temperature, but under fairly

constant conditions of on-chip voltage and frequency. The average (across all the thermal sensors) on-chip temperature was deterministically varied from $55^0$C to $75^0$C, for the three unique hardware parts, whose total power are HW1, HW2 and HW3, as shown in Fig. 9. This was achieved by iteratively and carefully controlled runtime alterations of fan speed, and processor voltage and frequency control. It may be noted that, for the same set-point voltage, a change in runtime temperature or frequency will cause a corresponding change in the current. To enable a more accurate estimation of runtime leakage and dynamic power, the VRM voltage set-point was dynamically adjusted in runtime to get a fairly constant voltage at the silicon, which was then used for the estimation of runtime total power. The average of the on-chip thermal sensors was fed into FirmLeak. The temperature profile across the processor chip is fairly uniform, with a variation of $2^0$C. This can be attributed to the homogeneous activity nature of the workload used for the evaluation. Across the entire range of parts and temperature, the estimate of runtime total power was found to have an average error of 5.1%, while the maximum errors was 7.9%.

In second experimental evaluation, chip frequency is varying under fairly constant conditions of on-chip voltage and temperature. The frequency is deterministically varied for range of -5% to +20% of nominal frequency conditions for the chip, as shown in Fig. 10. Across the range of three hardware parts and frequency, the estimate of runtime total power was found to have an average error of 3.8%, while the maximum error was 6.7%. In a third experimental evaluation, the chip VDD rail voltage is varied in runtime, but under fairly constant conditions of frequency and temperature, using techniques similar to previous experiments. For a more accurate estimation of runtime power, an estimate of on-chip silicon voltage was computed in runtime from the set-point voltage, and the runtime current drawn and IR drops. The same experiment was performed on three different hardware parts, as shown in Fig. 11. The average error, across the range of these three hardware parts and voltages, was observed to be around 2%, while the maximum error was 4.6%. Runtime power gating is known to cause transient runtime effects, as discussed in prior work [8, 9]. Due to certain hardware lab limitations beyond the scope of the paper, experimental evaluation of FirmLeak could not be done under these conditions. Highly heterogeneous workloads cause heterogeneity in temperature and voltage across the chip. But, we argue that a combination of the three experiments performed covers for such scenarios to a large extent, especially considering the accuracy requirements of the motivational use-cases FirmLeak is meant for.

## VI. SUMMARY AND CONCLUSIONS

In this paper, we present FirmLeak, a new framework that enables efficient runtime estimation of accurate and fine-grained leakage power by firmware, while accounting for power gating, core-core and chip-chip variations. We present how modeling leakage as PVT-independent power gating leakage abstracts at a per device type basis, which can be easily plugged into technology-specific leakage power model for each device, along with process, and runtime voltage and temperature information enables accurate runtime estimation of leakage power. Evaluation of FirmLeak on the POWER7+ processor chip shows that FirmLeak can be used for very accurate and efficient runtime estimation of leakage power. The effective application of FirmLeak for enabling accurate cloud-based billing and better fan-based power optimization is presented.

## VII. REFERENCES


[1] EPA report to congress on server and data center energy efficiency, August 2007.

[2] J. Hart et al., "3.6ghz 16-core SPARC SoC processor in 28nm," in Proc.2013 ISSCC (in presentation slide), pp. 48–49, 2013.

[3] W. Hu et al., "Godson-3b1500: A 32nm 1.35ghz 40w 172.8gflops 8-core processor," in Proc. 2013 ISSCC, pp. 54–55, 2013.

[4] K. Roy, et al., "Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits," Proceeding of the IEEE, vol. 91, no. 2, pp. 305-327, 2003.

[5] Wei Huang et al. "TAPO: Thermal-aware power optimization techniques for servers and data centers," Green Computing Conference and Workshops (IGCC), 2011.

[6] S. Borkar et al. ''Parameter variations and impact on circuits and microarchitecture,'' in Proc. Design Autom. Conf., Jun. 2003.

[7] M. Floyd et al. "Introducing the Adaptive Energy Management Features of the POWER7 chip," IEEE Micro, March/April 2011.

[8] M. Kondo, et al. "Design and Evaluation of Fine-Grained Power-Gating for Embedded Microprocessors," Proc. the DATE2014, March 2014.

[9] Z. Hu et al., "Microarchitectural techniques for power gating of execution units," in Proc. the 2004 ISLPED, pp. 32–37, 2004

[10] S. Taylor, "POWER7+™: IBM's next generation POWER microprocessor," Hot Chips 24, 2012.

[11] US Patent 8,527,794: "Realtime power management of integrated circuits," Ibrahim, A. and Dwarakanath, A. and Shimizu, D.P .

[12] US Patent 7,814,339: "Leakage power estimation," Monferrer, P.C. and Magklis, G. and Gonzalez, J. and Gonzalez, A.

[13] Shekhar Srikantaiah, Aman Kansal, and Feng Zhao. "Energy aware consolidation for cloud computing," In Proceedings of the 2008 conference on Power aware computing and systems (HotPower'08).

[14] Jimenez et al., "Energy-aware accounting and billing in large-scale computing facilities," IEEE Micro 31(3) (2011).

[15] Bellosa, F. "The benefits of event-driven energy accounting in power-sensitive systems," ACM SIGOPS European Workshop (2000).

[16] Wei Huang et al. "Accurate Fine-Grained Processor Power Proxies," In Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-45).

[17] Fluhr, E.J., et al. "POWER8: A server-class processor in 22nm SOI with 7.6 Tb/s off-chip bandwidth," In Proc. Int'l Solid State Circuits Conference (ISSCC), Feb. 2014.

[18] Z. Wang et al. "Optimal fan speed control for thermal management of servers," In Proc. of the ASME/Pacific Rim Technical Conf. and Exhibition on Packaging and Integration of Electronic and Photonic Systems, MEMS and NEMS (IPACK), July 2009.

[19] D. Shin et al. "Energy-optimal dynamic thermal management for green computing," In Proc. Of Intl. Conf. on Computer-Aided Design (ICCAD), November 2009.

[20] US 20130116963A1: "Minimizing Aggregate Cooling and Leakage Power with Fast Convergence," Malcolm S. et al.

[21] Dhanwada, N. et al. "Leakage Power Contributor Modeling," Design & Test of Computers, IEEE , vol.29, no.2, pp.71,78, April 2012.

[22] Dhanwada, N. et al. "Efficient PVT independent abstraction of large IP blocks for hierarchical power analysis," Computer-Aided Design (ICCAD), 2013 IEEE/ACM International Conference on, pp.458,465, 18-21 Nov. 2013.

[23] Liberty Reference Manual, Sep. 2008, Version 2008.09

[24] IBM Redbooks: "IBM Power 710 and 730 Technical Overview and Introduction," May 2013.