Harnessing Voltage Margins for Energy Efficiency in Multicore CPUs

George Papadimitriou[*] Manolis Kaliorakis[*] Athanasios Chatzidimitriou[*] Dimitris Gizopoulos[*]
Peter Lawthers[‡] Shidhartha Das[§]

[*] University of Athens, Greece {georgepap | manoliskal | achatz | dgizop}@di.uoa.gr
[‡] Applied Micro Circuits Corporation Deutschland GmbH, Munich, Germany plawthers@apm.com
[§] ARM Ltd., Cambridge, UK shidhartha.das@arm.com

ABSTRACT

In this paper, we present the first automated system-level analysis of multicore CPUs based on ARMv8 64-bit architecture (8-core, 28nm X-Gene 2 micro-server by AppliedMicro) when pushed to operate in scaled voltage conditions. We report detailed system-level effects including SDCs, corrected/uncorrected errors and application/system crashes. Our study reveals large voltage margins (that can be harnessed for energy savings) and also large V_{min} variation among the 8 cores of the CPU chip, among 3 different chips (a nominal rated and two sigma chips), and among different benchmarks.

Apart from the V_{min} analysis we propose a new composite metric (severity) that aggregates the behavior of cores when undervolted and can support system operation and design protection decisions. Our undervolting characterization findings are the first reported analysis for an enterprise class 64-bit ARMv8 platform and we highlight key differences with previous studies on x86 platforms. We utilize the results of the system characterization along with performance counters information to measure the accuracy of prediction models for the behavior of benchmarks running in particular cores. Finally, we discuss how the detailed characterization and the prediction results can be effectively used to support design and system software decisions to harness voltage margins for energy efficiency while preserving operation correctness. Our findings show that, on average, 19.4% energy saving can be achieved without compromising the performance, while with 25% performance reduction, the energy saving raises to 38.8%.

CCS CONCEPTS

• Hardware → Power and energy → Power estimation and optimization; • Hardware → Robustness → Hardware reliability → Process, voltage and temperature variations

MICRO-50, October 14–18, 2017, Cambridge, MA, USA

ACM ISBN 978-1-4503-4952-9/17/10...$15.00
https://doi.org/10.1145/3123939.3124537

KEYWORDS

Energy efficiency, voltage and frequency scaling, power consumption, error detection and correction, multicore CPUs characterization, micro-servers.

ACM Reference format:

G. Papadimitriou, M. Kaliorakis, A. Chatzidimitriou, D. Gizopoulos, P. Lawthers, and S. Das. 2017. Harnessing Voltage Margins for Energy Efficiency in Multicore CPUs. In *Proceedings of the 2017 50th IEEE/ACM International Symposium on Microarchitecture (MICRO-50).* ACM, Cambridge, MA, USA, 14 pages. DOI=https://doi.org/10.1145/3123939.3124537

1 INTRODUCTION

During chip fabrication, process variations can affect transistor dimensions (length, width, oxide thickness etc. [1]) which have direct impact on the threshold voltage of a MOS device [2]. As technology scales, the percentage of these variations compared to the overall transistor size increases and raises major concerns for designers, who aim to improve energy efficiency. This variation is classified as static variation and remains constant after fabrication. On top of that, transistor aging and dynamic variation in supply voltage and temperature, caused by different workload interactions, is also of primary importance. Both static and dynamic variations lead microprocessor architects to apply conservative guardbands (operating voltage and frequency settings) to avoid timing failures and guarantee correct operation, even in the worst-case conditions excited by unknown workloads [3, 4].

However, these guardbands impede the low power consumption and the high performance, which can be derived by reducing the supply voltage and increasing the operation frequency, respectively. To bridge the gap between energy efficiency and performance improvements, several hardware and software techniques have been proposed, such as Dynamic Voltage and Frequency Scaling (DVFS) [5]. The premise of DVFS is that the microprocessor's workloads as well as the cores' activity vary. Voltage and frequency-scaling during epochs where peak performance is not required enables a DVFS-capable system to achieve average energy-efficiency gains without affecting peak-performance adversely. At a specific frequency of operation, energy-efficiency gains are limited by guardbands that guarantee correct operation in the presence of dynamic margins.

Several techniques have been proposed [6, 7] that eliminate a subset of these guardbands for efficiency gains over and above what is dictated by design guardbands. However, all of these techniques are associated with significant design, test and measurement overheads that limit its application in the general case. For instance, in the Razor technique [6], support for timing-error detection and correction has to be explicitly designed into the processor micro-architecture which has significant verification overheads. Similarly, in adaptive-clocking approaches [7], extensive test and measurement effort is required for system sign-off. Ensuring the eventual success of these techniques requires a deep understanding of dynamic margins and their manifestation during normal code execution.

Revealing and fine-grained harnessing the pessimistic design-time voltage margins offers a significant opportunity for energy-efficient computing in multicore CPUs. The full energy savings potential can be exposed only when accurate *core*-to-*core*, *chip*-to-*chip*, and *workload*-to-*workload* voltage scaling variation is measured. When all these levels of variation are identified, system software can effectively allocate hardware resources to software tasks matching the capabilities of the former (undervolting potential of the CPU cores) and the requirements of the latter (for energy or performance).

Although characterization studies for CPUs and GPUs have been presented recently [4, 8 – 11], they primarily focus on coarse-grained identification of the V_{min} values, i.e. the voltage level at which no type of anomaly is observed in program execution of a particular core. Furthermore, these studies focus primarily on x86 and Power-series enterprise-class server systems whose summary is shown in Table 1; studies on GPU chips have been reported as well.

Table 1: Summary of studies on commercial chips.

ISA	Processor	Technology	Ref.
POWER 7 / 7+	IBM Power 750, 780	45 / 32 nm	[7, 8]
x86 – IA64 extension	Intel Itanium 9560	32 nm	[9, 10]
Nvidia Fermi / Kepler	GTX 480, 580, 680, 780	40 / 28 nm	[11]
ARMv8	APM X-Gene 2	28 nm	*This work*

1.1 Contributions of this Work

In this paper, we present the first detailed system-level voltage scaling characterization study for ARMv8-based multicore CPUs manufactured in 28nm. The study's backbone is a fully automated system-level framework built around AppliedMicro's (APM) X-Gene 2 micro-server. The automated infrastructure aims to increase the throughput of massive undervolting campaigns that require multiple benchmarks execution at several voltage supply levels of all individual cores. The characterization process requires minimal human intervention and records all possible abnormalities due to undervolting: *silent data corruptions* (*SDC*, i.e. program output mismatches without any hardware error notification), *corrected errors, uncorrected* (but *detected*) *errors* (provided by Linux EDAC driver [12]), as well as *application* and *system crashes* [13].

A second contribution of the paper towards the formalization of the behavior in undervolting conditions is the definition of a simple consolidated function. *Severity* function aggregates the effects of reduced voltage operation in the cores of a multicore CPU by assigning values to the different abnormal observations. The lower the voltage level, the higher the value of the severity function. The severity function assists an undervolting *classification* of the cores of a CPU chip for a given benchmark: different core, benchmark and voltage values lead to different severity patterns, some with an abrupt increase to the severity (i.e. the benchmark keeps executing correctly until a voltage level at which the system crashes), while others have a "smooth" severity increase while voltage is reduced (the system remains responsive throughout a range of voltage values but it generates ECC errors or produces SDCs). The fine-grained analysis of the behavior of the machine using the severity function can assist energy efficiency decisions for task-to-core allocation by the system software. Our comprehensive characterization for ARMv8-based multicore CPUs confirms that a different microarchitecture, circuit design or manufacturing technology exhibits different abnormal behavior when operating beyond nominal voltage conditions. Unlike previous studies for Intel systems with Itanium CPUs [9, 10], in our system silent data corruptions (alone or with ECC errors) appear at higher voltage levels than corrected errors alone for a significant set of benchmarks. In the previous studies [9, 10], the range of voltage levels with corrected errors only, offer a significant opportunity for energy savings without jeopardizing correctness of operation; this is *not* the case in the APM X-Gene 2 ARMv8-based CPUs of our study. Understanding the behavior in non-nominal conditions is very important for making software and hardware design decisions for improved energy efficiency that preserves correctness of operation.

Furthermore, we feed the characterization results along with performance counter measurements to a linear regression based statistical analysis model to predict both the V_{min} and the *severity* behavior of cores and benchmark combinations. Our study reveals that a simple linear regression model is efficient to predict the severity behavior of a core running a particular benchmark in off-nominal voltage conditions. The same model is also capable of accurately predicting the V_{min}, but this is also the case for the naïve prediction model of averaging the V_{min} values of the training dataset due to the limited range of dynamic V_{min} variation. We show that we can potentially achieve on average 19.4% energy savings without compromising the performance, while with 25% performance loss we can achieve 38.8% energy savings. The characterization modeling and the prediction results of our study can be effectively used to support design and system software decisions to harness voltage margins and thus improve energy efficiency while preserving operation correctness.

2 EXPERIMENTAL SETUP

In this section, we describe the system architecture, voltage and frequency domains of the APM X-Gene 2 micro-server. We also

present the automated characterization framework, with which we characterize three X-Gene 2 chips, and is the first contribution of this work.

2.1 System Architecture

The APM X-Gene 2 micro-server consists of eight 64-bit ARMv8-compliant cores. The X-Gene 2 architecture offers high-end processing performance and capabilities. For example, the X-Gene 2 subsystem features a Power Management processor (PMpro) and a Scalable Lightweight Intelligent Management processor (SLIMpro) to enable breakthrough flexibility in power management, resiliency and end-to-end security for a wide range of applications. The dedicated PMpro processor provides advanced power management capabilities, such as multiple power planes and clock gating, thermal protection circuits, Advanced Configuration Power Interface (ACPI) power management states and external power throttling support. The dedicated SLIMpro processor monitors system sensors, configures system attributes (e.g. regulate supply voltage, change DRAM refresh rate etc.) and accesses all error reporting infrastructure, using an integrated I2C controller as the instrumentation interface between the X-Gene 2 cores and this dedicated processor. SLIMpro can be accessed by the system's running Linux Kernel.

X-Gene 2 has three independently regulated power domains (as shown in Figure 1):

PMD (Processor Module): Each PMD contains two ARMv8 cores. Each of the two cores has separate instruction and data L1 caches, while they share a unified L2 cache. The operating voltage of all four PMDs together can change with a granularity of 5mV beginning from 980mV. While PMDs operate at the same voltage, each PMD can operate in a different frequency. The frequency can range from 300 MHz up to 2.4 GHz at 300 MHz steps.

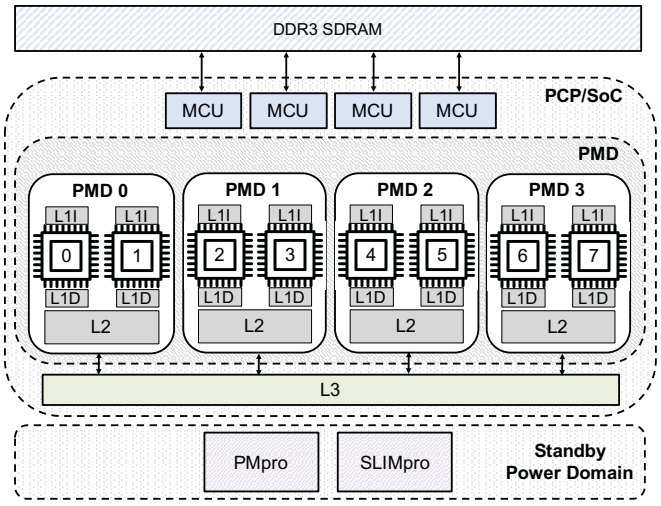

Figure 1: X-Gene 2 block diagram.

PCP (Processor Complex)/SoC: It contains the L3 cache, the DRAM controllers, the central switch and the I/O bridge. The PMDs do not belong to the PCP/SoC power domain. The voltage of the PCP/SoC domain can be independently scaled downwards with a granularity of 5mV beginning from 950mV.

Standby Power Domain: This includes the SLIMpro and PMpro microcontrollers and the interfaces for the I2C buses.

Table 2 summarizes the most important architectural and microarchitectural parameters of the APM X-Gene 2 micro-server that is used in the paper.

Table 2: Basic parameters of APM X-Gene 2.

Parameter	Configuration
ISA	ARMv8 (AArch64, AArch32, Thumb)
Pipeline	64-bit OoO (4-issue)
CPU	8 cores
Core clock	2.4 GHz
L1 Instr. cache	32KB per core (Parity Protected)
L1 Data cache	32KB per core (Parity Protected)
L2 cache	256KB per PMD (ECC Protected)
L3 cache	8MB (ECC Protected)
Technology	28 nm
Max TDP	35 W

2.2 Characterization Framework

The primary goals of the characterization framework used in this study are: (1) to identify the target system's limits when it operates at scaled voltage and frequency conditions, and (2) to record/log the effects of a program's execution under these conditions. The framework provides the following features:

- compares the outcome of the program with the correct output of the program when the system operates in nominal conditions to record Silent Data Corruptions (SDCs),
- monitors the exposed corrected and uncorrected errors from the hardware platform's error reporting mechanisms,
- recognizes when the system is unresponsive and restores it automatically,
- monitors system failures (crash reports, kernel hangs, etc.),
- determines the safe, unsafe and non-operating voltage regions for each application for all frequencies, and
- performs massive repeated executions of the same configuration.

The characterization framework (outlined in Figure 2) is fully automated, easily configurable by the user and can be embedded to any Linux-based system, with similar voltage and frequency regulation capabilities. To completely automate the characterization process, and due to the frequent and unavoidable system crashes that occur when the system operates in reduced voltage levels, we set up a Raspberry Pi board (see Figure 2) connected externally to the X-Gene 2 board as a watchdog monitor. The Raspberry Pi is physically connected to both the Serial Port, as well as to the Power and Reset buttons of the system board to enable physical access and remote control to the system.

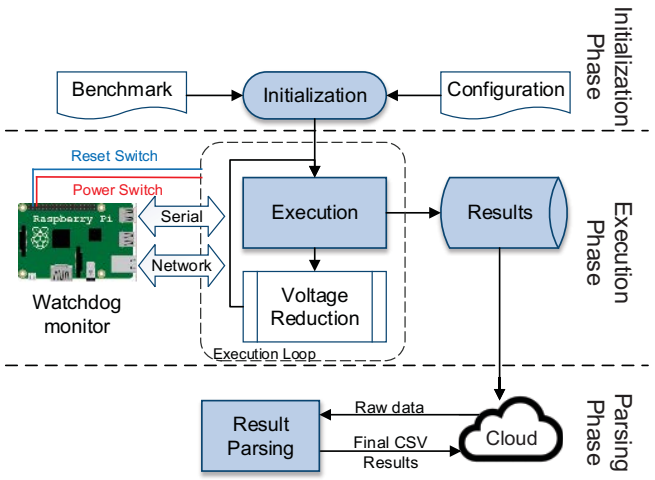

Figure 2: Characterization framework layout.

As shown in Figure 2, the characterization framework consists of three phases: initialization, execution, and parsing. During the *initialization* phase, a user can declare a benchmark list with corresponding input datasets to run in any desirable characterization setup. The characterization setup includes the voltage and frequency (V/F) values on which the experiment will take place and the cores where the benchmark will be run. The characterization setup depends on the power domains supported by the chip, but our framework is easily extensible to support the power domain features of different CPU chips. The *execution phase* consists of multiple runs of the same benchmark, each one representing the execution of the benchmark in a pre-defined characterization setup. The set of all the characterization runs running the same benchmark with different characterization setups represents a *campaign*.

In the *parsing phase* of our framework, all log files that are stored during the execution phase are parsed in order to provide a fine-grained classification of the effects observed for each characterization run. Note that, each run is correlated to a specific benchmark and characterization setup. The categories that are used for our classification are summarized in Table 3, but the parser can be easily extended according to the user's needs. For instance, the parser can also report the exact location that the correctable errors occurred (e.g. the cache level, the memory, etc.) using the logging information provided by the execution phase. At the end of the parsing step, all the collected results concerning the characterization (according to Table 3) and the severity function of each run are reported in CSV files.

2.2.1 Characterization Challenges. In this section, we discuss the most important challenges that were taken into consideration for the solid development of the characterization framework to ensure correct and accurate results.

Safe Data Collection. Given that a system operating beyond nominal conditions often has unexpected behaviors (e.g. file system driver failures), there is need to correctly identify and store all the essential information in log files (to be subsequently parsed and analyzed). The automated framework was developed to collect and store safely all the necessary information about the experiments. Therefore, after each run of the benchmark beyond nominal voltage conditions, the framework restores the microprocessor in nominal voltage conditions to store the log files and then it continues to the next experiment.

Table 3: Effects classification.

Effect	Description
NO (Normal Operation)	The benchmark was successfully completed without any indications of failure.
SDC (Silent Data Corruption)	The benchmark was successfully completed, but a mismatch between the program output and the correct output was observed.
CE (Corrected Error)	Errors were detected and corrected by the hardware (provided by Linux EDAC driver).
UE (Uncorrected Error)	Errors were detected, but not corrected by the hardware (provided by Linux EDAC driver [12]).
AC (Application Crash)	The application process was not terminated normally (the exit value of the process was different than zero).
SC (System Crash)	The system was unresponsive; meaning that the X-Gene 2 is not responding or the timeout limit was reached.

Failure Recognition. Another challenge is to recognize and distinguish the system and program crashes or hangs. This is a very important feature for the *parsing phase* to easily identify and classify the final results, with the most possible distinct information concerning the characterization.

Reliable Cores Setup. Another major challenge we also faced is that the characterization of a system is performed primarily by using properly chosen programs in order to provide diverse behaviors and expose all the potential deviations from nominal conditions. It is thus important to run the selected benchmarks in a *reliable cores setup*. This means that the cores, on which the benchmark runs, must be isolated and unaffected from the other active processes of the kernel in order to capture only the effects of undervolting on the studied benchmark. Further, to avoid any abnormal behavior sourcing from other cores of the microprocessor (not the one under characterization) and due to the fact that all the PMDs are in the same power domain, the framework sets the lowest frequency to all cores (300 MHz) but keeps the frequency high to the cores under characterization.

Massive Iterative Execution. The non-deterministic behavior of the characterization results due to several microarchitectural features makes it necessary to repeat the experiments multiple times with the same configuration to capture the divergences that may occur during different runs of the same configuration.

3 SYSTEM CHARACTERIZATION

We study the behavior of 3 different X-Gene 2 chips by using representative benchmarks from the SPEC CPU2006 suite to

explore the voltage guardbands for each core of the chip, and thus to detect the safe V_{min} in which the benchmarks can be executed correctly. We also study any abnormal behavior that can be exposed (SDC, ECC errors, crashes, etc.) below the safe V_{min} levels, for a comprehensive characterization. We present our findings for three different chips: TTT, TFF, and TSS. The TTT part is the "normal" part. The TFF is a fast corner part, which has high leakage but at the same time can operate at higher frequency. The TSS part is also a corner part which has low leakage and works at lower frequency. All chips have maximum frequency equal to 2.4 GHz, however, the TSS chip may have a lower voltage guardband than the TTT and TFF chips due to its lower power leakage.

3.1 Regions of Operation

Using the automated framework presented in subsection 2.2, we extensively characterize the three X-Gene 2 chips. The characterization process can reveal for each core of the CPU three different regions of operation when the microprocessor operates beyond nominal voltage conditions. These are the *safe* and *unsafe* operating regions and the region in which the system cannot operate (*crash region*).

To isolate the impact of temperature that can affect our results, apart from the isolation of system processes (see subsection 2.2.1), we also control the temperature by adjusting the CPU's fan speed accordingly. We stabilize the temperature at 43°C, and thus, all benchmarks complete their execution at the same temperature. In Figure 4 we present the results for 10 SPEC CPU2006 benchmarks [14]. All programs ran on a single core in each PMD at 2.4 GHz, while the remaining six cores (the other 3 PMDs) reliably operated at 300 MHz (see explanation in subsection 2.2.1). In order to consider the non-deterministic behavior of such experiments and maintain their statistical importance, we ran every undervolting campaign 10 times. Figure 4 presents in detail for all benchmarks the highest V_{min} values and the highest *crash* voltage values of the ten campaigns for the three different chips and all the cores of each chip. In all benchmarks, we can notice the three regions of operation according to the collected results. The regions are:

- *Safe* region (blue): The characterization runs that correspond to this region had a normal operation (NO) without any SDCs, errors or crashes.
- *Unsafe* region (grey): The characterization runs that correspond to this region generate an abnormal behavior (SDC, CE, UE, AC) but not a system crash.
- *Crash* region (black): This region includes voltage values in which at least one characterization run led to a system crash.

3.2 V_{min} Experimental Results

We experimentally obtain the V_{min} values of the 10 SPEC CPU2006 benchmark on the three X-Gene 2 chips (TTT, TFF, TSS), running the entire time-consuming undervolting experiment ten times for each benchmark. These experiments were performed during 6 months on a single X-Gene 2 machine. This part of our study focuses on a quantitative analysis of the safe V_{min} for diverse chips of the same architecture in order to expose the potential guardbands of each chip, as well as to quantify how the program behavior affects the guardband and to measure the core-to-core and chip-to-chip variation.

The voltage guardband for each program is the smallest (safe) margin between the nominal voltage of the microprocessor and its V_{min}. Our experiments were performed in two different frequencies: the highest available frequency of the X-Gene 2, which is 2.4 GHz, and 1.2 GHz. It is essential to highlight that the X-Gene 2 supports clock skipping and clock division, which, in combination, set the effective frequency of the PMD relative to its clock source. Clock ratios greater or less than 1/2 are implemented via clock skipping on the input clock. Clock ratio equal to 1/2 is implemented via clock division on the input clock. This means that, clock frequencies greater than 1.2 GHz have similar behavior as in 2.4 GHz, and frequencies less than 1.2 GHz have similar behavior as in 1.2 GHz. For this reason, we haven't characterized the chips in the intermediate frequencies.

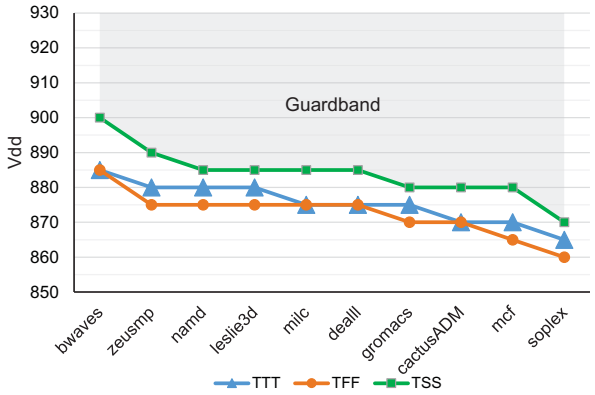

Figure 3: V_{min} results at 2.4 GHz for 10 SPEC CPU2006 programs on 3 different X-Gene 2 chips (TTT, TFF, TSS).

In the "normal" TTT part running at 1.2 GHz we observed that all programs have safe V_{min} at 760mV for all cores. Furthermore, no program in any core exposes any abnormal behavior (SDC, CE, crash, etc.) after the V_{min} and before the system crash. We observe only system crashes below the safe V_{min}. On the other hand, in the maximum frequency (and this is also the expectation for all intermediate frequencies between 2.4 GHz and 1.2 GHz as we described before) we observed divergences of the V_{min} values as shown in Figure 3 (blue line). For a significant number of benchmarks, we can see variations between different programs and different chips. Figure 3 represents the most robust core for each chip, and for these programs the V_{min} varies from 885mV to 860mV for TTT, from 885mV to 870mV for TFF and from 900mV to 870mV for TSS. Considering that the nominal voltage for the X-Gene 2 is 980mV, there is a significant reduction of voltage without affecting the correct execution of programs, which is equal to at least 18.4% for the TTT and TFF chip, and 15.7% for the TSS chip. We also notice in Figure 3 that the workload-to-workload variation remains the same across the 3 chips of the same architecture; however, there is a relatively large variation among the chips. This means that there is a program dependency of V_{min} behavior in all chips.

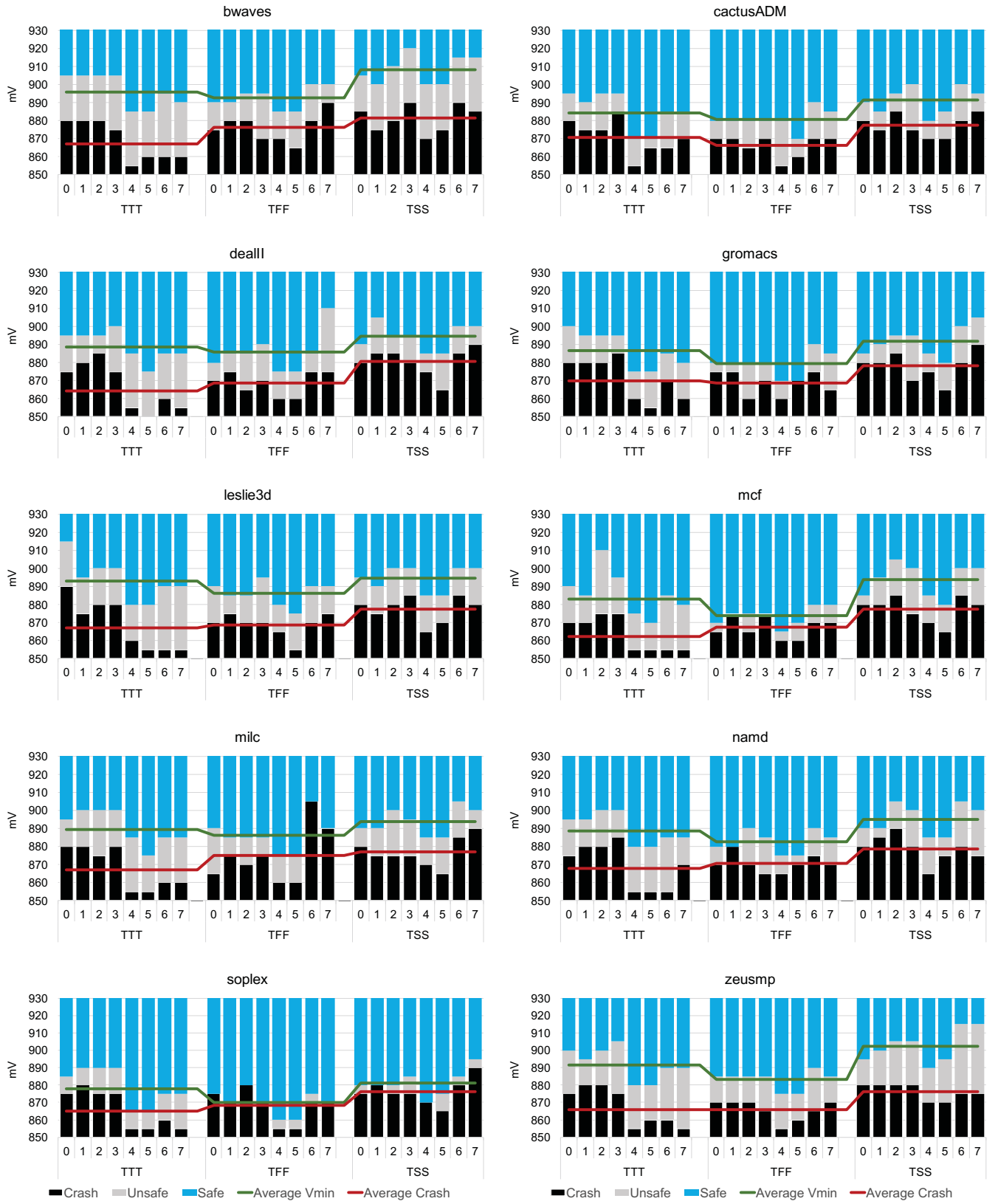

Figure 4: X-Gene 2 characterization results for 10 SPEC CPU2006 benchmarks (10 runs each) on three different chips (TTT, TFF, TSS). Blue represents the Safe region, grey represents the Unsafe region and black represents the Crash region.

3.3 Process Variation

Figure 4 presents the detailed information about the safe V_{min} for all benchmarks and cores of the three chips, as well as the range of the unsafe region. In this section, we discuss the core-to-core and chip-to-chip variation.

Core-to-Core Variation: There are significant divergences among cores for the same benchmark due to process variation. Process variations can affect transistor dimensions (length, width, oxide thickness etc.) which have direct impact on the threshold voltage of a MOS device, and thus, on the guardband of each core. This variation among cores of the same chip can result in high energy savings by using the appropriate task scheduling in combination with a comprehensive prediction. We present and discuss this method in the following sections (Section 4 and 5).

Chip-to-Chip Variation: As Figure 4 shows, PMD 2 (cores 4 and 5) is the most robust PMD for all three chips (up to 3.6% more voltage reduction compared to the most sensitive cores). Green line in Figure 4 presents the average V_{min}, and the Red line represents the average Crash voltage point for each chip. Thus, we can notice that the TFF chip has lower V_{min} points than the TTT chip, in contrast to TSS (the chip with lower leakage), which has significantly higher V_{min} points than the other two chips, and thus, lower power savings. For the unsafe region, on the other hand, we notice only small divergences among the chips.

3.4 Abnormal Behaviors below V_{min}

Previous studies on Intel Itanium CPUs [9, 10] have shown a large region of voltage values that contains only ECC *corrected* errors during undervolting. By reducing the voltage on those chips, the number of corrected errors increases gradually for quite many voltage steps until it exposes other types of abnormal behavior (SDCs, uncorrected errors, crashes). In such systems, ECC corrected errors can serve as proxies for the effects of undervolting. In contrast to these studies, a major finding of our characterization for ARMv8-compliant multicore CPUs is that *silent data corruptions appear at higher voltage levels than corrected errors alone for any benchmark*. In [9, 10], the reported range of voltage levels with corrected errors alone offers a significant opportunity for energy savings without jeopardizing correctness of operation. High correctable error rate is helpful to an ECC guided voltage speculation but this is not the case in the APM X-Gene 2 in our case.

To justify the differences between X-Gene 2 and previous studies on Itanium [9, 10], we developed and ran self-tests that separately stress each cache level independently as well as the ALU and FPU. Cache tests completely fill the cache arrays and flip all the bits of each cache block to check for cell bit errors during undervolting. ALU and FPU tests perform multiple different concurrent operations in each unit with random values to stress different paths and conditions. Through this component-focused stress process we observed the following: (1) SDCs occur when the pipeline gets stressed (ALU and FPU tests), and (2) the cache bit-cells safely operate at higher voltages (the cache tests crash in much lower voltages than the ALU and FPU tests). This observation leads us to conclude that the X-Gene 2 is more susceptible to timing-path failures than to SRAM array failures. In contrast, ECC corrections appear at a higher voltage on the Itanium compared to SDCs and system crashes. We attribute the increased robustness to timing-failures on the Itanium to circuit-level dynamic-margin mitigation techniques such as the capability to perform continuous clock-path de-skewing during dynamic operation [15]. The X-Gene 2 does not deploy such circuit-level techniques, and thereby, generates SDCs due to timing-path failures. Having the occurrence of *SDCs first*, it is not possible to easily guide the voltage speculation for prediction based on the manifested errors. For that reason, we propose the *severity function* both for quantifying the severity and for illustrating the scaling of abnormal behaviors due to voltage reduction, which is the second contribution of this paper. The new metric's contribution is twofold: (1) to aggregate the results produced by multiple runs, and (2) to quantify a microprocessor's ability to operate beyond nominal conditions and especially beyond the safe V_{min}.

3.4.1 Severity Function. Note that each characterization run can manifest multiple effects. For instance, in a run both SDC and CE can be observed; thus, both of them are reported for this run. Due to the non-determinism of the characterization in real hardware, all the information collected during multiple campaigns of the same benchmark (iterative execution) is also reported by our *severity* function. To quantify the criticality of the effects of different experimental runs of different campaigns with the same setup, we define the "severity function" S_v, where v is the voltage, as follows:

$$S_v = W_{SDC}\cdot\frac{SDC}{N} + W_{CE}\cdot\frac{CE}{N} + W_{UE}\cdot\frac{UE}{N} + W_{AC}\cdot\frac{AC}{N} + W_{SC}\cdot\frac{SC}{N}$$

In this function, the parameters *SDC*, *CE*, *UE*, *AC* and *SC* can take values from 0 to N (N is the number of runs at voltage level v), and represent the times that the effect appears to these runs (for example, if k of the N runs lead to UE, then parameter UE is set to k; the actual number of uncorrected errors during each run is not taken into consideration). Parameters W_{SDC}, W_{CE}, W_{UE}, W_{AC} and W_{SC} represent "weights" that can be arbitrarily set to characterize the severity of each effect of Table 3. The higher the weight, the more critical the effect is considered by our function, and the main role of these weights is to "translate" the behaviors (SDCs, etc.) to numbers in order to fit to the equation. We use the values presented in Table 4 as the values for our severity function (but different weight values can be also used according to the importance of each observed abnormal behavior in a particular system study).

Table 4: Weights used in our experiments.

Weight	Value
W_{SC}	16
W_{AC}	8
W_{SDC}	4
W_{UE}	2
W_{CE}	1
W_{NO}	0

As an example, Figure 5 shows the severity of *bwaves* benchmark on TTT chip, which has a significantly large *unsafe region*, and it also provides a smooth gradual increase of severity while the voltage is reduced. These results are derived from 10 executions of the same campaign. According to the severity value for each voltage level, one can decide if and when it is possible to reduce the voltage further (lower than the safe V_{min} where severity is 0) for more aggressive energy efficiency. The severity function is essential primarily for speculation (see details in section 4), because (1) it collects all needed information in one metric, (2) it incorporates the small divergences across multiple executions (10 executions in our case), (3) it measures the mean severity for each voltage step in each core, and (4) it assigns numbers to behaviors (SDC, CE, etc.), and thus, it is easy to use by any software daemon such as an online predictor.

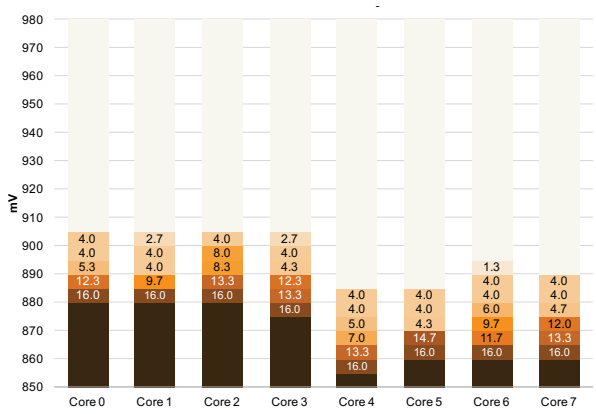

Figure 5: *bwaves* benchmark severity on TTT chip cores.

4 PREDICTION

Predicting safe voltage regions of the microprocessor using as input the performance counters provided by the system has recently gained the interest of the computer architecture community [9, 10, 17]. In this section, we study the feasibility of predicting the safe V_{min} (for a conservative prediction by taking into account only the guardbands – the V_{min} – for each core and workload), as well as a more aggressive prediction by using the severity values (described in section 3.4.1) using the microarchitectural events measured for the entire benchmarks execution provided by the X-Gene 2 hardware.

Specifically, we implemented linear regression analysis in all the cores of the three chips (TTT, TFF, and TSS) using all benchmarks from the SPEC CPU2006 suite with all their input datasets (40 programs). Due to space limitations, we illustrate only the most representative cases of our analysis targeting both the V_{min} and the severity values of the *most robust* core (Core 4) and the *most sensitive* core (Core 0) of the TTT chip. Our analysis shows that prediction using severity values is more efficient than targeting only the V_{min} point for any core and chip.

In our study, we used linear regression models, which are able to provide high prediction accuracy with a relatively small population of microarchitectural counters.

Linear regression functions of a small number of performance counters can be easily calculated on hardware, while non-linear models are more complex and more time consuming. Therefore, linear models are more suitable for online prediction purposes [18].

In general, regression techniques give the ability to calculate a function to predict a value of the dependent variable from a set of independent variables. Assuming a set of $x_1, x_2, x_3, \ldots, x_N$ independent variables and y the dependent variable, the classical linear regression model for y that we use in this analysis, is based on the Ordinary Least Squares (OLS) model. Specifically, it yields a set of weights β, one for each predictor variable x, and an error term e:

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + e_i$$

In this formula y_i is the i^{th} response value (in our case the V_{min} or the severity value), x_{ji} is the j^{th} microarchitectural counter (e.g. the L1 Cache Accesses) evaluated at the i^{th} observation, and e_i is the i^{th} statistical error. The goal of the regression analysis is to find the optimal values of the coefficients $\beta_1, \beta_2, \beta_3, \ldots, \beta_k$ so as to minimize the sum of the squares of the differences between the observed responses (values of the predicted variable) in the given dataset and those predicted by a linear function of a set of explanatory variables.

This analysis also provides a "coefficient of determination" (R^2) that indicates the proportion of the variance in the dependent variable that is predictable from the independent variables. The larger the values of R^2, the better fit the model provides, while the best fit exists when R^2 is equal to 1. The R^2 can be 0 when the model predicts the expected value disregarding the input features or even negative (because the model can be arbitrary worse). R^2 is an important indicator for linear regression analysis in order to quantify the accuracy of each model. However, to evaluate the accuracy of our prediction model for different test cases, we also use the Root Mean Square Error (RMSE) that represents the deviation between the predicted values and the observed values. The smaller the RMSE the more efficient the prediction model is.

Our analysis is based on four steps: (i) offline characterization (which was presented in Section 3), (ii) collection of all the performance counters provided by the X-Gene 2 system during nominal conditions, (iii) feature selection of the most important counters that mostly affect the prediction according to the test case (targeting either the V_{min} or the severity), and (iv) training and evaluation of the different test cases on which we implemented linear regression analysis. For feature selection and the linear regression model of our analysis we used the python libraries provided by [19]. As Figure 6 presents, in phase 1 we perform an extensive characterization, which exposes the regions of operation (Safe, Unsafe, Crash) and the severity values. In phase 2 we perform application profiling for all available performance counters. In phase 3, we train the predictor using the outputs from step 1 and 2, and in phase 4, we make the actual prediction evaluating the estimations using the test dataset. We further analyze the three steps (except for characterization, which is described previously) illustrating also the results of our

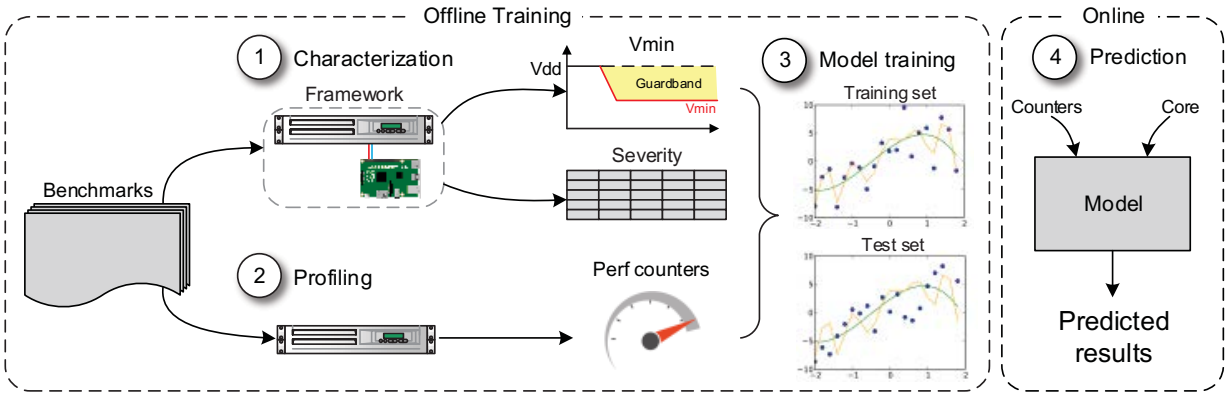

Figure 6: Overview of the prediction flow.

statistical analysis for the different test cases (targeting both the V_{min} and the severity).

4.1 Collection of all the Performance Counters

The X-Gene 2 provides 101 performance counters in total which report microarchitectural events of the entire system for individual cores, for the memory hierarchy (accesses and misses of all cache, TLB and page walks levels, unaligned accesses, prefetches, etc.), the pipeline (flushes, mispredictions, etc.), and the system (bus accesses, etc.). In our analysis, we used 26 SPEC CPU2006 benchmarks collecting the performance counters of the entire benchmarks using perf [20].

4.2 Selecting Counters for each Test Case

To reduce the population of performance counters used by our prediction model we implemented a feature selection technique called Recursive Feature Elimination (RFE) [19] for our statistical analysis concerning both the V_{min} and the severity values. Given an external estimator that assigns weights to features (e.g., a linear regression model) the goal of RFE is to select features by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features, and weights are assigned to each one of them. Then, features whose absolute weights are the smallest are pruned from the current set of features. This procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached. In our test cases, we eventually selected the 5 most efficient and representative events to predict the V_{min} and the severity values. The 5 most important features that were selected by RFE in our test cases are: (i) dispatched stalled cycles, (ii) exceptions taken, (iii) read data memory accesses, (iv) branch-target-buffer (BTB) mispredictions, and (v) conditional and indirect branches. Our model reports the impact of any architectural event that contributes to prediction, classified by its importance. We experimentally observe that the 5 aforementioned events provide the same accuracy as when we used more than 5 events, therefore no more are necessary.

4.3 Training and Evaluating the Test Cases

We analyze the results of our analysis for the three cases targeting both the V_{min} and the severity values for the most robust (Core 4) and the most sensitive core (Core 0) of the TTT chip. We present the three representative test cases of our study on the TTT chip, which are:

- 1st case: Predict V_{min} of the most sensitive core
- 2nd case: Predict severity of the most sensitive core
- 3rd case: Predict severity of the most robust core

In our analysis, we call *samples* all the information vectors that were used in our analysis and consist of the values of the dependent and independent variables used in our regression model. For all our experiments, we used the 80% of the population of the samples as the training set and the rest 20% as the test set for our prediction model. Finally, to evaluate the efficiency of our prediction model (apart from the R^2 and the RMSE) we used as baseline model the naïve prediction, which is the average of the target values (V_{min} or severity) of the samples of the training set.

4.3.1 1st case: Predict V_{min} of the most sensitive core: In our first case study, we evaluate the correlation of the performance counters with the V_{min} of the individual cores of the three chips. We discuss the results of our analysis in Core 0 that is the most sensitive core of the TTT chip. For our analysis, we used 40 samples for each core, which come from the full execution of 26 benchmarks from the SPEC CPU2006 suite with all of their input datasets (3 of them could not run correctly). Each sample consists of all the performance counters of each benchmark, while the target value of the model is the V_{min}. In general, the prediction model of the V_{min} for the individual cores gives us a good RMSE result (error equals 5mV or 0.51% of the nominal voltage), but the R^2 for that case is close to 0 which indicates that a small proportion of the variance in the dependent variable is predictable from the independent variables. Moreover, we observed that due to the narrow unsafe area observed (in Core 0 it is between 910mV and 885mV), the naïve prediction (using the average values of the training test) is equally efficient to predict the V_{min}.

4.3.2 2nd case: Predict severity of the most sensitive core: In the next two cases, we evaluate the efficiency of a linear regression model targeting the severity of an individual core as was defined in subsection 3.4.1. Firstly, we illustrate the results of our analysis for the most sensitive core of the TTT chip (Core 0). For our analysis, we used 100 samples from the unsafe region that were observed during the characterization phase. Each sample

corresponds to each reduction step of 5mV that was used during the characterization phase and consists of the microarchitectural counters running the benchmark in the nominal conditions and the voltage value of the characterization step. The target of our prediction model is the severity of Core 0 for a particular voltage value. Figure 7 presents the results of the prediction (blue line) and the test samples (black dots in the graph). The RMSE of the linear regression after the selection of the 5 most effective features with the RFE is 2.8 Severity units, while the RMSE of the naïve approach of using the average of the test dataset is 6.4 severity units indicating that our model is more efficient than both the baseline naïve approach for severity values and for the 1st case concerning the V_{min} point. Moreover, the R^2 for this case is *very high* 0.92 (very close to 1) that indicates that the linear model with the selected features is able to predict a large proportion of the variance in the dependent variable.

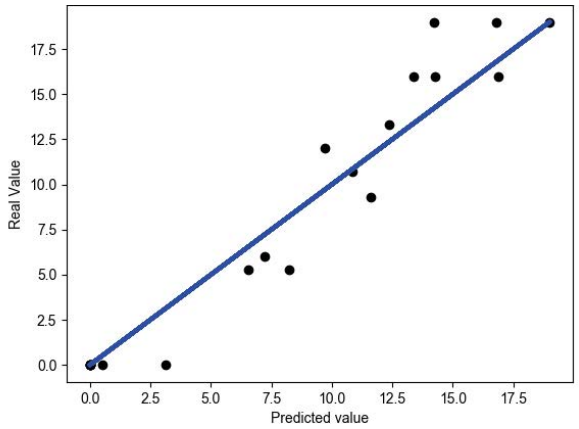

Figure 7: Severity prediction for most sensitive core (core 0).

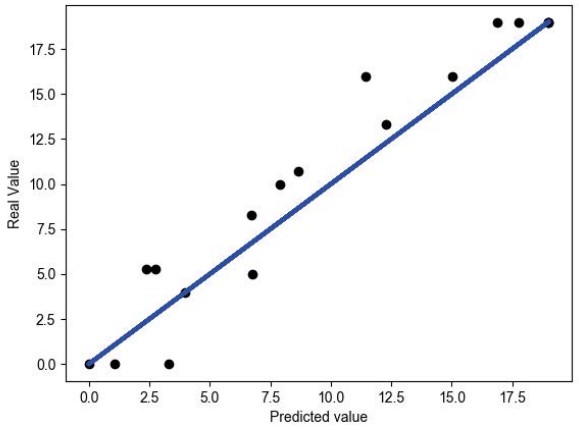

Figure 8: Severity prediction for most robust core (core 4).

4.3.3 3rd case: Predict severity of the most robust core: Finally, we present the results of our analysis for the most robust core of the TTT chip (Core 4). For our analysis, we used 90 samples as were presented in subsection 4.3.2, but now the target of our prediction model is the severity of Core 4 for a particular voltage value. Figure 8 presents the results of the prediction (blue line) and the test samples (black dots in the graph). The RMSE of the linear regression after the selection of the 5 most effective features with the RFE is 2.65 severity units, while the RMSE of the naïve approach of using the average of the test dataset is 6.9 severity units indicating that our model is more efficient than the baseline naïve approach. Moreover, the R^2 for this case is again *very high* 0.91 (very close to 1, which means the best fit) that indicates the efficiency of the linear model.

4.4 Undervolting Effects Mitigation

By combining our findings from the three test cases, it is clear that the prediction model using the severity values instead of a static V_{min} point is more efficient in predicting the safe V_{min} for each workload, as well as giving a flexibility to the predictor to be more aggressive due to the knowledge of the unsafe region. A static V_{min} point does not contain any information about the severity of operating at voltages below the safe V_{min}, but severity does so. Therefore, having knowledge about the severity below the safe V_{min} for each workload, the predictor can decide if it is possible to be more aggressive to set the voltage below the safe V_{min}, and thus, to save more power. We can also notice that the prediction based on the severity, not only is more efficient than the V_{min} point alone, but it also shows that it can fit effectively for each core, taking into account the process variation. The two different cases 2 and 3 (one for a sensitive core and one for a robust core) demonstrate that the linear regression model for severity values can be effective regardless the core-to-core variation (and consequently the chip-to-chip variation).

Depending on the actual characterization findings (V_{min} and severity) or the corresponding predicted values for a CPU core during undervolting, certain hardware-based or software-based mitigation approaches can be employed to maximize the energy savings while preserving the correctness of program execution.[1] The primary aspect that determines the most suitable approach is the *first observed (or predicted) effect* as undervolting goes down the voltage levels. We select the following behaviors using the severity function described in subsection 3.4.1 and used as the target function in the prediction. For each case we describe the behavior, discuss the severity function values and corresponding mitigation approaches.

Nothing abnormal (severity=0). The voltage range is predicted to be safe (above the V_{min} of a core); no mitigation action is required. System operation in this range is the most conservative option and no mitigation provision is needed. Energy-savings are the minimum.

[1] Note that our severity metric and prediction mechanism can be used above existing circuit-based techniques such as adaptive clocking. For instance, in the mechanism proposed in [38] adaptive-clocking can reduce the voltage at which SDCs occur. The frequency with which adaptation is deployed can be an input to our framework, thereby limiting the potential for performance degradation due to excessive deployment of adaptive-clocking induced frequency slowdown.

Corrected errors first (severity=1). This is a voltage range with predicted behavior as the one observed in [9, 10] for Intel's Itanium (not in our X-Gene 2 machine). In such a case, ECC hardware serves as a proxy for abnormal behavior due to undervolting but program operation is still correct. Significant energy savings can be obtained without any mitigation other than the ECC correction but going further down the voltage is risky.

SDCs alone (severity=4) or *with corrected and uncorrected errors* (severity=5-7). Voltage ranges with these predicted behaviors generate incorrect program outputs and require extra mitigation approaches. The characterization of our X-Gene 2 system shows that the first abnormal behavior generated by undervolting belongs here for the majority of benchmarks and corrected errors as observed in [9, 10] do not appear first alone in our system. In particular, the cases where SDCs appear alone (severity=4) are the worst ones since there is no indication about the malfunction of the system; these areas should be avoided. When an eventual SDC (output mismatch) is accompanied by corrected or uncorrected error notifications, recovery actions can be employed such as rollback to a previously stored check-point or program re-execution in safe voltage and frequency combinations. There are also many applications that can tolerate SDCs and benefit from the severity function. These applications are (1) approximate computing algorithms, (2) video streaming and other image and video processing, (3) security oriented applications such as jammer attacks detectors, etc. These applications are tolerant to faults, as they have minor impact on the returned output. For such applications, severity <=4 can be used for improving energy efficiency.

Application and system crashes with or without corrected and uncorrected errors (severity 8-19). Voltage levels with this predicted behavior (the result of massive hardware malfunction) are well beyond the limits of cores operation in undervolted conditions. Application or system unresponsiveness is systematic in these ranges and unless serious hardware re-design is employed these ranges are unusable.

5 ENERGY-PERFORMANCE TRADEOFFS

Since the linear regression analysis using the severity values seems very promising, comprehensively training the predictor with lots of data is necessary to guarantee the accurate prediction of any different program and dataset during the normal microprocessor operation. Further, one major challenge for the predictor is to be able to fit effectively on different cores and chips (due to process variation). As our analysis in the previous section shows, the linear prediction model is very efficient and accurate, by taking into account the process variation. However, process variation may lead to conservative energy savings while the microprocessor operates with real workloads. For that case, the predictor takes into account the different behaviors of each core in the microprocessor chip derived by the characterization. For instance, assume the TTT chip whose PMD 2 (cores 4 and 5) is the most robust and the PMD 0 (cores 0 and 1) is the most sensitive, for the majority of benchmarks. Consequently, the predictor not only sets the voltage according to the current workload (by monitoring the 5 representative performance counters), but it can also guide task scheduling so that tasks are assigned first to more robust cores to obtain higher power savings.

Note that the X-Gene 2 chip has one common power domain for all PMDs in the microprocessor, while the frequency granularity is per PMD (pair of cores). This means that, in a multicore execution, when for instance 4 workloads run on the system (two processes in PMD 0 and two in PMD 2), the predictor sets the voltage according to workload run on the most sensitive PMD (cores 0 and 1). Consider for example the *leslie3d* benchmark, as shown in Figure 4: the most robust PMD has safe V_{min} at 880mV, while the most sensitive PMD at 915mV in 2.4 GHz. By setting the voltage of the microprocessor chip at 915mV (for correct execution of all workloads), the measured energy savings will be 12.8%, while the most robust core could have 19.4%. On the other hand, considering again the TTT chip, by setting the frequency at 1.2 GHz, both robust and sensitive cores have safe V_{min} at 760mV, which means 69.9% energy savings, but with 50% performance loss.

The performance/energy trade-off though, can result in several energy saving steps. The predictor can monitor the architectural events separately for each core. According to the worst-case behavior of the core-benchmark pair, the predictor can decide what is the safe voltage for all the cores, which is practically the maximum among them. Therefore, depending on what program runs in any core, the predictor can recognize the core, in which the program can safely operate at the highest voltage. According to the predictor's decision the voltage can be set to the safest (*highest*) V_{min}.

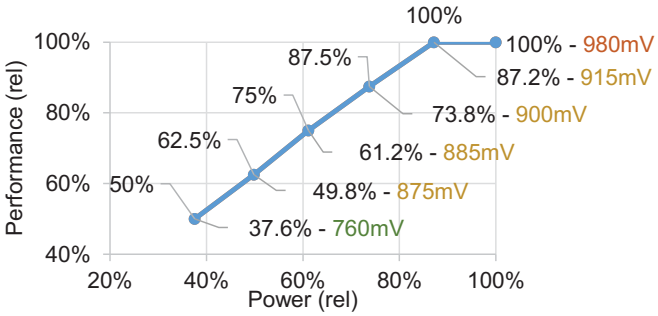

Figure 9: Tradeoffs for a workload of 8 benchmarks.

Figure 9 shows the potential savings for the case that 8 different benchmarks run simultaneously: *bwaves*, *cactusADM*, *dealII*, *gromacs*, *leslie3D*, *mcf*, *milc*, *namd*. By exploiting the predictor's results, 12.8% power savings can be obtained by adjusting the voltage to the TTT V_{min} without performance loss. Alternatively, the frequencies of the 2 weakest PMDs (0 and 1) can be reduced to 1.2 GHz (resulting in 25% performance loss) which will allow further reduction of the supply voltage to 885mV and energy savings up to 38.8%. Therefore, the predictor, apart from predicting the safe V_{min}, it can also assist task scheduling in conjunction to frequency scaling according to the current workload on the system to further improve energy efficiency.

6 DESIGN ENHANCEMENTS

Undervolting characterization studies such as the one we report in this paper can be used to provide hardware design recommendations for enhancements if the system (or its future revisions) is to be used in scaled voltage conditions for energy efficiency. There are some key hardware design guidelines that our analysis delivers for a system with similar behavior as the X-Gene 2 machine:

Stronger error protection. SECDEC ECC protection at the lower levels of the memory hierarchy does not provide enough protection at lower voltages. If (a) stronger ECC codes are employed [21, 22] and (b) more blocks are protected, SDC behavior with or without errors will have significant probability to be transformed to corrected errors behavior similarly to [9, 10]. Employing stronger ECC protection has been also reported in [23] for scaled voltage operation.

Hardware detectors. If stronger ECC protection is too costly other types of hardware support can be employed for voltage emergencies detection such as the skitter circuit [24 – 26] also cited in [17] or the monitoring circuits used in Power7+ designs [7].

Finer-grained voltage domains. Our characterization study shows that the coarse-grained voltage domains design of X-Gene 2 (a single voltage domain for all 8 cores) reduces the potential of energy savings since the voltage value of the domain is determined by its weakest core (the one with the higher V_{min}). If each PMD was designed to operate on a separate voltage domain (similarly to the independent frequency domains per PMD) more aggressive voltage scaling (and energy savings) would be have been possible.

Of course, all the above hardware design modifications have their own design complexity, area and performance implications which must be jointly considered with the potential of energy savings through undervolting.

7 RELATED WORK

Recently, the goal for improving microprocessors' energy efficiency, by reducing their supply voltage is a main concern of many scientific studies. For example, Ketkar et al. in [27] and Kim et al. in [28, 29] propose methods to maximize voltage droops in single core and multicore chips in order to investigate their worst-case behavior due to the generated voltage noise effects. Studies of Gupta et al. in [30] and Reddi et al. in [17] focus on the prediction of critical parts of benchmarks, in which large voltage noise glitches are likely to occur, leading to system malfunctions. In the same context, several studies either in the hardware or in the software level were presented to mitigate the effects of voltage noise [4, 24, 31 – 33] or to recover from them after their occurrence [34]. Gopireddy et al. in [35] presented a core that was designed for voltage scalability that can work in high-performance mode at nominal V_{dd} and in a very energy-efficient mode at low V_{dd}.

Apart from these studies that are mainly concentrated on the core and the voltage droops, Bacha et al. [9, 10] focus on the observation of the errors manifested on caches of a commercial Intel Itanium processor during the execution of benchmarks off-nominal voltage values. Papadimitriou et al. in [36] presented an experimental study that aims to identify the voltage margins in two different commercial x86-64 microprocessors; an ultra-low power and a high-end microprocessor. The authors in this work present the guardbands of these microprocessors, and compare them to the power and temperature savings, when they operate beyond nominal voltage conditions. Moreover, Wilkerson et al. [21], Chishti et al. [22] and Duwe et al. [23] propose several microarchitectural approaches to ensure the correct operation of caches in ultra-low voltage conditions. The characterization studies of commercial chips in off-nominal voltage conditions are limited [8 – 11, 37, 38] strengthening the purpose of the existence of our proposed framework that targets the APM X-Gene 2 micro-server. Similar characterization effort for emerging ARM-based enterprise server systems is sparse. Authors in [39 – 42] from ARM Research developed an electrical simulation framework for power-delivery analysis and used an on-chip voltage monitoring circuit to characterize supply voltage droops in a dual-core ARM Cortex-A57 cluster operating at 1.2 GHz. Regression analysis has been used in many performance and power studies [43 – 45], as well as in reliability estimation concerning soft errors [16, 18].

8 CONCLUSIONS

We presented a comprehensive characterization study of the X-Gene 2, a commercial 8-core 64-bit ARMv8 chip fabricated on 28nm provided by AppliedMicro (APM) when it operates in off-nominal voltage and frequency conditions. Our characterization revealed large voltage margins that can be translated into significant power savings and also large V_{min} variation among the 8 cores of the chip, among 3 different chips (a nominal rated and two sigma chips), and among different benchmarks. Moreover, the combination of our characterization results with a simple prediction scheme using a linear regression model that can guide task scheduling can lead to 19.4% energy savings without loss of performance, while with 25% performance loss we can achieve 38.8% energy savings in total.

ACKNOWLEDGMENT

This work is funded by the H2020 Framework Program of the European Union through the UniServer Project (Grant Agreement 688540) – http://www.uniserver2020.eu.

REFERENCES

[1] F. Salehuddin, I. Ahmad, F.A. Hamid, A. Zaharim, A. Maheran, A. Hamid, P. S. Menon, H. A. Elgomati, and B. Y. Majlis. 2012. Optimization of process parameter variation in 45nm p-channel MOSFET using L18 Orthogonal Array. In *Proceedings of IEEE International Conference on Semiconductor Electronic (ICSE '12).* Kuala Lumpur, Malaysia, 219-223. DOI: 10.1109/SMElec.2012.6417127

[2] W. Schemmert, and G. Zimmer. 1974. Threshold-voltage sensitivity of ion- implanted MOS transistors due to process variations. Electronics Letters, vol. 10, no. 9, pp. 151–152, May. DOI: 10.1049/el:19740115

[3] Norman James, Phillip Restle, Joshua Friedrich, Bill Huott, and Bradley McCredie. 2007. Comparison of split-versus connected-core supplies in the POWER6 microprocessor. In *Proceedings of the 2007 IEEE*

International Solid-State Circuits Conference (ISSCC '07). San Francisco, CA, USA, 298–604. DOI: 10.1109/ISSCC.2007.373412

[4] Vijay Janapa Reddi, Svilen Kanev, Wonyoung Kim, Simone Campanoni, Michael D. Smith, Gu-Yeon Wei, and David Brooks. 2010. Voltage Smoothing: Characterizing and Mitigating Voltage Noise in Production Processors via Software-Guided Thread Scheduling. In *Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-43)*. IEEE Computer Society, Washington, DC, USA, 77-88. DOI=http://dx.doi.org/10.1109/MICRO.2010.35

[5] Etienne Le Sueur and Gernot Heiser. 2010. Dynamic voltage and frequency scaling: the laws of diminishing returns. In *Proceedings of the 2010 international conference on Power aware computing and systems (HotPower'10)*. USENIX Association, Berkeley, CA, USA, 1-8.

[6] Dan Ernst, Nam Sung Kim, Shidhartha Das, Sanjay Pant, Rajeev Rao, Toan Pham, Conrad Ziesler, David Blaauw, Todd Austin, Krisztian Flautner, and Trevor Mudge. 2003. Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation. In *Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture (MICRO-36)*. IEEE Computer Society, Washington, DC, USA, 7-18.

[7] Yazhou Zu, Charles R. Lefurgy, Jingwen Leng, Matthew Halpern, Michael S. Floyd, and Vijay Janapa Reddi. 2015. Adaptive guardband scheduling to improve system-level efficiency of the POWER7+. In *Proceedings of the 48th International Symposium on Microarchitecture (MICRO-48)*. ACM, New York, NY, USA, 308-321. DOI: https://doi.org/10.1145/2830772.2830824

[8] Charles R. Lefurgy, Alan J. Drake, Michael S. Floyd, Malcolm S. Allen-Ware, Bishop Brock, Jose A. Tierno, and John B. Carter. 2011. Active management of timing guardband to save energy in POWER7. In *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-44)*. ACM, New York, NY, USA, 1-11. DOI=http://dx.doi.org/10.1145/2155620.2155622

[9] Anys Bacha and Radu Teodorescu. 2013. Dynamic reduction of voltage margins by leveraging on-chip ECC in Itanium II processors. In *Proceedings of the 40th Annual International Symposium on Computer Architecture (ISCA '13)*. ACM, New York, NY, USA, 297-307. DOI: http://dx.doi.org/10.1145/2485922.2485948

[10] Anys Bacha and Radu Teodorescu. 2014. Using ECC Feedback to Guide Voltage Speculation in Low-Voltage Processors. In *Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-47)*. IEEE Computer Society, Washington, DC, USA, 306-318. DOI: http://dx.doi.org/10.1109/MICRO.2014.54

[11] Jingwen Leng, Alper Buyuktosunoglu, Ramon Bertran, Pradip Bose, and Vijay Janapa Reddi. 2015. Safe limits on voltage reduction efficiency in GPUs: a direct measurement approach. In *Proceedings of the 48th International Symposium on Microarchitecture (MICRO-48)*. ACM, New York, NY, USA, 294-307. DOI: https://doi.org/10.1145/2830772.2830811

[12] The Linux Kernel Documentation (Parent Directory), Retrieved 2017 from https://www.kernel.org/doc/Documentation.

[13] George Papadimitriou, Manolis Kaliorakis, Athanasios Chatzidimitriou, Dimitris Gizopoulos, Greg Favor, Kumar Sankaran and Shidhartha Das. 2017. A System-Level Voltage/Frequency Scaling Characterization Framework for Multicore CPUs. In *13th IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE '17)*. Boston, MA, USA.

[14] John L. Henning. 2006. SPEC CPU2006 benchmark descriptions. SIGARCH Comput. Archit. News 34, 4 (September 2006), 1-17. DOI=http://dx.doi.org/10.1145/1186736.1186737

[15] Reid J. Riedlinger, Rohit Bhatia, Larry Biro, Bill Bowhill, Eric Fetzer, Paul Gronowski, and Tom Grutkowski. 2011. A 32nm 3.1 Billion Transistor 12-Wide-Issue Itanium® Processor for Mission-Critical Servers", In *Proceedings of the 2011 IEEE International Solid-State Circuits Conference (ISSCC '11)*. San Francisco, CA, USA, 84-86. DOI: 10.1109/ISSCC.2011.5746230

[16] Arijit Biswas, Niranjan Soundararajan, Shubhendu S. Mukherjee, and Sudhanva Gurumurthi. 2009. Quantized AVF: A means of capturing vulnerability variations over small windows of time. In *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE '09)*. Stanford University, CA, USA.

[17] Vijay Janapa Reddi, Meeta S. Gupta, Glenn Holloway, Gu-Yeon Wei, Michael D. Smith, and David Brooks. 2009. Voltage emergency prediction: Using signatures to reduce operating margins. In *Proceedings of the 15th International Conference on High-Performance Computer Architecture (HPCA '09)*, Raleigh, NC, USA 18–29. DOI: 10.1109/HPCA.2009.4798233

[18] Kristen R. Walcott, Greg Humphreys, and Sudhanva Gurumurthi. 2007. Dynamic prediction of architectural vulnerability from microarchitectural state. In *Proceedings of the 34th annual international symposium on Computer architecture (ISCA '07)*. ACM, New York, NY, USA, 516-527. DOI: https://doi.org/10.1145/1250662.1250726

[19] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Edouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. Machine Learning Research, vol. 12, pp. 2825-2830, October.

[20] Perf: Linux Profiling with Performance Counters. Retrieved 2017 from https://perf.wiki.kernel.org/index.php/Main_Page.

[21] Chris Wilkerson, Hongliang Gao, Alaa R. Alameldeen, Zeshan Chishti, Muhammad Khellah, and Shih-Lien Lu. 2008. Trading off Cache Capacity for Reliability to Enable Low Voltage Operation. In *Proceedings of the 35th Annual International Symposium on Computer Architecture (ISCA '08)*. IEEE Computer Society, Washington, DC, USA, 203-214. DOI=http://dx.doi.org/10.1109/ISCA.2008.22

[22] Zeshan Chishti, Alaa R. Alameldeen, Chris Wilkerson, Wei Wu, and Shih-Lien Lu. 2009. Improving cache lifetime reliability at ultra-low voltages. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-42)*. ACM, New York, NY, USA, 89-99. DOI=http://dx.doi.org/10.1145/1669112.1669126

[23] Henry Duwe, Xun Jian, Daniel Petrisko, and Rakesh Kumar. 2016. Rescuing uncorrectable fault patterns in on-chip memories through error pattern transformation. In *Proceedings of the 43rd International Symposium on Computer Architecture (ISCA '16)*. IEEE Press, Piscataway, NJ, USA, 634-644. DOI: https://doi.org/10.1109/ISCA.2016.61

[24] Meeta S. Gupta, Krishna K. Rangan, Michael D. Smith, Gu-Yeon Wei, and David Brooks. 2007. Towards a software approach to mitigate voltage emergencies. In *Proceedings of the 2007 ACM/IEEE International Symposium on Low Power Electronics and Design (ISPLED '07)*, Portland, OR, USA, 123-128. DOI: 10.1145/1283780.1283808

[25] R. Franch, P. Restle, N. James, W. Huott, J. Friedrich, R. Dixon, S. Weitzel, K. Van Goor, and G. Salem. 2008. On-chip timing uncertainty measurements on IBM microprocessors. In *Proceedings of the IEEE International Test Conference (ITC '08)*, Santa Clara, CA, USA, 1–7. DOI: 10.1109/TEST.2008.4700707

[26] Phillip J. Restle, Robert L. Franch, Norman K. James, William V. Huott, Timothy M. Skergan, Steven C. Wilson, Nicole S. Schwartz, Joachim G. Clabes. 2004. Timing uncertainty measurements on the power5 microprocessor. In *Proceedings of the 2004 IEEE International Solid-State Circuits Conference (ISSCC '04)*, San Francisco, CA, USA, 354–355. DOI: 10.1109/ISSCC.2004.1332740

[27] Mahesh Ketkar and Eli Chiprout. 2009. A microarchitecture-based framework for pre- and post-silicon power delivery analysis. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-42)*. ACM, New York, NY, USA, 179-188. DOI=http://dx.doi.org/10.1145/1669112.1669136

[28] Youngtaek Kim and Lizy Kurian John. 2011. Automated di/dt stressmark generation for microprocessor power delivery networks. In *Proceedings of the 17th IEEE/ACM international symposium on Low-power electronics and design (ISLPED '11)*. IEEE Press, Piscataway, NJ, USA, 253-258.

[29] Youngtaek Kim, Lizy Kurian John, Sanjay Pant, Srilatha Manne, Michael Schulte, W. Lloyd Bircher, and Madhu S. Sibi Govindan. 2012. AUDIT: Stress Testing the Automatic Way. In *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-45)*. IEEE Computer Society, Washington, DC, USA, 212-223. DOI=http://dx.doi.org/10.1109/MICRO.2012.28

[30] Meeta S. Gupta, Vijay Janapa Reddi, Glenn Holloway, Gu-Yeon Wei, and David M. Brooks. 2009. An event-guided approach to reducing voltage noise in processors. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '09)*. European Design and Automation Association, 3001 Leuven, Belgium, Belgium, 160-165.

[31] Russ Joseph, David Brooks, and Margaret Martonosi. 2003. Control techniques to eliminate voltage emergencies in high performance processors. In *Proceedings of the 2003 IEEE International Conference on High-Performance Computer Architecture (HPCA '03)*, Anaheim, CA, USA, 79–90. DOI: 10.1109/HPCA.2003.1183526

[32] Timothy N. Miller, Renji Thomas, Xiang Pan, and Radu Teodorescu. 2012. VRSync: characterizing and eliminating synchronization-induced voltage emergencies in many-core processors. In *Proceedings of the 39th Annual International Symposium on Computer Architecture (ISCA '12)*. IEEE Computer Society, Washington, DC, USA, 249-260.

[33] Michael D. Powell and T. N. Vijaykumar. 2003. Pipeline muffling and a priori current ramping: architectural techniques to reduce high-frequency inductive noise. In *Proceedings of the 2003 international symposium on Low power electronics and design (ISLPED '03)*. ACM, New York, NY, USA, 223-228. DOI=http://dx.doi.org/10.1145/871506.871562

[34] Meeta S. Gupta, Krishna K. Rangan, Michael D. Smith, Gu-Yeon Wei, and David Brooks. 2008. DeCoR: A Delayed Commit and Rollback mechanism for handling inductive noise in processors. In *Proceedings of the 2008 IEEE International Conference on High-Performance Computer Architecture (HPCA '08)*, Salt Lake City, UT, USA. DOI: 10.1109/HPCA.2008.4658654

[35] Bhargava Gopireddy, Choungki Song, Josep Torrellas, Nam Sung Kim, Aditya Agrawal, and Asit Mishra. 2016. ScalCore: Designing a core for voltage scalability. In *Proceedings of the 2016 IEEE International Conference on High-Performance Computer Architecture (HPCA '16)*, Barcelona, Spain, 681–693. DOI: 10.1109/HPCA.2016.7446104

[36] George Papadimitriou, Manolis Kaliorakis, Athanasios Chatzidimitriou, Charalampos Magdalinos, Dimitris Gizopoulos. 2017. Voltage Margins Identification on Commercial x86-64 Multicore Microprocessors. In *Proceedings of the 2017 IEEE 23rd International Symposium on On-Line Testing and Robust System Design (IOLTS '17)*. Thessaloniki, Greece, 51-56.

[37] Anys Bacha and Radu Teodorescu. 2015. Authenticache: harnessing cache ECC for system authentication. In *Proceedings of the 48th International Symposium on Microarchitecture (MICRO-48)*. ACM, New York, NY, USA, 128-140. DOI: https://doi.org/10.1145/2830772.2830814

[38] Sriram Sundaram, Sriram Samabmurthy, Michael Austin, Aaron Grenat, Michael Golden, Stephen Kosonocky, and Samuel Naffziger. 2016. Adaptive Voltage Frequency Scaling using Critical Path Accumulator implemented in 28nm CPU. In *Proceedings of the 2016 29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID '16)*, Kolkata, India, 565-566. DOI: 10.1109/VLSID.2016.106

[39] Paul N. Whatmough, Shidhartha Das, Zacharias Hadjilambrou, and David M. Bull. 2015. An all-digital power-delivery monitor for analysis of a 28nm dual-core ARM Cortex-A57 cluster. In *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC '15)*, San Francisco, CA, USA, 262-264. DOI: 10.1109/ISSCC.2015.7063026

[40] Paul N. Whatmough, Shidhartha Das, and David M. Bull. 2015. Analysis of adaptive clocking technique for resonant supply voltage noise mitigation. In *Proceedings of the 2015 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED '15)*, Rome, Italy, 128-133. DOI: 10.1109/ISLPED.2015.7273502

[41] Shidhartha Das, Paul Whatmough and David M. Bull. 2015. Modelling and characterization of the System-Level Power-Delivery Network for a Dual-Core ARM A57 Cluster in 28nm CMOS. In *Proceedings of the 2015 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED '15)*, Rome, Italy, 146-151. DOI: 10.1109/ISLPED.2015.7273505

[42] Paul Whatmough, Shidhartha Das and David M. Bull. 2017. Power Integrity Analysis of a 28 nm Dual-Core ARM Cortex-A57 Cluster Using an All-Digital Power Delivery Monitor. In *Journal of Solid-State Circuits (JSSC '17)*. vol. 52, no. 6, pp. 1643 – 1654, March. DOI: 10.1109/JSSC.2017.2669025

[43] Wenhao Jia, Kelly A. Shaw, and Margaret Martonosi. 2012. Stargazer: Automated regression-based GPU design space exploration. In *Proceedings of the 2012 IEEE International Symposium on Performance Analysis of Systems & Software (ISPASS '12)*. IEEE Computer Society, Washington, DC, USA, 2-13. DOI=10.1109/ISPASS.2012.6189201 http://dx.doi.org/10.1109/ISPASS.2012.6189201

[44] P. J. Joseph, Kapil Vaswani, Matthew J. Thazhuthaveetil. 2006. Construction and use of linear regression models for processor performance analysis. In *Proceedings of the 12th International Conference on High-Performance Computer Architecture (HPCA '06)*. Austin, TX, USA, 99–108. DOI: 10.1109/HPCA.2006.1598116

[45] Benjamin C. Lee and David M. Brooks. 2006. Accurate and efficient regression modeling for microarchitectural performance and power prediction. In *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems (ASPLOS XII)*. ACM, New York, NY, USA, 185-194. DOI: https://doi.org/10.1145/1168857.1168881