2016

PROCESSOR TEMPERATURE AND RELIABILITY ESTIMATION USING ACTIVITY COUNTERS

Mayank Chhablani
University of Massachusetts Amherst

PROCESSOR TEMPERATURE AND RELIABILITY ESTIMATION USING ACTIVITY COUNTERS

A Thesis Presented

by

MAYANK CHHABLANI

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL AND COMPUTER ENGINEERING

February 2016

Electrical and Computer Engineering

PROCESSOR TEMPERATURE AND RELIABILITY ESTIMATION USING ACTIVITY COUNTERS

A Thesis Presented

by

MAYANK CHHABLANI

Approved as to style and content by:

Israel Koren, Co-chair

C.M. Krishna, Co-chair

Sandip Kundu, Member

C. V Hollot, Department Chair
Electrical and Computer Engineering

To my parents.

ACKNOWLEDGMENTS

I take this opportunity to thank and recognize all the people who have helped me with this thesis. Firstly, I would like to express my heartfelt gratitude to my advisors Prof. Israel Koren and Prof. C. Mani Krishna. I thank them for their continuous support and guidance during the course of this research. Their unceasing support and thoughtful advice bolstered both the quality of this research as well as my spirit and my faith in the research process. Secondly, I extend my gratitude to Prof. Sandip Kundu for his valuable time and for being a part of my thesis committee and sharing his insightful advice and suggestions throughout this work.

I would also like to thank NSF for funding this project, and the Architecture and Real Time Systems (ARTS) Laboratory at the University of Massachusetts, Amherst for allowing me to work on it.

Finally, I am deeply grateful for the support and encouragement from my family and friends.

ABSTRACT

PROCESSOR TEMPERATURE AND RELIABILITY ESTIMATION USING ACTIVITY COUNTERS

FEBRUARY 2016

MAYANK CHHABLANI

B.E.E.C.E., UNIVERSITY OF RAJASTHAN, JAIPUR, INDIA

M.S.E.C.E., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Israel Koren and Professor C.M. Krishna

With the advent of technology scaling lifetime reliability is an emerging threat in high-performance and deadline-critical systems. High on-chip thermal gradients accelerates localised thermal elevations (hotspots) which increases the aging rate of the semiconductor devices. As a result, reliable operation of the processors has become a challenging task. Therefore, cost effective schemes for estimating temperature and reliability are crucial.

In this work we present a reliability estimation scheme that is based on a light-weight temperature estimation technique that monitors hardware events. Unlike previously proposed hardware counter-based approaches, our approach involves a linear-temporal-feedback estimator, taking into account the effects of thermal inertia. The proposed approach shows an average absolute error of <2.5 °C with standard deviation of <2 °C. Furthermore, if an on-chip temperature sensor is available, our modified technique can better tolerate ambient temperature variability.

We then present a counter-based technique to estimate the thermal accelerated aging factor (TAAF), which is an indicator of lifetime reliability. Results demonstrate that the estimation error is within $[-3, +5]$.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

Power densities have increased dramatically in recent years. High power density increases device temperature which can significantly elevate the hardware failure rate. In addition to high temperature, cycling between low and high temperature can increase failure rate. Thermal issues are prominent in cyber-physical systems which must often operate under harsh conditions. The Thermally Accelerated Age Factor (TAAF), which captures the effective age of a circuit as a multiple of its chronological age, usually rises roughly exponentially with temperature. Countermeasures, such as voltage and frequency scaling and throttling, exist for reducing thermal stress [18, 40]. However, such measures depend for their effectiveness on an effective way of monitoring on-chip temperature.

One design of thermal sensors uses simple ring oscillators or diode-based circuits [11]. The delay of a ring oscillator depends on the temperature providing an effective way for estimating temperature. Intel's Pentium 4 processor incorporates an on-die analog thermal diode, along with multiple sensing devices like resistance temperature detectors (RTDs), thermocouple and thermistors [31].

Moreover, Digital Thermal Sensors (DTS) have been incorporated into several Intel CPU families but software access is restricted to only core temperature resisters [5]. A DTS reports the difference between the current temperature and the maximum allowable junction temperature. Intel's Sandy-bridge and AMD's Quad-Core Opteron incorporate 12 and 38 thermal sensors, respectively [22, 12]. In these designs temperature sensing is done using a band-gap diode voltage comparator which converts voltage readings to temperature via a polynomial curve fit.

Direct measurement of temperature has some drawbacks: (1) The area of the sensor has to be large to provide high precision; (2) Sensors report the average temperature of the core which could miss localized hotspots; (3) The number of these sensors, their calibration and placement poses a serious concern as hotspots move over time, and (4) Sensor response is delayed due to thermal gradients along with process variations. Considering these issues with thermal sensors we need an alternative or supplement for thermal sensing; on-chip performance counters provide such an alternative.

Performance counters are available in today's high-end microprocessors for debugging and performance characterization and can easily be adapted to be used in thermal monitoring. Being programmable, these counters are also used to monitor certain events and activity levels such as L1-cache hits/ misses, functional unit access and branch-mispredictions allowing localized thermal sensing at a reasonably high accuracy. The number of counter registers also differs greatly between different micro-architectures. In addition to programmable counters, some processors also support fixed-function counters which provide limited programmability (i.e. they always count the same event, or they cannot be disabled).

Each performance counter is associated with a counter configurable control register (CCCR) and the event selection control registers (ESCRs) determine which event is to be counted [7]. The number of events captured by these event counters varies across processor families and their implementation. Also, there are limitations on how many events can be simultaneously measured. For example, AMD Athlon64, Opteron, and Phenom processors provide four performance counters to measure the hardware events experienced by application programs and system software [13]. Similarly, Intels SandyBridge has 3 fixed counters, 4 general-purpose counters and 4 Running Average Power Limit (RAPL) counters [36].

The RAPL energy counters monitor maximum average power. As a result, these counters may not be able to identify the localised thermal elevation events pertaining to specific block(s). Therefore, we have chosen performance counters to estimate the temperature of the hottest blocks.

1.1 Research Objective

Cyber-physical systems (CPS) are often used in life-critical applications, where the rate of fatal failures has to be kept very low. Therefore, fault-tolerance is crucial in CPS. However, the traditional approaches to fault-tolerance rely on a high level of redundancy which raises the computational load, thereby imposing a high thermal stress on the hardware. This, in turn, impacts the reliability and quality of control in CPS. As temperature (or thermal fluctuation) is strongly correlated to the aging rate of the processors, there is an urgent need for the early detection of localized thermal hotspots.

Since the maximum aging rate is dictated by the localised hotspots, our aim is to devise an estimation technique for the temperature and reliability at full-core level. The developed estimator(s) needs to be fast and robust, hence our objective is to devise such an estimator using a limited set of architectural performance counters.

1.2 Thesis Organization

Chapter 2 presents a literature survey, describing prior contributions in estimation of power and temperature. In Chapter 3 we present our experimental set-up, focusing on our estimation methodology and approach. Chapter 4 contains our simulation results, indicating the accuracy and robustness of our new approach for thermal and reliability estimation. A discussion and directions for possible extensions make up Chapter 5.

CHAPTER 2

REVIEW OF PRIOR WORK

In this chapter, we summarize some recent approaches to estimating power consumption and temperature.

2.1 Monitoring Power Consumption

Various authors have studied performance counters or architectural events in power-monitoring models. In [33], Singh *et al.* developed a linear model for power, on a per-unit basis. Their methodology uses a subset of performance counters, based on their correlation with power, and performs linear regression involving an ordinary least square (OLS) estimator. Policies were suggested to take corrective measures when the power envelope is breached. In [20], Isci and Martonosi proposed an online power estimation and synchronization model. Per-unit power measurement was done by sampling performance counters at fine-grain cycle granularity. In [30], Rodrigues *et al.* have shown that three performance counters can be used to estimate the dynamic power of processors with 95% accuracy.

Although power-aware-techniques can gauge the potential breach in the power budget/envelope, these techniques never account for the thermal impact of localized power density and hence the localized hotspots which degrade the performance and life span of a processor. Various thermal-aware computing techniques are discussed in [21], some of which are presented in the following section.

2.2 Thermal-Aware-Techniques

In [19] the authors describe a detailed thermal model, HotSpot, which represents the architectural blocks as an equivalent network of thermal resistances and capacitances with the power consumed by each functional unit acting as a current source. The temperature difference between two points is analogous to voltage and the resulting heat flow is analogous to current flow. Thermal capacitance measures the amount of heat required to raise the temperature by one unit; thermal resistance measures the amount of heat flow resulting from a unit difference in temperature.

Hotspot has three heat flow models. The lateral model expresses the flow of heat between a sub-unit and its neighbors. Two vertical models are provided to take into account the vertical effluxion of heat to the heat spreader, the heat sink and the on-chip interconnect and substrate. The Hotspot framework has been extensively used for thermal characterization, modelling and impact of temperature variations.

2.2.1 Online-Thermal Modelling

2.2.1.1 Using Sensors: Vigilance Over Thermal Barrier

Many researchers have assumed the existence of thermal sensors for monitoring core temperature [38]. Coskun *et al.* suggested a proactive temperature measurement technique using an Autoregressive Moving Average (ARMA) technique [9]. In this technique, a temperature trace of the current workload is collected from thermal sensors. This trace is sent to a predictor at run-time so as to monitor the change in thermal characteristics due to changes in the workload. Validation of this model, which takes place at run-time, incurs overhead because it requires computation of differential equations. Moreover, thermal sensors come with a few limitations: they may be too expensive and their placement is not trivial as hotspots move around during execution [24].

2.2.1.2 Using Hotspot: An Alternative To Thermal Sensors

Bao *et al.* proposed online thermal and power estimation using an iterative process that assumes an initial temperature estimate for power calculations [4]. They employed Hotspot for thermal analysis and used an iterative process for temperature estimation.

Lee and Skadron augmented the Hotspot model to monitor micro-architectural events using performance counters [24]. They employed the power model from [20] and integrated it with Hotspot in real-time. They have also described the variation of the gradient of maximum temperature across functional units.

Merkel and Bellosa suggested a hybrid approach using task activity vectors, where they sampled the utilization from performance counters every millisecond and estimated power and then, they used the Hotspot model to predict the temperature so as to avoid localized heating (hotspot) [26]. However, because the Hotspot model requires solving differential equations, it turned out to impose high overhead.

2.2.2 Offline-Modelling: Using Micro-architectural Events/Performance Counters

Reading temperature data from thermal sensors or incorporating Hotspot [19] or Temptor [16] at run-time seems to be a practical direction for thermal estimations; however, these methods have the aforementioned drawbacks. Chung and Skadron proposed a fine-grain localized temperature estimation technique using on-chip events [8][7]. They showed that by sampling performance counters at fine granularity and thereby using regression analysis, the temperature trace can be estimated. They employed similar metrics to those of [20] and used Hotspot to estimate the reference temperature data. Some of the limitations of this approach are that a two-step estimation process was used to get the reference temperature, i.e., first the power was estimated from the performance counters and then the temperature was estimated using Hotspot, which may have incurred some error in the reference temperature

itself. Moreover, their linear expression was accompanied by a large offset, which may result in overestimation when there is no activity.

While focusing on the Integer Register File, they indicated that a multi-valued regression based on the current and previous Integer Instructions Per Cycle (IIPC), sometimes results in reduced accuracy. Moreover, in [8], Chung and Skadron noted that the previous thermal traces does not contribute towards estimating the patterns of thermal fluctuations at run-time. Lee *et al.* predicted the localized temperature of a target functional unit, by using performance counters and performed DVFS, with the help of linear regression analysis [23].

While all the aforementioned research focused on monitoring the temperature of specific functional units, Upton and Hazelwood argued that linear regression is not a suitable choice for modelling full-core temperature based on performance counters and instruction stream (i.e., instruction category) [37]. They also showed that in some cases (arithmetic operations) the average error was significantly reduced, but because averaging out arithmetically can be skewed by the outliers, one may not be able to deal with a wide range of errors and hence, it can result in a misleading indicator. Moreover, [37] also indicated that the inclusion of temperature history drives the estimation in the opposite direction, i.e., temperature history restricts the thermal estimation as it is weighted by large factor in the estimation expression, hence, leading to more errors.

On the other hand, in [34], Kundu *et al.* presented a wavelet-based canonical spatio-temporal heat dissipation model for program traces. They used an integer linear programming formulation to rearrange program phases so as to maximize the temperature difference between a given pair of adjacent locations on the IC. This helps in incorporating appropriate performance and reliability guard bands within the design based on temperature estimate and also helps in selecting appropriate design packaging. They observed that the current temperature is determined not only by the current activity in that region, but also by the past activities in the surrounding regions. This indicates that previous temperature history does have an impact on the current temperature.

In Chapter 3, we show that the accumulation of thermal history over time has a significant contribution to thermal estimation. We also present a novel estimator switching approach in conjunction with temperature history which increases the estimation accuracy. The higher accuracy enables an effective back-tracking of TAAF, which we will be using as a proxy for lifetime-reliability. This enables efficient reliability-aware design and decisions [40].

CHAPTER 3

PROPOSED ESTIMATION SCHEME

In this chapter, we describe the entire simulation framework, specifically focusing on two approaches: impact of temperature history and novel estimator switching (expressions) using performance counters.

To facilitate the sampling of micro-architectural events and hence to gauge thermal behavior, we need a fully integrated performance, power and thermal model of the entire microprocessor. This framework will provide the activity, power and thermal traces which will form the basis for estimating temperature and reliability of the core. Architectural simulators can capture these events/activities depending on the workload characteristics, that determine activity and energy/power consumption in various functional units.

Thermal events have a longer time-constant than architecture-level events. We will use Hotspot for estimating the temperature, which will be used as a reference. Once the thermal behavior of various benchmarks are collected, we use machine learning approaches for selecting a limited set of performance counters and for generating estimator(s) for thermal estimation . One such approach is Correlation-based Feature Subset Selection, which can identify a representative set of on-chip events [15]. A second approach, the Time-Series Forecasting Model uses linear regression along with successive selection of potential performance counters for estimating temperature and reliability [17].

3.1 Experimental Environment

Figure 3.1 shows our simulation framework. Evaluation of system behavior at cycle-level, is done by using gem5 [6]. We obtain the activity data by reading performance counters at

cycle granularity for 20 million cycles (10ms) for the Alpha 21364 processor core operating at 2GHz, which is used in our experiments as a baseline core [29]; assuming a 90 nm technology. We have also validated our estimation methodology on another processor core, AMD Athlon64, which is scaled to 65 nm technology. A detailed description is provided below.

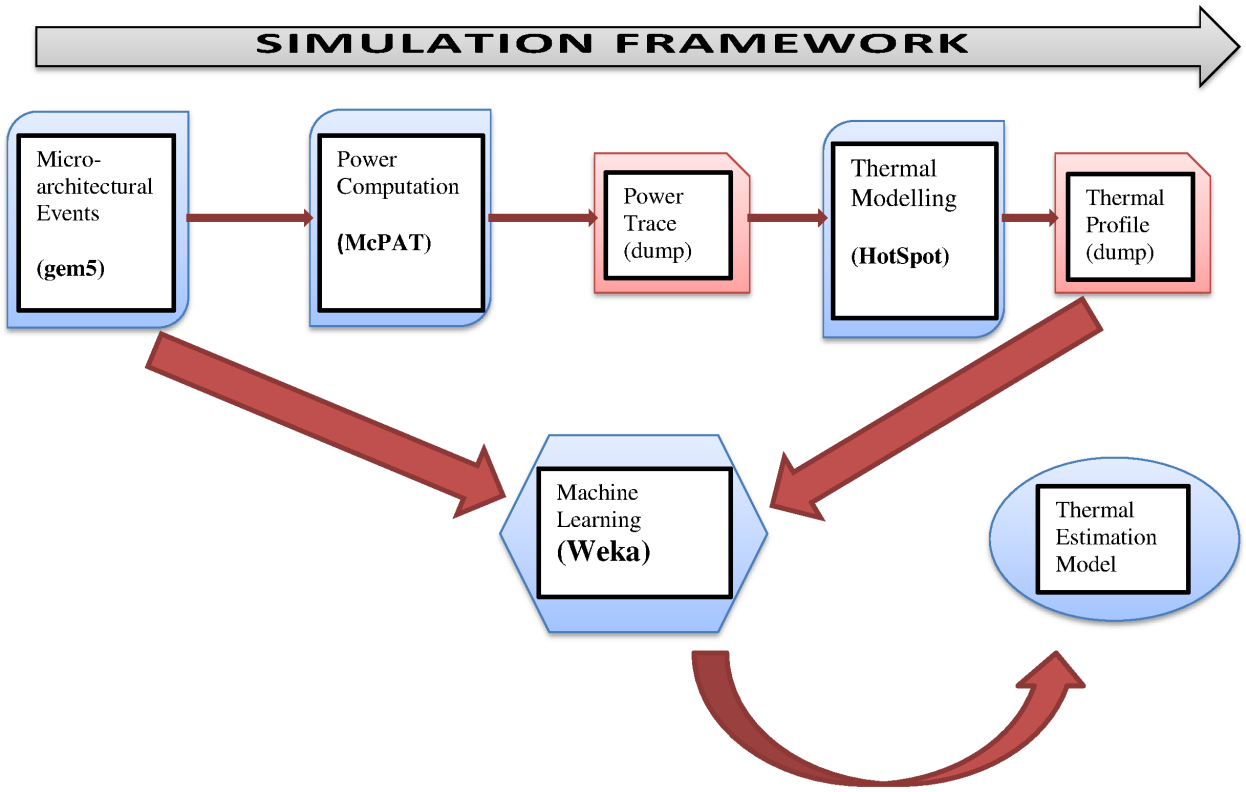


Figure 3.1: Simulation Framework

We have integrated gem5 with the power modelling tool, McPAT [25], that computes the power consumption for each functional unit. The execution traces obtained from the performance counters are dynamically fed to McPAT. We have taken into account both dynamic (caused by switching) and static (caused by leakage) power consumption. One advantage of using McPAT is that one can get the area specifications of individual functional units, which is helpful in thermal modelling.

Power traces for each unit (from McPAT), in conjunction with chip and packaging specifications drive the thermal model of HotSpot [19]. Table 3.1 summarizes the modified

configuration parameters[1] for HotSpot. The temperature model requires specification of the processor floorplan; we extracted the area specifications from McPAT and used hotfloorplan[2] for this purpose [19].

Table 3.1: HotSpot Configuration Parameters.

HotSpot Parameter	**Value**
Chip Thickness	0.15 mm
Core Area	241.883 mm^2
Convection Capacitance	140.4 $J\,K^{-1}$
Convection Resistance	0.7 $K\,W^{-1}$
Heat Sink Side	0.0526 m
Heat Spreader Side	0.026 m
Substrate Side	0.02 m
Ambient Temperature	45 °C

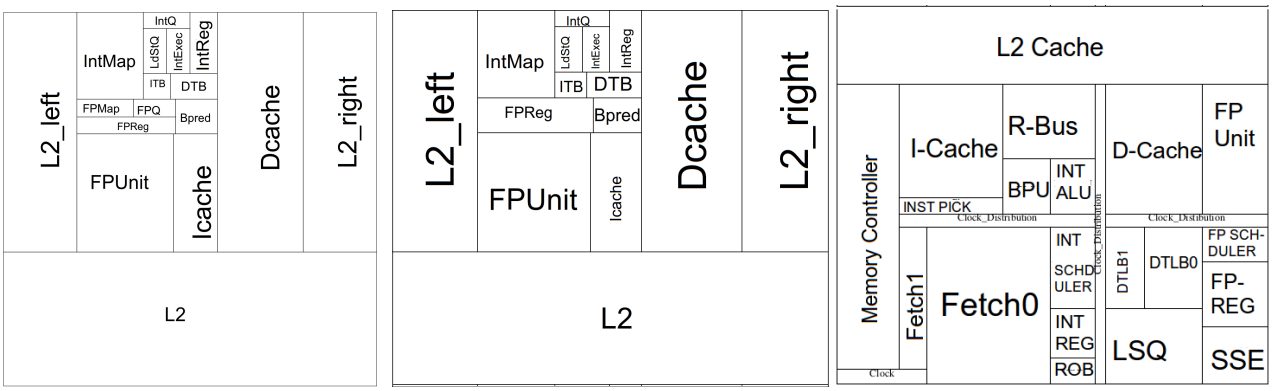
Since there is a limit on the number of events that can be monitored concurrently, we group highly correlated units within groups. We then focus on each such group. For example, we have observed that the floating point units such as Floating-Point Queue (FPQ), Floating-Point Map (FPMap) and Floating-Point Register (FPReg) can be grouped together because they show very similar thermal behavior and hence this thermal behavior can be estimated with the same set of performance counters. Therefore, we have merged FPQ, FPMap and FPReg unit into one single unit, thus obtaining a simplified floorplan of the Alpha 21264 as shown in Figure 3.2. While generating the simplified floorplan, using hotfloorplan [19], we aggregated the power of those units which were merged to "FPReg Unit" in the new floorplan.

3.1.1 Proposed Technique using Machine Learning Approaches

Once we generate the full thermal trace, we use an automated script to merge thermal traces with activity traces (performance counters), which in turn trigger Weka [17], a software

[1]We set the Hotspot's [41] parameters to fit the thermal characteristics of a single core die.

[2]HotFloorplan is a microarchitecture level thermal-aware floorplanning tool which makes use of simulated annealing algorithm for slicing floorplans.



(a) Original Alpha21364 Floorplan (b) Simplified Alpha21364 Floorplan (c) AMD Athlon64 Floorplan

Figure 3.2: Alpha21364 and AMD-Athlon64 Floorplan

tool consisting of a collection of machine learning techniques. Weka provides us with an estimation model which is then used to gauge the accuracy of estimation for different training and testing sets of benchmarks.

We use the following two approaches for thermal estimation: Correlation-based Feature Subset Selection (CfsSubsetEval) and Time-Series Forecasting Model. Both of these approaches are available in Weka.

3.1.1.1 Correlation-based Feature Subset Selection

Our goal is to identify a minimal set of performance counters for the temperature estimation of processor blocks. Clearly, we want those performance counters that are highly correlated with actual thermal behavior, yet largely uncorrelated with each other. In Machine Learning, identifying such a representative set (of features) is known as *Attribute/ Feature Selection*. Correlation-based Feature Subset Selection (CfsSubsetEval), considers the predictive ability of each feature (in our case, performance counters) amongst the set of features that are highly correlated with the temperature of the unit under consideration while having low inter-correlation amongst themselves [15]. In order to explore the subset-space we use the *Best-Fit* search algorithm along with *CfsSubsetEval*. Best-Fit searches the

space by greedy hill-climbing, augmented with backtracking [17]. The role of backtracking is to incrementally add performance counters which are highly correlated with temperature and abandon those which are correlated amongst each other. In this way we can explore all the performance counters but choose the representative ones, meeting our goal.

Also, we need to generate an estimator which can estimate the thermal variations of unknown working set/workload, which were not incorporated in the training set. Therefore, there is a need to have a good level of curve fitting to the data set (i.e., thermal trace), which can be achieved by one of the two methods *Random subsampling* and *k-fold cross validation*. In k-fold cross-validation, the data is randomly split into k mutually exclusive subsets (folds) of approximately equal size and the CfsSubsetEval algorithm is applied on training set and tested k times; each time it is trained on one of the k folds and tested using the remaining k-1 folds. In our case k=10. We used a k-fold cross validation scheme over random subsampling because in the latter case, test sets are not independently drawn with respect to the underlying distribution (i.e., thermal traces). After this process, we get the minimal set of performance counters, which are then fed to the Time-Series Forecasting Model in *Weka*, for developing an expression for thermal and reliability (or, TAAF) estimation.

3.1.1.2 Time-Series Forecasting Model

Due to thermal inertia, temperature changes relatively slowly with time. There is a lag between a change in power dissipation and temperature. Therefore, historical traces of temperature along with performance counters, play an important role in thermal estimation. Weka provides a *Time Series Forecasting Configuration* in which, a time-dependent series of observable variables (in our case, estimated temperature trace history and on-chip events) can lead to the development of an estimation expression using statistical technique of regression. We have chosen linear regression as the base learner for the estimation technique.

Weka's time series framework follows a machine learning approach to model aforementioned time-series. It encodes input activity data with time dependency via additional input

fields, representing in our case, historical thermal fluctuations. This process is called *flattening*. Once we have flattened our observable input variables, i.e., performance counters and temperature history, we can apply multiple linear regression (MLR) with the M5 Descriptor selection method, with the objective of minimizing the sum of squared residuals. The M5 Descriptor selection method is a process by which the features (performance counters, in our case) with the smallest regression coefficients are stepwise removed from the model until no reduction is observed in the average estimation error as given by Akaike Information Criteria[3] [3].

After applying MLR with the M5 Descriptor selection method, the following expression for estimation is generated by replacing $T^u_{actual}(n)$ with $T^u_{est}(n)$, which is the estimated thermal value at quantum n, with the initial condition $T^u_{est}(0) = T_{amb}$.

$$T^u_{actual}(n) = \sum_i \alpha_i C_i(n) + \beta T^u_{actual}(n-1) + \delta \tag{3.1}$$

Here, $T^u_{actual}(n)$ is the temperature of unit u, $C_i(n)$ is the ith performance counter, n is the sampling instance, $T^u_{actual}(n-1)$ is the temperature in the previous time step, T_{amb} is the ambient temperature, α and β are the empirical weight factors, representing the impact of current activity and thermal historical fluctuations on the present temperature reading, respectively, and δ is the offset of the estimation expression.

One of the key aspects of this linear temporal feedback model is that any temporal transient during the estimation due to application phase change will decay quickly, leading to convergence of estimated temperature with minimum error.

3.1.2 Estimation Methodology

There are several issues that must be considered when employing profiling techniques based on application characteristics. These include selection of representative sets/benchmarks,

[3]Akaike Criteria (AIC) not only rewards goodness of fit, but also includes a penalty that is an increasing function of the number of estimated parameters. The penalty discourages over-fitting and a smaller AIC is preferred.

the rate at which core-activity needs to be sampled and its effect on temperature peaks, the thermal history window and the threshold selection for our novel technique of estimator switching and its impact on estimation accuracy. We discuss these issues in detail below.

3.1.2.1 Selection of Benchmarks

Table 3.2 shows the benchmarks that we have chosen as representative subsets from the MiBench[14] and SPEC Benchmark suites [1] [2]. These benchmarks were chosen not only based on instructions type (i.e., integer-intensive and floating-point intensive) but also on the basis of thermal characteristics (i.e., slow & fast changing, gradually increasing thermal fluctuations). We have tested our estimator on individual suites and mixed suites, along with different training and testing sets.

Table 3.2: Set of Representative Benchmarks

Suite	**Benchmarks**
MiBench	adpcm, basicmath, bitcount, blowfish (encode & decode), crc, dijkstra, fft, fft-inverse gsm (encode & decode), jpeg (encode & decode), lame, patricia, qsort, rijndael-decode, sha, susan, typeset
SPEC2006	astar, bwaves, bzip2, calculix, dealII, gcc, h264ref, hmmer, mcf, namd, soplex, wrf
SPEC2000	ammp, applu, art, equake, mesa, mgrid

To model periodic tasks, we have employed these benchmarks with repetition: each of these benchmarks were executed for 4 billion cycles, after fast forwarding 1 billion cycles.

3.1.2.2 Exploration of Performance Counters

We examined the following performance counters which can be grouped as follows. The *CfsSubsetEval* method, as described in Section 3.1.1.1, is applied to select a limited subset of performance counters for thermal estimation. All the performance counter values were normalized w.r.t. sampling rate, i.e., events per cycle.

- **Instructions/Micro-Operations Per Cycle (IPC)**: IPC can be important to monitor because the number of instruction per cycle has a direct correlation with temperature due to unit's activity. In case of AMD Athlon, we focused on micro-ops per cycles instead of X86 complex instructions.

- **Functional Unit Access (FUA)**: The accesses to a particular unit plays an important role in projecting the thermal fluctuations in that unit. We explored the following performance counters related to FUA: IntRegAccess (IRA), FPRegAccess (FPRA), IntMapAccess (IMA), FPMapAccess (FPMA), IntQAccess (IQA), IntExecAccess (IEA), FPUnitAccess (FPUA), BPredAccess (BPUA), IssueRate.

- **Dispatch Stalls (D_Stalls)**: Here, dispatch stalls include stalls due to re-order buffer (ROB), load store queues (LSQ), reservation stations (RS), register map and register alias table (RAT). This counter has negative correlation with temperature, because while experiencing dependencies, activity in a unit decreases, thereby resulting in a gradual fall in temperature.

- **Hits/ Miss Counters**: The following counters can be grouped under this category: L2Misses (L2m), L1Hits (L1h), L1Misses (L1m), L2Hits (L2h). These counters also have a significant impact on power consumption. Hence, we have considered these as secondary counters while estimating thermal fluctuations.

- **Fetch and Speculative Counters**: Many instructions are speculatively executed in pipeline and may need to be flushed due to execution on false path. Therefore, flushing pipeline and execution of these instructions plays a significant role in thermal estimation. The counters explored under this category are: Number of Floating Point Instructions (NFP), Number of Fetched Instructions (Fetch_Insts), Branch Correctly Predicted (BCP), Branch Mis-predictions (BMP), Load-Store Instructions (LSInsts) and Branch Instructions (BrInsts).

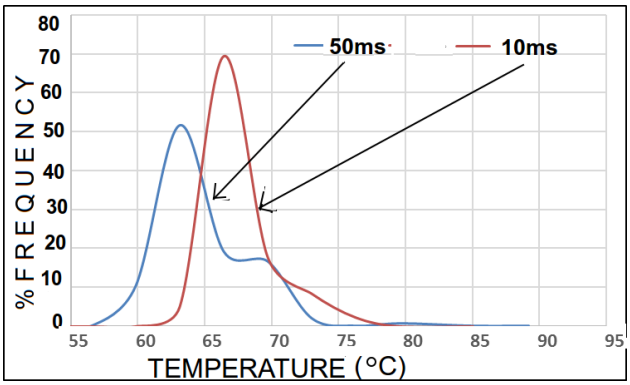


Figure 3.3: Sampling Window: Sampling at rate of (a) 10ms and (b) 50ms

3.1.2.3 Effect of Sampling Window

The sampling window is a key parameter which should be determined while profiling for thermal characteristics. An important question is how frequently should performance counters be sampled. Sampling too frequently may not produce a significant change in temperature and may unnecessarily interrupt the execution. On the other hand, sampling too infrequently may result in missing temperature peaks during intensive activities and may degrade our thermal management of the processor.

Experiments were carried out on the subset of benchmarks in Table 3.2 to determine the sampling frequency. Figure 3.3 shows the impact of sampling window on thermal characteristics of a set of benchmarks from Mibench and SPEC suites viz. *bitcount, namd, soplex, calculix* and *bwaves*.

Since the thermal characteristic remained essentially unchanged for sampling rates of $1M, 10M$ and $20M$ cycles, we show only the result for a sampling rate of $20M$ cycles to compare with that of sampling at $100M cycles$ (or, $50ms$). It is evident from Figure 3.3 that at $50ms$ sampling intervals, we are missing several high temperature peaks. This is also justified by our processor's thermal time constant which is $20ms$. Based on such data, we selected a sampling window of 20 million cycles, which is equivalent to a sampling interval of $10ms$ on a 2GHz processor. A sampling window of $10ms$ provides a natural opportunity for

software to read on-chip events [7] as it is the sampling granularity of commercial operating systems.

Moreover, for linear estimation technique to work sampling interval should be constant. Therefore, throughout our experiments we have sampled performance counters at a constant rate of $10ms$.

3.1.2.4 Threshold Detection for History-Based Dual Estimator

This section focuses on an approach to temperature estimation which involves dynamically selecting the appropriate estimator at run-time. As indicated in Equation 3.1, we have incorporated temperature history along with the current activity of the unit under consideration; this forms the basis of our approach.

As the name suggests, "**H**istory-Based **D**ual **E**stimator" is a technique wherein we switch from one expression to other at run-time based on the current activity of the unit; we have chosen Integer Register Access Per Cycle (IRA) and Number of Floating Point Instructions Per Cycle (NFP) as the base activity (b_act) for determining the threshold at which estimators will switch for Integer-Register File and Floating-Point Register File, respectively. In case of Scheduler (Integer/ Floating Point), we choose IssueRate as the base activity. Experiments show that these performance counters show a >0.97 correlation with temperature, and this correlation does not vary across technology node and architecture.

The two expressions are generated offline and are based on low and high base activity of the unit, namely, High Activity Estimator (*ha_est*) and Low Activity Estimator (*la_est*), as shown below.

$$T^u_{est}(n) = \begin{cases} \sum_i \alpha_i * C_i(n) + \beta * T^u_{est}(n-1) + \delta & \textit{if } b\_act \leq \tau; \\ \sum_j \theta_j * C_j(n) + \mu * T^u_{est}(n-1) + \gamma & \textit{otherwise} \end{cases} \tag{3.2}$$

where the first expression corresponds to *la_est*, the second to *ha_est* and τ represents the switching threshold. Also, the subset of performance counters for *la_est* may be different from that used by *ha_est*.

The choice of τ for these expressions is crucial because if this threshold is not chosen properly, estimators will be generated from a skewed thermal distribution and may result in an inaccurate estimation. Moreover, this threshold needs to be determined for each unit under consideration because of different base activity, (b_act), that needs to be taken into account. One obvious approach is to select τ to be the mean of the base activity. However, such a statistic can be misleading due to the possibility of large outliers. We have therefore chosen the median as the switching point for the estimator. There are exceptions to this choice. For floating point units, experimental results have shown that selecting τ to 10% of the base activity provides lower error.

Clearly, the full trace of activity is not available in practice during runtime. Instead, we have to select these parameters based on an offline study of a training set. Note that in a cyber-physical system, the task set is known in advance (even if the rate of invocation of individual tasks and the actual input data values are not).

CHAPTER 4

SIMULATION RESULTS

4.1 Temperature Estimation

The integer scheduler and integer register file are often two of the hottest units in the core [39, 27], resulting in a hotspot of up to 92.4 °C, unless dynamic thermal management (DTM) is provided. We therefore, focus on these units for our illustration here.

4.1.1 IntReg Unit Estimation

The following sections presents the integer register unit's estimation expression for two different benchmark families.

4.1.1.1 IntReg Unit Estimation: MiBench Suites

Table 4.1 shows the set of benchmarks used and expressions for Single and Dual Estimators which were generated from this experiment. Figure 4.1 shows the estimation error when employing the single estimator. This estimator takes into account both previous estimated temperature along with the current activity (i.e., intensity of current accesses to the IntReg Unit). An error in the range of -3.2°C to +2°C is observed; the question is whether using two estimators would improve the estimation accuracy.

Figure 4.1 answers this question: The dual estimator is more accurate.

Earlier, we described why the median is likely to be a better statistic than the mean in selecting a threshold for switching between estimators. The relative performance of these two statistics is evaluated in Figure 4.2. Using τ as the mean of the entire IRA (Integer Register Access Per Cycle) results in a skewed distribution and yields an underestimation of approx. 2°C relative to using the median of IRA for τ, for 60% of the samples.

Workloads	Dual Estimator (τ=Median=3.769)		Single Estimator
	la_est	*ha_est*	
bitcount, susan, jpeg-decode, lame, patricia, dijkstra, rijndael- encode, sha, adpcm, crc, fft, fft- inverse, gsm-encode	$T_{est}^{IntReg}(n) = 3.5634 * IRA + 0.1778 * T_{est}^{IntReg}(n-1) + 52.6467$	$T_{est}^{IntReg}(n) = 2.7008 * IRA + 0.4628 * T_{est}^{IntReg}(n-1) + 32.7468$	$T_{est}^{IntReg}(n) = 3.5496 * IRA + 0.2007 * T_{est}^{IntReg}(n-1) + 50.9166$

Table 4.1: [Alpha 21364] MiBench Workloads and Estimation Expressions when different inputs sets were considered

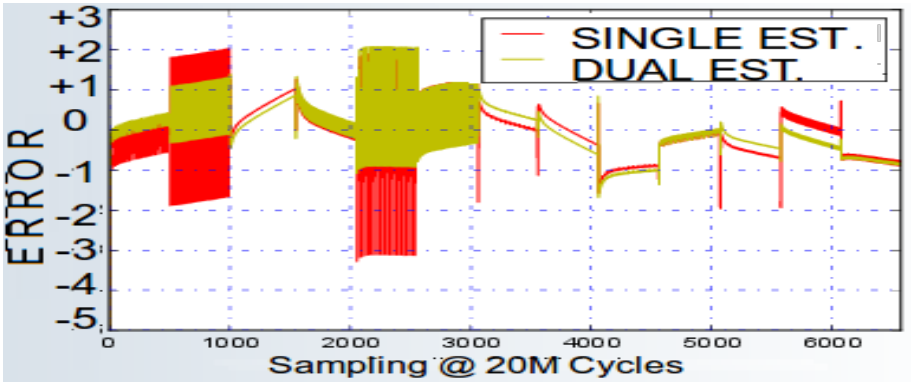


Figure 4.1: Temperature Error Variation for Dual Estimator and Single Estimator

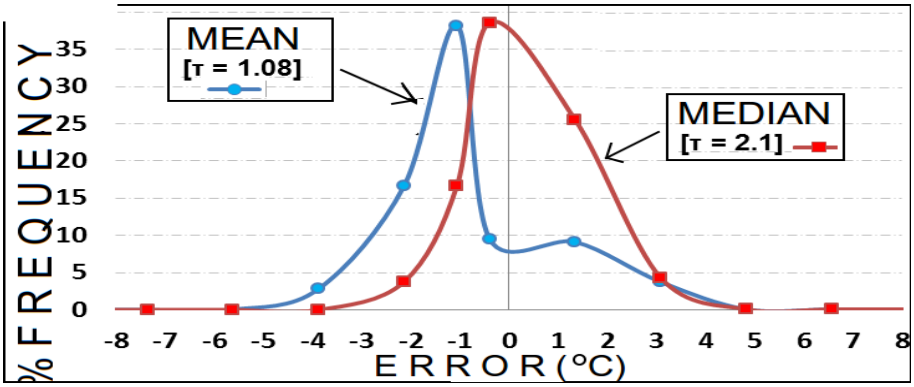
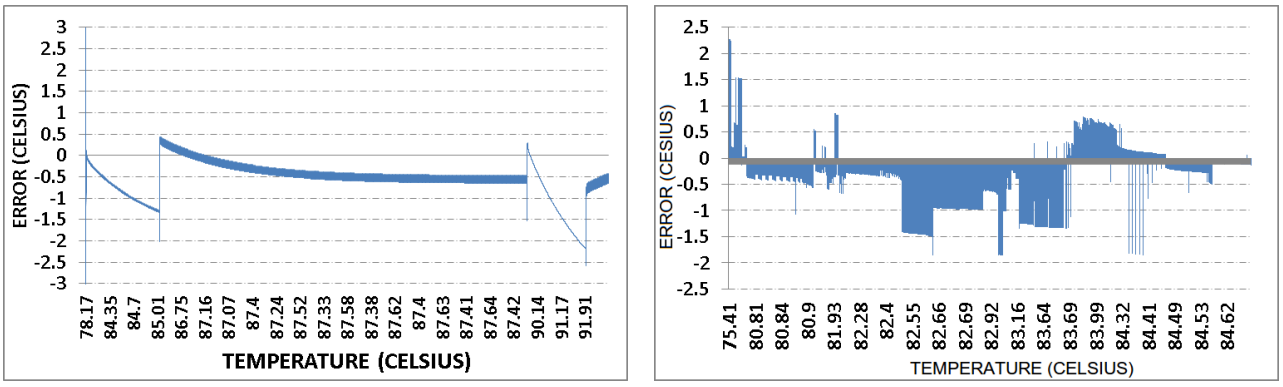


Figure 4.2: Threshold Selection



(a) Alpha21364 (b) AMD-Athlon64

Figure 4.3: MiBench Suites: Error Distribution in High Temperature Region (>75 °C) for both Alpha and Athlon Processor cores.

A better evaluation of our approach can be done when a working set different from that used in the training set is used as an input to the task sets (Table 4.1). We found a lower effective error swing contained within ±2°C, with an average absolute error of 0.697°C, with a standard deviation of 0.765. This error variation is in the medium temperature zone (i.e., greater than 60°C and less than or equal to 75°C). In the high temperature zone (i.e., greater than 75°C), the average absolute error is 0.48°C with a standard deviation of 0.717°C.

We also tested our approach using distinct training and testing sets. Figures 4.3 (a) and (b) show the errors for the worst case testing set, i.e., unknown set of workloads, illustrating the low error swing in high temperature zone for MiBench Suites across Alpha21364 and AMD-Athlon64 processor, respectively. The estimation absolute error is mostly below 2.5 °C. This indicates the effectiveness of our approach at high temperature, which is precisely where high accuracy is important.

Dual Estimator (τ=Median=1.75)	
la_est	*ha_est*
$T_{est}^{IntReg}(n) =$ $3.45 * IRA + 0.16 * T_{est}^{IntReg}(n-1) + 52.50$	$T_{est}^{IntReg}(n) =$ $1.87 * IRA + 0.75 * T_{est}^{IntReg}(n-1) + 12.70$

Table 4.2: [Alpha21364] MiBench Estimation Expression for one of training set.

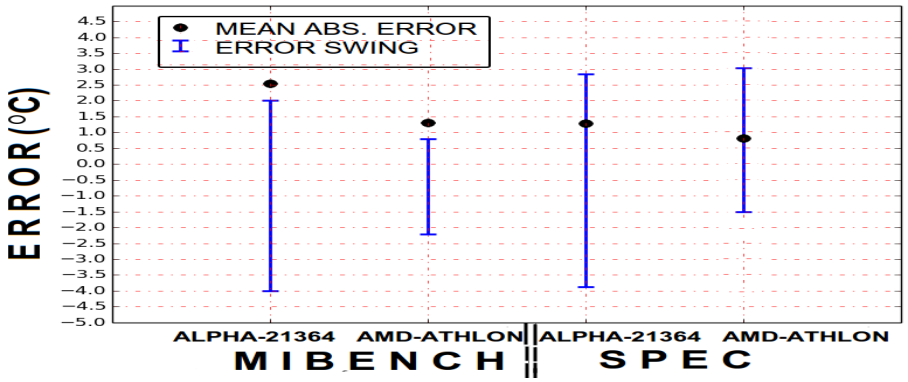


Figure 4.4: MiBench Suites: Overall error statistics across different training-testing sets

Figure 4.4 shows the overall error swing for the worst-case testing set on two different processors. On the average, shown as a black dot, our estimator performs fairly well. The error swing is within $[-4°C, +3°C]$ and an average error is $+1.2°C$ at maximum temperature of 83.5°C (averaged over all sets). Moreover, adding more performance counters did not help in improving the effective error.

As an alternative, one can also use the history of performance counters instead of using thermal history. A history window of five previous activities gave the same effect as our *History-Based Dual Estimator* approach.

4.1.1.2 IntReg Unit Estimation: SPEC Suites

Similar experiments were carried out on the SPEC benchmark suite. Table 4.3 shows three estimators; the dual estimator for our approach (for high and low activity) and the estimator derived using the approach in [7]. The latter resulted in error range of $[-9.19°C, +10.95°C]$ for the *bzip* benchmark. *By comparison, our approach has an error swing of* $[-1.52°C, +2.59°C]$.

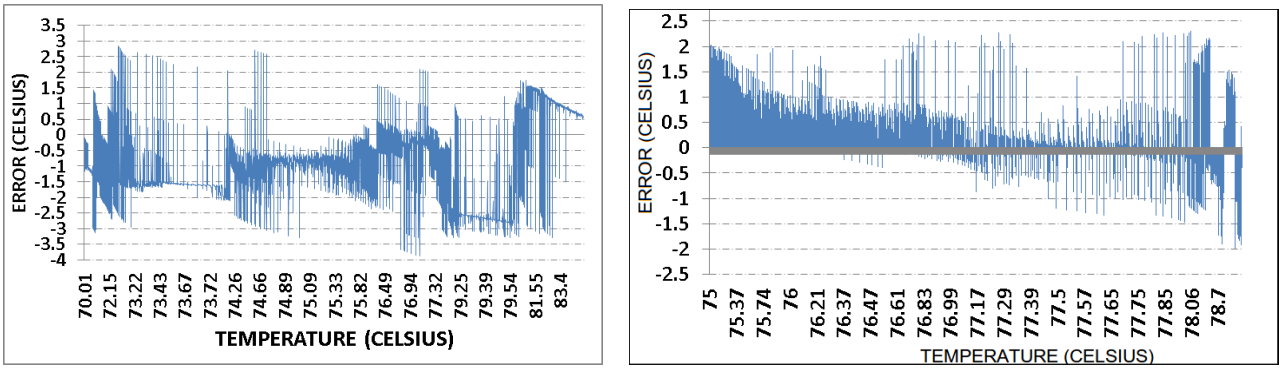
Figure 4.5 shows error variation in the high temperature zone. The error is limited to the range $[-3°C, +2.5°C]$. The estimate is much worse in low temperature regions where we underestimate up to -4°C; however, due to the nonlinear relationship between reliability and temperature, this will not greatly degrade the reliability estimates. Figure 4.4 shows

the error swing and average error for worst-case testing set, across Alpha and AMD-Athlon floorplan. Note that at the beginning, the estimate starts with a large error of up to -16°C but this error comes down after about 20 samples, and the estimation stabilizes itself towards higher accuracy. The rationale behind this behavior is that, when applying linear regression the initial condition is assumed to be error free but this is not the case when we initiate the estimation and because of geometric progression of the estimator, the error phases out.

Dual Estimator ($\tau = Median = 1.3$)		Baseline-Estimator based on [7]
Below Median	*Above Median*	
$T_{est}^{IntReg}(n) = 3.13 * IRA + 0.54 * T_{est}^{IntReg}(n-1) + 24.31$	$T_{est}^{IntReg}(n) = 3.00 * IRA + 0.34 * T_{est}^{IntReg}(n-1) + 35.06$	$T_{est}^{IntReg}(n) = 7.05 * IRA + 56.67$

Table 4.3: [AMD Athlon64] Comparative Study (SPEC Suite): Estimation Expression for one of the training set.

Our results show an average effective error swing, in the high temperature zone, to be between -3°C to +2.5°C and between -4.5°C to +4°C in the moderate temperature zone.



(a) Alpha21364 (b) AMD-Athlon64

Figure 4.5: SPEC Suites: Error Distribution in High Temperature Region (>75 °C) for both Alpha and Athlon Processor cores.

Thus far, all the expressions used for curve-fitting have been linear. We have also experimented using non-linear expressions; however, the estimation error turned out to be

much worse, approximately -15°C. Also, we tried the non-linear time series technique using Artificial Neural Networks[35]; this provided no improvement over the linear technique.

4.1.2 IntScheduler Unit Estimation

Across most of the workloads and execution phases, the Integer-Scheduler Unit (IntSchd) is the hottest unit in AMD-Athlon. We carried out experiments following the same methodology. In this case, Issue Rate seems to be highly correlated with the IntSchd unit. In this case also, we found that a dual-estimator provided better accuracy than a single estimator. Table 4.4 shows the dual-estimator and for the worst case testing set the accuracy of the estimator is within $[-2°C, +2.5°C]$ and $[-3.8°C, +4.5°C]$ for MiBench and SPEC Suites, respectively. Note also that our estimate of the peak temperature has an underestimate of

Dual Estimator (τ=Median=1.8)	
la_est	*ha_est*
$T_{est}^{IntSchd}(n) = 0.168 * IssueRate + 0.98 * T_{est}^{IntSchd}(n-1) + 1.106$	$T_{est}^{IntSchd}(n) = 6.553 * IssueRate + 0.5913 * T_{est}^{IntSchd}(n-1) + 16.29$

Table 4.4: [AMD-Athlon::SPEC-Suites] Estimation Expression for one of training set.

about +2°C at a peak (average) temperature of 88.3°C.

4.1.3 FPReg Unit Estimation

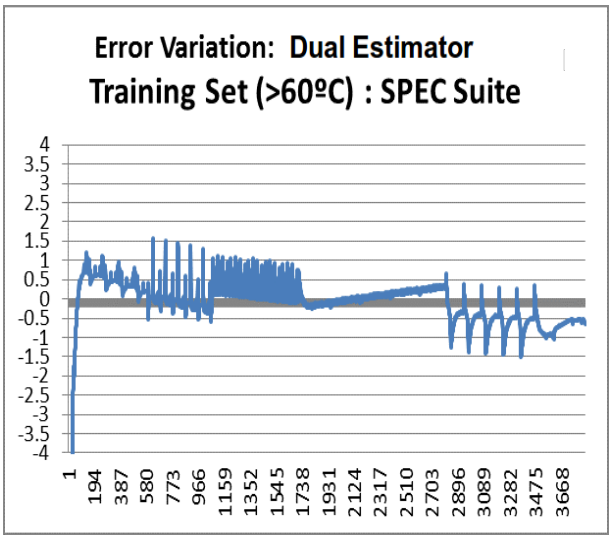
Another potential unit which need thermal monitoring is the Floating-Point Scheduler and Floating-Point Register (FPReg) Unit, where the impact of thermal variation can be up to 125°C, if no DTM control is applied [39]. This unit shows thermal fluctuations due to varying floating point activity. In the following sections, we present the estimation for two different benchmark suites.

4.1.3.1 FPReg Unit Estimation: SPEC Suites

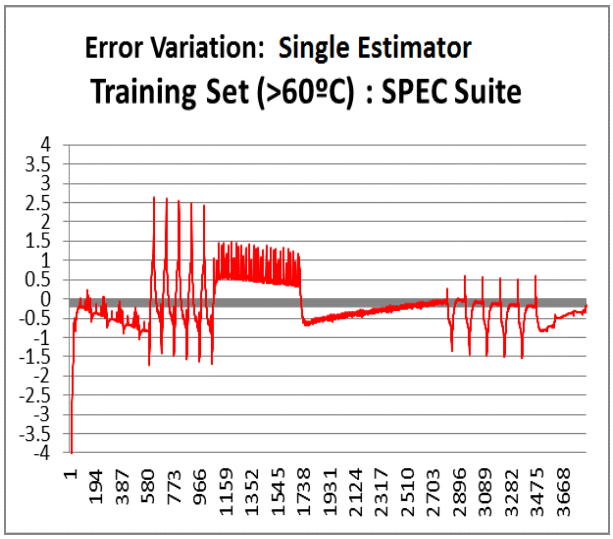
Similar to IntReg Unit, we have chosen a set of representative benchmarks for the estimation (training and testing) of FPReg Unit, as shown in Table 4.5.

Workloads	Single (Alpha-Core) Estimator	Single (Athlon-Core) Estimator
bwaves, milc, namd, dealII, soplex, calculix, mgrid, applu, mesa, art, equake and ammp	$T_{est}^{FPReg}(n) = 1.7129 * NFP + 12.618 * BMP + 0.9232 * T_{est}^{FPReg}(n-1) + 4.5787$	$T_{est}^{FPReg}(n) = 1.0357 * NFP + 8.241 * BMP + 0.7546 * T_{est}^{FPReg}(n-1) + 7.2774$

Table 4.5: [Alpha21364 and AMD-Athlon Core] SPEC Workloads and Estimation Expressions for FPReg Unit



(a) Error: Dual Estimator



(b) Error: Single Estimator

Figure 4.6: SPEC Suites: Comparison of Single Estimator vs. Dual Estimator

Figure 4.6 shows the estimation error when we have used single estimator vs. dual estimators. Unlike IntReg, here the improvement offered by the dual estimator is not much. We therefore decided to choose the single estimator approach for the thermal estimation of FPReg Unit. The overall error variation for worst case testing set is in the range of -2.8°C to +3.5°C, with an average absolute error of 1.8°C, as shown in Figure 4.7.

4.1.3.2 FPReg Unit Estimation: MiBench Suites

Similar experiments were done for MiBench benchmark suites. Table 4.6 shows the set

Workloads	Alpha-Estimator	Athlon-Estimator
basicmath, bitcount, fft, jpeg_decode, lame, patricia, qsort, rijndael, susan, typeset, crc, dijkstra and gsm	$T_{est}^{FPReg}(n) = 5.182 * NFP + 0.3782 * T_{est}^{FPReg}(n-1) + 40.701$	$T_{est}^{FPReg}(n) = 2.0825 * NFP + 0.3475 * T_{est}^{FPReg}(n-1) + 41.1412$

Table 4.6: MiBench Workloads and Estimation Expression for FPReg Unit

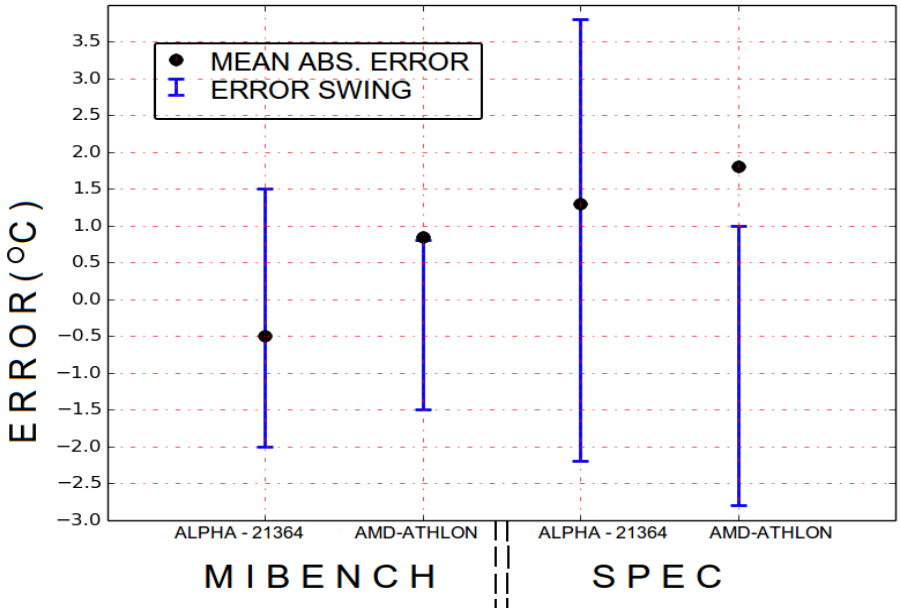


Figure 4.7: FPReg Unit: Overall error statistics for MiBench and SPEC Suites, across different processors.

of representative benchmarks which are employed for the estimation and the corresponding estimator, which is generated from the training set, for Alpha21364 and AMD-Athlon Processor cores.

The overall error variation for the worst case testing set is in the range of -2°C to +1.5°C, with an average absolute error of 0.8°C, as shown in Figure 4.7.

4.1.4 Floating-Point Scheduler Unit Estimation

We carried out experiments following the same methodology as before. In this case also, Issue Rate seems to be highly correlated with the Floating-Point Scheduler (FPSchd) Unit. Table 4.7 shows the estimators. For the worst case testing set the accuracy of the estimators is within [−2°C, +0.5°C] and [−2°C,+2°C] for MiBench and SPEC Suites, respectively. Note also that for the worst case testing set our estimate of the peak temperature has an underestimate of about +1.08°C at a peak temperature of 72.9°C.

MiBench-Estimator	**SPEC-Estimator**
$T_{est}^{FPSchd}(n) = 0.1478 * IssueRate + 0.6625 * T_{est}^{FPSchd}(n-1) + 21.084$	$T_{est}^{FPReg}(n) = 1.3005 * IssueRate + 0.7177 * T_{est}^{FPSchd(n-1)} + 16.52$

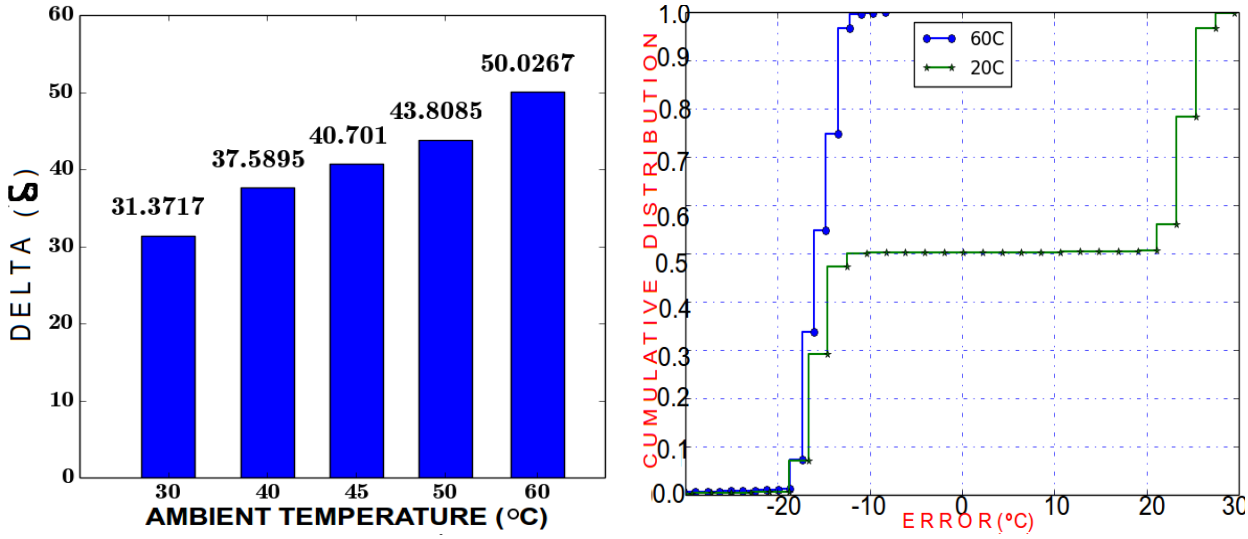
Table 4.7: Estimation Expression for Floating Point Scheduler Unit

4.2 Impact of Uncertainity in Ambient Temperature

The estimation accuracy which we have achieved is obtained at an ambient temperature of 45°C. But, in actual cases the ambient temperature may be different from the one on the basis of which the passive estimator(s) were generated. As a result, estimation will result in errors.

Figure 4.8(a) shows an almost linear increase in the offset term of the estimator with respect to ambient temperature. The effect of this variability on the estimation is evident from Figure 4.8(b), which shows the proportional shift in estimation (over/under estimation) of the FPReg Unit temperature, when tasks from MiBench suites are considered. In this case we have used a passive estimator, which was trained at 45°C and is then employed to track thermal fluctuations corresponding to an ambient temperatures of 20°C and 60°C.

In order to mitigate this, we have considered a design that includes an on-chip temperature sensor. Such a sensor will clearly reduce the impact of the variability in the assumed ambient temperature. However, on-line sensors have their own inaccuracies and the question



(a) Variation of offset term with Ambient Temperature

(b) Effect of Uncertainty in Ambient temperature

Figure 4.8: Variation of offset term in the estimator and its effect on FPReg temperature estimation error when the estimator was trained for an assumed ambient temperature of 45°C and tested on ambient temperatures of 20°C and 60°C

is whether the combined use of performance counters and sensors can yield a more robust estimator.

Low-Activity Estimator	**High-Activity Estimator**
$T_{est}^{IntSched}(n) = 0.05 * IssueRate + 3.3 * SR\_SCHD + 0.96 * T(n-1) - 0.25$	$T_{est}^{IntSched}(n) = 0.15 * IssueRate + 6.01 * SR\_SCHD + 0.94 * T(n-1) - 0.83$

Table 4.8: [AMD Athlon64] Estimators for the IntScheduler, based on IssueRate and the normalised on-chip sensor reading for the IntScheduler, SR_SCHD.

Table 4.8 shows INTScheduler temperature estimators (for SPEC06 Suites) when an on-chip thermal sensor has been integrated into the chip. The estimator is derived by combining thermal traces over a range of ambient temperatures, $20°C$ to $60°C$, with an interval of $10°C$. In addition, we have introduced an inaccuracy of $[-4°C, +4°C]$ in the thermal sensor [32] reading, so as to simulate a practical scenario. The injected inaccuracy is linear which is based on $2-$point calibration method as discussed in [10] and the sensor reading is obtained

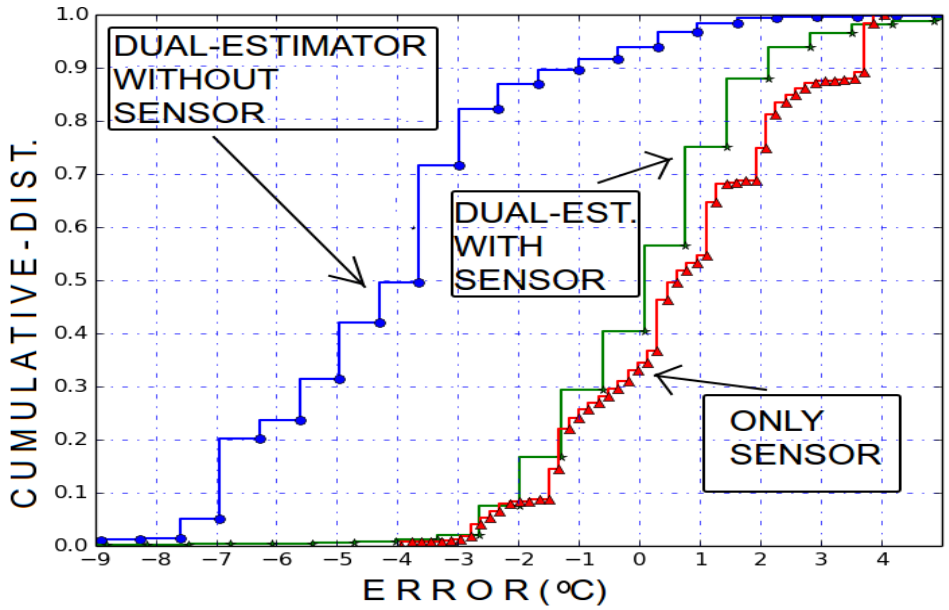


Figure 4.9: SPEC Suites: IntScheduler Unit (AMD Athlon) Error variation for one of the testing sets, when estimator was trained on a range of ambient (20°C - 60°C) and tested on ambient of 50°C.

as per the following equation.

$$\texttt{Sensor\_Response} = 0.837 * \texttt{Actual\_Temperature} + 11.34 \tag{4.1}$$

Figure 4.9 shows the variation in the INTScheduler temperature estimation error for one testing set. As evident, our approach performs fairly well over thermal sensors; thereby having better accuracy in estimation by almost +1°C. The overall estimation error across different training-testing sets for the selected range of ambient temperatures is within the tolerable range of $[-3°C, +3.88°C]$. Our experiments have shown that, while training the estimator, we need to train on the entire range of ambient temperatures and not just the outliers, otherwise the estimator may not be able to adequately tolerate the ambient temperature variability.

4.3 Reliability Estimation

In the past few years, instead of treating reliability improvement as an indirect benefit of thermal management, researchers have started to consider the reliability of VLSI circuits as an optimization criterion. In [40] and [18] the authors have presented reliability-aware dynamic scheduling techniques. These previous works suggest a potential for meaningful improvements in reliability. The basis of these techniques is the correct estimation of the failure rate of a processor. We can use thermal accelerated aging (TAAF) to estimate the impact on reliability and it is defined, for electomigration phenomenon, as:

$$TAAF(n) = \frac{MTTF_{amb}}{MTTF(n)} = e^{(E_a/\kappa)*(1/T_{amb}-1/T(n))} \tag{4.2}$$

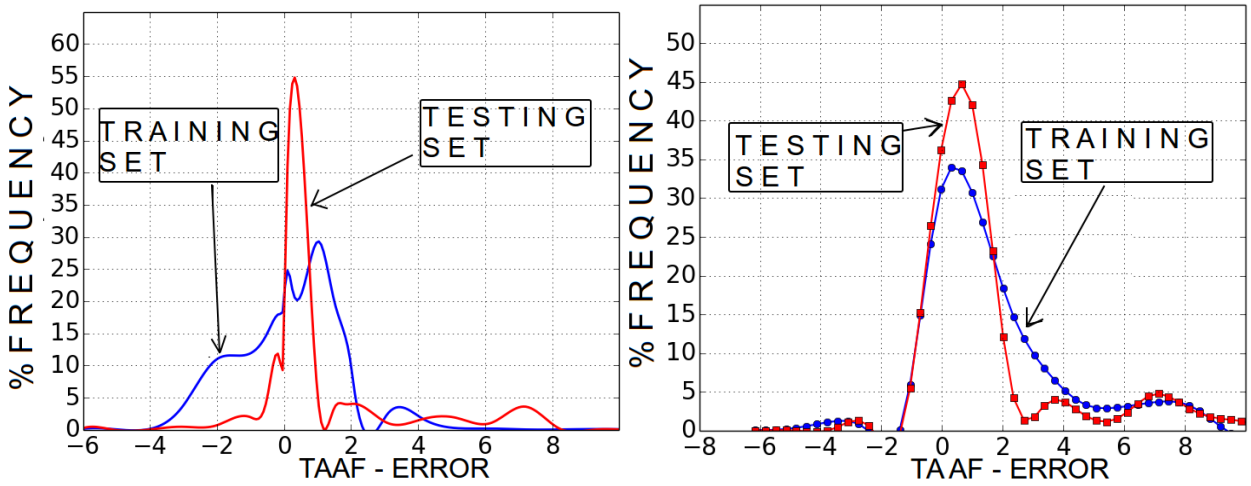
where $MTTF_{amb}$ is the Mean-Time-To-Failure at a given ambient temperature and $MTTF(n)$ is the Mean-Time-To-Failure corresponding to the estimated temperature at instance n, E_a is the activation energy, κ is the Boltzmann constant, T_{amb} is the ambient temperature in Kelvin and $T(n)$ is the absolute temperature at n.

Run-time estimation of reliability can allow balancing of the thermal stress in a multi-core environment. Using Equation 4.2 we monitor the TAAF by backtracking from the estimated temperature, as explained in Section 4.1.2. Figure 4.10(a) shows that the estimation error in TAAF is in the range of $[-4, +4]$.

We also made an attempt to estimate TAAF directly using performance counters. We followed the methodology outlined in Section 3.1.2, replacing $T(n)$ with $TAAF(n)$. Equation 4.3 shows the TAAF estimator for Integer Scheduler Unit which takes into account the impact of thermal stress, i.e., $TAAF(n-1)$.

$$\texttt{TAAF(n)} = 0.22 * \texttt{IssueRate} + 0.97 * \texttt{TAAF(n} - 1\texttt{)} + 0.05 \tag{4.3}$$

We assumed the initial condition as $TAAF(0) = 1$, i.e., no thermal age acceleration. Figure 4.10(b) shows that the estimation error in $TAAF$ is within $[-2, +5]$.



(a) Indirect TAAF Estimator (b) Direct TAAF Estimator

Figure 4.10: [SPEC Suites - AMD Athlon] TAAF Estimator for IntScheduler Unit using (a) Estimated Temperature (b) Direct TAAF monitoring using performance counters and accrued thermal stress

Although estimating $TAAF$ directly from performance counters is fast, there is an over-estimation associated with direct estimation. This indicates that faster thermal acceleration may force one to take premature reliability-aware decisions. Overall, both the indirect and direct estimations have a tolerable error level and the decision on which one to use will depend on the lifetime reliability requirements.

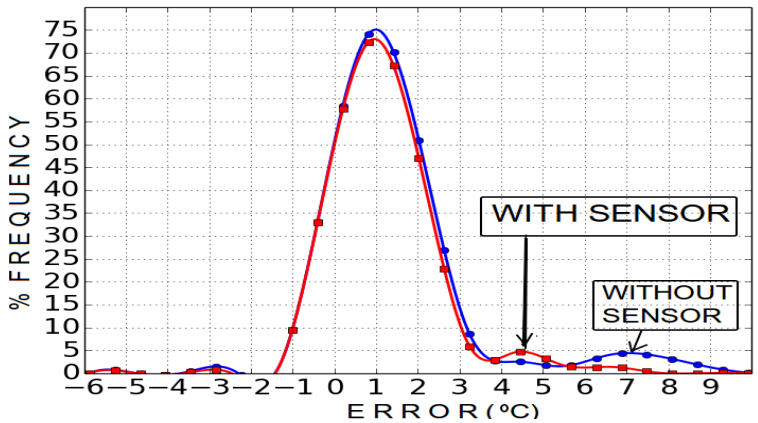


Figure 4.11: [AMD-Athlon: Testing Set] Error Variation in Direct Estimation of TAAF for IntScheduler Unit using performance counters and normalized sensor reading

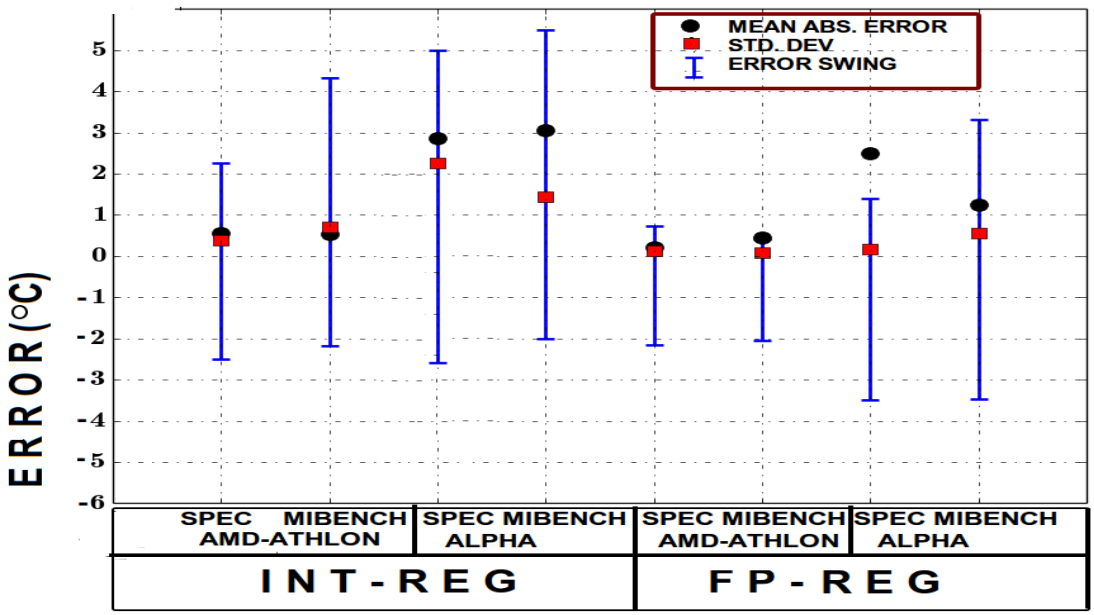


Figure 4.12: Overall Error Variation in Direct Estimation of TAAF across Alpha and AMD-Athlon Processor for IntReg and FPReg Unit.

We also tried to incorporate sensor reading into the TAAF estimation for the Integer Scheduler Unit. As evident from Figure 4.11, there is not a significant improvement in estimation accuracy by including sensor reading.

We did similar experiments for the IntReg and FPReg Units. Table 4.9 shows TAAF estimators which were generated using the same methodology as used before for temperature estimation. Figure 4.12 shows the overall error swing, mean absolute error and standard

IntReg Estimator	**FPReg Estimator**
$TAAF_{est}^{IntReg}(n) =$ $0.1976 * IRA + 0.77 * TAAF(n-1) + 0.947$	$TAAF_{est}^{FPReg}(n) =$ $0.636 * NFP + 0.832 * TAAF(n-1) + 0.627$

Table 4.9: [SPEC Suites] TAAF Estimators for the IntReg and FPReg Unit for AMD-Athlon Core.

deviation for the worst case testing set across two different floorplans, for IntReg and FPReg Units. As evident, the overall error is within a tolerable error range of [−3.2,+5.2], with an

average absolute error of +3. We also tried non-linear estimators but those estimators did not provide improvement over the linear (feedback) technique.

CHAPTER 5

CONCLUSIONS

In this work we have presented an effective online reliability monitoring technique using performance counters. The accuracy in the estimation of TAAF is within a fairly tight range. Our results show that the thermal history has a significant impact on reducing the offset term of the thermal estimator, reducing over/under estimation. We have also presented a novel dual estimator scheme that enhances the temperature estimation accuracy to an effective lower tolerable range of $[-3.5°C, +4.5°C]$. Compared to previously proposed techniques, our technique is lightweight, requiring just two or three multiplications and two additions.

We have also presented an approach to combine performance counter monitoring with on-chip sensor(s) to tolerate variability in ambient temperature. When properly selected and sampled at the appropriate rate, performance counters can form the basis of an effective and efficient mechanism for accurately guiding thermal and reliability control algorithms in processors.

A suggested direction of future study is to devise an online learning technique for minimizing the estimation error using weighted least square error (WLSE) algorithm [28]. WLSE estimation technique may minimize the error between the estimated temperature value and the actual sensor value by taking the weighted sum of errors and minimizing it with a set of weights at run-time. Another possible direction could be cross-unit thermal estimation by taking into account the appropriate placement of thermal sensors. That is, having the temperature estimate of one unit, reconstruct the thermal map for another unit.

BIBLIOGRAPHY

[1] Spec cpu2000 suite: The standard performance evaluation corporation. `https://www.spec.org/cpu2000/`, 2000.

[2] Spec cpu2006 suite: The standard performance evaluation corporation. `https://www.spec.org/cpu2006/`, 2006.

[3] Akaike, H. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19 (Dec 1974), pp. 716–723.

[4] Bao, M., Andrei, A., Eles, P., and Peng, Z. On-line thermal aware dynamic voltage scaling for energy optimization with frequency/temperature dependency consideration. In *Proceedings of the 46th ACM/IEEE on Design Automation Conference (DAC), 2009.* (July 2009), pp. 490–495.

[5] Berktold, M., and Tian, T. Cpu monitoring with dts/peci. `www.intel.com/content/dam/www/public/us/en/documents/white-papers/cpu-monitoring-dts-peci-paper.pdfpaperCPUMonitoringWithDTS/PECI`, 2010.

[6] Binkert, N., Beckmann, B., Black, G., Reinhardt, S. K., Saidi, A., Basu, A., Hestness, J., Hower, D. R., Krishna, T., Sardashti, S., Sen, R., Sewell, K., Shoaib, M., Vaish, N., Hill, M. D., and Wood, D. A. The gem5 simulator. *SIGARCH Comput. Archit. News*, 39 (Aug. 2011), pp. 1–7.

[7] Chung, S. W., and Skadron, K. Using on-chip event counters for high-resolution, real-time temperature measurements. In *Thermal and Thermomechanical Phenomena in Electronics Systems, 2006*, pp. 114–120.

[8] Chung, S. W., and Skadron, K. A novel software solution for localized thermal problems. In *Parallel and Distributed Processing and Applications*. Springer, 2006, pp. 63–74.

[9] Coskun, A. K., Rosing, T. S., and Gross, K. C. Proactive temperature management in mpsocs. In *Proceedings of the 13th international symposium on Low power electronics and design* (2008), ACM, pp. 165–170.

[10] Datta, B., and Burleson, W. Calibration of on-chip thermal sensors using process monitoring circuits. In *Proceedings of the 11th International Symposium on Quality Electronic Design (ISQED), 2010* (March 2010), pp. 461–467.

[11] Datta, B., and Burleson, W. Low-power, process-variation tolerant on-chip thermal monitoring using track and hold based thermal sensors. In *Proceedings of the 19th ACM Great Lakes Symposium on VLSI* (2009), GLSVLSI '09, ACM, pp. 145–148.

[12] Dorsey, J., Searles, S., Ciraula, M., Johnson, S., Bujanos, N., Wu, D., Braganza, M., Meyers, S., Fang, E., and Kumar, R. An integrated quad-core opteron processor. In *Proceedings of IEEE International Conference on Solid-State Circuits, 2007. ISSCC 2007. Digest of Technical Papers*, pp. 102–103.

[13] Drongowski, P. J. Basic performance measurements for AMD Athlon 64, AMD Opteron and AMD Phenom processors, September 2000.

[14] Guthaus, M., Ringenberg, J., Ernst, D., Austin, T., Mudge, T., and Brown, R. Mibench: A free, commercially representative embedded benchmark suite. In *IEEE International Workshop on Workload Characterization (WWC-4), 2001.* (Dec 2001), pp. 3–14.

[15] Hall, M. A. *Correlation-based Feature Subset Selection for Machine Learning.* PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

[16] Han, Y., Koren, I., and Krishna, C. Temptor: A lightweight runtime temperature monitoring tool using performance counters. In *Proceedings of the Third Workshop Temperature-Aware Computer Systems, in conjunction with ISCA-33, 2006*, pp. 17–28.

[17] Holmes, G., Donkin, A., and Witten, I. H. Weka: a machine learning workbench. In *Proceedings of the Second Australian and New Zealand Conference on Intelligent Information Systems.* (Nov 1994), pp. 357–361.

[18] Huang, L., Yuan, F., and Xu, Q. Lifetime reliability-aware task allocation and scheduling for mpsoc platforms. In *Proc. of the Conf. on Design, Automation and Test in Europe* (2009), European Design and Automation Association, pp. 51–56.

[19] Huang, W., Member, S., Ghosh, S., Velusamy, S., Sankaranarayanan, K., Skadron, K., Stan, M. R., Member, S., and Member, S. Hotspot: a compact thermal modeling methodology for early-stage vlsi design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 14 (May 2006), pp. 501–513.

[20] Isci, C., and Martonosi, M. Runtime power monitoring in high-end processors: Methodology and empirical data. In *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture* (2003), MICRO 36, IEEE Computer Society, pp. 93–104.

[21] Koren, I., and Krishna, C. Temperature-aware computing. *Sustainable Computing: Informatics and Systems*, 1 (2011), pp. 46–56.

[22] Kuppuswamy, R., Sawant, S., Balasubramanian, S., Kaushik, P., Natarajan, N., and Gilbert, J. Over one million tpcc with a 45nm 6-core xeon; cpu. In *Proceedings of IEEE International Conference on Solid-State Circuits - Digest of Technical Papers, 2009. ISSCC 2009.*, pp. 70–71.

[23] Lee, J. S., Skadron, K., and Chung, S. W. Predictive temperature-aware DVFS. *IEEE Transactions on Computers*, 59 (Jan 2010), pp. 127–133.

[24] Lee, K.-J., and Skadron, K. Using performance counters for runtime temperature sensing in high-performance processors. In *Proceedings of the 19th IEEE International Symposium on Parallel and Distributed Processing* (2005), pp. 4–8.

[25] Li, S., Ahn, J.-H., Strong, R., Brockman, J., Tullsen, D., and Jouppi, N. Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture, 2009. MICRO-42.* (Dec 2009), pp. 469–480.

[26] Merkel, A., and Bellosa, F. Task activity vectors: A new metric for temperature-aware scheduling. In *Proceedings of the 3rd ACM SIGOPS/EuroSys European Conference on Computer Systems* (2008), Eurosys '08, ACM, pp. 1–12.

[27] Mesa-Martinez, F. J., Nayfach-Battilana, J., and Renau, J. Power model validation through thermal measurements. In *Proceedings of the 34th Annual Int'l Symp. on Computer Architecture* (2007), ISCA '07, ACM, pp. 302–311.

[28] Monzingo, R. A., and Miller, T. W. Introduction to adaptive arrays, 1980.

[29] Mukherjee, S., Bannon, P., Lang, S., Spink, A., and Webb, D. The alpha 21364 network architecture. *Micro, IEEE 22* (Jan 2002), pp. 26–35.

[30] Rodrigues, R., Annamalai, A., Koren, I., and Kundu, S. A study on the use of performance counters to estimate power in microprocessors. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 60 (Dec 2013), 882–886.

[31] Rotem, E., Hermerding, J., Cohen, A., and Cain, H. Temperature measurement in the intel (r) coretm duo processor. `arxiv.org/pdf/0709.1861`, 2007.

[32] Sharifi, S., and Rosing, T. S. An analytical model for the upper bound on temperature differences on a chip. In *Proc. of the 18th ACM Great Lakes Symp. on VLSI* (2008), GLSVLSI '08, ACM, pp. 417–422.

[33] Singh, K., Bhadauria, M., and McKee, S. A. Prediction-based power estimation and scheduling for cmps. In *Proceedings of the 23rd international conference on Supercomputing* (2009), ACM, pp. 501–502.

[34] Srinivasan, S., Ganeshpure, K. P., and Kundu, S. A wavelet-based spatio-temporal heat dissipation model for reordering of program phases to produce temperature extremes in a chip. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 31 (2012), pp. 1867–1880.

[35] Suykens, J., and Vandewalle, J. Training multilayer perceptron classifiers based on a modified support vector method. *IEEE Transactions on Neural Networks*, 10 (Jul 1999), pp. 907–911.

[36] Thomas Willhalm, Roman Dementiev, P. F. Intel performance counter monitor - a better way to measure cpu utilization. `https://software.intel.com/en-us/articles/intel-performance-counter-monitor`, 2012.

[37] Upton, D., and Hazelwood, K. Evaluating linear regression for temperature modeling at the core level. In *Workshop on Duplication, Deconstructing, and Debunking* (2011), pp. 8–14.

[38] Zhou, H., Li, X., Cher, C.-Y., Kursun, E., Qian, H., and Yao, S.-C. An information-theoretic framework for optimal temperature sensor allocation and full-chip thermal monitoring. In *Proceedings of the 49th ACM/EDAC/IEEE on Design Automation Conference (DAC), 2012* (June 2012), pp. 642–647.

[39] Zhu, Y., and Albonesi, D. Localized microarchitecture-level voltage management. In *Proceedings of the IEEE Int'l Symp. on Circuits and Systems (ISCAS), 2006.* (May 2006), pp. 40–44.

[40] Zhuo, C., Sylvester, D., and Blaauw, D. Process variation and temperature-aware reliability management. In *Proceedings of the Design, Automation Test in Europe Conference (DATE), 2010*, pp. 580–585.

[41] Zoni, D., Corbetta, S., and Fornaciari, W. Hands: Heterogeneous architectures and networks-on-chip design and simulation. In *Proceedings of the 2012 ACM/IEEE International Symposium on Low Power Electronics and Design* (2012), ISLPED '12, ACM, pp. 261–266.