Data-Driven Structured Thermal Modeling for COTS Multi-Core Processors

Seyedmehdi Hosseinimotlagh, Daniel Enright, Christian R. Shelton, and Hyoseung Kim
University of California, Riverside
{shoss007, denri006, cshelton, hyoseung}@ucr.edu

***Abstract*—Thermal awareness is increasingly important for real-time systems deployed in harsh environments. As high chip temperature can cause frequency throttling or shutdown of processor cores at unexpected times, many real-time scheduling techniques have been developed to ensure continuous, fail-safe operation of safety-critical tasks with stringent timing constraints. However, their practical use remains largely limited due to the fact that it is extremely difficult to obtain a precise thermal model of commercial processors without using special measurement instruments or access to proprietary information, such as the power traces of micro-architectural units and detailed floorplans.**

In this paper, we propose a data-driven structured thermal modeling scheme that is directly applicable to commercial off-the-shelf multi-core processors used in real-time embedded systems. By using a small number of thermal profiles obtained from on-chip temperature sensors, our scheme can accurately predict the processor operating temperature under dynamic real-time workloads at various CPU frequencies and ambient conditions. The thermal model derived from our scheme is fast to converge and robust against different sources of errors. Our scheme is non-intrusive, meaning that it does not require changes to the software code or the hardware packaging of the target system. Furthermore, our scheme can estimate the relative power consumption of the processor for a given workload and clock frequency level. Experimental results from a multi-core ARM platform indicate that our scheme estimates the operating temperature with a maximum error of 2.5% while the latest prior work results in 23% error. This highly accurate modeling enables us to obtain the maximum achievable processor utilization that does not cause a thermal safety violation.

I. Introduction

A major concern in recent embedded systems is the high heat dissipation caused by complex applications running on high-performance multi-core systems-on-chips (SoCs). The high temperature also increases power consumption [2], reduces the chip reliability [35], [36], and leads to chip burnout. Due to these reasons, many modern operating systems (OSs) monitor SoC operating temperatures using on-chip temperature sensors to ensure thermal-safe operation.

The reactive thermal management mechanisms implemented to protect the system from burnout raise a challenging problem in real-time embedded systems. To protect the processor from thermal damage, OS thermal governors define a set of policies to dynamically throttle CPU core frequencies or shut cores down [5], [12], [37] when temperatures exceed operationally safe limits. Such thermal countermeasures, however, lead to timing unpredictability since the deadlines of tasks could be unexpectedly violated by reduced processing speed or temporarily unavailable CPU cores.

To prevent the negative impact of such performance disruption, extensive studies have been conducted including the techniques based on Dynamic Voltage Frequency Scaling (DVFS) [10], [26], [30], [34], [38] and forced sleeping during the execution of *hot* applications [4], [16], [22], [20], [19], [38]. Moreover, offline analysis has been proposed to guarantee the thermal safety of real-time tasks by bounding the maximum operating temperature in the steady [13], [14] and transient states [29] of a given system. These approaches assume a priori knowledge of a precise thermal model for temperature prediction, resource management, and task scheduling, and require accurate simulation tools or extra equipment for validation. Therefore, identifying an accurate thermal model of a given multi-core SoC is the fundamental requirement to substantiate these techniques in practice, and we call it a *thermal system identification* problem in this paper. Such a thermal model should also be in a format compatible with those used in prior work so that it is applicable to the timing and thermal analysis of a wide range of real-time systems.

Despite its importance, the thermal system identification of commercial off-the-shelf (COTS) processors still remains a challenging problem. Existing numerical simulation tools like HotSpot [18] can construct a compact thermal model for modern VLSI devices by using a resistance-capacitance (RC) thermal network to capture the transient temperature and generate a heatmap at each time instant. However, they are not difficult to use with modern COTS multi-core processors because the information required by these tools is proprietary and not publicly available. An exhaustive search approach for approximating this information through reverse engineering is time-consuming and prone to unacceptably high inaccuracies [31], especially for transient temperature estimation.

The latest work [1], [31], [32] partially addresses these issues by calibrating the parameters of thermal models without power traces and detailed floorplans. However, it is not applicable to systems using priority-based preemptive scheduling where execution patterns are dynamic [13], [14], [29], [1], [7]. Similar to HotSpot, it uses a time-driven prediction model where the temperature is calculated for each time instant under static workloads. This makes the model not only slow but also infeasible to capture the correct operating temperature when multiple tasks with different periods are concurrently scheduled. Due to the extensive transient temperature data needed for system identification, it suffers from the numerical

instability in the temperature model especially in the presence of different types of errors and sensor limitations.

In this paper, we propose a data-driven structured thermal modeling scheme for COTS multi-core processors in real-time embedded systems. Our scheme requires only a small number of temperature traces from on-chip thermal sensors which are standard in today's processors. To achieve a precise thermal model of SoCs, our scheme estimates the location of CPU cores on the chip floorplan and considers this information during thermal parameter estimation. This reduces the number of unknown parameters so that a thermal model can be estimated with fewer temperature profiles. Besides, it allows obtaining a model with robustness against noise, reduced order, and ensured stability. We also provide techniques to improve the accuracy of thermal parameters through the ensemble of measurements from different frequency levels, execution patterns, or redundant experiments. Unlike prior work, our scheme is particularly well suited for use in real-time systems because the thermal model obtained by our work does not assume fixed execution patterns of application tasks and is thus applicable to any systems with preemptive work-conserving schedulers.

Contributions. The contributions of this paper are as follows:

- We present a thermal modeling scheme that has low computational cost by design. Given that steady-state profiles are much compact than transient-state profiles, our scheme first estimates the thermal parameters of a given system using only steady-state profiles, and then uses transient-state data for calibration purposes.
- We characterize various sources of errors in thermal system identification, and reduce their negative effects through the multiple refinement stages of our scheme. Our scheme also enables locating errors in the temperature profiles.
- Our scheme identifies the relative distance between CPU cores and produces an estimated chip floorplan from temperature profiles. It can also estimate the relative power consumption for a given workload on each CPU core.
- We present techniques to further improve the accuracy of thermal parameters by exploiting the ensemble of measurement data obtained at various frequency and workload settings.
- The effectiveness and accuracy of our proposed scheme is demonstrated with extensive experiments on a real ARM embedded platform. The results show that our scheme estimates the operating temperature with the maximum error ratio of 2.5% while the state-of-the-art gives 23%. Our scheme also addresses the problems of the start-of-the-art that disallows preemptive scheduling and overestimates the maximum achievable utilization by up to 18%.

II. Related work

There exist well-known numerical tools to estimate the chip operating temperature. For instance, HotSpot [18] solves the system of differential equations using the fourth-order Runge-Kutta numerical method through fine-granularity iterations. ATMI [27] proposed an analytical method to provide explicit solutions to the heat equation more efficiently than the Finite element method (FEM) and finite-difference methods (FDM). The authors of [11], [21] constructed a thermal model with the measured power and temperature trace of each subsystem on a real mobile platform. Power Blurring [39] calculates temperature distributions using a matrix convolution technique, unlike the finite-element analysis (FEA) used in other simulation tools like HotSpot.

There are several studies that estimate or bound chip operating temperature based on HotSpot, IR thermal imaging, and hardware performance counter measurements. An analytical method proposed in [17] utilizes a reduced RC thermal model that considers only *vertical* thermal conduction while ignoring the lateral thermal conduction. This method requires significantly less computational effort and less memory demand for temperature estimation than tools like HotSpot. The neural network approach to temperature estimation introduced in [33] predicts the rate of temperature changes by using hardware performance counters and the current estimate of the thermal map. During dynamic task execution in the design space exploration process, several features are extracted for the model by recording data taken from performance counters and dynamic heatmaps generated from IR imaging. Their run-time method is able to estimate temperature changes, but cumulative error forces the absolute temperature estimation to diverge from the actual chip temperature. The authors of [6] proposed a simplified temperature model that considers the transient and steady state thermal behavior (temporal effect) of multi-core processors in addition to the thermal conductance and dependency (spatial effect) of nearby cores. Data gathered from offline simulation using Hotspot was used to generate a temperature model considering only the temporal effect and another model based on the regression analysis of that data was then used to consider the spatial effect. The combination of both approaches led to temperature overestimation of up to 6°C. However, to apply these approaches in practice, they require detailed information about device power traces and floorplans which are not publicly available for commercial SoCs. Furthermore, some of these approaches require highly precise thermal imaging from IR cameras, assuming the absence of cooling packages like heat-sinks and fans.

There have been other studies that focus on thermal system identification based on on-chip sensors and linear control systems [31], [32], [8], [1], [24], [23]. The authors of [31], [32] proposed a calibration-based method to predict thermal behavior by modeling the thermal impulse response of each CPU core with respect to core utilization. Similar to [8], they utilized a modified Generalized-Pencil-Of-Function (GPOF) [15] to estimate the impulse response of each application from utilization and temperature traces. In their work, the thermal effects due to conduction between CPU cores are taken into account by the transfer matrices constructed from self-core and other-core transfer functions. The fundamental contribution of [31], [32] is based on the assumption that a thermal model can be constructed using thermal fingerprint of different applications on SoCs. However, some of the key thermal parameters of SoCs are characteristics

of semiconductor technology and can be obtained independent of the workloads executing on the systems. Hence, only relying on thermal fingerprints may lead to inaccurate thermal system identification and a large stored volume of transfer functions in the profiling stage because each transfer function has to be estimated for each application on each CPU core at different frequency levels. Moreover, if a task is preempted by another task or is suspended, the current thermal model is no longer valid. In order to maintain consistency in the temperature estimation, for each context switch, a convolution of the utilization trace of the new application with the application's thermal impulse response vector has to be performed, which can cause additional errors in the temperature estimation of the whole SoC. This spatio-temporal thermal dependency is hard to capture in their model for preemptively-scheduled tasks on the same CPU core or concurrently-executing tasks on other CPU cores. Due to this reason, despite all the other benefits provided, the applicability of their approach to real-time systems is limited. In [23], the thermal parameters of mobile devices equipped with SoCs and cooling packages are estimated. In their work, the chip temperature is considered as a single value and thermal conduction between CPU cores and other subsystems are not taken into account. The parameter identification process proposed in [24] estimates thermal-coupling coefficients by running CPU benchmarks on CPU cores one at a time measuring the steady state data from measurements. Lastly, the authors of [1] proposed a thermal-isolation server to ensure the thermal safety of real-time multi-core systems. Their server policy introduces a new direction to real-time system design and has inspired other thermal-aware real-time systems [14], [13]. However, since the thermal modeling part of their work is almost similar to the thermal fingerprinting approach in [31], [32], they also have the same limitations mentioned above.

All aforementioned research suffers from two major problems. The first is the lack of *persistency of excitation* in system identification, which means that the collected input-output data may not be rich enough to capture all the thermal characteristics of the chip. The need for a sizable amount of noiseless data due to a large number of unknown parameters limits their applicability in practice. Second, an exhaustive search is required for thermal system identification. However, because of the existence of noises, poor precision of samples, and low sampling rate of on-chip sensors, an enormously high-order model can be obtained, which leads to heavy intensive computation in running the models with estimated parameters. A reduced order model is particularly important to achieve an accurate and fast thermal model as the number of CPU cores in embedded SoCs increases.

In this paper, we present a thermal modeling scheme solved by the matrix exponential method, which overcomes the limitations of prior work. Our scheme can estimate the thermal parameters of the chip that represent its semiconductor technology and cooling package, without requiring any information about the applications running on the system. We propose a two-stage scheme that first extracts the thermal information from quantized and imprecise measurements of steady-state temperature profiles with on-chip temperature sensors and estimates relative location of CPU cores on chip floorplans. With the information from the first stage, the second stage gathers information from a limited amount of transient data to complete the estimation of the thermal parameters. This structured estimation of thermal parameters ensures the stability of and accuracy of our thermal model and reduces the order of model while filtering out noises from various sources.

III. System Model

We consider a homogeneous multi-core processor where each CPU core uses the same microarchitecture. Each core is assumed to have a dedicated temperature sensor that is accessible by the OS at runtime. This assumption can be easily met in many commercial processors such as Samsung Exynos and Intel Core processors. We assume that the following information is not available to use: the chip floorplan, the exact locations of on-chip temperature sensors, and the power traces of the processor. In the rest of this section, we introduce the power and temperature model used in this paper.

A. Power Model

The total power consumption of CMOS circuits at time t is modeled as the summation of dynamic and static powers [3], i.e., $P(t) = P_S(t) + P_D(t)$. Static power P_S depends on the semiconductor technology and the operating temperature caused by current leakage. Hence, it can be modeled as: $P_S(t) = k_1\theta(t) + k_2$, where k_1 and k_2 are technology-dependent system constants, and $\theta(t)$ is the operating temperature [25]. Dynamic power $P_D(t)$ is the amount of power consumption due to the processor operating frequency f at time t, modeled as $P_D(t) = k_0 f(t)^s$, where s and k_0 are the system constants that depend on semiconductor technology.

B. Temperature Model

We consider the temperature model widely used in real-time systems [1], [7], [13], [14], [22], [24], which follows the well-known linear time-invariant (LTI) model. Hence, the temperature model for a multi-core CPU with n cores is given by the following equation:

$$[\boldsymbol{\theta}'(t)]_{n\times 1} = \mathbf{A}_{n\times n}\ [\boldsymbol{\theta}(t)]_{n\times 1} + \mathbf{B}_{n\times n}\ [\mathbf{P}(t)]_{n\times 1} \quad (1)$$

where $\boldsymbol{\theta}(t)$ is the $n \times 1$ matrix of the CPU core operating temperatures relative to the ambient temperature, and $\mathbf{P}(t)$ is the power consumption of all cores at time t. $\mathbf{A}$ is an invariant $n \times n$ matrix and it is based on the characteristics of the semiconductor technology. It quantifies the effect of conduction between adjacent cores, convection among all cores, and a difference between ambient and operating temperature.

$\mathbf{B}$ is the diagonal $n \times n$ matrix and it captures the effect of power consumption on the temperature of each core. For homogeneous multi-core CPUs, since all CPU cores follow the same power model, the matrix $\mathbf{B}$ can be represented as $b \times \mathbf{I}$, where $\mathbf{I}$ is the $n \times n$ identity matrix. Similar to $\mathbf{A}$, the values of b are invariant to the changes in power consumption.

Hence, the problem will be estimating the values of the matrices $\mathbf{A}$ and $\mathbf{B}$ ($= b \times \mathbf{I}$) without any prior knowledge or direct measurement of CPU power consumption. We will show that it is impossible to estimate the value of b without having any knowledge of CPU power consumption; instead, we will estimate $\mathbf{B} \times \mathbf{P}$ which matters in calculating the temperature in the LTI model.

C. Problem Description

Given a multi-core CPU equipped with on-chip temperature sensors, construct an accurate and fast thermal RC model by estimating $\mathbf{A}$ and $\mathbf{B} \times \mathbf{P}$ of the CPU from a limited number of temperature profiles, without requiring a priori knowledge of the floorplan, cooling package, and power traces.

IV. Limitations and Errors

Before continuing our discussion, we must identify some limitations to thermal parameter estimation on real-life platforms. Identifying these potential sources of error is critical to our data analysis and comprehension. It allows us to address the noise and limitations of our profile data set by preprocessing raw data and improve the accuracy of thermal parameters through the ensemble of measurements from different frequency levels and workload settings.

The largest sources of error in the raw data are the on-chip temperature sensors. In CPUs for embedded systems, the sampling rate of temperature sensors is, generally, very low. This limitation may cause inaccuracies in data measurements since sudden changes in CPU-core utilization and their effects on operating temperature may go undetected by the sensors. Furthermore, the location of temperature sensors on each CPU-core may vary and may not coincide with the cpu-core's hotspot. Additionally, the data sampled from the on-chip temperature sensors is subject to quantization and, thus, limits the precision of measurement. This imprecision can be exacerbated due to the superposition law in thermal modeling for multi-core CPUs [13]. Moreover, due to differences in the construction and architecture of on-chip thermal sensors, their response time may vary. Hence, the sensor data may not represent the actual temperature if the CPU utilization changes in a relatively short time interval. Over time, if there is no fluctuation in CPU utilization, the sensor data will converge to the actual CPU core temperature. Therefore, we can assume that this type of error only affects the transient-state data while the steady-state data is unaffected.

Our thermal model considers temperatures relative to the ambient temperature. Although we assume that the ambient temperature remains invariant during profiling, in reality, it may change even in a room or a thermal furnace. The fluctuation of the ambient temperature while profiling can introduce noise into the raw data. Heat convection caused by room fans, air conditioners, or active temperature control in a furnace can also lead to noisy thermal profiles, by directly affecting the heat transfer between the device and the surrounding environment.

V. The proposed scheme

In this section, we introduce our scheme for thermal system identification of a given multi-core processor. The entire workflow of the proposed scheme is illustrated in Fig. 1. The very first step is profiling steady-state temperature data for a set of designed workloads. Then, the scheme removes noise from the raw data set and detects possible inconsistencies in thermal profiles (②), and performs the floorplan estimation (③). By using the estimated floorplan template and the collected steady-state data, the value of the matrix $\mathbf{A}$ is estimated in terms of the power parameters (④). The parameters $\mathbf{B} \times \mathbf{P}$ are then estimated by analyzing a subset of the transient-state data at the final stage (⑤). One of the reasons that we propose dividing analysis into the steady-state and the transient-state stages (④ and ⑤) is to cope with the various types of errors that may be introduced during temperature profiling. If one tries to tackle this problem by using both data at the same time, errors in the transient-state data can adversely affect the characterization of the system in steady state because the transient-state data has a much larger number of data points than the steady-state data. Moreover, our proposed scheme has a low computational cost as it requires processing only a few data points in the steady-state stage to obtain the thermal characteristics of the system.

A. Thermal Analysis and Steady-State Profiling

The proposed scheme primarily uses the steady-state data for thermal parameter estimation since it is more robust to measurement errors than the transient-state data but still contains the required information about semiconductor technology and power consumption. Hence, before introducing the detailed stages of our scheme, we analyze the thermal model and provide the reasoning behind the steady-state profiling.

Our goal is to estimate the thermal parameters related to the steady-state data. After solving the first order equation of Eq. 1, we have

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}_{chip} + e^{(t-t_0)\mathbf{A}}\boldsymbol{\theta}_0 + \int_{t_0}^{t} e^{(t-s)\mathbf{A}}\mathbf{B}\mathbf{P}(s)ds \qquad (2)$$

where $[\boldsymbol{\theta}_{chip}]_{n\times 1}$ is the total heat dissipation caused by IP blocks on the chip and idle power of CPU cores. We assume that all IPs generate a constant amount of heat. This term captures the heat conduction between all parts of the chip and the CPU. When the CPU cores are idle for a long time, $\boldsymbol{\theta}_{chip}$ can be measured. It is worth noting that due to the location of CPU cores and their sensors on the floorplan, the steady-state temperature of each idle core can be different. The second term is the homogeneous solution which is the thermal response due to the initial temperature difference from the ambient. The third term is the non-homogeneous solution caused by the input power signal.

For thermal profiling, our scheme collects thermal data while the CPU is running at a fixed frequency level. It is worth noting that this is only for profiling purposes and the thermal model found by our scheme can be used with DVFS policies. Given that we analyze the data collected at a fixed

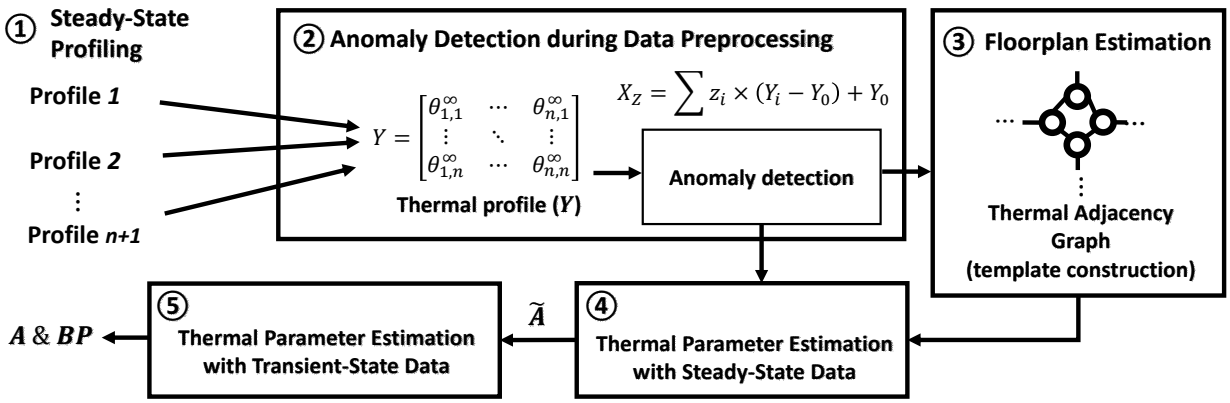


Figure 1: Proposed thermal system identification scheme.

CPU frequency, the total power can be considered as a function of temperature at any time instant t because the other factors remain invariant during task execution. For homogeneous multi-core CPUs, the power for each core is $P_S(t)$ when the CPU is idle, and $P_S(t)+P_D(t)$ when the CPU executes some workload and all cores are fully utilized. Hence, Eq. 2 can be written as:

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}_{chip} + e^{(t-t_0)\mathbf{A}}\boldsymbol{\theta}_0 - \mathbf{A^{-1}}\ (\mathbf{I} - \mathbf{e^{At}}\ )\mathbf{BP}. \quad (3)$$

In the steady state, the second term disappears because the steady-state operating temperature depends only on the power consumption (third term) and it is unaffected by initial temperature $\boldsymbol{\theta}_0$. Suppose $\boldsymbol{\theta}^\infty$ represents the operating temperature in the steady state, then:

$$\boldsymbol{\theta}^\infty = \lim_{t\to\infty} \boldsymbol{\theta}(t) = \boldsymbol{\theta}_{chip} - \mathbf{A^{-1}BP}. \quad (4)$$

We are interested in finding the value of matrices $\mathbf{A}$ and $\mathbf{BP}$. It is worth noting that the only control parameter is the power signal. It means that for each profiling procedure, it is possible to execute workload on any subset of CPU cores or hot-unplug them but the actual value of the power remains unknown. Let $[\mathbf{Y_i}]_{n\times 1}$ denote the operating temperature of the CPU cores when the i-th core is fully-utilized and $\mathbf{Y}_0$ represent the temperature of the CPU when all CPU cores are idle. Furthermore, let $[\mathbf{Y}]_{n\times n} = [\mathbf{Y_1 Y_2} \dots \mathbf{Y_n}]^T$ be the matrix of temperature profiles of the CPU in the steady state. For instance, $y_{i,j}$ in the row i and column j of $\mathbf{Y}$ is the temperature of the j-th CPU core when tasks are executing only on the i-th CPU core. Hence, according to Eq. 4,

$$\mathbf{Y} - [\mathbf{Y_0 Y_0} ... \mathbf{Y_0}]^T_{n\times n} = P_D\ (-\mathbf{A^{-1}B}). \quad (5)$$

We solve the equation to find the value of $\mathbf{A}$. Therefore,

$$\mathbf{A} = -P_D\ b\ (\mathbf{Y} - [\mathbf{Y_0 Y_0} ... \mathbf{Y_0}]^T)^{-1}. \quad (6)$$

By denoting $\tilde{\mathbf{A}} = (\mathbf{Y} - [\mathbf{Y_0 Y_0} ... \mathbf{Y_0}]^T)^{-1}$ and $\gamma = b\ P_D$, we have

$$\mathbf{A} = -\gamma\tilde{\mathbf{A}}. \quad (7)$$

Therefore, by estimating $\tilde{\mathbf{A}}$ and γ, we can determine the value of $\mathbf{A}$. $\tilde{\mathbf{A}}$ can be determined by profile without any knowledge of power consumption. The value of γ cannot be determined by the information from the steady-state profiles even with temperature profiles encompassing various CPU frequencies.

As shown in Eq. 7, we only need $n+1$ profiles to estimate the value of $\tilde{\mathbf{A}}$, i.e., one profile measured when all cores are off, and n profiles when each core is fully utilized one at a time. However, because of errors and limitations discussed in Section IV, there may be a considerable amount of noises in estimating the value of $\tilde{\mathbf{A}}$. If the profiles are noiseless, $\tilde{\mathbf{A}}$ will be a symmetric matrix with positive values on diagonal and non-positive values on non-diagonal elements. Zeros at $\tilde{a}_{i,j}$ of $\tilde{\mathbf{A}}$ represent that there is no heat conduction between cores i and j. It is noteworthy that a non-symmetric $\tilde{\mathbf{A}}$ caused by noisy data sets can lead to imaginary eigenvectors which is impossible in practice. We will later propose several methods to address different types of noises. Moreover, we will extend our analysis to reach a more precise $\tilde{\mathbf{A}}$ by fusing more data profiles. By using those techniques, it is not necessary to have the exhaustive combinations of $\mathbf{Y}_i$ for profiling purposes, but it is still possible to benefit from multiple auxiliary data obtained from the same core configuration.

One interesting property of $\mathbf{A}$ and $\tilde{\mathbf{A}}$ is that both have the same eigenvectors. Additionally the eigenvalues of $\mathbf{A}$ are $-\gamma$ times the eigenvalues of $\tilde{\mathbf{A}}$. We will use these properties to find the value of γ later and justify why it is impossible to estimate absolute power consumption from thermal profiling.

B. Anomaly Detection during Data Preprocessing

Compensating for various sources of errors in the steady-state data is critical to accurately estimate the system thermal parameters. For instance, the negative impact of limited sensor precision and transient noises can be reduced by applying a low-pass filter to each steady-state temperature profile before constructing an observation matrix. Beyond these basic methods, this section presents how our scheme detects inconsistencies and anomalies when multiple thermal profiles exist.

As we discussed in Sec. V-A, our scheme needs steady-state temperature profiles which are obtained when only one of the CPU cores is fully utilized at a time. If there are more profiled data, it is possible to detect if there is any inconsistency in some of the temperature profiles. For instance, if the ambient temperature in one profile is different from that in other profiles, it can be detected and rectified. Our post-processing error detection tests if the observed data $\mathbf{Y}_i$ are consistent with other auxiliary profiles that are obtained when

more than one CPU core is fully utilized. If any error is found from the observed data $\mathbf{Y}_i$, the corresponding column of $\mathbf{Y}$ will be rectified with the correct value.

We now present the details. Let $\mathbf{X_Z}$ denote the steady-state temperature of CPU cores. $\mathbf{Z} = \overline{z_1 z_2 \dots z_n}$ is the predicate that shows the status of CPU cores in the profile test where $z_i \in \{0, 1\}$ represents whether the i-th CPU core is fully utilized ($z_i = 1$) or not ($z_i = 0$). We are interested to test primary observed data Y_i by using auxiliary data $\mathbf{X}$s to detect the prospective error in Y. According to the thermal superposition theorem [13],

$$\mathbf{X}_Z = \sum_{i=1}^{n} z_i \times (\mathbf{Y}_i - \mathbf{Y}_0) + \mathbf{Y}_0. \tag{8}$$

Suppose that the available tests for a hypothetical 3-core CPU are $\mathbf{Y}_i$ for $i \in [1, 3]$ and the auxiliary test $\mathbf{X}_{\overline{101}}$. Hence, $\mathbf{X}_{\overline{101}} = \mathbf{Y}_1 + \mathbf{Y}_3 - \mathbf{Y}_0$. In the idle condition where there is no errors in the data, the equation much hold. Let's suppose there is an error data in one of them. We are interested to detect or correct it. By adding an error term ϵ to the equation, we have $\mathbf{X}_{\overline{101}} = \mathbf{Y}_1 + \mathbf{Y}_3 - \mathbf{Y}_0 + \epsilon_{1,3}$. If there is no error in data, one can assume that $\epsilon_{1,3} = [0, 0, 0]^T$. If there is an error in one of the tests, it can be detectable but not correctable. Now suppose that there is another test $\mathbf{X}_{\overline{011}}$ available, hence $\mathbf{X}_{\overline{011}} = \mathbf{Y}_2 + \mathbf{Y}_3 - \mathbf{Y}_0 + \epsilon_{2,3}$.

In order to detect and correct the error, we design a test in an n-hypercube format. Each corner of the hypercube represents one measurement setting $\mathbf{Z}$, and there is only a one-bit difference in the $\mathbf{Z}$ values of two neighboring corners. Therefore, the hamming distance of Zs of each pair of neighbor corners is 1. In this way, it is possible to examine the correctness of each test with its neighbors. It is also possible to spot the error with a sufficient number of tests.

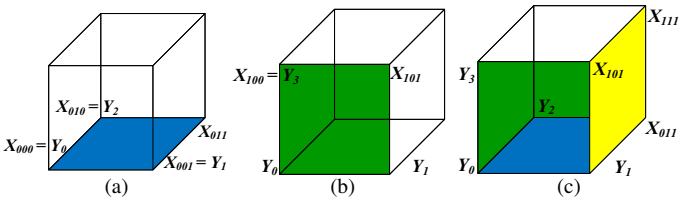


Figure 2: a) Auxiliary test for detect one error in either $\mathbf{Y}_1$ or $\mathbf{Y}_2$. b) Auxiliary test for detect one error in either $\mathbf{Y}_1$ or $\mathbf{Y}_3$. c) Second-tier testing of auxiliary tests.

We explain the procedure by making an example. Fig. 2a illustrates the auxiliary test $\mathbf{X}_{\overline{011}}$. The blue side shows the verification of the $\mathbf{Y}_1$ and $\mathbf{Y}_2$. If all the data are consistent, one can conclude that there is no single error in $\mathbf{Y}_1$ and $\mathbf{Y}_2$. If there is an error in one data an additional test is needed that we will explain later. The green side in Fig. 2b shows the verification for $\mathbf{Y}_1$ and $\mathbf{Y}_3$. If there is no single error, the data on this side is also consistent.

Now suppose that both sides are inconsistent, then it is easy to say that the data of $\mathbf{Y}_1$ is faulty because we assume that there is only one error in observation profiles and the edge of $(\mathbf{Y}_0, \mathbf{Y}_1)$ is the intersection of both sides. Since $\mathbf{Y}_0$ is zero base and not faulty, $\mathbf{Y}_1$ is noisy.

We now discuss the case where $\epsilon_{1,3} \neq 0$ and $\epsilon_{2,3} = 0$. To locate the error in the data, we use the auxiliary test that make a side with two of suspicious tests (the yellow side in Fig. 2c). In this case, we consider $\mathbf{X}_{\overline{111}}$ that can make a side with $\mathbf{X}_{\overline{001}}$, $\mathbf{X}_{\overline{011}}$ and $\mathbf{X}_{\overline{101}}$. If the data of this side is consistent, the error will belong to $\mathbf{Y}_1$, otherwise to the test $\mathbf{X}_{\overline{101}}$.

In generalization, any subset of verified steady-state temperature profiles $\mathcal{X} = \{X_{Z_1}, X_{Z_2}, \cdots, X_{Z_r}\}$ such that $Z_e = \oplus_{i=1}^{r} Z_i$ where $\oplus$ is the bitwise exclusive disjunction operator (XOR) can validate the data of the profile X_{Z_e}. By using it, the total number of auxiliary profiles with the hamming distance of 1 for a single anomaly detection is bounded by $\binom{n}{2} - 1$. For spotting the error, at most one extra profile with the hamming distance of 1 from two suspicious profiles is required. In fact, the profile that contains the error can be found by performing bitwise exclusive disjunction on predicates of inconsistent profiles. Only when there exist two inconsistent profiles, an extra auxiliary profile is required. The timing complexity of the proposed anomaly detection algorithm is $O(n^2)$ for detecting and correcting one single error.

By employing the same technique, it is easily possible to extend the proposed method for two errors in the tests by adding another tier to test the auxiliary tests with each other. Additionally, it is possible to conclude that there exists a single error or two errors in the data. It is important to mention that in this paper, it is assumed that errors in the two tests cannot conceal each other.

C. Floorplan Estimation

Hereby, we introduce a greedy algorithm to estimate the topography of the CPU cores in the chip. Later, we use this information to calibrate the thermal term $b\ P_D$ in the temperature model by using the transient-state data. One of the steps to refine the thermal parameters of a chip is to estimate the parameters complying with the floorplan. We present a mechanism to estimate the relative location of the CPU cores on a given chip.

The intuition behind our proposed algorithm is that the amount of heat dissipation from a heat source to a closer object is more than that to a farther object. Therefore, we expect that the temperature increase due to heat conduction of adjacent CPU cores is more than that of non-adjacent CPU cores. We introduce a fully-connected weighted graph where the edge weight represents the temperature increase of a CPU core when its neighboring cores are fully utilized. For instance, suppose there is a quad-core CPU where each core is labeled as C_i with $i \in [1, 4]$. The fully-connected graph of this CPU is illustrated in Fig. 3a. The weight of each edge, $w_{i,j}$, represents the temperature increase of a core C_j when another core C_i is fully utilized (i.e., $y_{i,j} - y_i^0$). It is worth noting that $w_{i,j} \neq w_{j,i}$ not only because of noise but also due to the location of the on-chip temperature sensor relative to the hotspot of each CPU core, although the amount of heat dissipation from each homogeneous core is ideally the same.

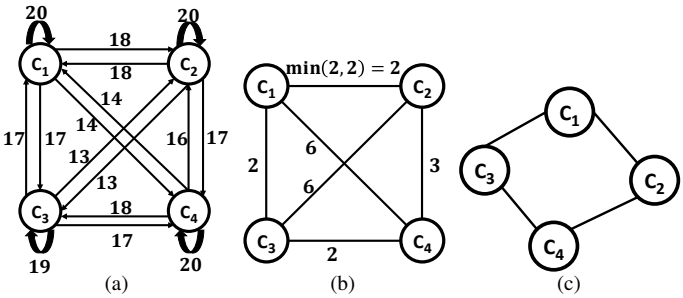


Figure 3: a) Example of adjacency graph of a quad-core CPU. b) Reduced uni-directed graph c) Final CPU affinity graph.

We are interested in the relative temperature increase of core C_i when it is fully utilized compared to when another core C_j is fully-utilized. Hence, we have $w'_{i,j} = y_{i,i} - y_{i,j}$. We also remove self-loops from the graph for simplicity. After this, the graph of Fig. 3 is reduced to a graph where all weights are positive. Next, we convert the bi-directed graph to a uni-directed graph by taking minimum of the edges in the different direction (i,e., $\min\{w'_{i,j}, w'_{j,i}\}$) as shown in Fig. 3b. It is noteworthy that one can use average, maximum or any reduction operation for this stage. As shown in Fig. 3b, the temperature increase of CPU core C_4 when the CPU core C_1 is fully utilized is because of heat conduction of CPU cores C_2 and C_3. In other words, the CPU core C_4 is heated up because of transitive heat transfer from C_1 to both C_2 and C_3 and then heat transfer occurs from these CPU cores to the CPU core C_4. Therefore, there is only transitive heat conductivity between the CPU core C_1 and the core C_4.

The floorplan estimation algorithm is shown in Algorithm 1. By using this graph (line 2-4) and also the aforementioned intuition, we estimate the relative location of the CPU cores on the floorplan. To do this end, we begin from one arbitrary core (line 9), let's say C_1, and find the minimum value of all weights connected to its corresponding node in the graph (line 29-31). In this example, the minimum weight is 2 for both CPU cores C_2 and C_3. Next, we find the minimum weight of CPU cores C_2 and C_3 from the unvisited node list, one at a time (line 16-25). We continue this step until all nodes are visited (line 8). If some of the visited CPU cores have the same value (within some error margin δ), they are connected. In this example, the error margin is 1 so the cores are connected as illustrated in Fig. 3c. The time complexity of this algorithm is $O(|E||V|)$ because each node is visited only once for its edges and the function `findMinEdges` can be $O(1)$ using precomputed tables or memoization.

Furthermore, if the temperature profiles of on-chip IPs such as an integrated GPU are available, the same approach can be applied to locate the corresponding IP by using the temperature increase when each CPU core is fully-utilized. In this way, there is only one edge from each CPU core C_i to the IP.

D. Thermal Parameter Estimation with Steady-State Data

We propose a method to calibrate the thermal parameters based on the floorplan estimation discussed in the previous section. The thermal parameter estimation without considering the floorplan may not be applicable in practice.[1] Some important properties of $\mathbf{A}$ according to heat transfer are as follows:

- $\mathbf{A}$ is a symmetric matrix.
- The elements on diagonal of $\mathbf{A}$ are negative.
- The zero values of non-diagonal elements represent a pair of corresponding CPU cores are not adjacent.
- All non-zero values of non-diagonal elements are positive.

These properties of $\mathbf{A}$ guarantee that there exist n real

Algorithm 1: Floorplan estimation algorithm

```
1  FloorPlanEst
       Input: Ã and error margin δ
       Output: G(V, E)
2      B = diag(Ã) * 1_{1,n} − Ã  /* 1 is the matrix of
        ones and diag gets diagonal elements of
        matrix                                    */
3      C = min(B,B^T)
4      Construct graph G' from C
5      Q = []  // empty queue
6      E = []  // edge set
7      V = []  // vertex set
8      while |V| ≠ |C| do
9          node = select randomly one unvisited node
            from G'
10         Q.push(node)
11         while |Q| > 0 do
12             node = Q.top()
13             node.visited = true
14             Q.pop()
15             neighbors = findMinEdges (node, δ)
16             foreach Node x ∈ neighbors do
17                 if x.visited = false then
18                     Q.push(x)
19                     V.insert(x)
20                     E.insert({node,x})
21                 end
22                 else if E.contains({node,x}) = false
                    then
23                     E.insert({node,x})
24                 end
25             end
26         end
27     end
28     return G(V, E)
29 Function findMinEdges(node, δ):
30     min = the minimum value of weights of edges
        with the source vertex node
31     return a list of nodes directly connected to node
        whose edges weights are less than min + δ
```

[1] This is because $\mathbf{Y}$ can be inaccurate and the properties of $\mathbf{A}$ may be violated if the floorplan is not considered.

negative eigenvalues and a set of n eigenvectors, one for each eigenvalue, that are mutually orthogonal. The negative values of all eigenvalues ensure the obtained thermal model is stable. This stage estimates the values of thermal parameters in way that it not only complies with the floorplan but also reduces the effect of noisy data observed data in $\mathbf{A}$.

The matrix $\tilde{\mathbf{A}}$ is constructed according to the estimated floorplan. If there is no thermal interference between two cores C_i and C_j in the estimated floorplan, the corresponding value in the matrix, i.e., $a_{i,j}$, is zero. A single value can be used for all adjacent CPU cores in the matrix $\tilde{\mathbf{A}}$ as they share the same heat dissipation increase value. It is worth noting that, depending on the user's accuracy demand, more distinct non-zero values may be used. For the explained example, the structure of matrix $\tilde{\mathbf{A}}$ will be

$$\tilde{A} = \begin{bmatrix} a_1 & a_5 & a_5 & 0 \\ a_5 & a_2 & 0 & a_6 \\ a_5 & 0 & a_3 & a_5 \\ 0 & a_6 & a_5 & a_4 \end{bmatrix}.$$

We propose a gradient descent algorithm to find the parameters according the estimated floorplan. For the calibration of the parameters, $\tilde{\mathbf{A}}$ is compared with the inverse of $\mathbf{Y}$, which means that $\tilde{\mathbf{A}}$ follows the estimated floorplan template and its inverse must be close to the temperature profiles. We propose the cost function as follows

$$\underset{a_i:\, i \in [1,r]}{\operatorname{argmin}} \; ||\tilde{\mathbf{A}}^{-1} - Y||_F^2 \tag{9}$$

where r is the number of unknown parameters. The sign of each parameter has to be carefully considered in the implementation, taking into account the properties of $\mathbf{A}$ discussed before (e.g., diagonal elements of $\tilde{A}$ should be positive and the rest be non-positive).

E. Thermal Parameter Estimation with Transient-State Data

In this section, we discuss how to estimate the matrix $\mathbf{B}$ for homogeneous multi-core platforms. As we mentioned in Sec. III-B, it is impossible to determine γ from the steady-state observations, hence, we must process the transient-state data.

As discussed in the steady-state section, if the power remains the same, the temperature equation will become Eq. 3. To estimate the value of $\mathbf{A}$, we only need γ which is commensurate to the power consumption. Substituting Eq.7 in Eq. 3, we have:

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}_{chip} + e^{(t-t_0)-\gamma\tilde{\mathbf{A}}}\boldsymbol{\theta}_0 + (\gamma\tilde{\mathbf{A}})^{-1}\,(\mathbf{I} - \mathbf{e}^{\mathbf{At}})\mathbf{bIP}. \tag{10}$$

Suppose that $\mathbf{V}$ and $\mathbf{\Lambda}$ are the eigenvectors and eigenvalues of $\tilde{\mathbf{A}}$, respectively. Therefore, $e^{(t-t_0)-\gamma\tilde{\mathbf{A}}}$ can be represented as $\mathbf{V}\; e^{(t-t_0)-\gamma\mathbf{\Lambda}}\; \mathbf{V}^{-1}$. From our proposed steady-state scheme, $\mathbf{V}$ and λ are determined. We can estimate the value of γ from the transient-state data of a singular observation. To better calibrate γ, multiple profiles may also be considered. Hence,

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}_{chip} + e^{-(t-t_0)\gamma\tilde{\mathbf{A}}}\boldsymbol{\theta}_0 + \tilde{\mathbf{A}}^{-1}\,(\mathbf{I} - \mathbf{e}^{\mathbf{A(t-t_0)}})\mathbf{H} \tag{11}$$

where $\mathbf{H}$ is an $n \times 1$ control signal whose i-th element is $h_i \in \{0, 1\}$. It is noteworthy that $h_i = 1$ when i-th core is fully utilized. By substituting the eigenvalues and eigenvectors, the equation can be represented as $\boldsymbol{\theta}(t) = \boldsymbol{\theta}_{chip} + V\; e^{-(t-t_0)\gamma\mathbf{\Lambda}}\; V^{-1}\boldsymbol{\theta}_0 + \tilde{\mathbf{A}}^{-1}\,(\mathbf{I} - \mathbf{V}\,\mathbf{e}^{-(\mathbf{t}-\mathbf{t_0})\gamma\mathbf{\Lambda}}\,\mathbf{V}^{-1})\mathbf{H}$. The only missing part in the temperature model is γ which can be estimated by curve fitting on transient-state temperature. The values of $P_d(t)$, $\boldsymbol{\theta}_{chip}$ and $\tilde{\mathbf{A}}$ change at different frequency levels but the values of $\mathbf{\Lambda}$, $\mathbf{V}$ and b remain invariant against frequency change. Although the value of γ can be estimated and is embedded in the temperature data, it is impossible to estimate absolute power even with temperature information at different frequency levels.

Since γ is a scalar value, we observed that curve fitting on only a few cores can provide good results. The simplest test for estimating the value of γ is when all cores are cooling down because the third term of Eq. 11 can be eliminated.

VI. Accuracy Enhancement

In this section, we extend our proposed scheme to improve accuracy using additional steady-state data from different core settings and extra data observed at different frequencies.

A. Use of Steady-State Data Ensembles

We discuss how to use the ensemble of extra observations at one frequency level to obtain a more accurate thermal observation profile. As discussed in the previous section, our method requires $n + 1$ observed steady-state temperature profiles for a n-core system to construct the matrix $\mathbf{Y}$: one profile when all cores are idle and n profiles when each of CPU core is only fully-utilized. Now, we extend our analysis to answer the following questions:

- Is it possible to collect different CPU core usage settings (other than the aforementioned $n + 1$ observed data) to construct $\mathbf{Y}$?
- Is it possible to have a more precise $\mathbf{Y}$ if there exist multiple instantiations from an identical setting and use all of them?
- Is it possible to construct $\mathbf{Y}$ with more than $n + 1$ steady-state data?

We are interested in generalizing the construction of $\mathbf{Y}$ to include any possible thermal traces of fully-utilized CPU core combinations and the ensemble of more thermal traces. It is noteworthy that if there is no noise in the observed data and also data is not quantized, no extra data or multiple instantiations are needed due to superposition law in Eq. 8.

Now, suppose we have m profiles such that $n \leq m \leq 2^n - 1$ and each profiling was done under a different CPU core setting Z_i. Based on that, we can construct the predicate matrix $[\mathbf{D}]_{m\times n} = [\mathbf{Z}_1\mathbf{Z}_2 \ldots \mathbf{Z}_m]^T$ from the auxiliary profiles in $\mathbf{U} = [\mathbf{X}_{z_1}\mathbf{X}_{z_2} \ldots \mathbf{X}_{z_m}]^T$. The matrix $\mathbf{Y}$ is then generated as follows

$$\mathbf{Y} = ((\mathbf{D} \times \mathbf{D}^T)^{-1} \times \mathbf{D} \times \mathbf{U}^T)^{-1}. \tag{12}$$

Because of the inversion in the equation, the equation works when the number of the profiles is more than the number of CPU cores. It also works when there exist enough orthogonal profiles, meaning that there is at least one profile for each CPU core where the core is idle. The matrix $\tilde{\mathbf{A}}$ is then calculated as explained in Sec. V-D.

B. Use of Multi-Frequency Data Ensembles

We now discuss how to obtain a more accurate $\mathbf{A}$ by using additional data collected at multiple frequency levels. As explained in Sec. III-B, the thermal parameter $\mathbf{A}$ is dependent on the semiconductor technology and remains invariant against frequency changes. Therefore, it would be logical to assume that having observed temperature profiles from different frequency levels would lead to an identical $\mathbf{A}$. However, due to the term γ in Eq. 7, $\tilde{\mathbf{A}}$ is proportional to the power consumption and the clock frequency of CPU cores. Based on this, we extend our scheme to answer the following questions:

- Is it possible to have an identical $\mathbf{A}$ from different Ys collected at different frequency levels?
- Is it possible to estimate relative power consumption at different frequency levels?

To answer these questions, we propose a thermal parameter estimation approach based on multi-frequency data ensembles. Let $\mathbf{Y}^i$ denote the temperature increase matrix constructed from the data when the CPU frequency is f_i, and γ_i denote its power effect on temperature at the frequency f_i. Unlike the method presented in Sec. V-D which estimates the parameters in $\tilde{\mathbf{A}}$, we will estimate $\tilde{\mathbf{A}}_1$, which is the base at f_1. Additionally, we consider $\gamma_1 = 1$ and estimate $\frac{\gamma_i}{\gamma_1}, \forall i > 1$. In such case, tracking the values of γ over different frequency levels gives the power consumption of CPU cores proportional to the γ_1 of the base frequency level f_1, and all $\mathbf{Y}$s share the same thermal parameter $\mathbf{A}$. Using the same procedure as in Sec. V-E allows estimating the value of γ_1; hence, all γs can be estimated. It is noteworthy that, even in this case, the actual value of the power consumption cannot be computed, but the relative power consumption at different frequency levels can be obtained. Therefore, one can see the effect of power consumption embedded in the temperature profiles.

We change the cost model of Eq. 9 to

$$\underset{\substack{a_j:\, j \in [1,r] \\ \frac{\gamma_i}{\gamma_1}:\, i \in [1,|f|]}}{\operatorname{argmin}} \sum_{1}^{|f|} ||\tilde{\mathbf{A}_1}^{-1} - \frac{\gamma_i}{\gamma_1}\mathbf{Y}^i||_F^2 \tag{13}$$

The problem is estimating the values of $\tilde{\mathbf{A}}_1$ and also $\frac{\gamma_i}{\gamma_1}$. One can expect that $f_i > f_j$ leads to $\gamma_i < \gamma_j$ due to dynamic power increase and this can be considered as a constraint in the implementation.

VII. Evaluation

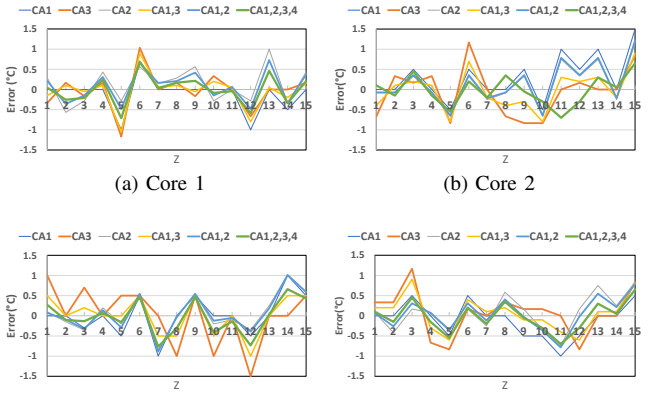
This section presents the experimental evaluation of our proposed scheme on a real embedded platform. We first evaluate the accuracy and validity of our techniques in thermal modeling and temperature prediction. We then conduct a case study to demonstrate the practical effectiveness and implications of our work in the context of real-time mixed-criticality systems (MCS).

A. Platform

We performed our experiments on an ODroid-XU4 development board [28] equipped with a Samsung Exynos5422 SoC based on the ARM big.LITTLE architecture. The Exynos CPU package contains two different quad-core CPU clusters of little Cortex-A7 and big Cortex-A15 cores. On-chip temperature sensors with a sampling rate of 10 Hz and precision of 1° C are available for each big CPU core as well as the GPU to measure the operating temperature[2]. The DTM throttles the frequency of the entire big CPU cluster to 900 MHz when one of its cores exceeds the hardware-defined maximum temperature threshold of 95° C. There is no active or passive cooling mechanism enabled on the CPU. The big CPU cluster frequency can be dynamically adjusted within the range of $[0.2, 2.0]$ GHz. However, for each experiment, it was pinned at a fixed frequency in the range of $[0.7, 1.4]$ GHz to avoid thermal violations that have occurred when all CPU cores run fully-utilized beyond 1.4 GHz in our environment where ambient temperature is regulated at around 21° C.

Table I: Descriptions of steady-state traces on big cores.

Case name	Experiments	Traces (decimal)	# of traces
$CA1$	one core	z_1, z_2, z_4, z_8	4
$CA3$	three cores	$z_7, z_{11}, z_{13}, z_{14}$	4
$CA2$	two cores	$z_3, z_5, z_6, z_9, z_{10}, z_{12}$	6
$CA1,3$	one core & three cores	$z_1, z_2, z_4, z_7, z_8, z_{11}, z_{13}, z_{14}$	8
$CA1,2$	one core & two cores	$z_1, z_2, z_3, z_4, z_5, z_6, z_8, z_9, z_{10}, z_{12}$	10
$CA1,2,3,4$	all cores	$z_1, z_2, \ldots, z_{15}$	15



(a) Core 1 (b) Core 2

(c) Core 3 (d) Core 4

Figure 4: The error of steady-state temperature of CPU cores by using different cases in construction of Y.

B. Anomaly Detection and Steady-State Data Ensembles

We evaluate our scheme in improving the accuracy of thermal parameter estimation by using the ensemble of steady-state profiles from multiple frequency levels. We apply the superposition theorem as in Eq. 8 to estimate the steady-state temperature of CPU cores when different subsets of them are fully-utilized. We use the data collected from a subset of all possible combinations of CPU cores at the fixed clock

[2]There is no temperature sensor for little cores since the power consumption and heat dissipation of the little cluster is substantially lower than that of the big cluster.

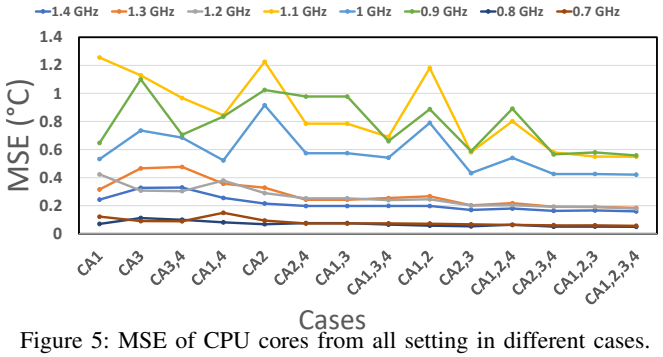


Figure 5: MSE of CPU cores from all setting in different cases.

frequency of 1.4 GHz.[3] The details on the subsets used to construct the matrix $\mathbf{Y}$ are given in Table I. Some cases contain the least number of orthogonal profiles (i.e., 4 profiles) while others contain more profiles. The case $CA1,2,3,4$ includes all profiles to construct the matrix $\mathbf{Y}$. The name of each case indicates the number of fully-utilized CPU cores. For instance, $CA1,2$ includes four profiles of one CPU core and six profiles of two cores.

Fig. 4 depicts the error in steady-state temperature of each CPU core under different utilization scenarios ($\mathbf{Y}$ from Eq. 12). As shown in the figure, using more profiles helps reduce the effect of noise in constructing $\mathbf{Y}$. The proposed anomaly detection mechanism presented in Sec. V-B was applied to the profiles and the outliers were excluded from consideration for $\mathbf{Y}$. For instance, applying this mechanism detected an anomaly in the profile $\mathbf{Z}_6$ by comparing it with the other profiles. Hence, while constructing $\mathbf{Y}$, we were able to find out that the anomaly was not because of an error in the other profiles but rather due to the error in $\mathbf{Z}_6$. The mean square errors (MSE) of the temperature model for all CPU cores under all different settings are shown in Fig. 5. The x axis is sorted in ascending order in terms of the number of profiles used during the construction of $\mathbf{Y}$. Some spikes in the results, e.g., the yellow line at $CA1,2$, are due to that additional traces contained noisy data. However, the error generally decreases as more profiles are used for the construction of $\mathbf{Y}$. This trend indicates that our proposed approach to considering data ensembles can reduce the negative impact of noisy data and improve the accuracy of thermal parameter estimation.

C. Floorplan Estimation

We validate our proposed floorplan estimation method on the Exynos5422 SoC. We draw the proposed adjacency graph according to the data collected at 1.4 GHz. Figures 6a-c show the steps to construct the final adjacency graph for Exynos5422. As we observe in Fig. 6c, the cores C_3 and C_4 have greater spatial proximity than cores C_3 and C_2. Although the physical layout of the cores on the CPU package is bi-symmetrical, the primary reason for the asymmetrical layout shown in Fig. 6c is that the location of the on-chip

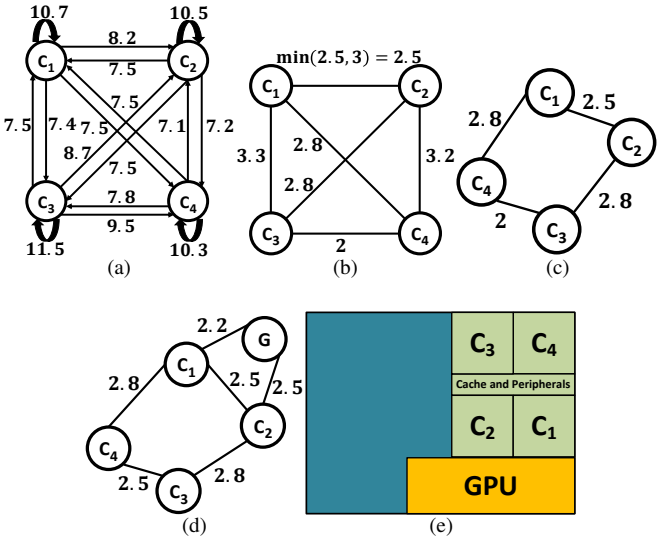


Figure 6: Floorplan estimation of Exynos5422 based on data of 1.4 GHz. (a) The fully-connected graph from the temperature increase data, (b) Graph reduction stage (c) The CPU affinity graph, (d) Estimation of GPU location relative to CPU location, and (e) The actual Exynos5422 floorplan [11].

temperature sensors on each processor may vary between cores. Additionally, the L2 cache and peripheral controllers may have an effect on the modeled spatial proximity between the core pairs of C_1, C_2 and C_3, C_4 . Using the same approach and GPU temperature data, we are also able to locate the embedded GPU by profiling the heat conduction between each CPU cores and the GPU. Fig. 6d depicts the final estimated locations of the embedded GPU and CPU cores on Exynos5422. Since this is similar to the actual layout reported in [11][4], we conclude that our floorplan estimation can provide a sufficient level of accuracy for thermal parameter calibration (Sec. V-D).

We construct the template of the matrix $\mathbf{A}$ to be compatible with the estimated floorplan. The matrix $\tilde{\mathbf{A}}$ at 1.4 GHz is then computed by using the steady-state data of $CA1,2,3,4$ at different frequency levels:

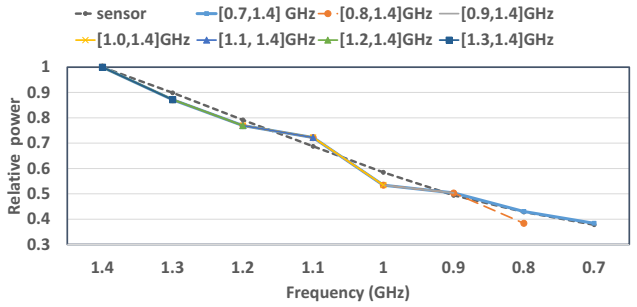
$$\tilde{\mathbf{A}} = \begin{bmatrix} 0.2961 & -0.1324 & 0 & -0.1194 \\ -0.1324 & 0.3017 & -0.1579 & 0 \\ 0 & -0.1579 & 0.3088 & -0.1269 \\ -0.1194 & 0 & -0.1269 & 0.2798 \end{bmatrix}.$$

D. Relative Power Estimation

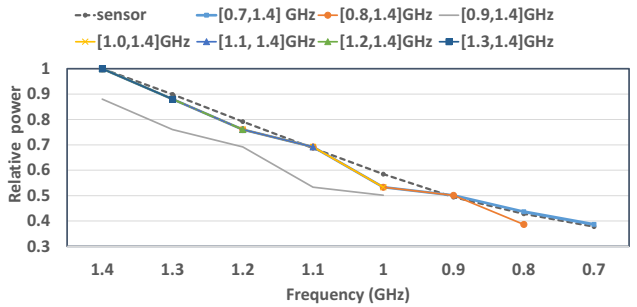
The relative power consumption of each CPU core can be estimated while estimating $\tilde{\mathbf{A}}$, as explained in Sec. VI-B. Fig. 7 illustrates the estimated power consumption. The results are obtained using profiles from different frequency ranges. As depicted in the figure, the estimated relative power closely follows the actual data collected from the built-in power sensors of the XU3 board that is equipped with the same Exynos5422 SoC.

[3]Note that the total number of core combinations is 2^4 in a quad-core system and the number of selecting a subset from the combinations is 2^{2^4}.

[4]The CPU core labeling in [11] is different from the labels in the driver and it is verified with infrared imaging captured by an FLIR A325sc IR camera [9].



(a)



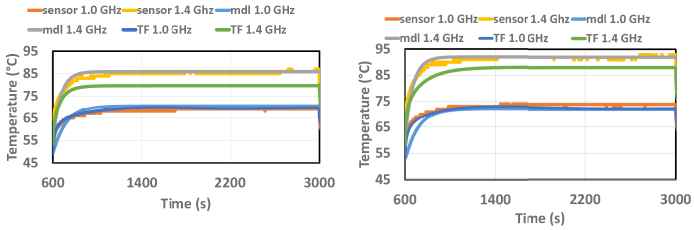
(b)

Figure 7: Power data of CPU cores in Exynos5422. (a) Comparison between the estimated relative power consumption and the normalized actual power data from built-in power sensors for $CA1$, and (b) Comparison for $CA1, 2, 3, 4$.

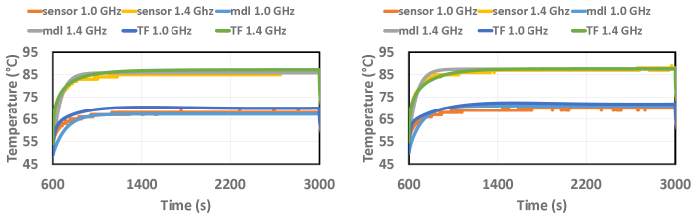
E. Temperature Prediction

We estimate the parameter γ_1 as the base parameter for the clock frequency 1.4 GHz in our experiments. Based on this, the absolute value of other γs can be determined. As discussed in Sec. V-E, we use the transient-state trace when all CPU cores are cooling down. After estimating the values of γs, we apply our model to predict the operating temperature of CPU cores.

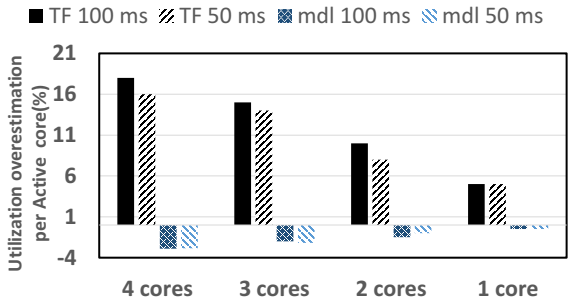
We evaluated our proposed scheme against the state-of-the-art power-agnostic thermal modeling method given in [1] (referred to as "TF"), which is based on the thermal fingerprint and calibration techniques [31], [32]. For the thermal profiling of TF, we left the CPU idle for 10 minutes before capturing the transient data of each thermal profile so that their model can estimates the initial states of transfer functions accurately. Next, we captured temperature data for all cores while each core was individually being fully-utilized for 30 minutes and then idle for 10 minutes. As in our scheme, a low-pass filter was applied to the raw data to reduce noise. This step gives us the idle steady state, active steady state, and transient state temperatures for every core. We performed the above steps when CPU operates at 1.0 GHz and 1.4 GHz. Then, we used Matlab's tfest function to estimate the self-core transfer functions [1] that represent the thermal response of each CPU core as a function of CPU core utilization. Finally, we performed an exhaustive search to tune the number of poles and zeros required for more accurate estimation of the transfer functions. The modeling results of TF have the average goodness fit of 89.96% at 1.0 GHz and 88.31% at 1.4 GHz,



(a) core 1 (b) core 2



(c) core 3 (d) core 4



(e)

Figure 8: (a-d) CPU temperature from sensor, our model, and TF when three cores are fully utilized. (e) Utilization overestimation error of server period of 50 and 100 ms at frequency level of 1.4 GHz.

which is reasonable given that the sensor precision is 1° C.

Fig. 8a-d depict the CPU temperatures when three CPU cores are fully utilized. The legends "sensor", "mdl", and "TF" refer to actual temperature collected from CPU sensors, the estimated temperature by our proposed scheme, and the state-of-the-art [1], respectively. Our proposed model can estimate the temperature values more accurately than TF especially when the frequency is higher (see the steady-state temperature of cores 1 and 2 at 1.4 GHz).[5] Since the precise estimation of the steady-state temperature is the key to ensure thermal safety, we expect that our proposed scheme can be effectively used in the thermal-aware design of COTS-based real-time systems.

One of the most important metrics during system design is the maximum achievable utilization (MAU) while the system remains thermally safe (i.e., a higher utilization than MAU leads to thermal violations). For instance, when designing a system using thermal-aware periodic servers [1], [14], [13], an underestimated steady-state temperature is particularly danger-

[5]We also observe that the underestimation error of TF increases with the number of fully-utilized CPU cores but demonstrate only the three-core case due to space limit.

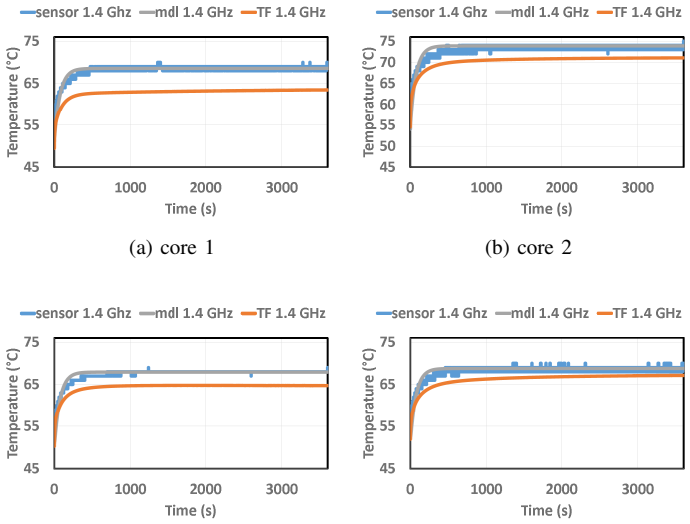
ous since it can lead to an overestimation of server budget, which in turn jeopardizes thermal safety. The amount of overestimation depends on temperature estimation accuracy, the number of active CPU cores, and the server settings used (i.e., period and budget replenishment policy). Hereby, we show in Fig. 8e the overestimated server utilization due to temperature estimation error under TF and our model when different number of CPU cores are active. The budget replenishment period of 50 ms and 100 ms and the polling server policy are used. In this figure, anything higher than zero means thermally unsafe. TF results in a large overestimation of MAU (up to 18% per core), and this trend increases with active core counts due to the thermal superposition. On the other hand, our scheme does not make any overestimation, so it is thermally safe. The underestimated MAU from our scheme is only less than 3% per core.

F. Case Study on MCS Application

We performed a case-study to evaluate the performance of our model for a mixed-critical Flight Management System (FMS) [1]. All low-criticality tasks are assigned to a polling server with a replenishment budget of 50 milliseconds and utilization of 50% on CPU core 1. High-criticality tasks are partitioned using worst-fit bin packing and executed on cores 2 and 4 using thermal-aware periodic servers [1], [14], [13] with a replenishment budget of 50 ms and utilization of 65%. Since TF assumes static scheduling and no preemption between servers, we used only one server per core for both ours and TF. All OS tasks are isolated on the little CPU cluster and CPU core 3 is idle. We carry on the experiment at CPU frequency of 1.4 GHz. As shown in Fig. 9, our scheme can estimate the temperature with a marginal error of 0.5° C while the TF method estimates the temperature with error that approaches 5.3° C. The ratio of the error to the temperature increase from the idle steady state is 23.07% for TF and 2.53% for our scheme. This large improvement in accuracy enables a thermally-safe system design (recall the impact of accuracy in utilization discussed in Sec. VII-E) and demonstrates the effectiveness of our work. Also, our scheme allows real-time tasks and thermal-aware servers to be scheduled preemptively, which TF cannot do.

VIII. Conclusion

In this paper, we proposed a novel and accurate scheme to estimate the thermal parameters of COTS multi-core processors for real-time embedded systems. By decomposing our estimation scheme into steady-state and transient-state stages, we substantially reduce the number of transient-state profiles needed to estimate the system's thermal parameters. We presented methods to improve the accuracy of our scheme by utilizing additional temperature profiles in the parameter estimation. Our proposed scheme is fast to converge and has a low computational cost for the prediction of chip operating temperature. Hence, it can be even used in an event-driven manner, e.g., at the arrival and the departure of each job of



(a) core 1 (b) core 2

(c) core 3 (d) core 4

Figure 9: CPU temperature from sensor, our model, and TF for FMS application at 1.4 GHz.

periodic real-time tasks, with negligible memory and computational overhead. We derived the thermal characteristics of a multi-core processor which remain unchanged across different frequency levels. We also showed the effectiveness of our scheme in extracting the relative power consumption information from temperature profiles.

There are several interesting directions for future work. Our scheme can be extended to identify the thermal models of SoCs under various cooling conditions or those with heterogeneous processor units. One may consider using statistical approaches to estimate the floorplan of SoCs rather than the fixed error margin used in this work. A formal mathematical analysis to quantify the robustness of data-driven thermal modeling against noisy profiles is an important research issue and will be a valuable addition to our work.

Acknowledgment

This work is in part supported by the National Science Foundation (NSF) grant 1943265.

References

[1] R. Ahmed, P. Huang, M. Millen, and L. Thiele. On the design and application of thermal isolation servers. *ACM Trans. Embed. Comput. Syst.*, 16(5s):165:1–165:19, Sept. 2017.

[2] R. Ahmed, P. Ramanathan, and K. K. Saluja. Necessary and sufficient conditions for thermal schedulability of periodic real-time tasks under fluid scheduling model. *ACM Transactions on Embedded Computing Systems*, 15(3):49, 2016.

[3] T. Chantem, R. P. Dick, and X. S. Hu. Temperature-aware scheduling and assignment for hard real-time applications on mpsocs. In *2008 Design, Automation and Test in Europe*, pages 288–293, March 2008.

[4] T. Chantem, X. S. Hu, and R. P. Dick. Temperature-aware scheduling and assignment for hard real-time applications on mpsocs. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 19(10):1884–1897, 2010.

[5] K. Choi, R. Soma, and M. Pedram. Fine-grained dynamic voltage and frequency scaling for precise energy and performance tradeoff based on the ratio of off-chip access to on-chip computation times. *IEEE transactions on computer-aided design of integrated circuits and systems*, 24(1):18–28, 2004.

[6] A. Das, A. Kumar, and B. Veeravalli. Reliability and energy-aware mapping and scheduling of multimedia applications on multiprocessor systems. *IEEE Transactions on Parallel and Distributed Systems*, 27(3):869–884, 2015.

[7] S. M. D'Souza and R. Rajkumar. Thermal implications of energy-saving schedulers. In *ECRTS*, 2017.

[8] T. J. A. Eguia, S. X. . Tan, R. Shen, E. H. Pacheco, and M. Tirumala. General behavioral thermal modeling and characterization for multi-core microprocessor design. In *2010 Design, Automation Test in Europe Conference Exhibition (DATE 2010)*, pages 1136–1141, 2010.

[9] FLIR A325sc. https://www.flir.com/products/a325sc, 2019.

[10] Y. Fu, N. Kottenstette, Y. Chen, C. Lu, X. D. Koutsoukos, and H. Wang. Feedback thermal control for real-time systems. In *2010 16th IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 111–120. IEEE, 2010.

[11] Y. H. Gong, J. J. Yoo, and S. W. Chung. Thermal modeling and validation of a real-world mobile ap. *IEEE Design Test*, 35(1):55–62, Feb 2018.

[12] S. Herbert and D. Marculescu. Analysis of dynamic voltage/frequency scaling in chip-multiprocessors. In *Proceedings of the 2007 international symposium on Low power electronics and design (ISLPED'07)*, pages 38–43. IEEE, 2007.

[13] S. Hosseinimotlagh, A. Ghahremannezhad, and H. Kim. On dynamic thermal conditions in mixed-criticality systems. In *2020 IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 336–349, 2020.

[14] S. Hosseinimotlagh and H. Kim. Thermal-aware servers for real-time tasks on multi-core gpu-integrated embedded systems. In *2019 IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 254–266, 2019.

[15] Y. Hua and T. K. Sarkar. Generalized pencil-of-function method for extracting poles of an em system from its transient response. *IEEE transactions on antennas and propagation*, 37(2):229–234, 1989.

[16] H. Huang, V. Chaturvedi, G. Quan, J. Fan, and M. Qiu. Throughput maximization for periodic real-time systems under the maximal temperature constraint. *ACM Trans. Embed. Comput. Syst.*, 13(2s):70:1–70:22, Jan. 2014.

[17] P.-S. Huang, Q.-C. Chen, C.-W. Huang, and S.-L. Tsao. An efficient thermal estimation scheme for microprocessors. In *2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 1–10. IEEE, 2014.

[18] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan. Hotspot: A compact thermal modeling methodology for early-stage vlsi design. *IEEE Transactions on VLSI systems*, 14(5):501–513, 2006.

[19] R. Jayaseelan and T. Mitra. Temperature aware task sequencing and voltage scaling. In *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*, pages 618–623. IEEE Press, 2008.

[20] P. Kumar and L. Thiele. System-level power and timing variability characterization to compute thermal guarantees. In *Proceedings of the seventh IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, pages 179–188. ACM, 2011.

[21] O. Kwon, W. Jang, G. Kim, and C. Lee. Accurate thermal prediction for nans (n-app n-screen) services on a smart phone. In *2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES)*, pages 1–10, 2018.

[22] Y. Lee, H. Chwa, K. G. Shin, and S. Wang. Thermal-aware resource management for embedded real-time systems. In *Embedded Software (EMSOFT), 2018 International Conference on*. IEEE, 2018.

[23] Y. Lee, E. Kim, and K. G. Shin. Efficient thermoelectric cooling for mobile devices. In *2017 IEEE/ACM International Symposium on Low Power Electronics and Design*, pages 1–6. IEEE, 2017.

[24] Y. Lee, K. G. Shin, and H. S. Chwa. Thermal-aware scheduling for integrated cpus–gpu platforms. *ACM Transactions on Embedded Computing Systems (TECS)*, 18(5s):1–25, 2019.

[25] Y. Liu, R. P. Dick, L. Shang, and H. Yang. Accurate temperature-dependent integrated circuit leakage power estimation is easy. In *2007 Design, Automation Test in Europe Conference Exhibition*, pages 1–6, April 2007.

[26] Y. Ma, T. Chantem, X. S. Hu, and R. P. Dick. Improving lifetime of multicore soft real-time systems through global utilization control. In *Proceedings of the 25th edition on Great Lakes Symposium on VLSI*, pages 79–82. ACM, 2015.

[27] P. Michaud and Y. Sazeides. Atmi: analytical model of temperature in microprocessors. In *Third Annual Workshop on Modeling, Benchmarking and Simulation (MoBS)*, volume 2, page 7, 2007.

[28] ODROID-XU4. http://www.hardkernel.com/, 2016.

[29] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel. Matex: Efficient transient and peak temperature computation for compact thermal models. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1515–1520. IEEE, 2015.

[30] F. Paterna and T. S. Rosing. Modeling and mitigation of extra-soc thermal coupling effects and heat transfer variations in mobile devices. In *2015 IEEE/ACM International Conference on Computer-Aided Design*, pages 831–838, Nov 2015.

[31] D. Rai and L. Thiele. A calibration based thermal modeling technique for complex multicore systems. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1138–1143. IEEE, 2015.

[32] D. Rai, H. Yang, I. Bacivarov, and L. Thiele. Power agnostic technique for efficient temperature estimation of multicore embedded systems. In *Proceedings of the 2012 international conference on Compilers, architectures and synthesis for embedded systems*, pages 61–70, 2012.

[33] M. Rapp, O. Elfatairy, M. Wolf, J. Henkel, and H. Amrouch. Towards nn-based online estimation of the full-chip temperature and the rate of temperature change. In *Proceedings of the 2020 ACM/IEEE Workshop on Machine Learning for CAD*, pages 95–100, 2020.

[34] O. Sahin and A. K. Coskun. Providing sustainable performance in thermally constrained mobile devices. In *2016 14th ACM/IEEE Symposium on Embedded Systems For Real-time Multimedia (ESTIMedia)*, pages 1–6, Oct 2016.

[35] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers. The impact of technology scaling on lifetime reliability. In *Dependable Systems and Networks, 2004 International Conference on*, pages 177–186. IEEE, 2004.

[36] R. Viswanath, V. Wakharkar, A. Watwe, V. Lebonheur, et al. Thermal performance challenges from silicon to systems. *Intel Technology Journal*, Q3, 2000.

[37] Y. Wang, K. Ma, and X. Wang. Temperature-constrained power control for chip multiprocessors with online model estimation. *ACM SIGARCH computer architecture news*, 37(3):314–324, 2009.

[38] S. Zhang and K. S. Chatha. Thermal aware task sequencing on embedded processors. In *Proceedings of the 47th Design Automation Conference*, pages 585–590. ACM, 2010.

[39] A. Ziabari, J.-H. Park, E. K. Ardestani, J. Renau, S.-M. Kang, and A. Shakouri. Power blurring: Fast static and transient thermal analysis method for packaged integrated circuits and power devices. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(11):2366–2379, 2014.