General Parameterized Thermal Modeling for High-Performance Microprocessor Design

Thom J. Eguia, Sheldon X.-D. Tan, *Senior Member, IEEE*, Ruijing Shen, *Student Member, IEEE*, Duo Li, *Student Member, IEEE*, Eduardo H. Pacheco, Murli Tirumala, and Lingli Wang, *Member, IEEE*

***Abstract*—This paper proposes a new parameterized dynamic thermal modeling algorithm for emerging thermal-aware design and optimization for high-performance microprocessor design at architecture and package levels. Compared with existing behavioral thermal modeling algorithms, the proposed method can build the compact models from more general transient power and temperature waveforms used as training data. Such an approach can make the modeling process much easier and less restrictive than before and, thus, more amenable for practical measured data. The new method, called *ParThermSID*, consists of two steps. First, the response surface method based on second-order polynomials is applied to build the parameterized models at each time point for all of the given sampling nodes in the parameter space. Second, an improved subspace system identification method, called *ThermSID*, is employed to build the discrete state space models, by construction of the Hankel matrix and state space realization, for each time-varying coefficient of the polynomials generated in the first step. To overcome the overfitting problems of the subspace method, the new method employs an overfitting mitigation technique to improve model accuracy and predictive ability. Experimental results on a practical quad-core microprocessor show that the generated parameterized thermal model matches the given data very well. The compact models generated by *ParThermSID* also offer two orders of magnitude speedup over the commercial thermal analysis tool *FloTHERM* on the given example. The results also show that *ThermSID* is more accurate than the existing *ThermPOF* method.**

***Index Terms*—Components, packaging and manufacturing technology, electronic packaging thermal management.**

I. Introduction

As VLSI technology is scaled into the nanometer region, the power density of high-performance microprocessors increases drastically. The exponential power density increase will, in turn, lead to average chip temperature to raise rapidly [1]. Higher temperature has significant adverse impacts on chip packaging cost, performance, and reliability. Excessive on-chip temperature leads to slower transistor speed owing to reduced carrier mobility, more leakage power consumption as leakage currents grow exponentially with temperature, higher interconnect resistance, and reduced reliability [8], [3].

The estimated temperature at the architecture level is vital for performing accurate power (especially leakage power), performance, reliability, wear-out, and aging analysis in the floor planning and packaging design [20]. As a result, design guided by temperature can be optimized theoretically without potential thermal problems. For the cycle-accurate architecture thermal simulation, the simulation time can be very long (several seconds) [14], [21]. For instance, for a 3-GHz CPU, 10 K clock cycles (typically used) is 3.3 μs. For 10 s, the number of time steps is 3 million. Although the simulation techniques have seen some progress recently [4], more efficient thermal simulators are still highly desired. To facilitate this temperature-aware architecture design, it is important to have accurate and fast thermal estimation at the architecture level. The demands for reliable and practical tools for thermal architecture modeling from both architecture and computer-aided design (CAD) tool communities could not be higher.

The traditional bottom-up approaches including finite element (FEM), finite difference (FDM), and computational flow dynamics (CFD)-based methods were widely used for thermal modeling and analysis in the past. They can be accurate when detailed thermal structures are known. However, these detailed models can be substantially large, which prevents their use in many practical problems. Static and transient thermal modeling methods at different levels (parts, package, board) have been proposed in the past. Many approaches try to use thermal resistance and capacitance with fixed topology networks subject to different thermal boundary conditions [11], [5], [2]. The main limitation of those methods is to determine appropriate *RC* values of elements, especially for complex geometries and boundary conditions. The *RC* values are typically determined and optimized against the field numerical or analytic results [7], [17] and measured data [19].

For thermal modeling at architecture level, existing work on HotSpot [9], [20] tries to solve this problem by generating the architecture thermal model in a bottom-up way based on the internal structure/architecture of the microprocessor. These bottom-up compact models, however, may suffer from accuracy loss, and compact models have to be calibrated with hardware or detailed simulation if more accurate models are required.

Manuscript received March 23, 2010; revised July 02, 2010, August 30, 2010; accepted September 14, 2010. Date of publication January 20, 2011; date of current version January 18, 2012. This work is supported in part by the National Science Foundation under Grant CCF-0448534 and Grant CCF-0902885, by Semiconductor Research Corporation (SRC) under Grant 2009-TJ-1991, and by the Science and Technology Commission of Shanghai Municipality under Grant 2009B021.

T. J. Eguia was with the Department of Electrical Engineering, University of California, Riverside, CA 92521 USA. He is now with West Digital Corporation, Irvine, CA 92612 USA.

S. X.-D. Tan and R. Shen are with the Department of Electrical Engineering, University of California, Riverside, CA 92521 USA (e-mail: stan@ee.ucr.edu).

D. Li was with the Department of Electrical Engineering, University of California, Riverside, CA 92521 USA. He is now with Synopsis Corporation, Mountain View, CA 94043 USA.

E. H. Pacheco and M. Tirumala are with the System Research Laboratory, Intel Corporation, Hillsboro, OR 97124 USA.

L. Wang is with the Department of Microelectronics, Fudan University, Shanghai 200433, China.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2010.2098054

Recently, a top-down behavioral architecture level thermal modeling method, *ThermPOF*, has been proposed [12], where temperature impulse responses are used to build the thermal models by the matrix pencil method.

In this paper, we propose a new parameterized thermal modeling approach for fast temperature estimation at the architecture and package levels for high-performance microprocessors. The new approach can build the behavioral thermal models from measured or simulated transient thermal and power information. The main advantage of the proposed modeling method over the existing black-box thermal modeling methods like *ThermPOF* [12] and *ParThermPOF* [13] is that the proposed method can accept *general* transient power and temperature waveforms. In contrast, existing black-box methods can only accept impulse/step power inputs and allow only one input to be excited at a time. Such relaxation on input powers make a huge difference as the new method is much more training friendly and general, as transfer function-like responses are typically difficult, even impossible (intractable), to obtain from the measurements. As before, the new method is a top-down, black-box approach, which means it does not require knowing any internal structure of the system. Lastly, it can accommodate a number of parameters, such as location of thermal sensors on a heat sink or thermal conductivity of heat sink materials, for example.

The new method, called *ParThermSID*, consists of two techniques. First, the response surface method (RSM) based on second-order polynomials is adopted to build the parameterized models at each time point for all of the given sampling nodes in the parameter space (except for time). Second, it applies an improved subspace system identification method, called *ThermSID*, to build the transient model for each time-varying coefficient of the polynomials generated in the first step. The subspace system identification method can accept general transient inputs and thus eliminates the need for impulse/step power inputs. The subspace system identification first generates the states of the desired models in terms of a Hankel matrix of Markov parameters from the measured input and output data via subspace projection and reduction. Then, the discrete state matrices are obtained through state matrix realizations. To overcome the over-fitting problem in the subspace method, *ThermSID* applies an over-fitting mitigation technique to pick up the best model among several that are built on partial training data to overcome the unavoidable overfitting problem associated with training-based modeling processes.

Experimental results on a real multicore microprocessor show that *ThermSID* and *ParThermSID* can provide thermal behavioral models that match the measured data very closely with similar accuracy to *ThermPOF* and *ParThermPOF*. The compact models generated by *ParThermSID* also offer two orders of magnitude speedup over the commercial thermal analysis tool *FloTHERM* on given examples. *ParThermSID* is also much more general and flexible than the recently proposed parameterized thermal modeling method *ParThermPOF*.

The remainder of this paper is organized as follows. Section II presents the thermal modeling problem we are trying to solve. Section III reviews the subspace system method for its use in thermal modeling, while Section III-B explores overfitting and the resulting mitigation techniques. Section IV reviews the response surface method and describes its use in parameterization. Finally, Section V presents the results of both *ThermSID* and *ParThermSID*, with Section VI concluding the paper.

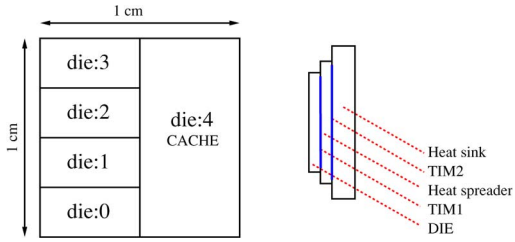


Fig. 1. Quad-core architecture.

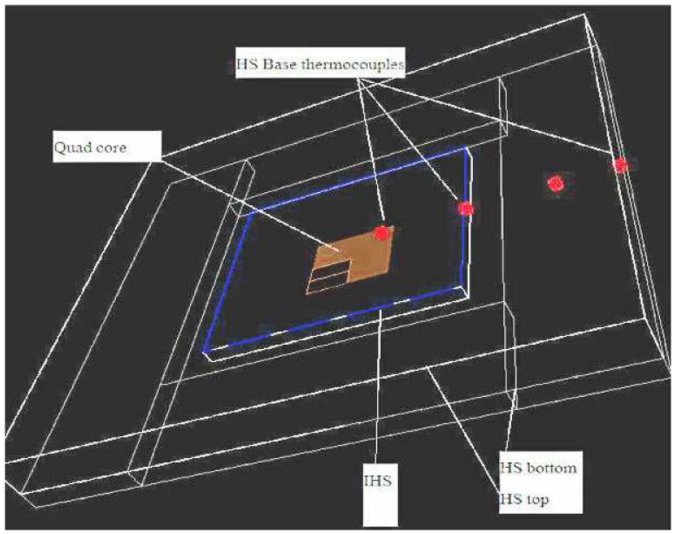


Fig. 2. 3-D structure of a quad-core processor.

II. Package-Level Parameterized Thermal Modeling Problem

Our modeling problem requires building parameterized thermal models considering both time and other variable parameters of multi-core processors. The goal is to build a behavioral model with power and temperature as its inputs and outputs. The behavioral model must be dependent not only on power inputs, but on different system parameters as well. Two types of parameters are considered in our modeling problems. The first is time, with the remaining parameters to be discussed below.

We validate the proposed thermal modeling method by specifically looking at a quad-core microprocessor architecture from our industry partner, Intel Corporation. The architecture of this multicore microprocessor, which contains four CPU cores (die 0 to die 3) and one cache core (die 4), is shown in Fig. 1. The temperatures are measured on the center of each die's bottom face.

Fig. 2 shows a 3-D structure of this quad-core microprocessor, where the CPU die (with quad-cores) is located at the bottom and is in contact with the integrated heat spreader (IHS) in blue. At the top is the heater sink (HS), which has both top and bottom parts. The thermocouples (thermal sensors) in red (online version) are used to measure the real temperatures

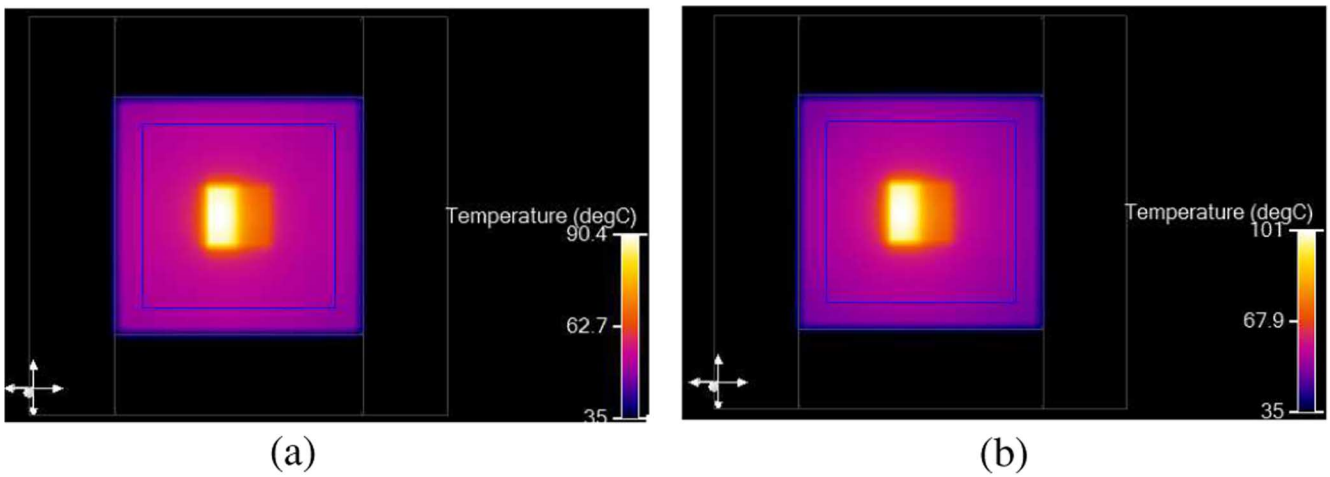


Fig. 3. Temperature distributions on the whole chip using different heat sink materials when all cores and cache are active. (a) Using copper heat sink. (b) Using aluminium heat sink.

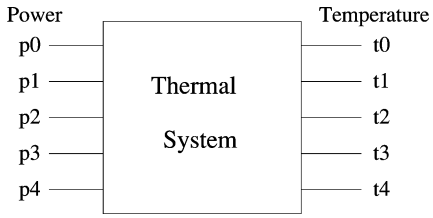


Fig. 4. Abstracted system.

at specific locations, such as 5, 15, and 25 mm away from the center of the heat sink. Typically, temperatures reduce as we move away from the center and bottom parts of the chip. Temperatures are hottest at the center of the core.

We also consider the thermal conductivity of the heat sink material as another parameter. Normally, a heat sink is made of either copper (Cu) or aluminum (Al). Cu and Al have different thermal conductivities, with Cu at 390 W/(m·K) and Al at 240 W/(m·K). Different heat sink materials may induce different temperature distributions on the chip. Fig. 3 shows the temperature distributions on the entire chip using a copper and an aluminum heat sink. The two figures show that the maximum temperature of the chip on a copper heat sink is 10 C° less than that of the aluminum. The price of a copper heat sink, however, is apparently much higher than aluminum, so designers are met with a tradeoff between hot-spot temperatures and package cost. In our work, we set up such a parameter to indicate the thermal conductivity of the heat sink material. Such parameterized thermal models may prove helpful for design exploration and optimization.

Finally, this quad-core processor can be abstracted into a linear system with five inputs, eight outputs, and two parameters, as shown in Fig. 4. The inputs are the power traces (p_0 to p_4) of all of the cores, and the outputs are the temperature responses on core0 to core3, cache, die, heat spreader, and heat sink for given parameters, which are represented by t_0 to t_4 in the figure. The parameters can be the location of the thermal sensors (distance to a center point) and thermal conductivity of the heat sink material, for example.

Instead of using transfer functions to model the systems [12], which require step/impulse power inputs, we propose using a discrete linear time-invariant system with state space models

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k + Du_k \end{aligned} \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$ is a stable matrix, $B \in \mathbb{R}^{n\times m}$, $B \in \mathbb{R}^{l\times n}$, and $D \in \mathbb{R}^{l\times m}$. The input vectors $u_k \in \mathbb{R}^{m\times 1}$ and output vectors $y_k \in \mathbb{R}^{l\times 1}$ are the measured power input traces and temperature responses, respectively. x_k are estimated state variables (called Kalman states).

Given input u_k and output y_k, the problem at hand is how to generate state matrices A, B, C, and D, where D is typically considered a matrix of zeros. Note that given the discrete system matrices, the continuous system matrices can be obtained using methods such as zeroth-order-hold, impulse invariance and Tustin approximation [6].

III. Subspace-Based Transient Thermal Modeling—ThermSID

Here, we introduce the transient thermal modeling algorithm—*ThermSID*. We first start with the introduction of the subspace identification method.

A. Subspace System Method

To determine the state matrices of a core's thermal system, we apply a recently proposed subspace system identification method, called Numerical Algorithm for Subspace State Space System Identification (N4SID), to build the thermal models [16].

The subspace method basically builds high-order prediction models by least-squares algorithm based on the transient inputs and outputs. The method differs from the previous method in that it builds (implicitly) the state variables (Kalman states) first, then realize the state matrices. The beauty of this method is that it is very numerically robust as only two numerical stable operations, QR (or its variant LQ) decomposition and singular value

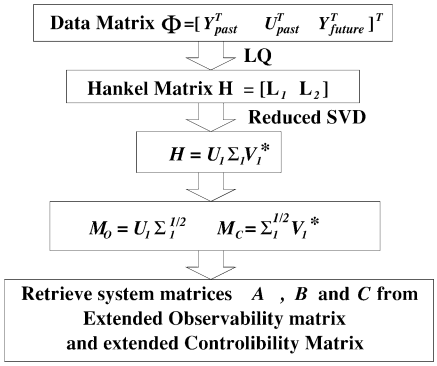


Fig. 5. Brief diagram of the subspace state-space system identification.

decomposition (SVD), are required and no iterative methods involved.

The whole algorithm flow is shown in Fig. 5. The algorithm consists of three major steps.

Step 1) Compute the Kalman states (implicitly) or, equivalently, state matrices (Hankel matrix) from the input and output data $Y_{\text{future}}, U_{\text{past}}, Y_{\text{past}}$. Numerically, this is done by the least squares minimization method (via LQ decomposition).

Step 2) Perform the SVD on the combined coefficient matrices $[L_1, L2]$ to obtain $\mathcal{M}_C$ and $\mathcal{M}_O$, which are the extended controllability and observability matrices.

Step 3) Realize the system matrices $[A, B, C]$ from $\mathcal{M}_C$ and $\mathcal{M}_O$ by least-squares-based regression.

A detailed description of the algorithm is given in the Appendix.

Compared with a previously proposed thermal modeling algorithm, *ThermPOF* [12], thermal modeling using the subspace identification method is more flexible and general. Since step responses are used during the training process in *ThermPOF*, a change to log-scale is necessary to better capture step response behavior. On the contrary, random power input traces and the resulting temperature responses of the desired system can be used directly as the input and output data sets for *ThermSID*, from which the system models are generated. Hence, it does not require the step temperature response as in *ThermPOF*. It also avoids log-scale change for time during the training process.

B. The Overfitting Problem

Simply applying the subspace-based method, however, can lead to overfitting problems during the modeling process. Overfitting occurs when a model describes random error or noise instead of the underlying behavior of the system, thus leading to poor predictive performance on new data sets.

Typically, overfitting occurs when a model is built to follow its training data so closely that it harms the model's predictive capabilities. The same phenomenon occurs when using the subspace system method to generate the model. Fig. 6 provides an example of this using a model of order 10. We can see that the model provides a near exact match of the training data used to generate it. However, simulating a new set of inputs shows that the model is unable to follow the trend of the verification data.

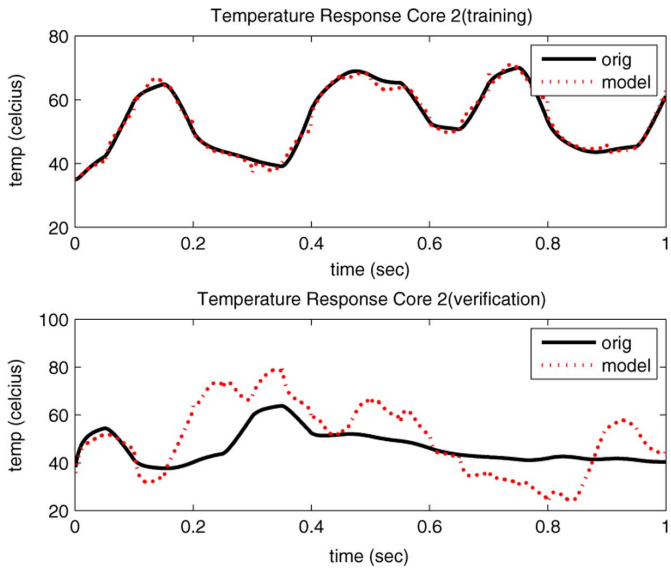


Fig. 6. Training and verification waveform comparison, $\text{model order} = 5$.

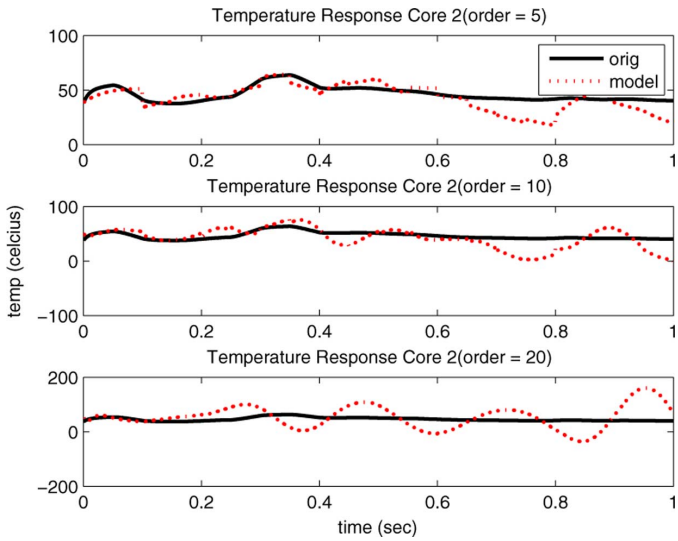


Fig. 7. Verification waveform comparison, model order = 5, 10, 20.

C. Proposed Mitigating Scheme

Improving the model's predictive capabilities requires minimizing the complexity of the model itself. Fig. 7 shows the simulation results of three models of different order that are generated using the same training set. The model at order 20 exhibits the worst fit, but improves progressively as the model order is reduced. At order 5, the fit of the model has significantly improved. Furthermore, our tests indicate that order 5 is the optimal setting for our model, with no improvement gains to be made at lower orders. Lowering the complexity of the model, however, adversely affects the fit of the training data. Fig. 8 shows this result, where the quality of the fit decreases with the order of the model. The significance of this result will be explained later.

Despite lowering the complexity, the subspace system method is still unable to provide a reasonably accurate model. This result suggests that overfitting is still occurring and may be

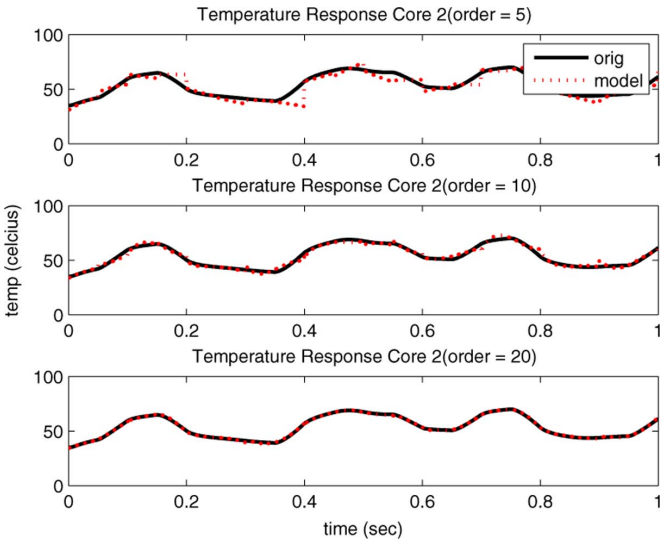


Fig. 8. Training waveform comparison, model order = 5, 10, 20.

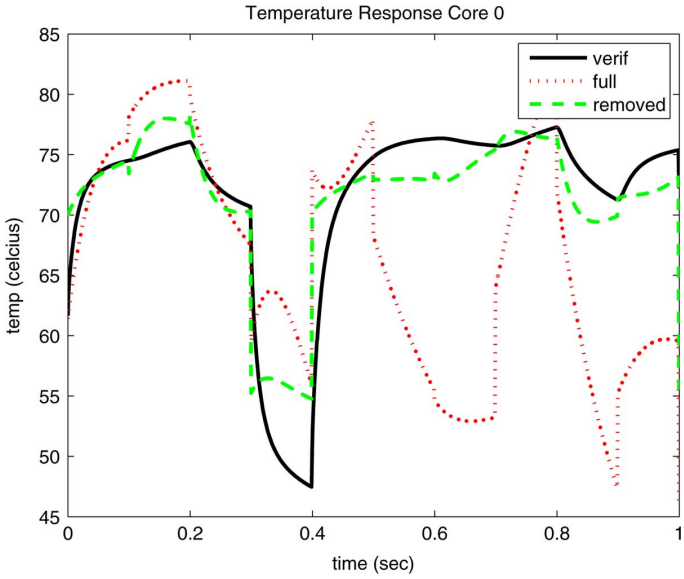


Fig. 10. Temperature response comparison between models 1 and 2 for Core 0.

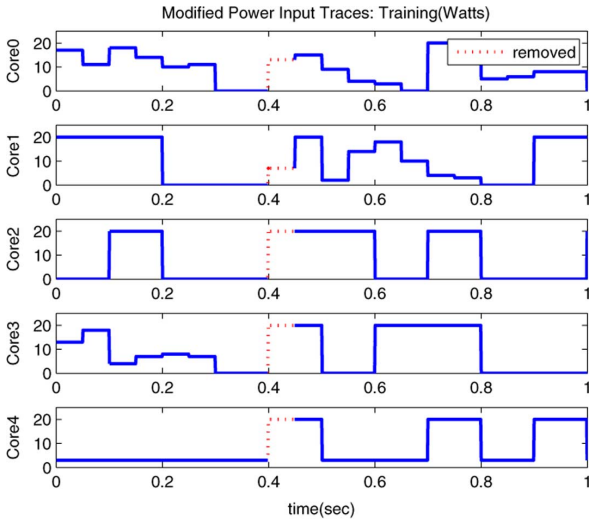


Fig. 9. Power input traces with section removed.

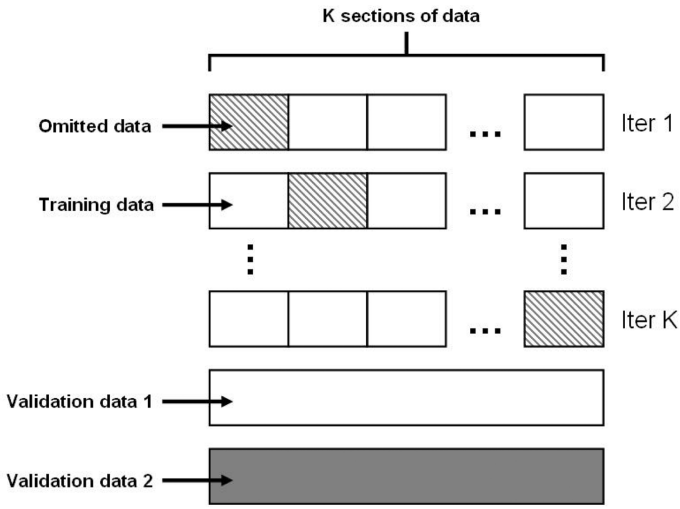


Fig. 11. Overfitting solution algorithm.

explored further. Several methods have been developed in the statistics community to mitigate the overfitting problem, which include early stopping, regularization, cross-validation, and Bayesian priors [10], [18]. For our problems, we observe that some sections of training data are more misleading than other sections due to highly correlated data that mask the underlying behavior of the system.

To show the effects of these correlated data, we shall observe the differences in accuracy of a model created by the training data presented in Fig. 9. First, we shall create model 1 using the entire data set, and then generate model 2 with the dotted red section removed. Fig. 10 shows the simulation results of Core 0 on a separate verification set. The dotted red line is the waveform created by model 1, while the dashed green line is created by model 2. The results show a distinct improvement in accuracy when a particular section of data is removed. This misleading section of data appears to be detrimental to the predictive capabilities of the model. In this particular section, a step input value is present in each of the four cores.

Based on such observations, we design a specific overfitting mitigation scheme in which we iteratively remove some portions of the training data to search for better models. An obvious solution may be to remove all sections where there are simultaneous step inputs. However, doing this will remove most of the data, leaving us with an unusable training set. Furthermore, not all simultaneous step input occurrences are necessarily misleading. Removing certain sections indiscriminately may, in fact, harm the accuracy of the generated model. Therefore, a method of detecting and removing these misleading sections of data proves necessary.

To detect misleading sections of data, we employ a methodology that requires both a training set and validation set. This could be done by splitting an existing data set in half, with one portion used for training and the other for validation. Such partitioning scheme is similar to the Early Stopping method [10]. As described in Fig. 11, the training set is further partitioned into K sections. In each iteration, a section of data is omitted,

ALGORITHM: THERMSID
Input: power input vector u_i for each input i,
temperature response vector y_j for each output j
Output: state space matrices A, B, C.

1) Form $W = [Y_{past}^T \quad U_{past}^T \quad Y_{future}^T]^T$.
2) Perform LQ decomposition on W. Form $\mathcal{H} = [L_1 L_2]$.
3) Perform SVD on $\mathcal{H} = [L_1 L_2] = U\Sigma V^*$.
4) Define controllability and observability matrices as shown in (23).
5) Retrieve state space matrices A, B, C from extended controllability and extended observability matrices.
6) Perform the overfitting mitigation scheme on top of steps (1)-(5) to improve the model predictiveness if necessary.

Fig. 12. Thermal modeling using subspace system identification method.

with the rest used as training data. A modeling error ϵ is computed in each iteration by comparing the model created using the training data i with the validation data set. In addition, the model is also compared to the entire training data set, resulting in a second modeling error value. The motivation for this comparison stems from the prior observation that the training waveform accuracy is lowered with low ordered models. It is further observed that removing sections of misleading data results in the improvement of the training waveform accuracy. Therefore, verification on two separate data sets provides a much greater insight into the predictive ability of the model. The final modeling error ϵ_i is thus calculated as the average of the errors for validation sets 1 and 2. Afterwards, the improved model can be chosen as the one with the lowest error value.

Fig. 12 presents the algorithm flow of the proposed thermal behavioral modeling method using the subspace system identification approach. Depending on the desired performance of the training data, the overfitting mitigation technique can be performed prior to utilizing the algorithm.

IV. PARAMETERIZED THERMAL MODELING METHOD—PARTHERMSID

To incorporate the parameter variables for the models, we apply the response surface method, which uses polynomials to represent the temperature responses with respect to the parameters. Then, the coefficients of the polynomials become functions of the time and power inputs, which are similar to the temperature responses for the nonparameterized problem and thus can be solved by ThermSID.

In summary, in order to build the parameterized behavioral model, we need to solve the following two problems: 1) finding response polynomial functions that can approximate the measured temperatures for all the controllable variables (parameters) with sufficient accuracy and 2) finding the state space models for polynomial coefficients [of the polynomials from 1)] to capture the behavior of the temperature. For problem 1), we introduce the response surface method to capture linear or nonlinear relationships between the parameters and responses (temperatures) at each time point. For problem 2), we apply ThermSID in the previous section to generate the dynamic models.

A. Response Surface Model for Thermal Response

Response surface methodology (RSM) explores the relationships between several input variables and one or more responses. The main principle of RSM is to use a set of designed experiments to obtain an optimal response. There are many applications of RSM in real industry, particularly in situations where several input variables potentially influence some performance measure or quality characteristic of the product or process [15]. This performance measure or quality characteristic is called the *response*, while the input variables are sometimes called *independent variables*.

Specifically, suppose that a response y depends on several controllable input variables $(\xi_1, \xi_2, \ldots, \xi_k)$

$$y = f(\xi_1, \xi_2, \ldots, \xi_k) + \varepsilon \tag{2}$$

where the form of the true response function f is unknown and perhaps very complicated. We need to minimize the error ε when building response surface models.

We must first transform the natural variables ξ in the range $[a, b]$ into coded variables z in the range $[-1, 1]$ using the linear transformation in

$$z_i = \frac{\xi_i - (b+a)/2}{(b-a)/2}. \tag{3}$$

After coding, the variable matrix Z, defined in (7), will have all orthogonal columns, which can reduce numerical errors and increase numerical stability.

At time t, a second-order response y depending on variables $(z_1, z_2, \ldots, z_k)$ can be written as

$$y = \vec{\beta}_0(t) + \sum_{j=1}^{k} \vec{\beta}_j(t) z_j + \sum_{j=2}^{k} \sum_{i<j} \vec{\beta}_{ij}(t) z_i z_j + \sum_{j=1}^{k} \vec{\beta}_j(t) z_j^2 + \varepsilon. \tag{4}$$

Notice that $\vec{\beta}_i(t) = \vec{\beta}_i(t, p_0, p_1, \ldots, p_n)$ is a vector function of time and input powers. If we let $z_{k+1} = z_1 z_2$, $z_{k+2} = z_2 z_3, \ldots, z_{k(k+1)/2+1} = z_1^2$, $z_{k(k+1)/2+2} = z_2^2, \ldots, \vec{\beta}_{k+1}(t) = \vec{\beta}_{12}(t)$, $\vec{\beta}_{k+2}(t) = \vec{\beta}_{23}(t), \ldots, \vec{\beta}_{k(k+1)/2+1}(t) = \vec{\beta}_{11}(t)$, $\vec{\beta}_{k(k+1)/2+2}(t) = \vec{\beta}_{22}(t), \ldots$, then (4) becomes

$$y = \vec{\beta}_0(t) + \sum_{j=1}^{q} \vec{\beta}_j(t) z_j + \varepsilon \tag{5}$$

which is a generalized linear regression model for coefficients $(\vec{\beta}_0(t), \vec{\beta}_1(t), \ldots, \vec{\beta}_q(t))$, where $q = k(k+3)/2$. We can use the least squares method to estimate the regression coefficients in the multiple linear regression model in (5).

Suppose that we have n observed responses $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ and for each y_i we have one set of parameter values $\mathbf{z_i} = (z_{i1}, z_{i2}, \ldots, z_{iq})$. (5) can then be written in matrix notation as

$$\mathbf{y} = \mathbf{Z}\vec{\boldsymbol{\beta}}(\boldsymbol{t}) + \boldsymbol{\varepsilon} \tag{6}$$

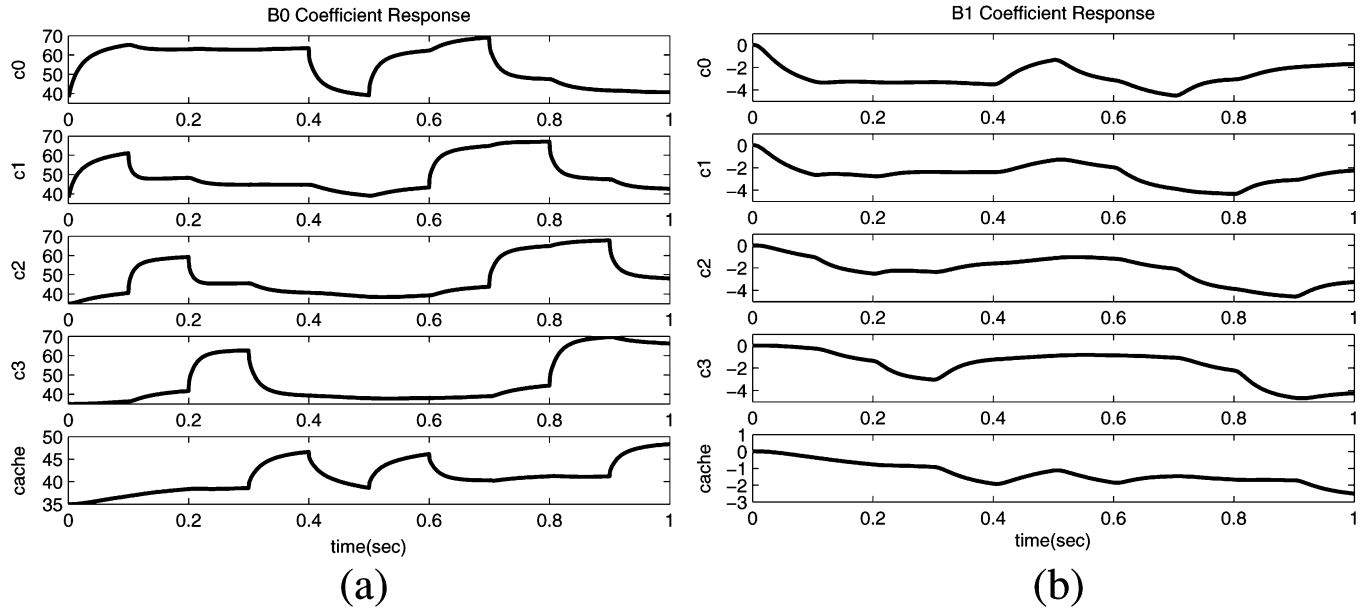


Fig. 13. Coefficient responses using training data as power inputs. (a) $\vec{\beta}_0$ coefficient response. (b) $\vec{\beta}_1$ coefficient response.

where

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\mathbf{Z} = \begin{bmatrix} 1 & z_{11} & z_{12} & \cdots & z_{1q} \\ 1 & z_{21} & z_{22} & \cdots & z_{2q} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & z_{n1} & z_{n2} & \cdots & z_{nq} \end{bmatrix}$$

$$\vec{\boldsymbol{\beta}}(\boldsymbol{t}) = \begin{bmatrix} \vec{\beta}_0(t) \\ \vec{\beta}_1(t) \\ \vdots \\ \vec{\beta}_q(t) \end{bmatrix}$$

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_0 \\ \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{bmatrix}. \tag{7}$$

We can minimize the squares of errors using the least squares estimator $\vec{\boldsymbol{\beta}}_{\mathbf{est}}(\boldsymbol{t})$ in

$$\vec{\boldsymbol{\beta}}_{\mathbf{est}}(\boldsymbol{t}) = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{y}. \tag{8}$$

In practice, we perform QR decomposition on $X = QR$, where Q is orthogonal matrix and R is an upper triangular matrix, to make the computation more numerically stable, thus obtaining $\mathbf{R}\vec{\boldsymbol{\beta}}(\boldsymbol{t}) = \mathbf{Q}^T\mathbf{y}$. After solving the linear equations, we get the estimated coefficient vector $\vec{\boldsymbol{\beta}}_{\mathbf{est}}(\boldsymbol{t})$.

B. Parameterized Thermal Behavioral Models

After we obtain the coded variable matrix X, the coefficients of our model can be computed using (8). At this point, we obtain the parameterized thermal model only on a single time point. We then compute the response surface models on all time points, which generates a set of response surface models, or more precisely, a set of coefficients, $\vec{\boldsymbol{\beta}}(\boldsymbol{t})$. Each $\vec{\beta}_i(t)$ is a multi-input and multi-output function of time

$$\vec{T}_{\text{coeff}}(t) = \vec{\beta}_i(t, p_0, p_1, \ldots, p_l). \tag{9}$$

In our specific case, $n = 4$ and $\vec{T}_{\text{coeff}}(t) \in R^5$ is a vector for the temperature coefficients at time t. Fig. 13 shows the training coefficient responses (from 0 to 1 s) of $\vec{\beta}_0$ and $\vec{\beta}_1$, which are both functions of time. Since we can consider the temperature response as a linear combination of such coefficients, the original thermal system is decomposed into a number of linear dynamic subsystems.

After we obtain the coefficients $\vec{\beta}_i$, which are function time and input powers, we can generate their system matrices $[A_i, B_i, C_i]$ for each coefficient $\vec{\beta}_i$ using *ThermSID*. Each coefficient, a special temperature, and the input powers, will become a multiple output and multiple input system based on our thermal models.

C. ParThermSID Algorithm Flow

Fig. 14 summarizes the algorithm flow *ParThermSID*. In this algorithm, the first three steps compute the coefficients of polynomials of parameters using response surface method. Step 4) performs *ThermSID* to construct the dynamic coefficient subsystems for each coefficient.

During the model evaluation, we first compute the responses from all the coefficient systems. After obtaining the coefficient responses, we can easily compute the temperature responses using (5).

Compared with *ParThermPOF* in [13], *ParThermSID* can handle multiple input multiple output thermal systems. Furthermore, it can work for general power inputs and does not rely on the step power input training data, which is necessary for *ParThermPOF*.

ALGORITHM: PARTHERMSID
Input: temperature response vectors y_i for each respective parameter, where $i = 1...l$, parameters p_j, where $j = 1...k$
Output: coefficient models for each $\beta_i(t)$, $i = 0...q$

1) Code parameter values z_j.
2) Form second order response equation in matrix notation as show in (5) for each time step t.
3) Calculate $\vec{\beta}_{est}(t)$ using (8) for each time step t.
4) Apply *ThermSID* to generate models for each coefficient $\beta_i(t)$ subsystem.

Fig. 14. Parameterized thermal modeling algorithm—*ParThermSID*.

V. EXPERIMENTAL RESULTS

This section verifies the effectiveness of the proposed thermal modeling methods. The proposed method using the subspace system identification method and overfitting technique is named *ThermSID*, while the parameterized version is known as *ParThermSID*. We perform our tests on a quad-core microprocessor architecture shown in Fig. 1 from our industry partner Intel Corp. Intel built the detailed quad-core process model using *FloTHERM*.[1] The detailed model follows closely to the actual quad-core processors. The experimental results were obtained from a Window PC workstation with Intel Core 2 CPU (2.4-GHz, 2-GB memory).

Both training and verification data sets were also provided by Intel. All algorithms were implemented in MATLAB 7.6.

A. Results for ThermSID With Overfitting Mitigation

In this experiment, we shall compare the performance of *ThermSID* with and without the overfitting mitigation technique. We generate a model with *ThermSID* using the entire training data set, and then generate a second model by employing the overfitting mitigation scheme. We then compare the accuracy of both models using a separate verification data set. The training data set excites each core with step power inputs that range between 0 to 20 W for a duration of 0 to 1 s. The verification data also excites the system with step inputs, but only with values of 0 or 20 W. Figs. 15 and 16 show the training and verification power traces, respectively. We partition the training data set into 20 sections during the overfitting mitigation process.

We verify our algorithms by comparing their results with the temperature response data of the verification set. Figs. 17 and 18 show the temperature response waveforms on Core 2 and the cache. The dotted line (*Sid*) represents the simulation results generated by *ThermSID* without overfitting, while the dashed line (*ThermSID*) represents the results generated by *ThermSID* with overfitting. Lastly, the black line is the measured temperature response from the verification data. The figures clearly show the improvements in accuracy when the overfitting mitigation process is employed in generating the models.

[1][Online]. Available: http://www.flomerics.com/products/flotherm/

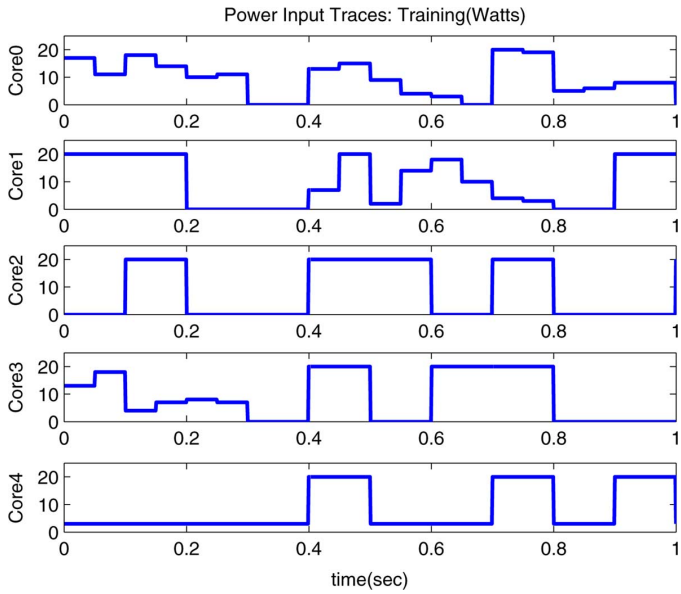


Fig. 15. Random power input traces for cores 0 to 3 and cache (set 1).

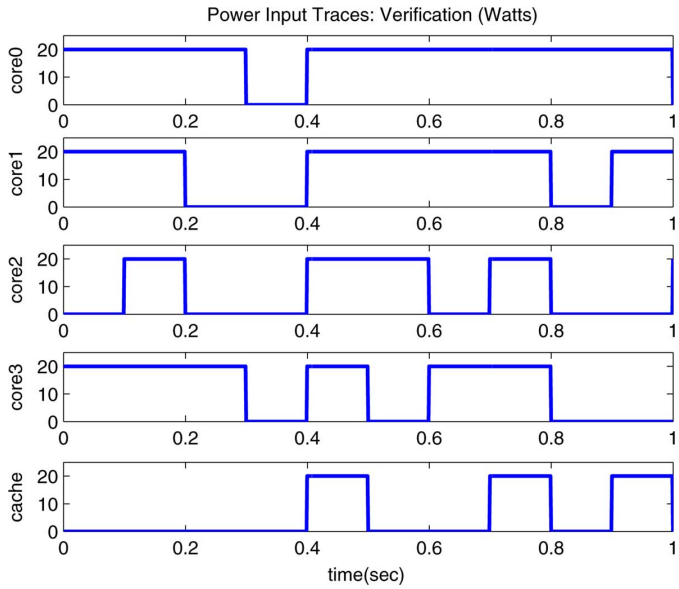


Fig. 16. Random power input traces for cores 0 to 3 and cache (set 2).

Table I provides the percentage error statistics of cores 0 to 3 and the cache. The mean error and standard deviation are calculated for both *Sid* and *ThermSID* models. Results show that *ThermSID* significantly reduces the mean and standard deviation of errors in all regions. The most notable improvement occurs in the cache, where the mean error drops from 21.17% to 2.21%.

We then compare the performance of *ThermSID* against a previously proposed thermal modeling method *ThermPOF*. Table III presents the error statistics of both methods. Note that since *ThermPOF* utilizes step responses to generate its model, we use a training data set with power inputs similar to that of Fig. 16 and their corresponding temperature response outputs. The models are then verified on a separate data set. With distinctly smaller mean error and standard deviation values,

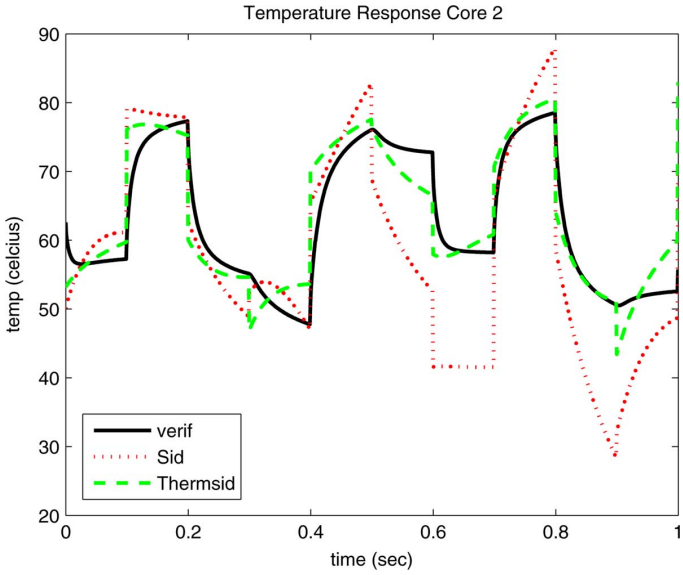


Fig. 17. Temperature response of the verification data, *Sid*, and *ThermSID* for Core 2.

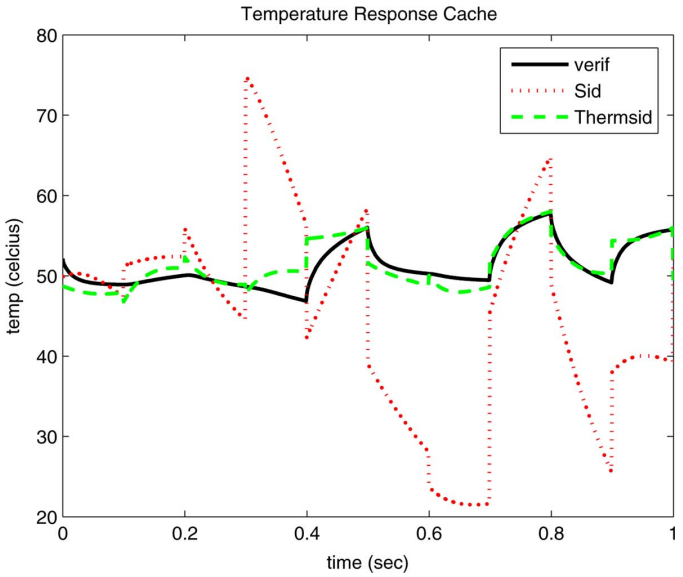


Fig. 18. Temperature response of the verification data, *Sid*, and *ThermSID* for cache.

our results suggest that *ThermSID* exhibits better accuracy than *ThermPOF*.

We further verify the overfitting mitigation method by applying it to a second training data set with greater predictive ability. As with the first test, the training data set is divided into 20 sections, and *Sid* is used to generate the model. Fig. 19 compares the temperature response of Core 1, from which we can see that the model created by *ThermSID* exhibits an improvement in accuracy. Table II shows that the improvements carry over to all regions of the core, with a decrease in both mean error and standard deviation. The results of our tests on training set 2 show that the overfiting mitigation scheme is still applicable even on data sets that exhibit a fair amount of accuracy.

TABLE I
STATISTICS OF ERRORS (%) BETWEEN GIVEN AND COMPUTED TEMPERATURES FOR VERIFICATION SET 1

core	Sid mean error	ThermSID mean error	Sid error std	ThermSID error std
Core 0	13.67	3.63	10.53	4.29
Core 1	8.51	4.03	7.10	4.32
Core 2	12.68	4.55	10.78	4.63
Core 3	15.17	5.72	13.50	4.97
Cache	21.17	2.21	18.64	2.10

TABLE II
STATISTICS OF ERRORS (%) BETWEEN GIVEN AND COMPUTED TEMPERATURES FOR TRAINING SET 2

core	Sid mean error	ThermSID mean error	Sid error std	ThermSID error std
Core 0	6.55	3.54	4.85	4.00
Core 1	6.06	2.70	5.94	2.96
Core 2	7.81	5.82	5.86	3.38
Core 3	9.40	7.58	6.61	7.19
Cache	9.37	4.77	5.12	3.32

TABLE III
STATISTICS OF ERRORS (%) BETWEEN GIVEN AND COMPUTED TEMPERATURES: THERMSID VERSUS THERMPOF

core	ThermPOF mean error	ThermSID mean error	ThermPOF error std	ThermSID error std
Core 0	0.74	0.16	0.29	0.11
Core 1	0.98	0.24	0.69	0.38
Core 2	3.11	0.34	1.31	0.46
Core 3	2.10	0.20	0.88	0.26
Cache	1.23	0.62	0.43	0.37

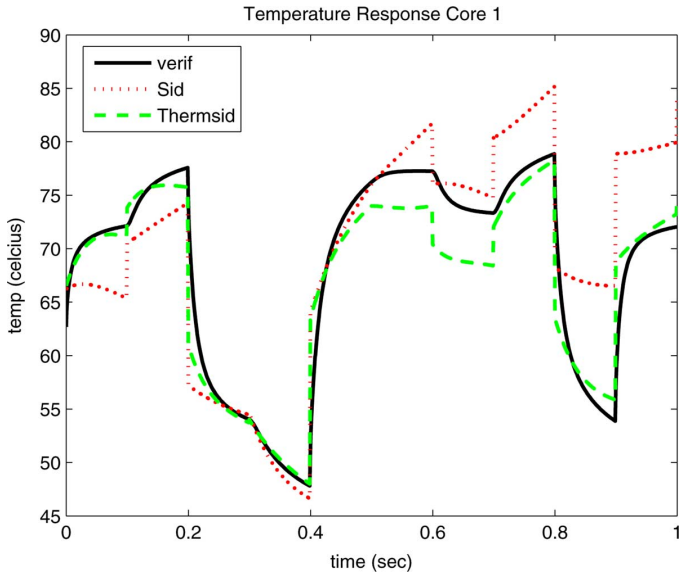
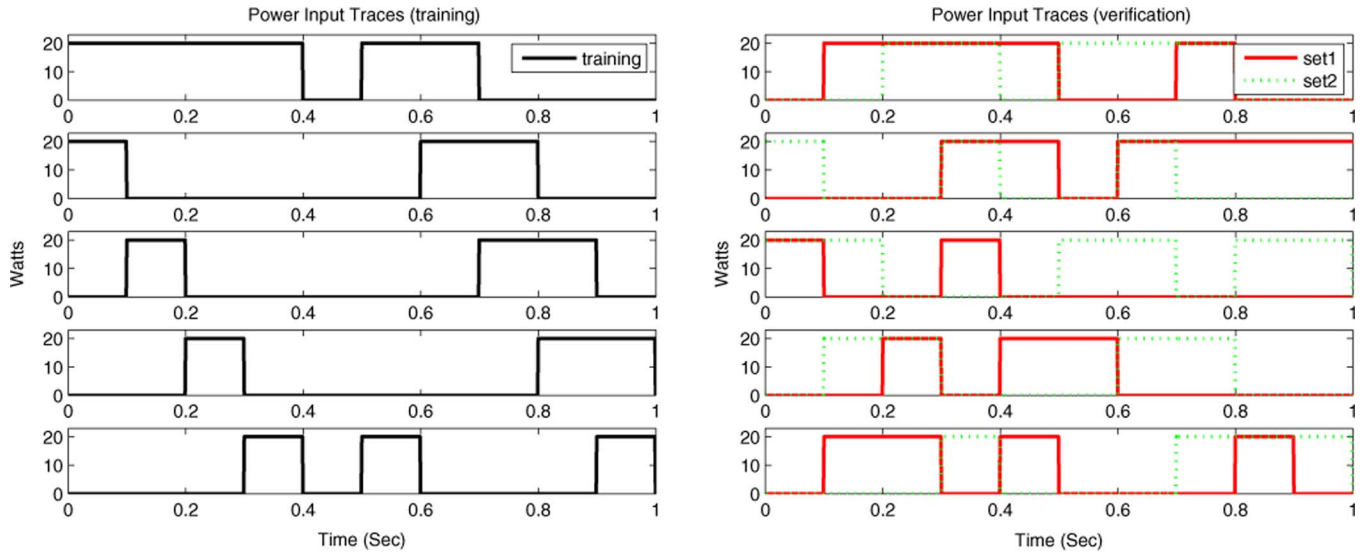


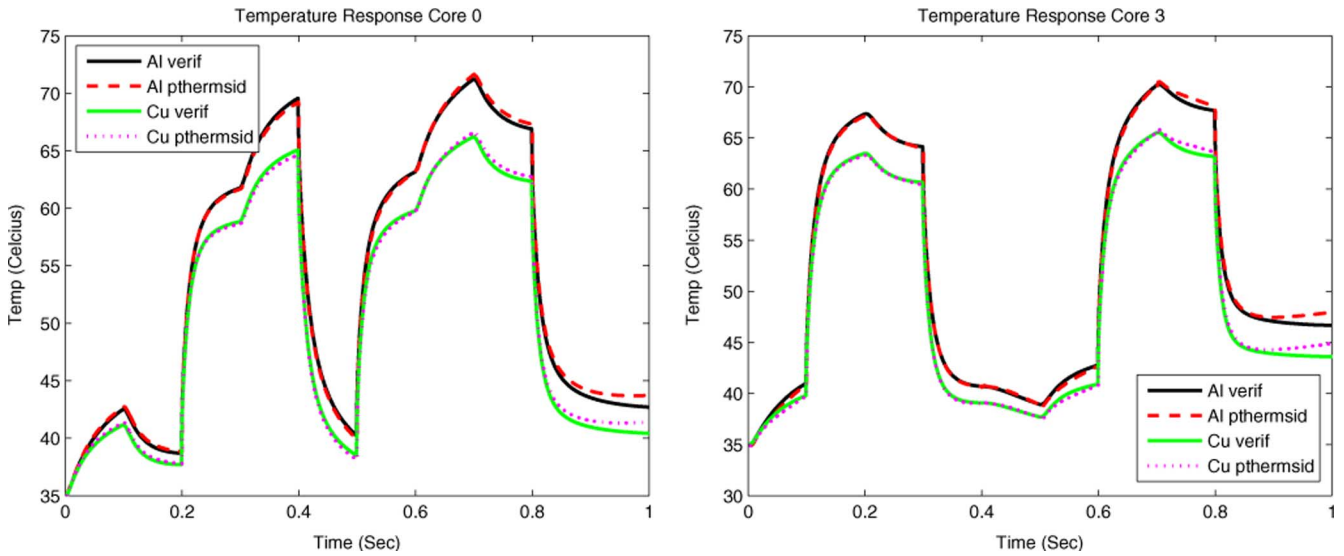
Fig. 19. Temperature response of the verification data, *Sid*, and *ThermSID* for Core 1.

Our experimental results show that utilizing the overfitting mitigation method in conjunction with the subspace-based modeling method significantly increases the accuracy of its generated model. Such an optimization approach is useful when sets of training data are scarce, and the present data is unable to generate an acceptably predictive model. While more optimal methods of model verification exist, the overfitting mitigation



(a) (b)

Fig. 20. Power input traces in watts. (a) Power input traces for training. (b) Power input traces for verification sets 1 and 2.



(a) (b)

Fig. 21. Temperature response waveforms considering aluminum and copper heat sink materials. (a) Temperature response waveforms at core 0. (b) Temperature response waveforms at core 3.

approach enables the creation of an improved model under such restrictive circumstances.

B. Result for ParThermSID

This section verifies the effectiveness of the proposed parameterized *ParThermSID*. An optimal training data set is used to generate the subthermal systems mentioned in Section IV-B, which are of order $m = 25$. The accuracy of the generated model is then measured using two verification sets. A comparison of error statistics between *ParThermSID* and *ParThermPOF* [13] is also provided.

The power inputs of both training and verification data, as shown in Fig. 20, excites each core with a series of step-like power inputs of 0 or 20 W for a duration of 0 to 1 s. Notice that more than one power inputs can be active in this case, which is different from the more restrictive requirement of the one-input-active power inputs in *ParThermPOF* [13]. A sample is provided at every 0.001 s, which gives a data set of approximately 1000 samples.

We verify the proposed method by comparing its results with the temperature responses of verification sets 1 and 2. Fig. 21 provides the temperature response waveforms on cores 0 and 3 using verification set 2. The graphs clearly show that the models

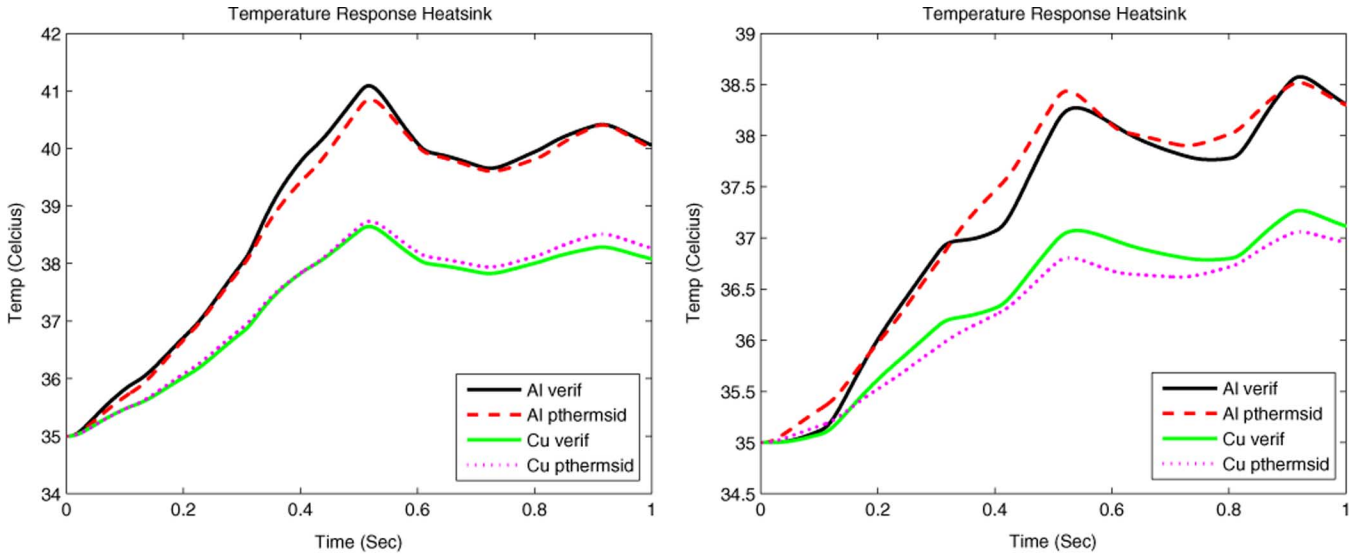


Fig. 22. Temperature response waveforms of the heat sink at different distances while considering aluminum and copper materials. (a) Temperature response waveforms at 0 mm. (b) Temperature response waveforms at 5 mm.

TABLE IV
STATISTICS OF ERROR PERCENTAGES AND MAXIMUM PEAK ERRORS FOR VERIFICATION SET 1. BOTH ALUMINUM AND COPPER ARE CONSIDERED. ALL VALUES ARE GIVEN AS PERCENTAGES

	Aluminum			Copper		
core	mean	std	maxpk	mean	std	maxpk
Core 0	0.4	0.48	0.25	0.58	0.57	0.51
Core 1	0.48	0.43	0.68	0.64	0.52	0.94
Core 2	0.62	0.49	0.67	0.82	0.61	0.91
Core 3	0.8	0.55	0.60	0.99	0.66	0.82
Cache	0.82	0.57	0.74	1.05	0.73	0.97
hs(0mm)	0.29	0.57	0.59	0.24	0.17	0.23
hs(5mm)	0.35	0.29	0.15	0.41	0.22	0.56
hs(15mm)	0.39	0.23	0.63	0.67	0.38	0.41
hs(25mm)	0.49	0.27	0.53	0.11	0.07	0.23

TABLE V
STATISTICS OF ERROR PERCENTAGES AND MAXIMUM PEAK ERRORS FOR VERIFICATION SET 2. BOTH ALUMINUM AND COPPER ARE CONSIDERED. ALL VALUES ARE GIVEN AS PERCENTAGES

	Aluminum			Copper		
core	mean	std	maxpk	mean	std	maxpk
Core 0	0.64	0.62	0.55	0.71	0.62	0.58
Core 1	0.53	0.57	0.21	0.63	0.59	0.22
Core 2	0.41	0.53	0.12	0.49	0.55	0.14
Core 3	0.58	0.65	0.39	0.64	0.68	0.22
Cache	0.58	0.50	1.08	0.63	0.52	1.05
hs(0mm)	0.55	0.32	0.32	0.20	0.17	0.52
hs(5mm)	0.67	0.41	0.91	0.43	0.33	1.17
hs(15mm)	0.39	0.30	1.10	0.82	0.30	0.48
hs(25mm)	0.63	0.32	0.98	0.11	0.08	0.12

follow the original waveforms very closely for both aluminum and copper heat sink materials.

Fig. 22 shows the temperature response waveforms of the heat sink taken at 0 and 5 mm away from the center using verification set 1. Once again, the generated models follow the trend of the original waveforms very closely. While the similarities may not be as pronounced at 5 mm, the differences are so minute that the resulting error percentages remain very small.

Tables IV and V provide the error statistics and maximum peak errors of verification sets 1 and 2. The mean error and standard deviation, as well as the maximum peak errors, are calculated for each core, the cache, and at distances of 0, 5, 15, and 25 mm from the center of the heat sink. Both aluminum and copper heat sink materials are also considered. Both tables show that the models generated by *ParThermSID* consistently exhibit less than 1% error in all three criteria.

In terms of CPU, *ParThermSID* is able to generate a model from training data in 5.52 s. The evaluation (1000 time steps) based on the parameterized model takes 0.48 s to finish. While *ParThermPOF* generates the similar model in 57.4 and the evaluation time (also 1000 time steps) is 0.21 s. Thus, *ParTermSID* is about 10× faster than *ParThermPOF*. The evaluation time of *ParThermPOF* is fast because the model is based on the partial fraction forms in terms of poles and zeros. By comparison, *FloTHERM*, which is a commercial analysis tool, performs modelling and simulation in about 34 min on a similar computing platform at Intel. As a result, we can see that *ParThermSID* is about two orders of magnitude faster than *FloTHERM* even though *ParThermsidSID* is implemented on MATLAB.

Lastly, Table VI provides a comparison between *ParThermSID* and *ParThermPOF*. Mean and maximum peak error comparisons are taken at specific parameters on the same quad-core microprocessor model using the same verification set. The second row, for example, which was taken at 5 mm from the center of an aluminum heat sink, shows that both models have less than 1% error in both criteria. The results show that *ParThermSID* can provide models of similar accuracy to *ParThermPOF*. However, while *ParThermPOF* is limited to using step/impulse responses for its training method,

TABLE VI
MEAN AND MAXIMUM PEAK ERROR COMPARISON BETWEEN *PARTHERMSID* AND *PARTHERMPOF* ON SEVERAL LOCATIONS CONSIDERING DIFFERENT PARAMETERS. ALL VALUES ARE GIVEN AS PERCENTAGES

	ParThermSID		*ParThermPOF*	
loc/param	mean	maxpk	mean	maxpk
hs(Al,0)	0.89	1.21	0.16	0.48
hs(Al,5)	0.83	0.59	0.49	0.35
hs(Al,15)	0.28	0.83	0.07	0.06
c0(Cu)	0.51	0.40	0.23	0.42
c2(Cu)	0.99	0.32	0.59	0.98
cache(Cu)	0.61	0.33	1.69	2.56
c1(Al)	0.52	0.03	1.28	0.28
c3(Al)	1.33	0.70	0.55	1.17

ParThermSID is able to utilize general input and measured output data for training as shown earlier. Furthermore, in the parameterizations process, *ParThermSID* is able to handle multi-input/multi-output systems, which consolidates the coefficient calculations in both linear and second-order equations.

VI. CONCLUSION

In this paper, we have proposed a new parameterized thermal behavioral modeling method. The new method, *ThermSID*, and its parameterized version, *ParThermSID*, build the parameterized dynamic thermal behavioral models from accurately computed thermal and power information. Compared to existing behavioral thermal modeling algorithms, the proposed method does not require transfer function like response for model training, which leads to greater flexibility during the modeling process. *ParThermSID* can include a number of parameters such as locations of thermal sensors in a heat sink, different components in packages, thermal conductivity of heat sink materials, etc. It is very suitable for thermal-related design space exploration and optimization where both dynamic behavior and design parameters need to be considered. Experimental results on a practical quad-core microprocessor have shown that the generated parameterized thermal models match the given power-thermal data very well. The compact models provided by *ParThermSID* offer two orders of magnitude speedup over the commercial thermal analysis tool *FloTHERM* on the given examples.

APPENDIX

Here, we give a brief introduction to the subspace identification method with some implementation details.

We suppose a dynamic thermal system is linear time-invariant (LTI) with noise as described in (1). For the system in (1), $h_k = CA^{k-1}B$ for $k \geq 1$ are called Markov parameters, which are the values of discrete-time impulse responses. The Hankel matrix of the Markov parameters can be defined as

$$\mathcal{H}_{i,j} = \begin{bmatrix} h_i & h_{i+1} & \dots & h_{i+j} \\ h_{i+1} & h_{i+2} & \dots & h_{i+j+1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{i+j} & h_{i+j+1} & \dots & h_{i+2j} \end{bmatrix}. \quad (10)$$

It can easily shown that

$$\mathcal{H}_{1,N-1} = \mathcal{M}_O(N)\mathcal{M}_C(N) \quad (11)$$

where $\mathcal{M}_C$ and $\mathcal{M}_O$ are the extended controlability and observability matrices, which have the following form:

$$\mathcal{M}_C(j) = [\,B \quad AB \quad \dots \quad A^{j-1}B\,]$$
$$\mathcal{M}_O(j) = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{j-1} \end{bmatrix}. \quad (12)$$

Considering s points of history and r points of future data, we first need to estimate the state variable $\hat{x}_k$ for the state x_k as

$$y_{\text{future}}(k) = \mathcal{M}_O(r)\hat{x}_k + S_{\text{future}}u_{\text{future}}(k) \quad (13)$$

where

$$y_{\text{future}}(k) = [y_k, y_{k+1} \dots, y_{k+r-1}]^T \quad (14)$$
$$u_{\text{future}}(k) = [u_k, u_{k+1} \dots, u_{k+r-1}]^T \quad (15)$$
$$S_{\text{future}} = \begin{bmatrix} 0 & 0 & \dots & 0 \\ CB & 0 & \dots & 0 \\ \vdots & & \ddots & 0 \\ CA^{r-2}B & CA^{r-3}B & \dots & 0 \end{bmatrix}. \quad (16)$$

It can be shown than we can approximate x_k in a least squares sense as [16]

$$\hat{x}_k = K_1 y_{\text{past}}(k-s) + K_2 u_{\text{past}}(k-s) + K_3 u_{\text{future}}(k) \quad (17)$$

where

$$y_{\text{past}}(k-s) = [y_{k-s}, y_{k-s+1}, \dots, y_{k-1}]^T$$
$$u_{\text{past}}(k-s) = [u_{k-s}, u_{k-s+1}, \dots, u_{k-1}]^T. \quad (18)$$

From (13) and (17), and use $\hat{x}_k$ in (13) to estimate $\hat{y}_{\text{future}}(k)$, $\hat{y}_{\text{future}}(k)$ becomes a linear combination of $y_{\text{past}}(k-s)$, $u_{\text{past}}(k-s)$ and $u_{\text{future}}(k)$. Rewrite it in matrix form as follows:

$$\hat{Y}^{\text{pred}}_{\text{future}} = L_1 Y_{\text{past}} + L_2 U_{\text{past}} + L_3 U_{\text{future}}, \quad (19)$$

where

$$\hat{Y}^{\text{pred}}_{\text{future}} = [\hat{Y}_{\text{future}}(s+1), \hat{Y}_{\text{future}}(s+2), \dots, \hat{Y}_{\text{future}}(N-r+1)]$$
$$Y_{\text{past}} = [y_{\text{past}}(1), y_{\text{past}}(2), \dots y_{\text{past}}(N-r-s+1)]$$
$$U_{\text{past}} = [u_{\text{past}}(1), u_{\text{past}}(2), \dots, u_{\text{past}}(N-r-s+1)]$$
$$U_{\text{future}} = [u_{\text{future}}(s+1), u_{\text{future}}(s+2), \dots, u_{\text{future}}(N-r+1)].$$

It can be proven that $[L_1\ L_2]$ will converge to $\mathcal{H}$, which is the Hankel matrix defined in (10). To obtain L_1 and L_2, we try to minimize

$$\min \left\| Y^{\text{data}}_{\text{future}} - \hat{Y}^{\text{pred}}_{\text{future}} \right\|^2_F \quad (20)$$

where $\|A\|^2_F = \text{Trace}(A * A)$.

Numerically, such optimization can be framed as a least square minimization by performing the orthogonal projection. Practically, such orthogonal projection is carried out by QR or LQ decomposition. From (19), we can also see that L_1 and L_2 are the projection of $\hat{Y}^{\text{pred}}_{\text{future}}$ onto Y_{past} and U_{past}, respectively. So, if we define data matrix

$$W = [Y^T_{\text{past}} \quad U^T_{\text{past}} \quad Y^T_{\text{future}}]^T \tag{21}$$

then the solution of the optimization problem in (20) can be obtained by LQ-factorization of W as

$$W = \begin{bmatrix} * & 0 & 0 \\ * & * & 0 \\ L_1 & L_2 & * \end{bmatrix} [\,Q_1 \quad Q_2 \quad Q_3\,]. \tag{22}$$

We perform SVD on $\mathcal{H} = [L_1 L_2] = U\Sigma V^*$, where Σ is used to identify the system order n. Afterwards, we can define the first n columns of U and V as U_1 and V_1, respectively, and the diagonal matrix with the first n singular values in Σ as Σ_1. Then, we define

$$\mathcal{M}_O = U_1 \Sigma_1^{1/2}, \quad \mathcal{M}_C = \Sigma_1^{1/2} V_1^*. \tag{23}$$

The system matrix C is the first n rows of $\mathcal{M}_O$, and B can be retrieved from the first n columns of $\mathcal{M}_C$. The matrix A can be obtained by

$$A = (\mathcal{M}_O^{\downarrow})^{+} \mathcal{M}_O^{\uparrow} \tag{24}$$

where X^+ means the pseudoinverse operation on X as $\mathcal{M}_O^{\downarrow}$ may not be a square matrix and invertible, and

$$\mathcal{M}_O^{\uparrow} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{M-1} \end{bmatrix} \quad \mathcal{M}_O^{\downarrow} = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{M-2} \end{bmatrix}. \tag{25}$$

REFERENCES

[1] Int. Technol. Roadmap for Semiconductors(ITRS) 2007 Ed. 2007 [Online]. Available: http://public.itrs.net

[2] A. Augustin, B. Maj, and A. Kostka, "A structure oriented compact thermal model for multiple heat source ASICs," *Mircroelectron. J.*, vol. 36, no. 8, pp. 700–704, Aug. 2005.

[3] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *Proc. Int. Symp. High-Performance Comp. Architecture*, 2001, pp. 171–182.

[4] X. Chen, R. P. Dick, and L. Shang, "Properties of and improvements to time-domain dynamic thermal analysis algorithms," in *Proc. Eur. Design and Test Conf. (DATE)*, Mar. 2010, pp. 1165–1170.

[5] F. Christiaens, B. Vandevelde, E. Beyne, R. Mertens, and J. Berghmans, "A generic methodology for deriving compact dynamic thermal models, applied to the PSGA package," *IEEE Trans. Compon., Packaging, Manuf. Technol. A*, vol. 21, no. 4, pp. 565–576, Dec. 1998.

[6] G. F. Franklin, M. L. Workman, and D. Powell, *Digital Control of Dynamic Systems*. Boston, MA: Addison-Wesley Longman, 1997.

[7] Y. C. Gerstenmaier and G. Wachutka, "Rigorous model and network for transient thermal problems," *Mircroelectron. J.*, vol. 33, pp. 719–725, Sep. 2002.

[8] S. Gunther, F. Binns, D. Carmean, and J. Hall, "Managing the impact of increasing microprocessor power consumption," *Intel Technol. J.*, 2001, 1st Quarter.

[9] W. Huang, M. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy, "Compact thermal modeling for temperature-aware design," in *Proc. Design Autom. Conf. (DAC)*, 2004, pp. 878–883.

[10] M. J. Kearns and U. V. Vazirani, *An Introduction to Computational Learning Theory*. Cambridge, MA: MIT, 1994.

[11] C. Lasance, H. Vinke, H. Rosten, and K.-L. Weiner, "A novel approach for the thermal characterization of electronic parts," in *Proc. IEEE 11th Annu. Semicond. Thermal Meas. Manag. Symp.*, 1995, pp. 1–9.

[12] D. Li, S. X.-D. Tan, E. H. Pacheco, and M. Tirumala, "Architecture-level thermal characterization for multi-core microprocessors," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 17, no. 10, pp. 1495–1507, Oct. 2009.

[13] D. Li, S. X.-D. Tan, E. H. Pacheco, and M. Tirumala, "Parameterized architecture-level dynamic thermal models for multicore microprocessors," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 15, no. 2, pp. 1–22, 2010.

[14] P. Liu, H. Li, L. Jin, W. Wu, S. X.-D. Tan, and J. Yang, "Fast thermal simulation for runtime temperature tracking and management," *IEEE Trans. Comput.-Aided Design (CAD) Integr. Circuits Syst.*, vol. 25, no. 12, pp. 2882–2893, Dec. 2006.

[15] R. H. Myers and D. C. Montgomery, *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. New York: Wiley-Interscience, 2002.

[16] P. V. Overschee and B. D. Moor, "N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems," *Automatica*, vol. 30, no. 1, pp. 75–93, 1994.

[17] H. Pape, D. Schweitzer, J. H. Janssen, A. Morelli, and C. M. Villa, "Thermal transient modeling and experimental validation in the european project PROFIT," *IEEE Trans. Compon. Packag. Technol.*, vol. 27, no. 3, pp. 530–538, Sep. 2004.

[18] P. D. Pedrod, "Bayesian averaging of classifiers and the overfitting problem," in *Proc. 17th Int. Conf. Machine Learning*, 2000, pp. 223–230.

[19] M. Rencz, G. Farkas, V. Székely, A. Poppe, and B. Courtois, "Boundary condition independent dynamic compact models of packages and heat sinks from thermal transient measurements," in *Proc. 5th Electron. Packaging Technol. Conf.*, 2003, pp. 479–484.

[20] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature aware microarchitecture," in *Proc. IEEE Int. Symp. Comput. Architecture (ISCA)*, 2003, pp. 2–13.

[21] Y. Yang, Z. P. Gu, C. Zhu, R. P. Dick, and L. Shang, "ISAC: Integrated space and time adaptive chip-package thermal analysis," *IEEE Trans. Comput.-Aided Design (CAD) Integr. Circuits Syst.*, vol. 16, no. 1, pp. 86–99, 2007.

Thom J. Eguia received the B.S. degree in electronic engineering and M.S. degree in electrical engineering from the University of California, Riverside, in 2008 and 2010, respectively.

He is currently a Design Engineer with West Digital Corporation, Irvine, CA. His research interests include thermal modeling and simulation for multicore microprocessors.

Sheldon X.-D. Tan (S'96–M'99–SM'06) received the B.S. and M.S. degrees in electrical engineering from Fudan University, Shanghai, China, in 1992 and 1995, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Iowa, Iowa City, in 1999.

He is currently a Professor with the Department of Electrical Engineering, University of California, Riverside. He was a faculty member with the Electrical Engineering Department, Fudan University, Shanghai, China, from 1995 to 1996. His research interests include modeling and simulation of analog/RF/mixed-signal and interconnect circuits, analysis and optimization of high-performance power and clock distribution networks, and architecture-level thermal, power, modeling, and simulation for multicore microprocessors and embedded system designs based on FPGA platforms. He also coauthored the books *Symbolic Analysis and Reduction of VLSI Circuits* (Springer/Kluwer, 2005) and *Advanced Model Order Reduction Techniques for VLSI Designs* (Cambridge University Press, 2007). He is currently is serving as an associate editor for *ACM Transactions on Design Automation of Electronic Systems* (TODAE), *Integration, The VLSI Journal,* and *Journal of VLSI Design.*

Dr. Tan was the recipient of the Outstanding Oversea Investigator Award from the National Natural Science Foundation of China (NSFC) in 2008, the National Science Foundation CAREER Award in 2004, the Best Paper Award from the 2007 IEEE International Conference on Computer Design (ICCD'07), a Best Paper Award Nomination from the 2005 and 2009 IEEE/ACM Design Automation Conferences, and the Best Paper Award from the 1999 IEEE/ACM Design Automation Conference. He served as a technical program committee member for DAC, ICCAD, ASPDAC, ASPDAC, BMAS, and ISQED.

Ruijing Shen (S'08) received the B.S. degree in electronic engineering and the M.S. degree in institute of Microelectronics from Tsinghua University, Beijing, China, in 2005 and 2007, respectively. Now she is a Ph.D. candidate in the department of electrical engineering in University of California, Riverside. Her current research interests include software thermal modeling for mobile computing system, statistical leakage analysis of VLSI, and Variational Capacitance Extraction and Modeling.

Duo Li (S'06) received the B.S. degree in computer science from Northeastern University, Shenyang, China, in 2003, the M.S. degree in computer science from Tsinghua University, Beijing, China, in 2006, and the Ph.D. degree in electrical engineering from the University of California, Riverside, in 2010.

He is currently with the IC Compiler (ICC) Timing Optimization Group, Synopsys Corporation, Mountain View, CA. His current research interests include model order reduction, fast circuit simulation, thermal modeling, thermal simulation, and static timing analysis.

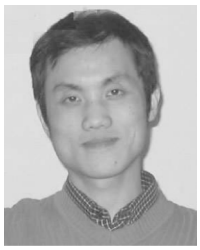
Eduardo H. Pacheco received the B.S. degree in energy from the Metropolitan University, Mexico City, Mexico , in 1994, the M.S. degree in solar energy from the National University of Mexico, Mexico City, Mexico, in 1998, the M.S. degree in mechanical engineering from the State University of New York, Stony Brook, in 2002, and the Ph.D. degree in chemical engineering from the University of North Dakota, Grand Forks, in 2004.

He was a Research Assistant with the Energy and Environmental Research Center in North Dakota, where he developed an electrothermal model for high-temperature fuel cells. He is currently with Intel Corporation, Hillsboro, OR, as a Thermal Engineer in the Mobility Group, and previously as a Thermal Engineer in Intel's Corporate Technology Group, Guadalajara, Mexico. His areas of interest are thermal management, heat transfer in human tissue, and high-temperature fuel-cell modeling. He has been a reviewer for the *Journal of Power Sources* and the *International Journal of Hydrogen Energy*.

Murli Tirumala received the B.Tech. degree from Osmania University, Hyderabad, India, in 1980, and the M.S. and Ph.D. degrees from Rensselaer Polytechnic Institute, Troy, NY, in 1986, all in chemical engineering. He specialized in phase change heat transfer and interfacial phenomena in thin films during his graduate and doctoral work.

He is a Principal Engineer with the Systems Technology Laboratory within Intel's Corporate Technology Group, Hillsboro, OR. In this capacity, he is responsible for researching and developing physical technologies and methodologies for future Intel platforms. He is also responsible for setting technology directions to enable cost effective solutions in thermal, acoustic, and system packaging disciplines. He has authored several technical papers in the areas of thermal management.

Dr. Tirumala is a member of Sigma Xi. He served as an associate editor of the IEEE TRANSACTIONS ON COMPONENTS AND PACKAGING TECHNOLOGIES and ASME's K16 committee.



Lingli Wang (M'99) received the M.S. degree from Zhejiang University, Hangzhou, China, in 1997, and the Ph.D. degree from Edinburgh Napier University, Edinburgh, U.K., in 2001, both in electrical engineering.

He was with Altera European Technology Center for four years. In 2005, he joined Fudan University, Shanghai, China, where he is currently a Full Professor with the State Key Lab of ASIC & Systems in the School of Microelectronics. His research interests include logic synthesis, reconfigurable computing, and quantum computing.