ABSTRACT

In this paper, we study a new architecture level thermal modeling problem from behavioral modeling perspective to address the emerging thermal related analysis and optimization problems for high-performance quad-core microprocessor designs. We propose a new approach to build the thermal behavioral models by using transfer function matrix from the measured thermal and power information at the architecture level. The new method builds behavioral thermal model using generalized pencil-of-function (GPOF) method, which was developed in the communication community to build the rational modeling from the measured data of real-time systems. To effectively model transient temperature changes, we propose two new schemes to improve the GPOF. First we apply logarithmic-scale sampling instead of traditional linear sampling to better capture the temperature changing characteristics. Second, we modify the extracted thermal impulse response such that the extracted poles from GPOF are guaranteed to be stable without accuracy loss. Experimental results on a practical quad-core microprocessor show that generated thermal behavioral models match the measured data very well.

1. INTRODUCTION

As CMOS technology is scaled into the nanometer region, the power density of high-performance microprocessors will increase drastically. The exponential power density increase will in turn lead to average chip temperature to raise rapidly [2]. Higher temperature has significant adverse impacts on chip performance and reliability. Excessive on-chip temperature leads to slower transistor speed, more leakage power consumption, higher interconnect resistance, and reduced reliability.

One way to mitigate the high temperature problem to put multiple CPU or cores into one single chip [9, 1, 3]. In this way, one can simply increase the total throughput by parallel computation, and have lower voltage and frequency to meet thermal constraints. But the thermal effects are influenced by the placement of cores and caches. So it is very important to consider the temperature during the floorplanning and architecture design of multi-core microprocessor.

The estimated temperature at the architecture level can then be used to perform power, performance, and reliability analysis, together with floorplanning and packaging design [12]. As a result, design decision is guided by temperature and design is optimized theoretically without potential thermal problems. To facilitate this temperature-aware architecture design, it is important to have accurate and fast thermal estimation at the architecture level. Both architecture and CAD tool community are currently lacking reliable and practical tools for thermal architecture modeling. Existing work on the HotSpot project [8, 12] tried to resolve this problem by generating the architecture thermal model in a bottom-up way based on the floorplanning of the function units. But this method is difficult to set up for new architecture with different thermal and packaging configurations. Also the resulting model work for only single CPU architecture and the accuracy may not sufficient as many approximations are made.

This paper, we propose a new thermal behavioral modeling approach for fast temperature estimation at the quad-core thermal architecture level at early design stage. The new approach builds the transfer function matrix from the measured architecture level thermal and power information. It first builds behavioral thermal model using generalized pencil-of-function (GPOF) method [6, 7, 11], which was developed in the communication community to build the rational modeling from the measured data of real-time and electromagnetism systems. However, direct use of GPOF will not generate stable useful thermal models. Based on the characteristics of transient temperature behaviors, we make two new improvements over the traditional GPOF: First we apply logarithmic-scale sampling instead of traditional linear sampling to better capture the temperatures change over the time. Second, we modify the extracted thermal impulse response such that the extracted poles from GPOF are guaranteed to be stable without accuracy loss. Experimental results on a practical quad-core microprocessor show that the generated thermal behavioral models can be built very efficient and the resulting model match well the measured temperature for non-training data.

The rest of this paper is organized as the follows: Section 2 presents thermal modeling problem we try to solve. Section 3 reviews a generalized pencil-of-function (GPOF) method for extracting the poles and residues from the transient response of a real-time system and electromagnetism. Section 4 presents our new thermal behavioral modeling approach based on the GPOF. Section 5 presents the experimental results and Section 6 concludes this paper.

2. ARCHITECTURE-LEVEL THERMAL MODELING PROBLEM

We first present the new thermal behavioral modeling problem. Basically we want to build the behavioral model, which is excited by the power input and product the temperature outputs for the specific locations in the floorplan of the quad-core microprocessor. Our behavioral models are created and calibrated with the measured temperature and
sient temperature can be then computed due to input port $i$ (3). 

function matrix

where $h_{ij}$ is the impulse response function for output port $j$ due to input port $i$.

Given a power input vector for each core $\mathbf{u}(t)$, the transient temperature can be then computed

$$y(t) = \int_0^t H(t - \tau)u(\tau)d\tau$$

Equation (2) can be written in frequency domain as in (3).

$$y(s) = \mathbf{H}(s)\mathbf{u}(s)$$

where $y(s)$, $\mathbf{u}(s)$ and $\mathbf{H}(s)$ are the Laplace transform of $y(t)$, $\mathbf{u}(t)$ and $\mathbf{H}(t)$, respectively. $\mathbf{H}(s)$ is called the transfer-function matrix of the system where each $h_{ij}(s)$ can be represented as the partial fraction form or the pole-residue form

$$h_{ij}(s) = \sum_{k=1}^n \frac{\tau_k}{s - p_k}$$

where $h_{ij}(s)$ is the transfer function between the $j$th input terminal and the $i$th output terminal; $p_k$ and $r_k$ are the $k$th pole and residue. Once transfer functions are computed, the transient responses can be easily computed.

The remaining important problem is to find the poles and residues for each transfer function $h_{ij}$ from the measured thermal and power information. It turns out the generalized pencil-of-function can be used for this propose. But we cannot simply apply GPOF method as we show in the Section 4. In the following section, we will briefly review the GPOF method before we present our improvements.

### 3. REVIEW OF GENERALIZED PENCIL-OF-FUNCTION METHOD

Generalized pencil-of-function (GPOF) method can be used to extract the poles and residues from the transient response of a real-time system and electromagnetism [6, 7, 11]. Specifically, GPOF can work for such a system that can be expressed in sum of complex exponentials:

$$y_k = \sum_{i=1}^{M} r_i e^{(p_i \Delta t)}$$

where, $k = 0, 1, ..., N-1$, is the number of sampled points, $r_i$ is the complex residues, $p_i$ are the complex poles, and $\Delta t$ is the sampling interval. $M$ is the number poles we used to build the transfer function. Let’s define

$$z_i = e^{(p_i \Delta t)}$$

which become the poles in Z-plane. For real value $y_k$, both $r_i$ and $p_i$ should be in complex conjugate pairs. Let’s define the new vector of node temperatures (in our problem) as $\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}_L$ where,

$$\mathbf{y}_i = [y_i, y_{i+1}, ..., y_{i+N-L-1}]^T$$

where $L$ can be viewed as sampling window size. Based on these vectors, we can define the matrices $Y_1$ and $Y_2$ as

$$Y_1 = [\mathbf{y}_0, \mathbf{y}_1, ..., \mathbf{y}_L-1]$$

$$Y_2 = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_L]$$

Then one can obtain the following relationship among the $Y_1$, $Y_2$ and the pole and residue vectors $Z_0$ and $R$ based on the structure of $Y_1$, $Y_2$:

$$Y_1 = Z_1 R Z_2$$

$$Y_2 = Z_1 R Z_0 Z_2$$

where

$$Z_1 = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ z_1 & z_2 & \ldots & z_M \\ \vdots & \vdots & \ddots & \vdots \\ Z_1^{N-L-1} \end{bmatrix}$$

$$Z_2 = \begin{bmatrix} 1 & z_1 & \ldots & z_1^{L-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & z_M & \ldots & Z_M^{L-1} \end{bmatrix}$$

$$Z_0 = \text{diag}[z_1, z_2, ..., z_M]$$

$$R = \text{diag}[r_1, r_2, ..., r_M]$$

Figure 1: Quad-core architecture

Figure 2: Abstrated system

Figure 3: Power Temperature

Power
p0 p1 p2 p3 p4
Temperature
0 1 2 3 4
Thermal
System
DIE
Heat spreader
TIM2
Heat sink
TIM1
CACHE

DIE
TIM1
Heat spreader
TIM2
Heat sink
TIM1
CACHE

1 cm
1 cm
So the problem we need to solve is to find the pole and residue vector \( Z_0 \) and \( R \) efficiently. It turns out that this can be easily computed by observing that

\[
Y_1^+ Y_2 = Z_1^+ R^{-1} Z_1^+ Z_1 R Z_0 Z_2 \\
Z_1^+ Z_0 Z_2 
\]  
(16)

Hence, the poles is the eigenvalues of \( Y_1^+ Y_2 \), where + indicate the (Moore-Penrose) pseudo-inverse, as \( Y_1 \) is not a square matrix. As a result, one can obtain the \( Z_0 \) by using

\[
Z = D^{-1} U M V 
\]  
(17)

where \( Z \) is a \( M \times M \) matrix and \( V \) and \( U \) comes from the singular value decomposition (SVD) of \( Y_1 \):

\[
Y_1 = U D V^H 
\]  
(18)

After the \( Z \) is computed, we can obtain the pole vector \( Z_0 \) by performing the eigen-decomposition of \( Z \), \( Z_0 = \text{eig}(Z) \), where \( \text{eig}(X) \) is to obtain the eigenvalue vector from matrix \( X \). Once \( Z_0 \) is obtained, we can compute the residue vector \( R \) by using either (10) or (11).

For GPOF method, it allows \( M \leq L \leq N - M \), which means that we can allow the different window size and pole numbers. Typically, choosing \( L = N/2 \) can yield better results.

4. NEW ARCHITECTURE-LEVEL THERMAL BEHAVIORAL MODELING METHOD

In this section, we present our new thermal behavioral modeling approach based on the GPOF method mentioned in the previous section.

For a linear time-invariant system, the sum of complex exponential form shown in (5) essentially is the impulse response in the s-domain. So we need to apply the GPOF method to the thermal impulse responses, which in general cannot be obtained directly from measurement. Instead, we measure the thermal step responses for each core (center of the core) excited by the same power inputs in the given multi-core microprocessor. Then impulse responses are obtained by performing the numerical differentiation of the step responses.

But directly applying the GPOF to the computed thermal impulse responses (from the measured data) may not lead to accurate and stable models as shown below. In the following, we will present two improvement schemes in the new method such that the resulting models are accurate and stable.

4.1 Logarithmic scale sampling for poles and residues extraction

The first problem we face for the thermal modeling is that linear sampling in the traditional GPOF method does not work for our thermal data.

According to GPOF method reviewed in Section 3, we know that matrices \( Y_1 \) and \( Y_2 \) are constructed from the sampled data and the sampling time interval \( \Delta t \) must be the same. However, how to obtain sample data from the observed temperature curve became a big issue in our CPU temperature simulation, because the step temperature response often goes up drastically in the first few seconds and gradually tends to reach a steady state after a relatively long time.

This can be observed in Fig. 3(a), which is step temperature responses for \( \text{core0} \) (die : 0) when only \( \text{core0} \) is driven by a step 20W power sources beginning at \( t = 0 \) (which is called active in this paper). The environment temperature (initial temperature when no input power at the beginning) is 35°C. We observe that almost all the temperature increase occurs within the first second, from 35°C to 57.9°C, where 61.1°C is the final temperature when \( \text{core0} \) reaches a steady state.

\[
y'(t) = y(ln(t) + t_0) 
\]  
(19)

where \( y'(t) \) is the response in normal time scale \( y(t) \) is the response in log-scale; \( t_0 \) is the offset for this transfer function.

4.2 Stable poles and residues extraction

4.2.1 Stable pole extraction

The second problem with the GPOF method is that it will not always generate stable poles for a given impulse response. Actually GPOF model can give a very good matching for a given impulse response for the sampled interval while using positive poles. But outside the sampled interval, the response from the model by GPOF can be unbounded due to the positive poles.

Fig. 4(a) shows the extracted impulse response compared to the original one for one of the cores. For this example, the sampled time interval is from 0 to 1000 seconds. Except for
the very beginning (we will address this issue later), it can be seen that the computed model matches very well with the original model from time 0 to the 1000s (the corresponding \( x = 18.55 \) in log-scaled x-axis with offset being 11.64). But outside the time interval, if we extend the time scale to \( 10^{10} \) seconds, they are significant difference between the two models. The computed models does not look like an impulse responses and will go unbounded actually owning to the positive poles. Fig. 5(a) shows the extracted poles where not all the poles extracted by GPOF are stable (negative real parts).

![Figure 4: Unstable and stable impulse response for Core0](image)

(a) Extracted impulse response with positive poles  
(b) Extracted impulse response with only negative poles

**Figure 4: Unstable and stable impulse response for Core0**

![Figure 5: Poles distributions of unstable and stable extracted transfer function](image)

(a) Extracted poles with positive poles  
(b) Extracted poles with only negative poles

**Figure 5: Poles distributions of unstable and stable extracted transfer function**

To mitigate this problem, we propose to extend the time interval for zero-response time. For any impulse response, after sufficient time, the response will become zero (or numerically become zero) as the area integration of the impulse curve below is a constant. By sufficiently extending the time interval for zero-response time in a impulse response, we can make all the poles stable. The reason is that if we have positive poles, after sufficient long time, the response will always go non-zero and eventually become unbounded assuming all the poles are different numerically, which is always true practically. If we ensure the zero response for sufficient long time, all the poles must be stable to have the zero responses for very long time as response contributed by those poles will decay to zero.

Using the same example, if we extend the time interval to \( 10^{10} \) seconds, which actually does not increase significantly in log-scale, all the extracted poles become stable. Fig. 5(b) shows the extracted poles by extending zero-response time to \( 10^{10} \)s where all the poles are stable (with negative real part) and Fig. 4(b) shows the extracted stable impulse response. For different problems, we may need to find such a sufficient time period. For all our problem, we find \( 10^{10} \)s seems such a good sufficient time for our example.

### 4.2.2 Stabilizing the starting response

After the log-scale sampling and numerical differentiation, the obtained impulse response become zero numerically for a short period as temperature changes at the very beginning is very slow. For example, we consider the temperature of core1 when only core0 is active. Assume that core0 is active at \( t = 0 \), in the first very short time, such as \( t = 10^{-4} \)s, temperature response of core1, due to the delay in thermal transmission, is probably still 0 and it may begin to increase at \( t = 10^{-3} \)s. Normally we consider the difference \( 10^{-4} \)s and \( 10^{-3} \)s as a small value, but in log-scale, this difference is translated to a period of time with zero responses at the beginning. And long zero-response time at beginning may cause the significant discrepancies as shown in Fig. 6(a), although the computed response tends to be accurate after some time period. This means this transfer function we obtained is not accurate enough. Fig. 6(b) shows a step response computed by the transfer function obtained in Fig. 6(a). Obviously, it has noticeable difference compared to the original one.

![Figure 6: Impulse and step response computed by inaccurate model with large error in the starting time.](image)

(a) Impulse response with large errors in the starting time.  
(b) The corresponding step response for the inaccurate model. Here the zero temperature means the room temperature.

**Figure 6: Impulse and step response computed by inaccurate model with large error in the starting time.**

The reason for this problem is that the log-scaled impulse response is different than the typically impulse response from physical RLC circuit in which the response goes to non-zero immediately after \( t = 0 \). To resolve this problem, we propose to truncate the beginning zero-response time such that responses go to non-zero numerically immediately. This can be achieved by setting threshold temperature to locate the new zero time. During the simulation process, in all the actual time before the new zero time, the response will be set to zero. Fig. 7 shows the impulse and step response computed by accurate model after truncating the beginning zeros.

### 5. Experimental Results

The proposed new algorithm has been implemented in Matlab 7.0 and tested on the quad-core microprocessor architecture shown in Fig. 1 from our industry partner. We first extracted transfer function matrix of the system through a training data set, which consists of the step responses for each core from other cores. After extracting the transfer functions, we could apply them to compute the thermal responses from any power inputs with any time varying inputs.
Our experimental data are each core’s temperatures measured directly from the center of the dies as we introduced in Section 2, which are provided by our industry partner. At the beginning all the cores are in zero state and have an initial environmental temperature 35°C. From $t = 0$ each core is excited by a step power input of 20W simultaneously. And the temperature of each core is collected from 0s to 1000s.

Now we verify the correctness of our model based on the given thermal data from our industry partner. For each transfer function, we set an order of 50. This is already enough for our model. In practice, temperature on each core or cache can be computed very fast by our model during any time interval. Because our model is directly based on the transfer function represented by poles and residues instead of state space equations.

We show our comparison results of core0 and cache in Fig. 8 and Fig. 9 under normal linear time scale and log-scale, respectively. The red solid curve represents the measured temperature and the blue curve represents our computed temperature. The simulation runs very fast and costs only few minutes. From these result figures, we can see that our model has very good accuracy. Actually, the temperatures of other cores match well too. We could not show them here due to limit of the maximum pages.

In Table 1 we show the temperatures when all the cores achieve the steady state and the differences percentage. The difference is only around 0.2%. Furthermore, Table 2 shows some statistical features of the differences over all the sampling time points, including the maximums, the means and the standard deviations. Also, the maximum and average percentages are given. From this table we can see the maximum difference is less than 0.5°C and 1% and the average difference is less than 0.3°C and 0.3% for all the cores.

### 6. CONCLUSION

In this paper, we proposed a new architecture level thermal behavioral modeling method. The new method, builds the thermal behavioral models by using transfer function matrix from the measured architecture thermal and power information. We applied the generalized pencil-of-function (GPOF) method to construct the impulse response functions from the measured thermal data. To effectively model the transient temperature changes, we have proposed two new schemes to improve the GPOF. First we applied logarithmic-scale sampling instead of traditional linear sampling to better capture the temperature changing characteristics. Sec-

<table>
<thead>
<tr>
<th>Table 1: Difference when temperatures achieve the steady state</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Core0</td>
</tr>
<tr>
<td>Core1</td>
</tr>
<tr>
<td>Core2</td>
</tr>
<tr>
<td>Core3</td>
</tr>
<tr>
<td>Cache</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 2: Features of the difference between measured and computed temperatures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Core0</td>
</tr>
<tr>
<td>Core1</td>
</tr>
<tr>
<td>Core2</td>
</tr>
<tr>
<td>Core3</td>
</tr>
<tr>
<td>Cache</td>
</tr>
</tbody>
</table>

Figure 7: Impulse and step response computed by accurate model with both improvements.

(a) Impulse response.  
(b) Step response.

Figure 8: Comparison results of core0’s temperature when all cores are active (driven by 20W powers).

(a) Temperature response of core0 in linear scale.  
(b) Temperature response of core0 in log-scale.

Figure 9: Comparison results of cache’s temperature when all cores are active (driven by 20W powers).

(a) Temperature response of cache in linear scale.  
(b) Temperature response of cache in log-scale.
ond, we modified the extracted thermal impulse response such that the extracted poles from GPOF are guaranteed to be stable without accuracy loss. Experimental results on a practical quad-core microprocessor have demonstrated that generated thermal behavioral models match the measured data very well. The proposed method can also be applied to thermal modeling of VLSI circuits at different granularity.

7. REFERENCES