Full-Chip Thermal Analysis for the Early Design Stage via Generalized Integral Transforms

Pei-Yu Huang and Yu-Min Lee, *Member, IEEE*

***Abstract*—The capability of predicting the temperature profile is critically important for timing estimation, leakage reduction, power estimation, hotspot avoidance and reliability concerns during modern IC design. This paper presents an accurate and fast analytical full-chip thermal simulator for early-stage temperature-aware chip design. By using the generalized integral transforms (GIT), an accurate formulation is derived to estimate the temperature distribution of full-chip with a truncated set of spatial bases which only needs very small truncation points. Then, we develop a fast Fourier transform like evaluating algorithm to efficiently evaluate the derived formulation. Experimental results confirm that the proposed GIT-based analyzer can achieve an order of magnitude speedup compared with a highly efficient Green's function-based thermal simulator. Finally, we propose a 3-D IC thermal simulator and demonstrate its efficiency and accuracy.**

***Index Terms*—Circuit simulation, generalized integral transforms (GITs), physical design, simulation, thermal analysis, 3-D IC.**

I. Introduction

THE power density of VLSI circuits increases monotonically as the CMOS technology scales down. The power dissipated by the circuits converts into heat. As a result, it raises the temperature of dies and induces hot spots. These thermal-related phenomena significantly degrade the performance and reliability of circuits [1]–[16]. For example, the resistance of copper interconnect increases 39% as the temperature rises from 20 °C to 120 °C, and the mean-time-to-failure of the interconnect exponentially decreases as the temperature increases [1]. To precisely predict the thermal impacts on design performance, an efficiently and accurately thermal analyzer is necessary in the temperature-aware design flow because it is usually a part of simulation kernel in the optimization loop and needs to be executed numerous times.

The thermal simulators can be categorized into two classes, numerical and analytical methods. The numerical methods use the finite difference method or the finite-element method (FEM) to transfer heat equations to resistance–capacitance (*RC*) network equations. Based on the *RC* network equations, several methods have been proposed to save the runtime. Wang *et al.* [2] utilized the alternating-direction-implicit method to split the equivalent *RC* system into different alternating directions, and alternately performed the line smooth scheme in each direction. In [3], the model order reduction technique was employed to improve the efficiency of transient analysis. Li *et al.* [4] applied the multi-grid method to speed up the convergence rate of iterative methods, and developed an order reduction scheme to save the runtime of dynamic thermal simulation. Because of the flexibility for dealing with the complicated structure, the numerical framework is the main stream in back-end design stages such as the post layout thermal verification.

As pointed out in [1], [5], and [6], temperature-aware design should be brought to early design stages such as thermal-aware floor-planning and placement. To give a reasonably accurate temperature prediction with little computational effort, [1] proposed a compact thermal model which modeled the package and interconnect layers as effective heat transfer coefficients for the boundary conditions of die. With the modeled heat transfer coefficients for the heat sink, prelayout interconnect and package, recently, an efficiently numerical thermal simulator developed by Yong *et al.* [7] is very suitable for early temperature-aware design stages. Because their simulator applies an adaptive discretization algorithm for spatial and temporal domains to analyze the temperature profile without degrading the accuracy, the number of temperature variables and simulating time steps can be significantly reduced.

The other category of thermal simulators being suitable for early design stages is the analytical method. The primary advantage of analytical approaches is that they avoid the volume meshing procedure of entire substrate, and have closed-form representations for the temperature distribution of the entire die. Hence, they are flexible to obtain the temperature distribution of certain user-specified regions without performing the thermal simulation for the entire die. Furthermore, based on the closed-form representations, the fast temperature evaluation of the die can be achieved for early design stages.

One analytical thermal solver is the Green's function-based method [6]. First, the steady-state Green's function of chip with a unity impulse power source is calculated. After that, its steady state temperature distribution with arbitrary power source map is got by taking the convolution of Green's function and its power density distribution with a table lookup method. To enhance the efficiency for lots of power sources, they used a series of cosine waveforms to approximate the power density map, and the temperature map of all grid cells were cast into the form of discrete cosine transform (DCT). Although their computational cost is $O(MN \log_2 MN)$, where M and N are numbers of divisions in the power density map along x- and y-directions, respectively, they can only provide the steady state thermal simu-

Manuscript received August 22, 2007; revised December 31, 2007. First published March 10, 2009; current version published April 22, 2009. This work was supported in part by National Science Council of Taiwan and by SoC Research Center of NCTU.

The authors are with the Department of Communication Engineering, National Chiao Tung University, Hsinchu 300, Taiwan, R.O.C. (e-mail: yumin@cm.nctu.edu.tw).

Digital Object Identifier 10.1109/TVLSI.2008.2006043

1063-8210/$25.00 © 2009 IEEE

lation. However, the dynamic thermal analysis is also necessary while performing the dynamic thermal management and run-time thermal analysis [1], [4], [7]. Moreover, as indicated in [6], a large number of truncation points for the Green's function is usually required to achieve an accurate solution.

To overcome these shortcomings, our major contributions are as follows.

- Compared with a highly efficient Green's function-based method [6], we improve the bound of the error decaying rate of analytical solution for the steady state temperature distribution and provide a transient temperature simulation by utilizing the generalized integral transforms (GITs) [17]–[19] to construct a set of spatial bases and calculate their time-varying coefficients. The proposed method can accurately estimate the temperature distribution of full-chip with very small truncation points (N_x and N_y) of spatial bases. The experimental results presented in Section V show that $N_x N_y$ can be far less than MN without losing any accuracy compared with [6].
- We develop a fast Fourier transform (FFT) like evaluating algorithm to efficiently evaluate the temperature map of all grid cells, and its computational cost is in the order of $O(MN \log_2 N_x N_y)$, where N_x and N_y are truncation points of bases in the x- and y-directions, respectively.
- We build an efficient 3-D IC thermal simulator by combining the GIT and numerical schemes, and its efficiency and accuracy are demonstrated by experimental results. Moreover, this hybrid scheme can be used to get more accurate temperature distribution with considering the different thermal conductivity of each stacked layer for the primary and secondary heat flow paths.

This paper is organized as follows. First, the thermal model for early design stages is presented in Section II. The GIT-based computational formula for the full-chip thermal simulation and the proposed evaluating algorithms are described in Section III. After that, the hybrid scheme of GIT-based thermal simulation method for the 3-D ICs and the package structures is addressed in Section IV. Finally, the experimental results and conclusions are given in Sections V and VI, respectively.

II. Thermal Modeling for Early Design Stages

A compact thermal structure of the chip, as illustrated in Fig. 1, can be used for early design stages. This model consists of three portions [1]: the primary heat flow path, the secondary heat flow path, and the heat transfer characteristic of each macro/block on the silicon die. The primary heat flow path is composed of thermal interface material, heat spreader and heat sink. The secondary heat flow path contains interconnect layers, input/output (I/O) pads and the print circuit board (PCB). The functional blocks are modeled as many power generating sources attached to a thin layer close to the top surface of die with the thickness being equal to the junction depth.[1] The major concerns of early-stage temperature-aware optimization procedure are to reduce the temperature or the thermal gradient of die. Here, we focus on estimating the temperature distribution.

[1]Because major part of currents only passes through the channel, this approximation is more reasonable than setting the power generating sources distributed to the entire die.

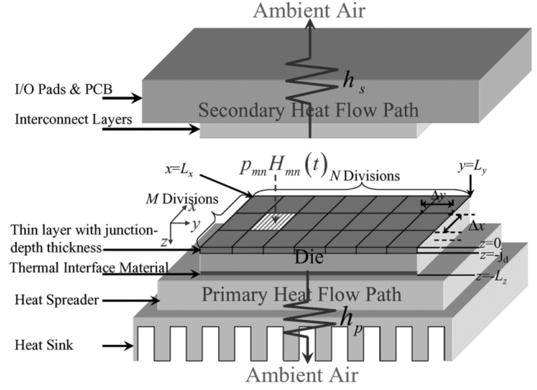

Fig. 1. Compact thermal model for early design stages.

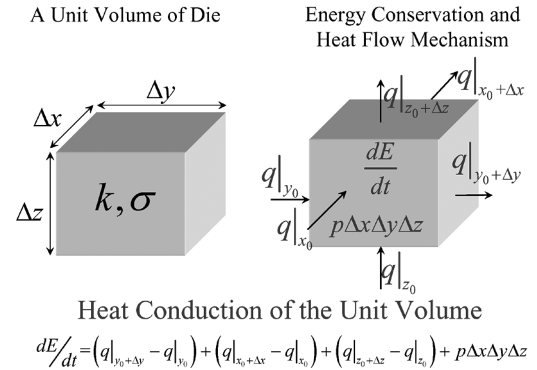

Fig. 2. Energy conservation law and the heat conduction equation. The dE/dt is the energy change rate for the unit volume and is equal to $\sigma\Delta x\Delta y\Delta z\partial T/\partial t$. The conduction heat flowing into the unit volume is equal to the sum of $q|_{x_0} = -\kappa\Delta y\Delta z\partial T/\partial x|_{x_0}$, $q|_{y_0} = -\kappa\Delta x\Delta z\partial T/\partial y|_{y_0}$ and $q|_{z_0} = -\kappa\Delta x\Delta y\partial T/\partial z|_{z_0}$. The conduction heat flowing outward the unit volume is the sum of $q|_{x_0+\Delta x} = -\kappa\Delta y\Delta z\partial T/\partial x|_{x_0+\Delta x}$, $q|_{y_0+\Delta y} = -\kappa\Delta x\Delta z\partial T/\partial y|_{y_0+\Delta y}$, and $q|_{z_0+\Delta z} = -\kappa\Delta x\Delta y\partial T/\partial z|_{z_0+\Delta z}$. The $p\Delta x\Delta y\Delta z$ is the energy generation rate of that unit volume. The κ and σ are the thermal conductivity, and the product of the material density and the heat in the unit volume, respectively. The p is the power consumption density of the unit volume.

According to energy conservation law, the changing rate of energy in a unit volume of substrate equals to the conduction heat through the unit volume [17]. Fig. 2 illustrates this heat conduction mechanism. Based on this heat conduction mechanism, the temperature $T_d(\mathbf{r}, t)$ of die can be governed by the following heat transfer equations [2], [4], [5], [7]:

$$\sigma(T_d)\frac{\partial T_d(\mathbf{r},t)}{\partial t} = \nabla \cdot (\kappa(T_d)\nabla T_d(\mathbf{r},t)) + p(\mathbf{r},t); \quad \mathbf{r} \in D \qquad (1)$$

$$\kappa(T_d)\frac{\partial T_d(\mathbf{r},t)}{\partial n_{b_s}} + h_{b_s}T_d(\mathbf{r},t) = f_{b_s}(\mathbf{r}). \qquad (2)$$

Here, $\mathbf{r} = (x, y, z)$, $\kappa(T_d)$ is the thermal conductivity (W/m $\cdot^\circ$ C) of die, $\sigma(T_d)$ is the product of the material density and the specific heat (J/m^3 $\cdot^\circ$ C) of die, $p(\mathbf{r}, t)$ is the power density of heat source (W/m^3), $D = (0, L_x) \times (0, L_y) \times (-L_z, 0)$ is the dimension of die, L_x and L_y are the lateral lengths of die, L_z is the thickness of die, b_s is any specific boundary surface of the die, h_{b_s} is the heat-transfer coefficient on b_s, $f_{b_s}(\mathbf{r})$ is an arbitrary function on b_s, and $\partial/\partial n_{b_s}$ is the differentiation along the outward direction which is normalized to b_s.

To provide reasonable accuracy of the temperature estimation with little computational effort during the early-stage temperature-aware optimization procedures, heat-transfer coefficients on the boundary surfaces of die should be appropriately modeled [1]. Based on the model proposed by [1], the heat transfer coefficients of primary heat flow path can be equalized to an effective heat transfer coefficient h_p by combining the effect of each component on the primary heat flow path. Since the detailed layout of interconnects is not available in early design stages, each interconnect layer is modeled as an equivalent thermal resistance based on the densities and the regularity structure assumption of metal and dielectric material [1]. Furthermore, the I/O pads and PCB can also be modeled as an effective thermal resistance by using the technique proposed by [20], [21]. With theses modeled thermal resistors, the technique shown in [15] can be utilized to find the equivalent heat-transfer coefficient h_s of these successively connected thermal resistors. After h_p and h_s have been obtained, $f_{b_s}(\mathbf{r})$'s for the top and bottom surfaces are set to h_sT_a and $-h_pT_a$, respectively [15], [20]. Here, T_a is the ambient air temperature. Because of the chip and package structures, the area of vertical surface is strictly less than the area of horizontal surface, and the thermal conductivity of air is much less than the thermal conductivities of primary and secondary heat flow paths. Therefore, the boundary condition of each vertical surface can be set to be adiabatic [6].

Generally, the values of $\kappa(T_d)$ and $\sigma(T_d)$ are temperature dependent. The difference of peak temperature is about 5 °C between the result with temperature-dependent thermal parameters and the result with constant thermal parameters at 25 °C [7]. In current VLSI design, the on-die temperture can be in the degree of 100 °C. Under this situation, this difference may lead to about 5% error for the peak temperature of die. Since the effort to amend this error is relatively high,[2] for practical purposes, these thermal parameters are usually treated as appropriate constants while performing temperature-aware floor-planning and placement [5], [11]–[14].

The value of each thermal parameter can be found by applying a simplified 1-D thermal model shown in Fig. 3 to estimate the roughly average rising temperature of the die with respect to the room temperature. After that, the thermal parameters are calculated at the average temperature. Please see Section V-A for the detail of using 1-D thermal model. By using these estimated thermal parameters, the error of peak temperature can be reduced.

With the above models, the heat diffusion equations for the rising temperature, $T(\mathbf{r},t) = T_d(\mathbf{r},t) - T_a$, of die in early design stages can be rewritten as

$$\sigma \frac{\partial T(\mathbf{r},t)}{\partial t} = \kappa \nabla^2 T(\mathbf{r},t) + p(\mathbf{r},t); \mathbf{r} \in D \tag{3}$$

$$\left.\frac{\partial T(\mathbf{r},t)}{\partial x}\right|_{x=0,L_x} = \left.\frac{\partial T(\mathbf{r},t)}{\partial y}\right|_{y=0,L_y} = 0 \tag{4}$$

$$\kappa \left.\frac{\partial T(\mathbf{r},t)}{\partial z}\right|_{z=-L_z} = h_p T(x,y,-L_z,t) \tag{5}$$

[2]To calibrate the difference caused by the temperature-dependent issue of thermal parameters, several iterative loops of the thermal simulation need to be executed.

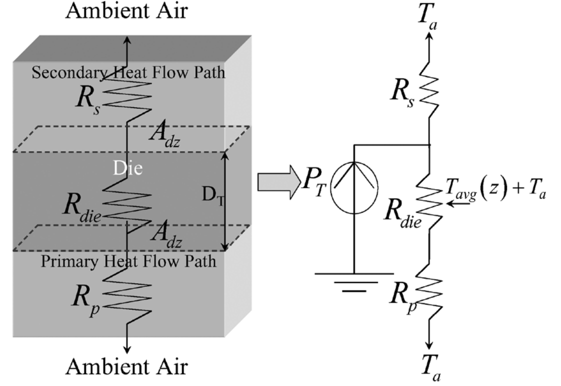

Fig. 3. Simplified 1-D thermal model for estimating the roughly average temperature of die. The modeled thermal resistance network is shown in the right-hand side. The values of thermal resistors are $R_s = 1/A_{dz}h_s$, $R_p = 1/A_{dz}h_p$ and $R_{\text{die}} = D_T/\kappa A_{dz}$. $T_{\text{avg}}(z)$ is the average rising temperature with respect to the room temperature T_a on the lateral planes at arbitrary z position of die. Here, R_{die} can be viewed as a variable resistor when obtaining $T_{\text{avg}}(z)$ at certain z position. P_T is the total average power consummation of die. A_{dz} is the cross area of die normal to the z-direction, and D_T is the thickness of die.

$$\kappa \left.\frac{\partial T(\mathbf{r},t)}{\partial z}\right|_{z=0} = -h_s T(x,y,0,t). \tag{6}$$

Here, κ and σ are the thermal conductivity, and the product of the material density and the specific heat of die got by using the roughly average temperature, respectively, and the initial condition $T(\mathbf{r},0) = 0$.

By discretizing the power generating source of die along the x- and y-directions into MN grid cells, as shown in Fig. 1, where M and N are numbers of divisions in x- and y-directions, respectively, the power density profile $p(\mathbf{r},t)$ in (3) can be written as

$$p(\mathbf{r},t) = \begin{cases} \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} p_{mn}(t)\Pi_{mn}(x,y), & \mathbf{r} \in D_p; \\ 0, & \mathbf{r} \in D \setminus D_p. \end{cases} \tag{7}$$

Here, $D_p = (0,L_x)\times(0,L_y)\times(-j_d,0)$, j_d is the junction depth of device, $\Pi_{mn}(x,y)$ is an indicative function with nonzero value being 1 only when (x,y) is in $[m\Delta x,(m+1)\Delta x] \times [n\Delta y,(n+1)\Delta y]$, $\Delta x = L_x/M$, $\Delta y = L_y/N$, m and n are indices of divisions, and $p_{mn}(t)$ is the power density waveform of grid cell (m,n) in the thin layer with thickness j_d.

For the dynamic thermal simulation, $p_{mn}(t)$ is a user-specified time-interval function with the magnitude of each interval being equal to the average power density of each time interval. We should note that the thermal time constant of heat conduction is much larger than the clock period of circuit [2], [16]. As indicated in [16], the temperature takes at least 100 K cycles to rise 0.1 °C. Practically, the time interval specified by the user can be much larger than the clock period of circuit. Moreover, when calculating the steady state temperature, the input power profile is usually set to the steady power profile (the average power profile for a very long time period estimation) [1], [2], [4], [6], [7], therefore, $p_{mn}(t)$ can be reasonably viewed as a step function with the magnitude being equal to its average power density for a long time period.

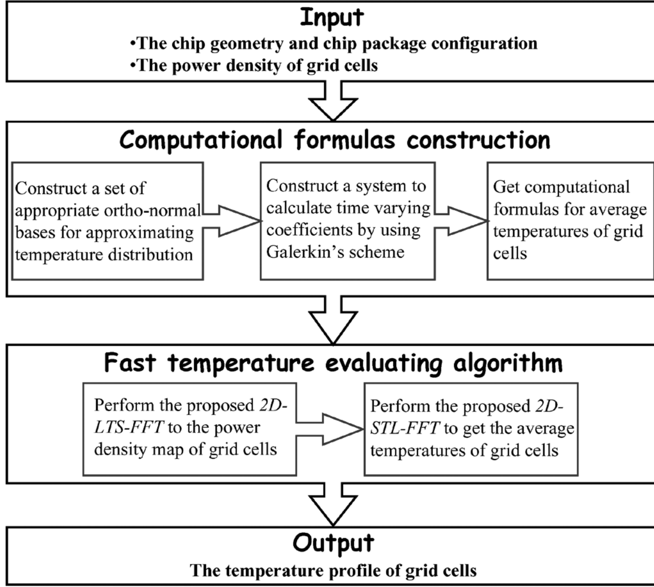

Fig. 4. Executing flow of the proposed GIT-based thermal simulator.

With the previous discussion and governing (3)–(6), our goal is to get the rising temperature distribution of the die corresponding to the ambient temperature.

III. Full-Chip Thermal Simulation

The executing flow of our GIT-based thermal simulator is summarized in Fig. 4. After the chip geometry, package configuration and power density of grid cells are given, the compact thermal model described in Section II is built.

Then, the GIT-based computational formulas for the full-chip temperature distribution are derived. As shown in the first major block (Computational formulas construction) of Fig. 4, three steps are involved to construct the formulas. In the beginning, a set of appropriate bases is generated by a system-compatible auxiliary problem.[3] After that, the temperature distribution can be expressed by these bases with suitable time-varying coefficients. With the Galerkin's scheme [17], [19], those time-varying coefficients can be found by an uncoupled system for estimating the temperature in the sense of least square residual approximation. Finally, the calculating formula of the average temperature for each specific grid cell is obtained by averaging the temperatures in that grid area.

After the temperature computational formulas are derived, we develop two efficient FFT like evaluating algorithms, *2D-LTS-FFT* and *2D-STL-FFT*, as shown in the second major block (fast temperature evaluating algorithm) of Fig. 4, to get the transformed coefficients for the power density map of grid cells and the desired temperature distribution, respectively.

In reality, the leakage power of chip is temperature dependent. Although our simulating flow does not include this issue, it can be easily handled by combining the temperature-power iterative framework [1], [7], [10] with the proposed algorithm, and the detail is presented in Appendix III.

In the rest of this section, each sub-block of two major blocks in Fig. 4, the error bound decaying rates of [6] and our GIT-based formula for the average steady-state temperature distribution, and the dynamic thermal simulation are discussed.

A. Auxiliary Problem for Generating Appropriate Spatial Bases

The auxiliary problem can be introduced by considering the homogeneous problem which the temperature distribution satisfies (3)–(6) with $p(\mathbf{r},t)=0$. As stated in [17]–[19], the auxiliary problem can be set to be the following Sturm–Liouville problem with specific boundary conditions:

$$\nabla^2\phi_{ilq}(\mathbf{r})+\lambda_{ilq}^2\phi_{ilq}(\mathbf{r})=0;\mathbf{r}=(x,y,z)\in D \tag{8}$$

$$\left.\frac{\partial\phi_{ilq}(\mathbf{r})}{\partial x}\right|_{x=0,L_x}=\left.\frac{\partial\phi_{ilq}(\mathbf{r})}{\partial y}\right|_{y=0,L_y}=0 \tag{9}$$

$$\kappa\left.\frac{\partial\phi_{ilq}(\mathbf{r})}{\partial z}\right|_{z=-L_z}=h_p\phi_{ilq}(x,y,-L_z) \tag{10}$$

$$\kappa\left.\frac{\partial\phi_{ilq}(\mathbf{r})}{\partial z}\right|_{z=0}=-h_s\phi_{ilq}(x,y,0). \tag{11}$$

The solutions of Sturm–Liouville problem form a set of completely ortho-normal spatial bases for the die, and the general forms of $\phi_{ilq}(\mathbf{r})$ and λ_{ilq}^2 can be obtained as follows [17]:

$$\phi_{ilq}(\mathbf{r})=\frac{\cos\left(\frac{i\pi x}{L_x}\right)\cos\left(\frac{l\pi y}{L_y}\right)\phi_q(x)}{\sqrt{N_{ilq}}} \tag{12}$$

$$\lambda_{ilq}^2=\lambda_{x_i}^2+\lambda_{y_l}^2+\lambda_{z_q}^2 \tag{13}$$

where i, l and q are non-negative integers, N_{ilq} is the normalized value being equal to $\theta_{il}L_xL_yN_{z_q}$, $\theta_{00}=1/2$, $\theta_{i0}=\theta_{0l}=1/4$, $\theta_{il}=1/8$ with $i\neq 0$ and $l\neq 0$, $\lambda_{x_i}^2=(i\pi/L_x)^2$, $\lambda_{y_l}^2=(l\pi/L_y)^2$

$$N_{z_q}=\frac{\left(h_p^2+\kappa^2\lambda_{z_q}^2\right)\left(\frac{\kappa h_s}{h_s^2+\kappa^2\lambda {z_q}^2}+L_z\right)+\kappa h_p}{\lambda_{z_q}^2} \tag{14}$$

and

$$\phi_q(x)=\kappa\cos\left(\lambda_{z_q}(z+L_z)\right)+\frac{h_p}{\lambda_{z_q}}\sin\left(\lambda_{z_q}(z+L_z)\right). \tag{15}$$

Here, each λ_{z_q} is a positive value which satisfies

$$\frac{\kappa^2\lambda_{z_q}^2-h_ph_s}{\kappa\lambda_{z_q}(h_p+h_s)}=\cot\left(\lambda_{z_q}L_z\right). \tag{16}$$

To obtain each λ_{z_q}, we apply Newton–Raphson method [22] to (16) with the initial guess of each q being $\pi q/L_z+0.2\pi/L_z$ because the period of the right hand side in (16) is equal to π/L_z.

Each $\phi_{ilq}(\mathbf{r})$ is called as an eigenfunction, λ_{ilq}^2 is its eigenvalue, and $\lambda_{x_i}^2$, $\lambda_{y_l}^2$, and $\lambda_{z_q}^2$ are eigenvalues in x-, y-, and z-directions, respectively. The physical meaning of $\phi_{ilq}(\mathbf{r})$ is that it presents the ilq^{th} free vibration with respect to the system described by (8)–(16), and its vibration frequencies are λ_{x_i}, λ_{y_l} and λ_{z_q} in x-, y-, and z-directions, respectively. The physical

[3]Several guidelines [17]–[19] need to be followed for choosing this auxiliary problem. First, the auxiliary problem should be as similar as possible to the original problem. Second, the generated bases have to be completely ortho-normalized to ensure the convergence in mean of the approximated temperature distribution. Finally, the ortho-normal bases should be time independent for the efficiency consideration.

meaning of λ_{ilq}^2 is that it presents the spectral magnitude of $\phi_{ilq}(\mathbf{r})$.

B. System Transformation for Time-Varying Coefficients

Since the generated bases $\{\phi_{ilq}(\mathbf{r})\}$ are completely orthonormal in the spatial domain of die, $T(\mathbf{r},t)$ can be approximated as the following finite integral transform pair [17]–[19]:

$$T(\mathbf{r},t) \approx \widehat{T}(\mathbf{r},t) = \sum_{q=0}^{N_z-1}\sum_{l=0}^{N_y-1}\sum_{i=0}^{N_x-1} \psi_{ilq}(t)\phi_{ilq}(\mathbf{r}) \quad (17)$$

$$\psi_{ilq}(t) = \int_{-L_z}^{0}\int_{0}^{L_y}\int_{0}^{L_x} T(\mathbf{r},t)\phi_{ilq}(\mathbf{r})dxdydz \quad (18)$$

where each $\psi_{ilq}(t)$ is an unknown transformed time-varying coefficient, and N_x, N_y and N_z are truncation points in x-, y-, and z-directions, respectively.

After utilizing the energy conservation law and Divergence Theorem [18], and executing a series of derivations, the following uncoupled system is established to find each time-varying coefficient function $\psi_{ilq}(t)$. The detail description is shown in Appendix I

$$\begin{cases} \sigma\psi'_{ilq}(t) = -\kappa\lambda_{ilq}^2\psi_{ilq}(t) + \widehat{p}_{ilq}(t) \\ \psi_{ilq}(0) = 0 \end{cases}$$
$$0 \le i \le N_x - 1, 0 \le l \le N_y - 1, 0 \le q \le N_z - 1 \quad (19)$$

where

$$\widehat{p}_{ilq}(t) = \int_{-j_d}^{0}\int_{0}^{L_y}\int_{0}^{L_x} p(\mathbf{r},t)\phi_{ilq}(\mathbf{r})dxdydz. \quad (20)$$

Since (19) is uncoupled for different "ilq", each $\psi_{ilq}(t)$ can be individually solved as

$$\psi_{ilq}(t) = \frac{1}{\sigma}\int_{0}^{t} \widehat{p}_{ilq}(\tau)e^{-\frac{k}{\sigma}\lambda_{ilq}^2(t-\tau)}d\tau. \quad (21)$$

For the steady-state simulation, $p_{mn}(t)$ is a step function with its magnitude being equal to the average power density of grid (m,n) for a long time period. Thus, t is set to be infinity to find the steady-state value of $\psi_{ilq}(\infty)$ which is $\widehat{p}_{ilq}(\infty)/(k\lambda_{ilq}^2)$. Therefore, the evaluation of steady-state temperature can be done without any time step approaching.

C. Average Rising Temperature Evaluation of Grid Cells

Generally speaking, hot spots occur in regions which are close to power sources. Hence, we focus on evaluating the average temperature of each grid cell on the top surface $(z=0)$ of die.[4] First, we present the formulation for calculating the average rising temperature of steady state and discuss its decaying rate of truncation error. Then, the fast evaluating algorithms are developed for realizing the formulation. Finally, the dynamic thermal simulation is given.

[4]Our method can be used to find the average temperature of each grid cell at arbitrary lateral plane of the die by substituting suitable z into the bases.

1) Steady-State Formulation: Plugging $\phi_{ilq}(\mathbf{r})$'s and $\psi_{ilq}(\infty)$'s into (17), the average rising temperature $\overline{T}_{mn}$ of steady state for each grid cell (m,n) on the top surface is

$$\overline{T}_{mn} = \frac{1}{\Delta x\Delta y}\int_{n\Delta y}^{(n+1)\Delta y}\int_{m\Delta x}^{(m+1)\Delta x} \widehat{T}(x,y,0,\infty)dxdy$$
$$= \sum_{l=0}^{N_y-1}\sum_{i=0}^{N_x-1} K_{il}\cos\left(\frac{i\pi(2m+1)}{2M}\right)\cos\left(\frac{l\pi(2n+1)}{2N}\right) \quad (22)$$

where

$$K_{il} = \frac{\widehat{P}_{il}}{\kappa}\sum_{q=0}^{N_z-1}\frac{\Gamma_q C_{ilq}}{N_{ilq}}\phi_q(0) \quad (23)$$

$$C_{ilq} = \begin{cases} \frac{\Delta x\Delta y}{\lambda_{ilq}^2}; & i=0, l=0 \\ \frac{4NL_y\Delta x\sin^2\left(\frac{l\pi}{2N}\right)}{l^2\pi^2\lambda_{ilq}^2}; & i=0, l\neq 0 \\ \frac{4ML_x\Delta y\sin^2\left(\frac{i\pi}{2M}\right)}{i^2\pi^2\lambda_{ilq}^2}; & i\neq 0, l=0 \\ \frac{16MNL_xL_y\sin^2\left(\frac{i\pi}{2M}\right)\sin^2\left(\frac{l\pi}{2N}\right)}{i^2l^2\pi^4\lambda_{ilq}^2}; & i\neq 0, l\neq 0 \end{cases} \quad (24)$$

and

$$\widehat{P}_{il} = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} p_{mn}\cos\left(\frac{i\pi(2m+1)}{2M}\right)\cos\left(\frac{l\pi(2n+1)}{2N}\right) \quad (25)$$

$$\Gamma_q = \frac{2\kappa}{\lambda_{z_q}}\cos\left(\lambda_{z_q}(L_z - j_d/2)\right)\sin\left(\lambda_{z_q}j_d/2\right) - \frac{2h_p}{\lambda_{z_q}^2}\sin\left(\lambda_{z_q}(L_z - j_d/2)\right)\sin\left(\lambda_{z_q}j_d/2\right) \quad (26)$$

where p_{mn} is the average power density of grid (m,n) for a long time period, and M and N are numbers of divisions in the x- and y-directions.

An error bound of $\overline{T}_{mn}$ calculated by (22) is given by *Theorem 1* in Appendix II. Based on *Theorem 1*, the error decaying rate of (22) is dominated by $i^2l^2\lambda_{z_q}((i\pi/L_x)^2 + (l\pi/L_y)^2 + \lambda_{z_q}^2)$. To compare the previous error decaying rate with the Green's function based method's [6], we set the boundary conditions and power source location to be the same with [6]. With these settings and Appendix II, the error decaying rate of our GIT-based formulation is in the order of $i^2l^2((i\pi/L_x)^2 + (l\pi/L_y)^2 + \lambda_{z_q}^2))$, and the error decaying rate of [6] is in the order of $i^2l^2\sqrt{(i\pi/L_x)^2 + (l\pi/L_y)^2}$. Therefore, the error decaying rate of the proposed GIT-based method is faster than [6]. The reason is that the bases in z-direction of the GIT-based method are different with [6], and our constructed bases can fully fill the eigen-space of heat diffusion equation. This fact leads to different coefficients in the approximating form even if the bases in x- and y-directions of our GIT-based method are the same with [6]. Furthermore, the error decaying rate of the proposed GIT-based method is not only faster than [6], the experimental results also show that it can maintain the same accuracy as [6] even if its truncation points, N_x and N_y are far less than the numbers of divisions, M and N.

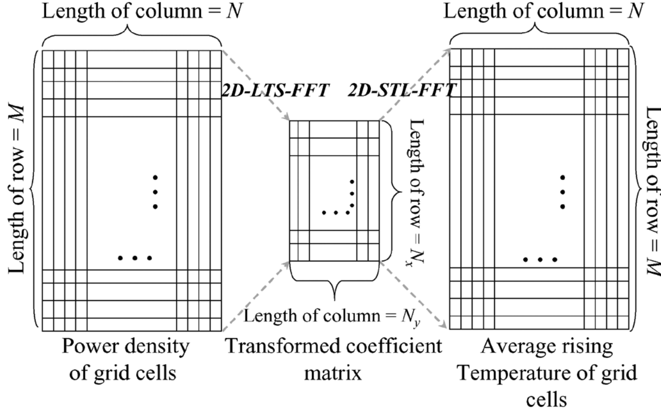

Fig. 5. Overview of using *2D-SLT-FFT* and *2D-LTS-FFT* to evaluate the average rising temperature of grid cells.

Algorithm Radix-two *1D-STL-FFT*
Input: *Complex vector* f *with length* $\widetilde{M}$
Output: *Complex vector* $\overline{F}$ *with length* $2M$

1	**Begin**
2	$f_{\mathbf{R}}$ = **Reverse-bit**(f) ;
3	$L = 4M/\widetilde{M}$;
4	$N_{SubDFTs} = \widetilde{M}/2$;
5	**For** $SubIndex$ = 0 to $N_{SubDFTs} - 1$
6	$k = L \times SubIndex$;
7	$i = 2 \times SubIndex$;
8	**For** $SubK$ = 0 to $L - 1$
9	$\overline{F}[k] = f_{\mathbf{R}}[i] + f_{\mathbf{R}}[i+1] \times e^{j2\pi \times SubK/L}$;
10	$k = k + 1$;
11	**EndFor**
12	**EndFor**
13	Apply the bottom up procedure of standard FFT to execute the Danielson-Lanczos Lemma $\log_2 \widetilde{M} - 1$ times for evaluating $\overline{F}$
14	**End**

Fig. 6. Procedure of *1D-STL-FFT*. The "$\mathbf{Reverse-bit}$" means the reverse-bit algorithm [22].

Although the truncation points N_xN_y can be far less than the number of grid cells MN, there is no actual efficiency improvement over [6] if we directly apply the standard FFT to evaluate each $\overline{T}_{mn}$. The reason is that the standard inverse fast Fourier transform (IFFT) needs to pad zeros to the input data when the dimension of input data is less than the dimension of output data, such as (22). Moreover, the dimension of output data in standard FFT is equal to the dimension of input data. However, the dimension of output data in (25) is only N_xN_y which is far less than its dimension of input data, MN. To overcome this limitation, we develop FFT like fast evaluating algorithms for our GIT formulation in the next subsection.

2) Fast Evaluating Algorithms for GIT Formulation: To efficiently realize our formulation for the steady-state temperature distribution, we first derive a one-dimensional radix-two-based FFT like algorithm for the length of output data being larger than the length of input data *1D-STL-FFT*. Then, based on *1D-STL-FFT*, we develop a 1-D FFT like algorithm for the length of output data being smaller than the length of input data *1D-LTS-FFT*. Finally, we extend these 1-D algorithms to two 2-D algorithms by the row-column procedure, and we call them as *2D-STL-FFT* and *2D-LTS-FFT*. Finally, these two algorithms are integrated to calculate (22) and (25). The computational complexity of our GIT-based thermal simulator can be analyzed to be only $O(MN \log_2 N_xN_y)$. The overview of the previous evaluating algorithms are shown in Fig. 5. Given the power density profile of chip, *2D-STL-FFT* computes the transformed coefficients of power density profile, and *2D-LTS-FFT* transforms these transformed coefficients to obtain the average rising temperature of grid cells.

a) 1D-STL-FFT: The prototype of *1D-STL-FFT* is

$$\overline{F}_k = \sum_{i=0}^{\widetilde{M}-1} f_i e^{j2\pi ik/2M}; \quad k = 0, \ldots, 2M-1 \tag{27}$$

where $\widetilde{M} < M$ and both are power of 2, $j = \sqrt{-1}$, and f_i's and $\overline{F}_k$'s are complex input and output data with lengths being equal to $\widetilde{M}$ and M, respectively.

Because the length of $\overline{F}_k$'s is larger than the length of f_i's, the zeros-padding step of f_i's like in the standard FFT algorithm needs to be avoided for saving the runtime. Therefore, the *1D-STL-FFT* algorithm shown in Fig. 6 is developed to calculate (27) without the zeros-padding.

In the beginning, the "$\mathbf{Reverse-bit}(f)$" reorders the input data for those sub discrete Fourier transforms (DFTs) which will be generated by recursively performing the Danielson–Lanczos Lemma (DL-Lemma) [22] to the prototype of *1D-STL-FFT* in (27). The DL-Lemma is used to rewrite the original DFT as the sum of two sub DFTs with half output length. One of the two is formed from the even-numbered points of the input data, and the other is formed from the odd-numbered points. In this step, the DL-Lemma is used recursively for these two sub DFTs. Because $\widetilde{M}$ is less than M, this bisecting procedure is executed only $\log_2 \widetilde{M}$ times, and we have $\log_2 \widetilde{M}$ bisecting levels.

After *Line* 2 in Fig. 6 is performed, the *1D-STL-FFT* algorithm evaluates the output of those L sub DFTs in the bottom level by using *Lines* 3 $\sim$ 12, and performs *Line* 13 to get the output of remaining levels.

An example with $M = 16$ and $\widetilde{M} = 8$ is given in Fig. 7(a). There are three bisecting levels, and four sub DFTs in the bottom level. After performing the reverse-bit algorithm to the input data, two phases are executed. The first phase is done by using *Lines* 3 $\sim$ 12 of Fig. 6. The second phase is to get the output of the remaining levels by executing the bottom up procedure of standard FFT as stated in *Line* 13 of Fig. 6.

The complexity of *1D-STL-FFT* is $O(M \log_2 \widetilde{M})$ since there are $\log_2 \widetilde{M}$ bisecting levels and each complexity is $O(M)$.

b) 1D-LTS-FFT: The prototype of *1D-LTS-FFT* is

$$\widehat{F}_i = \sum_{m=0}^{M-1} \widehat{f}_m e^{j2\pi im/2M}; \quad i = 0, \cdots, \widetilde{M}-1 \tag{28}$$

where $\widetilde{M} < M$, and $\widehat{f}_m$ and $\widehat{F}_i$ are real input and complex output data with lengths being equal to M and $\widetilde{M}$, respectively.

Applying the DL-Lemma to the prototype of *1D-LTS-FFT* for generating $\log_2(M/\widetilde{M}) + 1$ bisecting levels, $\widehat{F}_i$ can be written as the sum of $2M/\widetilde{M}$ sub DFTs. Each sub DFT has the same form as the *1D-STL-FFT* with the lengths of input and output being equal to $\widetilde{M}/2$ and $\widetilde{M}$, respectively.

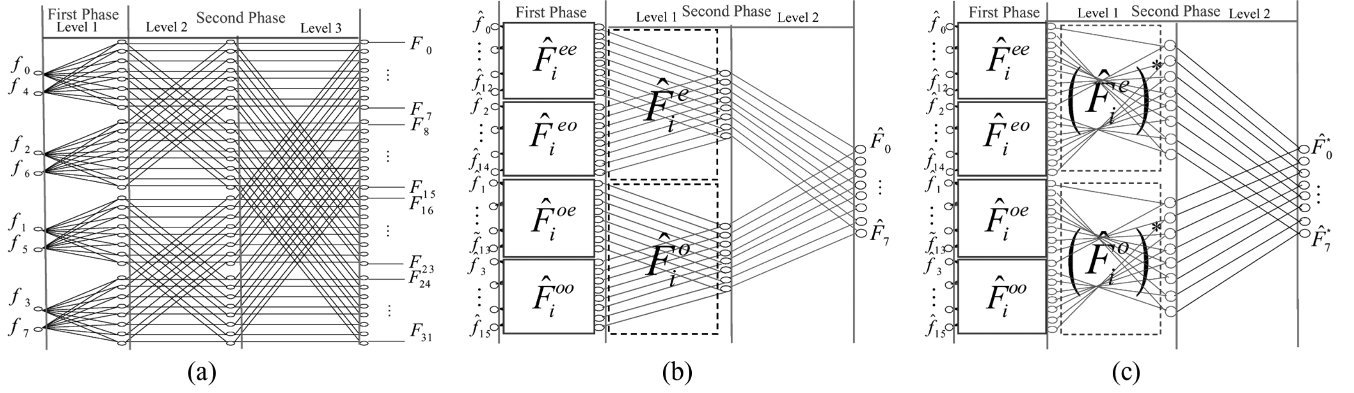

Fig. 7. Computational flow graphs of *1D-STL-FFT* and *1D-LTS-FFT* with $\widetilde{M} = 8$ and $M = 16$. (a) *1D-STL-FFT*. (b) *1D-LTS-FFT*. (c) *1D-LTS-FFT* for negative frequencies.

Algorithm Radix-two *1D-LTS-FFT*
Input: *Real vector* $\widehat{f}$ *with length* M
Output: *Complex vector* $\widehat{F}$ *with length* $\widetilde{M}$
1 **Begin**
2 $\widehat{f}_{\mathbf{R}}$ = **Reverse-bit**$(\widehat{f})$;
3 $N_{SubDFTs} = 2M/\widetilde{M}$;
4 **For** Sub_i = 0 to $N_{SubDFTs} - 1$
5 $Start = Sub_i \times \widetilde{M}$;
6 $End = Start + \widetilde{M}$;
7 $F_t\,(Start : End - 1)$ = *1D-LTS-FFT*$\Big(\widehat{f}_{\mathbf{R}}(\frac{Start}{2} : \frac{End}{2} - 1)\Big)$;
8 **EndFor**
9 $L = \widetilde{M}$;
10 **For** $level$ = 0 to $log_2(M/\widetilde{M})$
11 $n = 0$;
12 $Sub_i = 0$;
13 $N_{SubDFTs} = N_{SubDFTs}/2$;
14 **While** $Sub_i < N_{SubDFTs}$
15 **For** $i = 0$ to $\widetilde{M} - 1$;
16 $i^* = i + Sub_i \times \widetilde{M}$;
17 $F_t[i+n] = F_t[i^*] + F_t[i^* + \widetilde{M}] \times e^{j2\pi i/L}$;
18 **EndFor**
19 $Sub_i = Sub_i + 2$;
20 $n = n + \widetilde{M}$;
21 **EndWhile**
22 $L = 2 \times L$;
23 **EndFor**
24 $\widehat{F} = F_t\left(0 : \widetilde{M} - 1\right)$;
25 **End**

Fig. 8. Procedure of *1D-LTS-FFT*.

Two phases are utilized to evaluate $\widehat{F}_i$, and the *1D-LTS-FFT* algorithm is shown in Fig. 8. First, *Line* 2 performs the reverse-bit algorithm to the input data, and *Lines* 4 $\sim$ 8 use the *1D-STL-FFT* algorithm to obtain each bisected sub DFT. After each sub DFT has been done, a bottom up procedure is applied to the remaining $\log_2(M/\widetilde{M}) + 1$ bisecting levels for finding $\widehat{F}_i$, and the executing steps are from *Line* 9 to *Line* 24.

An example with $M = 16$ and $\widetilde{M} = 8$ is shown in Fig. 7(b). In the first phase, the input data are reordered by using the reverse-bit algorithm, and the reordered data are fed into the corresponding *1D-STL-FFT* blocks. This can be done by using *Lines* 3 $\sim$ 8 in Fig. 8. Then, the output of top block in the level 1 of the second phase is calculated by

$$\widehat{F}_i^e = \widehat{F}_i^{ee} + e^{j2\pi i/16}\widehat{F}_i^{eo} \tag{29}$$

Algorithm Radix-two *2D-STL-FFT*
Input: *Complex matrix* $\overline{K}$ *with length* $N_x \times N_y$
Output: *Complex matrix* $\overline{F}$ *with length* $2M \times 2N$
1 **Begin**
2 **For** i = 0 to $N_x - 1$
3 $T_{Row}(i, 0 : 2N-1)$ = *1D-STL-FFT*$\big(\overline{K}(i, 0 : N_y - 1)\big)$;
4 **EndFor**
5 **For** j = 0 to $2N - 1$
6 $\overline{F}(0 : 2M-1, j)$ = *1D-STL-FFT*$(T_{Row}(0 : N_x - 1, j))$;
7 **EndFor**
8 **End**

Fig. 9. Procedure of *2D-STL-FFT*.

and $\widehat{F}_i^o$ can be done by a similar way. Finally, $\widehat{F}_i$ is equal to

$$\widehat{F}_i = \widehat{F}_i^e + e^{j2\pi i/32}\widehat{F}_i^o. \tag{30}$$

The second phase is summarized in *Lines* 9 $\sim$ 24 of Fig. 8.

For the general case, the sub DFTs in each level of the second phase can be obtained by combining those sub DFTs of their previous level with the similar formula of (29) by replacing 16 to be $2^1\widetilde{M}, 2^2\widetilde{M}, \cdots, 2M$ in each level. The computational complexity of the first phase is $O(M \log_2 \widetilde{M})$ because the *1D-STL-FFT* needs to be executed $2M/\widetilde{M}$ times, and each complexity is $O(\widetilde{M} \log_2 \widetilde{M})$. The complexity is $O(M)$ for the second phase. Hence, the computational complexity of *1D-LTS-FFT* is $O(M \log_2 \widetilde{M})$.

c) Temperature Evaluation: The average rising temperature of steady state shown in (22) can be got as

$$\overline{T}_{mn} = \frac{1}{2}R_e\left\{\overline{F}_{m,n} + \overline{F}_{2M-(m+1),n}\right\} \tag{31}$$

where $R_e\{\cdot\}$ is the real part operator, and

$$\overline{F}_{k_1,k_2} = \sum_{i=0}^{N_x-1}\sum_{l=0}^{N_y-1} \overline{K}_{il} e^{\frac{j2\pi i k_1}{2M}} e^{\frac{j2\pi l k_2}{2N}}. \tag{32}$$

Here, $0 \le k_1 \le 2M - 1$, $0 \le k_2 \le 2N - 1$, $\overline{K}_{il} = K_{il}e^{j2\pi i/4M}e^{j2\pi l/4N}$, and each K_{il} is equal to (23).

To obtain $\overline{T}_{mn}$'s, the values of $\overline{F}_{k_1,k_2}$'s and K_{il}'s need to be known.

To calculate $\overline{F}_{k_1,k_2}$'s, a row-column-based *2D-STL-FFT* method is developed and shown in Fig. 9 by utilizing the *1D-STL-FFT* algorithm. *Lines* 2 $\sim$ 4 perform the *1D-STL-FFT*

Input: Geometries of die and package, and related thermal parameters. The steady power density of grid cells.
Output: The average steady-state rising temperature $\overline{T}_{mn}$ for each grid cell (m, n).

Pre-calculating stage

1. Set thermal parameters of die by using the roughly average temperature obtained by the simplified 1-D model described in section II.
2. Obtain the eigenfunctions and eigenvalues described in section III-A.
3. Obtain C_{ilq} and the summation term for each q in equations (24) and (23), respectively.

Post-calculating stage

1. Obtain $\widehat{P}_{il}$ by *2D-LTS-FFT* discribed in section III-C.2, and K_{il} in equation (23).
2. Obtain $\overline{K}_{il}$ in equation (32) and feed it into *2D-STL-FFT* described in section III-C.2. Then, apply equation (31) to obtain $\overline{T}_{mn}$.

Fig. 10. Simulating algorithm of the proposed steady-state thermal simulator.

for each row of the input matrix $\overline{K}$ which each (i, l) entry is $\overline{K}_{il}$, and *Lines* 5 $\sim$ 7 apply the *1D-STL-FFT* to each column of the output matrix got from the row procedure for obtaining the desired matrix $\overline{F}$ which each (k_1, k_2) entry is $\overline{F}_{k_1,k_2}$. The complexity for obtaining $\overline{F}_{k_1,k_2}$'s is $O(MN \log_2 N_x N_y)$ because the complexities of row and column procedures are $O(N_x N \log_2 N_y)$ and $O(NM \log_2 N_x)$, respectively.

To calculate each K_{il} from (23), $\widehat{P}_{il}$'s need to be known from (25). Therefore, the 2-D prototype with the similar form as (28) is needed to get related $\widehat{F}_{i,l}$'s for the input data being p_{mn}'s. A row-column-based *2D-LTS-FFT* algorithm can be constructed by using the similar procedure shown in Fig. 9 with the *1D-STL-FFT* replaced by the *1D-LTS-FFT*. The *2D-LTS-FFT* method is then used to get those related $\widehat{F}_{i,l}$'s. However, (31) cannot be utilized to calculate $\widehat{P}_{il}$'s because the lengths of those related $\widehat{F}_{i,l}$'s in the row and column directions are less than $2M$ and $2N$, respectively. Therefore, the complex conjugates of $\widehat{F}_{i,l}$'s are required to complete the calculation of $\widehat{P}_{il}$'s.

Fortunately, the complex conjugate of the output from each sub *1D-STL-FFT* in calculating $\widehat{F}_{i,l}$'s can be directly obtained by reversing these sub DFTs. Therefore, the complex conjugates of $\widehat{F}_{i,l}$'s can be got by reversing the data of F_t in *Line* 7 of Fig. 8, and performing *Lines* 9 $\sim$ 24 in Fig. 8 during the row-column procedure of $\widehat{F}_{i,l}$'s.

The complexity of row procedure for obtaining those related $\widehat{F}_{i,l}$'s is $O(MN \log_2 N_y)$ because the *1D-LTS-FFT* needs to be executed $2M$ times. The complexity of column procedure is $O(N_y M \log_2 N_x)$ because the *1D-LTS-FFT* needs to be executed N_y times. Hence, the complexity for obtaining $\widehat{F}_{i,l}$'s is $O(MN \log_2 N_x N_y)$. The complexity for calculating the complex conjugates of $\widehat{F}_{i,l}$'s is $O(MN) + O(N_y M)$ since only the second phase needs to be recomputed. Therefore, the complexity for computing (25) is $O(MN \log_2 N_x N_y)$.

From the previous discussion, we conclude that the complexity of our GIT based thermal simulator is $O(MN \log_2 N_x N_y)$. Finally, the completely proposed simulating algorithm is illustrated in Fig. 10.

3) Dynamic Thermal Simulation: While performing the dynamic thermal simulation, each $p_{mn}(t)$ can be modeled as a user-specified time interval function with the magnitude in each interval being equal to the average power in each time interval. By using (21), each time-varying coefficient $\psi_{ilq}^t \equiv \psi_{ilq}(t)$ is

$$\psi_{ilq}^t = \psi_{ilq}^{t-\Delta t} + \frac{\widehat{p}_{ilq}^t}{\kappa \lambda_{ilq}^2} \left(1 - e^{-\frac{k}{\sigma}\lambda_{ilq}^2 \Delta t}\right) \quad (33)$$

where Δt is the time step and is equal to the time interval of power density waveforms, $\widehat{p}_{ilq}^t$ is equal to (20) with $p(\mathbf{r}, t)$ being equal to the average power density profile in the time interval $(t - \Delta t, t)$, and $\psi_{ilq}^{t-\Delta t} = \psi_{ilq}(t - \Delta t)$.

After ψ_{ilq}^t's are calculated, the average temperature of each grid cell at the sampling time t can be obtained by (17) with the same evaluating method presented in Section III-C2. In addition, applying (33) to compute each ψ_{ilq}^t would not induce any unstable issue with a large Δt because (33) is the exact solution of the system (19), i.e., without the error caused from finite difference approximations such as the backward-Euler method, the trapezoidal method and the Runge–Kutta method. Furthermore, since the thermal time constant of heat conduction is much larger than the clock period of circuit [2], [16], the time step Δt can be far larger than the clock period of the circuit to save runtime with acceptable errors.

IV. Thermal Simulation for 3-D ICs and Package Structures

3-D ICs provide several advantages over 2-D ICs [8]. They provide the flexibility in system design, placement and routing, the suitability for circuits operating on different supply voltages and the capability of on-chip memory design. However, due to the high power density and the ill capability of heat dissipation, the thermal issue is one major concern for 3-D ICs.

Recently, the tradeoff between the circuit performance and the thermal issue of early-stage 3-D ICs design has been studied by estimating the uniform average temperature of each layer [9], [10]. To take into the thermal issue account for early-stage 3-D ICs design, Cong *et al.* [11], [12] applied the 1-D thermal model to predict the temperature cost for their floorplanning and placement algorithms, Goplen *et al.* [13] applied the FEM-based thermal simulator for their force-directed approach based standard cell placement method, and Balakrishnan *et al.* [14] utilized the state-of-the-art numerical method provided in [2] to obtain the temperature distribution as the cost function for their global placement engine.

The 1-D thermal model can quickly capture the average temperature of the region close to cells for each active layer but it loses the spatial temperature gradient. The numerical method needs to know temperatures of unnecessary sampling points, such as the points being far from the region of devices, because temperatures of sampling points depend on each other. To efficiently obtain the nonuniform temperature distribution without needing the temperatures of unnecessary sampling points, we develop a fast 3-D IC thermal simulator by combining the GIT and numerical schemes.

This hybrid scheme is developed for the structure of 3-D ICs in Fig. 11 by using the effective heat transfer coefficients for the primary and secondary heat flow paths. On the other hand, although different materials in the primary and secondary heat

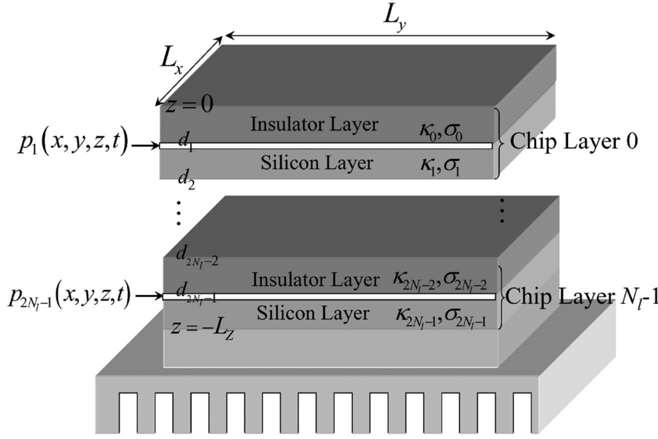

Fig. 11. Schematic diagram of a 3-D IC with N_l chip layers.

flow paths of Fig. 1 are described by effective heat transfer coefficients for the fast temperature estimation, those materials should be modeled as an inhomogeneous structure for the further accuracy consideration. Since its structure is similar to the multilayer structure of 3-D ICs, its inhomogeneity can also be handled by the proposed 3-D IC thermal simulator.

As shown in Fig. 11, the structure of 3-D ICs is a multilayer structure with stacking silicon and insulator layers one by one [8]–[10]. The power sources are distributed in a thin layer close to the top surface of each active silicon layer in the z-direction, and each insulator layer consists of Cu, ILD and glue materials. The heat transfer equations of 3-D ICs can be built by combining the governing equations of each layer with suitable boundary conditions. The heat diffusion equation inside each layer is similar to (3) with their corresponding thermal parameters κ_ζ and σ_ζ. Here, ζ is the layer index. The boundary conditions on the lateral surfaces are flux isolated, and the boundary conditions at $z = -L_z$ and $z = 0$ are convection types with equivalent heat-transfer coefficients h_p and h_s for the primary and secondary heat flow paths, respectively. With the GIT technique, the governing equations of 3-D ICs can be transformed into a 1-D subproblem by utilizing the following ortho-normal spatial bases in the x- and y-directions

$$\phi_{il}(x,y) = \frac{\cos\left(\frac{i\pi x}{L_x}\right)\cos\left(\frac{l\pi y}{L_y}\right)}{\sqrt{N_{il}}} \tag{34}$$

where $N_{il} = \rho_{il}L_xL_y$, $\rho_{00} = 1$, $\rho_{i0} = \rho_{0l} = 1/2$, and $\rho_{il} = 1/4$ with $i \neq 0, l \neq 0$. These ortho-normal spatial bases satisfy the following 2-D Sturm–Liouville problem:

$$\lambda_{il}^2\phi_{il}(x,y) = -\nabla^2\phi_{il}(x,y); \quad (x,y) \in D_{xy} \tag{35}$$

where $D_{xy} = (0, L_x) \times (0, L_y)$ and $\lambda_{il}^2 = \lambda_{x_i}^2 + \lambda_{y_l}^2$. The boundary conditions of (35) are flux isolated and equal to (9) with replacing $\phi_{ilq}(\mathbf{r})$ by $\phi_{il}(x,y)$.

Since $\phi_{il}(x,y)$'s are ortho-normal spatial bases, the approximated rising temperature $\widehat{T}(\mathbf{r},t)$ can be expressed as

$$\widehat{T}(\mathbf{r},t) = \sum_{i=0}^{N_x-1}\sum_{l=0}^{N_y-1}\psi_{il}(z,t)\phi_{il}(x,y) \tag{36}$$

where each $\psi_{il}(z,t)$ is an unknown function, and needs to be found.

Combining the interface conditions, the temperature continuity and the heat flux conservation law on the interface of two different layers, performing Galerkin's scheme along the x- and y-directions, and using (35), each $\psi_{il}(z,t)$ can be got by solving the following 1-D sub-problem:

$$\sigma_\zeta\frac{\partial}{\partial t}\psi_{il}(z,t) = \kappa_\zeta\left(\frac{\partial^2}{\partial^2 z}\psi_{il}(z,t) - \lambda_{il}^2\psi_{il}(z,t)\right) + \widehat{p}_{il,\zeta}(z,t) \tag{37}$$

$$\psi_{il}(z,t)\big|_{z=d_\zeta^+} = \psi_{il}(z,t)\big|_{z=d_\zeta^-} \tag{38}$$

$$\kappa_{\zeta-1}\left.\frac{\partial\psi_{il}(z,t)}{\partial z}\right|_{z=d_\zeta^+} = \kappa_\zeta\left.\frac{\partial\psi_{il}(z,t)}{\partial z}\right|_{z=d_\zeta^-} \tag{39}$$

$$\left.\frac{\partial\psi_{il}(z,t)}{\partial z}\right|_{z=0} = h_s\psi_{il}(0,t) \tag{40}$$

$$\kappa_{2N_l-1}\left.\frac{\partial\psi_{il}(z,t)}{\partial z}\right|_{z=-L_z} = h_p\psi_{il}(-L_z,t) \tag{41}$$

where $\widehat{p}_{il,\zeta}(z,t) = \int_0^{L_y}\int_0^{L_x} p_\zeta(x,y,z,t)\phi_{il}(x,y)dxdy$, ζ is the layer index and $1 \leq \zeta \leq 2N_l-1$, d_ζ is the position of the ζth interface in the z-direction, $p_\zeta(x,y,z,t)$ is the power density in the thin layer of the ζth active silicon substrate and is equal to zero as ζ is even (insulator layer), and each $\psi_{il}(z,0) = 0$.

Though the ortho-normal spatial bases in the z-direction of the above 1-D sub-problem can be analytically solved by the *sign-count* method [17], or (37)–(41) can be directly solved by [23], their computational efforts[5] are relatively high for the practical purpose. Hence, we adopt the numerical scheme to obtain $\psi_{il}(z,t)$ because its runtime is linear in the number of grid points along the z-direction.

By discretizing this 1-D sub-problem along the z-direction, the value of $\psi_{il}(z,t)$ at each grid point in the z-direction can be obtained by the following matrix equation:

$$\mathbf{G}_{il}\Psi_{il}(t) + \mathbf{C}\Psi_{il}'(t) = \mathbf{P}_{il}(t) \tag{42}$$

where $\Psi_{il}(t) = [\psi_{il}(z_0,t), \cdots \psi_{il}(z_r,t), \cdots \psi_{il}(z_{\Lambda-1},t)]^T$, z_r's are positions of grid points in the z-direction, $z_0 = 0$, $z_{\Lambda-1} = -L_z$, and Λ is the number of grid points. The $\mathbf{G}_{il}$'s and $\mathbf{C}$ are tri-diagonal and diagonal matrices, respectively, and $\mathbf{P}_{il}(t)$ is equal to $[0,\ldots,0,\widehat{p}_{il}(d_1,t),0,\ldots,0,\widehat{p}_{il}(d_3,t),0,\ldots,0,\widehat{p}_{il}(d_{2N_l-1},t), 0,\ldots,0]^T$.

When performing steady-state simulation, $\mathbf{P}_{il}(t)$ is a constant vector, and $\Psi_{il}'(t)$ is a zero vector. Hence, $\Psi_{il}(\infty)$ can be obtained without the time step evaluation. Moreover, because each $\mathbf{G}_{il}$ is tri-diagonal, each $\Psi_{il}(\infty)$ can be solved in linear

[5]The complexity of *sign-count* method [17] for obtaining the ortho-normal spatial bases in the z-direction for each "il" is proportional to "#Layers $\times \sum_{q=0}^{q=N_z-1} K_{ilq}$". Here, N_z is the truncation number in the z-direction, and K_{ilq} is the sign-count iterations for obtaining the eigenvalue λ_{ilq} of each ortho-normal spatial basis. The complexity of using [23] to obtain the spatial bases in the z-direction for each "il" is extremely high because it needs symbolic expression for the determinant of a #Layers $\times$ #Layers matrix and needs to perform the inverse Laplace transform.

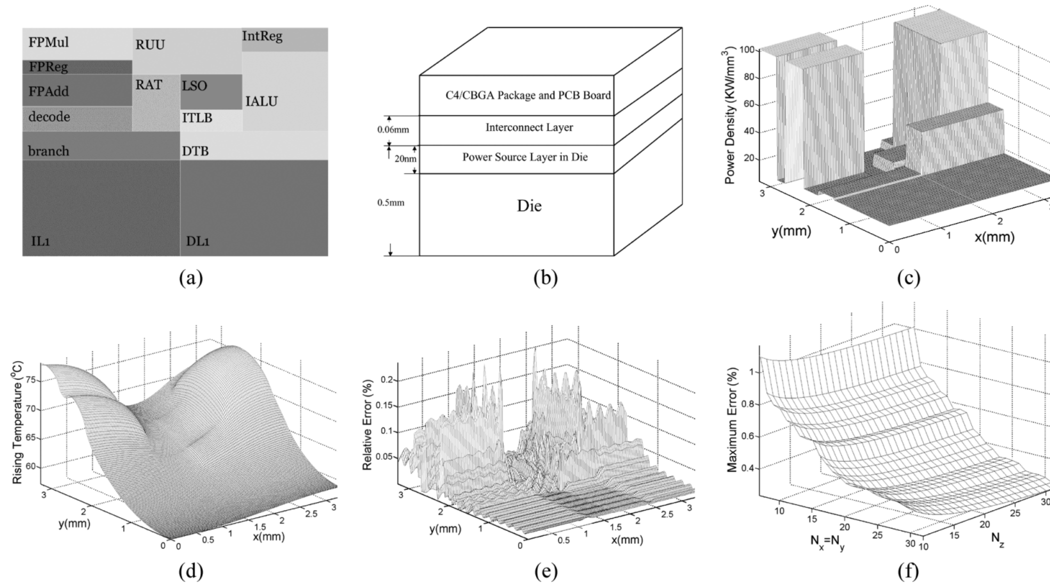

Fig. 12. Accuracy and the maximum error trend of a test chip. (a) Floorplan; (b) geometries of the test chip; (c) power distribution; (d) the rising temperature distribution of the top surface of the die; (e) the relative error distribution; and (f) the maximum relative error versus truncation point.

time. After solving $\Psi_{il}(\infty)$, the steady-state temperature of (36) at any z position of grid point can be cast into the similar form developed for 2-D ICs, and the proposed fast evaluating method can be used to calculate the temperature.

The transient analysis can be done by performing the time step evaluation to (42) for getting the value of $\Psi_{il}(t)$ at each time step, and then the proposed evaluating method is used to calculate the temperature at each time step. Note that, each $\mathbf{G}_{il}$ wouldn't change after functional blocks are replaced. Hence, once the LU decompositions of $\mathbf{G}_{il}$'s are done, they can be reused during the temperature-aware design flow.

V. EXPERIMENTAL RESULTS

We implement the proposed GIT-based thermal simulator and the Algorithm II of a highly efficient Green's function based method [6] in C++ language. The state-of-the-art FFT package, FFTW3 [24], is used to realize the DCT and IDCT for [6]. All methods are tested on a HP xw9300 workstation with 16 GB memory. The results are compared with a commercial computational fluid dynamic software ANSYS.

A. Accuracy and Fast Convergence of the GIT-Based Thermal Simulator

A chip, DEC Alpha 21264 [25], is employed to demonstrate the accuracy of our method, and its size is scaled down to 3.3 mm $\times$3.3 mm $\times$ 0.5 mm for the 65-nm technology. Its floorplan is shown in Fig. 12(a), and its die and package geometries are shown in Fig. 12(b). The equivalent thermal resistance of the package is set to be 45.5 °C/W [21].

The interconnect layer consists of 25% copper and 75% oxide with the thickness being equal to 0.06 mm, and its effective thermal conductivity is 101 W/(m·°C). The thickness of the power source layer is set to be 20 nm which is the nominal value of the device junction depth for the 65-nm technology [26]. The equivalent heat transfer coefficient of the primary heat flow path, h_p, is 8700 $\mathrm{W/(m^2 \cdot^\circ C)}$ [6], and the equivalent heat transfer coefficient of the secondary heat flow path, h_s, is 2017 $\mathrm{W/(m^2 \cdot^\circ C)}$.

To appropriately set the thermal conductivity of die, we apply the 1-D thermal model shown in Fig. 3 to compute the average temperature of die. To calculate the thermal resistance R_p, we apply the formula stated in [2], [15] to obtain $R_p = 1/(h_p A_{dz}) = 10.55\,^\circ$C/W. Here, A_{dz} is the cross area of die among the z-direction. The R_s is equivalent thermal resistance of the successively connected package and interconnect layers which is equal to 45.52 °C/W. Initially, $R_{\rm die}$ is calculated by using the thermal conductivity of die at the room temperature. After thermal resistances R_s, R_p and $R_{\rm die}$ are obtained, the average rising temperature $T_{\rm avg}$ of die is equal to

$$T_{\rm avg} = \frac{T_{\rm avg}(0) + T_{\rm avg}(-L_z)}{2} \tag{43}$$

where $T_{\rm avg}(0)$ and $T_{\rm avg}(-L_z)$ can be obtained by using the 1-D thermal model.

The $T_{\rm avg}$ got from (43) is the exact average rising temperature of die for its 1-D thermal model with given thermal resistors. Once $T_{\rm avg}$ is calculated, $R_{\rm die}$ is reset by using the thermal conductivity of die at $T_{\rm avg} + T_a$. This calculating procedure is repeated until $T_{\rm avg}$ converges. Here, the room temperature T_a is set to be 27 °C. With the above procedure, the average temperature is 90.9 °C, the thermal conductivity of die is 113.5 $\mathrm{W/(m \cdot^\circ C)}$, and $R_{\rm die} = 0.4\,^\circ$C/W.

The top surface of die is divided into 128 $\times$ 128 grid cells and the average power density profile is shown in Fig. 12(c). The average steady state rising temperature distribution on the top surface of the die computed by the proposed method with the truncation points being 32 in each x-, y-, and z-direction is shown in Fig. 12(d). The maximum relative error compared with the result of ANSYS is 0.24%, and its relative error distribution

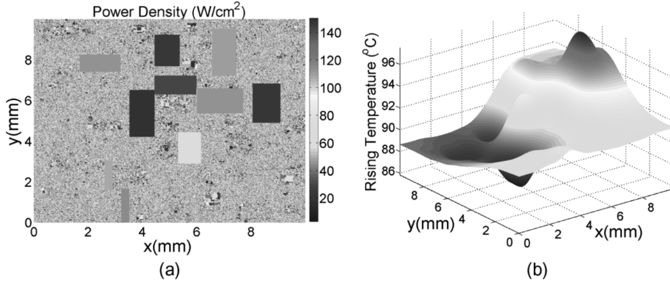

Fig. 13. Power density and temperature distribution of a 1 cm × 1 cm chip with one million functional blocks. (a) The power density distribution and (b) the rising temperature distribution.

TABLE I
ACCURACY AND RUNTIME COMPARISON OF THE PROPOSED GIT-BASED METHOD AND ALGORITHM II OF [6]

		Algorithm II of [6]	Our method
number of functional blocks		1 million	
number of grid cells		2^{20}	
number of bases		2^{22}	2^{11}
maximum error (%)		0.4143	0.3576
runtime (sec)	pre-calculating	2.47850	0.00005
	post-calculating	2.7642	0.1312
speedup (post-calculating)		21.0686	

is shown in Fig. 12(e). The relative error of each grid cell (m, n) is measured by

$$e_{mn} = \left| \frac{T_{mn}^{\text{ANSYS}} - \overline{T}_{mn}}{T_{mn}^{\text{ANSYS}}} \right| \tag{44}$$

where T_{mn}^{ANSYS} is the average rising temperature of grid cell (m, n) obtained by ANSYS. Note that, the $T_{\text{avg}}(0) = 65.14\,^{\circ}\text{C}$ got by the 1-D thermal model is consistent with the $T_{\text{avg}}(0) = 65.15\,^{\circ}\text{C}$ got by the proposed GIT-based method. This verifies the ability of 1-D thermal model for predicting the average temperature of the entire die.

To further demonstrate our fast error decaying rate, we plot the maximum relative errors with different truncation points in Fig 12(f). The result shows that the proposed GIT-based analyzer can achieve an extremely accurate solution even when the truncation points are very small.

B. Thermal Simulation for the Full-Chip Containing Lots of Functional Blocks

To demonstrate the capability of the proposed GIT-based method for the thermal simulation of full-chip with containing lots of functional blocks and the efficiency improvement over the Algorithm II of [6], a test chip with dimension of 1 cm × 1 cm × 0.5 mm and one million functional blocks is considered. The top surface of the chip is set to be adiabatic, and the power sources are assumed to be attached on the top surface of the die.[6] The setting is consistent with the setting in [6]. Fig. 13(a) shows the power density distribution of the functional blocks in W/cm^2. The top surface of the chip is divided into 1024 × 1024 grid cells. The truncation points of our GIT-based method are $16 \times 16 \times 8$ and the truncation points of [6] are 2048 × 2048 to achieve the same maximum error level. The average rising temperature distribution of the top surface got by our GIT-based method is shown in Fig. 13(b), and the maximum error is 0.3576% presented in Table I.

The runtime comparison is shown in Table I. The runtime of the post-calculating stage in our method is 0.1312 s while the runtime of the post-calculating stage in [6] is 2.7642 s. The speedup of our method over [6] is 21.07 at the post-calculating stage. This result demonstrates the substantial efficiency improvement of our thermal analyzer over [6].

[6]The power sources which are attached on the top surface of the die can be easily handled by deriving the integral transform pair with the assumption of the plane-power density on the top surface of die. The general solution can be found in [17]–[19].

C. Accuracy and Efficiency of the GIT-Based Thermal Simulator for the 3-D IC Thermal Analysis

To demonstrate the accuracy of our GIT-based thermal simulator for 3-D ICs, three chip layers are stacked and the power sources are distributed in three thin layers with the thickness being equal to the device junction depth. The lateral dimension of each chip layer is 3.3 mm × 3.3 mm. The thicknesses of insulator and silicon layers on both top and middle chips are scaled down to 15 and 10 μm, respectively. The thicknesses of insulator and silicon layers (including the substrate) for the bottom chip are 15 and 500 μm, respectively. The thermal parameters of each layer are referred to [9]. The top surface of each silicon layer is divided into 128 × 128 grid cells. The truncation point is 32 in each x- and y-direction, and the number of sampling points in the z-direction is 10 for each layer. Comparing with the result of ANSYS, our maximum error is 0.24% which demonstrates the accuracy of our method for 3-D ICs.

To show the efficiency of our GIT-based method for the cell-level thermal analysis in 3-D ICs, the top surface of each silicon layer is divided into 1024 × 1024 grid cells to mimic 1.05 million power sources. The truncation point and the number of sampling points in the z-direction are the same as the case of 128 × 128 grid cells. The average power density profile of each silicon layer is shown in Fig. 14(a), (c), and (e). The estimated average steady state rising temperature distribution on the top surface of each silicon layer is shown in Fig. 14(b), (d), and (f) from the top layer to the bottom layer. The runtime of our GIT-based method is 0.031 s for the pre-calculating stage (including the LU decomposition of each tri-diagonal matrix $\mathbf{G}_{il}$). The runtime of the post-calculating stage is only 0.48 s (including 0.016 s for calculating each $\Psi_{il}(\infty)$).

VI. CONCLUSION

An accurate and efficient GIT-based thermal simulator has been presented. Experimental results confirm its theoretical property which can achieve accurate results with sufficiently small truncation points. The proposed algorithm only takes 0.13 s for a chip with one million functional blocks and over one million grid cells, and 0.48 s for a 3-D IC with 3.146 million grid cells in the post-calculating stage to achieve accurately steady state temperature distribution. Therefore, the proposed GIT-based thermal simulator is very suitable for the thermal-aware design flow.

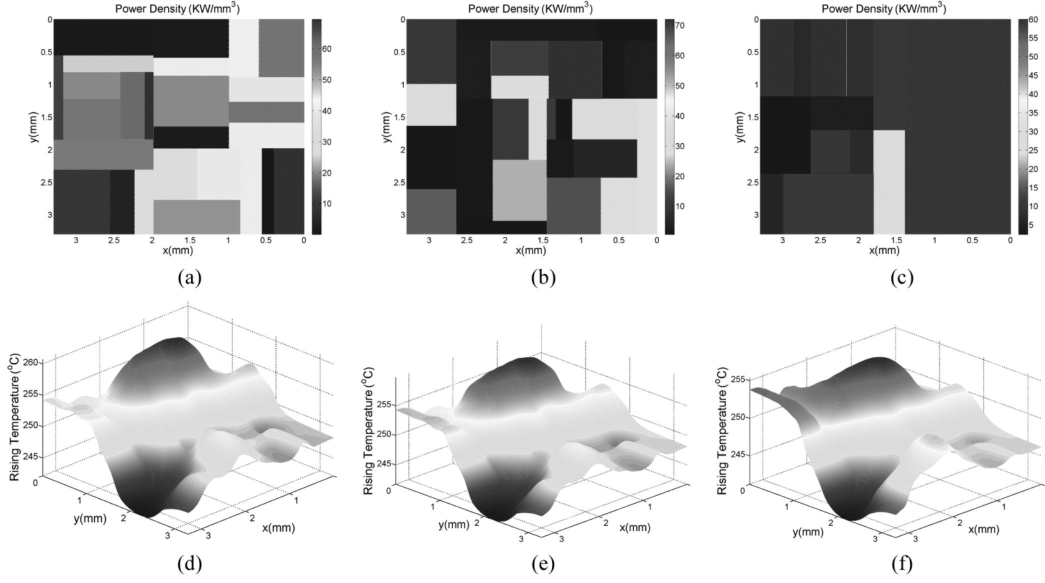

Fig. 14. Power density and temperature distribution of a test 3-D chip. (a)–(c) The power density distribution on the top surface of the top, middle, and bottom silicon layers, and (d)–(f) the temperature distribution on the top surface of the top, middle, and bottom silicon layers.

APPENDIX I
DERIVATION OF TIME-VARYING COEFFICIENTS FOR THE APPROXIMATED TEMPERATURE

To derive the uncoupled first-order differential (19) for each $\psi_{ilq}(t)$, both sides of (3) are multiplied by $\phi_{ilq}(\mathbf{r})$ and integrated over the region D of the die. After that, we have

$$\int_D \Delta^2 T(\mathbf{r},t)\phi_{ilq}(\mathbf{r})dv = \frac{\sigma}{\kappa}\int_D \frac{\partial T(\mathbf{r},t)}{\partial t}\phi_{ilq}(\mathbf{r})dv - \frac{1}{\kappa}\int_D p(\mathbf{r},t)\phi_{ilq}(\mathbf{r})dv \quad (45)$$

where $\int_D(\cdot)dv \equiv \int_{-L_z}^{0}\int_0^{L_y}\int_0^{L_x}(\cdot)dxdydz$.

The inward and outward flows of (45) must be balanced to satisfy the energy conservation law. Therefore, by applying the Divergence Theorem [18] to the left-hand side of (45), we have

$$\int_D \Delta^2 T(\mathbf{r},t)\phi_{ilq}(\mathbf{r})dv = \int_S \phi_{ilq}(\mathbf{r})\frac{\partial T(\mathbf{r},t)}{\partial n}ds - \int_D \nabla T(\mathbf{r},t)\cdot\nabla\phi_{ilq}(\mathbf{r})dv \quad (46)$$

where $\int_S(\cdot)ds$ is the surface integral of the boundary surface union S of the die, and $\partial/\partial n$ is the normal derivative on S in the outward direction.

By applying the Divergence Theorem to the second term in the right side of (46), we have

$$\int_D \nabla T(\mathbf{r},t)\cdot\nabla\phi_{ilq}(\mathbf{r})dv = \int_S T(\mathbf{r},t)\frac{\partial \phi_{ilq}(\mathbf{r})}{\partial n}ds - \int_D T(\mathbf{r},t)\nabla^2\phi_{ilq}(\mathbf{r})dv. \quad (47)$$

Inserting (47) into (46), and then putting the result into (45), we have

$$\sigma\int_D \frac{\partial T(\mathbf{r},t)}{\partial t}\phi_{ilq}(\mathbf{r})dv - \kappa\int_D T(\mathbf{r},t)\nabla^2\phi_{ilq}(\mathbf{r})dv = \kappa\int_S\left[\phi_{ilq}(\mathbf{r})\frac{\partial T(\mathbf{r},t)}{\partial n} - T(\mathbf{r},t)\frac{\partial\phi_{ilq}(\mathbf{r})}{\partial n}\right]ds + \int_D p(\mathbf{r},t)\phi_{ilq}(\mathbf{r})dv. \quad (48)$$

By plugging (8), (18), (4)–(6), and (9)–(11) into (48), we have the uncoupled first-order differential (19) for each $\psi_{ilq}(t)$.

APPENDIX II
ERROR BOUND ANALYSIS OF THE GIT-BASED STEADY-STATE TEMPERATURE FORMULATION

To proceed the error bound analysis of the GIT-based steady-state temperature formulation stated in Section III-C1, the following lemma is introduced.

Lemma 1: The magnitude of each time-varying coefficient $|\psi_{ilq}(\infty)|$ at the steady state is bounded by

$$|\psi_{ilq}(\infty)| \leq \begin{cases} \frac{2j_dP_T}{\lambda_{ilq}^2\kappa\sqrt{N_{ilq}}}\left(\frac{\kappa}{\lambda_{z_q}} + \frac{h_p}{\lambda_{z_q}^2}\right); & i=0, l=0 \\ \frac{4Nj_dP_T}{l\lambda_{ilq}^2\kappa\pi\sqrt{N_{ilq}}}\left(\frac{\kappa}{\lambda_{z_q}} + \frac{h_p}{\lambda_{z_q}^2}\right); & i=0, l\neq 0 \\ \frac{4Mj_dP_T}{i\lambda_{ilq}^2\kappa\pi\sqrt{N_{ilq}}}\left(\frac{\kappa}{\lambda_{z_q}} + \frac{h_p}{\lambda_{z_q}^2}\right); & i\neq 0, l=0 \\ \frac{8MNj_dP_T}{il\lambda_{ilq}^2\kappa\pi^2\sqrt{N_{ilq}}}\left(\frac{\kappa}{\lambda_{z_q}} + \frac{h_p}{\lambda_{z_q}^2}\right); & i\neq 0, l\neq 0 \end{cases} \quad (49)$$

where P_T is the total steady power consumption of die and j_d is the junction depth of device.

Since the time domain waveform of steady power profile can be treated as a step function, *Lemma* 1 can be easily proved by plugging (12) and (20) into (21), setting t to be infinity and with

several manipulations. With *Lemma* 1, an error bound of the GIT based formulation is given by the following theorem.

Theorem 1: The absolute error of average steady state temperature for each grid cell (m, n) by using the GIT-based formulation with truncation points of N_x, N_y and N_z in x-, y-, and z-directions is bounded by

$$\sum_{(i,l,q)\in S_1} \frac{\alpha_1\gamma_q}{i^2l^2\lambda_{ilq}^2} + \sum_{(i,q)\in S_2} \frac{\alpha_2\gamma_q}{i^2\lambda_{i0q}^2} + \sum_{(l,q)\in S_3} \frac{\alpha_3\gamma_q}{l^2\lambda_{0lq}^2} + \sum_{q\in S_4} \frac{\alpha_4\gamma_q}{\lambda_{00q}^2} \tag{50}$$

where

$$\gamma_q = \frac{1}{\lambda_{z_q}}\left(\kappa + \frac{h_p}{\lambda_{z_q}}\right)\frac{\kappa\lambda_{z_q}^2 + h_p^2}{\kappa\lambda_{z_q}^2 - h_p h_s}$$

and $S_1 = [1, N_x] \times (N_y, \infty) \times [0, \infty) \cup (N_x, \infty) \times [1, N_y] \times [0, \infty) \cup [1, N_x] \times [1, N_y] \times (N_z, \infty)$, $S_2 = [1, N_y] \times (N_z, \infty) \cup (N_x, \infty) \times [0, \infty)$, $S_3 = [1, N_x] \times (N_z, \infty) \cup (N_y, \infty) \times [0, \infty)$, $S_4 = (N_z, \infty)$, $\alpha_1 = 256M^2N^2j_dP_T/(\kappa L_xL_yL_z\pi^4)$, $\alpha_2 = 32M^2j_dP_T/(\kappa L_xL_yL_z\pi^2)$, $\alpha_3 = 32N^2j_dP_T/(\kappa L_xL_yL_z\pi^2)$, $\alpha_4 = 4j_dP_T/(\kappa L_xL_yL_z)$.

Proof: As pointed out in [18] and [19], (17) is convergent in mean when truncation points are infinities. Hence, the absolute truncation error is bounded as

$$|\epsilon_{mn}(z,\infty)| \leq \sum_{(i,l,q)\notin S} \left| \frac{\psi_{ilq}(\infty)}{\Delta x\Delta y} \int_{n\Delta y}^{(n+1)\Delta y} \int_{m\Delta x}^{(m+1)\Delta x} \phi_{ilq}(\mathbf{r})dxdy \right| \tag{51}$$

where $S = [0, N_x] \times [0, N_y] \times [0, N_z]$. Plugging (12) into (51), utilizing *Lemma 1* and with several manipulations, we can get the error bound (50). ■

Since the decaying rate of γ_q is dominated by $1/\lambda_{z_q}$, the error decaying rate of the GIT-based steady-state temperature formulation stated in Section III-C1 is dominated by $i^2l^2\lambda_{z_q}((i\pi/L_x)^2 + (l\pi/L_y)^2 + \lambda_{z_q}^2)$.

To compare the error bound of the GIT-based formulation with the error bound of [6] under the same boundary conditions and the plane-power source assumption, the error bound (50) can be simplified to

$$\sum_{(i,l,q)\in S_1} \frac{\alpha_1}{i^2l^2\lambda_{ilq}^2} + \sum_{(i,q)\in S_2} \frac{\alpha_2}{i^2\lambda_{i0q}^2} + \sum_{(l,q)\in S_3} \frac{\alpha_3}{l^2\lambda_{0lq}^2} + \sum_{q\in S_4} \frac{\alpha_4}{\lambda_{00q}^2} \tag{52}$$

where $\alpha_1 = 128M^2N^2P_T/(L_xL_yL_z\kappa\pi^4)$, $\alpha_2 = 16M^2P_T/(\kappa L_xL_yL_z\pi^2)$, $\alpha_3 = 16N^2P_T/(\kappa L_xL_yL_z\pi^2)$, and $\alpha_4 = 2P_T/(\kappa L_xL_yL_z)$.

The previous result shows that the error decaying rate of our GIT-based method can be in the order of $i^2l^2((i\pi/L_x)^2 + (l\pi/L_y)^2 + \lambda_{z_q}^2)$.

On the other hand, the error bound of the Green's function-based method shown in [6] can be similarly derived as

$$\sum_{(i,l)\in B_1} \frac{\beta_1}{i^2l^2\gamma_{il}} + \sum_{i\in B_2, l=0} \frac{\beta_2}{i^2\gamma_{il}} + \sum_{i=0, l\in B_3} \frac{\beta_3}{l^2\gamma_{il}} \tag{53}$$

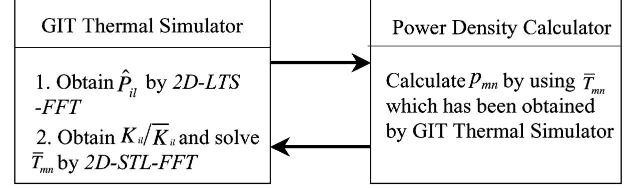

Fig. 15. Temperature-power iterative framework for dealing with the temperature dependence issue of leakage power.

where $\gamma_{il} = \sqrt{(i\pi/L_x)^2 + (l\pi/L_y)^2}$, $B_1 = (N_x, \infty) \times (N_y, \infty)$, $B_2 = (N_x, \infty)$, $B_3 = (N_y, \infty)$, $\beta_1 = 64M^2N^2P_T/(L_xL_y\kappa\pi^4)$, $\beta_2 = 8M^2P_T/(L_xL_y\kappa\pi^2)$, and $\beta_3 = 8N^2P_T/(L_xL_y\kappa\pi^2)$.

This bound shows that the error decaying rate of the Green's function based method [6] is in the order of $i^2l^2\sqrt{(i\pi/L_x)^2 + (l\pi/L_y)^2}$.

APPENDIX III
EXTENSION OF GIT FOR THE TEMPERATURE-DEPENDENCE ISSUE OF LEAKAGE POWER

By utilizing the temperature-power iterative frameworks [1], [7], [10], the proposed GIT-based method can be extended to consider the temperature-dependence issue of leakage power, as shown in Fig. 15. In the beginning, the power density profile is calculated at the room temperature and is immediately updated by applying the temperature-power iterative framework to the 1-D thermal model of the chip before carrying out the detail thermal simulation. Then, the temperature-power iterative framework is executed by recursively using the GIT thermal simulator and the power density calculator until they converge. Here, the precalculating stage only needs to be done once since it is independent of the power density profile.

Remarks: By integrating (17) from $z = -j_d$ to $z = 0$ and converting the result to the form which is suitable for performing *2D-STL-FFT*, a more accurate temperature estimation can be obtained. However, the difference between the top surface temperature distribution and the average temperature distribution for the power source layer is very small because the thickness of power source layer is very small.

ACKNOWLEDGMENT

The authors would like to thank the National Center for High-Performance Computing of Taiwan for the computer time and facilities.

REFERENCES

[1] 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 Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 5, pp. 501–513, May 2006.

[2] T.-Y. Wang and C. C.-P. Chen, "Thermal-ADI: A linear-time chip-level thermal simulation algorithm based on alternating-direction implicit (ADI) method," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 4, pp. 691–700, Aug. 2003.

[3] T.-Y. Wang and C. C.-P. Chen, "SPICE-compatible thermal simulation with lumped circuit modeling for thermal reliability analysis based on model reduction," in *Proc. Int. Symp. Quality Electron. Des.*, 2004, pp. 357–362.

[4] P. Li, L. T. Pileggi, M. Asheghi, and R. Chandra, "IC thermal simulation and modeling via efficient multigrid-based approaches," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 25, no. 9, pp. 319–326, Sep. 2006.

[5] J.-L. Tsai, C. C.-P. Chen, G. Chen, B. Goplen, H. Qian, Y. Zhan, S.-M. Kang, M. D. F. Wong, and S. S. Sapatnekar, "Temperature-aware placement for SOCs," *Proc. IEEE*, vol. 94, no. 8, pp. 1502–1518, Aug. 2006.

[6] Y. Zhan and S. S. Sapatnekar, "High efficiency Green function-based thermal simulation algorithms," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 26, no. 9, pp. 1661–1675, Sep. 2007.

[7] Y. Yang, Z. Gu, C. Zhu, R. P. Dick, and L. Shang, "ISAC: Integrated space-and-time-adaptive chip-package thermal analysis," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 26, no. 1, pp. 86–99, Jan. 2007.

[8] K. Banerjee, S. J. Souri, P. Kapur, and K. C. Saraswat, "3-D ICs: A novel chip design for improving deep-submicrometer interconnect performance and systems-on-chip integration," *Proc. IEEE*, vol. 89, no. 5, pp. 602–633, May 2001.

[9] G. L. Loi, B. T. Agrawal, N. Srivastava, S.-C. Lin, T. Sherwood, and K. Banerjee, "A thermally-aware performance analysis of vertically integrated (3-D) processor-memory hierarchy," in *Proc. Des. Autom. Conf.*, 2006, pp. 991–996.

[10] H. Hua, C. Mineo, K. Schoenfliess, A. Sule, S. Melamed, R. Jenkal, and W. R. Davis, "Exploring compromises among timing, power and temperature in three-dimensional integrated circuits," in *Proc. Des. Autom. Conf.*, 2006, pp. 997–1002.

[11] J. Cong, G. Luo, J. Wei, and Y. Zhang, "Thermal-aware 3D IC placement via transformation," in *Proc. Asia South Pacific Des. Autom.*, 2006, pp. 780–785.

[12] J. Cong, J. Wei, and Y. Zhang, "A thermal-driven floorplanning algorithm for 3D ICs," in *Proc. Int. Conf. Comput.-Aided Des.*, 2004, pp. 306–313.

[13] B. Goplen and S. Sapatnekar, "Efficient thermal placement of standard cells in 3D ICs using a force directed approach," in *Proc. Int. Conf. Comput.-Aided Des.*, 2003, pp. 86–89.

[14] K. Balakrishnan, V. Nanda, S. Easwarm, and S. K. Lim, "Wire congestion and thermal aware 3D global placement," in *Proc. Asia South Pacific Des. Autom.*, 2005, pp. 1131–1134.

[15] Y.-K. Cheng, P. Raha, C.-C. Teng, E. Rosenbaum, and S.-M. Kang, "ILLIADS-T: An electrothermal timing simulator for temperature-sensitive reliability diagnosis of CMOS VLSI chips," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 17, no. 8, pp. 668–681, Aug. 1998.

[16] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: Modeling and implementation," *ACM Trans. Arch. Code Opt.*, vol. 1, no. 1, pp. 94–125, Mar. 2004.

[17] M. D. Mikhailov and M. N. Ozisik, *Unified Analysis and Solutions of Heat and Mass Diffusion*. New York: Wiley, 1983.

[18] N. Y. Olcer, "On the theory of conductive heat transfer in finite region," *Int. J. Heat Mass Transfer*, vol. 7, pp. 307–314, 1964.

[19] M. D. Mikhailov, "General solutions of the heat equation in finite regions," *Int. J. Eng. Sci.*, vol. 10, pp. 577–591, 1972.

[20] J. Parry, H. Rosten, and G. B. Kromann, "The development of component-level thermal compact models of a C4/CBGA interconnect technology: The Motorola PowerPC 603 and PowerPC 604 RISC microprocessors," *IEEE Trans. Compon., Packag., Manuf. Technol. A*, vol. 21, no. 1, pp. 104–112, Mar. 1998.

[21] C. Lasance, H. Vinke, H. Rosten, and K.-L. Weiner, "A novel approach for the thermal characterization of electronic parts," in *Proc. IEEE Semi-Therm Symp.*, 1995, pp. 1–9.

[22] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C++*. Cambridge, U.K.: Cambridge Univ. Press, 2002.

[23] X. Lu, P. Tervola, and M. Viljanen, "A novel and efficient analytical method for calculation of the transient temperature field in a multi-dimensional composite slab," *J. Phys. A: Math. Gen.*, vol. 38, pp. 8337–8351, 2005.

[24] M. Frigo and S. G. Johnson, "FFTW Version 3.1 Package," 2004. [Online]. Available: http://www.fftw.org

[25] W. Liao, L. He, and K. Lepak, "Temperature-aware performance and power modeling," UCLA, Los Angeles, CA, Tech. Rep. UCLA Eng. 04-250, 2004.

[26] F. Lallement, B. Duriee, A. Grouillet, F. Amaud, B. Tavel, F. Wacquant, P. Stalk, M. Woo, Y. Erokhin, J. Scheuer, L. Gadet, J. Weeman, D. Distaso, and D. Lenoble, "Ultra-low cost and high performance 65 nm CMOS device fabricated with plasma doping," in *Symp. VLSI Technol. Dig. Tech. Papers*, 2004, pp. 178–179.

Pei-Yu Huang received the B.S. degree in electrical engineering from the National Taiwan University of Science and Technology, Taiwan, in 2004, and the M.S. degree from National Chiao-Tung University, Taiwan, in 2004, where he is pursuing the Ph.D. degree in the Department of Communication Engineering.

His research interests include computer-aided design of integrated circuits, thermal analysis, thermal optimization technique, and power grid analysis.

Yu-Min Lee (M'03) received the B.S. and M.S. degrees in communication engineering from the National Chiao-Tung University, Taiwan, in 1991 and 1993, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Wisconsin-Madison, Madison, in 2003.

Since 2003, he has been an Assistant Professor with the Department of Communication Engineering, National Chiao-Tung University. His research interests include computer-aided design on VLSI circuits with emphases on interconnect analysis and optimization, and circuit/thermal/electro-thermal simulation.

Dr. Lee was a recipient of the ISPD Best Paper Award in 2003.