A Fast Leakage Aware Thermal Simulator for 3D Chips

Hameedah Sultan
School of Information Technology
Indian Institute of Technology, New Delhi, India
Email: hameedah@cse.iitd.ac.in

Smruti R. Sarangi
Computer Science and Engineering
Indian Institute of Technology, New Delhi, India
Email: srsarangi@cse.iitd.ac.in

Abstract—**In this paper, we propose, *3DSim*, which is an ultrafast thermal simulator for 3D chips. It simulates the effects of both dynamic and leakage power. Our technique captures the steady state as well as the transient response with a high speed and good accuracy. *3DSim* uses an approach based on Green's functions, where a Green's function is defined as the impulse response of a unit power source. Our approach incorporates the effects of the leakage-temperature feedback loop, exploits the radial symmetry in the thermal profile, and uses Hankel transforms to yield a closed form solution for the leakage aware Green's function. To further speed up our technique, we use fast numerical discrete Hankel transforms, and pre-compute and store certain functions in a lookup table. Our approach fundamentally converts a 3D problem to a set of 1D problems, thus leading to a 68X speedup as compared to competing simulators with an error limited to 1.5 $^\circ$C .**

I. Introduction

With increased power density, chip temperature has been increasing steadily, resulting in several adverse effects. The most pronounced effect is the increase in leakage power. Leakage power is strongly dependent on temperature. Increase in leakage power further increases the on-chip temperature, resulting in a feedback effect. While there has been a significant amount of research in the area of fast thermal simulators [1], [2], [3], there is hardly any research that focuses on fast **leakage aware** thermal simulators [4].

Modeling leakage power is extremely necessary in modern day chips. In 45 nm technology and beyond, an estimated 30-50% percent of the total power is due to leakage [5]. Most existing thermal simulators incorporate leakage by computing the temperature profile, finding the leakage power and re-iterating several times to close the leakage-temperature feedback loop. This increases the run time by at least 3-5X (depending upon the number of iterations needed for convergence).

3D chips have an even more severe temperature issue. For 3D chips too, a fair number of thermal simulators exist[2], [3]. However, to the best of our knowledge, there is no 3D simulator that provides an analytical single-iteration solution for the chip temperature accounting for leakage, although it has been observed several times that high leakage is inevitable in such chips [6]. Furthermore, while calculating the transient response in 3D chips, state of the art simulators either ignore leakage, or suggest iterating multiple times until temperature and power values converge. However, we will justify in this paper that this method is prohibitively expensive.

Hence, we propose a novel 3D temperature simulator, *3DSim*, which is faster than contemporary simulators and takes into account the effect of leakage without requiring multiple iterations. Our simulator is capable of calculating the steady state as well as the transient thermal response. We propose a Green's function (impulse response of a power source) based methodology to obtain a **closed form solution** for temperature (in the transform domain). We also propose new algorithms for approximating sides and corners that are faster than the previous algorithms. We provide a method for further speed-up by using Hankel transforms instead of Fourier transforms, which are conventionally used in Green's function based methods. Further, we use Mathematica [7] to compute complex transforms and solve equations. We also use fast discrete Hankel transform algorithms, and deploy lookup tables to speed up the simulator further. Due to space constraints, we have moved some of the derivations and results to the Appendix. Our simulator is 68X faster than competing simulators for steady state simulations (HS3D [8] used by HotSpot and 3D-ICE [2]) and 71X faster for transient simulations. For accuracy, we compared our results against those obtained using a commercial CFD software, Ansys Icepak. The error is within 1.5°C , and is similar to that of the current simulators. In cases where the temperature dependence of leakage power changes, the Hankel transform based method, *3DSim* offers 82X speedup compared to Fourier transform based techniques (*3DSimF*).

II. Related Work and Background

A. Background

1) Leakage power: The leakage power, P_{leak} is exponentially dependent on temperature and can be described by the simplified BSIM 4 [9] model, as given by Equation 1.

$$P_{leak} \propto v_T^2 * e^{\frac{V_{GS}-V_{th}-V_{off}}{\eta * v_T}} (1 - e^{\frac{-V_{DS}}{v_T}}) \qquad (1)$$

where, v_T is the thermal voltage (kT/q), V_{th} is the threshold voltage, V_{off} is the offset voltage in the sub-threshold region and η is a constant. However, over the operating temperature range of real ICs, leakage may be assumed to be linearly dependent on temperature, as was shown experimentally in [10], [4]. The authors of these papers calculated the leakage power by taking a photograph of the power dissipation of functional units using IR cameras [10], and by performing HSpice simulations [4]. To verify these results, we calculated the leakage power using Equation 1 for the set of parameters shown in Table I. We then fit the results to a linear model using least squares based regression. The error using the linear model remained within 1% in the temperature range of 40-80°C . So we use a linear model given by Equation 2 to calculate leakage power as has been done in recent work [4].

$$P_{leak} = P_{leak0} + \beta(T - T_{amb}), \tag{2}$$

where, T_{amb} is the ambient temperature, P_{leak0} is the leakage power at ambient temperature, and $\beta = \frac{dP_{leak}}{dT}$ is a function of the electrical characteristics of the chip such as threshold voltage, and supply voltage.

TABLE I
PARAMETERS USED FOR CALCULATING LEAKAGE

Parameter	Value
T_{amb}	318.15 K
$V_{GS} = V_{DS}$	0.7 V
V_{th}	$0.15 - 0.002 * (T - Tamb)\ V$
V_{off}	0.0024 V
η	2
P_{leak0}	0.1 W

2) Green's function: The Green's function in the context of thermal simulation is the impulse response (temperature profile) of a unit point power source (the Dirac Delta function, δ). The final temperature profile is given by the convolution of the Green's function with the power profile (distribution of power within a chip) as follows:

$$\mathcal{T} = G \star P \tag{3}$$

Here, $\mathcal{T}$ is the temperature profile (change in temperature), $\star$ is the 2-D convolution operator, G is the Green's function, and P is the power map.

3) Hankel Transform: The 2-D Fourier transform of a radially symmetric function is equivalent to a zero order 1-D Hankel transform. The Hankel transform is defined as:

$$\mathcal{H}(f(r)) = H(s) = \int_0^\infty f(r)\mathcal{J}_0(sr)rdr \tag{4}$$

Here $\mathcal{J}_0$ is a Bessel function of the first kind of order 0, and $\mathcal{H}$ denotes the Hankel transform operator.

B. Related Work

For 2-D chips, a wide variety of techniques have been proposed for temperature simulation, which are fast as well as accurate [4], [11]. Most temperature simulation techniques solve the Fourier's law of heat conduction to arrive at a temperature simulation technique. However, this method is slow and the accuracy depends on the granularity of meshing.

In 3-D Thermal-ADI [3], the package and heat sink are modeled as convection boundary conditions represented by resistances on the 6 sides of the chip. The popular Hotspot [12] simulator uses the analogy between electrical and thermal circuits to solve the heat conduction equation. To close the leakage-temperature loop, they suggest iterating multiple times till convergence.

Another set of techniques first pre-compute the Green's function by applying a unit power source at the center of the die, and then calculate the temperature profile for a given power profile by convolving it with the pre-computed Green's functions [4], [1], [13]. In the Power Blurring approach [1], additional corrections are applied to the Green's functions for edges and corners. The authors propose a methodology for modeling the transient profile as well. But this method is not applicable to 3D chips.

In [4], a closed form temperature profile of a 2-D chip incorporating leakage has been obtained. They use a Green's function based method. Beginning with a linear model of leakage, they convolve the total power profile with the Green's function to obtain leakage aware temperature values. To convert the 2-D convolution to 1-D multiplication, they use the Hankel transform. To obtain the transient temperature profile, they incorporate a thermal capacitance. This is the work closest to our area. But this method is for 2D chips, and owing to the multiple heating effects in both lateral and vertical directions, this method cannot be trivially extended to 3D chips. We consequently developed a novel technique to realistically approximate the Green's functions in a 3-D chip.

A limited number of techniques have been proposed for 3-D chips. HS3D [8] extends the Hotspot temperature simulation technique [11] by adding multiple layers and vertical resistances for the interface material. We shall compare *3DSim* with this approach in Section IV. In 3D-ICE [2], the chip is discretized into several small cuboidal cells. The effect of incorporating microchannel cooling is studied by adding a convective term for the heat exchange by the fluids. The additional term is modeled as a temperature controlled heat source which translates to a voltage controlled current source in the equivalent RC circuit.

The **main focus of our technique is the speed of computation while considering leakage power**. The method of re-iterating till convergence consumes a lot of time, especially for transient analysis. Zhou et al. [6] show that ignoring temperature dependent leakage in modern-day chips can result in a 32% error. Furthermore, they also show that **a 20% reduction in peak temperature can be obtained by simply modeling temperature dependent leakage during floorplanning optimization, without any overheads. However, it increases the floorplanning run time by 56%.** Hence, often where speed is of utmost importance, researchers either ignore leakage, or make very crude approximations, compromising heavily on accuracy. Thus there is a strong need for a fast algorithm that can provide the steady state as well as the transient temperature profile, while incorporating the effects of leakage. We consequently devised a novel set of algebraic techniques to compute the leakage aware Green's function in a 3-D chip. We also propose novel approximations for sides and corners. We use a combination of several mathematical techniques and software to provide an analytical solution to a complex set of equations. Finally we compare our results against those obtained using a commercial CFD software.

III. Methodology

A. Model of the Chip

Let us assume that our chip has l active layers, and each layer has been discretized into an $n \times n$ grid.

When the chip is excited by an impulse power source, the temperature rise in a layer is affected by 3 factors: 1) dynamic power applied, 2) leakage power sources created in that layer, 3) leakage power sources created in other layers.

The secondary sources created in other layers make the calculation of the temperature profile of a layer very complex. Each layer heats every other layer, and the sources created in turn heat the other layers. Hence we propose a fast method for determining the final temperature values analytically in the next section. Note that in the rest of the discussion, we shall use the terms, Green's function and heat spreading function, interchangeably. Let us discuss the steady state solution and the transient solution separately.

B. Steady State Solution

1) Outline of the Method: This method broadly involves three steps:

1) **Pre-compute stage I(Offline):** Obtain the heat spreading function (Green's function) without leakage.
2) **Pre-compute stage II(Offline):** Calculate the leakage aware heat spreading function.
3) **Compute stage (Online):** Convolve it with the power profile to obtain the thermal profile

2) Obtaining the Green's Functions without Leakage: The Green's function can be pre-computed using either in-vivo thermal measurements, or a thermal simulator. It is obtained by applying an impulse power source in a layer, and measuring the temperature distribution across the chip. Let us call the temperature distribution in layer i because of a point impulse source in layer k as fsp_{ik}. These are then modified to obtain the leakage aware Green's functions.

3) 3D Leakage Aware Green's Function: When we apply a power source equal to $P_{dyn}\ Watts$, the temperature rise $\mathcal{T}$ according to Equation 3 is given by:

$$\mathcal{T} = f_{sp} \star (P_{dyn} + \Delta P_{leak}) \quad (5)$$

Let P_{dyn} be a delta function applied in layer k. We denote the temperature rise in layer i by $\mathcal{T}_{ik}$. As discussed in the previous Section, $\mathcal{T}_{ik}$ is affected by the dynamic power dissipated by layer k, as well as the leakage sources created in all the layers. From Equation 2, $\Delta Pleak_{ik} = \beta \mathcal{T}_{ik}$, which denotes the new leakage sources created in layer i, due to the source in layer k. Using these terms in Equation 5, we arrive at the following set of Equations:

$$\begin{aligned}\mathcal{T}_{1k} =& fsp_{1k} + \beta(fsp_{11} \star \mathcal{T}_{1k} + fsp_{12} \star \mathcal{T}_{2k} + fsp_{13} \star \mathcal{T}_{3k} + \\ & ... + fsp_{1l} \star \mathcal{T}_{lk}) \\ & ...\end{aligned}$$

To transform convolution to multiplication, we compute the 2-D Fourier Transform of the above Equations. This gives us a set of simultaneous linear equations in $\mathcal{F}(\mathcal{T}_{1k})$, $\mathcal{F}(\mathcal{T}_{2k})$, ... $\mathcal{F}(\mathcal{T}_{lk})$, where $\mathcal{F}$ denotes the Fourier transform operator. To solve this system of equations, we used Mathematica [7]. We make some more approximations and arrive at the analytical solution, which can be used directly to obtain the final temperature profile. We call this method *3DSimF*. However, it requires taking a 2-D transform, which is computationally expensive. To further speed-up the computation, we use the Hankel Transform. A 1-D zero order Hankel transform is equivalent to a 2-D Fourier Transform for a radially symmetric function. Thus the 2-D problem ($O(n^2)$ points) is reduced to a single dimension ($O(n)$ points).

Interested users may have a look at the detailed derivation in the Appendix.

$$\mathcal{H}(\mathcal{T}_{12}) = \frac{\begin{aligned}&\mathcal{H}(fsp_{12}) + 2\pi\beta(\mathcal{H}(fsp_{13})\mathcal{H}(fsp_{32}) - \mathcal{H}(fsp_{33})\mathcal{H}(fsp_{12}) \\ &\quad + ... + \mathcal{H}(fsp_{1l})\mathcal{H}(fsp_{l2}) - \mathcal{H}(fsp_{ll})\mathcal{H}(fsp_{12}))\end{aligned}}{1 - 2\pi\beta\left(\mathcal{H}(fsp_{11}) + \mathcal{H}(fsp_{22}) + ... + \mathcal{H}(fsp_{ll})\right)}$$
$$... \quad (6)$$

After computing the inverse Hankel transform of Equation 6, we can obtain the leakage aware Green's function. However, to calculate the final temperature profile, we would again need to take its transform. Hence we store it in the transform domain itself after converting the radial function to Cartesian coordinates.

4) Corrections for Edges and Corners: The boundary conditions are different at the sides and corners, and hence we need to treat these differently. By convolving the generic Green's function (obtained by applying a source at the center) with a power profile consisting of a Delta function at the corner, we should obtain the Green's function for the corner. But since we use 2-D FFT to perform the convolution, the temperature profile gets folded and split across opposite corners (because computing an FFT is equivalent to computing a circular convolution). Thus we obtain a temperature rise at all the four corners, even though we applied a source at only one corner. Hence to bring back the temperature values to the required locations, we add the values at the four opposite corners. In the cases of edges, we add the temperature values at the two opposite edges. Thus we obtain the corrected Green's functions.

In [1], the authors use the method of images and append multiple copies of the power map to account for the boundary conditions. This effectively achieves the same task that we perform. However, their method increases the size of the matrix used in convolution and hence the computation time.

5) Convolution with the Power Profile: The final step in calculating the thermal profile is to convolve the power map with the leakage aware Green's functions. The conventional approach is to compute the 2D FFTs of the dynamic power profile and the Green's function, and multiply them. After this, an inverse 2D-FFT operation yields the final thermal profile. However, we propose to save the leakage aware Green's functions in the transform domain itself. This will save us from calculating the inverse transform in step 2, and again the transform in step 3. In a 3-D scenario, the final thermal profile is obtained by adding the effects of dynamic power as well as the effects of leakage sources in each layer as given by Equation 7.

$$T_i = \mathcal{F}(\mathcal{T}_{i1})\mathcal{F}(\mathcal{P}_1) + \mathcal{F}(\mathcal{T}_{i2})\mathcal{F}(\mathcal{P}_2) + ... + \mathcal{F}(\mathcal{T}_{il})\mathcal{F}(\mathcal{P}_l) \quad (7)$$

where, $\mathcal{P}_i$ represents the sum of dynamic and leakage power sources in layer i, and T_i represents the final temperature rise in layer i.

C. Transient Solution

To calculate the transient profile, we add a capacitive term, which captures the temperature rise with time (as done in [4]). With no loss of generality, let us assume that the source is present in layer 2, and the chip has four active layers.

$$\begin{aligned}\mathcal{T}_{12} =& fsp_{12} + \beta\left(fsp_{11} \star \mathcal{T}_{12} + fsp_{12} \star \mathcal{T}_{22} + fsp_{13} \star \mathcal{T}_{32} + fsp_{14} \star \mathcal{T}_{42}\right) - \\ & C_1\left(fsp_{11} \star \frac{\partial T_{12}}{\partial t} + fsp_{12} \star \frac{\partial T_{22}}{\partial t} + fsp_{13} \star \frac{\partial T_{32}}{\partial t} + fsp_{14} \star \frac{\partial T_{42}}{\partial t}\right) \\ & \ldots\end{aligned} \quad (8)$$

We note that the non-capacitive terms on the right hand side of Equation 8 comprise the steady state response. Let us call this $\mathcal{T}_{12ss}$. We first compute the Hankel transform, to convert the convolution operations to multiplication. This results in a set of differential equations. We then compute the Laplace transforms in the time domain. This enables us to convert the differential equations to algebraic equations and we get a set of four linear equations. We finally arrive at:

$$\begin{aligned}\mathcal{L}(\mathcal{H}(\mathcal{T}_{12})) =& \mathcal{L}\left(\mathcal{T}_{12ss}\right) - \frac{2\pi C_1}{1-2\pi\beta fsp_{11}} \times \\ & \left(\mathcal{H}(fsp_{11})\mathcal{L}\left(\mathcal{H}\left(\frac{\partial T_{12}}{\partial t}\right)\right) + \mathcal{H}(fsp_{12})\mathcal{L}\left(\mathcal{H}\left(\frac{\partial T_{22}}{\partial t}\right)\right) + \right. \\ & \left. \mathcal{H}(fsp_{13})\mathcal{L}\left(\mathcal{H}\left(\frac{\partial T_{32}}{\partial t}\right)\right) + \mathcal{H}(fsp_{14})\mathcal{L}\left(\mathcal{H}\left(\frac{\partial T_{42}}{\partial t}\right)\right)\right) \\ & \ldots\end{aligned} \quad (9)$$

where, $\mathcal{L}$ is the Laplace transform operator.

Using properties of the Laplace transform, setting the temperature rise at $t = 0$ to be zero, and solving further, we arrive at Equation 10.

$$\begin{aligned}& s\mathcal{L}(\mathcal{H}(\mathcal{T}_{12})) = \mathcal{T}_{12ss} - s^2\frac{2\pi C_1}{1-2\pi\beta fsp_{11}}\left(\mathcal{H}(fsp_{11})\mathcal{L}(\mathcal{H}(T_{12})) + \right. \\ & \left. \mathcal{H}(fsp_{12})\mathcal{L}(\mathcal{H}(T_{22})) + \mathcal{H}(fsp_{13})\mathcal{L}(\mathcal{H}(T_{32})) + \mathcal{H}(fsp_{14})\mathcal{L}(\mathcal{H}(T_{42}))\right) \\ & \ldots\end{aligned} \quad (10)$$

where, s is the Laplace transform variable.

Let $f_{11} = \frac{2\pi C_1}{1-2\pi\beta fsp_{11}}$, which is constant with time.

The ideal approach would be to solve the set of equations in Equation 10 for $\mathcal{L}(\mathcal{H}(\mathcal{T}_{12}))$, calculate the inverse Laplace transform, followed by the inverse Hankel transform. However, the complexity involved quickly makes the problem intractable, even using numerical techniques.

Hence, we ignore the terms that have a minimal contribution to the computed temperature field. We only incorporate the first and second order effects produced by the current and adjoining layers (ignore third and higher order effects). This gives us two capacitive terms for each layer:

$$\begin{aligned}s\mathcal{L}(\mathcal{H}(\mathcal{T}_{12})) =& \mathcal{T}_{12ss} - s^2 f_{11}\left(\mathcal{H}(fsp_{11})\mathcal{L}(\mathcal{H}(T_{12})) + \mathcal{H}(fsp_{12})\mathcal{L}(\mathcal{H}(T_{22}))\right) \\ & \ldots \\ s\mathcal{L}(\mathcal{H}(\mathcal{T}_{42})) =& \mathcal{T}_{42ss} - s^2 f_{44}\left(\mathcal{H}(fsp_{43})\mathcal{L}(\mathcal{H}(T_{32})) + \mathcal{H}(fsp_{44})\mathcal{L}(\mathcal{H}(T_{42}))\right)\end{aligned}$$

Solving these, we get,

$$\mathcal{L}(\mathcal{H}(\mathcal{T}_{12})) = \frac{\mathcal{T}_{12ss} + sf_{22}\mathcal{H}(fsp_{22})\mathcal{T}_{12ss} - sf_{11}\mathcal{H}(fsp_{12})\mathcal{T}_{22ss}}{\begin{aligned}&s(-1 - f_{11}\mathcal{H}(fsp_{11})s - f_{22}\mathcal{H}(fsp_{22})s + \\ &f_{11}f_{22}\mathcal{H}(fsp_{12})\mathcal{H}(fsp_{21})s^2 - f_{11}f_{22}\mathcal{H}(fsp_{11})\mathcal{H}(fsp_{22})s^2)\end{aligned}}$$

Next we ignore the secondary feedback effects, (all terms except the first and third terms in the denominator, since the source is in layer 2), and calculate the numerical inverse Laplace transform using Mathematica:

$$\begin{aligned}\mathcal{H}(\mathcal{T}_{12}) =& \mathcal{H}(\mathcal{T}_{12ss}) - \frac{\mathcal{H}(\mathcal{T}_{22ss})f_{11}\mathcal{H}(fsp_{12})e^{-t/(f_{22}\mathcal{H}(fsp_{22}))}}{f_{22}\mathcal{H}(fsp_{22})} \\ & \ldots\end{aligned} \quad (11)$$

Now, to calculate the final leakage aware transient Green's function, we have to calculate the inverse Hankel transform of Equation 11. Thus we have:

$$\begin{aligned}& \mathcal{T}_{12} = \mathcal{T}_{12ss} - finv, \text{ where} \\ & finv = \mathcal{H}^{-1}\left(\frac{\mathcal{H}(\mathcal{T}_{22ss})f_{11}\mathcal{H}(fsp_{12})e^{-t/(f_{22}\mathcal{H}(fsp_{22}))}}{f_{22}\mathcal{H}(fsp_{22})}\right) \\ & \ldots\end{aligned} \quad (12)$$

$$finv = \mathcal{H}^{-1}\left(\frac{\mathcal{H}(\mathcal{T}_{2ss})C_1(1-2\pi\beta\mathcal{H}(fsp_{22}))\mathcal{H}(fsp_{12})e^{-t/(f_{22}\mathcal{H}(fsp_{22}))}}{C_2(1-2\pi\beta\mathcal{H}(fsp_{11}))\mathcal{H}(fsp_{22})}\right) \quad (13)$$

This equation is still too complex to be calculated analytically. To further simplify it, we note that the ratio of the terms $\frac{(1-2\pi\beta\mathcal{H}(fsp_{22})\mathcal{H}(fsp_{12})}{(1-2\pi\beta\mathcal{H}(fsp_{11})\mathcal{H}(fsp_{22})}$ is close to 1, when $h > \epsilon$, (where h is the Hankel transform variable, and $\epsilon \to 0$). So we replace this by a correction factor equal to 1.1 (based on empirical results).

$$finv_\epsilon^\infty = \mathcal{H}^{-1}\left(1.1 \times \frac{\mathcal{H}(\mathcal{T}_{22ss})C_1 e^{-t/(f_{22}\mathcal{H}(fsp_{22}))}}{C_2}\right) \quad (14)$$

where $finv_\epsilon^\infty$ is the value of $finv$ between ϵ and ∞. We further note that at lower frequencies ($h < \epsilon$),

$$finv_0^\epsilon = \int_0^\epsilon \frac{C_1}{C_2}\mathcal{H}(\mathcal{T}_{22ss})e^{-\frac{t}{f_{22}\mathcal{H}(fsp_{22})}}\mathcal{J}_0(hr)hdh \quad (15)$$

where $finv_0^\epsilon$ is the value of $finv$ between 0 and ϵ.

Since $\epsilon \to 0$, the product $hr \to 0$, and thus $\mathcal{J}_0(hr) \to 1$. At $h \to \epsilon$, $f_{22} = f_{220} = \frac{2\pi C_2}{1-2\pi\beta\mathcal{H}(fsp_{22})|_{h=0}}$,

Also, since $\delta(h)/h \gg \beta\mathcal{T}$ when $h < \epsilon$, we can ignore all the leakage terms. We can thus approximate $\mathcal{H}(\mathcal{T}_{22ss}) = \mathcal{H}(f_{sp22}) = \mathcal{H}(fsp_{22})|_{h=0}\delta(h)/h$. Thus we have,

$$\begin{aligned}finv_0^\epsilon &= \int_0^\epsilon \mathcal{H}(fsp_{22})|_{h=0}\frac{\delta(h)}{h}e^{-\frac{t}{f_{220}\mathcal{H}(fsp_{22})|_{h=0}\frac{\delta(h)}{h}}}hdh \\ &= \frac{(\mathcal{H}(fsp_{22})|_{h=0})^2 f_{220}}{\epsilon^2 t}\left(1 - e^{\frac{-t\epsilon^2}{f_{220}\mathcal{H}(fsp_{22})|_{h=0}}}\right)\end{aligned} \quad (16)$$

As h increases, $finv_0^\epsilon \to 0$, and $finv_\epsilon^\infty$ will dominate. The final transient Green's function can be found using Equation 17.

$$\begin{aligned}& \mathcal{T}_{12} = \mathcal{T}_{12ss} - finv_0^\epsilon - finv_\epsilon^\infty \\ & \ldots\end{aligned} \quad (17)$$

1) Techniques used for Further Speed-up: A continuous Hankel transform requires the calculation of the Bessel function for each value of r. To further speed-up the simulation, we use the technique proposed by Johnson [14] to compute the discrete Hankel transform (Equation 18):

$$\mathcal{H}(j_{0,k}) \approx \frac{1}{{j_{0,N+1}}^2} \sum_{n=1}^{N} \frac{2}{J_1^2(j_{0,n})} f(j_{0,n}/j_{0,N+1}) J_0(j_{0,k}j_{0,n}/j_{0,N+1}) \tag{18}$$

where $j_{0,k}$ is the k^{th} root of the Bessel function, and N is the defined range of the function whose Hankel transform is to be computed. To avoid computing the roots of the Bessel function in each iteration, we pre-compute the roots and store it in a look-up table.

IV. Evaluation

A. Architecture and Modeling of the 3D Chip

Our 3D chip has four active layers of Silicon, with four layers of thermal interface material (TIM) in between them. Each layer of Silicon has the dimensions: $1cm \times 1cm \times 0.015cm$ and has 16*16 = 256 grid points (deemed to be enough by [4]). The top layer is attached to a heat spreader of dimensions $2cm \times 2cm \times 1cm$, which is attached to a heat sink. The top layer of the spreader can be thought of as an isothermal surface (since it is attached to the heat sink), while all other external surfaces of the chip are adiabatic.

TABLE II
Parameters of the chip

Parameter	Value
No. of layers, l	4
No. of grid points ,n	16
β	0.0044
Die size	100 mm^2
Die thickness	0.15 mm
Silicon conductivity	100 W/m-K
TIM thickness	0.02 mm
TIM conductivity	4 W/m-K
Spreader thickness	10 mm
Spreader conductivity	400 W/m-K

We use the commercial CFD simulator, Ansys Icepak, for thermal simulation. It is based on the Fluent CFD package. Multilevel meshing has been used to separately mesh the chip and the spreader.

Our routines are written in R and Matlab for computing and manipulating Green's functions. We begin by applying a 1 W point power source at the center of the chip at grid point (9, 8) to obtain the Green's function without leakage from Ansys Icepak. We do this for each layer. Next we employ our methodology to compute leakage aware Green's functions. To validate our results, we need to calculate Green's functions with leakage from Icepak. We iteratively calculate the leakage of each core (using BSIM4 models), apply an equivalent amount of additional power to emulate leakage, and re-run the simulation. All our results have been obtained on an Intel i7 (2.8 GHz) desktop PC running 64-bit Windows 8.

For the transient version, it is extremely difficult to calculate the leakage aware Green's functions using a thermal simulator like Icepak. This is because in order to obtain the leakage converged temperature profile, we need to calculate the additional leakage power based on the temperature at a given time instant, and re-iterate till the power-temperature values converge. In a transient setting, where the temperature changes, we pretty much need steady state (leakage-converged) solutions at each point in time. This is prohibitively expensive. Hence, for validating our results, we choose a point in time (say t_i), and run the complete transient simulation (till steady state is achieved) multiple times with an additional leakage power each time, corresponding to the temperature rise at this time instant ($\mathcal{T}(t_i)$). We do this activity for a fixed number of chosen points in time, and interpolate the results in between.

B. Steady State Results

1) Accuracy: We compared our results with those obtained using Ansys Icepak. The method involves two stages: the pre-compute stage which is off-line, in which we calculate the leakage aware Green's functions, and the compute stage, which is online, in which we convolve the Green's function with the power profile.

Pre-compute stage: We found that the maximum error in calculating the Green's functions using the Fourier transform (*3DSimF*) was 0.2 °C in all cases. In terms of percentage, the maximum error was limited to 3%. Using Hankel transforms (*3DSim*), the percentage errors were lower in some cases, and higher in others. The average error was 5.5%. Although both these methods should yield the same accuracy, (*3DSim*) has a higher error since there are sophisticated packages available for computing the Fourier transform, whereas the code for the Hankel transform has been implemented by us and thus there are more issues with precision. We found out that the process of calculating the Hankel transform and the inverse Hankel transform yields an error between 1-7%, with an average error of 4%. To correct for these numerical errors, we added a correction factor of 1.04. This reduced our average error to less than 3.5%.

Compute stage: The error in the total temperature profile for the power profile described in Table III was limited to 0.46 °C using Fourier transforms and 1.5 °C using Hankel transforms.

TABLE III
Location and magnitude of dynamic power sources on the chip

Grid point	Layer	Power (W)
(12,5)	Layer 1	1 W
(6,5)	Layer 2	3 W
(9,8)	Layer 2	1 W
(9,8)	Layer 3	2 W
(12,13)	Layer 4	4 W

2) Speed: ***3DSimF* pre-compute stage**: Computing the Fourier transform of all the spreading functions requires 32 ms. Then it takes an additional 7 ms to calculate the leakage aware functions in the transform domain. These functions are then stored to be used later in the online stage.

***3DSimF* Compute stage**: The runtime of the online compute stage is 16 ms.

***3DSim* pre-compute stage**: To compute the Hankel transform of the Green's functions we need 0.96 s. The higher computation time needed to calculate the modified Green's functions using Hankel transforms is because we need to compute the results of Bessel functions. However, since this part is offline, and does not need to be computed again unless the parameters change, the higher simulation time is not a problem. Then, for the second sub-stage, it takes an additional 85 μs to calculate the leakage aware functions in the domain of Hankel transforms. Here, we have an 82X speedup as compared to *3DSimF*. This is due to the fact that the size of the problem is reduced by an order of magnitude, and since we calculate 16 Green's functions, we have significant savings in time.

***3DSim* Compute stage**: The running time of the compute stage is 16 ms for *3DSim* (almost the same as *3DSimF*). The results are summarized in Table IV.

To compute the same functions using Ansys Icepak, we require 2-3 hours, depending on the number of iterations needed for convergence. We also calculated the leakage converged temperature values for a similar chip configuration using the latest version of Hotspot (HS3D has now been completely incorporated into Hotspot [15]), and found the execution time to be 1.1 s (as compared to 16ms for *3DSim*). The authors of 3D-ICE[2] report similar execution times without considering leakage (using similar hardware). Therefore, our algorithm provides $\sim 68X$ speedup as compared to state of the art thermal simulators for steady state analysis of 3-D chips, with a similar, if not better, accuracy (see Table IV).

TABLE IV
SPEED AND ACCURACY OF POPULAR SIMULATORS (STEADY STATE)

Simulator	Execution Time	Error (°C)
Ansys	$2-4hours$	–
Hotspot (HS3D)	1.1 s	1.4 [15]
3D-ICE	1.1 s [2]	1.6
3DSim	0.016 s	1.5
3DSimF	0.016 s	0.46

C. Transient Results

For the transient temperature profile, the error obtained using our algorithm was limited to 6%. The execution time to obtain the leakage converged Green's functions depends on the number of time points chosen for simulation. For 100 points between 0 and 0.05 s, the CPU execution time remains limited to 3.5 s. As discussed earlier, obtaining the leakage converged Green's functions using simulators such as Hotspot and 3D-ICE is prohibitively expensive since multiple iterations must be run for each point in time. If five iterations are needed for convergence (as is the case in a majority of our experiments), then over 100 points in time, the execution time gets multiplied by 500. We simulated a model with a complexity similar to ours on 3D-ICE, and obtained the execution time for obtaining the transient profile to be 0.5 s. Thus to obtain a leakage converged transient profile, we need 250 s. Hence, our algorithm provides a 71 times speed-up over 3D-ICE.

V. CONCLUSION

In this work, we propose a fast analytical method for quickly modeling the temperature profile of a 3D chip, which includes the effects of leakage. We provide a rigorous method and a novel algebraic framework for modeling the cross layer heat transfer effects. We have evaluated our model in terms of accuracy and simulation time. Our accuracy (as measured against Ansys Icepak) is comparable and in some cases better than competing simulators in this area. Moreover, our algorithm is 68 times faster than competing simulators. The reasons for the speedup come from the fact that we develop a closed form solution, which incorporates the effect of leakage, in the transform domain. This eliminates the need to perform multiple iterations for leakage-temperature convergence and converts a 3-D problem to a primarily 1-D problem using the *3DSim* technique.

REFERENCES

[1] A. Ziabari, J.-H. Park, E. K. Ardestani, J. Renau, S.-M. Kang, and A. Shakouri, "Power blurring: Fast static and transient thermal analysis method for packaged integrated circuits and power devices," *VLSI Systems, IEEE Transactions on*, vol. 22, no. 11, pp. 2366–2379, 2014.

[2] A. Sridhar, A. Vincenzi, M. Ruggiero, T. Brunschwiler, and D. Atienza, "3D-ICE: Fast compact transient thermal modeling for 3D ICs with inter-tier liquid cooling," in *ICCAD, 2010*.

[3] T.-Y. Wang, Y.-M. Lee, and C. C.-P. Chen, "3D thermal-ADI: an efficient chip-level transient thermal simulator," in *ISPD, 2003*.

[4] S. R. Sarangi, G. Ananthanarayanan, and M. Balakrishnan, "Lightsim: A leakage aware ultrafast temperature simulator," in *ASPDAC, 2014*.

[5] S. Borkar, "Low power design challenges for the decade (invited talk)," in *ASPDAC, 2001*.

[6] P. Zhou, Y. Ma, Z. Li, R. P. Dick, L. Shang, H. Zhou, X. Hong, and Q. Zhou, "3D-STAF: scalable temperature and leakage aware floorplanning for three-dimensional integrated circuits," in *ICCAD, 2007*.

[7] I. Wolfram Research, "Mathematica," Illinios, 2012.

[8] W.-L. Hung, G. M. Link, Y. Xie, N. Vijaykrishnan, and M. J. Irwin, "Interconnect and thermal-aware floorplanning for 3D microprocessors," in *ISQED, 2006*.

[9] W. Liu, K. Cao, X. Jin, and C. Hu, "BSIM 4.0.0 technical notes," EECS Department, University of California, Berkeley, Tech. Rep. UCB/ERL M00/39, 2000.

[10] Y. Liu, R. P. Dick, L. Shang, and H. Yang, "Accurate temperature-dependent integrated circuit leakage power estimation is easy," in *DATE, 2007*.

[11] W. Huang, K. Skadron, S. Gurumurthi, R. J. Ribando, and M. R. Stan, "Differentiating the roles of IR measurement and simulation for power and temperature-aware design," in *ISPASS, 2009*.

[12] 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," *VLSI Systems, IEEE Transactions on*, vol. 14, no. 5, pp. 501–513, 2006.

[13] Y. Zhan and S. S. Sapatnekar, "High-efficiency green function-based thermal simulation algorithms," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 26, no. 9, pp. 1661–1675, 2007.

[14] H. F. Johnson, "An improved method for computing a discrete hankel transform," *Computer physics communications*, vol. 43, no. 2, pp. 181–202, 1987.

[15] R. Zhang, M. R. Stan, and K. Skadron, "Hotspot 6.0: Validation, acceleration and extension," University of Virginia, Tech. Rep., 2015.

Appendix

When we apply a power source equal to $P_{dyn}\ Watts$, the temperature rise $\mathcal{T}$ according to Equation 3 is given by Equation 19:

$$\mathcal{T} = f_{sp} \star (P_{dyn} + \Delta P_{leak}) \tag{19}$$

Let P_{dyn} be a delta function applied in layer k. the heat spreading function in layers 1 to l would be denoted by fsp_{1k}, fsp_{2k}, ... fsp_{lk}. Thus, fsp_{ik} denotes the effect in layer i, when a point source is applied in layer k. Let the temperature rise in layer i , when a point source is applied in layer k be denoted by $\mathcal{T}_{ik}$. As discussed in the previous Section, $\mathcal{T}_{ik}$ is affected by the dynamic power dissipated by layer k, as well as the leakage sources created in all the layers. From Equation 2, $\Delta Pleak_{ik} = \beta \mathcal{T}_{ik}$, which denotes the new leakage sources created in layer i, due to the source in layer k. Using these terms in Equation 19, and the fact that convolution of any function with a delta function is the function itself, we arrive at the following Equations:

$$\begin{aligned}
\mathcal{T}_{1k} =& fsp_{1k} + \beta(fsp_{11} \star \mathcal{T}_{1k} + fsp_{12} \star \mathcal{T}_{2k} + fsp_{13} \star \mathcal{T}_{3k} + \\
& ... + fsp_{1l} \star \mathcal{T}_{lk}) \\
& ... \\
\mathcal{T}_{lk} =& fsp_{lk} + \beta(fsp_{41} \star \mathcal{T}_{1k} + fsp_{42} \star \mathcal{T}_{2k} + fsp_{43} \star \mathcal{T}_{3k} + \\
& ... + fsp_{4l} \star \mathcal{T}_{lk})
\end{aligned} \tag{20}$$

To transform convolution to multiplication, we compute the 2-D Fourier Transform of the above Equations. This allows us to solve for $\mathcal{F}(\mathcal{T}_{ik})$, where $\mathcal{F}$ denotes the Fourier transform operator:

$$\begin{aligned}
\mathcal{F}(\mathcal{T}_{1k}) &= \frac{\begin{array}{r}\mathcal{F}(fsp_{1k}) + \beta(\mathcal{F}(fsp_{12})\mathcal{F}(\mathcal{T}_{2k}) + \mathcal{F}(fsp_{13})\mathcal{F}(\mathcal{T}_{3k}) \\ + ... + \mathcal{F}(fsp_{1l})\mathcal{F}(T_{lk}))\end{array}}{1 - \beta\mathcal{F}(fsp_{11})} \\
&... \\
\mathcal{F}(\mathcal{T}_{lk}) &= \frac{\begin{array}{r}\mathcal{F}(fsp_{lk}) + \beta(\mathcal{F}(fsp_{l1})\mathcal{F}(\mathcal{T}_{1k}) + \mathcal{F}(fsp_{l2})\mathcal{F}(\mathcal{T}_{2k}) \\ + ... + \mathcal{F}(fsp_{l(l-1)})\mathcal{F}(T_{(l-1)k}))\end{array}}{1 - \beta\mathcal{F}(fsp_{ll})}
\end{aligned} \tag{21}$$

This gives us a set of simultaneous linear equations in $\mathcal{F}(\mathcal{T}_{1k})$, $\mathcal{F}(\mathcal{T}_{2k})$, ... $\mathcal{F}(\mathcal{T}_{lk})$. To solve this system of equations, we used Mathematica [7].

Now, β is generally a very small number. For most modern day chips, it is of the order of 10^{-3}. Hence we neglect all terms containing powers of β greater than one. Assume we have the source in layer 2. We then arrive at Equation 22.

$$\mathcal{F}(\mathcal{T}_{12}) = \frac{\begin{array}{r}\mathcal{F}(fsp_{12}) + \beta(\mathcal{F}(fsp_{13})\mathcal{F}(fsp_{32}) - \mathcal{F}(fsp_{33})\mathcal{F}(fsp_{12}) \\ + ... + \mathcal{F}(fsp_{1l})\mathcal{F}(fsp_{l2}) - \mathcal{F}(fsp_{ll})\mathcal{F}(fsp_{12}))\end{array}}{1 - \beta\left(\mathcal{F}(fsp_{11}) + \mathcal{F}(fsp_{22}) + ... + \mathcal{F}(fsp_{ll})\right)} \tag{22}$$

In Equation 22, the first term fsp_{12} denotes the temperature rise in layer 1 due to the dynamic power present in layer 2. The second term, $\beta\mathcal{F}(fsp_{13})\mathcal{F}(fsp_{32})$, accounts for secondary effects of the dynamic power source. The dynamic power source creates leakage sources in the adjoining layer (layer 3), which in turn act as sources themselves and heat the other layers. The effect of these sources in layer 1 is captured by the second term.

Equation 22 can be used directly to obtain the final temperature profile. We call this method *3DSim$\mathcal{F}$*. However, it requires taking a 2-D transform, which is computationally expensive ($O(n^2)$). To further speed-up the computation, we observe that for large die sizes, the temperature profile is radially symmetric, and has information only in one direction. Hence we can convert Cartesian co-ordinates to polar co-ordinates and use the Hankel Transform. A 1-D zero order Hankel transform is equivalent to a 2-D Fourier Transform for a radially symmetric function. Thus the 2-D problem ($O(n^2)$) is reduced to a single dimension ($O(n)$). Now, according to the definition of the Hankel transform that we use (Equation 4), we require an additional factor of 2π when calculating the transform of a convolution of two functions.

$$\begin{aligned}
\mathcal{H}(\mathcal{T}_{12}) &= \frac{\begin{array}{r}\mathcal{H}(fsp_{12}) + 2\pi\beta(\mathcal{H}(fsp_{13})\mathcal{H}(fsp_{32}) - \mathcal{H}(fsp_{33})\mathcal{H}(fsp_{12}) \\ + ... + \mathcal{H}(fsp_{1l})\mathcal{H}(fsp_{l2}) - \mathcal{H}(fsp_{ll})\mathcal{H}(fsp_{12}))\end{array}}{1 - 2\pi\beta\left(\mathcal{H}(fsp_{11}) + \mathcal{H}(fsp_{22}) + ... + \mathcal{H}(fsp_{ll})\right)} \\
&...
\end{aligned} \tag{23}$$

After computing the inverse Hankel transform of Equation 23, we can obtain the leakage aware Green's function. However, to calculate the final temperature profile, we would again need to take its transform. Hence we store it in the transform domain itself after converting the radial function to Cartesian coordinates. Let us refer to this avatar of the simulator as *3DSim*.

Thus we divide step 2 into two sub-stages. In the first sub-stage, we compute the transform of the heat spreading functions (Fourier transform in the case of *3DSimF*, and Hankel transform in the case of *3DSim*). In the second sub-stage, we use Equations 22 for *3DSimF* and 23 for *3DSim*, to calculate the final Green's functions in the transform domain. The advantage of this approach is that whenever β changes (as a result of varying supply voltage or the threshold voltage, or voltage-frequency scaling), we will have to re-run only the second sub-stage, that is, the calculation of the final Green's functions.

Figure 1 shows one of the calculated steady state Green's functions (for layer 2) along with the one obtained from Icepak. The temperature profile rapidly decays down to less than 0.1°C within four grid points (0.25 cm), thereby indicating that the boundary effects should be minimal. Also the temperature profile is radially symmetric, as shown by the contour map of the Green's function in Figure 3. The temperature profile obtained when multiple power sources are applied is shown in Figure 2.

The transient temperature profile obtained is shown in Figure 4

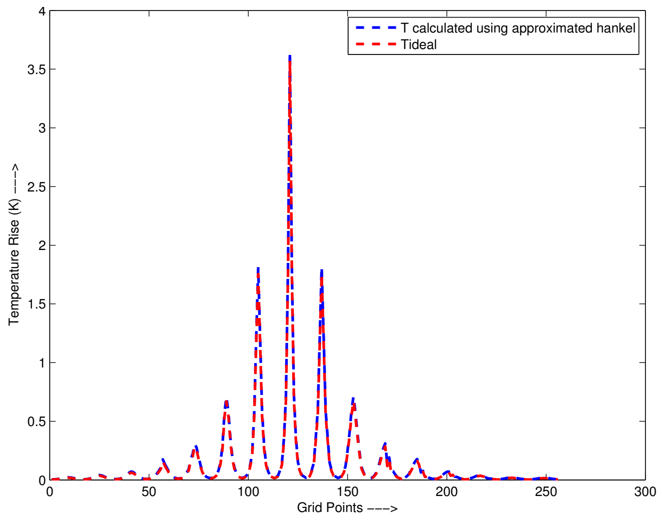

Fig. 1. Leakage Aware Green's function for Layer 2 with Source in Layer 3

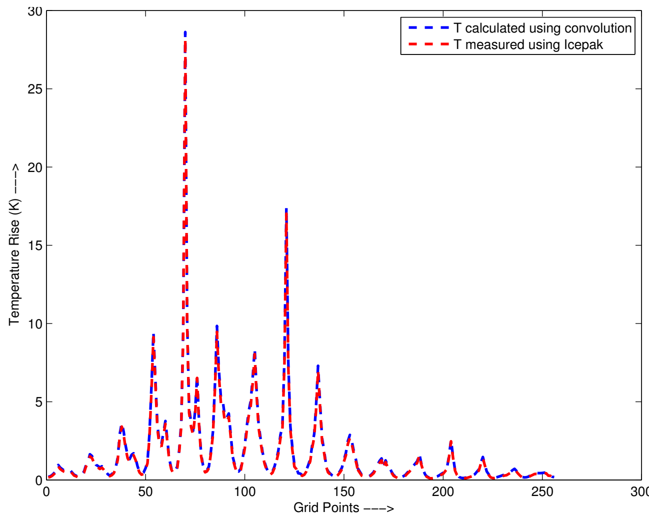

Fig. 2. Final temperature profile for Layer 2 for the power profile in Table III

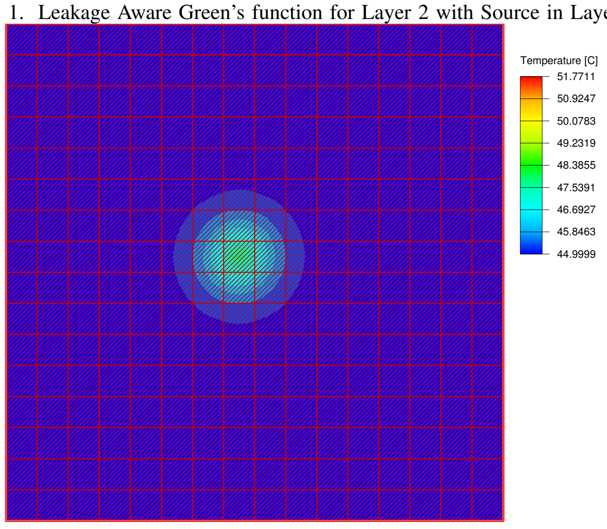

Fig. 3. Contour map for the Green's function in Figure 1

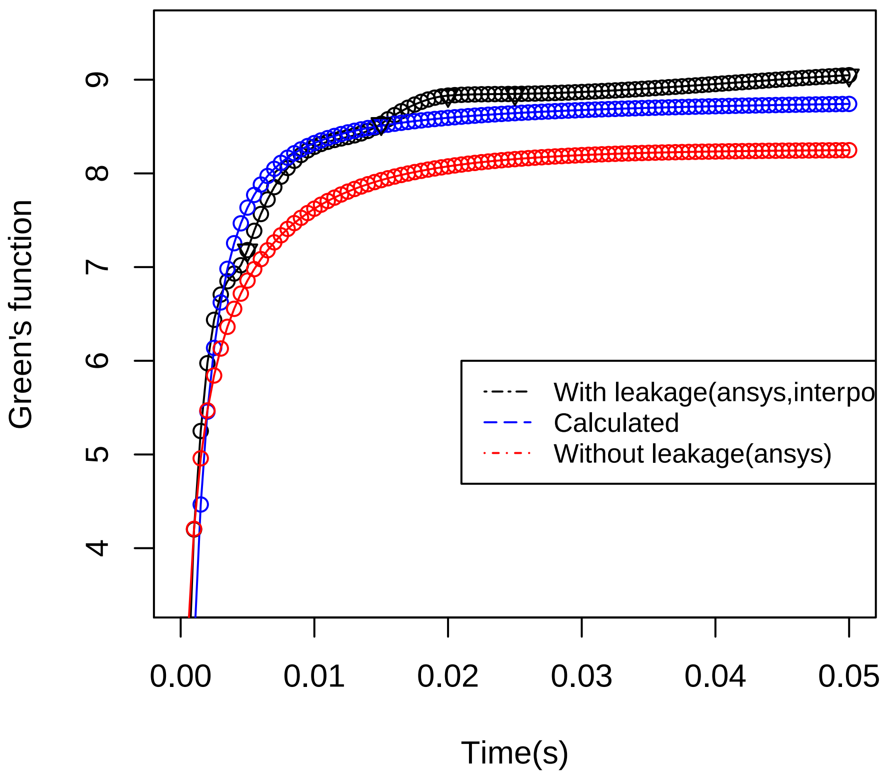

Fig. 4. Transient temperature profile for layer 2