INTERNATIONAL JOURNAL OF CIRCUIT THEORY AND APPLICATIONS
Int. J. Circ. Theor. Appl. 2011; **39**:905–921
Published online 28 June 2010 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cta.675

Finite difference schemes for heat conduction analysis in integrated circuit design and manufacturing

Yijiang Shen[1,*,†], Ngai Wong[1], Edmund Y. Lam[1] and Cheng-Kok Koh[2]

[1]*Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong*
[2]*School of Electrical and Computer Engineering, Purdue University, West Lafayette, U.S.A.*

SUMMARY

The importance of thermal effects on the reliability and performance of VLSI circuits has grown in recent years. The heat conduction problem is commonly described as a second-order partial differential equation (PDE), and several numerical methods, including simple explicit, simple implicit and Crank–Nicolson methods, all having at most second-order spatial accuracy, have been applied to solve the problem. This paper reviews these methods and further proposes a fourth-order spatial-accurate finite difference scheme to better approximate the PDE solution. Moreover, we devise a fourth-order accurate approximation of the convection boundary condition, and apply it to the proposed finite difference scheme. We use a block cyclic reduction and a recently developed numerically stable algorithm for inversion of block-tridiagonal and banded matrices to solve the PDE-based system efficiently. Despite their higher computation complexity than direct computation in a sequential processor, we make it possible for the very first time to employ a divide-and-conquer algorithm, viable for parallel computation, in heat conduction analysis. Experimental results prove such possibility, suggesting that applying divide-and-conquer algorithms, higher-order finite difference schemes can achieve better simulation accuracy with even faster speed and less memory requirement than conventional methods.

Received 26 June 2009; Revised 6 December 2009; Accepted 8 December 2009

KEY WORDS: boundary condition; heat conduction; partial differential equation (PDE); finite difference scheme; truncation error (TR); Crank–Nicolson (CN)

1. INTRODUCTION

Thermal effects on the reliability and performance of VLSI design have attracted increasing attention due to the rapid increase of power and package densities in recent years. Thermal effects are of great importance to a variety of problems including circuit design [1, 2], thermodynamical analysis [3], electrothermal stability [4] and thermal noise analysis [5, 6], etc. More and more efforts have been put into the development of numerical methods for the solution of heat conduction problems. For a homogeneous isotropic solid whose thermal conductivity is independent of the temperature, the heat conduction is described by

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}-\frac{1}{\kappa}\frac{\partial T}{\partial t}=0, \tag{1}$$

with T being the temperature in ^{o}C, $\kappa = K/\rho c$ is the diffusivity of the substance, where K is the thermal conductivity of the substance and ρ and c are the density and specific heat of the

*Correspondence to: Yijiang Shen, Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong.
†E-mail: yjshen@eee.hku.hk

substance, respectively. Although the thermal conductivity K is generally a function of temperature and position, due to its rather small variation in the conductor, K is often assumed constant when analyzing VLSI interconnect lines [7]. It is clear from [8] that the analytical solutions available for (1) are practically confined to linear problems on regions of simple shapes. If bodies of complicated shapes or non-linear boundary conditions have to be considered, one must resort to numerical methods. Some finite difference methods including explicit and implicit methods with first-order accuracy in time and second-order accuracy in space, i.e. truncation error TR = $O[(\triangle t), (\triangle x)^2, (\triangle y)^2, (\triangle z)^2]$, and the Crank–Nicolson (CN) method with the best accuracy TR = $O[(\triangle t)^2, (\triangle x)^2, (\triangle y)^2, (\triangle z)^2]$, can be found in [8, 9]. Since these methods are computationally intensive for 2-D or 3-D problems, Peaceman and Rachford [10] and Douglas and Gunn [11] developed the alternating direction implicit (ADI) method to overcome this difficulty. Basically, the ADI method is a process to reduce the 2-D or 3-D problems to a succession of two or three 1-D problems. Numerous efforts have been made to perform thermal analysis in VLSI circuits [12]. The finite difference method with equivalent RC model has been presented in [13, 14]. These methods suffer from superlinear runtime and memory usage for large-scale problems due to the complexity of solving large-scale matrix equations. Wang and Chen incorporated the finite difference methods with the ADI method and developed a linear-time chip-level dynamic thermal simulation algorithm, and applied the methods to the thermal simulation of 2-D and 3-D RC models [15, 16]. Because implicit and CN finite difference methods are unconditionally stable (whereas the explicit methods are not), they are often used in thermal simulation. However, due to computational complexity and numerical instability of the inversion of large-scale matrices, which are often banded, symmetrical and sparse in nature in thermal simulations, and the solution of linear equations that follows, existing finite difference methods are confined to at most second-order accuracy in both space and time.

Stability and memory problems arise when solving large-scale linear equation using direct computation such as LU decomposition. In [17, 18], block cyclic reduction is devised to solve block-tridiagonal systems. Despite its higher computation complexity, it can be applied in a divide-and-conquer approach with parallel computation. Another alternative is the direct inversion of block-tridiagonal and block-banded matrices, for which many elegant theoretical results have been proposed. The concept of semiseparable matrices [19] has been introduced and it has been shown that the inverse of a semiseparable matrix can be compactly represented by two sequences u_i and v_i [20, 21], or when the matrix is block-tridiagonal or banded, matrices U_i and V_i [22, 23]. However, even for modest size problems, the computation of U_i and V_i becomes unstable. The 'ratio' approach [24, 25] eliminated this instability for tridiagonal case, while for the general block-tridiagonal case, when a block factorization of the original block-tridiagonal matrix is used to construct a block Cholesky decomposition of its inverse, the construction of entries of the inverse is unstable despite the stable computation of the matrix ratios. A new representation [26] used n-1 Given's transformations and a vector of length n to represent a semiseparable matrix of dimension n with numerically stable computation. However, calculating the Given's transformation is computationally expensive. Reference [27] offers a compact representation for the inverses of block-tridiagonal and banded matrices that can be constructed in a computationally efficient and numerically stable manner. Two sequences, D_i which represents the diagonal blocks of the inverse and S_i the ratios of sequential elements of the sequence V_i, are used to construct the inverse of large-scale matrices in a numerically stable way. In [28], this representation is applied in a divide-and-conquer algorithm for both memory and computation benefits.

In this paper, we first conduct a revision of conventional finite difference methods. The revision is not meant to be exclusive, yet it includes explicit, implicit and CN methods which are most commonly employed to solve heat conduction problems. Then, we describe the divide-and-conquer scheme with fourth-order spatial accuracy. The proposed scheme possesses favorable features including:

(1) It has higher spatial accuracy and is more accurate than conventional finite difference schemes.
(2) It allows the use of direct computation, block cyclic reduction method or D_i, S_i method to solve block-tridiagonal systems.

DOI: 10.1002/cta

(3) Block cyclic reduction and D_i, S_i method are linearly proportional to the scale of the block-tridiagonal systems.
(4) Although block cyclic reduction and D_i, S_i method are computationally more expensive than direct computation, divide-and-conquer algorithm emerges for the very first time in the literature to solve large-scale block-tridiagonal systems.
(5) Thermal ADI is applied to the proposed schemes in 2-D and 3-D cases.

The remainder of the article is organized as follows. Section 2 reviews conventional finite difference methods in solving the heat conduction problem. An unconditionally stable finite difference method with fourth-order accuracy in space and second-order in time, together with a fourth-order accurate approximation of convection boundary condition, is proposed. In Section 3, we briefly describe block cyclic reduction and the inversion of large-scale matrices with D_i, S_i method and use them in the computation of the finite difference time domain (FDTD) methods, followed by a description of divide-and-conquer algorithm to solve large-scale block-tridiagonal systems. In Section 4, experimental results regarding linear flow of heat in an infinite region and simulation of interconnect thermal profile are presented. Finally, in Section 5, some concluding remarks and future insights are drawn.

2. FINITE DIFFERENCE FORMULATION OF HEAT CONDUCTION

Finite difference plays an important role in numerical analysis. It is one of the simplest ways of approximating a differential operator, and is extensively used in solving partial differential equations (PDEs) in applications including electromagnetic applications and thermal simulations such as (1). As mentioned earlier, ADI method is designed to reduce 2-D or 3-D problems to a succession of 1-D problems. Consequently, we focus our discussion on linear heat conduction in only one dimension. The results can be readily applied to higher dimensional problems using the ADI framework. The equation to be solved is

$$\frac{1}{\kappa}\frac{\partial T}{\partial t}=\frac{\partial^2 T}{\partial x^2}, \quad -\infty<x<\infty, \tag{2}$$

with T being a prescribed function of x and t.

The finite difference method relies on discretizing a function on a grid. Equation (2), which is one dimensional, is a second-order parabolic PDE which is applicable to simplified thermal simulation cases, such as an interconnect line passing over the substrate [7]. The first step to establish a finite-difference solution method of the PDE is to discretize the continuous x coordinate into a finite number of grid points with an interval ($\triangle x$). At a time step n with time interval ($\triangle t$), the temperature $T(x,t)$ at point i can be replaced by $T(i(\triangle x), n(\triangle t))$ which will be denoted by T_i^n in the rest of the article. Based on the Taylor series approach,

$$f(x+h)=f(x)+h\frac{\partial f(x)}{\partial x}+\frac{h^2}{2}\frac{\partial^2 f(x)}{\partial x^2}+\frac{h^3}{6}\frac{\partial^3 f(x)}{\partial x^3}+\frac{h^4}{24}\frac{\partial^4 f(x)}{\partial x^4}\ldots,$$

several finite difference approximations for $\partial^2 f/\partial x^2$ are included in Table I, where FD, BD and CD stands for Forward Difference, Backward Difference and Central Difference, respectively. The next step is to consider the time discretization problem. Three time-marching schemes can be applied to the left-hand side of (2) with respect to accuracy and stability [15, 16], namely, simple explicit method, simple implicit method and CN method. From Table I, it can be seen that CD methods have better truncation error than FD or BD methods, therefore CD methods will be used in this paper for the conduction of heat in an infinite region.

DOI: 10.1002/cta

Table I. Finite difference approximations.

Derivative	Finite Difference	Type	Error
$\frac{\partial^2 f}{\partial x^2}$	$\frac{f_{i+2}-2f_{i+1}+f_i}{(\triangle x)^2}$	FD	$O[(\triangle x)]$
	$\frac{f_i-2f_{i-1}+f_{i-2}}{(\triangle x)^2}$	BD	$O[(\triangle x)]$
	$\frac{f_{i+1}-2f_i+f_{i-1}}{(\triangle x)^2}$	CD	$O[(\triangle x)^2]$
	$\frac{-f_{i+2}+16f_{i+1}-30f_i+16f_{i-1}-f_{i-2}}{12(\triangle x)^2}$	CD	$O[(\triangle x)^4]$

2.1. Simple explicit method

Simple explicit method can be dated back to [8]. Applying the CD approximation with TR$=O[(\triangle x)^2]$ to approximate $\partial^2 T/\partial x^2$ in (2), at the point $x=i(\triangle x), i=0, 1, \ldots, N$, we have

$$T_{i+1}^t-2T_i^t+T_{i-1}^t-\frac{(\triangle x)^2}{\kappa}\frac{\mathrm{d}T_i^t}{\mathrm{d}t}=0. \tag{3}$$

For numerical implementation, it is also necessary to replace the time derivative in (2) by a difference. The simplest formula for $\partial T/\partial t$ is

$$\left[\frac{\partial T_i}{\partial t}\right]_{t=n(\triangle t)}=\frac{T_i^{n+1}-T_i^n}{(\triangle t)},$$

neglecting the second-order difference. Using this in (3) gives

$$T_i^{n+1}=r_1(T_{i+1}^n+T_{i-1}^n)-(2r_1-1)T_i^n, \tag{4}$$

where $r_1=\kappa(\triangle t)/(\triangle x)^2$. It should be noted that (4) gives the values of T at time $(n+1)(\triangle t)$ immediately in terms of those at time $n(\triangle t)$, and thus is called the explicit method. To find the stability condition of this method, we can substitute a trial solution

$$T_i^n=\lambda^k \mathrm{e}^{\mathrm{j}(i\pi/P)}, \tag{5}$$

where P is any nonzero integer, into (4) we obtain

$$\lambda=1-2r_1(1-\cos(\pi/P)). \tag{6}$$

Since we must have $|\lambda|\leqslant 1$ for nondivergence, the stability condition is

$$r_1=\kappa(\triangle t)/(\triangle x)^2\leqslant\tfrac{1}{2}. \tag{7}$$

This implies that as we decrease the spatial interval $(\triangle x)$ for better accuracy, we must also decrease the time step $(\triangle t)$ at the cost of more computation in order not to lose stability.

Considering now the possibility of finding difference schemes that represent the differential equation more accurately, it is natural to use the CD approximation with TR$=O[(\triangle x)^4]$ instead of the one with TR$=O[(\triangle x)^2]$. Applying the same formula for $\partial T/\partial t$ to (2), we have

$$T_i^{n+1}=r_2(-T_{i+2}^n+16T_{i+1}^n+16T_{i-1}^n-T_{i-2}^n)+(1-30r_2)T_i^n, \tag{8}$$

in which $r_2=\kappa(\triangle t)/(12(\triangle x)^2)=r_1/12$. Equation (8) also gives the value of T at time $(n+1)(\triangle t)$ explicitly. Similarly, to find the stability condition of (8), we substitute the same trial solution in (5) into (8) to get

$$\lambda=1-4r_2((\cos(\pi/p)-4)^2-9). \tag{9}$$

Since $0\leqslant(\cos(\pi/p)-4)^2-9\leqslant 16$ with $-1\leqslant\cos(\pi/p)\leqslant 1$, and we must have $|\lambda|\leqslant 1$ for nondivergence, the stability condition turns out to be

$$r_2\leqslant\tfrac{1}{64}\quad\text{or}\quad r_1\leqslant\tfrac{3}{16}. \tag{10}$$

The TR of this method is TR$=O[(\triangle t),(\triangle x)^4]$, thus this method is denoted as a 1–4 scheme.

DOI: 10.1002/cta

2.2. Simple implicit method

Here we consider implicit finite difference method, which can be obtained by applying the simple implicit update on the approximation of $\partial^2 T/\partial x^2$ in (2). If CD approximation with TR $= O(\triangle x)^4$ and the same formula for $\partial T/\partial t$ are applied to (2), we have

$$r_2 T_{i+2}^{n+1} - 16r_2 T_{i+1}^{n+1} + (30r_2+1)T_i^{n+1} - 16r_2 T_{i-1}^{n+1} + r_2 T_{i-2}^{n+1} = T_i^n, \tag{11}$$

where $r_2 = \kappa(\triangle t)/(12(\triangle x)^2) = r_1/12$. The value of T at time $(n+1)(\triangle t)$ is not explicitly given in (11), and the computation of T_i^{n+1} requires the inversion of a symmetric banded matrix with a bandwidth of 5. We will discuss the computation of T_i^{n+1} in Section 3.1. Now, to find the stability condition of (11), the trial solution in (5) is used in (11) to get

$$\lambda = \frac{1}{1+4r_2((\cos(\pi/p)-4)^2-9)}. \tag{12}$$

From the discussion in Section 2.1, we can see that $|\lambda| \leqslant 1$. Therefore, this implicit finite difference method is unconditionally stable with the best accuracy TR $= O[(\triangle t), (\triangle x)^4]$ which is also a 1–4 scheme.

2.3. A 2–4 finite difference scheme

We can construct an implicit CN-type scheme for the solution of (2) by replacing the integral in the relation

$$T^{n+1} = T^n + \int_T^{T+\triangle t} \frac{\partial T}{\partial t}\,\mathrm{d}t \tag{13}$$

with a trapezoidal rule, i.e.

$$T^{n+1} = T^n + \frac{\triangle t}{2}\left(\frac{\partial T^{n+1}}{\partial t} + \frac{\partial T^n}{\partial t}\right) + O((\triangle t)^3), \tag{14}$$

which can be interpreted as dealing with the time marching by taking the average of simple explicit and implicit methods in which the scheme takes the difference approximation of $\partial^2 T/\partial x^2$ and $\partial T/\partial t$ at the same time point, and takes the average of the CD approximations of $\partial^2 T/\partial x^2$ at the points n and $n+1$. If the CD approximation of $\partial^2 T/\partial x^2$ with TR $= O[(\triangle x)^4]$ is applied to this scheme, we have

$$\begin{aligned}&\frac{1}{2}\frac{-T_{i+2}^n + 16T_{i+1}^n - 30T_i^n + 16T_{i-1}^n - T_{i-2}^n}{12(\triangle x)^2}\\ &\quad + \frac{1}{2}\frac{-T_{i+2}^{n+1} + 16T_{i+1}^{n+1} - 30T_i^{n+1} + 16T_{i-1}^{n+1} - T_{i-2}^{n+1}}{12(\triangle x)^2} - \frac{1}{\kappa}\frac{T_i^{n+1} - T_i^n}{(\triangle t)} = 0,\end{aligned} \tag{15}$$

Further relaxation leads to

$$\begin{aligned}&r_3 T_{i+2}^{n+1} - 16r_3 T_{i+1}^{n+1} + (30r_3+1)T_i^{n+1} - 16r_3 T_{i-1}^{n+1} + r_3 T_{i-2}^{n+1}\\ &\quad = -r_3 T_{i+2}^n + 16r_3 T_{i+1}^n - (30r_3-1)T_i^n + 16r_3 T_{i-1}^n - r_3 T_{i-2}^n,\end{aligned} \tag{16}$$

where $r_3 = \kappa(\triangle t)/(24(\triangle x)^2) = r_1/24$. Equation (16) does not give the solution of T_i^{n+1} explicitly which requires the inversion of a matrix with the same structure as that in Section 2.2. Again to find the stability condition of (16), the trial solution in (5) is applied to (16) to get

$$\lambda = \frac{1-4r_3((\cos(\pi/P)-4)^2-9)}{1+4r_3((\cos(\pi/P)-4)^2-9)}, \quad |\lambda| \leqslant 1. \tag{17}$$

DOI: 10.1002/cta

Therefore, the proposed 2–4 scheme is unconditionally stable with the best accuracy TR = $O[(\triangle t)^2, (\triangle x)^4]$. In [15, 16], the CN method with TR = $O[(\triangle t)^2, (\triangle x)^2]$ is also unconditionally stable, which can be proved similarly, and requires the solution of a set of equations which can be expressed as

$$r_4 T_{i+1}^{n+1} - (2r_4+1)T_i^{n+1} + r_4 T_{i-1}^{n+1} = -r_4 T_{i+1}^n + (2r_4-1)T_i^n - r_4 T_{i-1}^n, \tag{18}$$

where $r_4 = \kappa(\triangle t)/(2(\triangle x)^2) = r_1/2$.

2.4. Boundary conditions

The heat transfer from the boundary surface to the ambient by convection can be expressed as

$$\kappa \frac{\partial T(x,t)}{\partial n_i} + h_i T(x,t) = h_i T_\infty, \tag{19}$$

where T_∞ is the ambient temperature, and h_i is the equivalent heat-transfer coefficient, and $\partial/\partial n_i$ is the differentiation along the outward direction normal to the boundary surface. Applying the boundary condition to the end points at $i(\triangle x)$, where $i=0$ and $i=N$, yields

$$\begin{aligned} -\kappa \frac{\partial T}{\partial x} + h_{0x} T &= h_{0x} T_\infty \quad \text{at } i=0 \quad \text{and} \\ -\kappa \frac{\partial T}{\partial x} + h_{Ix} T &= h_{Ix} T_\infty \quad \text{at } i=N \end{aligned} \tag{20}$$

where h_{0x} and h_{Ix} are the equivalent heat-transfer coefficients on the boundary $i=0$ and $i=N$, respectively. Take boundary $i=0$ as an example, according to (16), two virtual points T_{-1}, T_{-2} are needed. Our previous work in [29] used a second-order approximation of the convection boundary condition to simulate the interconnect temperature profile. However, a corresponding fourth-order accurate approximation of the boundary condition will ensure the fourth-order spatial accuracy since the overall accuracy of the proposed scheme could be degraded by a less accurate approximation of the virtual points. Using Taylor series expansion, finite difference approximations of $\partial f/\partial x$

$$\frac{f(x+3h) - 6f(x+2h) + 18f(x+h) - 10f(x) - 3f(x-h)}{12h}$$

and

$$\frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h}$$

have fourth-order accuracy, and the fourth-order accurate approximation of T_{-1}, T_{-2} can be derived as

$$\begin{aligned} T_{-1} &= \frac{T_3 - 6T_2 + 18T_1 - 10T_0 - r_{0x}(T_0 - T_\infty)}{3} \\ T_{-2} &= \frac{8T_3 - 45T_2 + 120T_1 - 80T_0 - 5r_{0x}(T_0 - T_\infty)}{3}, \end{aligned} \tag{21}$$

where $r_{0x} = 12(\triangle x)h_{0x}/\kappa$. Replacing the virtual points occurring on the boundary points in the difference equations with (21) at $i=0$ and $i=1$, we have the expression

$$\begin{aligned} &-\frac{8}{3}r_3 T_3^{n+1} + 18r_3 T_2^{n+1} - 72r_3 T_1^{n+1} + \left(1 + \left(\frac{170}{3} + \frac{11}{3}r_{0x}\right)r_3\right) T_0^{n+1} - \frac{11}{3}r_{0x} r_3 T_\infty \\ &= \frac{8}{3}r_3 T_3^n - 18r_3 T_2^n + 72r_3 T_1^n + \left(1 + \left(-\frac{170}{3} - \frac{11}{3}r_{0x}\right)r_3\right) T_0^n + \frac{11}{3}r_{0x} r_3 T_\infty, \quad i=0, \\ &= \frac{4}{3}r_3 T_3^{n+1} - 18r_3 T_2^{n+1} + (36r_3+1)T_1^{n+1} + \left(-\frac{58}{3} - \frac{1}{3}r_{0x}\right) r_3 T_0^{n+1} + \frac{1}{3}r_{0x} r_3 T_\infty \\ &= -\frac{4}{3}r_3 T_3^n + 18r_3 T_2^n + (-36r_3+1)T_1^n + \left(\frac{58}{3} + \frac{1}{3}r_{0x}\right) r_3 T_0^n - \frac{1}{3}r_{0x} r_3 T_\infty, \quad i=1. \end{aligned} \tag{22}$$

DOI: 10.1002/cta

Boundary condition expressions at $i=N-1$ and $i=N$ can be obtained similarly. Combining (16) and (22), the temperature profile at time slot $n+1$ can be computed by solving a block-tridiagonal system of equations $MX^{n+1}=Y^n$, where M is a banded matrix with a bandwidth of 5, $X^{n+1}=[T_1^{n+1}, T_2^{n+1}, \ldots, T_N^{n+1}]^{\mathrm{T}}$ is the node temperature at time slot $n+1$, and Y^n is the $N\times 1$ vector with elements corresponding to the right-hand side of (16) and (22).

2.5. Thermal alternating-direction-implicit method

In [11], Douglas and Gunn developed a general ADI scheme that is unconditionally stable and retains accuracy when applied to the 3-D problem. If δ_x^2, δ_y^2, δ_z^2 represent different aforementioned approximations to $\partial^2 T/\partial x^2$ in x, y and z direction, respectively, the 3-D heat conduction problem case can be reduced to a succession of 3 1-D problems and each of these 1-D problem can be solved using the proposed finite difference schemes. The ADI method can be described as

$$\begin{aligned}
\text{Step 1} &: \left(1-\frac{1}{2}r\delta_x^2\right)\Delta T^{n+(1/3)}=(r\delta_x^2+r\delta_y^2+r\delta_z^2)T^n\\
\text{Step 2} &: \left(1-\frac{1}{2}r\delta_y^2\right)\Delta T^{n+(2/3)}=\Delta T^{n+(1/3)}\\
\text{Step 3} &: \left(1-\frac{1}{2}r\delta_z^2\right)\Delta T^{n+1}=\Delta T^{n+(2/3)},
\end{aligned} \tag{23}$$

where

$$\begin{aligned}
\Delta T^{n+(1/3)} &\equiv T^{n+(1/3)}-T^n\\
\Delta T^{n+(2/3)} &\equiv T^{n+(2/3)}-T^n\\
\Delta T^{n+1} &\equiv T^{n+1}-T^n.
\end{aligned} \tag{24}$$

Thus, in step 1, the 1-D problem on x coordinate can be solved to produce $\Delta T^{n+(1/3)}$ as the input of the 1-D problem on y coordinate, then similarly in step2, the problem on y coordinate can be solved to produce $\Delta T^{n+(2/3)}$ as the input of the 1-D problem on z coordinate, finally, the computation of the problem on z coordinate in step 3 concludes the succession and gives the final results. By analogy, the 2-D ADI method can be similarly deduced which includes only two steps instead of 3 in the 3-D case.

3. SOLVING BLOCK-TRIDIAGONAL SYSTEMS

From Section 2, it can be seen that the simple explicit difference methods give the solution of T_i^{n+1} directly, while implicit method and the proposed 2–4 scheme require the solution of a symmetric banded system with a bandwidth of 5 with the structure

$$M=\begin{bmatrix}
30r+1 & -16r & r & 0 & 0 & \cdot & 0\\
-16r & 30r+1 & -16r & r & 0 & \cdot & 0\\
r & -16r & 30r+1 & -16r & r & \cdot & 0\\
 & \ddots & \ddots & \ddots & \ddots & \ddots & \\
0 & \cdot & r & -16r & 30r+1 & -16r & r\\
0 & \cdot & 0 & r & -16r & 30r+1 & -16r\\
0 & \cdot & 0 & 0 & r & -16r & 30r+1
\end{bmatrix}$$

DOI: 10.1002/cta

where $r = r_2$ for (11) and r_3 for (16) when no boundary condition is incorporated. Obviously, the symmetric banded system can be computed directly using LU decomposition, or conjugate gradient method as was conducted in [30]. However, when the system scale grows, stability and memory problems arise. Alternative algorithms which can be applied in divide-and-conquer schemes are considered as replacement of direct computation.

3.1. Inversion of banded matrices

In [27], the authors proposed a numerically stable algorithm, defining D_i and S_i sequences for the inversion of block-tridiagonal and banded matrices. Such a symmetric block-tridiagonal matrix A can be transformed into the form

$$A = \begin{pmatrix} A_1 & -B_1 & & & \\ -B_1^T & A_2 & -B_2 & & \\ & \ddots & \ddots & \ddots & \\ & & -B_{N_y-2}^T & A_{N_y-1} & -B_{N_y-1} \\ & & & -B_{N_y-1}^T & A_{N_y} \end{pmatrix}, \tag{25}$$

where each $A_i, B_i \in \mathbb{C}^{N_x \times N_y}$, thus $A \in \mathbb{C}^{N_y N_x \times N_y N_x}$ with N_y diagonal blocks of size N_x each. In our consideration, M is a banded matrix of bandwidth 5 which makes $N_x = 2$ and $N_y = N/N_x$, and A_i, B_i are block matrices of size 2×2 with

$$A_i = \begin{pmatrix} 30r+1 & -16r \\ -16r & 30r+1 \end{pmatrix}, \quad B_i = \begin{pmatrix} -r & 0 \\ 16r & -r \end{pmatrix}. \tag{26}$$

S_i sequence can be computed in $O(N_y N_x^3)$ operations using the following numerically stable recursion:

$$\begin{aligned} S_{N_y-1} &= B_{N_y-1} A_{N_y}^{-1}, \\ S_i &= B_i (A_{i+1} - S_{i+1} B_{i+1}^T)^{-1}, \quad i = N_y - 2, \ldots, 1. \end{aligned} \tag{27}$$

It is readily verified that the diagonal blocks of A^{-1}, denoted by sequence D_i, are given by the recursion

$$\begin{aligned} &D_1 = (A_1 - B_1 S_1^T)^{-1}, \quad D_{N_y} = A_{N_y}^{-1} (I + B_{N_y-1}^T D_{N_y-1} S_{N_y-1}), \\ &D_i = (A_i - B_i S_i^T)^{-1} (I + B_{i-1}^T D_{i-1} S_{i-1}), \quad i = 2, \ldots, N_y - 1, \end{aligned} \tag{28}$$

while the remaining block entries can be computed in a numerically stable way as

$$A_{ij}^{-1} = D_i S_i S_{i+1} \ldots S_{j-1}, \quad j > i. \tag{29}$$

Since A^{-1} is symmetric, combining (28) and (29), A^{-1} is computed. A fast computation of the product of the inverse of a block-tridiagonal matrix A and a vector x is also proposed in [27] with a computation time of $O(N_y N_x^2)$ as compared with $O(N_y^2 N_x^2)$ otherwise.

DOI: 10.1002/cta

3.2. Non-symmetric matrix arising from boundary conditions

It is noted that when applying boundary conditions (22), M becomes

$$\begin{bmatrix} \left(1+\left(\frac{170}{3}+\frac{11}{3}r_{0x}\right)r\right) & -72r & 18r & -\frac{8}{3}r & 0 & \cdot & 0 \\ \left(-\frac{58}{3}-\frac{1}{3}r_{0x}\right)r & (36r+1) & -18r & \frac{4}{3}r & 0 & \cdot & 0 \\ r & -16r & 30r+1 & -16r & r & \cdot & 0 \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \\ 0 & \cdot & r & -16r & 30r+1 & -16r & r \\ 0 & \cdot & 0 & \frac{4}{3}r & -18r & (36r+1) & \left(-\frac{58}{3}-\frac{1}{3}r_{0x}\right)r \\ 0 & \cdot & 0 & -\frac{8}{3}r & 18r & -72r & \left(1+\left(\frac{170}{3}+\frac{11}{3}r_{0x}\right)r\right) \end{bmatrix},$$

which is not symmetric, and its bandwidth increases. Fortunately, a smart trick can be applied: applying Gaussian elimination to the first and last two rows of M, we can transform B_1 and B_{N_y-1} into the same structure as B_i in (26) and $A_i, i=2,\ldots,N_y-1$ the same structure as in (26). These relaxations make the structure of M similar to that in (26), except that A_1 and A_{N_y} are not symmetric. A_1 and A_{N_y} are of the form

$$A_1 = \begin{pmatrix} \left(-1-\frac{1}{6}r_{0x}\right)r-\frac{1}{18} & -\frac{1}{9} \\ \left(-12+\frac{1}{6}r_{0x}\right)r+\frac{5}{36} & 27r+\frac{37}{36} \end{pmatrix},$$

$$A_{N_y} = \begin{pmatrix} 27r+\frac{37}{36} & \left(-12+\frac{1}{6}r_{0x}\right)r+\frac{5}{36} \\ -\frac{1}{9} & \left(-1-\frac{1}{6}r_{0x}\right)r-\frac{1}{18} \end{pmatrix},$$

It is noted that for every r_{0x}, if we have the freedom to define r, there will be a corresponding r that makes matrix A symmetric. If $r_{0x}=0$, for example, defining r as $\frac{1}{48}$ makes A symmetric. However, in thermal simulations, we do not have the flexibility of defining r according to the value of r_{0x}; both r_{0x} and r are designated. Therefore, it is unlikely that M will be symmetric. To overcome this, a symmetric preconditioner P of M is constructed as

$$\begin{bmatrix} \left(-1-\frac{1}{6}r_{0x}\right)r-\frac{1}{18} & \left(-12+\frac{1}{6}r_{0x}\right)r+\frac{5}{36} & r & 0 & 0 & \cdot & 0 \\ \left(-12+\frac{1}{6}r_{0x}\right)r+\frac{5}{36} & 27r+\frac{37}{36} & -16r & r & 0 & \cdot & 0 \\ r & -16r & 30r+1 & -16r & r & \cdot & 0 \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \\ 0 & \cdot & r & -16r & 30r+1 & -16r & r \\ 0 & \cdot & 0 & r & -16r & 27r+\frac{37}{36} & \left(-12+\frac{1}{6}r_{0x}\right)r+\frac{5}{36} \\ 0 & \cdot & 0 & 0 & r & \left(-12+\frac{1}{6}r_{0x}\right)r+\frac{5}{36} & \left(-1-\frac{1}{6}r_{0x}\right)r-\frac{1}{18} \end{bmatrix},$$

DOI: 10.1002/cta

with a much smaller condition number than that of M. Then an iterative preconditioned method is applied at each time slot $n+1$ to solve the equation

$$PX^{n+1}(j+1)=QX^{n+1}(j)+Y^n, \quad j=1,\ldots,k, \tag{30}$$

where k is the number of iterations and $Q=P-M$ is a very sparse matrix of size $N\times N$, with $Q(1,2)$, $Q(N,N-1)=(-12+\frac{1}{6}r_{0x})r+\frac{5}{36}+\frac{1}{9}$, and 0 otherwise. Starting the iteration at time n, using T_i^n as an initial guess for $X^{n+1}(1)$, $X^{n+1}(2)$ can be computed by first calculating the D_i, S_i sequences of P and then applying the fast algorithm in [27] to compute $X^{n+1}(2)=P^{-1}(QX^{n+1}(1)+Y^n)$. The iteration of computing $X^{n+1}(j+1)=P^{-1}(QX^{n+1}(j)+Y^n)$ continues until $X^{n+1}(j+1)$ is sufficiently close to $X^{n+1}(j)$.

3.3. Block cyclic reduction

The D_i, S_i method mentioned above is linearly proportional to the system scale. However, explicit representation of the inversion of block-tridiagonal matrix is not necessary in the solution of block-tridiagonal systems. Another candidate of divide-and-conquer algorithm without such problem is block cyclic reduction. We consider a block-tridiagonal linear system $Mx=v$, i.e.

$$E_jx_{j-1}+A_jx_j+F_jx_{j+1}=v_j, \quad j=1,\ldots,N_y, \tag{31}$$

where x_j and $v_j\in\mathbb{R}^2$, $E_j=-B_i^T$, $j=2,\ldots,N_y$, $F_j=-B_i$, $j=1,\ldots,N_y-1$ and $E_1=F_{N_y}=0$.

Consider three consecutive equations indexed $2j-1$, $2j$ and $2j+1$, in order to eliminate x_{2j-1} and x_{2j+1} in equation indexed $2j$, we multiply equation indexed $2j-1$ with $-E_{2j}(D_{2j-1})^{-1}$, equation indexed $2j+1$ with $-F_{2j}(D_{2j+1})^{-1}$ and add these to equation $2j$. The result is a new equation

$$E_{2j}^{(1)}x_{2j-2}+A_{2j}^{(1)}x_{2j}+F_{2j}^{(1)}x_{2j+2}=v_{2j}^{(1)}, \tag{32}$$

where

$$\begin{aligned}
E_{2j}^{(1)}&=-E_{2j}(D_{2j-1})^{-1}E_{2j-1},\\
A_{2j}^{(1)}&=A_{2j}-E_{2j}(D_{2j-1})^{-1}F_{2j-1}-F_{2j}(D_{2j+1})^{-1}F_{2j+1},\\
F_{2j}^{(1)}&=-F_{2j}(D_{2j+1})^{-1}F_{2j+1},\\
v_{2j}^{(1)}&=v_{2j}-E_{2j}(D_{2j-1})^{-1}v_{2j-1}-F_{2j}(D_{2j+1})^{-1}v_{2j+1}.
\end{aligned} \tag{33}$$

The elimination can of course only be performed if the diagonal block matrices are non-singular, which is true for M. Eliminating in this way the odd indexed unknowns, we obtain a block-tridiagonal reduced system for the even unknowns. The new system is half the size of the original system. We can iterate the procedure and eliminate again the odd unknowns in the new system. Suppose $N_y=2^k-1$, after $k-1$ iterations of elimination, we obtain a block system equation with only one unknown

$$A_{2^{k-1}}^{(k-1)}x_{2^{k-1}}=v_{2^{k-1}}^{(k-1)}, \tag{34}$$

and the rest of the unknowns can be computed by back substituting $x_{2^{k-1}}$ into upper level tridiagonal systems iteratively. It should be duly noted that both the D_i, S_i sequence method and block cyclic reduction can be applied in a divide-and-conquer fashion and thus parallel computation is possible for both memory and computation benefits.

DOI: 10.1002/cta

3.4. Divide-and-conquer algorithm

If we reorder (31) for the first step of block cyclic reduction by separating the unknowns with odd and even indices, we obtain:

$$\begin{pmatrix} D_1 & F \\ G & D_2 \end{pmatrix} \begin{pmatrix} x_o \\ x_e \end{pmatrix} = \begin{pmatrix} v_o \\ v_e \end{pmatrix}, \tag{35}$$

where D_1 and D_2 are diagonal block-matrices and F and G are block-bidiagonal. Eliminating the unknowns with odd indices $x_o = (x_1, x_3, \ldots)^{\mathrm{T}}$ means computing the reduced system of equations for the unknowns with even indices $x_e = (x_2, x_4, \ldots)^{\mathrm{T}}$ by forming the Schur complement:

$$(D_2 - GD_1^{-1}F)x_e = v_e - GD_1^{-1}v_o. \tag{36}$$

Such Schur complement can be formed by computing (3.3). By similarly eliminating the unknowns with even indices, the equivalent system is formed:

$$\begin{pmatrix} D_1 - FD_2^{-1}G & 0 \\ 0 & D_2 - GD_1^{-1}F \end{pmatrix} \begin{pmatrix} x_o \\ x_e \end{pmatrix} = \begin{pmatrix} v_o - FD_2^{-1}v_e \\ v_e - GD_1^{-1}v_o \end{pmatrix}, \tag{37}$$

which splits (35) into two independent block-tridiagonal systems for the two sets of unknowns x_o and x_e with half the size of the original system. They can be solved independently and this process can be seen as the first step of a divide-and-conquer algorithm and can be split iteratively into smaller subsystems to be solved on different processors. The application of D_i, S_i method in divide-and-conquer algorithm can also be found in [28], which suggests that although D_i, S_i method and block cyclic reduction method are more computationally expensive than direct computation, they are still worthy of investigation in divide-and-conquer algorithms when solving large-scale block-tridiagonal systems.

Incorporating the divide-and-conquer algorithm into the ADI method, we can solve 2-D and 3-D heat conduction problem efficiently. We first use the proposed finite difference scheme and boundary condition to formulate the problem, then, the 2-D or 3-D problem is reduced to 2 or 3 1-D problems using the ADI method and these 1-D problems are computed using the divide-and-conquer algorithm for the benefit of both computation time and memory requirement to give the final results.

4. EXPERIMENTAL RESULTS

In this section, some experimental results regarding linear heat conduction in one dimension described by (2) will be presented. A simple method of investigating the stability and accuracy of numerical solutions is by studying the case $T_0^0 = 1$, $T_i^0 = 0$, $i \neq 0$, which may be regarded as showing how the effect of a unit error is propagated through the system. The exact solution for this case is

$$T_i^n = \frac{1}{2(\pi r n)^{\frac{1}{2}}} \mathrm{e}^{-i^2/4rn}, \tag{38}$$

in which $r = r_1$ in (4). From Table II, in which $r = 0.25$ and all four methods approach the exact solution, it is noted that the proposed 2–4 scheme method approaches with the best accuracy. In Table III where $r = 0.5$, explicit method $\mathrm{TR} = O[(\triangle t), (\triangle x)^2]$ is at the edge of oscillation, and explicit method $\mathrm{TR} = O[(\triangle t), (\triangle x)^4]$ starts to oscillate, which is consistent with the stability condition in (10), and the proposed 2–4 scheme still approaches the exact solution with the best accuracy. In Table IV where $r = 0.6$, the explicit methods start to oscillate, while the implicit method, the CN 2–2 scheme and the proposed 2–4 scheme approach the exact solution. In these experiments, when r increases, the explicit methods start to diverge from the exact solution while the proposed 2–4 scheme approaches the exact solution with a better accuracy than the implicit method

DOI: 10.1002/cta

Table II. Experimental results $r=0.25, n=10$ (exact solution is computed using (38)).

i	0	1	2	3	4
Exact solution	0.1784	0.1614	0.1196	0.0725	0.0360
EM $O[(\triangle t),(\triangle x)^2]$	0.1762	0.1602	0.1201	0.0739	0.0370
EM $O[(\triangle t),(\triangle x)^4]$	0.1758	0.1599	0.1202	0.0742	0.0372
IM $O[(\triangle t),(\triangle x)^4]$	0.1864	0.1652	0.1169	0.0682	0.0337
CN $O[(\triangle t)^2,(\triangle x)^2]$	0.1832	0.1639	0.1182	0.0698	0.0344
Proposed 2–4 scheme	0.1787	0.1615	0.1195	0.0724	0.0360

Table III. Experimental results $r=0.5, n=5$ (exact solution is computed using (38)).

i	0	1	2	3	4
Exact solution	0.1784	0.1614	0.1196	0.0725	0.0360
EM $O[(\triangle t),(\triangle x)^2]$	0	0.3125	0	0.1563	0
EM $O[(\triangle t),(\triangle x)^4]$	−0.1146	0.4167	−0.0807	0.2083	−0.0260
IM $O[(\triangle t),(\triangle x)^4]$	0.1950	0.1680	0.1139	0.0646	0.0321
CN $O[(\triangle t)^2,(\triangle x)^2]$	0.1822	0.1636	0.1187	0.0702	0.0344
Proposed 2–4 scheme	0.1778	0.1612	0.1199	0.0729	0.0361

Table IV. Experimental results $r=0.6$, $n=5$ (exact solution is computed using (38)).

i	0	1	2	3	4
Exact solution	0.1629	0.1498	0.1167	0.0769	0.0429
EM $O[(\triangle t),(\triangle x)^2]$	−0.8355	1.0416	−0.5472	0.4752	−0.1296
EM $O[(\triangle t),(\triangle x)^4]$	−1.3966	1.5498	−0.9248	0.7008	−0.2333
IM $O[(\triangle t),(\triangle x)^4]$	0.1776	0.1570	0.1130	0.0695	0.0379
CN $O[(\triangle t)^2,(\triangle x)^2]$	0.1655	0.1516	0.1165	0.0755	0.0414
Proposed 2–4 scheme	0.1621	0.1496	0.1169	0.0775	0.0431

and the CN 2–2 scheme. From Table V, where $k=7$ and the number of nodes is $2\times(2^k-1)=254$, it is noted that explicit method and CN 2–2 scheme are much faster than implicit method and the proposed scheme. The computation complexity of D_i, S_i method is $O(N_y N_x^2)$, or $O(2N)$ where $N=N_x N_y$ is the scale of M and $N_x=2$. Cyclic reduction requires about 2.7 times more operations than Gaussian elimination and thus has a redundancy of 2.7 [17]. The computation time of using D_i, S_i method and block cyclic reduction are about 2 and 4 times of that using direct computation (sparse LU decomposition), respectively, which is consistent with their computation complexity. In Table VI, the CPU time of computing block-tridiagonal system using the divide-and-conquer algorithm shows that when the number of split subsystems increase to 128 and 256, the divide-and-conquer algorithm is faster than direct computation and when the scale of M reaches 7168, direct computation fails because of insufficient memory, the divide-and-conquer algorithm still solves the block-tridiagonal system with high speed. Consequently, despite the transportation overhead to different processors, experimental results in Table VI suggest that the divide-and-conquer algorithm actually prevails direct computation in both speed and memory requirement when the scale of the block-tridiagonal system is very large.

Figure 1 shows an interconnect line of length l, width w and thickness t_m and thermal conductivity k_m that passes over the Silicon substrate with an insulator of thickness t_{ins} and an insulator thermal conductivity k_{ins}. Although the thermal conductivity k_m is generally a function of temperature and position, due to its rather small variations in conduction, k_m is often assumed to be constant when analyzing VLSI interconnect lines. In addition, the four sidewalls and the top surface of the chip containing the interconnect lines are assumed to be completely insulated which is generally

DOI: 10.1002/cta

Table V. CPU time $r=0.25, n=1000$.

				Proposed 2–4 scheme		
	EM $O[(\triangle t),(\triangle x)^4]$	IM $O[(\triangle t),(\triangle x)^4]$	CN $O[(\triangle t)^2,(\triangle x)^2]$	Direct computation	D_i, S_i method	Block cyclic reduction
CPU time (s)	0.0232	2.6976	0.0401	2.2938	5.5738	9.8529

Table VI. CPU Time (s): N is the scale of M, m is the number of split subsystems.

N, m	$N=448$, $m=32$	$N=896$, $m=164$	$N=1792$, $m=128$	$N=3584$, $m=256$	$N=7168$, $m=1024$	$N=14336$, $m=2048$
Direct computation	0.0003	0.0005	0.0075	0.0856	Out of memory	Out of memory
Divide and conquer	0.0012	0.0018	0.0024	0.0158	0.0222	0.0243

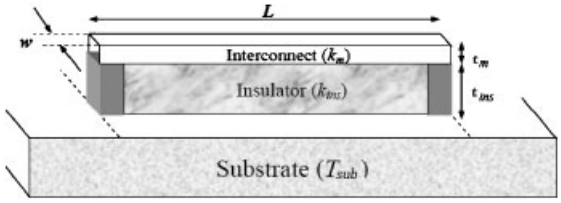

Figure 1. An interconnect line passing over the substrate, separated by an insulation layer.

valid [7]. This means that the interconnect line can only exchange heat with the external environment through the two vias at its two ends connected to the substrate. Also, for a global interconnect line whose length is much larger than its width and thickness, a 1-D heat diffusion equation along the length of the interconnect line is often employed. Based on these simplifying assumptions, (1) is reduced to

$$\frac{\partial^2 T}{\partial x^2}-\frac{1}{\kappa}\frac{\partial T}{\partial t}=-\frac{Q^*}{k_m}, \tag{39}$$

with T being a prescribed function of x and t. The effective volumetric heat generation Q^* is defined as the power dissipation minus the heat flow from the interconnect to the substrate per unit volume in [7, 12], and (21) can be further reduced as

$$\frac{\partial^2 T}{\partial x^2}-\frac{1}{\kappa}\frac{\partial T}{\partial t}=\lambda_1 T-\lambda_2 T_{\text{chip}}-\theta,$$

$$\text{in which } \lambda_1=\frac{1}{k_m}\left(\frac{k_{\text{ins}}}{t_m t_{\text{ins}}}-\frac{\rho_i \beta I_{\text{rms}}^2}{w^2 t_m^2}\right),$$

$$\lambda_2=\frac{1}{k_m}\frac{k_{\text{ins}}}{t_m t_{\text{ins}}} \quad \text{and} \quad \theta=\frac{1}{k_m}\frac{\rho_i I_{\text{rms}}^2}{w^2 t_m^2}, \tag{40}$$

where ρ_i is the electrical resistivity of the interconnect at the reference temperature and β is the temperature coefficient of resistance.

Assuming a uniform substrate temperature of $T_{\text{chip}}=100^oC$, the interconnect temperature profile after reaching steady state is shown in Figure 2(a), in which r is set to be $\frac{1}{4}$. It can be seen that interconnect temperature can reach as high as 140°C. Such a temperature can significantly increase the interconnect resistance, which in turn would increase the signal propagation delay in the interconnect line. In Figure 2(b), the transient temperatures of node 10 using the proposed 2–4 scheme and CN 2–2 scheme are presented, showing a transient temperature difference. In this

DOI: 10.1002/cta

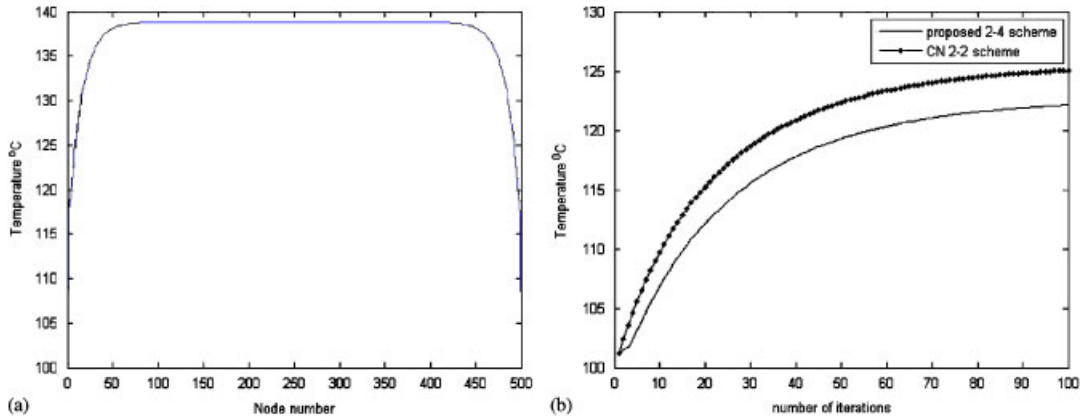

Figure 2. (a) Interconnect temperature profile reaching steady state and (b) transient temperature of node 10 using the proposed 2–4 scheme and CN 2–2 scheme.

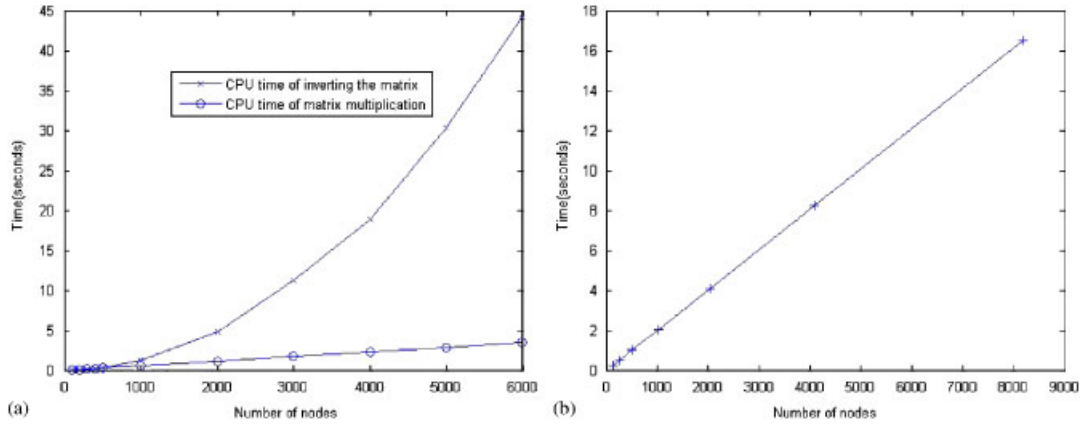

Figure 3. (a) The CPU time of inverting the matrix and matrix multiplication using D_i, S_i method and (b) the CPU time of block cyclic reduction method.

experiment, temperature change is over 40°C which does not justify the thermal diffusion constant leading to the nonlinear heat conduction problem. Nonetheless, the Kirchoffs transformation [30] can be used to remove this nonlinearity, which would still render the applicability of the proposed linear method to nonlinear heat conduction problems.

It is mentioned in Section 3.2 that in this interconnect thermal simulation, $r_{0x}=0$, value of r other than $\frac{1}{48}$ will make M not symmetric, and hence a preconditioned iterative method (30) is applied to solve equation $MX^{n+1}=Y^n$ at time $n+1$. It is verified that the iterative method converges when the spectral radius of $P^{-1}N$ is smaller than 1 which is true in our thermal simulation experiments, for example, when $r=0.25$, node number is 200, the spectral radius of $P^{-1}N$ is $0.8270<1$.

In Figure 3, the CPU time of the proposed scheme using D_i, S_i method and block cyclic reduction method is illustrated. As mentioned earlier, the banded matrix inversion and the matrix-vector multiplication can be computed in $O(N_y N_x^3)$ and $O(N_y N_x^2)$ operations, respectively. Figure 3(a) shows that in our experiments, where $N_x=2$ and the number of iterations is set to be 50, the CPU time of matrix inversion and matrix-vector multiplication are linearly proportional to the number of nodes. It should also be noted that in the experiments, the banded matrix inversion needs to be computed only once. Figure 3(b) shows that block cyclic reduction method is also linearly proportional to the number of nodes.

In Figure 4, 2-D thermal simulations of linear heat conduction in an infinite region with unit heat source in the middle of the plane and three randomly located heat sources with

DOI: 10.1002/cta

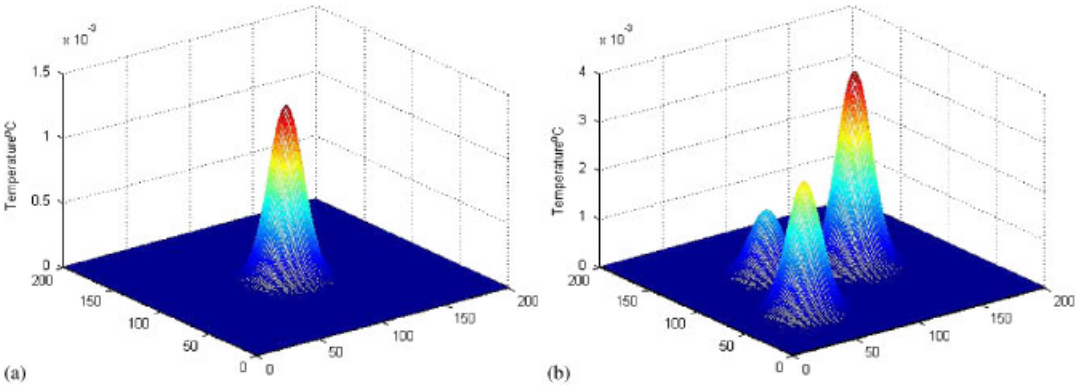

Figure 4. 2-D thermal simulation with (a) unit heat source in the middle of the plane and (b) randomly located heat sources with different inputs.

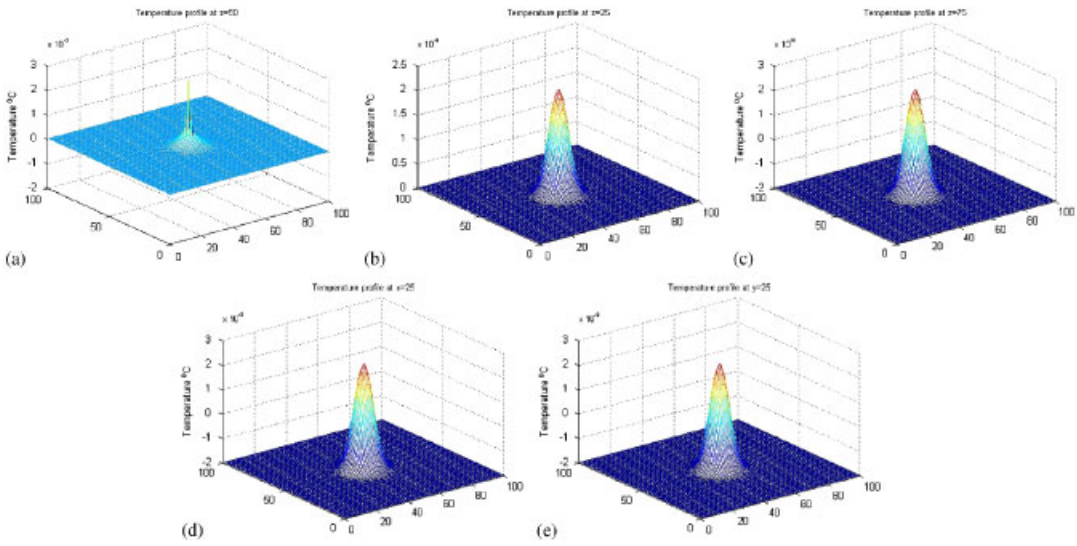

Figure 5. 3-D thermal simulation with unit heat source in the middle of the $100 \times 100 \times 100$ cube: (a) temperature profile at $z = 25$; (b) temperature profile at $z = 50$; (c) temperature profile at $z = 75$; (d) temperature profile at $x = 25$; and (e) temperature profile at $y = 25$.

different inputs show how heat is propagated through the system using the ADI method. In Figure 5, 3-D thermal simulation of linear heat conduction in an infinite region with unit heat source in the middle of the cube is demonstrated, in (b), (c), (d) and (e), identical temperature profiles at corresponding planes along three coordinate show that the heat in the center is radiated symmetrically through space which is consistent with our assumption. In both figures, the sum of temperature at all nodes remains the same during the heat conduction which proves that energy is preserved in the mesh nodes before heat conduction reaches the boundaries. Expectedly, all observations and advantages regarding the proposed 2–4 scheme over conventional schemes carry over to 2-D and 3-D scenarios.

5. CONCLUSION

We have presented in this article finite difference schemes for heat conduction analysis in circuit design and manufacturing. Explicit, implicit and CN methods using different finite difference schemes are contrasted. A fourth-order accurate approximation of the first-order spatial derivative is

DOI: 10.1002/cta

developed and is employed in the fourth-order approximation of the convection boundary condition. We have presented a fourth-order spatial-accurate finite difference scheme to better approximate the PDE solution. We have also proposed a divide-and-conquer approach to efficiently solve large-scale block-tridiagonal systems using D_i, S_i method and block cyclic reduction method. Experimental results have shown that using the proposed higher-order difference schemes in space will bring better simulation accuracy at the cost of computation time. However, the divide-and-conquer approach gives encouraging results in both speed and memory requirement. It is suggested that although D_i, S_i method and block cyclic reduction method are slower than direct computation, their application in divide-and-conquer algorithms gives them the edge when solving large-scale block-tridiagonal systems in parallel processors for both speed and memory benefits. It has also been shown that 2-D and 3-D thermal simulation can benefit from higher-order finite difference schemes with comparable computation to conventional second-order accurate methods, using ADI techniques.

REFERENCES

1. Conte G. A general method for small-signal circuit analysis considering thermal effects. *International Journal of Circuit Theory and Applications* 1978; **6**(1):41–47.
2. Mardiris VA, Karafyllidis IG. Design and simulation of modular 2^n to 1 quantum-dot cellular automata (QCA) multiplexers. *International Journal of Circuit Theory and Applications* 2009; **9999**(9999).
3. Weiss L, Mathis W. A thermodynamical approach to noise in non-linear networks. *International Journal of Circuit Theory and Applications* 1998; **26**(2):147–165.
4. Storti-Gajani G, Premoli A, Brambilla A. Electrothermal stability of uniform arrays of one-port elements. *International Journal of Circuit Theory and Applications* 2009; **37**(1):67–85.
5. Rawat TK, Parthasarathy H. On stochastic modelling of linear circuits. *International Journal of Circuit Theory and Applications* 2008; **9999**(9999).
6. Sedighi B, Bakhtiar MS. Linearity analysis in pipeline A/D converters. *International Journal of Circuit Theory and Applications* 2009; **37**(6):764–780.
7. Ajami AH, Banerjee K, Pedram M. Modeling and analysis of non-uniform substrate temperature effects in high performance VLSI. *IEEE Transactions on Computer Aided Design* 2005; **24**(6):849–861.
8. Carslaw HS, Jaeger JC. *Conduction of Heat in Solids*. Oxford University Press: London, 1948.
9. Ozisik MN. *Finite Difference*. CRC: New York, 1994.
10. Peaceman DW, Rachford HH. The numerical solution of parabolic and elliptic differential equations. *Journal of the Society for Industrial and Applied Mathematics* 1955; **3**:28–41.
11. Douglas J, Gunn JE. A general formulation of alternating direction methods-part I: parabolic and hyperbolic problems. *Numerische Mathematik* 1964; **6**:428–453.
12. Pedram M, Nazarian S. Thermal modeling, analysis and management in VLSI circuits: principles and methods. *Proceedings of IEEE*, *Special Issue on Thermal Analysis of ULSI* 2006; **94**(8):1487–1501.
13. Cheng YK, Raha P, Teng CC, Rosenbaum E, Kang SM. Illiads-t: an electrothermal timing simulator for temperature-sensitive reliability diagnosis of COSMOS VLSI chips. *IEEE Transactions on Computer-Aided Design* 1998; **17**:668–681.
14. Chen D, Li E, Rosenbaum E, Kang SM. Interconnect thermal modeling for accurate simulation of circuit timing and reliability. *IEEE Transactions on Computer-Aided Design* 2000; **19**:197–205.
15. Wang T-Y, Chen CC-P, 3-D Thermal-ADI: a linear-time chip level transient thermal simulator. *IEEE Transactions on Computer-Aided Design of Integrated Circuits And Systems* (*TCAD*) 2002; **21**(11):1343–1352.
16. Wang T-Y, Chen CC-P, Thermal-ADI: a linear-time chip-level dynamic thermal simulation algorithm based on alternating-direction-implicit (ADI) method. *IEEE Transactions on Very Large Scale Integration Systems* (*TVLSI*) 2003; **11**(4):691–700.
17. Gander W, Golub G. Cyclic reduction-history and applications. *Scientific Computing*: *Proceedings of the Workshop*, *10–12 March 1997*, Hong Kong, Springe, 1998; 73–85.
18. Golub G, Van Loan C. *Matrix Computations*. Johns Hopkins University Press: Baltimore, 1996.
19. Rozsa P. Band matrices and semi-separable matrices. *Colloquia Mathematical Societatis Janos Bolyai* 1986; **50**:229–237.
20. Asplund E. Inverses of matrices a_{ij} which satisfy a_{ij}=0 for $j>i+p$. *Mathematical Scandinavica* 1959; **7**:57–60.
21. Asplund SO. Finite boundary value problems solved by Green's matrix. *Mathematical Scandinavica* 1959; **7**:49–56.
22. Rozsa P. On the inverse of band matrices. *Integral Equations and Operator Theory* 1987; **50**:82–95.
23. Rozsa P, Bevilacqua R, Favati P, Romani F. On the inverse of block tridiagonal matrices with applications to the inverses of band matrices and block band matrices. *Operator Theory*: *Advances and Applications* 1989; **40**:447–469.
24. Concus P, Golub GH, Meurant G. Block preconditionings for the conjugate gradient method. *SIAM Journal on Scientific Computing* 1985; **6**:220–252.

DOI: 10.1002/cta

25. Meurant G. A review on the inverse of symmetric block tridiagonal and block tridiagonal matrices. *SIAM Journal on Matrix Analysis and Applications* 1992; **13**(3):707–728.
26. Vandebril R, Barel MV, Mastronardi N. A note on the representation and definition of semiseparable matrices. *Numerical Linear Algebra with Applications* 2005; **12**(8):839–858.
27. Jain J, Cauley S, Li H, Koh C, Balakrishnan V. Numerically stable algorithms for inversion of block tridiagonal and banded matrices. *Technical Report*, Electrical and Computer Engineering, Purdue University, West Lafayette, 2007.
28. Cauley S, Jain J, Koh C, Balakrishnan V. A scalable distributed method for quantum-scale device simulation. *Journal of Applied Physics* 2007; **101**(12):123–715.
29. Shen Y, Wong N, Lam EY, Interconnect thermal simulation with higher order spatial accuracy. *ser. APCCAS 2008*, 2008; 566–569.
30. Akturk A, Goldsman N, Metze G. Self-consistent modeling of heating and mosfet performance in 3-d integrated circuits. *IEEE Transactions on Electron Devices* 2005; **52**(11):2395–2403.

DOI: 10.1002/cta