Thermal-ADI—A Linear-Time Chip-Level Dynamic Thermal-Simulation Algorithm Based on Alternating-Direction-Implicit (ADI) Method

Ting-Yuan Wang and Charlie Chung-Ping Chen

***Abstract*—Due to the dramatic increase of clock frequency and integration density, power density and on-chip temperature in high-end very large scale integration (VLSI) circuits rise significantly. To ensure the timing correctness and the reliability of high-end VLSI design, efficient and accurate chip-level transient thermal simulations are of crucial importance. In this paper, we develop and present an efficient transient thermal-simulation algorithm based on the alternating-direction-implicit (ADI) method. Our algorithm, thermal-ADI, not only has *a linear run time and memory requirement,* but is also *unconditionally stable,* which ensures that time step is not limited by any stability requirement. Extensive experimental results show that our algorithm is not only orders of magnitude faster than the traditional thermal-simulation algorithms, but also highly accurate and efficient in memory usage.**

***Index Terms*—Alternating-Direction-Implicit (ADI), design automation, finite difference methods, thermal simulation, temperature.**

I. Introduction

DUE TO THE relentless push for high-speed, high-performance, and high-component density, power density and on-chip temperature in the high-end very large scale integration (VLSI) circuits rises significantly. The 1999 International Technology Roadmap for Semiconductors (ITRS) shows that the maximum power and number of metal layers will significantly increase for the future high-performance microprocessor unit (MPU). The characteristics are described in Table I. This trend shows the importance of thermal issues in VLSI design. A comprehensive analysis of the thermal effects in high-performance VLSI has been discussed recently [1]–[4]. This analysis shows that the maximum temperature on the chip increases dramatically with scaling because of the increase of interconnect levels, current density, and thermal coupling, as well as the introduction of low-k dielectrics.

High temperature not only causes timing failures for both transistors and interconnects, but also degrades chip reliability. For example, the electromigration (EM) effect for the interconnects is exponentially proportional to the temperature, not to mention electrostatic discharge (ESD) or other effects. For the next-generation process, the low dielectric constant (low-k) materials will exaggerate the thermal effects because of their low thermal conductivity. To effectively analyze the thermal distribution and locate the hot spots, chip-level thermal analyses are of crucial importance. Furthermore, for the finite thermal conductivity of the complicated packaging problem, the uniform heat distribution does not guarantee the uniform temperature profile. Thus, it is valuable to know the temperature profile and hot spots, not only for the steady state, but also the transient state.

Several approaches have been proposed to perform thermal analysis [5]–[7]. However, due to the complexity of solving the large-scale matrix, the existing direct matrix-solving algorithms suffer from superlinear run time and memory consumption for large-scale problems. In this paper, we propose an algorithm using the alternating-direction-implicit (ADI) method [8], [9] to simulate the temperature profile. Our method, thermal-ADI is not only unconditionally stable, but also has a linear run time and memory requirement. The experimental results show orders of run-time improvement over the traditional methods.

The remainder of this paper is organized as follows. Thermal-simulation physics is discussed in Section II. Section III presents an overview of the numerical formulation of the heat transfer. Section IV introduces the ADI method in our simulation. The implementation and experimental results are presented in Section V, followed by the conclusion in Section VI.

II. Thermal-Simulation Physics

As shown in Fig. 1, the temperature distribution in a chip is governed by the following partial differential equation of heat conduction from the law of energy conservation [10]:

$$\rho c_p \frac{\partial T(\vec{r},t)}{\partial t} = \bigtriangledown \cdot \left[\kappa(\vec{r},T) \bigtriangledown T(\vec{r},t)\right] + g(\vec{r},t) \qquad (1)$$

subject to the following thermal boundary conditions:

$$\kappa(\vec{r},T)\frac{\partial T(\vec{r},t)}{\partial n_i} + h_i T(\vec{r},t) = f_i(\vec{r}_{s_i},t) \qquad (2)$$

where T is the time-dependent temperature at any $\vec{r}$ is the density of the material, c_p is the specific heat, κ is the thermal conductivity, g is the heat energy generation rate, h_i is the

Manuscript received February 19, 2002. This work was supported in part by the National Science Foundation under Grant CCR-0093309, by the Intel Corporation, and by Faraday, Inc.

T.-Y. Wang is with the Department of Electrical and Computer Engineering, University of Wisconsin–Madison, Madison, WI 53706 USA (e-mail: wangt@cae.wisc.edu).

C. C.-P. Chen is with the Graduate Institute of Electronics Engineering and Department of Electrical Engineering, National Taiwan University, Taipei 106, Taiwan (e-mail: cchen@cc.ee.ntu.edu.tw).

Digital Object Identifier 10.1109/TVLSI.2003.812372

TABLE I
ITRS: TREND OF THE MPU TECHNOLOGY REQUIREMENTS: NEAR TERM (1999–2005) AND LONG TERM (2008–2014)

Year	1999	2000	2001	2002	2003	2004	2005	2008	2011	2014
Technology Node(nm)	180	-	-	130	-	-	100	70	50	35
Maximum Power (W)	90	100	115	130	140	150	160	170	174	183
Clock(MHz)	1200	1321	1454	1600	1724	1857	2000	2500	3004	3600
# of wiring levels	7	7	7	8	8	8	9	9	10	10
Current Density (A/cm^2) with wire at $105°C$	5.8e5	7.1e5	8.0e5	9.5e5	1.1e6	1.3e6	1.4e6	2.1e6	3.7e6	4.6e6

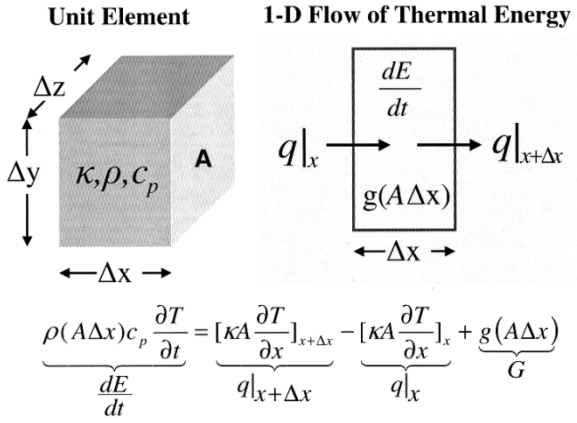


Fig. 1. Conservation of energy and the heat-conduction equation. The one-dimensional system is shown on the right-hand side, and the heat equation derived by energy conservation is shown at the bottom. The conduction heat into the system is $q|_x = -[\kappa A(\partial T/\partial x)]_x$. The conduction heat out of the system is $q|_{x+\Delta x} = -[\kappa A(\partial T/\partial x)]_{x+\Delta x}$. dE/dt is the change rate of energy stored inside the system. G is the rate of energy generation inside the system.

heat-transfer coefficient on the boundary surface of the chip, $f_i(\vec{r}_{s_i}, t)$ is an arbitrary function on the boundary surface s_i, and $\partial/\partial n_i$ is the differentiation along the outward direction normal to the boundary surface s_i.

In general, thermal conductivity $\kappa(\vec{r}, T)$ is dependent on the position and temperature. The heat-generation rate $g(\vec{r}, t)$ arises from the power consumption in gates and power/ground/clock interconnects. The power consumption of gates is caused by short-circuit and leakage currents, and the power consumption of power/ground/clock interconnects is generated by self-generating. The energy generation rate in the interconnects can be expressed as $J^2_{\rm rms} \cdot R_\rho$[6], where $J_{\rm rms}$ is the rms current density, and R_ρ is the temperature-dependent resistivity of the interconnects.

Note that the thermal time constant of the heat conduction is orders larger than the clock cycle, which means that the temperature variation caused by transient currents is small. Thus, the average power in one or several periods are used in the heat-generation rate defined above to simulate the transient temperature profile. The full-chip temperature profile is supposed to become stable when the thermal steady state is reached after enough time. There are three different situations for the boundary conditions.

- *Specified Temperature*

  The temperature is prescribed along the boundary surface, i.e.,

$$T(\vec{r}, t) = f_i(\vec{r}_s, t).$$

  This is the boundary condition of the first kind.

- *Heat Flux*

  The specified heat flux along the boundary surface can be expressed as

$$\kappa(\vec{r}, T)\frac{\partial T(\vec{r}, t)}{\partial n_i} = q_{s_i}$$

  where q_{s_i} is the heat flux on the boundary surface s_i. This is the second kind of boundary condition. For the adiabatic boundary condition, we have $q_{s_i} = 0$.

- *Convection Boundary Condition*

  The heat transfers from the considered boundary surface s_i to the ambient by convection can be expressed as

$$\kappa(\vec{r}, T)\frac{\partial T(\vec{r}, t)}{\partial n_i} + h_i T(\vec{r}, t) = h_i T_\infty$$

  where T_∞ is the ambient temperature, and h_i is the equivalent heat-transfer coefficient. This is the boundary condition of the third kind.

Section III shows the discretization process.

III. FINITE-DIFFERENCE FORMULATION OF THE HEAT CONDUCTION

The two-dimensional (2-D) heat-conduction equation can be rewritten from (1) as

$$\frac{\partial T(x,y,t)}{\partial t} = \alpha\frac{\partial^2 T(x,y,t)}{\partial x^2} + \alpha\frac{\partial^2 T(x,y,t)}{\partial y^2} + \frac{1}{\rho c_p}g(x,y,t) \tag{3}$$

where $\alpha = \kappa/\rho c_p$. This equation is a second-order parabolic partial differential equation. There are two steps to build the finite-difference method. The first step to establish the finite-difference method of the partial differential equation is to discretize the continuous space domain into a mesh with a finite number of grid points. As illustrated in Fig. 2, the temperature $T(x, y, t)$ at each point in the chip will be replaced by $T(i\Delta x, j\Delta y, n\Delta t)$, which will be denoted as $T^n_{i,j}$ for the remainder of this paper. The first-order partial derivative of T with respect to x is converted to the following expression by the forward-difference representation:

$$\frac{\partial T}{\partial x}\Big|^n_{i,j} = \frac{T^n_{i+1,j} - T^n_{i,j}}{\Delta x} + O(\Delta x) \approx \frac{T^n_{i+1,j} - T^n_{i,j}}{\Delta x}$$

where the truncation error is $O[\Delta x]$. Similarly, the central-difference representation of the second-order partial derivative of T with respect to x can be expressed as

$$\begin{aligned}\frac{\partial^2 T}{\partial x^2}\Big|^n_{i,j} &= \frac{T^n_{i+1,j} - 2T^n_{i,j} + T^n_{i-1,j}}{(\Delta x)^2} + O(\Delta x)^2 \\ &\approx \frac{T^n_{i+1,j} - 2T^n_{i,j} + T^n_{i-1,j}}{(\Delta x)^2} \\ &= \frac{\delta^2_x T^n}{(\Delta x)^2}\end{aligned}$$

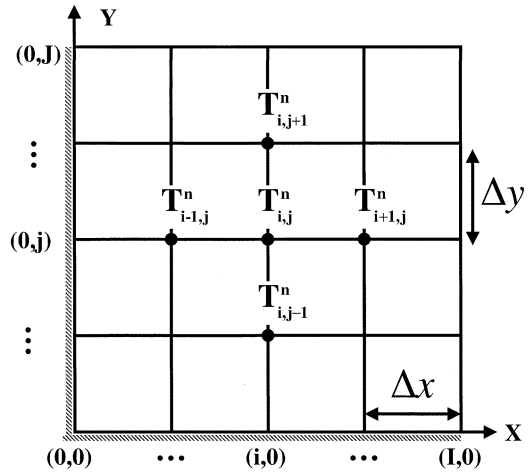


Fig. 2. Finite-difference mesh on the $x-y$ plane.

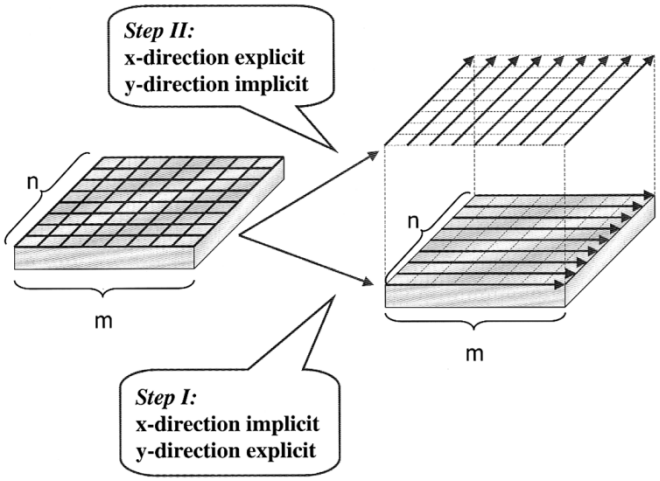


Fig. 3. Illustration of the ADI method.

where the truncation error is $O[(\Delta x)^2]$, and $\delta_x^2 T^n = T^n_{i-1,j} - 2T^n_{i,j} + T^n_{i+1,j}$.

The next step is to consider the time-marching problem for the finite-difference equations. Since (1) comes from the law of energy conservation, it can be explained physically as the increasing rate of the stored energy in a control unit volume being equal to the net rate of energy transferring into the control volume and the generation rate of heat energy in the control volume. By applying the forward difference with time on the left-hand side of (3), and replacing the space-domain approximation terms on the right-hand side, we have the following expression:

$$\frac{T^{n+1}_{i,j} - T^n_{i,j}}{\Delta t} = \alpha \left[\frac{\delta_x^2 T^?}{(\Delta x)^2} + \frac{\delta_y^2 T^?}{(\Delta y)^2} \right] + \frac{1}{\rho c_p} g_{i,j}. \quad (4)$$

The term on the left-hand side in (4) relates to the energy stored from time step n to $n+1$ in the control unit volume. However, during the time period from n to $n+1$, what time step will be used to update the terms on the right-hand side relating to the net rate of energy transferring into the control volume? There are three choices of time marching in the terms expressed with question marks in (4) [11], [12].

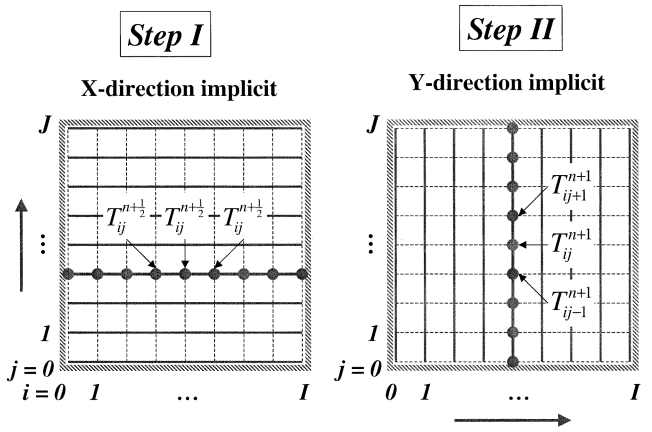


Fig. 4. For *Step I*, the x-direction is implicit and the y-direction is explicit. For each j, there are $I+1$ equations corresponding to $I+1$ points. Note that each temperature point $T^{n+1/2}_{i,j}$ at value j is related to two unknowns $T^{n+1/2}_{i-1,j}$ and $T^{n+1/2}_{i+1,j}$, which introduces a tridiagonal matrix for each j. *Step II* has a similar process to *Step I*.

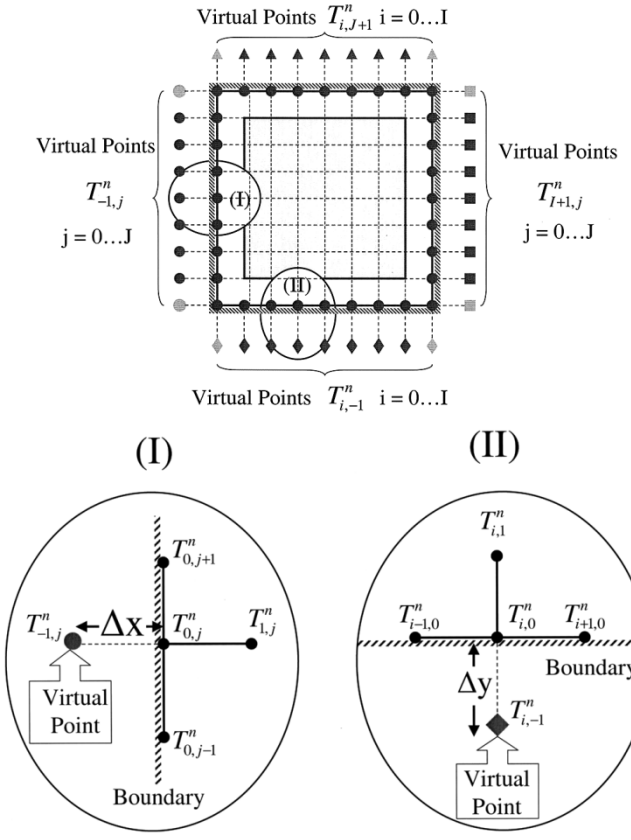


Fig. 5. Boundary conditions.

- *Simple Explicit Method*

  Applying the explicit update on the right-hand side of (4) at time step n, we get

$$\frac{T^{n+1}_{i,j} - T^n_{i,j}}{\Delta t} = \alpha \left[\frac{\delta_x^2 T^n}{(\Delta x)^2} + \frac{\delta_y^2 T^n}{(\Delta y)^2} \right] + \frac{1}{\rho c_p} g_{i,j}. \quad (5)$$

Note that there is only one unknown in each equation with respect to point (i,j). Thus, there is no matrix inversion

TABLE II
DIFFERENCE EQUATIONS OF THE PEACEMAN–RACHFORD METHOD ON THE BOUNDARY POINTS

Step I

$$(2+2r_x\beta_{0x})T_{0,j}^{n+\frac{1}{2}} - 2r_xT_{1,j}^{n+\frac{1}{2}} = r_yT_{0,j-1}^{n} + (2-2r_y)T_{0,j}^{n} + r_yT_{0,j+1}^{n} + 2r_xr_{0x} + \frac{\Delta t}{2\rho c_p}g_{0,j}$$

$$-2r_xT_{I-1,j}^{n+\frac{1}{2}} + (2+2r_x\beta_{Ix})T_{I,j}^{n+\frac{1}{2}} = r_yT_{I,j-1}^{n} + (2-2r_y)T_{I,j}^{n} + r_yT_{I,j+1}^{n} + 2r_xr_{Ix} + \frac{\Delta t}{2\rho c_p}g_{I,j}$$

Step II

$$(2+2r_y\beta_{0y})T_{i,0}^{n+1} - 2r_yT_{i,1}^{n+1} = r_xT_{i-1,0}^{n+\frac{1}{2}} + (2-2r_x)T_{i,0}^{n+\frac{1}{2}} + r_xT_{i+1,0}^{n+\frac{1}{2}} + 2r_yr_{0y} + \frac{\Delta t}{2\rho c_p}g_{i,0}$$

$$-2r_yT_{i,J-1}^{n+1} + (2+2r_y\beta_{Jy})T_{i,J}^{n+1} = r_xT_{i-1,J}^{n+\frac{1}{2}} + (2-2r_x)T_{i,J}^{n+\frac{1}{2}} + r_xT_{i+1,J}^{n+\frac{1}{2}} + 2r_yr_{Jy} + \frac{\Delta t}{2\rho c_p}g_{i,J}$$

where

$$r_{0x} = \frac{\Delta x h_{0x}T_{\infty,0x}}{\kappa}; \quad r_{Ix} = \frac{\Delta x h_{Ix}T_{\infty,Ix}}{\kappa}; \quad r_{0y} = \frac{\Delta y h_{0y}T_{\infty,0y}}{\kappa}; \quad r_{Jy} = \frac{\Delta y h_{Jy}T_{\infty,Jy}}{\kappa};$$

$$\beta_{0x} = (1+\frac{\Delta x h_{0x}}{\kappa}); \quad \beta_{Ix} = (1+\frac{\Delta x h_{Ix}}{\kappa}); \quad \beta_{0y} = (1+\frac{\Delta y h_{0y}}{\kappa}); \quad \beta_{Jy} = (1+\frac{\Delta y h_{Jy}}{\kappa})$$

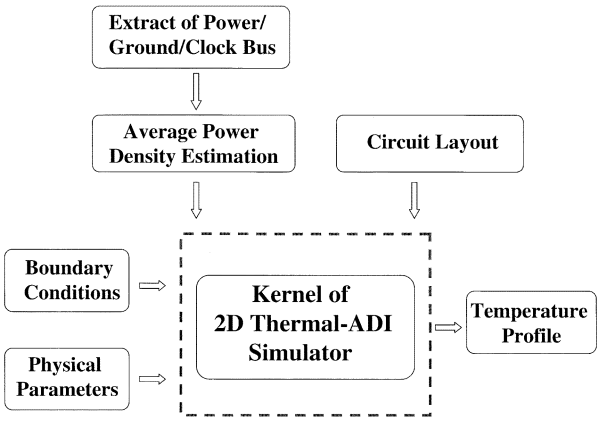


Fig. 6. Flowchart of the thermal-ADI simulation.

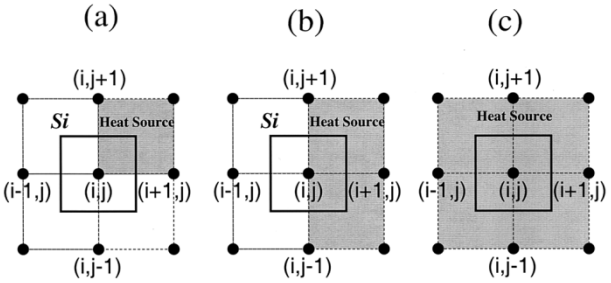


Fig. 7. Three different situations of the layout extraction.

needed. The truncation error is $O[(\Delta t),(\Delta x)^2,(\Delta y)^2]$. The stability constraint γ can be described as follows:

$$\gamma = \alpha\Delta t\left(\frac{1}{(\Delta x)^2}+\frac{1}{(\Delta y)^2}\right) \le \frac{1}{2}. \tag{6}$$

This constraint restricts the size of the time step Δt for the given space increment Δx and Δy.

- *Simple Implicit Method*

Applying the simple implicit update on the right-hand side of (4) at time step n, we get

$$\frac{T_{i,j}^{n+1}-T_{i,j}^{n}}{\Delta t} = \alpha\left[\frac{\delta_x^2T^{n+1}}{(\Delta x)^2}+\frac{\delta_y^2T^{n+1}}{(\Delta y)^2}\right]+\frac{1}{\rho c_p}g_{i,j}. \tag{7}$$

There are five unknowns in each equation at point (i,j). This approach involves large-scale matrix inversion, and the truncation error is $O[(\Delta t),(\Delta x)^2,(\Delta y)^2]$. However, the advantage of the method is that there is no restriction on the step size because of the unconditional stability.

- *Crank–Nicolson Method*

Crank and Nicolson developed another method by taking the average of the simple explicit update and the simple implicit update on the right-hand side of (4). Thus, they obtained

$$\frac{T_{i,j}^{n+1}-T_{i,j}^{n}}{\Delta t} = \alpha\left[\frac{\delta_x^2T^{n+1}+\delta_x^2T^{n}}{2(\Delta x)^2}+\frac{\delta_y^2T^{n+1}+\delta_y^2T^{n}}{2(\Delta y)^2}\right]+\frac{g_{i,j}}{\rho c_p}. \tag{8}$$

There are five unknowns in each equation at point (i,j) for the time step at $n+1$. The Crank–Nicolson method not only has the best accuracy with the truncation error $O[(\Delta t)^2,\ (\Delta x)^2,(\Delta y)^2]$, but also unconditional stability. Thus, the difference equation of the heat conduction derived from the Crank–Nicolson method can be expressed as follows:

$$\begin{aligned} &-r_xT_{i-1,j}^{n+1}-r_yT_{i,j-1}^{n+1}+2(1+r_x+r_y)T_{i,j}^{n+1}\\ &\quad -r_xT_{i+1,j}^{n+1}-r_yT_{i,j+1}^{n+1}\\ &\quad = r_xT_{i-1,j}^{n}+r_yT_{i,j-1}^{n}+2(1-r_x-r_y)T_{i,j}^{n}\\ &\quad + r_xT_{i+1,j}^{n}+r_yT_{i,j+1}^{n}+\frac{2\Delta t}{\rho c_p}g_{i,j},\\ &\quad i=1,2,\ldots,I-1; \quad j=1,2,\ldots,J-1 \end{aligned} \tag{9}$$

where $r_x = \alpha\Delta t/(\Delta x)^2$ and $r_y = \alpha\Delta t/(\Delta y)^2$.

However, there are some difficulties for solving the above equations. Considering a 2-D mesh with size $(I+1)$ by $(J+1)$, the number of variables needed for this system is $N = (I+1)(J+1)$, which requires a matrix A with size $N\times N$ to store

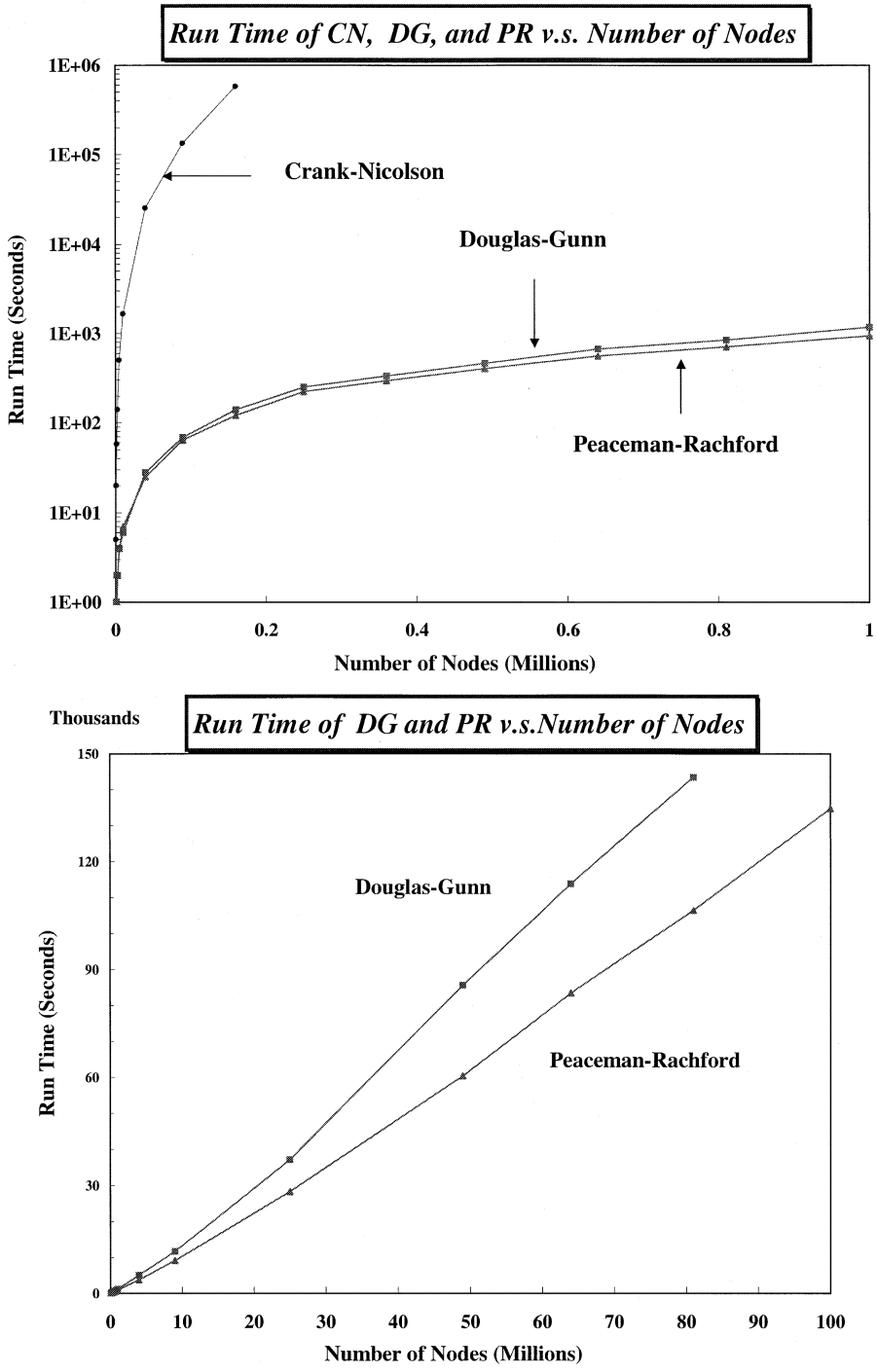


Fig. 8. Run-time comparisons of the Crank–Nicolson, the Douglas–Gunn, and Peaceman–Rachford methods.

the coefficients. To solve the equations $Ax = b$ by LU decomposition or Cholesky decomposition, the run-time and memory requirements are superlinear even with sparse matrix techniques.

IV. ADI Methods

Since the Crank–Nicolson method is computationally intensive, Peaceman and Rachford [8] and Douglas and Gunn [9] developed the ADI method to overcome this difficulty. Basically, the ADI method is a process to reduce the two- or three-dimensional problems to a succession of two or three one-dimensional problems, as shown in Fig. 3. Two detailed approaches are discussed as follows.

A. Peaceman–Rachford Method

First, rearranging (8) and having the following expression:

$$\left(1-\frac{r_x}{2}\delta_x^2\right)\left(1-\frac{r_y}{2}\delta_y^2\right)T^{n+1} = \left(1+\frac{r_x}{2}\delta_x^2\right)\left(1+\frac{r_y}{2}\delta_y^2\right)T^n+\frac{\Delta t}{\rho c_p}g. \quad (10)$$

Peaceman and Rachford proposed an ADI method to solve the above equations by two steps.

- **Step I**

$$\left(1-\frac{r_x}{2}\delta_x^2\right)T^{n+1/2}=\left(1+\frac{r_y}{2}\delta_y^2\right)T^n+\frac{\Delta t}{2\rho c_p}g. \quad (11)$$

- **Step II**

$$\left(1-\frac{r_y}{2}\delta_y^2\right)T^{n+1}=\left(1+\frac{r_x}{2}\delta_x^2\right)T^{n+1/2}+\frac{\Delta t}{2\rho c_p}g. \quad (12)$$

This algorithm completes the time step from n to $n+1$ with two sub-time steps: from n to $n+1/2$ and from $n+1/2$ to $n+1$. The detailed process is shown in Fig. 4. In *Step I*, it applies the implicit update in the x-direction and the explicit update in the y-direction. For each j value, there are $I+1$ equations for the corresponding points $T_{i,j}^{n+1/2}$, where $i=0,\ldots,I$. Since each temperature point $T_{i,j}^{n+1/2}$ is related to two temperature points $T_{i-1,j}^{n+1/2}$ and $T_{i+1,j}^{n+1/2}$, the coefficient matrix is in tridiagonal form, which can be easily solved with linear run time. In *Step II*, it applies the implicit update in the y-direction and the explicit update in the x-direction similar to *Step I*.

The detailed difference equations in *Step I* for each j in (11) can be derived as follows:

$$\begin{aligned}&-r_xT_{i-1,j}^{n+1/2}+(2+2r_x)T_{i,j}^{n+1/2}-r_xT_{i+1,j}^{n+1/2}\\&\quad=r_yT_{i,j-1}^{n}+(2-2r_y)T_{i,j}^{n}+r_yT_{i,j+1}^{n}+\frac{\Delta t}{\rho c_p}g_{i,j},\\&\qquad i=1,2,\ldots,I-1;\quad j=1,2,\ldots,J-1. \quad (13)\end{aligned}$$

Similarly, the difference equations in *Step II* for each i in (12) can be expressed as

$$\begin{aligned}&-r_yT_{i,j-1}^{n+1}+(2+2r_y)T_{i,j}^{n+1}-r_yT_{i,j+1}^{n+1}\\&\quad=r_xT_{i-1,j}^{n+1/2}+(2-2r_x)T_{i,j}^{n+1/2}+r_xT_{i+1,j}^{n+1/2}+\frac{\Delta t}{\rho c_p}g_{i,j},\\&\qquad i=1,2,\ldots,I-1;\quad j=1,2,\ldots,J-1. \quad (14)\end{aligned}$$

Now, we can reduce the 2-D problems into a succession of two one-dimensional problems by the Peaceman–Rachford ADI method. For every one-dimensional problem, the tridiagonal matrix is solved by the Thomas algorithm [11] with time complexity $O(n)$.

B. Douglas–Gunn Method

Douglas and Gunn developed another general ADI method that is unconditionally stable and second-order accurate. As shown in [9], (8) can be rewritten as

$$T^{n+1}-T^n=\frac{r_x\delta_x^2}{2}(T^{n+1}+T^n)+\frac{r_y\delta_y^2}{2}(T^{n+1}+T^n)+\frac{\Delta t}{\rho c_p}g. \quad (15)$$

According to the Douglas–Gunn method, (15) is solved by two sub-time steps.

- **Step I**

$$T^{n+1/2}-T^n=\frac{r_x\delta_x^2}{2}(T^{n+1/2}+T^n)+r_y\delta_y^2T^n+\frac{\Delta t}{\rho c_p}g. \quad (16)$$

TABLE III
RUN TIME OF THE CRANK–NICOLSONPEACEMAN–RACHFORD, AND DOUGLAS–GUNN METHODS

# Nodes	CN (seconds)	PR (seconds)	DG (seconds)
2500	141	2	2
4900	502	4	4
10000	1665	7	6
40000	25303	25	28
90000	133433	64	69
160000	576750	121	140
250000	2520000	223	252
360000	-	297	336
490000	-	405	465
640000	-	561	672
810000	-	711	845
1000000	-	942	1182
4000000	-	3772	5047
9000000	-	9113	11729
25000000	-	28235	37122
49000000	-	60375	85625
64000000	-	83500	113830
81000000	-	106525	143414
100000000	-	134708	-

- **Step II**

$$T^{n+1}-T^n=\frac{r_x\delta_x^2}{2}(T^{n+1/2}+T^n)+\frac{r_y\delta_y^2}{2}(T^{n+1}+T^n)+\frac{\Delta t}{\rho c_p}g. \quad (17)$$

Since the Douglas–Gunn method uses a similar ADI method to the Peaceman–Rachford method, we will ignore the detailed discussion about the time marching. After derivation, the difference equations in *Step I* for each j value are as follows:

$$\begin{aligned}&-r_xT_{i-1,j}^{n+1/2}+2(1+r_x)T_{i,j}^{n+1/2}-r_xT_{i+1,j}^{n+1/2}\\&\quad=r_xT_{i-1,j}^{n}+2\,r_yT_{i,j-1}^{n}+2(1-r_x-2r_y)T_{i,j}^{n}\\&\qquad+r_xT_{i+1,j}^{n}+2r_yT_{i,j+1}^{n}+\frac{2\Delta t}{\rho c_p}g_{i,j},\\&\qquad i=1,2,\ldots,I-1;\quad j=1,2,\ldots,J-1. \quad (18)\end{aligned}$$

The difference equations in *Step II* for each i value are as follows:

$$\begin{aligned}&-r_yT_{i,j-1}^{n+1}+2(1+r_y)T_{i,j}^{n+1}-r_yT_{i,j+1}^{n+1}\\&\quad=(r_xT_{i-1,j}^{n+1/2}-2r_xT_{i,j}^{n+1/2}+r_xT_{i+1,j}^{n+1/2})+r_xT_{i-1,j}^{n}+r_yT_{i,j-1}^{n}\\&\qquad+2(1-r_x-r_y)T_{i,j}^{n}+r_xT_{i+1,j}^{n}+r_yT_{i,j+1}^{n}+\frac{2\Delta t}{\rho c_p}g_{i,j},\\&\qquad i=1,2,\ldots,I-1;\quad j=1,2,\ldots,J-1. \quad (19)\end{aligned}$$

Similarly, the above tridiagonal equation sets can be solved with a linear run time.

Consider a given mesh with size $N=(I+1)\times(J+1)$. In *Step I*, there are $J+1$ values of j, and each j takes time $O(I+1)$ to solve the tridiagonal system. Therefore, the run time is $O(2\times(J+1)\times(I+1))$ for a complete ADI iteration. The time complexity of the thermal-ADI method is then $O(K2N)=O(N)$, where K is the iteration number to reach steady state. This analysis means that the thermal-ADI method is linear with respect to the total number of nodes. In Section IV-C, we will consider the boundary conditions.

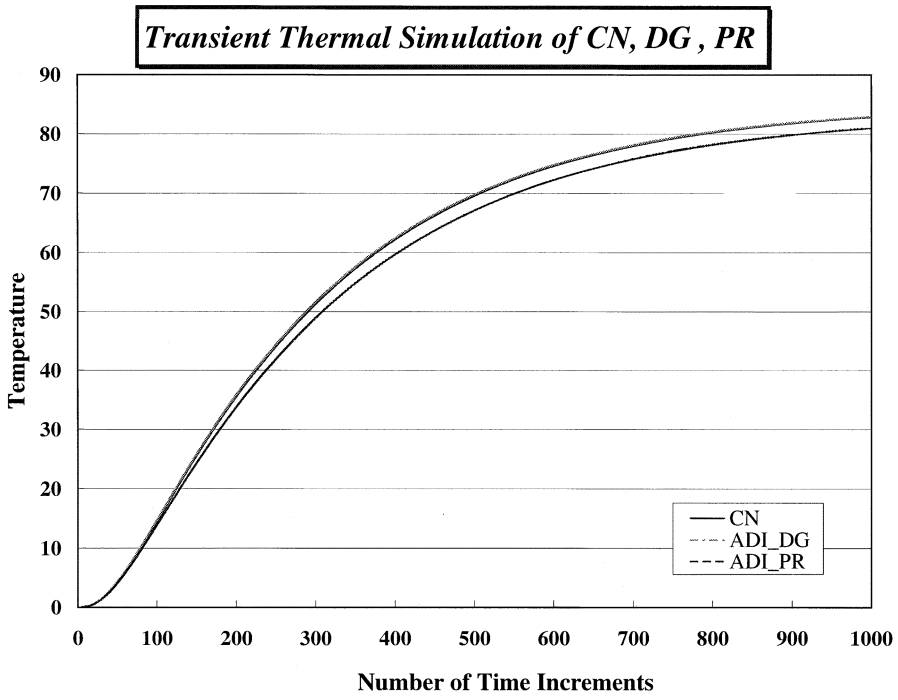


Fig. 9. Transient simulation results of the Crank–Nicolson, Douglas–Gunn, and Peaceman–Rachford methods at a point in the chip.

C. Boundary Conditions

The difference equations of heat conduction by the Cranck–Nicolson method in (9) for $i = 1,2,\ldots,(I-1)$ and $j = 1,2,\ldots,(J-1)$ only provide $(I-1)\times(J-1)$ algebraic equations, but contain $(I+1)\times(J+1)$ unknown temperature nodes $T^n_{i,j}$. Similarly, the difference equations for the Peaceman–Rachford method and the Douglas–Gunn method in (13) and (14) and (18) and (19), respectively, need two extra equations for each i or j value. Hence, the points on the boundary conditions are needed to make the number of equations equal to the number of unknowns. Suppose that we have the following boundary conditions:

$$\begin{aligned} -\kappa\frac{\partial T}{\partial x} + h_{0x}T &= h_{0x}T_\infty, && \text{at } i = 0 \\ \kappa\frac{\partial T}{\partial x} + h_{Ix}T &= h_{Ix}T_\infty, && \text{at } i = I \\ -\kappa\frac{\partial T}{\partial y} + h_{0y}T &= h_{0y}T_\infty, && \text{at } j = 0 \\ \kappa\frac{\partial T}{\partial y} + h_{Jy}T &= h_{Jy}T_\infty, && \text{at } j = J \end{aligned} \tag{20}$$

where h_{0x}, h_{Ix}, h_{0y}, and h_{Jy} are the equivalent heat-transfer coefficients calculated from the equivalent thermal resistance on the boundary $i = 0$, $i = I$, $j = 0$, and $j = J$, respectively, and T_∞ is the ambient temperature.

In order to achieve the second-order accuracy, the central-difference approximation will be used to discretize the boundary conditions. First, we introduce the virtual temperature nodes $T^n_{-1,j}$, $T^n_{I+1,j}$, $T^n_{i,-1}$, and $T^n_{i,J+1}$ by expanding the region with Δx to the left and right and with Δy to the bottom and top, as shown in Fig. 5, and then apply the central-difference approximation to discretize the boundary condition, e.g., at $i = 0$ and time step $n + 1/2$ in (20). We then have

$$-\kappa\frac{T^{n+1/2}_{1,j} - T^{n+1/2}_{-1,j}}{2\Delta x} + h_{0x}T^{n+1/2}_{0,j} = h_{0x}T_\infty, \qquad \text{at } i = 0. \tag{21}$$

Thus, the virtual point can be expressed as

$$T^{n+1/2}_{-1,j} = T^{n+1/2}_{1,j} + \frac{2\Delta x h_{0x}}{\kappa}\left(T_\infty - T^{n+1/2}_{0,j}\right). \tag{22}$$

The other virtual points $T^{n+1/2}_{I+1,j}$, $T^{n+1/2}_{i,-1}$, and $T^{n+1/2}_{i,J+1}$ can be derived via the same way. These derived virtual points can then be used to replace the virtual points occurring on the boundary points in the difference equations. For example, the difference equations of the Peaceman–Rachford method at points (0, j) in *Step I* have the expression

$$\begin{aligned} &-r_x T^{n+1/2}_{-1,j} + (2 + 2r_x)T^{n+1/2}_{0,j} - r_x T^{n+1/2}_{1,j} \\ &\quad = r_y T^n_{0,j-1} + (2 - 2r_y)T^n_{0,j} + r_y T^n_{0,j+1} + \frac{\Delta t}{\rho c_p} g_{0,j}. \end{aligned} \tag{23}$$

The virtual point $T^{n+1/2}_{-1,j}$ in the above equation can be replaced by the expression in (22), and then the difference equations of *Step I* at points (0, j) for the boundary conditions are

$$\begin{aligned} &(2 + 2r_x\beta_{0x})T^{n+1/2}_{0,j} - 2r_x T^{n+1/2}_{1,j} \\ &\quad = r_y T^n_{0,j-1} + (2 - 2r_y)T^n_{0,j} + r_y T^n_{0,j+1} + 2r_x r_{0x} + \frac{\Delta t}{\rho c_p} g_{0,j}. \end{aligned} \tag{24}$$

where $\beta_{0x=1+(\Delta x\, h x_{0x})/\kappa}$ and $r_{0x} = (\Delta x\, h_{0x} T_\infty)/\kappa$.

The difference equations of the Peaceman–Rachford method on the boundary $i = I$, $j = 0$, and $j = J$ can be obtained similarly and are shown in Table II. The difference equations on the boundary for the Crank–Nicolson method and the Douglas–Gunn algorithm can be derived similarly.

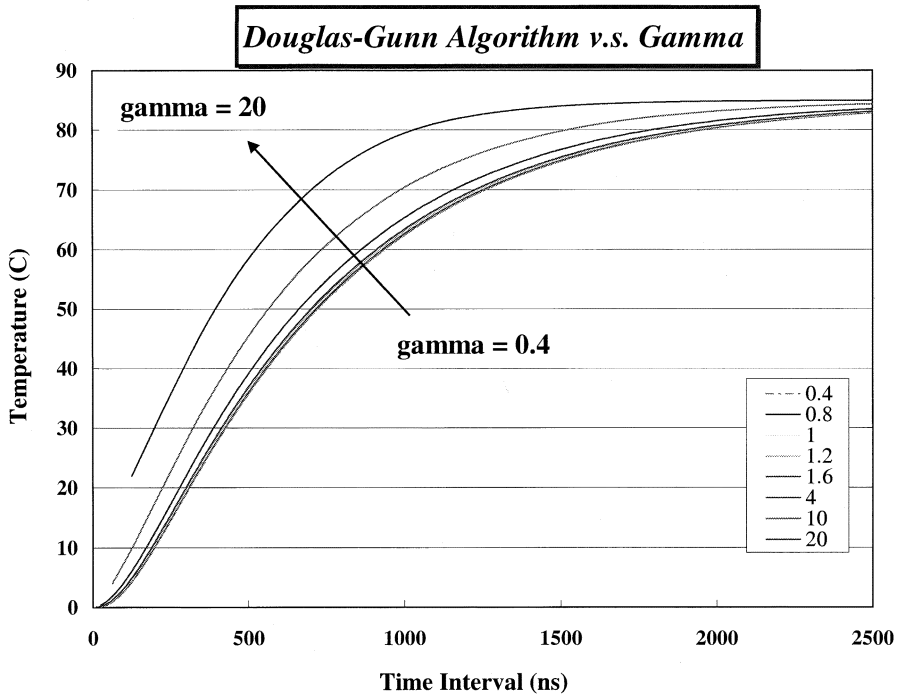


Fig. 10. Douglas–Gunn method is unconditionally stable. Since the γ values exceed the criteria value 0.5, the results are still stable rather than oscillating.

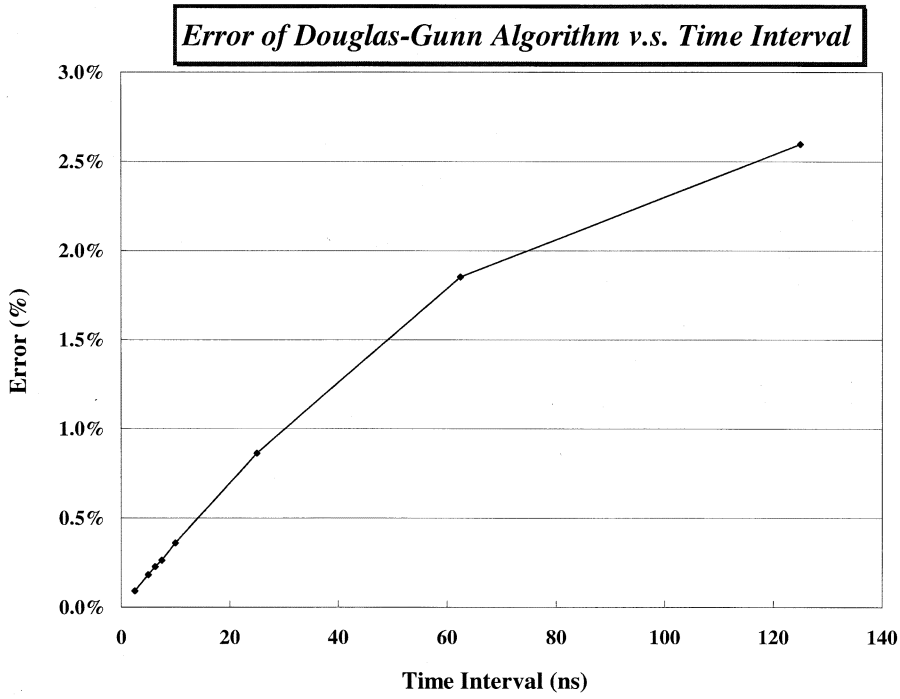


Fig. 11. Errors of the Douglas–Gunn method compared to the Crank–Nicolson method at time step 1000, as shown in Fig. 10.

V. Implementation and Experimental Results

The flowchart of the transient thermal simulation by the ADI method is shown in Fig. 6. The main features of implementation are as follows.

- The core part of the thermal-ADI is a fast 2-D transient thermal simulator. The first step for the thermal simulation is to partition the chip into a mesh, and choose the size of Δt, Δx, and Δy. The second step is to extract the layout of transistors and metal wires from circuits. There are three situations of layout extraction, as shown in Fig. 7. The simulator reads the given circuit descriptions, the coordinates of the gates, metal wires, and power/ground/clock interconnects for thermal simulation, and then calculates the information at each node on the mesh according to the physical parameters at that point. Finally, it iteratively calculates the temperature at each node by the ADI method and outputs the temperature profile.

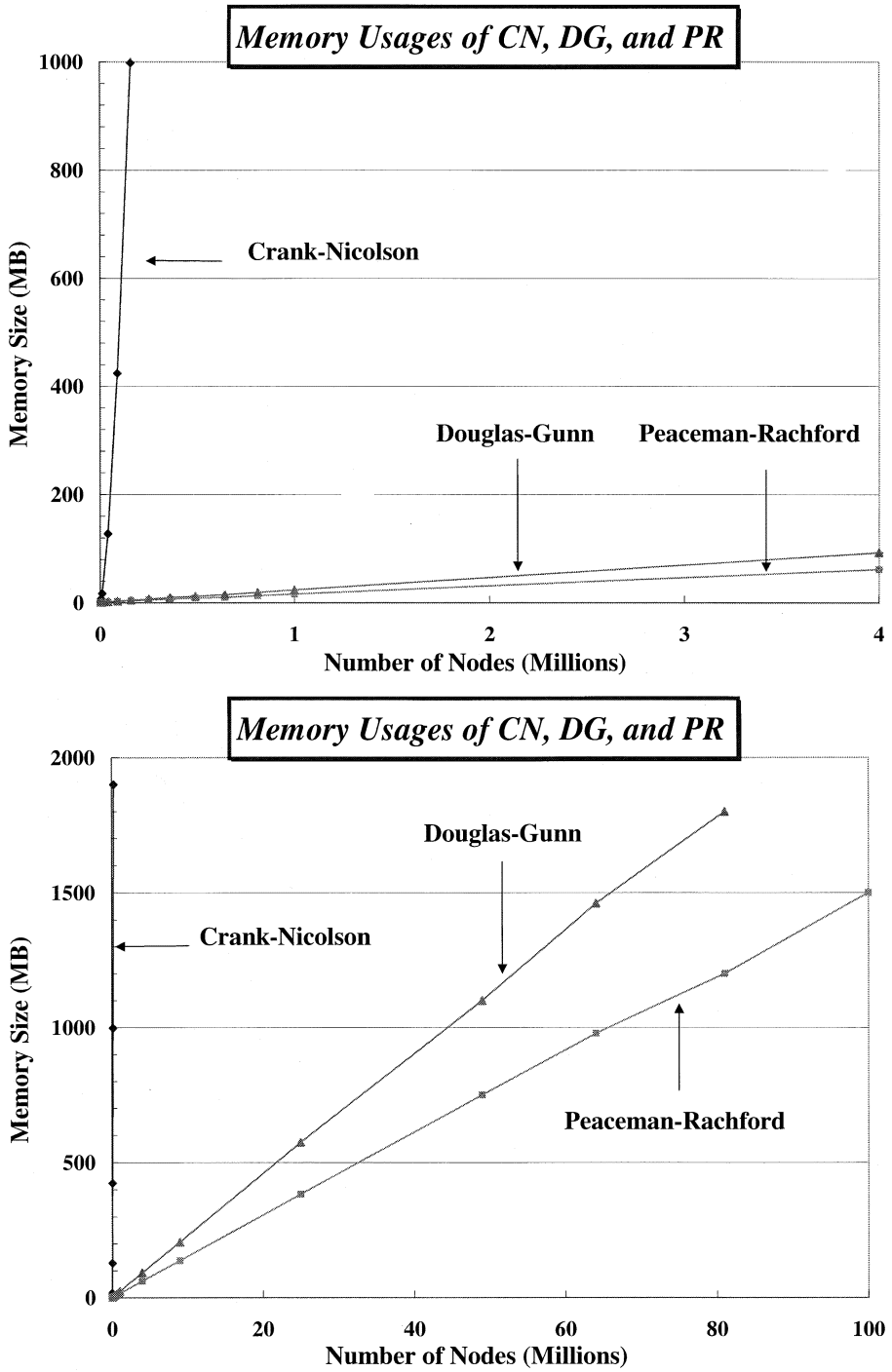


Fig. 12. Comparison of memory usages of the Crank–Nicolson, Douglas–Gunn, and the Peaceman–Rachford methods.

- We use the band structure to implement the Crank–Nicolson method in order to reduce the run time and the memory usage. Therefore, the chip with mesh points $N = (I + 1) \times (J + 1)$, we need a matrix A with size $N \times (I + 2)$ rather than $N \times N$.

The thermal-ADI simulation was implemented with C++ language, and was executed on an Alpha workstation with dual SLOTB 667-MHz Alpha 21264 processors.

The run-time comparison of the Crank–Nicolson, the Peaceman–Rachford, and the Douglas–Gunn methods is shown in Fig. 8 and Table III. All the methods are tested with 500 time iterations. As can be seen in Fig. 8, the run time of the Douglas–Gunn and Peaceman–Rachford methods is linearly proportional to the number of nodes, as shown with the node number up to 10^8. On the other hand, the run time of the Crank–Nicolson method increases dramatically. Note that the scale in the y-axis is logarithm in the top figure.

The temperature of transient thermal simulation at a random chosen point is shown in Fig. 9. This is done by a 100×100 mesh with 1000 time steps. The difference of the Douglas–Gunn method compared to the Crank–Nicolson method is less than 0.1%, and the difference of the Peaceman–Rachford method

TABLE IV
MEMORY USAGES OF THE CRANK–NICOLSON, PEACEMAN–RACHFORD, AND DOUGLAS–GUNN METHODS

# Nodes	CN (Mb)	PR (Mb)	DG (Mb)
4900	7.048	1.104	1.144
10000	17	1.192	1.272
40000	128	1.68	1.992
90000	424	2.48	3.184
160000	998	3.592	4.848
250000	1900	5.016	6.976
360000	-	6.752	9.68
490000	-	8.8	12
640000	-	10	15
810000	-	13	19
1000000	-	16	24
4000000	-	62	93
9000000	-	138	207
25000000	-	383	574
49000000	-	749	1100
64000000	-	978	1460
81000000	-	1200	1800
100000000	-	1500	-

compared to the Crank–Nicolson method is 2.25%. The transient temperature curve of the Douglas–Gunn method is very close to the curve of the Crank–Nicolson method.

In order to test the property of unconditional stability, the stability constraint γ in (6) was simulated from 0.4 to 20, as shown in Fig. 10, where 1/2 is the stability limit. As can be observed in Fig. 10, the Douglas–Gunn method is unconditionally stable because there is no oscillation with $\gamma \geq 1/2$. Fig. 10 also shows that the larger Δt of the Douglas–Gunn algorithm turns out to have the bigger deviation away from the curve of the Crank–Nicolson method.

The errors of the Douglas–Gunn method are shown in Fig. 11 with different γ at time step 1000. Here, we only change the value of Δt in γ while keeping the other factors fixed. The errors increase with respect to the γ value.

As illustrated in Fig. 12 and Table IV, the memory usages of the Douglas–Gunn and Peaceman–Rachford methods are linearly proportional to the number of nodes up to 10^8. However, the memory usage of the Crank–Nicolson method increases dramatically. This traditional method using a lot of memory space limits the simulation size of the problems.

VI. CONCLUSIONS

An efficient thermal-ADI algorithm for transient thermal simulation has been developed. The unconditional stability and a linear run-time and memory requirement have been demonstrated. The numerical simulation has also shown that the thermal-ADI algorithm not only speeds up the run-time orders of magnitude over the Crank–Nicolson method, but also reduces the memory usages. The error of the Douglas–Gunn algorithm can be reduced to 0.1% with a suitable choice of Δt and time step.

REFERENCES

[1] K. Banerjee and A. Mehrotra, "Global (interconnect) warming," *IEEE Circuits Devices Mag.*, vol. 17, pp. 16–32, Sept. 2001.

[2] K. Banerjee, M. Pedram, and A. H. Ajami, "Analysis and optimization of thermal issues in high-performance VLSI," in *ACM/SIGDA Int. Physical Design Symp.*, Apr. 2001, pp. 230–237.

[3] K. Banerjee, A. Mehrotra, A. Sangiovanni–Vincentelli, and C. Hu, "On thermal effects in deep sub-micron VLSI interconnects," in *36th ACM/IEEE Design Automation Conf.*, 1999, pp. 885–891.

[4] V. Székely, "Tracing the thermal behavior of ICs," *IEEE Des. Test Comput.*, vol. 15, pp. 14–21, Apr.–June 1998.

[5] Y.-K. Cheng, P. Raha, C.-C. Teng, E. Rosenbaum, and S.-M. Kang, "ILLIADS-T: An electrothermal timing simulator for temperature-sensitive reliability diagnosis of CMOS VLSI chips," *IEEE Trans. Computer-Aided Design*, vol. 17, pp. 668–681, Aug. 1998.

[6] D. Chen, E. Li, E. Rosenbaum, and S.-M. Kang, "Interconect thermal modeling for accurate simulation of circuit timing and reliability," *IEEE Trans. Computer-Aided Design*, vol. 19, pp. 197–205, Feb. 2000.

[7] Z. Yu, D. Yergeau, and R. W. Dutton, "Full chip thermal simulation," in *Int. Quality Electronic Design Symp.*, Mar. 2000, pp. 145–149.

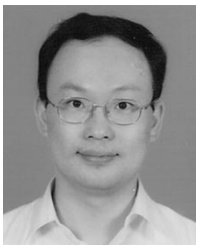
[8] D. W. Peaceman and H. H. Rachford, Jr., "The numerical solution of parabolic and elliptic differential equations," *J. Soc. Ind. Appl. Math.*, vol. 3, pp. 28–41, 1955.

[9] J. Douglas, Jr. and J. E. Gunn, "A general formulation of alternating direction methods—Part I: Parabolic and hyperbolic problems," *Numer. Math.*, vol. 6, pp. 428–453, 1964.

[10] M. N. Ozisik, *Boundary Value Problems of Heat Conduction*. New York: Dover, 1968.

[11] ——, *Finite Difference Methods in Heat Transfer*. Boca Raton, FL: CRC, 1994.

[12] G. E. Myers, *Analytical Methods in Conduction Heat Transfer*, 2nd ed. Schenectady, NY: Genium, 1987.



Ting-Yuan Wang received the B.S. degree in physics from the University of Cheng-Kung University, Tainan, Taiwan, R.O.C., in 1990, the M.S. degrees in physics and electrical and computer engineering from the National Taiwan University, Taipei, Taiwan, R.O.C., in 1992, and the University of Wisconsin–Madison, in 1998, respectively. He is currently working toward the Ph.D. degree in electrical and computer engineering at the University of Wisconsin–Madison.

Since 1999, he has been a Research Assistant with the VLSI-EDA Research Group, University of Wisconsin–Madison. In Fall 2000, Spring 2001, and Spring 2002, he was a Teaching Assistant with the Department of Electrical and Computer Engineering, University of Wisconsin–Madison. His research interests include computer-aided design on VLSI circuits and systems, with emphases on power delivery optimization design, efficient thermal simulation, and integrated circuit (IC) reliability.

Mr. Wang received the Best Paper Award nomination from the International Symposium on Quality Electronic Design (ISQED) in 2002, and the Best Paper Award from the Association for Computing Machinery (ACM)/Special Interest Group on Design Automation (SIGDA) International Symposium on Physical Design (ISPD) in 2003.

Charlie Chung-Ping Chen received the B.S. degree in computer science and information engineering from the National Chiao-Tung University, Hsinchu, Taiwan, R.O.C., in 1990, and the M.S. and Ph.D. degrees in computer science from The University of Texas at Austin, in 1996 and 1998.

From 1996 to 1999, he was a Senior CAD Engineer with the Strategic CAD Laboratories, Intel Corporation, where he was in charge of several interconnect and circuit synthesis projects with the Micro-Processor Group. Since 1999, he has been an Assistant Professor with the Department of Electrical and Computer Engineering, University of Wisconsin–Madison. His research interests are in the areas of computer-aided design and microprocessor circuit design with an emphasis on interconnect and circuit optimization, as well as signal integrity analysis and optimization.

Prof. Chen received the D2000 Award presented by the Intel Corporation and the 1999 and 2001 National Sciences Foundation Faculty Early CAREER Development Award. He also received the 2002 Special Interest Group on Design Automation (SIGDA)/Association for Computing Machinery (ACM) Outstanding Young Faculty Award, the 2002 Peter Schneider Faculty Development Award, and the Best Paper Award from ACM/SIGDA International Symposium on Physical Design (ISPD) in 2003.