Adaptive Chip-Package Thermal Analysis for Synthesis and Design

Yonghong Yang† Zhenyu (Peter) Gu‡ Changyun Zhu† Li Shang† Robert P. Dick‡

†ECE Department
Queen's University
Kingston, ON K7L 3N6, Canada
{4yy6, 4cz1}@qlink.queensu.ca, li.shang@queensu.ca

‡EECS Department
Northwestern University
Evanston, IL 60208, U.S.A.
{zgu646, dickrp}@ece.northwestern.edu

Abstract

Ever-increasing integrated circuit (IC) power densities and peak temperatures threaten reliability, performance, and economical cooling. To address these challenges, thermal analysis must be embedded within IC synthesis. However, detailed thermal analysis requires accurate three-dimensional chip-package heat flow analysis. This has typically been based on numerical methods that are too computationally intensive for numerous repeated applications during synthesis or design. Thermal analysis techniques must be both accurate and fast for use in IC synthesis.

This article presents a novel, accurate, incremental, self-adaptive, chip-package thermal analysis technique, called ISAC, for use in IC synthesis and design. It is common for IC temperature variation to strongly depend on position and time. ISAC dynamically adapts spatial and temporal modeling granularity to achieve high efficiency while maintaining accuracy. Both steady-state and dynamic thermal analysis are accelerated by the proposed heterogeneous spatial resolution adaptation and temporally decoupled element time marching techniques. Each technique enables orders of magnitude improvement in performance while preserving accuracy when compared with other state-of-the-art adaptive steady-state and dynamic IC thermal analysis techniques. Experimental results indicate that these improvements are sufficient to make accurate dynamic and static thermal analysis practical within the inner loops of IC synthesis algorithms. ISAC has been validated against reliable commercial thermal analysis tools using industrial and academic synthesis test cases and chip designs. It has been implemented as a software package suitable for integration in IC synthesis and design flows and has been publicly released.

1. Introduction

Integrated circuit (IC) densities and performance requirements are continuously increasing. The crucial task of managing the resulting increase in power density and peak IC temperature is becoming more difficult [1], [2]. Current architectural-level design automation and synthesis tools have multiple design metrics, such as power consumption, temperature, performance, cost, and reliability. IC designs must carefully trade off these metrics. However, if not properly addressed, increased IC temperature affects other design metrics including performance (via decreased transistor switching speed and increased interconnect latency), power and energy consumption (via increased leakage power), reliability (via electromigration, hot carrier effects, oxide thermal breakdown, etc.), and price (via increased system cooling cost). Considering thermal issues during IC synthesis and design is now necessary. When determining the impact of each decision in the synthesis or design process, the impacts of changed thermal profile on performance, power, price, and reliability must be considered. This requires repeated use of fast, accurate thermal analysis tools during synthesis.

The IC thermal analysis problem may be separated into two subproblems: steady-state (or static) analysis and dynamic analysis. Steady-state analysis determines the temperature profile to which an IC converges as time approaches infinity, given power and thermal conductivity profiles. Dynamic thermal analysis determines the temperature profile of an IC at any time given an initial temperature, power, heat capacity, and thermal conductivity profiles.

Numerical analysis techniques were also proposed to characterize the thermal profile of on-chip interconnect layers [3–5]. Recently, Skadron et al. developed steady-state and dynamic thermal analysis tools for microarchitectural evaluation [6]. Neither the matrix techniques of the steady-state analysis tool nor the lock-step fourth-order Runge-Kutta time marching technique used for dynamic analysis make use of spatial or asynchronous temporal adaptation; accuracy or performance suffer. Researchers have proposed quad-tree mesh refinement for thermal analysis [7], but did not consider local temporal adaptation. Li et al. proposed an efficient multigrid modeling technique to conduct full-chip steady-state thermal analysis [8]. Although the advantages of heterogeneous element discretization is noted, no systematic adaptation method is provided. Zhan and Sapatnekar [9] proposed a steady-state thermal analysis method based on Green's function that was accelerated by using discrete cosine transforms and look-up table. However, these methods [8], [9] do not support dynamic thermal analysis.

Existing IC thermal analysis tools are capable of providing either accuracy or speed, but not both. Accurate thermal analysis requires expensive computation for many elements in some regions, at some times. Conventional IC thermal analysis techniques ensure accuracy by choosing uniformly fine levels of detail across time and space, i.e., they use equivalent physical sizes or time step durations for all thermal elements. The large number of elements and time steps resulting from such techniques makes them computationally intensive and, therefore, impractical for use within IC synthesis. This article presents validated, synthesis-oriented IC thermal analysis techniques that differ from existing work by doing operation-by-operation dynamic adaptation of temporal and spatial resolution in order to dramatically reduce computational overhead without sacrificing accuracy. Experimental results indicate that the proposed spatial adaptation technique improves CPU time by 21.64–690.00× and that the temporal adaptation technique improves CPU time by 122.81–337.23×. Although much faster than conventional analysis techniques, the proposed techniques have been designed for accuracy even when this increases complexity and run time, e.g., by correctly modeling the dependence of thermal conductivity on temperature. These algorithms have been validated against FEMLAB, a reliable commercial finite element physical process modeling package, and a high-resolution spatially and temporally homogeneous initial value problem solver. Experimental results indicate that using existing thermal analysis techniques within IC synthesis flow would increase CPU time by many orders of magnitude, making it impractical to synthesize complex ICs. The proposed techniques make both dynamic and static thermal analysis practical within the inner loop of IC synthesis algorithms. They have been implemented as a software tool called ISAC that has been publicly released [10].

This article is organized as follows. Section 2 gives a motivating example, which illustrates the need for fast and accurate thermal analysis during IC synthesis and suggests techniques to reach this goal. Section 3 describes the model, algorithms, and implementation of ISAC, a fast and accurate steady-state and dynamic thermal analysis tool. Section 4 presents experimental results validating ISAC and demonstrating the dramatic performance advantages resulting from spatial and temporal adaptation during thermal analysis. Section 5 presents conclusions.

2. Motivating Examples

In this section, we use a thermal-aware IC synthesis flow to demonstrate the challenges of fast and accurate IC thermal modeling. Figure 1 shows an integrated behavioral-level and physical-level IC synthesis system [11]. This synthesis system uses a simulated annealing algorithm to jointly optimize several design metrics, including perfor-

This work is supported in part by the NSERC Discovery Grant #388694-01, and in part by the NSF under award CNS-0347941.

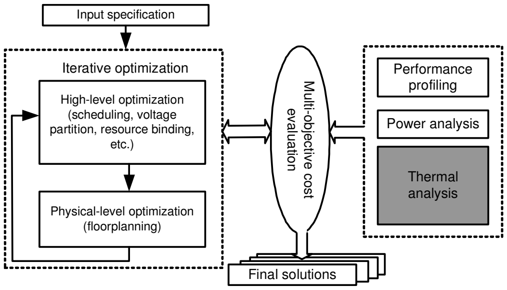
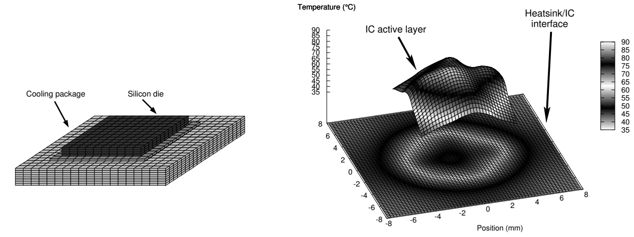

Input specification
Iterative optimization
High-level optimization (scheduling, voltage partition, resource binding, etc.)
Physical-level optimization (floorplanning)
Multi-objective cost evaluation
Performance profiling
Power analysis
Thermal analysis
Final solutions

Figure 1. Thermal-aware synthesis flow.



(a) Silicon chip and package. (b) Temperature profile for active layer and heatsink.

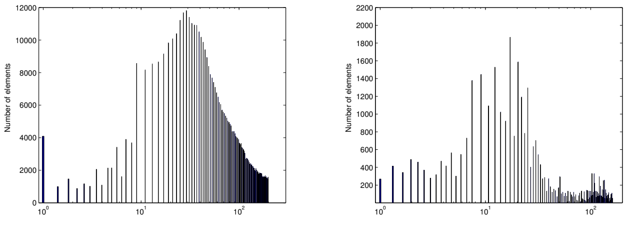
Figure 2. Thermal analysis during IC synthesis.

mance, area, power consumption, and peak IC temperature. It conducts both behavioral-level and physical-level stochastic optimization moves, including scheduling, voltage assignment, resource binding, floorplanning, etc. An intermediate solution is generated after each optimization move. A detailed two-dimensional power profile is then reported based on the physical floorplan. Thermal analysis algorithms are invoked to guide optimization moves.

As illustrated by the example synthesis flow for each intermediate solution, detailed thermal characterization requires full chip-package thermal modeling and analysis using numerical methods, which are computationally intensive. Figure 2 shows a full chip-package thermal modeling example from an IBM IC design (see Section 4.1 for more detail). The steady-state thermal profile of the active layer of the silicon die in conjunction with the top layer of the cooling package, shown in Figure 2(b), were characterized using a multigrid thermal solver by partitioning the chip and the cooling package into 131,072 homogeneous thermal elements. Without spatial and temporal adaptation, the solver requires many seconds or minutes when run on a high-performance workstation. Compared to steady-state thermal modeling, characterizing IC dynamic thermal profile is even more time consuming. IC synthesis requires a large number of optimization steps; thermal modeling can easily become its performance bottleneck.

A key challenge in thermal-aware IC synthesis is the development of fast and accurate thermal analysis techniques. Fundamentally, IC thermal modeling is the simulation of heat transfer from heat producers (transistors and interconnect), through silicon die and cooling package, to the ambient environment. This process is modeled with partial differential equations. In order to approximate the solutions of these equations using numerical methods, finite discretization is used, i.e., an IC model is decomposed into numerous three-dimensional elements. Adjacent elements interact via heat diffusion. Each element is sufficiently small to permit its temperature to be expressed as a difference equation, as a function of time, its material characteristics, its power dissipation, and the temperatures of its neighboring elements.

In an approach analogous to electric circuit analysis, thermal RC (or R) networks are constructed to perform dynamic (or steady-state) thermal analysis. Direct matrix operations, e.g., inversion, may be used for steady-state thermal analysis. However, the computational demand of this technique hinders its use within synthesis. Dynamic thermal analysis may be conducted by partitioning the simulation period into small time steps. The local times of all elements are then advanced, in lock-step, using transient temperature approximations yielded by difference equations. The computation complexity of dynamic thermal analysis is a function of the number of grid elements and time steps. Therefore, to improve the efficiency of thermal modeling, the key issue is to optimize the spatial and temporal modeling granularity, eliminating non-essential elements and stages.



(a) Inter-element thermal gradient (b) Normalized maximum step size

Figure 3. The potential of adaptive thermal modeling.

There is a tension between accuracy and efficiency when choosing modeling granularity. Increasing modeling granularity reduces analysis complexity but may also decrease accuracy. Uniform temperature is assumed within each thermal element. Intra-element thermal gradients are neglected. Therefore, increasing spatial modeling granularity naturally increases modeling errors. Similarly, increasing time step size may result in failure to capture transient thermal fluctuation or may increase truncation error when the actual temperature functions of some elements are of higher order than the difference equations used to approximate them.

IC thermal profiles contain significant spatial and temporal variation due to the heterogeneity of thermal conductivity and heat capacity in different materials, as well as varying power profiles resulting from non-uniform functional unit activities, placements, and schedules. Figure 3(a) shows the inter-element thermal gradient distribution using homogeneous meshing of the example shown in Figure 2. The histogram is normalized to the smallest inter-element thermal gradient. This figure contains a wide distribution of thermal gradients: heterogeneous spatial element discretization refinement based on thermal gradients has the potential to improve performance without impacting accuracy.

For dynamic thermal simulation, the size of each thermal element's time steps should permit accurate approximation by the element difference equations. An IC may experience different thermal fluctuations at different locations. Therefore, the best sizes of time steps for elements at different locations may vary. Figure 3(b) shows the maximum potential time step size of each individual block based local thermal variation; local adaptation of time step sizes has the potential to improve performance without impacting accuracy.

3. Thermal Analysis Model and Algorithms

This section gives details on the proposed thermal analysis techniques.

3.1. IC Thermal Analysis Problem Definition

IC thermal analysis is the simulation of heat transfer through heterogeneous material among heat producers (e.g., transistors) and heat consumers (e.g., heat sinks attached to IC packages). Modeling thermal conduction is analogous to modeling electrical conduction, with thermal conductivity corresponding to electrical conductivity, power dissipation corresponding to electrical current, heat capacity corresponding to electrical capacitance, and temperature corresponding to voltage.

The equation governing heat diffusion via thermal conduction in an IC follows.

$$\rho c\frac{\partial T(\vec{r},t)}{\partial t} = \bigtriangledown(k(\vec{r})\bigtriangledown T(\vec{r},t)) + p(\vec{r},t) \qquad (1)$$

In Equation 1, ρ is the material density; c_p is the mass heat capacity; $T(\vec{r},t)$ and $k(\vec{r})$ are the temperature and thermal conductivity of the material at position $\vec{r}$ and time t; and $p(\vec{r},t)$ is the power density of the heat source. Note that, in reality, the thermal conductivity, k, also depends on temperature (see Section 3.5). ISAC supports arbitrary heterogeneous thermal conduction models. For example, a model may be composed of a heat sink in a forced-air ambient environment, heat spreader, bulk silicon, active layer, and packaging material or any other geometry and combination of materials.

In order to do numerical thermal analysis, a seven point finite difference discretization method can be applied to the left and right side of Equation 1, i.e., the IC thermal behavior may be modeled by decom-

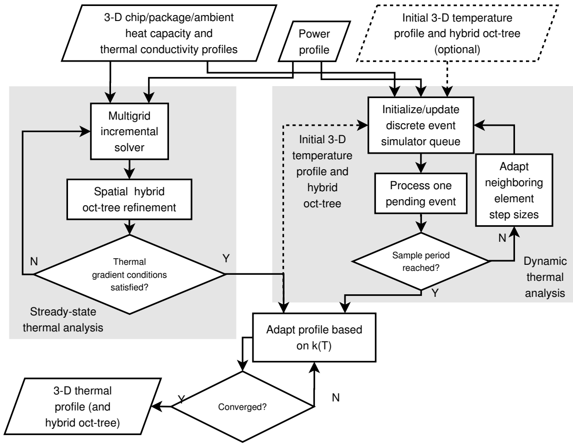

3-D chip/package/ambient heat capacity and thermal conductivity profiles
Power profile
Initial 3-D temperature profile and hybrid oct-tree (optional)
Multigrid incremental solver
Spatial hybrid oct-tree refinement
Thermal gradient conditions satisfied?
N
Y
Stready-state thermal analysis
Initialize/update discrete event simulator queue
Initial 3-D temperature profile and hybrid oct-tree
Process one pending event
Adapt neighboring element step sizes
Sample period reached?
N
Y
Dynamic thermal analysis
Adapt profile based on k(T)
Converged?
Y
N
3-D thermal profile (and hybrid oct-tree)

Figure 4. Overview of ISAC.

posing it into numerous cubic elements, which may be of non-uniform sizes. Adjacent elements interact via heat diffusion. Each element has a power dissipation, temperature, thermal capacitance, as well as a thermal resistance to adjacent elements. The discretized equation at an interior point of a homogeneous material follows.

$$\rho c V \frac{T^{m+1}_{i,j,k} - T^{m}_{i,j,k}}{\Delta t} = -2(G_x + G_y + G_z)T^{m}_{i,j,k} + G_x T^{m}_{i-1,j,k} + G_x T^{m}_{i+1,j,k} + G_y T^{m}_{i,j-1,k} + G_y T^{m}_{i,j+1,k} + G_z T^{m}_{i,j,k-1} + G_z T^{m}_{i,j,k+1} + V p_{i,j,k} \qquad (2)$$

Given that Δx, Δy, and Δz are discretization steps in dimension x, y and z, $V = \Delta x \Delta y \Delta z$. G_x, G_y and G_z are the thermal conductivities between adjacent elements. They are defined as follows: $G_x = k\Delta y \Delta z/\Delta x, G_y = k\Delta x \Delta z/\Delta y$, and $G_z = k\Delta x \Delta y/\Delta z$. Δt is the discretization step in time t. For steady-state analysis, the left term in Equation 2 expressing temperature variation as function of time, t, is dropped. For either the dynamic or steady-state version of the problem, the equations for all IC elements can be represented as a matrix. Although it is possible to directly solve this problem, the computational expense is prohibitive.

3.2. ISAC Overview

Figure 4 gives an overview of ISAC, our proposed incremental, self-adaptive, chip-package, thermal analysis tool. When used for steady-state thermal analysis, it takes, as input, a three-dimensional chip and package thermal conductivity profile, as well as a power dissipation profile. A multigrid incremental solver is used to progressively refine thermal element discretization to rapidly produce a temperature profile.

When used for dynamic thermal analysis, in addition to the input data required for steady-state analysis, ISAC requires the chip-package heat capacity profile. In addition, it may accept an initial temperature profile and efficient element grid. If these inputs are not provided, the dynamic analysis technique uses the steady-state analysis technique to produce its initial temperature profile and element grid. It then repeatedly updates the local temperatures and times of elements at asynchronous time steps, appropriately adapting the step sizes of neighbors to maintain accuracy.

As described in Section 3.5, after analysis is finished, the temperature profile is adapted using a feedback loop in which thermal conductivity is modified based upon temperature. Upon convergence, the temperature profile is reported to the IC synthesis tool or designer.

3.3. Spatial Adaptation in Thermal Analysis

In this section, we present an efficient technique for adapting thermal element spatial resolution for thermal analysis. This technique uses incremental refinement to generate a tree of heterogeneous parallelepipeds that supports fast thermal analysis without loss in accuracy. Within ISAC, this technique is incorporated with an efficient multigrid numerical analysis method, yielding a complete steady-state thermal analysis solution. Dynamic thermal analysis also benefits from the proposed spatial adaptation technique due to the dramatic reduction of the number of grid elements that must be considered during time marching simulation.

Algorithm 1 hybrid_tree_traversal($node_{root}$)

```
1:  if $node_{root}$ is a leaf node then
2:      Add $node_{root}$ to $contour_{finest\_level}$; return $finest\_level$
3:  end if
4:  for each intermediate child $chi\_node_i$ do
5:      $level_{chi\_node_i} = hybrid\_tree\_traversal(chi\_node_i)$
6:      $level_{min} = \min(level_{min}, level_{chi\_node_i})$
7:  end for
8:  for each intermediate child $chi\_node_i$ do
9:      if $level_{chi\_node_i} > level_{min}$ then
10:         Add $chi\_node_i$ to $contour_{chi\_node_i - 1}, \ldots, contour_{level_{min}}$
11:     end if
12: end for
13: Add $node_{root}$ to $contour_{level_{min}-1}$
14: return $level_{min}$-1
```

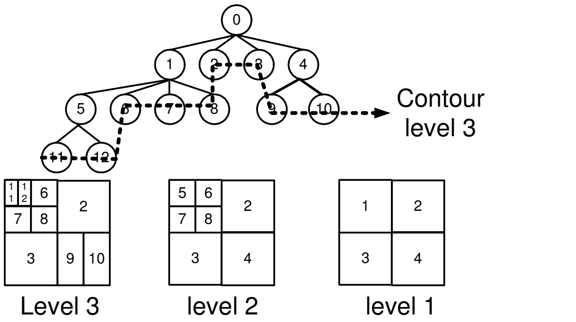


Figure 5. Heterogeneous spatial resolution adaptation.

3.3.1. Hybrid Data Structure. Efficient spatial adaptation in thermal analysis relies on sophisticated data structures, i.e., it requires the efficient organization of large data sets, representation of multi-level modeling resolutions, and inter-level transition. The proposed technique is supported by a hybrid oct-tree data structure, which provides an efficient and flexible representation to support spatial resolution adaptation. A hybrid oct-tree is a tree that maintains spatial relationships among parallelepipeds in three dimensions. Each node may have up to eight immediate children. Figure 5 shows a hybrid tree representation. For the sake of simplicity, a two-dimension quad-tree is shown instead of a three-dimension hybrid oct-tree. In the hybrid oct-tree, different modeling resolutions are organized into contours along the tree hierarchy, e.g., the contour formed by the leaf nodes represent the finest spatial resolution (in this example, elements 2,3,6,7,. . . ,12). Heterogeneous spatial resolution may result in a thermal element residing at multiple resolution levels, e.g., element 2 resides at level 1, 2, and 3. This information is represented as nodes existing in multiple contours in the tree.

Spatial resolution adaption requires two basic operations, partitioning and coarsening. In a hybrid oct-tree, partitioning is the process of breaking a leaf node along arbitrary orthogonal axes, e.g., nodes 9 and 10 result from refining node 4. Coarsening is the process of merging direct sub-nodes into their parent, e.g., node 11 and 12 merged into node 5. To conduct multi-resolution thermal analysis, we proposed an efficient contour search algorithm, with computational complexity $O(N)$, to determine thermal grid elements belonged to the same resolution level. As shown in Algorithm 1, leaf nodes are assigned to the finest resolution level (lines 1–3). The resolution level of a parent node of a subtree equals the minimal resolution level of all of its intermediate children nodes minus one (lines 4–7 and 13). An element may reside in multiple resolution levels (lines 8–12). As will be explained later, this algorithm provides an efficient solution to traverse different spatial resolutions, thereby supporting efficient multigrid thermal analysis.

3.3.2. Multigrid Method. Since directly solving the system of linear equations resulting from a large problem instance is intractable, more efficient numerical methods are used to solve the heat diffusion problem. The multigrid method is an iterative method of solving (typically sparse) systems of linear equations. It solves this problem by constructing a multi-level scheme, which greatly improves the efficiency of removing low frequency solution errors common for conventional iterative methods [12]. A description of this technique is shown in

Algorithm 2.

3.3.3. Incremental Analysis. Upon initialization, the steady-state thermal analysis tool generates a coarse homogeneous oct-tree based on the chip size. Iterative temperature approximation is repeated until convergence to a stable profile. Elements across which temperature varies by more than a user-specified threshold are further partitioned into sub-elements. For each ordered element pair, (i, j), given that T_i is the temperature of element i and that S is the temperature threshold, the new number of elements, Q, along some partition g follows.

$$Q = \lceil \log_2 (T_i - T_j / S) \rceil \tag{3}$$

For each element, i, partitions along three dimensions are gathered into a three-tuple (x_i, y_i, z_i) that governs partitioning element i into a hybrid sub oct-tree. The number of sub-elements depends on the ratio of the temperature difference to the threshold. Therefore, some elements may be further partitioned and local thermal simulation repeated. Simulation terminates when all element-to-element temperature differences are smaller than the predefined threshold, S. This method focuses computation on the most critical regions, increasing analysis speed while preserving accuracy.

3.4. Temporal Adaptation in Thermal Analysis

ISAC uses an adaptive time marching technique for dynamic thermal analysis. This technique is loosely related to the adaptive Runge-Kutta method [13] described in Section 2. The computational cost of a finite difference time marching technique is $\sum_{e \in E} u_e c_e$ where E is the set of all elements, u_e is the number of time steps for a given element, and c_e is the time cost per evaluation for that element. For Runge-Kutta methods, assuming a constant evaluation time and noting that all elements experience the same number of evaluations, run time can be expressed as $uc \sum_{e \in E} n_e$ where n is the number of a block's transitive neighbors. For these methods, element time synchronization permits evaluation amortization, eliminating the need to repeatedly evaluate transitive neighbors, yielding a time cost of $|E|uc$.

Analysis time is classically reduced by attacking u, either by using higher-order methods that allow larger steps under bounded error or by adapting global step size during analysis, e.g., the adaptive Runge-Kutta method. However, much greater gains are possible. As noted in Section 2, the requirement that all thermal elements be synchronized in time implies that, at each time step, all elements must have their local times advanced by the smallest step required by any element in the model. As indicated by Figure 3(b), this implies that most elements are forced to take unnecessarily small steps.

Although many time marching numerical methods for solving ordinary differential equations are based on methods that do not require explicit differentiation, these methods are conceptually based on repeated Taylor series expansions around increasing time instants. Revisiting these roots and basing time marching on Taylor series expansion allows element-by-element time step adaptation by supporting the extrapolation of temperatures at arbitrary times.

For many problems, the differentiation required for calculating Taylor series expansions is extremely complicated. Fortunately, for the dynamic IC thermal analysis problem, little more than the Laplace transform and linearity theorem are needed. Noting the definitions in Equation 2, and given that $T_n(t)$ is the temperature of element n at time t, G_{in} is the thermal conductivity between elements i and n, N_i are element i's neighbors, M is the neighbor depth, $\alpha_i = \sum_{n \in N_i} G_{in}$, and

$$\beta_i(t, M) = \begin{cases} \sum_{n \in N_i} T_n(t, M) \cdot G_{in} + V p_i & \text{if } M = 0 \\ V p_i & \text{otherwise} \end{cases} \tag{4}$$

the nearest-neighbor approximation of temperature of element i at time $t + h$ follows.

$$T_i(t + h, M) = \beta_i(t + h, M - 1)/\alpha_i + \frac{T_i(t) - \beta_i(M - 1)/\alpha_i}{e^{(h \cdot \alpha_i)/(\rho c_{pi} V)}} \tag{5}$$

under boundary conditions determined by the chip, package, and cooling solution.

Note that the potentially differing values of step size, h, and local time, t, for all thermal elements implies that the number of transitive temperature extrapolations necessary for an element to advance by one time step may not be amortized over multiple uses, as in the case in the lock-step Runge-Kutta methods. As a result, for three-dimensional thermal analysis, the number of evaluations, e, is related to the transitive neighbor count, d, as follows:

$$e = |E| (4/3d^3 + 2d^2 + 8/3d) \tag{6}$$

i.e., the discretized volume of the implied octahedron.

Algorithm 2 Multigrid cycle

```
Pre-smoothing step: Iteratively relax initial random solution. {HF error eliminated.}
subtask Coarse grid correction
    Compute residue from finer grid.
    Approximate residue in coarser grid.
    Solve coarseer grid problem using relaxation.
    if Coarsest level has been reached then
        Directly solve problem at this level.
    else
        Recursively apply the multigrid method.
    end if
    Map the correction back from the coarser to finer grid.
end subtask
Post smoothing step: Add correction to solution at finest grid level.
Iteratively relax to obtain the final solution.
```

In summary, although it is common to improve the performance of time marching techniques by increasing their orders, thereby increasing their step sizes, for the IC thermal analysis problem greater gains are possible by decoupling element local times, allowing most elements to take larger than minimum-sized steps. However, this requires explicit differentiation and prevents the amortization of neighbor temperature extrapolation, increasing the cost of using higher-order methods relative to that of using fully synchronized element time marching techniques. As demonstrated in Section 4, this trade-off is an excellent one: the third-order element-by-element adaptation method yields speed-ups ranging from 122.81–337.23× when compared to the fourth-order adaptive Runge-Kutta method.

We now describe the element-by-element step size adaptation methods used by ISAC to improve performance while preserving accuracy. As illustrated in the right portion of Figure 4, dynamic analysis starts with an initial three-dimensional temperature profile and hybrid oct-tree that may have been provided by the synthesis tool or generated by ISAC using steady-state analysis; a chip/package/ambient heat capacity and thermal conductivity profile; and a power profile. After determining the initial maximum safe step sizes of all elements, ISAC initializes an event queue of elements sorted by their target times, i.e., the element's current time plus its step size. The element with the earliest target time is selected, its temperature is updated, a new maximum safe step size is calculated for the element, and it is reinserted in the event queue. The event queue serves to minimize the deviation between decoupled element current times, thereby avoiding temperature extrapolation beyond the limits of the local time bounded-order expansions. The new step size must take into account the truncation error of the numerical method in use as well as the step sizes of the neighbors. Given that h_i is element i's current step size, v is the order of the time-marching numerical method, u is a constant slightly less than one, y is the error threshold, $dT_i/dt(t)$ is the derivative of i's temperature as a function of time at time t, and t_i is i's current time, the safe next step size for a block, regardless of its neighbors, follows.

$$s_i(t_i) = u \cdot \sqrt[v]{\frac{y}{\left| \frac{dT_i}{dt}(t_i) \cdot \frac{3}{2} \cdot h_i - \frac{3}{4} \cdot h_i \left(\frac{dT_i}{dt}(t_i) + \frac{dT_i}{dt}(t_i + \frac{3}{4} \cdot h_i) \right) \right|}} \tag{7}$$

This method of computing a new step size is based on the literature [14]. However, it uses non-integer test step sizes to bracket the most probable new step size.

It is necessary to further bound the step size to ensure that the local times of neighbors are sufficiently close for accurate temperature extrapolation. Given that N_i is the set of i's neighbors and w is a small constant, e.g., 3, the new step size follows.

$$h'_i = \min \left(s_i(t_i), \min_{n \in N_i} (w \cdot (t_n + h_n - t_i)) \right) \tag{8}$$

For efficiency, the h_n of a neighbor at its own local time is used.

This temporal adaptation technique based upon Equations 4, 5 and 8 is general, and has been tested in first-order, second-order, and third-order numerical methods. As indicated in Section 4.2, the result is a 122.81–337.23× speedup without loss of accuracy when compared to the fourth-order adaptive Runge-Kutta method.

3.5. Impact of Variable Thermal Conductivity

Thermal conductivity for a material is its ratio of heat flux density to temperature gradient. The thermal conductivity of a material, e.g., silicon, is a function of temperature, T. An ICs thermal conductivity, $k(\vec{r}, T)$, is also a function of position, $\vec{r}$. Most previous fast IC thermal analysis work ignores the dependence of thermal conductivity on temperature, approximating it as a constant. This introduces inaccuracy in analysis results. In contrast, ISAC models thermal conductivity as a function of temperature.

Position and temperature dependent thermal conductivity follows: $k = k_0 \cdot T/300^{-\alpha/C}$, where k_0 is the material's conductivity value at temperature 300 K, α is a constant for the specific material. Recalculating the thermal conductivity value after each iteration for all the elements would be computationally expensive. In order to maintain both accuracy and performance, ISAC uses a post-processing feedback loop to determine the impact of variations in thermal conductivity upon temperature profile. As described in Section 4.1, the consequences were over 5 K improvements in peak temperature accuracy when compared with a model assuming constant thermal conductivity.

3.6. The use of ISAC in IC Synthesis

As explained in Section 2, ISAC was developed primarily for use within IC synthesis, although it may also be used to provide guidance during manual architectural decisions. ISAC may be used to solve both the steady-state and dynamic thermal analysis problems described in Section 3.1. For use in steady-state analysis, ISAC requires three-dimensional chip-package profiles of thermal conductivity and power density. The required IC power profiles are typically produced by a floorplanner used within the synthesis process [11], [15], [16]. It produces a three-dimensional steady-state temperature profile. When used for dynamic thermal analysis, ISAC requires three-dimensional chip-package profiles of temperature, power density, heat capacity, (optionally) initial temperature, and an elapsed IC duration after which to report results. It produces a three-dimensional temperature profile at any requested time.

Both steady-state and dynamic thermal analysis solvers within ISAC have been accelerated, using the techniques described in Sections 3.3 and 3.4, in order to permit efficient use after each tentative change to an IC power profile during synthesis or design. Use within synthesis has been validated (see Section 4) by integrating ISAC within a behavioral synthesis algorithm [11].

4. Experimental Results

In this section, we validate and evaluate the performance of ISAC. Experiments were conducted on Linux workstations of similar performance. Evaluation focuses on accuracy and efficiency. ISAC supports both steady-state and dynamic thermal analysis. Steady-state thermal analysis is validated against FEMLAB, a widely-used commercial physics modeling package, using two actual chip designs from IBM and the MIT Raw group. Dynamic thermal simulation is validated against a fourth-order adaptive Runge-Kutta method using a set of synthesis benchmarks. Efficiency determines the feasibility of using thermal analysis during synthesis and design. To characterize the efficiency of ISAC, we compare it with other popular numerical analysis methods by conducting steady-state and dynamic thermal analysis on the power profiles produced during IC synthesis.

4.1. Steady-State Thermal Analysis Results

This section reports the accuracy and efficiency of the steady-state thermal simulation techniques used in ISAC. We first conduct the following experiments using two actual chip designs. The first IC is designed by IBM. The silicon die is 13 mm×13 mm×0.625 mm, which is soldered to a ceramic carrier using flip-chip packaging, and attached to a heat sink. A detailed 11×11 block static power profile was produced using a power simulator. The second IC is a chip-level multiprocessor designed by the MIT Raw group. This IC contains 16 on-chip MIPS processor cores organized in a 4×4 array. The die area is 18.2 mm×18.2 mm. It uses an IBM ceramic column grid array package with direct lid attach thermal enhancement. The static power profile is based on data provided in the literature [17]. We validate ISAC by comparing its results with those produced by FEMLAB, a widely-used commercial three-dimensional finite element based physics modeling package. Table 1 provides thermal analysis results produced by ISAC and FEMLAB for these ICs.

Average error, e_{avg} will be used as a measure of difference between thermal profiles:

$$e_{avg} = 1/N \sum_{i=0}^{N-1} \left| T_i - T_i' \right| / T_i \qquad (9)$$

where N is the total number of elements on the surface of the active layer of the silicon die modeled by ISAC. T_i and T_i' are the temperatures of element i reported by FEMLAB and ISAC, respectively. This is conservative. If comparisons were made in degrees Kelvin instead of degrees Celsius, the reported percentage error would be even lower.

In Table 1, the second and third columns show the peak and average temperatures of the surface of the active layer of the silicon dies of these chips, as reported by ISAC. Compared to FEMLAB, the average errors, e_{avg}, are 1.7% and 0.7%. The next four columns show the efficiency of ISAC in terms of CPU time, speedup, memory use, and number of elements. For comparison, the next three columns show the efficiency using a multigrid analysis technique with homogeneous meshing. These results clearly demonstrate that element resolution adaptation allows ISAC to achieve dramatic improvements in efficiency compared to the conventional multigrid technique. CPU times decrease to 3.6% and 0.14% and memory usage decreases to 5.6% and 2.4% of the times and memory required by the homogeneous technique. Note that multigrid steady-state analysis itself is a highly efficient approach [8]. Using FEMLAB, both simulations take at least 20 minutes.

Existing IC thermal simulators neglect the dependence of thermal conductivity on temperature, potentially resulting in substantial errors in peak temperature. In previous work, this error was not detected during validation because the models against which they were validated also used constant values for thermal conductivity. Temperature varies through the silicon die. Therefore, ignoring the dependence of thermal conductivity on temperature may introduce significant errors.

The last two columns of Table 1 show the peak and average temperatures, reported by FEMLAB, using thermal conductivities at 25 °C, i.e., room temperature. It shows that, for both chips, the peak temperatures are underestimated by approximately 5 °C. This effect will be even more serious in designs with higher peak temperatures. Note that the source of inaccuracy is not the specific value of thermal conductivity chosen. No constant value will allow accurate results in general: an accurate IC thermal model must consider the dependence of silicon thermal conductivity upon temperature.

To further evaluate its efficiency, we use ISAC to conduct thermal analysis for the behavioral synthesis algorithm described in Section 2. This iterative algorithm does both behavioral-level and physical-level optimization. In this experiment, ISAC performs steady-state thermal analysis for each intermediate solution generated during synthesis of ten commonly-used behavioral synthesis benchmarks.

Table 2 shows the performance of ISAC when used for steady-state thermal analysis during behavioral synthesis. The second, third, and fourth columns show the overall CPU time, speedup, and average memory used by ISAC to conduct steady-state thermal analysis for all the intermediate solutions. Column five shows the average error compared to a conventional homogeneous meshing multigrid method, whose overall CPU time and average memory use are shown in columns six and seven. ISAC achieves almost the same accuracy with much lower run-time overhead. The last column shows the CPU time used by the behavioral synthesis algorithm. Comparing column two and column seven makes it clear that, when used for steady-state thermal analysis, ISAC consumes only a fraction of the CPU time required for synthesis: it is feasible to use ISAC during synthesis.

4.2. Dynamic Thermal Analysis Results

In this section, we evaluate the accuracy and efficiency of the dynamic thermal analysis techniques used in ISAC. Heterogeneous spatial resolution adaptation was evaluated via steady-state thermal analysis. We will now focus on evaluating the proposed temporal adaption tech-

Table 1. Steady-state architectural thermal analysis evaluation

	ISAC							Multigrid_HM			Const. *k*	
Test cases	Peak	Average	Error	CPU	Speedup	Memory	Elements	CPU	Memory	Elements	Peak	Average
	temp. (°C)	temp. (°C)	(%)	time (s)	(×)	use (KB)		time (s)	use (KB)		temp. (°C)	temp. (°C)
IBM chip	90.7	54.8	1.7	0.08	27.50	252	1,800	2.2	4,506	32,768	85.2	53.8
MIT Raw	88.0	81.3	0.7	0.01	690.00	108	888	6.9	4,506	32,768	83.1	77.5

Table 2. Steady-state thermal analysis in IC synthesis

	ISAC				Multigrid_HM		HLS
Problem	CPU	Speedup	Mem.	Error	CPU	Mem.	CPU
	time (s)	(×)	(KB)	(%)	time (s)	(KB)	time (s)
chemical	0.78	53.06	265	0.35	41.39	4,506	40.02
dct_wang	2.52	37.08	264	0.24	93.43	4,506	301.37
dct_dif	2.40	37.63	266	1.50	90.31	4,506	71.60
dct_lee	6.10	27.64	268	0.50	168.60	4,506	132.15
elliptic	2.31	32.38	267	0.43	74.79	4,506	38.07
iir77	3.35	29.27	265	0.20	98.06	4,506	77.93
jcb_sm	1.63	21.64	277	0.13	35.27	4,506	151.95
mac	0.26	79.08	264	0.12	20.56	4,506	12.32
paulin	0.13	202.85	264	0.25	26.37	4,506	4.06
pr2	8.29	22.53	285	0.55	186.75	4,506	220.81

Table 3. Dynamic thermal analysis evaluation

	ISAC-2nd-order			ISAC			ARK	HLS
Problem	CPU	Error	Speedup	CPU	Error	Speedup	CPU	CPU
	time (s)	(%)	(×)	time (s)	(%)	(×)	time (s)	time (s)
chemical	79.80	0.01	135.74	48.19	0.02	224.75	10831.24	82.43
dct_wang	77.56	0.05	88.20	29.35	0.05	233.06	6840.83	110.24
dct_dif	104.47	0.03	71.02	57.67	0.02	128.65	7420.05	68.15
dct_lee	360.02	0.03	61.38	179.93	0.03	122.81	22097.77	90.51
elliptic	48.68	0.02	179.75	25.95	0.02	337.23	8750.10	80.79
irr77	97.82	0.02	111.00	48.87	0.02	222.15	10857.33	110.01
jcb_sm	73.19	0.05	108.42	32.00	0.04	247.98	7935.64	160.52
mac	19.80	0.01	97.50	14.48	0.01	133.30	1930.00	29.05
paulin	16.20	0.02	109.83	10.86	0.02	163.86	1779.10	7.73
pr2	82.65	0.02	106.82	34.08	0.03	259.04	8828.50	123.43

nique. We apply this technique to second-order (ISAC-2nd-order) and third-order (ISAC) numerical methods, which are then used to conduct dynamic thermal analysis on power profiles produced by our thermal-aware behavioral synthesis algorithm during optimization. Efficiency and accuracy are compared with a fourth-order adaptive Runge-Kutta method, which uses global temporal adaptation. The CPU time of ISAC is also compared to the CPU time for IC synthesis.

We use the same set of benchmarks described in the previous section. To generate dynamic power profile, on-line power analysis is conducted during synthesis using the switching activity model proposed in the literature [18]. In this model, input data with a Gaussian distribution are fed through an auto-regression filter to model the dependence of switching activity on operand bit position.

Table 3 shows the experimental results for dynamic thermal analysis. For each benchmark, column two and column five show the CPU time used by ISAC-2nd-order and ISAC to conduct dynamic thermal analysis for all the intermediate solutions generated by the behavioral synthesis algorithm. Column eight shows the CPU time used by the fourth-order adaptive Runge-Kutta method. In comparison, our techniques consistently speeds up analysis by at least two orders of magnitude, as indicated by column four and column seven. As shown in column three and column six, ISAC produces results that deviate from those of the adaptive fourth-order Runge-Kutta method by no more than 0.05%. The last column shows the CPU time used by the behavioral IC synthesis algorithm. As indicated in the table, it would be impractical to use the adaptive fourth-order Runge-Kutta method within IC synthesis due to its high computational overhead. The CPU times required by the proposed thermal analysis techniques are similar to those required by IC synthesis. Therefore, it is practical to incorporate them within IC synthesis.

5. Conclusions

This article has presented spatially and temporarily adaptive techniques for steady-state and dynamic thermal analysis during IC synthesis and design. The proposed techniques were used on a number of IC designs and synthesis test cases and validated against FEMLAB, a reliable commercial finite element physical process modeling package, and a high-resolution spatially and temporally homogeneous solver. Dynamic spatial and temporal adaptation result in 21.64–690.00× and 122.81–337.23× speedups, respectively, while preserving accuracy. Moreover, improvements in the underlying model, e.g., considering the dependence of thermal conductivity on temperature, have allowed accuracy improvements of 5 °C when compared with IC thermal models that neglect this dependence. The proposed techniques make accurate dynamic and static thermal analysis practical within the inner loops of IC synthesis algorithms. They have been implemented as a software tool called ISAC that has been publicly released [10].

References

[1] International Technology Roadmap for Semiconductors, http://public.itrs.net.

[2] J. Srinivasan, *et al.*, “The impact of technology scaling on lifetime reliability,” in *Proc. International Conf. Dependable Systems and Networks*, June 2004, pp. 177–186.

[3] T.-Y. Chiang, K. Banerjee, and K. C. Saraswat, “Analytical thermal model for multilevel VLSI interconnects incorporating via effect,” *IEEE Electron Device Letters*, vol. 23, no. 1, pp. 31–33, Jan. 2002.

[4] D. Chen, *et al.*, “Interconnect thermal modeling for accurate simulation of circuit timing and relability,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 2, pp. 197–205, Feb. 2000.

[5] Z. Lu, *et al.*, “Interconnect lifetime prediction under dynamic stress for reliability-aware design,” in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2004, pp. 327–334.

[6] K. Skadron, *et al.*, “Temperature-aware microarchitecture,” in *Proc. Int. Symp. Computer Architecture*, June 2003, pp. 2–13.

[7] T. Smy, D. Walkey, and S. Dew, “Transient 3D heat flow analysis for integrated circuit devices using the transmission line matrix method on a quad tree mesh,” *Solid-State Electronics*, vol. 45, no. 7, pp. 1137–1148, July 2001.

[8] P. Li, *et al.*, “Efficient full-chip thermal modeling and analysis,” in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2004, pp. 319–326.

[9] Y. Zhan and S. S. Sapatnekar, “A high efficiency full-chip thermal simulation algorithm,” in *Proc. Int. Conf. Computer-Aided Design*, Oct. 2005.

[10] “Incremental self-adaptive chip-package thermal analysis software,” ISAC link at http://www.ece.queensu.ca/hpages/faculty/shang/projects.html and http://www.eecs.northwestern.edu/~dickrp/projects.html.

[11] Z. P. Gu, *et al.*, “TAPHS: Thermal-aware unified physical-level and high-level synthesis,” in *Proc. Asia & South Pacific Design Automation Conf.*, Jan. 2006.

[12] W. Briggs, *A Multigrid Tutorial*. SIAM Press, 1987.

[13] S. S. Rao, *Applied Numerical Methods for Engineers and Scientists*. Prentice-Hall, Englewood Cliffs, NJ, 2002.

[14] W. H. Press, B. P. F. S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in FORTRAN: The Art of Scientific Computing*. Cambridge University Press, 1992.

[15] J. Cong and M. Sarrafzadeh, “Incremental physical design,” in *Proc. Int. Symp. Physical Design*, Apr. 2000.

[16] W. Choi and K. Bazargan, “Incremental placement for timing optimization,” in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2003.

[17] J. S. Kim, *et al.*, “Energy characterization of a tiled architecture processor with on-chip networks,” in *Proc. Int. Symp. Low Power Electronics & Design*, Aug. 2003, pp. 424–427.

[18] A. Raghunathan, N. K. Jha, and S. Dey, *High-level Power Analysis and Optimization*. Kluwer Academic Publishers, Boston, 1997.