Unified Temperature-Aware Incremental High-Level and Physical-Level Synthesis

Zhenyu (Peter) Gu* Yonghong Yang‡ Jia Wang† Li Shang‡ Robert P. Dick†

* Synplicity Inc
Sunnyvale, CA 94086, U.S.A.
zygu@synplicity.com † EECS Department

Northwestern University
Evanston, IL 60208, U.S.A.
dickrp@northwestern.edu
jwa112@eecs.northwestern.edu

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

***Abstract*—Thermal effects are becoming increasingly important during integrated circuit design. Thermal characteristics influence reliability, power consumption, cooling costs, and performance. It is necessary to consider thermal effects during all levels of the design process, from the architectural level to the physical level. This is challenging because design-time temperature prediction requires access to floorplans, wire models, power profile information, and a chip-package thermal model. Temperature-aware design and synthesis necessarily couple architectural-level design decisions (e.g., scheduling) with physical design (e.g., floorplanning), and modeling (e.g., wire and thermal modeling).**

This paper proposes an efficient and accurate temperature-aware high-level synthesis system that makes use of integrated high-level and physical-level design techniques. Voltage islands are automatically generated via slack distribution and voltage partitioning algorithms in order to reduce the design's power consumption and peak temperature. The proposed system was used to synthesize a number of benchmarks, yielding designs that trade off peak temperature, integrated circuit area, and power consumption. In comparison with an existing power-aware high-level synthesis algorithm, the proposed techniques reduce peak temperature by 12.5 °C on average. Under a constraint on peak temperature, integrated circuit area is reduced by 9.9% on average.

I. Introduction

Increasing performance requirements and system integration are dramatically increasing integrated circuit (IC) power density and cooling costs. Thermal issues are now central to the design of ICs, including both high-end instruction processors in general-purpose computers and high-performance application-specific integrated circuits (ASICs) in portable electronic consumer devices. Peak local temperature influences the reliability, packaging costs, cooling costs, bulk, and performance of ICs; these considerations can be particularly important for portable devices.

Increasing IC power consumption raises average and peak temperatures. Temperature-dependent reliability problems account for a significant proportion of electronic failures [1], most of which are due to electromigration, thermal stress, temperature-dependent dielectric breakdown, and negative bias temperature instability. Power and temperature variation can also lead to significant timing uncertainty, requiring more conservative timing margins, thereby reducing performance. Designers must frequently trade off other design metrics, such as performance, area, and cooling costs, to meet tight temperature constraints. The interaction of power and temperature constraints with other design metrics further increases system complexity. As projected by the International Technology Roadmap for Semiconductors (ITRS) [2], further process scaling will be bounded by power consumption and heat dissipation below 65 nm: it is critical to address the energy and thermal issues during on-chip system design to enable future technology scaling.

Thermal problems cannot be well solved at any single level of the design process. Thermal characterization therefore requires detailed physical information, including an IC floorplan, power profile, as well as interconnect and chip-package thermal models. Temperature optimization therefore requires a cross-layer high-level and physical-level design flow. At the architectural level, reducing supply voltage can reduce IC power consumption and temperature. At the physical level, efficient floorplanner is critical to correctly implement temperature-aware techniques made by high-level design flow while maintaining other design metrics, such as performance, chip area, and cooling cost. This requires a unified high-level and physical-level infrastructure.

Incremental synthesis is a promising design technique that may be used to unify high-level synthesis and physical design. It improves the quality of results by maintaining important physical-level properties across consecutive physical design changes, many of which are triggered by architectural changes [3]–[5]. Moreover, it dramatically improves synthesis time by reusing and building upon high-quality, previous physical design solutions that required a huge amount of time and effort to produce.

This paper presents an incremental, temperature-aware, voltage selection, floorplanning, high-level synthesis system called TAPHS. The proposed incremental synthesis techniques rapidly learn the impact of architectural changes on floorplan-dependent characteristics (e.g., peak temperature, interconnect structure, area, and energy consumption) and concurrently optimize IC thermal profile, area, and energy consumption under performance constraints.

II. Related Work

In this section, we survey related work in the two main research areas in which TAPHS is rooted: (1) high-level and physical-level co-synthesis and (2) temperature-aware analysis and design.

A number of researchers have considered the impact of physical details, e.g., floorplanning information, on behavioral synthesis [6]–[9]. Interconnect and interconnect buffers are now first-order timing and power considerations in VLSI design [10]. This change has complicated both design and synthesis. It is no longer possible to accurately estimate the power consumption and performance of a design without first knowing enough about its floorplan to predict the structure of its interconnect. For this reason, a number of researchers have worked on interconnect-aware behavioral synthesis algorithms [11]–[14]. These approaches typically use a loosely-coupled independent floorplanner for physical estimation. There are two drawbacks of this approach. First, the independent floorplanner may not be stable, i.e., a small change in the input netlist may result in a totally different floorplan. This results in a behavioral synthesis algorithm that bases its moves on cost functions without continuity.

This work is supported in part by SRC awards 2007-HJ-1593 and 2007-TJ-1589, NSF award CNS-0347941, and NSERC Discovery Grant #388694-01.

Second, even if the floorplanner is stable, creating a floorplan from scratch after each behavioral synthesis move is not efficient. The new floorplan typically only requires small changes to the previous one. Recently, incremental floorplanning and synthesis [4] were used to tightly couple high-level and physical synthesis, thereby dramatically improving their combined performance and quality [3].

Recent studies on thermal issues focused on thermal modeling, optimization, and run-time management. Full-chip thermal modeling and analysis during synthesis were rarely considered in the past due to the formidable computational demand. Recently, a number of IC thermal modeling approaches have been proposed [15]–[19]. Architecture-level thermal modeling and management techniques were proposed to improve the thermal characteristics of microprocessors [20] and on-chip networks [17]. With increasing power densities and reducing feature sizes, temperature-related reliability problems such as electromigration and stress migration voiding are becoming increasingly important. Recent studies [21]–[23] have proposed numerical and analytical modeling techniques to characterize the thermal profile of on-chip interconnect layers. Recently, thermal issues have also been considered during chip cell-level placement [24, 25], three-dimensional IC floorplanning [26], temperature-aware high-level synthesis by introducing area redundancy [27], and high-level temperature-aware resource binding and allocation [28]. Liu et al. [29] proposed an algorithm that solved the voltage partitioning problem under the constraint of a set of predefined voltages. The use of voltage islands is effective in reducing power consumption and therefore temperature. Wu et al. [30] proposed a novel approach to improve the voltage assignment by automatic outlier detection algorithm followed with incremental placement. Voltage island generation has recently been incorporated with physical design [31]–[33].

III. Motivating Example

In this section, we use an example IC design to demonstrate the challenges of temperature optimization in high-level synthesis. Figure 1 shows an IC floorplan produced by an integrated high-level synthesis and floorplanning algorithm. In this figure, the numbered rectangles are functional units, e.g., adders, multipliers, dividers, or registers. Using thermal analysis, as described in Section IX, the IC thermal profile is determined. The temperature of each functional unit is indicated by its brightness: brighter functional units are hotter. 85 °C is a typical thermal emergency threshold to ensure reliable operation. In this example, functional units temperatures higher than 85 °C are white. 29 of the functional units are operating at dangerously high temperatures: this chip is likely to suffer from failure caused by temperature-related reliability problems, e.g., electromigration or decreased charge carrier mobility. Note that producing the detailed chip thermal profile in Figure 1 requires detailed physical information, i.e., a floorplan, a power profile, and a chip-package thermal model. Therefore, stand-alone high-level synthesis algorithms have no means of detecting, let alone correcting, thermal problems.

High-level synthesis provides numerous temperature optimization opportunities. Reducing supply voltage reduces power consumption, hence temperature, but may also impair performance. Recent work on voltage islands has proposed operating different regions of an IC at different voltages. Figure 2 illustrates the floorplan of an IC using voltage islands. In this design, functional units are assigned to contiguous voltage islands with different supply voltages. The brightnesses of the thick functional unit boundaries indicate their voltages. In this example, three voltage islands are used. As in Figure 1, functional units violating the 85 °C temperature constraint are white.

A comparison of Figures 1 and 2 indicates that voltage islands can dramatically improve thermal conditions. The number of functional units with temperatures above the temperature constraint decreased from 29 to 19. However, as shown in Figure 2, localized hot spots still exist. The remaining hot spots are primarily the result of local peaks in power density. Therefore, thermal analysis algorithms are invoked to guide optimization moves in the high-level design.

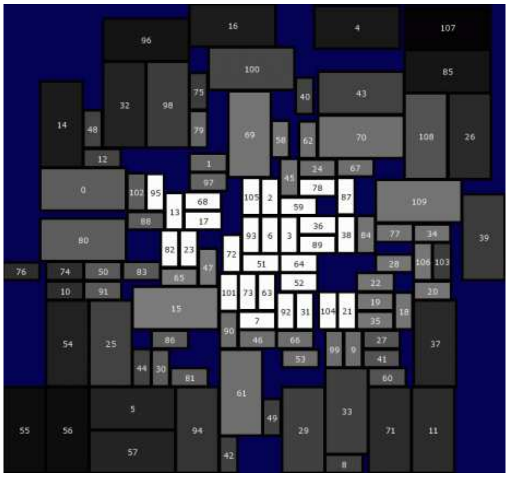


Fig. 1. Post-synthesis thermal profile without voltage islands.

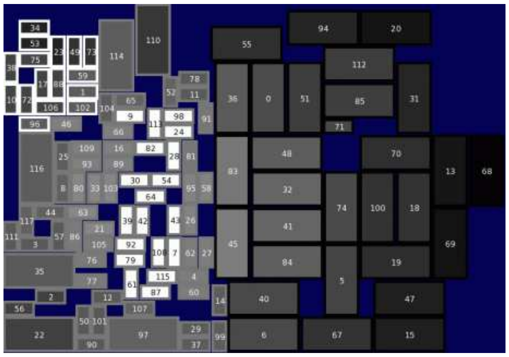


Fig. 2. Post-synthesis thermal profile with voltage islands.

Our study suggests using integrated high-level and physical-level temperature optimization techniques, including multiple operating voltages and appropriate scheduling. Many of the techniques to optimize IC thermal properties also impact other design metrics such as area and power consumption. We have considered the side effects of a number of techniques, proposing those that allow improvements to thermal properties while maintaining good design quality in terms of area, performance, and power consumption.

Using voltage islands has a significant impact on chip area and performance as well as increasing the complexity of floorplanning. Voltage islands require the addition of voltage converters and delivery circuits, as well as on-chip level shifters to support communication among functional units in different voltage islands. Moreover, reduced supply voltage requires a longer clock period to compensate for reduced switching speeds. In order to use voltage islands, a synthesis algorithm must wisely choose the island for each functional unit and appropriately allocate timing slack to allow scheduling in the high-level design. In physical-level design, high-quality incremental floorplanning is necessary to form voltage islands based on voltage assignment from high-level decision as well as maintaining other design objectives such as area, total wire length. This tightly couples the architectural and physical levels of design.

Facing these design challenges, a high-quality temperature-aware synthesis system must incorporate temperature optimization techniques into a unified high-level and physical level design flow, as

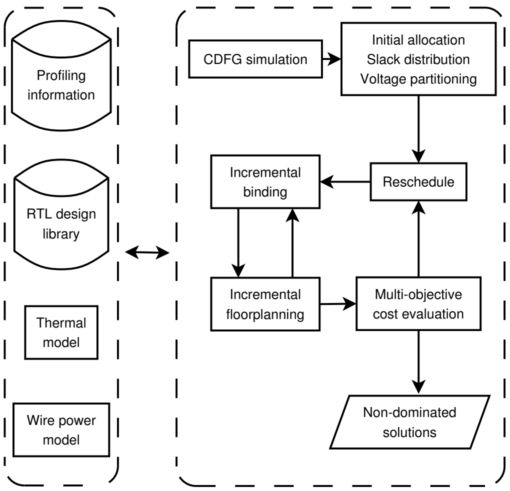


Fig. 3. Incremental high-level synthesis algorithm

well as striking wise tradeoffs among conflicting design goals.

IV. Overview of TAPHS

In this section, we give an overview of TAPHS: our incremental temperature-aware physical and high-level synthesis system. TAPHS considers the thermal impact of both logic and interconnect power dissipation. It automatically plans voltage islands and schedules operations to reduce IC power consumption and peak temperature.

Figure 3 illustrates the main algorithms used in TAPHS. First, the control and data flow graph is simulated with typical input traces in order to determine the power consumption of each operation and data transfer edge. The profile information, an RTL design library, floorplanner, and thermal model are used to evaluate the IC area, performance, power and temperature profile. Slack distribution, voltage clustering, and voltage island aware floorplanning are used to generate voltage islands for use in the initial solution: a fully parallel implementation. There are two loops within the high-level synthesis algorithm. In the outer loop, the clock period of the design is iteratively changed from the minimum to maximum potentially feasible value. Incremental rescheduling, resource sharing, resource splitting (i.e., the opposite of resource sharing), and slack distribution are used to generate valid solutions. In the inner loop, back-tracking iterative improvement is used to optimize the RTL architecture, considering multiple objectives, e.g., peak temperature, area, or power consumption. A *dominated* solution is inferior to some other previously encountered solution in all costs. Non-dominated solutions are preserved in a solution cache, from which the designer may choose based upon the desired, and available, trade-offs among costs.

A high-quality incremental floorplanner was developed [34] and incorporated into TAPHS. Each time the high-level synthesis algorithm needs temperature and physical information to guide its moves, it extracts that information from the current, incrementally-generated, floorplan. In addition, costs derived from the floorplan are also used to guide high-level synthesis moves. By using incremental floorplanning, closer interaction between high-level synthesis and physical design is possible, i.e., the high-level synthesis algorithm may determine the impact of potential changes to binding upon physical attributes such as maximum IC temperature, area, and interconnect energy consumption.

As shown in Section III, using multiple voltage islands can help reducing temperature. There are several works that mainly focus on the scheduling problem for voltage scaling [35]–[38]. Chen et al. [38] proposed a fast heuristic approach to predict the optimum dual-voltages by looking at the lower bound of the power consumption in the given circuit. However, this only works for two voltage levels. In the following sections, a more general approach will be proposed for multiple voltage levels. Since multiple voltage assignment and partitioning is a hard problem, it was split into two sub-problems: slack distribution and voltage partitioning.

V. Slack Distribution

In order to allow voltage scaling, it is necessary to appropriately distribute scheduling slack among different operations. An operation's *slack* is the difference between its latest start time and earliest start time. If operation start times are determined with an as-soon-as-possible schedule (ASAP), the executions of most operations will be immediately followed by other operations. As a result, it is impossible to assign an operation to a lower-voltage functional unit without violating timing constraints based on ASAP operation start times. TAPHS redistributes slack among operations in order to support a reduced energy assignment of functional units to voltage islands. Note that voltage assignment (described in Section VI) may not arrive at an energy-optimal solution, as it is necessary to constrain the off-chip overhead that would result from numerous power regulators. As shown in Figure 3, slack distribution occurs before voltage partitioning.

Assume that control data flow graphs have been partitioned into same-slack paths, as described later in this section. Given a single path composed of sequential operations, the slack distribution problem is equivalent to deciding the execution times of each operation such that energy consumption is minimized under a hard constraint on path execution time. We shall use the following variables and constants:

- d is the execution time of each functional unit;
- D is the bound on path execution time;
- p is the set of all operations on the path;
- v is the voltage of an operation's functional unit;
- V_t is the threshold voltage constant;
- K is an execution time constant;
- E is the total path energy consumption;
- e is the energy required for an operation;
- C is the switched capacitance constant of an operation's functional unit; and
- α is the alpha power law constant [39].

$$d = \frac{Kv}{(v - V_t)^{\alpha}} \tag{1}$$

A very low value of v will generally imply an unacceptable path delay that will be prevented by the constraint on Equation 7. Therefore, we may assume V_t is small, thus

$$d \simeq \frac{Kv}{v^{\alpha}} \tag{2}$$

$$v = \left(\frac{d}{K}\right)^{\frac{1}{1-\alpha}} \qquad \text{by (2)} \tag{3}$$

$$e = Cv^2 \tag{4}$$

$$e = c\left(\frac{d}{K}\right)^{\frac{2}{1-\alpha}} \qquad \text{by (2) and (4)} \tag{5}$$

$$E = \sum_{i \in p} C_i \left(\frac{d_i}{K_i}\right)^{\frac{2}{1-\alpha}} \tag{6}$$

$$\min_{\substack{\forall i \in p \\ v_i}} \quad \sum_{i \in p} C_i \left(\frac{d_i}{K_i}\right)^{\frac{2}{1-\alpha}} \quad \text{subject to the constraint } D \geq \sum_{i \in p} d_i \tag{7}$$

Note that a decrease in v implies an decrease e, which implies an increase in d. Therefore, for minimal E,

$$D = \sum_{i \in p} d_i \tag{8}$$

Consider the delay and energy trade-off for an arbitrary pair of operations:

$$d_{12} = d_1 + d_2 \tag{9}$$

$$e_{12} = e_1 + e_2 \tag{10}$$

$$e_{12} = \frac{C_1}{{K_1}^{\frac{2}{1-\alpha}}} \left(d_1\right)^{\frac{2}{1-\alpha}} + \frac{C_2}{{K_2}^{\frac{2}{1-\alpha}}} \left(d_{12} - d_1\right)^{\frac{2}{1-\alpha}} \tag{11}$$

Take the derivative of e_{12} with respect to d_1, set to zero, and solve to find d_2/d_1 for minimal E.

$$\frac{d_2}{d_1} = \left(\frac{\frac{C_1}{{K_1}^{\frac{2}{1-\alpha}}}}{\frac{C_2}{{K_2}^{\frac{2}{1-\alpha}}}} \right)^{\frac{1-\alpha}{1+\alpha}} \tag{12}$$

This optimal delay ratio for two operations may be used to compute the optimal delay ratio for an arbitrary pair of operations. These ratios can be scaled by a dynamically-computed value, N, to ensure that the constraint on line 7 is honored.

$$N = \sum_{i \in p} \frac{d_i}{d_1} \tag{13}$$

$$\forall_{i \in p}\ d_i = \left(\frac{\frac{C_1}{{K_1}^{\frac{2}{1-\alpha}}}}{\frac{C_i}{{K_i}^{\frac{2}{1-\alpha}}}} \right)^{\frac{1-\alpha}{1+\alpha}} \frac{D}{N} \qquad \text{by (11)} \tag{14}$$

$$\forall_{i \in p}\ d_i = \sqrt[3]{\frac{C_i {K_i}^2}{C_1 {K_1}^2}} \cdot \frac{D}{N} \qquad \text{by fixing } \alpha = 2 \tag{15}$$

Equations 11 and 15 yield the optimal time, d_i, to dedicate to each operation. By granting slack to each operation in the path such that its time is proportional to its time share, we allow the voltage island generation algorithm the opportunity to assign functional units to voltage islands such that energy consumption may be minimized under a hard constraint on path execution time (please see Section VI).

Thus far, we have discussed individual operation paths. However, it is necessary for TAPHS to determine slack distributions along numerous paths in arbitrary directed acyclic graphs of operations. Assigning time shares eventually has the effect of (temporarily) fixing operation start times. These start times may influence the earliest start times and latest finish times of operations on other paths; in order to avoid deadline violations, slack distribution is conducted on operation paths in order of increasing path slack. In order to generate paths, a modified depth-first search is conducted on a graph in which each vertex is an operation labeled with its slack and each edge is a data dependency. Vertex children are visited in the order of increasing slack, thereby guaranteeing that vertices on multiple paths will be included in minimal-slack paths.

As shown in Algorithm 1, starting from the minimal-slack path, TAPHS incrementally assigns extra clock cycles to operations. At each step, it locates the operation, j, for which the current allocated time, t_j, differs most from d_j (Step 8) and assigns it an additional clock cycle (Step 9). It is unlikely that this will result in deadline violations on other paths because slack distribution is carried out on paths in order to increase slack. However, an optimal algorithm would consider the graph structure instead of simplifying the problem by decomposing it into path-based subproblems. Therefore, slack distribution on a given path is prevented from delaying any node so much that slack becomes negative on other paths on which the node lies. After slack sharing is done for a given path, the slacks of all nodes are recomputed and slack distribution proceeds for the next path. The proposed slack distribution metric is optimal for continuous slack values. Although our high-level synthesis infrastructure supports multi-cycling, discretization to clock cycles must ultimately be done, potentially introducing suboptimality. The clock period is concurrently optimized, which may partially mitigate this effect. Recently, Ghiasi et al. [40] proposed a min-cost flow algorithm that optimally solves problem with linear cost functions. However, it is not applicable for this problem because the cost function is nonlinear.

Algorithm 1 Slack distribution procedure

```
 1: Compute all operation slacks
 2: Group operations into same-slack paths, $P$
 3: Sort paths $P$ in order of increasing slack
 4: for all $p \in P$ do
 5:     while slack remains on $p$ do
 6:         $\forall_{i \in p}\ t_i$ is the time assigned to operation $i$
 7:         Operation $i = \underset{j}{\arg\min} \sqrt[3]{\frac{C_i {K_i}^2}{C_1 {K_1}^2}} \cdot \frac{D}{N} - t_j$ by Equation 15
 8:         Assign one additional clock cycle to operation $i$
 9:     end while
10:     Recompute all operation slacks
11: end for
```

VI. Voltage Partitioning

TAPHS uses on-chip voltage islands to optimize IC thermal profiles and energy consumption. On-chip voltage islands are generated in two stages. *Voltage partitioning* classifies on-chip functional units into different voltage levels to maximize overall power and energy savings hence potential chip temperature reduction. *Voltage island generation* is then conducted via incremental floorplanning to produce and optimize on-chip voltage islands.

Given a set of functional unit minimum voltages, and a constraint on the maximum number of permissable voltages, voltage partitioning determines the voltages to be used and assigns each functional unit a single voltage. We propose an optimal voltage partitioning algorithm with time complexity in $\mathcal{O}\left(N^2\right)$ where N is the number of different functional unit minimum voltages. It conducts voltage allocation and assignment to minimize energy consumption subject to timing constraints. Note that this also minimizes power consumption because the execution period is fixed. The minimum voltages of functional units may come from a continuous range, or be discrete. In addition, an integer K is used to indicate the number of voltage levels that may be used in the synthesized IC. The algorithm determines K voltage levels, and assigns one of these K voltage levels to each functional unit.

First, let us define the optimal voltage partitioning problem.

Problem Definition *Given N functional units, $\{FU_i\}$, and an input K, find an optimal voltage partition, Ψ^K_{opt}, containing K voltage clusters, $\{{\psi_{opt}}_j\}_{j=1,\ldots,K}$, such that the total energy consumption is minimized, i.e.,*

$$\Psi^K_{opt} = \underset{\Psi^K \in \text{All possible partitions}}{\text{argmin}} E(\Psi^K) \tag{16}$$

where $E(\Psi^K) = 1/2 \sum_{l=1}^{N} C_l \times V^2_{\psi_j}$. C_l is the switching capacitance of FU_l, i.e., the product of average run-time switching activity and capacitance of this functional unit, $FU_l \in {\psi_{opt}}_j, j = 1, \ldots, K$.

For each functional unit, FU_l, the minimal allowed supply voltage, $V^{min}_{FU_l}$, is the minimum voltage that permits it to complete an operation within the time allotted by the slack distribution algorithm described in Section V. In general, the latency of a functional unit increases with the reduction of supply voltage. Then, in a voltage partition Ψ_i^S with S clusters, to satisfy the deadline constraints of functional units, for each cluster, $\psi_j = \{FU_{j1}, \ldots, FU_{jn}\}$, $\psi_j \in \Psi_i^S$, the supply voltage, V_{ψ_j}, is greater than or equal to $\max\{V_{FU_{jt}}^{\min}\}_{t=1,\ldots,n}$, i.e., the maximum of the minimal supply voltages of the functional units in this cluster.

We next present an example to illustrate the voltage partitioning problem. Consider a circuit design with five functional units as shown in Figure 4. Figure 4(a) shows one possible two-voltage-cluster partition, which contains two voltage clusters, $\psi_1 = \{FU_1, FU_2\}$ and $\psi_2 = \{FU_3, FU_4, FU_5\}$. The minimal supply voltage of ψ_1, $V_{\psi_1} = 1.0\,\mathrm{V} \geq \{0.8\,\mathrm{V}(\mathit{FU1}), 1.0\,\mathrm{V}(\mathit{FU2})\}$. The minimal allowed supply voltage of ψ_2, $V_{\psi_2} = 2.0\,\mathrm{V} \geq \{1.2\,\mathrm{V}(\mathit{FU3}), 1.5\,\mathrm{V}(\mathit{FU4}), 2.0\,\mathrm{V}(\mathit{FU5})\}$. Figure 4(c) shows the energy consumptions of different two-voltage-cluster partitions, which are derived using a linear scan of the voltage cut along the functional unit list. This list is sorted in order of increasing required minimal allowed supply voltage. This figure shows that the energy function is not monotonic in the cut location. Therefore, to find the two-voltage-cluster partition with minimal total energy consumption, an exhaustive search along the sorted functional unit list, i.e., a linear scan, is used. This leads to $\mathcal{O}(N)$ time complexity. Then, to find an optimal voltage partition with K voltage clusters, the exhaustive method leads to $\mathcal{O}\left(N^{K-1}\right)$ complexity, i.e., the complexity increases exponentially with the number of voltage clusters.

An optimal voltage partitioning algorithm of $\mathcal{O}\left(N^2\right)$ complexity:

We propose an optimal voltage partitioning algorithm of $\mathcal{O}\left(N^2\right)$ time complexity. Its pseudo-code is shown in Algorithm 2, which is described recursively. $\mathit{Partition}()$ has five input/output parameters. $\mathit{FU\_list}$ is a reference to the sorted functional unit list. Start and End designate the sub-list, the portion of the original list that must be partitioned. Initially, $\mathit{Start} = 0$ and $\mathit{End} = N$, indicating that the entire list must be partitioned. To yield K voltage clusters, M is the required number of partition cuts, and $M = K - 1$. $\mathit{OptTable}$ is an $M \times N$ look-up table, which stores intermediate optimal partitions of functional unit sub-lists, including the partitioning or cut location information as well as the corresponding energy consumption. More specifically, $\mathit{OptTable}[i][j]$ stores the optimal i-cut partition of the functional unit sub-list containing functional units from FU_i to FU_{END}.

$\mathit{Partition}()$ is invoked recursively when $M > 1$ (line 3–13). For each sub-partitioning (M cuts) on a sub-list (from Start to End), the optimal solution is derived using linear scan to examine the M^{th} cut, ⇒⇒**i.e. cut** M**,**⇐⇐ from Start to End, which is combined with the optimal solution of the sub-partitioning ($M - 1$ cuts) on its sub-list (from i to End) by accessing $\mathit{OptTable}$. When $M = 1$, the algorithm uses a linear scan to find the optimal cut in the targeted sub-list (line 16).

Next, we present Lemma 1, which guarantees the optimality of the algorithm. The algorithm uses combinational linear scans to explore all the possible partitioning combinations of the sorted list, including the optimal solution.

Lemma 1: *In an optimal partition Ψ_{opt}^K with K voltage clusters, $\{\psi_{opt,1}, \ldots, \psi_{opt,K}\}$, ordered by increasing voltage, $V_{\psi_{opt,1}} < V_{\psi_{opt,2}} < \cdots < V_{\psi_{opt,K}}$, the minimal allowed voltage of any functional unit FU_j in the optimal i_{th} voltage cluster is greater than the voltage level of adjacent lower-voltage cluster, i.e., $V_{\psi_{opt,i-1}} < V_{FU_j}^{min} \leq V_{\psi_{opt,i}}, \forall FU_j \in \psi_{opt,i}$, where $V_{FU_j}^{min}$ is the minimum allowed voltage for functional unit FU_j.*

Proof: Assume for the sake of contradiction, there exists a $FU_j \in \psi_{opt,i}$ such that $V_{\psi_{opt,i-1}} \geq V_{FU_j}^{min}$. If $V_{FU_j}^{min} = V_{\psi_{opt,i-1}}$, this contradicts the claim $V_{\psi_{opt,i}} > V_{\psi_{opt,i-1}}$. If $V_{FU_j}^{min} < V_{\psi_{opt,i-1}}$, then we can simply move FU_j to partition $\psi_{opt,i-1}$, which results in a lower energy consumption partition.

Algorithm 2 $\mathit{Partition}(*\mathit{FU\_list}, \mathit{Start}, \mathit{End}, M, *\mathit{OptTable})$

```
 1: check $OptTable[M][Start]$, and return if solved
 2: /* Recursive search the optimal $M$-cut partition */
 3: if $M > 1$ then
 4:     $C \leftarrow 0$
 5:     for $(i \leftarrow Start; i \leq End; i++)$ do
 6:         $Partition(*FU\_list, i, End, M - 1, *OptTable)$
 7:         $E^M_{M^{th}=i} \leftarrow C \times (V^{\min}_{FU_{i-1}})^2 + OptTable[M-1][i]$
 8:         $C += C_{FU_i}$
 9:     end for
10:     /* Determine the energy-optimal $M$-cut partition */
11:     $E^M_{opt}(Start, End) \leftarrow min\{E^M_{M^{th}=i}\}_{i \leftarrow Start,\ldots,End}$
12:     $cut^M_{opt}(Start, End) \leftarrow i$ if $E^M_{M^{th}\leftarrow i} = E^M_{opt}(Start, End)$
13:     $OptTable[M][Start] \leftarrow \mathrm{pair}(E^M_{opt}(Start, End), cut^M_{opt}(Start, End))$
14: else
15:     /* Linear scan to find the energy-optimal two-voltage-cluster
        partition */
16:     $Linear\_Scan(*FU\_list, End, \&E^2_{opt}(Start, End),$
        $\&cut^2_{opt}(Start, End))$
17:     /* Update $OptTable$ with the optimal two-voltage-cluster
        partition */
18:     $OptTable[1][Start] \leftarrow \mathrm{pair}(E^2_{opt}(Start, End),$
        $cut^2_{opt}(Start, End))$
19: end if
```

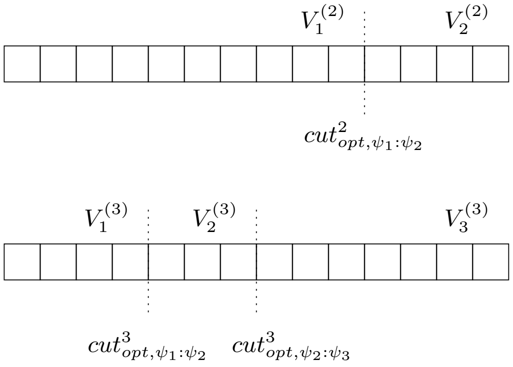

$V_1^{(2)}$
$V_2^{(2)}$
$cut^2_{opt,\psi_1:\psi_2}$
$V_1^{(3)}$
$V_2^{(3)}$
$V_3^{(3)}$
$cut^3_{opt,\psi_1:\psi_2}$
$cut^3_{opt,\psi_2:\psi_3}$

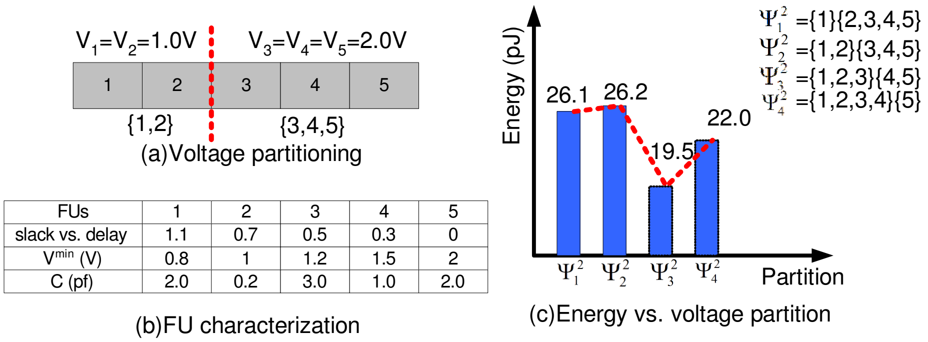
Fig. 5. Lemma 2 for optimal partitioning with $K = 3$ voltage clusters.

Lemma 1 implies that the optimal partitioning can be found by partitioning the sorted functional unit list. The time complexity of using a combinational linear scan to find the optimal K partitions on a sorted list with N functional units is $\mathcal{O}\left(N^{K-1}\right)$. To improve computation efficiency, as shown in the proposal Algorithm 2, we use a data structure, called $\mathit{OptTable}$, to store optimal sub-partitions. In this algorithm, there are a total of $\mathcal{O}(MN)$ table entries in $\mathit{OptTable}$, and each table entry requires an $\mathcal{O}(N)$ time linear search to find the optimal solution. Therefore the total running time is $\mathcal{O}\left(MN^2\right)$, or $\mathcal{O}\left(N^2\right)$ if M is bounded by a small constant.

Next, we will present Lemma 2, which is used to guarantee the optimality of Algorithm 2.

Lemma 2: In a sorted functional unit list with increasing minimum allowed voltages, if the cuts of an optimal partition Ψ_{opt}^K with K voltage clusters are $\{cut^K_{opt,\psi_1:\psi_2}, \ldots, cut^K_{opt,\psi_{K-1}:\psi_K}\}$, and the cuts of an optimal voltage partition Ψ_{opt}^{K+1} with $K+1$ voltage clusters are $\{cut^{K+1}_{opt,\psi_1:\psi_2}, \ldots, cut^{K+1}_{opt,\psi_K:\psi_{K+1}}\}$, then $cut^{K+1}_{opt,\psi_i:\psi_{i+1}}$ must be in the range between $cut^K_{opt,\psi_{i-1}:\psi_i}$ and $cut^K_{opt,\psi_i:\psi_{i+1}}$, i.e., $cut^K_{opt,\psi_{i-1}:\psi_i} \leq cut^{K+1}_{opt,\psi_i:\psi_{i+1}} \leq cut^K_{opt,\psi_i:\psi_{i+1}}$.

Proof: First, we will give the proof by contradiction for two cuts, i.e., from K=2 (two voltage clusters) to K=3 (three voltage clusters),



(a)Voltage partitioning

FUs	1	2	3	4	5
slack vs. delay	1.1	0.7	0.5	0.3	0
V^{min} (V)	0.8	1	1.2	1.5	2
C (pf)	2.0	0.2	3.0	1.0	2.0

(b)FU characterization

(c)Energy vs. voltage partition

Fig. 4. Voltage partitioning example.

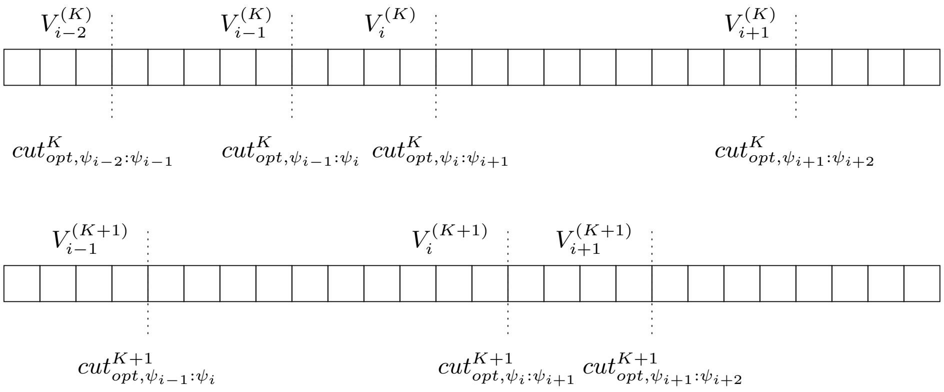

$V_{i-2}^{(K)}$
$V_{i-1}^{(K)}$
$V_{i}^{(K)}$
$V_{i+1}^{(K)}$
$cut^K_{opt,\psi_{i-2}:\psi_{i-1}}$
$cut^K_{opt,\psi_{i-1}:\psi_i}$
$cut^K_{opt,\psi_i:\psi_{i+1}}$
$cut^K_{opt,\psi_{i+1}:\psi_{i+2}}$
$V_{i-1}^{(K+1)}$
$V_{i}^{(K+1)}$
$V_{i+1}^{(K+1)}$
$cut^{K+1}_{opt,\psi_{i-1}:\psi_i}$
$cut^{K+1}_{opt,\psi_i:\psi_{i+1}}$
$cut^{K+1}_{opt,\psi_{i+1}:\psi_{i+2}}$

Fig. 6. Property 1 of Lemma 2 for optimal partitioning with $K+1$ voltage clusters.

then extend to general case. We will prove the two properties that guarantee the correctness of Lemma 2.

Given $cut^2_{opt,\psi_1:\psi_2}$ is the optimal partition, which generates two voltage clusters ψ_1 and ψ_2. The voltages of the two clusters are $V_1^{(2)}$ and $V_2^{(2)}$. Next, assume for K=3, both cuts, $cut^3_{opt,\psi_1:\psi_2}$ and $cut^3_{opt,\psi_2:\psi_3}$ of the optimal partition are to the left of $cut^2_{opt,\psi_1:\psi_2}$ (shown in Figure 5). The voltages of three voltage clusters are $V_1^{(3)}$, $V_2^{(3)}$, and $V_3^{(3)}$, with $V_1^{(3)} < V_2^{(3)} < V_3^{(3)}$. Notice that $V_2^{(2)} = V_3^{(3)}$. Let's tentatively move $cut^3_{opt,\psi_2:\psi_3}$ to the same place as optimal one-cut partition, $cut^2_{opt,\psi_1:\psi_2}$. We define E^3_{inc} as the energy increase and E^3_{dec} as the energy decrease due to this move. Set α includes the functional units between $cut^3_{opt,\psi_2:\psi_3}$ and $cut^3_{opt,\psi_1:\psi_2}$. Set β includes the functional units between $cut^3_{opt,\psi_2:\psi_3}$ and $cut^2_{opt,\psi_1:\psi_2}$. The resulting energy changes of the $K=3$ partition from moving $cut^3_{opt,\psi_2:\psi_3}$ follow:

$$E^3_{inc} = 1/2 \sum_{FU_i \in \alpha} C_i((V_1^{(2)})^2 - (V_2^{(3)})^2) \tag{17}$$

$$E^3_{dec} = 1/2 \sum_{FU_i \in \beta} C_i((V_3^{(3)})^2 - (V_1^{(2)})^2) \tag{18}$$

On the other hand, if we move $cut^2_{opt,\psi_1:\psi_2}$ to the same position as $cut^3_{opt,\psi_2:\psi_3}$, then the overall energy of the $K=2$ partition changes as follows:

$$E^2_{inc} = 1/2 \sum_{FU_i \in \beta} C_i((V_2^{(2)})^2 - (V_1^{(2)})^2) \tag{19}$$

$$E^2_{dec} = 1/2 \sum_{FU_i \in \kappa} C_i((V_1^{(2)})^2 - (V_2^{(3)})^2) \tag{20}$$

where set κ includes all the functional units in the left side of $cut^3_{opt,\psi_2:\psi_3}$.

Note that $cut^2_{opt,\psi_1:\psi_2}$ is the optimal two-partition cut. Therefore, $E^2_{inc} > E^2_{dec}$. $V_2^{(2)} = V_3^{(3)}$ and $\alpha \in \kappa$, therefore $E^3_{dec} = E^2_{inc} > E^2_{dec} > E^3_{inc}$. This means that moving $cut^3_{opt,\psi_2:\psi_3}$ to $cut^2_{opt,\psi_1:\psi_2}$ reduces energy consumption. Therefore, the current partition, i.e., placing both cuts to the left of $cut^2_{opt,\psi_1:\psi_2}$, is suboptimal. Similarly, we can prove that placing both cuts to the right of $cut^2_{opt,\psi_1:\psi_2}$ is also suboptimal.

Now, let's extend this proof to the general case of unbounded number of cuts (M) and partitions (K), and $M = K-1$. Given an optimal partitioning with K voltage clusters, an optimal partitioning of $K+1$ voltage clusters must have the following properties: (1) there must be at least one cut between $cut^K_{opt,\psi_{i-1}:\psi_i}$ and $cut^K_{opt,\psi_i:\psi_{i+1}}$; (2) if there are two cuts between $cut^K_{opt,\psi_{i-1}:\psi_i}$ and $cut^K_{opt,\psi_i:\psi_{i+1}}$, then there must be a cut between $cut^K_{opt,\psi_i:\psi_{i+1}}$ and $cut^K_{opt,\psi_{i+1}:\psi_{i+2}}$, and a cut between $cut^K_{opt,\psi_{i-2}:\psi_{i-1}}$ and $cut^K_{opt,\psi_{i-1}:\psi_i}$. The intuition behind these two properties is that they imply that optimal solutions follows the rule $cut^K_{opt,\psi_{i-1}:\psi_i} \leq cut^{K+1}_{opt,\psi_i:\psi_{i+1}} \leq cut^K_{opt,\psi_i:\psi_{i+1}}$ because if there exists a solution that satisfies Property 2, then it cannot meet the requirement for Property 1. This is due to the fact that the total number of cuts is only increased by 1 from the K partition case to $K+1$ partition case. Therefore, in an optimal solution, there exists one and only one cut between $cut^K_{opt,\psi_{i-1}:\psi_i}$ and $cut^K_{opt,\psi_i:\psi_{i+1}}$.

Property 1: Consider Figure 6. Given an optimal solution of the voltage partitioning problem with K voltage clusters, $(\{cut^K_{opt,\psi_1:\psi_2}, \ldots, cut^K_{opt,\psi_{K-1}:\psi_K}\})$ and an optimal solution of the voltage partitioning problem with $K+1$ voltage clusters $(\{cut^{K+1}_{opt,\psi_1:\psi_2}, \ldots, cut^{K+1}_{opt,\psi_K:\psi_{K+1}}\})$, assume that $cut^{K+1}_{opt,\psi_i:\psi_{i+1}}$ is to the right of $cut^K_{opt,\psi_i:\psi_{i+1}}$, i.e., no cut exists between position $cut^K_{opt,\psi_{i-1}:\psi_i}$ and $cut^K_{opt,\psi_i:\psi_{i+1}}$.

Here, set α includes the functional units between $cut^K_{opt,\psi_i:\psi_{i+1}}$ and $cut^{K+1}_{opt,\psi_i:\psi_{i+1}}$. Set β includes the functional units between $cut^{K+1}_{opt,\psi_{i-1}:\psi_i}$ and $cut^{K+1}_{opt,\psi_i:\psi_{i+1}}$. Moving $cut^{K+1}_{opt,\psi_i:\psi_{i+1}}$ left to the same position as $cut^K_{opt,\psi_i:\psi_{i+1}}$ results in the following energy changes:

$$E_{inc}^{K+1} = 1/2 \sum_{FU_i \in \alpha} C_i((V_{i+1}^{(K+1)})^2 - (V_i^{(K+1)})^2) \qquad (21)$$

$$E_{dec}^{K+1} = 1/2 \sum_{FU_i \in \beta} C_i((V_i^{(K+1)})^2 - (V_i^{(K)})^2) \qquad (22)$$

Moving $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ right to the same position as $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$ results in the following energy changes:

$$E_{inc}^{K} = 1/2 \sum_{FU_i \in \kappa} C_i((V_i^{(K+1)})^2 - (V_i^{(K)})^2) \qquad (23)$$

$$E_{dec}^{K} = 1/2 \sum_{FU_i \in \alpha} C_i((V_{i+1}^{(K)})^2 - (V_i^{(K+1)})^2) \qquad (24)$$

Here, set κ includes the functional units between $cut_{opt,\psi_{i-1}:\psi_i}^{K}$ and $cut_{opt,\psi_i:\psi_{i+1}}^{K}$. Since $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ is the optimal cut for partition with K voltage clusters. Hence, $E_{inc}^{K} > E_{dec}^{K}$. From Equations 21–24, we can see that $E_{dec}^{K+1} > E_{inc}^{K}$ since $\kappa \in \beta$. We also can get $E_{dec}^{K} > E_{inc}^{K+1}$ because $V_{i+1}^{(K)} > V_{i+1}^{(K+1)}$. Therefore, $E_{dec}^{K+1} > E_{inc}^{K} > E_{dec}^{K} > E_{inc}^{K+1}$. This contradicts the fact that the current partition with $K+1$ voltage clusters is optimal. Therefore, $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$ is to the left of $cut_{opt,\psi_i:\psi_{i+1}}^{K}$. In the same way, we can prove that $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$ is to the right of $cut_{opt,\psi_{i-1}:\psi_i}^{K}$. Therefore, $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ is between $cut_{opt,\psi_{i-1}:\psi_i}^{K}$ and $cut_{opt,\psi_i:\psi_{i+1}}^{K}$.

Property 2: Consider Figure 7. Given an optimal voltage partitioning solution with K voltage clusters ($\{cut_{opt,\psi_1:\psi_2}^{K}, \ldots, cut_{opt,\psi_{K-1}:\psi_K}^{K}\}$) and an optimal voltage partitioning solution with $K+1$ voltage clusters ($\{cut_{opt,\psi_1:\psi_2}^{K+1}, \ldots, cut_{opt,\psi_K:\psi_{K+1}}^{K}\}$), assume that $cut_{opt,\psi_{i-1}:\psi_i}^{K+1}$ and $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$ are between $cut_{opt,\psi_{i-1}:\psi_i}^{K}$ and $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ and there is no cut between $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ and $cut_{opt,\psi_{i+1}:\psi_{i+2}}^{K}$.

Moving $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$ to the same position as $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ results in the following energy changes:

$$E_{inc}^{K+1} = 1/2 \sum_{FU_i \in \beta} C_i((V_i^{(K)})^2 - (V_i^{(K+1)})^2) \qquad (25)$$

$$E_{dec}^{K+1} = 1/2 \sum_{FU_i \in \alpha} C_i((V_{i+1}^{(K+1)})^2 - (V_i^{(K)})^2) \qquad (26)$$

Here, set α includes the functional units between $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$ and $cut_{opt,\psi_i:\psi_{i+1}}^{K}$. Set β includes the functional units between $cut_{opt,\psi_{i-1}:\psi_i}^{K+1}$ and $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$.

Moving $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ to the same position as $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$ results in the following energy changes:

$$E_{inc}^{K} = 1/2 \sum_{FU_i \in \alpha} C_i((V_{i+1}^{(K)})^2 - (V_i^{(K)})^2) \qquad (27)$$

$$E_{dec}^{K} = 1/2 \sum_{FU_i \in \kappa} C_i((V_i^{(K)})^2 - (V_i^{(K+1)})^2) \qquad (28)$$

Here, set κ includes the functional units between $cut_{opt,\psi_{i-1}:\psi_i}^{K}$ and $cut_{opt,\psi_i:\psi_{i+1}}^{K+1}$. Since $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ is the optimal cut for partitions with K voltage clusters, therefore $E_{inc}^{K} > E_{dec}^{K}$. From above equation, we can see that $E_{dec}^{K+1} > E_{inc}^{K}$ because $V_{i+1}^{(K+1)} > V_{i+1}^{(K)}$. We also can get $E_{dec}^{K} > E_{inc}^{K+1}$ since $\beta \in \kappa$. Therefore, $E_{dec}^{K+1} > E_{inc}^{K+1}$. This contradicts the fact that the current partitioning solution with K voltage clusters is optimal. Therefore, there must be a cut between $cut_{opt,\psi_i:\psi_{i+1}}^{K}$ and $cut_{opt,\psi_{i+1}:\psi_{i+2}}^{K}$. In the same way, we can prove there must be a cut between $cut_{opt,\psi_{i-2}:\psi_{i-1}}^{K}$ and $cut_{opt,\psi_{i-1}:\psi_i}^{K}$.

VII. Floorplanning with Voltage Islands

In order to support temperature-aware, incremental, unified high-level and physical-level optimization, it was necessary to incorporate a high-quality, incremental floorplanner within TAPHS. New algorithms were developed and incorporated into this floorplanner to directly support physical-level temperature optimization and indirectly support architectural-level temperature optimization.

A. Floorplanner Representation and Cost Function

The floorplanner within TAPHS is based on the Adjacent Constraint Graph (ACG) representation [41]. An ACG is a constraint graph with exactly one geometric relationship between every pair of modules. ACGs have invariant structural properties that allow the number of edges in the graph to be bounded. Operations on ACGs have straightforward meanings in physical space and change graph topology locally; they require few, if any, global changes. The operations of removing and splitting modules are designed to reflect high-level operation to functional unit binding decisions. To obtain the physical position of each module, packing based on longest path computation is employed. Simulated annealing is used to obtain an initial floorplan. A weighted sum of the area and the interconnect power consumption is calculated for use as the floorplanner cost function, i.e.,

$$A + w \sum_{e \in E} C_e D_e \qquad (29)$$

where A is the area, w is the power consumption weight, E is the set of all wires, e is an interconnect wire, C_e is the unit-length switched capacitance for the data transfer along e, and D_e is the length of e, which is calculated as Manhattan distance between the two modules connected by the wire. Using this cost function, we optimize the interconnect power consumption, interconnect delay, and area of the floorplan. The resulting floorplan will be improved during the subsequent incremental floorplanning high-level synthesis moves. Therefore, the number of simulated annealing iterations is bounded to reduce synthesis time.

After each high-level synthesis move, the previous floorplan is modified by removing or splitting a module. The modules and switched capacitances are updated based upon the impact of these merges and splits. The floorplan is then re-optimized with a greedy iterative improvement algorithm using the same cost function as the simulated annealing algorithm. There are two reasons to use a greedy algorithm during this stage of synthesis: (1) re-optimization requires fewer global changes and less hill climbing and (2) perturbations resulting from high temperatures may disrupt high-quality floorplan structures.

After determining the best binding across all the possible values of *csteps*, another simulated annealing floorplanning run is used for that binding. This final floorplanning stage occurs only once for every synthesis run. Therefore, it is acceptable to use a slower, but higher-quality, annealing schedule than those in the inner loop of high-level synthesis, thereby improving integrated circuit area and interconnect power consumption.

During the annealing schedule, we use a constant multiplicative cooling factor, r, i.e.,

$$T^{+} = r \times T \qquad (30)$$

where T is the current temperature and $T+$ is the temperature during the next iteration. The number of the perturbations for the initial floorplanning run, the floorplanning for each value of *csteps* run, and the final floorplanning are related as follows: $1/2/20$. The number of perturbations per round for the greedy iterative improvement algorithm is the same as that for final floorplanning run.

B. Voltage Island Implementation in Floorplanning

As described in previous section, voltage island generation was introduced into the high-level synthesis system in order to improve

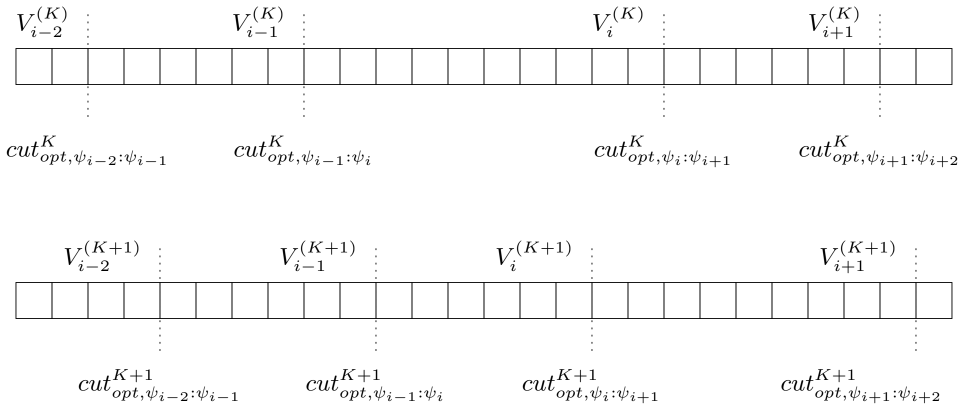

$V_{i-2}^{(K)}$
$V_{i-1}^{(K)}$
$V_{i}^{(K)}$
$V_{i+1}^{(K)}$
$cut^{K}_{opt,\psi_{i-2}:\psi_{i-1}}$
$cut^{K}_{opt,\psi_{i-1}:\psi_{i}}$
$cut^{K}_{opt,\psi_{i}:\psi_{i+1}}$
$cut^{K}_{opt,\psi_{i+1}:\psi_{i+2}}$
$V_{i-2}^{(K+1)}$
$V_{i-1}^{(K+1)}$
$V_{i}^{(K+1)}$
$V_{i+1}^{(K+1)}$
$cut^{K+1}_{opt,\psi_{i-2}:\psi_{i-1}}$
$cut^{K+1}_{opt,\psi_{i-1}:\psi_{i}}$
$cut^{K+1}_{opt,\psi_{i}:\psi_{i+1}}$
$cut^{K+1}_{opt,\psi_{i+1}:\psi_{i+2}}$

Fig. 7. Property 2 of Lemma 2 for optimal partitioning with $K+1$ voltage clusters.

thermal profiles and reduce energy consumption. Therefore, the floorplanner should keep functional units assigned to the same voltage level contiguous in order to minimize the need for level converters and simplify power distribution. The floorplanner must still honor the elements in the original cost function shown in Equation 29. Pair-wise weighted edges were added between all pairs of functional units operating at the same voltage, yielding the following updated cost function:

$$n\sqrt{A} + 2n\sum_{v\in V} L_v + \sum_{e\in E} C_e D_e \tag{31}$$

where A is the area, n is the number of functional units, V is the set of all functional unit pairs at the same voltage, v is a pair of functional units sharing the same voltage, L_v is the separation between a pair of functional units sharing the same voltage, E is the set of all interconnects, e, is an interconnect line, C_e is the unit-length switched capacitance for the data transfer along e (zero in the case of no communication), and D_e is the length of e. This approach generates contiguous voltage islands, as well as optimizing the interconnect power consumption and area.

Figure 2, described in Section III, shows an example of the results produced by this floorplanning algorithm. TAPHS rapidly generated this result using only pair-wise edges for functional unit clustering, i.e., hierarchical floorplanning was not required. Note that functional units operating at the same voltage are contiguous. In some cases, keeping voltage levels contiguous and minimizing wire length results in a slight area penalty. This is to be expected, regardless of the quality of a floorplanner, because it is rare for a minimal-area solution to have contiguous voltage levels and minimal interconnect power consumption. During incremental improvement, operation merging (functional unit resource sharing) combines functional units with other compatible functional units, always merging from the lower-voltage functional unit to the higher-voltage functional unit (please see Section IV).

VIII. Thermal Modeling

As mentioned in Section IV, thermal modeling and analysis are used in the inner loop of the optimization flow to provide direct guidance for temperature optimization. Therefore, our previous work, a compact chip-package thermal model [42], has been integrated into TAPHS to determine the thermal profile of our system.

The thermal model has been validated against the COMSOL Multiphysics software package (formerly FEMLAB) [43], an accurate but slow commercial finite-element based solver. It exhibited less than 2.5% estimation error when measured on the Kelvin scale. In the following experiments, each chip design is attached to a copper heat sink using forced-air cooling. We model two thermally conductive paths: heat dissipates from the silicon die through the cooling package to the ambient environment and through the package to the printed circuit board. We use an ambient temperature of 45 °C and a silicon thickness of 200 µm. In high-end microprocessor systems, due to the efficient cooling design, more than 80% of heat is dissipated through the first conductive path. In portable consumer electronics, due to the tight cooling budget and limited cooling space, the impact of the secondary conductive path may be significant.

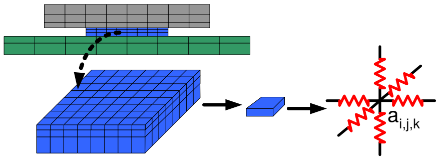


Fig. 8. Full chip-packaging thermal model.

Figure 8 illustrates the compact thermal model. Each material layer (e.g., silicon die, cooling package, and substrate carrier) is modeled with multiple layers of thermal elements. Each layer is partitioned into homogeneous thermal tiles, and each thermal tile is then modeled with inter-layer and intra-layer thermal resistors. Thermal resistances are determined based on material properties and tile geometries. Different layers use different tile granularity to strike a good trade-off between estimation accuracy and efficiency. To estimate the power consumption of each tile, we currently ignore the self-heating effect of on-chip interconnect, and only consider the power consumption of active devices [42]. Note that this work considers the impact of wire capacitance on the power consumption of drivers and repeaters. Based on the floorplanning information, we compute the power consumptions of thermal tiles using the average power dissipated by the functional units within each thermal tile. The power consumption of functional units with portions in multiple tiles is divided appropriately among the tiles. Instead of characterizing thermal profile from scratch after every incremental change to the power profile, the numerical thermal analysis method is initialized with the thermal profile associated with the previous power profile, thereby accelerating convergence after incremental changes to the power profile.

In general, chip thermal profile exhibits both temporal and spatial variations. Most of the benchmarks in this work have overall execution delays less than or comparable to the active layer element thermal time constant; therefore, temporal variation is negligible. Therefore, we focus on characterizing spatial thermal variation using steady-state thermal analysis. Chip thermal characteristics are estimated based on the chip power distribution averaged over the period or the deadline for periodic benchmarks and aperiodic benchmarks respectively.

IX. Experimental Results

In this section, we present experimental results for the TAPHS temperature-aware high-level synthesis system, including the temperature optimization techniques described in Sections V, VI, and VII. The circuits described in this section were synthesized using a register transfer level (RTL) design library [44] based on the TSMC 0.18 µm process. The experiments were conducted on AMD Athlon-based Linux workstations with 512 MB–1 GB of random access memory. No IC synthesis runs required more than 1,195 s of CPU time.

A. Benchmarks

We used TAPHS to synthesize 13 synthesis benchmarks. *Chemical* and *IIR77* are infinite impulse response (IIR) filters used for signal processing. *DCT_IJPEG* is the Independent JPEG Group's implementation of discrete cosine transform (DCT). *DCT_Wang* and *DCT_Lee* are DCT algorithms named after their inventors. All DCT algorithms work on 8 × 8 pixel of arrays. *Elliptic*, an elliptic wave filter, comes from the NCSU CBL (North Carolina State University Collaborative Benchmarking Laboratory) high-level synthesis benchmark suite [45]. *Jacobi* is the Jacobi iterative algorithm for solving a fourth-order linear system. *WDF* is a finite impulse response (FIR) wave digital filter. The largest benchmark, Jacobi, has 24 multiplications, 8 divisions, 8 additions, and 16 subtractions. In addition, we generated two large CDFGs using a pseudo-random graph generator [46]. Random100 has 20 additions, 15 subtractions, and 19 multiplications. Random200 has 39 additions, 44 subtractions, and 36 multiplications. The same sample periods (deadlines) were used for the benchmarks when evaluating each synthesis technique.

B. Multiobjective Results

Table I shows the results of doing full multiobjective optimization of peak temperature, area, and energy consumption with three voltage levels. In total, we compared 13 benchmarks. For each benchmark, the table shows non-dominated solutions produced by TAPHS. Due to space constraints, we sorted the solutions for each problem in order of increasing peak temperature and uniformly eliminated all but four solutions. For each solution, the left column indicates the name of the benchmark. The next three columns show the peak temperatures, areas, and power consumptions of solutions produced without using voltage islands. Area is reported as a percentage of the area of the an initial solution without resource sharing or voltage islands. The floorplanner typically has an area efficiency ranging from 75%–90% for these benchmarks. From these solutions, it should be clear that it is possible to trade off peak temperature for area as long as a thermal model is available during multiobjective synthesis. However, improving both objectives requires architectural-level and physical-level temperature optimization techniques.

The next three columns show the results produced using voltage islands. From these columns, it is clear that voltage islands yield significant improvements in peak temperature, area, and power consumption. For example, the peak temperatures of the lowest peak temperature solutions to each problem were reduced by an average of 12.5 °C.

Figure 9 shows only the lowest peak temperature for each benchmark after synthesis with voltage islands, and without voltage islands. This figure indicates that voltage islands can substantially reduce IC peak temperature, and that the relative contribution depends on the benchmark.

In addition, given the same area, TAPHS achieves lower peak temperatures for most benchmarks. For example, the peak temperature of *pr2* was reduced from 95.8 °C to 88.4 °C with the same area. Similar reductions were possible for *dct_dif*, *dct_ijpeg*, and *dct_lee*.

TABLE I
Comparison of non-dominated (multiobjective) results with three voltage levels

Example	No voltage islands			Voltage islands		
	Peak T (°C)	Area (%)	Power (W)	Peak T (°C)	Area (%)	Power (W)
chemical	123.4	116.6	2.18	98.0	142.4	1.60
	123.6	112.0	2.18	100.4	121.7	1.62
	123.7	109.3	2.18	103.3	112.7	1.59
	128.6	112.9	2.24	110.3	95.0	1.62
dct_dif	79.0	87.9	0.85	67.3	92.5	0.60
	79.7	78.6	0.83	67.6	81.5	0.58
	80.3	83.7	0.85	69.8	83.4	0.61
	80.1	81.4	0.84	69.3	74.9	0.57
dct_ijpeg	126.0	118.2	2.44	113.6	117.6	1.99
	129.4	107.2	2.39	115.8	114.9	2.03
	129.5	104.5	2.41	118.6	99.9	2.00
	130.6	104.7	2.40	118.9	102.0	2.03
dct_lee	71.5	98.9	0.79	63.7	106.4	0.59
	71.8	95.6	0.79	65.5	119.2	0.61
	71.9	99.9	0.79	64.6	106.3	0.59
	75.0	87.8	0.80	65.2	100.4	0.60
dct_wang	70.7	101.3	0.70	59.8	109.8	0.42
	68.2	97.5	0.68	59.1	116.0	0.43
	68.5	108.1	0.68	60.1	108.0	0.42
	70.4	89.1	0.70	59.8	102.8	0.44
elliptic	136.8	105.5	2.55	111.6	122.6	2.04
iir77	94.5	105.0	1.57	73.7	119.7	0.94
	97.7	93.1	1.56	74.6	115.7	0.94
	99.0	93.1	1.57	76.5	94.9	0.96
jacobi	54.2	64.4	0.25	51.8	81.5	0.20
	53.9	65.5	0.25	52.1	77.7	0.20
	53.8	63.2	0.24	52.9	69.2	0.21
	54.9	59.4	0.25	52.5	69.4	0.21
pr1	98.0	104.0	1.49	82.5	106.1	1.10
	97.4	103.1	1.52	84.8	92.6	1.10
pr2	95.4	103.8	1.67	87.9	110.2	1.44
	97.3	89.2	1.68	87.5	100.6	1.45
	95.8	98.4	1.67	88.0	105.4	1.45
	99.3	91.2	1.68	88.4	98.4	1.44
random100	71.6	100.0	0.85	66.0	98.8	0.63
	72.1	99.2	0.85	65.7	99.6	0.62
	72.7	99.7	0.86	67.6	85.1	0.67
	73.2	85.4	0.86	67.2	87.3	0.64
random200	90.8	90.2	1.77	81.4	112.0	1.37
	91.1	93.0	1.77	83.2	90.2	1.37
wdf	75.6	108.0	0.75	68.0	104.5	0.59
	74.8	96.9	0.73	67.8	101.8	0.59

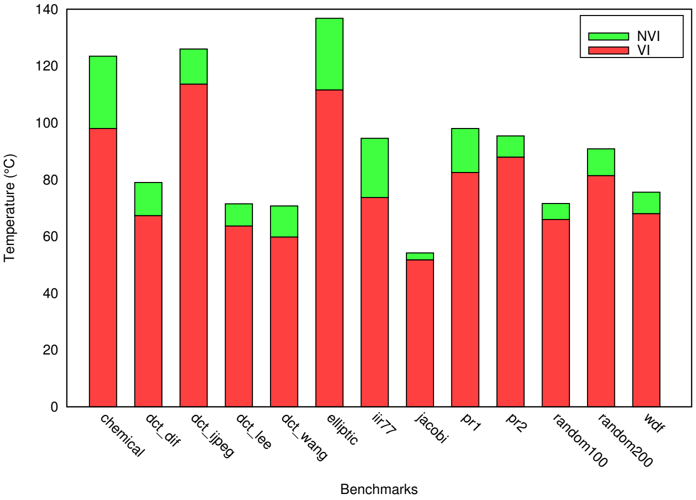


Fig. 9. Peak temperature comparison for three voltage levels case.

TABLE II
COMPARISON OF NON-DOMINATED (MULTIOBJECTIVE) RESULTS WITH TWO VOLTAGE LEVEL

	No voltage islands			Voltage islands		
Example	Peak T (°C)	Area (%)	Power (W)	Peak T (°C)	Area (%)	Power (W)
chemical	123.4	116.6	2.18	104.1	113.3	1.64
	123.6	112.0	2.18	104.5	114.5	1.65
	123.7	109.3	2.18	106.0	117.9	1.70
	128.6	112.9	2.24	107.1	106.0	1.68
dct_dif	79.0	87.9	0.85	71.5	81.2	0.62
	79.7	78.6	0.83	72.4	80.7	0.64
	80.3	83.7	0.85	71.9	77.7	0.60
	80.1	81.4	0.84	71.8	72.4	0.61
dct_ijpeg	126.0	118.2	2.44	115.6	104.7	2.02
	129.4	107.2	2.39	118.2	107.9	2.08
	129.5	104.5	2.41	119.4	101.7	2.06
dct_lee	71.5	98.9	0.79	63.5	125.7	0.61
	71.8	95.6	0.79	65.1	97.6	0.61
	71.9	99.9	0.79	64.8	107.3	0.62
	75.0	87.8	0.80	66.8	100.3	0.63
dct_wang	70.7	101.3	0.70	61.1	104.3	0.47
	68.2	97.5	0.68	60.2	96.2	0.46
	68.5	108.1	0.68	61.6	92.3	0.48
	70.4	89.1	0.70	60.5	102.5	0.49
elliptic	136.8	105.5	2.55	116.6	112.3	2.09
	139.7	89.4	2.51	116.7	106.9	2.09
	141.0	105.5	2.48	118.6	106.9	2.09
	142.8	92.4	2.51	120.7	111.1	2.09
iir77	94.5	105.0	1.57	77.5	102.4	1.01
jacobi	54.2	64.4	0.25	52.1	70.4	0.20
	53.9	65.5	0.25	52.9	68.2	0.20
	53.8	63.2	0.24	52.8	65.8	0.21
	54.9	59.4	0.25	52.9	63.8	0.20
pr1	98.0	104.0	1.49	86.3	106.0	1.25
	97.4	103.1	1.52	88.0	102.6	1.28
	97.9	98.3	1.50	88.9	98.5	1.27
	97.4	100.0	1.53	89.8	103.9	1.28
pr2	95.4	103.8	1.67	89.0	102.0	1.47
	97.3	89.2	1.68	89.7	95.7	1.47
	95.8	98.4	1.67	90.9	97.2	1.48
	99.3	91.2	1.68	91.3	99.0	1.47
random100	71.6	100.0	0.85	64.8	108.2	0.64
	72.1	99.2	0.85	66.0	96.3	0.64
	72.7	99.7	0.86	65.9	100.6	0.64
	73.2	85.4	0.86	66.7	91.2	0.65
random200	90.8	90.2	1.77	86.3	84.1	1.47
	91.1	93.0	1.77	85.6	82.8	1.48
	91.2	94.7	1.76	86.1	79.5	1.46
	91.3	91.8	1.77	86.2	88.3	1.47
wdf	75.6	108.0	0.75	71.1	95.1	0.63
	74.8	96.9	0.73	73.0	84.0	0.64
	76.0	93.2	0.72	73.2	86.5	0.64
	79.3	91.8	0.75	73.6	82.3	0.63

TABLE III
COMPARISON OF NON-DOMINATED (MULTIOBJECTIVE) RESULTS WITH FOUR VOLTAGE LEVEL

	No voltage islands			Voltage islands		
Example	Peak T (°C)	Area (%)	Power (W)	Peak T (°C)	Area (%)	Power (W)
chemical	123.4	116.6	2.18	96.9	131.5	1.55
	123.6	112.0	2.18	99.9	129.4	1.61
	123.7	109.3	2.18	101.8	116.1	1.56
	128.6	112.9	2.24	101.6	122.8	1.57
dct_dif	79.0	87.9	0.85	66.8	105.7	0.59
	79.7	78.6	0.83	67.5	91.0	0.56
	80.3	83.7	0.85	66.8	94.8	0.57
	80.1	81.4	0.84	67.0	84.6	0.55
dct_ijpeg	126.0	118.2	2.44	111.7	113.7	1.92
	129.4	107.2	2.39	115.8	116.4	2.05
	129.5	104.5	2.41	115.0	108.1	2.05
	130.6	104.7	2.40	116.7	102.4	1.96
dct_lee	71.5	98.9	0.79	62.9	109.8	0.56
	71.8	95.6	0.79	63.9	109.6	0.59
	71.9	99.9	0.79	63.9	111.6	0.58
	75.0	87.8	0.80	64.1	98.2	0.58
dct_wang	70.7	101.3	0.70	57.5	111.9	0.39
	68.2	97.5	0.68	57.6	112.3	0.40
elliptic	136.8	105.5	2.55	116.4	109.6	1.98
	139.7	89.4	2.51	115.5	104.8	1.98
	141.0	105.5	2.48	116.8	104.8	1.98
	142.8	92.4	2.51	121.9	93.8	1.98
iir77	94.5	105.0	1.57	74.4	112.6	0.91
jacobi	54.2	64.4	0.25	52.3	67.3	0.20
	53.9	65.5	0.25	52.1	69.6	0.20
	53.8	63.2	0.24	53.2	66.2	0.21
	54.9	59.4	0.25	52.8	67.0	0.21
pr1	98.0	104.0	1.49	80.3	107.6	1.07
	97.4	103.1	1.52	83.4	111.2	1.10
	97.9	98.3	1.50	84.1	101.4	1.08
pr2	95.4	103.8	1.67	86.7	106.9	1.44
	97.3	89.2	1.68	89.2	110.2	1.47
	95.8	98.4	1.67	88.4	99.7	1.44
	99.3	91.2	1.68	93.1	93.7	1.53
random100	71.6	100.0	0.85	63.3	104.5	0.60
	72.1	99.2	0.85	64.3	98.4	0.60
	72.7	99.7	0.86	64.8	106.2	0.60
	73.2	85.4	0.86	65.2	95.5	0.62
random200	90.8	90.2	1.77	80.5	94.4	1.35
	91.1	93.0	1.77	79.9	98.7	1.37
	91.2	94.7	1.76	80.7	94.9	1.38
	91.3	91.8	1.77	83.2	80.8	1.39
wdf	75.6	108.0	0.75	71.1	95.1	0.63
	74.8	96.9	0.73	73.0	84.0	0.64
	76.0	93.2	0.72	73.2	86.5	0.64
	79.3	91.8	0.75	73.6	82.3	0.63

In addition to reducing peak temperature, the proposed techniques can also be used to reduce area given a fixed peak temperature. When constraining temperature to the lowest temperature solution found without temperature optimization techniques, using voltage islands reduced area by, on average, 9.9%.

As shown in Table II and Table III, all the benchmarks were also run using two voltage levels and four voltage levels. The lowest peak temperatures were reduced by an average of 9.85 °C for two voltage levels, 12.5 °C for three voltage levels, and 12.83 °C for four voltage levels, compared to the results without voltage islands. These results indicate that temperature reduction is significant when moving from two voltage levels to three voltage levels and minor from three voltage levels to four voltage levels. As shown in Figure 10, we also found that due to the differing properties (such as benchmarks size, CDFG graph structure, etc.) of each benchmark, the number of voltage levels permitting maximum temperature reduction differs. In general, temperature reduces with an increasing number of voltage levels. However, changes in floorplanning prevent this trend from being entirely consistent.

C. Comparison with Liu's Algorithm

In their 2007 Design Automation Conference paper [29], Liu et al. proposed a voltage selection algorithm. This algorithm takes, as input, a small set of permissable voltages for each functional unit. These permissable voltages all come from the same small set of discrete voltages. The algorithm selects a subset of these voltages and assigns a voltage in this subset to each functional unit. They claim to have solved the same voltage selection problem considered in our conference paper [47]. This is false; functional units in our problem definition may have voltage constraints in a continuous range (see Section VI for a detailed problem definition). They also claim that the problem we solved is $\mathcal{NP}$-complete. This is false (see the polynomial-time optimal algorithm presented in Section VI).

Liu et al. claimed identical power consumption results for our algorithm and theirs. They ensured this result by constraining the

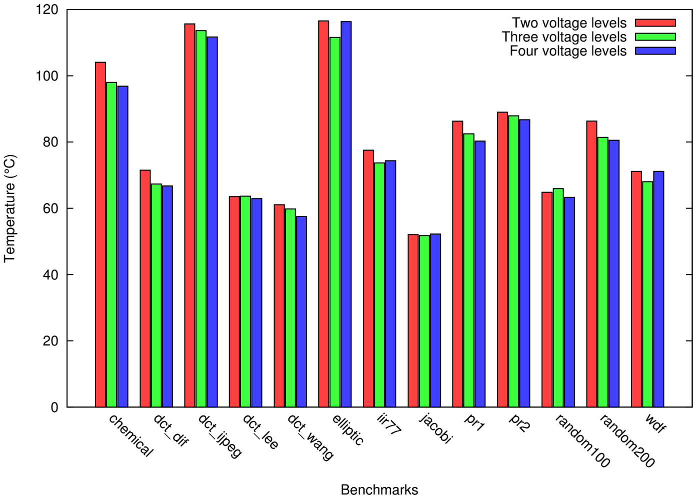


Fig. 10. Peak temperature reduction with different number of voltage levels.

maximum voltages of the functional units provided to our algorithm to a small set of discrete values. In order to determine the impact of their change in problem definition upon solution quality, we determined the change in power consumption resulting from constraining all functional unit maximum voltages to five values uniformly distributed from the minimum to the maximum voltages in our technology library. Three final voltage levels were permitted. Table IV shows the increase in power consumptions resulting from using the problem formulation proposed in Liu et al. Their problem formulation produces 6.46% higher power consumption, on average.

Liu et al. also indicated that they ran their algorithm on 10 pseudo-random benchmarks and that this was the same evaluation technique used in our conference paper. This is false; our conference paper presented results for 13 benchmarks, 11 of which were real (primarily signal processing) applications taken from previous publications. They also claimed an improvement in run time over the algorithm presented in our conference paper. In fact, if the number of functional unit voltage constraints were limited to a small set of voltages, the length of the arrays partitioned in Section VI should obviously be bounded by the number of discrete voltages (five, in Liu et al.'s publication). Our algorithm would also be very fast under those circumstances. However, we do not recommend such a problem definition due to the resulting degradation in solution quality. We indicated in person, to one of Liu's co-authors, the portions of their paper we believe to be misleading in the hope that it will be corrected.

X. Conclusions

In this paper, we have described TAPHS, a temperature-aware high-level synthesis system that uses a tightly-integrated thermal model and incremental floorplanner to optimize IC peak temperatures, areas, and power consumptions, while meeting performance constraints. In order to optimize peak temperature, it was necessary to tightly integrate floorplanning, wire modeling, power profile generation, and chip-package thermal analysis with high-level synthesis. Experimental results indicate that TAPHS is able to trade off peak temperature, IC area, and power consumption. The proposed techniques allowed a reduction in peak temperature of 12.5 °C, on average. We have also found that temperature optimization can allow significant improvements in IC area under temperature constraints. We conclude that it is important to incorporate temperature optimization in high-level synthesis to support continued increases in device and power density.

TABLE IV
Comparison with Liu's, Lee's, and Chang's algorithm

Example	Power of proposed algorithm (W)	Power of Liu's algorithm (W)	Improvement (%)
chemical	2.392	2.583	8.0
dct_dif	0.928	1.003	8.1
dct_ijpeg	2.857	2.857	0.0
dct_lee	0.905	0.985	8.9
dct_wang	0.580	0.580	0.0
elliptic	3.284	3.733	13.7
iir77	1.412	1.575	11.6
jacobi_sm	1.730	1.954	13.0
mac	2.792	3.144	12.6
paulin	1.050	1.051	0.1
pr1	1.638	1.873	14.4
pr2	2.268	2.293	1.1
random100	0.977	0.995	1.9
random200	1.966	1.966	0.0
random300	3.187	3.320	4.2
wdf	0.880	0.932	5.9

References

[1] L.-T. Yeh and R. C. Chu, *Thermal Management of Microelectronic Equipment: Heat Transfer Theory, Analysis Methods, and Design Practices*. New York, NY: ASME Press, 2002.

[2] "International Technology Roadmap for Semiconductors," 2006, http://public.itrs.net/.

[3] Z. P. Gu, et al., "Incremental exploration of the combined physical and behavioral design space," in *Proc. Design Automation Conf.*, June 2005, pp. 208–213.

[4] O. Coudert, et al., "Incremental CAD," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2000, pp. 236–244.

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

[6] D. W. Knapp, "Fasolt: A program for feedback-driven data-path optimization," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, no. 6, pp. 677–695, June 1992.

[7] J. P. Weng and A. C. Parker, "3D scheduling: High-level synthesis with floorplanning," in *Proc. Design Automation Conf.*, June 1992.

[8] Y. M. Fang and D. F. Wong, "Simultaneous functional-unit binding and floorplanning," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 1994.

[9] W. E. Dougherty and D. E. Thomas, "Unifying behavioral synthesis and physical design," in *Proc. Design Automation Conf.*, June 2000.

[10] J. Cong and Z. Pan, "Interconnect performance estimation models for design planning," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, pp. 739–752, June 2001.

[11] R. Mehra, L. M. Guerra, and J. M. Rabaey, "Low power architecture synthesis and impact of exploiting locality," *J. VLSI Signal Processing*, vol. 13, no. 8, pp. 877–888, Aug. 1996.

[12] P. Prabhakaran and P. Banerjee, "Simultaneous scheduling, binding and floorplanning high-level synthesis," in *Proc. Int. Conf. VLSI Design*, Jan. 1998.

[13] L. Zhong and N. K. Jha, "Interconnect-aware high-level synthesis for low power," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2002, pp. 110–117.

[14] A. Stammermann, et al., "Binding, allocation and floorplanning in low power high-level synthesis," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2003.

[15] Y. Cheng, et al., "ILLIADS-T: An electrothermal timing simulator for temperature-sensitive reliability diagnosis of CMOS VLSI chips," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 8, pp. 668–681, Aug. 1998.

[16] Z. Yu, et al., "Fast placement-dependent full chip thermal simulation," in *Proc. Int. Symp. VLSI Tech., Systems, & Applications*, Apr. 2001, pp. 249–252.

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

[18] H.-S. Wang, et al., “Orion: A power-performance simulator for interconnection networks,” in *Proc. Int. Symp. Microarchitecture*, Dec. 2002, pp. 294–305.

[19] X. Guo, et al., “The creation of compact thermal models of electronic components using model reduction,” in *Proc. Semiconductor Thermal Measurement & Management Symp.*, Mar. 2004, pp. 104–110.

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

[21] K. Banerjee, et al., “On thermal effects in deep sub-micron VLSI interconnects,” in *Proc. Design Automation Conf.*, June 1999, pp. 885–891.

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

[23] 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.

[24] C. Tsai and S. Kang, “Cell-level placement for improving substrate thermal distribution,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 2, pp. 253–266, Feb. 2000.

[25] B. Goplen and S. Sapatnekar, “Efficient thermal placement of standard cells in 3D ICs using a force directed approach,” in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2003, pp. 86–89.

[26] J. Cong, J. Wei, and Y. Zhang, “A thermal-driven floorplanning algorithm for 3D ICs,” in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2004, pp. 306–313.

[27] J.-P. Weng and A. C. Parker, “Taking thermal considerations into account during high-level synthesis,” *VLSI Design*, vol. 5, no. 2, pp. 183–193, 1997.

[28] R. Mukherjee, S. Öğrenci Memik, and G. Memik, “Temperature-aware resource allocation and binding in high-level synthesis,” in *Proc. Design Automation Conf.*, June 2005, pp. 196–201.

[29] H. Liu, W. Lee, and Y. Chang, “A provably good approximation algorithm for power optimization using multiple supply voltages,” in *Proc. Design Automation Conf.*, June 2007, pp. 887–890.

[30] H. Wu and M. D. F. Wong, “Improving Voltage Assignment by Outlier Detection and Incremental Placement,” in *Proc. Design Automation Conf.*, June 2007, pp. 459–464.

[31] Y. Cai, et al., “A thermal aware floorplanning algorithm supporting voltage island for low power soc design,” *Integrated Circuit and System Design*, Aug. 2005.

[32] W.-L. Hung, et al., “Temperature-aware voltage islands architecting in system-on-chip design,” in *Proc. Int. Conf. Computer Design*, Oct. 2005.

[33] R. L. Ching, et al., “Post-placement voltage island generation,” in *Proc. Int. Conf. Computer-Aided Design*, nov. 2006.

[34] Z. P. Gu, et al., “Unified incremental physical-level and high-level synthesis,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, Sept. 2007.

[35] S. Raje and M. Sarrafzadeh, “Variable voltage scheduling,” in *Proc. Int. Symp. Low Power Electronics & Design*, Aug. 1995, pp. 9–14.

[36] J. Chang and M. Pedram, “Energy minimization using multiple supply voltages,” in *Proc. Int. Symp. Low Power Electronics & Design*, Aug. 1996, pp. 157–162.

[37] K. Usami and M. Horowitz, “Cluster voltage scaling technique for low-power design,” in *Proc. Int. Symp. Low Power Electronics & Design*, Apr. 1995, pp. 3–8.

[38] C. Chen, A. Srivastava, and M. Sarrafzadeh, “On gate level power optimization using dual-Supply voltages,” *IEEE Trans. VLSI Systems*, vol. 9, no. 5, pp. 616–629, Oct. 2001.

[39] K. A. Bowman, et al., “A physical alpha-power law MOSFET model,” *IEEE J. Solid-State Circuits*, vol. 34, pp. 1410–1414, Oct. 1999.

[40] S. Ghiasi, et al., “A unified theory of timing budget management,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 11, pp. 2364–2375, Nov. 2006.

[41] H. Zhou and J. Wang, “ACG–Adjacent constraint graph for general floorplans,” in *Proc. Int. Conf. Computer Design*, Oct. 2004.

[42] L. Shang, et al., “Thermal modeling, characterization and management of on-chip networks,” in *Proc. Int. Symp. Microarchitecture*, Dec. 2004, pp. 67–80.

[43] COMSOL Multiphysics. COMSOL, Inc. http://www.comsol.com/products/multiphysics/.

[44] A. Raghunathan and N. K. Jha, “SCALP: An iterative-improvement-based low-power data path synthesis system,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, no. 11, pp. 1260–1277, Nov. 1997.

[45] “NCSU CBL high-level synthesis benchmark suite,” www.cbl.ncsu.edu/benchmarks.

[46] R. P. Dick, D. L. Rhodes, and W. Wolf, “TGFF: task graphs for free,” in *Proc. Int. Wkshp. Hardware/Software Co-Design*, Mar. 1998, pp. 97–101.

[47] 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, pp. 879–885.