N-Version Temperature-Aware Scheduling and Binding

Yousra Alkabani
CS Department
Rice University
Houston, TX
yousra@rice.edu

Farinaz Koushanfar
ECE Department
Rice University
Houston, TX
farinaz@rice.edu

Miodrag Potkonjak
CS Department
UCLA
Los Angeles, CA
miodrag@cs.ucla.edu

ABSTRACT

Technology scaling to nanometer nodes causes growing increase in power density and especially leakage that in turn result in locally hot regions on the chip. In this paper, we introduce a novel methodology for temperature-aware design. The methodology embeds N-versions of the scheduler and binder such that the thermal profiles of the versions are distant from each other. Next, instead of using only one version of the scheduler and binder, a rotation of N-versions of the scheduler and binder is constructed for balancing the thermal profile of the chip. We propose a linear programming framework that takes the multiple versions as the input, and constructs the thermal-aware rotational scheduling and binding by selecting the N most efficient versions and by determining the duration of each version. Our experimental evaluation shows a very low overhead and an average 5% decrease in the steady-state peak temperature produced on the benchmark designs compared to using a schedule that balances the amount of usage of different modules.

Categories and Subject Descriptors

B.6.3 [**Hardware**]: Logic Design—*Design Aids*

General Terms

Algorithms and Design

Keywords

Themperature control, High-level synthesis, and N-variants

1. INTRODUCTION

The intense feature scaling of CMOS has been driven by the growing application demands and pursuit of improved performance, as envisioned by Moore's law. Aggressive scaling lowers the cost-per-function, but it simultaneously escalates the device density and computational speed. The power density (i.e., power consumption per unit area) is also growing. The increased power generates heat on the chip. Since the heat propagation is slow compared with the switching activity of the IC, the heat would be concentrated at local regions, or so called *hotspots* [4, 8, 3]. The heat gradient increase would result in thermal stress that can speed up chip aging due to negative bias temperature instability, electromigration, or gradual dielectric breakdown. Therefore, circuit reliability would degrade.

The excessive increase in design complexity, power density, heat gradient, and unreliability of the scaled CMOS devices, has made thermal-aware design and optimization a strong research focus. In the ASICs domain, several solutions at different levels of design abstraction including scheduling, resource allocation, binding, floorplanning, and placement were developed [5, 16, 14, 11, 12, 10]. The common denominator for the existing work is that by assuming a certain model, they perform optimization (often iterative ones) that finds one optimized solution at the target level of the design abstraction.

In this paper, we introduce a paradigm shift by devising a flexible synthesis methodology that forms N-versions of the scheduling and binding solution each with unique thermal characteristics. The N-versions are simultaneously embedded into one design. During the operation, the versions would rotate such that each version would be used for a predefined time duration.

Our contributions are (1) Introduction of the concept of using rotational N-version scheduler and binder. (2) Designing an algorithm for finding the N-versions. (3) Methodology for construction and low-overhead implementation of the N-versions. (4) Development of a linear program that uses the thermal properties of each version to find the length and duration of N-versions. (5) Evaluation of the new method by comparison of the most balanced scheduler and binder.

2. RELATED WORK

In the past few years, a number of new approaches for thermal effect modeling and thermal-aware design has emerged. Banerjee et al. introduced a method for designing temperature and reliability aware cost trade-off with respect to power, performance and cooling [3]. Efficient modeling of the chip level and architecture level thermal characteristics have been proposed [4, 8]. The impact of thermal energy on power consumption was also studied [9]. Energy minimization has been addressed for a variety of systems including real-time [7], and under impact of manufacturing variability [2].

Chu and Wong proposed thermal-aware placement using a matrix synthesis method [5]. Tsai and Kang devised a standard cell placement tool for balancing the chip thermal distribution [16]. HotFloorplan is a floorplanning tool that manages the chip lateral heat propagation [14].

Mukherjee and Memik proposed a multistage integrated temperature optimization at the architectural synthesis lev-

ISLPED'09, August 19–21, 2009, San Francisco, California, USA.

els [11]. They develop an iterative optimization method for scheduling and binding that gets feedback the post-floorplan thermal simulations. Ni and Memik studied thermal-induced leakage power optimization by redundant resource allocation [12]. Lim and Kim formulated the thermal-aware binding problem into a problem of repeated utilization of network flow method [10]. Shang et al. explain the challenges the power and temperature optimization pose for high-level synthesis researchers and summarizes the research progress in the field [15]. Zhang and Chatha present approximation algorithms for temperature-aware scheduling; for a set of periodic tasks executing on a processor the latency is minimized subject to thermal constraints [17]. Note that there is also a vast body of literature for thermal-aware design for MPSoC and programmable devices that are outside the scope of this paper. To the best of our knowledge, this is the first work that considers combining multiple schedules and allocation for efficient thermal management at the high level synthesis.

3. FLOW

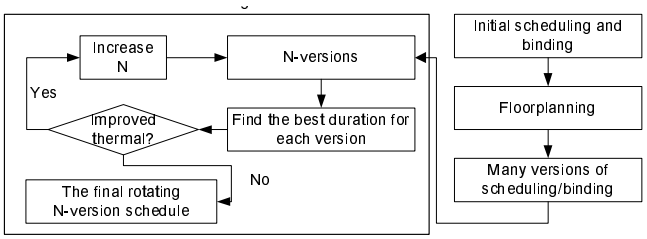

Figure 1: Flow of the rotational N-version thermal-aware scheduling and binding.

Figure 1 presents the rotational N-version thermal-aware scheduling and binding flow. The initial scheduling and binding is performed by list scheduling with a fixed timing constraint that minimizes the number of resources. A force-directed floorplanner is used after resource selection such that the similar resources are placed far from each other. This is to maximize the number of alternatives for a resource such that the temperature increase is independent among the alternatives. Finally, the maximally constrained minimally constraining rule is adopted for creation of many versions of the scheduler and binder. Next, the linear programming method finds N out of the several available scheduling and resource binding versions and their running durations, such that the selected versions have the smallest peak thermal energy dissipation.

4. N-VERSION SCHEDULING / BINDING

In this section, we describe how we generate the Multi-version schedules. Algorithm 1 shows the main steps for generating the N'-versions (N' $\geq$ N). The inputs to the Algorithm are G the CDFG of the circuit, and N' the number of generated versions. The output is the N' versions of the scheduling and binding method.

The first few steps generate a layout for different resources on a grid. In Step 2, we find the lower bound on each resource type denoted by R_t^{lb} using list scheduling. We do not change the critical path timing because in many of the DSP applications that we target, the throughput must remain constant. Next we choose the number of resources to be used for each module type (denoted by R_t in Step 3). R_t must be greater than or equal R_t^{lb} to guarantee that we do not alter the timing. In Step 4, we generate a force directed layout that ensures that resources of the same type are placed furthest away from each other. Finally, in Step 5, for each resource we determine its coordinates on the grid.

The remainder of the steps of the Algorithm 1 form N' versions of the scheduler and binder. For each version, we do both scheduling and binding using a maximally constrained minimally constraining heuristic described as follows. In Step 8, we select a center resource for each module type that is the most frequently used resource in the pertinent version. In line 10, we compute a priority pair (p_r, n_r) for each module. p_r is proportional to the distance from the center resource and n_r is the number of neighbors on the grid. The center resource of each type gets the highest priority ($p_r = 1$) to be used in a control step (maximally constrained). Further resources with larger number of neighbors have lower priority. We bind the operations to resources in order of priority. To break the ties among the resources with the same pr value, we use the number of neighbors of the resource n_r, where the resource with the smaller value of n_r has a higher priority (minimally constraining).

Alg. 1 Generation of N'-versions of the schedules.

Input: $CDFG, N'$
Output: S'_n

```
1 begin
2     Compute Rlb_t- using list scheduling;
3     Choose R_t such that R_t >= Rlb_t;
4     Generate layout grid for R_t;
5     Compute co-ordinates (x_t, y_t) for R_t on the grid;
6     for k = 1 : N'
7         for each resource type t;
8             Choose a center resource n_{c_t};
9             for each resource r ∈ R_t;
10                compute the priority pair (p_r, n_r)
11        time = 1;
12        ready = operations without predecessors;
13        while There exists an unscheduled operation;
14        begin
15            Schedule a maximum subset of ready
              Operations with least mobility are scheduled,
              Resources with min (p_i, n_i) are bound;
16            time = time + 1;
17            Add operations whose predecessors
              are done to ready;
18        end
19    end
```

5. ROTATIONAL N-VERSION METHOD

Thermal model. The compact thermal model that we use to predict the temperature rise due to using a certain version. The thermal energy of each module can be estimated by considering three different parameters: the power consumed at that module, the activity of the module in the schedule, and the exchange of energy with each of its neighboring modules.

We model the stationary state of the IC, where its produced thermal energy is equal to the energy that it transfers

to the environment. The two key assumptions are: (i) the chip is small relative to the environment and therefore, the environment does not change its temperature due to the heat conducted by the IC; and (ii) the rate of thermal change is much slower than the chip's clock frequency f_c and thus, one can consider the usage rate of one module in a certain scheduling round to be a good approximation of the cumulative impact of that module on the temperature.

The compact model is based on the Fourier conduction equations with constant thermal properties that is known to be a linear elliptic boundary value problem. Elliptic boundary value problems are a class of problems which do not involve the time variable, and instead only depend on space variables [6]. The thermal energy of the module at coordinate (i,j) is denoted by $Q_{i,j}$ respectively. Based on the Fourier conduction equations, the Thermal energy ($Q_{i,j}$) of the module (i,j) can be written as:

$$\begin{aligned} Q_{i,j} &= k_{Si/env} * A_{Si/env} * (T_{i,j} - T_{env}) \\ &+ k_{Si/Si} * A_{Si/Si} * (T_{i,j} - T_{i,j-1}) \\ &+ k_{Si/Si} * A_{Si/Si} * (T_{i,j} - T_{i,j+1}) \\ &+ k_{Si/Si} * A_{Si/Si} * (T_{i,j} - T_{i-1,j}) \\ &+ k_{Si/Si} * A_{Si/Si} * (T_{i,j} - T_{i+1,j}). \end{aligned} \quad (1)$$

Linear program for selection of the N-versions and their durations. We describe how we generate the rotational schedule using a linear program. The linear program takes as input the N'-versions that are constructed in the previous section. Next it selects the N-versions out of N' for embedding in the chip. The linear program also assigns a duration to each of the N selected final schedules.

The linear program is shown in Algorithm 2. The objective function (shown in Step 1) is to minimize the maximum temperature on the chip. This is followed by four types of constraints. Step 2 shows the first constraint type that represents local Newton heat laws. $Q_{i,j}$ and $T_{i,j}$ are variables representing the thermal energy generated by and the temperature of the resource at the coordinate (i,j) respectively.

The second type of constraints are shown in Step 3. This constraint represents the local thermal energy generation which is a function of the schedules. $P_{k,i,j}$ is a constant representing the average power generated by the resource at (i,j) in version k. p_k is a variable denoting the fraction of time this schedule is to be used. Step 4 shows the global constraints for the maximum temperature on the grid, where each resource temperature must not exceed the maximum temperature. Finally step 5, shows the total activity constraint that sets the sum of all the fractions p_k to 1. To have a low-overhead implementation for the rotational N-version method, we construct the FSM (F_r) of the rotational schedule from the FSM (F_1) of one of the versions. F_r has $logN$ extra inputs added to F_1 that are used as the key to select a version (I_{key}). The number of outputs of F_r are the same as the number of outputs of F_1. The construction is done similar to the method described in [1].

Alg. 2 The linear program to generate the rotational schedule.

Input: $grid$, N'-versions
Output: S_r

1 **Objective function**
min T_{max};
2 **Constraint type 1**
for each resource $r_{i,j}$ on $grid$
Satisfy equation 1;
3 **Constraint type 2**
for each resource $r_{i,j}$ on $grid$
$Q_{i,j} = p_1 P_{1,i,j} + p_2 P_{2,i,j} + ... + p'_N P_{N',i,j}$;
4 **Constraint type 3**
for each resource $r_{i,j}$ on $grid$
$T_i, j <= T_{max}$;
5 **Constraint type 4**
$p_1 + p_2 + ... + p'_N = 1$;

6. EXPERIMENTAL EVALUATIONS

We evaluate the rotational N-version method on different benchmarks from HYPER extracted from [13]. The benchmark names are shown in the second column of Table 1. The benchmarks lee, arai, and dir are 8 point fast discrete cosine algorithms with sharply different structures. Specifically, lee is Lee's recursive sparse matrix factorization algorithm, arai is Arai-Agui-Nakajima algorithm, and dir is the direct generic definition of DCT-I algorithm. feig is Feig's fast 2D 8x8 DCT with provably minimal number of multiplications. The benchmarks aircraft and honda are two industrial strength mechanical controllers.

The CDFG of the benchmarks are extracted in graphviz format. Matlab is used to read the CDFG, implement the multiple-version scheduling and binding and for the LP problem formulation and solving. We also generate HotSpot floorplan and power trace files of the schedules to evaluate the temperatures. We use the default values in HotSpot.

Table 1 shows the improvement in the maximum temperature when we use the minimum resources computed by the list scheduler. The comparison is made to the scheduling and binding method that most balances the usage of the modules. This balanced version is found by giving equal priorities to all the modules in Algorithm 1. The first column in the table represent the benchmark name. The second and third columns show the number of ALU operations (aop) Multiplication operations (mop) in the CDFG of the benchmark. The fourth and fifth columns demonstrate the lower bound on the number of ALUs ($A\#$) and Multipliers ($M\#$). The number of schedules selected by the linear program ($N\#$) is shown in the sixth column. The maximum temperature for the balanced scheduling and binding method (T_b), the maximum temperature in the rotational N-version (T_r) method (both in °C), as well as percentage improvement (I) in the Temperature are presented in the last three columns respectively. Temperatures are computed using HotSpot. The maximum improvement is above 11.7% and on the average the improvement is 4.9%.

Table 1: Max temp. improvement (min resources).

Name	aop	mop	A#	M#	N#	T_b(C)	T_r(C)	I%
arai	**39**	**5**	8	1	2	69.52	61.41	11.7
lee	**37**	**20**	4	4	2	64.99	62.57	3.7
honda	**70**	**34**	12	8	2	67.56	66.22	2.0
dir	**77**	**47**	11	11	3	64.22	61.45	4.3
aircraft	**147**	**127**	15	16	4	65.67	60.73	7.5
feig_dct	**505**	**78**	48	18	8	74.75	71.67	4.1

We next evaluate the performance of the rotational N-version method compared with the balanced version by adding resources. Table 2 shows the maximum temperature im-

provement when we use 10% extra resources. The first column shows the benchmark name. The second and third columns show the number of ALUs and Multipliers used in each benchmark. The fourth column shows the number of schedules produced by the linear program. The last three columns show the maximum temperatures in the balanced schedule and the rotational schedule in degree Celsius and the percentage improvement. The maximum improvement is about 19%, while the average is 7.4%.

Table 2: Max temp. improvement (add resources).

Name	A#	M#	N#	T_b(C)	T_r(C)	I(%)
arai	9	2	2	69.61	61.4	11.8
lee	5	5	3	63.57	61.22	3.7
honda	14	9	3	67.73	61.62	9.0
dir	13	13	2	68.25	62.05	9.1
aircraft	17	18	6	71.48	57.9	19.0
feig_dct	53	20	3	74.32	72.27	2.8

To study the overhead of embedding multiple schedules, we use ABC synthesis tool to estimate the area overhead of a single schedule and the rotational schedule. The area overhead for Table 1 is shown in Table 3. The first column shows the benchmark name. The second column shows the area in terms of the number of literals for the chip using a single schedule denoted by *orig*. The third column represents the area for the new schedule denoted by *new*. The maximum overhead is less than 5%. Note that benchmark 9 has zero overhead because it uses only one schedule. On the average the overhead is 1.3%. This shows the very low overhead of rotating among the versions. Note that the power overhead of the N-version method is proportional to its area overhead. The timing overhead is zero since all of the versions satisfy the timing constraint.

Table 3: Area overhead of the N-versions in Table 1.

Name	Orig (lit)	New (lit)	%
arai	99738	99738	0.2
lee	83369	83519	0.2
honda	211627	213597	0.9
dir	229201	239662	0.6
aircraft	322173	338011	4.9
feig_dct	712040	738329	3.7

7. CONCLUSION

We introduced a new temperature-aware scheduling and resource allocation method that combines N different versions of scheduling and binding. The combination was done by rotating between the N versions and by running each of the versions for a certain duration of time. Maximally constrained minimally constraining scheduling method was used for the efficient design and implementation of the multiple versions. We presented a linear programming formulation of the scheduling rotation that selects N out of many versions and determines the duration of each version. Evaluation of the method on standard benchmarks showed the low overhead of implementing the multiple versions in one design, and selection of the best value for N. Our experimental results shows that using the new method, an average of about 5% reduction in the peak temperature is obtained on the benchmarks in comparison with the scheduling and binding method where the usage of all resources are balanced.

8. REFERENCES

[1] Y. Alkabani and F. Koushanfar. N-variant IC design: methodology and applications. In *DAC*, pages 546–551, 2008.

[2] Y. Alkabani, T. Massey, F. Koushanfar, and M. Potkonjak. Input vector control for post-silicon leakage current minimization in the presence of manufacturing variability. In *DAC*, pages 606–609, 2008.

[3] K. Banerjee, S.-C. Lin, and V. Wason. Leakage and variation aware thermal management of nanometer scale ICs. In *IMAPS-Workshop*, 2004.

[4] Y. Cheng, P. Raha, C. Teng, E. Rosenbaum, and S. Kang. ILLIADS-T: an electrothermal timing simulator for temperature-sensitive reliability diagnosis of CMOS VLSI chips. *IEEE Trans. on CAD*, 17(8):668–681, 1998.

[5] C. Chu and D. Wong. A matrix synthesis approach to thermal placement. In *ISPD*, pages 163–168, 1997.

[6] L. Evans. *Partial Differential Equations*. American Mathematical Society, 1998.

[7] I. Hong, D. Kirovski, G. Qu, M. Potkonjak, and M. Srivastava. Power optimization of variable-voltage core-based systems. *IEEE Trans. on of Integrated Circuits and Systems*, 18(12):1702–1714, 1999.

[8] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. Stan. HotSpot: a compact thermal modeling methodology for early-stage VLSI design. *IEEE Trans. on VLSI*, 14(5):501–513, 2006.

[9] W. Liao, L. He, and K. Lepak. Temperature and supply voltage aware performance and power modeling at microarchitecture level. *IEEE Trans. on CAD*, 24(7):1042–1053, 2005.

[10] P. Lim and T. Kim. Thermal-aware high-level synthesis based on network flow method. In *CODES+ISSS*, pages 124–129, 2006.

[11] R. Mukherjee and S. Memik. An integrated approach to thermal management in high-level synthesis. *IEEE Trans. on VLSI*, 14(11):1165–1174, 2006.

[12] M. Ni and S. Memik. Thermal-induced leakage power optimization by redundant resource allocation. In *ICCAD*, pages 297–302, 2006.

[13] K. Rao and P. Yip. *Discrete Cosine Transform*. Academic Press, 1990.

[14] K. Sankaranarayanan, S. Velusamy, M. Stan, and K. Skadron. A case for thermal-aware floorplanning at the microarchitectural level. *Journal of Instruction-Level Parallelism*, (7), 2005.

[15] L. Shang, R. Dick, and N. Jha. *High-Level Synthesis Algorithms*, chapter High-Level Synthesis Algorithms for Power and Temperature Minimization, pages 285–297. Springer, 2008.

[16] C. Tsai and S. Kang. Standard cell placement for even on-chip thermal distribution. In *ISPD*, pages 179–184, 1999.

[17] S. Zhang and K. Chatha. Approximation algorithm for the temperature-aware scheduling problem. In *ICCAD*, pages 281–288, 2007.