Microarchitecture Floorplanning for Sub-threshold Leakage Reduction *

Hushrav D. Mogal and Kia Bazargan
University of Minnesota, Twin Cities
Minneapolis, MN 55414
{mhush,kia}@umn.edu

Abstract

Lateral heat conduction between modules affects the temperature profile of a floorplan, affecting the leakage power of individual blocks which increasingly is becoming a larger fraction of the overall power consumption with scaling of fabrication technologies. By modeling temperature dependent leakage power within a microarchitecture-aware floorplanning process, we propose a method that reduces sub-threshold leakage power. To that end, two leakage models are used: a transient formulation independent of any leakage power model and a simpler formulation derived from an empirical leakage power model, both showing good fidelity to detailed transient simulations. Our algorithm can reduce subthreshold leakage by upto 15% *with a minor degradation in performance, compared to a floorplanning process that does not model leakage. We also show the importance of modeling whitespace during floorplanning and its impact on leakage savings.*

1. Introduction and motivation

With scaling of technologies, leakage power takes up increasing fractions of the total power of a design [2]. As is predicted by the ITRS [13], this trend is further exacerbated in sub-100nm technologies due to increased device densities leading to a larger number of leaky devices and hence greater power densities. Exponential interdependence of subthreshold leakage on temperature can further cause problems like thermal runaway. These factors not only challenge circuit designers of mobile applications but also increase packaging costs.

Microarchitecture floorplanning has gained increased importance over the last few years due to its ability to exploit lateral conduction to reduce on-chip temperatures of the core blocks. In [12] the authors present HotFloorplan, a floorplanning tool to reduce the maximum on-chip temperature. This is done by modeling the chip as a network of thermal resistors using HotSpot [9], computing the steady state temperature of each block of the floorplan during the simulated annealing based floorplanning process and incorporating the maximum temperature of the floorplan in the cost function formulation. The authors in [8] perform a similar computation but instead of detailed thermal models use power densities as a measure of the temperature of the block. Although reducing maximum on-chip temperature does play an important role in reducing packaging costs, it does not directly tackle the problem of leakage reduction. This is because a larger module having a lower temperature than a smaller module having a higher temperature may dissipate more leakage power. This is very possible in current day microprocessors which employ large caches to improve performance. Also, the phenomenon of thermal runaway cannot be addressed directly using such a cost formulation. The authors in [16] explore different architectures via floorplanning to achieve a given performance objective with certain thermal constraints. In [5] the authors present a clustered microarchitecture wherein instruction scheduling is applied to control the average temperature of the chip thereby also targeting leakage reduction. Although all these works use temperature in their floorplanning objective, they do not employ accurate models of subthreshold leakage dependence on temperature which could pose a serious limitation given the importance of leakage power dissipation in future technologies.

This work attempts to reduce the subthreshold leakage power (hereby referred to as leakage) considering its exponential dependence on temperature by microarchitecture floorplanning. Our main contribution is in deriving a formulation for capturing the rise in leakage power of a given floorplan thereby obviating the need to perform time consuming transient simulations during the floorplanning process. Our leakage formulation is independent of any model used to describe leakage power dissipation in a transistor. We call this the **transient** model. A simpler formulation dependent on the leakage power model is also explored and is shown to have good fidelity with detailed simulations for a given floorplan. This is called the **simple** model. Lastly, we model whitespace in the floorplan to analyze its impact on reducing leakage power.

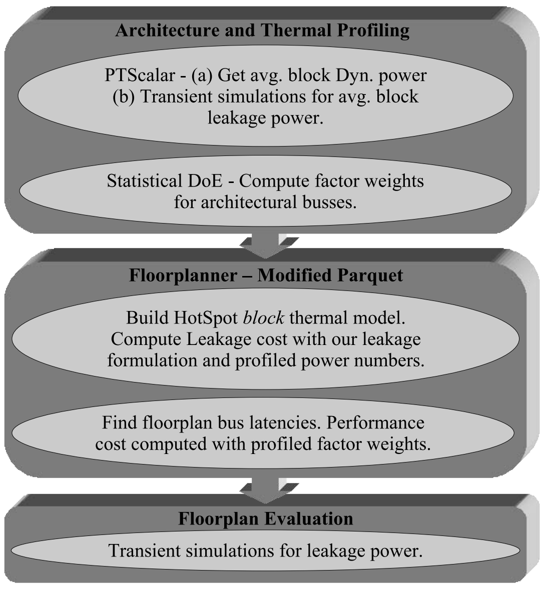
The rest of this paper is organized as follows. In Section 2 we explain the overall flow of our leakage reduction methodology briefly describing the tools used for thermal simulation and give the transistor leakage model used for accurate computation of leakage power. Section 3 details our leakage aware floorplanner deriving the transient formulation for leakage power, a simpler model dependent formulation and the whitespace modeling algorithm. Section 4 provides the experimentation results followed by the conclusion in Section 5.

2. The flow of our method

Our floorplanning framework revolves around the non-slicing floorplanner Parquet [1]. The overall flow of our methodology is shown in Figure 1 with a brief description given below.

Thermal profiling using detailed instruction level simulations to obtain average dynamic and leakage power numbers for the different microarchitecture blocks.

*This work was partly supported by a grant from DARPA under contract number N66001-04-8909



Architecture and Thermal Profiling
PTScalar - (a) Get avg. block Dyn. power (b) Transient simulations for avg. block leakage power.
Statistical DoE - Compute factor weights for architectural busses.
Floorplanner – Modified Parquet
Build HotSpot *block* thermal model. Compute Leakage cost with our leakage formulation and profiled power numbers.
Find floorplan bus latencies. Performance cost computed with profiled factor weights.
Floorplan Evaluation
Transient simulations for leakage power.

Figure 1. Overall flow of our leakage reduction and evaluation methodology.

Architectural profiling with statistical Design of Experiments (DoE) to reduce the number of simulations and gauge the impact of microarchitectural busses on performance [11]. We compute factor weights for each critical microarchitecture bus.

Floorplanning using Parquet to include leakage and performance costs. The input to our floorplanner consists of the profiled average powers and factor weights computed above. Temperature is computed using the *block* model in HotSpot. The leakage is computed according to two formulations

- Leakage power model independent **transient** formulation: This captures the interdependence of leakage on temperature without performing expensive transient simulations during the floorplanning phase.
- Leakage power model dependent **simple** formulation: This is derived from the transient formulation and depends on the leakage power model used.

Evaluation of the optimized floorplan using detailed instruction level transient simulations and the HotSpot *grid* model to compute the leakage power.

The following sections elaborate on each of these steps in the overall flow.

2.1. Architecture profiling and thermal modeling

We run our leakage reduction formulation on a microarchitecture based on the Alpha 21264 design scaled down to $130nm$.The configuration details for each block of the microarchitecture are shown in Table 1. Only the core blocks are floorplanned and the L2 cache is wrapped around the core so as to obtain a square die of side $15.9mm$.

Prior to floorplanning, we compute the average dynamic power consumption of each block using the PTScalar framework described in [10]. The idea is similar to WATTCH [3] wherein during each simulation cycle the number of accesses to every block in the microarchitecture is monitored to compute its activity, which is then used to calculate its dynamic power.

Block Name	Description
Register Update Unit(RUU) size	16 Instructions
Load Store Queue(LSQ) size	8 Instructions
Fetch queue size	4 Instructions
Issue width	4 Instructions / cycle
Decode width	4 Instructions / cycle
Commit width	4 Instructions / cycle
Integer ALU (IALU)	3 adder and 1 multiplier unit
Floating point ALU (FPALU)	1 adder and 1 multiplier unit
Branch predictor	Combined, Bimodal with 2K entries and 2 level with 1K entries and 8-bit history
Instruction TLB (ITLB)	64 entry fully associative with 30 cycle miss penalty
Data TLB (DTB)	128 entry fully associative with 30 cycle miss penalty
Branch Target Buffer	512 entries and 4-way associative
L1 Data cache	16KB, 4-way set-associative, with 32B blocks and 1 cycle latency
L1 Instruction cache	16KB, direct mapped, with 32B blocks and 1 cycle latency
L2 cache	256KB, 4-way set-associative, with 64B blocks and 6 cycle latency

Table 1. Microarchitecture configuration

We conduct profiling runs for a set of 10 Spec2000 [14] benchmarks consisting of 5 integer (bzip2, gcc, gzip, crafty, twolf) and 5 floating point benchmarks (mesa, art, mgrid, swim, equake). The benchmarks are run for 500 million instructions after an initial architectural warm up period of a 100 million instructions in order to avoid any cold start effects. We also allow an additional thermal warmup of 200 million instructions before monitoring any temperature dependent statistics. The main purpose of these simulations is to arrive at floorplan (i.e. temperature) independent average powers dissipated by each of the blocks assuming the instruction mix covers a wide range of applications.

We use HotSpot [9] for thermal modeling in our overall flow. HotSpot uses the well known duality between thermal power dissipation and current flow in an electrical network by constructing an RC model of resistors and capacitors. The equation governing heat dissipation in a thermal system is given by

$$C\,\frac{dT}{dt} + G\,T \quad = \quad P \tag{1}$$

where G and C are the thermal conductance and capacitance matrices respectively, T is the temperature vector and P is the power vector giving the temperature and power at each node in the thermal RC network respectively. Transient simulations for leakage power computation are performed by integrating Equation (1) over fixed size intervals using HotSpot. The leakage numbers are used as weighting constants when computing the leakage penalty using the **simple** model because different types of blocks (for instance logic vs memory) having the same area and temperature consume different amounts of leakage power.

During floorplanning each block is represented by a node in the RC network. This is referred to as the *block* model in [12] and is used for its computatational efficiency inside the floorplanning optimization loop.

Once we obtain an optimized floorplan the evaluation phase consists of running detailed simulations on all 10 benchmarks for 500 million instructions after an architectural warmup of 100 million instructions and a thermal warmup of 200 million instructions. Similar to the profiling phase, we use HotSpot's *grid* model [12] for better accuracy at the cost of increased runtime.

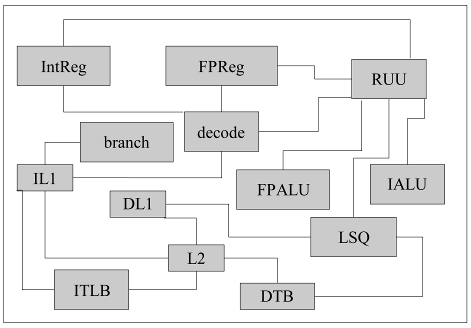


Figure 2. Busses modeled for performance evaluation. The fetch unit is considered part of IL1.

2.2. Performance modeling

One measure of performance during floorplanning is the widely used half perimeter wirelength metric (HPWL). However, as pointed out in [12] and [11], since this metric lacks the notion of criticality of a wire, floorplans with smaller total wirelength may have a worse performance than those with larger HWPL.

As leakage reduction tends to create whitespace between blocks in a floorplan, which in turn hurts performance, we wire-pipeline long busses to avoid large performance penalties in the microarchitecture. Towards this goal the authors in [6] consider both the architectural and physical design spaces in their floorplanning tool to devise an optimal floorplan meeting the input constraints. The work in [11] uses a statistical Design of Experiments (DOE) approach so as to avoid running a prohibitively large number of experiments to gauge the impact of each microarchitecture bus on performance. We use the weights in [11] (also called factors) obtained using the DOE approach for critical busses in the floorplan to evaluate their impact on performance. Briefly, the authors use a 2 level, resolution III fractional factorial design wherein busses are assigned latencies from 0 to their maximum value. The SimpleScalar simulator [4] is modified to incorporate the additional latencies on the wires and a set of experiments[1] performed to determine the weights of the busses. For our purposes we consider the 17 critical busses as shown in Figure 2. Multiple factor interactions are ignored in our work since they are negligible compared to single factor interactions. The main assumption made here is that these factors affect the system response monotonically. The wirelength term during floorplanning is computed as the sum of weighted latencies, i.e. $HPWL_W = \sum_{i=1}^{n} w_i \cdot l_i$, where w_i is the factor weight and l_i is the latency of bus i. The latency of a particular wire is determined based on the microarchitecture frequency of operation fed to the floorplanner. Although the main thrust of this work is not in modeling performance we intend to show savings in leakage through floorplanning with a tolerable loss in performance.

3. Floorplanning details

Floorplanning is performed using Parquet v4.0 [1], a simulated annealing based floorplanner, in fixed-outline mode with a given bound on the total whitespace. The individual blocks can have aspect ratios between 1 and 3 whereas the overall microarchitecture core aspect ratio is provided as an input to the floorplanner. We have added two temperature-specific moves to the annealing move-set, presented in Section 3.2. The following annealing cost function is used

$$\begin{aligned} C &= \alpha \cdot A + \beta \cdot HPWL_W \\ &= +\gamma \cdot L_P + (1-\alpha-\beta-\gamma) \cdot AR \end{aligned} \quad (2)$$

where A is the summation of the excess width and height of the floorplan, AR is the aspect ratio, $HPWL_W$ is the weighted sum of bus latencies as given in Section 2.2 and L_P is a term to capture the leakage power dependence on temperature. Section 3.1 shows how L_P is calculated. The coefficients α, β and γ are used to weight each of the cost terms.

3.1. Transient formulation independent of leakage power model

As mentioned previously, each block in the floorplan is modeled as a node in the thermal network given by Equation (1). Ideally, the power dissipated at any node in the network is a constant. However, considering leakage power, the total power dissipated at any node is now a function of the temperature T. One method of solving this system is to assume a constant power P_t at time t for a given time step Δt, integrate Equation (1) over that interval and compute the temperature $T + \Delta T$ at the new time $t + \Delta t$. This can then be used to update the power at time step $P_{t+\Delta t}$ and we repeat the process till convergence is achieved. Performing such a transient analysis is highly time consuming at each step of floorplanning and is only done during thermal profiling and leakage evaluation as described in Section 2.1. We avoid such expensive iterative computations with an approximation to the leakage power.

Rewriting Equation (1) with the temperature dependence of power gives

$$\tfrac{dT}{dt} = C^{-1} P(T) - C^{-1} G\, T \quad (3)$$

In the above equation $P(T)$ is the power dissipated at temperature T. Writing the finite difference approximation for the above equation for time Δt we obtain

$$\begin{aligned} \frac{\Delta T}{\Delta t} &= C^{-1} P(T_1) - C^{-1} G\, T_1 \\ \Rightarrow T_2 &= T_1 + (C^{-1} P(T_1) - C^{-1} G\, T_1)\, \Delta t \end{aligned} \quad (4)$$

where T_1 and T_2 are the block temperatures at times t and $t + \Delta t$ respectively. Expanding $P(T)$ in terms of its first order Taylor series assuming a small increment in temperature ΔT we get

$$\begin{aligned} P(T + \Delta T) &= P(T) + \frac{dP}{dT} \cdot \Delta T \\ \Rightarrow P(T_2) &= P(T_1) + \left.\frac{dP}{dT}\right|_{T_1} (T_2 - T_1) \end{aligned} \quad (5)$$

Combining Equations (4) and (5) and writing them in matrix form we obtain

$$P(T_2) = P(T_1) + J\Big|_{T_1} C^{-1} \Big[P(T_1) - G\, T_1 \Big] \Delta t \quad (6)$$

where $J\Big|_{T_1}$ is the jacobian of power P wrt temperature T evaluated at T_1. Let T_1 be the temperature attained with blocks dissipating their average dynamic power P_D.

$$\begin{aligned} G\, T_1 &= P_D \\ i.e.\ T_1 &= R\, P_D \end{aligned} \quad (7)$$

[1]given N factors, the number of experiments run is the nearest highest power of 2, i.e. for 17 factors we run 32 experiments

where $R = G^{-1}$ is the transfer thermal resistance matrix [15]. Also let $P_L(T_1) = P(T_1) - P_D$ be the leakage power dissipated at T_1. Using this in Equation (6) we obtain

$$\frac{P_L(T_2) - P_L(T_1)}{\Delta t} = J\Big|_{T_1} C^{-1}\left[P(T_1) - G\,T_1\right]$$

$$\Rightarrow \frac{\Delta P_L}{\Delta t} = J\Big|_{T_1} C^{-1}\,P_L(T_1) \qquad (8)$$

Equation (8) gives us the "temperature dependent" increase in leakage power over time Δt given the average dynamic power dissipation P_D. We use this as the formulation for leakage power, L_P, in Equation (2). The formulation is unable to capture a highly non-linear rise in leakage power e.g., with temperatures high enough to facilitate thermal runaway. However, given the large time constants of thermal power dissipation compared to the frequency of operation of the chip, we can reasonably assume a linear increment in leakage power. Since we try to minimize Equation (8) during floorplanning, this also reduces the temperature of the blocks helping to avoid thermal runaway. Equation (8) gives us the increment in power which could be used in rejecting floorplans with thermal runaway. Performing transient simulations would require an additional runtime overhead of $O(n \cdot N^2)$ where n is the number of iterations to converge and N is the total number of blocks in the floorplan.

3.2. Heuristic moves

We perform two additional kinds of moves in addition to those provided by Parquet described below.

1. **Temperature Move:** We randomly choose among the top 5 hottest and coolest blocks in the floorplan and swap them.
2. **Power Move:** Consider the transfer thermal resistance matrix in Equation (7). Element R_{ij} corresponds to the increase in temperature of block i for a unit change in power of block j, i.e. $R_{ij} \cdot P_j$ gives us the influence of block j on block i, where P_j is the power dissipated by block j. We randomly choose a block i among the top 5 most leaky blocks and place it next to the block exerting the least influence on block i, i.e. the block with the minimum $R_{ij} \cdot P_j$ value.

3.3. Simple formulation dependent on leakage power model

The subthreshold leakage power model for a particular microarchitecture block is given below [10]

$$P_L(T) = A\,T^2\,e^{-\left(\frac{\alpha\,V_{dd}+\beta}{T}\right)} \qquad (9)$$

where $P_L(T)$ is the leakage power dissipated at temperature T, A is proportional to the area of the block, V_{dd} is the supply voltage magnitude and α and β are empirical constants whose values depend on whether the block is a functional unit such as IALU or a memory unit such as a cache. Using the transient formulation we recompute Equation (8) as

$$\frac{\Delta P_L}{\Delta t}\Bigg| = C^{-1}A^2T^2e^{-2\left(\frac{\alpha\,V_{dd}+\beta}{T}\right)}\left[2T + \alpha V_{dd} + \beta\right] \qquad (10)$$

The thermal capacitance of a block is directly proportional to its area. Expanding the exponential in its Taylor series and observing

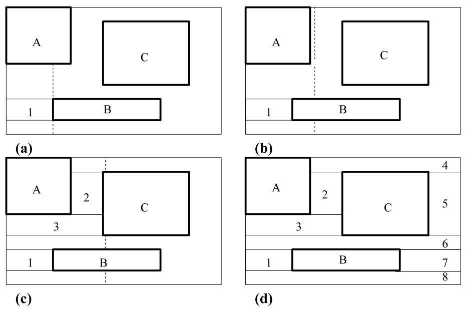


Figure 3. Execution of sweepline on a toy example. Disjoint dotted lines are whitespace intervals showing the advance of the sweep-line at event points. A, B, C are blocks and 1-8 are whitespaces filled in. Two event points between (c) and (d) are not shown.

the dominant term gives us a cubic polynomial in T. We therefore use the cubic formulation shown below

$$L_P = k\,A\,T^3 \qquad (11)$$

where k is a proportionality constant obtained using the average leakage power dissipation of the blocks during their profiling runs. Intuitively the model makes sense since larger blocks (i.e. more leaky transistors) with higher temperature dissipate more leakage power. The proportionality constant accounts for the fact that different types of blocks having the same area dissipate different amounts of leakage power at the same temperature. Equation (11) offers the advantage that it is simple to calculate compared to (8) which incurs an additional overhead in computing the leakage powers and the jacobian for each block at temperature T. Note that term T is calculated using HotSpot and does indeed consider the interaction between neighboring modules.

3.4. Whitespace modeling

Parquet uses a non-slicing sequence pair representation to perform the floorplanning. Since whitespaces influence the temperature distribution of the chip, they need to be accounted for during the floorplanning process. In [8] which also uses Parquet, the authors increase the area of certain blocks to fill the whitespace. It is not clear to us as to how this could be done in a non-slicing floorplan. Although our work does not directly exploit the available whitespace during fixed die floorplanning, we need to model whitespace for the annealer engine to evaluate the temperature distribution (leakage cost) of the floorplan accurately. For this purpose we model whitespaces as additional blocks in the floorplan with zero power dissipation. Our algorithm for computing whitespaces is based on the sweepline paradigm using a balanced interval tree [7] and takes as input the blocks in the floorplan, their total number and the width W and height H of the floorplan.

Figure 3 shows the sweepline execution on a small example. Event point 1 creates whitespace block 1 and two new whitespace intervals above and below block B(Figure $3(a)$). Event point 2 creates a new whitespace interval to the right of block A(Figure $3(b)$). At event point 3 we find overlaps creating whitespace blocks 2 and 3 and two new whitespace intervals, one above block C and the other between blocks B and C(Figure $3(c)$). The runtime complexity of our algorithm is $O(NlogN + K)$ where N

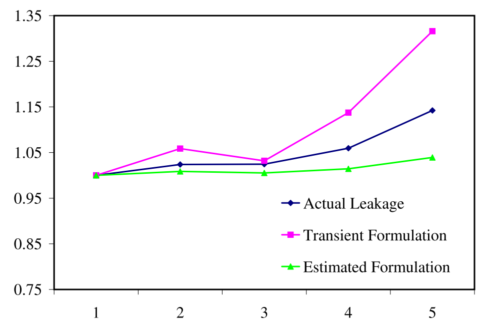


Figure 4. Normalized fidelity plots for eqn. (8) **'transient', eqn.** (11) **'estimated' and accurate transient simulations 'accurate'. Floorplans are numbered in increasing value of 'accurate' leakage for better visualization.**

is the number of blocks in the floorplan and K is the number of whitespace blocks (which is at most linear in N).

4. Results

We ran our experiments on an Intel$^{®}$ 3.2 GHz CPU with 2GB RAM. To evaluate the leakage power of each floorplan, detailed instruction level transient simulations were performed for all 10 benchmarks for 500 million instructions after architectural and thermal warmups of 100 million and 200 million instructions respectively. Each run of Parquet chooses the best among 10 generated floorplans.

4.1. Fidelity of our formulation

To compute the fidelity of our leakage cost function used by the floorplanner in evaluating different floorplans in terms of their overall leakage cost, we generate a set of 5 floorplans (optimized with different leakage weights) and compute their leakage according to Equations (8) and (11). We only chose to compare with accurate leakage power simulations of benchmark bzip2 due to its higher average chip temperature and leakage power compared to the other benchmarks. We call the leakage obtained through these formulations the transient, estimated and actual leakage respectively. Figure 4 shows normalized plots of the three formulations with respect to the leakage of floorplan 1 for each formulation. As can be seen, all three curves track each other well with the transient formulation having high sensitivity to floorplans with different leakage. Moreover, the plot also shows that we can use our simple estimation method with low computational complexity and yet achieve similar results compared to the transient formulation.

4.2. Leakage savings

Figure 5 compares the gains in leakage of two floorplans, one with (transient case) and the other without (base case) leakage optimization. We compare their leakage and performance (in terms of CPI) for each of the 10 benchmarks.

From the figure we observe that leakage optimization results in 15% savings on an average with a performance penalty of under 5%. Both floorplans (i.e. with and without leakage consideration) were constrained to have a maximum whitespace of 10%. We also observe from the figure that optimizing the floorplan with the simple formulation from Equation (11) (called the estimate case) results in similar leakage savings.

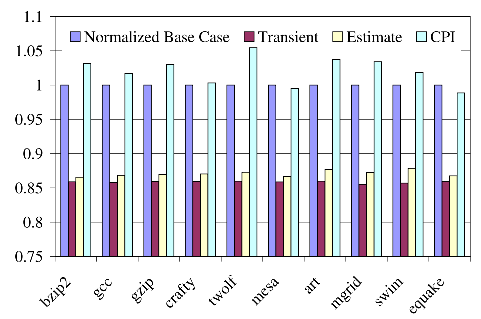


Figure 5. Normalized graphs of the leakage power and CPI in comparison with a floorplan optimized without leakage called Base Case. Y-axis begins at 0.75**.**

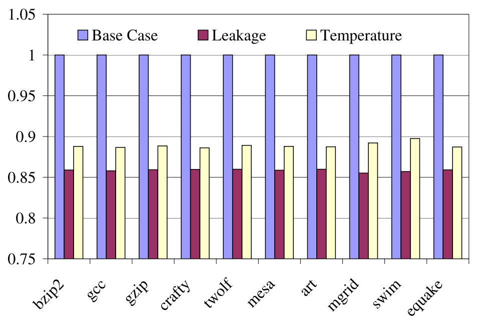


Figure 6. Normalized graphs of leakage power with maximum temperature and transient formulation as objective functions compared to a base floorplan without leakage optimization. Y-axis begins at 0.75**.**

4.3. Maximum temperature reduction

Similar to [12] and [8], most works on thermal-aware floorplanning aim at minimizing the maximum temperature of the floorplan. As mentioned previously, the same cannot be applied to the case of leakage minimization since large blocks like the data caches although have a lower temperature could have more leakage due to their larger area. This could be the result of, for instance, placing high temperature modules next to the large leaky modules. Figure 6 compares the leakage savings when the thermal objective function is to reduce the maximum chip temperature as opposed to our leakage objective in Equation (8). The figure shows that temperature reduction results in floorplans with an average leakage about 4% more than the case with leakage optimization. We believe this difference will increase for lower technology nodes, which we will investigate in our future work. Additionally, as discussed in Section 3.1, we have no way of detecting thermal runaway in the floorplan with such an objective function. It must be noted that we observed similar reductions in peak temperature of these two formulations compared to the floorplan optimized only for performance, indicating that our formulation in Equation (8) is effective in reducing peak chip temperature.

4.4. Whitespace considerations

We perform floorplanning with 10% and 15% whitespace(WS) and compute the leakage using transient simulations. Figure 7 shows the results. It also shows the impact of not modeling whitespace (No WS Modeling) during floorplanning. Negligible differences are observed in leakage savings of floorplans with 10% and

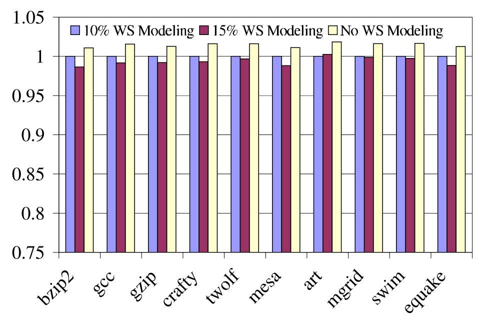


Figure 7. Normalized graphs showing the impact of WS on leakage reduction. Bar 1, 2 and 3 are leakge with 10%, 15% and no WS modeling during floorplan optimization. Y-axis begins at 0.75.

Floorplan scheme	Leakage Modeled	L2 cache partitioned	Whitespace modeled	Runtime ($secs$)
A	No	No	No	0.24
B	Yes	No	No	19
C	Yes	Yes	No	235
D	Yes	Yes	Yes	520

Table 2. Runtimes for different floorplanning schemes

15% whitespace. Moreover, modeling whitespace (as blocks with zero dynamic power dissipation) has little impact on the overall leakage savings. We believe the primary reason for such behavior is because the simulations were performed with an ambient temperature of $40°C$ which is an optimistic estimate of the temperature inside a CPU tower. Our simulations resulted in maximum block temperatures of about a $100°C$. Due to its exponential dependence, small differences in temperature at higher temperature ranges result in larger changes in leakage power than large differences in temperature at lower temperature ranges. This also points to the important fact that increased whitespace in the floorplan yields diminishing leakage power savings when the average chip temperature is not very high.

4.5. Runtimes

Table 2 lists the runtimes of various schemes employed in the floorplanning process. Scheme A only models performance. Scheme B models leakage of the various blocks in the floorplan using Equation (8) but no consideration is given to whitespaces. Schemes C and D also model leakage but unlike B they partition the surrounding L2 cache into smaller blocks for better thermal modeling accuracy. Scheme D additionally includes whitespace modeling. As observed in Section 4.4 the difference between C and D in terms of leakage is negligible. Comparing B and C we found that partitioning the L2 cache into smaller blocks does not impact the overall results but runtime increases considerably due to the increased overhead of thermal modeling and leakage computations. However, considering the fact that L2 leakage increases with future technology nodes, schemes that improve its thermal modeling will be essential to accurate leakage cost computation.

5. Conclusions and future work

This paper presents new formulations for subthreshold leakage power reduction enabled via floorplanning. Our results show an improvement on average of 15% compared to the case when no leakage power optimization is performed, with a tolerable loss in performance. We show that our leakage formulation has good fidelity with transient simulations justifying its use during the floorplanning phase. We also show the effect of increasing whitespace in the floorplan. As future work, we would like to explore floorplanning with different microarchitectures and smaller technology nodes where leakage power dissipated is a larger fraction of the system dynamic power.

6. Acknowledgements

We would like to acknowledge Vidyasagar Nookala [11] in providing us with factor weights used in performance estimation.

References

[1] S. Adya and I. Markov. Fixed-outline floorplanning: enabling hierarchical design. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 11(6):1120–1135, Dec 2003.

[2] S. Borkar. Design challenges of technology scaling. *Micro, IEEE*, 19(4):23–29, Jul-Aug 1999.

[3] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: a framework for architectural-level power analysis and optimizations. In *ISCA '00: Proceedings of the 27th annual international symposium on Computer architecture*, pages 83–94, New York, NY, USA, 2000. ACM Press.

[4] D. Burger and T. M. Austin. The simplescalar tool set, version 2.0. *SIGARCH Comput. Archit. News*, 25(3):13–25, 1997.

[5] P. Chaparro, J. Gonzalez, and A. Gonzalez. Thermal-aware clustered microarchitectures. In *ICCD '04: Proceedings of the IEEE International Conference on Computer Design (ICCD'04)*, pages 48–53, Washington, DC, USA, 2004. IEEE Computer Society.

[6] J. Cong, A. Jagannathan, G. Reinman, and M. Romesis. Microarchitecture evaluation with physical planning. In *DAC '03: Proceedings of the 40th conference on Design automation*, pages 32–35, New York, NY, USA, 2003. ACM Press.

[7] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, MA, 1989.

[8] Y. Han, I. Koren, and C. A. Moritz. Temperature aware floorplanning. In *Second Workshop on Temperature-Aware Computer Systems(TACS-2)*, 2005.

[9] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. Stan. Hotspot: a compact thermal modeling methodology for early-stage vlsi design. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 14(5):501–513, May 2006.

[10] W. Liao, L. He, Lepak, and K.M. Temperature and supply voltage aware performance and power modeling at microarchitecture level. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 24(7):1042–1053, July 2005.

[11] V. Nookala, Y. Chen, D. J. Lilja, and S. S. Sapatnekar. Microarchitecture-aware floorplanning using a statistical design of experiments approach. In *DAC '05: Proceedings of the 42nd annual conference on Design automation*, pages 579–584, New York, NY, USA, 2005. ACM Press.

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

[13] SIA. *International Technology Roadmap for Semiconductors*, 2001.

[14] SPEC. *Standard Performance Evaluation Corporation CPU2000 Benchmarks*.

[15] C.-H. Tsai and S.-M. S. Kang. Standard cell placement for even on-chip thermal distribution. In *ISPD '99: Proceedings of the 1999 international symposium on Physical design*, pages 179–184, New York, NY, USA, 1999. ACM Press.

[16] Y.-W. Wu, C.-L. Yang, P.-H. Yuh, and Y.-W. Chang. Joint exploration of architectural and physical design spaces with thermal consideration. In *ISLPED '05: Proceedings of the 2005 international symposium on Low power electronics and design*, pages 123–126, New York, NY, USA, 2005. ACM Press.