LEAF: A System Level Leakage-Aware Floorplanner for SoCs

Aseem Gupta†, Nikil D. Dutt†, Fadi J. Kurdahi†

†Center for Embedded Computer Systems
University of California, Irvine
Irvine, CA 92697 USA
{aseemg,dutt,kurdahi}@uci.edu

Kamal S. Khouri‡, Magdy S. Abadir‡

‡Design Technology Organization
Freescale Semiconductor Inc.
Austin, TX 78729 USA
{kamal.khouri,m.abadir}@freescale.com

Abstract— Process scaling and higher leakage power have resulted in increased power densities and elevated die temperatures. Due to the interdependence of temperature and leakage power, we observe that the floorplan has an impact on both the temperatures and the leakage of the IP-blocks in a system on chip (SoC). Hence, in this paper we propose a novel system level Leakage Aware Floorplanner (LEAF) which optimizes floorplans for temperature-aware leakage power along with the traditional metrics of area and wire length. Our floorplanner takes a SoC netlist and the dynamic power profile of functional blocks to determine a placement while optimizing for temperature dependent leakage power, area, and wire length. To demonstrate the effectiveness of LEAF, we implemented our methodology on ten industrial SoC designs from Freescale Semiconductor Inc. and evaluated the trade-off between leakage power and area. We observed up to 190% difference in the leakage power between leakage-unaware and leakage aware floorplanning.

I. Introduction

Process scaling has enabled electronic devices to offer much higher computational power and performance at the expense of increasing power densities. Leading semiconductor chip makers have already announced a discontinuation of increase in clock frequencies because of high operating temperatures resulting from the power consumption [1]. In order to reduce switching delays of transistors and to reduce dynamic power consumption, CMOS devices are scaled down along with the supply voltage (V_{DD}) and the transistor threshold voltage (V_{th}). This causes an increase of up to 5X in the leakage power dissipation per technology generation [2]. This has resulted in leakage power becoming a major part of the total power dissipation in a chip. In addition, the leakage current of a transistor increases with increasing die temperatures.

Floorplanning at the system level is the placement of functional IP-blocks with uncertain dimensions, but with fixed area. The objective of floorplanning is to determine a layout of blocks while optimizing the total area of the chip and the total wire length.

The following three observations motivate our research:

1. The subthreshold current (the main component of leakage current) of a transistor has been shown to have a super-linear dependency on temperature [3]. For newer technologies, the size of a transistor is even smaller and hence the sensitivity of leakage power due to temperature is even more pronounced.
2. Different functional blocks have different dynamic power dissipation profiles and hence produce varying local temperatures. The die temperature for a IP-block in a system-on-chip (SoC) is not confined to the block itself and effects the temperatures of all its neighboring blocks because of thermal diffusion. Thus the placement of blocks in the SoC determines the temperatures of the blocks. A functional block can have widely different temperatures for the different floorplans of the same SoC.
3. The floorplan has a direct effect on the leakage power of the SoC. Indeed, it has been shown that different floorplans have different leakage power [4].

The above observations, when considered together, lead us to believe that floorplanning should have an effect on the leakage power. This motivated us to investigate temperature dependent leakage power-aware floorplanning for SoCs at the system level. A leakage power-aware floorplanner considers the dynamic power profile of the blocks to calculate the block temperatures using which the leakage power of the SoC is estimated. The floorplan is then optimized for leakage power along with area and wire length.

The main contribution of our work is a system level Leakage Aware Floorplanner (LEAF) which optimizes floorplans for temperature-aware leakage power. Our floorplanner takes SoC netlist and dynamic power profile of the functional blocks to determine a placement while optimizing for temperature dependent leakage power, area, and wire length.

II. Related Work

The related work can be divided into three classifications: dependency of leakage power on temperature, classical floorplanning algorithms, and thermal-aware floorplanning.

In the area of temperature dependent leakage power S. Nassif et al. [5] propose a model estimating the leakage power for a chip while considering power supply and temperature variations. W. Liao et al. [6] also propose a temperature dependent leakage power model with an assumption that the complete chip is at one uniform temperature.

Floorplanning to reduce area and wire length has been a well studied area of research. There are slicing floorplanning algorithms [7] and non-slicing floorplanning algorithms [8]. The most widely used algorithm is the simulated annealing based slicing tree algorithm by Wong et al. [7]. Sarrafzadeh and Wong [9] present a comprehensive survey of floorplanning algorithms in their book.

1-4244-0630-7/07/$20.00 ©2007 IEEE.

Recent research on thermal-aware floorplanning has focused on managing peak temperatures at the micro-architectural level for concerns about reliability, design of cooling mechanisms, and Dynamic Thermal Management (DTM) schemes. The objective of these thermal-aware floorplanners is to reduce the peak temperature reached by any of the blocks during operation, using different layouts. Sankaranarayanan et al. have proposed thermal-aware floorplanning at the micro-architectural level for processors [10], [11]. Their objective also is to reduce the peak temperature. A study on the floorplanner's effectiveness in lowering the maximum processor temperatures was done by Han et al. [12]. Ekpanyapong et al. [13], Healy et al. [14], and Cong et al. [15] have proposed thermal-driven and communication profile-driven floorplanning with the objective to reduce peak temperatures. Hung, Vijaykrishnan et al. [16] have used floorplanning with genetic algorithms to reduce the peak temperatures and optimize area. Reduction of clock tree power using activity based register clustering and thermal aware placement has been proposed by Cheon et al. [17] and Obermeier et al. [18] respectively.

All of the above described works aim at peak temperature reduction and not at leakage power reduction. Moreover their work focuses on micro-architectural level where there is a lot of homogeneity in the activities of different blocks of the pipeline. As we see in the next section, floorplans with lower peak temperatures do not necessarily have a lower leakage power. Our work focuses on leakage power management using system level floorplanning for SoCs where the block activities are much more non-uniform. Thus for SoCs there is a lot more potential for leakage aware floorplanning.

III. Leakage-Aware vs. Thermal-Aware Floorplanning

Our experiments on an industrial SoC design have shown the disconnect between peak temperature and leakage power for different floorplans of a SoC. A floorplan when compared to another can exhibit a higher leakage power for a lower peak temperature. Fig. 1 shows the distribution of peak temperature and leakage power for various floorplans of the same SoC from Freescale Semiconductor Inc. We have used a system level temperature and floorplan-aware leakage power estimation tool $STEFAL$, which is described in detail in [4] and is summarized briefly in Section V. $STEFAL$ estimates the temperature dependent leakage power of a SoC while considering the floorplan and the dynamic power profiles of the blocks. Floorplans A through D are marked in increasing order of their peak temperatures. Floorplan A has the minimum peak temperature; however, its leakage power is not the least. The lowest leakage power was observed for Floorplan B, but its peak temperature is higher by about 23^oC than that of Floorplan A. Neither Floorplan C, with maximum leakage power, is the floorplan with the highest peak temperature, nor does Floorplan D, with maximum peak temperature, have the highest leakage power. This example experimentally establishes the following:

(i) Different floorplans of a SoC have different leakage power dissipation, which is the motivation for LEAF.

(ii) There exists no definite correlation between the leakage power and the peak temperature of a floorplan.

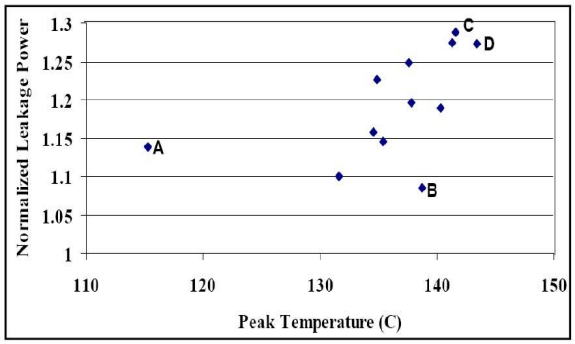

Fig. 1. Peak Temperature and Leakage Power for Various Floorplans of a SoC

(iii) Leakage power-aware floorplanning is distinct from thermal-aware floorplanning.

IV. Background

This section covers some of the fundamentals of effect of thermal diffusion and simulated annealing floorplanning.

A. Thermal Diffusion Among Blocks

The power consumed during transistor switching and during standby, due to leakage, is dissipated as heat. The principal job of the package and cooling mechanisms is to facilitate the transfer of this heat from the die to the outside environment. Regions of the chip with high power densities usually have higher temperatures as well. But this may not always be true because the heat of a block is not confined to itself and tends to move from a high temperature region to a low temperature region, primarily by the mechanism of conduction. The temperature of a particular block in the chip depends on the power densities of the adjacent blocks as well. Han et al. [12] have shown that in a core, a block whose power density is 5X the power density of another block, has its temperature lower than that of the other block by 12^oC. They also show that the temperatures of two blocks can differ by as much as 55^oC. Hence the effects of diffusion cannot be ignored and the floorplan becomes a key component while computing block temperatures from their known power densities.

B. Slicing Tree Floorplanning with Simulated Annealing

A slicing floorplan is a rectangular area that is sliced recursively by a horizontal or a vertical cut into a set of rectangular rooms to accommodate the blocks. Simulated annealing is a generic probabilistic meta-algorithm for the global optimization problem for a large search space [19]. Each step of the algorithm replaces the current solution by a random nearby solution, chosen with a probability that depends on the difference between the corresponding cost function values and on a global parameter Simulated Annealing Temperature (SAT), which is gradually decreased during the process. The dependency is such that the current solution changes almost randomly when SAT is large, but smaller cost function solutions are increasingly chosen as the SAT goes to zero. The algorithm employs a random search which not only accepts changes that decrease the cost function, but also some changes that increase it, which saves the method from becoming stuck at a local minima.

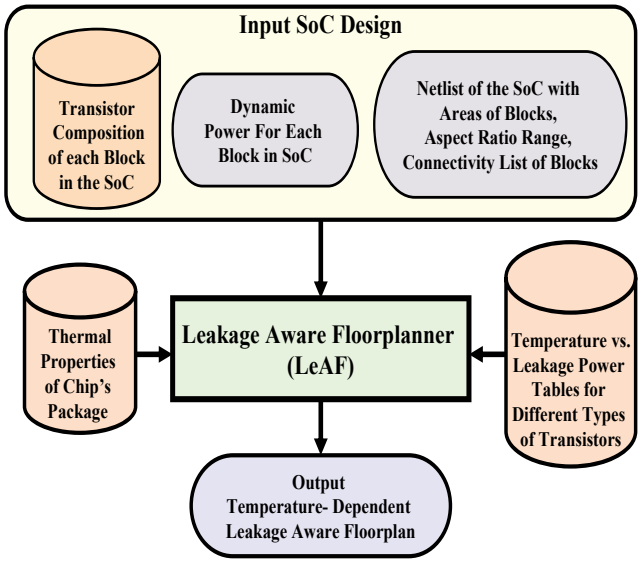

Fig. 2. Overview of LEAF

We have used slicing tree based simulated annealing floorplanning in our work. We briefly introduce some of the other terminology. Total area of the chip is the sum of ***active area*** (sum of the areas of the transistors of all the blocks) and ***inactive area*** (sum of the areas of dead space, interconnects routed between the blocks, and connecting ports of the blocks).

In slicing tree floorplanning based on simulated annealing algorithm, a new candidate floorplan solution is generated by making a move randomly on the current floorplan solution. Three types of moves are defined, which are applied on the current solution to generate more candidate solutions. They are: swap two adjacent blocks, complement a chain of non-zero length, and swap an adjacent block and a cut. After making the moves, the cost function is calculated for the candidate solutions and the simulated annealing algorithm decides which moves will be accepted.

V. Leakage-Aware Floorplanner (LEAF)

Fig. 2 shows an overview of LEAF. LEAF is motivated by our observations in Section III which show that there is a significant difference in leakage power among floorplans of the same SoC. Similar to a traditional leakage-**un**aware floorplanner LEAF also takes the netlist of the SoC which has the areas of the functional blocks, the range of aspect ratio for each block, and the connectivity list between the blocks. Unlike traditional floorplanners, LEAF also takes as input the dynamic power profile of the functional blocks. This dynamic power profile of the SoC can be obtained by profiling the application on system- or RT- level simulators. Additionally, LEAF uses the transistor composition of the functional blocks. Each IP-block is synthesized into a combination of many different types of transistors. The transistor composition for the blocks contains details about the different types of transistors used within the block and the number of each type of transistors. Different types of transistors have different leakage power and exhibit different temperature sensitivity. Using a library of temperature vs. leakage power tables for the different types of transistors and the thermal properties of chip's package, LEAF outputs a system level placement while optimizing the temperature dependent leakage power, area, and wire length.

Fig. 3 presents the details of LEAF and the pseudocode is shown in Fig 4. The left side of Fig. 3 shows the schematic diagram of LEAF. The cost function of a traditional simulated annealing based slicing tree floorplanner is supplemented with leakage power. This leakage power is both temperature-aware and floorplan-aware. Hence, in Fig. 3, we have added a box to calculate temperature- and floorplan-aware leakage power, labeled Ⓐ. Thus the cost function of LEAF is now a function of total area, wire length, and leakage power:

$$\begin{aligned} Cost\ Function = {} & W_{Area} * S_{Area} * Total\ Area \\ & + W_{Length} * S_{Length} * Wire\ Length \\ & + W_{Leakage} * S_{Leakage} * Leakage\ Power \end{aligned} \quad (1)$$

where $W_{Area}, W_{Length}, W_{Leakage}$ are the respective weights and $S_{Area}, S_{Length}, S_{Leakage}$ are the respective scaling factors. The weights are relative such that:

$$W_{Area} + W_{Length} + W_{Leakage} = 1 \quad (2)$$

The total area is the sum of active area and inactive area. Active area is the sum of the areas of the transistors of all the blocks and inactive area is the sum of the areas of dead space, interconnects routed between the blocks, and connecting ports of the blocks.

$$Total\ Area = Active\ Area + Inactive\ Area \quad (3)$$

The wire length is the sum of the lengths of all the connecting wires between the blocks in the connectivity list. The floorplanner estimates the wire length without doing actual routing by using the Manhattan distance between the blocks. Manhattan distances are calculated to get the length of wire between two connected blocks. Let (x_i, y_i) be the location of block i:

$$WireLength = \sum_{(i,j) \in\ connectivity\ list} |x_i - x_j| + |y_i - y_j| \quad (4)$$

The leakage power of the floorplan for which the cost function is to be calculated is floorplan and temperature dependent. The leakage power is estimated using a tool called $STEFAL$ [4] which is described only briefly here.

LEAF uses the leakage power to compute the leakage-aware cost function which in turn drives the floorplanner. The output floorplans are optimized for leakage power, area and wire length, and their respective weights determine the degree of influence. As we will see in Section VI the output floorplans do have a reduced leakage power. Since the leakage power, area, and wire length are weighted in the cost function from Eqn. (2), there is a tradeoff between leakage power and other optimization parameters.

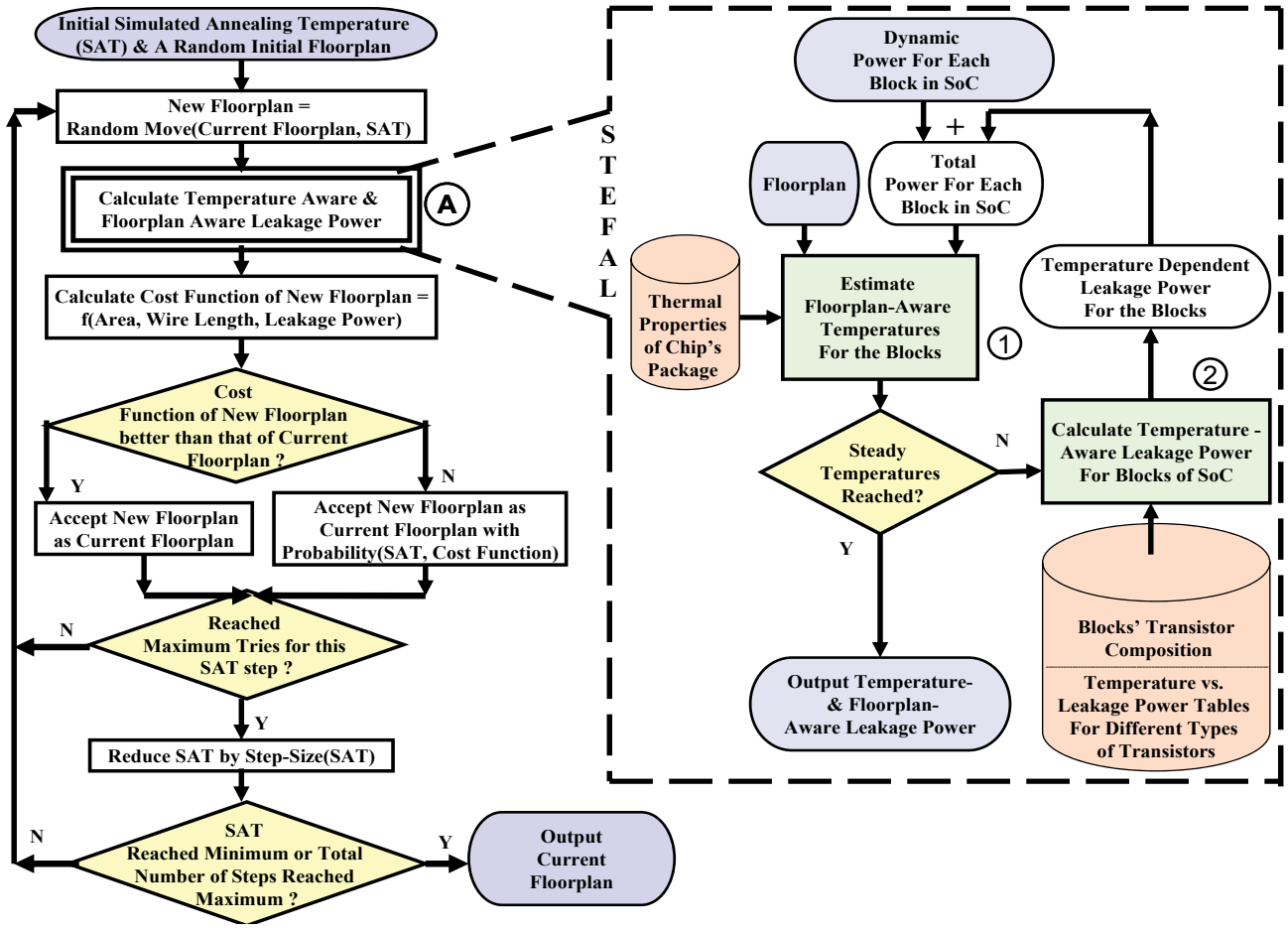

Fig. 3. Leakage Power Aware Floorplanner (LEAF) with Overview of STEFAL

LEAF()

Input SoC_{Desc} ▷ Input SoC Description with netlist
▷ Input dynamic power & transistor composition for all blocks,
▷ and package characteristics
Input DP_{SoC}, $Trns$, and Pkg
▷ Input temperature-leakage tables for all transistor types
Input LP_{Tbl}
Initialize SAT
$Flp \leftarrow$ RANDOM(SoC_{Desc})
while $(SAT > 0)$
  $N\_Steps \leftarrow 0$
  while $(N\_Steps < MAX\_STEPS)$
    $New_{Flp} \leftarrow$ RANDOMMOVE(Flp)
    $LP_{Flp} \leftarrow$STEFAL$(Flp, DP_{SoC}, Pkg, Trns, LP_{Tbl})$
    $Cost_{Flp} \leftarrow$COST$(LP_{Flp}, Area_{Flp}, WireLen_{Flp})$
    $LP_{New} \leftarrow$STEFAL$(New_{Flp}, DP_{SoC}, Pkg, Trns, LP_{Tbl})$
    $Cost_{New} \leftarrow$COST$(LP_{New}, Area_{New}, WireLen_{New})$
    if $(Cost_{New} < Cost_{Flp})$ $Flp \leftarrow New_{Flp}$
    if $Cost_{New} \geq Cost_{Flp}$
      $Prob \leftarrow$ PROBABILITY$(SAT, Cost_{Flp})$
      $Flp \leftarrow New_{Flp}(Prob)$
    $N\_Steps \leftarrow N\_Steps + 1$
  endwhile
  $step\_size \leftarrow$ FUNCTION(SAT)
  $SAT \leftarrow SAT - step\_size$
endwhile
Output Flp

Fig. 4. Pseudocode for LEAF

A. STEFAL

The right side of Fig. 3 expands the box Ⓐ for calculating temperature- and floorplan-aware leakage power into the $STEFAL$ estimator tool used. The system level temperature and floorplan aware leakage ($STEFAL$) estimator tool is presented in [4] and is briefly described here. This leakage power estimation tool takes the inputs of the dynamic power profiles of all the blocks in the SoC and the floorplan for which the cost function has to be calculated. The procedure *Estimate Floorplan-Aware Temperatures for the Blocks*, labeled ①, estimates the temperatures of the blocks using the input floorplan and total power (Dynamic + Leakage) for the blocks, along with the library of thermal properties of the chip's package. Procedure ① uses a freely available tool HotSpot [20] to do the temperature estimation. The procedure *Calculate Temperature Aware Leakage Power for Blocks of the SoC*, labeled ②, calculates temperature-aware leakage power using the block temperatures estimated by ①, library of blocks' transistor composition, and the library of temperature vs. leakage power tables for different types of transistors. The temperature vs. leakage power tables are pre-generated library tables at 65nm technology and have the leakage power values for each type of transistor from 27^oC to 150^oC. For each block, Procedure ② calculates the leakage power by multiplying the number of transistors of each type in its composition by the corresponding leakage power at the block's temperature (from the temperature vs. leakage power tables).

For each block, the leakage power calculated by Procedure

② is added to its dynamic power to get its total power which is then used to estimate the block temperatures using Procedure ①. This is done iteratively to find the stable operating temperatures of the IP-blocks and the leakage power at stable temperatures. At stable temperatures all the dynamic and the leakage power is dissipated to the environment by the package. If the maximum difference in the block temperatures of two consecutive iterations is less than a small value, ϵ, the SoC has reached steady state temperature and the iteration is terminated. The estimation tool then outputs the total leakage power of the SoC.

B. Assumptions

LEAF is based on several assumptions as detailed below:

- We used industrial SPICE-like simulation models to generate the temperature vs. leakage power tables for different types of transistors that were used in our industrial designs.
- Currently we do not consider the interconnect power and assume that the dynamic power remains constant with the floorplan. This is a part of our future work.
- We assume that the application's profile is available, which can be obtained by system- or RTL-level simulation of the application. In some cases different benchmarks may have to be averaged to get an average dynamic power profile.

VI. Results

We applied LEAF on ten industrial SoC designs from Freescale Semiconductor's PowerQUICC family of SoCs. Because of the proprietary nature of the data we provide only the normalized values in results. The influence of leakage power on the floorplanning can be adjusted using the weight of leakage power ($W_{Leakage}$) from Eqn. (1). For a higher value of $W_{Leakage}$, the output floorplans are expected to have less leakage. However, for a higher value of $W_{Leakage}$, the area and the wire length of the output floorplans increase because their degree of influence on the cost function is decreased. For all the results that we present in this paper the wire length of the output floorplans meet the wire length constraints of the SoC designs provided by the designer. There is a trade-off between area and leakage power because of the relative weights in Eqn. (2). However, we do not claim that the leakage power of a SoC can be reduced by increasing its area alone. We will point out some instances where a floorplan had a lower leakage power as well as a lower area. To compare leakage-**un**aware floorplanning with LEAF, we vary the degree of leakage awareness by varying $W_{Leakage}$. We executed LEAF for different values of $W_{Leakage}$ by varying $W_{Leakage}$ from 0 (LEAF becomes leakage-**un**aware) to 1 (LEAF optimizes **only** for leakage power). We collected data on the leakage power and the inactive area (as a % of the active area) of the resultant floorplans. Though different designs have different area constraints, floorplans with less than 30% inactive area are generally acceptable [21]. Conservatively we assume that the floorplans with less than 20% inactive area will be feasible and give supplementary comments about them. Many floorplans have very high inactive area and these supplementary comments will help identify the feasible floorplans.

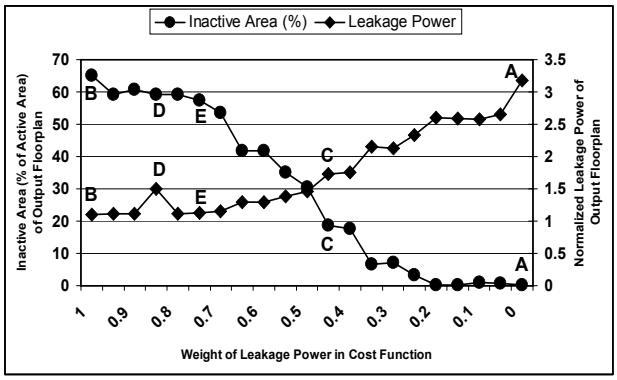

Fig. 5. Results for Design-I

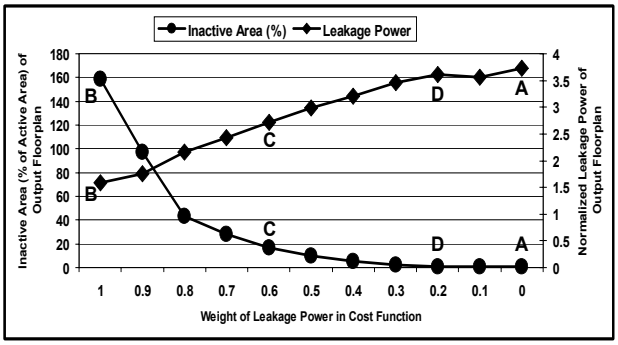

Fig. 6. Results for Design-II

A. Design-I

This design is for a relatively small SoC with eight blocks. We executed LEAF for a set of different values of $W_{Leakage}$ by incrementing $W_{Leakage}$ in steps of 0.05 thus outputting 21 floorplans. The results are shown in Fig. 5. The difference in the leakage power for leakage-unaware ($W_{Leakage} = 0$, Floorplan A) and leakage-only aware ($W_{Leakage} = 1$, Floorplan B) floorplans is 190%. The difference in the leakage power between Floorplan C (the floorplan with $\approx 20\%$ inactive area) and Floorplan A is 83.6%. Thus leakage aware floorplanning can significantly reduce leakage power of SoCs. We also observe that the for Floorplans D and E, there was a decrement in the area along with a decrement in the leakage power as well. Thus there is no definite indication that the leakage power of a floorplan can be reduced by increasing its area alone and designers can use LEAF for leakage power optimization at the system level.

B. Design-II

This design is a medium sized industrial SoC with twenty-three blocks and the resultant floorplans from 11 executions of LEAF by varying $W_{Leakage}$ in steps of 0.1 are in Fig. 6. The difference in the leakage power for leakage-unaware (Floorplan A) and leakage-only aware (Floorplan B) floorplanning is 134.8% and the difference in the leakage power between Floorplan C (the floorplan with $\approx 20\%$ inactive area) and Floorplan A is 37.3%. We also observe that Floorplans D through A are compact and have very small inactive area, but their leakage power differs by 5.8%.

TABLE I
SUMMARY OF DESIGNS-IV THROUGH X

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
	$W_{Leakage} = 0$		$W_{Leakage} = 1$		Difference	Normalized Leakage of	Difference
Design	Inactive	Normalized	Inactive	Normalized	in Leakage	the Floorplan with	in Leakage
(Number of blocks)	Area	Leakage	Area	Leakage	(3)-(5)	$\approx$ 20% Inactive Area	(3)-(7)
Design IV(10)	2.23%	4.1	68.94%	2.61	56.81%	3.47	18.06%
Design V(6)	0.98%	3.07	34.84%	2.24	36.85%	2.48	23.74%
Design VI(12)	0.30%	4.74	78.53%	3.01	57.43%	4.62	2.68%
Design VII(17)	0.22%	10.76	93.98%	5.34	101.54%	8.78	22.63%
Design VIII(31)	1.08%	11.19	120.0%	5.42	106.14%	8.95	25.01%
Design IX(9)	0%	1.51	46.05%	1.0	51.12%	1.32	14.8%
Design X(18)	0.31%	3.69	67.38%	2.27	62.82%	3.33	11.0%

C. Design-III

This design is the largest industrial SoC made available to us with forty-nine blocks. For this SoC the five outputs from $W_{Leakage} = 0$ to $W_{Leakage} = 0.4$ in steps of 0.1 had an inactive area of less than 6%. However, the difference in the leakage power of the floorplans is 29.2%. Thus even for small inactive area, LEAF optimized floorplans for significant leakage power savings. This design offered an interesting design space because of its size and nature. Out of the forty-nine IP-blocks in the SoC, many blocks were of small area. Because of this the floorplans were compact and had a small inactive space area of less than 6%. The floorplan for $W_{Leakage} = 1$ had an inactive area of 103.53% and its leakage power was 67.4% less than the floorplan for $W_{Leakage} = 0$.

D. Results on Designs-IV to X

The results for industrial Designs-IV through X are summarized in Table I. For each of these designs LEAF was executed 11 times for a range of values of $W_{Leakage}$ from 0 to 1 in steps of 0.1. Columns (2) and (3) have the inactive area and the normalized leakage power at $W_{Leakage} = 0$ respectively while for $W_{Leakage} = 1$, they are shown in Columns (4) and (5) respectively. The difference between the leakage of the leakage-unaware and the leakage-only aware floorplans (Columns (3) and (5)) is in Column (6). We observe that among these designs, Design-VIII had the largest difference of 106.14%. For most of the designs, 9 out of the 11 floorplans had less than 20% inactive area and Design-V had 10 such floorplans. This indicates that a majority of points in the design space have low inactive area while being leakage aware. The normalized leakage power for the floorplan with $\approx$ 20% inactive area is shown in Column (7). Column (8) shows the difference in the leakage power between leakage-unaware floorplan (Column 3) and the floorplan with $\approx$ 20% inactive area (Column (7)). We observe a maximum difference of 25.01% in the leakage power for Design-VIII. These results on industrial designs further support the rationale behind leakage power aware floorplanning at the system level.

E. Summary of Results

From the ten industrial designs, we observed that: there is a significant variation in the leakage power for various floorplans of the same SoC and indeed, leakage aware floorplanning can output floorplans with lower leakage power.

We executed LEAF using cygwin, which provides a linux-like environment for windows operating system, on a Intel Centrino 1.8 GHz processor with 512 MB RAM. The runtime overhead for LEAF compared to leakage-**un**aware floorplanning was only about **4%**, which we believe is reasonable.

VII. CONCLUSION

In this paper we demonstrated the impact of floorplanning on the leakage power of a SoC and we proposed a novel system level leakage power-aware floorplanner LEAF, which optimizes floorplans for leakage power. LEAF is generic and can be useful for a wide variety of SoCs. In our future work we plan to explore leakage aware floorplanners which can guide the floorplanning towards a floorplan with lower leakage.

Acknowledgment: This work was supported by Freescale Semiconductor Inc.

REFERENCES

[1] Stephan Ohr, "Efforts heat up to remove processor hot spots," *EE Times*, February 2005.
[2] V. De and S. Borkar., "Technology and design challenges for low power and high performance," *ISLPED*, 1999.
[3] MW. Liu et al., *BSIM3v3 MOSFET Model,* Silicon and Beyond-Advanced Device Models and Circuit Simulators, World Scientific Pub. Co., 2000.
[4] Aseem Gupta et al.,"STEFAL: A System Level Temperature- and Floorplan-Aware Leakage Power Estimator for SoCs,"*VLSI Design Conference*, 2007.
[5] H. Su, S. Nassif et al., "Full Chip Leakage Estimation Considering Power Supply and Temperature Variations," *ISLPED*, 2003.
[6] W. Liao et al., "Microarchitecture Level Power and Thermal Simulation Considering Temperature Dependent Leakage Model," *ISLPED*, 2003.
[7] D. F. Wong et al., "A new algorithm for floorplan design," *DAC*, 1986.
[8] P. Guo et al., "An O-tree Representation of Non-Slicing Floorplan and Its Application," *DAC*, 1999.
[9] M. Sarrafzadeh and C. K. Wong, *An Introduction to VLSI Physical Design,* McGraw-Hill Higher Education, 1996.
[10] K. Sankaranarayanan et al., "A Case for Thermal-Aware Floorplanning at the Microarchitectural Level," *Journal of Instruction Level Parallelism*, 2005.
[11] K. Sankaranarayanan et al., "Microarchitectural Floorplanning for Thermal Management: A Technical Report," *University of Virginia*, 2005.
[12] Y. Han et al.,"Temperature Aware Floorplanning," *Workshop on Temperature Aware Computer Systems*, June 2005.
[13] M. Ekpanyapong et al., "Thermal Aware 3-D Microarchitectural Floorplanning," *Technical Reports, Georgia Tech*, December 2004.
[14] Healy, Ekpanyapong et al., "Microarchitectural Floorplanning Under Performance and Thermal Tradeoff,"*DATE*,2006.
[15] J. Cong, J. Wei et al.,"A Thermal-Driven Floorplanning Algorithm for 3D ICs,"*ICCAD-2004*.
[16] W.L. Hung et al.,"Thermal-Aware Floorplanning Using Genetic Algorithms," *ISQED*, 2005.
[17] Y. Cheon et al., "Power-Aware Placement," *DAC*, 2005.
[18] B. Obermeier et al., "Temperature Aware Global Placement," *ASP-DAC*, 2004.
[19] S. Kirkpatrick et al., "Optimization by Simulated Annealing," *Science, Vol. 220*, 1983.
[20] K. Skadron et al., "Control-theoretic Techniques and Thermal-RC Modeling for Accurate and Localized Dynamic Thermal Management," *HPCA*, 2002.
[21] Personal communication with designers at Freescale Semiconductor Inc.