Power pre-characterized meshing algorithm for finite element thermal analysis of integrated circuits

Shohdy Abdelkader
Mentor Graphics Egypt

Alaa ELRouby
Faculty of Engineering and Natural Science,
Yildirim Beyazit University

Mohamed Dessouky
Mentor Graphics Egypt

***Abstract*—In this paper we present an adaptive meshing technique suitable for steady state finite element (FE) based thermal analysis of integrated circuits (ICs). The algorithm presented is a non iterative one where the technology used is first pre-characterized. The characterization results are then used for scanning the layout to detect high power regions then fine meshing them. Finally, the analysis is done only once. This makes it faster than conventional iterative adaptive meshing methods. The algorithm results showed comparable accuracy and better performance when compared to the flux based (iterative) and the power aware (non iterative) algorithms.**

I. Introduction

Meshing is a key factor in the FE process, and specially the thermal analysis one. It has a big impact on the analysis results. A tradeoff has always existed between accuracy and speed when meshing any design. High accuracy usually demands a fine mesh size. On the other hand, a finer mesh means a bigger system which also means more time to solve.

This paper is organized such that in section II, it describes the challenges facing adaptive meshing of ICs, along with common adaptive meshing techniques stated in literature. Followed by section III, where it describes in detail the proposed meshing algorithm. Finally, the testing results in section IV.

II. Adaptive meshing in thermal analysis of ICs

The mesh is designed to have each element with homogenous physical properties. For thermal analysis, the thermal conductivity should be the same across the element volume. Unfortunately nowadays layouts are full of wiring and devices specially those made for digital circuits. It is not practical to have a mesh with the constraint aforementioned. So for thermal analysis of ICs, the meshing engine produces elements containing different materials. Then it calculates an equivalent thermal conductivity in the element. Various ways exist in literature to calculate the equivalent values [1], [2], and [3]. One more challenge facing the adaptive meshing in ICs, is that mesh refinement involves more than adding extra nodes and elements. Since for each mesh configuration, a calculation has to be done for the densities of the materials inside each element so as to calculate its thermal conductivity. This process is usually called property extraction. So a refined mesh basically means a new set of thermal conductivity values and somehow a different system to solve.

Adaptive meshing can be divided into three main categories [4]. The p-refinement, r-refinement, and the h-refinement. In the p-refinement, the mesh is unchanged but the order of the polynomial FE basis functions is increased adaptively. While in the r-refinement, element type and number of mesh nodes are kept constant, but nodes positions are changed adaptively to focus on areas with sharp solution gradients. For h-refinement, the same type of finite element is used in all the mesh elements but the element size is changed. Of the three types, the h-refinement is the most popular due to its simplicity. Two main approaches for the h-refinement [5] exists in the literature, error based and flux based.

The error based approach starts by doing a FE analysis with a coarse grid, then uses the results to do an estimation of the error inside each element. Mesh is then refined in places where the error is above a certain threshold. The process keeps going on until the error goes below a threshold. A good example for this technique can be found in [6]. The main drawback of this technique is that the process of the error estimation itself is a complex process mathematically and computationally. Various techniques for calculating the error can be found in [7], and [8]. For the flux based approach, a FE analysis is also done with a coarse grid, where the results are used to calculate the gradient in each element. The mesh is then refined at places where the gradient is highest. This process is repeated iteratively until the difference in results between subsequent runs goes below a certain threshold. This makes it more popular over the error based as the gradient calculation is way simpler and faster than the error estimation. This approach was demonstrated in [9], [1], and [10]. The main disadvantage in both of the approaches described above is that they are both iterative. This requires doing multiple analysis runs, where each run consists of a meshing step, property extraction, and the system solution. The property extraction step involves a considerable amount of geometrical processing to calculate the densities of the materials inside each element and then the thermal conductivity. For the typical size of nowadays chips, this process may take more time than the analysis step.

A non iterative power aware meshing technique is found in [11]. The core idea of the algorithm is to scan the layout to detect the areas with high power, and then fine mesh these areas. This algorithm suffers from two main disadvantages. First, the method used for detecting high power areas is not accurate enough specially when operating at the boundaries of high power regions. Second, the fine mesh size used is not dependant on the power of the region being meshed. This may

978-3-9815370-8-6/17/$31.00 ©2017 IEEE

cause over meshing of some areas, leading to a bigger system without a real gain in the accuracy.

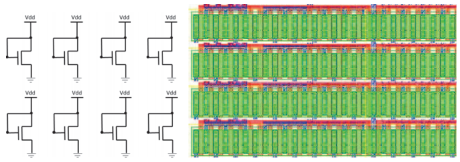
III. Proposed Algorithm

The core idea of the proposed algorithm is to detect high power areas ahead of the analysis, and design the mesh to be fine enough around these areas. Thus, we can get good enough results without any need for iteration. This done on two steps which will be described in the following two sections.

A. Technology characterization

From the error vs. power density graph in [11], we can conclude that the higher the power density in each block, the smaller the mesh size needed to mesh the area of this block. To do this, we have to characterize the technology node we are using. This is a step that is needed only once per technology node and is not design or layout dependent step. The goal of this step is to generate a lookup table, where we can find the smallest mesh size needed for each power density value.

The first step of the characterization is to find out the maximum power density possible for this technology pd_{max}. To achieve this, a dummy design is used. This design consists of an array of identical interdigitized transistors like those in figure 1. Each transistor has its source connected to ground, while the gate and the drain are connected together to VDD to allow the maximum possible current through the transistor. Finally, the power generated in the array is calculated then divided by the area of the layout to get pd_{max}. Since the end result is normalized to the area, the transistor size and the number of transistors used in the array are arbitrary.



(a) Schematic (b) Layout

Fig. 1: Array for maximum power density calculation

Now that the maximum power density in known, we can start the characterization steps. This is done by a set of thermal analysis iterations on the layout created earlier. Each characterization step aims to find the smallest possible mesh for a given pd. The first step goes as follows, we use the layout with the maximum power values calculated earlier. An iterative thermal analysis process is done. This process starts by doing an analysis with a coarse mesh, and keep refining it until the average error between the last two iterations is below a certain threshold, in our case 0.01 C^o was used. Once the threshold is reached, we record the final mesh size as the one corresponding to the power density used in this characterization step. For the other characterization steps, the process is exactly the same, except that we scale down the power used in the transistors by an arbitrary factor. We keep doing the characterization steps until the power used is below the power of a single transistor in the original layout. The flow chart for the whole characterization process is depicted in figure 2.

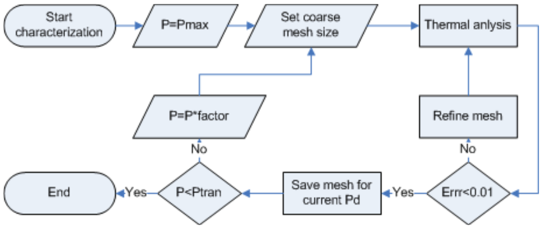
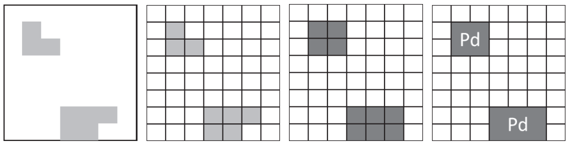


Fig. 2: Characterization process

B. Detection and meshing of high power areas

The main goal of this step, is to detect and mark the areas with high power density. Most of nowadays chips usually have their peak power consumption localized in a number of blocks in the chip, where the rest of the chip has much lower power consumption. The high power areas we are trying to detect are basically the bounding boxes of these blocks. The basic detection algorithm steps are summarized in figure 3. To explain the algorithm, let's assume we have a design with a couple of high power blocks as those marked in gray in figure 3a. The detection process starts by calculating the average power density for the whole design pd_{avg}. This is defined as the total power divided by the total area of the chip. The layout is then divided into small tiles. The power density inside each tile pd_{tile} also defined as the tile power divided by the tile area is then calculated. Next, we loop on all the tiles and select only those with power density greater than the pd_{avg} as in figure 3b. These selected tiles are geometrically merged together to form bigger polygons. We then calculate the bounding box for each polygon as seen in figure 3c. These bounding boxes are actually the rectangles bounding the power blocks, which is almost what we are looking for. The final step is to annotate each of these boxes with its power density as in figure 3d so as to be used during the meshing step. One final thing to note is that the smaller the tiling size the better. More tiles will lead to better resolution for the detection. On the other hand very small tiles will make the power detection step computationally heavy. A reasonable value for the tile size is $\frac{1}{100}min(width, length)$. Where width and length are the dimensions of the chip.



(a) Power map (b) Selected tiles (c) Bounding boxes (d) Detected areas

Fig. 3: High power areas detection

The final step remaining is to apply the meshing using the lookup table generated and the high power areas detected. In this step, we start by applying the biggest mesh size in the table for the whole layout. Next we loop on all the high power areas detected, for each area, we use pd calculated to lookup the mesh size from the table. If the exact pd value does not exists, we ceil pd to the nearest value before the lookup. Finally we refine the mesh at the area with the mesh size we looked up from the table. Once this is done for all the high power areas, we do the property extraction and then the thermal analysis.

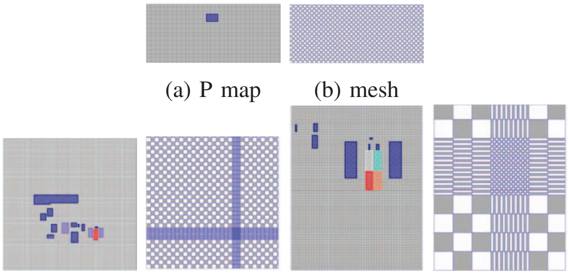
IV. Testing results

During the testing of the proposed algorithm, we had to evaluate its accuracy and performance. To evaluate the accuracy, we did a thermal analysis using our generated mesh, and compared it to a reference analysis. We then generated an error map. An error map is simply the absolute difference between the algorithm results and the golden results at every point in the layout. From the error map we then calculate the maximum and the average error in temperature. The reference result used above, is simply a result from a simulation with a much finer mesh. In order to decide this fine mesh size, we do a number of consecutive simulations starting with a coarse mesh, and reduce the mesh size with each run. We then calculate the error between the runs. We kept doing that until the maximum error between consecutive runs is below 0.1 C^o. This was done for each layout we used during the testing.

As for the performance, we measure the runtime of the algorithm itself along with the runtime of the solver using the generated mesh. For the runtime of the algorithm, this includes the meshing and property extraction. The results are then compared to the flux based algorithm from [9], [1], and [10] and the power aware algorithm from [11]. For the flux based algorithm we used a 0.1 C^o threshold for maximum error between each consecutive iterations. Since our focus is only on the effect of the proposed meshing algorithm on the results and system size, we can use any arbitrary FE solver without caring much about the solver performance itself. In our research we used the thermal solver from ANSYS® to examine the proposed meshing algorithm. For simplicity we used conforming rectangular mesh in our testing.

We used the designs from [11] to test the proposed meshing algorithm, all of which are full chips with various sizes, functionalities, and power consumption. All our simulations were steady state ones. They were done assuming free convection from all sides of the chip with an ambient temperature of 27 C^o. For each design, we show the power map, and output mesh from the proposed algorithm. This is followed by the temperature map from our golden solution, the temperature map from the proposed meshing algorithm, and the error map. Since all the maps are 3D, it may be hard to demonstrate it on a 2D view. For the sake of clarity, we drew only 2D maps, where all the maps are taken from the plan right at the top of the silicon substrate. We selected this position as it is where all the transistors -the heat sources- actually exist.

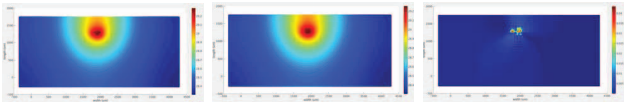
Our results are summarized in the following tables. In tables I and II we compare the runtime of the flux based and the power aware algorithms to the proposed algorithm. Followed by tables III and IV where we compare the error in the FE results from the proposed algorithm to the flux based and the power aware. In table V, we have a summary of the physical and electrical properties of the three chips we used in our simulations. This is followed by the lookup table generated from the characterization process in table VI. The error is based on the comparison to the reference analysis. Finally, table VII which summarizes the size of the FE problem from the reference mesh, the flux based algorithm, and our algorithm. One thing to be noted about the performance results, is that the time taken to do the meshing and the property extraction is larger than that taken to solve the resulting system. This is because the resulting system is not that big to solve. Since the proposed algorithm does the meshing step only once, it saves a lot of time compared to the iterative Flux based algorithm.



(a) P map (b) mesh

(c) P map (d) mesh (e) P map (f) mesh

Fig. 4: Power map and mesh for three chips



(a) Golden (b) Algorithm (c) Error map

Fig. 5: Chip 1 analysis results

V. Conclusions

An adaptive meshing technique suitable for steady state FE based thermal analysis of integrated circuits was presented. The algorithm is a non iterative one, based on power pre-characterization of the technology used. The algorithm showed around 40% speed up compared to the power aware algorithm and 75% speed up when compared to the flux based.

References

[1] Y. Yang, Z. Gu, C. Zhu, R. Dick, and L. Shang, “Isac: Integrated space-and-time-adaptive chip-package thermal analysis,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 26, no. 1, pp. 86–99, Jan 2007.

[2] S.-C. Lin and K. Banerjee, “An electrothermally-aware full-chip substrate temperature gradient evaluation methodology for leakage dominant technologies with implications for power estimation and hot-spot management,” in *Computer-Aided Design, 2006. ICCAD ’06. IEEE/ACM International Conference on*, Nov 2006, pp. 568–574.

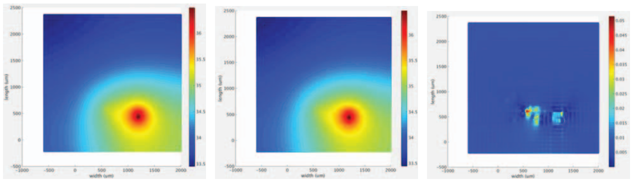
[3] P. Li, L. Pileggi, M. Asheghi, and R. Chandra, “Ic thermal simulation and modeling via efficient multigrid-based approaches,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 25, no. 9, pp. 1763–1776, Sept 2006.

TABLE I: Performance comparison to the flux based algorithm

Design	Flux Based algorithm (sec)				Proposed algorithm (sec)			Reduction
	iter. count	meshing	analysis	total	meshing	analysis	total	
Chip 1	5	681	8	689	152	2	154	77.6%
Chip 2	7	849	17	866	140	1	141	83.7%
Chip 3	6	59	11	70	18	0.5	18.5	73.5%

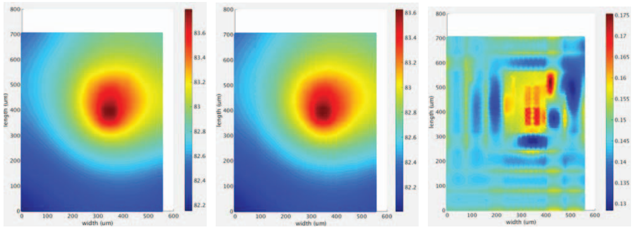
TABLE II: Performance comparison to the power aware algorithm

Design	Power aware algorithm (sec)			Proposed algorithm (sec)			Reduction
	meshing	analysis	total	meshing	analysis	total	
Chip 1	270	1	271	152	2	154	43.1%
Chip 2	241	2	243	140	1	141	41.9%
Chip 3	19	2	21	18	0.5	18.5	11.9%



(a) Golden (b) Algorithm (c) Error map

Fig. 6: Chip 2 analysis results



(a) Golden (b) Algorithm (c) Error map

Fig. 7: Chip 3 analysis results

TABLE III: Error compared to the flux based algorithm

Design	Flux based error		Proposed algorithm error	
	max	avg	max	avg
Chip 1	0.02 C^o	0.001 C^o	0.04 C^o	0.001 C^o
Chip 2	0.09 C^o	0.01 C^o	0.05 C^o	0.07 C^o
Chip 3	0.88 C^o	0.016 C^o	0.17 C^o	0.14 C^o

TABLE IV: Error compared to the power aware algorithm

Design	Power aware error		Proposed algorithm error	
	max	avg	max	avg
Chip 1	0.082 C^o	0.003 C^o	0.04 C^o	0.001 C^o
Chip 2	0.25 C^o	0.016 C^o	0.05 C^o	0.07 C^o
Chip 3	0.89 C^o	0.17 C^o	0.17 C^o	0.14 C^o

[4] Y. Zhao, S. Ho, and W. Fu, “A novel adaptive mesh finite element method for nonlinear magnetic field analysis,” *Magnetics, IEEE Transactions on*, vol. 49, no. 5, pp. 1777–1780, May 2013.

[5] Z. Cendes and D. Shenton, “Adaptive mesh refinement in the finite element computation of magnetic fields,” *Magnetics, IEEE Transactions on*, vol. 21, no. 5, pp. 1811–1816, Sep 1985.

TABLE V: Chips data summary

Design	Dimensions	Area	Power
Chip 1	4.6x2.0 mm	9.2 mm^2	72 mW
Chip 2	2.6x2.5 mm	6.5 mm^2	240 mW
Chip 3	560x700 μm	0.35 mm^2	170 mW

TABLE VI: Characterization results

$pd(W.m^{-2}) \times 10^6$	0.76	3.804	5.32	6.08	7.61
Mesh(μm)	80	40	20	10	10

TABLE VII: Simulated system size

Design	System nodes count		
	Reference	Flux based	Algorithm
Chip 1	1.129M	2.2K	4.78K
Chip 2	803.418K	5.13K	7.92K
Chip 3	1.172M	3.648K	2.046K

[6] J.-F. Remacle, P. Dular, F. Henrotte, A. Genon, and W. Legros, “Error estimation and mesh optimisation using error in constitutive relation for electromagnetic field computation,” *Magnetics, IEEE Transactions on*, vol. 31, no. 6, pp. 3587–3589, Nov 1995.

[7] J. F. Remacle, C. Geuzaine, P. Dular, H. Hedia, and W. Legros, “Error estimation based on a new principle of projection and reconstruction,” *Magnetics, IEEE Transactions on*, vol. 34, no. 5, pp. 3264–3267, Sep 1998.

[8] F. Meyer and D. Davidson, “Error estimates and adaptive procedures for the two-dimensional finite element method,” *Electronics Letters*, vol. 30, no. 12, pp. 936–938, Jun 1994.

[9] Y.-K. Cheng, P. Raha, C.-C. Teng, E. Rosenbaum, and S.-M. Kang, “Illiads-t: an electrothermal timing simulator for temperature-sensitive reliability diagnosis of cmos vlsi chips,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 17, no. 8, pp. 668–681, Aug 1998.

[10] S. Matsutomo, S. Noguchi, and H. Yamashita, “Adaptive mesh generation method utilizing magnetic flux lines in two-dimensional finite element analysis,” *Magnetics, IEEE Transactions on*, vol. 48, no. 2, pp. 527–530, Feb 2012.

[11] S. Abdelkader and A. E. El-Rouby, “Power-aware meshing algorithm for thermal analysis of integrated circuits,” *IEEE Transactions on Components, Packaging and Manufacturing Technology*, vol. PP, no. 99, pp. 1–9, 2016.