A Recovery Algorithm for Frequency-Domain Layered Finite Element Analysis of Large-Scale High-Frequency Integrated Circuits

Dan Jiao, *Senior Member, IEEE*

***Abstract*—A recovery algorithm is proposed for the frequency-domain layered finite element analysis of large-scale high-frequency integrated circuits. Given the solution in one layer, this algorithm can recover the solutions in other layers with single-ayer computational complexity. Numerical and experimental results are given to demonstrate its validity.**

***Index Terms*—Finite element method (FEM), high frequency, integrated circuits (ICs).**

I. Introduction

HIGH-frequency digital, mixed-signal, and radio frequency (RF) integrated circuit (IC) design demands accurate full-wave analysis for pre-layout design optimization and post-layout performance verification. However, traditional full-wave modeling techniques suffer from the well-known problems of large memory requirement and long CPU run time. Although efficient algorithms have been studied to mitigate this problem [1], very large-scale IC design: 1) demands very large-scale electromagnetic (EM) solutions, which cannot be offered by many current computational EM techniques and 2) imposes many unique modeling challenges that are totally new to the EM community [2]. Therefore, it is of critical importance to develop high-capacity EM methods amenable for ICs to drive continual very large scale integration (VLSI) revolution.

In [3], a layered finite element method (FEM) was developed for high-frequency modeling of large-scale 3-D on-chip circuits. In this method, first, the matrix system of the original 3-D problem is reduced to that of 2-D layers. Second, the matrix system of 2-D layers is reduced to that of a single layer. The computational complexity only involves solving a single layer irrespective of the original problem size. The method is shown to possess a high capacity to solve large-scale IC problems. In this letter, we propose a recovery algorithm to further improve the capability of the layered FEM. Given the solution in one layer, this algorithm can recover the solutions in other layers. In addition, the recovery scheme only involves single-layer computational complexity.

II. Formulation

The electric field $\mathbf{E}$ inside 3-D ICs satisfies the second-order vector wave equation

$$\nabla \times \left(\mu_r^{-1} \nabla \times \mathbf{E}\right) - k_0^2 \overline{\varepsilon}_r \mathbf{E} = -jk_0 Z_0 \mathbf{J} \quad \text{in } V \qquad (1)$$

subject to certain boundary conditions. A finite-element solution of (1) and its boundary conditions results in a matrix equation. The matrix can be extremely large for realistic on-chip circuit problems, which constitutes a computational challenge. To overcome this challenge, first we reduce a 3-D layered system matrix to a matrix that only involves 2-D surface unknowns in each layer [3]. If the reduced matrix is within the capability of available computational resources, we stop and solve the reduced system matrix as a whole without further reduction. If not, we continue to reduce the dimension of the system matrix to the size that we can handle such as a single-layer one [3].

The aforementioned method was developed based on the observation that in many IC design cases only one layer needs to be preserved in the final system matrix. For instance, for power grid design, transistor switching occurs at the bottom-most layer. This layer also features the maximum voltage droop. For RF IC design, the topmost layer is often the layer to be calculated because most RF components are fabricated therein. However, there are many other cases in which multiple layers/blocks are of interest. For example, transistor switching occurs in one circuit block while the crosstalk noise of interest is located in the other block. In this case, a recovery algorithm becomes necessary to obtain the solution in other layers/blocks from the layer/block being calculated. To describe the proposed recovery algorithm, we will begin with a general methodology description, and then proceed to the detailed formulation. In what follows, L denotes the number of layers; x_i denotes the unknowns on surface i; and layer i consists of surfaces i and $i+1$.

Now assuming after solving layer/block i we want to know the solution in layer/block j. Without loss of generality, we assume that layer i is to the left of layer j. As depicted in Fig. 1(a), we first project the contributions from the layers right to the jth layer to layer j. Mathematically, it resembles superposing a submatrix $\mathbf{P}_{R,j}$ on the original submatrix of layer j as shown in Fig. 1(b). We then translate the solution of the i-th layer to the j-th layer. It is equivalent to assembling a translation submatrix $\mathbf{T}$ which consists of $\mathbf{T_i}$ and $\mathbf{T_j}$. The resultant matrix is depicted in Fig. 1(c). With the solution in layer i known, the matrix system in Fig. 1(c) can be readily transformed to that shown in Fig. 1(d), in which matrix $\mathbf{T_i}$ is multiplied by x_{i+1}, which is known from the solution in layer i, to form the right hand side of layer j. Clearly, the resultant matrix system for layer j only involves single-layer unknowns, which can be readily solved.

Next, we give the formulation of $\mathbf{P}_{R,j}$ and $\mathbf{T}$. We represent the surface-based matrix in each layer as the form shown in

Manuscript received February 12, 2006; revised May 7, 2007. This work was supported by an Intel Corporation Grant.

The author is with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: djiao@purdue.edu).

Digital Object Identifier 10.1109/LMWC.2007.901752

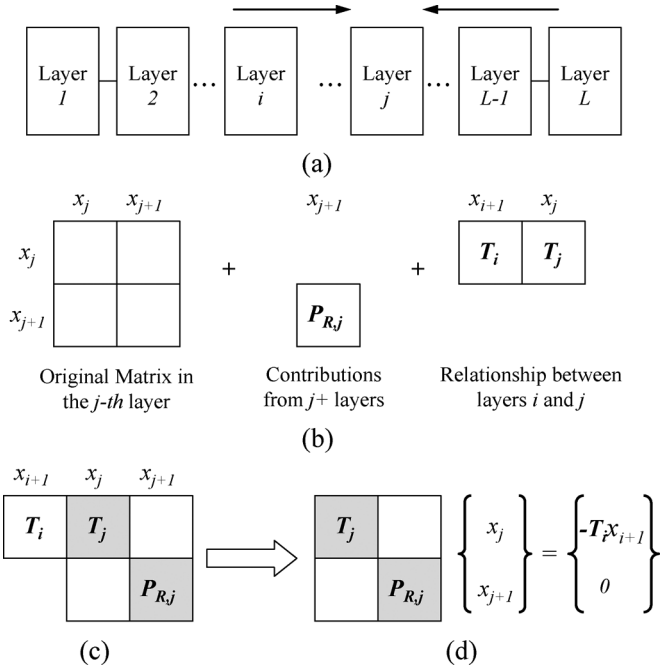


Fig. 1. Illustration of the recovery scheme.

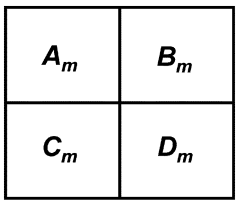


Fig. 2. Illustration of the surface-based matrix in each layer.

Fig. 2, in which m denotes the layer index. $\mathbf{P}_{R,j}$ can be obtained by the following numerical procedure:

$$\mathbf{D}'_L = \mathbf{D}_L.$$

From $m = L$ to $(j+1)$, do

$$\begin{aligned} \mathbf{P}_{\mathbf{R},m-1} &= \mathbf{A}_m - \mathbf{B}_m \left(\mathbf{D}'_m\right)^{-1} \mathbf{C}_m \\ \mathbf{D}'_{m-1} &= \mathbf{D}_{m-1} + \mathbf{P}_{\mathbf{R},m-1}. \end{aligned} \tag{2}$$

end

Matrix $\mathbf{T}$ can be obtained recursively as shown in (3).

$$\mathbf{T}_i = \mathbf{C}_{i+1}; \mathbf{T}_j = \mathbf{D}_{i+1}$$

From $m = (i+1)$ to $(j-2)$, do

$$\begin{aligned} \mathbf{T}_i &= -\mathbf{C}_{m+1}(\mathbf{T}_j + \mathbf{A}_{m+1})^{-1}\mathbf{T}_i \\ \mathbf{T}_j &= \mathbf{D}_{m+1} - \mathbf{C}_{m+1}(\mathbf{T}_j + \mathbf{A}_{m+1})^{-1}\mathbf{B}_{m+1}. \end{aligned} \tag{3}$$

end

The matrix operation in (2) is equivalent to eliminating all the unknowns on the surfaces right to layer j. The matrix operation in (3) is equivalent to eliminating the intermediate surface unknowns between layer i and layer j. Therefore, both (2) and (3) can be conducted using symmetric backward Gaussian elimination. The computational cost of (2) and (3) scales linearly with the cost of eliminating single-layer unknowns.

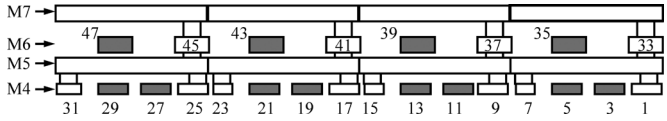


Fig. 3. Cross-sectional view of an on-chip power grid example (Horizontal axis is x, vertical axis is y).

The memory usage is also modest: linearly proportional to the storage of a single layer matrix.

If all the layers are of interest, we do not directly translate the solution in layer i to layer j, instead we recover the solutions layer by layer as the following:

$$\begin{aligned} x_{i-m} &= \mathbf{B}_{i-m}^{-1}\left[b_{i-m+1} - (\mathbf{A}_{i-m} + \mathbf{A}_{i-m+1})x_{i-m+1} \right. \\ &\quad \left. - \mathbf{B}_{i-m+1}x_{i-m+2}\right] \\ b_m &= 0; \quad m = 1, 2, \cdots, i-1 \quad \text{(For layers 1 to } i-1\text{)} \\ x_{i+m+1} &= \mathbf{B}_{i+m}^{-1}\left[b_{i+m} - (\mathbf{A}_{i+m-1} + \mathbf{A}_{i+m})x_{i+m} \right. \\ &\quad \left. - \mathbf{B}_{i+m-1}x_{i+m-1}\right] \\ b_{i+m+1} &= 0; \quad m = 1, 2, \cdots, L-i \,\text{(For layers } (i+1) \text{ to } L\text{)}. \end{aligned} \tag{4}$$

In (4), matrices $\mathbf{D}$ and $\mathbf{C}$ are not involved because for any layered structure, they are equal to $\mathbf{A}$ and $\mathbf{B}$, respectively in each layer [3]. Again, (4) only involves single-layer computational complexity.

III. Numerical Results

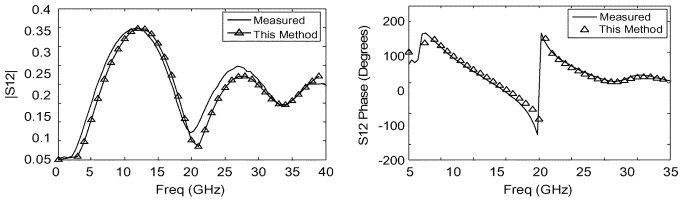
To validate the accuracy of the proposed algorithm, we considered a benchmark on-chip power grid example, the cross section of which is shown in Fig. 3. The structure involves four pitches and four metal layers from M4 to M7. The shaded wires are VSS (ground) rails, and the others are VCCs (power lines). A via exists wherever like rails cross each other. The wire width in M4 is 0.42 μm; that in M6 is 0.72 μm. The pitch is 7.2 μm. A length of 0.49 μm is considered in z direction (the direction that is perpendicular to the paper). Forty-eight ports are sampled at the near and far ends of the M4 and M6 wires. The ports are ordered from the right to the left. M4 wires are ordered first. For each wire, the near end is ordered first. In Fig. 3, the odd-number ports are labeled. The even-number ports are located at the far end of each wire.

Since the cross section in y–z plane has the minimal size, the layer growth direction is chosen to be x. Using the proposed method, a matrix block is formed for each pitch. In total four matrix blocks are formed. In each block, with the volume unknowns eliminated, only unknowns residing on the port surfaces are preserved, which renders six surface-unknown based submatrices. When one port is excited, only the associated matrix block is preserved in the system, with the contribution from other matrix blocks incorporated through the projection matrices. The solution in other layers is then recovered with the proposed recovery algorithm. Table I lists the S-parameters at 5 GHz simulated by the proposed algorithm in comparison with the results obtained from a conventional FEM. Excellent agreement is observed, which is as expected because the proposed recovery algorithm does not make any theoretical approximations.

Next, we extend the structure to incorporate 27 pitches along x. We also incorporate 16 power rails in M8 for C4 bump landing to model a realistic on-chip example. The 16

TABLE I
SIMULATED S-PARAMETERS IN COMPARISON WITH THE RESULTS OBTAINED FROM A STANDARD FEM

	S11	S12	S17	S1,17	S23, 45
Standard	-0.8655 +j3.2366e-004	0.1284 -j1.9149e-004	0.0894 -j9.2546e-007	0.0743 -j1.5077e-004	0.0914 +j2.6871e-005
This Method	-0.8655 +j3.2668e-004	0.1284 -j1.8880e-004	0.0894 -j1.6844e-006	0.0743 -j1.4943e-004	0.0914 +j2.4221e-005



(a) (b)

Fig. 4. Crosstalk of a 3-D on-chip interconnect structure. (a) Magnitude. (b) Phase.

wires are unevenly distributed in M8. In total, 388 ports are generated, which lead to 389 layers in the final system. Since every layer is of interest here, without the proposed recovery algorithm, we have to solve the 389 layers as a whole to obtain the crosstalk between ports, which failed on a 2 GB computer. With the proposed recovery algorithm, we only need to deal with single-layer computational complexity to extract the 388×388 S-parameter matrix. At 10 GHz, The calculated S_{11} is $-0.9071 + j0.0015$, $S_{1,216}$ is $0.046{+}j5.47e{-}4$, and $S_{1,385}$ is $2.67e-4-j3.607e{-}5$, where port 1 is the first M4 VCC port in the first pitch, port 216 is the M6 VCC port in the first pitch, and port 385 is the M8 VCC port in the 27th pitch.

Finally we simulated the crosstalk of a large-scale on-chip interconnect structure (E14 structure in [4], [5]) that was fabricated on a test chip using conventional Si processing techniques. In [3], we reduced the original system matrix to a single-layer one that consists of the two ports of interest to simulate the crosstalk. Here, we keep two layers and translate the solution of one layer to the other to examine the validity of the proposed recovery algorithm. As shown in Fig. 4, the proposed algorithm matches measurements very well.

IV. CONCLUSION

In this letter, we propose a recovery algorithm to obtain the solution in other layers from the solution calculated in one layer in the framework of the layered FEM. Both CPU run time and memory usage scale linearly with single-layer computational complexity. The algorithm is very powerful in performing crosstalk analysis of large-scale ICs, for which a standard FEM will be computationally prohibitive due to large memory requirement.

ACKNOWLEDGMENT

The author would like to thank M. J. Kobrinsky, Intel Corporation, for providing measured data.

REFERENCES

[1] W. C. Chew, J. M. Jin, E. Michielssen, and J. M. Song, Eds., *Fast and Efficient Algorithms in Computational Electromagnetics*. Norwood, MA: Artech House, 2001.

[2] D. Jiao, C. Dai, S.-W. Lee, T. R. Arabi, and G. Taylor, "Computational electromagnetics for high-frequency IC design," in *Proc. IEEE Int. Symp. Antennas Propag.*, 2004, pp. 3317–3320.

[3] D. Jiao, S. Chakravarty, and C. Dai, "A layered finite element method for electromagnetic analysis of large-scale high-frequency integrated circuits," *IEEE Trans. Antennas Propag.*, vol. 55, no. 2, pp. 422–431, Feb. 2007.

[4] M. J. Kobrinsky, S. Chakravarty, D. Jiao, M. Harmes, S. List, and M. Mazumder, "Experimental validation of crosstalk simulations for on-chip interconnects at high frequencies using S-parameters," in *Proc. IEEE 12th Topical Meeting Elect. Perform. Electron. Packag.*, 2003, pp. 329–332.

[5] M. J. Kobrinsky, S. Chakravarty, D. Jiao, M. C. Harmes, S. List, and M. Mazumder, "Experimental validation of crosstalk simulations for on-chip interconnects using S-parameters," *IEEE Trans. Adv. Packag.*, vol. 28, no. 1, pp. 57–62, Feb. 2005.