A Layout-Driven Yield Predictor and Fault Generator for VLSI

Alexander R. Dalal, *Member, IEEE,* Paul D. Franzon, *Member, IEEE,* and Michael J. Lorenzetti

Abstract—**IC faults arise from manufacturing defects. Layout analysis is required to determine the yield that is limited by these defects and the probability-graded fault lists that can be used to determine the optimal testing sequence. This requires 1) Accurate analytical expressions relating layout geometries to likelihood of opens and shorts appearing during fabrication, and 2) efficient CAD methods to extract these geometries from the layout. This paper describes all these aspects as implemented in one unified CAD tool. Accuracy is enhanced through the addition of edge information and the validation of shorting defects**

I. Introduction

DEFECTS arising during the manufacturing process may lead to faults in the fabricated ICs. Using defect monitors, it is possible to extract from the fabrication line the relative probabilities of different defect types occurring, and the probability distribution of defect count versus defect size [1]. This information can be combined with layout information to give the following:

1. Predicted circuit yield
2. Probability-graded fault lists

The predicted circuit yield is instrumental for calculating production cost and required volume [1] [2]. It can be used to guide the choice of layout style [3]. The probability graded fault lists can be used to shorten average IC testing times [5].

Not all manufacturing defects result in faults, as shown in Fig. 1. For this reason, a layout analysis tool is required to determine the defect-to-fault mapping for yield prediction and probability-graded fault list generation. Yield is calculated by ascertaining the *critical area* for each defect. In our calculations, we deal with opens and shorts only and assume defects to have a circular nature. As the circular assumption is also used when analyzing defect monitor data the actual defect shape is immaterial.

Manuscript received August 30, 1991; revised August 19, 1992.

A. R. Dalal was formerly with MCNC and North Carolina State University. He is now with Sun Microsystems Computer Corporation, 2550 Garcia Avenue, Mountain View, CA 94043-1100.

P. D. Franzon is with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh NC 27695-7911.

M. J. Lorenzetti was formerly with MCNC. He is now with Mentor Graphics Corporation, 8005 SW Boeckman Road, Wilsonville, OR 97070-7777.

IEEE Log Number 9205042.

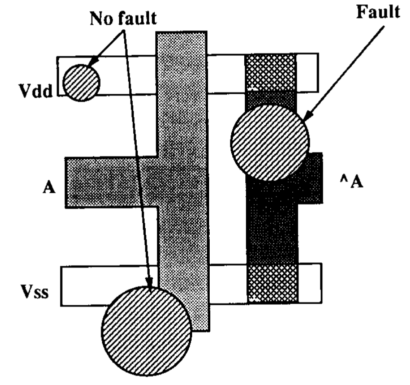


Fig. 1. Defect to fault mapping in a simple layout.

The definition of critical area is that area in which the center of a defect must fall in order to produce a fault. The critical area is always less than or equal to the IC area.

A probability-graded fault list is generated by listing the circuit faults along with the type and size range of the fault-inducing defect. Defect probability information, extracted from the process monitors, can then be used to probability-grade these faults. Ordering the anticipated faults by the order of occurrence allows reduced time spent on testing, because testing is usually terminated when the first fault is discovered.

There are a number of approaches to calculating critical area and/or generating probability-generated fault lists from the layout:

1. *Statistical Monte-Carlo Analysis:* [6] Graded fault lists are determined probabilistically. Post-processing is required to obtain the critical area. This approach is computationally expensive and useful only for small layouts.

2. *Virtual Mask Approach:* The layout is replaced with a simplified layout image, from which an approximate critical area only is calculated [2] [7]. It cannot produce fault information.

3. *Analytic Approaches:* Critical area is calculated by directly examining the layout mask and applying geometric analysis techniques. Two methods exist:

(a) *Scan Line Method:* The layout is scanned, in a similar fashion to a Design Rule Checker, to determine critical area only [8], or a non probability-graded fault list [9].

(b) *Layout Tiling Method:* Tiles indicating exact layout locations where defects may map into faults are generated. Geometric merging and resizing functions are then applied to arrive at an exact value of the critical area. The tiles can also be manipulated to provide fault information. Schvan [15] first described a method to carry this out, suggesting an iterative process. The method described here requires fewer steps and is more accurate.

In this paper we describe how we implemented the Layout Tiling Method to produce a CAD tool that calculates critical area with the same computational complexity as in the Scan Line method [8] and requires fewer steps than Schvan's approach [15], while simultaneously generating probability-graded fault lists with far greater efficiency than with the Monte Carlo approach [6].

We first develop the analytical models required for defect-to-fault mapping and critical area calculation. We then present the implementation of a critical area calculator that can accurately determine the defect critical regions and generate a graded fault list for dense VLSI circuit layouts. The complexity of the calculator algorithms is discussed next. Their efficiency stems from the fact that only a single check of the entire layout is needed.

II. Geometric Analysis of Critical Area

Defects occur during the water processing sequence and can lead to failures altering the electrical behavior of an integrated circuit. Two factors determine the likelihood of a defect causing a failure:

1. The probability of occurrence for defects of different sizes; and
2. The susceptibility of individual parts of the layout to defects.

Not all defects lead to failures in an IC. Instead, defects map into faults only if they fall in a critical area on the layout. The critical area is a function of the defect type and layout topology. The most common failures in layouts are: *Contact failures*, *intralayer shorts*, *interlayer shorts* and *intralayer opens* [11]. Modes describing the critical area for each type of failure are given in equations (1) through (4), [1], [10], [11]. The terms l, s, w, and d describe a layout of rectangular tiles with length l and width w, spaced apart by a distance s, on which defects of diameter d fall. As defect monitors measure the probability of contact failure, the critical area for contact failures is given in terms of the number of contact structures:

$$A_{\text{critical}_{\text{contact}}} \propto N_{\text{contacts}}. \tag{1}$$

For interlayer shorts and opens, the defect monitors measure defect probability and diameter distribution, and the critical areas associated with a single space between two lines and for a single line are:

$$A_{\text{critical}_{\text{short}}} = \begin{cases} 0 & \text{if } 0 \le d \le s \\ l(d-s) & \text{if } s < d \end{cases} \tag{2}$$

$$A_{\text{critical}_{\text{open}}} = \begin{cases} 0 & \text{if } 0 \le d \le w \\ l(d-w) & \text{if } w < d. \end{cases} \tag{3}$$

Equations (2) and (3) assume that wires with arbitrarily small widths and spacings are still good. If they are not (and width/space can be determined from measurement) then s in (2) and w in (3) are reduced appropriately. Under the reasonable assumption that interlayer shorts are caused solely by pinholes, the critical area is simply the cumulative overlap area between the two layers, i and j, involved,

$$A_{\text{critical}_{ij}} = A_{\text{overlap}_{ij}}. \tag{4}$$

The critical area for shorting or opening defects may be interpreted as a rectangular tile of length l and height ($h = d - s$) or ($h = d - w$), from (2) and (3), as indicated by the hatched area in Fig. 2. Defects of diameter less than s or w cannot cause shorting or opening faults, respectively. Thus, narrow spacings or widths imply a smaller minimum required defect size, $d_{\min}$,to cause a fault than large tile spacings or wide tiles. The tile height in Fig. 2 varies linearly with the defect diameter d and is offset by the spacing s or the tile width w, for a given value of d. Mask regions where tiles are narrowly spaced are characterized by critical area rectangles with greater height for a given defect size, than regions with wide spacing between adjacent tiles. We can now establish a direct correlation between tile spacing s, minimum required defect size $d_{\min}$, and critical area height ($d - s$), for different values of s. A smaller tile spacing means that a smaller minimum defect size is required to cause a fault. Thus, the critical area is greater for a given defect size. In a formal notation:

$$\begin{aligned} s_1 < s_2 &\Rightarrow d_{\min_1} < d_{\min_2} \\ &\Rightarrow (d - s_1) > (d - s_2)\ \forall d > 0. \end{aligned} \tag{5}$$

A similar argument applies to opening faults. Data from defect monitors has indicated a $1/x^m$ distribution of frequency of occurrence versus defect size [1], where the value of m is typically $2.8 < m < 3.1$. Smaller defects are more likely to occur and therefore, layout regions with small tile widths or inter-tile spacings exhibit greater susceptibility to failures than regions with large tile widths or spacings, because $d_{\min}$ is smaller than the former case. For the purpose of fault grading according to likelihood of occurrence, all that is needed then, is a sorted list of layout critical area regions, ordered by a minimum defect size necessary for failure. Fault grading is performed in three steps:

Step 1: Layout tiling into rectangular regions: A *manhattan style* VLSI mask, where conductors and active regions may only feature exact 90° bends, can be decomposed into a physically identical mask of purely rectangular tiles. Polygons are tiled into rectangles by the use of algorithms employed for Design Rule Checking. Such a layout may be described in set notation by two sets of elements and an *adjacency* mapping relationship be-

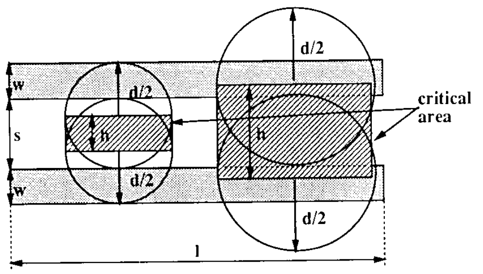


Fig. 2. The critical area for shorts between adjacent lines is given by $h \times l$ when the defect diameter d is greater then the spacing s.

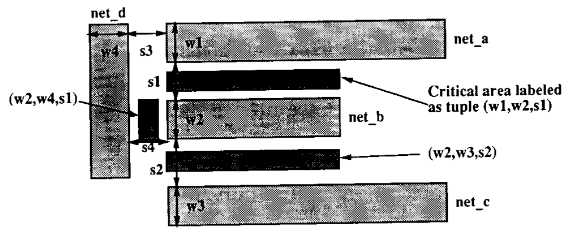


Fig. 3. Adjacencies between conductors. The critical areas for a defect size $d > d_{\min}$ and for tuples associated with net$_b$ are shown.

tween them for each layer, such as diffusion, polysilicon and metal. For layer l, the sets are:

$$W_l = \{w_{l_0}, w_{l_1}, \cdots, w_{l_n}\}, \tag{6}$$

the set of all rectangular conductor segments, and

$$S_l = \{s_{l_0}, s_{l_1}, \cdots, s_{l_n}\}, \tag{7}$$

the set of all rectangular spacings between adjacent conductors in W_l. Each member of W_l and S_l has associated with it relevant geometrical information, such as length, height and absolute layout coordinates. The adjacency mapping relation forms 3-tuples $(w_{l_i}, w_{l_j}, s_{l_h})$ or $(s_{l_i}, s_{l_j}, w_{l_h})$, consisting of two adjacent, non-abutting conductor rectangles and the spacing between them, or two adjacent, non-abutting spacing tiles and the conductor between them, for shorting or opening faults, respectively. A wire segment w_{l_i}, may occur in more than one tuple, because it may be adjacent and non-abutting to more than one other segment, as indicated for conductor net$_b$ in Fig. 3.

Step 2: Extraction of $d_{\min}$ for each region: Each 3-tuple uniquely defines a potential critical area region for defects of the appropriate diameter. The minimum required defect size for failure in the critical area region, described by a tuple, may be found from the geometrical parameters associated with the members of the tuple.

Step 3: Sorting the list of step 2 in order of increasing $d_{\min}$: The probability of failure is inversely proportional to $d_{\min}$. The list of critical areas obtained in the previous step may thus be sorted in order of increasing $d_{\min}$ values to produce a probability-graded list of faults for the layout.

Note that this list of faults may be extracted very efficiently. Only a single pass over the layout is required: Tiling the mask into rectangles, performing parameter extraction and sorting. These lists of faults are indexed by layout feature. To convert this into a list indexed by circuit element it is necessary to determine the probability of a defect of size greater than $d_{\min}$, add the failure probabilities for different layout features associated with the same circuit element and resort the new list. This is not currently implemented.

The individual defect critical regions for a mask layer can now be processed further to obtain the total critical area at different defect sizes. Critical area extraction is performed in four steps:

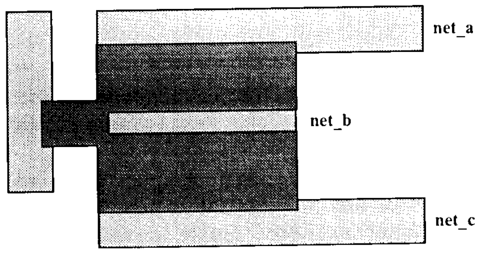


Fig. 4. Merged critical areas for a larger defect size.

Step 1: Critical region selection: The ordered list of tuples for shorting and opening faults on layer i are scanned to select only those regions with a value of $d_{\min}$ less than the current defect diameter for critical area processing.

Step 2: Critical area tile generation: For each of the selected regions, a rectangle of length l and height $(d - s)$, in the case of shorting faults, or $(d - w)$ in the case of opening faults, is generated. For example, the critical short areas associated with net$_b$ and a certain defect size are shown in Fig. 3.

Step 3: Tile merging and retiling: These rectangles are merged and retiled to account for possible tile overlapping. For example, the merged critical areas for a larger defect size are shown for net$_b$ in Fig. 4. (Note this step was not done in the fault grading process as tiles of size $d_{\min}$ will have little overlap.)

Step 4: Critical tile area summation: The areas of the resulting rectangles are summed up to arrive at the total critical area value for shorting or opening faults on layer i, due to defects of diameter d.

For the next larger defect size, d', the existing rectangles from the previous iteration are simply resized to reflect the increased tile height:

$$\begin{aligned}\Delta h &= (d' - s) - (d - s) \\ &= d' - d.\end{aligned} \tag{8}$$

Additionally, the sorted lists of tuples are scanned again to generate tiles for regions which may have become defect critical at the new defect diameter. All rectangles are again checked for overlapping before being summed up to give the new value of the critical area.

III. Implementation

The fault list generator and critical area calculator are implemented in the *Cadence* environment [13] because it already incorporated the required polygon manipulation routines as part of the design rule checker, thus saving programming effort. Each physical shape in a VLSI layout is represented by an entry into the design database, which stores geometrical, physical, and electrical properties of the object. The program flow for fault generation and critical area calculation are similar, so the steps for the two are combined to the maximum extent possible. The sequence of operations performed is described below:

Step 0: Initially, the total area of the layout, which the critical area must never exceed, is calculated from geometrical information available in the design database.

Step 1: Individual layers are preprocessed.

1. All overlapping shapes on a layer are merged. This step assures us that all segments belonging to the same electrical circuit node are a single physical object in the database, and is important for *tile validation*. Merging is necessary because the Cadence framework does not support a canonical representation similar to *Magic* [14].

2. An initial set of critical area tiles is generated by tiling the shapes on the respective layer themselves, in the case of opening faults, or the spacing between the shapes, in the case of shorting faults.

3. In the case of shorting faults, tile validation is performed on this initial set of rectangles. Tile validation is required to ascertain that defect critical regions for shorting faults between conductors are generated only for conductors in different electrical nets. Tile A in Fig. 5 is a valid shorting fault tile, while tile B is not, because it shorts together two wire segments in the same set. Such situations are not uncommon. For example, two successive jogs in polysilicon to avoid a metal to thinox via or a short layer change. Tile validation implements the adjacency mapping previously mentioned. The database is queried to find the neighbors of each potential critical area tile, on the current layer. If they are in the same net, the tile is invalid and is removed from the database. After completion of tile validation, we have the tuples required for fault-list generation and critical area calculation.

Step 2: The valid tiles obtained in the previous step are sorted by their small dimension for fault grading purposes. It is assumed that the small dimension of each tile corresponds to $d_{\min}$.

Step 3: The tiles are prebloated to account for the *edge effect*. For opening and shorting faults, circular defect critical regions exist on the sides of the rectangular tiles, as demonstrated in Fig. 6. These regions are termed *edge regions* [11] and are bound by circular arc segments, which also grow with increasing defect size. The rectangular tiles from the previous step are stretched by a precalculated factor on either side, to account for the edge regions without the need to generate, resize, and measure circular regions, which is significantly more compute-intensive. The equivalent area added is [11]

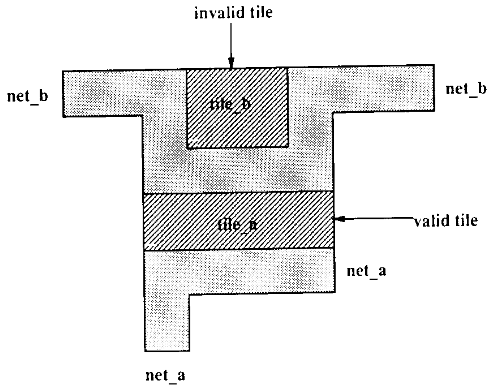


Fig. 5. Tile validation.

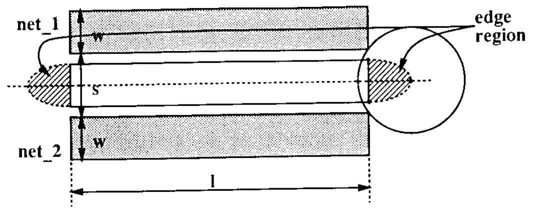


Fig. 6. Edge effect for shorting faults.

$$A_{\text{critical}} = \pi \frac{d^2}{16} + \left(\frac{d}{2}\right)\left(\frac{d}{2} - s\right). \tag{9}$$

Step 4: The rectangular tiles are now iteratively resized for each defect in a list sorted by increasing defect diameter, and the critical area calculated after each iteration.

1. Due to the sorting of tiles by increasing $d_{\min}$, the list need only be processed up to the first tile which does not meet the existence criterion for critical area:

$$d_{\min} \leq d \tag{10}$$

For all tiles obeying (10) the dimension corresponding to $d_{\min}$ is resized to be the same as A_{critical}/l as given in (2) and (3).

2. After resizing, all tiles are merged and tiled again to avoid double counting of possible overlap areas.

3. The total critical area for the current defect size is then calculated as the sum of the areas of the rectangles generated in the previous step.

This step is repeated for all defect sizes at which critical area information is desired. If the calculated critical area exceeds the total chip area for a given defect diameter, the program indicates this and simply sets the critical area equal to the total layout area for this and all remaining defect sizes.

IV. Complexity Analysis

The critical operations in the calculations are polygon tiling, rectangle merging and resizing and sorting. Sorting is performed by the *Mergesort* algorithm, which has a worst-case complexity $O(N \log (N))$. Resizing N rectangles has complexity $O(N)$. It may be performed iteratively by adding the sizing factor to coordinates of the upper, left-hand corner, and subtracting it from the coordinates of the lower, right-hand corner for each rectangle. We do not know the complexity of Cadence's polygon tiling and rectangle merging algorithms, but the best published result for the latter operation [12], which is the critical one, is $O[N \log (N) + p \log (N^2/p)]$, where N is the number of rectangles, and p is the number of intersecting edges. In the former case, polygons may be tiled into rectangles by sorting the coordinates of the polygon vertices in an x-coordinate and y-coordinate order and then performing comparisons between the two lists, all of which have worst-case complexity of $O(N \log (N))$. Thus, the overall complexity of the Layout Tiling Method is similar to existing implementation of the Scan Line Method [8] for critical area calculation. The overall complexity is also similar to Schvan's layout tiling approach, except that Schvan's approach requires multiple passes over the layout, where ours requires only one. Thus our approach is better by a constant factor. Being deterministic with complexity of $O[N \log (N)]$, our fault grading approach is significantly faster than VLASIC [6], and allows automated, probability-graded fault list generation for sizable VLSI layouts.

Unfortunately, the actual runtime is only reasonable because we had only an interpreted version of the polygon manipulation routines interface available to us. For example, the analysis of an 8,000 transistor chip consumed 3.5 hours of CPU time on a Microvax II under Ultrix 2.2. We estimate that use of the compiled versions of these routines would allow an order of magnitude speed improvement.

V. Results

The program produces a two-part report. Part one generates a list of the names of circuit nets shorted or opened, along with the minimum required defect size for each fault to occur, sorted in ascending order of $d_{\min}$. The list may then be postprocessed with the McYield [16], [7] system, which allows defect density information from the process line to be entered, to obtain an accurate grading of faults, according to the likelihood of occurrence. The second part of the output is a table of layout critical area versus defect diameter, for each selected defect type and layer. Again, this data is in a McYield-compatible format, and McYield is used to calculate the actually predicted yield from the critical area data.

VI. Conclusion

We have presented an efficient approach to probability-graded fault list generation, and critical area calculation for IC yield production. The approach was also efficient to program because it was built on top of existing design rule checking routines. No critical area estimation tool provides exact answers. However, here the accuracy of the tool is enhanced by including in the critical area calculations adjustments for defects occurring at the end of a feature and validating shorts before including the associated critical area in the sum. It would be possible to make the approach more efficient by going to an entirely graph-based approach [11], thus avoiding the physical tile generation step.

References

[1] C. H. Stapper, "Modeling of defects in integrated circuit photolithographic patterns," *IBM J. Res. Develop.*, vol. 28, no. 4, pp. 549-557, July 1984.

[2] W. Maly, "Modeling of lithography related yield losses for CAD of VLSI circuits," *IEEE Trans. Computer-Aided Design*, vol. CAD-4, no. 3, 166-177, July 1985.

[3] M. J. Lorenzetti, "The effect of channel router algorithms on chip yield," *Proc. MCNC Int. Workshop on Layout Synthesis*, May, 1990.

[4] A. V. Ferris-Prabhu, "Yield implications and scaling laws for submicrometer devices," *IEEE Trans Semicond. Manufact.*, vol. 1, no. 2, pp. 44-61, May 1988.

[5] W. Maly, "Optimal Order of the VLSI IC Testing Sequence," *Proc., ACM/IEEE Design Automation Conf.*, June 1986, pp. 560-566.

[6] D. M. H. Walker, *Yield Simulation for Integrated Circuits*. Boston: Kluwer, 1987.

[7] M. J. Lorenzetti, P. Magill, A. Dalal and P. Franzon, "McYield: A CAD Tool for Functional Yield Projections for VLSI," in *Proc. 1990 Int. Workshop on Defect and Fault Tolerance in VLSI Systems*, Nov., 1990 pp. 100-110.

[8] J. P. de Gyvez and J. A. G. Jess, "A layout defect-sensitive extractor," in *Dig. Tech. Papers, IEEE Int. Conf. Computer-Aided Design*, Nov. 1989, pp. 538-541.

[9] P. Nigh and W. Maly, "Layout-driven defect-sensitive extractor," in *Dig. Tech. Papers, IEEE Int. Conf. Computer-Aided Design*, Nov. 1989, pp. 154-157.

[10] A. V. Ferris-Prabhu, "Modeling the critical area yield forecasts," *IEEE J. Solid State Circuits*, vol. SC-20, no. 4, pp. 874-880, Aug. 1985.

[11] A. R. Dalal, "An Exact Critical Area Calculator and Fault Generator for VLSI Layouts," M.S. thesis, North Carolina State University, 1990.

[12] F. P. Preparato and M. I. Shamos, *Computational Geometry. An introduction*. New York: Springer-Verlag, 1985.

[13] Cadence Design Systems, *Edge, Design Framework Manual, Vol. 1*, 2.

[14] G. S. Taylor and J. K. Ousterhout, "Magic's incremental design-rule checker," in *Proc. 21st ACM/IEEE Design Automation Conf.*, June 1984, pp. 160-165.

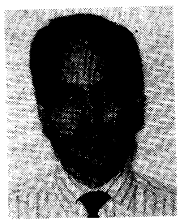
[15] P. Schvan, D. Y. Montuno, and R. Hadaway, "Yield projections based on electrical fault distribution and critical structure analysis," in *Proc. Int. Workshop on Defect and Fault Tolerance in VLSI Systems*, Springfield, MA, Oct. 6-7, 1988.

[16] M. Lorenzetti, P. Magill, A. Dalal, and P. Franzon, "Functional Yield Projections Based on Efficient Extraction of Critical Areas," MCNC Tech. Rep. TR89-53.

Alexander Dalal (M'91) received the B.S. degree in electrical engineering from the University of Notre Dame in 1988 and the M.S. degree in computer engineering from North Carolina State University in 1990.

In 1990, he joined Intel Corporation's Microprocessor Components Group and in 1992 moved to Sun Microsystems Computer Corporation. In both of these positions he has been involved in the development of floating point arithmetic circuits and full chip power network design and analysis. His research interests include low-power BiCMOS circuits, and automated circuit simulation environments.

Mr. Dalal is a member of Eta Kappa Nu and Tau Beta Pi.



Paul D. Franzon (S'85-M'88) received his B.Sc. degree in Physics, B.E. (Hons) and Ph.D. degrees in electrical engineering from the University of Adelaide, Adelaide, Australia in 1983, 1984, and 1990 respectively.

Before completing the Ph.D. he had worked at AT&T Bell Laboratories in Holmdel, NJ in 1986 and 1987 and with the Australian Defense Science and Technology Organization. In these positions he worked on problems in wafer scale integration, IC yield modeling, and VLSI design. In 1987 he cofounded Communica Data Communications Engineers. Since 1989, he has been an Assistant Professor in the Department of Electrical and Computer Engineering at North Carolina State University. His research interests include CAD and CAE for signal integrity management in packaging ICs, for design-for-manufacturability, and for system-level design, as well as design methodologies and techniques for very high speed and low power circuits. He is the editor and author of a book on MCMs.

Dr. Franzon is a member of ACM and ISHM.

Michael J. Lorenzetti received the B.S. degree in electrical engineering from the Illinois Institute of Technology in 1978 and the M.S. and Ph.D. degrees in electrical engineering from The University of Texas at Austin in 1980 and 1983, respectively.

He has been active in research and development of physical design automation algorithms at VR Information Systems (1978-1983), The GE Microelectronics Center (1983-1985) and MCNC (1985-1990) where he was also an Adjunct Associate Professor in the Electrical and Computer Engineering Department at North Carolina State University. He is currently developing physical design tools at Mentor Graphics Corporation.

Dr. Lorenzetti is a member of the ACM.