Yield Analysis of Nano-Crossbar Arrays For Uniform and Clustered Defect Distributions

Onur Tunali
Istanbul Technical University, Turkey
onur.tunali@itu.edu.tr

Mustafa Altun
Istanbul Technical University, Turkey
altunmus@itu.edu.tr

***Abstract*—During the fabrication of nano-crossbar arrays, certain amount of defective elements are introduced to the end product which affect the yield drastically. Current literature regarding the yield analysis of nano-crossbar arrays is very rough and limited to the uniform distribution of defect occurrence with a few exceptions. Since density feature of crossbar architectures is the main attracting point, we perform a detailed yield analysis by considering both uniform and non-uniform defect distributions. Firstly, we briefly explain the present algorithms and their features used in defect tolerant logic mapping. Secondly, we explain different defect distributions and logic function assumptions used in the literature. Thirdly, we formalize an approximate successful mapping probability metric for uniform distributions and determine area overheads. After that, we apply a regional defect density analysis by comparing uniform and clustered defects to formulate a looser upper bound for area overheads regarding clustered distributions. Finally, we conduct extensive experimental simulations with different defect distributions.**

***Index Terms*—Nano-crossbar; Area Yield; Defect Tolerance**

I. INTRODUCTION

Advancements in nanofabrication produce novel emerging technologies as an answer to longstanding integration and miniaturization issues of electronic circuits. These developments lead to programmable circuit architectures based on nano-crossbar arrays which operate similarly to conventional programmable logic arrays (PLA's), molecular switch crossbar arrays, and resistive crossbar logic [1] and [2]. Two fully operational implementations as a nanoprocessor and a finite-state machine are shown to be feasible in [3] and [4].

In short, a nano-crossbar array is constructed from two layers of orthogonal wires/lines. Every crosspoint/junction acts as a switching element [2]. Since nano-crossbars are dominantly produced with bottom-up fabrication techniques that generates uniform and dense structures, higher defect rates are more likely to occur. Projected maximum defect rates present in end products are potentially to deviate between 15% and 20% [1]. Furthermore, physically realized structures given in [3] and [4] confirm this range.

High defect rates of nano-crossbars complicate the logic mapping that generally necessitates area overheads [5]. In this paper, we use the term *yield* as the percentage of a nano-crossbar utilized to realize a given logic function. For example, a logic function can optimally be realized with a 6×4 size crossbar and if an efficient algorithm is able to find a valid mapping in the presence of defects with the same size, yield is 100%. In a different scenario, if a less efficient algorithm is able to find a valid mapping with at least 12×8 size crossbar, yield is $\frac{6\times4}{12\times8} = 25\%$. Since yield values are directly related to the efficiency of the used algorithm, we implement two state-of-the-art defect tolerant logic mapping algorithms to evaluate our yield results for a reliable analysis.

A. Previous Works

Yield analysis of nano-crossbars for uniformly distributed defects is first conducted partially in [6] as an area overhead calculation. Even though necessary size of a nano-crossbar for a successful logic mapping is presented as a lower bound, we show that this limit is too generous. Required yield is more than adequate for most of the cases. Second comprehensive yield analysis is conducted in [7]. They performed an elementary logic mapping algorithm with a long runtime constraints to acquire reliable results. Nevertheless, inefficiency of the methods results in poor yield outcomes ranging between 5% to 20%. Furthermore, benchmark logic functions chosen for simulations have rather small sizes, so results are unlikely to be realistic for logic functions with larger sizes.

Apart from these works, a clustered distribution specific study is presented in [8] which also used an elementary heuristic algorithm for logic mapping. This approach also produced low yield results as well. Another important tendency worth mentioning, majority of logic mapping methods in the literature utilize 2.25 times area overheads [5]. We show that this practice is an excessive measure and disregards the features of logic functions and nano-crossbars. Our method specifically tailors the yield for maximum values considering a variety of parameters such as defect rates, logic function sizes, and area overheads.

B. Contributions and Organization

Our contributions are as follows: 1) We cover both uniform and clustered defect distributions; 2) We introduce an approximate formalization for an optimized yield considering uniform distributions; 3) We propose a method to examine defect distributions by dividing crossbars into sub regions; 4) By comparing uniform and clustered defects, we formulate a loose upper bound for yield considering clustered distributions; and 5) We show that our method is adaptive to changing parameters of logic functions and nano-crossbars.

The rest of the paper are as follows. In Section II, we give preliminary concepts. In Section III, we elaborate on different defect distributions and logic functions. In Section IV, we study our yield model. In Section V, we present experimental results and Section III concludes the paper.

This work is supported by the EU-H2020-RISE project NANOxCOMP #691178 and the TUBITAK-Career project #113E760.

978-1-5386-1911-7/17/$31.00 ©2017 IEEE

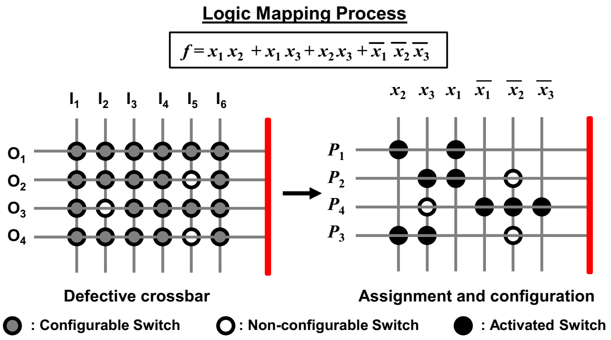
Logic Mapping Process
$f = x_1x_2 + x_1x_3 + x_2x_3 + \overline{x_1}\,\overline{x_2}\,\overline{x_3}$
I_1 I_2 I_3 I_4 I_5 I_6
O_1
O_2
O_3
O_4
x_2 x_3 x_1 $\overline{x_1}$ $\overline{x_2}$ $\overline{x_3}$
P_1
P_2
P_4
P_3
Defective crossbar
Assignment and configuration
: Configurable Switch
: Non-configurable Switch
: Activated Switch

Fig. 1. Defect tolerant logic mapping of a given logic function.

II. PRELIMINARIES

A. Nano-crossbar Model

A nano-crossbar array consists of two layers of orthogonal lines. Vertical and horizontal lines can be regarded as inputs and outputs, respectively. Crosspoints/junctions of an array are utilized as switching elements. A switch can be *activated* and *deactivated* according to preference. A distinction between a functional and a defective crosspoint switch is stated as follows: 1) *Configurable (functional)*: Switch works properly, so both activation and deactivation are possible; and 2) *Non-configurable (defective)*: Switch is defective, so it is always on a deactivated mode. This defect model is also named as *stuck-at open*. In terms of logic mapping, this type of defects are more manageable. In addition, it is possible to manipulate fabrication process in favor of non-configurable switches. For this reason, this paper only considers stuck-at open defects.

B. Logic Function

A *logic function* is a two-level Boolean function consist of *literals* (variables and their negations) in a Sum-of-Products form, for example $f = P_1 + P_2 + P_3 + P_4 = x_1x_2 + x_1x_3 + x_2x_3 + +\overline{x_1}\ \overline{x_2}\ \overline{x_3}$.*Optimal area size* of a logic function is defined as the minimum area for the realization of the function as $M \times N = (\#\ of\ products \times \#\ of\ literals)$. For example, $f = x_1x_2 + x_1x_3 + x_2x_3 + +\overline{x_1}\ \overline{x_2}\ \overline{x_3}$ has 4 products and 6 literals, so $M = 4$ and $N = 6$. *Logic Inclusion Ratio* (IR) is defined as a ratio of the literal count of a function to the optimal area size. For example, if f has a literal count of 9 and optimal area sizes are $M = 4$ and $N = 6$, then IR = 9 /24 = 37%. For logic mapping, IR shows percentage of the switches used to realize a given function in optimal area. Conversely, (1-IR) shows the the percentage of the unused switches.

C. Logic Mapping Process

In order to map a logic function to a nano-crossbar, input and output lines are used as literals and products of a logic function. In a defect-free nano-crossbar, every switch is configurable, so assignment and configuration processes for a logic function realization are straightforward. However, defects require a mapping process to avoid non-configurable switches. Here, we should consider that if a literal is included in a product, crosspoint at the intersection of the corresponding input and output lines must be activated; otherwise, corresponding switch is left as deactivated.

A successful mapping means that no literal is assigned to a non-configurable switch. In Figure 1, a successful mapping process is shown. Different mapping methods such as integer linear programming, satisfiability, graph embedding and bipartite matching are used [5]. In this paper, we choose two approaches: a greedy heuristic method with matrix matching having the smallest runtime [9] and a memetic algorithm using bipartite matching having the highest success rate [10].

D. Nano-crossbar Yield

We define the yield as the percentage of a nano-crossbar utilized to realize a given logic function. We use area overhead coefficients concerning the input and output lines, for our yield formulation.1) *Input and output area overhead coefficients* (k_o, k_i) show the number of used input and output lines to the optimal ones. For example, $k_o = 1.5, k_i = 1.5$ means we have a crossbar size of $1.5.M \times 1.5.N$ where $M \times N$ is the optimal size. 2) *Yield* is the ratio of optimal area size to the used area overhead as $\frac{M.N}{k_o.M.k_i.N} = \frac{1}{k_o.k_i}$. For example, when an optimal size crossbar is used for a logic mapping, $k_o = k_i = 1$ and $\frac{1}{k_o.k_i} = 1$ which means 100% yield. When area overheads are $k_o = k_i = 1.1$, yield decreases to $\frac{1}{k_o.k_i} = 0.82$ which is 82%.

III. DEFECT AND LITERAL DISTRIBUTIONS

In uniform distributions, defect occurrence probability on every switch is a constant and an independent value. Figure 2 (c) shows the defect patterns of an uniform distribution. Black points indicate configurable switches and white points indicate defective (non-configurable) switches. For clustered distributions, defects tend to cumulate around each other, so it is not suitable to use a constant defect rate. We use a general clustering modeling used in [11] also adopted in [8].

In short, this model initializes random probabilities for each switch at the beginning and then appoints new defect probability to each switch by using the formulation in (1). Choosing the range of neighbors determines the clustering patterns of a distribution. Figure 2 (a)-(b) show different clustering patterns of defect distributions. Mathematical notation of introducing a defect to a switch during a time interval Δt is as follows:

$$p(\Delta t|k, l_1, ...k_n) = c(x, y) + bk + \sum_{l=0}^{n} b_i l_i \qquad (1)$$

where $c(x, y)$ is a susceptibility factor, b is a global clustering factor, b_i is a local clustering factor, k is the number of defects already present, and l_i is the number of defects present in neighboring circuit area. We use the same values of the given parameters as chosen in [8] with the same defect rate as 20%.

In addition, we also study the literal distribution of logic functions. In Figure 2 (d), literal distribution, meaning activated switches, of a logic function is shown. Black points indicate activated switches and white points indicate unused switches. Before explaining the distribution of literals, we remind the reader that nano-crossbars have both variables and their negations as input lines as shown in Figure 1. An important feature affecting the distribution of literals is that a product

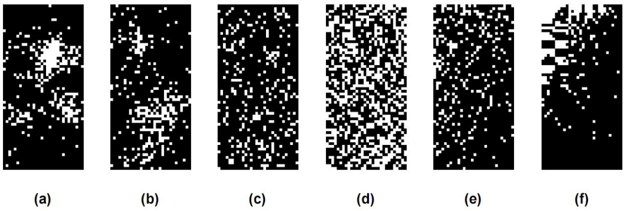

Fig. 2. Defect rate is 20% : (a) - (b) denser to looser clustering patterns (c) uniform distribution (d) a logic function and (e) -(f) sorted uniform and clustered distributions.

of a logic function cannot have a variable and its negation at the same time. Based on this observation, two constraints for the distribution of switches are as follows: 1) At most, half of the switches are activated in an output line; and 2) Distribution of activated switches in an output line cannot be fully random since an activated switch corresponding to a specific input line necessitates a deactivated switch corresponding to the negated input line. It is important to comply with these constraints while generating a random logic function, otherwise yield analysis simulations do not reflect the true nature of results.

IV. YIELD ANALYSIS OF NANO-CROSSBARS

A summary of yield parameters is as follows:

$IR_{\{r\}}$: Logic inclusion ratio of the r^{th} product
$M \times N$: Optimal size of an array
k_i : Input coefficient $\quad k_o$: Output coefficient
P_d : Defect rate $\quad Psuc$:Probability of a successful mapping
A_i: Number of defective switches in i^{th} region
D_i : Density of i^{th} region

A. Uniform Distribution

Yield is closely related to matching probability of products of a logic function with outputs of a nano-crossbar. In order for a product to be successfully matched to an output line, all activated switches must be defect-free. Since $IR_r.N$ gives the number of activated switches in a product, $(1-P_d)^{IR_r.N}$ gives the probability of successful matching of a product. Therefore, if an array has M different output lines, matching probability of a product to an output line can be formulated with $[1-(1-(1-P_d)^{IR_r.N})^M]$. Finally, multiplication of matching probability of all products gives the probability of a successful mapping. Formulation is:

$$Psuc = \prod_{t=0}^{M-1} \left[1-(1-(1-\frac{P_d}{k_i})^{IR_{\{t+1\}}.N})^{M.k_o-t}\right]. \quad (2)$$

Additionally, regarding (2) $M.k_o$ shows the number of output lines including area overhead and $\frac{P_d}{k_i}$ shows the decrease in the defect rate to take into account that mapping with redundant input lines are easier. Eventually, finding k_i and k_o maximizing the $Psuc$ significantly increases the likelihood of finding a valid mapping for a given logic function. Determining the maximum yield requires the satisfactions of two following constraints.

$$minimize\Big(k_o, k_i \geq 1\Big) \quad (3)$$

$$maximize\Big(Psuc(k_i, k_o)\Big) \quad (4)$$

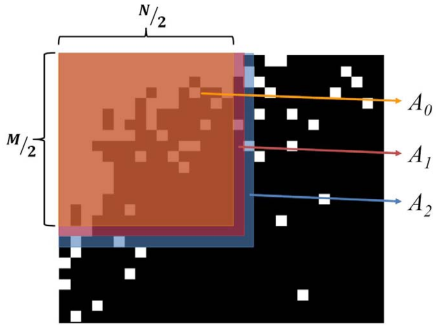

Fig. 3. Subareas used for calculating the regional densities.

B. Clustered Distribution

Conventional wisdom indicates that cluster is in the eye of the beholder, so a formal yield analysis is difficult due to changing defect rates of nano-crossbar regions. For this reason, we find an upper bound for yield for clustered defects.

First, we sort crossbars with uniform and clustered defects. We decreasingly arrange input lines left to right and output lines top to bottom according to the number of defects. Figure 2 (e)-(f) shows the sorted crossbars. After that, we divide crossbars into subregions starting from the top left corner and expanding the region one input and output line at a time. Figure 3 shows progress of this operation. Finally, we compare the defect densities of uniform and clustered distributions.

In order to find densities of regions, we define a base case A_0 as the number of defects in a $\frac{M}{2} \times \frac{N}{2}$ area and A_i's as the number of defects in $(\frac{M}{2}+i) \times (\frac{N}{2}+i)$ areas. We found this base case experimentally as a threshold eliminating the fluctuations of subregion densities. After that, following formula is used for finding region densities.

$$D_i = \begin{cases} \frac{A_0}{\frac{M}{2} \times \frac{N}{2}}, & i = 1 \\ \frac{(A_i - A_{i-1})}{\frac{M}{2}+\frac{N}{2}+1}, & i > 1 \end{cases} \quad (5)$$

We repeat this process by increasing the defect rate of uniform distribution until our condition of all regional densities of uniform distribution are equal or greater than those of clustered distribution. In finding this rate, we exploit (3) and (4) to determine maximum yield upper bound for clustered defects. In Figure 4, defect density comparisons are given with increasing P_d until our condition is met. After conducting extensive simulations with different IRs and optimal sizes, we determine that when defect rate of uniform distribution is 1.5 times greater than defect rate of clustered distribution.

V. EXPERIMENT RESULTS

We implement the algorithms and simulations using MATLAB with a sample size of 100 defective nano-crossbars for each case. All experiments run on a 3.30GHz Intel Core i7 CPU (only single core used) with 8GB memory.

First, we show the yield results of random benchmarks in Table I. Even though *IR* and optimal size values are same, results differ considerably for benchmarks complying the logic function constraints and the rest. These results confirm our claim that fully randomly generated benchmarks are unfit for a proper yield analysis.

TABLE II

SUCCESS RATES OF TUNALI AND YUAN'S ALGORITHMS WITH PROPOSED AND 1.5 TIMES LARGER AREA OVERHEAD COEFFICIENTS, $P_d = 20\%$

Benchmarks				**Tunali** [9]										**Yuan** [10]									
				Optimum		Proposed						$k_o, k_i = 1.5, 1.5$		Optimum		Proposed						$k_o, k_i = 1.5, 1.5$	
No	Name	Size	IR	Uni	Clu	Uni	k_o	k_i	Clu	k_o	k_i	Uni	Clu	Uni	Clu	Uni	k_o	k_i	Clu	k_o	k_i	Uni	Clu
1	xor5	16×10	50%	76%	16%	100%	1.4	1.4	100%	1.6	1.6	100%	100%	78%	12%	100%	1.4	1.4	100%	1.6	1.6	100%	100%
2	squar5	32×10	50%	12%	0%	100%	1.5	1	100%	1.8	1	100%	100%	14%	0%	100%	1.5	1	100%	1.8	1	100%	100%
3	bw	87×10	40%	60%	10%	100%	1.2	1	100%	1.2	1	100%	100%	60%	14%	100%	1.2	1	100%	1.2	1	100%	100%
4	ex5p	256×16	50%	0%	0%	100%	1.3	1	100%	1.5	1	100%	100%	0%	0%	100%	1.3	1	100%	1.5	1	100%	100%
5	apex4	438×18	46%	0%	0%	100%	1.2	1	100%	1.3	1	100%	100%	0%	0%	100%	1.2	1	100%	1.3	1	100%	100%
6	sao2	58×20	36%	0%	0%	100%	1.5	1	100%	1.5	1.4	100%	100%	0%	0%	100%	1.5	1	100%	1.5	1.4	100%	100%
7	table3	175×28	41%	0%	0%	100%	1.4	1	100%	1.4	1	100%	100%	0%	0%	100%	1.4	1	100%	1.4	1	100%	100%
8	t481	481×32	31%	0%	0%	100%	1.2	1	100%	1.2	1.1	100%	100%	0%	0%	100%	1.2	1	100%	1.2	1.1	100%	100%
9	table5	158×34	35%	0%	0%	94%	1.3	1.2	100%	1.6	1.5	100%	100%	0%	0%	100%	1.3	1.2	100%	1.6	1.5	100%	100%
10	duke2	87×44	20%	0%	32%	100%	1.3	1.2	100%	1.6	1.5	100%	100%	0%	50%	100%	1.3	1.2	100%	1.6	1.5	100%	100%
11	apex1	206×90	10%	0%	58%	100%	1.2	1	100%	1.4	1.3	100%	100%	0%	70%	100%	1.2	1	100%	1.4	1.3	100%	100%
12	apex3	280×108	8%	0%	40%	100%	1.2	1	100%	1.2	1.2	100%	100%	0%	86%	100%	1.2	1	100%	1.2	1.2	100%	100%

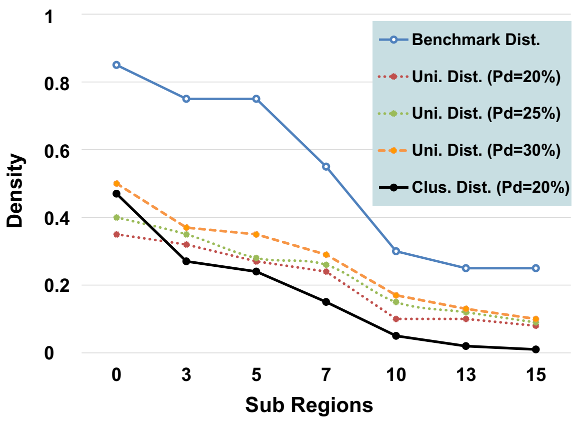

Fig. 4. Comparison of regional densities for uniform and clustered distributions considering different P_d values of uniform defects.

TABLE I

SUCCESS RATES OF TUNALI'S ALGORITHM WITH DIFFERENT YIELD VALUES REGARDING RANDOM BENCHMARKS; DEFECT RATE $P_d = 20\%$

Bench		$k_o, k_i = 1, 1$		$k_o, k_i = 1.1, 1.1$		$k_o, k_i = 1.2, 1$		$k_o, k_i = 1, 1.2$	
No	*IR*	Uni	Clu	Uni	Clu	Uni	Clu	Uni	Clu
1*	40%	0%	0%	0%	0%	0%	0%	0%	0%
2	40%	0%	0%	4%	0%	16%	0%	10%	0%
3*	35%	0%	0%	0%	0%	100%	8%	100%	2%
4	35%	4%	0%	100%	0%	100%	8%	100%	10%
5*	30%	100%	0%	100%	64%	100%	90%	100%	84%
6	30%	100%	0%	100%	64%	100%	75%	100%	76%
7*	25%	100%	6%	100%	98%	100%	98%	100%	98%
8	25%	100%	16%	100%	100%	100%	100%	100%	100%

* generated according to the logic function constraints in Section 4.

Second, we use industrial benchmarks. We utilize both Tunali's algorithm which has a low runtime [9] and Yuan's algorithm which is slower but has higher success rate [10] to evaluate our yield values found with equations (3) and (4). Results of conducted simulations are given in Table II. In addition to our yield values, we include $k_o, k_i = 1.5, 1.5$ area overheads which is a common practice in the literature. It is clear from the table that our area overhead results for uniform and clustered defect distributions always ensure a valid mapping considering both algorithms. Our findings, questioning the common practices in the literature, can be summarized as: 1) Every logic function requires different area overhead coefficients and they are not necessarily the same; 2) Our yield values ensure both algorithms to find a valid mapping; and As a general tendency, increasing the area coefficient belonging to the larger dimension of nano-crossbar enhance the probability of finding a valid mapping.

VI. CONCLUSION

We present yield analysis of nano-crossbar arrays covering defects with uniform and non-uniform distributions. We comment on rough yield estimations used in the literature by showing that yield is significantly related to 1) efficiency of the used mapping algorithms, 2) logic functions to be implemented, and 3) defect distributions.

REFERENCES

[1] M. Haselman and S. Hauck, "The future of integrated circuits: A survey of nanoelectronics," *Proceedings of the IEEE*, vol. 98, no. 1, pp. 11–38, 2010.

[2] D. Alexandrescu, M. Altun, L. Anghel, A. Bernasconi, V. Ciriani, L. Frontini, and M. Tahoori, "Logic synthesis and testing techniques for switching nano-crossbar arrays," *Microprocessors and Microsystems*, vol. 54, pp. 14–25, 2017.

[3] H. Yan, H. S. Choe, S. Nam, Y. Hu, S. Das, J. F. Klemic, J. C. Ellenbogen, and C. M. Lieber, "Programmable nanowire circuits for nanoprocessors," *Nature*, vol. 470, no. 7333, pp. 240–244, 2011.

[4] J. Yao, H. Yan, S. Das, J. F. Klemic, J. C. Ellenbogen, and C. M. Lieber, "Nanowire nanocomputer as a finite-state machine," *Proceedings of the National Academy of Sciences*, vol. 111, no. 7, pp. 2431–2435, 2014.

[5] O. Tunali and M. Altun, "A survey of fault tolerance algorithms for reconfigurable nano-crossbar arrays," *ACM Computing Surveys*, 2017.

[6] H. Naeimi and A. DeHon, "A greedy algorithm for tolerating defective crosspoints in nanopla design," in *Field-Programmable Technology, 2004. Proceedings. 2004 IEEE International Conference on.* IEEE, 2004, pp. 49–56.

[7] Y. Su and W. Rao, "Defect-tolerant logic mapping on nanoscale crossbar architectures and yield analysis," in *Defect and Fault Tolerance in VLSI Systems, 2009. DFT'09. 24th IEEE International Symposium on.* IEEE, 2009, pp. 322–330.

[8] Y. Yellambalase and M. Choi, "Cost-driven repair optimization of reconfigurable nanowire crossbar systems with clustered defects," *Journal of Systems Architecture*, vol. 54, no. 8, pp. 729–741, 2008.

[9] O. Tunali and M. Altun, "Permanent and transient fault tolerance for reconfigurable nano-crossbar arrays," *Transactions on Computer Aided Design*, vol. 99, 2016.

[10] B. Yuan, B. Li, T. Weise, and X. Yao, "A new memetic algorithm with fitness approximation for the defect-tolerant logic mapping in crossbar-based nanoarchitectures," *Evolutionary Computation, IEEE Transactions on*, vol. 18, no. 6, pp. 846–859, 2014.

[11] C. H. Stapper, "Simulation of spatial fault distributions for integrated circuit yield estimations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 8, no. 12, pp. 1314–1318, 1989.