Process Multi-Circuit Optimization

Arun Lokanathan
Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556
alokanat@cse.nd.edu

Jay Brockman
Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556
jbb@cse.nd.edu

Abstract: This paper describes the implementation of a concurrent methodology for integrated circuit optimization that spans the fabrication process design and circuit design disciplines. Results from this methodology show substantial performance gains as compared to a separate process/circuit optimization and a substantial time savings over an "all-at-once" combined optimization.

1. Introduction

For a given semiconductor technology (CMOS, bipolar, BiCMOS, GaAs) and a given circuit topology (a particular datapath or a memory architecture, for example), the performances of an integrated circuit are generally determined by two distinct "camps" of designers: the device and fabrication process designers, and the circuit designers. Device and process designers control the electrical characteristics of the devices that they fabricate, while circuit designers seek to develop optimal circuits with those devices. Circuit optimization and process optimization are logically treated as separate problems, where circuit designers assume a fixed set of device model parameters (Spice device models) and adjust circuit geometries to enhance circuit performances, and process engineers strive to fabricate devices that conform to the circuit designers' assumptions. Caught in the middle, however, is the question of how to target the device characteristics. This question is becoming increasingly important as we move to systems-on-a-chip, such as mixed logic and memory (e.g. [13]), where different parts of a chip may "prefer" higher or lower threshold voltages, oxide thicknesses, etc. Advanced or exotic processing techniques may sometimes be able to produce different devices for different circuits, but in general this is quite expensive and should be avoided.

The problem of optimizing devices for multiple circuits on a single die is challenging for several reasons. First of all, individual device model parameters cannot be treated as independent design variables as they are correlated through the fabrication process. Secondly, device parameters cannot be properly chosen if the circuit designs themselves are mistargetted; in other words, device characteristics should ideally be optimized for the best circuits that could be designed with those devices. This demands that circuit, device, and process design be considered concurrently.

In theory, concurrent process, device, and circuit optimization could be represented as a "flat" problem, with the fabrication process controls and circuit geometries as the design variables and circuit performances as the objective, that is solved "all-at-once" using process, device, and circuit simulators to estimate state values. This, however, is practically infeasible for several reasons: obviously, because of the size of the problem and complexity of the tools involved, and more subtly, because it ignores the natural and logical organizational partitioning between the circuit and process design domains.

This paper describes the implementation and application of Process Multi-Circuit Optimization (PMCO), a strategy that enables process designers and circuit designers to work concurrently and coordinatedly toward the common goal of optimizing system-level IC performance. Results of benchmarking the concurrent methodology against separately optimized circuits and devices, as well as against an "all-at-once" joint optimization approach are presented. As examples of different circuits with different device needs, an output buffer is optimized for speed and power and an SRAM cell is optimized for access time and read and write reliability. Process and device simulations for a twin-tub CMOS process are done with pdFab ([11]) and circuit simulations are done with a commercial version of Spice. To facilitate the optimization, an extensible object-oriented environment has been developed that integrates the various simulation and optimization tools. Results show substantial aggregate circuit performance gains over separate optimizations, and substantial time savings over the all-at-once optimization.

2. Background and Related Work

Consider the problem of optimizing Φ, the aggregate performance of an integrated circuit composed of subcircuits with performances $f_1, f_2 \ldots f_m$, as shown in Figure 1. The subcircuit performance vectors $f_1, f_2 \ldots f_m$ depend upon the respective subcircuit geometries $G_1, G_2 \ldots G_m$ and the set of device model parameters D. The set of fab process inputs P, determines the values of the device model parameters, D. The goal of process-circuit co-optimization is to determine values for the process inputs P and circuit geometries $G_1, G_2 \ldots G_m$, that optimize the system performance, Φ. The

1-58113-049-x-98/0006/$3.50

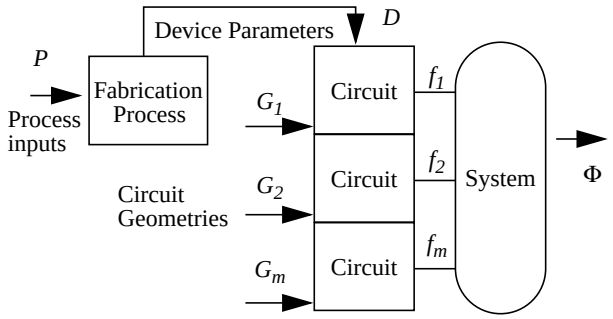

Figure 1. Variables in Integrated Circuit Optimization

system objective function must reflect the multiple performance requirements of the circuits that the IC system is composed of—common choices for Φ include linear combination of performances, sum of squares, or sum of exponentiated terms ([1]). We cast the IC optimization problem as a multi-objective optimization problem using a technique called goal programming ([6]).

In the past, work in the area of integrated circuit optimization has focussed predominantly on techniques for local design improvement in either the circuit design discipline or the process design discipline. In fact, circuit optimization is a relatively mature area; a survey of relevant literature can be found in [1]. Statistical circuit optimization that further accounts for process variations has also been studied extensively, and a number of formulations and techniques, based on traditional optimization approaches, have been proposed in the past to solve the problem ([2], [3], [4]). The examples in the literature almost exclusively demonstrate single-domain optimization. Circuit optimization examples adjust circuit geometries with a view to achieving acceptable circuit performances within some range of variation of device model parameters and/or operating conditions. Process optimization examples adjust process inputs to achieve acceptable device model parameter values. Previous work reported in [5] that does address optimization across the process and circuit design disciplines, uses a stochastic algorithm to optimize a process and single-circuit system in a sequential all-at-once manner. In contrast, the PMCO method is a concurrent deterministic method that may be used to optimize a fab process and a set of circuits in an independent yet coordinated manner.

Process Multi-Circuit Optimization (PMCO), the methodology of this paper, draws on ideas from the general area of Multidisciplinary Design Optimization (MDO) [7], and is based specifically on the Concurrent Subspace Optimization (CSSO) algorithm [8]. MDO algorithms start with a complex coupled system-level optimization problem, decompose the problem into smaller approximate subproblems that are solved independently (by domain-specific subspace design teams), and coordinate the solutions of the subproblems to obtain a system-level optimal solution.

3. Process Multi-Circuit Optimization

The PMCO method optimizes the overall system of Figure 1 by formulating a common system-level optimization problem, and then solving it in a concurrent manner. The IC optimization problem is cast using the goal programming formulation which attempts to simultaneously minimize (or maximize) multiple circuit performance functions below (or above) user-specifiable targets. The minimization of several functions $f_1, f_2 \ldots f_m$ is converted to the following scalar constrained nonlinear optimization problem:

$$
\begin{array}{ll}
\text{Minimize} & \gamma \\
\text{w.r.t.} & \gamma, P, G_1, G_2 \ldots G_m \\
\text{subject to} & f_1 - \gamma\alpha_1 \le \delta_1 \\
 & f_2 - \gamma\alpha_2 \le \delta_2 \\
 & \ldots \\
 & f_m - \gamma\alpha_m \le \delta_m
\end{array}
$$

The values $\delta_1, \delta_2 \ldots \delta_m$ are targets or goals that are set for the performances $f_1, f_2 \ldots f_m$ respectively. $\alpha_1, \alpha_2 \ldots \alpha_m$ are weights used to adjust the relative rigidity with which the targets need to be met. γ, a scalar, is introduced as an independent design variable as well as the objective function of the transformed optimization problem. The multiple objective functions of the original problem are then transformed into the constraints of the goal programming formulation. Assuming positive weights α_i, as γ is decreased, each of the performance functions $f_i(D(P), G_i)$, $i = 1 \ldots m$ need to be decreased in order to satisfy their respective constraints in the goal programming problem. This leads to all the functions f_i being simultaneously minimized, if such a point exists.

The overall flow of the PMCO method, first proposed in [9], is shown in Figure 2. The algorithm starts with the current set of values for the design variables $X^k = [P^k, G_1^k \ldots G_m^k]$ at the k^{th} outer iteration. It first performs what is termed as a "system analysis" which evaluates the following: (i) values of device parameters, D^k, and device parameter sensitivities with respect to process inputs, $\nabla_P D^k$, via process simulation, (ii) values of the performances of each circuit, f_i^k and their partial sensitivities with respect to geometries and device model parameters, namely $\nabla_{G_i} f_i^k$ and $\nabla_D f_i^k$. In other words, the system analysis computes process and circuit function values and gradients at the current design point.

The algorithm then launches several independent subspace optimization tasks, one in the process domain and one corresponding to each circuit. Within the process optimization, the process designers calculate device model parameters as a function of process inputs via process

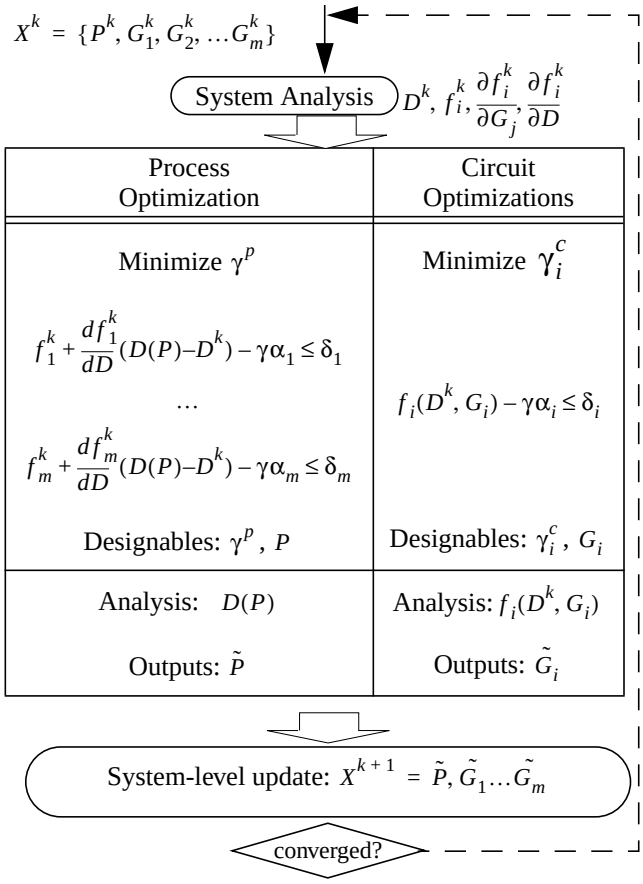

Figure 2. The Concurrent Process-Circuit Codesign Method

simulation. In order to approximate circuit performances, they use a linear mapping of device model parameters to circuit performances using the currently available gradient information, namely, the matrix $\nabla_D f_i^k$. Due to the linear $D \to f$ assumption, move limits must be imposed on the design variables of the process subspace optimization, namely the process inputs P. A user-selectable constant move limit (defaulting to 25% of the global range) is used in this work. In the circuit optimizations, circuit designers use the currently available device model parameter set, and adjust circuit geometries to optimize individual circuit performances. The process-optimal and circuit-optimal design vectors are concatenated to obtain the system design vector for the next iteration.

The key advantages of the PMCO method are (i) the subspace optimization tasks are of smaller dimensionality than the whole system-level problem, leading to robust convergence characteristics (in general, it is reasonable to expect optimization algorithms to converge better on smaller-dimensional problems) (ii) the process and circuit subspace optimization tasks may proceed concurrently, thereby decreasing design time, (iii) there is a certain degree of design freedom within the process and circuit optimization tasks—one may use different move limits, tolerance settings or even different optimization strategies in the different tasks. We will illustrate these benefits via a series of examples in the next section.

4. Results

A series of examples is used demonstrate the two main points that motivate the PMCO method: (i) in order to optimize an integrated circuit, we should adjust not only the circuit geometries but also the fabrication process control inputs, and (ii) in solving the combined process-circuit optimization problem, the PMCO method is more efficient and practical than an all-at-once approach. The first example optimizes the a CMOS buffer while a second optimizes an SRAM cell. A third example then optimizes a system composed of both the buffer and SRAM circuits.

The examples use a twin-tub CMOS process modeled by 75 processing steps in pdFab ([11]), which maps 120 process inputs to 26 Spice level 3 MOS transistor model parameters. Because only a few process variables generally impact device/circuit performances significantly, a simple sensitivity-based variable screening step was performed initially to identify the process inputs that had the most impact on the performances of the buffer and SRAM circuits ([14]).

In each example, the optimization problem is first solved by adjusting only circuit design variables and then solved by adjusting process inputs as well as circuit geometries. This brings out the benefit of combined process-circuit optimization over circuit-only optimization. To demonstrate the efficiency of the concurrent PMCO approach, the combined process-circuit optimization problem is solved using two methods: (i) an all-at-once method that treats the entire process-circuit system as a monolithic whole and "synchronously" optimizes the system over all the variables of the system (process and circuit variables), and (ii) the concurrent PMCO method.

The results indicate that while the choice of circuit geometries is responsible for large performance improvements, the performances of optimal-geometry circuits can be further improved significantly via an optimal choice of device characteristics (and hence process inputs). The third example also demonstrates the application of the concurrent methodology to multiple circuits as well as its capability to negotiate process trade-offs. Furthermore, the concurrent PMCO algorithm is found to be more efficient than the all-at-once approach in terms of elapsed time.

4.1 Optimization of CMOS Buffer

The buffer circuit is a 4-stage static CMOS inverter chain that may be used for driving off-chip loads or large on-chip capacitive loads such as clock networks—the designable geometries are the widths of the PMOS and NMOS transistors of inverter chain. The performances are delay and power dissipation. Both quantities are measured via a transient simulation in SmartSpice [12]. The delay is measured across the 50% Vdd points and the power is measured via a power meter along the lines of [10].

Consider the case for which the delay of the buffer needs to be minimized subject to a power constraint, as is the typical requirement for high-speed (as opposed to low-power) systems.

Minimize	$\text{delay}(P, G_1)$
subject to	$\text{power}(P, G_1) \le 2.5\text{m}$
wrt.	P, G_1

where P is the set of 5 significant process variables determined via variable screening and G_1 is the set of 8 transistor widths.

Variable	Initial	Circuit Only	Process Circuit
Ntubbox/Temp(C)	1100.0	1100.0	1113.0
gateox2/Temp (C)	900.0	900.0	760.4
gateox3/Temp (C)	880.0	880.0	740.4
padox4/Temp (C)	1030.0	1030.0	1050.0
passiv/Temp (C)	840.0	840.0	834.0
buffer/wn5 (μm)	3.0	22.5	20.3
buffer/wn6 (μm)	3.0	51.7	44.5
buffer/wn7 (μm)	3.0	139.8	114.8
buffer/wn8 (μm)	3.0	297.0	263.1
buffer/wp5 (μm)	4.0	33.2	31.8
buffer/wp6 (μm)	4.0	84.9	80.1
buffer/wp7 (μm)	4.0	189.7	153.4
buffer/wp8 (μm)	4.0	445.5	434.9
buffer/power (mW)	1.32	1.94	2.30
buffer/delay (ns)	18.0	0.97	0.66

Table 1: CMOS Buffer Design Points

Table 1 lists the values of process inputs, circuit geometries, and circuit performances at the key design points. The initial design point (initial process, 3μ NMOS, 4μ PMOS transistors) resulted in a delay of 18.0 ns and a power dissipation of 1.32 mW. The performances were first optimized by adjusting circuit geometries alone, keeping the process inputs constant, as is done in conventional circuit optimization. As shown in the "circuit-only" column of the table, circuit optimization reduced buffer delay from 18.0 ns to 0.97 ns (a 95% reduction). The optimization problem was then solved by adjusting both the process inputs as well as the circuit geometries—the optimal design after process-circuit optimization produced a delay of 0.66 ns (an additional 32% reduction from the circuit-only optimal point). The 2.5 mW power constraint is satisfied by all the design points—the power dissipation is traded off, however, to decrease delay from the initial to the optimal points.

In order to compare the PMCO method with an all-at-once method, the above problem was solved using both methods. Table 2 lists the times spent by the all-at-once and concurrent methods. The all-at-once column is straightforward— the total time is the sum of time spent in function calls, time spent in gradient calls and time spent in line searches by the optimizer. Remember that every time the all-at-once method makes a function call, it involves a process simulation followed by a circuit simulation. For the PMCO column, we have to add the system analysis time with the larger of the process optimization time and circuit optimization time, over all iterations of the algorithm. To clarify this point, we illustrate the flow of the PMCO algorithm using the timeline diagram of Figure 3—the algorithm proceeds by iteratively performing a system analysis followed by concurrent process and circuit optimization tasks. The function calls, gradient calls and line search calls in the PMCO algorithm are made to the respective subspace approximations that involve only one simulation tool (either process or circuit, not both, as in the all-at-once case). The net result is that the all-at-once approach takes 10.79 hours whereas the concurrent optimization takes 7.97 hours to arrive at the optimal process-circuit point.

(Time in hours over all iterations)	All-at-once	PMCO	
		Process	Circuit
System Analyses	-	1.2	
Function calls	1.21	0.90	0.39
Gradient calls	4.90	3.10	1.97
Line searches	4.68	0.81	2.94
Total Time	10.79	7.97	

Table 2: Timing Analysis

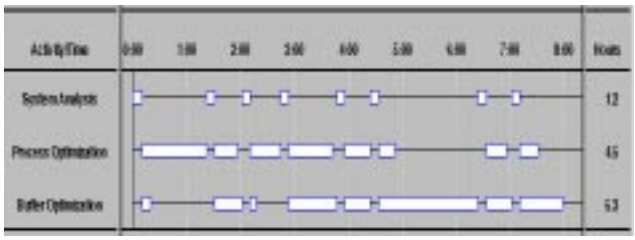

Figure 3. Time line for Concurrent Optimization

4.2 Optimization of SRAM Cell

This example optimizes the access time, read safety and write reliability of a standard 6-transistor SRAM cell (connected between two bit lines and a word line in the usual manner). Figure 4 shows the 2-dimensional space of bit line voltages divided into regions of safe read, write-0 and write-1. The quantity v_{sr0}, the minimum voltage required on a bit line to ensure a safe read-0 operation,

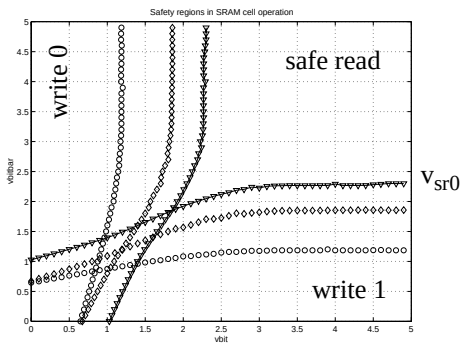

Figure 4. SRAM Cell Read Safety/Write Reliability

defines the boundary between the read and write regions. A higher v_{sr0} implies higher read safety but lower write reliability. We will require that v_{sr0} be made as low as possible but no lower than 1.5 V. This leads to the following optimization formulation:

Minimize	accesstime(P, G_2)
and	$v_{sr0}(P, G_2)$
subject to	$v_{sr0}(P, G_2) \geq 1.5$
wrt.	P, G_2

The performances at the initial, circuit-optimal and combined process-circuit optimal design points are shown in Table 3 (the input variables have been left out for brevity). The initial access time of 2.66 ns was reduced to 0.97 ns via circuit optimization. Process optimization accounted for a further drop in access time from 0.97 ns to 0.79 ns. The quantity v_{sr0} was reduced in both cases from 2.27 V to 1.50 V, which maximizes read safety subject to the write reliability constraint. Combined process-circuit optimization took 8.02 hours via the all-at-once approach as opposed to 6.14 hours via the concurrent method.

	Initial	Circuit Only	Process Circuit
sram/accesstime (ns)	2.66	0.97	0.79
sram/vsr0 (V)	2.27	1.50	1.50

Table 3: SRAM Cell Optimization Design Points

4.3 Process Multi-Circuit Optimization

This example illustrates the applicability of the concurrent methodology to multiple circuits as well as its ability to address competing performance concerns via process trade-offs. It is required to find a set of process inputs and circuit geometries that minimize buffer delay as well as SRAM cell accesstime and read safety, while satisfying constraints on buffer power dissipation and SRAM cell write reliability. This leads to the following optimization formulation:

Minimize	delay(P, G_1)
and	accesstime(P, G_2)
and	$v_{sr0}(P, G_2)$
subject to	$v_{sr0}(P, G_2) \geq 1.5$
and	power$(P, G_1) \leq 2.5$m
wrt.	P, G_1, G_2

Variable	Initial	Circuit-only	Process Circuit
buffer/power	1.32 mW	1.94 mW	2.3 mW
buffer/delay	18.0 ns	0.97 ns	0.69 ns
sram/accesstime	4.93 ns	1.59 ns	0.82 ns
sram/vsr0	2.05 V	2.10 V	1.51 V

Table 4: Combined Optimization Design Points

Table 4 shows the circuit performances at the initial design point, the circuit-optimal design point and the process-circuit optimal design point. Circuit-only optimization improves the performances of both circuits—the buffer delay is reduced to 18.0 ns from 0.97 ns, the SRAM access time is reduced from 4.93 ns to 1.59 ns. v_{sr0}, however, is slightly increased to 2.10 V, and is undesirable for read safety—circuit-only optimization maximize read safety at the given initial process point. Combined process-circuit optimization, on the other hand, arrives at a combination of process inputs and circuit geometries that produces an SRAM v_{sr0} of 1.51V, thereby maximizing read safety as desired. Furthermore, it improves the buffer delay to 0.69 ns (a 28% improvement over the circuit-only optimal point) and improves the access time of the SRAM to 0.82 ns (a 48% improvement over the circuit-only optimal point). Process-circuit optimization via the concurrent PMCO method thus produces a significantly better design point than merely circuit optimization. Again, this example was solved using both the concurrent method and the all-at-once method—the concurrent method took 6.61 hours compared with 17.1 hours of the all-at-once approach. The reason for the time savings of the PMCO method over the all-at-once approach is that the PMCO method solves several smaller problems independently—a trend that may be expected to get better as the number of circuits being optimized increases.

Finally, Table 5 summarizes the key design points, namely, the initial point, the point after circuit optimization, and points after process optimization targeting the buffer alone, the SRAM cell alone and then both simultaneously. Observe that the process point that optimizes buffer alone is detrimental to the SRAM cell access time; the process point that optimizes the SRAM alone penalizes buffer delay. The PMCO method produces a process point that is beneficial to the system as a whole, in that while it sacrifices SRAM access time somewhat, it produces the fastest buffer and SRAM cell while satisfying constraints on power dissipation and SRAM read safety and write reliability.

Design Point	Buffer		SRAM cell		
	Delay	Power	Access time	vsr0	
Initial	18.0 ns	1.32 mW	4.93 ns	2.05 V	Slow buffer, slow SRAM, Unsafe read
After circuit optimization	0.97 ns	1.94 mW	1.59 ns	2.10 V	Room for improvement
Buffer-optimal process	0.66 ns	2.30 mW	1.56 ns	1.50 V	Fast buffer, safe SRAM read, slow SRAM
SRAM-optimal process	0.70 ns	2.48 mW	0.79 ns	1.50 V	Fast SRAM, safe SRAM read slow buffer
Joint optimal process	0.69 ns	2.30 mW	0.82 ns	1.50 V	Fast buffer, fast SRAM, safe SRAM read

Table 5: Process Trade-offs

5. Conclusion

This paper has demonstrated that, in order to design optimal integrated circuits, it is important not only for circuit designers to adjust circuit geometries but also for fabrication process engineers to adjust process controls for suitable device characteristics and better overall system performance. The results presented in this paper indicate that significant performance improvements can be achieved by performing combined process multi-circuit optimization. There are two distinct ways in which combined process-circuit optimization may be performed—either the entire system may be optimized all-at-once or the concurrent process multi-circuit optimization (PMCO) method may be used. The PMCO method produces similar design points as the all-at-once approach in significantly less time. The concurrent PMCO method is useful in optimizing a fab process targeting multiple circuits that may embody competing performance concerns.

6. Acknowledgments

This work was supported in part by grants from Motorola Semiconductor Products Sector, NSF and NASA.

7. References

[1] R. K. Brayton, and R. Spence, "*Sensitivity and optimization*", Elsevier Scientific, Amsterdam, 1980.

[2] J. C. Zhang and M. A. Styblinski, "Yield and Variability Optimization of Integrated Circuits", Kluwer Academic Publishers, 1995.

[3] K. J. Antreich, H. E. Graeb, and C. U. Weiser, "Circuit Analysis and Optimization Driven by Worst-Case Distances", *IEEE Transactions on CAD*, vol. 13, no. 1, January 1994.

[4] P. Feldmann and S.W. Director. Accurate and efficient evaluation of circuit yield and yield gradients. In *IEEE International Conference on Computer-Aided Design, ICCAD-90*, pages 120-123, November 1990.

[5] M. A. Styblinski and L. J. Opalski, "Algorithms and Software Tools for IC Yield Optimization Based on Fundamental Fabrication Parameters", *IEEE Transactions on CAD*, vol. CAD-5, no. 1, pp. 79-89, January 1986.

[6] R. K. Brayton, S. W. Director, G. D. Hachtel, L. Vidigal, "A New Algorithm for Statistical Circuit Design Based on Quasi-Newton Methods and Function Splitting", *IEEE Transactions on Circuits and Systems*, vol. CAS-26, pp. 784-794, 1979.

[7] R. J. Balling, and J. Sobieszczanski-Sobieski, "Optimization of Coupled Systems: A Critical Overview of Approaches", In *Proceedings of The AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Panama City, FL, September 7-9, 1994.

[8] J. E. Renaud, "An Optimization Strategy for Multidisciplinary Systems Design", *Proceedings of the 9th International Conference on Engineering Design*, August 1993.

[9] Arun N. Lokanathan, Jay B. Brockman and John E. Renaud, "A Methodology for Concurrent Fabrication Process/Cell Library Optimization", In *Proceedings of the 33rd IEEE/ACM Design Automation Conference*, June 1996.

[10] Sung Mo Kang, "Accurate Simulation of Power Dissipation in VLSI Circuits", *IEEE Journal of Solid-State Circuits*, vol. SC-21, no. 5, October 1986.

[11] "*Pdfab User's Reference Manual*", PDF Solutions, San Jose, CA 1996.

[12] "SmartSpice/UTMOST III Users Manual", Silvaco International, Santa Clara, CA, 1995.

[13] P. M. Kogge, T. Sunaga, E. Retter, *et al*, "Combined DRAM and Logic Chip for Massively Parallel Applications," *16th IEEE Conference on Advanced Research in VLSI*, Raleigh, NC, IEEE Computer Society Press #PR07047, March 1995, pp. 4-16

[14] Arun N. Lokanathan, "Concurrent Process Multi-Circuit Optimization", Ph.D. thesis, University of Notre Dame, Notre Dame, IN, 1998.