Global Optimization Approach to Transistor Sizing for High Performance CMOS VLSI circuits[1]

Sharad Mehrotra, Paul Franzon[2], Wentai Liu[3]

NCSU-VLSI 93-10

November 1993

Picosecond Digital Electronics Laboratory

Department of Electrical and Computer Engineering

North Carolina State University

Raleigh NC 27695-7911

Abstract

A stochastic global optimization approach is presented for skew minimization in CMOS VLSI circuits. This is a direct search strategy for the best design among feasible ones, with the designer determining when the search is stopped. Through examples, we show the power of this technique in quickly obtaining very good designs, even for constrained problems.

---

[1]This report has been submitted to *31st ACM/IEEE Design Automation Conference,*, October 1993

[2]partially supported by MIP-901704 and an NSF National Young Investigator award

[3]partially supported by MIP-9212346

1 Introduction

In designing high performance CMOS circuits, it is often necessary to properly size the various transistors in a skeletal cell, in order to meet performance requirements. For example, transistor sizing has been extensively used for delay and power optimization of digital CMOS circuits. [3], [8]. In this paper, we describe a sizing program based on stochastic optimization of a model function. This is a direct search method which is more accurate than traditional nonlinear optimization programs, and is considerably faster than simulated annealing.

There are two basic approaches for transistor sizing that have been explored by various researchers. The first approach involves developing a simplified model of signal delay through a CMOS gate, either analytically [3], or by macromodels based on simulations [8]. Then, the delay of the entire cell is computed using this model. The transistor sizing problem is then formulated as an optimization problem with the objective of minizing the delay, as predicted by the model. Some optimization problems have a special form to guarantee efficiency and accuracy of the optimization process, e.g. the posynomial objective function, used in [3], [5]. Otherwise, a nonlinear optimization tool is required.

The second approach involves coupling a circuit simulator to a nonlinear optimization tool (or tool set). For example, Ochotta et. al. [13] employ an augmented asymptotic waveform evaluation technique to evaluate the behavior of each circuit visited by a simulated annealing program. In Delight.Spice [12], a set of nonlinear optimization programs are integrated with the SPICE circuit simulator.

The first, 'equation based', approach, though effective for delay optimization, is difficult to generalize to other problems, such as delay skew optimization, as is required for wave-pipelined circuits. Skew, in general, cannot be estimated using a single process corner or data vector. Maintaining accuracy in estimating the spread in delay over input data vector and process variations using a simple model is very difficult. The only solution is to evaluate each sizing scheme through detailed circuit simulations, to account for data and process variations. This makes objective computation a very expensive task, even for very small sized circuits.

The second, 'simulation based' approach has the combined computational burden of running a non-linear optimization program and running at least one full circuit simulation each time the objective function needs to be evaluated. There are some additional hindrances in employing a conventional non-linear optimization program for this task:

1. Gradient information is very difficult to obtain. Though there are numerical optimization techniques which do not require explicit gradient information, these techniques tend to be slow. Also, they try to evaluate the gradient through perturbation. This implies a further increase in the number of objective evaluations.

2. Transistor sizes can only be varied in certain quanta. Most numerical optimization techniques operate on a continuous parameter range. Hence the final solution might be an infeasible sizing scheme. Moving the solution to the closest feasible size may lead to a sub-optimal solution.

3. Constraints in the optimization problem, e.g. area and power, further complicate the optimization task.

4. The user has no direct control over the optimizer, i.e., the optimization task is not interactive, and it is difficult for the engineer to use his or her judgement in guiding the optimizer.

5. The optimization routines look for strict local minima. Usually, the designer is interested only in obtaining a rough approximation to a globally optimal solution. To achieve the global minimum, the optimizer has to be run from multiple, random, initial solutions. Even then, there are not even theoretical guarantees of achieving a globally optimal solution, except for some very restricted problems.

One way of avoiding gradient evaluation and restricting the solutions to feasible sizings is by employing simulated annealing [6] for optimization. However, simulated annealing programs tend to be prohibitively slow, specially when each objective evaluation is so expensive.

In view of these difficulties, we present a new approach to transistor sizing in small, high performance circuit blocks. This approach is based on stochastic modeling of the circuit responses of interest. It is a direct search for the best design among feasible ones. No gradient computations are required. The designer has direct control over the number of simulations conducted, and the search process can be stopped any time the designer is satisfied with the best solution produced thus far. The stochastic model helps in identifying the most promising design based on the existing information about the problem. Only the most promising designs are simulated. Hence simulations are organized naturally and efficiently. The method is capable of identifying the global minimum much faster than exhaustive search.

Recently, global optimization methods have been proposed for optimization of continuous functions based on stochastic modeling [17], [10]. Since this approach is relatively unknown in the circuit optimization domain, we will briefly review its important features in the next section. In section III we will propose an optimization algorithm consistent with the stochastic modeling philosophy. In section IV we present results obtained on two example circuits. Finally, section V is devoted to a discussion of the scope and limitations of this method.

2 Optimization by stochastic modeling of objective function

Consider the following unconstrained optimization problem:

$$\min_x \quad \zeta(x) \quad x \in A \subset R^d \tag{1}$$

where x is a d dimensional vector, A is a finite subset of R^d. $\zeta(x)$ is the objective function whose value at any $x \in A$ can be determined only through an expensive simulation. Besides this, there is very little information about the objective function. Suppose the objective function is perceived to be continuous and "smooth", but not unimodal. In such a situation, it is reasonable to approximate $\zeta(x)$ by a simpler function and to perform an optimization on this simplified function. However, a global polynomial approximation is inappropriate; information is lost in fitting the model to data, unless the degree of the polynomial is as large as the data set. A better approach to function approximation is needed.

Recently, stochastic models have been proposed to capture complex objective functions [14]. With this approach, the value of the unknown function at each point in A is assumed to be a random variable. Then, the unknown function itself is a *sample path of a stochastic function.* In the general case, a stochastic function $\phi(x)$ is defined by a family of multidimensional probability distributions $F_{x_1,\ldots,x_m}(y_1,\ldots,y_m) = P(\phi(x_i) < y_i, i = 1,\ldots,m)$. For example, if this distribution is joint Gaussian, then the stochastic function is described by the *a priori* average function $\mu(x)$ and covariance $\sigma(x_i, x_j)$. If k values of the stochastic function are known, e.g. $\phi(x_i) = \zeta(x_i), i = 1,\ldots,k$ then the *conditional distribution* of $\phi(x)$ at any x is normal with the mean value

$$\begin{aligned} m_k(x \mid \phi(x_i) &= \zeta(x_i), i = 1,\ldots,k) = \mu(x) + \\ &(\sigma(x,x_1),\ldots,\sigma(x,x_k))R_k^{-1}(\zeta(x_1)-\mu(x_1),\ldots,\zeta(x_k)-\mu(x_k))^T \end{aligned} \tag{2}$$

and variance

$$\begin{aligned} s_k^2(x \mid \phi(x_i) &= \zeta(x_i), i = 1,\ldots,k) = \sigma(x,x) - \\ &(\sigma(x,x_1),\ldots,\sigma(x,x_k))R_k^{-1}(\sigma(x,x_1),\ldots,\sigma(x,x_k))^T, \end{aligned} \tag{3}$$

where R_k^{-1} is the inverse of the $k \times k$ covariance matrix of the random process at the locations where the function values are known.

Using a stochastic model, and a set of "measurements" on the objective function $\zeta(x)$, prediction of the value of $\zeta(x)$ at untried points can be made using the conditional distribution of $\phi(x)$. This prediction is only probabilistic, i.e., at each x, there is a probability distribution associated with the possible values for $\zeta(x)$, specified by the mean and variance given above. This allows a search strategy for points of small $\zeta(x)$ to be formulated, based on the conditional distribution. It seems more likely to find a point with small function value where $m_k(x \mid .)$ is small. However, large values of $s_k^2(x \mid .)$ indicate regions of great uncertainity, i.e. regions where function values can differ greatly from the conditional mean. Hence a rational choice has to be discriminate between points of small mean but large variance or points of small variance but somewhat larger mean. We shall briefly review some proposed algorithms that fit the stochastic modeling paradigm. For a good discussion on stochastic modeling of unknown functions, see reference [14]. Global optimization methods are further described by Mockus in [10].

2.1 Algorithms based on Statistical Modeling

Several algorithms for optimization using a stochastic model function have been investigated. We summarize some interesting approaches and finish with the P-Algorithm which forms the basis of our optimization procedure. These approaches essentially differ in the stochastic model chosen and the method used for minimizing the model function.

In Groch et. al.[4], the model in the multidimensional case is not really a stochastic function. Instead, the conditional mean and variance of the one dimensional Weiner process are generalized. The choice of the next point of evaluation is made by minimizing:

$$r_i^{k+1}(x) \;=\; m_i^k(x) - c\sigma_i^k(x)\ ,\ \ i = 1, \ldots, n\ , \tag{4}$$

where the experimental region A is divided into N disjoint simplicial subregions and c is a constant, determining the weight given to variance with respect to the mean. The addition of each new point causes the experimental region to be further sub-divided. The attractive feature of this method is that it is simple; r can be minimized analytically. Only an approximation to the global optimum can be located by minimizing r. A local optimization has to be then performed.

In Adachi et. al.[7], the model function is a stationary stochastic process. The conditional mean is an interpolating function. This interpolating function is exact at the data points and its derivatives are easy to compute. The variance of approximating the true response by this function is also easy to compute. The optimal point is chosen by minimizing the interpolating function starting from the smallest data point found thus far, subject to a constraint on the coefficient of variation. The coefficient of variation is the ratio of the mean and variance of the conditional distribution. The procedure is iterative, each iteration leading to (hopefully) a different local optimum of the objective function, so that, eventually, all local optima can be located. The auxiliary computations are expensive, namely the use of a nonlinear optimizer at every iteration to find the minimum of the interpolating function.

Bernardo et. al. [1], employ a stochastic model function to perform yield optimization of electronic circuits. Their approach relies heavily on designer intervention to identify significant parameters through parameter effect plots and also to identify promising sub-regions in the design space.

The P-algorithm was developed and characterized in [19], [20]. It is an iterative procedure. At each iteration, a new observation point x_{k+1} is chosen that has the highest probability of $\phi(x_{k+1})$ being smaller than y_{ok} which is some chosen value smaller than the mean value of ϕ at each point in A. i.e.,

$$x_{k+1} = \text{Arg max}_{x \in A} P_x(y_{ok}) \tag{5}$$

is chosen as the next observation point where, y_{ok} is some value less than $min_{x \in A} m_k(x \mid x_i, \zeta(x_i), i = 1, \ldots, k)$, and

$$P_x(y_{ok}) = \text{Probability}(\phi(x) \leq y_{ok}) \, . \tag{6}$$

Based on rather intuitive axioms, it is shown in [17] that $\phi(x)$ can be assumed to be a Gaussian random variable whose conditional mean $m_k(x)$ and variance $s_k(x)$ is given as:

$$m_k(x \mid x_i, \zeta(x_i), i = 1, \ldots, k) = \sum_{i=1}^{k} w_i^k \zeta(x_i) \tag{7}$$

$$s_k^2(x \mid x_i, \zeta(x_i), i = 1, \ldots, k) = \gamma_k \sum_{i=1}^{k} (\sigma(x, x) - \sigma(x, x_i)) w_i^k \tag{8}$$

where w_i^k are weights chosen such that $\sum_{i=1}^{k} w_i^k = 1$ and $m_k(x \mid x_i, \zeta(x_i), i = 1, \ldots, k) = \zeta(x_i)$ at the k observed points, i.e. the mean value interpolates the known responses. In [20] it is proved that a sequence of points thus generated, converges to the global optimum of $\zeta(x)$.

The P-algorithm is a general formulation of a strategy to maximize the information gained by each function evaluation, and is quite easy to implement. However, in the implementation of the P-algorithm [17], several decisions have to be made that affect the speed and the accuracy of the method, namely the following:

- The appropriate form of the weighting functions w_i has to be chosen.

- The appropriate form of the covariance $\sigma(x, y)$ has to be chosen. It has been suggested that σ should be such that $(\sigma(x, x) - \sigma(x, z)) = \|x - z\|$ where $\|x - z\|$ is the Euclidean norm.

- An appropriate search method for finding x_{k+1} must be devised. This could be another multimodal optimization problem.

- An appropriate value of y_{ok} has to be chosen. It has been shown in [20] that too small a value of y_{ok} leads to the points of greatest uncertainity. If y_{ok} greater than or equal to $min_{x \in A} m_k(x \mid x_i, \zeta(x_i), i = 1, \ldots, k)$, then the next point chosen will be those x values that attain this minimum.

  In the next section, we detail our implementation of the P-algorithm. The process is iterative, hence the user can stop the iterations any time the best existing solution is satisfactory. The number of simulations to be run are directly controlled by the user. The algorithm only identifies the most promising points for simulation.

3 Implementation

The P-Algorithm is a formalization of an intuitive search strategy, and in fact, provides a framework for devising global optimization algorithms. We now describe our particular implementation of the P-Algorithm.

1. Choose k points x_i, $i = 1, \ldots, k$ uniformly from A using Latin Hypercube Sampling [9] and compute $\zeta(x_i)$ by simulation. Start iteration $l = 1$.

2. Using the BLUP and MSE expression in [15] (see Appendix 1, equations 17 and 19), find the mean $m(x_j)$ and variance $s(x_j)$ at $N \gg k$ uniformly distributed points in A.

3. Find the smallest value of $m(x_j)$, i.e.

$$m_l(x) \quad = \quad \min_{j \in 1 \ldots N} \; m(x_j) \; . \tag{9}$$

   Let $y_{ok}^l = m_l(x) - \epsilon_l$. At each x_j find the probability $P_{x_j}(y_{ok}^l)$.

4. Choose m_l points with largest probability $P_x(y^l_{ok})$ from the N points.

5. Compute $\zeta(x)$ at the n_l points found above. If $\min_{j\in 1\ldots n_l}\zeta(x_j)$ is satisfactory, then stop, else continue

6. $k = k + n_l$. If $k > K_{max}$, then stop, else $l = l + 1$, go to step 2.

This algorithm is parameterized by the constants k, n_l, ϵ_l and K_max. These parameters have to be adapted to the specific problem or left to the designer's judgement. For example $k = 10 * d$, $n_l = 2d$, where d is the dimensionality of the design space A, were found to be good values for problems below. In this way the designer directly controls the number of simulations to be run. The choice of N search points can be suitably biased by the designer's judgement, and can account for constraints on the design space, e.g. if the design region is a polytope constrained by linear inequalities on the design variables. Remember that only the mean and variance has to be estimated at the N points, which is quite inexpensive.

This leads us to another very important question, namely, handling constraints on the design. If the constraints are linear, they can be handled very naturally by screening each of the N points for violation. When the constraints are implicit and can be checked for only after simulation, e.g. a maximum delay or power restriction when doing a skew optimization, the procedure needs to be modified. There are two immediate possibilities. If the constraint can be evaluated through the same simulation, then another stochastic model can be used to model the constraint. A certain tradeoff has to be established between evaluation of this secondary model and the actual objective. If a penalty method is used to incorporate the true objective and the constraint in a single objective function, then the optimization task is a little more difficult, as the overall model might have to account for disparate variations. Alternatively, the constraint can be modeled piece-wise linearly by performing a separate experiment [16]. Then the optimization task is considerably simpler. The choice of method depends on the severity of the constraints. For a very tight constraint, it makes sense to pay close attention to the constraining function and use the first of the above approaches. For a loose constraint, the second approach would be more suitable. In the first example given below, the first approach was adopted for meeting a maximum delay requirement in addition to optimizing the delay skew.

In the next section, the algorithm described above is executed on two examples that have to be optimized for maximum delay variation. In the first example, the optimization is done with a maximum delay constraint. The second is an essentially unconstrained optimization problem.

4 Optimization examples

4.1 Delay Controlled Elements for Wave-pipelined circuits

The design of wave-pipelined circuits involves very careful control of the delay of each path in the combinational blocks. Techniques have been proposed by De Micheli et. al. [18] for balancing the path delays by inserting active delay elements. For ECL technology, they have shown how the delay of each gate can be accurately controlled through the tail current. For CMOS gates, however, the delay is data dependent. For the CMOS NAND gate, for example, the rising delay is substantially smaller when both inputs switch from 1 to 0, as opposed to one input being fixed at 1 and the other switching from 1 to 0. One way of avoiding this data dependence is to use the cross coupled biased-CMOS NAND gate shown in figure 1 [2]. This gate, however, consumes substantial static power when the output is low. Another gate structure suitable for wave pipe-lining is shown in figure 2. Here the transistor M3 is used to add extra resistance to the pull-up chain to reduce the effect of the simultaneous switching of both inputs. It also has the deleterious effect of slowing down the circuit. Hence a proper balance has to be struck between the maximum delay through the gate, as well as the data-dependent spread [11]. The delay spread has to be minimized over process variations also. Of course, the easiest parameters to control in this optimization are the transistor sizes. Hence the goal of the optimization is to obtain a suitable sizing scheme such that the delay spread through each circuit block is minimized, subject to a constraint on the maximum delay through the circuit.

The optimization problem is formalized as follows:

$$\begin{aligned} \text{Find } x^* &= \text{Arg } \min_{x \in A} \max_V \delta^*(x) && (10) \\ \text{subject to} \quad & \max_V \text{delay(x)} \leq D_{max} \,. && (11) \end{aligned}$$

Here V denotes the nominal and the four process corner MOSFET models. A is the hypercube formed by restricting the widths of M1-M3 between 3.6μm and 10μm, and V_{bias} between 0.0 and 2.0 V. The widths of N1 and N2 are constrained to be one-half the width of M1 and M2 respectively. x is an arbitrary vector of feasible transistor widths and bias voltage. Note that the minimum allowed feature size is 0.6μm and hence the widths of N1 and N2 were restricted to vary in quanta of 0.6μm only. The skew $\delta^*(x)$ is defined as the variation in delay through the circuit shown in figure 2 over the six possible input transitions (see figure 3), and the $delay(x)$ is the largest delay over these input transitions. D_{max} is the maximum delay constraint.

The models for delay and skew were initially established by simulating $k = 100$ different sizing schemes, selected randomly using Latin Hypercube Sampling [9]. The first row of table 1 shows the sizing scheme with the best skew value, satisfying the delay constraint, among these 100 points. This sizing scheme is not feasible. The second row shows the nearest feasible point to this sizing and the delay and skew value for that circuit.

Since the number of possible sizings is small, all the feasible alternatives (216 distinct sizing schemes) were evaluated with five values of bias voltage ranging from 0 to 2.0V using equations 17 and 19. This constitutes an exhaustive search of the design space using the models. Since the smallest possible value for the skew is 0, y_{ok} was chosen to be zero. 20 feasible sizes and bias voltages with the largest probability that satisfied the delay constraint $D_{max} \leq 1ns$ were chosen for resimulation. Table 1 shows the results for these sets of simulations. The best sizing in the second set was considerably better than the results of the first 100 samples and was considered quite suitable for design and hence no further simulations were performed. The total time taken for simulation was 540 cpu seconds on a DECstation 5000 while the overhead of model building and searching was less than 1 cpu second.

This example illustrates how the optimization procedure is employed. The search space is pruned by the designer's judgement and a good solution is found with very few simulations. In the next example, the methodology is further expanded to include continuous variables with a very different objective formulation.

4.2 CMOS Clock Driver Circuit

The second example we present here is the skew optimization of a single to differential input clock driver circuit shown in figure 4. It is desired to obtain a signal and it's complement from this circuit's outputs such that there is minimum skew between the two signals, i.e. to minimize $\delta^* = \max(\delta_h, \delta_l)$, as shown in figure 5. Again, this skew has to be minimized over the process variations. This optimization has to be done using a suitable transistor sizing scheme. The absolute delay through this circuit is not a concern, nor is the power consumption as this is a custom circuit block used only sparingly in the design. Hence the optimization is essentially unconstrained. To test the scope of the technique, temperature and power supply variations were also considered. The model was built over the space of transistor sizes, process, temperature and power supply variations. As in the previous example, process variations were considered by simulating each sizing scheme over the 4 process corners and the nominal process. The problem is formalized as follows:

$$\text{Find } x^* \quad = \quad \text{Arg } \min_{x \in A} \max_V \delta^*(x) \tag{12}$$

Here, A is the hypercube formed by restricting the widths P1-P6 between 3.6μm to 12μm, the temperature variation between 25-75° and V_{dd} between 4.75-5.25 V. As before the widths of N1-N6 are constrained to be one-half the widths of P1-P6 respectively. V represents the process variations considered. The optimization problem is different here as the worst delay skew over all *external noise* factors, i.e., temperature and power supply variations, has to be minimized over the *internal noise* factors, i.e. the process variations. For this problem, the sizing provided by the resident circuit design expert had a worst case skew of 290ps (row 1 of table 2). For optimization purposes, the skew model was built using $k = 100$ simulations, selected randomly using Latin Hypercube Sampling again. First, the effect of temperature and supply variation was factored out. To do this, 1000 random points were sampled in the 6-dimensional space of the transistor widths (A_w). At each of these 1000 points, the *model* in equations(17) and (19) (see Appendix) was evaluated for 9 different combinations of the supply voltage and temperature variations. The largest value of the probability $P(y_{ok})$ (equation 6) over these 9 combinations was found for each of the 1000 points. This value was used to estimate the likelihood of a sizing being the best. i.e. :

$$x_w^* \quad = \quad \text{Arg } \min_{x_w \in A_w} \max_E P_{x_w}(y_{ok}) \tag{13}$$

was the target for further simulation. Here x_w is the transistor width vector, A_w is the space of permissible transistor widths and E represents the temperature and voltage variations. Again y_{ok} was chosen to be 0.0 which is the minimum possible value of the skew. From the 1000 sizing schemes evaluated, the 10 sizing schemes (instead of only one as suggested by equation 13) with the largest probability were chosen for further simulation. These schemes were verified using the 5 process parameters and the 4 corners of power supply and temperature fluctuations. The smallest worst-case skew among these sizings was only 110 ps, a significant improvement over the expert's design (row 2 of table 2). The total simulation time was 640 cpu seconds on a DECstation 5000 and the overhead of model building and optimization was less than 10 cpu seconds.

These examples illustrate the power of this approach in evaluating transistor sizing schemes efficiently for different optimization problems. The stochastic model is able to capture the relationship of any performance measure to the transistor sizing. The model accuracy is reflected in the few subsequent evaluations required to arrive at very good designs.

5 Discussion

When optimizing high performance combinational sub blocks, there is some little *a priori* knowledge about the objective function, e.g., the difference between the maximum and minimum delay will always be greater than zero and less than some upper bound on the delay through the block. Also, the objective function is expected to behave fairly smoothly. Besides this, little can be said. Gradients are very hard to obtain as analytical modeling is difficult. Each objective function evaluation is going to be expensive, if the evaluation is made through a full circuit simulation, especially for reasonably large blocks. The number of input parameters is usually fairly small, in the 5-20 range, corresponding to the number of transistors that can be sized independently. There might exist some correlation between the input parameters. The algorithms based on stochastic modeling fit well in this scenario. The main idea behind these algorithms is to maximize the chances of improving the objective function after each evaluation. Almost any kind of a priori information can be used to speed up the search for optimal sizing. For example, it's quite easy to restrict the investigation performed by this algorithm to the part of the design space that is perceived by the designer to be most promising. The examples given in the previous section show how the method is capable of optimizing complex sizing objectives, based only on simulation results. Another extremely attractive feature of this algorithm is its flexibility. Since each objective evaluation is guaranteed to improve the best solution found thus far, the designer can stop the investigation at any time when the attained solution is deemed to be acceptable. However, this can also be perceived as a draw back of the algorithm since there is no implicit convergence criterion, i.e. location of a stationary point. The only way to perceive convergence is if successive iterations return the same solution. Another limitation of the method is that the cost of model estimation grows rapidly with the number of data points n. Each invocation of the model equations (17 and 19) requires the inversion of a $n \times n$ matrix. If, however, many model evaluations are made on the same data set (as in step 2 of the algorithm), then the matrix inversion needs to be done only once. This method should work very well for problems of dimensionality up to 25-30. Beyond that, the simple random sampling of step 2 is not sufficient to guarantee close to optimal solutions.

6 Conclusion and Future Work

We have demonstrated how an algorithm based on stochastic modeling is capable of solving some very difficult transistor sizing problems. This algorithm is a general formulation of a search strategy for the optimal sizing scheme. The only information required by the algorithm is the value of the objective function which can be calculated through a circuit simulation. No gradient information is required. The main thrust of the search is to maximize the probability of improving the best known solution with each calculation of the true objective function. This is consistent with the aim of obtaining good sizing schemes, evaluated through circuit simulations, while minimizing simulation time. An improved solution is found at each step of the algorithm and so the search can be stopped any time the best solution found thus far is considered satisfactory. The examples illustrate that very good solutions can be quickly found using this approach.

Further work is needed on the program interfaces. Process variations were considered only using the process corner models. If more detailed models of process variations are given, the procedure needs to be modified. For example, if the process variations can be specified by a normal distribution, then the stochastic model can be built to include the process parameters as variables also. Further investigation is required to do a worst-case analysis in this scenario.

7 Acknowledgements

The authors thank Tom Gray of IBM and Toby Schaffer of NCSU for help with the design examples.

References

[1] M. C. Bernardo, R. Buck, L. Liu, W. A. Nazaret, J. Sacks, and W. J. Welch. Integrated circuit design optimization using a sequential strategy. *IEEE Transactions on Computer Aided Design*, CAD-11:pp. 353–360, March 1992.

[2] D. Fan, C. T. Gray, W. Farlow, T. Hughes, W. Liu, and R. Cavin. A CMOS parallel adder using wave pipelining. In *Proc. of MIT/Brown Conference on Advanced Research in VLSI and Parallel Systems*, pages pp. 147–164, 1992.

[3] J. P. Fishburn and A. E. Dunlop. TILOS: a posynomial programming approach to transistor sizing. In *Proc. Int. Conf. Computer Aided Design*, pages pp. 326–328, Nov. 1985.

[4] A. Groch, S. W. Director, and L. M. Vidigal. A new global optimization mehtod for electronic circuit design. *IEEE Transactions on Circuits and Systems*, CAS-32:160–169, 1985.

[5] B. Hoppe, G. Neuendorf, D. Schmitt-Landsiedel, and W. Specks. Optimization of high speed CMOS logic circuits with analytical models for signal delay, chip area, and dynamic power dissipation. *IEEE Transaction on Computer-Aided Design*, CAD-9(3):pp. 236–247, 1990.

[6] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):pp. 671–680, 1983.

[7] Jin-Qin Lu and Takehiko Adachi. A parameter optimization method for electronic circuit design using stochastic model function. *Electronics and Communications in Japan*, 75(4):13–25, 1992.

[8] Mark D. Matson and Lance G. Glasser. Macromodeling and optimization of digital MOS VLSI circuits. *IEEE Transactions on Computer Aided Design*, CAD-5(4):pp. 659–679, Oct 1986.

[9] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, May 1979.

[10] J. B. Mockus. *Bayesian approach to global optimization: Theory and applications.* Kluwer Academic Publishers, 1989.

[11] W. Van Noije, C. T. Gray, W. Liu, T. Hughes, and R. Cavin. CMOS sampler with 1GBits/s bandwidth with 25ps resolution. In *Proceedings of the Custom Integrated Circuits Conference*, pages 27.5.1–27.5.4, 1993.

[12] W. Nye, D. Riley, A. Sangiovanni-Vincentelli, and A. Tits. DELIGHT.SPICE: An Optimization-Ba sed System for the Design of Integrated Circuits. *IEEE Trans. on Computer-Aided Design*, CAD-7(4):pp. 501–519, April 1988.

[13] Emil S. Ochotta, Rob A. Rutenbar, and L. Richard Carley. Equation-free synthesis of high-performance analog circuits. In *Proc. of MIT/Brown Conference on Advanced Research in VLSI and Parallel Systems*, pages 129–143, 1992.

[14] A. O'Hagan. Curve fitting and optimal design for prediction. *Journal of the Royal Statisitcal Society*, 40(1):pp. 1–42, 1978.

[15] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):pp. 409–435, 1989.

[16] S. Simovich, S. Mehrotra, P. Franzon, and M. Steer. Delay and reflection noise macromodeling for signal integrity management of PCBs and MCMs. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, to appear in, 1993.

[17] Aimo Torn and Antanas Zilinskas. *Global Optimization*. Springer-Verlag, 1989.

[18] D. Wong, G. De Micheli, and M. Flynn. Designing high-performance digital circuits using wave pipelining. In *VLSI'89*, pages pp. 241–252, 1989.

[19] A. Zilinskas. Axiomatic approach to statistical models and their use in multimodal optimization theory. *Mathematical Programming*, 22:pp. 104–116, 1982.

[20] A. Zilinskas. Axiomatic characterization of a global optimization algorithm and investigation of its search strategy. *Operations Research Letters*, 4(1):pp. 35–39, 1985.

Appendix

The model used for the problem above was:

$$Y(x) \;=\; \beta_0 + \sum_{j=1}^{d} \beta_j x_j \;+\; \phi(x) \tag{14}$$

where x is the d dimensional parameter vector and $x_0 = 1$. β is a $(d+1) \times 1$ vector of unknown coefficients. $\phi(.)$ is a random process with mean zero and covariance:

$$V(w, x) \;=\; \sigma^2 \Pi_{i=1}^{d} exp(- \mid w_j - x_j \mid) \tag{15}$$

Suppose that n values of $Y(x)$ are given at sample points $s_i, \ldots, s_n$. These values are in the $n \times 1$ vector ζ_S. F is the $n \times (d+1)$ matrix of the n parameter vectors at the sample sites augmented by a unit vector, i.e.

$$F \;=\; \begin{bmatrix} 1 & s_{11} & \ldots & s_{1d} \\ \vdots & & & \\ 1 & s_{n1} & \ldots & s_{nd} \end{bmatrix} \tag{16}$$

Then the Best Linear Unbiased Predictor $\hat{y}$ of $Y(.)$ is given as:

$$\hat{y} \;=\; X\hat{\beta} + r'(x)R^{-1}(\zeta_S - F\hat{\beta}) \tag{17}$$

where

$$\hat{\beta} \;=\; (F'R^{-1}F)^{-1}F'R^{-1}\zeta_S \tag{18}$$

where, as before, R is the $n \times n$ covariance matrix of the stochastic function $\phi(.)$ at the n sample locations, and r is the $n \times 1$ vector of covariances $V(x, s_k)$, $k = 1, \ldots, n$, and $X = [1\ x_1 \ldots\ x_d]$. The Mean Squared Error (MSE) is given as:

$$\text{MSE}(\hat{y}(x)) \;=\; \sigma^2 \left([X\ r'(x)] \begin{bmatrix} 0 & F' \\ F & R \end{bmatrix}^{-1} \begin{bmatrix} X' \\ r(x) \end{bmatrix} \right) \tag{19}$$

	M1	M2	M3	vbias	delay	skew
Best Random Sizing	6.15e-06	8.79e-06	8.55e-06	.98	9.9-10	7.7e-10
Closest Feasible Sizing	6.0e-6	8.4e-6	8.4e-6	.98	1.0e-09	7.7e-10
Best Sizing after Optimization	7.2e-6	8.4e-6	9.6e-6	0.0	8.8e-10	5.4e-10

Table 1: Results for Delay Controlled Element

	M1	M2	M3	M4	M5	M6	skew
Designer's choice	9.6e-6	4.8e-6	4.8e-6	9.6e-6	4.8e-6	4.8e-6	2.9e-10
Optimal Point	7.2e-6	3.6e-6	8.4e-6	9.6e-6	9.6e-6	4.8e-6	1.1e-10

Table 2: Results for Clock Driver Circuit

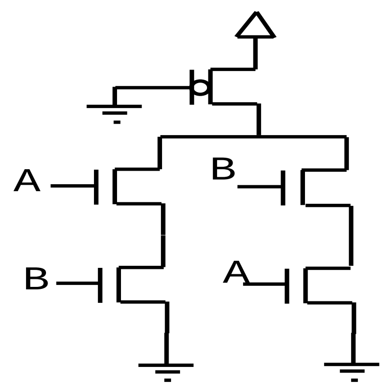
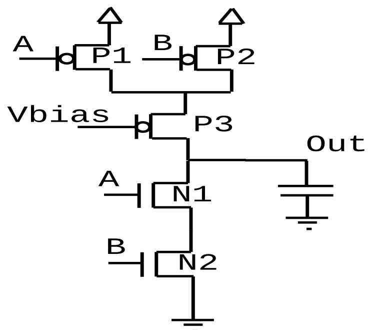
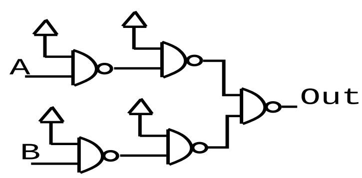


Figure 1: Cross coupled NAND gate



Delay Controlled Circuit Element



Test Circuit

Figure 2: Circuit Block for Wave-pipelining

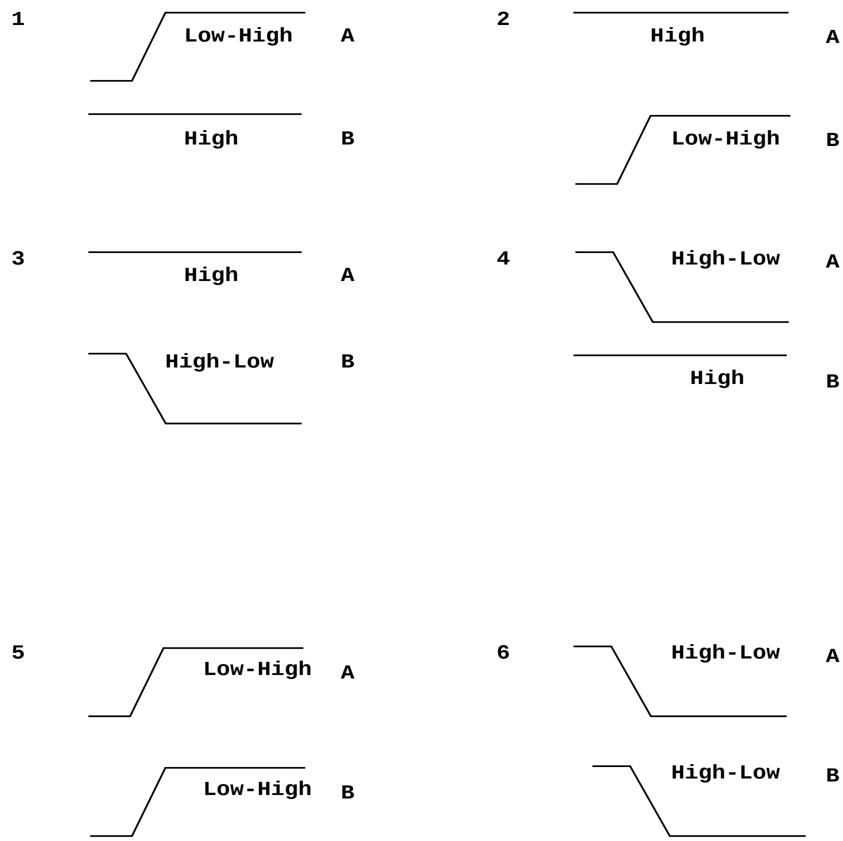


Figure 3: Possible input transitions

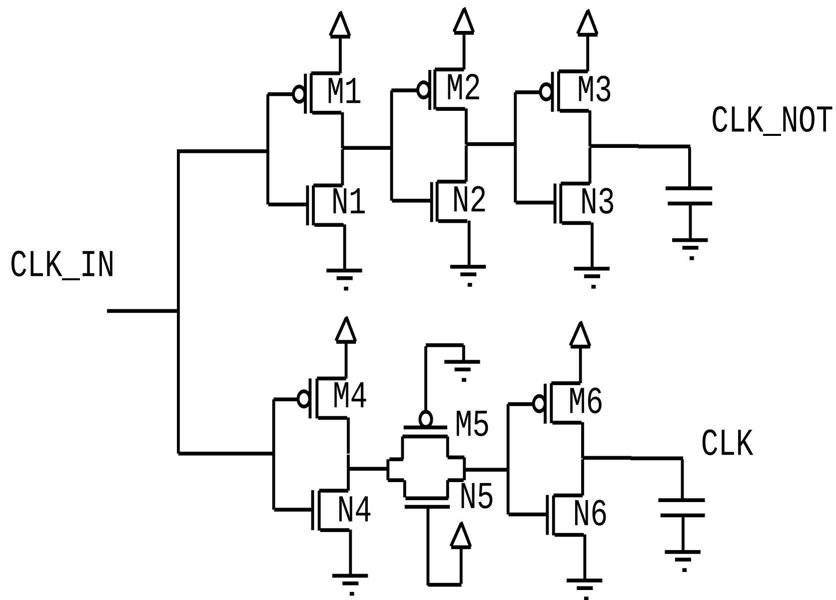


Figure 4: Clock Driver Circuit

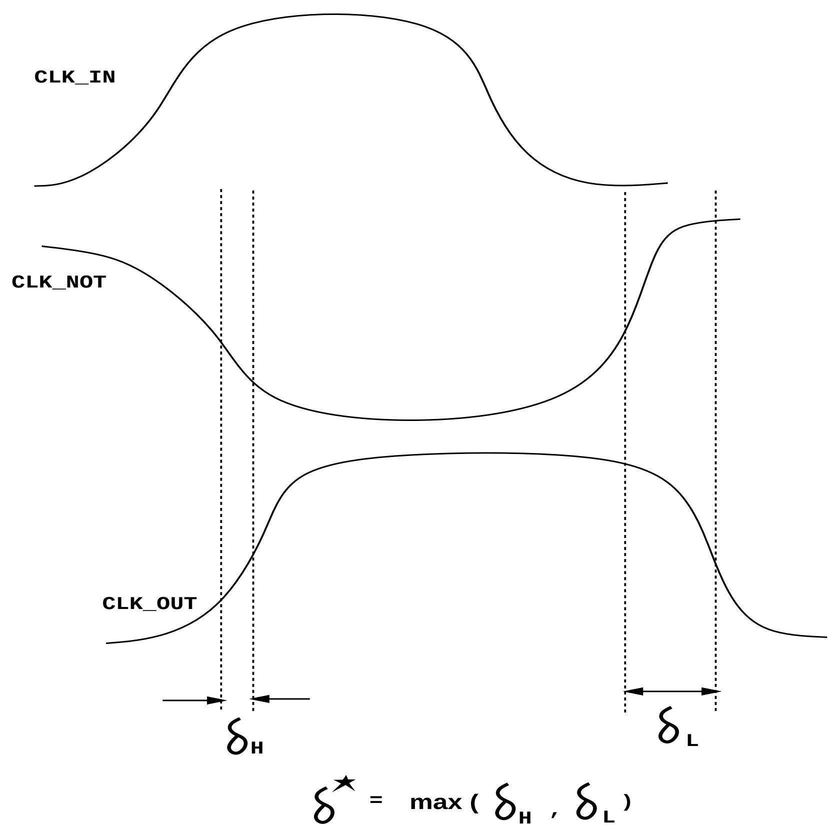


Figure 5: Skew Definition