A Global Driver Sizing Tool for Functional Crosstalk Noise Avoidance

Murat R. Becer, David Blaauw*, Supamas Sirichotiyakul*, Rafi Levy+, Chanhee Oh*, Vladimir Zolotov*, Jingyan Zuo* and Ibrahim N. Hajj
Coordinated Science Laboratory, University of Illinois at Urbana-Champaign
*Advanced Tools Group, Motorola Inc. Austin, TX, + Motorola Semiconductor Israel Ltd., Israel

Abstract

As coupling noise analysis and estimation is reaching a relative maturity with recent efforts, more effort is needed in correcting and/or avoiding failures that can be caused by coupling noise. In this paper, we present a global driver sizing tool which can be used in a complete noise avoidance tool along with other techniques such as wire spacing and wire sizing. The proposed approach is used along with ClariNet [1], which is a recent noise analysis tool, in a greater effort towards a total signal integrity solution. We first present the analytical, linear interconnect model used. We then show how this model is used to provide neccesary information for global driver sizing along with our novel algorithm. We finally present results on two industrial circuits including a large high performance control block.

1. Introduction

Functional noise failures have become a significant design and verification issue for large and high performance designs. Due to the current technology trends, the ratio of crosstalk capacitance to the total capacitance of a wire is increasing [2]. On the other hand, more aggressive and less noise immune circuit structures such as dynamic logic are being used more commonly due to performance requirements. These facts have brought a significant noise problem in high performance designs.

In noise analysis vocabulary, the nets on which crosstalk noise is injected by one or more of its neighbors are called the victim nets whereas the nets that inject this noise are called the aggressor nets. The effects of this noise from capacitive coupling can be divided into two. One is functional noise which is reffered to the type of noise that occurs on a victim net which is being held quiet by a driver. Crosstalk noise on such a victim causes a glitch which may propagate to a dynamic node or a latch, changing the circuit state and causing a functional failure. The other type of noise due to capacitive coupling is called noise on delay which is reffered to the type of noise that occurs when two capacitively coupled nets switch simultaneously. Depending on the direction of these transitions, the delays on both nets are affected [3], [5]. The focus of this paper is functional noise.

Several papers can be found in recent literature on crosstalk noise. The work in [12] derives bounds for crosstalk using a lumped model ignoring interconnect resistance while [4] introduces a simple upper bound for crosstalk under ramp inputs. The peak noise expression in [12] is extended by [9] and [6] to handle resistive interconnects. In another recent work on crosstalk in interconnects [10], [11], the authors introduce simple to compute noise metrics for crosstalk amplitude and pulse width in resistive, capacitively coupled lines. The derived expressions are also used to motivate circuit design techniques, such as transistor sizing and layout techniques such as wire width optimization to reduce crosstalk. Also in [13], the authors propose two interconnect layout design methodologies for minimizing the crosscoupling effect in data path design.

One of the better tools that recent efforts on detection and estimation of the effects of crosstalk noise have resulted in, is ClariNet [1] which uses fast linear simulation techniques to evaluate noise on signal lines. Clarinet speeds the analysis of large designs by using noise filters based on reduced interconnect representations and pruning the nets coupled to a signal net. ClariNet also incorporates logic and timing correlations into noise analysis to reduce pessimism.

In this paper, we focus on the next stage of the larger signal integrity problem: How do we avoid functional noise failures estimated by a noise analysis tool (in this case ClariNet)? Techniques that can be used for noise avoidance are wire spacing (which decreases coupling capacitance), wire widening (which decreases wire resistance and increses wire grounded capacitance) and driver sizing. The former two methods are suitable for incorporation with a router for an estimated pre-route noise avoidance tool [14]. But once the circuit is routed, it is not desirable to use those techniques which would require re-routing. In this paper, we will present a novel approach for a post-route noise avoidance tool which

*This work has been done in Advanced Tools Group, Motorola Inc. Austin, TX during author's summer internship

0-7695-1025-6/01 $10.00

utilizes driver sizing.

The paper is organized as follows. Section 2 defines the problem and the issues associated with it. Section 3 introduces the analytical linear model used and presents its accuracy. We discuss our proposed algorithm in section 4. In section 5, we give results obtained by running our tool on industrial circuits including a large, high performance control block. Section 6 contains some closing remarks.

2. Issues with Driver Sizing

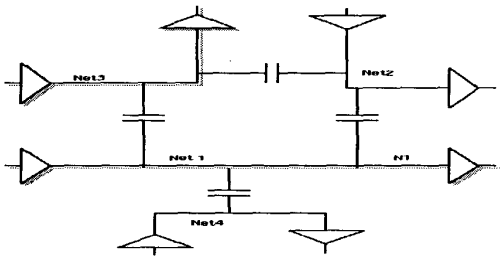

Figure 1. Capacitively coupled interconnects

Figure 1 shows several nets randomly coupled to each other, representing a general case in a practical situation. If we assume that net 1 is being held low and vary the driver strenght of driver 1 (victim) and look at the noise peak seen at node N1, we see that, as the driver strength of the victim driver is increased (i.e. its pull down power is increased by widening the NMOS transistor), the amount of noise induced on this net decreases. This is the motivation behind using driver sizing for noise avoidance. Now, let's assume that net 3 is being held low and the other drivers are switching. By increasing the driver strenght of driver 1, one actually worsens the noise induced on net 3 since net 1 is one of the aggressors of net 3. The fact that a net can both be an aggressor and a victim makes the driver sizing problem more complex and justifies the requirement of a global algorithm which takes into account all the interrelations among the nets of a block on which driver sizing will be applied. Only such an approach would guarantee that while trying to fix one problem, one does not create a new one.

3. The Analytical Model

A good coupled interconnect and gate model is neccesary for our approach to be successful. The model should be analytical which would let us see the effects of model parameters (such as the driver size) while not sacrificing speed and accuracy. Figure 2 shows the linear model we use in this work. This is an improved version of the model proposed in [3]. For aggressor drivers, a standard Thevenin model is used whereas the victim drivers are modeled as a linear holding resistor. Interconnects are modeled as pi circuits. The reader is referred to [1] for a detailed discussion on how the victim holding resistance is calculated.

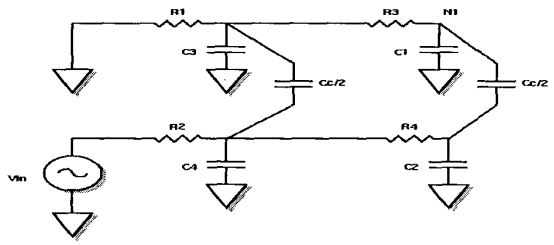

Figure 2. The linear model used in this work

After finding the Laplace domain expression for the voltage waveform on node N1, the time at which the peak noise will occur is calculated. By inserting this value into the analytical time domain voltage equation, one finds the peak noise in terms of the parameters of the circuit. The transfer function from the aggressor driver to the output node of the victim net represents a 4^{th} order system. We neglect the higher order terms, retaining a 2^{nd} order system.

$$\frac{V_{N_1}}{V_{in}} = \frac{a_1 s^2 + a_0 s}{b_2 s^2 + b_1 s + 1} \tag{1}$$

The coefficients a_0, a_1, b_1 and b_2 are in terms of the circuit parameters.

Assuming a ramp input at V_{in} with a rise time of t_r, we can express V_{N_1} as shown in Eqn. (2).

$$V_{N_1}(s) = \frac{V_{dd}(1 - e^{-st_r})}{t_r b_2} \left(\frac{r_0}{s} + \frac{r_1}{s - p_1} + \frac{r_2}{s - p_2} \right) \tag{2}$$

where p_1 and p_2 are the roots of $b_2 s^2 + b_1 s_1 + 1 = 0$.

When the inverse Laplace transform is calculated, we get the following time domain voltage equation

$$v_{N_1}(t) = \begin{cases} \frac{V_{dd}}{b_2 t_r} \left(r_0 + r_1 e^{p_1 t} + r_2 e^{p_2 t} \right) \text{ when } & t \le t_r \\ \frac{V_{dd}}{b_2 t_r} \left[r_1 \left(e^{p_1 t} - e^{p_1 (t - t_r)} \right) \right] + & \\ \frac{V_{dd}}{b_2 t_r} \left[r_2 \left(e^{p_2 t} - e^{p_2 (t - t_r)} \right) \right] & \text{when } t > t_r \end{cases} \tag{3}$$

where the residues have been found to be,

$$\begin{aligned} r_0 &= a_0 b_2 \\ r_1 &= -\frac{b_2 (a_1 + p_2 a_0 b_2)}{b_1 + 2 p_2 b_2} \\ r_2 &= -\frac{b_2 (p_2 a_0 b_2 + a_0 b_1 - a_1)}{b_1 + 2 p_2 b_2} \end{aligned} \tag{4}$$

By taking the derivative of Eqn.(3) and equating to 0, one gets the time at which the peak noise occurs on node N_1.

$$t_{p_1} = \frac{1}{p_2 - p_1} ln(-\frac{r_1 p_1}{r_2 p_2}) \quad (5)$$

$$t_{p_2} = \frac{1}{p_2 - p_1} ln\left[(-\frac{r_1 p_1}{r_2 p_2})(\frac{1 - e^{p_1 t_r}}{1 - e^{p_2 t_r}})\right] \quad (6)$$

Eqn.(5) is valid if $t_p \leq t_r$ and Eqn.(6) is valid if $t_p > t_r$. The peak noise can be calculated analyticially, when the valid t_p value is inserted back into Eqn.(3).

Note that these equations have also been calculated by Kahng et.al. in [7]. The accuracy of the model with respect to SPICE has been tested on a random logic block of 415 nets and the average error in peak noise voltage has been found to be 5%. Although we will not be using this model to calculate the noise voltages (ClariNet already does that), it is important to check its accuracy.

4. Global Driver Sizing

We use the model shown in Figure 2 to be able to analytically estimate the sensitivity of crosstalk noise on the circuit paramaters. Figure 3 shows the variation of peak noise at node N_1 with respect to R_1 in our model which represents the driver of the victim net.

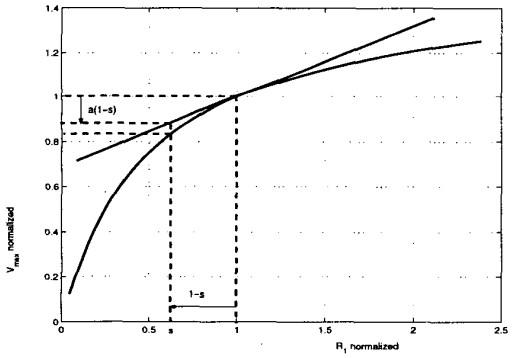

Figure 3. Rate of change of normalized V_{max} wrt normalized R_1

Note that this curve is normalized with respect to both V_{max} and R_1. Let us define the normalized slope shown in the figure as a. If R_1 is scaled by s to $R_1 \times s$, then the normalized V_{max} reduces by $a(1-s)$. If we multiply this value by the current $V_{noise12}$ (noise on net 1 induced by aggressor 2), we get an estimated reduction of noise on net 1 induced by driver 2:

$$change\_R1_{1-2} = a \times (1 - s_1) \times V_{noise12} \quad (7)$$

This is a first order estimation of the reduction of noise on net 1 induced by net 2. As can be seen from Figure 3, this reduction will always be an underestimation of the actual reduction if R_1 is scaled by s. If R_1 were to be scaled up, then this method would overestimate the increase in the noise amount.

On the other hand, we can also find the amount of change in noise when R_2 (which represents the driver of the aggressor net) is scaled using the same approach. Figure 4 shows the normalized variation of V_{max} with respect to R_2.

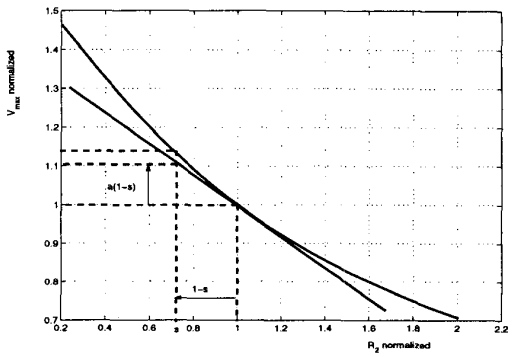

Figure 4. Rate of change of normalized V_{max} wrt normalized R_2

Thus the estimated change in V_{max} due to a scaling of R_2 by s can be written as:

$$change\_R2_{1-2} = b \times (1 - s_2) \times V_{noise12} \quad (8)$$

where b is the normalized slope shown in Figure 4

If R_2 is scaled down (i.e. if $s < 1$), the estimated noise will be slightly less than it actually is and if R_2 is scaled up (i.e. if $s > 1$), the same argument applies. From experimentation on several practical circuits and interconnects, we have observed that the normalized slope with respect to R_1 (term a in Equation (7)) is larger in absolute value than the normalized slope with respect to R_2 (term b in Equation (8)). Since the normalized slopes are directly related with the estimation error of this approach, this property means that noise will be overestimated most of the time, which makes this a safe method.

Let us return to the coupled interconnect cluster in Figure 1 and assume that after an accurate noise analysis, it is found that some of these nets have noise above the user specified threshold and some do not. We define $slack_i$ as the voltage difference between the allowed noise and actual noise on the

net:

$$slack_i = allowed\_noise_i - actual\_noise_i \quad (9)$$

So if $slack_i < 0$, then net_i is failing the noise analysis and needs to be corrected. If $slack_i > 0$, then net_i is 'OK' in terms of functional noise. Our goal is to find a scale s_i for each driver such that $slack_i > 0$ for all nets, i.e. no functional noise violations on the interconnect cluster, using driver sizing. If this is not possible, the uncorrectable nets' drivers will be scaled to a default value and the solution for the rest of the circuit will be seeked.

If we look at net_i as a victim net, the change of noise on net_i due to scaling of $driver_i$ will be:

$$\begin{aligned}\sum_{j=1,j\neq i}^{n}(1-s_i)a_{ij}V_{noiseij} &= (1-s_i)\sum_{j=1,j\neq i}^{n}a_{ij}V_{noiseij} \\ &= (1-s_i)A_i \quad (10)\end{aligned}$$

where

$$a_{ij} = \frac{\partial V_{maxij}}{\partial R_i}\frac{R_i}{V_{noiseij}} \quad (11)$$

Here, a_{ij} is the rate of change of normalized noise on net_i induced by $driver_j$ (V_{maxij}) with respect to $victim_i$'s normalized holding resistance (R_i) and $V_{noiseij}$ is the noise on net_i induced by $driver_j$ at the current point.

On the other hand, the change of noise on net_i due to scaling of $driver_{j\neq i}$ will be:

$$(1-s_j)b_{ij}V_{noiseij} \quad (12)$$

where

$$b_{ij} = \frac{\partial V_{maxij}}{\partial R_j}\frac{R_j}{V_{noiseij}} \quad (13)$$

Here, b_{ij} is the rate of change of normalized noise on net_i induced by $driver_j$ (V_{maxij}) with respect to $aggressor_j$'s normalized thevennin resistance (R_j) and $V_{noiseij}$ is the noise on net_i induced by $driver_j$ at the current point.

So the total change of noise voltage on net_i due to the scaling of all drivers $1 \ldots n$ by $s_1 \ldots s_n$ is:

$$\begin{aligned}total\_red_i &= (1-s_i)\sum_{j=1,j\neq i}^{n}\frac{\partial V_{maxij}}{\partial R_i}R_i + \\ &\quad \sum_{j=1,j\neq i}^{n}(1-s_j)\frac{\partial V_{maxij}}{\partial R_j}R_j \quad (14)\end{aligned}$$

$$\begin{aligned}total\_red_i &= (1-s_i)A_i + (1-s_1)A_1 + (1-s_2)A_2 + \\ &\quad \ldots + (1-s_{i-1})A_{i-1} + \\ &\quad (1-s_{i+1})A_{i+1} + \ldots + (1-s_n)A_n \quad (15)\end{aligned}$$

To make this net free of noise, we should set,

$$total\_red_i = -slack_i \quad (16)$$

If we set $1 - s_i = x_i$ and carry out this procedure for all nets, then we get a set of linear equations in the form $Ax = b$, where A is an $n \times n$ matrix (n is the number of nets) and b is an $n \times 1$ vector which has the negatives of the slack values for all nets. The x vector is the solution vector which will give the correct s (scale) values.

Notice that the above explained method is missing one important point. When the derivatives in the second part of Equation (14) are found, we cannot use the parameters in the original circuit since R_i will have scaled to $s_i R_i$, and this will change the slope. As a matter of fact $V_{max} = f(circuitparameters, s_i, s_j)$, is a function of s_i and s_j as well as the circuit parameters. To rid ourselves from this difficulty, we employ a few iterations. As a starting point, we use the following heuristic method:

4.1. Method to find the initial $(s_1 \ldots s_n)$ vector

In this section, we present a method which brings the initial solution to a point from which a minimal number of iterations are required. We neglect the effect of aggressor size changes and assume only victim drivers are scaled. If a victim is also an aggressor, a possible change in its size is ignored when looking at it as an aggressor. Using this method, we can calculate the initial s_i's using the following formula

$$(1-s_i)\sum_{j=1,j\neq i}^{n}\frac{\partial V_{maxij}}{\partial R_i}R_i = -slack_i \quad (17)$$

This can be thought as a 'local driver sizing' method where the fact that a victim can also be an agressor is not taken into account. This method finds s_i, then proceeds to find s_{i+1} not remembering that $driver_i$ is scaled to s_i. These initial victim driver scales are used to calculate b_{ij}'s in Eqn.(13). We will show results of this initial solution in the following section along with the global driving sizing results, which finds $s_1 \ldots s_n$ at once by solving a larger system.

4.2. Global driver sizing algorithm

- We take into account nets that have negative slacks
- We take into account nets that have positive slacks but not the ones that have $slack > MAX\_SLACK$
- If a net has a $slack > MAX\_SLACK$, then its driver is scaled to 1

There is another constraint that should also be taken into account. As a result of our algorithm, a scale can end up being greater than one. Since scaling up a driver resistance means scaling down the gate, this might have an undesired effect on delay requirements. So, there is a maximum scale, MAX_SCALE for each driver.

Also note that, we fit Chebyshev polynomials to the V_{max} function in Equation 3 and use this fit to evaluate the coefficients of the derivatives we need in our approach. This is a fast and accurate numerical derivation method [8].

We present our global driver sizing algorithm in Figure 5:

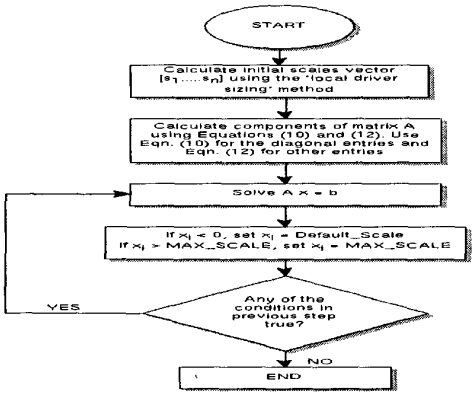

Figure 5. Global driver sizing algorithm

5. Results

We have implemented the global driver sizing algorithm presented in Figure 5 in C language on Sun environment. As a first example, we will present results on a small sized logic block *sip_lcl_top* which has 415 nets. Our run flow is as follows. We first run Clarinet and then our global driver sizing tool on its results. The data that we obtain from Clarinet are, the parasitic interconnect parameters that we use in our analytical model and information on each net (noise amount, slack). Results are presented in Table 1.

Block name: sip_lcl_top	
# of nets: 415	
Initial failing # of nets found by Clarinet: 40	
Scale	# of drivers
> 4x	6
4x (Default)	20
2x – 4x	11
1.33x – 2x	26
1x – 1.33x	17
< 1x	47
# of nets failing after implementing these suggestions: 9	

Table 1. Results of global driver sizing algorithm on sip_lcl_top

Scale numbers mentioned in Table 1 correspond to $1/s_i$ values found by our algorithm. Remember that scaling the resistance that represents a driver by s maps to scaling the driver size by $1/s$.

The number of nets used for the calculations by our algorithm was 127. This means that, although there were 40 nets in violation initially, there were a total of 127 nets interconnected to each other forming a cluster. Note that, the number of nets scaled to default (0.25) is 20. This means the algorithm estimated that we are not able to fix functional noise on these nets by scaling their drivers. But as a matter of fact, we had only 9 violations when we implemented the suggestions recommended by our algorithm. This is due to the pessimism of our approach mentioned as in section 4. One other measure of the quality of our algorithm is the slacks obtained after implementing the suggestions. We set a target positive slack of 20 mV during the run and it was observed that the average positive slack in the corrected nets was 22 mV.

We present the results if we let the algorithm stop after the 'local driver sizing' step, in Table 2.

Block name: sip_lcl_top	
# of nets: 415	
Initial failing # of nets found by Clarinet: 40	
Scale	# of drivers
> 4x	3
4x (Default)	15
2x – 4x	5
1.33x – 2x	9
1x – 1.33x	8
# of nets failing after implementing these suggestions: 21	

Table 2. Results of local driver sizing algorithm on sip_lcl_top

First of all, notice that there are 40 drivers scaled. This is a result of looking at victim nets only due to the local approach. When we compare the nets in violation after the suggestions are implemented with those from the global driver sizing run, it is seen that the same 9 nets fail and there is an addition of 12 nets in the local case. The average slack of these 12 nets is -11mV. The global approach corrects these nets which are missed in the local approach.

The second example that we present is a large high performance control block *qctl_s* of around 46000 nets. Results can be seen in Table 3.

Due to the size of this block, we ran our global driver sizing tool on a sub-block of 299 nets which had noise violations. The average positive slack in the corrected nets was 81 mV while the target positive slack was 30 mV.

The results at the end of local driver sizing step are also presented in Table 4.

As can be seen in these results, the global driver sizing algorithm reduces the number of violations dramatically (77%

Block name: qctl_s	
# of nets: 46207	
Initial failing # of nets found by Clarinet: 299	
Scale	# of drivers
> 4x	24
4x (Default)	152
2x – 4x	37
1.33x – 2x	39
1x – 1.33x	23
< 1x	24
# of nets failing after implementing these suggestions: 31	

Table 3. Results of global driver sizing algorithm on qctl_s

and 90% in presented examples) while controlling the positive slack on corrected nets accurately (within 50mV). These results also show that, local driver sizing method is a good starting point for our global driver sizing approach.

Block name: qctl_s	
# of nets: 46207	
Initial failing # of nets found by Clarinet: 299	
Scale	# of drivers
> 4x	44
4x (Default)	44
2x – 4x	70
1.33x – 2x	72
1x – 1.33x	69
# of nets failing after implementing these suggestions: 59	

Table 4. Results of local driver sizing algorithm on qctl_s

6. Conclusion and Future Work

A novel global driver sizing algorithm and its implementation has been presented. Our approach takes into account the fact that a net can be both a victim and an aggressor, creating and solving a system which contains data on the entire block that the tool is run on. An analytical interconnect model which we used to find sensitivities and its accuracy were presented. We have also introduced a local driver sizing algorithm, which is used as the starting point of our global approach. The same mathematical sensitivity approach using an analytical interconnect model is also utilized for other noise avoidance techniques such as wire sizing and wire seperation which we will present in a future paper. The tool presented in this paper can be used as part of a comprehensive noise avoidance tool which also can be incorporated into a noise analysis tool to constitute a full signal integrity solution for functional capacitive noise in deep submicron designs. Results presented in section 5 show the accuracy of our model along with the effectiveness of driver sizing in noise avoidance.

References

[1] S. Alwar, D. Blaauw, A. Dasgupta, A. Grinshpon, R. Levy, C. Oh, B. Orshav, S. Sirichotiyakul, and V. Zolotov. Clarinet: A noise analysis tool for deep submicron design. In *Proceedings of Design Automation Conference DAC*, pages 233–238, June 2000.

[2] S. I. Association. The international technology roadmap for semiconductors, 1999.

[3] M. R. Becer and I. N. Hajj. An analytical model for delay and crosstalk estimation with application to decoupling. In *Proceedings of the IEEE International Symposium on Quality Electronic Design ISQED*, pages 51–57, March 2000.

[4] A. Devgan. Efficient coupled noise estimation for on-chip interconnects. In *Proceedings of the IEEE International Conference on Computer-Aided Design, ICCAD-97*, pages 147–153, 1997.

[5] P. D. Gross, R. Arunachalam, K. Rajagopal, and L. T. Pileggi. Determination of worst-case aggressor alignment for delay calculation. In *Proceedings of the IEEE International Conference on Computer-Aided Design, ICCAD-98*, 1998.

[6] C. Guardini, C. Forzan, B. Franzini, and D. Pandini. Modeling the effect of wire resistance in deep submicron coupled interconnects for accurate crosstalk based net sorting. In *Proceedings of PATMOS-98*, pages 8.2.1–8.2.10, October 1998.

[7] A. B. Kahng, S. Muddu, and D. Vidhani. Noise and delay uncertainty studies for coupled rc interconnects. In *Proceedings of ASIC/SOC Conference*, pages 3–8, 1999.

[8] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C The Art of Scientific Computing*. Cambridge University Press, New York, NY, 1992.

[9] T. Stohr, H. Alt, A. Hetzel, and J. Koehl. Analysis, reduction and avoidance of crosstalk on VLSI chips. In *Proceedings of International Symposium on Physical Design*, pages 211–218, 1998.

[10] A. Vittal, L. H. Chen, M. Marek-Sadowska, K. P. Wang, and S. Yang. Crosstalk in VLSI interconnections. *IEEE Transactions on Computer Aided Design*, 18:1817–1824, December 1999.

[11] A. Vittal, L. H. Chen, M. Marek-Sadowska, K. P. Wang, and X. Yang. Modeling crosstalk in resistive VLSI interconnections. In *Proceedings of International Conference on VLSI Design*, pages 470–475, January 1999.

[12] A. Vittal and M. Marek-Sadowska. Crosstalk reduction for VLSI. *IEEE Transactions on Computer Aided Design*, 16:290–298, March 1997.

[13] J. S. Yim and C. M. Kyung. Reducing crosscoupling among interconnect wires in deepsubmicron datapath design. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 485–490, 1999.

[14] H. Zhou and D. F. Wong. Global routing with crosstalk constraints. In *Proceedings of Design Automation Conference DAC*, pages 374–377, 1998.