Extended Global Routing With RLC Crosstalk Constraints *

James D. Z. Ma
ECE Department
University of Wisconsin, Madison

Lei He
ECE Department
University of Wisconsin, Madison

Abstract

Experiments using large industrial benchmark circuits show that up to 25% of nets may have RLC crosstalk violations at 3GHz clock. We develop an accurate and extremely efficient length-scaled K_{eff} (LSK) model for long-range RLC crosstalk, and formulate an extended global routing problem (denoted as GSINO) to leverage simultaneous shield insertion and net ordering with the RLC crosstalk constraints. We then propose an effective three-phase GSINO algorithm to completely eliminate the RLC crosstalk violations. Compared to the best alternative method we have studied, the GSINO algorithm reduces the chip area by up to 21%.

1. INTRODUCTION

As VLSI technology advances, crosstalk becomes increasingly critical [1, 2]. Routing techniques such as net ordering [3, 4, 5], spacing [6], and layer assignment [7] have been studied to reduce the capacitive crosstalk within a routing channel or switchbox. To overcome the limited flexibility of channel or switchbox routing, simultaneous track and layer assignment as well as pseudo pin assignment has also been developed at the full chip level [8, 9]. Furthermore, a global routing adjustment procedure [10] has been developed via iterative region-based crosstalk estimation and reduction considering track assignment. Moreover, an extended global routing problem has been proposed to take into account layer/track assignment and a single shield insertion into each routing region [11].

A few recent works have addressed crosstalk avoidance techniques for both capacitive and inductive crosstalk. Examples include: shielding [12], simultaneous shield insertion and net ordering (SINO) [13, 14], twisted bundle layout structure [15], and differential signaling [16]. However, there has been no in-depth study on automatic global routing that is able to consider the above techniques.

Different from capacitive crosstalk which exists only between adjacent wires, inductive crosstalk has a long-range effect, i.e., it may affect both adjacent and non-adjacent wires. Therefore, global routing considering inductive crosstalk is much more difficult than global routing considering only capacitive crosstalk [11]. Our major contributions in this paper include the following:

- We propose a simple yet accurate length-scaled K_{eff} (LSK) model for long-range RLC crosstalk. The new model has negligible running time and an error of about 15% compared to the SPICE simulation under the distributed RLC circuit model.
- We formulate an extended global routing problem (denoted as GSINO) to meet RLC crosstalk constraints with consideration of simultaneous shield insertion and net ordering, and develop the following three-phase algorithm: in Phase I, we partition the crosstalk bound under the LSK model among routing regions, synthesize a global routing solution, and reserve as well as minimize the shielding area needed to meet the partitioned crosstalk bound; in Phase II, we find within each routing region a minimum area SINO solution that meets the crosstalk bound; in Phase III, we apply a greedy algorithm to eliminate remaining (but very limited) crosstalk violations and reduce routing congestion.
- Experiments using large industrial benchmark circuits show that up to 25% of nets may have RLC crosstalk violations at 3GHz clock. Yet our GSINO solutions are able to completely eliminate the RLC crosstalk violations, and reduce the chip area by up to 21% when compared to the best alternative solutions we have studied.

The rest of the paper is organized as follows: Section 2 introduces the preliminaries and proposes the LSK model. Section 3 formulates the GSINO problem and develops a three-phase GSINO algorithm. Section 4 presents the experiment results, and concludes the paper with discussions on future work.

2. RLC NOISE MODELING

2.1 Preliminaries

We consider over-the-cell routing for global interconnects in this paper. We denote m signal nets as $\mathcal{N} = \{N_1, N_2, ..., N_m\}$. Each net N_i ($1 \leqslant i \leqslant m$) has a number of pins ($s_{i0}, s_{i1}, ..., s_{in}$), where s_{i0} is the source and s_{ij} ($1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant n$) is the sink. We assume that there is a pair of routing layers with no obstacles. One routing layer is for horizontal wires, and the other for vertical wires. Further, the routing layers are divided by pre-routed power/ground (P/G) networks into routing *regions* $\mathcal{R} = \{R_1, R_2, ..., R_p\}$. Each region R_k ($1 \leqslant k \leqslant p$) has the size of $w \times h$. The number of horizontal (vertical) tracks available for each region

*This research is partially supported by NSF CAREER Award CCR-0093273 and Intel. We used computers donated by HP and SUN Microsystems. Address comments to lhe@ece.wisc.edu.

is the horizontal (vertical) capacity $HC(R_k)$ ($VC(R_k)$). A track can be used by a segment of either a net or a shield. A shield is a wire directly connected (without through devices) to P/G networks. One convenient way to connect shields is to add vias between shields and P/G networks. Moreover, we assume that all global interconnects have the same driver resistance and loading capacitance, and all wires (except P/G wires) have the same width, spacing, and thickness. Further, P/G wires are wide enough that there is *no* crosstalk between regions separated by P/G wires.

According to [13, 14], two signal nets N_1 and N_2 are *sensitive* to each other if a switching signal on N_1 causes N_2 to malfunction (due to extraordinary crosstalk or delay variation). In this case we call N_1 an aggressor for N_2 and N_2 a victim of N_1. We assume that there is no coupling between different routing regions and two non-sensitive nets do *not* switch simultaneously, and define the *sensitivity rate* of N_i as the ratio of the number of aggressors for N_i to the total number of signal nets. The simultaneous shield insertion and net ordering (SINO) problem has been studied to find the minimum area solution with the following RLC crosstalk constraints: all signal nets are capacitive crosstalk free (i.e., no sensitive nets are adjacent to each other) and has inductive crosstalk (or equivalently, the total crosstalk) less than a given threshold. Compared to the dense wiring fabric scheme [17], the SINO scheme achieves up to 42% area reduction with RLC crosstalk constraints [18]. In this paper, we will develop an extended global routing algorithm leveraging SINO to minimize the chip area with RLC crosstalk constraints. An accurate and extremely efficient model for RLC crosstalk, such as the LSK model to be developed below, is of paramount importance for such a routing algorithm.

2.2 LSK Model

To develop and verify the LSK model, we consider the derived ITRS $0.10\mu m$ technology (see Table 1 [19]). We use SPICE to compute the noise voltage induced on a signal net in the SINO solution under the following distributed RLC circuit model: we divide each wire into $100\mu m$-long segments and connect each shield to P/G wires for every $100\mu m$-long segment.[1] Each segment is modeled by an RLC π-model, with a coupling capacitance to its adjacent segment that does not belong to the same wire. There is a coupling inductance between any two segments if they are not orthogonal to each other. The capacitance and inductance are calculated based on [12, 20]. We consider the maximum noise under the worst-case signal switching pattern, and measure the noise at the input of the receiver. Such noise may introduce logic failure, and reduce device reliability even if the noise is not over the logic threshold value.

A formula-based K_{eff} model has been proposed in [13] to characterize inductive coupling between two signal nets. It uses a formula to calculate the coupling coefficient K_{ij} between two signal nets N_i and N_j. The total amount of inductive coupling K_i induced on N_i is $\sum_{j \neq i} K_{ij}$ for all signal nets that are sensitive to N_i. The K_{eff} model is easy to compute and convenient to use at a higher design level

[1] Compared to shields that are connected to P/G wires only at two ends in [16], our better-connected shields provide closer current return paths and reduce much more crosstalk in our experiments.

Vdd	1.05V	load capacitance	60fF
frequency	3GHz	wire width	$1.0\mu m$
input rising time	$33ps$	wire thickness	$1.1\mu m$
driver resistance	150Ω	wire spacing	$0.8\mu m$

Table 1: Interconnect specifications based on derived ITRS 0.10μm technology.

or an earlier design stage. Further, K_i has a high fidelity in the following sense [21]: for a SINO solution of multiple nets with a fixed wire length, a signal net with a higher K_i value given by the K_{eff} model also has a higher SPICE-computed noise voltage. To generalize this model, we present in Figure 1 the SPICE-computed noise voltage versus the wire length. Each curve represents the noise voltage values for a signal net in a given SINO solution where all nets have uniform wire length. One can see that the noise voltage is roughly a linearly increasing function of the wire length, and this observation holds for a large number of SINO solutions in our experiments considering different design and fabrication technologies.

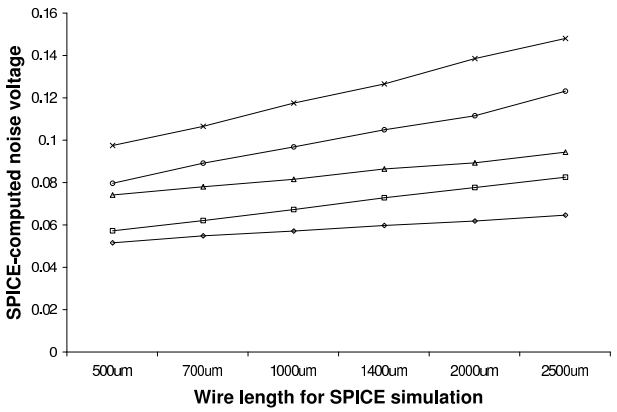

Figure 1: SPICE-computed noise voltage versus wire length for given min-area SINO solutions.

Based on the generalization of the above observation, we propose the following length-scaled K_{eff} model (in short, LSK model): if the interconnect from the source to a sink of net N_i goes through a few routing regions where the total inductive coupling for the interconnect is K_i^j in region R_j and l_j is the length of N_i in R_j, then $\sum_j l_j \cdot K_i^j$ can be used as a figure of merit for the RLC crosstalk voltage at the above sink. Please note that the LSK model can only be applied to a net N_i with the following property: none of the neighboring wires of N_i switches simultaneously with N_i. All nets in a SINO solution satisfy this property.

Further, we compute the RLC crosstalk voltage directly from $\sum_j l_j \cdot K_i^j$ by looking up a table with two columns, one for $\sum_j l_j \cdot K_i^j$, and the other for the crosstalk voltage. To simplify the table building, we first generate a number of SINO solutions for a single routing region where $\sum_j l_j \cdot K_i^j$ is equivalent to $l \cdot K_i$ with l being the length of the single region and K_i being computed by the K_{eff} model. We then compute the corresponding crosstalk via SPICE simulation for $l = 1000\mu m, 1400\mu m$, and $2000\mu m$. Our table used in

the paper contains 100 entries, with crosstalk voltage values from 0.10V to 0.20V, which is about 10% $\sim$ 20% of the voltage supply Vdd.

Note that we assume a uniform driver and receiver for all interconnects, and the aforementioned table should be re-computed for a different combination of driver and receiver. Our future work includes generalization the LSK model for non-uniform drivers and receivers, and alternative ways to compute crosstalk voltage from $\sum_j l_j \cdot K_i^j$, specifically considering multiple-region SINO solutions for higher accuracy.

2.3 Model Verification

We first consider two-pin nets such as wires in the bus structure. We assume that all nets have uniform wire length and are divided into a row of 10 cascaded routing regions, each $200\mu m$ long. Each net segment is assigned with a random crosstalk bound K_{th} under the K_{eff} model, and the SINO algorithm is carried out within each region to meet the given crosstalk bound. We consider two bus structures with 32 and 64 signal nets respectively, and compare the noise values given by the LSK model via look-up table with those given by SPICE simulation in Figure 2. One can easily see that in the vast majority of cases, the error is less than $\pm 15\%$.

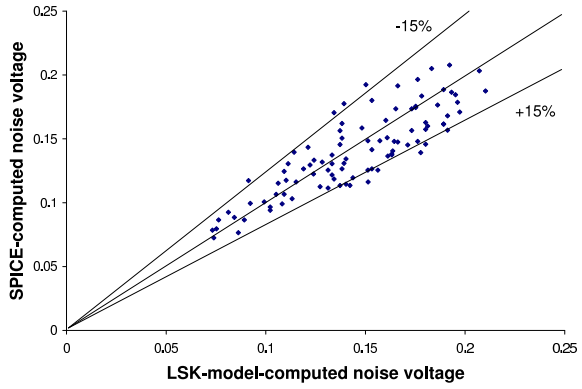

Figure 2: Comparison of LSK-model and SPICE-computed noise voltage for two-pin nets.

We then verify the LSK model for a multi-pin signal net. As shown in Figure 3, NET_1 has its source in Region 1 and two sinks in Region 2 and Region 3, respectively. It also has a different neighboring wires in different regions. Once again, we randomly assign K_{th}'s to all net segments and perform SINO within each region. We then compare the crosstalk voltage values under the LSK model and SPICE simulation for sinks s_1 and s_2. As indicated in Table 2, the errors are less than 20%.

In [10, 11], the capacitively coupled length has been used to model the capacitive crosstalk. Even though no SPICE simulation is presented to verify its accuracy, the model can be viewed as a length-scaled model with a constant coupling (and length scaling) coefficient over the chip when the wire width and spacing is uniform. However, the coupling coefficient in our LSK model is no longer a constant due to the long-range effect of the RLC crosstalk, and is a function of the routing solution. Therefore, the routing solutions in [10, 11] may not be applicable to the RLC crosstalk.

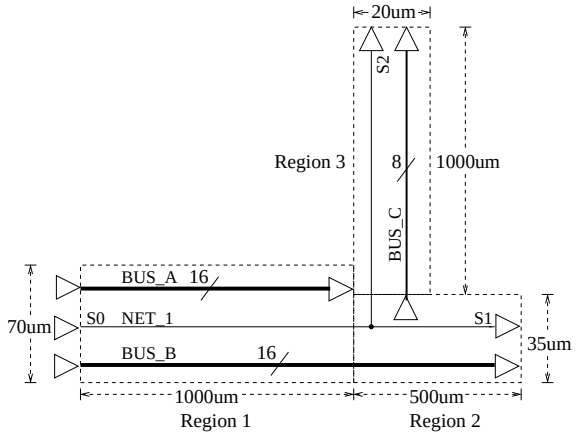

Figure 3: Illustration of a multi-pin signal net with its neighboring wires in three regions.

	LSK model	SPICE	error (%)
s_1	0.118V	0.1412V	-16.43%
s_2	0.163V	0.1376V	18.46%

Table 2: Comparison of LSK model and SPICE-computed noise voltage for the three-pin net in Figure 3.

3. GSINO FORMULATION AND ALGORITHM

3.1 Formulation and Algorithm Overview

The GSINO problem decides a Rectilinear Steiner Tree RST_i for each net N_i such that N_i can be routed with either L- or Z-shaped wires in a sequence of routing regions, and finds a SINO solution within each region such that the given RLC crosstalk constraint is satisfied for each net. Its objective is to minimize the total wire length and chip area.

The GSINO problem has a high complexity, as even its sub-problem SINO is NP-hard [13]. Therefore, we propose the following three-phase heuristic algorithm (see Figure 4). In Phase I, we partition the crosstalk budget among routing regions for each signal net, and perform global routing with consideration of allocating and minimizing shielding area. In Phase II, we simply find SINO solutions within each routing region for partitioned crosstalk bounds. In Phase III, we carry out post-SINO refinement to eliminate remaining (but very limited) crosstalk violations and reduce the routing congestion. We present details of Phase I and III below, and refer the readers to [13] for the SINO algorithm in Phase II. Overall, the GSINO problem can be viewed as an extended global routing problem similar to [11], because net segments are assigned not only to regions but also to tracks.

GSINO Algorithm
Phase I: Crosstalk budget partitioning and global routing.
Phase II: SINO within each region.
Phase III: Post-SINO refinement.

Figure 4: Overview of the GSINO algorithm.

3.2 Phase I Algorithm

For simplicity of presentation, we assume a uniform constraint of the crosstalk voltage for all sinks. Both our algorithm and program implementation, however, can handle non-uniform crosstalk constraints. For crosstalk budget partitioning, we first map the crosstalk voltage into a LSK value by the table look-up approach in Section 2. We then find $L_{e,ij}$, the semi-perimeter of the bounding box for source s_{i0} and a specific sink s_{ij} of signal net N_i. We assume that no detours are allowed in our routing algorithm such that $L_{e,ij}$ is exactly the wire length from the source to the sink in the final routing solution. Finally, we compute the inductive coupling bound as $K_{th} = \frac{l \cdot K}{L_{e,ij}}$ if N_i has a segment routed through the given region with length l. For a net segment on the common paths from source to multiple sinks, its K_{th} is the minimum of those bounds evaluated by individual paths. Experimental results in Section 4 will show that this uniform partitioning provides a good starting point for better crosstalk budgeting to reduce the congestion in Phase III.

With K_{th} available for each net segment, we apply the iterative deletion (ID) algorithm [22] to synthesize a global routing solution with shielding area (i.e., a number of tracks) properly reserved and more importantly minimized. We choose the ID algorithm because it is able to consider all the signal nets simultaneously and thus is independent of the routing order. The ID algorithm is less efficient but may lead to better solutions compared to other order-dependent routing approaches. Nevertheless, our algorithm framework can be applied to other routing algorithms.

In the ID algorithm, we first define the *net connection graph* for net N_i as an undirected graph $G_i = (V_i, E_i)$, where V_i is the set of regions within the bounding box for the pins of net N_i and there is an edge $e_{i,jk} \in E_i$ if $V_{i,j}$ and $V_{i,k}$ are adjacent regions. G_i is *admissible* if it contains at least one connection tree of N_i for all the pins. We say that an edge e_i is *non-critical* if G_i remains admissible after e_i is removed from G_i. On the contrary, an edge e_i is *critical* if its removal causes G_i to be non-admissible. Once a non-critical edge becomes an critical edge, it is "frozen" and will not be non-critical. Let Ω be the set of the admissible graphs we construct for all the nets. We iteratively delete the "worst" non-critical edge from Ω until Ω is a net connection forest which means that all admissible graphs reduce to connection trees. To determine which of the edges to be eliminated from Ω in the ID process, a weight is assigned to the horizontal edge as

$$w(e) = \alpha \cdot f(wire\_length) + \beta \cdot HD(R_i) + \gamma \cdot HOFR(R_i) \quad (1)$$

where f is the relative wire length, i.e., the wire length normalized with respect to the estimated wire length of the Rectilinear Steiner Minimum Tree ($RSMT$) for the current net. Further, $HD(R_i)$ is the relative routing density defined as $\frac{HU(R_i)}{HC(R_i)}$. $HC(R_i)$ is the horizontal capacity of the region, and $HU(R_i) = N_{ns} + N_{ss}$ is the utilization of horizontal tracks, with N_{ns} being the number of net segments in the region and N_{ss} being the number of shield segments needed by the min-area SINO solution to satisfy the K_{th} constraint for each net segment. Moreover, $HOFR(R_i)$ is the relative horizontal overflow, i.e., the number of overflow net segments over the routing capacity. Finally, α, β, and γ are three constants and can be tuned for different objectives. Generally, γ is much larger than α and β so that virtually no overflow is allowed in the final global routing solution. One possible set of values is α=2, β=1, and γ=50. Similarly the weight for the vertical edge can also be defined.

We use a modified depth-first search algorithm to mark all the non-critical edges of each admissible graph in Ω. After removing the edge of the maximum weight in each iteration, we update the weight of those affected edges and re-mark the non-critical edges. Advanced data structures, such as buckets, can be used to speed up the selection of the maximum weighted non-critical edge and updating the weight of edges. The ID algorithm is summarized in Figure 5. A straightforward implementation of the ID algorithm has time complexity of $O(M^2P)$, and sophisticated data structures may result in $O(M^2 \log P)$ time, where M is the total number of edges in Ω and P is the total number of pins.

ID Algorithm
For each net N_i
  construct G_i and mark non-critical edges.
Let Ω be the set of G_i's.
Repeat
  e = maximum weighted non-critical edge in Ω.
  remove e from the corresponding G_i.
  re-mark the non-critical edges in G_i.
  update the weight of the affected edges.
Until Ω is a net connection forest.

Figure 5: ID algorithm.

a_1 = -0.448	a_2 = 0.624	a_3 = 21.67	a_4 = -29.87
a_5 = 0.384	a_6 = -0.337	a_7 = -22.83	a_8 = 31.45
a_9 = -0.12	a_{10} = 0.202	a_{11} = 5.47	a_{12} = -6.28

Table 3: Values of the coefficients in Formula (2).

Please note that given number of net segments N_{ns} in the region, the K_{th} bound for every net segment, and their sensitivities (S_i's), number of shield segments N_{ss} can be estimated by

$$\begin{aligned} N_{ss} = {} & a_1 \cdot \frac{1}{N_{ns}} \cdot \sum_{i=1}^{N_{ns}} K_{th,i} \cdot \sum_{i=1}^{N_{ns}} S_i^2 + a_2 \cdot \sum_{i=1}^{N_{ns}} S_i^2 \\ & + a_3 \cdot \frac{1}{{N_{ns}}^2} \cdot \sum_{i=1}^{N_{ns}} K_{th,i} \cdot \sum_{i=1}^{N_{ns}} S_i^2 + a_4 \frac{1}{N_{ns}} \cdot \sum_{i=1}^{N_{ns}} S_i^2 \\ & + a_5 \cdot \frac{1}{N_{ns}} \cdot \sum_{i=1}^{N_{ns}} K_{th,i} \cdot \sum_{i=1}^{N_{ns}} S_i + a_6 \cdot \sum_{i=1}^{N_{ns}} S_i \\ & + a_7 \cdot \frac{1}{{N_{ns}}^2} \cdot \sum_{i=1}^{N_{ns}} K_{th,i} \cdot \sum_{i=1}^{N_{ns}} S_i + a_8 \cdot \frac{1}{N_{ns}} \cdot \sum_{i=1}^{N_{ns}} S_i \\ & + a_9 \cdot \sum_{i=1}^{N_{ns}} K_{th,i} + a_{10} \cdot N_{ns} + a_{11} \cdot \frac{1}{N_{ns}} \cdot \sum_{i=1}^{N_{ns}} K_{th,i} \\ & + a_{12} \end{aligned} \quad (2)$$

where the values of coefficients are listed in Table 3. Different from the shielding estimation formula with uniform K_{th} in [21], we replace the terms containing uniform K_{th} with the arithmetic mean of non-uniform K_{th}'s. In Figure 6 we verify it by comparing the numbers of shields estimated by Formula (2) with those obtained by *individual* SINO solutions. The error is less than 10% for *each and every* SINO solution.

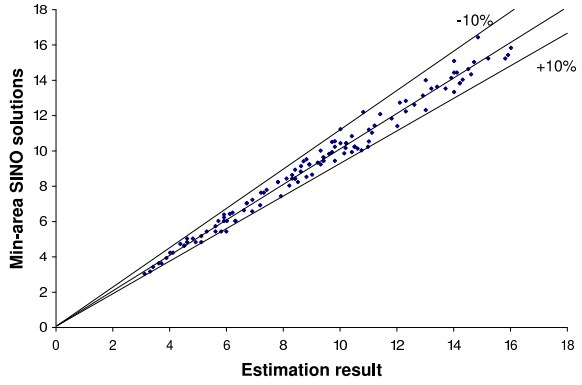

Figure 6: Comparison of numbers of shields between Formula (2) and min-area SINO solutions. All errors are less than 10%.

It is worthwhile to emphasize that according to Formula (2), the number of shields in a region is proportional to the density of sensitive nets. Because net segments in different regions are not sensitive to each other and the weight of edge (1) considers the density of sensitive nets in order to reduce such density, our new ID algorithm is able to distribute evenly the senisitive nets across the chip and therefore reduce the densities of sensitive nets in regions as well as the total number of shields. This is verified by the experimental comparison between numbers of shields for *i*SINO and GSINO (see Section 4).

3.3 Phase III Algorithm

As shown in Figure 7, Phase III contains two passes of greedy-based refinements (denoted as RF) to eliminate crosstalk violations and reduce routing congestion. The SINO algorithm from [13] is based on simulated annealing. The crosstalk constraint is implemented as a component of the cost function for the simulated annealing. A very limited number of crosstalk violations may exist after Phase II. To implement a "better" SINO algorithm for Phase II such that all net segments satisfy the *sub-optimally* distributed crosstalk constraint within *each* and *every* region may lead to over-design. Therefore, we choose to eliminate remaining crosstalk violations through post-SINO refinement rather than a "better" SINO algorithm.

There are two loops in the first pass to eliminate crosstalk violations. The outer loop picks the signal net N_i with the most severe crosstalk violation, and the inner loop locates the least congested routing region R_i through which N_i is routed through. Further, the inner loop allows exactly *one* more shield to be added into R_i, and uses Formula (2) to decide how much the K_{th} can be reduced for the net segment of N_i. Finally, SINO is invoked again in R_i with respect to the reduced K_{th}. This inner loop is iterated until we eliminate the crosstalk violation for N_i. The outer loop is stopped when there are no more crosstalk-violating nets.

There are also two loops in the second pass to reduce congestion. In the outer loop, we start with the most congested region R_i and compute *slack* as K_i - K_{th} for all the nets that are routed through region R_i. To avoid being too greedy, in the inner looper we allow only *one* shield to be removed at a time. Again, Formula (2) is used to decide how much increase on K_{th} is allowed. The inner loop iteration is stopped when one shield can be removed according to Formula (2). We then re-run SINO under the increased K_{th}'s. If there are no crosstalk violations, we accept the new SINO solution, and update the congestion of this region and the *slack*'s of affected nets. Otherwise, we keep the original SINO solution. The outer loop is stopped when no reduction on *slack*'s is possible without causing crosstalk violations.

Pass 1: Eliminate crosstalk violations
Repeat
  N_i = the net with most severe crosstalk violation
  Repeat
    R_i = least congested region containing N_i
    decrease K_{th} for N_i's segment by
    allowing one more shield in R_i
    re-do (SINO, R)
  Until N_i has no crosstalk violation
Until no nets have crosstalk violation

Pass 2: Reduce routing congestion
Repeat
  R_i = the most congested region
  compute *slack* for nets routed through region R_i
  Repeat
    N_i = net with the most *slack*
    increase K_{th} for N_i's segment in R_i by N_i's *slack*
  Until one shield can be removed from R_i
  new_SINO = **re-do** (SINO, R)
  if (new_SINO causes no crosstalk violations) **then**
    SINO=new_SINO
    update congestion of region R_i.
    update the *slack*'s of affected nets.
Until no reduction on *slack* is possible without causing crosstalk violation

Figure 7: Post-SINO Refinement (RF) algorithm.

4. EXPERIMENTAL RESULTS AND DISCUSSIONS

We have implemented the three-phase GSINO algorithm using C/C++ on a UNIX workstation, and applied it to the ISPD'98/IBM benchmark suite shown in Table 4. We use the placement generated by DRAGON [23], and set the crosstalk constraint as 0.15V, around 15% of the supply voltage Vdd for all sinks. Based on the capacities of the routing regions and the interconnect design specifications in Table 1, we derive the routing region's physical size for each benchmark circuit.

We compare GSINO with the following two approaches:

1. ID+NO: Traditional global router (ID) to minimize wire length and congestion, followed by net ordering (NO) within each region to eliminate as much capacitive coupling as possible. In order to make fair comparisons, we still consider the weight function (1) but without N_{ss} in $HU(R_i)$, as no shields are used in ID+NO.

2. *i*SINO: Traditional global router followed by SINO within each individual routing region and post-SINO refinement.

	number of nets	number of regions	number of pins	region's capacity	region's physical size ($\mu m \times \mu m$)
ibm01	13056	$64 \times 64 = 4096$	45815	V:12 H:14	25× 30
ibm02	19291	$80 \times 64 = 5120$	79033	V:22 H:34	40× 65
ibm03	26104	$80 \times 64 = 5120$	80193	V:20 H:30	40× 60
ibm04	31328	$96 \times 64 = 6144$	94756	V:20 H:32	40× 60
ibm05	29647	$128 \times 64 = 8192$	127509	V:42 H:63	80× 115
ibm06	34395	$128 \times 64 = 8192$	125880	V:20 H:33	40× 60

Table 4: Benchmark circuits information.

4.1 Crosstalk Violations

In Table 5, we present the numbers of crosstalk-violating nets in three solutions. Since ID+NO is not aware of an RLC crosstalk constraint during global routing stage, up to 25% of nets may have crosstalk violations.[2] Both *i*SINO and GSINO approaches apply SINO to meet the crosstalk bound. Before post-SINO refinement, there are only a few crosstalk-violating nets (less than 2%). The RF algorithm is able to completely remove the remaining crosstalk violations.

4.2 Overflow

In Table 6 we compare the numbers of overflow net segments of three solutions. It is easy to see that the RF algorithm is effective not only to eliminate the crosstalk violations (as shown in Table 5), but also to reduce the routing congestion in terms of number of overflow net segments. The overflow reduction is up to 33% for *i*SINO over *i*SINO-RF (i.e., *i*SINO without post-SINO refinement), and up to 34% for GSINO over GSINO-RF (i.e., GSINO without post-SINO refinement). In addition, since the shielding area is not estimated or properly reserved during the global routing stage, the *i*SINO approach leads to up to 98% more overflow net segments compared with ID+NO. The GSINO approach reduces the number of overflow net segments by up to 54% compared with *i*SINO. More importantly, GSINO only has slightly more overflow net segments than ID+NO.

4.3 Shielding Area

The number of shield segments used in the *i*SINO and GSINO approaches is compared in Table 7. Once again the RF algorithm is effective to reduce the number of shield segments needed to meet the crosstalk constraint by up to 25% for *i*SINO over *i*SINO-RF, and up to 22% for GSINO over GSINO-RF. On average, the number of shield segments used in *i*SINO is 14% of the total number of net segments for 30% sensitivity rate, and 19% for 50% sensitivity rate. For GSINO, the percentages are 10% and 14%, respectively. Therefore, GSINO is more area efficient than *i*SINO. As stated before, GSINO is able to distribute evenly the sensitive nets across the chip and therefore reduce the densities of sensitive nets in regions as well as the total number of shields.

[2]ID+NO may still lead to a routing solution where sensitive nets are routed adjacent to each other. In this case, neighboring wires may switch simultaneously and capacitive crosstalk exists between them. In turn, the LSK model may not be applied to compute the crosstalk. To estimate the crosstalk violations for ID+NO solutions, we ignore the capacitive crosstalk and therefore assume that the LSK model can be applied. This assumption in general *under-estimates* the crosstalk violations for ID+NO solutions.

4.4 Average Wire Length

As SINO solution does not change the wire length solution, *i*SINO has the same wire length as ID+NO. On the other hand, the GSINO approach has an average of 11% (for 30% sensitivity rate) and 15% (for 30% sensitivity rate) wire length overhead compared with *i*SINO. It is worthwhile to point out that the SINO solution has a relatively smaller delay per unit length as no neighboring wires switch simultaneously [18]. Therefore, the performance penalty due to the increase on wire length is expected to be tolerable.

4.5 Chip Area

In Table 9, we calculate the chip area by the product of maximum row and column lengths and compare the results for the three approaches. *i*SINO has a large chip area overhead compared to ID+NO, 23% on average for 30% sensitivity and 18% on average for 50% sensitivity. GSINO reduces the area overhead to 8% and 11%, respectively.

4.6 Impact of Sensitivity Rate

In Tables 5–9, we consider two sensitivity rates 30% and 50%. In case of 30%, a signal net is sensitive to random 30% of other signal nets across the chip. We observe that when the sensitivity rate decreases from 50% to 30%, GSINO results in reduced number of overflow net segments (24% on average), reduced number of shield segments (26% on average), reduced wire length (33% on average), and reduced chip area overhead (27% on average). The sensitivity rate in the real chip designs is likely to be lower than 30% and 50% used in this paper. The reduced sensitivity rate will lead to even smaller overhead in terms of the chip area and average wire length.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have shown that even when the net ordering is performed to reduce capacitive crosstalk, up to 25% of nets may still have RLC crosstalk violations at 3GHz clock. RLC crosstalk will become increasingly important as more designs will operate at giga-hertz clocks in the near future.

To satisfy RLC crosstalk constraints, we have developed an effective GSINO routing algorithm with consideration of simultaneous shield insertion and net ordering. Compared to the best alternative method (i.e., *i*SINO) we have studied, the GSINO algorithm reduces the chip area by up to 21%. More importantly, compared to the conventional routing solution that minimizes wire length and congestion but may have up to 25% RLC crosstalk violations, the GSINO solutions increase the chip area only by 10% on average, and increase the total wire length only by 14% on average.

number of crosstalk-violating nets					
	sensitivity rate = 30%				
	ID+NO	*i*SINO-RF	*i*SINO	GSINO-RF	GSINO
ibm01	1982 (15.18%)	177 (1.36%)	0	79 (0.61%)	0
ibm02	3370 (17.46%)	321 (1.61%)	0	148 (0.77%)	0
ibm03	5085 (19.47%)	365 (1.40%)	0	196 (0.75%)	0
ibm04	5392 (17.21%)	545 (1.74%)	0	302 (1.02%)	0
ibm05	4528 (15.27%)	552 (1.86%)	0	209 (0.70%)	0
ibm06	4951 (14.39%)	398 (1.16%)	0	163 (0.47%)	0
	senstivity rate = 50%				
	ID+NO	*i*SINO-RF	*i*SINO	GSINO-RF	GSINO
ibm01	2695 (20.64%)	272 (2.08%)	0	113 (0.87%)	0
ibm02	4386 (22.73%)	448 (2.32%)	0	185 (0.96%)	0
ibm03	6237 (23.89%)	663 (2.54%)	0	327 (1.25%)	0
ibm04	6201 (19.79%)	891 (2.27%)	0	346 (1.10%)	0
ibm05	7348 (24.78%)	912 (3.08%)	0	362 (1.22%)	0
ibm06	6752 (19.63%)	605 (1.76%)	0	259 (0.75%)	0

Table 5: Numbers of crosstalk-violating nets for ID+NO, *i*SINO, and GSINO solutions. *i*SINO-RF is *i*SINO without post-SINO refinement, and GSINO-RF is GSINO without post-SINO refinement. The data in the parentheses are percentages with respect to the total numbers of signal nets.

number of overflow net segments					
		senstivity = 30%			
	ID+NO	*i*SINO-RF	*i*SINO	GSINO-RF	GSINO
ibm01	576 (1.12%)	982 (1.91%)	890 (1.73%)	618 (1.05%)	465 (0.79%)
ibm02	1519 (1.08%)	2448 (1.74%)	2153 (1.53%)	2033 (1.30%)	1345 (0.86%)
ibm03	2359 (1.94%)	3637 (2.99%)	3151 (2.59%)	2227 (1.64%)	2091 (1.54%)
ibm04	2953 (1.89%)	4669 (3.01%)	4065 (2.62%)	3013 (1.75%)	2513 (1.46%)
ibm05	6453 (1.53%)	9237 (2.19%)	6706 (1.59%)	6679 (1.45%)	5205 (1.13%)
ibm06	3483 (1.61%)	4846 (2.24%)	3657 (1.69%)	3489 (1.46%)	2963 (1.24%)
		Sensitivity = 50%			
	ID+NO	*i*SINO-RF	*i*SINO	GSINO-RF	GSINO
ibm01	576 (1.12%)	1250 (2.43%)	1132 (2.20%)	759 (1.29%)	647 (1.10%)
ibm02	1519 (1.08%)	3109 (2.21%)	3011 (2.14%)	2206 (1.41%)	1877 (1.20%)
ibm03	2359 (1.94%)	4524 (3.72%)	3649 (3.00%)	2891 (2.13%)	2606 (1.92%)
ibm04	2953 (1.89%)	5475 (3.53%)	4949 (3.19%)	3460 (2.01%)	2995 (1.74%)
ibm05	6453 (1.53%)	11051 (2.62%)	8647 (2.05%)	8199 (1.78%)	6863 (1.49%)
ibm06	3483 (1.61%)	6013 (2.78%)	4781 (2.21%)	4540 (1.90%)	3848 (1.61%)

Table 6: Numbers of overflow net segments of ID+NO, *i*SINO, and GSINO solutions. The data in parentheses are percentages with respect to the total numbers of net segments. Please note that the numbers of overflow net segments of ID+NO is independent of sensitivity rate.

number of shield segments				
	sensitivity rate = 30%			
	*i*SINO-RF	*i*SINO	GSINO-RF	GSINO
ibm01	7859 (15.28%)	7021 (13.65%)	7414 (12.60%)	6308 (10.72%)
ibm02	24116 (17.14%)	21119 (15.01%)	22543 (14.41%)	18351 (11.73%)
ibm03	20179 (16.59%)	17876 (13.17%)	16397 (12.08%)	13886 (10.23%)
ibm04	28233 (18.20%)	22741 (14.66%)	25664 (14.91%)	20707 (12.03%)
ibm05	60779 (14.41%)	53781 (12.75%)	50945 (11.06%)	42286 (9.18%)
ibm06	33293 (15.39%)	28554 (13.20%)	29276 (12.25%)	22800 (9.54%)
	sensitivity rate = 50%			
	*i*SINO-RF	*i*SINO	GSINO-RF	GSINO
ibm01	14304 (24.31%)	12774 (21.72%)	10256 (17.43%)	8331 (14.16%)
ibm02	36709 (26.09%)	33023 (23.47%)	26627 (17.02%)	24295 (15.53%)
ibm03	32111 (26.40%)	28951 (23.98%)	25152 (18.53%)	20116 (14.82%)
ibm04	39076 (25.19%)	32344 (20.85%)	30554 (17.75%)	27386 (15.91%)
ibm05	93047 (22.06%)	77651 (18.41%)	70246 (15.25%)	63245 (13.73%)
ibm06	47592 (22.00%)	37251 (17.22%)	35538 (14.87%)	30974 (12.96%)

Table 7: Numbers of shield segments of *i*SINO and GSINO solutions. The data in the parentheses are percentages with respect to the total numbers of net segments.

The majority of running time in the current three-phase GSINO algorithm is consumed by the ID-based global rout-

average wire length						
		sensitivity rate = 30%)			sensitivity rate = 50%)	
	ID+NO	*i*SINO	GSINO	ID+NO	*i*SINO	GSINO
ibm01	639	639 (0%)	704 (10.17%)	639	639 (0%)	731 (14.40%)
ibm02	724	724 (0%)	805 (10.06%)	724	724 (0%)	843 (16.71%)
ibm03	647	647 (0%)	726 (12.21%)	647	647 (0%)	769 (18.85%)
ibm04	748	748 (0%)	830 (10.96%)	748	748 (0%)	881 (17.78%)
ibm05	695	695 (0%)	758 (9.06%)	695	695 (0%)	792 (13.95%)
ibm06	769	769 (0%)	841 (9.36%)	769	769 (0%)	873 (13.52%)

Table 8: Average wire lengths (μm) of ID+NO, *i*SINO, and GSINO solutions. The data in the parentheses are the average increase on wire lengths compared to ID+NO solutions.

chip area					
		sensitivity rate = 30%			
	ID+NO	*i*SINO-RF	*i*SINO	GSINO-RF	GSINO
ibm01	1533 × 1824	1678 × 2033 (22.00%)	1647 × 1975 (16.33%)	1641 × 1899 (11.14%)	1604 × 1870 (7.27%)
ibm02	3004 × 3995	3362 × 4465 (25.08%)	3301 × 4296 (18.16%)	3218 × 4210 (12.89%)	3139 × 4122 (7.82%)
ibm03	3178 × 3852	3591 × 4130 (21.15%)	3504 × 4093 (17.16%)	3352 × 4106 (12.43%)	3302 × 3995 (7.75%)
ibm04	3861 × 3910	4225 × 4387 (22.78%)	4208 × 4280 (19.30%)	4159 × 4192 (15.48%)	4159 × 3940 (8.54%)
ibm05	9837 × 7286	12087 × 7514 (26.71%)	11532 × 7456 (19.96%)	10853 × 7471 (13.12%)	10379 × 7423 (10.74%)
ibm06	5002 × 3795	5468 × 4249 (22.39%)	5396 × 4158 (18.20%)	5370 × 4212 (19.15%)	5328 × 3894 (9.30%)
		sensitivity rate = 50%			
	ID+NO	*i*SINO-RF	*i*SINO	GSINO-RF	GSINO
ibm01	1533 × 1824	1705 × 2126 (29.63%)	1697 × 2053 (24.59%)	1682 × 1936 (16.47%)	1615 × 1898 (9.62%)
ibm02	3004 × 3995	3597 × 4511 (35.20%)	3513 × 4381 (28.24%)	3307 × 4265 (17.52%)	3216 × 4147 (11.13%)
ibm03	3178 × 3852	3624 × 4255 (25.96%)	3549 × 4187 (21.38%)	3401 × 4145 (15.15%)	3365 × 4074 (11.99%)
ibm04	3861 × 3910	4309 × 4458 (27.24%)	4265 × 4313 (21.84%)	4209 × 4173 (16.34%)	4174 × 3989 (10.29%)
ibm05	9837 × 7286	12590 × 7692 (35.11%)	11794 × 7502 (23.44%)	10924 × 7502 (14.34%)	10755 × 7441 (11.65%)
ibm06	5002 × 3795	5582 × 4310 (26.73%)	5514 × 4278 (24.26%)	5516 × 4273 (24.16%)	5497 × 3923 (13.60%)

Table 9: Chip areas ($\mu m \times \mu m$) of ID+NO, *i*SINO, and GSINO solutions. The data in the parentheses are the increase on chip areas compared to ID+NO solutions.

ing phase. A more efficient global router will be developed or be integrated into the GSINO framework. Further, we plan to explore alternative solutions for crosstalk budget partitioning.

6. REFERENCES

[1] L. Pileggi, "Coping with RC(L) interconnect design headaches," in *ICCAD*, pp. 246–253, Nov. 1995.

[2] K. L. Shepard and V. Narayanan, "Noise in deep submicron digital design," in *ICCAD*, 1996.

[3] T. Gao and C. L. Liu, "Minimum crosstalk channel routing," in *ICCAD*, pp. 692–696, Nov. 1993.

[4] T. Gao and C. L. Liu, "Minimum crosstalk switchbox routing," in *ICCAD*, pp. 610–615, Nov. 1994.

[5] D. A. Kirkpatrick and A. L. Sangiovanni-Vincentelli, "Techniques for crosstalk avoidance in the physical design of high-performance digital systems," in *ICCAD*, pp. 616–619, Nov. 1994.

[6] K. Chaudhary, A. Onozawa, and E. S. Kuh, "A spacing algorithm for performance enhancement and cross-talk reduction," in *ICCAD*, 1993.

[7] S. Thakur, K.-Y. Chao, and D. F. Wong, "An optimal layer assignment algorithm for minimizing crosstalk for three layer VHV channel routing," in *ISCAS*, 1995.

[8] R. Kay and R. A. Rutenbar, "Wire packing: A strong formulation of crosstalk-aware chip-level track/layer assignment with an efficient integer programming solution," in *ISPD*, April 2000.

[9] C.-C. Chang and J. Cong, "Pseudo pin assignment with crosstalk noise control," in *ISPD*, April 2000.

[10] T. Xue and E. S. Kuh, "Post global routing crosstalk synthesis," *TCAD*, pp. 1418–1430, Dec. 1997.

[11] H. Zhou and D. F. Wong, "Global routing with crosstalk constraints," *TCAD*, pp. 1683–1688, November 1999.

[12] L. He, N. Chang, S. Lin, and O. S. Nakagawa, "An efficient inductance modeling for on-chip interconnects," in *CICC*, pp. 457–460, May 1999.

[13] L. He and K. M. Lepak, "Simultaneous shielding insertion and net ordering for capacitive and inductive coupling minimization," in *ISPD*, 2000.

[14] K. M. Lepak, I. Luwandi, and L. He, "Simultaneous shield insertion and net ordering for coupled RLC nets under explicit noise constraint," *DAC*, 2001.

[15] G. Zhong, H. Wang, C.-K. Koh, and K. Roy, "A twisted bundle layout structure for minimizing inductive coupling noise," in *ICCAD*, 2000.

[16] Y. Massoud, J. Kawa, D. MacMillen, and J. White, "Modeling and analysis of diferential signaling for minimizing inductive cross-talk," in *DAC*, 2001.

[17] S. Khatri, A. M. Mehrotra, R. K. Brayton, and A. Sangiovanno-Vincentelli, "A novel VLSI layout fabric for deep sub-micron applications," in *DAC*, 1999.

[18] J. D. Ma and L. He, "Formulae and application of interconnect estimation considering shield insertion and net ordering," in *ICCAD*, 2001.

[19] Semiconductor Industry Association, *International*

Technology Roadmap for Semiconductors, 1999.

[20] J. Cong, L. He, A. B. Kahng, D. Noice, N. Shirali, and S. H.-C. Yen, "Analysis and justification of a simple, practical 2 1/2-d capacitance extraction methodology," in *DAC*, pp. 627–632, 1997.

[21] J. D. Ma, A. Parihar, and L. He, "Pre-routing estimation of shielding for RLC signal integrity," in *ICCD*, 2001.

[22] J. Cong and B. Preas, "A new algorithm for standard cell global routing," *Integration, the VLSI Journal*, pp. 49–65, November 1992.

[23] M. Wang, X. Yang, and M. Sarrafzadeh, "DRAGON2000: Standard-cell placement tool for large industry circuits," in *ICCAD*, 2000.