True Crosstalk Aware Incremental Placement with Noise Map

Haoxing Ren
ECE Department, UT Austin
& IBM Corporation
Austin, TX 78758
haoxing@us.ibm.com

David Z. Pan
ECE Department
University of Texas at Austin
Austin, TX 78712
dpan@ece.utexas.edu

Paul G. Villarrubia
Microelectronics Division
IBM Corporation
Austin, TX 78758
pgvilla@us.ibm.com

ABSTRACT

Crosstalk noise has become an important issue as technology scales down for timing and signal integrity closure. Existing works to fix crosstalk noise are mostly done at the routing or post routing stage, which may be too late. Since placement determines the overall routing congestion, which correlates with the coupling capacitance, which in turn correlates with the crosstalk noise, placement shall be a good level to do early noise mitigation. The only existing work for the crosstalk aware placement (to our best knowledge) is [1], which uses the coupling capacitance map to guide placement. However, crosstalk is determined not only by the coupling capacitance, but also by many other factors, such as the driver resistance of the victim net and the coupling location (near source vs near sink coupling) [2]. In this paper, we introduce a novel concept of noise map which takes those factors into account. Guided by this accurate noise map explicitly, we propose an incremental placement technique to mitigate noise without disturbing the global placement order. Our incremental placement has two key steps, namely noise aware cell inflation and local refinement. Experimental results on industrial circuits show that our approach is able to reduce the number of top noise nets by 25% and improve the timing (300ps on the worst slack), with no wire length penalty or CPU overhead. Our incremental approach is also able to maintain the placement stability.

1. INTRODUCTION

Coupling capacitance between adjacent nets has become dominant in the interconnect capacitance as technology scales down [3]. The crosstalk noise introduced by coupling capacitance is a large bottleneck for design closure. Numerous techniques, such as driver sizing, wire spacing, buffer insertion and simultaneously buffering and routing [4] [5] [6] [7] have been developed to mitigate crosstalk noise, but they are mostly done at the routing or post routing stage. It might be difficult to clean up all the noise issues at the last stage of physical design. Crosstalk mitigation during the early design stage, e.g., placement, is therefore needed.

To our best knowledge, there is very little work that deals with crosstalk *explicitly* during placement. There have been a number of studies on wiring congestion reduction during placement [8] [9] [10]. Most recently, [1] attempted to mitigate noise during placement using an estimated coupling capacitance map. Wiring congestion to certain extend, determines the coupling capacitance, which in turn affects the crosstalk noise. So it is natural to alleviate congestion during placement to reduce coupling capacitance, and hopefully to reduce crosstalk. But this linkage (from congestion to coupling to noise) is not necessarily *linear*. For crosstalk noise, the coupling capacitance is just one parameter. Other parameters, such as the driver strength and the coupling location, may affect the crosstalk noise significantly [2]. A *true* crosstalk aware placement shall take into account of a more accurate crosstalk modeling.

For early crosstalk estimation and modeling, [11] introduced a probabilistic method to estimate coupling capacitance. It abstracts an RC tree and estimates noise using PRIMA [12]. This approach does consider the driver strength and wiring topology, but it is too costly to compute the noise using the PRIMA method. Moreover, the modeling error with a global router also makes such computation unnecessary during placement. The noise and congestion relationship is also hidden in the pole-zero model, which is hard to be utilized by placement. The 2-π model from [2] is an elegant analytic model which takes into consideration of many first-order parameters such as the coupling capacitance, driver and wire resistance. We will use this model and incorporate it into a concept of *crosstalk noise map* to guide placement directly.

The placement mitigation for noise shall not completely change the entire picture of the initial timing driven placement. So it is desirable to get high degree of placement stability during noise mitigation. [1] uses a partition based algorithm with whitespace insertion during the placement. This approach might change the global placement significantly [13]. In this paper, we present a true crosstalk aware, incremental placement technique. Our main contributions include:

- We introduce the concept of *crosstalk noise map*, which takes into consideration of the key parameters for crosstalk. To generate such map, we propose a more accurate coupling capacitance estimation model during placement, using real routing data and curve fitting. An interesting observation is that the coupling capacitance will saturate after certain congestion threshold.

- We show that the crosstalk map not necessarily matches the coupling capacitance map and the congestion map. Thus noise aware placement should be guided *directly* by the noise metric, not by the congestion or coupling capacitance metrics.

- Guided by this accurate noise map *explicitly*, we propose a two-step incremental placement technique to mitigate noise without disturbing the global placement order.

Experimental results on industrial circuits show that our approach is able to reduce the number of top noise nets by 25% and improve the timing (300ps on the worst slack), with no wire length penalty or CPU overhead. Our incremental approach is also able to maintain the placement stability well.

0-7803-8702-3/04/$20.00 ©2004 IEEE.

The rest of the paper is organized as follows. Section 2 describes background information on congestion estimation and noise analysis. Section 3 illustrates the relationship of the congestion map versus the coupling capacitance map, and introduces the concept of the noise map. Section 4 presents the cell inflation and local refinement incremental placement technique guided by noise map. The experimental results are shown in section 5, followed by the conclusion in section 6.

2. PRELIMINARIES

2.1 Congestion Analysis

Global router divides the chip routing areas into $m \times n$ routing grids. It assigns each net on those grids based on Steiner tree [14]. Techniques such as rip-up-and-reroute [15] or multicommodity flow [16] might be applied to reduce overflow. The output of a global router is a matrix of routing resource and routing demand for each grid edge. The routing resource is the number of available routing tracks on a grid edge, and the routing demand is the number of routing tracks required to route nets assigned to pass this grid edge. Fast global routers usually only perform two-layer assignment: vertical and horizontal. The horizontal routing resource includes available tracks of all the horizontal routing layers, and the vertical routing resource includes available tracks of all the vertical routing layers. Note that blockage and reserved wiring can be removed from routing resource, for example, grids over large SRAM or Register Array (RA) will have less routing resources. Therefore the impact of SRAM and RA on coupling capacitance should be modelled.

The congestion W_e on a grid edge e is defined as:

$$W_e = D_e / R_e \tag{1}$$

where D_e is the routing demand on edge e, R_e is the routing resource. Each grid has four edges. We use the average edge congestion as the grid congestion, i.e.,

$$W(x,y) = (W_{le}(x,y) + W_{re}(x,y) + W_{ue}(x,y) + W_{de}(x,y))/4 \tag{2}$$

where W_{le},W_{re}, W_{ue}, and W_{de} are the congestion of the left, right, up and down edge of grid (x,y) accordingly.

2.2 Noise Estimation using 2-π Model

Given an RC network, there are various methods to estimate the receiver noise [17] [18] [19] [20] [21] [22]. We use the 2-π model because it gives simple yet accurate close form solution for both peak noise and noise width [2]. Suppose there is a victim net with an aggressor net shown in Figure 1:

According to the 2-π model [2], the peak noise and the noise pulse width can be estimated by the following formulae:

$$v_{max} = \frac{t_x}{t_r}(1 - e^{-t_r/t_v}) \tag{3}$$

$$t_{width} = t_r + t_v ln\left[\frac{1 - e^{-2t_r/t_v}}{1 - e^{t_r/t_v}}\right] \tag{4}$$

where t_x is the Elmore delay of the coupling capacitance with the near source wire, t_r is the transition time of the aggressor net, and t_v is the Elmore delay of the victim net.

In practice, both peak noise and pulse width are used to determine if a receiver will fail. From [2], the peak noise amplitude and noise width product (AW) at the receiver can be written as:

$$AW = (R_d + R_s)C_x f(x) \tag{5}$$

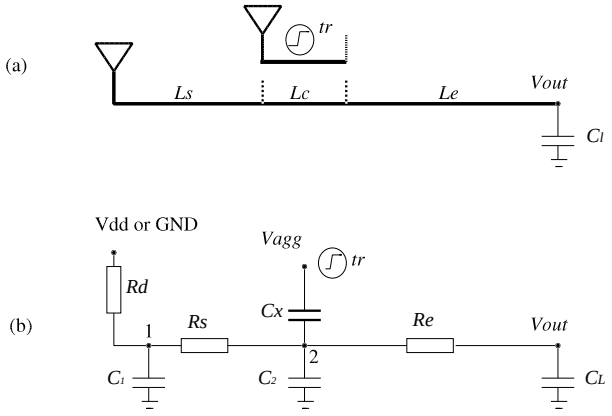

Figure 1: (a) The layout of a victim net and an aggressor above it. (b) The 2-π crosstalk noise model

where R_d is the drive resistance, R_s is the near source wire resistance (i.e., from driver to the coupling segment) of the sink net, C_x is the coupling capacitance, $f(x) = \frac{e^x - e^{-x}}{1 - e^{-x}} ln \frac{e^x - e^{-x}}{1 - e^{-x}}$ and $x = t_r/t_v$. It can be shown that $f(x) \in [ln2, 1]$. Thus to the first order, we can estimate AW as:

$$AW \approx (R_d + R_s)C_x \tag{6}$$

The above simple noise metric captures the two most important parameters that determine the crosstalk noise. The first parameter $(R_d + R_s)$ is the total resistance (including both gate and wire) from the driver to the coupling location. It captures how strong a driver is to bring the noise back to the steady state, as well as the coupling location such as near source (smaller R_s) versus near sink coupling (larger R_s). The second parameter C_x is the coupling capacitance, as expected.

3. CROSSTALK NOISE MAP WITH COUPLING CAPACITANCE MODELING

Crosstalk noise is correlated with coupling capacitance, and coupling capacitance in turn is correlated with routing congestion. Intuitively, a placement with less congestion should have less noise. Therefore, given an existing placement, reducing the congestion on the hot spots will result in a placement with better noise margin [1]. This requirement will raise the question of how to find those regions where reducing congestions shall lead to the largest noise reduction. The most congested regions are not necessary those regions according to the AW formula. Although the coupling capacitance in those hot spots are higher than other regions, the noise on each sink also depends on the wire topology and driver resistances. A further noise estimation based on wire topology, driver resistance and coupling capacitance is needed in this case. In this section, we will first illustrate our coupling capacitance estimation method, then we will introduce the concept of noise map, which is more accurate than either congestion or coupling capacitance metric to explicitly guide the noise aware placement.

3.1 Coupling Capacitance Estimation with Curve Fitting

Previous studies have shown that we can use the congestion map produced by a global router to estimate the coupling capacitance. To accurately estimate the coupling capacitance on a single net is not possible at this stage, simply because there are multiple possible local routing solution. However, it is reasonable to estimate

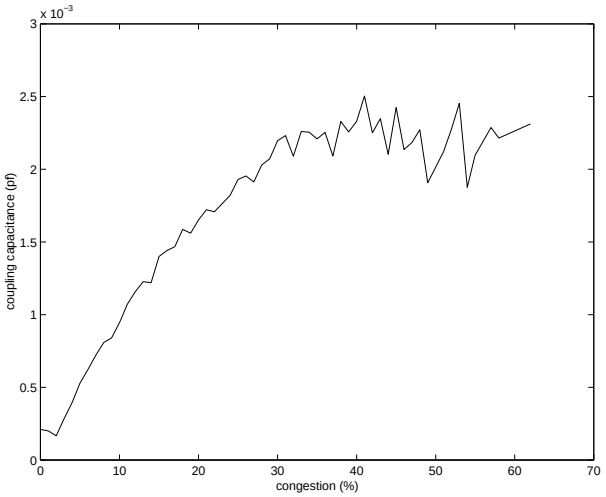

Figure 2: Average net coupling capacitance verse grid congestion for ckt1

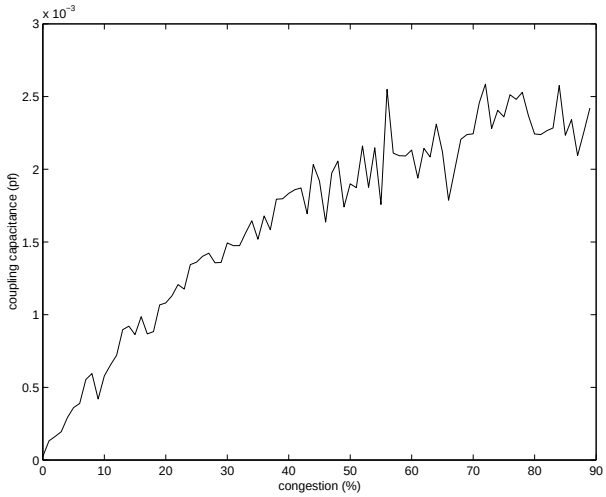

Figure 3: Average net coupling capacitance verse grid congestion for ckt2

the average coupling capacitance for all the nets in a global routing grid. We assume there is a relation between global routing congestion and the coupling capacitance. This relation can be find by by curve fitting. This approach can be applied to any routing and silicon technology. The detail of this approach will be given in the rest of this section. Previously, [1] used a model to estimate coupling capacitance based on unit wire length capacitance for each layers. [11] proposed a probabilistic model which considers all the possible wire assignments inside a grid. Although these models can all give reasonable coupling capacitance estimation from congestion, they are not generalized to fit different global routers and silicon technologies.

We first run global router to produce a global congestion map $W(x,y)$ described in section 2. Following the global router, local router will be run. Then we run extraction to generate coupling capacitance for each wire segment. We distribute the coupling capacitance on each wire segment to the global routing grid based on the following equation:

$$C_g(i,j) = C_n(i,j)/N_w(i,j) \tag{7}$$

where $C_n(i,j)$ is the coupling capacitance between wire segments i and j, $N_w(i,j)$ is the number of global routing grids where i and j overlaps, $C_g(i,j)$ is the coupling capacitance distributed to each one of those routing grids. Summing up the coupling capacitance for all the nets passing a grid, we can get the total coupling capacitance on each grid:

$$C_t(x,y) = \sum_{(i,j)\in E(x,y)} C_g(i,j) \tag{8}$$

where $C_t(x,y)$ is the total coupling capacitance of grid (x,y), $E(x,y)$ is the set of coupling edges that cover grid (x,y). The average net coupling capacitance of a grid $C_a(x,y)$ can be computed with:

$$C_a(x,y) = C_t(x,y)/N_g(x,y) \tag{9}$$

where $N_g(x,y)$ is the total number of net covering grid (x,y).

Using the congestion map $W(x,y)$ and the average net coupling capacitance map $C_a(x,y)$, we can obtain the correlation between congestion and coupling capacitance. We have run two industry circuits ckt1 and ckt2 and generated the correlation pictures in Figures 2 and 3.

From the figures, we can see that for some designs, when congestion is large enough, the average coupling capacitance does not increase accordingly. For example, the coupling capacitance of ckt1 does not increase linearly when congestion is over 40 percent. On the contrary, it even decrease a little when congestion is over 50 percent. We speculate that it is because local router does not match well with the congestion analysis of global router in some regions with higher routing congestion. Intuitively, our result shows that those nets on global grids with congestion more than certain threshold will have almost the same coupling capacitance. By curve fitting, we can use following equation to approximate this relationship:

$$C_a(W) = \alpha(1 - e^{-\beta W}) \tag{10}$$

where $C_a(W)$ is the average coupling capacitance for nets in a grid with congestion W, α and β are extracted constant for each technology. Using this formula, we can estimate coupling capacitance for nets on each grid (x,y) when we have the congestion map:

$$C_a(x,y) = C_a(W(x,y)) \tag{11}$$

3.2 Noise Map Generation

We need to identify areas where the improvement in congestion would contribute most on noise, i.e., the noise amplitude-width product(AW) of all sinks. As Equation (6) shows, AW depends on coupling capacitance, driver resistance and wire resistance. The coupling capacitance map only has part of information needed for noise estimation. And since coupling capacitance does not increase when congestion is high as shown in section 3.1, only using coupling capacitance to guide placement will not be effective. Therefore, we introduce a new concept noise map to capture the complete noise information and use it to guide placement.

The noise of each grid $N(x,y)$ is defined as the total contribution of all the crosstalk within the grid to the AW of all nets. Here we only count the AW of one sink for each net. To compute $N(x,y)$, we need to first analyze the AW contribution of net i in a grid. From Equation (6), we have

$$AW(i) = (R_d(i) + R_s(i))C_x(i) \tag{12}$$

where $R_d(i)$ is driver output resistance of net i, $R_s(i)$ is the upstream wire resistance, and $C_x(i)$ is the estimate coupling capacitance of

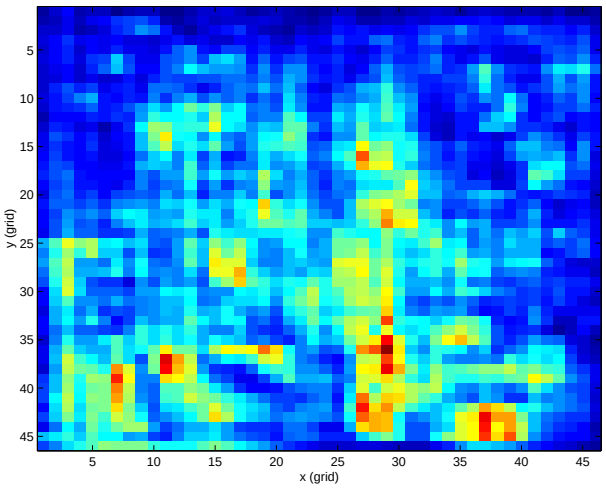

Figure 4: Congestion map for ckt1

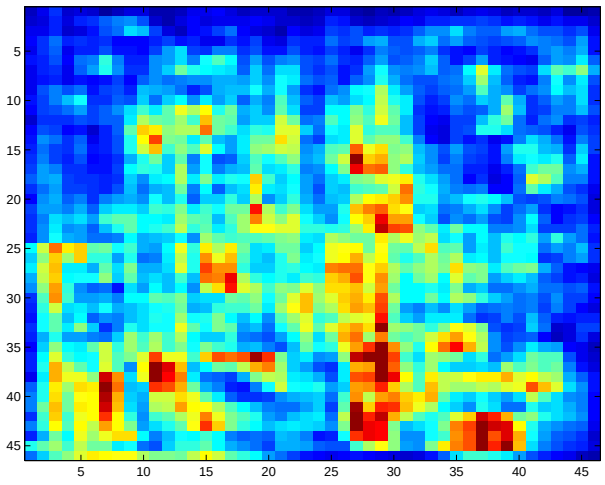

Figure 5: Coupling capacitance map for ckt1

net i. R_s can be computed as $r_w L$, where r_w is the unit wire resistance given by technology parameters, L is the distance of the source and the current grid. We use a single average capacitance to estimate all the coupling to any net inside a grid. Using that model, if net i is in grid (x,y), the $C_x(i)$ is just $C_a(x,y)$ computed in the previous section. The noise $N(x,y)$ of grid (x,y) can be computed as:

$$N(x,y) = \sum_{i \in (x,y)} AW(i) \tag{13}$$

It adds up all the AWs of nets passing grid (x,y). Note that although we do not use timing window in the noise analysis because the aggressor and victim net pairs can not be identified at this step, we believe it could still estimate the total noise in a region to the first order.

Figures 4, 5 and 6 compare the congestion map $W(x,y)$, the coupling capacitance map $C_a(x,y)$ and the noise map $N(x,y)$ for ckt1. The red areas have higher value than the blue areas. We can see that the coupling capacitance map is similar to the congestion map except that the hotspots are even bigger than those of the congestion map. This is because the coupling capacitance does not increase linearly with congestion when congestion is higher than certain threshold as we described in section 3.1. The regions with highest noise value are almost a subset of regions with higher coupling capacitance. This is because coupling capacitance is a big differentiator in Equation (12). Higher coupling capacitance will usually cause bigger noise. However, if the nets in those regions have shorter wire length or strong drivers, their noise value tend to be smaller. That is why some areas with higher coupling capacitance are not highly noisy regions.

4. INCREMENTAL PLACEMENT FOR NOISE REDUCTION

After identifying the noise regions with noise map, we will perform incremental placement migration to reduce the congestion of those regions in order to reduce the coupling capacitance and noise in turn [1]. The constraint is that we should not perturb the existing placement too much to change the timing characteristics from run to run. We propose a two-step incremental placement mitigation technique to reduce noise while preserving the global placement order. The first step is to inflate cells in the noise regions. It is similar to the placement density control methods used in [23] [1]. The difference is that we do the spreading as a post-placement incremental process instead of spreading cells during the global placement. This will help to preserve the placement order. The second step uses local refinement technique to improve the local congestion. This step will further improve the congestion in the noise region via the recursive min-cut algorithm. We will describe these two steps and the placement flow in following subsections.

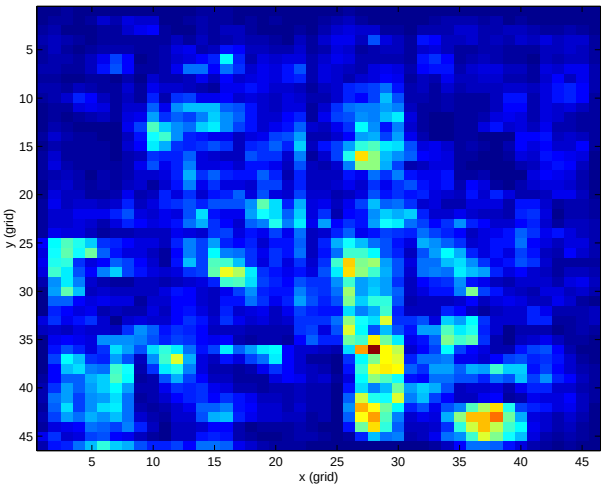

Figure 6: Noise map for ckt1

4.1 Noise Aware Cell Inflation

A popular way to reduce congestion in hot spots is to allocate whitespace [10] [8] [23] in the congested regions. This technique can be used for noise mitigation as well. One can either insert pseudo cells into those regions or inflate cells in those regions. Typical placement engines [24] [9] [25] recursively bisect or quadrisect the chip area into smaller bins. We refer each bisection or quadrisection as a cut. The bin size decreases as the cut number increases. Usually whitespace is allocated between cuts [8], we propose to incrementally add whitespace to the existing placement, then rerun placement starting from a late cut to remove overlaps. This approach can help preserve the placement stability to the existing

placement, and have more accurate estimation for congestion and noise.

Since we start from an existing placement, we use the cell inflation instead of pseudo cell insertion to add white space into congested regions. We inflate cells based on the noise map and pin density. The total whitespace inserted into a bin is proportional to the noise of this bin over the average noise.

$$\Delta A(x,y) = \begin{cases} (\frac{N(x,y)}{N_{avg}} - 1)A(x,y) & N(x,y) > N_{avg} \\ 0 & N(x,y) \leq N_{avg} \end{cases} \quad (14)$$

where $A(x,y)$ is the total cell area in bin (x,y), $\Delta A(x,y)$ is the total cell inflation area in bin (x,y), and N_{avg} is the average noise across the chip. If the noise value is equal to or less than the average noise, no cell inflation will happen. To not overflow the chip area, we also compare the total inflation against the available chip area and scale the inflation for each bin to be less than certain ratio of the total available area.

Inside a bin, we scale the inflation on each cell based on its pin density. The pin density $d(i)$ of a cell i is defined as the ratio of pin numbers $k(i)$ over the cell area $a(i)$:

$$d(i) = k(i)/a(i) \quad (15)$$

The inflation area $\Delta a(i)$ for each cell i can be computed by following equation:

$$\Delta a(i) = \frac{d(i)}{\sum_{k\in(x,y)} d(k)} \Delta A(x,y) \quad (16)$$

After the cell inflation, we start a new placement from the N_1 cut. The placement will follow a mixed quadratic placement and partition flow [24] [23]. We also run greedy detailed placement algorithms, i.e. cell swapping, to further improve the wire length after the recursive min-cut. To prevent the detailed placement from disturbing global placement order, we set a maximal movement limit for each cell.

This placement step is essentially an overlap removal procedure that removes the overlaps caused by inflated cells. The merit of this method is that it can produce a legal placement while minimizing total wire length. By starting from a late cut, the global relative order of each cells are kept unchanged as well.

Figure 7 is the placement of ckt1 before cell inflation. Figure 8 is the placement after cell inflation and replacement. Cells that are inflated are marked with red color. We can see the relative position of the noise area are kept. Note that the cell inflation is only used for placement step, the real size of any cell is not changed, therefore the total area of the chip would keep the same.

4.2 Local Refinement

The cell inflation technique given in section 4.1 will spread cells out from noise region, however since we did not use the real size for cells in the noise region, the local relative order for cells in those region might not be optimized for congestion. The objective of this step is to only optimize for those regions using the original cell size while keeping the placement order of other regions. We use a simple but effective way to achieve this. We fix the placement of cells in other regions, which means we do not allow movement of those cells in this step. Since the noise regions are usually just a small fraction of the chip area, this method will only replace a small number of cells while fix the majority of cells. This way will guarantee to keep the overall order of existing placement. In the regions where we want to reduce noise, we will run recursive min-cut algorithm. Similar to the previous step, we do not start recursive min-cut algorithm from the first cut, instead, we start it from a late cut N_2. This will prevent cells from moving far away

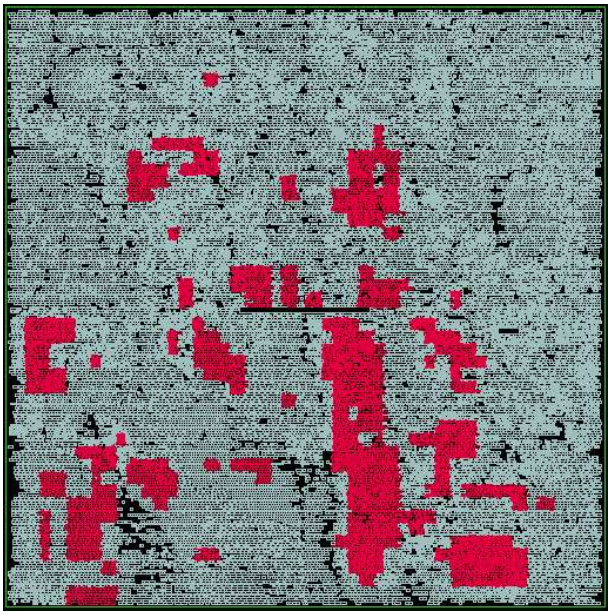

Figure 7: Placement of ckt1 before cell inflation

from its original location. N_2 is chosen based on the ratio of the size of the noise region and the chip size. Figure 9 shows the placement of ckt1 after the local refinement.

The recursive min-cut algorithm will reduce cut crossing, which is equal to reduce congestion and coupling capacitance on the cut lines. [9] has shown that recursive min-cut works well on producing congestion-free placements. Besides of keeping the global stability, another effect of fixing other cells is that we can actually spread the unfixed cells. Suppose we only leave cells in a highly noisy region movable and cells elsewhere fixed, the recursive min-cut algorithm will evenly distribute cells in that region to the entire chip area if it starts from the first cut. If we start from cut N_2, cells will be evenly spread to the entire grid resulting from N_2 cut. Our experimental result shows that this technique alone can significantly reduce peak function noise. However, if we start it from an early cut, which has large bin size, cell would end up far away from their original positions. On the other hand, if we start from a later cut number and bin size is small, it will spread less and might not reduce congestion. Therefore, we need to first spread out those cells before running local refinement. That is the other reason why we use the post-placement cell inflation technique and local refinement approach together. The cell inflation technique will inflate cells, and local refinement will shuffle the local areas in the expanded regions to improve wire length.

4.3 Incremental Placement Flow

Using the cell inflation introduced in section 4.1, we will achieve the spreading in noisy regions, then run the local refinement technique describe in section 4.2 to further reduce noise at the hot spots. Putting them all together, we propose an incremental placement flow illustrated by Algorithm 1:

The runtime overhead for this placement flow includes global routing, noise map generation, post placement cell inflation and local refinement. The runtime of noise map generation is linear to the number of grids. Since the placements for both cell inflation and local refinement all start from late placement cuts, and we only make a small fraction of the cells movable during local refinement, they run relatively fast compared to the global placement.

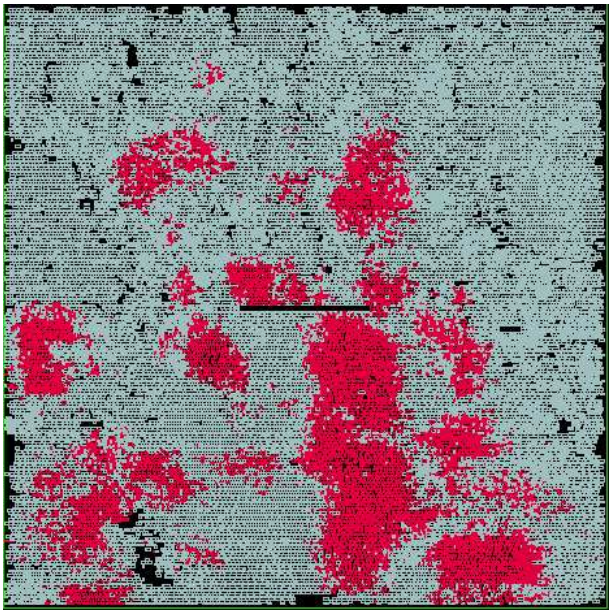

Figure 8: Placement of ckt1 after cell inflation

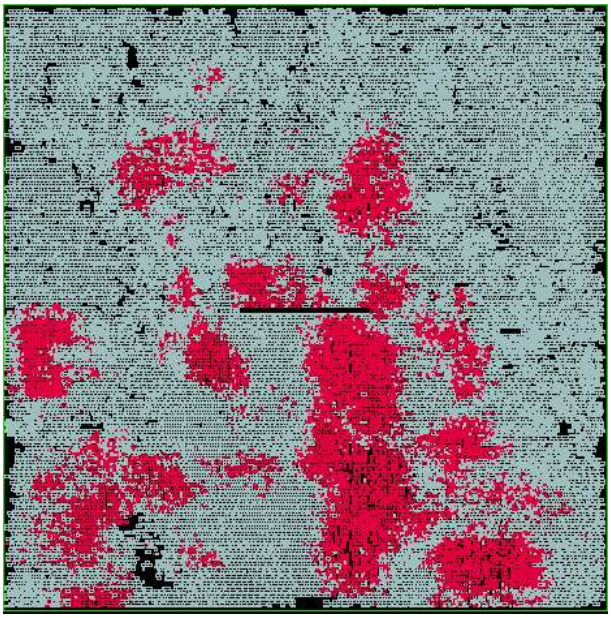

Figure 9: Placement of ckt1 after local refinement

Algorithm 1 Re-placement flow for noise reduction

1: Starting from an existing placement
2: Generate congestion map $W(x,y)$ from global router
3: Generate coupling capacitance map $C_a(x,y)$ from congestion map $W(x,y)$ using Equation (10)
4: Generate noise map $N(x,y)$ from coupling capacitance map $C_a(x,y)$, cell output resistance and wire length estimation from global router using Equations (12) and (13)
5: Post placement cell inflation (section 4.1)
6: Local Refinement in top noisy regions(section 4.2)

Detailed placement also runs very fast. Therefore, using a faster global router, our flow will not increase the runtime significantly.

Note that although in this paper, we focus on the incremental placement to mitigate crosstalk noise, the same concept of the crosstalk map can be used to spread the cells during global placement too. Those flows adjust the placement density in between each placement cuts based on congestion estimation [24] [8]. Instead of using the congestion estimation, one can use the noise estimation proposed in section 3.2 to guide placement density control. In theory, the noise estimation map can be generated with any level of congestion estimation.

5. EXPERIMENTAL RESULTS

The overall placement and noise analysis baseline flow is given by Algorithms 2 (BF).

Algorithm 2 Baseline flow (BF) for placement and noise analysis

1: Synthesized netlist
2: Placement and physical synthesis
3: Routing
4: Extraction
5: Noise and delay analysis

Our proposed flow (PF) using noise map is shown in Algorithm 3. We will also run a flow CF similar to PF, which uses the congestion map to guide incremental placement instead of using the noise map.

Algorithm 3 Proposed flow (PF) for placement and noise analysis)

1: Synthesized netlist
2: Initial placement and physical synthesis
3: Incremental placement for noise reduction
4: Routing
5: Extraction
6: Noise and delay analysis

The algorithm is implemented in C++ and tested on the IBM AIX 43P-S85 servers. The placement tool used in this flow is the IBM CPlace [26]. CPlace has several placement engines. In our experiment, we uses the quadratic placement engine called QPS in the initial placement and a recursive min-cut engine ACG [23] in the replacement for noise reduction. The physical synthesis used in this flow is IBM Placement Driven Synthesis (PDS). We use IBM Xrouter to perform global and local routing, IBM ChipEdit to do RC extraction, IBM 3Dnoise to perform noise analysis and IBM Einstimer to do timing analysis with coupling noise.

We will compare the final noise and delay of PF against the baseline BF which does not run incremental placement. We will also compare PF and CF to evaluate the effectiveness of noise reduction using the noise map vs. using congestion map. We run those three flows on two industry designs ckt1 and ckt2. The characteristic of these two designs are given in Table 1.

Table 1: Testcase Size and Technology

Design	cells	nets	technology	metal layers
ckt1	68563	66966	0.13um	4
ckt2	76733	71077	0.13um	6

We run global and detailed routing, parasitic extraction, noise analysis and timing report at the end of each flow. Table 2 gives the noise distributions for the final result of flow BF, CF and PF. These

results are given by IBM 3Dnoise and Einstimer. The noise slack in Table 2 indicates how much noise slack a sink has before fail. Small slack means large noise at sinks. Not all the net sinks are listed in the table. Those net sinks with noise smaller than a threshold are ignored. It shows that the proposed flow PF has the best noise reduction among all the three flows. On average PF reduces the number of sinks with top noise slack (0-0.4) by 25%.

Table 2: Noise distribution for net sinks

Noise slack	ckt1			ckt2		
	BF	*CF*	*PF*	*BF*	*CF*	*PF*
0-0.1	13	16	11	0	2	0
0.1-0.2	27	12	9	13	10	10
0.2-0.3	78	74	67	30	31	25
0.3-0.4	92	89	70	49	41	34
total	210	201	157	92	85	69

Table 3 gives the final worst negative slack with coupling noise for each flow. We can see PF also improves the circuit performance better than CF does.

Table 3: Worst negative slack with coupling noise

Design	BF (ns)	CF (ns)	PF (ns)
ckt1	-3.495	-3.425	-3.161
ckt2	-1.612	-1.519	-1.453

The estimated congestion and noise distribution for the final result of flow BF, CF and PF are shown in figure 10 and 11 for ckt1. All the estimated noise and congestion values are based on grids. We can see that the top noise reduction of PF over BF is about 30%, while CF over BF is only 5%. Note that although CF also improves the congestion of BF, the noise reduction is not as large as that of PF. It is clear that using noise map to guide placement can produce better noise reduction.

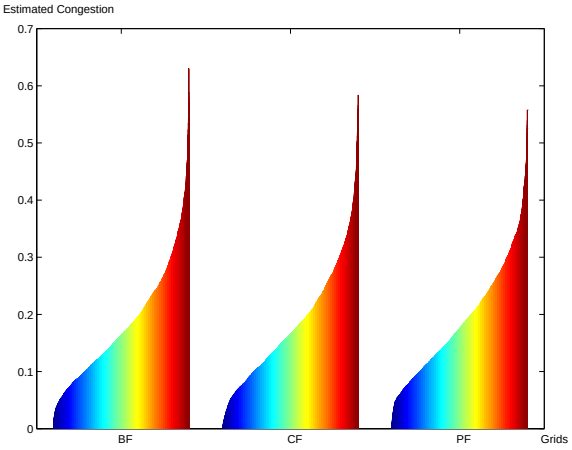

Figure 10: Estimated congestion distribution for ckt1

The runtime comparison of flow PF and BF is shown in Table 4. PF has three more CPU consuming steps than BF, i.e. global routing, cell inflation and local refinement. The noise map generation step is negligible. For ckt1, PF spends more CPU time than BF, however, for ckt2, PF spends less CPU time. This is because the local router actually runs faster when there is less congestion. Therefore, PF may use less total amount of CPU (e.g., ckt2).

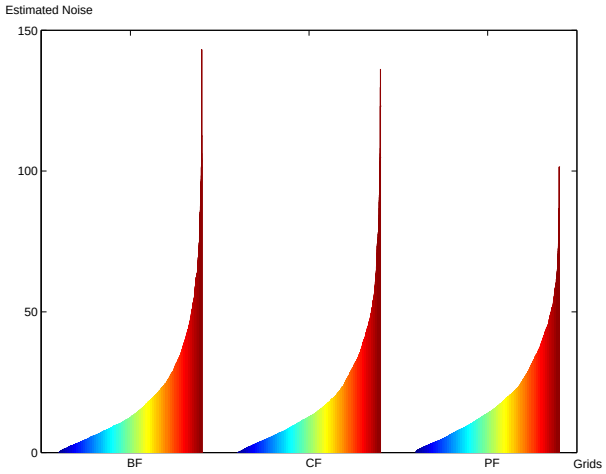

Figure 11: Estimated noise distribution for ckt1

Table 4: CPU time comparison

Design	ckt1		ckt2	
Flow	BF	PF	BF	PF
global routing	n/a	98.5	n/a	102.4
cell inflation	n/a	390	n/a	540
local refinement	n/a	62	n/a	49
final routing	2810.5	2730.6	7163.3	5298.1
subtotal	2810.5	3281.1	7163.3	5989.5

To verify the placement stability, we also compare the total wire length (TWL) of the flow BF, CF and PF in Table 5. We can see the TWL differences among these three flows are negligible.

Table 5: Total wire length (TWL) comparison

Design	ckt1			ckt2		
Flow	BF	CF	PF	BF	CF	PF
TWL ($\times 10^6$)	10.9	10.8	10.9	10.8	10.7	10.7

6. CONCLUSIONS

In this paper, we introduce a novel noise map to model crosstalk noise and then use it to guide a two-step incremental placement to mitigate noise. We show that the noise map is not necessarily the same as the congestion or coupling capacitance map. Guided by this more accurate metric, we propose the cell inflation and local refinement to reduce crosstalk noise. It can achieve 25% top noise reduction and timing improvement while keeping the placement stability. We show much better results compared to that from the congestion or coupling guided incremental placement.

7. ACKNOWLEDGMENT

The authors would like to thank Eric Foreman, Shaodi Gao, Haihua Su, William Livingstone, and Chuck Alpert at IBM Corporation for their helpful discussions.

8. REFERENCES

[1] J. Lou and W. Chen, "Crosstalk-aware placement," *IEEE Design & Test of Computers*, pp. 24–32, Jan. 2004.

[2] J. Cong, D. Z. Pan, and P. V. Srinivas, "Improved crosstalk modeling for noise constrained interconnect optimization," in *Proc. Asia and South Pacific Design Automation Conf.*, Jan. 2001.

[3] Semiconductor Industry Association, *International Technology Roadmap for Semiconductors*, 2003, http://public.itrs.net/.

[4] I. H. R. Jiang, Y. W. Chang, and J. Y. Jou, "Crosstalk driven interconnect optimization by simultaneous gate and wire sizing," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 999–1010, Sept. 2000.

[5] T. Xiao and M. Marek-Sadowska, "Gate sizing to eliminate crosstalk induced timing violation," in *Proc. IEEE Int. Conf. on Computer Design*, pp. 186–191, 2001.

[6] C. J. Alpert, A. Devgan, and S. T. Quay, "Buffer insertion for noise and delay optimization," in *Proc. Design Automation Conf.*, pp. 362–367, 1998.

[7] C.-P. Chen and N. Menezes, "Noise-aware repeater insertion and wire sizing for on-chp interconnect using hierarchical moment-matching," in *Proc. Design Automation Conf.*, pp. 502–506, June 1999.

[8] U. Brenner and A. Rohe, "An effective congestion driven placement framework," in *Proc. Int. Symp. on Physical Design*, pp. 6–11, 2002.

[9] A. E. Caldwell, A. B. Kahng, and I. L.Markov, "Can recursive bisection alone produce routable, placements?," in *Proc. Design Automation Conf.*, pp. 477–482, 2000.

[10] X. Yang, B.-K. Choi, and M. Sarrafzadeh, "Routability-driven white space allocation for fixed-die standard-cell placement," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 410–419, Apr. 2003.

[11] M. R. Becer, D. Blaauw, R. Panda, and I. N. Hajj, "Early probabilistic noise estimation for capacitively coupled interconnects," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 337–345, Mar. 2003.

[12] A. Odabasioglu, M. Celik, and L. T. Pileggi, "Prima: passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 645–653, Aug. 1998.

[13] S. N. Adya, I. Markov, and P. Villarrubia, "On whitespace and stability in mixed-size placement and physical synthesis," in *Proc. Int. Conf. on Computer Aided Design*, pp. 311–318, 2003.

[14] C. Chiang, C. K. Wong, and M. Sarrafzadeh, "A weighted steiner tree-based global router with simultaneous length and density minimization," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1461–1469, Dec. 1994.

[15] B. S. Ting and B. N. Tien, "Routing techniques for gate array," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 301–312, Oct. 1983.

[16] R. C. C. IV, J. Li, and C. K. Cheng, "A global router with a theoretical bound on the optimal solution," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 208–216, Feb. 1996.

[17] T. Sakurai, "Closed-form expressions for interconnection delay, coupling, and crosstalk in VLSIs," *IEEE Trans. on Electron Devices*, vol. 40, pp. 118–124, 1993.

[18] A. Vittal and M. Marek-Sadowska, "Crosstalk reduction for VLSI," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, pp. 290–98, 1997.

[19] H. Kawaguchi and T. Sakurai, "delay and noise formulas for capacitively coupled distributed RC lines," in *Proc. Asia and South Pacific Design Automation Conf.*, pp. 35–43, Jan. 1998.

[20] A. Devgan, "Efficient coupled noise estimation for on-chip interconnects," in *Proc. Int. Conf. on Computer Aided Design*, pp. 147–153, Nov. 1997.

[21] A. Vittal, L. Chen, M. Marek-Sadowska, K.-P. Wang, and S. Yang, "Crosstalk in VLSI interconnections," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 2, pp. 1817–24, 1999.

[22] A. B. Kahng, S. Muddu, and D. Vidhani, "Noise and delay uncertainty studies for coupled rc interconnects," in *IEEE International ASIC/SOC Conference*, pp. 3–8, 1999.

[23] C. J. Alpert, G.-J. Nam, and P. G. Villarrubia, "Effective free space management for cut-based placement via analytical constraint generation," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1343–1353, Oct. 2003.

[24] H. Eisenmann and F. M. Johannes, "Generic global placement and floorplanning," in *Proc. Design Automation Conf.*, pp. 269–274, 1998.

[25] M. Wang, X. Yang, and M. Sarrafzadeh, "Dragon2000: Standard-cell placement tool for large industry circuits," in *Proc. Int. Conf. on Computer Aided Design*, pp. 260–263, 2000.

[26] P. Villarrubia, G. Nusbaum, R. Masleid, and P. T. Patel, "IBM RISC chip design methodology," in *Proc. IEEE Int. Conf. on Computer Design*, pp. 143–147, 1989.