Shield Count Minimization in Congested Regions

Prashant Saxena
Intel Corporation
2111 NE 25th Ave
Hillsboro OR 97124 USA
prashant.saxena@intel.com

Satyanarayan Gupta
Intel Corporation
17 Mahatma Gandhi Road
Bangalore 560001 India
satyanarayan.gupta@intel.com

ABSTRACT
With worsening crosstalk in nanometer designs, it is becoming increasingly important to control the switching cross-coupling experienced by critical wires. This is commonly done by adding shields adjacent to these wires. However, the number of wires requiring shields in high frequency designs becomes extremely large, resulting in a large area impact. We address this problem at both the methodological and algorithmic levels in this paper, integrating the traditionally separate steps of power and signal routing in a safe manner to minimize the number of shields required to satisfy all shielding constraints. We postpone the power routing in middle metal layers to *after* critical signal nets and their shields have been laid out (with maximal shield sharing), and then try to construct a fine-grained power grid out of the already routed shields. Given a routing on a metal layer, our adaptive power routing algorithm adds provably fewest new power lines to complete the power grid on that layer. Our approach has proven highly effective while designing some high frequency blocks of a commercial gigahertz range microprocessor.

Categories and Subject Descriptors
B.7.2 [**Integrated Circuits**]: Design Aids – *layout, placement and routing.*

General Terms
Algorithms, Performance, Design.

Keywords
Layout, Routing, Power routing, Noise, Crosstalk, Shielding, High performance design, Domino circuits.

1. INTRODUCTION
Process scaling usually results in an increase in interconnect noise caused due to switching cross-capacitance that makes it increasingly harder to design correct circuits [3]. Furthermore, switching cross-capacitance increases the difficulty of converging high performance circuits by widening the transition windows of signals to account for the crosstalk-on-delay effects due to unpredictable transition states of neighboring signals [2]. In view of this, designers attempt to minimize switching cross-capacitance by shielding sensitive signals using power (i.e., Vdd or Vss) nets. However, with the importance of interconnect switching cross-capacitance in high frequency designs growing with each process generation, the proportion of signals requiring shielding (and consequently, the area used up by shields) is also growing. Since design blocks usually end up being wire-limited in their layout, any decrease in the number of required shields can translate directly into an area saving on silicon. Furthermore, detailed shielding requirements are available only at late stages in the design process, at which time the silicon real estate available to lay out each converged design block may already have been frozen. In such a scenario, if the layout of a converged design block cannot be carried out in its planned area, it can cause extensive delays due to negotiations with and redesign of surrounding design blocks.

The shielding approach proposed in this paper optimizes layout area by integrating two traditionally disjoint phases of the layout process, viz., power routing and signal routing. We postpone the completion of the power routing to the signal routing phase, at which time signal shielding requirements are also used to complete the power routing (in addition to the usual power delivery integrity constraints). Along with this adaptive power routing, we rely on the aggressive use of shield sharing optimizations. In spite of its simplicity, our approach proves surprisingly effective in congested layout regions in which a large proportion of nets are susceptible to crosstalk. Furthermore, given a routing of critical nets and their shields on a metal layer, our adaptive power routing is optimal in the number of new power lines that it adds to complete the power grid on that layer.

As described in Section 3.3, adaptive power routing can result in mismatches between the power grids of adjacent design blocks. Therefore, it cannot be used on the upper metal layers where long range matching of the power networks is a stringent requirement. However, it works very well on intermediate metal layers such as M3 and M4 where the power networks of adjacent design blocks do not have to match up exactly with each other (since they are densely connected with one another through the upper layers)[1]. However, while perturbing the power grid, the adaptive power routing ensures that no region wider than the designated power pitch is left without any power lines; the power pitch is the maximum separation between successive power lines that

ISPD'02, April 7-10, 2002, San Diego, California, USA.

[1] Simulations show that even if the power grids of adjacent blocks are perfectly matched on M3 and M4, a large fraction of any current between two points in these layers in the power grids of different blocks actually flows through the power grids on the upper layers due to their lower resistivity.

guarantees acceptable IR drop, inductive noise[2] and reliability. Furthermore, the power pitch also enables the analysis of the power grid even if all the power lines have not yet been finalized (by modeling the grid on a layer by virtual power lines uniformly separated by the power pitch; shields added later only improve the power grid characteristics yet further). This methodology works best on layers with fine-grained grids.

In the remainder of this paper, we first outline traditional approaches to power and sensitive signal routing in Section 2, describing how previous industrial and academic approaches to the interconnect crosstalk problem differ from the approach that we have adopted. Then, in Section 3, we present methodological, algorithmic and some implementation details about our approach to shield count minimization. Finally, we detail the experimental evaluation of our work in Section 4, followed by some concluding remarks in Section 5. The work described in this paper was used in the design of a portion of a commercial gigahertz range microprocessor that taped out in 1999.

2. PRIOR WORK

Traditional physical design flows first lay out the power routing (along with other special signals such as clocks), and then route the signal nets. When a noise-sensitive signal net is being routed, it is explicitly shielded on one or both sides (as required) using a Vdd/Vss shield. If such a power line is already present in the neighborhood of the signal net's proposed routing, the signal may be routed next to it. Otherwise, a new power line is explicitly added solely for the purpose of shielding the signal. Little or no attempt is made to exploit the flexibility in routing of the signal nets in order to minimize the number of shields required. In any case, the power routing done earlier is not disturbed. This impacts the layout density adversely in two ways. Firstly, the number of shield wires that are needed in order to satisfy the shielding requirements on all noise-sensitive signal nets using a greedy shielding scheme can be substantially larger than the number of shields that suffices under an intelligent shield-sharing regime. Secondly, since the power lines are fixed during signal net routing, they may not get optimally used for shielding if signal nets requiring shielding cannot be routed in the tracks adjacent to these lines. Furthermore, prerouted power lines can result in wasted space in gridless routing environments when performance and noise constraints cause different signal nets to be sized to different widths (that may be non-integral multiples of the minimum wire width). This wastage occurs because the space available on one side of the power line may be insufficient to fit a sized signal track. Even if there is unused space available on the other side of the power line, the power line cannot be shifted in traditional routing methodologies to allow the sized signal track to be laid out. We are not aware of any published work or unpublished practice that performs adaptive non-uniform power routing or explicit shield sharing optimization (although [9], discussed in the next paragraph, looked at a few similar ideas).

Over the past few years, there have been several papers that have attempted to optimize routing to combat on-chip crosstalk (see, for instance, [4]-[6],[8],[10]-[13]). Routing fabrics such as the Dense Wiring Fabric outlined in [7] (in which every alternate track is reserved for a power line) eliminate capacitive crosstalk at the cost of considerable routing area. Recently, there has also been some work [9] that deals with integrated shielding, net ordering and power grid design. This work studies shielding structures under uniform power pitches, and under limited perturbations of the power grid. But, in contrast to our approach, this work too begins with a prerouted power grid and limits the maximum perturbation of any power line to a small, pre-determined number (say, k)[3] of tracks. However, in practice, power grid perturbation is a binary property – either the power lines in two adjacent design blocks must match exactly on any given layer, or they must remain disconnected from each other on that layer (since adding single layer doglegs to connect power lines offset from each other creates blockages that worsen the routability of signal lines on that layer tremendously). Therefore, the cost of offsetting two corresponding power lines in adjacent design blocks relative to each other is independent of the size of the offset. Our approach exploits this property to create the maximum flexibility for shield count minimization, in contrast to restricting the perturbation search space as in [9]. This increases the likelihood that the power lines in the grid double up as shields for sensitive nets, decreasing the overall count of Vdd/Vss tracks.

3. SHIELD COUNT MINIMIZATION

As mentioned earlier, our approach (overviewed in Figure 1) consists of an adaptive power routing algorithm and a shield sharing optimization algorithm. Unlike traditional routing flows that complete the power routing before doing any signal routing, we postpone detailed power routing to later in the layout flow, integrating it adaptively into the subsequent signal routing step as described below. Thus, the tracks that would have been used up for the power delivery network are left available for critical signal routing, in the hope that the shields required by these signals will take care of much of the power delivery requirements. Of course, if any region ends up with locally insufficient shields subsequent to the critical signal routing, we insert additional power lines there as part of the adaptive power routing algorithm in order to maintain power delivery integrity. This approach works best on layers having fine-grained power grids with narrow, closely spaced power lines, so that minimum power-width[4] shields can seamlessly be used for power delivery.

3.1 Shield Sharing Optimization

Prior to layout, the timing and noise optimization of the circuit performs the sizing of devices and wires as well as generates shielding requirements for each net (i.e., decides whether the net requires no shielding or shielding on one/both sides). These shielding requirements are used to drive our shield sharing optimization algorithm as follows. Critical signal nets are ordered by the degree of freedom in their placement[5] and by the number of shields required by them, and then routed gridlessly with the most

[2] A fine-grained power grid ensures that every switching signal has a Vdd and a Vss line close by, thus guaranteeing that a large number of current return paths form small inductive loops.

[3] As a matter of fact, [9] recommends limiting k to 1.

[4] Note that the minimum width of a power line is usually greater than that of a generic signal line.

[5] The net placement constraints can arise from diverse sources such as the bounding box of a driver and its critical sinks, or the need to avoid the RC penalty of vias close to the driver(s).

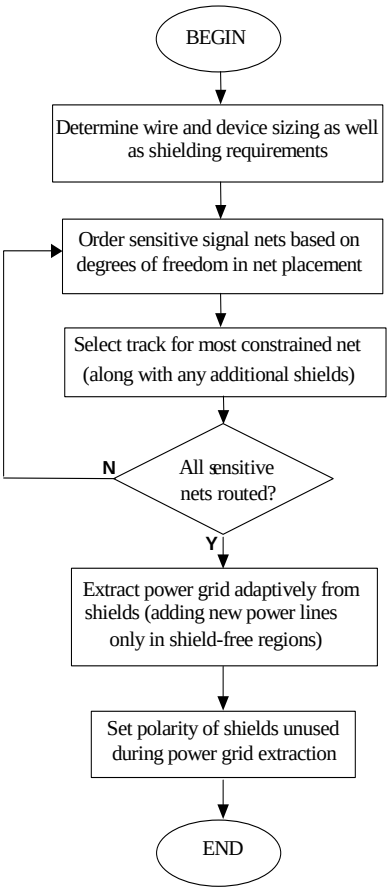

Figure 1. An overview of our approach to shield count minimization.

constrained nets being routed first (along with their associated shields). Each individual net is placed so as to reuse previously routed shields (including any prerouted power lines) as much as possible. This ordering of the nets, which is dynamically updated during the routing process, ensures a high degree of shield sharing between the signal nets requiring shielding, thus minimizing the number of new shields that need to be inserted. At this stage, the polarity of any newly inserted shields is left unassigned.

This high level description of shield sharing optimization can be customized to the specific constraints of the design methodology within which it is to be applied. We used shield count minimization within the framework of a domino synthesis system, where it was complicated by the presence of several prerouted nets. Furthermore, the library and routing templates were such that each major trunk at the output of a cell had to be routed on one of the tracks available over its driving cell (and, in the case of multiple parallel drivers, within the cumulative span of its drivers[6]). This created strong constraints on the flexibility in the placement of each net. We classified all the sensitive nets into a small number of buckets based on the width of their placement ranges. This was followed by the slicing of the entire layout into parallel tiles, each of width equal to the width of the smallest bucket. We then processed these tiles starting from one end of the layout and moving across to the other end. While processing a tile, we routed the nets belonging to the current width bucket that were to be placed within that tile. From this set of nets, the doubly shielded nets were placed first, followed by the singly shielded nets. Of the nets available for placement at any stage, we selected a net that maximized sharing with any shields already present in the current tile and adjacent tiles (breaking ties by the number of half-shielded tracks available in the tile after the placement of that net). Once all the tiles in the layout had been processed, we re-sliced the layout again with tiles of width equal to the width of the next larger bucket, in order to place the nets belonging to this bucket using the procedure described above. This was repeated until all the bucket widths had been processed. Although we tried several heuristics, we obtained the best results with the above approach that bucketed the nets based on their placement flexibility and then placed the most constrained nets first. Our bucketing heuristic worked well because the distribution of the placement ranges of the nets was clustered around a small number of distinct peaks (corresponding to the bucket widths), thus allowing some decorrelation between the ordering of the nets based on full/half/no shielding and that based on placement spans.

[6] Placing a trunk outside the span of its driver(s) creates a RC bottleneck because of the resistance of the vias required to shift routing tracks at the output of the driver(s); this resistance sees the downstream capacitance of the entire net and all its sinks.

3.2 Hierarchical Routing Revisited

In contrast to traditional hierarchical routing in which global routing results in the assignment of the nets to a sequence of global routing cells (GRCs) without doing any exact routing (or even track assignment) within those cells, we do an exact routing for all the trunks of the net. Although the routes are brought close to their sinks, no attempt is made to actually connect the nets to the sinks. This is motivated by our observation that the majority of detailed routing effort is spent on the connection of the nets to the pins; in contrast, the exact routing of the major trunks requires comparatively little processing time. The traditional hierarchical paradigm sufficed in the pre-nanometer era when routing abstracted to the granularity of GRCs yielded good estimates of congestion and net-length based delay. However, the increasing impact of crosstalk on performance [2][3] has made net adjacencies important (for crosstalk to be predicted) at the global routing level. This motivates a new global/detailed routing abstraction in which the major trunks of the net are routed exactly during "global" routing, but no effort is spent on the compute-intensive activity of actually hooking up the net to its pins; instead, the trunks are just brought to the neighborhood of the sink cells, with the actual hookups being performed during "detailed" routing. This abstraction enables a good prediction not only of net length and congestion but also of delay (inclusive of the impact of coupling) and signal integrity effects. The shield sharing optimization heuristics described in Section 3.1 dovetail naturally into this abstraction by focusing on the exact placement of trunks and shields, thus yielding the adjacencies among and exact spacings between trunks and their neighbors. This allows a high degree of predictability in the final noise and delay in the interconnects at the block-level global routing stage itself without excessive runtime penalty.

3.3 Adaptive Power Routing

The shield sharing optimization phase, during which all the crosstalk-sensitive nets are routed, is followed by the adaptive

power routing phase. During this phase, we ensure that the design is supplied with adequate power. In other words, we ensure that the design has alternating Vdd and Vss tracks separated by no more than the designated power pitch (say, *P*). However, before assigning a Vdd or Vss line to one among a target set of tracks, we try to reuse any pre-existing shield in that region, setting its polarity as required. If this fails, we search for any shields lying in the region between the previous power line and the desired track(s) for the new power line. If it exists, the shield lying in this region that is farthest from the previous power line is now treated as part of the power grid and assigned to a polarity opposite to that of the previous power line. (Thus, the separation between this reused shield and the previous power line ends up being smaller than the power pitch *P* in this case. We now attempt to route our next power line in a track approximately *P* units beyond this reused shield.) A new power line is explicitly added only if we are unable to find any shield in the entire region between the previous power line and the track one power pitch beyond it. This minimizes the number of inserted power lines while still guaranteeing power delivery integrity, thus decreasing the routing congestion and area significantly. Furthermore, even in the case where an additional power line must be inserted, our algorithm allows better routing completion by allowing the power line to be assigned to any unused track over the entire region (of width *P*). In contrast, traditional (non-adaptive) power routing requires that the power line be added to a region just one or two tracks wide.

Finally, all the shields in the design that have not yet had their polarities assigned (i.e., are not an essential part of the power delivery network) are arbitrarily assigned to Vss or Vdd.

The operation of the adaptive power routing algorithm is illustrated in Figure 2. We start from the lower boundary of the design block and search for a shield within the first *P/2* units from the boundary, starting from *a* and searching back towards the boundary. Let *a'* be the first shield found in this process (thus, it is the shield in the desired region that is farthest from the boundary). We now locate the track *b* that is *P* units beyond *a'* (and not *a*), and look for an existing shield starting from *b* and moving back towards *a'*. Let *b'* be the shield found as a result of this search. If there exist no shields in the region [*b',c*] (i.e. the tracks in the *P* units beyond *b'*), we explicitly insert a power line *c'* in the first available track starting from *c* and searching backwards towards *b'*. This iterative process continues until the upper boundary of the design block has been reached.

Theorem: *If there exists a way of inserting power lines to complete the power grid (with a given power pitch) on a metal layer on which some nets and their associated shields have been routed, the adaptive power routing algorithm does the power grid completion with the fewest additional power lines possible.*

Proof: Consider a layout in which additional power lines have been added in an optimal fashion. We shall use induction to show that the i^{th} newly added power line inserted during adaptive power routing cannot be at a smaller coordinate than the i^{th} newly added power line in the optimal solution (with the origin lying at the bottom of the layout and the newly added power lines ordered by increasing coordinate). Therefore, the number of newly added power lines required to cover the entire design block with a valid power grid in the adaptive power routing case will be no greater than in the case of the optimal layout. For the base case of the induction, consider, in the optimal layout, the newly added power line with the lowest coordinate. Let the track containing this power line be denoted by *t*, and the track *P* units below it (if it exists) be denoted by *t'*. (See Figure 3.) Then, there must exist a prior shield in at least one of the tracks in the region *[t', t)* (else, there is a violation of the power pitch). Of all these shields, let the one in the track closest to *t* (and at a coordinate less than that of *t*) lie in track *u*. (Note that it is possible for *u* and *t'* to refer to the same track). In the adaptive power routing algorithm, a search for the "next" power line that begins in the interval *[0, t')* will not add a new power line (since there is, by definition of the base case, no shield-free area wider than *P* in that region). A search for the next power line that begins in the interval *[t', u)* (if *t'* is distinct from u) will also not add a new power line, since it will not proceed to any coordinate lower than that of *u* before finding a shield to reuse. The only remaining case is that of a search beginning in the interval *[u, t)*. Let *v* denote the track *P* units beyond *u*. It follows that the coordinate of *v* is no less than that of *t*. Since the adaptive power routing algorithm tries to add a new power line to the first available track below *v* (after it has failed to find a suitable shield to reuse), it will certainly go no further than track *t* (that is currently vacant, since it contains a *newly added* shield in the optimal layout) to add a new power line, thus proving the base case. The same argument holds for the induction step also, with the only difference being that the tracks in the *P* units just below the current newly added power line in the optimal layout (say, in track *t*) might contain another newly added power line (say, in track *w*). However, in this case, the induction hypothesis guarantees the existence of a newly added power line (say, in track *w'*) in the adaptively power routed layout (corresponding to the power line added in the optimal solution in *w*) that lies in a track with coordinate no less than that of *w*, so that the induction argument goes through as before if *w'* < *t*. (Note that if *w'* >= *t*, the induction hypothesis is trivially true). This proves our induction hypothesis, and thus the optimality of adaptive power routing in the number of newly added power lines. ❑

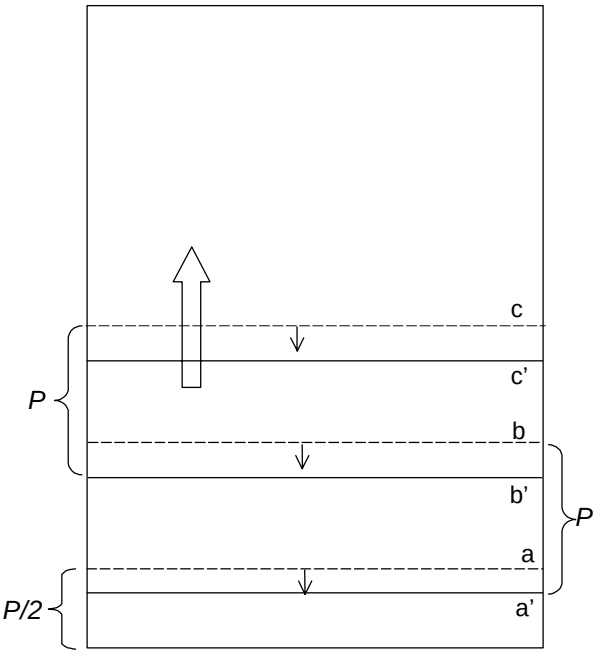

Figure 2. Adaptive power routing.

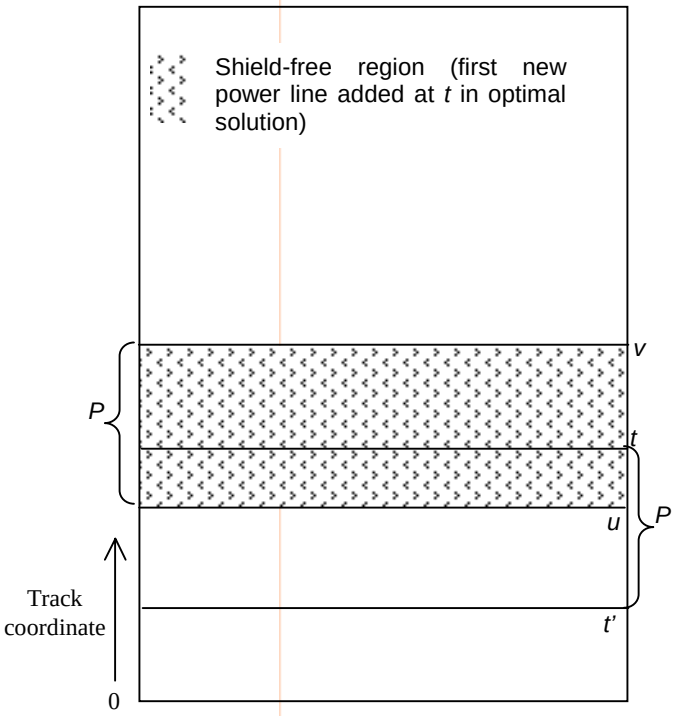

Figure 3. The base case of the induction.

Note that pathologically congested layouts may not allow any new power lines in some shield-free regions. But this is highly unlikely since our approach completes the power grid after *only the critical nets* have been routed (and critical nets are also usually the ones that require shielding). So, if all tracks have been used up, it is likely that some of them contain shields that can be reused for the power grid. Thus, if there exist large shield-free regions that are highly congested, the problem is probably because of poor placement resulting in an unfeasible routing problem, or because of the false characterization of a large number of nets as critical.

If p denotes the minimum routing pitch, the number of tracks that need to be inspected while identifying the "next" power line (either among existing shields or by explicit addition) is at most P/p. Therefore, if the total number of tracks in the design block is T, the complexity of adaptive power routing is bounded above by $O(TP/p)$. Thus, since $P/$p is a predetermined constant, the algorithm is linear in the number of tracks in the block.

4. DESIGN RESULTS

The shield count minimization techniques described in this paper were implemented as a part of the backend of a domino synthesis package. They were successfully applied to several testcases as well as to the taped-out design of a critical domino block, all taken from a leading edge commercial microprocessor operating in the gigahertz range. The techniques demonstrated custom-quality routing and shielding in each case.

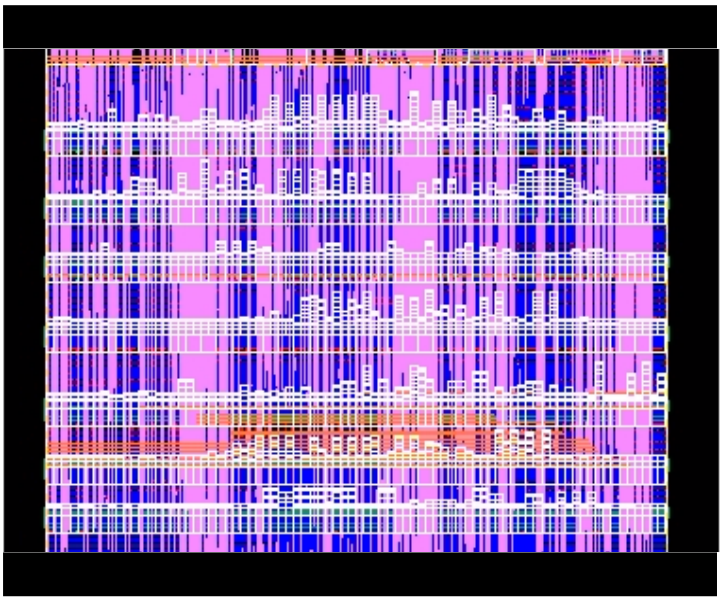

Figure 4. Layout of a high frequency microprocessor logic block produced using adaptive power routing and shield sharing.

The layout of the taped-out design block mentioned above is depicted in Figure 4. The block consists of 373 complex domino clusters with high average fanins and fanouts (with average fanout[7] of 3.5 and maximum fanout of 36, excluding clocks) and 122 I/O pins. The size of the block is approximately 0.45 mm x 0.36 mm in a 0.18µ process technology. The horizontal direction in the figure corresponds to the metal layers M2 and M4, while the vertical direction corresponds to M3. Metal M1 has no preferred routing direction. The data flow direction is roughly along M3, although there is little structural regularity. The routing for this block was completed in M1, M2 and M3, with very limited use of M4. The shield count minimization techniques discussed in this paper were applied to the interconnect trunks routed in M3. The block contained 363 block-level (signal) nets to be routed, in addition to 38 prerouted clock and signal nets and 30 prerouted power lines on M3. Owing to the single M3 trunk topology adopted for routing in this block (in order to decrease the RC bottlenecks at vias), each signal net required no more than a single M3 track. The M3 track structure was constructed on the fly (with variable track widths) based on the sizing constraints on the various nets routed on M3. In the final layout, the M3 layout map had 109 tracks that required no shielding, 43 tracks that required half shielding (i.e. a Vdd/Vss line in at least one of its two adjacent tracks), and 62 tracks that required shielding on both sides. In addition, the placement flexibility for 63 M3 trunks was only 6µ wide, while that for 13 other M3 trunks was only 12µ

[7] In comparison, static CMOS circuits usually have average fanout less than 2, resulting in much lesser wiring complexity per cell in comparison to domino logic. For instance, fanout data for a regression suite of 15 industry testcases reported in [1] indicates that, on the average, 64.07% of the nets in each design have one sink, whereas another 19.66% have two sinks.

wide. The remaining signals had greater (although bounded) flexibility in the placement of their M3 trunks. The layout was heavily wire-limited on M3.

The shielding constraints were satisfied by the insertion of only 73 new shields. In contrast, a Dense Wiring Fabric (DWF) routing structure [7] would have required 185 new shields, whereas a greedy shielding algorithm (similar to those in state-of-the-art commercial routers) could have required as many as 140-160 new shields. Furthermore, the adaptive power routing algorithm was able to reuse 23 shields to construct the power grid on M3, needing to insert only 4 additional lines explicitly for power delivery. This is in contrast to the approximately 25 or so lines that would have been required for an *a priori* grid construction. The DWF approach respects the explicit shielding and power pitch constraints faced by our design[8]; however, compared to DWF, our approach represents a saving of 25.175% in total track count on the this block – a saving that is "real" as shown by the valid operation of taped-out silicon. The runtime overhead for our shield sharing and adaptive power routing algorithms on the routing flow was insignificant. Prior to the application of our approach, it had not been possible to fit all the shields required for the converged design block within its planned area using state-of-the-art routing techniques. It has been estimated by the technical leads for the relevant portion of the chip containing this design block that its manual redesign and layout would have significantly impacted the tapeout date of the entire microprocessor project. In contrast, our approach allowed us to rapidly generate a layout that obeyed all shielding requirements and power delivery integrity constraints and yet fit into its planned area. The final signoff on the block was done using a combination of in-house and vendor tools for performance, noise and reliability analysis. Subsequent to tape-out of the microprocessor, valid operation without limiting frequency has been measured on high volume product and at clock frequencies >2GHz.

5. CONCLUDING REMARKS

In this paper, we have addressed the problem of routing in highly congested regions containing a large number of crosstalk sensitive signals. We have proposed the use of sophisticated shield-sharing heuristics as well as the postponement of the completion of a fine-grained power grid on intermediate metal layers to after the critical signals and their associated shields have been routed. At that stage, we propose the extraction of the power grid from the shields routed earlier, thus obviating the need to add power lines exclusively for power delivery in many parts of the layout. We have also proven that our adaptive power routing algorithm can do power grid completion optimally. We have demonstrated the effectiveness of our techniques by presenting design results on a taped-out domino block from a commercial microprocessor. We feel that these techniques are a natural fit for many wire-dominated layouts where a large proportion of wires are delay critical or susceptible to noise failures – a class that includes the full-chip and inter-block hierarchies of an increasingly large fraction of all static CMOS layouts also.

[8] A direct comparison of our approach with [9] is difficult because of the numerous net placement, preroute and explicit shielding constraints in our design. In order to handle these, the algorithms of [9] would have to be heavily constrained and would therefore lose much of their potential benefits.

6. ACKNOWLEDGMENTS

We would like to thank Barbara Chappell who was the technical lead for the domino synthesis project that included the work described here; we benefited much from her willingness to share her vast experience and expertise. We would also like to acknowledge the support provided by Hien Le and Wilfred Gomes during the layout of the circuit block described in Section 4, as well as the valuable feedback provided by the reviewers of this paper.

7. REFERENCES

[1] Boese, K.D., Kahng, A.B., and Mantik, S. On the Relevance of Wire Load Models. SLIP '01, 91-97.

[2] Cooke, L.H., Goossens, M., Hoxey, P., Inoue, T., Overhauser, D., Saxena, P., and Singh, R. Signal Integrity Effects in System-on-Chip Designs – A Designer's Perspective, in Signal Integrity Effects in Custom IC and ASIC Designs. Singh, R. (ed.). Wiley-Interscience & IEEE Press, 2001, 1-11.

[3] Gal, L. On-chip Crosstalk – The New Signal Integrity Challenge. CICC '95, 12.1.1-12.1.4.

[4] Gao, T., and Liu, C.L. Minimum Crosstalk Switchbox Routing. Integration VLSI Journal, 1995, 161-180.

[5] Jiang, I.H.R., Pan, S.R., Chang, Y.W., and Jou, J.Y. Optimal Reliable Crosstalk-driven Interconnect Optimization. ISPD '00, 128-133.

[6] Kay, R., and Rutenbar, R.A. Wire Packing: A Strong Formulation of Crosstalk-aware Chip-level Track/Layer Assignment with an Efficient Integer Programming Solution. ISPD '00, 61-68.

[7] Khatri, S.P., Mehrotra, A., Brayton, R.K., Sangiovanni-Vincentelli, A., and Otten, R.H.J.M. A Novel VLSI Layout Fabric for Deep Submicron Applications. DAC '99, 491-496.

[8] Kirkpatrick, D.A., and Sangiovanni-Vincentelli, A.L. Techniques for Crosstalk Avoidance in the Physical Design of High Performance Digital Systems. ICCAD '94,. 616-619.

[9] Ma, J.D.Z., and He, L. Formulae and Applications of Interconnect Estimation Considering Shield Insertion and Net Ordering. ICCAD '01, 327-332.

[10] Saxena, P., and Liu, C.L. A Postprocessing Algorithm for Crosstalk-driven Wire Perturbation. IEEE Trans. CAD, 2000, 691-702.

[11] Tseng, H.P., Scheffer, L., and Sechen, C. Timing and Crosstalk-driven Area Routing. DAC '98, 378-381.

[12] Vittal, A., and Marek-Sadowska, M. Crosstalk Reduction for VLSI. IEEE Trans CAD, 1997, 290-298.

[13] Zhou, H., and Wong, D.F. Global Routing with Crosstalk Constraints. DAC '98, 374-377.