Design Topology Aware Physical Metrics for Placement Analysis

Shyam Ramji and Nagu R. Dhanwada
IBM Microelectronics,
Hopewell Junction, NY.
{ramji,nagu}@us.ibm.com

ABSTRACT
Traditionally placement evaluation metrics have been based on wirelength and congestion measures and are independent of the logic network topology. However, the actual timing measure, which is used in a design closure loop, is path-based and dependent on the network topology. In this paper, we propose a design-topology aware metric that encapsulates the structural property of the circuit and physical goodness of the given placement. We present such a metric which is based on path monotonicity and an efficient method to compute this measure for a given placement. This method involves abstract path generation, clustering based region refinement and physical monotonicity analysis. Experimental results on real industry designs, using a commercial strength design closure flow, establish the usefulness of this metric in predicting the quality of a given placement with respect to design timing closure.

Categories and Subject Descriptors
D.3.3 [**CAD**]: Physical Design, Placement, Floorplan.

General Terms: Algorithms, Design, Experimentation.

Keywords: Placement, Wirelength, Path-Monotonicity, Timing Analysis.

1. INTRODUCTION
Typical stages in an ASIC design flow include logic synthesis, floorplanning, place & route and timing analysis steps. At each step in the design process different metrics are used to evaluate the quality of the design state. For instance, at the logic synthesis stage, several metrics like literal count, cell count, area etc have been used to measure the quality of a logic network. Similarly at the physical design stage, metrics like wirelength, peak and average congestion, and cut size have been used to predict the quality of a placed netlist. Integrated flows that address timing closure use timing analysis at different levels to predict the design performance.

Recent works [3][10] attempt to measure the physical goodness of logic network during synthesis stage and thereby define metrics (in the logic network space) that capture physical information (wirelength, congestion) at an early stage. In the physical design space the tools tend to assume a logic network[1] independent view of the design. The logic network topology aware measure at the physical design stage is the timing information, which is more an implicit way of accounting for the logic network structure. Moreover, most timing-driven physical design techniques (placement, floorplanning) tend to interact with the timing analysis engine to correct the initial solution that is generally obtained through optimization of physical metrics (which refer to wirelength and congestion estimates). While this *feedback* mechanism to correct the current solution and search the physical design space toward achieving timing closure is computationally efficient, most algorithms to correct timing violations do not have a global view of the solution space.

The placement phase in physical design has been well studied for over two decades and it still remains a challenge with the growing circuit size and the dominance of interconnect delay in deep-submicron design. Increasing focus on integrated design flows for timing closure, has led to a renewed interest in timing driven placement. Existing timing-driven placement approaches can be broadly classified as: path-based, net-based and delay budgeting based. The net-based algorithms tend to control the path delays through weighting the critical nets [6] while delay budgeting schemes apply a delay bound on the nets [9]. Path-based schemes [7] tend to model the timing information accurately but have an exponential complexity which does not scale very well. Therefore, most recent works are net-based [8][9] where timing analysis is performed to identify the critical paths and weight the critical nets, while the core placement engine works on weighted net length minimization. The placement phase itself is modeled as a minimization problem with the objective function being any of wirelength, squared wirelength, cutsize, or a combination of those. The essence of these timing-driven placement techniques is to reflect the path-based timing information to the net-based model that most analytical, iterative (simulated annealing, genetic algorithms etc), and partitioning based placement algorithms adopt. While this alleviates the complexity of explicitly modeling the structural information through paths, it also does not capture the "design dependent" nature of the circuit topology accurately. Furthermore, existing physical metrics to measure the quality of placement use variations of wirelength estimation (bounding box or Steiner route lengths) and estimated wiring congestion. Even without additional timing-driven constraints, [2] rightly point out that generating optimal placements in terms of wirelength measure while assuring routability is by itself a complex problem.

Motivated by these issues, [4] propose a top-down partitioning based placement with analytic techniques for timing optimization.

GLSVLSI'03, April 28-29, 2003, Washington, DC, USA.

[1] In this paper, *network topology or structural information* refers to physical paths between specific gates in a logic network along with fan-in fan-out logic cones and shared sub-paths.

This is based on the proven performance of most partitioning based placers in terms of wirelength, congestion and practical runtimes [1]. In a recent work, [5] attempt to address the limitations of modeling timing criticality in most net-based partitioning techniques by proposing a multi-objective minimization function which allows a mix of path-based and net-based optimization. Though path-based or circuit topology aware placement algorithms is the right direction for timing-driven placement, most of these approaches ignore the potential for circuit optimization or re-synthesis toward timing-closure. While timing-analysis would identify the top *K* critical paths, rectifying the timing behavior could come from re-synthesis/timing correction as well as re-placement [11]. From this perspective, most timing-driven placement algorithms tend to over-constrain the placement problem. So, in the context of physical synthesis, it is imperative that the quality of a given global placement might need to be judged without explicit timing information while still incorporating both structural (path-embedding) and physical (wirelength and congestion) measures. In this work, we motivate the need for integrating the structural view of a design along with other physical metrics in arriving at a good solution during the physical design stages of floorplanning and placement. We present such a metric based on path monotonicity that evaluates the quality of a given placement in relation to its logic structure, and validate this experimentally within a timing closure flow.

The rest of this paper is organized as follows: section 2 presents the motivation for this work in the context of a physical synthesis flow. Section 3 presents our proposed approach for computing a topology aware metric. Experimental results on real industry designs are presented in section 4. Section 5 concludes the paper and discusses future work.

2. MOTIVATION

Figure 1 shows a typical physical design flow for timing closure. The physical design phase starts with an initial floorplanning step that fixes the location of the macros and the IO cells in the design. The detailed placement of the leaf-cells is then accomplished using a placement tool. The solution is evaluated using typical physical metrics like wirelength and congestion, and iteratively refined to generate a final floorplan along with initial placement of leaf cells. The selected solution from this iterative process is taken through the timing closure steps which attempt to optimize both the placement and the logic network to meet circuit timing constraints. Most known algorithms for floorplanning and detailed placement attempt to optimize some variation of wirelength for ease of modeling it as a constraint. The wirelength measure only reflects the global goodness of the solution but does not directly indicate the timing quality of the solution. Timing analysis on the placed netlist is the ultimate measure that guarantees timing closure. Therefore there is a need for a path-based non-explicit timing measure which captures the design topology and spatial distribution (placement) of the cells. Selecting a floorplan and initial placement that is to be optimized by the physical synthesis phase based on wirelength measure alone may not be adequate.

We conjecture the need for design topology awareness in any physical metric that will reflect the quality of the current placement solution. Significance of this approach would be to arrive at a *topology aware* initial solution in the physical design space, which has an *average goodness* of timing paths that can be further optimized through explicit timing driven methods to converge faster towards timing closure. Also, during early stages of the physical design flow, for instance the floorplanning phase, where the complete timing information (timing budgets) may not be available, incorporating a flavor of the structural information along with other physical metrics (like wirelength and congestion) would result in a good initial floorplan that can be further improved through timing correction techniques.

The focus of this work is in developing such a topology aware metric for analyzing a given placement and to establish its correlation to explicit timing through experiments. The metric provides the ability to distinguish solutions based on structure awareness that were conventionally judged only using wirelength measure. We present a general scheme to analyze the placement quality using this metric and an application of this in arriving at a good initial solution (floorplan and placement) for a physical synthesis tool. As detailed in the following sections, we also present schemes where the model for placement analysis is generated only once for a given design netlist, but the iterative floorplan/placement changes are analyzed incrementally to reflect the quality of physical embedding of the logic network.

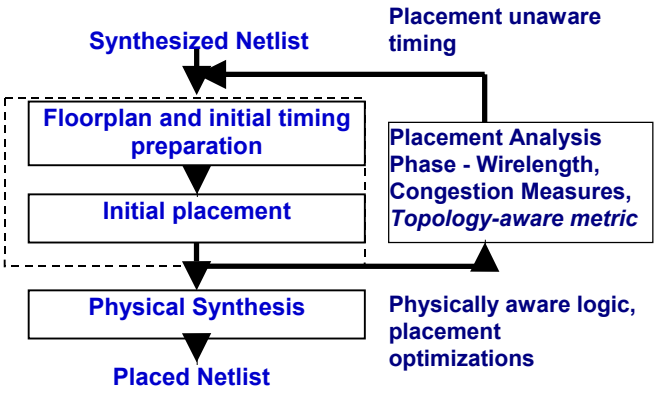

Figure 1: Physical Design Flow for Timing Closure.

3. PHYSICAL METRIC COMPUTATION

In this section we present the overall approach with the sub-steps in our placement analysis scheme. Figure 2 shows the main steps in the proposed metric computation scheme. This scheme involves computing monotonicity [3] of physical paths, which can be viewed as a topological ordering of cells based on their signal paths. The basic idea comes from the fact that the best timing solution would be achieved if all paths were strictly monotonic [3]. Defining monotonicity (straightness) of logic path embeddings is a non-trivial task on its own. Furthermore, capturing the spatial distribution of leaf-cell placement in conjunction with physical path monotonicty compounds the problem. In the proposed approach, we overcome this problem by capturing path information in the logic network within a static abstract model and by super-imposing placement information onto this abstract model for path monotonicity analysis.

From the input design netlist, an abstract/virtual model of the netlist is constructed, which is referred to as *netlist abstraction*. The model consists of "abstract paths" without the detailed view of intermediate logic between certain *marked* objects of interest (detailed in section 3.1) in the original netlist. By marking objects like latches and large macros in the netlist, the logical boundary for abstract paths is identified. Each abstract path (defined by the logical boundary of marked objects) has a corresponding physical

region based on the placement of marked objects. The structural placement analysis is conducted on the physical region associated with each abstract path.

The path information is then refined based on shared logic detection and a clustering approach to minimize the number of distinct abstract paths, redefining the physical regions within which analysis will be done. The steps of abstraction and region refinement divide a given layout into analysis regions. The monotonicity of physical paths belonging to each analysis region is used to define the metric for a given placement solution. Figure 3 shows a sample logic network and the abstract paths for the given marked objects L1, L2, L3 and L4. The physical regions corresponding to the abstract paths are shown in Figure 4.

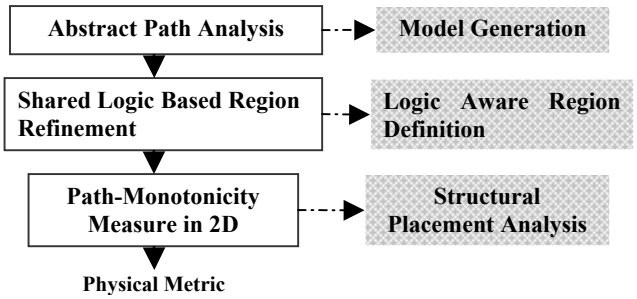

Figure 2: Proposed Metric Computation Approach

3.1 ABSTRACT PATH ANALYSIS

The role of netlist abstraction is in identifying direct and indirect paths involving macros, I/Os, and other design objects of interest while abstracting away the intermediate logic. The abstraction method employs a hybrid, controlled path enumeration approach to generate a static model that captures the logical nature (structure of netlist in terms of paths) of the design. The abstraction methodology [12] retains all design objects of interest, while abstracted cells in the input netlist are represented as directed hyper-edges with constraint annotations. The representation of abstracted cells as attributed hyper-edges, instead of complex (i.e., hierarchical) blocks, helps to preserve the signal path oriented behavior of the design. This model generation is done only once for a design netlist. Different placement solutions of the design netlist are evaluated using the same abstract model of the netlist.

The input to the abstraction process is the design netlist, which can be represented as a hypergraph*, DG (V, E),* where *V* is the set of nodes representing the cells, and *E*, the set of hyperedges representing the nets. There are two distinct phases in the abstract model creation process, namely: (1) *Marking phase*, where design objects of interest are identified in the original netlist. The result of the marking phase is a subset of design objects from the original design graph, $V' \subseteq V$. For the example in Figure 3, the marked objects of interest are L1, L2, L3, and L4. (2) *Abstract network generation or tracing phase*, wherein paths between objects of interest are identified as abstract or virtual interconnections between marked objects. The tracing algorithm starts from an object of interest, searching for a path leading to another object of interest through a modified depth first search on the hypergraph, *DG (V, E).*

The tracing phase accepts the set of objects of interest V' identified by the marking phase, and constructs an edge attributed hypergraph, $AG(V', E')$, which represents the abstract model. E' is the set of abstract paths and is defined as follows:

$$E' = \{e_i(u,v), i = 1..n \mid u, v \in V', u \neq v\}$$

Each abstract path represented by an edge $e_k(u,v) \in E'$, consists of a set of physical paths, which is the attribute on the edges of the abstract graph, $AG(V', E')$ and is given by,

$$\forall e_k(u,v) \in E', Attrib(e_k(u,v)) = \{p_1^k, p_2^k, ..., p_m^k\}$$

Where, each physical path p_j^k defined by the attribute of the edge $e_k(u,v)$ (abstract path), is represented as a set of cells that exist on the physical path between *u* and *v*.

$$p_j^k = \{c \mid c \in C_j^k, C_j^k \subseteq V - V'\}$$

Where C_j^k is the abstracted logic (set of cells) associated with the physical path, p_j^k. For the example shown in Figure 3, abstract paths are shown in dotted lines as AbsPath 1, AbsPath 2, and AbsPath 3. Placement locations of the vertices of an edge (abstract path) in *AG* identify the region in which the path monotonicity is analyzed for that abstract path. Complexity of the tracing phase is controlled through a dynamic programming implementation. There are other parameters built into the tracing mechanism that controls the number of logic levels and fanout of the intermediate nodes which further help bound the complexity of path tracing.

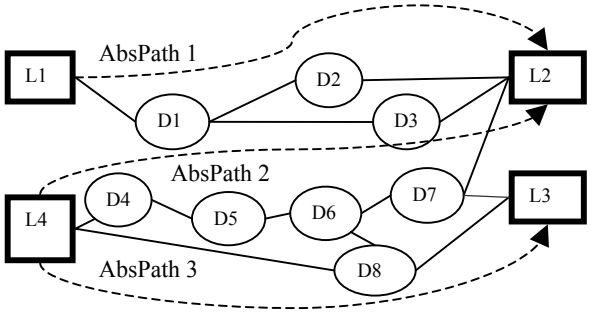

Figure 3: A Sample Design Netlist and Abstract Paths

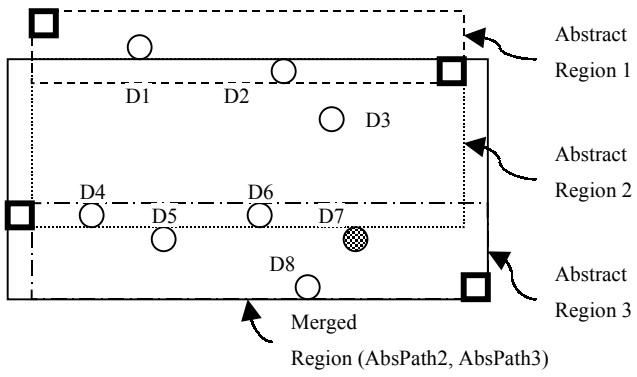

Figure 4: Analysis Regions Defined by Abstract Paths

3.2 LOGIC AWARE REGION DEFINITION

The abstract model generation described above attempts to cluster (or group) the logic cells that lie on a physical path between a pair of marked objects. For example, in Figure 3, AbsPath 1 groups D1, D2 and D3 based on the physical paths between marked objects L1 and L2. However, the logic cells that lie on a sub-path that is common to multiple abstract paths would result in shared logic that is part of multiple clusters (groups) defined by the corresponding abstract paths. This is illustrated in Figure 3, where AbsPath 2 (D4, D5, D6, D7) and AbsPath 3 (D4, D5, D6, D7, D8), have shared logic cells (D4, D5, D6, D7). Treating these abstract paths independently for region based placement analysis may lead to a misleading measure of physical path-monotonicity, since the extent of shared logic may skew the monotonicity measure of some of the abstract paths. In this case, based on the shared logic cells information, one might want to analyze the abstract paths AbsPath 2 and AbsPath 3 together to define a merged region (AbsPath2, AbsPath3) (shown in Figure 4) for placement analysis. We use a clustering-based approach to globally solve the path and region redefinition problem based on shared logic detection across abstract path. The information about logic sharing between abstract paths is maintained from the path tracing operation defined in section 3.1.

We construct an undirected graph, *shared logic graph (SLG)* whose nodes are abstract paths and edges represent existence of shared-logic between two abstract paths. There is a cost associated with every edge in the graph based on the number of shared logic cells and physical proximity of regions corresponding to the abstract paths. It must be mentioned that the nodes in this graph represent only those abstract paths that have common (shared) logic between them. In the given example, AbsPath 2 and Abs Path 3 will be the only two nodes in the graph, since Abs Path 1 has no shared logic with the other two. We use a simple clustering heuristic that attempts to merge nodes in the *SLG* based on the edge weights with a threshold on the cluster size as well as the size of the analysis regions corresponding to the cluster. It is important to consider the latter factor since it would have a direct bearing on the accuracy of the subsequent placement analysis step. At the end of this region definition step, we would have the abstract model with the refined physical regions based on logic sharing information across abstract paths. This is represented by the modified graph $AG'(V', E'')$, where the edge set $E'' \subseteq E'$ is the reduced set of abstract paths that would be used for placement analysis.

3.3 STRUCTURAL PLACEMENT METRIC

In this section, we describe the actual computation of the metric given the placement information of the leaf cells. The preceding sections discussed the logic abstraction and region definition process, both of which would be combined with the placement information in the analysis phase. The structure-aware physical metric consists of measuring the non-monotonic component of the total path length measure. Each abstract path is a collection of physical paths and the analysis scheme associates two measures with each abstract path – the path length and the monotonicity measure. The path length (PL) component corresponding to each abstract path is the length of the longest physical path contained in it. The longest physical path information is computed in the tracing phase of the netlist abstraction process (section 3.1) and is stored in the model. The path monotonicity (PM) measure is computed based on the containment of abstracted logic placement within the physical regions defined by the abstract paths. Figure 5 shows the region defined (solid rectangle) for an abstract path and the placement of the abstract logic (cells D4, D5, D6, D7 and D8) in between. The criterion for monotonicity measure is primarily based on the containment of the logic cells within the defined region. Recall that the region definition itself is based on the placement of the marked objects of interest (like latches, large macros, IOs). As shown in the figure 5, the region definition includes the marked objects in entirety. Using the abstract path information and the defined physical region, the monotonicity measure for an abstract path represented by edge $e_k \in E''$ of $AG'(V', E'')$, is calculated as follows:

$$M(e_k) = \frac{N_c^k}{N^k} * \frac{1}{1+W} \quad \text{..... (1)}$$

Where, N^k is the number of unique cells in the abstract path, e_k, given by

$N^k = \left| \bigcup_{j=1}^{|Attrib(e_k)|} p_j^k \right|$, and N_c^k is the number of cells contained in the region defined for the abstract path e_k. W is the weight associated with path non-monotonicity and is given by

$$W = \begin{cases} (MaxWt/K) * x & if, x \le K \\ MaxWt & if, x > K \end{cases} \quad \text{..... (2)}$$

Where, x is the average distance from the edge of the defined region by which the cells fail containment.

The weight is assigned based on a relaxation window defined by factor K (symmetric distance by which the physical region window is relaxed on all four sides). Intuitively the relaxed window definition accommodates the containment failure of leaf-cells that might be shared between different abstract paths and hence cannot be fully contained in the region defined for each. The non-monotonicity weight assigned to each abstract path itself is a function of the extent of deviation of the leaf-cell placement from the defined region.

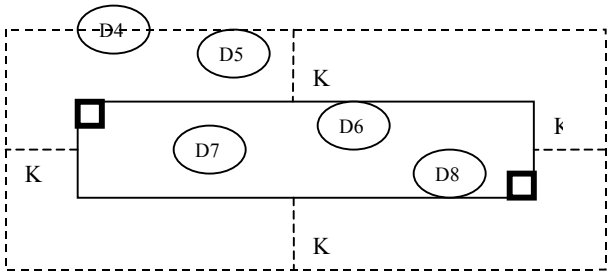

Figure 5: Path Monotonicity Computation

For example, in figure 5, the cells D6, D7 and D8 are contained in the basic physical region while cells D4 and D5 fail the containment criteria. So, for this example, N=5 and N_c = 3. The non-monotonic weight associated with this path is a linearly scaled function (Eq. 2) of the average cell deviation upto the relaxed window limit from the region edge defined for the abstract path. For the experiments conducted we used a MaxWt of 10 and a window relaxation factor of 500 design units. As can be seen from (Eq. 1) a completely monotonic path would have a non-monotonicity weighting of zero

(**W=0**) and hence the monotonicity of the abstract path, *e*, would be 1 (**M(*e*)=1**). For paths that are partially monotonic (those that satisfy containment of a subset of cells in the path), the weighting scheme in (Eq. 2) is scaled based on the average distance of failure up to the relaxed window limit. For cells that fall beyond the relaxed window, a constant weight (MaxWt) is assigned. The MaxWt of 10 chosen for these experiments reflects the granularity of impact the non-monotonic cells placements would have on the monotonicity measure for each abstract path. The relaxation window limit of 500 was selected based on empirical observation of the maximum distance of failure from region edge for all the abstract paths across designs.

Total Path Monotonicity Measure, $\boldsymbol{TPM} = \sum_{e \in E''} M(e)$

Total Path Length, $\boldsymbol{TPL} = \sum_{e \in E''} PathLength(e)$

Degree of Monotonicity, $\boldsymbol{\alpha} = TPM / |E''|$, Where $|E''|$ is the number of abstract paths in the logic network after shared-logic based path (and region) redefinition. The proposed metric is given by,

$$Metric = (1 - \alpha) * TPL$$

Intuitively, the metric reflects the non-monotonic component of the total path length as measured across all the abstract paths. The lower the numerical value for a solution, the better the quality of placement as predicted by this metric. This measure captures both the physical goodness of the embedding (through the monotonicity measure) as well as the explicit path length measure. Such a combined metric reflects truly the quality of placement solution with respect to the structure of the logic network as compared to a single cumulative measure like total wirelength or total path length.

Table I: Design Characteristics

Network	# of Abs. Paths	# of Cells	# of Nets
Design I	163462	73 K	74 K
Design II	266955	58 K	59 K
Design III	7707194	132 K	147 K
Design IV	6037967	216 K	223 K
Design V	8460662	303 K	330 K
Design VI	8421541	300 K	323 K

4. EXPERIMENTAL RESULTS

The program for metric computation was implemented in C++ in a physical design environment ChipBench [13]. Experimental results are presented for real industry designs whose characteristics are shown in Table I. The experimental setup involved generating different placement solutions by varying placement tool [16] parameters as well as inducing changes to the floorplan and computing the metric for each instance. The best solution as predicted by the metric was compared against the timing results generated before the start of detailed physical synthesis. Each of these placement instances was also taken through a physical synthesis tool [14] to record the final timing. The aim of this experimental setup is to show that the placement solutions that measure up to a better value (lower) of the metric also have a better timing solution, while the same cannot be said of wirelength optimal solutions.

The results presented are not to highlight the absolute quality of the timing solutions but more to explicate the correlation of the proposed metric with timing measure. Our goal in conducting these experiments is to state the correlation while we envision *greater benefit in using such a metric in the context of a placement tool generating solutions that optimized for this metric along with wirelength, instead of purely optimizing for the latter*. In other words, the solutions presented in Table II would be part of the solution space searched by a placement tool that attempts to arrive at a good solution with respect to the proposed measure. Since the current work is to establish the usefulness of such a topology-aware measure, the focus was not on integrating this metric with a placement algorithm.

Table II: Results for Industry Designs

Design	WL X 10^6	Metric X 10^6	Figure Of Merit (initial)	# of Neg. Slack Paths (initial)
Design I – P1	12.475	2.720	-4746	6324
Design I – P2	11.855	2.820	-5888	7065
Design I –FP1	12.612	2.170	-3998	5475
Design II – P1	10.261	2.081	-27202	21524
Design II – P2	9.536	2.576	-30065	21860
Design III – P1	104.149	59.80	-19863	52563
Design III – P2	103.074	65.00	-19821	54980
Design III – P3	102.416	59.00	-19230	52991
Design IV – P1	49.579	13.90	-9728	8438
Design IV – P2	47.815	13.70	-8131	7301
Design IV – P3	47.672	12.20	-8474	6466
Design IV – P4	45.213	11.10	-6908	6125
Design V – P1	135.285	19.40	-14519	30578
Design V – P2	132.714	17.00	-13414	27486
Design V – P3	130.728	17.20	-14441	25717
Design V – P4	129.318	15.00	-13961	27131
Design VI - FP1	128.405	15.25	-12984	24798
Design VI - FP2	166.848	20.71	-23732	47091

The results shown in Table II indicate the extent of correlation between the quality of solution predicted by the metric and the timing results at the start of the detailed physical synthesis phase. Each row indicates a specific design with placement variations, either by controlling the placement tool parameters (indicated by Design-PX) or through floorplan changes (indicated by Design-FPX). The second column shows the wirelength (net half-perimeter) as measured by the placement tool. The third column shows the physical metric reflecting the non-monotonic component of the total path length (as described in Section 3). The initial timing measure is presented through two values indicated in columns three and four of Table II. The first of these, Figure of Metric (FOM), is a measure of the cumulative slack of all negative slack cells in the design. The

closer this value is to zero, the better the timing characteristics of the design [15]. The second measure is the total number of negative slack paths in the design. The timing measures reported in the table corresponds to the design state before physical synthesis phase in the design closure flow (figure 1). As seen in the table, for Design I, the best placement solution predicted by the metric corresponds to instance Design I–FP1, which has the lowest non-monotonic path length component and with respect to timing measure, it has the least FOM and number of negative paths. However, the best wirelength solution instance, Design I–P2, does not correspond to the best timing solution. A similar observation can be made for Design II, while Designs III, IV and V show that the best solution predicted by the metric is also wirelength optimal. In case of Design III, with respect to placement instances Design III–P1, and P2, the metric predicts P1 to be better than P2, which can also be perceived as the better timing solution, since the FOMs are comparable but the number of negative paths being significantly smaller for P1 compared to P2. Design-VI is a variant of the netlist for Design-V, for which the metric prediction correlates with the best timing and wirelength solutions. For most designs in these experiments the best solution predicted by the metric also have the best post-physical synthesis timing measures.

The results were obtained by running the metric computation program on a quad processor RS6000 S80 machine running at 270Mhz with 2GB of memory. In all the experiments, the abstract model generation involved *marking* the latches and the large arrays in the design. The logic depth setting for the abstract path generation was set at a reasonably high number (250) to capture a large extent of the logic structure between marked objects. This number was arrived at empirically through observing the invariance in the number of abstract paths generated beyond this logic depth threshold. Once the abstract model is generated, run times for computing the non-monotonic path length component for these designs range from 5 to 20 minutes.

5. CONCLUSIONS

In this paper we have motivated the concept of design-topology awareness in a placement analysis metric, presented such a measure and an efficient approach to compute it. Through experimental results in a commercial strength timing closure setup on real designs, we have established the usefulness of such a design-topology aware metric through its correlation with early timing results. The metric computation approach presented in this paper includes novel techniques for path-behavior preserving logic abstraction, shared logic based abstract path clustering, and physical region based monotonicity analysis.

In certain cases, the best solution predicted by the metric (also the best timing solution) also happens to be wirelength optimal. However, the proposed design topology based metric has the ability to identify timing friendly solutions that are not wirelength optimal. Therefore, examining such a metric along with traditional wirelength based measures to analyze placement solutions for a physical synthesis flow, would lead to generating a more timing-friendly placement that can potentially reduce the design closure time.

While the proposed metric has the ability to distinguish solutions based on timing, it still does not incorporate any flavor of congestion. Future work is focused on addressing the following two aspects: (1) Extending the current framework to integrate both wiring congestion and design-topology awareness which would facilitate the metric to select a solution that is good in terms of initial timing and wirability. (2) Explore the feasibility and techniques required to incorporate the path monotonicity based metric along with the traditional wirelength measure, as a part of the objective function for placement algorithms.

6. REFERENCES

[1] A. E. Cadwell, A. B. Kahng, and I. Markov, “Can Recursive Bisection alone Produce Routable Placements?”, Proc. of ACM/IEEE DAC, 2000, pp.477-482.

[2] A. B. Kahng, I. Markov “Analytical Optimization of Signal Delays in VLSI Placement”, Proc 3rd Annual IBM ACAS Conference, Feb 2002, pp. 62-68.

[3] W. Gosti, A. Narayan, R.K. Brayton, A. L. Sangiovanni-Vincentelli, ”Wireplanning in Logic Synthesis”, Proc. ICCAD 1998. pp. 26 – 33.

[4] S-L Ou, M. Pedram, ”Timing-driven Placement Based on Partitioning with Dynamic Cut-net Control”, Proc. DAC 2000, pp. 472-476.

[5] C. Ababei, N. Selvakkumaran, K. Bazargan, G. Karypis,

”Multi-objective Circuit Partitioning for Cutsize and Path-Based Delay Minimization”, Proc. ICCAD 2002., pp.181-186

[6]”Timing Influenced Layout Design”, M. Burstein and M. N. Youssef, Proc. 22nd ACM/IEEE Design Automation Conference, 1985, pp. 124-130.

[7] W. E. Donath, et al, ”Timing Driven Placement Using Complete Path Delays”, Proc. 27th ACM/IEEE Design Automation Conference 1990, pp. 84-89.

[8] H. Eisenmann, F.M. Johannes, ”Generic Global Placement and Floorplanning”, Proc. of the 35th ACM/IEEE Design Automation Conference, 1998, pp. 269-274.

[9] D. J. Huang and A. B. Kahng,”Partitioning-Based Standard-Cell Global Placement with an Exact Objective”, Proc. ACM/IEEE ISPD, 1997, pp. 18 - 24.

[10] P. Kudva, A. Sullivan, and W. Dougherty, ”Metrics for Structural Logic Synthesis”, Proc of ICCAD 2002, pp.551-556

[11] W. Chen, C.T. Hsieh, M. Pedram, “Simultaneous Gate Sizing and Placement”, IEEE Trans. CAD, Vol 19, Feb 2000, pp.206-214.

[12] N. R. Dhanwada, et.al , “Method for Optimizing a VLSI Floorplanner using a path based hyper-edge representation”, US Patent No. FIS920020131US1, Filed December 2002.

[13] J. Y. Sayah et. al “Design Planning for High-Performance ASICs”, IBM Journal of Research and Development, July 1996.

[14] W. Donath, P. Kudva, L. Stok, P. Villarrubia, L. Reddy, A. Sullivan, K. Chakraborthy, “Transformational Placement and Synthesis”, Proc. DATE 2000, pp. 194-201.

[15] C. Alpert, G-J. Nam, P.Villarrubia “Free Space Management for Cut-Based Placement”, Proc. ICCAD 2002, pp. 746-751.

[16] S. Hojat, P.Villarrubia, “An Integrated Placement and Synthesis Approach for Timing Closure of PowerPC MicroProcessor”, Proc of ICCD 1997 , pp.206-210.