Int. J. Open Problems Compt. Math., Vol. 2, No. 3, September 2009
ISSN 1998-6262; Copyright ©ICSRS Publication, 2009
www.i-csrs.org

Interconnect Synthesis in High Speed Digital VLSI Routing

Moustafa A. Sayed and Ehab Y. Abdel Maksoud

Department of Engineering Mathematics and Physics,
Faculty of Engineering, Cairo University, Egypt.

Abstract

The advent of the nanotechnology has introduced new challenges and non-conventional problems to high speed digital Very Large Scale Integrated (VLSI) design. Moreover, the resultant progress of manufacturing technology is widening the gap between current Computer Aided Design (CAD) tools and VLSI technologies. This is reflected clearly in the IC design process where the Integrated Circuit (IC) flow has become very iterative, especially in the back end phase. These returns to the complexity of placement and routing phases, and the inadequate approximations used for interconnect modeling and characterization. In this paper, we introduce a new performance-wise approach that combines connection graphs with a basic area routing algorithm to complete the routing without or with minimal iterations. In this approach, the routing elements are ordered with respect to its space not with respect to their relative position, to modify its complexity and optimize memory requirements to fit the problem in hand. This can help to yield a time complexity of $O(\log\log S)$ *and a memory complexity in worst case* $O(S)$*, where* S *is the size of the space. Instead of employing the traditional flow that uses inaccurate interconnect models and characterization techniques, and then analyze the output to point out errors in the resulting layout, we propose Interconnect Synthesis as a framework. Within this framework, we introduce an enhanced routing algorithm that outperforms traditional routers and present a more accurate interconnect modeling and signal characterization techniques for* *interconnect synthesis. Consequently, the output of the routing phase will no longer suffer from the error in the timing or the signal integrity constraints. Finally, the analysis phase will become merely a verification phase, not a process for re-iteration.*

Keywords: Detailed routing, Global routing; High speed digital VLSI design, Interconnect synthesis, Interval labeling scheme, Line Search routers, Sequential routing.

1 Introduction

The advent of the nanotechnology in the Very Large Scale Integrated (VLSI) community has been accompanied with several new challenges that impose physical limitations on the VLSI process. These challenges are compounded by the relentless advancement in manufacturing technology, where the increase in integration density is pushing towards greater design complexities. Furthermore, within the tight time-to-market window frame, we aim for optimal solutions under the limitations mentioned. Thus, under all these constraints, the role of Computer Aided Design (CAD) becomes major and indispensable. Unfortunately, current CAD technologies can't hold up with the growing limitations of the manufacturing technology and can't adequately reap the full potential of the available technologies. One of these critical limitations CAD technologies are facing in the Deep Sub Micron (DSM) designs is interconnects, as discussed in the International Technology Roadmap for Semiconductors (ITRS) roadmap [1]. Interconnects, once been ignored, have dominated the design process and are a major bottleneck in today's design flows.

In [2], Peterson introduced the key challenges of the Application Specific Integrated Circuits (ASIC) design in the upcoming nanotechnologies. He mentioned seven main key challenges which are: the process challenge and its physical limitations, the design complexity challenge, the performance challenge, the power density challenge, the quality and reliability challenge, and the productivity level challenge. With a deeper look, we'll find that interconnect plays an important role in most of these challenges. At the physical level, signal integrity and delay are of main concerns. At the design complexity level, interconnects and their prediction at early stages are increasing the complexity design process. At the performance level, interconnects delay are increasing as they don't scale down at the same rate as devices. At the chip density level, interconnects and interconnect repeaters are expected to occupy a large portion of the design. At the productivity level, interconnects are considered a bottleneck in today's design flows and lead to several long iterative cycles before the design is complete. In the current DSM technologies, the main limitations of interconnects that have a major impact on the physical design process includes hard timing constraints, signal integrity, manufacturing variability, yield, and reliability. These limitations not only affect the performance of the design, in fact they could lead to failure of operation in several cases.

Routing is a very wide field with several research results in the literature. As a broad classification, there are two main techniques of routing [3]. The first class aims for routing a group of nets simultaneously, as in Over The Cells (OTC) routing [4] and Hierarchical routing [5–8]. Others, such as Tile Expansion Shortest Path router [9] and the XRouter [10, 11], are sequential routing which route each net sequentially. A more detailed survey can be found in [3, 12]. The main problem is that all the routing models are NP-complete, and the techniques that route nets simultaneously are heuristic in nature with inferior output quality. Hence, sequential routing as a class of more deterministic nature, where the router concentrates on one route at each time is pursued in practice and more widely discussed. As for the main drawbacks of sequential routers: "net ordering" and "rip-up & re-route" techniques, several researches have been conducted in this issue, and can be incorporated in any of the sequential routing techniques [9 – 11, 13].

Another point, which is important for DSM designs, is wire sizing and spacing, which has taken a lot of attention lately in DSM research and is considered one of the drawbacks in the area of the routing techniques. Yet, spacing and sizing can be considered as a post-processing step by using an adequate grid/line spacing in the layout so that the compaction of the layout will succeed in producing the required Design Rule Check (DRC) correct output [3, 12].

When studying the characteristics of the signal on a wire, we mean that we want to evaluate some properties of the signal to make sure that the logical signal values confirm with the physical ones. Correct functionally are achieved through correct voltage levels and correct timing. These two terms are encapsulated in terms of signal integrity. Signal integrity includes the delay of the signal, which is defined as the time for the signal to transit from one logical level to another. It also includes the overshoot and undershoots due to the inductive effect of the line, where the signal oscillates around its asymptotic value. Currently, most available researches include only delay in their routing techniques. One of the most famous and widely used techniques to calculate delay is Elmore delay technique [14, 15]. Other formulas are used to calculate capacitive crosstalk. Unfortunately, the existing approaches use poor interconnect models and inaccurate techniques for signal characterization (to calculate delay, overshoot, or undershoot) so they can't detect any real violations in signal integrity, hence they reduce the iterations in the design process but not to an acceptable limit. Very few routers support RLC models during routing. Recently, there has been several works [15 – 18] that takes the inductive effect in account during global routing to accommodate the RLC effect. Unfortunately, the formulas used are low order formulas to characterize some of the signal characteristics to cover all possible interconnect cases. Therefore, this can't catch several signal integrity effects, and are not effective on the long run for next generation manufacturing technologies.

One of the interesting concepts that were introduced in the Printed Circuit Boards (PCBs) flow is Interconnect Synthesis (IS) [19]. IS technology, owned by Mentor Graphics Corporate (MCG), and is implemented in its backend flow that presents an effective way to overcome. This technology presents an effective way to overcome the re-iterative nature of the physical design process. According to the traditional design flow, a synthesis/verification step is a main component of this flow. In synthesis, due to incomplete information of the layout, the tools either don't accurately model the interconnects or leave it for the verification step to point out the timing and signal problems. On the other hand, the IS technology proposes a new approach to solve the problem. The core of this technology is to use a transmission line simulator capable of analyzing nets in real-time during floor planning, placement, and routing. Using this accurate model converts the electrical constraints by user (like timing and signal integrity constraints) into geometric and physical constraints for the tools to follow. In fact, it can be incorporated in any of the proposed design stages to give much better performance of the system and make the routers job tractable.. In additional, using such high accuracy in synthesis can help to eliminate or reduce re-iterations in design.

In this paper, we propose a high performance routing algorithm and the data structure that can achieve high speed routing with adequate memory requirements. The routing process consists of two consecutive phases: global routing and detailed routing. The global routing phase is used to find an initial and rough solution of the given routing problem. To speed up and guarantee the success of the routing, the detailed router uses this solution to search for the exact path of the interconnects. The proposed routing algorithm is an interval labeling algorithm that is capable to achieve the minimum cost path, while using line search techniques for high performance. In this work, we consider the cost of the path as a function only of delay. To make the router more accurate, we integrate IS [19] within the system to guide the router in its search for the shortest path efficiently. We have two phases for IS. The first phase is done before routing, and is done once per technology to generate the synthesis formulas. The second phase is done during routing to synthesize the different routing parameters. Unlike traditional approaches that find a solution without taking several important constraints or handles them but with low accuracy, and then depend on the testing phase to reveal the problems in the proposed solution, our approach mainly depends on using accurate synthesis, which means that CAD tools are to understand the constraints and apply them during processing to accurately synthesize the design. Simulation studies are used to evaluate the efficiency and the merits of the proposed routing algorithm and interconnect synthesis methodology approach to produce better results that reduces the error in the computed delay of the routed layout and minimizes the overall running time, relative to the previous traditional approaches.

The remainder of this paper is organized as follows: Section 2 describes the proposed high speed routing model, and presents a global overview of the whole system, the main modules and how they interact. Section 3 presents the proposed high speed routing algorithm. Section 4 introduces the proposed accurate signals characterization technique of the interconnect synthesis and presents an algorithm that integrate interconnect synthesis with routing module. Section 5 presents target VLSI applications. Section 6 discusses the simulation results to evaluate the proposed approaches. The conclusion is provided in section 7. Finally, the open problem of the paper and the future work are drawn in Section 8.

2. The Proposed High Speed Router for Digital VLSI Design

The VLSI routing system under consideration is a multi-stage processing system. Due to the complexity of the process and the long run time, the process is divided into multiple stages. In the proposed system, we don't define the number of levels of routing nor do we go into the details of the inputs and the outputs of each stage. The first stages are used to identify any over-constraints in the given input layout and give a coarse solution for the routing problem. As we go through more levels, the more physical details are determined. Therefore, the proposed system, like most routing systems, consists of two main phases: global routing and detailed routing, as shown in Figure 1.

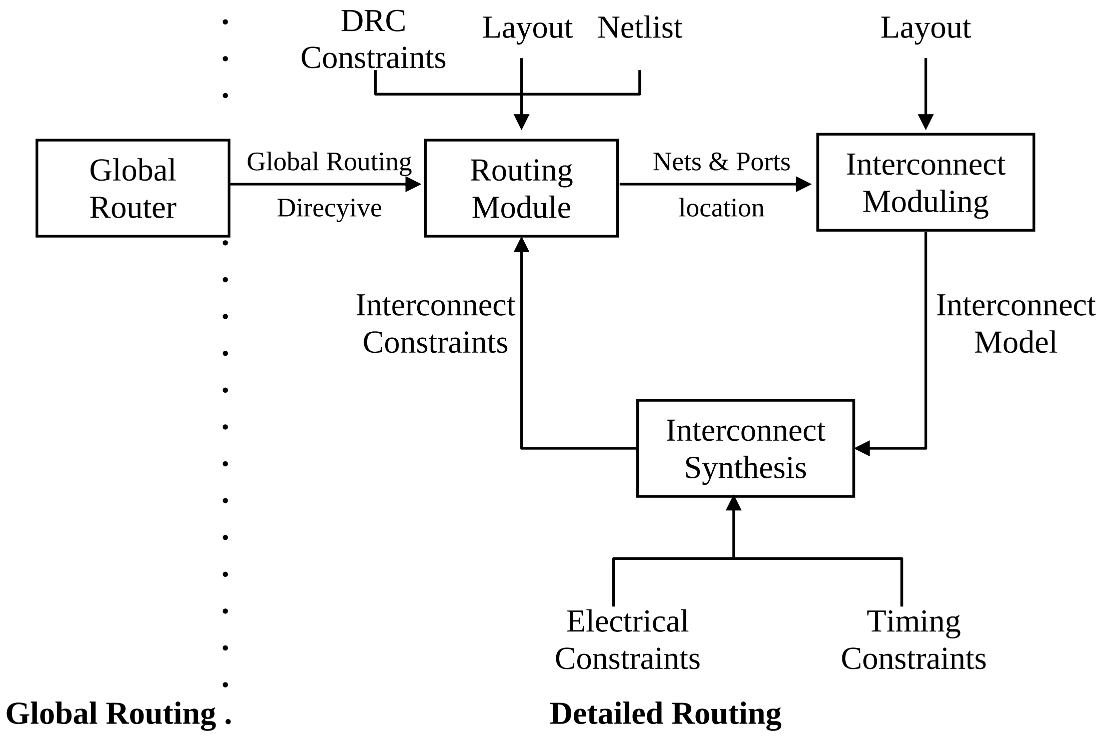


Figure 1: The complete proposed routing system.

2.1 Global routing

In the global routing phase, it's required to find an initial and rough solution of the routing problem under consideration. There are no constraints on the higher levels of this router. First, the routing area is divided into coarse regions. Then, the global router processes the net lists and determines through which regions each net will pass through. The exact location of these nets is to be determined in the next phase by the detailed router. The size of the routing regions depends on the routing system. Figure 2 shows how the layout is divided. The rectangular white grids are the coarse routing regions which the global router is working within, while the gray regions represent the layout cells. Figure 2(a) shows the bins of a gate array or sea of gates design, while Figure 2(b) gives the routing regions for standard cell design, and Figure 2(c) represents the divisions for more non-uniform block based layouts.

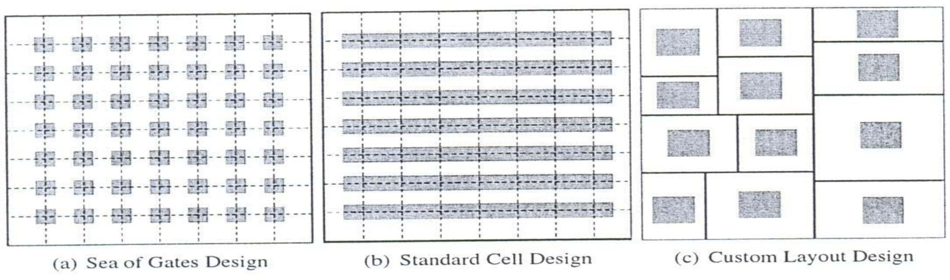


Figure 2: Classification of the routing region in global routing.

This division can be represented by a graph G given in Figure 3, whose vertexes V represent the routing regions, and the set of edges E represent the connections to neighboring regions. Given a set of nets to be routed, N , where the ports of each net is located in one of the routing regions V , it's required to connect these nets and determine their routes through the graph G . Every edge e is given a weight to represent its wire capacity (the number of wires that can pass through it) called the boundary capacity or supply capacity (e), and another value to represent the number of wires routed through it called flow ($flow(e)$).

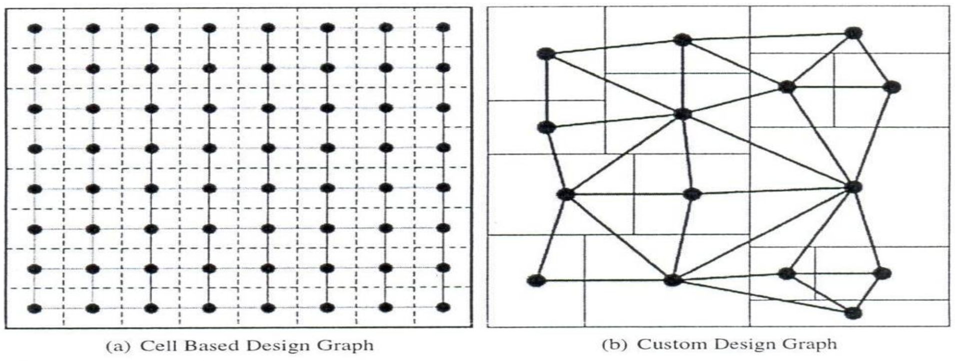


Figure 3: The graph representation of the layout

2.2 Detailed router

The detailed routing phase receives directives from the global routing stage(s) to speed up and guarantee the success of the routing in this stage. In general, we assume that these directive include the bins through which the interconnects lie, rough estimation of spacing, and initial wire sizing for signal integrity limits. It's required by the detailed router to find the exact path of the interconnects. The proposed detailed routing system, as shown in Figure 1, is composed of three main modules: the Routing Module, the Interconnect Modeling (IM) module, and the Interconnect Synthesis (IS) module.

2.2.1 The routing module

The routing module is the starting point to the detailed routing process. It is responsible for routing the nets in a constraint-driven environment. After the global router finishes its processing, it passes to the detailed router its global routing and constraints on nets. This module accepts as inputs:

1. **The Layout**, which describes the placement of the standard cells and the location of unconnected ports.
2. **The Net list**, which shows the terminals that belong to the same net and should be connected.
3. **Constraints**, which includes:
    - (a) **The Design Rule Checking (DRC) Constraints**: These are the geometrical constraints [3, 12] that the router should satisfy during router.
    - (b) **The Global Routing Directives**: These are global constraints on the path of the wire, the width, and the spacing of critical nets.

Other constraints can be defined and handed over the detailed router.

(c) **Interconnect Synthesis Constraints**: These constraints are more accurate than the global router directives on the wires, in terms of width, spacing, and length.

The routing module receives from the IS module the geometrical constraints required to satisfy the signal integrity constraints. This is done on several phases: First, the synthesis phase, where the geometrical constraints are synthesized based on a rough estimation for the possible paths and the router chooses the best path. Second, the analysis phase, where the nets that are completely routed should be accurately analyzed to ensure that no constraints are violated. Finally, the incremental analysis phase, that is to re-consider nets which have already been routed, but are affected by the newly routed nets in their neighborhood. According to the analysis/synthesis of the IS module routing is performed, net ordering can be changed, rip-up and re-route might become necessary, and the cost function can be computed.

2.2.2 The interconnect modeling module (IM module)

This module is responsible for modeling the interconnects and providing the interconnect synthesis module with an accurate model of the interconnect. Modeling the interconnect mainly depends on the interconnect length, yet there are other criteria that should be considered. This also includes width and location of the net with respect to the devices, and its location with respect to other nets in the layout. It's the responsibility of the IM module to assign each interval of the path with an appropriate RLC value. It also has to take in consideration capacitive and inductive cross coupling. The more accurate the models are used, the more exact are the single characterizations results. Most important is that each module is built independent of how other modules perform their processing to preserve modularity of the design.

2.2.3 The interconnect synthesis module (IS module)

This module is responsible to supply the router with the several geometrical constraints on nets including width, spacing, and length to satisfy the signal integrity and timing constraints. It takes as inputs:

1. **Timing constraints**, which give the maximum delay allowed for each net. These constraints come from the logical synthesis phase.
2. **Electrical constraints**, which include the maximum overshoot and undershoot.
3. **Interconnect model**, which gives the resistance, capacitance, and inductance of the model per segment.

Based on the provided model, there are three main functionalities required by the IS Module:

1. **Interconnect Synthesis**: It is used to compute the required geometrical constraints on the wire, as in width, length, and spacing. Since the exact interconnect path is not yet known, approximate models for the interconnects can be used. Moreover, it's required to choose the best path for the router to follow in its search, by comparing the signal characteristics along different paths.
2. **Interconnect Analysis**: For critical nets, after being routed, it's required to analyze the signal to ensure that no constraint has been violated.
3. **Incremental Analysis**: Since the whole layout is not completely routed, new information is provided as new nets are routed. It's required to be able to incrementally analyze the interconnect that might be adversely affected by neighboring signals.

It should be noted that due to the large number of nets, using the highest accuracy with all of the nets becomes too expensive in terms of computational cost. Therefore, the IS module should include several levels of accuracy for the analysis/synthesis of the nets. It can include simple capacitive models, RC models, RLC models, or transmission line models. Depending on the model used, simple analytical formulas can be used or more complex IS computation can be performed. In this work, we focus on interconnect synthesis and don't support all the signal integrity constraints. We only concentrate on the delay of the wire, and provide a very accurate signal characterization technique that can be used for interconnect synthesis in high speed VLSI design.

3. The Proposed Performance-wise Routing Algorithm

There are mainly two main algorithmic approaches: Area Routers and the Line Search routers. Line search routers are known for its efficient performance and area routers are known for optimal path. In the proposed routing algorithm, we combine these two approaches and present an efficient router in terms of speed and memory by using an interval labeling scheme as in [11]. This algorithm is an application to Dijkstra's algorithm [20] using Breadth First Search (BFS) technique. The main objective we pursue here is to achieve the performance of a line search algorithm to find the shortest path in terms of time and memory complexity. Hence, we find that studying the orthogonal line intersection problem will lead us to more efficient algorithm. The orthogonal line intersection problem is one of the well-known problems in Computational Geometry which is defined as follows: Given a set of horizontal lines S and one vertical line l, it's required to find the subset $S_\perp$ of S that intersect l. To solve this problem, we first introduce the Adjacency Map [21] and the Hive Graph [22] that will help in our algorithm.

3.1 Adjacency Map and Hive Graph

The notation of the adjacency map was first introduced by Lipski and Preparata [21]. Given a set S of n horizontal segments, the vertical adjacency map $G(S)$ is constructed by interconnecting the horizontal segments in S using vertical (infinite, semi-infinite, or finite) segments as follows: from each endpoint of the segment in S, draw two rays shooting upward and downward respectively until they meet other segment in S except possibly at an endpoint. This creates a planar subdivision $G(S)$ with $O(n)$ vertices, which are the joints of the horizontal and vertical segments. $G(S)$ can be represented in $O(n)$ space using the adjacency lists associated with the vertices. We call the edges supported by the horizontal and vertical segments horizontal and vertical edges, respectively.

Chazelle noticed in [22] that the adjacency map is a useful tool which, when modified appropriately, can be used to handle the orthogonal segment intersection problem efficiently. His modification of the vertical adjacency map is called the hive graph. A hive graph $H(S)$ is derived from $G(S)$ by adding only vertical segments to $G(S)$, to speed up the search process, while maintaining $O(n)$ vertices and $O(n)$ space representation. However, it has the important property that each face may have, in addition to its four (or fewer) corners, at most two extra vertices, one on each horizontal edge. Figure 4 shows a vertical adjacency map and its corresponding hive graph, in which the additional vertical edges are depicted as dashed lines. By assuming that the endpoints of the segments in S all have distinct x- and y-coordinates, as in [22] did, one can conclude that each face of $H(S)$ has $O(1)$ vertices on its boundary. Given a query segment $(x; y_1, y_2)$, the segment intersection query can be handled as follows. We first find the face in $H(S)$ that contains the endpoint (x, y_1) in $O(\log n)$ time by using one of the well known planar point location algorithms [9, 23]. Then, we traverse a portion of $H(S)$ from bottom up following the direction from (x, y_1) to (x, y_2). Only a constant number of vertices are visited between two consecutive encounters of the horizontal edges that intersect the query segment.

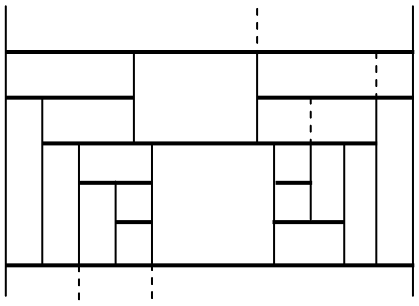


Figure 4: A hive graph

Our work makes use of the Hive Graph [22] as the basic data structure for the orthogonal line intersection problem. Several modifications are considered to make it more efficient and practical. First, our graph $G = (V, E)$ is defined, such that each node V represents a single horizontal level of lines. Edges E, are directed edges that are all in one direction, assume bottom-up. Hence, search is done only in one direction, i.e. we modify the fractional cascading scheme to be a single segment per line. Moreover, we assume that the pre-processing time is ignorable, as obstacles found in the layout at the start are very few and most of the horizontal line segments are free for routes to be occupied by nets. As new nets are routed, the horizontal line segments are segmented into smaller ones. The Stratified Tree (STree) [24] is used as the main data structure for the modified Hive Graph. Each horizontal level of lines is saved in the STree, and the end point (rightmost or leftmost) of the line is used as a key for the different operations. Fractional cascading is done through pointers between line segments and their children.

Our search graph start by dividing our layout into multiple alternating layers, with a preferred routing direction (horizontal or vertical). Each pair of adjacent layers (having horizontal or vertical directions) is treated as a single routing plane. Each plane consists of vertical and horizontal lines that represent free space in the layout for routing while occupied are represented as disconnections in the lines.

The correspondence between the graph and the layout model is interval based not line based. Each line is labeled with cost and interval leading to it. These intervals represent our vertexes in our graph. Edges between the vertexes are determined by the neighborhood function. Any two intervals connected through a Fractional Cascading pointer are connected in the graph (even if they are not adjacent). Also edges are labeled by a scalar function that is in terms of delay. To make our search more directed we use a A^* search algorithm where the cost function is a Rubin-like cost [25].

An incremental definition of the cost function during routing, as we move from interval (node) u to the interval v is given as:

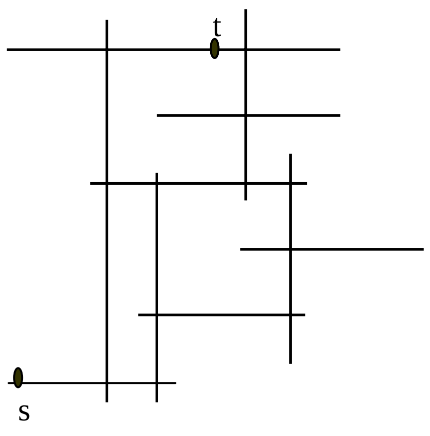
$$\delta(v) = \delta(u) + Delay(u,v) + (Delay_{Man.}(v,t) - Delay_{Man.}(u,t)) \qquad (1)$$

where $(Delay_{Man.}(v,t) - Delay_{Man.}(u,t))$ is the expected manhattan delay between nodes u and v, $Delay(u,v)$ is the actual delay of the path from u and v, $\delta(u)$ and $\delta(v)$ are the cost functions from the source s to the nodes u and v, respectively.

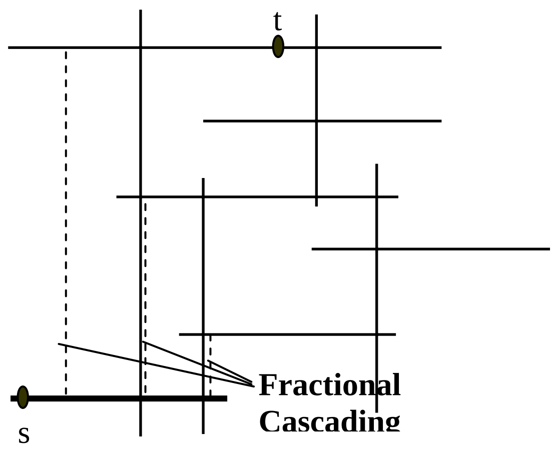
The algorithm starts by labeling the zero detour space (the paths that have a delay equal to the minimum delay which is usually part of the manhattan paths). If no path is found in all routing planes at this cost, then the detour space is increased until a path is found, or there are no more intervals to label. This step has a time complexity of $O((\delta_{path} + 1) \times l)$, where δ_{path} is the detour of the path from the source to the target, and l is the number of layers.

Each layer is labeled in the BFS fashion. The interval containing the source s port is first label with a cost of δ, as shown in Figure 5(a). We define a $1D$ bucketing data structure where bucket is a priority queue. The $1D$ bucket is divided in terms of the cost value, and the priority queue is defined in terms of the manhattan delay between the interval and the target. All neighboring lines are then labeled, and inserted in their position in the data structure,as shown in Figure 5(b). Next, the minimum interval of cost δ is retrieved for further processing. Neighboring lines are reached in $O(1)$ time using the fractional cascading pointers, while finding orthogonal line segment requires $O(\log \log S)$ time, using the STree line search algorithm [24].

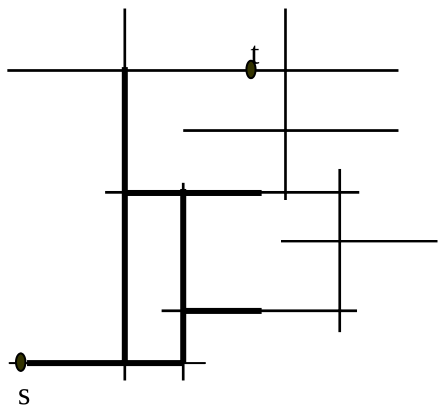
As for labeling the orthogonal lines, we wish to label only the orthogonal intervals with the top-most or bottom-most intersecting labeled intervals. This can be done by labeling them with the first labeled intersecting interval, as shown in Figures 4(c) and 4(d). This is because, while we label the plane, the intervals that are furthest are labeled first.



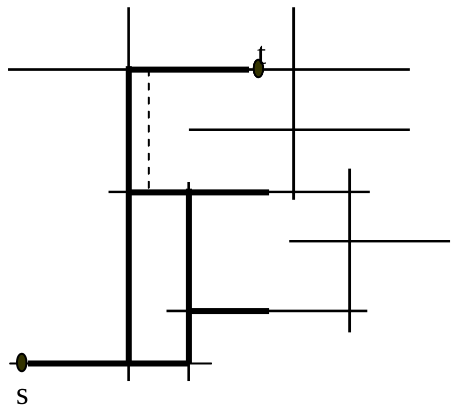
(a) Initial Layout.



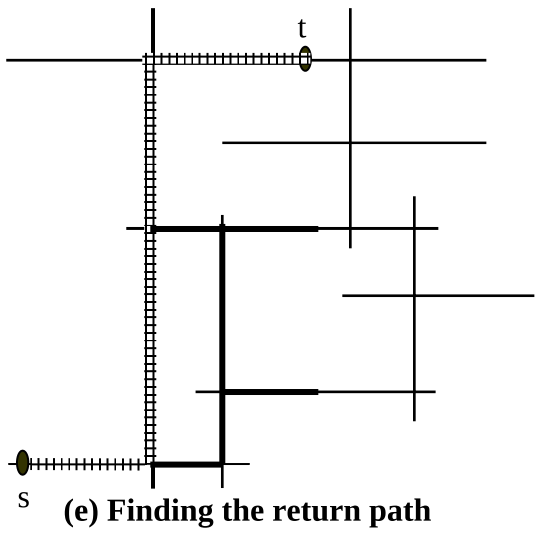
(b) Labeling Initial Interval.



(c) Top-most Labeling Intervals



(d) Bottom-most Labeling Intervals.



(e) Finding the return path

Figure 5: New Routing Algorithm Example

Each line segment is labeled with different intervals, each with a cost value. Since we are moving monotonically on different cost values, then no old interval can be re-labeled. Moreover, the labeling function is also monotonic on the same line, so labeling of lines includes only the insertion of new intervals and look-up for an interval with a given cost value, that is the current detour value. Hence, we check the interval with largest cost assigned to know if the line has been labeled for the current detour value or not. If it hasn't been labeled, then we can insert a new label. A simple data structure for this is a stack, which allows the label insertion to be done in $O(1)$ time. The steps of the proposed routing algorithm are given as follows:

1. Defined the starting interval as s and the end interval as t.
2. Set the current detour value $\delta = 0$.
3. Label the starting interval (this is the part of the line that has zero detour, not necessarily the whole line) l_s with $(s, 0)$.
4. Add l_s to the $1D$ bucketing data structure $BQ(\delta)$.
5. While target t not reached, repeat for each layer:

    5.1 While $BQ(\delta)$ not empty

    5.1.1 Set the current interval $l_{current}$ = Dequeue from $BQ(\delta)$.

    5.1.2 For each child line segment of $L_{current}$, that is $L^i_{current}$, connected

a fractional cascading pointer to the next line segment L_{next}:

5.1.2.1 Get the orthogonal line that intersect $L^i_{current}$ found in the label:

```
            $L_{\perp}$ .
        5.1.2.2 Find the first $L_{\perp}^{j}$ (get it through fractional cascading from $L_{\perp}^{j-1}$
                recursively) that connects $L_{current}^{i}$ and $L_{next}$ .
            5.1.2.2.1 Compute the cost value of $l_{\perp}^{j}$ (that lies on $L_{\perp}^{j}$ ), that is
                      $\delta\,(l_{\perp}^{j})$ .
            5.1.2.2.2 If the line segment $L_{\perp}^{j}$ containing $L_{\perp}^{j}$ has been labeled
                      before with the value $\delta\,(l_{\perp}^{j})$ , then continue to the
                      next orthogonal line
            5.1.2.2.3 Label $L_{\perp}^{j}$ with $(l_{current}, \delta\,(l_{\perp}^{j}))$ .
            5.1.2.2.4 Enqueue $l_{\perp}^{j}$ in $BQ\,(\delta\,(l_{\perp}^{j}))$ .
        5.1.2.3 Using fractional Cascading, label the next interval $l_{next}$ , if not
                labeled before and if connected to the $l_{current}$ through the
                orthogonal interval $(l_{\perp}^{j})$ .
            5.1.2.3.1 Compute the cost value of $l_{next}$ , that is $\delta\,(l_{next})$ .
            5.1.2.3.2 If the line segment $L_{next}$ that contains $l_{next}$ has been
                      labeled before with the value $\delta\,(l_{next})$ , then continue to
                      the next neighboring interval.
            5.1.2.3.3 Label $l_{next}$ with $(l_{\perp}^{j}, \delta\,(l_{next}))$ .
            5.1.2.3.4 Enqueue $l_{next}$ in $BQ\,(\delta\,(l_{next}))$ .
    5.2 Increment $\delta$ .
```

Since the proposed routing algorithm combines the Breadth First Search (BFS) technique with the STreet line search algorithm, it must be limited by the complexities of the STree algorithm [24]. Therefore, the algorithm has a time complexity of $O\,(\log\log S)$ and a memory complexity in worst case $O\,(S)$, where S is the size of the space (universe). It's quite clear that the time complexity of the algorithm outperforms any other router presented so far and achieves very high speeds as will be shown by the simulation results. Moreover, the size S is greatly reduced by the memory optimization techniques proposed to be applicable in large DSM designs. In the next section, starting from this algorithm, we pursue a more accurate interconnect analysis/synthesis procedures allowing it to have more computational time than other traditional approaches.

4. Interconnect Synthesis

In this section, we present the Interconnect Synthesis methodology. The core of this technology is to use a transmission line simulator capable of analyzing nets in real-time during floor planning, placement, and routing. Using this accurate model converts the electrical constraints by user (like timing and signal integrity constraints) into geometric and physical constraints to guide the router in its search for the shortest path. In addition, using such high accuracy in synthesis will eliminate or reduce iterations in design. In this work, we aim to handle the moderate lines that can be modeled as RC or RLC lines. Moreover, the technique proposed can be augmented by traditional analytical models for less critical nets, to reduce the computational cost. Consequently, we recommend simple analytical delay formulas for short lines and non-critical moderate lines. As for critical moderate lines (lines that have a delay close to the maximum allowed delay), we propose a more accurate approach for interconnect analysis and synthesis. The approach presented, using parametric model order reduction, can be extended to transmission line models as shown in [26], to handle long global wires.

We also limit ourselves in this research to certain aspects of the Interconnect Synthesis problem:

1. We concentrate on the delay of interconnect. Yet the technique developed can be extended to other signal characteristics as overshoot, undershoot, and crosstalk.
2. Synthesis of interconnect geometry according to the allowed line delay.
3. We take into account the line delay not including the driver delay.
4. The input signal driving the line is arbitrary; yet defining the input reduces the computational cost during routing.
5. The tree structure is a simple ladder circuit. Again, the proposed technique can be extended to multi-terminal nets and trees. In practice, circuit design techniques aim for effective fan outs around four for optimal performance [27], which leads to few branching cases (two branch points in maximum).

4.1 Formulation of circuit equations

Modeling the transmission line in a set of cascaded interconnected RLC segment, as shown in Figure 5, is one of the most accurate form of the interconnect modeling. Formulating the circuit equations can be derived through one of the circuit analysis techniques such as the Modified Node Analysis (MNA), nodal formulation, or sparse tableau. Using MNA for the lumped ladder circuit in Figure 5, the following first order differential algebraic equations system is derived:

$$M \frac{dx(t)}{dt} = -F\ x(t) + P\ u(t) \tag{2}$$

$$y(t) = P^T\ x(t) \tag{3}$$

where:

$$x(t) = \begin{bmatrix} V(t) \\ I(t) \end{bmatrix} \quad , \quad y(t) = \begin{bmatrix} i_{source} \\ v_{output} \end{bmatrix}$$

$x(t)$ is the state vector, vector $y(t)$ is the output(s) of the system, and vector $u(t)$ is the input stimuli. Matrix $V(t)$ is the voltage for the nodes of the system, including the output port. Matrix $I(t)$ is the branch current matrix, including the input port. M and F are matrices of size $n \times n$, while P is of size $n \times 2$, defined as follows:

$$M = \begin{bmatrix} C & 0 \\ 0 & L \end{bmatrix} \quad , F = \begin{bmatrix} 0 & F_{12} \\ -F_{21}^T & R \end{bmatrix} \text{ and } \quad P = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

where C, L, and R are the capacitance, inductance, and resistance matrices, respectively. P is the matrix that selects the inputs and the outputs of the system from $x(t)$ and $u(t)$. Matrix F_{12} covers the voltages in the state vectors while the matrix $-F_{21}^T$ covers the current variables.

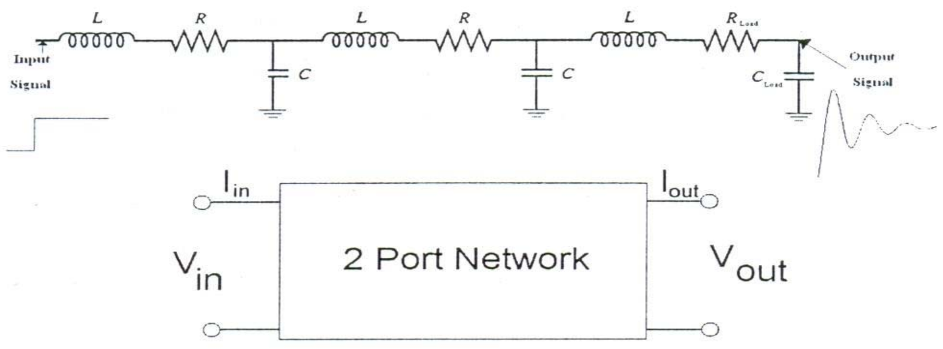


Figure 5: Lumped RLC model.

4.2 Parametric model order reduction

Several techniques are proposed for PMOR simulation [28]. In this work we apply the PMOR technique which is similar to that in [29], where our system equations, 2 and 3, are defined in the s-domain as follows:

$$E(s)\,X(s) = P\,U(s) \tag{4}$$

$$Y(s) = P^T\,X(s) \tag{5}$$

where $E(s)$ is defined as:

$$E(s) = F + sM \tag{6}$$

Then we defined the RLC system in terms of the interconnect parameters (geometrical parameters). These parameters are: the width of the interconnect, ω, the thickness of the dielectric between metal layers, d_C, the interconnect thickness, τ, and the of the interconnect segment, l, which is considered constant. Since the exact path is not known, it's suitable to assume a uniform RLC lumped model along the path. The modeling equations for C, R, and L are given by:

$$C = \in_m \frac{\omega l}{d_C} = C_0 + \frac{\Delta C}{C_0} C_0 \tag{7}$$

where $\in_m$ is the dielectric permeativity, C is the capacitance matrix, C_0 is the nominal capacitance matrix, and $\frac{\Delta C}{C_0}$ is the normalized variation in the capacitance parameter.

$$R = \rho_C \frac{l}{\omega\tau} = R_0 + \frac{\Delta R}{R_0} R_0 \tag{8}$$

where ρ_C is the conductor resistivity, R is the resistance matrix, R_0 is the nominal resistance matrix, and $\frac{\Delta R}{R_0}$ is the normalized variation in the resistance parameter.

$$L = 2l\,[\ln\left(\frac{2l}{\omega+\tau}\right) + 0.5 - k] = L_0 + \frac{\Delta L}{L_0} L_0 \tag{9}$$

where k is a constant (given in cm in this formula only) that depends on ω and τ, and it is defined as $0 < k < 0.0025$ according to [28, 29]. L is the inductance matrix, L_0 is the nominal inductance matrix, and $\frac{\Delta L}{L_0}$ is the normalized variation in the inductance parameter.

By using equations 7, 8, and 9, we can define F and M in equation 6 as follows:

$$M = M_0 + p_1 M_1 + p_2 M_2 \tag{10}$$

$$F = F_0 + p_3 F_1 \tag{11}$$

where

$$M_0 = \begin{bmatrix} C_0 & 0 \\ 0 & 0 \\ 0 & L_0 \end{bmatrix}, \quad M_1 = \begin{bmatrix} C_0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & L_0 \end{bmatrix},$$

$$F_0 = \begin{bmatrix} 0 & F_{12}^{(0)} \\ -F_{21}^{(0)T} & R_0 \end{bmatrix}, \quad F_1 = \begin{bmatrix} 0 & F_{12}^{(1)} \\ -F_{21}^{(1)T} & R_0 \end{bmatrix}$$

and the parameters $p_i, i = 1,2,3$, are defined as:

$$p_1 = \frac{\Delta R}{R_0} s \quad , \quad p_2 = \frac{\Delta C}{C_0} s \quad , \quad p_3 = \frac{\Delta L}{L_0} s$$

Using the parametric model defined in equations 10 and 11, and substituting it into equation 4, we get:

$$(M_0 + p_1 M_1 + p_2 M_2)\, X(s) = -(F_0 + p_3 F_1)\, X(s) + PU(s) \tag{12}$$

By reducing the system using the congruence transformation with V, multiplying equation 12 by V^T from the left, and setting $X(s) = V\, Z(s)$, equations 12 and 5 become:

$$M(p)\, Z(s) \;=\; -G(p)\, Z(s) + V^T\, P\, U(s) \tag{13}$$

$$Y(s) \;=\; P^T\, V\, Z(s) \tag{14}$$

where $M(p) = V^T M_0 V + \sum_i p_i V^T M_i V$ and $F(p) = V^T F_0 V + \sum_i p_i V^T F_i V$

By putting, $\hat{M}_i = V^T\, M_i\, V$, $\hat{F}_i = V^T F_i\, V$,

and $\hat{M}(p) = \hat{M}_0 + \sum_i p_i\, \hat{M}_i$, $\hat{F}(p) = \hat{F}_0 + \sum_i p_i\, \hat{F}_i$, where $i = 1,2,3$, we get:

$$\hat{M}(p)\, X(s) \;=\; -\hat{F}(p)\, X(s) + \hat{P}\, U(s) \tag{15}$$

$$Y(s) \;=\; \hat{P}^T\, X(s) \tag{16}$$

From equations 15 and 16, the transfer function is given by:

$$H_q(s) = \hat{P}^T \left(\hat{F}(p) + \hat{M}(p)\right)^{-1} \hat{P} \tag{17}$$

4.3 Krylov sub-space

To reduce the input system, we must compute the Krylov subspace to construct V. We construct the input Krylov subspace which meets only the first q moments as proved by Daniel in [29]. The Krylov subspace is defined as:

$$K_q(A, B) \;=\; colspan\,\{B, AB, A^2B, ..., A^qB\} \tag{18}$$

The columns of V are composed of the Krylov subspace basis:

$$V \;=\; span\, K_q\,(A, B) \tag{19}$$

where A is the system matrix and B is the input matrix.

From equations 12 and 5, assuming that the matrix $(F_0 + p_0 M_0)$ is singular, our transfer function can be written in the form:

$$H(s) \;=\; P^T \left(R_0 + \sum_i R_i(p) \right)^{-1} P \tag{20}$$

where, $R_0 = F_0 + p_0 M_0$, $R_i(p) = p_i F_i$ or $R_i(p) = p_i M_i$,
and p_0 is a constant scalar value such that the matrix $R_0(p)$ is not singular.
By further simplification we get:

$$H(s) \;=\; P^T \left(I + (R_0)^{-1} \sum_i R_i(p) \right)^{-1} (R_0(p))^{-1} P \tag{21}$$

By setting the Krylov matrices as follows:

$$A = (R_0)^{-1} \sum_i R_i(p) \quad , \quad B = (R_0(p))^{-1} P \tag{22}$$

and substituting in equation 21, we get:

$$H(s) \;=\; P^T \, (I + A)^{-1} \, B \tag{23}$$

By expanding equation 23, we get:

$$H(s) \;=\; \sum_{i=0}^{q} A^i \, B \;=\; \sum_{i=0}^{q} m^{(i)} \tag{24}$$

where $m^{(i)}$ are the system moments:

$$m^{(i)} \;=\; \left[(R_0)^{-1} \sum_i R_i(p) \right]^i B \tag{25}$$

The recursive form for the moments of the system is defined as:

$$m^{(i)} \;=\; \sum_{k=0}^{i} (R_0)^{-1} \, R_{i-k}(p) \, m^{(k)} \tag{26}$$

where $m^{(0)} \;=\; (R_0)^{-1} \, B$

Therefore, the Krylov subspace spanned by V is constructed using a suitable Model Order Reduction (MOR) algorithm as Arnoldi [30], where:

$$\begin{aligned} spancol\,(V) \;&\supseteq\; spancol\,\{m^{(0)}, m^{(1)}, \ldots\ldots, m^{(q)}\} \\ &\supseteq\; spancol\,\{B, AB, A^2B, \ldots\ldots, A^q B\} \end{aligned} \tag{27}$$

4.4 Delay formula

For the transfer function in equation 17, the size of the matrices $\hat{F}_i$ and $\hat{M}_i$ is $q \times q$, where q has a maximum order of three. Using symbolic toolbox, the inverse is computed and the Pade' approximation [31] is obtained for the input-output voltage:

$$H_q(s) \;= \frac{V_{out}(s)}{V_{in}(s)} = \; k \, \frac{(1 + a_1 s + a_2 s^2)}{(b_0 + b_1 s + b_2 s^2 + s^3)} \tag{28}$$

where a_0, a_1, b_0, b_1 and b_2 are all functions of the parameters p_i (the maximum degree of any parameter term is two). To this point symbolic manipulation of the transfer functions stops, and approximate values of all the parameters are evaluated except for the Laplace valiable, s, and the target parameter, p, to be synthesized.

We can factorize the denominator of the passive system in hand, to compute the output response as follows:

$$V_{out}(s) = k\,\frac{(1+a_1 s + a_2 s^2)}{(s-\in_1)(s-\in_2)(s-\in_3)}\,V_{in}(s) \tag{29}$$

where $\in_i \in C$, are the poles/eigenvalues of the system and $\mathrm{Re}(\in_i) < 0$, $i = 1,2,3$.

We can have a ramp input or step input (or any other function, but taking in account computational overhead during routing). In case of a step function (just for illustration) and by using partial fractions, we can write equation 29 in the form:

$$V_{out}(s) \;=\; k\left(\sum_i \frac{\alpha_i}{s-\in_i} + \frac{1}{s}\right) \tag{30}$$

where the residues, $\alpha_i, i = 1,2,3$, are defined as:

$$\alpha_i \;=\; \frac{1 + a_1 \in_i + a_2 \in_i^2}{(\in_i - \in_j)\,(\in_i - \in_g)}\text{, where } i \neq j \neq g. \tag{31}$$

By returning to the time domain, we get:

$$V_{out}(t) \;=\; k\left(\sum_i \alpha_i\, e^{(\in_i t)} + 1\right) \tag{32}$$

Now, we have an equation where $\in_i$ and α_i are both parametric. To solve this equation, we first set $e^t = \dfrac{1}{1-1.88t}$:

$$V_{out}(t) \;=\; k\left(\sum_i \frac{\alpha_i}{(1-1.88\in_i t)} + 1\right) \tag{33}$$

$$\left(\prod_i (1-1.88\in_i t)\right)\frac{V_{out}}{k} \;=\; \sum_{i,j,g} \alpha_i (1-1.88\in_j t)(1-1.88\in_g t) \tag{34}$$

where $i \neq j \neq g$.

From the Theory of equations, we know that:

$$\in_1 + \in_2 + \in_3 = -b_2 \quad , \quad \in_1\in_2 + \in_2\in_3 + \in_1\in_3 = b_1 \quad , \quad \in_1\in_2\in_3 = -b_0$$

By setting $f(t,b_i) = (1 + 1.88^3 b_0 t^3 + 1.88^2 b_1 t^2 + 1.88 b_2 t)$ and substituting in equation 33, we get:

$$f(t,b_i)\,\frac{V_{out}(t)}{k} \;=\; \sum_{i,j,g} [\alpha_i (1-1.88\in_j t)(1-1.88\in_g t)] + f(t,b_i) \tag{35}$$

By substituting equation 31 in equation 35 and simplifying, we get:

$$f(t,b_i)\,\frac{V_{out}(t)}{k} \;=\; a_2 + 1.88\,a_1 t + 1.88^2 t^2 + f(t,b_i) \tag{36}$$

As we mentioned before, a_i and b_i are both functions of the second degree of p_i and t. By substituting for all the unknowns except one, we get a second/third degree equation in the geometrical/delay parameter, respectively. Solving the equation will yield the required value. Consequently, the 50% delay, t, or geometric constraint, p, can be computed from the given formula. For higher accuracies, we can use higher order formulas that give less error but require higher computational cost during runtime as given in Table 1.

Table 1: Different Formulas.

Formula	**Maximum error**	**Degree of t in the eq.**	**Degree of p in the eq.**
$e^t = 1/(1-1.88\,t)$	10%	3	2
$e^t = 1/(1-0.69\,t)^2$	5.6%	6	4
$e^t = 1/(1-0.41\,t)^3$	4%	9	6
$e^t = 1/(1-0.29\,t)^4$	3.2%	12	8

In the case where the reduced formula contained sine or cosine functions (complex powers), we consider the second order transfer function:

$$H_q(s) = k\left(\frac{1+a_1 s}{b_0 + b_1 s + s^2}\right) \;=\; k\left(\frac{1+a_1 s}{(s-\in_1)(s-\overline{\in}_1)}\right) \tag{37}$$

where each $\in_i$, $i = 1,2$, is a first degree equation of the parameters p_i.

From the Theory of Equations:

$$\in_1 + \overline{\in}_1 = -b_1 \qquad , \qquad \in_1 \overline{\in}_1 = b_0$$

Calculating the residues, $\alpha_i, i = 1,2$:

$$\alpha_1 \;=\; \left(\frac{1+a_1\,\in_1}{(\in_1 - \overline{\in}_1)}\right) \quad , \quad \alpha_2 \;=\; \left(\frac{1+a_1\,\overline{\in}_1}{(\overline{\in}_1 - \in_1)}\right) \;=\; \overline{\alpha}_1$$

The output function is:

$$\begin{aligned} V_{out}(t) \;&=\; k\;(\alpha_1 e^{(\in_1 t)} + \overline{\alpha}_1 e^{(\overline{\in}_1 t)} + 1) \\ &=\; k\left[\exp[(\in_1 + \overline{\in}_1)\frac{t}{2}]\,(f_{\cos}(t,p) - f_{\sin}(t,p)) + 1\right] \end{aligned} \tag{38}$$

where

$$f_{\cos}(t,p) = (\alpha_1 + \overline{\alpha}_1)\cos\left(i\,(\in_1 - \overline{\in}_1)\frac{t}{2}\right) = \alpha_1 \cos\left(\sqrt{4b_0 - b_1^{\,2}}\;\frac{t}{2}\right) ,$$

$$f_{\sin}(t,p) = i\,(\alpha_1 - \bar{\alpha}_1)\,\sin\left(i\,(\in_1 - \bar{\in}_1)\frac{t}{2}\right) = \frac{-(2-b_1\alpha_1)}{\sqrt{4b_0 - b_1^{\,2}}}\,\sin\left(\sqrt{4b_0 - b_1^{\,2}}\;\frac{t}{2}\right)$$

By substituting $e^t = \dfrac{1}{1-1.88\,t}$ (or any higher order formula), we get:

$$V_{out}(t) \;=\; k\left[\left(\frac{1}{1+0.94b_1\,t}\right)(\,f_{\cos}(t,p) - f_{\sin}(t,p)) + 1\right] \tag{39}$$

The derived equation is nonlinear in both t and the parameter p_i:

$$V_{out}(t)\,\frac{f_1(b,t)}{k} \;=\; f_{\cos}(t,p) - f_{\sin}(t,p) + f_1(b,t) \tag{40}$$

where: $f_1(b,t) \;=\; 1 - 0.94b_1\,t$

To solve this equation, since we are concentrated only with delay, so we approximate the sinusoidal functions till the point where they turn to zero(that is within the interval $[0,\frac{\pi}{2}]$):

$\cos t \;=\; 1 - 0.421t^2$, where $t \in [0,\frac{\pi}{2}]$ and the maximum error is 3.87%

and $\sin t \;=\; 0.9t - 0.1t^3$, where $t \in [0,\frac{\pi}{2}]$ and the maximum error is 4.88%

This leads to:

$$V_{out}(t)\,\frac{f_1(b,t)}{k} \;\approx\; \tilde{f}_{\cos} + \tilde{f}_{\sin} + f_1(b,t) \tag{41}$$

where

$$\tilde{f}_{\cos} = \alpha_1\left(1 - 0.421(4b_0 - b_1^{\,2})\frac{t^2}{4}\right),\quad \tilde{f}_{\sin} = (2-\alpha_1 b_1)\left(0.9\frac{t}{2} - 0.1(4b_0 - b_1^{\,2})\frac{t^3}{8}\right)$$

For higher order systems, we'll get equations containing higher degrees. Analytically, we can solve equations up to the fourth degree. As for higher degrees equations, Abel proved that there cannot be an analytical solution. In that case a numerical solution is required which will have a higher cost (knowing a suitable initial solution will reduce it greatly). Consequently, a similar derivation can be derived for a transfer function of higher q order.

4.5 Interconnect synthesis integration with routing module

As discussed before, the router, during its search, doesn't know the exact path, layer, or number of vias of the interconnect. Consequently, statistical assumptions and simple formulas are valid at this stage (but this is more accurate than these assumptions made at the global routing phase). To integrate IS within the router, we first assume a minimum width and manhattan path of the interconnect. According to the resulting solution, we calculate the delay of the system and the slack, which is the difference between the maximum allowed delay and the expected delay. If any time violation occurs, one of the layout optimization techniques [32,33] and the parameterized formulas provided from PMOR can be used to optimize the wire geometry. If a suitable optimization doesn't exist, then it is impossible to route the net using the manhattan path, even if exists, and re-placement is necessary. Next, the maximum allowed length of the interconnect is computed through linear extrapolation in the look-up table of formulas [34]. If the router can route the net accurately under the given constraints, with a large slack (say 50% of the allowed delay or more) then analysis is not required. Otherwise, either a rip-up and re-route is required or a more thorough analysis is required. In general, since the proposed algorithm is a sequential router, incorporating a rip-up and re-route strategy is crucial in completing the routing phase. The used strategy in this work was merely an implementation of the technique proposed in [11], which is used to find the minimum set of nets to be removed, with a very small overhead that is acceptable in practice. The algorithm that integrates the interconnect synthesis with routing module, which is called the Interconnect Synthesis driven routing algorithm, can be written in steps as follows:

1. Assume manhattan length and minimum width of the interconnect. Calculate the delay of the line in this case using the equations 36 and 41.
2. If the delay is greater than the maximum allowed delay, net can't be routed and re-placement is required.
3. Calculate the slack, slack = Maximum allowed delay – Delay for manhattan path.
4. If slack is less than 25%
   4.1 Calculate the required interconnect geometry constraints using the equations 36 and 41.
   4.2 After routing, a more detailed analysis of the line is required.
5. If slack is larger than 25%, then search through the table for the maximum allowed length that the slack remains less than 25%.
6. Route the net.
7. If the length constrained was not satisfied, check if changing the interconnect geometry is possible and re-route.
8. Otherwise, rip-up and re-routing is required to find a shorter path, using the technique in [11].
9. Go to the next net, and repeat the IS procedure.

4.6 Size of the reduced system

As proved in [35], for the parameters, n, the maximal size of the reduced individual matrices for any parameter, m_q, the order of the reduced parameterized system, q, is given by:

$$q = \frac{(m_q + p - 1)!}{m_q!\,(p-1)!} \tag{42}$$

For $m_q = 1$ and $p = 3$, the reduced system is of size $q = 3$. For higher accuracy, we can increase m_q, but to get a reduced system of reasonable size, one or two parameters is to be used. The proposed system here depends on three parameters: $\frac{\Delta R}{R_0}, \frac{\Delta L}{L_0}$ and $\frac{\Delta C}{C_0}$, other approaches can use other parameters. For variables that have few discrete values, as in d_C and τ which can have less than ten possible values (the maximum number of layers in current technologies), they can be added as an additional dimension to the look-up table. The resulting formulas give excellent accuracy and requires an adequate computational cost for interconnect synthesis.

4.7 Computational cost

One question remains unanswered: Can we really eliminate iterations using IS? In general, this highly depends on the accuracy of our synthesis approach. The higher the accuracy, the more computations done and the less number of iterations required. Thus, there is no unique synthesis approach and no best approach; it's merely a trade-off between a single run time, and the number of expected iterations. Using the proposed IS technique will greatly reduce the error in the computed delay of the routed layout, which in turns reduces the overall iterations. This will greatly depend on the accuracy of the interconnect models used. PMOR here was customized for calculating delay of the ladder circuits, yet the approach we took can be applied to other signal integrity issues and other types of multi-terminal circuits.

Before we give a computational analysis of the proposed technique for interconnect synthesis, we need to differentiate between on-line and off-line running. We use on-line running to refer to the computation done within the router inner routing loop. On the other hand, off-line routing is used to refer to PMOR computations done before routing process. The off-line computation involves mainly the computation of the reduced order parameterized transfer function in equation 17, then symbolically finding the time domain relation between the time delay and the interconnect geometry parameters. In this computation, the PMOR formulas are calculated once per technology and not for every layout, which is a very small cost in favor of the high accuracy attained. As for the on-line computation it will be dedicated to computing the delay of the interconnect, the suitable width for layout optimization, or the maximum allowed length. The cost of the on-line computations is no more than solving a third degree equation given in equation 36 or 41.

5. Target VLSI applications

Interconnects have proved to be a bottleneck in several applications. In Mentor Graphics Corporate (MGC) Egypt, the Intellectual Property Division (IPD) is facing typical interconnect problems with every tape out. To overcome these problems, a research is conducted in the Intellectual Property Department (IPD) to reach a long term solution for interconnects in the DSM technologies. The work in this paper proposes the concept of IS and implements it in a detailed router as a part of a complete back end flow for physical synthesis. The target applications that the IPD team is mixed signal Application Specific IC (ASIC) designs, and in this work we concentrate on high speed digital VLSI circuits. These circuits have the following specifications:

1. Number of gates 10-20 Kgate (kilo-gate). Although designs in the industry reaches 4-10 MGates (mega-gate).
2. Area is less than $1\,mm^2$, which is typically 1 – 2 cm^2.
3. Clock frequency starts from 500 MHz and reaches GHz ranges in some applications.
4. Globally Synchronous Digital circuits.
5. Standard cell ASIC layouts.
6. Static CMOS logic families. We don't support dynamic logic or differential pair circuits (we don't support matched length constraints).
7. Routing constraints don't include power integrity constraints. We focus on timing on and signal integrity taking in account RLC effect of interconnects at such high frequencies.
8. DSM manufacturing technologies (Nanotechnologies).

Although the target digital circuits are smaller than typical ASIC designs, yet our approach can be extended to such large complex designs by including one or more global routing stages before the detailed router.

6. Simulation Study

To evaluate the efficiency of the proposed routing approaches and interconnect synthesis (IS) methodology, and their effectiveness to help the router in its task, we conducted extensive simulation study on the VLSI routing model, which was illustrated in section 2. An advantage of the simulation results is that they have been conducted on industrial chips. These chips were mainly mixed signal chips, where the digital part that had very high frequencies (order of GHz).

In the first part of this study, we considered the proposed interval labeling routing algorithm that was implemented from scratch. The algorithm is a basically a variant of Dijkstra's shortest path algorithm that uses BFS search strategy [20]. We defined the various parameters defining the search graph as:

1. **Graph nodes**: The intervals that are subset of line segments.

2. **Graph edges**: The fractional cascading pointes.
3. **Edge label**: The detour value, yet other enhanced cost functions could be used as in [33] for a time driven routing.

The core of our system is the routing algorithm. It communicates with OpenAccess for file Input/Output operations. These data are saved in a suitable data structure managed by OpenAccess. The graph that represents the layout is then constructed using the following data structures:

1. **Priority Queue:** This data structure is used in various data, like storing the nets, labels of an interval, and the labeled intervals.
2. **1D bucketing data structure:** This is used to save the intervals being labeled. Each bucket of the data structure is a priority queue for a given label value. In each priority queue, nets are sorted according to their distances from the target for the Breadth First Search (BFS) strategy [20].
3. **The modified STree:** Each row of line segments is saved in a separate modified STree. The adjacent STrees, are connected through the fractional cascading pointers.

The algorithm proceeds in searching and connecting interconnects. In case that a net can't be routed, the rip up and re-route module [11] is called to rip up the obscuring nets, and re-insert them in the proper position in the net priority queue. Finally, the layout is saved by OpenAccess in its internal format, yet other standard formats can be generated from OpenAccess, such as GDS file, using the utilities offered with the Library of Efficient Data Structure and Algorithms (LEDA) [36].

The interval labeling routing algorithm was developed on a Sun Solaris platform. The algorithm was designed and implemented using an Object Oriented paradigm in ANSI C (this makes porting to other platforms a simple task). The development environment we used was:

1. Operating System (OS) platform: SunOS 5.51
2. Hardware (HW) platform: Single processor sparc processor operating at 380MHz.
3. Machine HW: SUNW, Ultra_5_10.
4. Integrated Development Environment: Sun Studio 11.
5. Compiler: Sun C++ compiler version 5.8.

The algorithm implemented can be used for different types of designs, yet the test cases we used were only standard cell based layouts. We have two main layout examples in TSMC 13 technology. The first is a simple small circuit of buffers for the purpose of development, which was used at the first phases of development. This example has an order of a hundred chips and nets. The other example is the digital block of the new Universal Serial Bus (USB) circuit. The functionality will not affect the running of the algorithm, yet we're more concerned with the physical properties of the layout as shown in Table 2.

Table 2: Chip sizes.

	Operating frequency	**Area**	**Order of Number of Gates/Nets**
Chip 1	–	Less than 0.01 mm^2	100
Chip 2	500 MHZ	$1mm^2$	1 kilo-gate

We compare our routing algorithm with Silicon Ensemble [37], one of the well-known industrial routing tools. In our testing we used two main layouts. Our comparison will be in terms of performance mainly. By analyzing our simulation results in Table 3, we can see that in the first example the two routers have very similar running time. This returns to the overhead of the initializations done in the algorithm on both sides and the small running time. In the second example the difference performance becomes more tangible and we can see an overall out-performance of our algorithm.

Table 3: Comparing routing times.

	The proposed router alg.	Silicon Ensemble
Chip 1	3 minutes	4 minutes
Chip 2	45 minutes	1 hour

In the sense of the physical design process in overall, the second example took several iterations before we reached the final routing stage which we include here in our results. On the other hand, we expect that the incorporation of the interconnect analysis and synthesis module with our router will perform much better. On the level of a single routing iteration, we expect that our router would be slower as the speedup achieved will be utilized in the interconnect analysis and synthesis. Consequently, we expect our algorithm will speed up the overall routing process and not a single iteration.

The second part of this study conducted on the Interconnect Synthesis driven routing algorithm. As discussed before, we have two phases for IS. The first phase is done before routing, and is done once per technology to generate the synthesis formulas. We refer to this phase as off-line routing. The second phase, interconnect synthesis, is done during routing to synthesize the different routing parameters, which we refer to as on-line routing. During the first phase, we concentrate on accuracy and ignore the running time. We compare the accuracy of derived formulas with Eldo [38] simulations and show the range of validity. The simulation of the interconnect was done on two environments. Eldo simulations were conducted on a Unix platform:

1. OS platform: Linux Redhat.
2. HW platform: Single processor Intel processor operating at 3.19GHZ.
3. Eldo v6.3.
4. Calibre PEX v9.3

The PMOR simulations were performed on a Windows platform:

1. SW platform: Windows XP.
2. HW platform: Single processor Intel processor operating at 3.19 GHZ.
3. Mathematica v5.1.
4. Cygwin v5.1.

In this case, the main objective is to test the accuracy of the generated IS formulas derived through PMOR. To do that, we need to build the look-up table [34] of the synthesis formulas. This approach allows us to achieve higher accuracy on the expense of a very high memory cost and a large computational overhead of pre-routing simulations. In general, this method tries to define the different parameters that affect the delay along the interconnect and builds a n-dimensional table that includes the delay. Therefore, in this study, we proceed as follow:

1. We start by a parametric model for the interconnect, where we use simple models as those suggested in equations 2 and 3. The nominal values of R_0, L_0, and C_0 are provided through TSMC13 manual.
2. In the suggested models, we have three parameters, which are $\frac{\Delta R}{R_0}, \frac{\Delta L}{L_0}$, and $\frac{\Delta C}{C_0}$. The range of these parameters for a single interconnect segment of length $100\mu m$ is from zero to three. This means that we assume the parameters can increase four times of their original values.
3. We use a PMOR system [30] that is implemented on Mathematica to reduce the system to order two and three PMOR models.
4. We repeat this step for different interconnect lengths, where the actual interconnect length is defined in terms of the number of segments. The number of segments also determines the order of the original system before PMOR.
5. We use symbolic manipulation to drive the parametric Pade' approximation as in equation 28.
6. Finally, the synthesis formulas are directly derived using the equations 36 and 41.

An extensive simulation study was made in two directions. Results in Table 4 and Table 5 are considered as a sample of this numerical study. In the first direction, we compared the delay of the synthesis table to Eldo simulations for interconnects with different number of segments (length values), while maintain the width to minimum value. The errors in these cases are shown in Table 4. In these result, one segment is equivalent to $100\mu m$ length. In the second part of simulations, we simulated the interconnect for different number of segments and width (from minimum w up-to $4w$) using Eldo. Then for each structure, we calculated the optimal width where the delay is set to that calculated by Eldo. The errors in the optimal width are found in Table 5.

Table 4: Error in delay.

	20 segment	60 segment	100 segment
Minimum width w	1%	4%	9%
Width of 2 w	3%	5%	12%
Width of 3 w	4%	7%	18%
Width of 4 w	8%	8%	20%

Table 5: Error in optimal width.

	20 segment	60 segment	100 segment
Minimum width w	1%	2%	5%
Width range 2 w	2%	3%	6%
Width range 3 w	3%	4%	7%
Maximum width 4 w	5%	5%	8%

Results in Tables 4 and 5 shows that for nominal widths we can get very excellent results, yet as we get further away the error increases dramatically. At maximum length the error becomes very inaccurate. This is mainly because interconnects at these lengths (10 mm) are not accurately modeled using lumped RLC elements. Also, for very wide interconnects of width 4w we find that the error is also inadequate. Hence, we conclude that the look-up table could only be used for wires of moderate length with excellent accuracy. Also, the relative variation in the parameter should be kept as close as possible to unity for sufficient accuracy.

7. Conclusion

Interconnects play a major role in the design flow at every stage. In fact, new design flows have been proposed to replace the current flow to overcome this problem. This idea might appear appealing, but converting to a totally new flow is not that practical. A more practical approach is to modify the current flow to allow the support of interconnects at different level of abstraction and with the absence of complete information to produce accurate results. In this paper, we proposed a high speed router and a more accurate interconnect characterization methodology than the traditional delay and signal integrity formulas. Through a deep understanding of the concept of Interconnect Synthesis (IS), a new approach is presented to solve the new challenges accompanied by interconnects. In this approach, we persuaded a more solid and mathematical path. Understanding that we are facing a well-known orthogonal segment intersection problem, we developed a suitable data structure and algorithm. The IS was incorporated in the detailed routing phase. Yet it should be employed in the rest of the physical design phases for higher accuracy. If we carefully examine the proposed approach, we can see that we have two contradicting demands. A routing algorithm that has a very high processing requirement, and an interconnect synthesis module that has an increasing demand in processing requirements. Hence, a trade off is an adequate solution where we have increased the speed of the algorithm on the expense of its memory usage leaving more room for the IS module for its complex computation. In overall, the speedup in algorithm and improvement in accuracy of the synthesis formulas are promising. Using a single synthesis formula for all possible cases and independent of the used technology encouraged us to choose an efficient simulating technique, Parametric Model Order Reduction (PMOR), that had low computational complexity and yielded accuracy of very high order. This technique reduces the system to a smaller one that has equivalent behavior and allows us to defined the synthesis formulas in terms of the interconnect geometric parameters. Then, by using simple approximations we can derive a lookup table of simple formulas (not delay values) that could be used inside the routers internal loop. Results of a numerical study, using a large number of test cases, showed that the proposed approach is capable to yield better results that greatly reduces the error in the computed delay of the routed layout, which in turn reduces the overall iterations, and minimizes the overall running time, relative to the previous traditional approaches, for a wide variety the system parameters.

8. Open Problem

The topics we have handled in this work were not completely covered. There are several unanswered questions that remain open and deserve to be researched. In the routing algorithm, for example: What is the order of nets that should be used in routing [19]? Can we improve the rip-up and re-route strategy we are using [11]? How can we improve the memory requirements of our algorithm? Can we estimate congestion in the cost function [39]? How can expert rules be embedded to improve the quality of the routed layout [40]?How can we handle variable width in routing [41]?

On the other hand, in PMOR we have several unanswered questions like: How can to extend our approach further on the other signal integrity issues, such as crosstalk, attenuation, signal reflection and ringing [5, 15, 16, 18]? What are the error bounds for our approach as there is no proof for error bounds of Krylov subspaces? Can we extend our approach to solve higher degree formulas without numerical iterations? Other ideas come on mind and deserve to be researched like: Complex frequency hoping and Transmission line PMOR [42]. It's obvious; this point has opened a door for more questions that deserves the attention of researcher in this field. Finally, a more admiring question presents itself: Can we incorporate interconnect synthesis in other phases of the IC flow? How can this be done? All these questions are left as open questions for further research and work.

ACKNOWLEDGEMENTS

The authors would like to thank the referees for their highly valuable comments and suggestions that greatly helped to improve the earlier version of this work. All of their suggestions were incorporated directly in the text.

References

[1] Semiconductor Industry Association. International Technology Roadmap for Semiconductors. ITRS. Available:http://www.itrs.ntrs /publntrs.nsf.

[2] C. Peterson. Risks and opportunities in ASIC design. Intel Developer update Magazine, Febraury 2002.

[3] S.P. Khatli, Robert K. Brayton, A. Mehrotra, Alberto L. Sangiovanni-Vincentelli, and M.R. Prasad. Routing techniques for deep sub-micron technologies. Technical Report UCBIERL M99/15, EECS Department, University of California, Berkeley, 1999.

[4] S. Bhingarde, S. Madhwapathy, A. Panyam, and N. Sherwani. An efficient four layer over-the-cell router. Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS '94), pp. 187-190, 1994.

[5] J. Cong, J. Fang, and Y. Zhang. Multi-level approach to full-chip gridless routing. Proc. lEEE/ACM Int. Conf. on Computer-aided design (lCCAD '01), Piscataway, NJ, USA, pp. 396-403, 2001.

[6] J. Cong, M. Xie, and Y. Zhang. An enhanced multi-level routing system. Proc. IEEE/ACM Int. Conf. on Computer-aided design (ICCAD '02), New York, NY, USA, pp. 51-58, 2002.

[7] T.-Y. Ho, Y.-W. Chang, S.-J. Chen, and D. T. Lee. A fast crosstalk-and performance-driven multi-level routing system. Proc. IEEE/ACM Int. Conf. on Computer-aided design (ICCAD '03), Washington, DC, USA, pp. 382-389, 2003.

[8] S.-P. Lin and Y.-W. Chang. A novel framework for multi-level routting considering routability and performance. Proc.lEEE/ACM Int. Conf. on Computer-aided design (ICCAD '02), New York, NY, USA, pp. 44-50, 2002.

[9] Z. Xing and R. Kao. Shortest path search using tiles and piecewise linear cost propagation. IEEE Trans. on CAD of Integrated Circuits and Systems, 21 (2):145-158, 2002.

[10] T. Stohr, M. Alt, A. Hetzel, and J. Koehl. Analysis, reduction and avoidance of crosstalk on VLSI chips. Proc. Int. Symp. on Physical design (lSPD '98), New York, NY, USA, pp. 211-218, 1998.

[11] A. Hetzel. A sequential detailed router for huge grid graphs. Proc. Conf. on Design, Automation and Test in Europe (lSPD '98), Washington, DC, USA, pp. 332-339, 1998.

[12] E. Kuh and T. Ohtsuki. Recent advances in VLSI layout. IEEE Proceedings Special Issue on Computer-Aided Design, 78(2):237-263, Feb., 1990.

[13] A. B. Kahng, K. Masuko, and S. Muddu. Analytical delay models for VLSI interconnects under ramp input. Proc. IEEE/ACM Int. Conf. on

Computer-aided design, San Jose, California, United States, pp. 30-36, 1996.

[14] D. Zhou and R.-M. Li. Design and verification of high-speed VLSI physical design. Computer Science and Technology, 20(2):147-165, 2005.

[15] R. Venkatesan, J. A. Davis, and J. D. Meindl. Compact distributed RLC interrconnect models - part iv: Unified models for time delay, crosstalk, and repeater insertion. IEEE Trans. Electron Devices, 50: 1094-1102, 2003.

[16] J. Xiong and L. He. Full-chip routing optimization with RLC crosstalk buddgeting. IEEE Trans. on CAD of Integrated Circuits and Systems, 23(3):366-377, 2004.

[17] K. Baneljee and A. Mehrotra. Accurate analysis of on-chip inductance effects and implications for optimal repeater insertion and technology scaling. Proc. IEEE Symp. on VLSI Circuits, pp. 195-198, 2001.

[18] X. Huang, Y. Cao, D. Sylvester, T. King, and C. Hu. Analytical performance models for RLC interconnects. Proc. IEEE International ASIC-SoC Conference, Sep., 2002.

[19] P. Groeneveld. Wire ordering for detailed routing. IEEE Design & Test of Computers, 6(6):6-17, Dec., 1989.

[20] E. W. Dijkstra. A discipline of programming. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1997.

[21] W. Lipski and F. Preparata . Segment, rectangles, contours. Algorithms, 2: 63-76, 1985.

[22] B. Chazelle. Filtering search: a new approach to query answering. SIAM J. Computing, 15(3):703-724, 1986.

[23] N. Sarnak and R. E. Tarjan. Planar point location using persistent search trees. Communications of the ACM, 29:669-679, 1986.

[24] R. Dementiev, L. Kettner, J. Mehnert, and P. Sanders. Engineering sorted list data structure for 32 bit key. Proc. 6th Workshop on Algorithm Engineering and Experiments and the 1st Workshop on Analytic Algorithmic and Combinatory (ALENEX-04), New Orleans, LA, USA, pp. 142-151, Jan., 2004.

[25] F. Rubin. The Lee path connection algorithm. IEEE Trans. Computers, C-23:907-914, 1974.

[26] G. Shi and C.J.Richard Shi. Parametric reduced order modeling for interconnect analysis. Proc. Conf. on Asia South Pacific Design Automation (ASP-DAC '04), Piscataway, NJ, USA, pp. 774-779, 2004.

[27] I. Sutherland, B. Sproull, and D. Harris. Logical effort: designing fast CMOS circuits. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[28] L. Feng. Parameter independent model order reduction. Mathematics and Computers in Simulation, 68(3):221-234, 2005.

[29] L. Daniel, O.C. Siong, L.S. Chay, K.H. Lee, and J. White. Multi-parameter moment matching model reduction approach for generating geometrically parameterized interconnect performance models. Trans. on Computer Aided Design of Integrated Circuits, , May 2004.

[30] E. B. Rudnyi and J. G. Korvink. Model order reduction for large scale engineering models developed in ANSYS. Lecture Notes in Computer Science, 3732:349-356. 2006.

[31] P. Feldmann and R. W. Freund. Efficient linear circuit analysis by Pade' approximation via the Lanczos process. Proc. Conf. on European design automation (ASP-DAC '04), Los Alamitos, CA, USA, pp. 170-175, 1994.

[32] J. Cong and Z. Pan. Interconnect performance estimation models for design planning. IEEE Trans. on Computer-aided design, 20:739-752, Jun. 2001.

[33] J. Cong, L. He, K.-Y. Khoo, C.-K. Koh, and D. Z. Pan. Interconnect design for deep submicron ICs. Proc. Int. Conf. on Computer-aided design (ICCAAD '97), pp. 478-485, 1997.

[34] J. Lillis and P Buch. Table look-up methods for improved performance driven routing. Proc. 35th annual Conf. on Design Automation (DAC '98), New York, NY, USA, pp. 368-373, 1998.

[35] L. Daniel, O.C. Siong, L.S. Chay, K.H. Lee, and J. White. Multi-parameter moment matching model reduction approach for generating geometrically parameterized interconnect performance models. Trans. on Computer Aided Design of Integrated Circuits, 23(5):678-93, May 2004.

[36] K. Mehlhorn, S. Naher, and C. Uhrig. The LEDA platform of combinatorial and geometric computing. Proc. 24th Int. Colloquium on Automata, Languages and Programming (ICALP '97), London, UK, pp. 7-16, 1997.

[37] Cadence Design Systems, Inc. Silicon Ensemble Getting Started Guide.

[38] Mentor Graphics Corporate, Inc. Eldo User's Manual, 1998.

[39] C. W. Sham and E. F. Young. Congestion prediction in early stages. Proc. 5th Int. Workshop on System level interconnect prediction (SLIP '05), New York, NY, USA, pp. 91-98, 2005.

[40] R. Jobbani and D. P. Siewiorek. Weaver: a knowledge-based routing expert. In DAC '85: Proc. 22nd ACM/IEEE Conf. on Design automation, New York, NY, USA, pp. 266-272, 1985.

[41] Y.M Kureichik, B.K.Lebedev, and E.Y. Nuzhnov. A maze router for tracks with diffferent width. IEEE Int. Conf. on Artificial Intelligence Systems, pp.407-410, Sep., 2002.

[42] P. Gunupudi, R. Khazaka, and M. Nakhla. Analysis of transmission line circuits using multi-dimensional model reduction techniques. IEEE Trans. on Advanced Packaging, 25(2):174-180, May 2002.