Timing-Driven Routing for Symmetrical Array-Based FPGAs

YAO-WEN CHANG
National Chiao Tung University
KAI ZHU
Triscend Corp.
and
D. F. WONG
University of Texas at Austin

In this paper we present a timing-driven router for symmetrical array-based FPGAs. The routing resources in the FPGAs consist of segments of various lengths. Researchers have shown that the number of segments, instead of wirelength, used by a net is the most critical factor in controlling routing delay in an FPGA. Thus, the traditional measure of routing delay on the basis of geometric distance of a signal is not accurate. To consider wirelength and delay simultaneously, we study a model of timing-driven routing trees, arising from the special properties of FPGA routing architectures. Based on the solutions to the routing-tree problem, we present a routing algorithm that is able to utilize various routing segments with global considerations to meet timing constraints. Experimental results show that our approach is very effective in reducing timing violations.

Categories and Subject Descriptors: B.7.1 [**Integrated Circuits**]: Types and Design Styles—*Gate arrays*; B.7.2 [**Integrated Circuits**]: Design Aids—*Placement and routing*; J.6 [**Computer Applications**]: Computer-Aided Engineering

General Terms: Algorithms, Design, Experimentation, Measurement, Performance

Additional Key Words and Phrases: Computer-aided design of VLSI, field-programmable gate array, layout, synthesis

The work of Yao-Wen Chang was partially supported by the National Science Council of Taiwan, ROC, under grant NSC-87-2215-E-009-041.
The portion of the work done by Kai Zhu in this paper was done while he was with The University of Texas at Austin.
The work of D. F. Wong was partially supported by the Texas Advanced Research Program under grant 003658288.
Authors' addresses: Y.-W. Chang, Department of Computer and Information Science, National Chiao Tung University, Hsinchu, 300, Taiwan; email: ywchang@cis.nctu.edu.tw; K. Zhu, Triscend Corp., 301 N. Whisman Rd., Mountain View, CA 94043; email: zhu@triscendcorp.com; D. F. Wong, Department of Computer Sciences, University of Texas at Austin, Austin, TX 78712; email: wong@cs.utexas.edu.

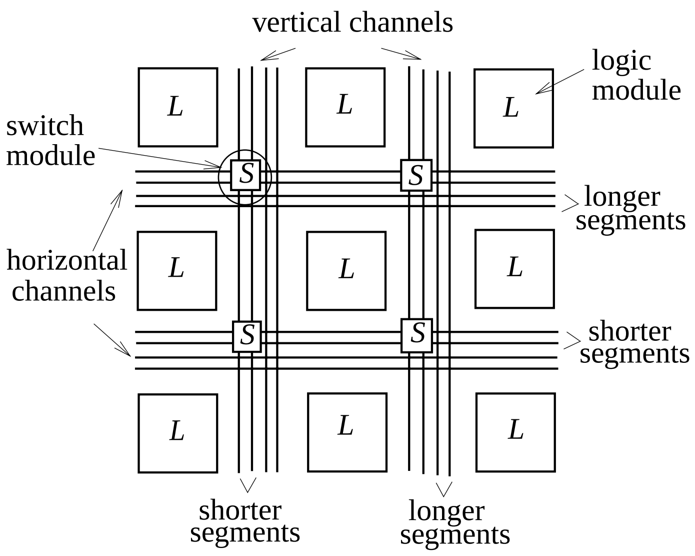

Fig. 1. The symmetrical array-based FPGA model.

1. INTRODUCTION

Symmetrical array architecture is used in several popular commercially available FPGAs [Lucent Technologies 1996; Xilinx 1996]. A symmetrical array-based FPGA consists of two-dimensional array logic modules interconnected by vertical and horizontal routing channels [Brown et al. 1992] (see Figure 1 for a typical FPGA model). The logic modules, denoted by L in Figure 1, can be customized to realize logic functions. The routing channels comprise general routing resources used to connect logic-module pins. An intersection area of horizontal and vertical routing channels is referred to as a *switch module*, denoted by S in Figure 1. The switch modules house programmable switches. FPGA routing uses programmable switches (inside S) to make connections. The switches usually have high resistance and capacitance, and thus incur significant delays. To improve circuit performance and maintain reasonable routability simultaneously, the routing channels are usually segmented, and thus routing tracks consist of wires with a versatile set of lengths [Xilinx 1996] (see Figure 2). Longer segments are intended for high fan-out, time-critical signal nets. On the other hand, shorter segments are intended for short connections so as to avoid wasting routing resources. To achieve the goal of improving circuit performance without much sacrifice on routability, it is essential for a routing algorithm to utilize these various routing segments effectively.

Many routing algorithms for symmetrical array-based FPGAs are reported in the literature [Alexander et al. 1995; Alexander and Robins 1995; Brown et al. 1992; Chang et al. 1994; Chen et al. 1995; Frankle 1992; Lee and Wu 1995; Lemieux and Brown 1993; Lemieux et al. 1997; McMurchie and Ebeling 1995; Palczewiski 1992]. Most of these algorithms, however, either aim at routability only, without considering timing constraints

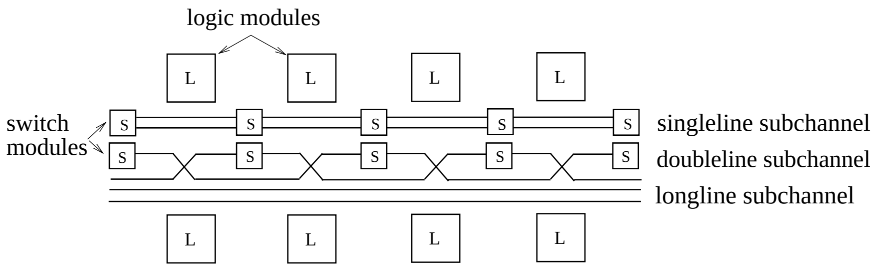

Fig. 2. Detailed structure of a routing channel..

[Brown et al. 1992; Chang 1994; Palczewiski 1992], or consider only one type of wire segment [Alexander 1995; Alexander and Robins 1995; Brown et al. 1992; Chang et al. 1994; Chen 1995; Lee and Wu 1995; Lemieux et al. 1997]. Research on timing-driven routing for symmetrical array-based FPGAs with multilength segments is very limited. An FPGA routing algorithm that considers routability, delay, and multilength segments is described in Lemieux and Brown [1993]. The algorithm is an extension of the detailed routing algorithm of Brown et al. [1992] to handle the selection of different segments for detailed routing. However, a drawback to the approach in Lemieux and Brown [1993] is that the choice of segments is restricted to the routing paths defined by a conventional global router, and thus it is hard for the router to utilize the segments efficiently.

Researchers have shown that, due to the segmented architectures of FPGA routing channels, the number of segments, instead of wirelength, used by a net is the most critical factor in controlling routing delay in an FPGA [Fallah and Rose 1992; Khellah et al. 1993]. Thus, the traditional measure of routing delay on the basis of the geometric distance of a signal is not accurate for the FPGA with multiple segment lengths. For example, in Figure 3, both nets 1 and 2 have the same geometric distance. However, the signal delay of net 2 is significantly larger than that of net 1 because net 2 uses more switches to make the connection.

To consider wirelength and delay simultaneously, we study a model of a timing-driven routing-tree problem, arising from the special properties of FPGA routing architectures. Based on the solution to the routing-tree problem, we present a timing-driven routing algorithm for symmetrical array-based FPGAs. The routing algorithm has the following distinct features, when compared to previous work:

—Utilization of routing resources is considered from a global viewpoint. The routing algorithm uses a hierarchical top-down approach and considers all available routing resources in assigning routing paths to nets.

—The algorithm explicitly includes timing constraints as its inputs and aims at routing all the nets to meet the constraints. To perform timing-driven routing, the timing constraints are transferred into the delay bounds on the source-to-sink paths of nets. Further, the routing algo-

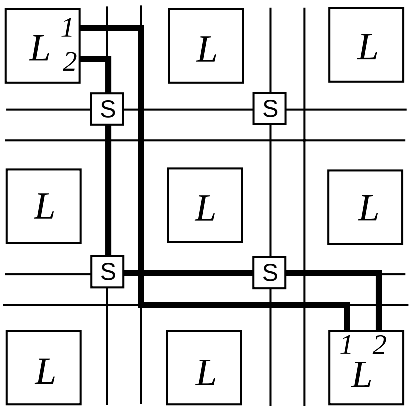

Fig. 3. Routing on different types of segments. Net 2 has larger signal delay than Net 1 because it uses more switches.

rithm can be used as a direct follow-up to timing-driven placement [Hauge et al. 1987; Marek-Sadowska 1993].

—Detailed routing is performed simultaneously with global routing. All nets are routed at the same time, and thus the net-ordering problem can be avoided during both global and detailed routing stages.

Experimental results show that our timing-driven router substantially reduces the number of connections violating timing constraints.

The rest of the paper is organized as follows. Section 2 gives the formulation of FPGA routing architectures and the timing-driven routing. Section 3 describes all ingredients in the timing-driven routing. Finally, experimental results are presented in Section 4.

2. PROBLEM FORMULATION

A horizontal (vertical) channel is the routing space between two adjacent rows (columns) of logic modules. A routing channel is divided into a set of *subchannels*. Subchannels are distinguished by the relative lengths of their segments, and each subchannel consists of a set of equal-length routing segments. Figure 2 shows an example of a horizontal channel with three subchannels, namely *single-line subchannel*, *double-line subchannel*, and *long-line subchannel*. The single-line subchannel consists of single-length segments that form a grid of horizontal and vertical lines that intersect at switch modules. The double-line subchannel contains double-length segments composed of a grid of segments twice as long as the single-length ones. The long-line subchannel contains segments that run the entire vertical or horizontal channels. This model of channel segmentation is used in popular commercial FPGAs [Lucent Technologies 1996; Xilinx 1996].

In timing-driven physical layout, the timing constraints can be transferred into the delay bounds on nets [Frankle 1992; Hauge et al. 1987]. Every net consists of a *source* pin and a set of *sink* pins. The timing constraints on a net are specified as the delay bound from the source to each sink. The goal of timing-driven routing is thus to route the nets so that delay constraints are satisfied.

It is shown in Khellah et al. [1993] that the number of segments (and thus the number of switches) a net has to pass through is the most critical factor in controlling routing delay in an FPGA. The delay model based on the number of switches used by a net is sufficiently accurate for the purpose of timing-driven routing for FPGAs (see Zhu and Wong [1998] for examples of transferring timing constraints into the upper bounds on the numbers of switches used by source-to-sink paths of nets based on the Elmore delay model [Elmore 1948]). Note that the number of switches along the routing paths of the net is known if the global routing paths of a net are specified in terms of subchannels.

We formulate the timing-driven routing problem for the symmetrical array-based FPGAs as follows.

—*The FPGA timing-driven routing problem (FPGA-TRP)*

Instance: Given an FPGA architecture, a netlist of the circuit, and timing constraints specified as the upper bounds on the number of switches used along all source-sink pairs of nets.

Objective: Determine the sequence of wire segments and switches for each net so that the net delay bound is satisfied.

3. TIMING-DRIVEN ROUTING ALGORITHM

3.1 Overview

The routing algorithm is based on the hierarchical top-down strategy for traditional ASICs [Lauther 1987; Marek-Sadowska 1986; Suaris and Kedem 1989], with several distinctive features to deal with the special properties of FPGA routing architectures. We first review the generic hierarchical approach and then present the unique features of our router. The generic hierarchical approach proceeds as follows. Starting with the entire circuit, a cut line is chosen in some way (such as min-cut [Breuer 1977; Suaris and Kedem 1989]) to divide the circuit into two parts. Across the cut line there are *routing sections* that represent routing spaces on the chip. (See Figure 4 for a clarification of the terminology.) A routing section is usually a channel with a routing capacity specified by the channel width. Each connection crossing the routing section is assigned a cost defined by total wire length and channel densities. The routing at each hierarchical level assigns connections to routing sections based on a particular algorithm, such as the linear assignment method [Papadimitriou and Steiglitz 1982]. After finishing routing at this hierarchical level, both subcircuits separated by the cut line are routed by the same method recursively. The algorithm terminates when the subcircuits are small enough so that they can be solved easily.

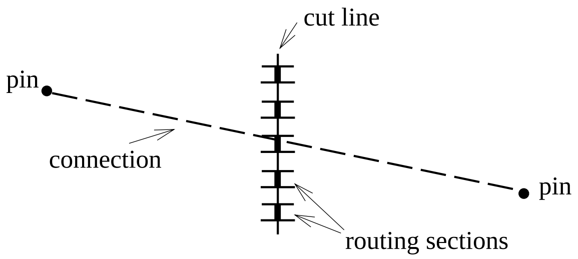

Fig. 4. Illustration of terminology in the hierarchical routing.

To deal with the timing-driven routing for symmetrical array-based FPGAs, we add the following distinctive features to the aforementioned generic hierarchical approach:

(1) A routing section on a cut line is a subchannel instead of a channel. So the router will automatically choose appropriate subchannels (and thus segments) for nets at each hierarchical level.

(2) A routing tree is constructed for each net. Here, a routing tree of a net is a spanning tree connecting all pins of the net, subject to the constraints that all paths from the source to sinks must satisfy their delay bounds. In Section 3.2 we formulate a timing-driven routing-tree problem to consider the constraints.

(3) After constructing a routing tree, the net delay bounds need to be distributed among the edges of the routing tree in order to perform hierarchical routing. The algorithm for distributing delay bounds is described in Section 3.3.

(4) A new cost function for linear assignment should be defined to consider timing constraints. We define the cost function in Section 3.4.

(5) At each hierarchical level, the delay bound for a connection crossing a cut line needs to be redistributed among subconnections (i.e., each subconnection gets its own delay bound) for the next level of hierarchical routing. The redistribution of delay bounds should facilitate routing at subsequent hierarchical routing levels. Delay-bound redistribution is discussed in Section 3.5.

(6) Detailed routing is performed simultaneously with global routing, and is discussed in Section 3.6.

3.2 Timing-Driven Routing Trees

The goal of timing-driven routing is to construct a routing tree for each net such that all timing constraints are satisfied and the routing cost is minimized (to save routing resources and reduce capacitance). Routing costs usually include wire lengths and routing congestion. As in conventional technology, geometric distance is a suitable measurement for routing cost. Due to the segmented structure of routing resources in FPGAs, however, routing delay is generally not proportional to the geometric distance. Thus, the traditional measure of routing delay that is based on the geometric distance of a signal is not accurate for an FPGA with multiple segment lengths. To perform timing-driven routing, we formulate the following routing-tree problem with two *independent* weights, one for routing cost and the other for routing delay.

—*The bounded-delay minimum-cost spanning-tree problem* (*BDMCSTP*)

Instance: Given a graph $G = (V, E)$, $V = \{s, t_1, \ldots, t_{|V|-1}\}$, a routing cost function $r(e) \in \mathbf{Z}^+$, a delay cost function $d(e) \in \mathbf{Z}^+$ for each $e \in E$, and a delay bound $B(t_i) \in \mathbf{Z}^+$ for each simple path from s to t_i.

Objective: Find a spanning tree T for G satisfying $\Sigma_{e \in p(s, t_i)} d(e) \leq B(t_i)$ for every simple path p from s to t_i in T such that $\Sigma_{e \in T} r(e)$ is minimized.

The decision version of the BDMCSTP is described below:

—*The bounded-delay bounded-cost spanning-tree problem* (*BDBCSTP*)

Instance: The instance of the BDMCSTP, with an additional routing cost bound $C \in \mathbf{Z}^+$.

Question: Is there a spanning tree T for G satisfying $\Sigma_{e \in p(s, t_i)} d(e) \leq B(t_i)$ for every simple path p from s to t_i in T such that $\Sigma_{e \in T} r(e) \leq c$?

3.2.1 *Complexity of the Routing-Tree Problem.* In this section we discuss the complexity of the above problems. Work on spanning-tree construction with simultaneous multiple objective optimization (e.g., delay and wire length) is reported in Awerbuch et al. [1990]; Cong et al. [1992]; Ho et al. [1991]; and Khuller et al. [1993]. While the literature deals with the case where the objectives are mutually *dependent* (e.g., delay and wire length are both measured by geometric distance), our formulation considers the case where the objectives are *independent*, which, as mentioned earlier, is essential for timing-driven routing on the FPGA architecture with multiple segment lengths.

We show that the satisfiability problem of boolean formulas in the 3-conjunctive normal form, 3SAT, is polynomially reduciable to the BDBCSTP, i.e., 3SAT $\leq_p$ BDBCSTP. Since 3SAT is NP-complete [Garey and Johnson 1979], the BDBCSTP in general is NP-hard. More specifically, we have the following theorem.

THEOREM 1. *BDBCSTP is NP-complete.*

The BDMCSTP is NP-hard. Since the general BDMCSTP is NP-hard, whether there exists a polynomial-time algorithm for the problem is an open question. Hence, we are interested in finding an approximation algorithm with a guaranteed performance bound. However, we can show that if there exists a polynomial-time algorithm $\mathcal{A}$ for the BDMCSTP without triangle inequality, then $\mathcal{A}$ can be used to solve the 3SAT problem, which is a contradiction if $P \neq NP$. Specifically, the following theorem holds.

THEOREM 2. *If $P \neq NP$ and $\rho \geq 1$, there is no polynomial-time approximation algorithm with performance bound ρ for the general BDMCSTP.*

The triangle inequality for the routing cost usually holds in practical applications. Given a graph $G = (V, E)$, let the routing costs of its minimum spanning tree measured on r, MST_r, and an optimal tree, T_{BDMCST}, to the BDMCSTP on G be Φ_{MST_r} and Φ_{BDMCST}, respectively. We have the following theorem.

THEOREM 3. $\Phi_{MST_r} \leq \Phi_{BDMCST} \leq (|V| - 1)\Phi_{MST_r} \leq (|V| - 1)\Phi_{BDMCST}$, *if the routing cost satisfies the triangle inequality.*

To save space, see Chang [1996] for the proofs of Theorems 1–3.

3.2.2 *The Timing-Driven Routing-Tree Heuristic*. Since the general BDMCSTP is NP-hard, we resort to a heuristic to obtain efficient solutions. For timing-driven routing, we use the following cost functions. The routing cost of an edge $e = (u, v)$ is defined as the Manhattan distance

$$r(e) = |x_u - x_v| + |y_u - y_v|, \tag{1}$$

where (x_u, y_u) and (x_v, y_v) are the coordinates of the pins u and v, respectively. The delay measurement (number of switches used) is given by

$$d(e) = \max\{d_{min}, \alpha r(e)\}, \tag{2}$$

where d_{min} is the minimum number of switches required to connect u and v (we discuss d_{min} in detail later), and $0 < \alpha \leq 1$ is a constant that satisfies the following inequality:

$$D_{G,d}(s, t_i) \leq B(t_i), \ \ \forall t_i \in V, \tag{3}$$

and at the same time maximizes the left-hand side of the above inequality; here $D_{G,d}(s, t_i)$ is the delay cost of the shortest path from s to t_i on G. Recall that the minimum length of routing segment is the span of one logic module, thus the routing cost function $r(e)$ in the delay function $d(e)$ in Eq. (2) represents the maximum possible delay between two pins u and v. The constant α is used to capture the usage of longer routing segments for satisfying stringent delay bounds. Smaller α gives smaller delay cost $d(e)$, implying that the routing between u and v needs to use longer segments. The value of α is computed by Inequality (3) on all source-sink pairs of the net. Thus, for the source-sink pairs for which Inequality (3) is strictly less than the corresponding delay bound $B(t_i)$, there will be more alternatives for the routing to meet the delay bounds.

To compute d_{min}, we construct a graph H as follows. Represent each pin and routing segment by a vertex. Connect two vertices by an edge with unit weight if there is a switch between the two vertices. That is, the switch

```
Algorithm: $PrimBDRT(G, B, s)$
Input: $G = (V, E)$; $B(t_i), \forall t_i \in V$; $s$.
Output: $T = (V_T, E_T)$.
1  $V_T \leftarrow \{s\}$; $E_T \leftarrow \emptyset$;
2  while $(|V_T| < |V|)$ do
3      Among all edges $e_i = (u, v) \in E$, $u \in V_T$, and $v \in V \backslash V_T$,
       satisfying $D_{T,d}(s, u) + d(e_i) \leq B(v)$, pick $e$ such that
       $r(e) = \min_i r(e_i)$;
4      if such an edge exists
5          $V_T \leftarrow V_T \cup \{v\}$;
6          $E_T \leftarrow E_T \cup \{(u, v)\}$;
7      else
8          Pick $e = (u, v)$ such that $r(e) = \min_i r(e_i)$;
9          $V_T \leftarrow V_T \cup \{v|$ vertex $v$ in $STP_{G,d}(s, v)\}$;
10         $E_T \leftarrow E_T \cup \{e|$ edge $e$ in $STP_{G,d}(s, v)\}$;
11 $T \leftarrow$ shortest-paths tree of $T$ measured on delay;
12 Output $T = (V_T, E_T)$.
```

Fig. 5. Algorithm for constructing bounded-delay routing trees.

either connects a pin with a routing segment or connects a routing segment to another segment through a switch module. Such a graph H is referred to as a *connectivity graph*. The delay bound d_{min} is the length of the shortest path between u and v on H, and can be computed by a shortest-path algorithm [Papadimitriou and Steiglitz 1982]. The connectivity graph is used again in defining the cost function for linear assignment in Section 3.4.

With cost functions (1) and (2), we use the following heuristic to construct bounded-delay routing trees. The heuristic, $PrimBDRT$, follows the approach of Prim's minimum spanning-tree construction [Prim 1957], and is a variant of the tree construction used in Awerbuch et al. [1990]; Cong et al. [1992]; Ho et al. [1991]; and Khuller et al. [1993], which measure both routing and delay costs based on geometric distance. We grow a tree $T = (V_T, E_T)$ incrementally, starting from the source s. At each step, we choose an edge $e = (u, v)$, where $u \in V_T$ and $v \in V \backslash V_T$ such that the routing cost of the edge is minimum and the delay constraint from s to v is satisfied. If no such edge exists, we add the shortest path from s to v on G measured on delay cost d, $STP_{G,d}(s, v)$ to the routing tree. Since adding the path could violate the tree property, we compute the shortest-path tree of T measured on delay. Figure 5 summarizes the algorithm. The time complexity of $PrimBDRT$ is $O(|V|^3)$. In particular, the heuristic enjoys the performance bound listed in Theorem 3, since the triangle inequality holds for the routing cost.

3.3 Delay-Bound Distribution

In this section we discuss the distribution of delay bounds on source-sink pairs to the edges of a routing tree. We first introduce a few terms. In a routing tree T, there is only one edge between any sink and its parent. Let

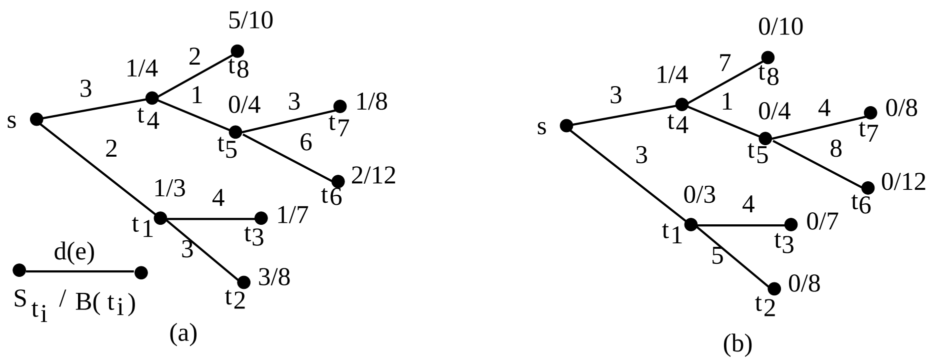

Fig. 6. (a) Slack computation; (b) delay-bound distribution.

the edge between the sink t_i and its parent be e_{t_i}. For any sink t_i, there is a unique path from the source s to t_i in T. Let $b(e_{t_i})$ be the delay bound allocated to the edge e_{t_i}. The *slack* of a sink t_i, denoted by S_{t_i}, is defined as the difference between the delay bound $B(t_i)$ and the sum of delay bounds of all edges along the path from s to t_i, i.e., $S_{t_i} = B(t_i) - \Sigma_{e \in p(s,\, t_i)} b(e)$, where $p(s, t_i)$ is the path from s to t_i in T. The slack for the source s is defined as zero. Figure 6(a) shows a routing tree with delay bounds on the sinks, a distribution of delay bounds on the edges and the corresponding slacks.

The delay-bound distribution is to allocate the delay bounds on the edges of the routing tree T such that for every sink t_i in T, the slack $S_{t_i} \geq 0$. The algorithm for delay-bound distribution is listed in Figure 7. First, the algorithm allocates to every edge e_{t_i} in T an initial delay bound $b(e_{t_i}) = d(e_{t_i})$, i.e., the associated edge delay cost (line 1). Note that with such initial delay bound allocation, the slacks of all sinks in the routing tree T constructed by the Prim BDRT algorithm in Section 3.2 are guaranteed to be non-negative. After the initial allocation, it is possible that the delay bounds on the edges can be relaxed further under certain conditions. Let T_{t_i} be the subtree of T rooted at sink t_i. A sink t_i is a *relaxable sink* if the slack of every sink in T_{t_i} is greater than zero. The following theorem gives the necessary and sufficient condition for further relaxing edge delay bounds.

THEOREM 4. *The delay bound on an edge e_{t_i} can be relaxed further if and only if the sink t_i is a relaxable sink.*

PROOF. *(If)*. Suppose that the sink t_i is relaxable. By definition, all sinks in the subtree T_{t_i} must also be relaxable. Let t'_i be the sink in T_{t_i} with the minimum slack among all the sinks in T_{t_i}. The delay bound of edge e_{t_i} can be increased by $S_{t'_i}$ and no sink in T will have negative slacks. Thus the delay bound of edge e_{t_i} can be relaxed. Notice that the existence of sinks with positive slacks is not sufficient to relax the delay bounds on edges. For

```
Algorithm: Delay_Bound_Distribution($T, B, d$)
Input: $T = (V, E)$; $B(t_i), \forall t_i \in V$; $d(e), e \in E$
Output: $T$ with delay-bound distribution.
1 $b(e_{t_i}) \leftarrow d(e_{t_i}), \forall t_i \in V$;
2 while (there exist relaxable sinks) do
3     Compute slacks $S_{t_i}, \forall t_i \in V$;
4     Label every sink $t_i \in V$ as either relaxable or unrelaxable;
5     Pick a relaxable sink $t_i$ with minimum $S_{t_i}$, let $b(e_{t_i}) \leftarrow b(e_{t_i}) + S_{t_i}$.
```

Fig. 7. Algorithm for delay bound distribution.

example, the sink t_4 in Figure 6(a) has slack 1, but t_4 is not a relaxable sink. The reason is that one of t_4's children, sink t_5, has zero slack.

(Only If). Suppose a sink t_i is not relaxable. By definition, either $S_{t_i} = 0$ or there is at least one sink in the subtree T_{t_i} which has zero slack. If the delay bound for e_{t_i} is increased, there must be at least one sink in T_{t_i} which will have negative slack. Thus the delay bound for e_{t_i} cannot be relaxed. □

All the relaxable sinks in T can be easily identified in $O(|V|)$ time. After allocating the initial delay bounds on edges, the algorithm picks a relaxable sink t_i, if any, with the minimum slack, and relaxes the the delay bound on e_{t_i} by the amount of S_{t_i} (lines 3-5). The process repeats until there is no relaxable sink in T. Figure 6(b) shows the routing tree after applying the algorithm on the tree shown in Figure 6(a). Notice that the slack of t_4 does not change, since t_4 is not a relaxable sink.

Each of steps 3 and 4 can be done in one traversal of the routing tree T. In each iteration, the number of relaxable sinks is reduced at least by one. Thus there are at most $|V|-1$ iterations in the while loop, and the running time of the algorithm is $O(|V|^2)$.

3.4 Cost Function for Linear Assignment

A cut line at a routing hierarchical level cuts through a set of subchannels. A routing section on the cut line is a subchannel. Let C_{ij} be the cost of routing connection i through subchannel j. The cost C_{ij} consists of three terms:

$$C_{ij} = C_{ij}^{(1)} + C_{ij}^{(2)} + C_{ij}^{(3)}. \tag{4}$$

The first term $C_{ij}^{(1)}$ depends on whether the connection i can reach subchannel j:

$$C_{ij}^{(1)} = \begin{cases} 0 & \text{if connection } i \text{ can reach subchannel } j, \\ \infty & \text{otherwise.} \end{cases}$$

Reachability can be determined by a breadth-first search on the connectivity graph as defined in Section 3.2.

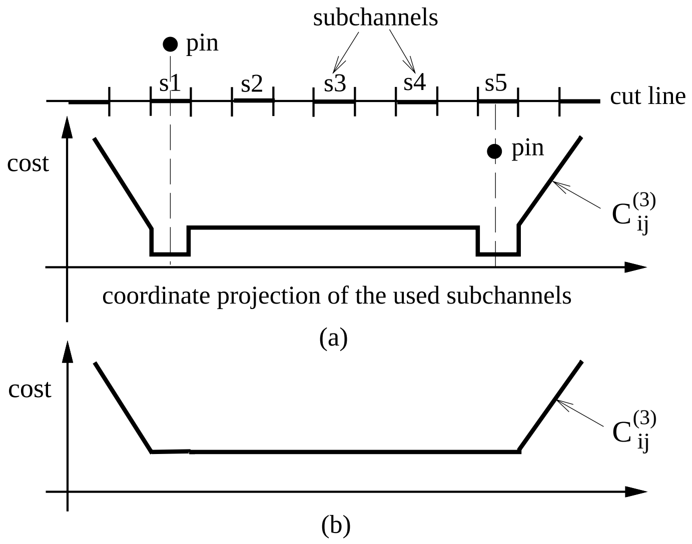

Fig. 8. The third term $C_{ij}^{(3)}$ of the cost function for linear assignment. (a) Case for our timing-driven routing; (b) case for nontiming-driven routing.

The second term intends to utilize the routing segments evenly according to the connection length and its delay bound:

$$C_{ij}^{(2)} = a\left|\frac{l_i}{U_i} - L_j\right|,$$

where l_i is the Manhattan distance of the connection i; U_i is the delay bound of connection i; L_j is the length of routing segments in the subchannel j; and $a > 0$ is a constant. Note that l_i/U_i is the average Manhattan distance in routing the connection using at most one switch. Thus the closer l_i/U_i is to the length of segments in a subchannel, the lower the cost.

A bend in routing a net requires at least one switch. So it is preferable for routing a net with fewer bends to meet delay bounds. The third term $C_{ij}^{(3)}$ of the cost function is defined on the basis of this consideration and is illustrated in Figure 8(a). There are two subchannels ($s1$ and $s5$ in Figure 8) on the cut line through which the connection will have the minimum possible number of bends. Thus C_{ij} has the minimum value at these two subchannels. The cost of routing through the subchannels within the bounding box of the connection is a constant greater than the minimum cost. The costs for routing through subchannels outside the bounding box of the connection are proportional to the Manhattan distances between the subchannels and the bounding box. Note that the cost function illustrated in Figure 8(b) corresponds to those used in Lauther [1987]; Marek-Sadowska [1993]; and Suaris and Kedem [1989].

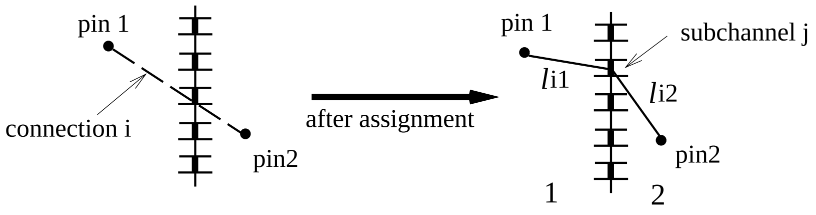

Fig. 9. Delay bound redistribution: simple case.

3.5 Delay-Bound Redistribution

After assigning connections to the subchannels on the cut line, the delay upper bounds for the connections need to be redistributed among the new connections created by the cut line.

We first consider a simple case where the connection crossing the cut line does not share pins with any other connection crossing the cut line (see Figure 9). Assume the connection i is assigned to the subchannel j. Let $i1$ and $i2$ be the new connections created on the two sides of the cut line. Let l_{i1} (l_{i2}) be the Manhattan distance between pin 1 (pin 2) and the subchannel j. The delay bound U_i of the connection i is redistributed among the new connections $i1$ and $i2$ proportional to their distances to the assigned subchannel, i.e.,

$$U_{i1} = \frac{l_{i1}}{l_{i1} + l_{i2}} U_i,$$

and

$$U_{i2} = \frac{l_{i2}}{l_{i1} + l_{i2}} U_i,$$

where U_{i1} and U_{i2} are the delay bounds for the new connections $i1$ and $i2$, respectively.

In general, connections of the same net crossing the cut line may share pins. If connections that share pins are not assigned to the same subchannel, delay bounds are redistributed for every connection independently, as discussed in the above simple case. If several connections that share pins are assigned to the same subchannel, the delay bound redistribution will be performed as illustrated in Figure 10. In Figure 10, three connections i, j, and k belong to the same net. Connections i and j share a pin on side 1, and connections i and k share another pin on side 2. All three connections are assigned to the same subchannel. Thus the new connections $i1$ and $j1$ ($i2$ and $k2$) are the same. The delay bound for a new connection created on the side where the connections share a pin is chosen to be the minimum of the delay bounds. In Figure 10, the delay bounds for new connections $i1$ ($j1$) and $i2$ ($j2$) are

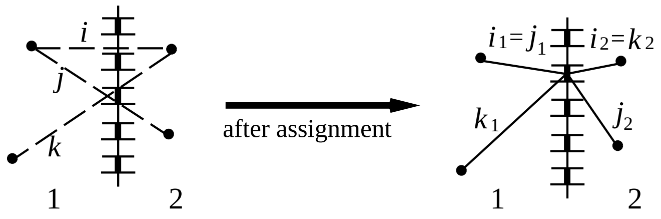

Fig. 10. Delay bound redistribution: general case.

$$U_{i1} = U_{j1} = min\left\{\frac{l_{i1}}{l_{i1} + l_{i2}}U_i, \frac{l_{j1}}{l_{j1} + l_{j2}}U_j\right\}$$

and

$$U_{i2} = U_{k2} = min\left\{\frac{l_{i2}}{l_{i1} + l_{i2}}U_i, \frac{l_{k2}}{l_{k1} + l_{k2}}U_k\right\},$$

where U_{i1}, U_{i2}, U_{j1}, and U_{k2} are the delay bounds for the new connections $i1, i2, j1$, and $k2$, respectively. For a new connection on the side that does not share pins with other connections, the delay bound is the difference between the delay bound for the original connection and that allocated to the new connection on the other side of the cut line. In Figure 10, the delay bounds for the new connections $j2$ and $k1$ are $U_{j2} = U_j - U_{j1}$, and $U_{k1} = U_k - U_{k2}$, respectively.

3.6 Detailed Routing

Detailed routing can be easily incorporated into global routing at every hierarchical level. After global routing, the connections assigned to every subchannel are known. For each subchannel, construct a bipartite graph $G = (V_1 \cup V_2, E)$. A connection or a set of connections belonging to the same net and assigned to the subchannel is represented by a node $u \in V_1$. A routing segment in the subchannel is represented by a node $v \in V_2$. There is an edge between a node $u \in V_1$ and a node $v \in V_2$ if and only if all the pins of the connection(s) represented by u can reach the routing segment represented by v. Detailed routing is then solved optimally in polynomial time using a bipartite matching algorithm [Papadimitriou and Steiglitz 1982]. If detailed routing cannot route all the connections assigned to a subchannel, the capacity of the subchannel is reduced appropriately and the global routing at this hierarchical level is routed again. The iterative procedure continues until either all connections are routed or a predefined routing completion rate is achieved.

One of the advantages of performing detailed routing and global routing simultaneously is that any overestimate in global routing can be corrected immediately and accordingly.

4. EXPERIMENTAL RESULTS

The proposed timing-driven routing algorithm was implemented in C on a SUN SPARC Ultra I workstation. Table I gives the names of the circuits, the numbers of connections (the numbers of source-sink pairs) in each circuit, and the sizes of the circuits in terms of the number of logic modules in the FPGA. Note that this set of circuits is used in Fallah and Rose [1992].

There are 20 tracks per channel. Each horizontal and vertical channel consists of three subchannels that contain 8, 4, and 4 tracks of single lines, medium lines with a length of 4 units, and long lines, respectively. These parameters of routing segmentation are close to the LUT-based architecture used in the Lucent Technologies FPGAs [Lucent Technologies 1996]. All three types of subchannels can be connected with each other via switch modules.

We randomly picked a source from the pins of a net and made the other sinks. Let $l(s, t_i)$ be the Manhattan distance between the source s and a sink t_i. The delay bound $B(t_i)$ was chosen randomly from the three intervals $[0.3l(s, t_i), 0.8l(s, t_i)]$, $[0.4l(s, t_i), 0.9l(s, t_i)]$, and $[0.5l(s, t_i), 1.0l(s, t_i)]$, representing the, respectively, tight, medium, and loose cases for timing constraints. If the delay bound $B(t_i)$ is less than d_{min},[1] then set $B(t_i) = d_{min}$. Each circuit is routed by the algorithm and the percentage of source-sink pairs violating the delay bounds is computed. The results are shown in the column "Timing-driven" in Table II. For comparison, we also routed the circuits by the same routing algorithm, with the cost function $C_{ij}^{(3)}$ in Eq. (4) being set according to the cost function illustrated in Figure 8(b); this leads to a nontiming-driven routing approach [Lauther 1987; Marek-Sadowska 1993; Suaris and Kedem 1989]. The results are given in the column "Nontiming-driven" of Table II. For all the circuits, the timing-driven routing algorithm substantially reduces the percentage of connections violating the delay bounds. The percentage of reduction in timing violations is listed in the column "Improv." in Table II. The results show that averages of 70%, 80%, and 86% reductions were obtained for the tight, medium, and loose cases, respectively. The results show that the percentage of timing constraints that are satisfied depends on the given delay bounds. The tighter the delay bounds, the harder it is for both routers to satisfy specified timing constraints. Our router is quite efficient; runtime ranged from seconds for the smallest circuit (BUSC) to about 10 minutes for the largest circuit (K2).

Note that we do not compare our results with the work of Alexander and Robins [1995]; Brown et al. [1992]; Chen et al. [1995]; Lee and Wu [1995]; and Lemieux et al. [1997] because it considers only one type of wire segment while ours is based on multilength segments (which is the case in most commercial architectures).

[1] Recall that, in Section 3.2, d_{min} is the minimum number of switches required to connect two pins.

Table I. Benchmark Circuits

Circuit	#Connections	Circuit Sizes
BUSC	392	12 x 11
X1	436	13 x 12
TooLarge	519	14 x 13
VDA	722	16 x 15
DMA	771	17 x 15
ALU4	851	18 x 16
K2	1256	21 x 19

Table II. Percentage of Connections Violating Timing Constraints after Detailed Routing Completion

	$B(t_i) \in [0.3l(s,t_i),\ 0.8l(s,t_i)]$			$B(t_i) \in [0.4l(s,t_i),\ 0.9l(s,t_i)]$			$B(t_i) \in [0.5l(s,t_i),\ 1.0l(s,t_i)]$		
Circuit	Timing-driven (%)	Non-timing driven (%)	Improv. (%)	Timing-driven (%)	Non-timing driven (%)	Improv. (%)	Timing-driven (%)	Non-timing driven (%)	Improv. (%)
BUSC	4.80	22.88	79.02	0.56	11.80	95.25	0.00	6.46	100.00
X1	5.27	25.33	79.19	1.06	23.22	95.43	0.00	12.37	100.00
TooLarge	6.84	25.17	72.82	3.08	18.02	82.91	1.76	11.21	84.23
VDA	9.40	31.64	70.29	5.22	16.72	68.78	2.84	11.49	75.28
DMA	8.03	25.57	68.60	2.41	16.33	85.24	1.20	8.43	85.77
ALU4	12.58	33.37	62.30	5.43	25.12	78.38	2.29	16.91	86.46
K2	14.58	35.85	59.33	12.88	29.79	56.76	5.83	21.01	72.25
Average	-	-	70.22	-	-	80.39	-	-	86.28

5. CONCLUSIONS

We have presented a timing-driven router for FPGAs with segments of various lengths. The router is based on a hierarchical strategy and is suited for the special properties of FPGA routing architectures. Experimental results show that our router is very effective in reducing the number of connections violating timing constraints.

ACKNOWLEDGMENTS

The authors thank Steve Brown and Baharam Fallah of the University of Toronto for providing us with the benchmark circuits, Nick Haruyama for helpful discussions, and Cherng-Shiuan Wang for implementing the detailed router.

REFERENCES

ALEXANDER, M. J., COHOON, J. P., GANLEY, J. L., AND ROBINS, G. 1995. Performance-oriented placement and routing for field-programmable gate arrays. In *Proceedings of the European Conference EURO-DAC '95 with EURO-VHDL '95 on Design Automation* (Brighton, UK, Sept. 18–22), G. Musgrave, Ed. IEEE Computer Society Press, Los Alamitos, CA, 80–85.

ALEXANDER, M. J. AND ROBINS, G. 1995. New performance-driven FPGA routing algorithms. In *Proceedings of the 32nd ACM/IEEE Conference on Design Automation* (DAC

'95, San Francisco, CA, June 12–16, 1995), B. T. Preas, Ed. ACM Press, New York, NY, 562–567.

AWERBUCH, B., BARATZ, A., AND PELEG, D. 1990. Cost-sensitive analysis of communication protocols. In *Proceedings of the Ninth Annual ACM Symposium on Principles of Distributed Computing* (PODC '90, Quebec City, Canada, Aug. 22–24), C. Dwork, Ed. ACM Press, New York, NY, 177–187.

BREUER, M. A. 1988. A class of min-cut placement algorithms. In *A Compendium of Papers from the Design Automation Conference on Twenty-Five Years of Electronic Design Automation* (DAC '88), A. R. Newton, Ed. ACM Press, New York, NY, 142–148.

BROWN, S. D., ROSE, J., AND VRANESIC, Z. 1992. A detailed router for field-programmable gate arrays. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 11*, 5, 620–627.

CHANG, Y.-W. 1996. Routing architectures and algorithms for field-programmable gate arrays. Tech Rep.. University of Texas at Austin, Austin, TX.

CHANG, Y.-W., THAKUR, S., ZHU, K., AND WONG, D. F. 1994. A new global routing algorithm for FPGAs. In *Proceedings of the 1994 IEEE/ACM International Conference on Computer-Aided Design* (ICCAD '94, San Jose, CA, Nov. 6–10, 1994), J. A. G. Jess and R. Rudell, Eds. IEEE Computer Society Press, Los Alamitos, CA, 356–361.

CHEN, C.-D., LEE, Y.-S., WU, C.-H., AND LIN, Y.-L. 1995. TRACER-fpga: A router for ram-based FPGA's. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 14*, 3 (Mar.), 371–374.

CONG, J., KAHNG, A., ROBINS, G., SARRAFZADEH, M., AND WONG, C. K. 1992. Provably good performance-driven global routing. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 11*, 6 (June), 739–752.

ELMORE, W. C. 1948. The transient response of damped linear networks with particular regard to wide band amplifiers. *J. Appl. Phys. 19*, 55–63.

FALLAH, B. AND ROSE, J. 1992. Timing-driven routing segment assignment in FPGAs. In *Proceedings of the Canadian Conference on VLSI* (Halifax, Nova Scotia, Oct. 18-20, 1992), 124–130.

FRANKLE, J. 1992. Iterative and adaptive slack allocation for performance-driven layout and FPGA routing. In *Proceedings of the 29th ACM/IEEE Conference on Design Automation* (DAC '92, Anaheim, CA, June 8–12), D. G. Schweikert, Ed. IEEE Computer Society Press, Los Alamitos, CA, 536–542.

GAREY, M. AND JOHNSON, D. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., New York, NY.

HAUGE, P. S., NAIR, R., AND YOFFA, E. J. 1987. Circuit placement for predictable performance. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design* (ICCAD, San Jose, Calif., Nov. 9 - 12), IEEE Computer Society Press, Los Alamitos, CA, 88–91.

HO, J.-M., LEE, D. T., CHANG, C.-H., AND WONG, C. K. 1991. Minimum diameter spanning trees and related problems. *SIAM J. Comput. 20*, 5 (Oct. 1991), 987–997.

KHELLAH, M., BROWN, S., AND VRANESIC, Z. 1993. Modelling routing delays in SRAM-based FPGAs. In *Proceedings of the Canadian Conference on VLSI* (Banff, Alberta, Canada, Nov. 14–16, 1993), 14.

KHULLER, S., RAGHAVACHARI, B., AND YOUNG, N. 1993. Balancing minimum spanning and shortest path trees. In *Proceedings of the 4th Annual ACM/SIAM Symposium on Discrete Algorithms* (Austin, Tex., Jan. 25 - 27), ACM Press, New York, NY, 243–250.

LAUTHER, U. 1987. Top-down hierarchical global routing for channelless gate arrays based on linear assignment. In *Proceedings of the IFIP Conference on VLSI* (Aug. 10-12, 1987), North-Holland Publishing Co., Amsterdam, The Netherlands, 109–120.

LEE, Y.-S. AND WU, C.-H. 1995. A performance and routability driven router for FPGAs considering path delay. In *Proceedings of the 32nd ACM/IEEE Conference on Design Automation* (San Francisco, CA, June 12–16, 1995), ACM Press, New York, NY, 557–561.

LEMIEUX, G. AND BROWN, S. D. 1993. A detailed router for allocating wire segments in FPGAs. In *Proceedings of the 4th Workshop on ACM/SIGDA Physical Design* (April 1993, Lake Arrow, CA), ACM Press, New York, NY.

LEMIEUX, G. G. F., BROWN, S. D., AND VRANESIC, D. 1997. On two-step routing for FPGAS. In *Proceedings of the 1997 International Symposium on Physical Design* (ISPD '97, Napa Valley, CA, Apr. 14–16, 1997), A. B. Kahng and M. Sarrafzadeh, Eds. ACM Press, New York, NY, 60–66.

LUCENT TECHNOLOGIES INC. 1996. ORCA OR2CxxA (5.0 V) and OR2TxxA (3.3 V) series field-programmable gate arrays. Lucent Technologies, Inc., Allentown, PA.

MAREK-SADOWSKA, M. 1986. Route planner for custom chip design. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design* (San Jose, CA, Nov. 10-13, 1986), ACM Press, New York, NY, 246–249.

MAREK-SADOWSKA, M. 1993. Issues in timing driven layout. In *Algorithmic Aspects of VLSI Layout*, D. T. Lee and M. Sarrafzadeh, Eds. World Scientific Publishing Co., Inc., River Edge, NJ.

MCMURCHIE, L. AND EBELING, C. 1995. PathFinder: A negotiation-based performance-driven router for FPGAs. In *Proceedings of the Third International ACM Symposium on Field-Programmable Gate Arrays* (FPGA '95, Monterey, CA, Feb. 12–14), P. K. Chan and J. Rose, Eds. ACM Press, New York, NY, 111–117.

PALCZEWSKI, M. 1992. Plane parallel: A maze router and its application to FPGAs. In *Proceedings of the 29th ACM/IEEE Conference on Design Automation* (DAC '92, Anaheim, CA, June 8–12), D. G. Schweikert, Ed. IEEE Computer Society Press, Los Alamitos, CA, 691–697.

PAPADIMITRIOU, C. H. AND STEIGLITZ, K. 1982. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ.

PRIM, R. C. 1957. Shortest connecting networks and some generalizations. *Bell Syst. Tech. J. 36*, 1389–1401.

SUARIS, P. AND KEDEM, G. 1989. A quadrisection-based combined place and route scheme for standard cells. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 8*, 3 (Mar.), 234–244.

XILINX. 1994. *The Programmable Logic Data Book*. Xilinx, San Jose, CA.

ZHU, K. AND WONG, D. F. 1998. Switch bound allocation for maximizing routability in timing-driven routing of FPGAs. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 17*, 4, 316–323.

Received: October 1997; revised: March 1999; accepted: March 1999