Published in Proceedings of IEEE 1994 Custom Integrated Circuits Conference, San Diego, California, May 1994.

Minimizing Interconnection Delays in Array-based FPGAs

Muhammad Khellah, Stephen Brown, and Zvonko Vranesic

Department of Electrical and Computer Engineering
University of Toronto
E-mail: khellah | brown | zvonko@eecg.toronto.edu

Abstract

This paper provides research results that suggest ways in which the speed-performance of array-based FPGAs, like those from Xilinx, can be improved through enhancing their interconnect. Using an experimental approach, we study this issue from both the perspective of improving the routing architectures of the chips, as well as the CAD tools used to route circuits. The basic conclusions reached are: the lengths of wire segments in the interconnect dramatically affects speed-performance, it is crucial to limit the number of programmable switches that signals pass through in series, the impact of decisions made by the CAD routing tools is very significant, and the CAD tools should consider *both* speed-performance and area utilization, not just focus on one goal.

1 Introduction

Over the last decade, Field-Programmable Gate Arrays (FPGAs) have emerged as a key technology for implementing circuits in VLSI. A number of different types of FPGAs are currently available from various manufacturers. One of the key features distinguishing different classes of FPGAs is the technology used to implement the user-programmable switches, examples of which are SRAM, EPROM, EEPROM, and antifuse. Because of its widespread use, being offered in FPGAs manufactured by Xilinx, Altera, AT&T, Atmel and Algotronix, this paper assumes that programmable switches are based on SRAM technology.

Since they are user-programmable, FPGAs considerably reduce manufacturing and prototyping time, as well as design costs, and are thus more attractive than alternative technologies like Mask Programmable Gate Arrays (MPGAs). On the other hand, the inherent drawbacks of user-programmability are reduced logic capacity and speed-performance, and this engenders limitations on the range of applications for which FPGAs are suitable. In this paper, we focus on improving the speed-performance of FPGAs.

Speed-performance is limited by two main factors: combinational delays in logic blocks, and propagation delays through the interconnect's programmable switches. Unlike mask-programmed technologies where combinational delay dominates, in FPGAs interconnect accounts for a very significant 40-60 percent of total delay [1]. Two key factors affect delays due to interconnect in an FPGA: 1) the *routing architecture*, which comprises the wires and switches used to interconnect the logic blocks, and 2) the CAD tools used to implement circuits.

In previous research [2] [3], the speed-performance of FPGA routing architectures has been studied, but only for row-based devices such as those from Actel. In this paper, we investigate ways of decreasing interconnect delays for FPGA architectures that are array-based, like those from Xilinx. One motivation for this research is the results of recent benchmarks [4] that have shown that array-based FPGAs currently offer lesser speed-performance than row-based FPGAs or Complex PLDs (CPLDs). As one of the key differences between these devices, the interconnect structure is a prime candidate for enhancing speed-performance.

To experiment with a range of routing architectures, a general model for FPGAs is needed. Fig. 1 shows the model used for this study. It consists of a two-dimensional array of logic blocks, vertical and horizontal routing channels, and I/O cells around the chip periphery. The logic blocks contain combinational and sequential circuit elements, while the routing channels comprise the wire segments and switches used to interconnect the logic blocks. Wire segments exist in both vertical and horizontal *tracks*. For the small example in Fig. 1, there are four tracks per channel and each logic block has two *pins* that appear on each of its sides. Although not shown in the figure, the C blocks contain switches for connecting the pins of the logic blocks to the wire segments, and the S blocks house switches that join one wire segment to another. The routing switches are pass-transistors controlled by SRAM cells. Wire segments can be of any *length*, where the length of a wire segment is defined as the number of logic blocks it spans. As an illustration, Fig. 2 shows a section of a horizontal channel with wire segments of length 1, 2 and 3. Our FPGA model allows virtually any channel segmentation scheme.

In early FPGAs, tracks consisted mostly of short wire segments of length one, and longer segments could be formed by joining together two or more of these short segments via routing switches. While this provides for good utilization of the wires in the sense that there are no long segments that might be wasted on short connections, requiring that long connections pass through several switches in series severely impairs speed-performance. This follows because a pass-transistor

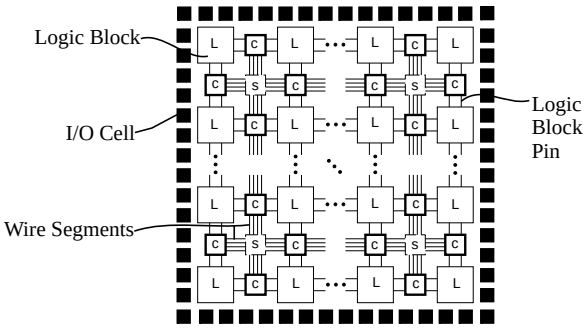

Figure 1 - *General Model of an Array-based FPGA.*

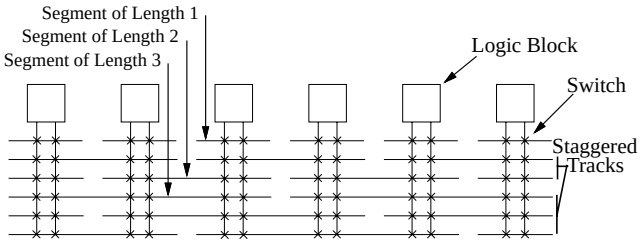

Figure 2 - *A Channel with Wire Segment Lengths of 1, 2, and 3.*

switch has significant series resistance and parasitic capacitance. To address these issues, recent SRAM-based FPGAs include wire segments of various lengths, but no previous study has investigated the trade-offs involved in selecting a particular segmentation scheme. Thus, in studying the performance of routing architectures, the key aspect considered here is the effects of wire segment lengths; we provide a comprehensive investigation of this issue, considering not only speed-performance, but also area utilization.

The following section briefly describes the CAD routing tools used in this study, and Section 3 presents the experimental results and conclusions. Section 4 contains final remarks.

2 CAD Routing Tools

Typically, several CAD tools are employed when implementing a circuit in an FPGA. Following initial entry of the design (via schematic capture, HDL, etc.), typical steps in a CAD system include [5]: logic optimization, technology mapping, placement, and routing. This paper focuses only on the routing CAD tools. We adopt the traditional CAD approach for routing a device with channels by dividing the problem into two steps: global routing and detailed routing.

2.1 The Global Router

A global router chooses channels for each net and leaves the task of allocating specific wire segments and switches to detailed routing. The global router used in this study is called PgaRoute [6]. Since it is based on point-to-point connections, as its first step PgaRoute divides any multipoint nets in the circuit into a set of two-point connections. When routing, the main objective of PgaRoute is to select channels for each connection in such a way that the number of connections assigned per channel is balanced over the whole FPGA. Intuitively, this is a reasonable goal for routing an FPGA, since the number of tracks per channel is strictly limited.

In addition to balancing channel usage, PgaRoute can also (optionally) minimize the number of bends that each of the two point connections incurs [7]. A *bend* occurs at an S block if a connection has to turn in order to reach its destination. Minimizing bends is important with respect to speed-performance, as will be discussed in Section 3. While routing, PgaRoute uses a cost function that keeps count of the number of tracks per channel and through an iterative process of routing, removing and re-routing connections achieves a balanced usage of the channels. If minimizing bends is also desired, the router has additional parameters [7], that exploit such things as functional and electrical equivalence of logic block pins, to reduce the number of bends incurred for each routed connection.

2.2 The Detailed Router

The detailed router used in this study is called SEGA [8], for SEGment Allocator, and was developed specifically for array-based FPGAs. SEGA can be used over a wide range of routing architectures, and so is suitable as a research tool for evaluation of different segmentation schemes. The input to SEGA is a netlist of two-point connections, which is the output of the global router. To route the connections, SEGA allocates wire segments according to a cost function, basing its decisions on either of two goals: 1) optimize for area or 2) optimize for speed. For 1), only the *routability* of the circuit is considered, which means the cost function focuses only on the task of successfully routing 100% of the connections in a circuit. In mode 2), SEGA selects the routes that have the best speed performance. As a research vehicle, SEGA can use one of several cost functions to assess the speed-performance of a detailed route; Table 1 summarizes the cost functions [10]. To illustrate the possible impact of detailed routing on speed-performance, Section 3 compares results produced by each cost function in the table.

Cost Function	Description
Area	optimize for routability only
Seg_Len	minimize lengths of wire segments used
Num_Seg	minimize the number of wire segments used
Seg_Len + Num_Seg	combination of the above cost functions
Analytic_Model	use an analytic model [9] to find delays
Net_Routing	use analytic model, but also focus on re-use of wire segments for connections on same net

Table 1 - *Detailed Router Cost Functions.*

3 Experimental Results

This section describes experimental results that show the effects of both CAD routing tools and channel segmentation on speed-performance. The results are presented for a set of 13 industrial benchmark circuits (from MCNC), ranging in size from 202 to 2135 two-point connections.

3.1 Architectural Assumptions

The following assumptions, most of which are based on previous studies on FPGAs [11], are used:

i. All routing channels have an equal number of tracks, *W*.
ii. Wherever a logic block pin can connect to a channel, it can be connected to all *W* tracks.
iii. Each wire segment that enters an S block can pass straight through, or turn right or left, and only wire segments of equal length can be joined together.
iv. All wire segments in a given track have the same length.
v. Tracks are *staggered*, meaning that wire segments of equal length in adjacent tracks start at different positions. For an illustration, see Fig. 2.

3.2 Effect of CAD Routing Tools on Routing Delays

This section illustrates the effects of both global routing and detailed routing on speed-performance. The experimental approach is as follows: for the global routing experiment, each of the benchmark circuits was routed twice: once with PgaRoute's bend reduction feature turned off, and then with bend reduction turned on. Each solution was detailed routed

using SEGA, with the cost functions shown in Table 1. The FPGA was assumed to have 30 tracks per channel, with tracks having segments of length one, two and three[1]. After routing each circuit, an analytic delay model [9] was employed to measure the average delay for each net. Table 2 gives a summary of the results of these experiments. Each number in the table represents the average net delay (in ns.) for the 13 benchmarks.

Referring to Table 2, enabling the bend reduction feature clearly produces better speed-performance results. The reason is that connections with longer straight sections are produced, which allows the detailed router to make use of longer wire segments. Connections routed with a small number of long wires need to pass through fewer switches than if they were routed with a larger number of short wire segments.

Table 2 also shows that the detailed router has a significant effect on speed-performance. The various cost functions in SEGA yield different average routing delays. The results for the Area cost function show that focusing only on routability gives less than optimal routing delays, as would be expected. The Seg_Len row indicates that very poor speed-performance results if the router considers only the lengths of wire segments. The intent of this function is to prevent the assignment of long wires to short connections to minimize capacitive loading, but comparison to the Num_Seg row shows this to be a poor strategy. Minimizing the number of segments that connections pass through yields among the lowest delays and, since combining this with Seg_Len worsens the results, seems to be the most important goal. Since for the Analytic_Model SEGA calculates accurate estimates of real delays, comparing Num_Seg to Analytic_Model shows that the simple cost function that counts the number of switches traversed by a connection is a good approach.

Finally, the bottom row in Table 2 shows that dividing a multi-point net into a set of two-point connections can have negative effects on speed performance. For Net_Routing, SEGA tries to re-assemble multipoint nets by focusing on not only speed-performance (using Analytic_Model), but also on re-using wire segments for multiple connections that are part of the same net. This is an important point, because Net_Routing achieves the best results.

3.3 Effect of Channel Segmentation on Routing Delays

The previous section showed that in order to achieve good speed-performance of circuits implemented in FPGAs, CAD routing tools should be carefully designed to make good use of the FPGA's wire segments. In this section, we wish to determine what channel segmentation schemes result in the lowest interconnect delays. Stated more precisely, given *W* tracks per channel, from a speed-performance point of view what percentage of tracks should have wire segments of length 1, what percentage should be length 2, length 3, and so on?

As a first experiment, Fig. 3 shows the effects of having all tracks with equal segment lengths of 1, 2, 3, up to 8. There are two curves in the figure; the solid curve represents speed-performance, and the dotted curve shows effects on area. For the solid curve, each point was produced as follows: using the CAD tools that provided the best results from the previous section, routing of each circuit was attempted starting with a low value of *W*. If routing failed, *W* was incremented by one until the circuit could be fully routed. As before, the results are averaged over all 13 benchmarks.

1. With 30 tracks per channel, there are many segmentation schemes. The results provided in Table 2 are averaged over all possible segmentations.

SEGA Cost Function	PgaRoute without Bend Reduction	PgaRoute with Bend Reduction
Area	16.7	12.8
Seg_Len	19.0	16.9
Num_Seg	14.9	11.9
Seg_Len + Num_Seg	16.4	13.3
Analytic_Model	14.8	11.9
Net_Routing	13.0	10.1

Table 2 - *Effect of CAD Routing Tools on Routing Delays.*

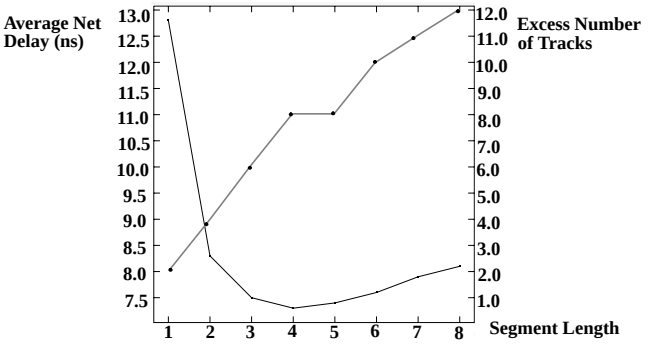

Figure 3 - *Effect of Segment Lengths on Speed and Area.*

The most important point to notice from Fig. 3 is that segmentation length can have a huge effect on net delay. Referring to the solid curve, average net delay decreases by more than 40 percent when the segment length is increased from 1 to 4. Also, increasing segment length beyond 4 slightly degrades performance. This occurs because as segment length is increased the number of switches that connections need to pass through will decrease. However, once the segments become longer than the connections themselves, excess capacitance results in increased delays. The other interesting point illustrated by the dotted curve in the figure is that as segment lengths increase, the number of tracks that the CAD tools require in order to implement the circuits also increases. This has two effects: it contributes to increased capacitive loading, and it results in wasted chip area. Considering both curves in Fig. 3, it seems likely that in order to decrease interconnect delays and yet avoid an excessive number of tracks, it would be better to have a mixture of tracks, some with long segments and others with short ones.

The above experiment was repeated with channels having combinations of segments of length 1, 2, and 3 and the results appear in Table 3. Each entry in the table represents the average net delay produced by a particular segmentation scheme. The horizontal axis represents the percentage of tracks that are of length 3, the vertical axis is percentage length 2, and the remaining tracks are length 1. In the table, the shades of gray represent relative speed-performance; the lighter the shade, the better the result. The minimum delay achieved is 7.4 ns.,

100	8.3										
90	8.4	8.2									
80	8.4	8.1	8.0								
70	8.6	8.3	8.0	7.9							
60	9.0	8.5	8.2	7.9	7.7						
50	9.5	8.9	8.4	7.9	7.7	7.6					
40	10.3	9.5	8.7	8.2	7.9	7.5	7.5				
30	11.1	10.1	9.4	8.7	8.2	7.7	7.5	7.4			
20	11.9	11.0	10.2	9.3	8.6	8.0	7.7	7.4	7.5		
10	12.4	11.7	10.9	10.1	9.2	8.3	7.9	7.6	7.4	7.5	
0	12.8	12.5	11.8	10.8	9.9	9.0	8.4	7.9	7.5	7.4	7.5
	0	10	20	30	40	50	60	70	80	90	100

Table 3 - *Routing Delays of Segmentation Schemes.*

100	1	5											
90	1	4	1	5									
80	1	3	1	4	1	5							
70	1	4	1	4	1	5	1	5					
60	0	4	1	4	1	4	1	5	1	6			
50	0	4	0	4	0	4	1	5	1	5	1	6	
40	0	3	1	4	1	4	1	4	1	5	1	5	1	6
30	0	4	1	4	1	4	1	5	0	4	1	5	1	5	1	6
20	0	4	0	4	0	4	0	5	1	4	1	5	1	5	1	6	1	6
10	0	3	0	4	0	4	1	4	1	5	1	5	1	5	1	5	1	6	2	6
0	0	2	0	3	0	4	0	4	1	5	1	5	1	5	1	5	1	5	1	6	2	6
	0		10		20		30		40		50		60		70		80		90		100	

Table 4 - *Area Penalty for Segmentation Schemes.*

with several combinations being close to this value. A characteristic shared by all segmentations close to the minimum delay is that more than 80 percent of their tracks have segment length greater than 1. This leads to a fundamental conclusion that most of the tracks in a channel should have wire segments longer than length 1, and only a few tracks need short wire segments. In fact, since the result for 100 percent length 3 is near the minimum, this indicates that it may be acceptable if none of the tracks have length 1 segments. We address this last point more closely below.

Intuitively, some tracks with length 1 segments should provide speed-performance savings, particularly for short connections. Since Table 3 contradicts this, we performed an additional experiment in which, for all segmentations in Table 3, only the delays of connections of length 1 and 2 were measured. Although the numbers are not shown here, the results showed that including length 1 segments improves the performance of short connections by about 12 percent on average. This additional information leads us to select the combinations indicated by **bold** outlines in Table 3 as the best segmentation scheme candidates.

The above experiments were repeated for all possible combinations of tracks with segments of length 1, 2, 3 **and** 4. The results showed a negligible improvement over the values in Table 3, which means that, from a speed-performance perspective, it is not necessary to include wire segments of length greater than 3 [1]. This is also supported by Fig. 3, which shows minimum delays around length 3 or 4.

The above conclusions consider only speed-performance and do not account for effects on area. However, Fig. 3 shows that increasing segment lengths can have a great effect on the number of tracks needed, and thus area should not be ignored. To address this point, Table 4 shows the number of tracks in excess of the minimum achievable for each segmentation scheme from Table 3. Consider first only the numbers in the columns shaded grey in Table 4, which provide the excess tracks for the same CAD tool cost functions used in Table 3. Referring to the table, the numbers show that the "best" segmentation schemes from Table 3 imply 5 or 6 extra tracks per channel. For the circuits being routed, this corresponds to a significant increase in tracks of about 30 percent. Now consider the unshaded columns in Table 4, which correspond to the same information as the shaded ones, except that they show how many excess tracks would be needed if, instead of using the speed-performance cost function, the CAD tools had focused only on routability. In this case, only 1 extra track would be needed. The basic conclusion that can be reached from this experiment is that, while CAD tools must use speed-performance oriented cost functions to achieve low routing delays, for connections whose delays are not important the CAD tools should focus on routability, so that fewer tracks per channel will be needed. Since in real circuits many nets are not time-critical, employing a combination of speed- and area-based cost functions is reasonable in practice.

1. This comment applies to wire segments used for general purpose interconnect. An FPGA should always contain some long wires that span the entire device, for distributing high fan out nets like clocks.

4 Conclusions

This paper has addressed several key issues associated with channel segmentation for array-based FPGAs. Conclusions supported by the research results are that CAD routing tools have a significant impact on speed-performance, and the lengths of wire segments in the channels has a very large effect on interconnect delays. The single most important factor, from the perspective of both CAD and architecture, is to limit the number of switches that signals pass through in series. Also, the effect on area has been considered, and the basic conclusion is that while CAD tools should focus on minimizing delays for time-critical nets, in order to reduce the number of tracks needed routability should be prioritized for non-critical nets.

5 References

[1] J. Frankle, "Iterative and Adaptive Slack Allocation for Performance-driven Layout and FPGA Routing," 29th *ACM/IEEE DAC*, pp. 536-542, June 1992.

[2] K. Zhu and D. Wong, "On Channel Segmentation Design for Row-Based FPGAs," *ICCAD*, pp. 26-29, 1992.

[3] K. Roy and M. Mehendale, "Optimization of Channel Segmentation for Channelled Architecture FPGAs," *CICC*, pp. 4.4.1-4.4.4, 1992.

[4] PREP Benchmark Suite #1, Version 1.2, Programmable Electronics Performance Corporation, March 1993.

[5] S. Brown, R. Francis, J. Rose and Z. Vranesic, "Field-Programmable Gate Arrays," Kluwer Academic Publishers, 222 pages, 1992.

[6] J. Rose, "Parallel Global Routing for Standard Cells," *IEEE Trans. CAD*, Vol. 9, No. 10, pp. 1085-1095, Oct. 1990.

[7] B. Tseng, J. Rose and S. Brown, "Using Architectural and CAD Interactions to Improve FPGA Routing Architectures," *First International ACM/SIGDA Workshop on FGPA*, pp. 3-8, February 1992

[8] G. Lemieux and S. Brown, "A Detailed Router for Allocating Wire Segments in FPGAs," *ACM Physical Design Workshop*, CA, 1993.

[9] M. Khellah, S. Brown, and Z.Vranesic, "Modelling Routing Delays in SRAM-based FPGAs," *CCVLSI*, Banff, Canada, Nov. 1993.

[10] M. Khellah, Technical Report, University of Toronto.

[11] J. Rose and S. Brown, "Flexibility of Interconnection Structures in FPGAs," *IEEE JSSC*, Vol. 26 No. 3, pp. 277-282, March 1991.