

Modelling Routing Delays in SRAM-based FPGAs

Muhammad Khellah, Stephen Brown and Zvonko Vranesic

Department of Electrical and Computer Engineering

University of Toronto

Toronto, Canada M5S 1A4

E-mail: khellah|brown|zvonko@eecg.toronto.edu

Abstract

This paper presents an efficient technique for estimating the propagation delay of signals in SRAM-based FPGAs, by using an analytic model of MOS integrated circuits. The model provides a facility for experimenting to find the effect that different routing structures in SRAM-based FPGAs have on the speed-performance of implemented circuits. To illustrate the applicability of the technique, two examples of its use are provided.

1 Introduction

Having been introduced in 1985 by Xilinx, several different types of FPGAs are now commercially available from an assortment of companies. While each manufacturer's product offers unique features, all FPGAs share some common characteristics, like an array of logic blocks and programmable interconnect resources. A number of technologies are currently used for implementing the programmable elements in the interconnect, including static RAM, EPROM, EEPROM, and anti-fuse. This paper focuses on SRAM technology because it is widely used, being offered in FPGAs manufactured by Xilinx[1], Altera[2], AT&T [3], Atmel[4] and Algotronix[5].

A key aspect in the design of an FPGA is its routing architecture, which comprises the resources that are used to interconnect the device's logic blocks. In early FPGAs [1], the interconnect consisted mostly of short wire segments that spanned the length or width of one logic block. Longer segments could be formed by joining together two or more of these short segments via programmable routing switches. While this approach provides for good utilization of the wire segments 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 SRAM-based FPGAs normally use pass-transistors to implement routing switches and this kind of switch has significant series resistance and parasitic capacitance. To address these issues, more recent

SRAM-based FPGAs include wire segments of various lengths, but research has not yet determined what specific segmentation schemes produce the best results.

A hypothetical SRAM-based FPGA is depicted in Figure 1. The figure shows an FPGA with a two-dimensional array of logic blocks, and both horizontal

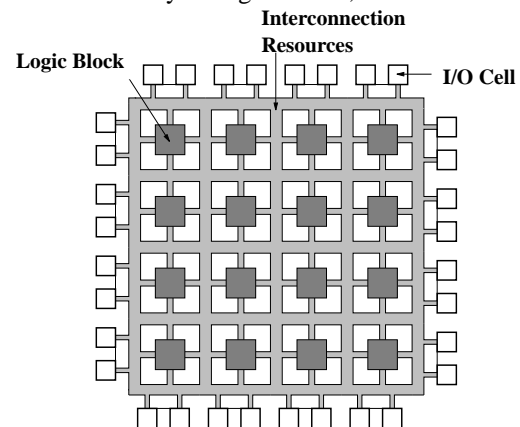


Figure 1 - A Hypothetical SRAM-based FPGA.

and vertical routing channels¹. Although the figure does not include the details of the interconnection resources, routing switches would exist for two purposes: 1) to connect the pins of the logic blocks to the wire segments in the channels, and 2) to connect one wire segment to another. Two examples of how SRAM cells could be used to control the routing switches in this type of FPGA are illustrated by Figure 2.

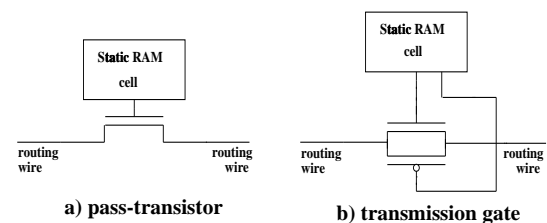


Figure 2 - SRAM-based Routing Switches.

Figure 2(a) shows an SRAM cell controlling a sin-

1. While commercial FPGA architectures differ, this paper assumes the general structure shown in Figure 1, without loss of generality.

gle NMOS pass-transistor and Figure 2(b) represents an SRAM cell controlling two pass-transistors connected as a transmission gate. The relative merits of the two types of routing switches shown in Figure 2 are evaluated in Section 3 of this paper.

The purpose of the work presented in this paper is to facilitate experimentation with different routing architectures by providing a computationally efficient technique for estimating the speed-performance of SRAM-based FPGAs that use pass-transistors to implement routing switches. The technique is based on an analytic model [6] that sets upper and lower bounds for signal delay in MOS integrated circuits. Routing switches are represented in the model as simple resistors and capacitors, which is discussed in detail in Section 2. Section 3 compares the delay estimates produced by the analytic model with true transistor delays as measured by the de facto industry standard HSPICE simulator [7]. Section 4 provides some examples that illustrate the usefulness of the delay estimation technique, and Section 5 provides concluding remarks.

2 Delay Modelling of Pass-Transistor Routing Switches

As mentioned above, the analytic model developed in [6] is employed to estimate routing delays in SRAM-based FPGAs. In this model, MOS transistors are represented as constant RC elements. While other more sophisticated models have appeared in the literature [8] [9], the method in [6] has been chosen because it is easy to implement, computationally simple to evaluate, and we can show that it is accurate for our purposes. For brevity, the internal workings of the analytic model are not explained here, but we describe the way in which inputs to the model are generated from routed FPGA circuits. The reader interested in the details of the model itself can refer to the original publication.

The input to the analytic model is an RC tree network, which is a set of interconnected resistors with capacitors from each node in the network to ground. As output, the model produces an upper and lower bound for the delay from the source node of the network to each of the sink nodes.¹ In order to generate the required RC tree for a net routed in an FPGA, the pass-transistor switches and wire segments involved must be modelled by resistive and capacitive elements. A net is routed in an FPGA by selecting wire segments to connect to each logic block pin on the net and then linking these wire segments together through routing switches. For any given net, there usually will be many

1. Our implementation of the analytic model, written in the C programming language, is available on request by anonymous ftp.

ways in which the required connectivity can be achieved, but ultimately one specific path through the routing channels will be chosen by the CAD routing tools.

An example of a small (two-point) net is illustrated in Figure 3. It shows three alternative ways of

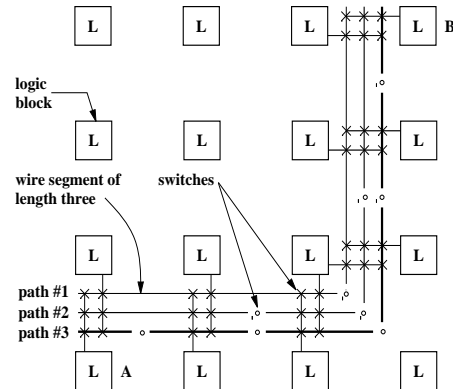


Figure 3 - Alternative Routing Choices for a Net in an SRAM-based FPGA.

connecting a pin at logic block A to a pin at block B. In the figure, routing switches are represented in two ways: a switch that connects a logic block pin to a wire segment is shown as an X, and a switch that connects one wire segment to another appears as a small circle (for clarity, only the routing switches in the rightmost and lower routing channels are shown in the figure, and only one pin is drawn for each logic block). The paths in the figure are labelled as paths #1, #2, and #3 for later reference. In order to build an RC tree from the information depicted in Figure 3, the following questions must be answered:

- How do we model a switch that is turned ON?
- How do we model a switch that is turned OFF?
- How do we model a wire segment?

These questions are addressed in the following subsections.

2.1 Modelling an ON Routing Switch

An ON switch can be modelled as a resistor bounded by two capacitors to ground, as shown in Figure 4c [10].

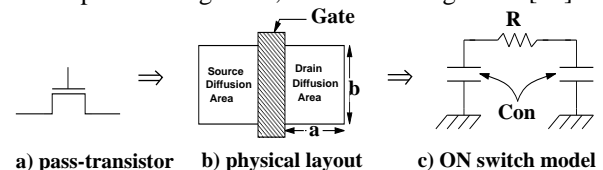


Figure 4 - Modelling an ON Routing Switch.

The resistor represents the ON resistance of the switch and the capacitors corresponds to the parasitic capacitance at the drain and source of the device. This simple model can be applied to an NMOS or PMOS pass-transistor, or a transmission gate. At first glance, this model

may not seem appropriate because the resistance of a MOS transistor is non-linear, but the next subsection (along with Section 3) shows that it is possible to pick a value of resistance that accurately reflects the delay characteristics of the device.

2.1.1 The Resistance of an ON Switch

To determine the resistance of an ON switch, we use the method described in [11], as follows. One side of the transistor is connected to a DC voltage source that can be varied from zero to five volts and the other side of the device is connected to ground. As the voltage across the switch is increased, the current is measured (using HSPICE) and the resistance is calculated using Ohm's law. As an example, Figure 5 shows the

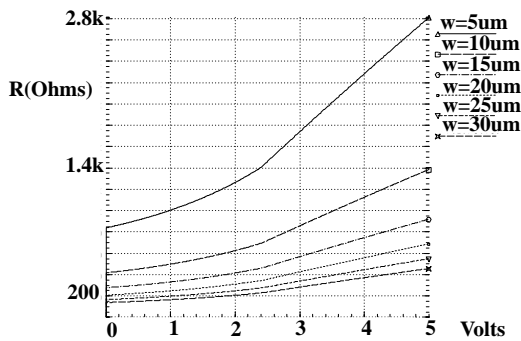


Figure 5 - Measuring the Resistance of an NMOS Pass-Transistor Switch.

results of simulations to measure the ON resistance of an NMOS pass-transistor assumed to be built using a 0.8μ BiCMOS technology [12]. Each of the curves in the figure represents a different value of transistor channel width, from 5 microns for the highest curve to 30 microns for the lowest. As the figure shows, the ON resistance is a monotonically increasing non-linear function of the applied voltage. The same experiment was performed for a transmission gate switch and it was found to have a lower ON resistance than the single NMOS pass-transistor, as one would intuitively expect. This is discussed further in Section 3.

2.1.2 The Capacitance of an ON Switch

The capacitance at the source (or drain) of the ON switch can be approximated as follows [13]:

$$C_{on} = C_{drain}(C_{source}) = C_{diffusion} + \frac{1}{2} \times C_{gate} \quad 1$$

$C_{diffusion}$ consists of two parts, called the junction area capacitance and junction sidewall capacitance. $C_{diffusion}$ is then defined according to:

$$C_{diffusion} = C_{ja} \times (ab) + C_{jsw} \times (2a + 2b) \quad 2$$

where C_{ja} is the junction area capacitance per square and C_{jsw} is the junction sidewall capacitance per unit length. Note that C_{ja} and C_{jsw} are functions of the

voltage across the diffusion/substrate junction, but we approximate this by taking the average value. The variables a and b are the dimensions of the diffusion area as illustrated by Figure 4(b). The C_{gate} term in Equation 1 can be approximated as

$$C_{gate} = \frac{\epsilon_0 \times \epsilon_{sio2}}{T_{ox}} \times A \quad 3$$

where ϵ_0 and ϵ_{sio2} are the permittivity of vacuum and silicon dioxide respectively, A is the active area of the gate and T_{ox} is the thickness of the thin oxide between the gate and the substrate.

2.2 Modelling an OFF Routing Switch

An OFF switch can be modelled simply as two capacitors, called C_{off} , separated by an infinitely large resistance. The value of C_{off} is equal to $C_{diffusion}$ as given by Equation 2.

2.3 Modelling a Wire Segment

A wire segment in an FPGA may span the length or width of a single logic block or it may be longer. Since wire segments are implemented as metal lines they have negligible resistance, but their parasitic capacitance can be significant. Thus, a wire segment is modelled as a capacitor, C_{ws} , between the node corresponding to the wire segment and ground. As in the case of the diffusion capacitance (Equation 2), C_{ws} consists of two parts: area and sidewall capacitance, as defined in Equation 4:

$$C_{ws} = C_{ma}(SegArea) + C_{msw}(SegPerimeter) \quad 4$$

C_{ma} and C_{msw} are the metal area capacitance per square and the metal sidewall capacitance per unit length, respectively.

Routing Switch Transistor Width (um)	C_{off} (ff)	C_{gate} (ff)	C_{on} (ff)	Ron_max * Con (psec)
5	4.9	7.9	8.8	24.7
10	8.8	15.8	16.7	23.3
15	12.7	23.7	24.6	22.5
20	16.7	31.6	32.5	22.3
25	20.6	39.5	40.4	22.1
30	24.6	47.4	48.3	22.0

Table 1 - Capacitance values for a 0.8μ BiCMOS N-Channel Pass-Transistor Switch.

Typical values of capacitance as calculated by the above equations are listed in Table 1. The table gives the values of C_{off} , C_{gate} , and C_{on} for several values of transistor channel width, assuming an NMOS transistor built in the same 0.8μ BiCMOS technology [12]. In addition, the right-most column of Table 1 shows the product of C_{on} with the maximum switch resistance (R_{on_max}) taken from Figure 5. This column indicates the inherent delay of the switch, and shows that the

delay levels off as the transistor size increases (because R_{on_max} decreases, but Con increases). This issue is discussed in more detail in Section 4.

2.4 Example of an RC Tree Network

As an example of an RC tree network, Figure 6 shows

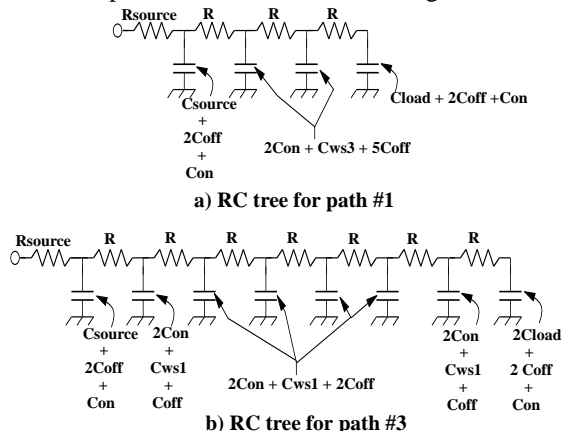


Figure 6 - Example of an RC Tree Network.

the RC tree corresponding to paths #1 and #3 for the net illustrated in Figure 3. For each of the paths, it is assumed that the logic block driving the net is characterized by an output resistance and capacitance called R_{source} and C_{source} . Also, a load capacitance called C_{load} is added at the sink end of the path. Referring to Figure 6a, it can be seen that path #1 involves three resistors, labelled R , that correspond to routing switches. Also, the values of the capacitors account for the number of routing switches in path #1 that are turned ON, the number of OFF switches that “hang on” the wire segments in the path, and the capacitance of the wire segments (C_{ws3} represents the capacitance of a wire segment that spans 3 logic blocks). Similarly, Figure 6b corresponds to path #3, showing that this path involves many more series resistors (seven) because it passes through more series routing switches. In the next section, we will compare the speed performance of both paths, using both HSPICE and our delay model.

3 Delay Model versus HSPICE

This section shows that the delay estimates produced by our technique are accurate as compared to HSPICE simulations. For this experiment, we assume that 50 (0.8u-BiCMOS n-channel) pass-transistors are connected in series and each transistor has a channel width of 15 microns¹. The input to the first transistor is a step function that is varied from zero to five volts. The last transistor is connected to a capacitive load. Values

1. The same experiment was carried out using different transistor sizes, with similar results.

for Con are taken from Table 1. For the ON resistance, results are given for two values: R_{on_ave} and R_{on_max} . R_{on_avg} is the average value of the switch resistance (480 Ω) over the voltage range given in Figure 5 and R_{on_max} is the maximum value of resistance (900 Ω) in Figure 5. Figures 7 and 8 show the results of comparing delay estimates produced by our method with HSPICE. Both figures give the delay at the output of each pass-transistor in the sequence. As Figure 7

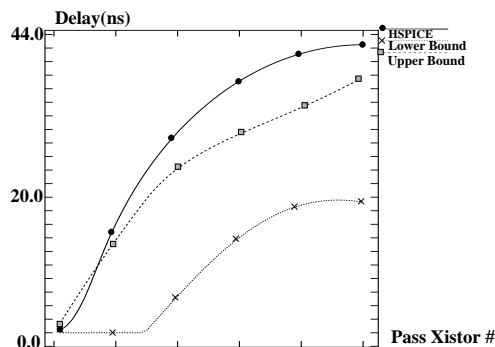


Figure 7 - Comparison of the Delay Model Results with HSPICE Using R_{on_ave} .

indicates, the delay estimates produced by the analytic model do not bound the real delays when using R_{on_avg} , but Figure 8 shows that the real delays are bounded by the model when R_{on_max} is used. Since the number of transistors in series in this experiment is

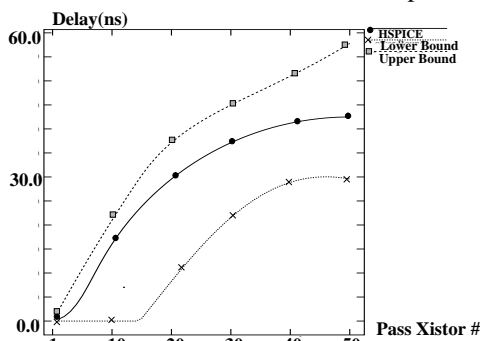


Figure 8 - Comparison of the Delay Model Results with HSPICE Using R_{on_max} .

more than would normally be expected in an FPGA, Figure 8 shows that if the switch resistance is set to R_{on_max} , the analytic model provides an accurate estimate of the delay of routing switches in SRAM-based FPGAs.

As an example of real nets in an FPGA, Table 2 shows the delays of both paths #1 and #3 from Figure 3 as calculated by both HSPICE and the analytic model. In this case, we show the results for both a single pass-transistor and a transmission gate for comparison purposes. Columns 2 to 5 show the delay of path #1, produced by HSPICE simulation for transmission gate (TG) switches, HSPICE simulation with NMOS pass-

transistor (**PT**) switches, and by the lower (**L**) and upper (**U**) bounds of the delay model (the RC-tree was shown in Figure 8a). The same information for path #3 is given in columns 6 to 9. Again, values are shown for a range of transistor widths¹ and indicate that the delay levels off as the transistor size is increased. The table shows that path #1 is much faster than path #3 (by an average of 55 percent) because of the lesser number of routing switches traversed. It also shows that the delay model accurately bounds HSPICE simulations (compare the **PT** column with the **L** and **U** columns for both paths).

Width (um)	Path #1 Delay (ns)				Path #3 Delay (ns)			
	TG	PT	L	U	TG	PT	L	U
5	0.8	0.9	0.6	1.0	2.2	2.4	1.8	3.0
10	0.8	0.7	0.5	0.8	2.1	1.8	1.4	2.4
15	0.8	0.6	0.4	0.8	2.1	1.7	1.2	2.3
20	0.8	0.6	0.4	0.8	2.1	1.6	1.2	2.2
25	0.9	0.6	0.4	0.8	2.1	1.6	1.2	2.2
30	1.0	0.6	0.4	0.8	2.1	1.5	1.2	2.2

Table 2 - Delay of Path #1 and Path #3 (Figure 3)

Comparison between the **TG** and the **PT** results for the two paths shows that the transmission gate switch actually performs worse than the single NMOS transistor. While this might seem surprising at first, it occurs because, although the transmission gate has a lower resistance than a single pass-transistor (as mentioned before), the PMOS transistor adds a significant amount of parasitic capacitance. This capacitance is particularly important in an FPGA because it not only affects the *Con* values, but also *Coff* (Recall from Figure 6 that many *Coff* capacitors “hang on” the wire segments in an FPGA).

Having illustrated the accuracy of the delay modelling technique, we will now present two examples of its use: 1) to evaluate different cost functions in a CAD routing tool for FPGAs, and 2) to evaluate the effect of routing switch size on the speed performance of real FPGA circuits.

4 Examples Utilizing the Delay Estimation Technique

The experimental results presented in this section are based on real industrial benchmark circuits implemented in a hypothetical FPGA (Figure 1). Several stages are involved in a CAD system that maps a circuit into an FPGA [14], including logic optimization, technology mapping, placement, and routing. Since this paper is concerned with measuring the speed-performance of the final routed circuits, we focus only on the final step in the CAD system, in which the routing

1. For the transmission gate case, the channel width numbers are for the NMOS transistor, with the PMOS being twice as large.

resources are allocated to implement the required connectivity between logic blocks. Some characteristics of the benchmark circuits, which are in the appropriate format for input to the CAD routing tool, are listed in Table 3. The table gives the size of each circuit, both in terms of the number of two-point connections² and the number of connected logic blocks. Each of the circuits is from the Microelectronics Centre of North-Carolina (MCNC) benchmark suite.

Circuit Name	# Logic Blocks	# Two-Point Connections
too_large	156	519
example2	120	444
vda	210	722
alu2	143	511
alu4	255	851

Table 3 - Characteristics of Benchmark Circuits

The FPGA used to implement the benchmarks comprises a routing architecture that consists of wire segments of various lengths. For all of the examples given here, we average the results over many segmentation schemes that have segments of lengths one, two and three.

4.1 Evaluating the Speed-Performance of Different Cost Functions

As illustrated earlier in the previous sections, the specific wire segments chosen for a connection routed in an SRAM-based FPGA can have a significant effect on speed-performance. In order for a CAD routing tool to account for this, it must have some means of evaluating the speed-performance of routing alternatives. This ability has been implemented into a CAD routing tool for FPGAs, called SEGA [15] [16] by using a cost function, called $Delay(p)$. When routing a connection as part of a circuit, SEGA assesses the speed of each possible path by evaluating $Delay(p)$ either as:

$$Delay(p) = w_1 \times NumSeg(p) + w_2 \times LenSeg(p) \quad 5$$

or as:

$$Delay(p) = Analytic(RC) \quad 6$$

For brevity, the details concerning Equation 5 will not be described, but the basic idea is that it accounts for the number of wire segments in a path, p , using the $NumSeg(p)$ term, and the lengths of those segments via the $LenSeg(p)$ term. This type of cost function has also been utilized in other routing tools for FPGAs [17] [18]. However, it is not clear, what weights, w_1 and w_2 , should be used in equation 5. In Equation 6, $Delay(p)$

2. The CAD routing tool requires that multi-point nets are divided into two-point connections. However, all delay measurements are calculated for multi-point nets because the delay of individual connections is affected by being part of a larger net.

is evaluated by representing p as an RC tree and then estimating its delay using the analytic model, as was shown for the example in Section 3. Table 4 compares the quality of results produced by the above two cost functions where the numbers in the table (given in ns) represent the average delay of a net in each circuit as estimated by the analytic model¹. Comparing the results in column 2 to those in columns 3 and 4 of the table shows that decreasing the number of segments (and so the number of ON switches) a net has to pass through, is the most critical factor in reducing its routing delay. Similar speed performance is also found by the analytic delay function as given in column 5.

Circuit Name	EQ 5 with w1 = 1, w2 = 0	EQ 5 with w1 = 0, w2 = 1	EQ 5 with w1 = 1, w2 = 1	Analytic Delay Function (EQ6)
too_large	12.7	18.5	13.7	12.6
example2	8.7	12.5	9.3	8.6
vda	16.5	24.2	18.1	16.4
alu2	9.0	13.2	10.0	9.1
alu4	13.3	19.7	14.7	13.4
Average	12.0	17.6	13.2	12.0

Table 4 - Speed Performance Results Using Four Delay Cost Functions

4.2 Measuring the Effect of Switch Transistor Size on Speed-performance

Table 5 shows the results of an experiment to measure the effect that varying the size (channel width) of an NMOS pass-transistor switch has on the speed-performance of the benchmark circuits. In the experiment, each of the circuits was routed using SEGA with routing switch size that varies from 5 to 30 μm . As in earlier experiments, the transistor RC parameters were taken from the 0.8 μm BiCMOS technology.

Circuit Name	5	10	15	20	25	30
too_large	12.8	10.9	10.4	10.3	10.3	10.4
example2	9.1	7.8	7.4	7.4	7.4	7.5
vda	17.1	14.5	13.8	13.7	13.7	13.7
alu2	9.7	8.3	7.9	7.9	7.9	8.0
alu4	14.8	12.6	12.0	11.0	11.9	11.9
Average	12.7	10.8	10.3	10.1	10.2	10.3

Table 5 - Effect of Switch Transistor Size on Speed-Performance

The table shows that increasing the switch channel width from 5 to 15 microns significantly reduces delay, but there are diminishing returns beyond that point. This result is consistent with the data in Table 1, where the inherent delay of a single pass-transistor was shown to level out beyond a size of 15 microns. As stated earlier with respect to Table 1, the reason that

1. Recall that the analytic model produces both an upper and lower bound for the delay. For the results given here, we always use the upper bound.

the delay flattens out as transistor size increases is that, although resistance drops as the transistor size is increased, parasitic capacitance increases.

5 Conclusions and Future Work

In this paper, we have described an efficient technique for estimating routing delays in SRAM-based FPGAs. The technique has been shown to be accurate by comparison to HSPICE simulations and is important because it facilitates experimentation into the speed-performance of different FPGA routing architectures. We have illustrated the applicability of the technique by providing two example of its use.

In future work, the technique will be used for extensive experimentation into the effect of different FPGA routing architectures on speed-performance. This includes a study of channel segmentation as well as other routing architecture parameters.

References

- [1] W. Carter, K. Duong, R. H. Freeman, H. Hsieh, J. Y. Ja, J. E. Mahoney, L. T. Ngo and S. L. Sze, "A User Programmable Reconfigurable Gate Array," *Proc. 1986 Custom Integrated Circuits Conference*, May 1986, pp. 233-235.
- [2] ALTERA Corporation, "FLEX Programmable Logic," *Product Information Bulletin*, September 1992.
- [3] AT&T Microelectronics, "Optimized Reconfigurable Cell Array (ORCA) Series Field-Programmable Gate Arrays," *Advanced Data Sheet*, February 1993.
- [4] Concurrent Logic Inc., "CLi6000 Series Field-Programmable Gate Arrays Data Sheets," May 1992.
- [5] Algotronix, "CAL1024 Preliminary Data Sheets," 1992.
- [6] J. Rubinstein, P. Penfield and M. Horowitz, "Signal Delay in RC Tree Networks," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, Vol. CAD-2, No. 3, July 1983.
- [7] Meta-Software, Inc. "HSPICE User's Manual," 1991.
- [8] A. C. Deng, and Y. C. Shiau, "Generic Linear RC Delay Modeling for Digital CMOS Circuits," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, VOL. 9, No. 4, April 1990.
- [9] L. T. Pillage and R. A. Rohrer, "Asymptotic Waveform Evaluation for Timing Analysis," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, VOL. 9, No. 4, April 1990.
- [10] C. Mead and L. Conway, "Introduction to VLSI Systems," Addison-Wesley Publishing Company, 396 pages, 1980.
- [11] Adel S. Sedra and Gordon W. Roberts, "SPICE for Micro-electronic Circuits," Saunders College Publishing, 630 pages, 1985.
- [12] Canadian Microelectronics Corporation, "Design Rules for CMC 0.8-Micron BiCMOS," February 1993.
- [13] Neil Weste and Kamran Eshraghian, "Principles of CMOS VLSI Design," Addison-Wesley Publishing Company, 531 pages, 1988.
- [14] S. Brown, R. Francis, J. Rose and Z. Vranesic, "Field-Programmable Gate Arrays," Kluwer Academic Publishers, 222 pages, 1992.
- [15] G. Lemieux and S. Brown, "A Detailed Router for Allocating Wire Segments in FPGAs," *ACM Physical Design Workshop*, California, April 1993.
- [16] S. Brown, G. Lemieux and M. Khellah, "Segmented Routing for Speed-Performance and Routability in Field-Programmable Gate Arrays," Submitted to *VLSI Design Journal* in June 1993.
- [17] J. Greene, V. Roychowdhury, S. Kaptanoglu, and A. El Gamal, "Segmented Channel Routing," *Proc. 27th Design Automation Conference*, pp. 567-572, June 1990.
- [18] K. Roy and M. Mehendale, "Optimization of Channel Segmentation for Channelled Architecture FPGAs," *Proc. 1992 Custom Integrated Circuits Conference*, pp. 4.4.1-4.4.4., May 1992.