Flexibility of Interconnection Structures for Field-Programmable Gate Arrays

Jonathan Rose, *Member, IEEE*, and Stephen Brown, *Student Member, IEEE*

Abstract—**This paper explores the relationship between the routability of a field-programmable gate array (FPGA) and the flexibility of its interconnection structures. The flexibility of an FPGA is determined by the number and distribution of switches used in the interconnection. While good routability can be obtained with a high flexibility, a large number of switches will result in poor performance and logic density because each switch has significant delay and area.**

The minimum number of switches required to achieve good routability is determined by implementing several industrial circuits in a variety of interconnection architectures. These experiments indicate that high flexibility is essential for the connection block that joins the logic blocks to the routing channel, but a relatively low flexibility is sufficient for switch blocks at the junction of horizontal and vertical channels. Furthermore, it is necessary to use only a few more routing tracks than the absolute minimum possible with structures of surprisingly low flexibility.

I. Introduction

THE architecture of field-programmable gate arrays (FPGA's) is a new and difficult science because the programmable nature of the devices requires complex building blocks. The architecture of an FPGA consists of its logic block function, interconnection structure, and I/O block design. In previous work, we have investigated the effect of logic block functionality on the area of FPGA's [1], [2]. This paper focuses on the design of the interconnection structure and studies the effect of the flexibility on routability and wiring resource requirements. Routability is the percentage of the total number of connections that are successfully routed.

The FPGA was introduced in [3] and newer versions have been presented in [4]–[11]. In some cases these architectures were driven by process technology and the assumption that a user would be able to hand-tune an implementation. In other cases, the architectures were a result of a natural progression of PLD's. It is now clear that these devices will soon be so complex that users must have CAD tools to aid in their design. Thus the architecture should be heavily influenced by its ability to ease the automated design process. Consequently, architectural decisions should be made as the CAD tools are crafted, and should be driven by the characteristics of the CAD algorithms. We use this approach to explore routing structures for FPGA's.

Manuscript received August 1, 1990; revised November 9, 1990. This work was supported by NSERC Operating Grant URF0043298, a research grant from MICRONET, a research grant from ITRC, and a research grant from Bell-Northern Research.

The authors are with the VLSI Research Group, Department of Electrical Engineering, University of Toronto, Toronto, Ont., Canada M5S 1A4.

IEEE Log Number 9041737.

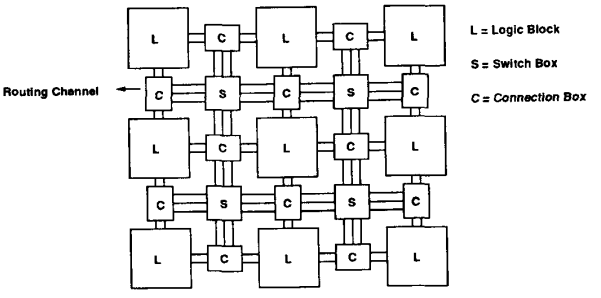


Fig. 1. General routing structure of an FPGA.

There has been little prior work on the FPGA interconnection structure. An earlier version of this paper appeared in [12]. Greene *et al.* [13] present a short discussion on channel segmentation design in the context of a segmented channel routing algorithm.

Fig. 1 depicts the general structure of the FPGA that we consider. The logic blocks (L blocks in the figure) connect to the horizontal and vertical channels using a connection block (C block). Connections are switched at the intersection of horizontal and vertical channels by a switch block (S block). The flexibility of the S and C blocks is the key to obtaining efficient use of the routing area. Flexibility can be loosely defined as the number of choices offered to each wire entering an interconnection block. A high flexibility means a large number of choices, and a low flexibility implies few choices. If the blocks are more flexible than is required to obtain complete connectivity, then there will be more switches than necessary. Since switches are implemented either as pass transistors [4], [9], anti-fuses [6], or EPROM [8], [10], they have significant parasitic capacitance and series resistance, and can take up significant area. Thus an excessive number of switches means that the FPGA will be unnecessarily slow and large. Conversely, if the interconnection structure is insufficiently flexible, then either 100% routing completion will be impossible or a large number of routing tracks per channel will be required and area will be wasted.

The particular questions this paper addresses are as follows.

1) What is the effect of connection (C) block flexibility on the routing completion rate? Experiments indicate that the connection block should have high flexibility because it is otherwise unlikely that any path will be able to connect at its terminating point.

0018-9200/91/0300-0277$01.00 ©1991 IEEE

2) What is the effect of switch (S) block flexibility on the routing completion rate? Results show that low flexibility is sufficient for switch blocks because a cascade of switch blocks will provide many possible paths, even when there are only a few choices at one block.

3) How do the switch block and connection block flexibilities interact? Results indicate that connection block flexibility and switch block flexibility will trade off to achieve the same routability, up to a point. However, unless the flexibilities remain within a particular range, the architecture will require an excessive number of switches.

4) What is the effect of connection and switch block flexibility on the number of tracks per channel required to achieve 100% routing? Experiments show that there is a minimum flexibility beyond which it is easy to achieve near to the minimum possible number of tracks.

This paper is organized as follows. Section II defines the routing structures and implementation procedure used in these experiments. Section III describes the experimental results and Section IV concludes.

II. Experimental Definitions and Procedure

These questions are answered by implementing a set of industrial circuits in a variety of interconnection structures and measuring the percentage routing completion and required number of tracks per channel for each circuit.

The logic block used in these experiments is the one resulting from our previous study [1], [2]. It contains a four-input lookup-table-based combinational block, a D flip-flop, and a tristate output. In the previous study, this block achieved the minimum total area for 12 industrial circuits when compared to many other logic blocks with differing numbers of inputs, including and excluding a flip-flop, over a wide range of programming technologies (the method of FPGA customization).

Another important design parameter in the routing structure is the number of sides of the logic block T on which each logical pin appears physically. If T is high then the routing problem is easier, but greater area and delay occur due to the increased number of switches. For the experiments in this paper, T is set to be two. This was determined by several global routing experiments that showed that a significant improvement in the number of tracks required is achieved for the $T = 2$ case over the $T = 1$ case, but showed diminishing returns for $T > 2$. We note that it is possible to achieve similar flexibility by using $T = 1$ and having the router make use of the functional equivalence of the logic block inputs.

The implementation procedure described in Section II-B will be used to transform each circuit into a variety of FPGA's with interconnection structures of varying flexibility. The purpose is to study the effect of the flexibility of the connection block and the switch block on routing completion rate and routing track requirements. In the following section the flexibility of the interconnection structures are defined. The subsequent section gives the implementation procedure.

A. Flexibility Definitions

The nature of the switch block under consideration is illustrated in Fig. 2. The flexibility of the switch block, F_s, is defined to be the total number of possible connections

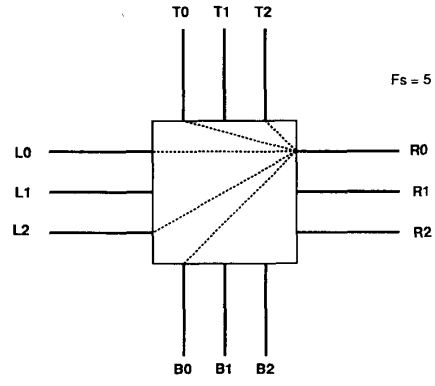


Fig. 2. Definition of flexibility of switch block.

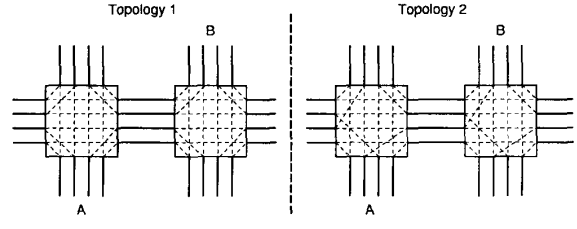


Fig. 3. Two switch block topologies.

offered to each incoming wire. (Note: this definition is different from the one that appeared in [12], where F_s was the number of connections going to each opposing side.) For the example shown in Fig. 2, wire $R0$ can be connected to two wires on the top side ($T0, T2$), two wires on the left side ($L0, L2$), and one wire on the bottom side ($B0$) so that $F_s = 5$, if all the other wires are similarly connected. The implementation of the connections depends on the programming technology—they can be SRAM-driven transistor switches [4], anti-fuses [6], or EPROM-driven transistors [8], among others.

The topology of the interconnections (the actual pattern of the switches) can be very important. It is possible to choose two different topologies with the same flexibility measure (F_s) that result in very different routability. As an illustration of this, consider the two different topologies shown in Fig. 3. Each switch block has the same flexibility measure, $F_s = 2$. Assume that a global router has specified that the two points A and B must be connected by traveling through the two switch blocks shown. It is not possible to reach point B from point A with Topology 1, while it is for Topology 2. Topology 2 is able to make the connection because it is asymmetric about the horizontal and vertical axes. For the results presented in this paper, Topology 2 is used. For higher values of F_s, switches are added such that the basic pattern is preserved.

Fig. 4 illustrates an example connection block. The channel wires flow uninterrupted through the connection block and are connected to logic block pins via a set of switches. An X indicates the position of a switch. The flexibility of the connection block, F_c, is defined as the number of channel wires to which each logical pin can connect. For the example shown in Fig. 4, logic block pin $P0$ can connect to channel

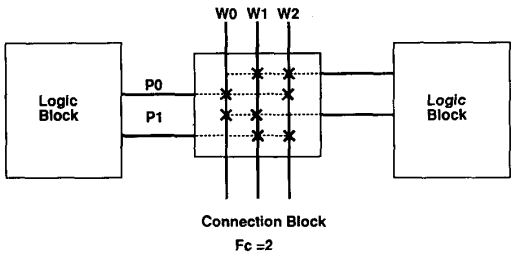


Fig. 4. Definition of flexibility of connection block.

wires $W0$ and $W2$ and so F_c is 2, as all the other pins are connected similarly.

B. Implementation Procedure

The procedure described below realizes each circuit as an FPGA. The inputs are the circuit netlist, the interconnection structure of the switch block (characterized by F_s), and the connection block (characterized by F_c). There are two kinds of output that are produced:

1) the percentage of nets that were routed successfully, given a particular number W of tracks per channel;
2) the number of tracks per channel W required to route 100% of the connections with a given F_s and a given ratio of F_c / W.

The implementation procedure for each circuit is as follows.

1) Perform the technology mapping [14], [15] from the original circuit into the logic block. The result of this step is a new netlist that interconnects only logic blocks, and is functionally equivalent to the original circuit.

2) Perform the placement of the resulting netlist. This is done using the Altor placement program [16], which is based on the min-cut placement algorithm. Altor makes the array as square as possible.

3) Perform the global routing of the circuit. Global routing determines the path of channels that each wire is to take. This procedure gives the number of tracks that would be required in each channel if the switch and connection blocks had full flexibility. The approach used is similar to the LocusRoute standard cell global routing algorithm described in [17].

For the set of parameters F_s, F_c, do the following.

4) Perform the detailed routing of the circuit. For each wire's global path, determine the exact wires and explicit connections in the switch blocks and connection blocks. We have developed a new routing algorithm that is able to take the configuration of the S and C blocks and W as input [18]. If a specific W is given as input, the detailed routing determines the percentage routing completion. Alternatively, if the desired output is the number of tracks required to achieve 100% routing, then the detailed router is invoked with an increasing number of tracks until complete routing is achieved.

III. Experimental Results

The implementation procedure was performed on five circuits from four sources: Bell-Northern Research, Zymos, and two different designers at the University of Toronto. Table I gives the name, size (number of logic blocks and two-point connections), source, and function of each circuit.

TABLE I
Experimental Circuit Characteristics

Circuit	#Blocks	#Conn	Source	Type
BUSC	109	392	UTD1	Bus controller
DMA	224	771	UTD2	DMA controller
BNRE	362	1257	BNR	Random logic and data path
DFSM	401	1422	UTD1	State machine
Z03	586	2135	Zymos	8-b multiplier

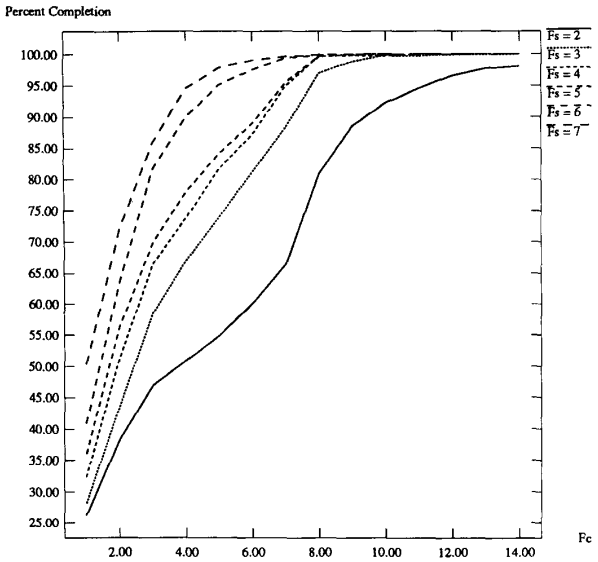


Fig. 5. Percent completion versus F_c for the circuit BNRE.

A. Connection Block Flexibility and FPGA Routability

Fig. 5 is a plot of the percentage routing completion versus connection block flexibility F_c for the circuit BNRE. Each curve in the figure corresponds to a different value of switch block flexibility F_s, which ranges from 2 to 7. The number of tracks W is set to 14, which is two greater than the minimum possible number of tracks as indicted by the global router. The figure indicates that the routing completion rate is very low for small values of F_c and only achieves 100% when F_c is at least one-half of W. The figure also shows that the increasing switch block flexibility improves the completion rate at a given F_c, but to get near 100% the value of F_c must always be high.

Table II summarizes the results for the other circuits. It gives the minimum value of F_c and F_c / W required to achieve 100% routing completion for each circuit for nine values of F_s. W is fixed at two greater than the minimum. The value "nr" in the table indicates that 100% routing was not achieved.

The key observation from the data of Table II is that F_c must be close to W for high routing completion, that is, F_c / W must be between 0.5 and 1. The reason for this is that F_c / W is the probability that a wire arriving on a particular

TABLE II
MINIMUM F_c REQUIRED FOR 100% COMPLETION

Circuit	W	F_s	100% F_c	F_c/W
BUSC	11	2	nr	nr
BUSC	11	3	8	0.73
BUSC	11	4	7	0.64
BUSC	11	5	7	0.64
BUSC	11	6	6	0.54
BUSC	11	7	6	0.54
BUSC	11	8	6	0.54
BUSC	11	9	6	0.54
BUSC	11	10	6	0.54
DMA	12	2	nr	nr
DMA	12	3	8	0.67
DMA	12	4	8	0.67
DMA	12	5	7	0.58
DMA	12	6	7	0.58
DMA	12	7	7	0.58
DMA	12	8	7	0.58
DMA	12	9	7	0.58
DMA	12	10	7	0.58
BNRE	14	2	nr	nr
BNRE	14	3	12	0.86
BNRE	14	4	10	0.71
BNRE	14	5	9	0.64
BNRE	14	6	8	0.57
BNRE	14	7	8	0.57
BNRE	14	8	8	0.57
BNRE	14	9	8	0.57
BNRE	14	10	8	0.57
DFSM	13	2	nr	nr
DFSM	13	3	10	0.77
DFSM	13	4	9	0.69
DFSM	13	5	8	0.61
DFSM	13	6	8	0.61
DFSM	13	7	7	0.54
DFSM	13	8	8	0.61
DFSM	13	9	7	0.54
DFSM	13	10	7	0.54
Z03	13	2	nr	nr
Z03	13	3	10	0.77
Z03	13	4	9	0.69
Z03	13	5	9	0.69
Z03	13	6	9	0.69
Z03	13	7	7	0.54
Z03	13	8	7	0.54
Z03	13	9	7	0.54
Z03	13	10	7	0.54

track at the C block is able to connect to the required physical pin. As F_c/W decreases from 1, it is more likely that wires will not be able to connect to the desired pin, and so routing completion declines.

B. Switch Block Flexibility and FPGA Routability

Fig. 6 is a plot of the percentage routing completion versus F_s, for values of F_c ranging from 1 to W. This plot is for the circuit BNRE. W is again set to 14, two more than the minimum possible number of tracks as indicted by the global router. Clearly, if F_c is high enough, then very low values of F_s can achieve 100% routability. These are low since the maximum value of F_s is $3\times W$. For the results in Fig. 6 this maximum is 42, whereas 100% routing completion is often achieved for $F_s = 3$. This makes intuitive sense because even for $F_s = 3$ every incoming wire to the switch block has at least one way of connecting to each outgoing side. In the general case, it is possible to show that the number of different paths

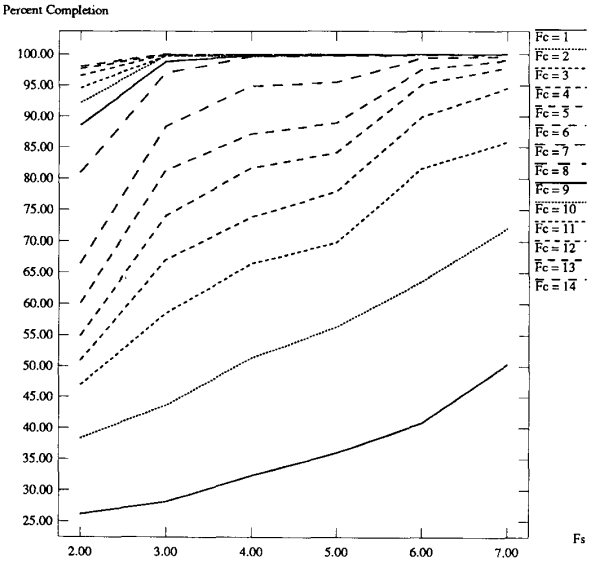


Fig. 6. Percent completion versus F_s for the circuit BNRE.

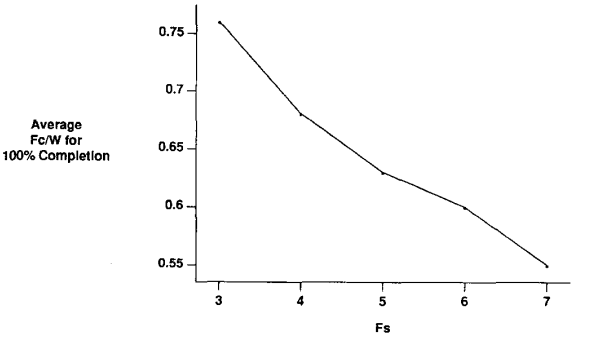


Fig. 7. Average F_c/W for 100% completion versus F_s.

between the initial physical pin and the terminating C block of a two-pin wire is given by

$$\#\text{paths} = F_c \times \left(\frac{F_s}{3}\right)^N$$

where N is the number of switch blocks in the global path. The average value of N for these circuits is approximately 3, and so for the average connection, using $F_s = 6$ and $F_c = 10$, there are 80 different paths. For $F_s = 9$ there are 270 paths. Thus a small increase in flexibility of the switch block vastly increases the number of paths, and hence the routability.

For lower values of F_c, we observe that increasing F_s improves routing completion, up to a point. Fig. 7 is a plot of the average value of F_c/W required to achieve 100% completion versus F_s over all of the circuits. It shows that a more flexible switch block can compensate for a less flexible connection block.

C. Track Count Requirements

Table III gives the number of tracks required to achieve 100% routing completion for a set of different routing architectures for circuit BNRE. Each entry in the table corre-

TABLE III
TRACK COUNT REQUIREMENTS FOR BNRE
(MINIMUM = 12)

F_s	F_c/W 0.6	0.7	0.8	0.9	1.0
2	> 35	> 35	29	20	20
3	18	15	15	13	13
4	16	13	13	13	13
5	15	13	12	13	13
6	15	13	13	13	13
7	14	12	13	13	12

TABLE IV
AVERAGE EXCESS TRACK COUNT REQUIREMENTS OVER ALL CIRCUITS

F_s	F_c/W 0.6	0.7	0.8	0.9	1.0
2	inf	inf	12	10	9.2
3	7.6	3.2	2.0	0.6	0.6
4	3.2	1.4	0.8	0.4	0.4
5	2.2	1.2	0.6	0.4	0.2
6	2.0	0.8	0.6	0.2	0.2
7	1.2	0.0	0.4	0.4	0.0

sponds to a routing architecture with a different pair of $F_s, F_c/W$ values. The value of F_c/W is held constant as W is changed. The minimum possible track count for that circuit is 12, as indicated by the global router. The table shows that with low flexibilities ($F_s \geq 3$ and $F_c/W \geq 0.6$) it is possible to achieve 100% routing completion using very near to the minimum possible number of tracks.

Table IV gives a summary of the results for all the circuits. Each entry in this table is the average number of tracks in excess of the minimum required to achieve 100% routing completion over all of the circuits. The value is "inf" for those cases where 100% routing was not achieved. These results are surprisingly clear: for $F_s = 2$, the penalty in tracks is quite high— greater than nine tracks. For $F_s \geq 3$ excluding the case $F_s = 3$ and $F_c/W = 0.6$, the increase in required tracks over the minimum is no more than three. Thus with very low flexibilities it is possible to achieve a number of tracks only slightly more than the minimum possible.

D. Architectural Decisions

The actual choice of interconnection flexibility must be based on the implementation cost, in area and delay. Since these depend on the programming technology, it is not possible to make general conclusions. It is instructive, however, to measure the cost of an interconnection structure in terms of the total number of switches per tile. A tile is the section of the FPGA that would be replicated across the entire chip. It includes the logic block and the switching structures on two sides of the block. The number of switches in a tile is a function of $W, F_s, F_c/W$, P, and T, where P is the number of logical pins on the block and T is the number of sides of the block on which each logical pin appears. The number of switches in a tile is the sum of the number of switches in the connection and switch blocks. These are given by

$$\#\text{switches in switch block} = 2 \times F_s \times W$$

$$\#\text{switches in connection block} = T \times P \times F_c.$$

TABLE V
NUMBER OF SWITCHES PER TILE FOR EACH ARCHITECTURE

F_s	F_c/W 0.6	0.7	0.8	0.9	1.0
2	434	482	440	331	360
3	259	236	258	241	260
4	262	231*	249	267	286
5	276	257	254	293	312
6	306	283	301	319	338
7	313	285	327	345	336

Table V gives the number of switches used to implement circuit BNRE in a variety of architectures, calculated using the above equations. The values of W are taken from Table III. The number of switches per track increases as flexibility (F_s and F_c/W) increases, but the total number of tracks may decrease as shown in Table III. Hence there should be an architecture that exhibits a minimum total number of switches. This is evident in Table V as the minimum occurs at 231 switches with $F_s = 4$ and $F_c/W = 0.7$. Note that several neighboring architectures have similar switch counts. For all of the test circuits the minimum number of switches occurred when the architecture's parameters were in the range $3 \leq F_s \leq 4$ and $0.7 \leq F_c/W \leq 0.9$.

IV. CONCLUSIONS AND FUTURE WORK

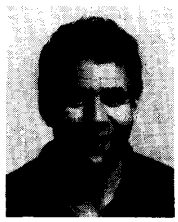
This paper has explored the relationship between routing structure flexibility and routability of FPGA's. The principal conclusions are that connection blocks should have high flexibility to achieve high-percentage routing completion, but that switch blocks require limited flexibility. Furthermore, it has been shown that with low flexibilities it is necessary to use only a small number of tracks in excess of the minimum. Finally, it has been shown how to choose the flexibility parameters so as to minimize the number of switches in the routing architecture.

In the future we will investigate routing architectures with segments of varying lengths and architecture that are optimized for performance.

REFERENCES

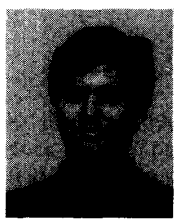
[1] J. S. Rose, R. J. Francis, P. Chow, and D. Lewis, "The effect of logic block complexity on area of programmable gate arrays," in *Proc. CICC*, May 1989, pp. 5.3.1–5.3.5.
[2] J. S. Rose, R. J. Francis, D. Lewis, and P. Chow, "Architecture of field-programmable gate arrays: The effect of logic block functionality on area efficiency," *IEEE J. Solid-State Circuits*, vol. 25, no. 5, pp. 1217–1225, Oct. 1990.
[3] W. Carter *et al.*, "A user programmable reconfigurable gate array," in *Proc. CICC*, May 1986, pp. 233–235.
[4] H. Hsieh *et al.*, "A 9000-gate user-programmable gate array," in *Proc. CICC*, May 1988, pp. 15.3.1–15.3.7.
[5] H. Hsieh *et al.*, "Third-generation architecture boosts speed and density of field-programmable gate arrays," in *Proc. CICC*, May 1990, pp. 31.2.1–31.2.7.
[6] A. El Gamal *et al.*, "An architecture for electrically configurable gate arrays," *IEEE J. Solid-State Circuits*, vol. 24, no. 2, pp. 394–398, Apr. 1989.

[7] M. Ahrens *et al.*, "An FPGA family optimized for high densities and reduced routing delay," in *Proc. CICC*, May 1990, pp. 31.5.1–31.5.4.
[8] S. C. Wong, H. C. So, J. H. Ou, and J. Costello, "A 5000-gate CMOS EPLD with multiple logic and interconnect arrays," in *Proc. CICC*, May 1989, pp. 5.8.1–5.8.4.
[9] Plessey Semiconductor, Swindon, England, ERA60100 preliminary data sheet, 1989.
[10] C. Marr, "Logic array beats development time blues," *Electron. Syst. Des. Mag.*, pp. 38–42, Nov. 1989.
[11] K. Kawana *et al.*, "An efficient logic block interconnect architecture for user-programmable gate array," in *Proc. CICC*, May 1990, pp. 31.3.1–31.3.4.
[12] J. S. Rose and S. Brown "The effect of switch box flexibility on routability of field programmable gate arrays," in *Proc. CICC*, May 1990, pp. 27.5.1–27.5.4.
[13] J. Greene *et al.*, "Segmented channel routing," in *Proc. 27th Design Automation Conf.*, June 1990, pp. 567–572.
[14] K. Keutzer, "DAGON: Technology binding and local optimization by DAG matching," in *Proc. 24th Design Automation Conf.*, June 1987, pp. 341–347.
[15] R. J. Francis, J. Rose, and K. Chung, "Chortle: A technology mapping program for lookup table-based field programmable gate arrays," in *Proc. 27th Design Automation Conf.*, June 1990, pp. 613–619.
[16] J. Rose, Z. Vranesic, and W. M. Snelgrove, "ALTOR: An automatic standard cell layout program," in *Proc. Can. Conf. VLSI*, Nov. 1985, pp. 168–173.
[17] J. Rose, "LocusRoute: A parallel global router for standard cells," in *Proc. 25th Design Automation Conf.*, June 1988, pp. 189–195.
[18] S. Brown, J. S. Rose, and Z. Vranesic, "A detailed router for field programmable gate arrays" in *Proc. 1990 Int. Conf. Computer-Aided Design (ICCAD)*, Nov. 1990, pp. 382–385.



Jonathan Rose (S'79–M'86) received the B.A.Sc. degree in engineering science in 1980, and the M.A.Sc. and Ph.D. degrees in electrical engineering in 1982 and 1986, respectively, from the University of Toronto, Toronto, Ont., Canada.

During the summer of 1983 he was with Bell-Northern Research Ltd., Ottawa, Ont., Canada, in the Integrated Circuits CAD/CAM group. From 1986 to 1989 he was a Research Associate in the Computer Systems Laboratory at Stanford University, Stanford, CA. In 1989 he joined the faculty of the University of Toronto, where he is currently an Assistant Professor of Electrical Engineering. His research interests include CAD and architecture for field-programmable gate arrays, automatic layout, and parallel CAD algorithms.



Stephen Brown received the B.Sc.Eng. degree from the University of New Brunswick, Fredericton, NB, Canada, in 1985, and the M.A.Sc. degree from the University of Toronto, Toronto, Ont., Canada, in 1987, both in electrical engineering. Presently, he is a Ph.D candidate in the Department of Electrical Engineering at the University of Toronto. His research interests include computer-aided design and VLSI systems.