A Low-Power Field-Programmable Gate Array Routing Fabric

Mingjie Lin, *Member, IEEE*, and Abbas El Gamal, *Fellow, IEEE*

***Abstract*—This paper describes a new programmable routing fabric for field-programmable gate arrays (FPGAs). Our results show that an FPGA using this fabric can achieve 1.57 times lower dynamic power consumption and 1.35 times lower average net delays with only 9% reduction in logic density over a baseline island-style FPGA implemented in the same 65-nm CMOS technology. These improvements in power and delay are achieved by 1) using only short interconnect segments to reduce routed net lengths, and 2) reducing interconnect segment loading due to programming overhead relative to the baseline FPGA without compromising routability. The new routing fabric is also well-suited to monolithically stacked 3-D-IC implementation. It is shown that a 3-D-FPGA using this fabric can achieve a 3.3 times improvement in logic density, a 2.51 times improvement in delay, and a 2.93 times improvement in dynamic power consumption over the same baseline 2-D-FPGA.**

***Index Terms*—Field-programmable gate arrays (FPGAs), low-power, performance analysis, routing architecture/fabric.**

I. Introduction

POWER consumption is a primary concern of designing integrated circuits in deeply scaled CMOS technologies. This is especially the case in field-programmable gate arrays (FPGAs) because they are significantly less power efficient than cell-based ASICs [1], [2]. Recent studies have shown that FPGAs are between 9 and 12 times less power efficient than cell-based ASICs [3]–[7]. Today, this power inefficiency problem continues to preclude the use of FPGAs in low power applications, and if not appropriately addressed could undermine future improvements in their logic capacity and delay.

The power consumed in an FPGA core[1] consists of both static and dynamic components. Although static power has been increasing with technology scaling, it constitutes only 10% of the total power consumed in an FPGA today [1], [2]. Furthermore, known techniques for reducing it such as programmable supply voltage, multiple gate oxide thicknesses, multiple transistor threshold voltages, and power gating, have already been applied to FPGAs [1], [2] [8]. Dynamic power, on the other hand, constitutes over 90% of the power consumed in FPGAs and is the main source for their power inefficiency relative to cell-based ASICs. This power inefficiency is caused by the high programming overhead of FPGAs, including the pass-transistor switches, MUXes, buffers, and configuration memory used in the programmable routing fabric, which occupies 50%–80% of the silicon area [3]. This results in much longer routed net lengths relative to cell-based implementations, and hence higher capacitive loading. An even more significant factor than the increase in net length is the loading presented by the programming overhead itself (see Sections II–II-C). As a result of these two factors, 60%–80% of the total FPGA core dynamic power is consumed in the programmable routing fabric [9], [10].

From this discussion, it is clear that reducing interconnect power consumption in an island-style FPGA must involve reducing routed net lengths and/or programming overhead. Routed net lengths can be reduced by using only short interconnect segments in the routing channels. Unfortunately, using only short segments in island-style FPGA only marginally improves power consumption and results in longer delays (see Sections II– II-B). To reduce programming overhead, one way is to decrease the switch box and/or connection box flexibilities. This, however, can decrease routability, increase routed net lengths, and ultimately result in longer delay and higher dynamic power. The main contribution of this paper is showing that significant interconnect power saving can be achieved by re-architecting the programmable routing fabric such that both routed net lengths and programming overhead are reduced without adversely affecting routability or delay.

A. Results Summary and Outline

We present a new FPGA programmable routing fabric that achieves significantly lower power consumption and delay than island-style fabric by using only short interconnect segments and reducing the interconnect loading due to the programming overhead.[2] We show that an FPGA using this fabric, referred to henceforth as *new 2-D-FPGA*, can achieve on average a 1.57 times lower dynamic power consumption and 1.35 times lower geometric average pin-to-pin to delay with only 9% increase in silicon area relative to a baseline island-style FPGA, referred to henceforth as *baseline 2-D-FPGA* (see Sections II–II-A and [12]).

The top level architecture of an FPGA using the new routing fabric is depicted in Fig. 1. It consists of an array of merged *Routing and Logic blocks* with horizontal and vertical routing channel overlay. Only short interconnect segments are used in the routing channels to reduce routed net lengths. The routing block integrates the functions of the connection and switch

Manuscript received June 09, 2007; revised January 15, 2008 and July 20, 2008. First published March 16, 2009; current version published September 23, 2009. This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) and in part by the Space and Naval Warfare Systems Command (SPAWAR) System Center under Grant N66001-04-1-8916.

The authors are with the Department of Electrical Engineering, Stanford University, Stanford, CA 94305-9505 USA.

Digital Object Identifier 10.1109/TVLSI.2008.2005098

[1]In this paper, we do not consider the power consumed in FPGA I/Os.

[2]A very preliminary version of this paper was partially presented in [11]. This paper provides a more complete discussion of the motivation for developing this fabric and performance comparison for both 2-D and 3-D-FPGA.

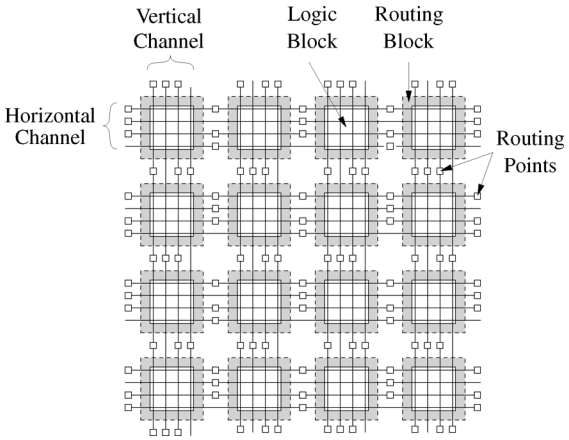

Fig. 1. New routing fabric: Array of logic-routing blocks with horizontal and vertical channel overlay.

boxes in a conventional FPGA. It provides connectivity for logic block inputs and outputs and performs signal bends and fan-outs. The *routing points* are used to: 1) form local connections between neighboring logic blocks (LBs) without going to channels; 2) connect routing block inputs and outputs to channel segments; and 3) chain channel segments together to form longer segments without entering routing blocks. The loading on the interconnect segments is reduced by:

1) using significantly smaller number of pass-transistor switches to connect routing block inputs to logic block inputs than the number used in the connection box of the baseline FPGA;
2) re-architecting the way the logic block outputs connect to the fabric to reduce the loading on logic block output segments and provide direct connectivity between neighboring logic block;
3) eliminating the pass-transistor switches from the interconnect segments in the channels.

As we shall see in Section V, these reductions in programming overhead are achieved without compromising routability.

The architecture of the new routing fabric is also motivated by research on monolithically stacked 3-D-IC, whereby active devices are lithographically built in between metal layers [13], [14]. In addition to achieving higher logic density, vertical stacking reduces interconnect length, hence reducing interconnect delay and power consumption. Considering a 3-D-FPGA where the programming overhead is stacked on top of the LBs (see Fig. 2), a recent study [12] quantified the performance benefits of a monolithically stacked 3-D-FPGA. It was shown that such a 3-D-FPGA can achieve 3.2 times higher logic density, 1.7 times lower critical path delay, and 1.7 times lower dynamic power consumption than a baseline 2-D-FPGA fabricated in the same 65-nm technology node. These improvements were achieved with *no* change in the logic or routing architecture of the baseline 2-D-FPGA. In this paper, we show that a 3-D-FPGA using our new routing fabric, referred to henceforth as *new 3-D-FPGA*, can achieve average improvement of 3.3 times in logic density, 2.38 times in critical path delay, 2.51 times in geometric average pin-to-pin delay, and 2.93 times in dynamic power consumption over the baseline 2-D-FPGA.

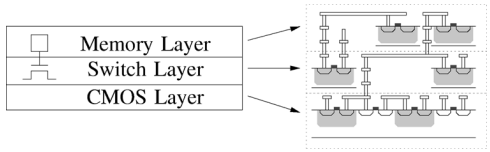

Fig. 2. Monolithically stacked 3-D-FPGA concept.

The rest of the paper is organized as follows. In the following section, we describe the 2-D-FPGA architecture used as a baseline for performance comparison and discuss several factors that motivated the development of the new routing fabric. In Section III, we describe the new routing fabric in details. In Section IV, we describe the CAD flow used for mapping designs into the new FPGA. In Section V, we quantify the performance improvements of the new 2-D-FPGA over the baseline 2-D-FPGA. Finally, in Section VI, we quantify the performance improvements of a monolithically stacked implementation of the new FPGA over the baseline 2-D-FPGA.

II. Preliminaries

We first describe the baseline 2-D-FPGA architecture that we use in explaining the motivation for the new fabric design and in comparing their performance. We then discuss the factors that motivated the development of the new programmable routing fabric.

A. Baseline 2-D-FPGA

We assume an island-style FPGA comprising a 2-D array of LBs interconnected via a programmable routing fabric [15] (see Fig. 3). To ensure the fairness of our performance comparison studies, the architectural parameters of baseline 2-D-FPGA are chosen according to both previous studies [16], [17] and our own investigations using TORCH design tool [18].

- *Logic Block*: We assume a Virtex II style logic block comprising four slices, each consisting of two four-input Lookup Tables (LUTs), two flip-flops (FFs), and programming overhead. Note that the design of our logic block is a simplified version of the Virtex II logic block detailed in [15]. Our choice of cluster and LUT sizes follows the results of [17], where under a fixed routing architecture (using wire segment length 4 and 50% pass transistors and 50% tri-state buffers in routing switches), an island-style FPGA with the cluster size 8 and LUT size 4 consumes the lowest energy. We also assume throughout the paper that the new 2-D and 3-D FPGAs use the same logic block as the baseline 2-D-FPGA.
- *Routing Channel Width*: The routing fabric comprises horizontal and vertical segmented routing channels. The number of routing tracks is determined empirically. In the routability comparison in Sections V–V-A, we use the minimum number of routing tracks needed to successfully route the benchmark circuits in both the new FPGA and the baseline. In the delay and power comparison in Sections V-V-E and V-V-F, we choose the number of routing tracks to be 15% higher than the minimum required to avoid excessive routing congestion [19]–[22].

TABLE I
BASELINE 2-D-FPGA PARAMETER VALUES

Tile Width L	Array Size $N \times N$	LB Buffer Size, b	LB Inputs K_i	LB Outputs K_o	SB Width / Density W, d	CB Flexibility F_c
4100λ	64 or 100	4	32	8	72, 3	0.5

Segment Type	Single	Double	Length-3	Length-6
Number of Tracks	32	36	30	72

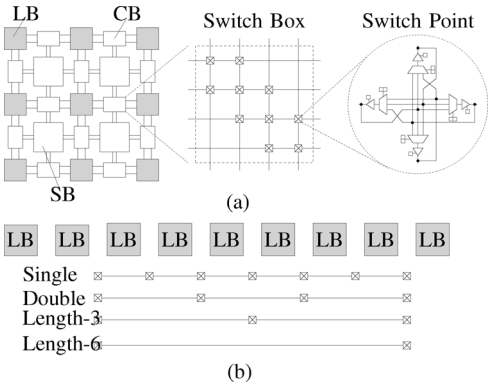

Fig. 3. (a) Baseline 2-D-FPGA architecture. (b) Segmented track types used.

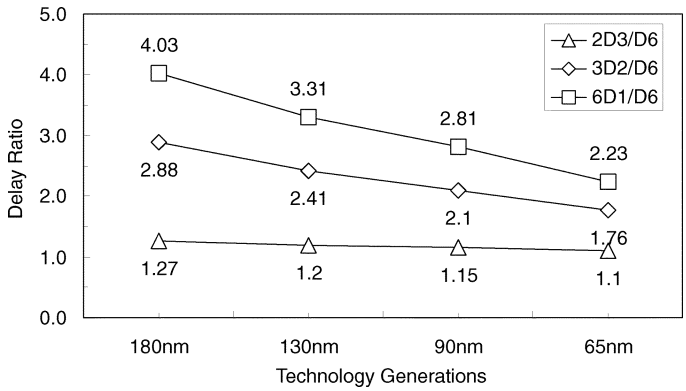

Fig. 4. Delay benefits of long interconnect relative to short interconnects of equivalent total length for four technology nodes. Normalized net delays, $6D_1/D_6$, $3 - D_2/D_6$, and $2 - D_3/D_6$, for four technology nodes, where D_i is the delay of a segment of length i tiles, are plotted. The sizes of the buffers for interconnect segments Length-3 and Length-6 are optimized for each technology node as discussed in [12].

- *Routing Channel Segmentation*: The channel segmentation for the baseline FPGA comprises sets of staggered single, double, length-3, and length-6 segmented tracks. Each segment consists of two unidirectional metal wires. The segments can be connected via connection boxes (CBs) only to the inputs and outputs of the first and last LBs it spans. Segments can be connected to each other via switch boxes (SBs). To ensure the quality of routing channel segmentation for the baseline FPGA, we start with a segmentation based on [15] and [23], and use the TORCH routing channel segmentation design tool [18] to optimize it for 65-nm CMOS technology node. The optimization resulted in a 9% improvement in delay-power product over the Virtex II based segmentation.
- *Design of CB and SB*: The design of the CB and SB follow [24] and the design of the switch point is MUX-based as described in [25] [see Fig. 3(a)]. The logic design of a switch point is much more complicated than a pass-transistor switch, which consists of a single transistor controlled by one configuration memory bit. Each LB is assumed to have K_i input pins and K_o output pins that can be connected to routing channel.

Table I lists the key architecture parameter values for the baseline FPGA used in the performance comparisons.

We now discuss various factors that motivated the development of our new routing fabric.

B. Using Only Short Segments

FPGA routing fabrics, such as the one used in the baseline FPGA described above, employ a mixture of short and long interconnect segments in order to reduce net delay. Short segments are beneficial to routability but can result in higher net delay. Using long interconnects, however, results in longer routed net lengths and larger silicon area and hence higher power consumption.

A previous study [12] has argued that the delay advantage of long interconnect segments (Lengths 3 and 6) diminishes as CMOS technology scales below 100 nm due to the much larger increase in wire parasitics relative to transistor parasitics. This observation is demonstrated in Fig. 4, which compares the delay of implementing a net spanning six tiles using 2 Length 3 segments ($2 - D_3$), 3 Double segments ($3 - D_2$), or 6 Single segments ($6D_1$) normalized by the delay of implementing the net in a single Length 6 segment (D_6). Thus, the larger the normalized delay, the greater the benefit of using longer segments. From the figure we see, for example, that the relative delay $6D_1/D_6$ decreases from 4.03 in 180-nm technology to 2.23 in 65-nm technology. As long segments are expensive in area due to large buffers and provide less routing flexibility than shorter segments, their cost effectiveness decreases with technology scaling.

This finding has motivated us to consider routing fabrics with only short segments. In the following routability study, we quantify the improvement in net length of routed circuits using only short segments, in which we consider a baseline 2-D-FPGA with only single and double interconnects, henceforth referred to as *2-D-FPGA(1,2)*. The important point here is that the average segment length is considerably shorter than in the baseline 2-D-FPGA.

In order to make the comparison fair, we equalize the routability of the 2-D-FPGA(1,2) to that of the baseline 2-D-FPGA using the methodology outlined in Sections V–V-B, which results in each channel having 24 single and 96 double

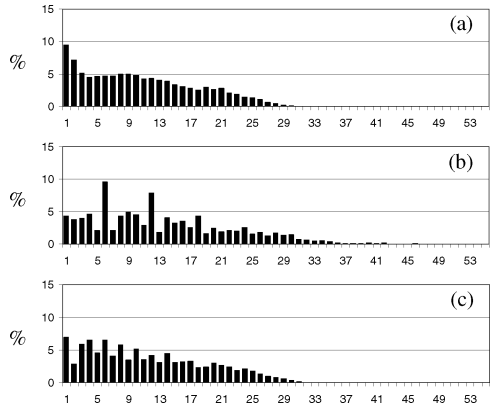

Fig. 5. Routed net length distributions for ALU4 benchmark circuit. (a) Manhattan distance. (b) baseline 2-D-FPGA. (c) 2-D-FPGA(1,2).

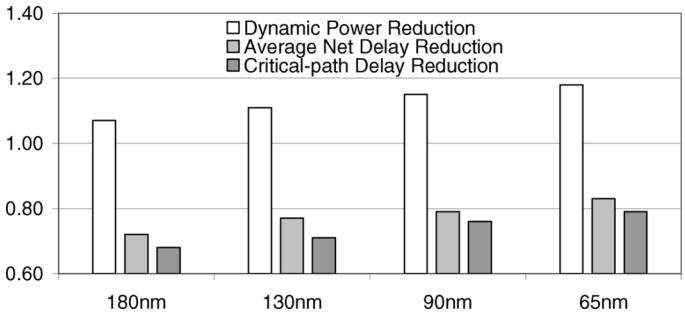

Fig. 6. Average routing fabric power consumption and delay for 2-D-FPGA(1,2) relative to baseline 2-D-FPGA for four technology nodes.

segment tracks. We placed and routed the 20 largest MCNC benchmark circuits [26] in both the baseline 2-D-FPGA and the 2-D-FPGA(1,2). Fig. 5 compares their pin-to-pin net length distributions to the Manhattan distance for the ALU4 benchmark circuit [26]. These results are obtained using timing-driven placement for 2-D-FPGA and separate routing runs for the 2-D-FPGA and the 2-D-FPGA(1,2). Note that the net length distribution in the baseline 2-D-FPGA(1,2) is much closer to that of the Manhattan distance. On average, the net length in the 2-D-FPGA(1,2) is around 16% shorter than that in the baseline 2-D-FPGA. This reduction in net length results in a reduction in dynamic power consumption as illustrated in Fig. 6. Note that the power consumption improves for all technology nodes, and that the improvement factor increases with technology scaling because of the increase in wire parasitics. The improvement in dynamic power, however, comes at the expense of an increase in delay as shown in Fig. 6. As we shall see, the new routing fabric achieves significant improvements in both power and delay relative to the baseline 2-D-FPGA(1,2).

C. Reducing Loading due to Programming Overhead

Interconnect loading in FPGAs is dominated by loading due to the programming overhead. To illustrate this, Fig. 7 plots the fraction of capacitive loading due to the switch point (SP), connection box pass-transistor switches (PTS), and metal wire (MW) in Single and Double segments in the baseline 2-D-FPGA for four technology nodes. Note that even though the fraction of loading due to metal wire has been increasing with technology scaling, the loading due to the pass-transistor switches and switch point still dominates. In fact, the fraction due to the pass-transistor switches has increased with technology scaling. As such, the design of the new routing fabric focuses on reducing the loading due to the programming overhead as quantified in Sections V–V-D.

D. Improving Routability

While using only short segments should improve routability, reducing programming overhead could have an adverse effect on routability, which could in turn hurt power and delay. The new routing fabric reduces programming overhead without reducing routability. This is achieved through judicious design of the routing point and the routing block, including the extended switching capability of the routing block, as detailed in the Section III.

E. Compatibility With 3-D Implementation

Our routing fabric is also motivated by monolithic stacking. Because the programmable routing (switches and configuration memory) is stacked on top of the logic blocks and buffers and the LB inputs and outputs "come up" to the routing fabric, it is natural to integrate the functionalities of the interconnect and switch boxes associated with each LB into a single routing block. This allows for sharing of resources and further reduction in interconnect loading.

III. New Routing Fabric

As discussed in the introduction (see Fig. 1), the new fabric consists of an array of routing blocks (RBs) with horizontal and vertical routing channel overlay.

A. Routing Block

The routing block integrates the functions of both the connection and switch boxes of the island-style FPGA. Fig. 8 shows how the routing block implements the function of a switch box. As in a conventional switch box, the RB function is parameterized by the width W, which is the number of input ports on each side, and the switching width d, which is the number of possible connections for each port [24]. A routing block input line entering at one side connects to inputs of d different MUXes on each of the two perpendicular sides. For example in Fig. 8(b), the input line $i_{1,t}$ enters the left side of the routing block and connects to the inputs of MUXes with outputs $O_{1,t}$, $O_{2,t}$, $O_{3,t}$ on the top side and three MUXes with outputs $O_{1,b}$, $O_{2,b}$, and $O_{3,b}$ on the bottom side. Note that unlike conventional switch box design, our routing block does not include straight connections. Such connections are provided by the interconnect segments in the routing channel overlay as shown in Fig. 12. Each routing block MUX output drives an interconnect segment and/or an input line to a neighboring routing block through a routing point.

Figs. 9 and 10 show how the routing block implements the function of a connection box. Each routing block input can connect to n_i LB inputs via pass-transistor switches as shown in Fig. 9. The value of n_i is chosen so that each LB input can connect to the same number of routing block inputs as the number of interconnect segments an LB can connect to in the baseline

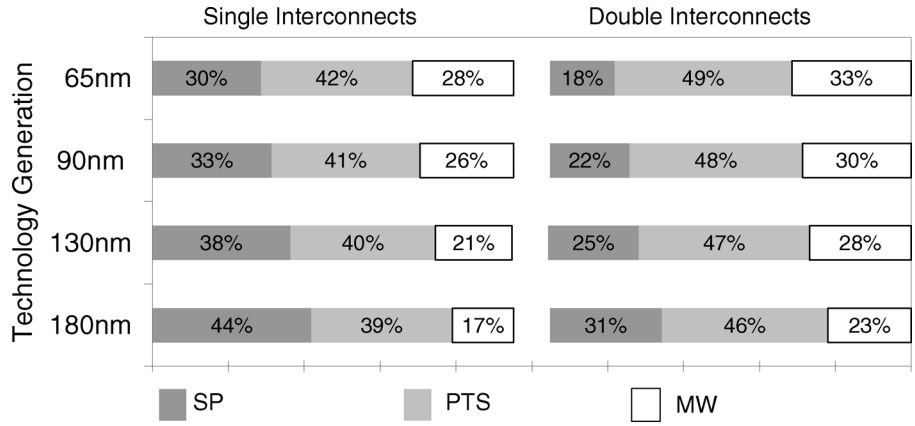

Fig. 7. Delay breakdown of Single and Double interconnect delays between connection box pass-transistor switches (PTS), switch points (SP), and metal wire (MW) for baseline 2-D-FPGA.

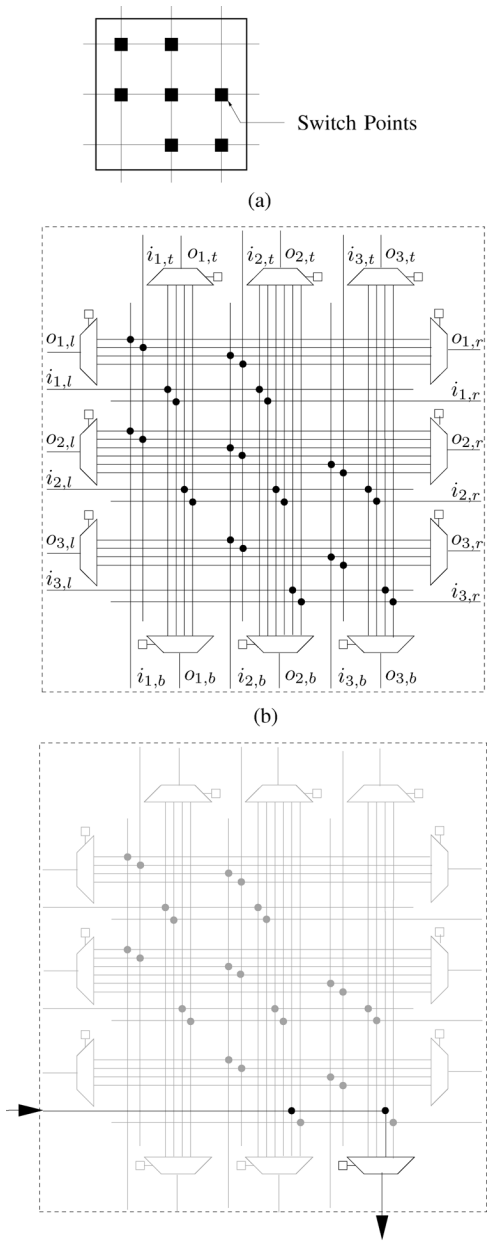

Fig. 8. (a) Switching capability of a routing block. (b) Routing block with $W = 3$ and $d = 3$ (connections to LB inputs and outputs not shown). (c) Example signal turn.

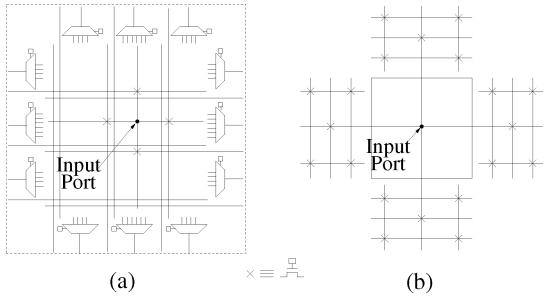

Fig. 9. (a) LB input can connect to several routing block inputs. Each routing block input can connect to n_i LB inputs. (b) An LB input can connect to several segments in the baseline fabric. Note the similarity between the two connectivities. The much-simplified example of $W = 3$, $n_i = 1$, $K_i = 2$, and $K_o = 1$ is shown.

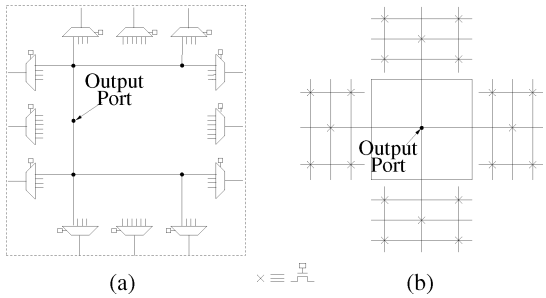

Fig. 10. (a) LB output can connect to n_o routing block MUX inputs on each side. (b) An LB output can connect to segments in the baseline fabric. Note the large difference between the two connectivities. The much-simplified example of $W = 3$, $n_o = 2$, $K_i = 2$, and $K_o = 1$ is shown.

fabric. However, and *bypass*. the loading on a routing block input segment is significantly lower than that on an interconnect segment in the baseline fabric. Thus, we maintain the same connection box flexibility for the LB inputs in our fabric, but with significantly lower segment loading. In Sections V–V-D, we provide a detailed comparison between the loading on a routing block input and on a Single segment in the baseline fabric.

In addition to the connection through switch points, the architecture of the routing block allows for *extended* switching width. As shown in Fig. 11, a signal entering a routing block can loop back twice into it and exit to a perpendicular direction

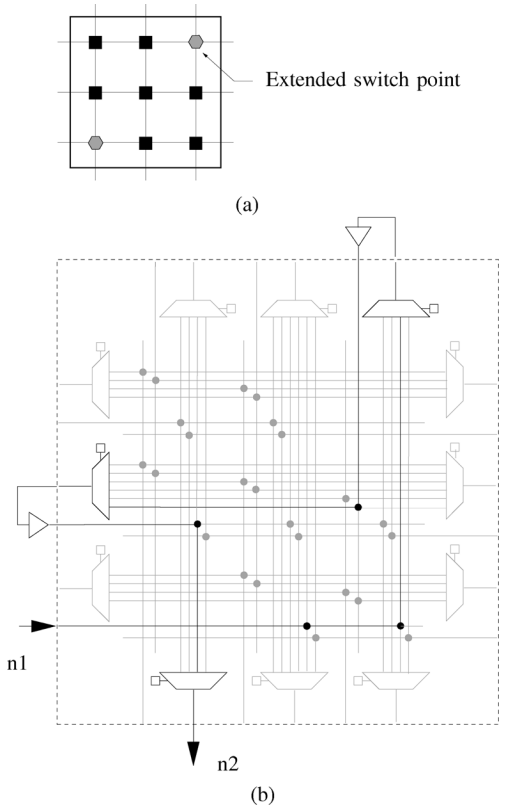

Fig. 11. (a) Extended routing capability of routing block. (b) Example of an extended signal connection.

if it cannot do so directly. As we demonstrate later, such extended switching significantly improves routability.

B. Routing Channel Overlay

Each routing channel comprises only Single and Double segmented tracks. Each segment consists of two unidirectional wires. The inputs and outputs of the routing blocks and the channel segments can be programmably connected using the routing points as depicted in Fig. 12. Segments can be chained together to form longer segments, henceforth referred to as *bypass interconnect*, by appropriately setting the states of the routing points without entering routing blocks. The segments can also be connected via routing points to routing blocks to connect to LB inputs and outputs, make bends, or fan-out.

C. Connections Between Routing Block and Routing Channel Overlay

Fig. 10 shows how an LB output is connected to the fabric. Each LB output is connected to n_o different MUX inputs on each side of the routing block. This is quite different from the way LB outputs are connected in the baseline fabric and is a key factor in achieving lower interconnect loading. Note that n_o is chosen to be significantly lower than the number of segments an LB output can connect to in the baseline fabric. This results in a significant reduction in loading on the LB output segment as quantified in Sections V–V-D. Interestingly, this reduction in LB output connectivity does not adversely affect routability (see Sections V–V-A for a more detailed discussion of this key point).

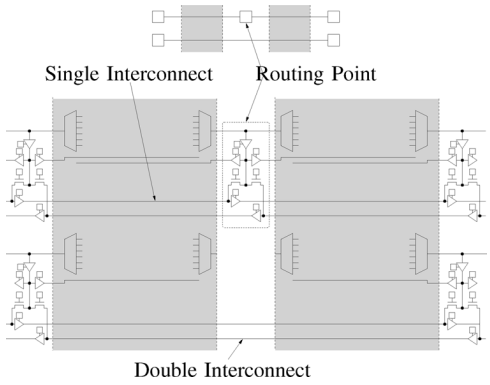

Fig. 12. Single and double interconnects and their connections through routing points to the routing block. Top: symbolic representation from Fig. 1. Bottom: detailed schematic with only horizontal channels shown.

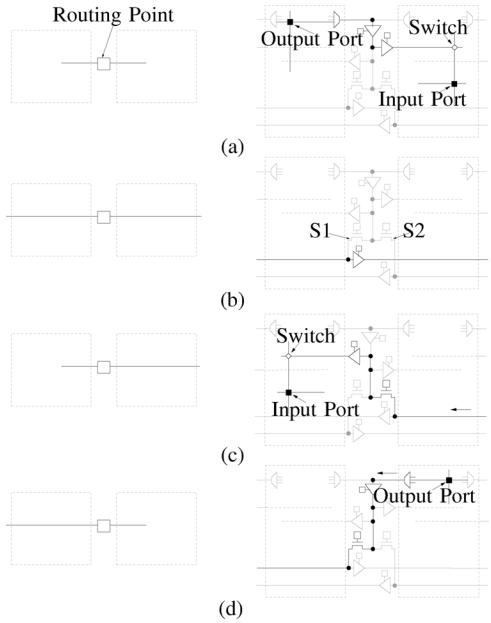

Fig. 13. (a) Connection from LB output to neighboring LB input. (b) Bypass interconnect. (c) Connection from a segment to an LB input. (d) Connection from an LB output to a segment.

The routing fabric allows for two types of net connections: *local* A local connection is used to directly route one output of an LB to an input of its neighboring LB without using routing channel segments. A bypass connection is used to route a longer net without entering intermediate routing blocks. Fig. 13(a) shows how an output of an LB can be directly connected to the input of a neighboring LB without using channel segments. Fig. 13(b) shows how a bypass interconnect is implemented. Note that by turning off the two pass transistor switches S1 and S2, a local connection can be simultaneously made between the two routing blocks. This routing point resource sharing improves routability. Turning off these pass transistors also reduces the loading on the bypass interconnect. Fig. 13(c) and (d), respectively, shows how an LB input and output can be connected to a segment. Note that only buffers that are needed to establish the connection are turned on. This again helps reduce the loading on the connection, thus reducing its delay and power consumption (see Sections V–V-D).

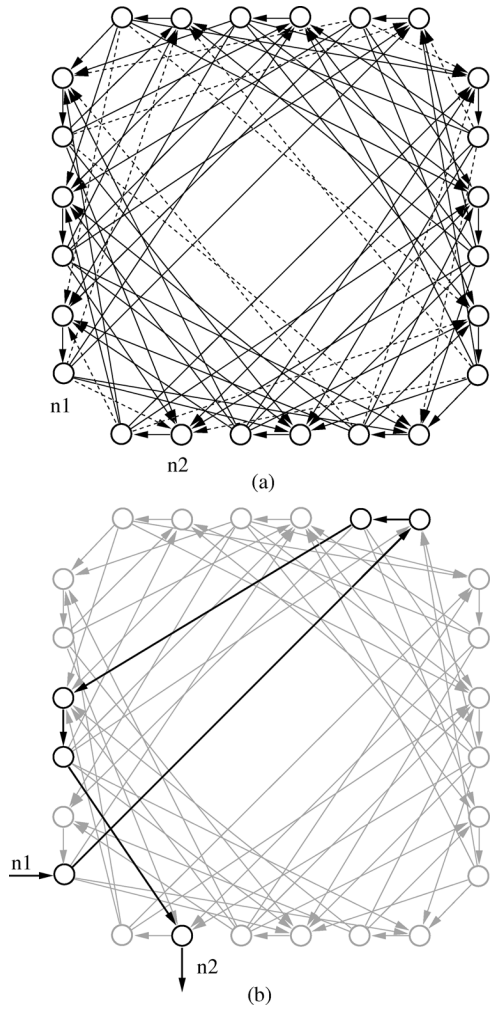

Fig. 14. (a) Routing graph for routing block with $W = 3$ and $d = 3$. (b) Extended connection between n_1 and n_2.

IV. CAD Tools

To map designs into the new FPGA, we use the logic packing and placement modules of VPR [27]. We modify the VPR router [27], [28] to accommodate the differences between the routing architecture of our new fabric and the island-style fabric.

Fig. 14(a) shows an example routing graph for a routing block with width $W = 3$ and switching with $d = 3$. Note that d is the number of possible connections from an input port to each of the routing block sides it connects to, which plays a similar role to the parameter F_s in the baseline FPGA [24]. Each routing block input and output is represented by a node (so there are $2W$ nodes on each side). Solid edges correspond to direct connections while dashed edges correspond to extended connections. Fig. 14(b) shows how an extended connection from input node n_1 to output node n_2 is implemented using direct connections. The example corresponds to Fig. 11.

The routing algorithm used is based on the Pathfinder negotiated congestion algorithm in [28]. It is described in Algorithm 1, where:

- A_{ij} is the criticality of the connection from the source of net i to one of its sinks j;
- d_n is the intrinsic delay of routing node n;
- p_n is the present congestion cost of node n.

Initially, nets are routed one at a time using the shortest path it can find without considering interconnect segment or logic block pin overuse. Each iteration of the router consists of sequential net rip-up and reroute according to the lowest cost path available. The cost of using a routing resource is a function of its current overuse and any overuse that occurred in prior routing iterations. By gradually increasing the cost of an oversubscribed routing resource, the algorithm forces nets with alternative routes to avoid using that resource, leaving it to the net that most needs it.

The main difference between this algorithm and VPR is that we keep track of visited nodes during the breadth-first-search to improve the run time. This is described in lines 11, 12, and 22. This modification is needed because the extended switching capability of the routing block explained earlier results in many local cycles in the routing graph.

Algorithm 1 Congestion/Delay Routing Algorithm

1: $A_{ij} \leftarrow 1$ for each signal net i and each sink j
2: **while** shared routing nodes exist **do**
3: **for all** nets i **do**
4: rip up routing tree RT_i
5: initialize the queue PQ
6: **for all** sinks t_{ij} **do**
7: enqueue each node n in RT_i at costs $A_{ij}d_n$ to PQ
8: **while** t_{ij} is not found **do**
9: dequeue node m with the lowest cost from PQ
10: **for all** fanout node n of m **do**
11: **if** node n is unseen **then**
12: mark node n as seen
13: enqueue n to PQ with the cost of $A_{ij}d_n + (1 - A_{ij})d_n p_n$
14: **end if**
15: **end for**
16: **for all** node n in the routed path t_{ij} to s_j **do**
17: update the cost of node n
18: add n to RT_i
19: **end for**
20: **end while**
21: **end for**
22: mark all nodes in PQ as unseen
23: update A_{ij} for net i
24: **end for**
25: **end while**

V. New 2-D-FPGA Versus Baseline 2-D-FPGA

In this section, we compare a 2-D-FPGA using the new routing fabric to the baseline 2-D-FPGA in terms of routability, programming overhead, logic density, delay, and power consumption. We assume a 65-nm CMOS technology and the Berkeley Predictive Technology Model (BPTM) for devices and interconnect.

A. Routability

To compare the routability of the new fabric to that of the baseline 2-D-FPGA, we placed and routed the 20 largest MCNC benchmark circuits in both architectures. We varied the routing channel width and found the minimum track count $T_{\min}$ for each design mapped to each architecture. Note that, in general, T is greater or equal to the routing block width W. For example, in our baseline FPGA routing channel, we have 32 Single, 36 Double, 30 Length-3, and 72 Length-6 tracks, which corresponds to $T = 170$ and $W = 32 + 36/2 + 30/3 + 72/6 = 72$. In this section, we choose $T_{\min}$ to compare routability between baseline the new and baseline FPGAs. For the new 2-D-FPGA, we assumed that each routing channel contains 28 Single, 64 Double, and 36 Length-3 routing tracks. In varying the channel width we maintained the same fractions of each interconnect segment type. For the baseline, we use a fraction of 0.19 for Single, 0.21 for Double, 0.18 for Length-3, and 0.42 for Length-6 segment tracks. For the new 2-D-FPGA, we maintained a ratio of 0.22, 0.5, and 0.28 for the number of Single, Double, and Length-3 segment tracks, respectively. Table II compares 1) the minimum channel width $T_{\min}$ in segmented tracks for the baseline 2-D-FPGA and the new 2-D-FPGA, 2) the geometric average of pin-to-pin net lengths $\overline{L}$ where the pin-to-pin net length is the sum of segment lengths used in its routing, and 3) the geometric average of the number of bends $\overline{S}$ used to route each pin-to-pin net segment. Note that on average, the new routing fabric requires 50% fewer tracks per channel than the baseline 2-D-FPGA. This is because of the use of only short segments and the additional switching capability of the routing block. These $T_{\min}$ values correspond to a reduction in average routing block width of around 20% over the switch box width in the baseline 2-D-FPGA. The new routing fabric also reduces the average pin-to-pin net segment length by around 16%. Finally, note that the average number of bends $\overline{S}$ increases slightly in the new 2-D-FPGA due to the use of only short segments.

To investigate the improvement in routability due to the extended switching capability of the routing block, we disabled this feature and rerouted the benchmark circuits. We found that the saving in minimum channel width drops from 57% to 34%.

TABLE II
ROUTABILITY COMPARISON BETWEEN THE NEW 2-D-FPGA (NEW) AND THE BASELINE 2-D-FPGA (BL). $T_{\min}$ IS THE MINIMUM NUMBER OF SEGMENTED TRACKS NEEDED, $\overline{L}$ IS THE GEOMETRIC AVERAGE OF PIN-TO-PIN NET LENGTHS MEASURED IN THE NUMBER OF LOGIC BLOCKS THE NET SPANS, AND $\overline{S}$ IS THE GEOMETRIC AVERAGE OF THE NUMBER OF BENDS

Circuit	$T_{\min}$		$\overline{L}$		$\overline{S}$	
	BL	NEW	BL	NEW	BL	NEW
alu4	56	24	12.79	11.54	3.61	3.88
apex2	58	31	12.23	10.43	3.23	3.28
apex4	53	36	11.57	9.19	2.79	2.69
bigkey	37	21	19.98	18.22	4.41	5.55
clma	76	42	21.11	17.51	4.37	5.23
des	40	14	18.35	15.63	3.42	3.94
diffeq	39	27	9.13	7.81	2.99	3.01
dsip	32	14	20.77	18.23	4.59	5.11
elliptic	76	36	17.02	13.77	3.34	4.42
ex1010	83	47	13.95	14.56	3.71	4.03
ex5p	75	38	10.19	9.51	2.59	3.07
frisc	83	46	14.91	12.09	3.53	3.87
misex3	65	33	12.01	9.12	3.21	3.32
pdc	112	49	18.98	16.83	3.72	4.17
s298	43	26	14.03	11.03	3.33	3.87
s38417	75	33	10.09	8.23	2.27	3.02
s38584	59	33	12.31	9.79	2.79	3.13
seq	73	35	12.21	10.51	3.21	3.69
spla	94	52	17.72	14.43	3.89	3.81
tseng	43	27	10.78	9.51	3.13	3.41

B. Programming Overhead

In this section, we compare the programming overhead of the new 2-D-FPGA to that of the baseline 2-D-FPGA. For the baseline 2-D-FPGA, we assume the architecture parameter values given in Table I. For the new 2-D-FPGA, we assumed that each routing channel contains 28 Single, 64 Double, and 36 Length-3 routing tracks (so $W = 28 + 64/2 + 36/3 = 72$). This choice of routing channel segmentation was obtained using TORCH [18]. Each routing block input line connects to inputs of $d = 3$ MUXes in each of the two perpendicular directions and can be programmably connected to $n_i = 8$ LB inputs through pass transistor switches (see Fig. 9). Equivalently, each LB input can connect to $8 \times 72 \times 4/16/4 = 36$ input lines from each side of the routing block, which is the same as the number of segments an LB can connect to in the baseline 2-D-FPGA. Each LB output can connect to two MUX inputs ($n_o = 2$) on each of the four sides of the routing block (see Fig. 10). As Table II shows, the $T_{\min}$ values for the new 2-D-FPGA correspond to a reduction in average routing block width of around 20% over the switch box width in the baseline 2-D-FPGA. Therefore, $W = 72$ for the new 2-D-FPGA achieves equal or better routability than the assumed $W = 72$ for the baseline 2-D-FPGA.

Table III list the programming overhead per tile for the new 2-D-FPGA versus the baseline 2-D-FPGA. Note that the programming overhead of the routing block, which implements the functions of two connection boxes and a switch box, is lower than that of the switch box by itself. This is because, in the new 2-D-FPGA, 1) the number of pass-transistor switches on an LB input is much lower than that on an interconnect segment in the baseline 2-D-FPGA, 2) there are no pass-transistor switches on LB output segments, and 3) there are no pass-transistor switches on the interconnect segments in the channels.

An important point here is that the number of interconnect segments an LB output can connect to in the new 2-D-FPGA is reduced to 8 from 36 (16 Single, 9 Double, 5 Length-3 and 6 Length-6) in the baseline 2-D-FPGA. However, the number of RB inputs that an LB input can connect to is the same for the new and baseline 2-D-FPGA (14 Single, 16 Double, and 6 Length-3 in the new 2-D-FPGA and 16 Single, 9 Double, 5 Length-3, and 6 Length-6 in the baseline 2-D-FPGA). The fact that an LB input can connect to many more short segments in the new 2-D-FPGA and the extended switching capability of the routing block appear to be more than enough to compensate for the reduction in the LB output connectivity. To demonstrate that a similar reduction in LB output connectivity would hurt routability in the baseline 2-D-FPGA, we decreased the LB output connectivity to the connection box in the baseline 2-D-FPGA from 36 to 8. We placed and routed the MCNC benchmark circuits and found

TABLE III
COMPARISON OF PROGRAMMING OVERHEAD PER TILE FOR BASELINE 2-D-FPGA AND THE NEW 2-D-FPGA. S: SWITCHES. T: TRI-STATE BUFFERS. I: INVERTERS. M: MEMORY BITS

Baseline 2D-FPGA	
Logic Block	M: 1049
Interconnects	T: 96 M: 96
2 Connection Boxes	S: 1440 M: 1440
1 Switch Box	S: 3456 T: 864 I: 1728 M: 2592
Total	S: 4896 T: 960 I: 1728 M: 5177

New 2D-FPGA	
Logic Block	M: 1049
Interconnects	T: 288 M: 288
Routing Block	S: 2880 T: 432 M: 3312
Total	S: 2880 T: 720 M: 4649

that this reduction in LB output connectivity results in a 7% increase in the minimum channel width required to successfully route the designs. Furthermore, the average power consumption increased by 9% and delay increased by 13% as a result of this decrease in LB output connectivity.

C. Logic Density

We use the Cadence GSCLib3.0 technology-independent library and Virtuoso to estimate the layout area for the logic block and buffers in the same manner as in [12]. The routing block area is estimated using custom layout. For the baseline 2-D-FPGA, we assume the architecture parameter values in Section II and the interconnect buffer sizes used in [12]. The 2-D-FPGA(1,2) has the same architecture as the baseline except that it has 24 Single and 48 Double segment tracks in each routing channel. For the new 2-D-FPGA, we assume the architecture parameter values in the previous subsection and buffer size 4 for Single interconnects, 6 for Double interconnects, 8 for buffers driving the routing block input, and 6 for shared MUX output buffer. The MUXes and the pass-transistor switches that connect segments to routing blocks use size 4 transistors.

The estimated size of a single tile in the new 2-D-FPGA is $4280\lambda \times 4280\lambda$ compared to $3846\lambda \times 3846\lambda$ for the 2-D-FPGA(1.2) and $4100\lambda \times 4100\lambda$ for the baseline 2-D-FPGA [12]. The slight increase of 9% in the tile area of the new 2-D-FPGA over the baseline 2-D-FPGA is due to the use of larger pass-transistor switch sizes to improve interconnect delay.

D. Interconnect Loading

To quantify the reduction in interconnect loading in the new fabric relative to that in the baseline 2-D-FPGA(1,2), we consider three scenarios, a local connection between two neighboring LBs (see Fig. 15), a connection spanning 3 LBs (see Fig. 16), and a connection spanning 6 LBs with one signal bend (see Fig. 17). Table IV compares the capacitive loading for the three scenarios. To help understand where the reduction in loading comes from, the loading in each case is divided into several components as shown in the figures. From the table, we see that the total loading in the new fabric is 55% lower than that in the baseline fabric for the local connection, 60% lower for the length 3 connection, and 39% for the length 6 connection. For the local connection (Fig. 15), the loading savings from the output segment and the interconnect segment are about the same. For the other second scenario (Fig. 16), the largest source of reduction in the loading due to the interconnect segment (component 3). While the Single segment in the baseline fabric with CB connectivity $F_c = 0.5$ has 20 $(= (16 + 4) \times 0.5 \times 2)$ pass transistors connected to it from the input and output segments of the two neighboring LBs, the routing block input has only eight pass transistors in addition to six MUX inputs connected to it. The second largest source of loading reduction is the output segment (component 1). The output segment in the baseline with $W = 72$ has 36 pass transistors connected to, while in the new fabric it has only eight MUX inputs connected to it. In addition, there is a small reduction in the loading due to the lower loading of the routing point in the new fabric relative to the switch point in the baseline fabric. For the third scenario (Fig. 17), the largest source of reduction in the loading is the interconnect segment (component 3). However, as shown in Table IV, the parasitic loading due to the signal bend (component 5) in the new routing fabric is significantly larger than that in the baseline fabric. This is because in the new fabric a bend results in high wire loading (see Fig. 8) relative to the baseline. As a result of the high signal bend cost, the loading reduction in the third scenario 3 is only 39%.

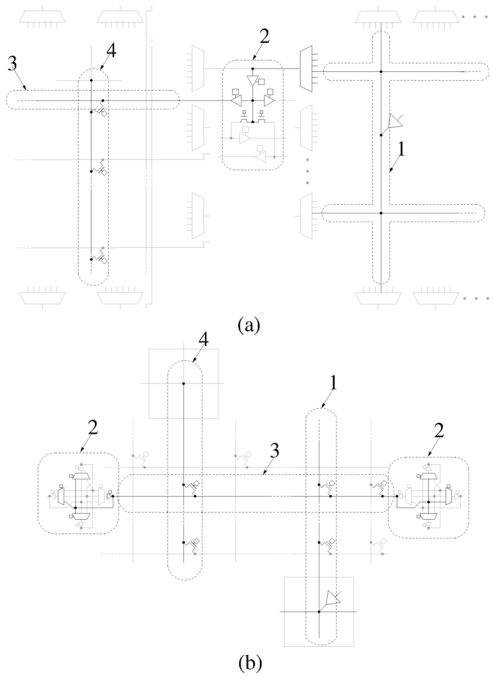

Fig. 15. Local connection in new fabric (a) and in baseline fabric (b). The numbers correspond to (1) output segment, (2) routing point/ switching point, (3) interconnect segment, and (4) input segment.

E. System Delay

To compare the system delay of the new 2-D-FPGA to that of the baseline 2-D-FPGA, we use two metrics; the improvement in the geometric average of the pin-to-pin net delays, and the improvement in critical path delay, which includes the LB delays along the path. By improvement here we mean the ratio of the

TABLE IV
COMPARISON OF CAPACITIVE LOADING FOR DIRECT CONNECTION AND LENGTH-3 CONNECTION IN 2-D-FPGA(1,2) AND NEW 2-D-FPGA

Component	Direct (×100 fF)		Length-3 (×100 fF)		Length-9 (×100 fF)	
	Baseline	New	Baseline	New	Baseline	New
Output Segment (1)	44.9	23.0	44.9	23.0	44.9	23.0
SP/RP (2)	12.8	5.7	12.8	11.4	51.2	21.0
Interconnect Segment (3)	36.6	19.1	96.9	47.6	193.9	95.1
Input Segment (4)	43.4	41.1	43.4	41.1	43.4	41.1
Signal Bend (5)	-	-	-	-	30.2	80.3
Total	137.7	88.9	198.0	123.1	363.5	260.6

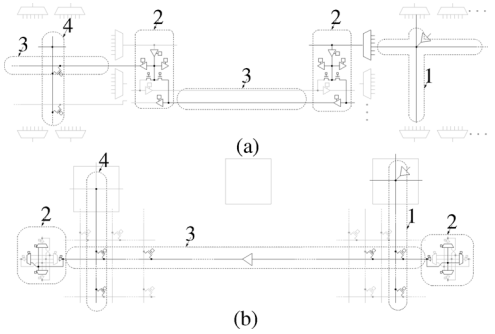

Fig. 16. Length 3 net connection in new fabric (a) and in baseline fabric (b). The numbers correspond to (1) output segment, (2) routing point/ switching point, (3) interconnect segment, and (4) input segment..

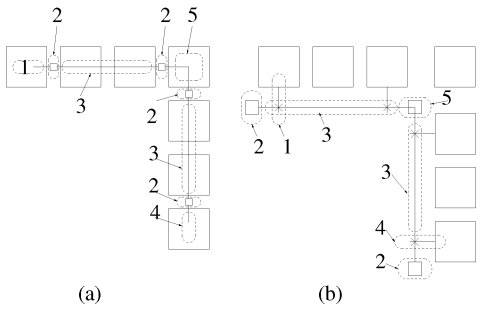

Fig. 17. Length 6 net connection with a bend in new fabric (a) and in baseline fabric (b). The numbers correspond to (1) output segment, (2) routing point/ switching point, (3) interconnect segment, (4) input segment, and (5) signal bend.

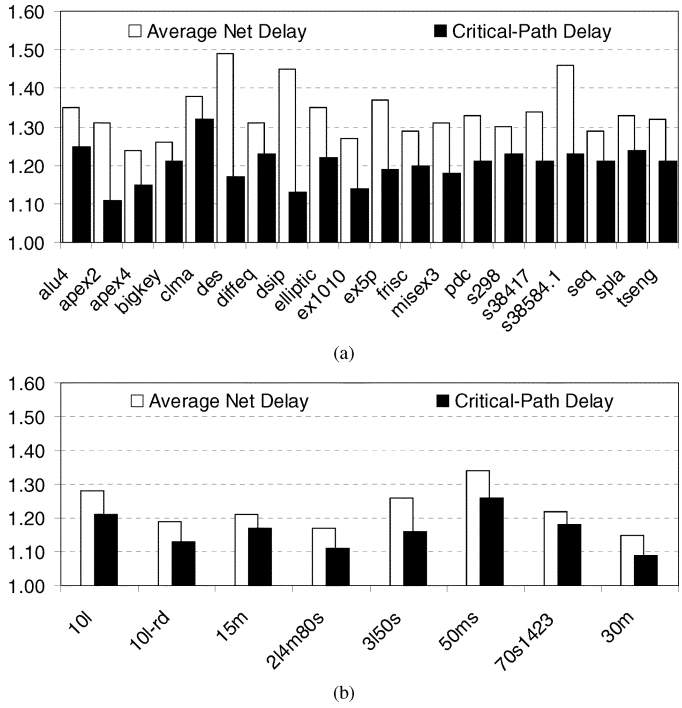

Fig. 18. Delay improvements of new 2-D-FPGA over baseline 2-D-FPGA for (a) MCNC benchmark circuits and (b) benchmark designs synthesized with QUIP toolkit from Altera.

delay in the baseline 2-D-FPGA to that in the new 2-D-FPGA. We use two sets of benchmark designs, the 20 largest MCNC benchmark circuits and 8 designs synthesized by QUIP toolkit from Altera.[3] The circuits in the second set are around 4 times larger than the largest MCNC circuit. Results for both benchmark sets are plotted in Fig. 18. Note that for the MCNC benchmarks, the improvements over the baseline 2-D-FPGA range from 1.24 to 1.49 times for the geometric average pin-to-pin delay and from 1.11 to 1.32 times for the critical path delay. On average, pin-to-pin net delay improves by 1.34 times and critical path delay improves by 1.21 times over the baseline 2-D-FPGA. The improvement for the QUIP benchmarks ranges from 1.15 to 1.34 times for the geometric average pin-to-pin delay and from 1.09 to 1.26 times for the critical path delay. On average, pin-to-pin net delay improves by 1.23 times and critical path delay improves by 1.16 times over the baseline 2-D-FPGA. The reason for the lower improvements in the large designs can be attributed to the choice of segmentation. We believe that larger improvements can be achieved by optimizing the segmentation in the new fabric, for example, using TORCH [18].

[3]Quartus II University Interface Program. 2005.

F. Dynamic Power

Dynamic power consumption consists of three components, the power consumed in the logic blocks P_{LB}, the power consumed in the interconnects P_{int}, and the power consumed in the clock networks P_{clk}. In this paper, we assume that the new 2-D-FPGA uses the same logic block as the baseline 2-D-FPGA. Although potentially the logic block in the new FPGA can have different glitching characteristics that can cause changes in its power consumption, in our study, we assume the component of dynamic power consumed by the logic block in the new 2-D-FPGA is the same as that in the baseline 2-D-FPGA.

We quantify the improvement in the total dynamic power consumption, ξ, between the baseline 2-D-FPGA and the new

2-D-FPGA both implemented in 65-nm technology using the equation (see detailed derivation in [12])

$$\xi = \frac{1}{\left(\phi_{\rm LB} + \frac{\phi_{\rm int}}{\xi_{\rm int}} + \frac{\phi_{\rm clk}}{\xi_{\rm clk}}\right)} \quad (1)$$

where $\phi_{\rm LB}$, $\phi_{\rm int}$, and $\phi_{\rm clk}$ are the fraction of dynamic power consumed in the logic blocks, the interconnect, and the clock network of the baseline 2-D-FPGA, respectively ($\phi_{\rm LB} + \phi_{\rm int} + \phi_{\rm clk} = 1$), and $\xi_{\rm int} \geq 1$ is the ratio of the dynamic power consumed by the interconnects in the 2-D-FPGA to that in the new 2-D-FPGA for a particular benchmark circuit in MCNC suite.

We choose $\phi_{\rm LB} = 0.15$, $\phi_{\rm int} = 0.65$, and $\phi_{\rm clk} = 0.2$ to be consistent with recent studies [9], [10]. Because the component of dynamic power consumed in the interconnects is proportional to the total capacitance of all signal nets for a fixed activity factor, we set $\xi_{\rm int}$ equal to the ratio of the total signal net capacitance of the baseline 2-D-FPGA to that in the new 2-D-FPGA for each placed and routed benchmark circuit. We use the same procedure to estimate the dynamic power improvement factor for the clock network $\xi_{\rm clk}$. We assume the H-tree clock distribution network with distributed buffering [16], [29], [30]. The change of the power consumption in clock network is mainly due to the area change in the new FPGA. We compute the dynamic power improvement ξ for each of the 20 MCNC benchmark circuits assuming a 64×64 LB array for both the baseline 2-D-FPGA and the new 2-D-FPGA implemented in 65-nm technology. The results in Fig. 19(a) show a 1.49 to 1.85 times improvement in total dynamic power with an average improvement of 1.56 times. The same procedure is repeated to compute the dynamic power improvement ξ for each of the eight benchmark circuits synthesized with QUIP toolkit assuming a 100×100 LB array for both the baseline 2-D-FPGA and the new 2-D-FPGA implemented in 65-nm technology. The results in Fig. 19(b) show a 1.31 to 1.56 times improvement in total dynamic power with an average improvement of 1.47 times. Again, the reason for the reduction in improvement versus the MCNC designs is the choice of segmentation.

G. Performance Comparison Summary

Table V summarizes the results of Sections V-V-C, V-E, and V-F. While the 2-D-FPGA(1,2) has better power consumption and logic density than the baseline 2-D-FPGA, it has worse delay. The new 2-D-FPGA achieves significantly better power consumption than the 2-D-FPGA(1,2), while improving delay over the baseline 2-D-FPGA with only a small penalty in logic density.

VI. New 3-D-FPGA Versus Baseline 2-D-FPGA

In this section, we compare the performance of a monolithically stacked 3-D-FPGA using the new fabric to the baseline 2-D-FPGA. First, we discuss how the 3-D-FPGA may be implemented.

A. 3-D Implementation Feasibility

The new 3-D-FPGA can be realized by stacking three active layers on top of a standard CMOS layer with a total of 12 metal

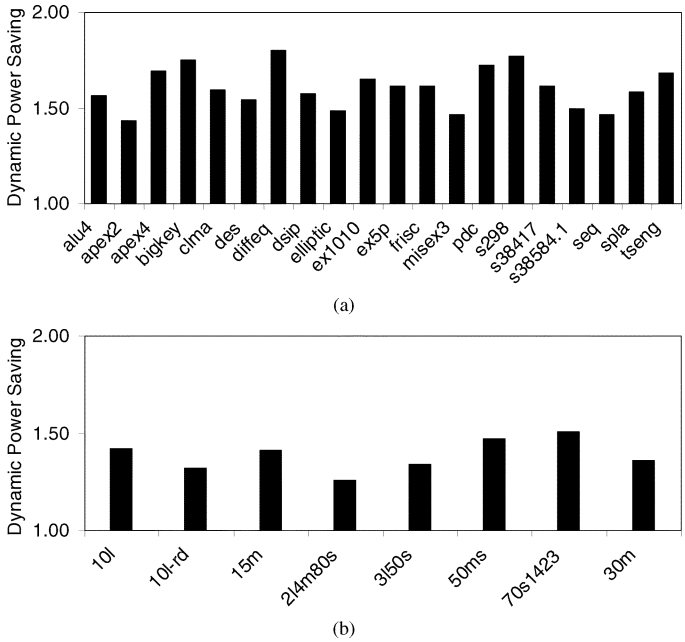

Fig. 19. Dynamic power saving of new 2-D-FPGA over baseline 2-D-FPGA for (a) MCNC benchmark circuits and (b) QUIP toolkit benchmark designs.

TABLE V
Performance Improvements

	Logic Density	Pin-to-Pin Delay (Critical Path Delay)	Dynamic Power
Baseline	1.00	1.00 (1.00)	1.00
Baseline (1,2)	1.14	0.91 (0.83)	1.18
New	0.92	1.31 (1.19)	1.54

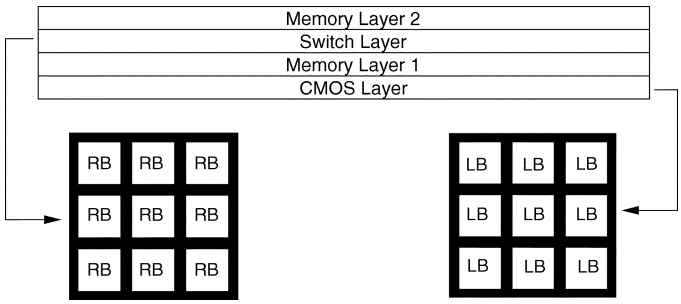

Fig. 20. Active layers of a 3-D-FPGA using proposed routing fabric. (RB: Routing block, LB: Logic block.

layers interspersed between them (see Fig. 20). The stacked active layers consist of: 1) a first configuration memory layer to program the logic blocks and tri-state buffers, 2) a switch layer comprised only of NMOS devices, and 3) a second configuration memory layer for programming the switches in the switch layer. Because the switch layer has only NMOS device, all buffers are located in the bottom CMOS layer. The use of two memory layers, instead of one as assumed in [12], provides better local vertical connectivity and relaxes the requirement on memory cell size. The 3-D-FPGA is completely tileable, so we focus on the implementation of a single tile consisting of a stack of one logic block and interconnect tri-state buffers, one routing block, and their configuration memory.

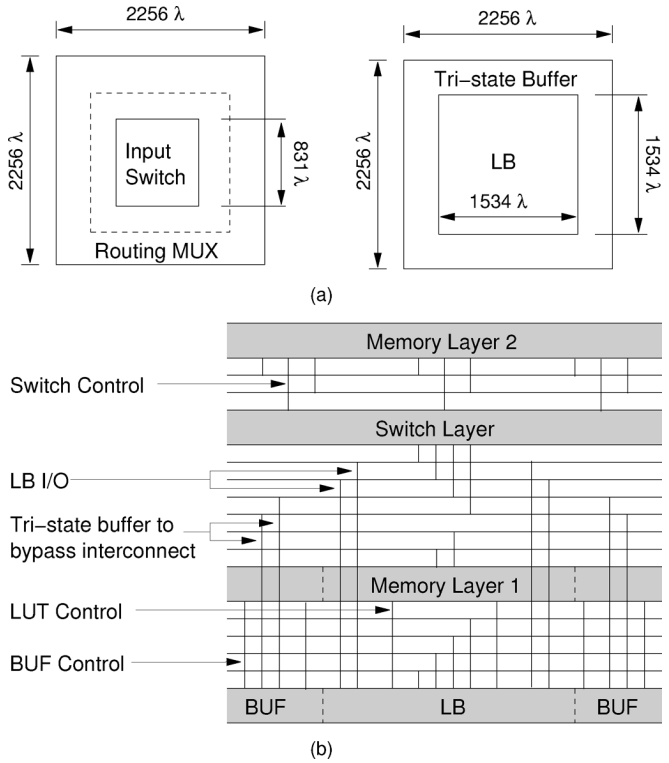

Fig. 21. (a) Tile size and (b) Layer and metal assignment in the new 3-D-FPGA.

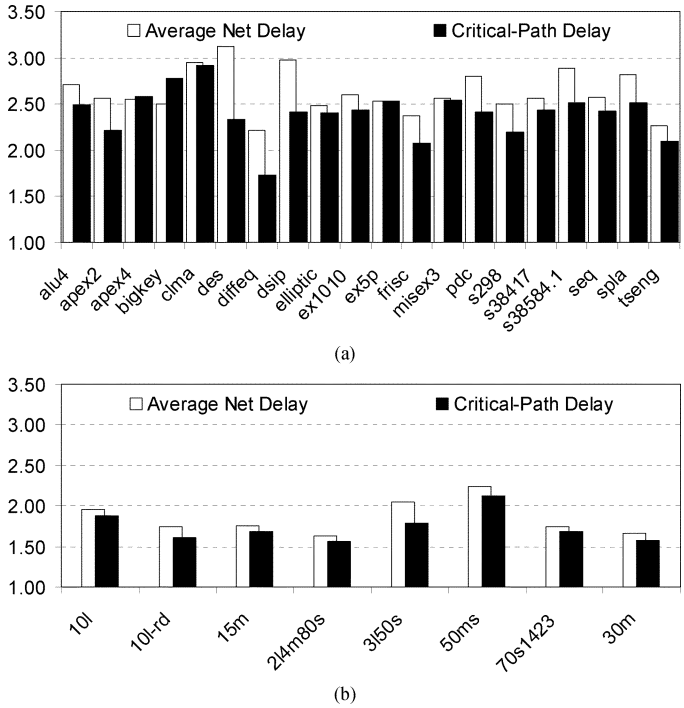

Fig. 22. Delay improvements of new 3-D-FPGA over baseline 2-D-FPGA for MCNC benchmark circuits. (a) Geometric average pin-to-pin delay. (b) Critical path delay.

Fig. 21(b) shows how the metal layers are allocated between the active layers. The bottom layer is a standard CMOS layer with both NMOS and PMOS devices available and is used to implement the logic block and buffers. A minimum of four layers of metal are assumed for signal and power and ground connections. The second layer is the configuration memory layer for the logic block and buffers. Two layers of metal are assumed for this layer. The third layer is the switch layer, where the routing block and interconnect segments are implemented. We assume four layers of metal for this layer. The top layer is the configuration memory layer for the switch layer, which requires two layers of metal. Thus, a minimum of 12 layers of metal are needed to implement this architecture. The vertical lines in Fig. 21(b) depict the vertical connections between different layers. Note that a connection that spans more than two layers is implemented using multiple vias between consecutive active layers. To estimate the tile area of the new 3-D-FPGA, we use the same methodology as in Section V-C. Our estimated footprint size of the new 3-D-FPGA tile is around $2256\lambda \times 2256\lambda$ [Fig. 21(a)]. The logic block and buffer tile and the routing block in the new 3-D-FPGA have similar areas. This is compared to a tile size of $4100\lambda \times 4100\lambda$ for the baseline 2-D-FPGA [12] and corresponds to an improvement in logic density of around 3.3 times, which is slightly better than reported in the previous study [12].

The reason for using two configuration memory layers versus one as in [12] is to relax the area requirement for the configuration memory cell and to provide greater and simpler vertical connectivity.

B. Performance Comparison

As in Section V, we compare the system delay performance using two metrics, the improvement in the geometric average of the pin-to-pin net delays, and the improvement in critical path delay, which includes the LB delays along the path. Results for the largest 20 MCNC benchmark circuits are plotted in Fig. 22. Note that the improvements over the baseline 2-D-FPGA range from 2.31 times to 3.19 times for the geometric average pin-to-pin delay and from 1.82 times and 3.01 times for the critical path delay. On average, there is a 2.59 times delay improvement in pin-to-pin net delay and 2.51 times delay improvement in critical path delay over the baseline 2-D-FPGA. The improvement for the QUIP benchmarks ranges from 1.72 to 2.31 times for the geometric average pin-to-pin delay and from 1.63 to 2.23 times for the critical path delay. On average, pin-to-pin net delay improves by 2.03 times and critical path delay improves by 1.92 times over the baseline 2-D-FPGA.

To obtain the power consumption improvement of new 3-D-FPGA over the baseline 2-D-FPGA, we used the same estimation method as detailed in Section V-F. Again, we computed the dynamic power improvement ξ for each of the 20 MCNC benchmark circuits assuming a 64×64 LB array for both the baseline 2-D-FPGA and the new 3-D-FPGA implemented in 65-nm technology. Our results in Fig. 23 show a 2.59 times to 3.24 times improvement in total dynamic power with an average improvement of 2.91 times. The same procedure is repeated to compute the dynamic power improvement ξ for each of the seven benchmark circuits synthesized with QUIP toolkit assuming a 100×100 LB array for both the baseline 2-D-FPGA and the new 3-D-FPGA implemented in 65-nm technology. The results

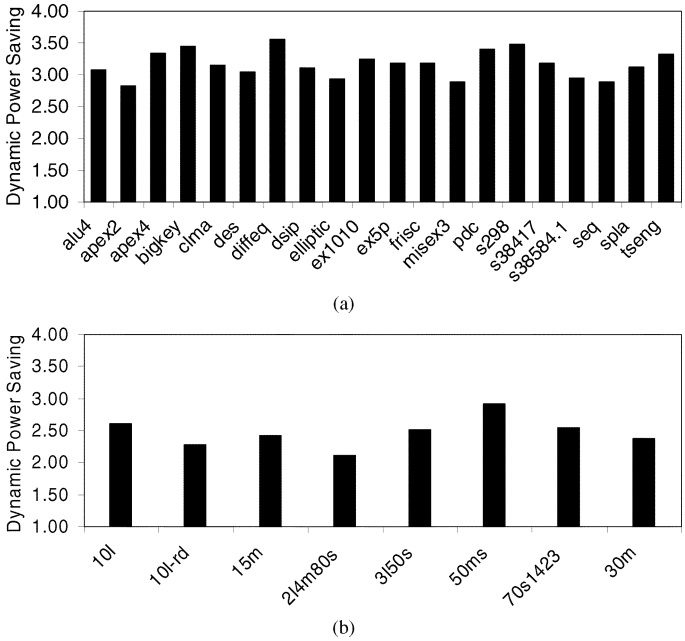

Fig. 23. Dynamic power saving of new 3-D-FPGA over baseline 2-D-FPGA for MCNC benchmark circuits.

in Fig. 23(b) show a 2.33 to 3.02 times improvement in total dynamic power with an average improvement of 2.61 times.

VII. Conclusion

The power inefficiency of FPGAs relative to cell-based ASICs is a major impediment to their adoption in a broad range of applications and to the continuing improvement in their logic density and performance. In an effort to address this major problem, we proposed a new FPGA routing fabric and showed that an FPGA that uses this fabric can achieve 1.57 times reduction in the overall dynamic power consumption and 1.35 times improvement in average net delays with only 9% reduction in logic density over an island-style baseline 2-D-FPGA implemented in the same 65-nm CMOS technology. The improvements in delay and dynamic power consumption are achieved by using only Single and Double interconnect segments to reduce routed net lengths and by reducing the component of interconnect loading due to programming overhead without adversely affecting routability.

The improvement results are for specific choice of segmentation and transistor and buffer sizes. Further improvements can be achieved by optimizing segmentation using TORCH [18]. This would be particularly important for large designs. Power and logic density can be further improved at the expense of extra delay by using smaller transistor and buffer sizes. To illustrate this, we reduced the MUX transistor sizes from 4 to 3, the pass transistor switch sizes from 4 to 3, and the LB input/output buffer sizes by a factor of 2. This resulted in 11% reduction in silicon area and 7% reduction in power consumption with only 4% increase in delay. An area of improvement in the new fabric is to reduce the high loading incurred by a signal bend. Although the cost of a signal bend can be factored in the placement and routing cost, it would be interesting to investigate architectural ways to reduce it.

The new fabric is well-suited to monolithically stacked 3-D implementation. We showed that a 3-D-FPGA using the new routing fabric can be implemented using a four-layer stack consisting of a CMOS layer, a switch transistor layer, and two memory layers with 12 metal layers between them. This new 3-D-FPGA achieves a 3.3 times improvement in logic density, a 2.51 times improvement in delay, and a 2.93 times improvement in dynamic power consumption over the same baseline 2-D-FPGA. Much work remains to verify the general validity of these performance improvements. However, the fact that the performance improvements presented here are indeed very large warrants continued investigation.

Acknowledgment

The authors would like to thank the anonymous reviewers for many constructive comments that have greatly improved the presentation of this paper.

References

[1] "Power Consumption in 65 nm FPGAs." White Paper Xilinx, Inc., 2006.
[2] "Stratix III Programmable Power." White Paper Altera, Inc., 2006.
[3] I. Kuon and J. Rose, "Measuring the gap between FPGAs and ASICs," in *Proc. 2006 ACM/SIGDA 10th Int. Symp. Field-Programmable Gate Arrays*, 2006, pp. 21–30.
[4] F. Li, Y. Lin, L. He, and J. Cong, "Low-power FPGA using pre-defined dual-Vdd/dual-Vt fabrics," in *Proc. 2004 ACM/SIGDA 12th Int. Symp. Field-Programmable Gate Arrays*, 2004, pp. 42–50.
[5] Y. Lin, F. Li, and L. He, "Circuits and architectures for field programmable gate array with configurable supply voltage," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 13, no. 9, pp. 1035–1047, Sep. 2005.
[6] J. H. Anderson and F. N. Najm, "Low-power programmable routing circuitry for FPGAs," in *Proc. IEEE Int. Conf. Comput.-Aided Design*, 2004, pp. 602–609.
[7] F. Li, D. Chen, L. He, and J. Cong, "Architecture analysis and automation: Architecture evaluation for power-efficient FPGAs," in *Proc. 2003 ACM/SIGDA 10th Int. Symp. Field-Programmable Gate Arrays*, 2003, pp. 175–184.
[8] T. Tuan, S. Kao, A. Rahman, S. Das, and S. Trimberger, "A 90 nm low-power FPGA for battery-powered applications," in *Proc. 2006 ACM/SIGDA 10th Int. Symp. Field-Programmable Gate Arrays*, 2006, pp. 3–11.
[9] L. Shang, A. S. Kaviani, and K. Bathala, "Dynamic power consumption in Virtex-II FPGA family," in *Proc. 2002 ACM/SIGDA 10th Int. Symp. Field-Programmable Gate Arrays*, 2002, pp. 157–164.
[10] V. George, "Low energy field-programmable gate array," Ph.D. dissertation, Univ. of California, Berkeley, 2000.
[11] M. Lin and A. El Gamal, "A routing fabric for monolithically stacked 3-D-FPGAs," in *Proc. 2007 ACM/SIGDA 10th Int. Symp. Field-Programmable Gate Arrays*, 2007, pp. 1–10.
[12] M. Lin, A. El Gamal, Y.-C. Lu, and S. Wong, "Performance benefits of monolithically stacked 3-D-FPGA," in *Proc. 2006 ACM/SIGDA 10th Int. Symp. Field-Programmable Gate Arrays*, 2006, pp. 113–122.
[13] A. R. Joshi and K. C. Saraswat, "Nickel induced crystallization of a-Si gate electrode at 500C and MOS capacitor reliability," *IEEE Trans. Electron Devices*, vol. 50, no. 4, pp. 1058–1062, Apr. 2003.
[14] S. M. Jung, J. Jang, W. Cho, J. Moon, K. Kwak, B. Choi, B. Hwang, H. Lim, J. Jeong, J. Kim, and K. Kim, "The revolutionary and truly 3-dimensional 25F2 SRAM cell technology with the smallest S3 (Stacked single-crystal Si) cell, 0.16 μm^2, and SSTFT(Stacked single-crystal thin film transistor) for ultra high density SRAM," in *Tech. Dig. 2004 VLSI Technol. Symp.*, 2004, pp. 228–229.
[15] "Virtex-II Cpmplete Datasheet (All Four Modules)," Xilinx, Inc., 2007.
[16] F. Li, L. He, and J. Cong, "Architecture evaluation for power-efficient FPGAs," in *Proc. 2003 ACM/SIGDA 11th Int. Symp. Field-Programmable Gate Arrays*, 2003, pp. 175–184.
[17] F. Li, Y. Lin, L. He, D. Chen, and J. Cong, "Power modeling and characteristics of field programmable gate arrays," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.* , vol. 24, no. 11, pp. 1712–1724, Nov. 2005.

[18] "TORCH: A tool for segmented routing channel design in FPGAs, "A routing fabric for monolithically stacked 3-D-FPGAs"," in *Proc. 2008 ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2008, pp. 131–138.
[19] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*. Norwell, MA: Kluwer, 1990.
[20] V. Betz and J. Rose, "Directional bias and non-uniformity in FPGA global routing architectures," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design. ICCAD-96. Dig. Tech. Papers.,* , Nov. 10–14, 1996, pp. 652–659.
[21] V. Betz and J. Rose, "Effect of the prefabricated routing track distribution on FPGA area-efficiency," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 6, no. 9, pp. 445–456, Sep. 1998.
[22] V. Betz and J. Rose, "FPGA routing architecture: Segmentation and buffering to optimize speed and density," in *Proc. 1999 ACM/SIGDA 7th Int. Symp. Field Programmable Gate Arrays*, 1999, pp. 59–68.
[23] W. Xu, VPR for Virtex [Online]. Available: http://www-unix.ecs.umass.edu/~wxu/jbits/VPR_for_Virtex.htm.
[24] V. Betz, J. Rose, and A. Marquardt, Eds., Architecture and CAD for Deep-Submicron FPGAs Norwell, MA, Kluwer, 1999.
[25] G. Lemieux and D. Lewis, "Circuit design of routing switches," in *Proc. ACM/SIGDA 10th Int. Symp. Field-Programmable Gate Arrays*, 2002, pp. 19–28.
[26] S. Yang, "Logic Synthesis and Optimization Benchmarks," Tech. Rep. Microelectron. Center of North Carolina, 1991, 3.0.
[27] V. Betz and J. Rose, "VPR: A new packing, placement and routing tool for FPGA research," in *Proc. 7th Int. Workshop Field-Programmable Logic Applicat.*, 1997, pp. 213–222.
[28] C. Ebeling, L. McMurchie, S. Hauck, and S. Burns, "Placement and routing tools for the Triptych FPGA," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 3, no. 4, pp. 473–482, Apr. 1995.
[29] D. Liu and C. Svensson, "Power consumption estimation in CMOS VLSI chips," *IEEE J. Solid-State Circuits*, vol. 29, no. 6, pp. 663–670, Jun. 1994.
[30] E. Friedman, "Clock distribution design in VLSI circuits—An overview," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 1993, pp. 1475–1478.

Mingjie Lin (S'01–M'08) received the B.S. degree in engineering from Xi'an Jiaotong University, Xi'an, China, in 1993, the M.S. degree in mechanical engineering and electrical engineering from Clemson University, Clemson, SC, in 2001, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 2008.

His current research interests include next-generation FPGA architectures and reconfigurable parallel many-core computer architecture.

Abbas El Gamal (S'71–M'73–SM'83–F'00) received the B.Sc. degree in electrical engineering from Cairo University, Cairo, Egypt, in 1972, and the M.S. degree in statistics and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1977 and 1978, respectively.

From 1978 to 1980, he was an Assistant Professor of electrical engineering with the University of Southern California, Los Angeles. Since 1981, he has been with Stanford University, where he is currently a Professor of electrical engineering and the Director of the Information Systems Laboratory. From 1984 to 1986, he served as the Director of the LSI Logic Research Laboratory. In 1986, he cofounded Actel Corporation and served as its Chief Scientist. In 1990, he cofounded Silicon Architects and served as its Chief Technical Officer until it was acquired by Synopsys in 1995. From 1997 to 2003, he was a Principal Investigator on the Stanford Programmable Digital Camera project. His research contributions have spanned several areas in information theory, digital imaging, and integrated circuit design and design automation. He has authored and coauthored more than 180 papers and holds 30 patents in these areas. He has served on the board of directors and advisory boards of several silicon valley companies.