Measuring and Utilizing the Correlation Between Signal Connectivity and Signal Positioning for FPGAs Containing Multi-Bit Building Blocks

Andy Ye and Jonathan Rose

The Edward S. Rogers Sr. Department of Electrical and Computer Engineering
University of Toronto
10 King's College Road
Toronto, Ontario
Canada M5S 3G4
email: {yeandy, jayar}@eecg.utoronto.ca

ABSTRACT

As the logic capacity of FPGA increases, there has been a corresponding increase in the variety of FPGA building blocks. From a mere collection of the conventional logic blocks, FPGAs now can include digital signal processors, multipliers, multi-bit addressable memory cells, and even processor cores; and one of the common characteristics of these new building blocks is their multi-bit design, where each block is designed specifically to process several bits of data at a time. This multi-bit processing paradigm is significantly different from the single-bit processing design of the conventional FPGA logic blocks; and it creates differentiation in signals through its bussed structures. Consequently, this paper examines the correlation between the positions of the signals in buses and the connectivity of these signals. Based on the correlation measurements, a multi-bit routing architecture is then proposed along with its routing tool. It is experimentally shown that, comparing to the conventional routing architectures, the multi-bit architecture requires 12% less area to implement; and in particular, it needs 27% less routing switches to connect its multi-bit blocks to their routing tracks, and 18% less configuration memory to store the configuration information.

1. INTRODUCTION

Over the years, there has been a dramatic increase in the variety of Field-Programmable Gate Array (FPGA) building blocks. Evolving from a mere combination of simple logic blocks and bit-addressable memory cells, FPGAs now can include digital signal processing blocks, multipliers, multi-bit addressable memory, and even processor cores; and one significant difference between these new building blocks and the classical logic block design is in the way that they process data. On one hand, the classical logic blocks are designed to process one bit of data at a time; on the other, the new building blocks are designed to process multiple bits of data simultaneously; and the use of these multi-bit processing elements presents new opportunities for exploiting datapath regularity.

In particular, while the conventional FPGA logic blocks are connected to FPGA routing through individual input and output signals, the inputs and outputs of the multi-bit building blocks can be grouped into buses; and the positions of signals in these buses often strongly correlate to the way that they connect. This strong correlation between the connectivity and the physical positioning of signals can create opportunities for FPGA architects to selectively remove routing switches from the routing fabric of an FPGA and to share configuration memory — all, at the same time, maintaining the original routability of the architecture. Since routing switches often consume a significant amount of FPGA area, reducing the total number of routing switches and their configuration memory in an FPGA can significantly increase its area efficiency, especially for implementing large arithmetic-intensive datapath circuits, including computer graphics, multimedia, digital signal processing, and Internet routing applications.

Several FPGAs containing only multi-bit building blocks have been proposed over the years [1]–[12]. While they come in a wide variety of routing architectures, in this work, we focus on the study of FPGAs containing segmented-style routing resources [13]. In particular, we empirically measure the correlation between signal connectivity and the position of these signals in buses for several datapath circuits. The correlation is then used to remove routing switches and to share configuration memory in order to increase the overall area efficiency of the routing fabric. Note that the primary reason for the choice of segmented-style routing resources is due to the fact that these are the building blocks of many state-of-the-art commercial FPGAs (including the Altera Flex, Stratix, and Cyclone series [14] and Xilinx 5200, Virtex, and Spartan families [15] of FPGAs); and with their ever-increasing logic capacity, commercial FPGAs are being increasingly used to implement large datapath-intensive applications.

For any new FPGA architecture, it is essential to have a set of automated design tools that can make the effective use of its new architectural features. In particular, the sharing of configuration memory place new demand on Computer Aided Design (CAD) tools. As a result, a set of datapath-oriented CAD tools, including synthesis [16], packing [17], placement [18], and routing tools, have been developed at the University of Toronto; and in this paper, we focus on the particular problem of automated routing. Routing for architectures containing

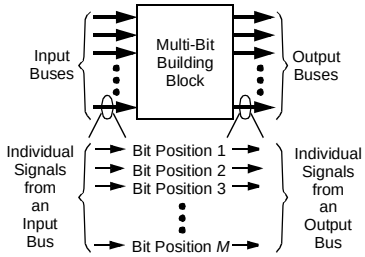


Fig. 1. Multi-Bit Building Block Interface

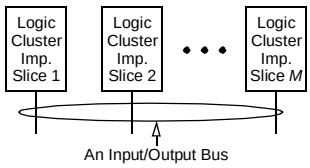


Fig. 2. Modeling the Bussed Interface

configuration-memory sharing switches is more difficult than classical routing [13] [19] [20] [21] [22], since the router has to properly model the new architectural features; and, additionally, in order to effectively utilize these added features, the router has to preserve the regularity of datapath circuits throughout the routing process, while, at the same time, attempting to achieve the conventional routing objectives of minimizing congestion and critical path delay. As a result, this paper presents a new routing algorithm that is an evolution of the classical Negotiated-Congestion (NC) routing algorithm [20]. The algorithm leverages many existing features of the classical router while incorporating several new metrics, which are designed to measure the regularity of datapath circuits, into the traditional cost functions of congestion and critical path delay.

While two previous papers [23] [24] have examined the area efficiency of datapath-oriented FPGAs, this work advances the research in two fundamental ways. First, it is the first to statistically quantify the routing demand of datapath circuits; and these statistics are valuable for both designing specialized datapath-oriented FPGAs and for improving the connectivity of multi-bit building blocks in conventional FPGAs. Secondly, this paper proposes a new routing algorithm for configuration memory sharing resources (which can be use to take advantage of the statistics in order to improve area efficiency).

The rest of this paper is organized as follows: Section 2 presents the correlation measurements; Section 3 and Section 4 describe the routing architecture and its corresponding routing algorithm, respectively; Section 5 presents the experimental results on the area efficiency of the proposed architecture; and concluding remarks are presented in Section 6.

2. SIGNAL CONNECTIVITY VS. SIGNAL POSITIONING

As shown in Figure 1, the input and output signals of a multi-bit block can be grouped into buses; and each signal in a bus can be associated with an unique integer number indicating the bit position of the signal in the bus. Note that this bussed structure arises from the regularity of datapath circuits, where a datapath is created by duplicating a single sub-design, called a *bit-slice*, multiple times. For circuits implemented using conventional logic blocks, a majority of this regularity is often destroyed by the CAD tools during the optimization process; for circuits implemented using the multi-bit blocks, on the other hand, these bit-slices are routinely preserved for the purpose of multi-bit processing; and since the primary purpose of FPGA routing is to provide connectivity between the input and output signals of various FPGA building blocks, it is important to examine the relationship between the connectivity and the bit positions of these signals.

Table 1. Two-Terminal Connections for $M = 2$

Circuit Name	Two-Terminal Connections				
	Src. = 1, Sink = 1	Src. = 1, Sink = 2	Src. = 2, Sink = 1	Src. = 2, Sink = 2	Per Cir. Total
code_seq_dp	40%	22%	12%	26%	873
dcu_dpath	39%	11%	11%	39%	2247
ex_dpath	38%	13%	12%	37%	7127
exponent_dp	30%	19%	20%	30%	1399
icu_dpath	39%	15%	11%	35%	8160
imdr_dpath	34%	14%	16%	36%	3030
incmod	31%	25%	19%	26%	2248
mantissa_dp	36%	17%	15%	33%	2554
multmod_dp	26%	25%	24%	25%	3645
pipe_dpath	40%	16%	10%	34%	1134
prils_dp	31%	23%	20%	27%	983
rsadd_dp	38%	16%	13%	34%	740
smu_dpath	35%	17%	16%	33%	1211
ucode_dat	39%	12%	11%	38%	3304
ucode_reg	47%	14%	4%	36%	191
Total	**36%**	**16%**	**14%**	**34%**	**38846**

Since multi-bit blocks do come in many different physical forms, this work uses a model containing M logic clusters [13] to capture their common characteristics of bussed interface. It is assumed that each logic cluster consists of a group of four tightly connected Look-Up Tables (LUTs) and four flip-flops, and all buses in a block is of width M, where M is called the granularity of the block. As shown in Figure 2, the model assumes that each cluster is responsible for generating one bit in each bus — formed by either a group of multi-bit block input signals or a group of multi-bit block output signals. To map a datapath circuit onto the multi-bit blocks, logic in each bit-slice is first grouped into a series of logic clusters using the datapath-oriented packing algorithm as described in [17]. The same algorithm is then used to group M identical logic clusters from the neighboring bit-slices into a multi-bit block.

Once a circuit is mapped into multi-bit blocks, the connectivity between the multi-bit block input and output signals can be measured using *two-terminal connections*, where each con-

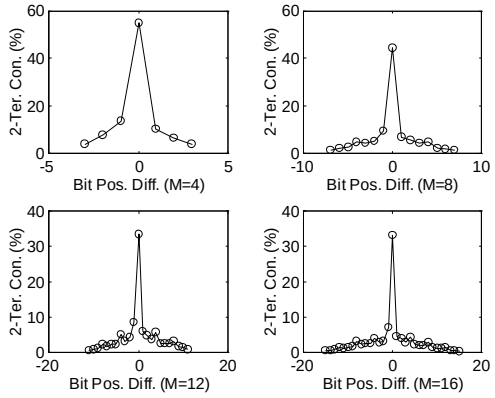


Fig. 3. Correlation Between Bit-Position Difference and Signal Connectivity

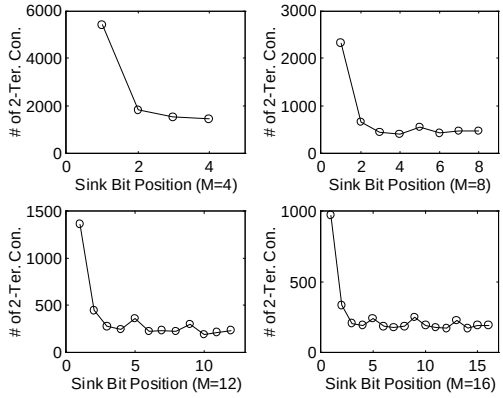


Fig. 4. Number of Inputs that Outputs at Bit Position 1 are Connected To

nection consists of a connected pair of a multi-bit block input (called the *sink* of the connection) and a multi-bit block output signal (called the *source* of the connection). We then classify these two-terminal connections based on the bit positions of their sources and sinks.

In particular, for the granularity value of two, all connections can be classified into four types as shown in Table 1 — namely connections with both sources and sinks from bit position 1, connections with both sources and sinks from bit position 2, connections with sources from bit position 1 and sinks from bit position 2, and connections with sources from bit position 2 and sinks from bit position 1. The table then shows the number of connections in each type as a percentage of the total number of two-terminal connections for fifteen benchmark circuits from the Pico-Java processor [25] in column 2, 3, 4, and 5, respectively; and column 6 shows the total number of two-terminal connections in each circuit. As shown, a majority (70%) of the connections in these benchmark circuits have the same source and sink bit positions. Connections with different source and sink bit positions, on the other hand, consist of only 30% of the total number of two-terminal connections.

This strong correlation between the bit position difference and the signal connectivity is observed across all granularity values; and Figure 3 plots the difference against the number of two-terminal connections (as a percentage of the total number of two-terminal connections) for the granularity of 4, 8, 12, and 16. As shown, connections with the same source and sink bit positions (where the difference value is 0) consist of from over 30% (for $M = 12$ and $M = 16$) to over 70% (for $M = 2$) of the total number of two-terminal connections; and these percentage values are significantly greater than all other percentage values.

The correlation exists still at the level of individual bit positions. In particular, Figure 4 shows the connectivity of multi-bit block output signals at bit position 1 in terms of the number of input signals that the outputs are connected to. As shown, for all granularity values, the output signals are connected to a significantly higher amount of input signals that are also from bit position 1 than inputs from any other bit positions; and the same trend is observed for outputs from other bit positions (figures not shown due to space limitations).

The major architectural conclusion that can be drawn from these observations is that, for multi-bit blocks, a significant amount of connectivity should exist between the inputs and outputs that are from the same bit positions, and input and output signals from distinct bit positions, on the other hand, would require much less connectivity. Note that this observation is contrary to the connectivity requirements of the conventional FPGA logic blocks, where architects strive to uniformly distribute the connections of each logic block output signal to all available logic block input signals [13].

For two-terminal connections that have the same source and sink bit positions, we can further group some of these signals into M-bit wide buses where each signal in a bus has a distinct source/sink bit position, and all the signals in the bus originate from a common multi-bit block and terminate at another. The number of signals that exist in these buses is shown in Figure 5 for the benchmark circuits over a range of granularity values (M). Here, there are two lines in the figure, where the top line indicates the percentage of two-terminal connections that have the same source and sink bit positions, and the bottom line shows the percentage of two-terminal connections that can be grouped into M-bit wide buses. As shown, the number of signals that can be grouped into M-bit wide buses consists of a significant proportion (from 25% to 60%) of the total number of connections; and since all signals in each bus share a single source and a single sink block, these signals can be a potential source of redundant information in FPGA routing configuration. This redundancy, in turn, can be exploited by FPGA architects to improve the area efficiency of FPGAs — where a single configuration memory bit can be

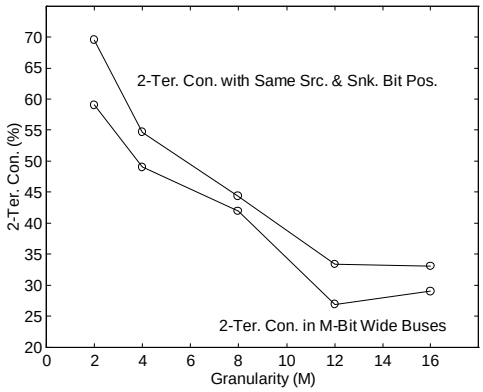


Fig. 5. Two-Terminal Connections in M-Bit Wide Buses

used in place of multiple bits of memory to store the identical configurations.

Having observed the correlation, one question that naturally arises is how exactly this correlation can be turned into area savings; and one of the best ways to address the question is through a set of empirical studies, where an FPGA architecture with varying switch patterns is first defined and then used to implement a set of benchmark circuits. Such an approach is used in this study; and the architecture used in the study is discussed next.

3. THE MULTI-BIT ROUTING ARCHITECTURE

Turning the observations above in signal connectivity into actual area savings would require the design of a concrete routing architecture; and such an architecture (called the multi-bit routing architecture) is used in this work. As shown in Figure 6, the architecture contains two types of routing tracks — the single-bit tracks and the multi-bit tracks; and each track is connected to several multi-bit block input and output signals through a set of routing switches (denoted by an X in the figure).

The single-bit tracks are similar, in structure, to the conventional FPGA routing tracks; and their switch patterns are designed to uniformly distribute multi-bit block output connections to all available multi-bit block inputs [13]. The multi-bit tracks, on the other hand, are organized into M-bit wide buses; and, in a bus, each track is assigned a unique bit position number, ranging from 1 to M. These numbers are then used to match the bit positions of the signals that the tracks are connected to. For example, a track at bit position 1 can only be connected to multi-bit block input and output signals that are also from bit position 1; and in general, a track at bit position x can only be connected to input and output signals also from bit position x. Consequently, increasing the number of multi-bit tracks in the architecture only increases the availability of one type of two-terminal connections, namely two-terminal connections with the same source and sink bit positions.

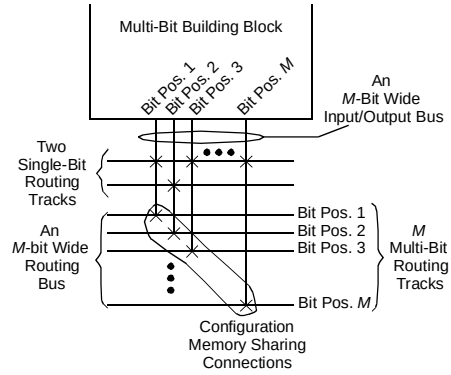


Fig. 6. Multi-Bit Routing

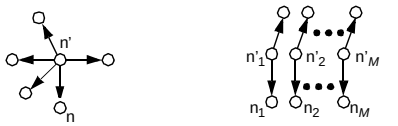
Increasing the number of single-bit tracks, on the other hand, uniformly increases the availability of all types of two-terminal connections.

Furthermore, the routing switches in each routing bus (which consists of a group of M multi-bit routing tracks) are further grouped into M-bit wide groups in order to exploit the observed redundant configuration information; and each group, as shown in Figure 6, shares a single set of configuration memory. The sharing of configuration memory, however, complicates the design of the routing tools; and in the next section, a new NC-based routing algorithm is introduced. The algorithm accommodates configuration memory sharing; and both the architecture and the algorithm are then used to empirically measure the overall effect of this correlation between signal connectivity and signal positioning on the area efficiency of FPGAs.

4. ROUTING ON THE CONFIGURATION-MEMORY SHARING ROUTING RESOURCES

As an NC-based routing algorithm, routing is performed in multiple routing iterations; and each iteration is controlled by a set of cost functions, which measure the delay and the congestion of each route. These cost functions consist of a collection of cost metrics; and each metric is updated at the end of each routing iteration based on the current and the historical routing results.

During a routing iteration, two-terminal connections that can be grouped into M-bit wide buses (as defined in Section 2) are first routed. In particular, all signals in the bus are routed through the multi-bit routing tracks (which are connected by the configuration-memory sharing switches) as a single group. Then a bit chosen at random from the bus is routed through the single-bit tracks as an individual signal. These two routing solutions — one consists of a series of multi-bit routing tracks and the other consists of a series of single-bit routing tracks — are then compared based on their congestion and delay metrics. If the single-bit tracks provide a better solution (because of much lower delay or congestion), the multi-bit track solution is abandoned in favour of routing all the signals in the bus



(a) Single-Bit Expansions (b) Multi-Bit Expansions

Fig. 7. Expansion Topologies

through the single-bit tracks. Otherwise the multi-bit track solution is used to route the bus.

For two-terminal connections that cannot be grouped into buses, on the other hand, they are routed as individual signals. Here each signal is routed through both the single-bit and the multi-bit tracks in order to obtain the best routing solutions (which means routing solutions with the best possible combination of delay and congestion).

4.1. Expansion Topologies

In conventional FPGAs, each architecture can be represented by a graph called the *routing resource graph*, where nodes represent the routing resources, and the edges represent the routing switches. To route a signal in this graph, one uses a series of wave-front expansions, where the initial wave front consists of only the source node of the signal; and at each stage of the expansion, the node with the lowest possible cost is selected from the front. The selected node is then substituted by its neighbouring nodes to expand the current front; and the expansion continues until all sinks of the signal are reached.

The multi-bit routing architecture, on the other hand, can also be represented by a routing resource graph. Here each node in the graph represents either a multi-bit block input pin, a multi-bit block output pin, a continuous wire segment between two routing switches on a single-bit routing track, or a similar wire segment on a multi-bit routing track. Each edge of the graph represents a routing switch that connects these nodes together. Similarly signals can also be routed through the graph using the technique of wave-front expansion. In particular, when routing an individual signal through the single-bit routing tracks, the expansion appears exactly the same as the topology of the conventional wave-front expansions; and an example of such an expansion is shown in Figure 7(a), where the wave front is expanded from a single node denoted by n', to five neighbouring nodes (one of which is denoted by n in the figure).

Routing either a bus or an individual signal through a routing bus (which consists of M multi-bit routing tracks), however, would require the simultaneous expansion of M wave fronts, where one wave front is used to keep track of a bit in the bus. The multiple fronts are needed in order to fully account for the expansion costs since groups of M switches in each routing bus are collectively controlled by a single set of configuration memory. Consequently, no individual connections can be made in isolation. Instead, wire segments must be connected together in M-bit wide groups.

An example of such an expansion topology is shown in Figure 7(b), where a set of wave fronts, each containing one node, denoted by n'_1 through n'_M in the figure, is expanded into two sets of nodes, where one of the sets is denoted by n_1 through n_M. This added expansion topology requires the design of a set of expansion cost functions so the single-bit expansions (such as the one shown in Figure 7(a)) can be fairly compared to the multi-bit expansions (such as the one shown in Figure 7(b)).

4.2. Expansion Costs

Since the single-bit expansion topology is identical to the conventional expansion topology, the conventional cost function as defined [13] is used in this study for these expansions. In particular, the following equation is used:

$$\begin{aligned}\text{expansion\_cost}(n) = {} & [1 - \text{criticality}] \times C(n) + \\ & \text{criticality} \times D(n) + \text{future\_expasion\_cost}(n)\end{aligned} \quad . \qquad (1)$$

Here, $C(n)$ is defined to be the *accumulated congestion cost* of all nodes in an expansion path that connects the source of the signal all the way to node n; and it is calculated based on the following formula:

$$C(n) = \text{congestion\_cost}(n) + C(n'), \qquad (2)$$

which states that the accumulated congestion cost of any node, n, is equal to the sum of the accumulated congestion cost of n' — the node immediate proceeds n on the expansion path — and the *congestion cost* of n. This *congestion cost* is a function of the *capacity* (which is equal to the maximum number of times that a node can be legally used) of node n and the number of times that n is actually being used [13].

Also in Equation 1, $D(n)$ is defined to be the *delay cost* of the expansion path that connects the source of the signal to node n. Both $C(n)$ and $D(n)$ are scaled by the *criticality* of the signal (which is a fraction between 0 and 1). A high criticality value means that as compared to other signals, the signal that is being routed has a higher delay value; therefore, a larger proportion of the expansion cost should be equal to the delay cost. Otherwise, the signal has a lower delay value; and accumulated congestion cost should be the larger proportion of the expansion cost. Finally, the final term in the equation represents the *future expansion cost*, which is an estimation on the additional delay and accumulated congestion cost that can incur in the path that connects node n to the sink of the signal.

When routing a bus of two-terminal connections through the multi-bit expansion topology, on the other hand, one have to deal with the delay and accumulated congestion of M signals and M nodes instead of one. In this case, the Equation 1 is used to calculate the expansion cost for each signal and its corresponding node; and the maximum of these costs is then used

as the overall cost of the expansion, or more formally, the expansion cost in this case is defined to be:

$$\text{expansion\_cost}'(n_1, n_2, \ldots n_M) = \max(\text{expansion\_cost}(n_1), \text{expansion\_cost}(n_2), \ldots, \text{expansion\_cost}(n_M)) \quad (3)$$

When routing a single bit of a signal through the same expansion topology, however, there will be only one delay cost; and this cost reflects the delay of the signal that is being routed. There are, however, M different accumulated congestion costs — one for each node of the expansion. As a result, Equation 3 no longer applies; and to accommodate, the accumulated congestion cost is redefined to be the maximum of all accumulated congestion costs, as follows:

$$C'(n_1, n_2, \ldots n_M) = \max(\text{congestion\_cost}(n_1), \text{congestion\_cost}(n_2), \ldots, \text{congestion\_cost}(n_M)) + C'(n'_1, n'_2, \ldots n'_M) \quad (4)$$

This accumulated congestion cost is then used in place of $C(n)$ in Equation 1 to calculate the overall expansion cost of the topology.

5. RESULTS

The router is then used with a set of datapath-oriented CAD tools to map the fifteen benchmark circuits (presented in Section 2) onto several variants of the multi-bit routing architecture presented in Section 3; and these variants primarily differ in the composition of their routing tracks where each architecture contains a different amount of single-bit and multi-bit tracks. From these architectures, the architecture with the best area is identified for each circuit; and it is then compared with a base architecture that contains only single-bit routing tracks (tracks that strictly distribute the connections of each multi-bit block output uniformly across all available multi-bit block inputs) for the number of routing switches, the amount of configuration memory (SRAM), and the area usage of routing resources.

For the study, it is assumed that each circuit is placed onto a square FPGA that contains just enough multi-bit building blocks to accommodate the given circuit. It is also assumed that each block contains just four clusters, and each cluster contains four four-input LUTs and four DFFs. The number of input signals per multi-bit block is assumed to be 40; and the number of output signals per multi-bit block is assumed to be 16. These signals are then grouped into ten 4-bit wide input buses and four 4-bit wide output buses, respectively. The routing switches that connect the wire segments together are assumed to have a disjoint switch block topology [26] (for both the single-bit and the multi-bit tracks). It is also assumed that each multi-bit block input/output pin is connected to 40%/25% of the single-bit tracks in a routing channel, and each input bus/output bus is connected to 40%/25% of the 4-bit wide routing buses. Finally each wire segment is assumed to continuously expand two multi-bit building blocks for either the single-bit or the multi-bit tracks; and all transistors are properly sized in each architecture to ensure good performance and efficient area usage.

Table 2. Track Count Per Channel

Circuit	Single-Bit + Multi-Bit		Single-Bit Only
	#S.B. Tracks	#M.B. Tracks	
code_seq_dp	37 (90%)	8	41
dcu_dpath	27 (48%)	36	56
ex_dpath	39 (45%)	52	86
exponent_dp	41 (63%)	36	65
icu_dpath	39 (45%)	60	86
imdr_dpath	47 (66%)	32	71
incmod	37 (69%)	32	54
mantissa_dp	42 (55%)	52	77
multmod_dp	57 (92%)	8	62
pipe_dpath	32 (94%)	8	34
prils_dp	33 (85%)	20	39
rsadd_dp	22 (59%)	32	37
smu_dpath	34 (79%)	16	43
ucode_dat	29 (49%)	44	59
ucode_reg	13 (46%)	36	28
Average	**35 (63%)**	**31**	**56**

Table 2 shows the number of single-bit tracks and the number of multi-bit tracks per channel that are required to implement each benchmark circuit in column 2 and column 3, respectively. It also shows the number of tracks that is needed to implement the same circuit using purely single-bit tracks in column 4. (Shown in parentheses in column 2 is the value of column 2 as a percentage of the value presented in column 4.) As shown, the multi-bit tracks can be used to significantly reduce the number of single-bit tracks in an architecture; and for a vast majority (ten) of the circuits shown in Table 2, the total number of single-bit tracks in each is reduced by over 30% when compared to the full single-bit track implementations; and the average number of single-bit tracks per channel is reduced by over 37%.

The direct benefit of reduction in single-bit tracks is in the reduction of routing switches that connect the multi-bit blocks to their routing tracks; and Table 3 shows the reduction in column 2, 3, and 4. Here column 2 lists the total number of switches (both for connecting the multi-block input and output signals) required for architectures containing both the single-bit and the multi-bit tracks; and column 3 lists the same number for architectures containing only single-bit tracks. The percentage reduction is then listed in column 4; and as shown, one can achieve an overall routing switch reduction of over 27% through the use of multi-bit routing tracks.

Reducing the number of single-bit tracks also reduces the number of SRAM bits required to configure the routing resources because of configuration memory sharing; and column 5, 6, and 7 of Table 3 show the SRAM reduction figures. As shown, architectures containing both the single-bit and the

Table 3. Switch Count and SRAM Bit Count

Circuit	Switches in Routing			SRAM Bits in Routing		
	S.B. + M.B.	S.B. Only	% Red.	S.B. + M.B.	S.B. Only	% Red.
code_seq_dp	18768	19688	4.7%	20700	21206	2.4%
dcu_dpath	47880	72072	34%	53739	68922	22%
ex_dpath	194304	308352	37%	178464	245872	27%
exponent_dp	34048	41984	19%	32128	37312	14%
icu_dpath	239568	380184	37%	222425	303147	27%
imdr_dpath	92664	117288	21%	95013	99225	4.2%
incmod	54264	62928	14%	55062	61845	11%
mantissa_dp	74240	99840	26%	66752	81152	18%
multmod_dp	127680	131880	3.2%	116550	121065	3.7%
pipe-dpath	20416	20416	0.0%	24969	25375	1.6%
prils_dp	21216	20800	-2.0%	22802	23738	3.9%
rsadd_dp	13608	15960	15%	17031	18648	8.7%
smu_dpath	28800	32256	11%	32184	35532	9.4%
ucode_dat	71048	99600	29%	72459	93209	22%
ucode_reg	3072	3552	14%	4356	4896	11%
Total	**1041576**	**1426800**	**27%**	**1014634**	**1241144**	**18%**

multi-bit tracks consume about 18% less SRAM bits than architectures containing only single-bit routing tracks.

The area savings that can be achieved through the reduction of routing switches and SRAM bits, however, are offset by the larger switch block size in architectures that use multi-bit routing tracks; and this increase in switch block size is primarily due to the increase in the total number of routing tracks. As shown in Table 2, the average track count per channel, including both the single-bit tracks and the multi-bit tracks, actually increases from the 56 tracks of the full single-bit track implementations to the 66 tracks of the combined single-bit and multi-bit implementations. This increase in track count also increases the number of routing switches required in each switch block (which should be differentiated from the routing switches that connect multi-bit blocks to the routing tracks), and consequently reduces the overall area savings.

Taking into consideration the increase in switch block size, Table 4 summarizes the actual area savings/increases for each benchmark circuit. Here the area is measured using the *equivalent minimum-width transistor area* as described in [13], where the total area required to implement an FPGA is normalized against the area that is required to implement a minimum-width transistor. As shown, ten of the fifteen circuits require less area to implement when using multi-bit routing tracks; and these circuits consist of 82% of the total area of all benchmark circuits. Finally the overall area reduction for these benchmark circuits is over 12%; and in particular, the two largest circuits (ex_dpath and icu_dpath) did particularly well, and achieve an area reduction of 22% and 18% each.

Table 5 shows that the use of multi-bit routing tracks has little impact on the overall delay of a circuit. In particular, column 2 of the table shows the critical path delay for architectures containing both single-bit and multi-bit routing tracks; and column 3 shows the delay for architectures containing

Table 4. Routing Area

Circuit	Routing Area (10e5)		
	Single-Bit + Multi-Bit	Single-Bit Only	% Reduction
code_seq_dp	2.69	2.66	-1.1%
dcu_dpath	7.24	8.81	18%
ex_dpath	26.9	34.6	22%
exponent_dp	5.04	5.40	6.7%
icu_dpath	33.5	40.7	18%
imdr_dpath	11.7	12.9	9.3%
incmod	7.23	7.27	0.55%
mantissa_dp	10.5	11.6	9.5%
multmod_dp	14.8	14.7	-0.68%
pipe_dpath	2.82	2.71	-4.1%
prils_dp	2.80	2.54	-10%
rsadd_dp	2.14	2.09	-2.4%
smu_dpath	4.02	4.16	3.4%
ucode_dat	10.2	11.6	12%
ucode_reg	0.536	0.565	5.1%
Total	**142**	**162**	**12%**

Table 5. Routing Delay

Circuit	Routing Delay (ns)		
	Single-Bit + Multi-Bit	Single-Bit Only	Change
code_seq_dp	6.47	6.51	-0.61%
dcu_dpath	6.67	9.83	-32%
ex_dpath	20.1	20.1	0.0%
exponent_dp	8.96	9.15	-2.1%
icu_dpath	12.0	11.6	+3.5%
imdr_dpath	19.1	18.2	+4.9%
incmod	19.1	19.6	-2.6%
mantissa_dp	8.14	7.86	+3.6%
multmod_dp	15.6	11.8	+32%
pipe_dpath	6.65	7.01	-5.1%
prils_dp	15.1	13.9	+8.6%
rsadd_dp	13.3	12.8	+3.9%
smu_dpath	9.92	10.5	-5.5%
ucode_dat	7.80	8.86	-12%
ucode_reg	1.99	1.96	+1.5%
Geo. Avg.	**9.92**	**10.0**	**-0.8%**

only single-bit routing tracks. Finally, the change in critical path delay is listed in column 4. Overall there is a slight reduction of 0.8% in critical path delay for all benchmark circuits. In the worst case, one circuit (multmod_dp) has a 32% increase in critical path delay; but another circuit (dcu_dpath), however, has a 32% reduction in critical path delay.

6. CONCLUSIONS

This paper examines the correlation between signal connectivity and signal positioning for multi-bit building blocks in FPGAs. It is shown that, for their bussed structures, multi-bit

block input and output signals with the same bit positions are much more likely to connect together; and based on the observation, a routing architecture is designed to use dedicated routing tracks to enrich the connectivity between these identical bit positions. Configuration-memory sharing is then used to reduce the amount of memory required to configure these tracks.

Using the proposed architecture and a specialized routing algorithm, it is empirically shown that the correlation can be directly translated into a 27% reduction in the number of routing switches, which connect the multi-bit blocks to their tracks. Furthermore, 18% less configuration memory bits are needed to control the routing resources; and one can achieve an overall area saving of 12%.

7. REFERENCES

[1] D. Chen and J. Rabaey, "A Reconfigurable Multiprocessor IC for Rapid Prototyping of Algorithmic-Specific High-Speed DSP Data Paths," *JSSC*, Dec. 1992, pp.1895–1904.

[2] A. Yeung and J. Rabaey, "A Reconfigurable Data Driven Multi-Processor Architecture for Rapid Prototyping of High Throughput DSP Algorithms," *HICCS*, Jan. 1993, pp.169–178.

[3] R. Bittner and P. Athanas, "Wormhole Run-time Reconfiguration," *FPGA*, Feb. 1997, pp.79–85.

[4] D. Cherepacha and D. Lewis, "DP-FPGA: An FPGA Architecture Optimized for Datapaths," *VLSI Design*, 1996, pp.329–343.

[5] C. Ebeling, et. al, "RaPiD — Reconfigurable Pipelined Datapath," *FPL*, Aug. 1996, pp. 237–241.

[6] J. Hauser and J. Wawrzynek, "Garp: A MIPS Processor with a Reconfigurable Coprocessor," *FCCM*, Apr. 1997, pp.24–33.

[7] E. Waingold, et. al, "Baring It All to Software: Raw Machines," *IEEE Computer*, Sep. 1997, pp.86–93.

[8] T. Miyamori and K. Olukotun, "A Quantitative Analysis of Reconfigurable Coprocessors for Multimedia Applications," *FCCM*, Apr. 1998, pp.2–11.

[9] A. Marshall, et. al, "A reconfigurable arithmetic array for multimedia applications," *FPGA*, Feb. 1999, pp.135–143.

[10] A. Alsolaim, et. al, "Architecture and Application of a Dynamically Reconfigurable Hardware Array for Future Mobile Communication Systems," *FCCM*, Apr. 2000, pp.205–214.

[11] S. Goldstein, et. al, "PipeRench: a reconfigurable architecture and compiler," *IEEE Computer*, Apr. 2000, pp.70–77.

[12] K. Leijten-Nowak and J. van Meerbergen, "An FPGA architecture with enhanced datapath functionality," *FPGA*, Feb. 2003, pp.195–204.

[13] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*, Feb. 1999, Kluwer Academic Publishers.

[14] *Altera Documentation Library*, Sep. 2004, Altera Corporation.

[15] *Xilinx Data Sheets*, 2004, Xilinx Inc.

[16] A. Ye, J. Rose, and D. Lewis, "Synthesizing Datapath Circuits for FPGAs with Emphasis on Area Minimization," *FPT*, Dec. 2002, pp.219–227.

[17] A. Ye and J. Rose, "Using Multi-Bit Logic Blocks and Automated Packing to Improve Field-Programmable Gate Array Density for Implementing Datapath Circuits," *FPT*, Dec. 2004, pp.129–136.

[18] A. Ye, "Field-Programmable Gate Array Architectures and Algorithms Optimized for Implementing Datapath Circuits," *Ph.D. Thesis*, Jun. 2004, University of Toronto (http://www.eecg.toronto.edu/~jayar/pubs/theses/Ye/AndyYe.pdf).

[19] C. Lee, "An Algorithm for Path Connections and its Applications," *IRE Trans. on Electronics Computing*, Vol. 10, 1961, pp.346–365.

[20] C. Ebeling, L. McMurchie, S. Hauck, and S. Burns, "Placement and Routing Tools for the Triptych FPGA," *Trans. on VLSI*, Dec. 1995, pp.473–482.

[21] J. Swartz, V. Betz and J. Rose, "A Fast Routability-Driven Router for FPGAs," *FPGA*, Feb. 1998, pp.140–149.

[22] P. Chan and M. Schlag, "New Parallelization and Convergence Results for NC: A Negotiation-Based FPGA Router," *FPGA*, Feb. 2000, pp.165–174.

[23] A. Ye, J. Rose, and D. Lewis, "Architecture of Datapath-Oriented Coarse-Grain Logic and Routing for FPGAs," *CICC*, Sep. 2003, pp.61–64.

[24] A. Ye and J. Rose, "Using Bus-Based Connections to Improve Field-Programmable Gate Array Density for Implementing Datapath Circuits," *FPGA*, Feb. 2005, To Appear.

[25] *Pico-Java Processor Design Documentation*, Sun Microsystems, 1999.

[26] H. Hseih, *et al.*, "Third-Generation Architecture Boosts Speed and Density of Field-Programmable Gate Arrays," *CICC*, Mar. 1990, pp.31.2.1–31.2.7.