

# The Effect of Multi-bit Based Connections on the Area Efficiency of FPGAs Utilizing Unidirectional Routing Resources

Omesh Mutukuda, Andy Ye, Gul Khan

Department of Electrical and Computer Engineering, Ryerson University  
350 Victoria Street, Toronto, Ontario, Canada M5B 2K3

omesh.mutukuda@ee.ryerson.ca

aye@ee.ryerson.ca

gnkhan@ee.ryerson.ca

**Abstract**— Field Programmable Gate Arrays (FPGAs) are increasingly being used to implement large datapath-oriented applications that are designed to process multiple-bit wide data. Studies have shown that the regularity of these multi-bit signals can be effectively exploited to reduce the implementation area of datapath circuits on FPGAs that employ the traditional bidirectional routing. Most of modern FPGAs, however, employ unidirectional routing tracks which are more area and delay efficient. No study has investigated the design of multi-bit routing architectures to effectively transport multiple-bit wide signals using unidirectional routing tracks. This paper presents such an investigation of architectures which employ multi-bit connections and unidirectional routing resources to exploit datapath regularity. It is experimentally shown that unidirectional multi-bit routing architectures are 8.6% more area efficient than the conventional routing architecture. This paper also determines the most area efficient proportion of multi-bit routing tracks.

## I. INTRODUCTION

Many of the applications implemented on Field Programmable Gate Arrays (FPGAs) are largely arithmetic based. These applications typically contain many datapath components that are designed to process multiple-bit wide data. The size and complexity of these applications demand FPGAs with large logic capacities and routing networks. The routing networks often overshadow the computing elements on the major performance metrics of area and delay. Specifically over 70% of the total FPGA area is often devoted to routing resources [1]. This motivates us to explore the architectural aspects of designing efficient FPGA routing resources for implementing large datapath-oriented applications on FPGAs.

The focus of this work is on FPGAs containing unidirectional routing resources - routing resources that employ directional wiring with single non-tristate drivers instead of traditional bidirectional wires. This work is relevant to current FPGA research since unidirectional wiring is used in many commercial FPGA architectures [2] [3]. Additionally, commercial FPGAs are being used to implement large datapath rich applications such as digital signal processing (DSP), computer vision, medical imaging and code breaking of cryptographic algorithms. Since datapath applications are designed to process multiple-bit wide data, circuits would

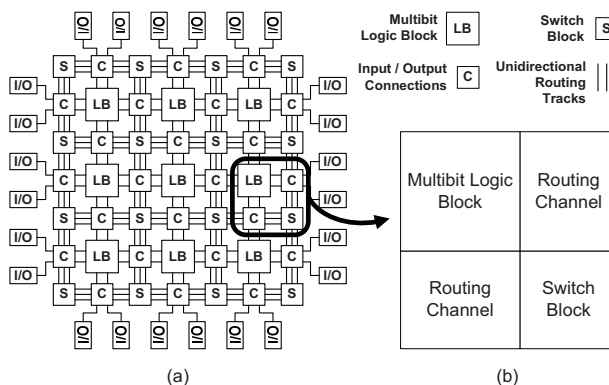


Fig. 1. (a) Island-style FPGA (b) FPGA tile

require specialized routing resources that can efficiently transport multiple-bit wide signals from one computing element to another.

Previous studies [4] - [8] have proposed various FPGA architectures which contain specialized computing elements designed to process multiple-bit wide data. None of the studies, however, have investigated the design of specialized routing resources that can effectively transport multiple-bit wide signals on the unidirectional routing architecture.

In order to investigate the effects of multi-bit signals with unidirectional routing, a set of datapath-rich benchmark circuits are implemented on multi-bit and non-multi-bit architectures. To this end, a Computer-Aided Design (CAD) tool employing placement and routing algorithms is used. Modifications are made to support multi-bit and unidirectional architectural features. Optimal architectural parameters, described later in this paper, are intelligently chosen to experimentally obtain area, delay and track segment results over a set of benchmark circuits. In order to preserve the regularity (amount of related signals travelling from a common source to a common destination) of the benchmark circuits, their netlists are generated using datapath-oriented synthesis [9] and packing [10] tools for use during the placement and routing operations in this paper.

TABLE I  
IMPACT OF ROUTING ON TOTAL FPGA AREA

W	A <sub>input</sub>	A <sub>sw.block</sub>	A <sub>routing</sub>	A <sub>FPGA</sub>	A <sub>routing</sub> / A <sub>FPGA</sub>
4	396	262	658	8,507.72	7.73%
8	873	524	1397	9,246.96	15.11%
12	1350	845	2195	10,044.64	21.85%
16	1587	1049	2635	10,485.44	25.13%
20	2064	1369	3433	11,283.12	30.43%
24	2301	1573	3874	11,723.92	33.04%
28	2538	1894	4432	12,281.61	36.08%
32	2775	2098	4872	12,722.40	38.30%
36	3252	2418	5670	13,520.09	41.94%
<b>40</b>	<b>3489</b>	<b>2622</b>	<b>6111</b>	<b>13,960.88</b>	<b>43.77%</b>
<b>48</b>	<b>3963</b>	<b>3146</b>	<b>7109</b>	<b>14,959.36</b>	<b>47.52%</b>
<b>52</b>	<b>4200</b>	<b>3467</b>	<b>7667</b>	<b>15,517.05</b>	<b>49.41%</b>
<b>64</b>	<b>4911</b>	<b>4195</b>	<b>9106</b>	<b>16,956.32</b>	<b>53.70%</b>
88	6574	5768	12342	17,754.01	55.78%
100	7285	6613	13898	20,191.77	61.12%
120	8470	7866	16336	21,747.94	63.90%

This paper is structured as follows. Section II introduces the multi-bit connections used in this study. Section III describes the multi-bit architecture and its implementation in detail. Section IV presents the experimental results, and Section V outlines the conclusions.

## II. MULTI-BIT BASED CONNECTIVITY

The multi-bit architecture considered in this study is based on the island style topology shown in Fig. 1(a) which is segmented into tiles as shown in Fig. 1(b). An FPGA tile consists of a multi-bit logic block, one vertical and one horizontal routing channel along with a switch block where the two routing channels intersect. Each multi-bit logic block contains  $M$  configurable logic blocks ([11] describes these configurable logic blocks in detail) where  $M$  is the granularity of the architecture [12]. Furthermore each multi-bit logic block is attached to  $I$   $M$ -bit wide input buses and  $N$   $M$ -bit wide output buses. The multi-bit logic blocks are interconnected through routing channels each containing  $W$  routing tracks. In this study the architectural parameters  $M=4$ ,  $I=10$  and  $N=4$  are used since previous work on bidirectional routing architectures show that this combination of values results in low area consumption [12]. Since this study employs unidirectional routing,  $W$  must always be an even number of tracks to accommodate signals travelling in both the forward and reverse directions.

Each routing track is comprised of a series of wire segments spanning a logical length of  $L$  multi-bit logic blocks. As shown in Fig. 2, connections are made from the end of each segment to the start of another segment along a channel using programmable routing switches. These routing switches consist of 1) a multiplexor to select both horizontal and

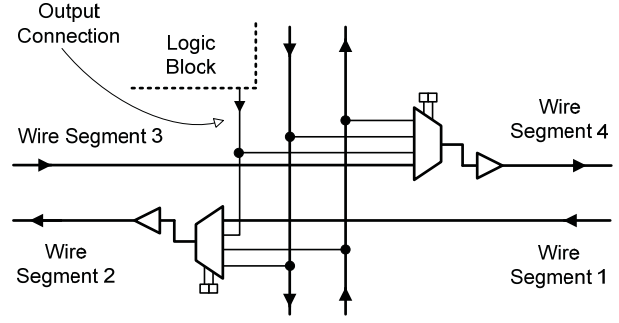


Fig. 2. Switch block connections for a horizontal routing channel

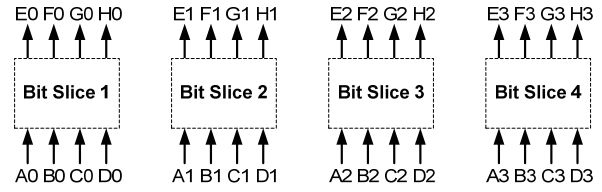


Fig. 3. Bit-slice partitioned datapath circuit

vertical wire segments, allowing signals to turn or extend further along the channel and 2) a buffer to drive signals along the respective wire segment [13]. Although not displayed in Fig. 2, it is important to note that similar connections would exist for the vertical tracks to form a complete switch block. The connection pattern of the routing switches inside a switch block is defined by its topology and in this study, the disjoint topology [14] is used as it is best suited for segmented architectures [11].

Logic block input and output pins connect the multi-bit logic block to adjacent routing channels using input connections and output connections. The fraction of routing tracks that connect to each input pin is defined as  $F_{ci}$  while the number of tracks connected to each input pin is  $\lceil F_{ci} \times \frac{W}{2} \rceil \times 2$ . Similarly,  $F_{co}$  represents the fraction of tracks each output pin drives. However, output connections can only be made to the routing switch multiplexors of wires that begin nearby (as shown in Fig. 2) due to design restrictions of unidirectional routing. This restriction combined with the staggered starting positions of the wires (discussed later in this section) results in  $\lceil F_{co} \times \frac{W}{2} \rceil \times 2$  connections to wires that begin in adjacent switch blocks (per output pin). The architecture described thus far shall for the remainder of this paper be known as the conventional routing architecture. In this work, the active area is measured in terms of minimum-width transistor area and the overall FPGA area consumed by logic and routing resources is calculated using the following formula:

$$\text{Area} = \sum_{\text{All Trans.}} \left( 0.5 + \frac{\text{Drive Strength of Current Trans.}}{2 \times \text{Drive Strength of Min. Width Trans.}} \right)$$

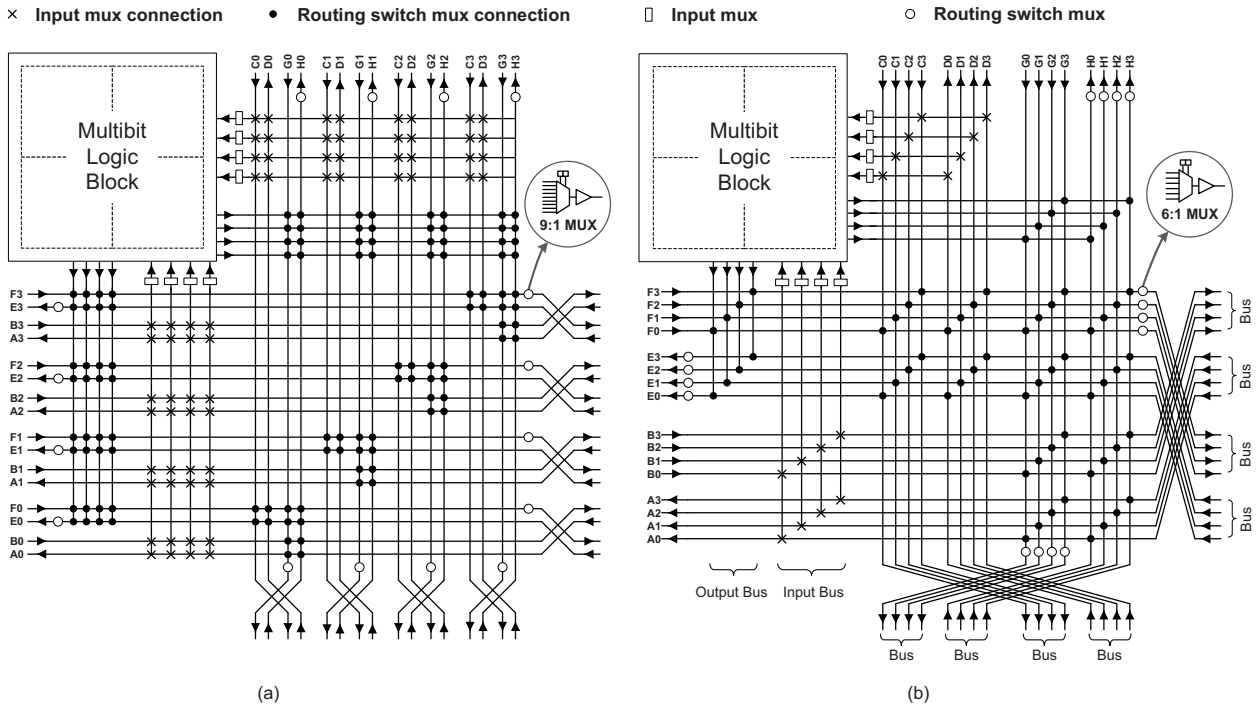


Fig. 4. Implementation of a bit-slice circuit on an FPGA tile using (a) conventional singular routing and (b) routing buses

Table I summarizes the area of routing resources (consisting of input and switch block connections) and total area of a conventional FPGA tile for increasing values of  $W$ . In these calculations  $M=4$ ,  $N=4$ ,  $I=10$ ,  $L=2$ ,  $F_{ci}=0.5$ ,  $F_{co}=0.25$  and a disjoint switch topology are used. The transistor and buffer sizes for all area calculations are obtained from the 90nm process specifications in [15]. The transistors inside logic blocks are sized according to the methods described in [11]. Columns 2 and 3 list the input and switch block area and column 4 lists the total routing area. Column 5 shows the total FPGA tile area including that of the multi-bit logic block and the final column shows the percentage of total FPGA area that the routing resources occupy. The values highlighted in bold are the results associated with typical channel widths for successful routing of circuits with the given parameters. It can be observed that every measurement of area listed in Table I increases as function of  $W$  and for typical channel widths, the programmable routing resources occupy over 40% of the total FPGA area. It is possible to alleviate some of this area by replacing conventional unidirectional tracks with multi-bit-oriented unidirectional routing buses that employ multi-bit signals from a common source to a common destination. It is important to note that observations similar to those from Table I were the motivation of employing multi-bit based connections to transport multi-bit signals on bidirectional routing architectures in [12].

In order to illustrate the advantages of multi-bit routing, consider mapping a datapath circuit onto an FPGA tile. The circuit is segmented into 4 bit-slices in which each bit-slice has 4 inputs and 4 outputs as shown in Fig. 3. Assuming the logic of each bit-slice can fit within a single configurable logic block, for  $M=4$ , a multibit logic block is used to house the 4 configurable logic blocks containing the entire datapath circuit. As shown in Fig. 4 (a), at a minimum, 16 bit wide routing channels are required to transport all the signals to and from the multi-bit logic block. Each of the white circles in Fig. 4 (a) represent a routing switch which includes an  $X:1$  multiplexer and a driving buffer.  $X$  represents the amount of multiplexer inputs determined by counting the total number of black circles on the associated track and the track itself. These routing switches are arranged according to the disjoint topology. The white squares represent  $Y:1$  input multiplexers where  $Y$  equals the number of connections between wires in a channel and a particular input pin marked by an 'x' in Fig. 4 (a). This example assumes 50% connectivity of input pins and full connectivity of the output pins wherever possible ( $F_{ci} = 0.5$  and  $F_{co} = 1$ ). According to Fig. 4 (a), there are 16 routing switches and 8 input switches. Each routing switch employs a 9:1 multiplexer while each input switch employs an 8:1 multiplexer.

Note that Fig. 4 (a) illustrates two essential details of a practical design using one common tile layout. The first being a staggered starting position of wires [13] leaves tracks labelled A0-A3, B0-B3, C0-C3 and D0-D3 (in this example)

TABLE II  
ACTIVE-AREA OF CONVENTIONAL AND BUS-BASED FPGA TILES

Conventional		Bus-based		$A_{BUS} / A_{BIT}$
W	$A_{BIT}$	$W_{BUS}$	$A_{BUS}$	
8	9,246.96	2	8,685.98	94%
16	10,485.44	4	9,155.50	87%
24	11,723.92	6	10,096.50	86%
32	12,722.40	8	10,425.64	82%
40	13,960.88	10	10,732.42	77%
48	14,959.36	12	11,025.08	74%
56	15,957.84	14	11,307.98	71%
64	16,956.32	16	11,583.74	68%
72	18,194.81	18	13,113.11	72%
80	19,193.29	20	13,379.11	70%
88	20,191.77	22	13,641.64	68%
96	21,190.25	24	13,901.29	66%
104	22,188.73	26	14,158.53	64%
112	23,187.21	28	14,413.72	62%
120	24,185.69	30	14,667.13	61%

without any routing switches since these wire segments do not start at this tile. This leads to the next detail of requiring track shifts between pairs of  $2L$  wires as shown at the bottom and right edges of Fig. 4 (a). The implementation of these track shifts and the staggering start positions of wires require only a single tile to be designed such that signals can still flexibly traverse the FPGA. This however requires the channel width to be a multiple of  $2L$  [13].

Alternatively Fig. 4 (b) illustrates an architecture which replaces all routing tracks of the previous example with 4 4-bit wide routing buses and groups the input and output pin connections into 4-bit wide input buses and output buses. Multi-bit based connection patterns are then used to connect the buses together. In particular, a bit in one bus can only be connected to a bit of the same position from another bus. The same multi-bit logic block of the previous example is used. Notice both designs require the same number of routing switches, input switches and routing tracks to implement the circuit. Fig. 4 (b) however requires smaller input and routing switch multiplexors, specifically of size 2:1 (75% reduction) and 6:1 (33% reduction) respectively. This reduction occurs due to a much sparser switch block and input connection pattern where wires of each input/output bus only connect to routing bus tracks of the same bit positions.

Table II lists the active area of a conventional unidirectional tile and that of a unidirectional bus-based tile for increasing values of  $W$  and  $W_{bus}$ , where  $W_{bus}$  is the number of  $M$ -bit wide routing buses. The area calculations use  $M=4$ ,  $N=4$ ,  $I=10$ ,  $L=2$ ,  $F_{ci}=0.5$ ,  $F_{co}=0.25$  and a disjoint switch topology. For the purpose of this analysis, these tiles are assumed to only model circuits containing  $M$ -bit wide interconnected datapath components. As shown in column 5, the use of buses to route datapath signals can reduce area by

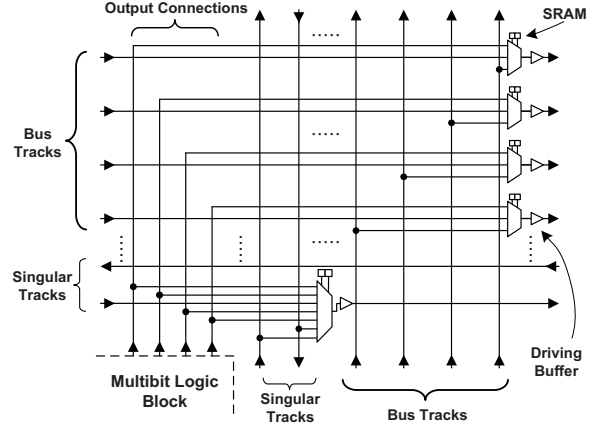


Fig. 5. Horizontal switches and multi-bit logic block output connections

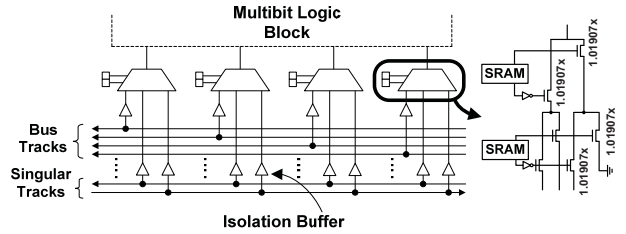


Fig. 6. Multi-bit logic block input connections

23% (for  $W=40$ ) and 30% (for  $W=80$ ). Larger area savings can be obtained for larger channel widths.

### III. MULTI-BIT ARCHITECTURE AND IMPLEMENTATION

As shown thus far, implementing ideal datapath circuits on a purely bus based routing architecture can significantly improve the area efficiency of FPGAs. However, practical circuits also contain irregular signals (single-bit wide signals or multi-bit wide signals that shift bit positions between their source logic blocks and their destination logic blocks). To accommodate these irregular signals, pairs of conventional routing tracks (using the same connection patterns as those used in the conventional unidirectional routing architecture) are used to augment the routing buses (multi-bit tracks) to form the multi-bit routing architecture [12]. For the remainder of this paper the conventional routing tracks will be referred to as singular tracks.

The multi-bit architecture is composed of multi-bit logic blocks interconnected by vertical and horizontal channels of routing tracks. As displayed in Fig. 5 and Fig. 6, the routing channels contain both singular tracks and  $M$ -bit wide buses of channel widths  $W_f$  and  $W_c$  respectively. Each multi-bit logic block contains  $M$  configurable logic blocks whose input and output connections connect directly to those of the multi-bit logic block. Each configurable logic block is composed of  $N$  basic logic elements (BLE) and share  $I$  inputs. A BLE as defined in [11] consists of one  $k$  input look-up-table and a D

TABLE III  
INTERNAL MULTI-BIT LOGIC BLOCK DELAYS

Delay Description	Delay (ns)
BLE output to CLB output pin	0
CLB input pin to BLE input	0.6077
BLE output to BLE input in the same CLB	0.05793
BLE input to BLE in combinational mode	0.2391
BLE input to storage component within BLE in sequential mode	0.2347
BLE storage component to BLE input in sequential mode	0.140

flip flop. Wire segments are connected together and to logic block output pins using multiplexors and driving buffers where SRAM components are used to control the select lines of the multiplexor [16] as shown in Fig. 5. Additional buffers (called isolation buffers and shown in Fig. 6) are added to isolate each track from the electrical effects of the input connections [12]. Finally, the input and output buses of the multi-bit logic block are distributed uniformly among its four sides. This is possible due to the logical equivalency between input pins and between output pins respectively.

#### A. Buffer Sizing and Delay Model

In order to generate realistic data on the behaviour of multi-bit connections on unidirectional routing, logic and routing components must be modelled based on a modern process technology. This study uses accurate area and timing estimates based on 90nm CMOS process estimates and optimized for FPGA architectures with  $N=4$ ,  $I=10$ ,  $L=2$ . The following area and delay information is extracted from [15] [17], whose transistor-level models are based on the Berkeley Predictive Technology Model (BPTM) [18]. As shown in Fig. 6, the input multiplexors are built as a tree of pass-transistors where each transistor is of size 1.01907 times that of a minimum-width transistor. Similarly routing switch multiplexors are built with pass-transistors of size 1.82646 times that of a min-width transistor while the driving buffer is designed as a three-stage buffer of size 12.324 min-width transistor area units. The delay of an input connection starting from the routing track through the isolation buffer and the multiplexor to the multi-bit logic block input pin is 0.07428ns. The routing switches consisting of a multiplexor and driving buffer) have an intrinsic delay of 0.07115ns. Table III lists the delays for paths through logic block components such as input pins, output pins and BLEs. Also listed are timing estimates for specific paths when the BLEs are in sequential or combinational states [15] [17].

#### B. Parameters

Overall, there are 12 variables used to parametrically describe the multi-bit architecture. These parameters can be categorized as follows: multi-bit logic block parameters, routing track dimensions and connection parameters.  $N$ ,  $I$ ,  $k$  and  $M$  as defined before describe the size of the multi-bit logic

block along with the number of BLEs and their size.  $L_f$ ,  $L_c$ ,  $W_f$  and  $W_c$  describe the dimensions of the routing tracks and channels. Finally  $F_{cif}$ ,  $F_{cic}$ ,  $F_{cof}$ ,  $F_{coc}$  and  $T_s$  define the input and switch block connectivity of the routing tracks. Notice that each of these parameters have been described in the previous section, however, separate parameters have been allocated for components relating to singular routing tracks (subscripted  $f$ ) and routing-bus tracks (subscripted  $c$ ).  $T_s$  describes the number of routing switches, their connections and their topological arrangement within the switch block.

The combination of these parameters generates an extremely large design space requiring exploration that is beyond the scope of this study. Therefore, most of these parameters are set to values determined to be optimal from previous architectural FPGA studies. Internal logic block parameters  $N$  and  $I$  are set to 4 and 10 respectively as [19] has shown these to be efficient for bidirectional non-multi-bit based FPGAs. Additionally, the value of  $k$  is set to 4 since [20] and [21] have shown a size 4 LUT yields a minimum in total routing area. The granularity  $M$  is set to 4 since it has been empirically shown to yield the most area efficient results by [12].  $T_s$ , for the multi-bit architecture, is the disjoint switch block topology as it is ideal for segmented architectures [11] and widely used.  $F_{cif} = F_{cic} = .5$ ,  $F_{cof} = F_{coc} = .25$ ,  $L_c = 2$  and  $L_f = 2$  are used in this work. The studies done in [11] and [12] find these independent variables result in efficient area results for both singular-bidirectional and multi-bit-bidirectional architectures.  $W_f$  and  $W_c$  are the dependent variables of this study.

## IV. RESULTS

To empirically evaluate the effect of multi-bit connection patterns on the area efficiency of unidirectional routing architectures, 15 benchmark circuits [12] consisting of datapath components from Sun Microsystems' Pico-Java processor [22] are implemented. Each of these circuits are synthesized and mapped onto multi-bit logic blocks using datapath-oriented synthesis and packing tools based on [9] [10] and [23]. These tools are essential in preserving the regularity of the interconnected datapath components. The modified simulated annealing placement algorithm of the place-and-route toolset VPR (described in [24]) serves to physically arrange the packed logic blocks on a 2-dimensional grid. In this work, the VPR router is modified to connect the placed logic blocks together using a combination of unidirectional buses and singular tracks. These modifications involve changes to the routing resource graph within VPR along with area and timing updates to the architectural file. This is done with an emphasis on optimizing delay and minimizing area by reducing routing demand. The multi-bit routing tool differs significantly from the conventional routing tool due to grouped wavefront expansion methods in their routing resource graphs and new congestion costs [25]. Additionally the router forces the multi-bit signals and singular signals to be routed on their respective resource types unless one type of resource is highly congested.

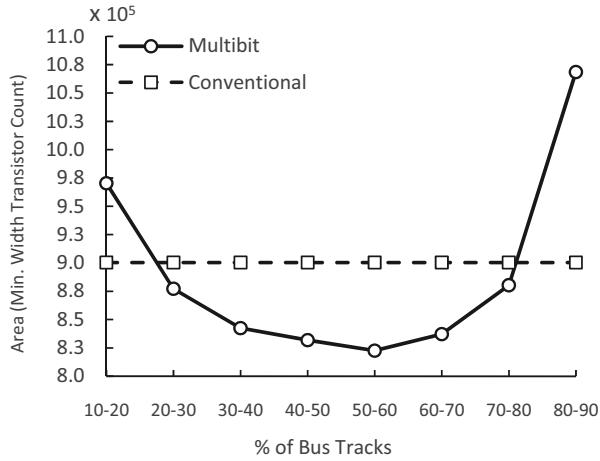


Fig. 7. Average area as a function of the percentage of routing bus tracks

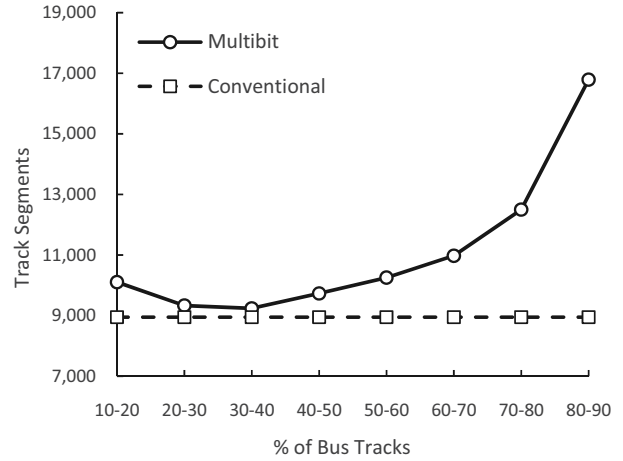


Fig. 8. Average number of track segments as a function of the percentage of routing bus tracks

TABLE IV  
ROUTING AREA ORGANIZED BY PERCENTAGE OF MULTI-BIT SIGNALS

% Range of Multibit Signals	Benchmark Circuit	% of Multibit Signals	Routing Area (min. width transistor count)	
			Multibit	Conventional
10-20%	multmod_dp	18.71	912194	946289
	Average		<b>912194</b>	946289
20-30%	prils_dp	29.14	214759	233488
	code_seq_dp	29.37	315211	298276
	Average		<b>264985</b>	265882
30-40%	exponent_dp	31.36	238851	274516
	incmod	37.41	431668	465967
	pipe_dpath	39.42	217886	255391
	Average		<b>296135</b>	331958
40-50%	smu_dpath	41.08	332538	353157
	imdr_dpath	42.88	571594	628280
	mantissa_dp	44.11	7.81E+05	8.56E+05
	icu_dpath	48.60	1590000	1910000
	ucode_dat	48.93	952775	1140000
	Average		<b>845655.2</b>	977410
50-60%	ex_dpath	50.25	4.01E+06	4.88E+06
	rsadd_dp	51.06	147503	1.67E+05
	dcu_dpath	54.01	871663	1010000
	Average		<b>1.68E+06</b>	2.02E+06
60+%	ucode_reg	65.64	6.88E+04	8.33E+04
	Average		<b>6.88E+04</b>	8.33E+04

In this investigation, the benchmark circuits are implemented on both conventional and multi-bit architectures to compare their performance. In order to fairly assess the area results, the same routing tool (the multi-bit routing algorithm in [24] [25]) is used for every experiment, eliminating any

effects arising due to routing algorithm variations. Analysis of both implementations is achieved by constraining the routing bus channel width  $W_c$  and then attempting to successfully route the circuit with a minimum number of singular tracks  $W_f$  using the binary search algorithm of the router. The conventional implementation involves constraining  $W_c$  to zero, thereby forcing the router to use only singular unidirectional tracks. The multi-bit architecture is evaluated by routing the benchmark circuits over a range of  $W_c$  values. Each benchmark circuit is routed with fixed values of  $W_c$  starting with  $2M$  bus tracks (8 tracks in this experiment) and incremented by  $2M$  tracks to an upper limit of 120 tracks (30 buses). These circuit implementations are then sorted according to percentile ranges representing the proportion of routing bus tracks as a function of total routing tracks in a routing channel. The results with minimum area for each of the 15 benchmark circuits are chosen and arithmetically averaged for each percentile range. Similarly the minimum amount of total track segments are determined for each circuit implementation and arithmetically averaged for each percentile range. Finally the best critical-path delays of the multi-bit implementation are determined for each circuit and compared against the conventional implementations.

#### A. Effect of Routing Buses on Area

Fig. 7 is a graph of the average area consumed by 15 benchmark circuits over a range of 8 percentile ranges. The solid curve represents the average area for 15 benchmark circuit implementations which fall in the listed percentile ranges. The dashed line represents the average conventional implementation area of the 15 benchmark circuits. The percentile range (0 - 10%) is not present in the plot since there are only a few circuit designs in the benchmark set utilizing this range of routing bus tracks. When 10% - 30% of the tracks in a channel are routing bus tracks, an increase in area is observed. This occurs due to the inability of input pins to

TABLE V  
CRITICAL PATH DELAYS OF ROUTED BENCHMARK CIRCUITS

Benchmark Circuit	Critical Path Delay (s)	
	Conventional	Multi-bit
code_seq_dp	6.05E-09	5.36E-09
dcu_dpath	3.38E-09	2.76E-09
ex_dpath	1.64E-08	1.62E-08
exponent_dp	8.20E-09	8.00E-09
icu_dpath	1.17E-08	1.17E-08
imdr_dpath	1.55E-08	1.54E-08
incmod	1.59E-08	1.48E-08
mantissa_dp	4.35E-09	3.90E-09
multmod_dp	1.29E-08	1.20E-08
pipe_dpath	6.09E-09	5.85E-09
prils_dp	9.78E-09	7.08E-09
rsadd_dp	1.34E-08	1.32E-08
smu_dpath	1.28E-08	1.27E-08
ucode_dat	3.58E-09	3.12E-09
ucode_reg	1.42E-09	1.42E-09
<b>Geometric Mean</b>	<b>7.78E-09</b>	<b>7.21E-09</b>

connect such few routing buses at the current value of  $F_{cic}$ . The generated number of routing buses remains unused while the router attempts to reroute these signals using singular routing tracks, resulting in an increase routing area. A similar observation is made for the bidirectional multi-bit architecture in [12]. The 50% - 60% range of routing bus tracks achieves the greatest area efficiency with an 8.6% routing area reduction over the conventional architecture. As the percentage of bus tracks increase past 70%, the number of constrained bus tracks will exceed the amount actually required by each circuit by a factor of 2 (recall unidirectional routing requires an even number of tracks/buses). The router uses these excess bus tracks to route singular signals, resulting in drastically higher area consumption.

Table IV displays the best implementation area results of each benchmark circuit for both multi-bit and conventional implementations. The results are then categorized into percentile ranges based on the regularity of each circuit. Column 3 lists this regularity (in ascending order) as the percentage of total signals in each circuit that are grouped into 4-bit wide multi-bit buses. Arithmetic averages are computed and displayed for each percentile range and implementation type. As shown, almost all benchmark circuits routed on the multi-bit architecture are more area efficient than those routed on the conventional architecture for every proportion of multi-bit signals listed in Table IV. Additionally, it can be seen that larger circuits containing a higher proportion of datapath circuits tend to realize larger area savings.

#### B. Delay and Track Segment Results

Fig. 8 plots the number of wire segments utilized per circuit, averaged over 15 benchmark circuits as a function of the percentage of routing bus tracks. It is observed that the best

multi-bit architecture (with 50% to 60% multi-bit tracks) employs 14.6% more track segments over the conventional architecture.

Table V summarizes the critical path delays of the entire set of benchmark circuits implemented on the most area efficient conventional and multi-bit architectures. The geometric mean is calculated for each type of architectural implementation and is displayed. As shown by the geo-mean values, the multi-bit architectures perform slightly better than the conventional architecture. Additionally, almost all of the multi-bit circuit implementations show a slight performance increase.

#### V. CONCLUSION

This study has explored the effect on FPGA area efficiency of multi-bit connections using unidirectional routing in order to efficiently implement arithmetic intensive circuits. Initially a simple theoretical datapath circuit is mapped onto conventional and bus-only architectures where the total area results of each are compared. From these results the estimates and limits on area efficiency by using routing buses are found. In order to accommodate the usage of non-ideal signals in modern circuits, pairs of singular signals are added to the routing buses to form the multi-bit routing architecture. The actual effectiveness of this multi-bit architecture is determined by comparing the implementation area of 15 benchmark circuits mapped on multi-bit and conventional architectures. It is found that the best architecture consists of 50% to 60% routing bus tracks with an average routing area reduction of 8.6% over the best conventional architecture.

#### REFERENCES

- [1] J. Rose, A. El Gamal, and A. Sangiovanni-Vincentelli, "Architecture of Field-Programmable Gate Arrays," *Proc. IEEE*, vol. 81, pp. 1013-1029, Jul. 1993.
- [2] D. Lewis *et al.*, "The Stratix II logic and routing architecture," in *Proc. ACM Int. Symp. Field-Programmable Gate Arrays*, 2005, pp. 14-20.
- [3] "Xilinx Data Sheets," Xilinx Inc., San Jose, CA, 2004 [Online]. Available: <http://www.xilinx.com>
- [4] K. Leijten-Nowak and J. van Meerbergen, "An FPGA architecture with enhanced datapath functionality," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2003, pp. 195-204.
- [5] C. Ebeling, D.C. Cronquist, and P. Franklin, "RaPiD - Reconfigurable Pipelined Datapath," in *Proc. Int. Workshop Field-Programmable Logic Appl.*, 1996, pp. 126-135.
- [6] D.C. Chen and J.M. Rabaey, "A reconfigurable multiprocessor IC for rapid prototyping of algorithmic-specific high-speed DSP data paths," *IEEE J. of Solid-State Circuits*, vol. 27, pp. 1895-1904, Dec. 1992.
- [7] A. Marshall *et al.*, "A reconfigurable arithmetic array for multimedia applications," in *Proc. ACM/SIGDA Int. Symp. Field Programmable Gate Arrays*, 1999, pp. 135-143.
- [8] D. Lewis and D. Cherepacha, "DP-FPGA: An FPGA Architecture Optimized for Datapaths," *J. VLSI Des.*, vol. 4, pp. 329-343, 1996.
- [9] A. Ye, J. Rose, and D. Lewis, "Synthesizing datapath circuits for FPGAs with emphasis on area minimization," in *Proc. Int. Conf. Field-Programmable Tech.*, 2002, pp. 219 - 226.
- [10] A. Ye and J. Rose, "Using multi-bit logic blocks and automated packing to improve field-programmable gate array density for implementing datapath circuits," in *Proc. Int. Conf. Field-Programmable Tech.*, 2004, pp. 129-136.
- [11] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep Submicron FPGAs*. Norwell, MA: Kluwer, 1999.

- [12] A. Ye and J. Rose, "Using bus-based connections to improve field-programmable gate-array density for implementing datapath circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, pp. 462-473, May 2006.
- [13] G. Lemieux, E. Lee, M. Tom, and A. Yu, "Directional and single-driver wires in FPGA interconnect," in *Proc. Int. Conf. Field-Programmable Tech.*, 2004, pp. 41-48.
- [14] H. Hsieh *et al.*, "Third-generation architecture boosts speed and density of field-programmable gate arrays," in *Proc. IEEE Custom Integrated Circuits Conf.*, 1990, pp. 31.2/1 - 31.2/7.
- [15] I. Kuon and J. Rose. (2008, February). *iFAR – intelligent FPGA Architecture Repository* [Online]. Available: <http://www.eecg.utoronto.ca/vpr/architectures/>
- [16] G. Lemieux and D. Lewis, "Circuit design of routing switches," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2002, pp. 19-28.
- [17] I. Kuon and J. Rose, "Area and delay trade-offs in the circuit and architecture design of FPGAs," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2008, pp. 149-158.
- [18] W. Zhao and Y. Cao, "New Generation of Predictive Technology Model for Sub-45nm Design Exploration," *IEEE Trans. on Electron Devices*, vol. 53, pp. 585-590, Nov. 2006.
- [19] V. Betz and J. Rose, "How much logic should go in an FPGA logic block," *IEEE Des. Test of Comput. Mag.*, vol. 15, pp. 10 -15, Jan. 1998.
- [20] J. Rose, R.J. Francis, D. Lewis, and P. Chow, "Architecture of field-programmable gate arrays: the effect of logic block functionality on area efficiency," *IEEE J. Solid-State Circuits*, vol. 25, pp. 1217-1225, Oct. 1990.
- [21] E. Ahmed and J. Rose, "The Effect of LUT and cluster size on deep-submicron FPGA performance and density," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, pp. 288 - 298, Mar. 2004.
- [22] Pico-Java Processor Design Documentation, Sun Microsystems, Santa Clara, CA, 1999.
- [23] J. Luu *et al.*, "VPR 5.0: FPGA cad and architecture exploration tools with single-driver routing, heterogeneity and process scaling," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2009, pp. 133-142.
- [24] A. Ye, "Field-Programmable Gate Array Architecture and Algorithms Optimized for Implementing Datapath Circuits," Ph.D. dissertation, Univ. of Toronto, Toronto, ON, Canada, 2004.
- [25] A. Ye and J. Rose, "Measuring and utilizing the correlation between signal connectivity and signal positioning for FPGAs containing multi-bit building blocks," in *Proc. Int. Con. Field Programmable Logic and Applications*, 2005, pp. 159-166.