A Hexagon-Based Honeycomb Routing Architecture for FPGA

Kaichuang Shi, Hao Zhou, Lingli Wang*
State Key Laboratory of ASIC and System
Fudan University, Shanghai, China
*llwang@fudan.edu.cn

***Abstract*—Field Programmable Gate Arrays (FPGAs) are widely used for their flexibility and short time to market. FPGA routing architecture design is the key problem due to the fact that it plays a dominant role in the area, delay and power. Most of modern FPGAs are island-style which provide abundant vertical and horizontal tracks to guarantee the circuit designs can be routed successfully. Most connections in placed netlists are diagonal which may lead to passing through extra turning switches, resulting in increased delay cost and high routing density. In this paper, we propose a hexagon-based honeycomb FPGA routing architecture to improve the routability and performance. In honeycomb architecture, there are three kinds of routing channels which can provide more freedom to decrease the turning switches on the routing paths. In addition, the router lookahead algorithm is enhanced to support the honeycomb architecture which is then evaluated by the enhanced VTR with provided benchmarks. The experimental results show that the honeycomb architecture can improve the minimum routing channel width by 7.7% compared with traditional rectangular architecture with length-1 wires. In addition, the honeycomb architecture can achieve 9.9% improvement on the routed wirelength, 11.5% on the critical path delay and 12.4% on the area-delay product.**

I. Introduction

Compared to application-specific integrated circuits (ASICs), FPGAs have superiority in flexibility, time to market and non-recurring engineering cost. However, circuit implementations in FPGA often need more area, delay and power than ASIC circuits [1]. Studies show that the routing architecture contributes to most of the FPGA's area, delay and power [2]. So how to optimize the routing architecture is the key problem to improve the FPGA performance.

The FPGA routing architecture includes the routing wire segments and the programmable switches. In academic researches, the routing architecture is mainly based on the connection blocks (CBs) and switch blocks (SBs). CBs provide programmable switches to interface wire segments and logic block (LB) pins while SBs are used to connect different wire segments. Traditional rectangular routing architecture includes vertical and horizontal routing channel tracks where the diagonal connections in circuits need to pass through extra programmable switches in SBs. To alleviate the problem, many researches have been carried out in routing architecture designs. In this paper, we propose a novel routing architecture to improve the routability and performance of FPGAs. Our contributions include:

- We propose a hexagon-based honeycomb FPGA routing architecture. In the honeycomb FPGA architecture, the basic tile is hexagonal while the basic tile is rectangular in the traditional island-style FPGA architecture. Each tile is surrounded by three kinds of channels.
- We define four parameters to model the honeycomb architecture. In order to evaluate the performance of the honeycomb architecture, we enhance the Routing Resource Graph (RRG) generator and the router lookahead algorithm in the latest VTR 8 [3]. Besides, a depopulation strategy for the hexagonal SBs is implemented to improve the area efficiency of the honeycomb architecture.
- We evaluate the performance of the honeycomb architecture whose area and delay parameters are extracted from COFFE 2 [4] with VTR benchmarks [5]. Experimental results show that the honeycomb architecture can improve the minimum routing channel width by 7.7% compared with classical rectangular architecture with length-1 wires. In addition, it can achieve 9.9% improvement on the routed wirelength, 11.5% on the critical path delay and 12.4% on the area-delay product.

The rest of this paper is organized as follows. Section II introduces the traditional FPGA routing architecture and the related work. Section III gives the proposed hexagon-based honeycomb FPGA routing architecture. Section IV discusses the enhancement of routing algorithm in VTR 8. Section V presents the experimental methodology and results compared with traditional island-style rectangular architecture. Section VI concludes this paper with future work.

II. Background and related work

A. Routing Architecture of Island-Style FPGAs

Modern island-style FPGA is composed of an array of rectangular tiles which are interconnected by vertical and horizontal routing channels as shown in Fig. 1. Each tile consists of an LB, two CBs and an SB. The routing channel width is described by W and the wire length is presented by L. Input and output CB flexibility, $F_{c,in}$ & $F_{c,out}$ mean the fraction of wire segments that an LB input and output pin can connect to respectively. The SB flexibility F_s is the number of other wires to which an incoming wire can connect inside the SBs.

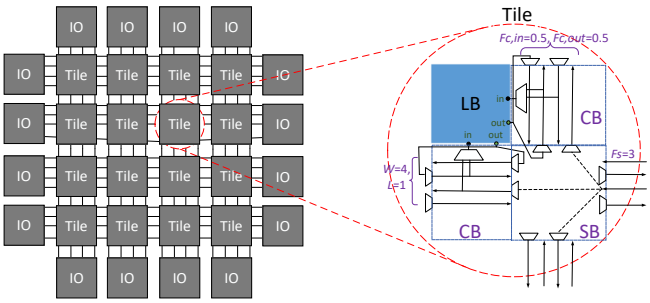

Fig. 1. Island-style FPGA architecture.

B. Related Work

In the exploration of non-rectangular architecture, S. Chaudhuri et al. [6] do a purely mathematical analysis about the possibility of using hexagonal and octagonal tiling patterns replacing Manhattan grid in FPGAs, which shows that the average interconnect length can be well improved at the cost of increased area. In [7], S. Chaudhuri discusses two new diagonal routing tracks in addition to the vertical and horizontal tracks and presents the performance evaluation. Diagonal interconnect architecture has also been used in ASIC and can achieve great delay improvement [8] [9]. The honeycomb architecture has been used in many applications. In [10], the honeycomb mesh is considered as a multiprocessor interconnection network. In [11], A. Thomas et al. present a new multi-grained dynamically reconfigurable honeycomb architecture which targets to improve array utilization, reusability and post-compile flexibility. Reference [12] proposes a honeycomb Network-on-Chip (NoC) architecture and results show it achieves great improvement in power, area and delay than the mesh topology. In this paper, we propose a hexagon-based honeycomb routing architecture for FPGA.

III. Honeycomb Architecture

In this section, we propose a hexagon-based honeycomb FPGA routing architecture. Besides, we enhance VTR 8 to evaluate the honeycomb architecture.

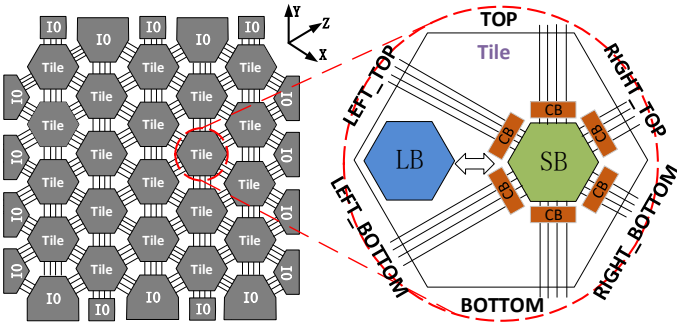

Fig. 2. The proposed honeycomb architecture.

A. Honeycomb Architecture

Fig. 2 shows the honeycomb architecture we propose, in which the basic tile is hexagonal while the basic tile is rectangular in the classical island-style FPGA architecture. Each tile which is composed of an LB, an SB and six CBs is surrounded by routing channels from six sides and has six neighboring tiles. LB pins can connect to the routing channel tracks from the corresponding side through programmable switches in CBs. The routing channel tracks can connect with each other through programmable switches in SBs which is similar to the traditional island-style FPGA architecture. The difference is that each SB in the honeycomb architecture has six sides. Besides, there are three kinds of routing channels in the honeycomb architecture which are called X channel, Y channel and Z channel respectively as Fig. 2 shows.

Fig. 3 shows one routing path example to compare the traditional architecture with the honeycomb architecture. Assuming that only length-1 wires are available, eight wire segments are needed to connect LB *A* with LB *B* in the traditional rectangular architecture. In the honeycomb architecture, it only needs six wire segments to connect LB *A* with LB *B* which can save two wire segments. Besides, the longer the net is, the more wire segments can be saved. Hence, the honeycomb architecture can possibly reduce the number of wire segments (with driving MUXes) on the critical path. In addition, the honeycomb architecture can release routing congestion due to the less wire segments.

Fig. 4 shows a comparison between the routing resources in the honeycomb architecture and the traditional rectangular architecture. Hops mean the number of wire segments needed to set up a connection between two LBs [13]. Evidence shows that the honeycomb architecture with length-1 wires can connect to 1.5 times as many LBs as the traditional rectangular architecture with length-1 wires can connect to through the same hops.

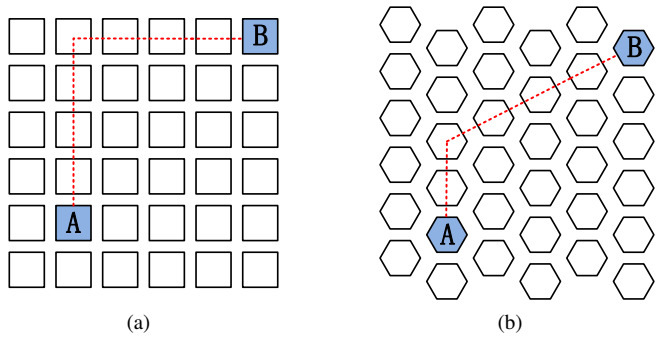

Fig. 3. A routing path example to compare the (a) traditional rectangular architecture with (b) honeycomb architecture.

B. Modelling Parameters

To model the honeycomb architecture, four parameters are defined. We define *PinLocation* to assign the location of each LB pin optionally according to our needs. The optional locations contain {RIGHT_TOP, RIGHT_BOTTOM, BOTTOM,

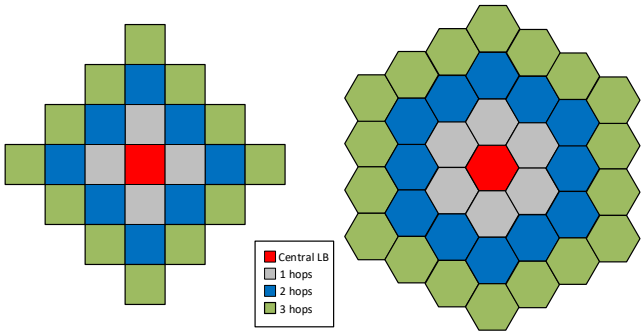

Fig. 4. The hops distribution comparison in the (a) traditional rectangular architecture and (b) honeycomb architecture.

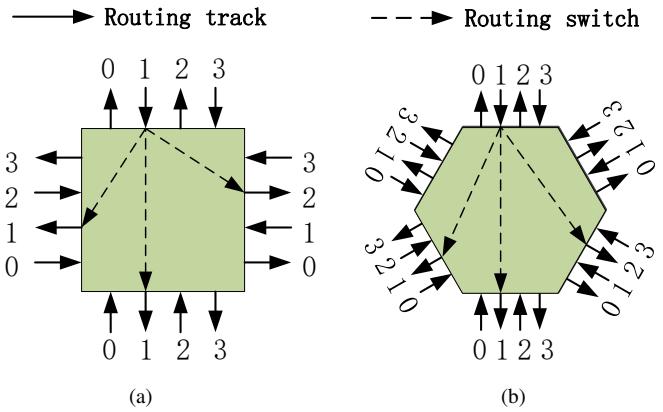

Fig. 5. The Wilton SB pattern in the (a) traditional rectangular architecture and (b) honeycomb architecture.

LEFT_BOTTOM, LEFT_TOP, TOP} which correspond to six sides of the hexagonal LB respectively as Fig. 2 shows. Similar to the traditional rectangular architecture, we define F_c (includes $F_{c,in}$ and $F_{c,out}$) and F_s to describe the connections between LB pins and wire segments and the connections between different wire segments respectively. $F_{c,in}$ and $F_{c,out}$ define the fractions of wire segments that an LB input pin and an LB output pin can connect to respectively. F_s defines the number of wire segments that an incoming wire can connect inside an SB. The specific connections in CBs follow a round-robin scheme to increase the track diversity.

C. SB Enhancement

Traditional SB has four sides. Fig. 5 (a) shows the Wilton SB pattern in traditional rectangular architecture. Each wire segment can connect to three wire segments on the other three sides. For the honeycomb routing architecture, each LB is surrounded by routing channels from six sides, which leads to each SB has six sides. Evidence shows that F_s = 5 in the universal hexagonal SBs will bring extra area cost [14]. So a depopulation strategy is implemented to reduce the number of programmable switches and the routing area. Fig. 5 (b) shows the depopulated hexagonal Wilton SB pattern. One incoming wire connects to three wires located in the three opposite sides. There is no connections between the incoming wire and the wires in its two neighboring sides. The depopulation scheme is eliminating the connections which will not be used in the shortest path. The connections between wires in adjacent sides with angles less than 90-degree lead to backward connections which are wasteful. Therefore, we have depopulated the SB pattern and set F_s = 3. The specific wire-to-wire connections follow the traditional SB pattern with four sides.

D. Enhanced RRG Generator

FPGA routing architecture is modelled by the routing resource graph in which LB pins and wire segments are represented by routing nodes and the programmable inter-connections between different nodes are denoted by edges. To support the honeycomb architecture, we rewrite the RRG generator in VTR which can generate three kinds of routing channels as we describe in Section III-A. The starting points of the wire segments are stagger to enhance the FPGA routability. In addition, the RRG generator can automatically generate the routing nodes and edges according to the parameters described in Section III-B and Section III-C.

E. Area and Delay Modelling

We use the area and delay models in VTR 8 to estimate the whole FPGA area and the critical path delay. VTR measures the area in *minimum-width transistor areas* (MWTAs) [15] and VTR uses Elmore delay model to estimate the delay. The default delay and area models are still suitable for the hon-eycomb architecture. However, the resistance and capacitance per unit length in terms of LBs of the wire segments need to be amended. The area of LB is calculated by summing the areas of all the transistors. The relationships between the area S and the side length l of a square can be seen in equation (1) and the relationship between the area $S^{'}$ and the height $l^{'}$ of a regular hexagon can be seen in equation (2). Assuming the LB areas are the same in the traditional rectangular architecture and honeycomb architecture ($S = S^{'}$), we obtain an expression as shown in (3) after combining equations (1)(2). Therefore, the resistance and capacitance per unit length in honeycomb architecture are 1.07 times of that in traditional rectangular architecture. Most of the wire delay is mainly concentrated on the driving MUX [2], so the increase in delay is negligible. The area and delay parameters which are required in the VTR architecture file are all extracted from COFFE 2 [4] at the 22nm technology node. COFFE 2 is a fully automated transistor sizing tool for FPGAs which measures the area and delay by relying on HSPICE simulation.

$$S = l^2 \tag{1}$$

$$S^{'} = 6 \times \frac{l^{'}}{\sqrt{3}} \times \frac{l^{'}}{2} \times \frac{1}{2} = \frac{\sqrt{3}}{2} l^{'2} \tag{2}$$

$$l^{'} = 1.07l \tag{3}$$

F. Layout Strategy

In this section, we want to discuss the layout strategy of the honeycomb architecture. The 60-degree lines can only be achieved physically by non-standard processes and there are still some challenges about the deployment of 60-degree wiring [9]. So another layout can be proposed as shown in Fig. 6 (b) except the layout described in Section III-A. If we move the leftmost and rightmost vertices of a regular hexagon inward, the regular hexagon becomes a rectangle with an aspect ratio of 2 as Fig. 6 (a) shows. The 45-degree metal lines can be achieved easily [8], so the honeycomb architecture can also be realized physically through this layout strategy. In the future, we will also explore the fabrication effect of the honeycomb architecture.

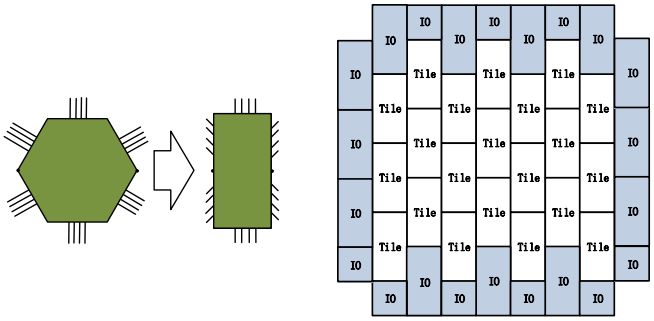

Fig. 6. The layout strategy of the honeycomb architecture. (a) The hexagon becomes a rectangle with an aspect ratio of 2. (b) The possible layout.

IV. Algorithm Enhancement

VTR 8 uses a routing algorithm [3] which is based on the Pathfinder negotiated congestion algorithm. The router lookahead is used to estimate the remaining delay from an intermediate node to the target node by calculating the numbers of segments needed to reach the target node. We enhance the router lookahead to support the honeycomb architecture. First, we introduce how to calculate the point-to-point shortest distance between two LBs in the honeycomb architecture. In honeycomb architecture, we firstly use a novel coordinate system for ease of calculating the distance as Fig. 7 shows. The coordinate of the new coordinate system is represented by (x, y) while (x', y') for original VTR coordinate system. VTR uses Cartesian coordinate system and the coordinate of the bottom left block is $(0, 0)$. The two coordinate systems can convert to each other as equations (4)(5) shows, where $\lfloor \cdot \rfloor$ represents the *flooring* function. In this new coordinate system, the coordinates in the same Y channel have the same x coordinate and the coordinates in the same X channel have the same y coordinate. The coordinates in the same Z channel have the same difference in x and y coordinates.

Assuming that there are two LBs and their coordinates are (x_1, y_1) and (x_2, y_2) respectively. We use D to represent the shortest distance between the two LBs. If the two LBs are located in the same Y channel, then $D = |y_2 - y_1|$. In contrast, if the two LBs are located in the different Y channels, there are three cases to discuss. Assuming that $x_2 > x_1$, we define $\Delta x = x_2 - x_1$, $\Delta y = y_2 - y_1$. D can be calculated by (6). The case of $x_2 < x_1$ is similar. Put all cases together, D can be calculated by (7). The expected cost of two LBs is the number of the segments needed multiply by the delay of passing through one wire segment. Therefore, assuming the wire length is 1, the number of segments needed is the distance of the two LBs which can be calculated by (7). Similar, if the wire length is longer than 1, the number of wire segments needed can be calculated by (8). In this equation, $\lceil \cdot \rceil$ represents the *ceiling* function and L is the wire length.

$$x = x' \tag{4}$$

$$y = \begin{cases} y' & x < 3 \\ y' + \lfloor (x' - 1)/2 \rfloor & x \geq 3 \end{cases} \tag{5}$$

$$D = \begin{cases} \Delta x - \Delta y & \Delta y < 0 \\ \Delta x & 0 \leq \Delta y \leq \Delta x \\ \Delta y & \Delta y > \Delta x \end{cases} \tag{6}$$

$$D = \begin{cases} \max(|\Delta x|, |\Delta y|) & \Delta x \cdot \Delta y \geq 0 \\ |\Delta x| + |\Delta y| & \Delta x \cdot \Delta y < 0 \end{cases} \tag{7}$$

$$n = \begin{cases} \left\lceil \frac{||\Delta x| - |\Delta y||}{L} \right\rceil + \left\lceil \frac{\min(|\Delta x|, |\Delta y|)}{L} \right\rceil & \Delta x \cdot \Delta y \geq 0 \\ \left\lceil \frac{|\Delta x|}{L} \right\rceil + \left\lceil \frac{|\Delta y|}{L} \right\rceil & \Delta x \cdot \Delta y < 0 \end{cases} \tag{8}$$

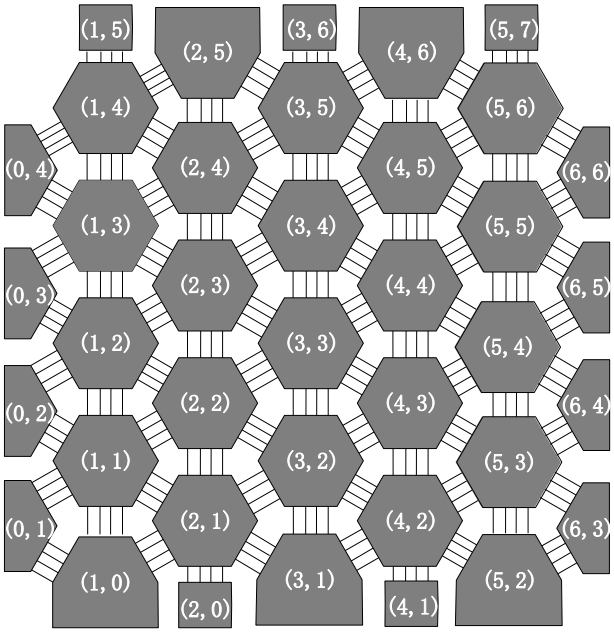

Fig. 7. The new coordinate system for the honeycomb architecture.

V. Experimental Results

In this section, we describe the experimental methodology and the baseline architecture. Then, we use the enhanced VTR along with the provided benchmark set to compare the traditional rectangular architecture and the honeycomb architecture in the routability, area and delay.

A. Experimental Methodology

In this paper, we use an island-style rectangular FPGA architecture as the baseline architecture. Its delay and area parameters that are used in VTR architecture files are extracted from COFFE 2 at the 22nm technology node. Each LB is composed of ten BLEs and each BLE can operate as a 6-input LUT or two 5-input LUTs. DSP blocks are 36*36 fractuable multipliers and memories are configurable 32K block RAMs. CB flexibility is set to $F_{c,in}$ = 0.1 & $F_{c,out}$ = 0.1 while SB flexibility is set to F_s = 3. The SB pattern is Wilton. In the following section, we evaluate FPGA routability with W_{min}, the minimum routing channel width of each circuit. Then, we evaluate the area and delay at a constant routing channel width of W = 300 which is reasonable in prior work [16].

B. Routability Comparison

In the traditional rectangular architecture, there are two kinds of routing channels (vertical and horizontal channels) while there are three kinds of routing channels in the honeycomb architecture. The routing channel width in traditional rectangular architecture is 1.5 times of that in honeycomb architecture under the same total routing channel tracks. So, we compare 1.5 times of W_{min} in the honeycomb architecture with the W_{min} in the traditional rectangular architecture to evaluate the routability fairly. In addition, we set $F_{c,in}$ = 0.15 & $F_{c,out}$ = 0.15 in honeycomb architecture to ensure the LB pins can connect to the same number of routing channel tracks as that in traditional rectangular architecture under the same total routing channel tracks. The routing resources are all length-1 wires. The experimental results are shown in TABLE I. *HC* means the honeycomb architecture while *REC* means the traditional rectangular architecture. Experimental results show that the honeycomb architecture can improve the minimum routing channel width by 7.7% compare to the traditional rectangular architecture.

TABLE I
Comparison of minimum routing channel width between the honeycomb architecture and the rectangular architecture

Circuit	W_{min}		
	HC	REC	Ratio
arm_core	64	108	88.9%
bgm	38	64	89.1%
blob_merge	38	60	95.0%
boundtop	16	24	100.0%
ch_intrinsics	16	26	92.3%
diffeq1	22	44	75.0%
diffeq2	22	34	97.1%
LU8PEEng	52	90	86.7%
LU32PEEng	80	120	100.0%
mcml	52	78	100.0%
mkDelayWorker32B	16	20	120.0%
mkPktMerge	20	26	115.4%
mkSMAdapter4B	30	48	93.8%
or1200	38	70	81.4%
raygentop	22	42	78.6%
sha	32	52	92.3%
stereovision0	26	46	84.8%
stereovision1	34	70	72.9%
stereovision2	56	92	91.3%
stereovision3	16	26	92.3%
Av. Improvement		**7.7%**	

[a] Ratio is the results of $1.5W_{min}$ in the honeycomb architecture divided by W_{min} in rectangular architecture.

C. Area and Delay Comparison

In this section, we compare the honeycomb architecture with traditional rectangular architecture in the area and delay. The routing channel width is set to 200 in the honeycomb architecture to keep the total number of routing channel tracks same as that in the baseline architecture. In addition, we set $F_{c,in}$ = 0.15 & $F_{c,out}$ = 0.15 in the honeycomb architecture to ensure the LB pins can connect to the same number of routing channel tracks as that in the traditional rectangular architecture. The routing architecture consists of only length-1 wires. Experimental results show that the honeycomb architecture can improve the routed wirelength by 9.9% and the critical path delay by 11.5% with 1% area reduction compare to the traditional rectangular architecture as TABLE II shows. The small reduction in area is due to the depopulation of SBs. The SBs in the perimeter of FPGA have less programmable switches. For example, the SBs in the leftmost and rightmost columns have no programmable switches and only half SBs in the top and bottom rows have programmable switches.

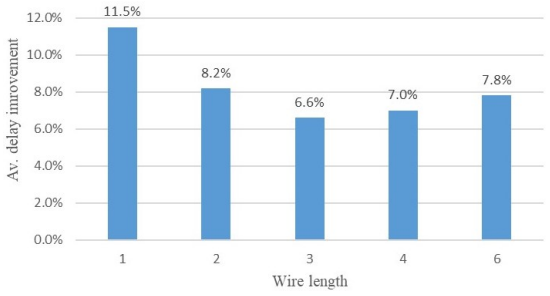

Fig. 8. The delay improvement of the honeycomb architecture with different wire lengths.

D. Effect of Wire Length

To explore the effect of wire length on the performance improvement of the honeycomb architecture, we compare the honeycomb architecture with the traditional rectangular architecture with different wire segments. Five kinds of wire segments with length-{1,2,3,4,6} are used. As Fig. 8 shows, experimental results show that the longer wire segments have less improvement on critical path delay than length-1 wire. As the wire delay mainly concentrate on the driving MUX, less wire segments can be saved on the critical path which leads to

TABLE II
PERFORMANCE OF THE HONEYCOMB ARCHITECTURE COMPARED WITH TRADITIONAL RECTANGULAR ARCHITECTURE

Circuit	Routed Wirelength			Total Area (10^6)			Critical Path Delay (ns)			Area-Delay Product (10^6)		
	HC	REC	Ratio	HC	REC	Ratio	HC	REC	Ratio	HC	REC	Ratio
arm_core	139072	154558	90.0%	82.67	83.10	99.5%	17.87	21.11	84.7%	1477.12	1753.88	84.2%
bgm	180943	193787	93.4%	136.57	137.15	99.6%	16.48	18.23	90.4%	2250.74	2500.12	90.0%
blob_merge	43967	47704	92.2%	39.17	39.45	99.3%	7.26	9.19	79.0%	284.24	362.57	78.4%
boundtop	602	591	101.9%	5.54	5.65	98.2%	1.67	2.12	78.5%	9.25	12.00	77.1%
ch_intrinsics	509	613	83.1%	4.60	4.69	98.2%	2.35	2.27	103.3%	10.81	10.65	101.5%
diffeq1	4365	4982	87.6%	6.86	6.97	98.4%	13.26	14.17	93.6%	90.88	98.71	92.1%
diffeq2	3165	3674	86.1%	6.86	6.97	98.4%	10.69	11.72	91.2%	73.26	81.68	89.7%
LU8PEEng	203701	218796	93.1%	131.02	131.58	99.6%	68.53	79.64	86.1%	8979.41	10478.49	85.7%
LU32PEEng	913439	1012027	90.3%	457.67	458.84	99.7%	66.51	81.30	81.8%	30440.53	37302.41	81.6%
mcml	402294	435223	92.4%	378.99	380.01	99.7%	56.18	65.84	85.3%	21291.85	25020.39	85.1%
mkDelayWorker32B	12295	11894	103.4%	108.27	108.78	99.5%	9.69	10.15	95.5%	1049.50	1104.06	95.1%
mkPktMerge	7808	7757	100.7%	31.61	31.86	99.2%	4.93	5.31	92.8%	155.76	169.10	92.1%
mkSMAdapter4B	7829	8177	95.7%	15.26	15.43	98.9%	5.43	5.51	98.6%	82.94	85.03	97.5%
or1200	28188	31937	88.3%	29.45	29.69	99.2%	12.18	12.92	94.3%	358.78	383.62	93.5%
raygentop	9731	12416	78.4%	18.79	18.98	99.0%	3.70	4.38	84.4%	69.46	83.17	83.5%
sha	12681	14026	90.4%	13.53	13.69	98.8%	9.74	12.09	80.5%	131.70	165.53	79.6%
stereovision0	33432	37729	88.6%	48.13	48.45	99.3%	2.84	3.06	92.6%	136.46	148.43	91.9%
stereovision1	65505	83256	78.7%	60.91	61.28	99.4%	5.29	5.90	89.7%	322.50	361.74	89.2%
stereovision2	371172	437600	84.8%	330.49	331.45	99.7%	15.12	18.70	80.9%	4997.21	6196.53	80.6%
stereovision3	284	342	82.9%	1.11	1.15	96.6%	1.89	2.19	86.1%	2.11	2.53	83.2%
Av. Improvement		**9.9%**			**1.0%**			**11.5%**			**12.4%**	

less improvement on the delay in the honeycomb architecture with the longer wire segments.

VI. DISCUSSION AND CONCLUSION

In this paper, we propose a hexagon-based honeycomb routing architecture. The RRG generator and routing algorithm in VTR is enhanced to support the honeycomb architecture. Experimental results show that the honeycomb architecture can improve the minimum routing channel width by 7.7% compared with rectangular architecture with length-1 wires. Besides, the honeycomb architecture can achieve 9.9% improvement on the routed wirelength and 11.5% on the critical path delay and 12.4% on the area-delay product.

In the future, there are still many researches to be carried out about the honeycomb architecture. The placement and routing algorithms need to be enhanced to support the honeycomb architecture better. Besides, we will focus on the routing channel segmentation with different lengths and SB pattern design on the honeycomb architecture and try to find a near-optimal routing architecture.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under grant 61971143.

REFERENCES

[1] I. Kuon and J. Rose, "Measuring the Gap Between FPGAs and ASICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 203–215, 2007.

[2] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for deep-submicron FPGAs*. Kluwer Academic Publishers, 1999.

[3] K. E. Murray *et al.*, "VTR 8: High Performance CAD and Customizable FPGA Architecture Modelling," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 13, no. 2, pp. 1–55, 2020.

[4] S. Yazdanshenas and V. Betz, "COFFE 2: Automatic modelling and optimization of complex and heterogeneous FPGA architectures," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 12, no. 1, pp. 1–27, 2019.

[5] J. Luu *et al.*, "VTR 7.0: Next generation architecture and CAD system for FPGAs," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 7, no. 2, pp. 1–30, 2014.

[6] S. Chaudhuri, S. Guilley, P. Hoogvorst, and J.-L. Danger, "Efficient tiling patterns for reconfigurable gate arrays," in *Proceedings of the 2008 international workshop on System level interconnect prediction*, pp. 11–18, 2008.

[7] S. Chaudhuri, "Diagonal tracks in FPGAs: a performance evaluation," in *Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays*, pp. 245–248, 2009.

[8] M. Igarashi *et al.*, "A diagonal interconnect architecture and its application to RISC core design," in *2002 IEEE International Solid-State Circuits Conference. Digest of Technical Papers (Cat. No.02CH37315)*, vol. 2, pp. 166–167, 2002.

[9] H. Chen, C.-K. Cheng, A. Kahng, I. Mandoiu, Q. Wang, and B. Yao, "The Y architecture for on-chip interconnect: analysis and methodology," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 4, pp. 588–599, 2005.

[10] I. Stojmenovic, "Honeycomb networks: Topological properties and communication algorithms," *IEEE Transactions on parallel and distributed systems*, vol. 8, no. 10, pp. 1036–1042, 1997.

[11] A. Thomas, M. Rückauer, and J. Becker, "HoneyComb: A multi-grained dynamically reconfigurable runtime adaptive hardware architecture," in *2011 IEEE International SOC Conference*, pp. 335–340, IEEE, 2011.

[12] A. Yin, N. Chen, P. Liljeberg, and H. Tenhunen, "Comparison of mesh and honeycomb network-on-chip architectures," in *2012 7th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, pp. 1716–1720, 2012.

[13] P. B. Minev and V. S. Kukenska, "The Virtex-5 routing and logic architecture," *Annual Journal of Electronics, Technical University of Sofia*, vol. 3, pp. 107–110, 2009.

[14] G.-M. Wu, M. Shyu, and Y.-W. Chang, "Universal switch blocks for three-dimensional FPGA design," *IEE Proceedings-Circuits, Devices and Systems*, vol. 151, no. 1, pp. 49–57, 2004.

[15] C. Chiasson, *Optimization and modeling of FPGA circuitry in advanced process technology*. PhD thesis, University of Toronto, 2013.

[16] K. Shi, H. Zhou, X. Zhou, and L. Wang, "GIB: A Novel Unidirectional Interconnection Architecture for FPGA," in *2020 International Conference on Field-Programmable Technology (ICFPT)*, pp. 174–181, 2020.