Routing Optimization For Hybrid FPGAs

Chi Wai Yu[1], Wayne Luk[2], Steven J.E. Wilton[3], Philip H.W. Leong[4]

[1,2] *Dept of Computing, Imperial College London, London England*
[1,2] cyu,wl@doc.ic.ac.uk

[3] *Dept of Electrical and Computer Engineering, University of British Columbia, Vancouver, B.C., Canada*
[3] stevew@ece.ubc.ca

[4] *Dept of Computer Science and Engineering, Chinese University of Hong Kong, Hong Kong*
[4] phwl@cse.cuhk.edu.hk

***Abstract*—This paper optimizes the routing structure for hybrid FPGAs, in which high I/O density coarse-grained units are embedded within fine-grained logic. This significantly increases the routing resource requirement between elements. We investigate the routing demand for hybrid FPGAs over a set of domain-specific applications. The trade-off in delay, area and routability of the separation distance between coarse-grained blocks are studied. The effects of adding routing switches to the coarse-grained blocks and using wider channels near them to meet extra routing demand are examined. Our optimized architectures are compared to existing column based architecture. The results show that (1) there is 44% tracks usage at the edge of the embedded blocks, (2) both the separation of embedded blocks and addition of switches to embedded blocks can increase the area and delay performance by 48.4% compared to column based FPGA architecture, (3) wider channel width reduces the area of highly congested system by 34.9%, but it cannot further improve the system with separation of embedded blocks and additional switches on embedded blocks.**

I. Introduction

Although general-purpose FPGAs are suitable for many applications, there are cases where they do not provide the required speed, density, or power consumption. Hybrid FPGAs are similar to generic FPGAs, but contain application-specific embedded blocks to improve the efficiency of computations within a given application domain. Unlike general-purpose FPGAs which contain relatively simple embedded blocks (DSPs and memories), the embedded blocks in hybrid FPGAs can be large and complex. Ho et al. [1] propose a domain-specific hybrid FPGA architecture for computationally expensive floating point (FP) applications to achieve 18 times area reduction. The floating point unit (FPU) in this architecture is specifically optimized for FP addition and multiplication.

It has been shown that the presence of a large embedded block affects the routing demand within the fine-grained logic. Altera removes the large MegaRAM Blocks in Stratix-III device since these large blocks create a disruption for the routing fabric [2]. This suggests that the extra routing demands imposed by the large embedded block should be accommodated in the design of the routing architecture of the fine-grained logic. Existing commercial devices such as Xilinx Virtex 5 arrange the small embedded blocks like memory and DSP in columns. This arrangement may not be efficient for large blocks.

Programmable routing between logic and I/O pads in traditional fine-grained island FPGA consumes about 70% of the area in a die and contributes significantly to delay [3]. Betz et al. [4] examine the best routing track distribution on FPGA, but this may not be suitable for hybrid FPGA.

In this paper, we examine the routing structure of hybrid FPGA with large embedded coarse-grained blocks. Specifically, the key contributions of this paper are:

- we show experimentally that the presence of large embedded blocks affect the routability,
- we propose three routing optimizations: finding optimized separation distance between EBs, adding routing switches on the top of the EBs, and inserting tracks near the EBs to meet the extra routing demand,
- we evaluate architectures with the proposed optimizations, showing that they improve 48.4% of the area-delay product over the column based architecture.

To facilitate comparison, we make use of a hybrid FPGA with large embedded blocks for floating point computation [1]; our techniques can be generalized to cover other FPGAs with other large embedded blocks.

II. Baseline Architecture

A. Fine/Coarse-grained Architecture

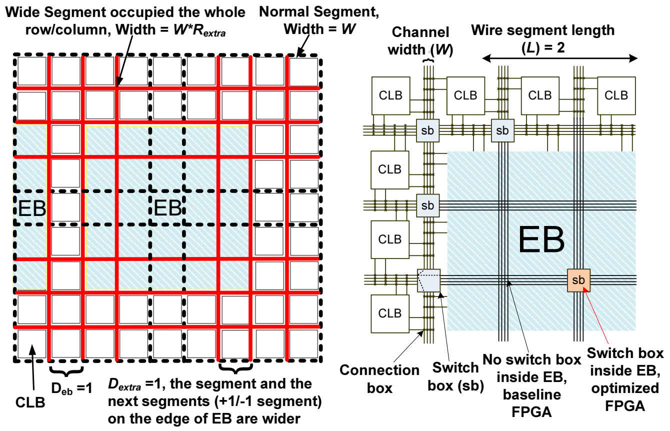
In the hybrid FPGA, coarse-grained EBs are surrounded by fine-grained CLBs and they are connected by horizontal and vertical wire channels as shown in Figure 1a. We assume the fine-grained element is a configurable logic block (CLB) similar to a Virtex II slice. The CLB is a cluster of *2* basic logic elements (BLEs) containing *4*-LUT, flip flop (FF), fast carry chains, internal multiplexers and XOR gates. We adopt an enhanced version of double precision floating point units (FPUs) [1]. There are two floating point adders/subtracters, two floating point multipliers and five word blocks in each FPU, which are connected by bus based wires. A word block consists of 64 identical bitblock with LUT and FF. The area model of the FPU in [1] does not include the routing tracks for fine-grained routing in FPGA. The routing wires consist of over 70% of total FPGA area [3] for large channel widths. Therefore, we add 70% extra area to the FPU for the vertical, horizontal routing tracks and switches. The resulting area of each FPU is 214 tiles. Each tile consists of a CLB, its

978-1-4244-4377-2/09/$25.00 © 2009 IEEE

associated interconnects, buffers and configuration bits. From the study in [5], the best interface between tiles should follow the following rules: (1) FPUs should be close to square, (2) FPUs should be positioned in the centre of the FPGA, (3) the FPU pins should be evenly distributed on four sides of the FPU.

B. Routing Architecture

Figure 1b shows our assumed routing architecture. CLBs and EBs are connected to W parallel routing tracks of segment length L using connection boxes. We adopt $L = 4$ which gives the best area-delay product [4]. W is constant for this baseline architecture, however we will introduce heterogeneous channel width near EBs to meet the routing demand found in Section IV. Segment channels are intersected by a switch box. There are no switch boxes inside EBs, so changes in wire direction are not allowed. The area and delay model of the wire is based on PTM 0.13μm, 1.3V CMOS process [6]. We estimate the routing area in our architecture by using the model in [4]. We count the area of connection multiplexers from logic blocks to tracks and tri-state buffers in routing switches in term of minimum-width transistors per CLB tile.


Wide Segment occupied the whole row/column, Width = $W*R_{extra}$
Normal Segment, Width = W
EB
EB
CLB
D_{eb} =1
D_{extra} =1, the segment and the next segments (+1/-1 segment) on the edge of EB are wider
Channel width (W)
Wire segment length (L) = 2
CLB
sb
EB
Connection box
Switch box (sb)
No switch box inside EB, baseline FPGA
Switch box inside EB, optimized FPGA

(a) Hybrid FPGA routing architecture

(b) Detailed routing architecture in hybrid FPGA

Fig. 1. The hybrid FPGA architecture with optimization parameters.

III. Experimental Methodology

A. Floating Point Benchmarks

To explore the routing architecture of a hybrid FPGA, we use nine double precision floating point applications as benchmarks. They are: (a) *dscg*, digital sine-cosine generators (8 FPUs), (b) *bfly*, Fast Fourier Transform circuits (8 FPUs), (c) *fir4*, 4-tap finite impulse response filters (8 FPUs), (d) *ode*, ordinary differential equation solvers (8 FPUs), (e) *mm3*, 3x3 matrix multipliers (8 FPUs), (f) *bgm*, a Monte Carlo simulations of interest rate model derivatives circuit (7 FPUs), (g) *nbody*, a force pipeline for an N-body solver (5 FPUs), (h) *syn2* (3 FPUs) and (i) *syn7* (16 FPUs). The last two are synthetic floating point benchmarks generated by a synthetic benchmark generator. Floating point hybrid FPGAs can implement these circuits more efficiently than fine-grained FPGA, since the embedded FPUs are able to do most of the floating point computation.

B. Evaluation Tool

We analyze the wirelength, timing and area of our routing architecture using a place and route tool for hybrid FPGA called VPH [7]. VPH is a modified version of the VPR tool. It supports embedded blocks, memories, multipliers, carry chains and user constraints. The positions of EBs, additional switches on EBs and extra tracks around EBs can be specified in the user constraints.

IV. Routing Demand

Commercial devices [2] embed smaller blocks such as DSP and memory, which are normally less than 10 CLB tiles in column based arrangement. The column based architecture may not be efficient for large embedded coarse-grained blocks, with over 100 CLB tiles. Therefore, we compare the performance of the fine-grained FPGA, column based FPGA, and the baseline architecture for large EBs.

A. Netlength Demand

The average netlength is important since longer netlength requires more routing resources for a net. We examine the netlength and wirelength of fine-grained FPGA, column based and baseline hybrid FPGAs with FPUs as embedded blocks. The EBs in the baseline architecture are closely packed together, with D_{eb}=0. The aspect ratio of EBs in column based FPGA is 2, and are evenly distributed.

On average, the embedded FPUs in the baseline FPGA increase the netlength of a net by 1.4 times compared to the fine-grained FPGA. In a fine-grained FPGA, all the user logic is implemented in CLBs, which are more flexible to move closer to reduce net delay. In a hybrid FPGA, most of the computation logic and nets are in the fast FPUs, which reduces the CLB usage and number of nets. Therefore, the embedded FPU improves the area and delay. The remaining small amount of nets in the hybrid FPGA is used to connect EBs and CLBs. However, the large EBs with longer perimeter are not flexible to move. They require long length routing nets to connect to other elements.

The column based FPGA is 20.8% slower and has 18.2% longer netlength than the baseline FPGA. The floating point adders and multipliers are normally connected in series. The tall FPU and the long distance between FPU columns cause the longer nets and delay between EBs.

B. Congested Region

Next, we investigate the most congested region in both baseline and column based hybrid FPGA. We examine the wire segments at the edge and one segment next to the edge of EB in the baseline FPGA. On average, tracks in these segments are only 17.82% of the total tracks in the FPGA, but about 44% of the tracks are used for routing, making it the most congested region.

In a particular example of *bgm*, Figure 2 and Figure 3 show the track usage along X and Y channel of *bgm* in the baseline FPGA and the column based FPGA respectively. The peak track usage in both systems is at the edge of EBs. This congestion is caused by the large amount of net connecting from EB to CLBs or another EB. Many CLBs move to the EBs to reduce the net delay. Therefore, the wire density at this area is very high. As shown in the figures, column based FPGA spreads the tracks better than the baseline one. The difference is because of the EB columns are evenly distributed in the column based system. There are enough space to place CLBs around, so the density of connection in this area is reduced and spread to another region.

Although the speed of the baseline FPGA is faster than the column based FPGA, more routing tracks are used, which lead to 50.4% decrease in area-delay product. In order to retain the speed advantage of the baseline FPGA, we propose three routing optimization schemes to reduce its routing area.

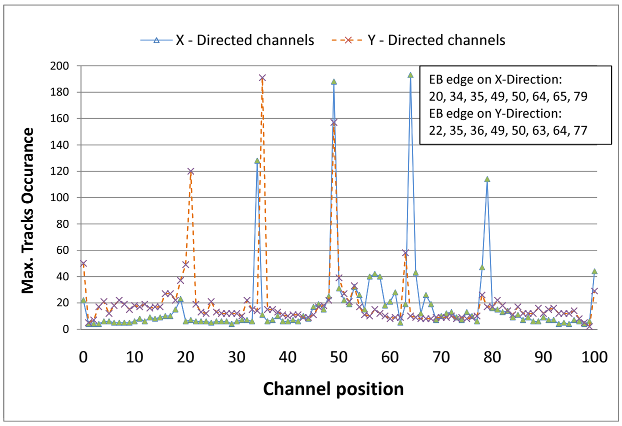


Fig. 2. Track usage along X-Y channels of *bgm* in the baseline FPGA (100x100 CLBs)

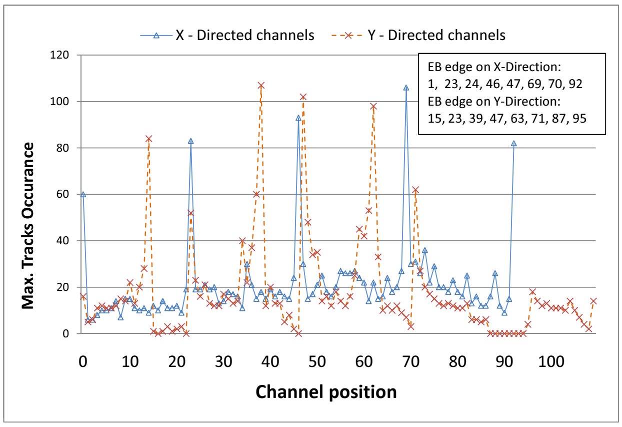


Fig. 3. Track usage along X-Y channels of *bgm* in the column based FPGA (93x110 CLBs)

V. Optimization of Routing

A. Separation Distance between EBs

The distance between EBs can be varied as shown in Figure 1a. Larger D_{eb} can reduce the routing stress to use less routing channel like column based FPGA, but the trade-off is having longer net delay. We examine the trade-off of different D_{eb} in delay and area in the baseline architecture without switches in EB.

We place and route the benchmarks in different D_{eb} by using 20% tracks more than minimum channel to avoid congestion. We study the routing area-delay product. to determine which D_{eb} is the most optimized for both area and delay. The result in Figure 4 (dot line) shows D_{eb}=4 is the best combination, which is at least 6% better than others. Since the routing stress around EBs is minimum when the EBs are 4 CLBs away of each other, which results in less routing resources are used. The improvement in routing area is more significant than loss in speed.

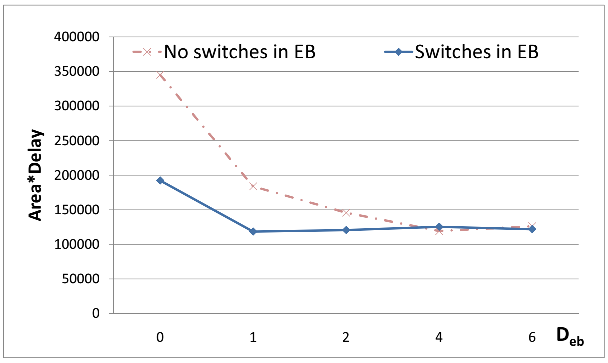


Fig. 4. Average area*delay at different D_{eb}

B. Additional Routing Switches in EBs

In this section, we show that if we extend the routing grid over the embedded block by adding switches within the embedded block, routability can be improved. Figure 1b shows the additional switches which allow the direction of signals inside EBs to be changed.

1) Area overhead of additional switches: The routing wires and transistors of switch boxes are on different metal layers in the FPGA. The area of routing wires and switch boxes increase by different amount when the channel width W increases. Schmit and Chandra [8] found that 100% of the area underneath the switch box is occupied by switch point transistors when W<49 in a 0.35μm process. Otherwise, the switch box area is mainly occupied by the routing wires. The wirebound of the routing area is W=49. We adopt this result to estimate the area trade-off when adding switch boxes in an EB. We estimate the wirebound area is W=33. We account for the area overhead of the additional switches when the channel width is less than 33, where the area of EB would be increased by 33.27/W.

2) Performance: We study the area-delay product in different EB separation distances as shown in Figure 4. There are no switch box area overhead through out the experiment since the minimum channel width is about 50, the routing wires are dominating the routing area. Additional routing switches in EBs result in a significant reduction of 48.9% in the required channel at D_{eb}=0. This is because the additional switches split the long and inflexible straight wires inside EBs, which are more flexible to route a net.

Switches in EB achieve the best area-delay product at D_{eb}=1, which has the same performance as no switches in EB at D_{eb}=4. There is 65.69% reduction of area-delay product compared to the baseline architecture and 48.4% improvement in performance compared to column base architecture. The switches in EB increase the routability significantly, and separating EBs slightly is enough to minimize the minimum channel width and obtains highest performance.

C. Extra Routing Tracks

Betz et al. [9] suggested using wider channel in the center of fine-grained FPGA. This architecture did not improve the routability. It is because all circuits are forced to route most of their connections through the predefined wide channel. But those connections can be spread out in uniform FPGA. The large EBs introduce large routing demand at the edge of EBs, which cannot be spread out easily. In this section, we show that extra routability can be provided with low overhead by strategically inserting extra tracks in the channels surrounding the embedded blocks.

Figure 1a shows an example of employing extra routing tracks around the EBs. The normal segment width is W, R_{extra} is the ratio of number of tracks in wider segment to the tracks in normal segment. D_{extra} is the distance (in term of CLB length) from the edge of EB, within this distance, the width of the segment is $W * R_{extra}$. We investigate the impact of R_{extra} and D_{extra} on routability, area and delay of the baseline hybrid FPGA, and the optimized FPGA in previous sections. We select four systems to evaluate:
(1) $FPGA_A$: D_{eb}=0, no switches in EBs,
(2) $FPGA_B$: D_{eb}=0, switches in EBs,
(3) $FPGA_C$: D_{eb}=4, no switches in EBs,
(4) $FPGA_D$: D_{eb}=1, switches in EBs.

The area-delay product of using extra routing tracks in these systems is shown in Figure 5. Extra routing tracks are efficient to increase the performance of the highly congested $FPGA_A$ by 34.9% at R_{extra}=3 and D_{extra}=2. The reason is that the wide segment routes address most of the high density connections from EBs in the congested region, and only a small amount of connections are used in normal segments. As R_{extra} increases, the number of tracks in normal segments decreases.

Surprisingly, $FPGA_C$ and $FPGA_D$ are less efficient starting from R_{extra}=1.5 in both D_{extra}=1 and D_{extra}=2. $FPGA_C$ and $FPGA_D$ are very flexible to route, but the minimum channel width cannot be further reduced. Therefore, the extra tracks near the edge of EBs introduce additional area to the optimized system.

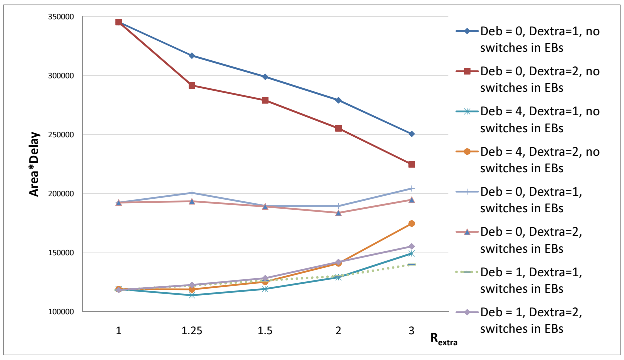


Fig. 5. The area*delay at different R_{extra} and D_{extra}

VI. Conclusions

Hybrid FPGAs require high routing resource demand. We explore four interconnection parameters in order to reduce routing area and net delay. First, we show the extra demand of routing requirements by examining channel width, segment length and netlength. Second, we examine the effect of the separation distance between EBs. Third, we study the trade-off when we add switches to embedded blocks, such that change in routing direction is allowed inside the blocks. Finally, we add extra wires surrounding coarse-grained units to accommodate high density connection. Future work includes extending the study to cover a wide range of applications, exploring the yield problem in process variation of architectures with the purposed routing optimizations.

Acknowledgement

We thank EPSRC and Xilinx for their support.

References

[1] C.H. Ho, C.W. Yu, P.H.W. Leong, W. Luk and S.J.E. Wilton, "Domain-Specific Hybrid FPGA: Architecture and Floating Point Applications," in *Proc. FPL*, 2007, pp. 196 – 201.

[2] D. Lewis, E. Ahmed, D. Cashman, T. Vanderhoek, C. Lane, A. Lee, and P. Pan, "Architectural enhancements in Stratix-III^{TM} and Stratix-IV^{TM}," in *Proc. FPGA*, 2009, pp. 33–42.

[3] I. Kuon and J. Rose, "Measuring the Gap between FPGAs and ASICs," *IEEE Trans. CAD*, vol. 26, no. 2, pp. 203–215, 2007.

[4] V. Betz and J. Rose, "FPGA Routing Architecture: Segmentation and Buffering to Optimize Speed and Density," in *Proc. FPGA*, 1999, pp. 59–68.

[5] C.W. Yu, J. Lamoureux, S.J.E. Wilton, P.H.W. Leong and W. Luk, "The Coarse-Grained/Fine-Grained Logic Interface with Embedded Floating-Point Arithmetic Units," *IJRC*, vol. 2008, Article ID 736203, 10 pages, 2008.

[6] *Predictive Technology Model (PTM)*, http://www.eas.asu.edu/~ptm/.

[7] C.W. Yu, "A Tool for Exploring Hybrid FPGAs," in *Proc. FPL PhD forum*, 2007, pp. 509 – 510.

[8] H. Schmit and V. Chandra, "FPGA Switch Block Layout and Evaluation," in *Proc. FCCM*, 2002, pp. 11–18.

[9] V. Betz and J. Rose, "Effect of the Prefabricated Routing Track Distribution on FPGA Area-Efficiency," *IEEE Trans. VLSI*, vol. 6, no. 3, pp. 445–456, 1998.