2011 21st International Conference on Field Programmable Logic and Applications

On Timing Yield Improvement for FPGA Designs Using Architectural Symmetry

Haile Yu, Qiang Xu
Dept. of Computer Science and Engineering
The Chinese University of Hong Kong
email: {hlyu,qxu}@cse.cuhk.edu.hk

Philip H. W. Leong
School of Electrical and Information Engineering
The University of Sydney
email: philip.leong@sydney.edu.au

***Abstract*—As semiconductor manufacturing technology continues towards reduced feature sizes, timing yield will degrade due to increased process variation. This work proposes the use of architectural symmetry in FPGA so that multiple timing-equivalent configurations can be derived from a single initial implementation, allowing the application of post-silicon tuning to mitigate process variation effects. Experimental results on twenty MCNC benchmark circuits for various process technologies demonstrate timing yield improvement using the proposed method.**

I. Introduction

As the transistor feature size is continuously scaled down, process variation has become increasingly severe, posing a great challenge to the timing yield of integrated circuits. It was shown in [1] that FPGA devices can suffer from performance degradation by a factor of 2 and an increase in leakage power consumption by a factor of 3. In contrast to ASICs, FPGA designs are placed on the device after they are fabricated. This provides an opportunity to customize the design at run-time, leveraging process variation for yield enhancement.

In this work, we propose a design methodology that combines configuration-level manipulation and fine-grained design tuning for timing yield improvement. Utilizing homogeneity and symmetry of an FPGA's architecture, multiple configurations can be generated by rotating and flipping an initial configuration. These different configurations, with equivalent functionality and timing, can be regarded as "candidate configurations". In the presence of process variation, the performance of a configuration is dependent on the individual chip's variation characteristic. The best configuration for a particular FPGA is determined by measuring the performances of all candidate configurations. Given the optimized configuration, a fine-grained design tuning technique can further improve timing through swapping neighboring logic elements (LEs). MCNC benchmark circuits are implemented using VPR [2]. Process variations are modeled with a predictive technology model and applied to netlists of benchmark circuits. The simulation results show our method achieves evident improvement in timing yield.

To the best of the authors' knowledge, this is the first work by using FPGA architectural symmetry and a mixture of course-grained and fine-grained design tuning for timing yield enhancement. The contributions can summarized as follows

1) A novel use of symmetry in homogeneous FPGAs to improve timing performance and yield in the presence of process variations.
2) A scheme to manipulate configurations to obtain different layouts in a symmetrical FPGA by rotation and flipping.
3) A fine-grained tuning scheme which employs logic element swaps to further improve timing.

The rest of the paper is organized as follows. Section II surveys related works and describes our motivation. A variation model is presented in section III and in section IV, the proposed architectural modification is described. The tool flow is presented in section V. In section VI, we present experimental results along with an analysis. Finally, conclusions are drawn in section VII.

II. Background

A. Previous Works

Several methods have been proposed to improve timing yield of FPGA design by statistical static timing analysis (SSTA) in placement [3] and routing [4]. Improvement beyond the tradeoff between performance and yield can be achieved by exploiting the reconfigurability of FPGAs. Given the variation characteristics for each individual FPGA, chipwise optimization effectively enhances timing yield [5][6][7]. This kind of approach is referred to as variation aware design (VAD) [8].

Although VAD seems to be a straightforward method, there remains some difficult problems, particularly with respect to the mass production of FPGAs. (1) *Cost of variation characterization:* A VAD method requires delay information for each circuit element of the FPGAs, which increases the difficulty and testing cost compared to traditional pass/fail testing. Such costs can be suppressed by reducing measurement resolution, however the effectiveness of VAD would be greatly degraded. (2) *Design tool execution-time:* VAD requires chip-specific design flows which are extremely inconvenient compared with the current method of producing non chip specific designs. (3) *Design tool fluctuation:* Many design tools give different

978-0-7695-4529-5/11 $26.00 © 2011 IEEE
DOI 10.1109/FPL.2011.105

results each time they are executed and may represent another source of performance variation. It has been reported that the VPR placement algorithm fluctuates by about 7% between average and best [9]. Performance variation caused by the design tools may therefore be larger than that due to process variations.

To tackle the problems above, Matsumoto et. al. used mutually exclusive critical path configurations (MECPCs) to improve timing yield [8]. MECPCs are generated by finding different routings of a fixed placement, which are equivalent in functionality and comparable in timing performance. The configuration with best timing performance for each individual FPGA is selected for the final implementation. Assuming 30% (σ/μ) variation in threshold voltage, with 10 MECPCs, the average critical path delay is reduced by up to 5%, with a corresponding 50% decrease in standard deviation.

B. Motivation

MECPC is a very straightforward method to improves timing yield by utilizing the FPGA's reconfigurability. However, there are several limitations associated with this approach. (1) Since MECPCs share a fixed placement, variation in the delay of logic elements (LEs) are not addressed. (2) The design must be routed a large number of times, greatly increasing design time. (3) Fluctuation of routing results may mean that some configurations are always slow and cannot contribute to timing improvement.

By leveraging FPGA architecture symmetry, we proposed a new approach to enhance timing yield. Given a symmetrical architecture, multiple equivalent configurations can be transformed from an initial design by configuration rotation and flipping, which should be more efficient in terms of CAD tool run-time than MECPCs. Fine-grained design tuning can further improve timing performance. Due to random variation, two neighboring LEs with identical nominal delay can be different. Post-layout LE swapping allocates faster resources to critical paths, improving timing.

III. Variation Model

For the purposes of this paper, process variation can be modeled as [10],

$$X_{total} = X_{d2d} + X_{wid}$$

where X_{d2d} and X_{wid} respectively denote the die-to-die and within-die variation. The inter-die variation affects all the components of a chip identically. The intra-die variation includes systematic and random variation. Systematic variation is, in turn, subject to spatial correlation within the chip.

A. Spatial Correlation

Devices that are located close to each other typically exhibit similar characteristics. The authors in [11] performed extensive critical dimension measurements on fabricated dies and proposed a simple piecewise linear model to capture the effect of spatial correlation, which is expressed as

$$\rho = \begin{cases} 1 - \frac{d}{D_L}(1-\rho_B) & d \le D_L \\ \rho_B & d \ge D_L \end{cases}$$

where ρ_B denotes the characteristic correlation baseline, and D_L denotes the characteristic correlation length. Physically, D_L is related to the gradient of the within-die systematic variation and naturally fall at roughly half the chip length; ρ_B is determined by the relative magnitudes of within-die and die-to-die variation components. ρ_B increases as the fraction of total variation accounted for by die-to-die variation increases. According to [11], D_L is half of the die size, and ρ_B is 0.1 and 0.32 respectively for vertical direction and horizontal direction. The spatial correlation model is applied to model the entire wafer and thus within-die systematic variation and within-wafer die-to-die correlation are considered.

B. Random Variation

We combine all sources of random variation to V_{th} variation, which is expressed as [12],

$$\sigma_{V_{th}} = \frac{C}{\sqrt{L_{eff} W_{eff}}} \quad (1)$$

where C is a technology-dependent parameter, which is a function of gate oxide thickness T_{ox} and doping concentration N_a; and L_{eff} and W_{eff} represent the effective channel length and width. As is the case with measured devices, smaller transistors have larger variation.

By applying the 16 nm predictive technology model (PTM) [13] to the equation 1, $\sigma_{V_{th}}$ can be calculated. The VPR architecture model [2] is used for the primitive circuits, and HSPICE Monte-Carlo simulation using $\sigma_{V_{th}}$ is used to obtain the statistical delay distribution in table I. This table gives the delay variation of the different combinational FPGA primitives (logic element (L.E.), local interconnect (Local), connection box (C.B.) and switch box (S.B.)) for the technology studied.

TABLE I
σ_D/μ_D of Delay Primitives for PTM 16*nm*

L.E.	Local	C.B.	S.B.
15.8%	14.6%	12.5%	8.84%

IV. Proposed Architecture

In this work, we adopt a homogeneous island-style FPGA architecture as described in VPR [2], which is conceptually symmetric. In VPR, delays for all CLB input pins to LE inputs pins are identical. Local interconnect delay (LE output pins to LE input pins within CLB) are assumed identical, and global interconnects of the same type have equivalent timing performance. In addition to these assumptions, several architectural modifications are made to model

configuration rotation and flipping, as well as fine-grained design tuning.

A. Modification for Configuration Rotation and Flipping

Given a symmetric FPGA architecture, an initial design can be modified by configuration rotation and flipping of the tiles to offer an additional seven primary permutations as shown in figure 1 [14].

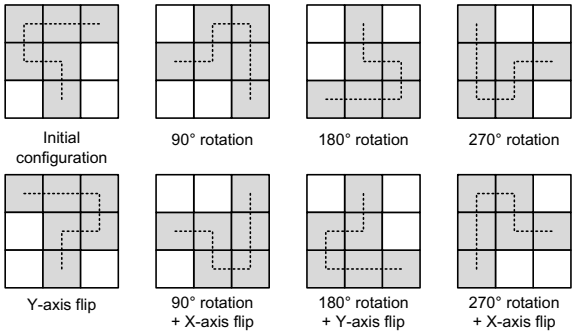
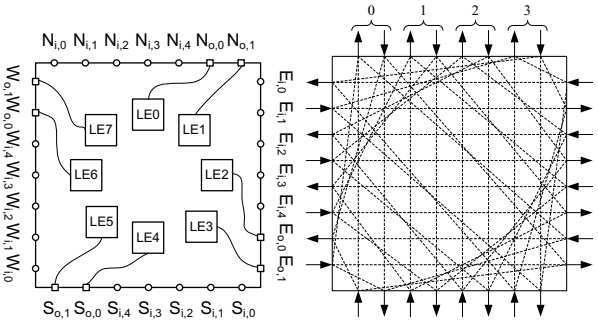


Fig. 1. Rotation and Flipping

To facilitate configuration rotation and flipping for an island-style FPGA, the two rules described below must be satisfied.



(a) Symmetric pin allocation along the edges of CLB

(b) Structure of switch box

Fig. 2. Detailed FPGA fabrics for symmetric architecture

(1) *Configurable logic blocks are rotatable.* From figure 1, we observe that configuration-level rotation and flipping of tiles can be performed by applying the corresponding action to each individual CLB. For example, if the entire configuration is rotated clockwise by $90°$, each individual CLB must be relocated and internally rotated in the same way. It requires CLB I/O pins must be evenly allocated on the four edges, which means on each edge of CLB, there should be identical number of I/O pins. Therefore, the number of LEs within a CLB must be an integer multiple of 4, to guarantee that each edge of CLB contains the same number of output pins. According to the empirical formula in [15], the number of input pins per CLB I, the number of LEs per CLB N and the number of LUT inputs K are related as

$$I = \frac{K}{2}(N+1)$$

. In the example in figure 2(a), $K = 4$ and $N = 8$ so $I = 18$. However, if I is 18 as calculated by the empirical equation, the number of input pins on each of CLB would not be identical, therefore breaking symmetry. To satisfy even allocation of input pins along the CLB, I is rounded up to 20. We also assume that the chip pins, including general I/O, clock and power supply are evenly allocated on 4 edges of entire chip. This ensures that the logic circuits in the CLB are equivalent after relocation, rotation and flipping, and is independent of global routing.

(2) *Global routing is symmetrical.* In this work, a directional routing architecture is adopted. The flexibility of switch box (SB) F_s is 3 and the type of SB is "subset" as shown in figure 2(b). Two neighboring wire segments with opposite directions are grouped as one "track". The output pin around SB only receives signals from input pins on the same track. Therefore, any interconnect net can be only composed of wire segments in the same track. By rotating and flipping a particular global routing path, seven different routes can be obtained. As rotation does not change relative location between two LEs, the routing is simply rotated in the same manner. Flipped paths would be composed of mirrored routing resources. Therefore, as long as the original design is implemented successfully, no routing congestion would be induced by rotation and flipping. The symmetrically physical design ensures that the eight primary configurations are equivalent in terms of timing performance. Full connection flexibility from CLB I/O pin to tracks is assumed.

For an $n \times n$ FPGA, assume LE i is located in CLB (x,y). The CLB can be rotated and flipped via the formula below.

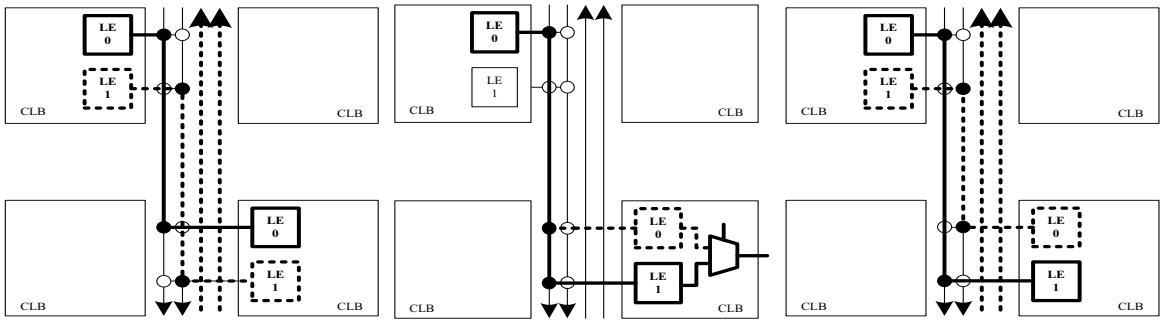
$$\begin{aligned}(x,y) &\xrightarrow{90°C.W.rotation} (y, n-x)\\ (x,y) &\xrightarrow{\text{y-axis flipping}} (n-x, y)\\ (x,y) &\xrightarrow{\text{x-axis flipping}} (x, n-y)\end{aligned}$$

The LE index i is unchanged by flipping. Rotation of i is performed using the following formula.

$$i \xrightarrow{90° \text{ C.W. rotation}} (i+1) \ mod \ 8$$

B. Modification for Fine-Grained Design Tuning

To facilitate LE swaps, an additional multiplexor is needed to select from two neighboring LEs $2i$ and $2i+1$, $i = 0, 1, \ldots, n$, assuming there are $2n$ LEs in a CLB. This is illustrated in figure 4. If a CLB contains $2n$ LEs, n additional multiplexors are needed. Although these multiplexors are not included in the default VPR architecture, they are available in Xilinx FPGAs to optimism large multiplexors [16]. In this work, we use such a multiplexor to facilitate on-chip self-testing of the relative speed of neighboring LEs, detailed in the following paragraph. As design tuning



(a) Step 1: To extract critical path from initial design (b) Step 2: To modify critical path with duplications and measure performance (c) Step 3: To swap LEs for timing performance improvement

Fig. 3. An illustration from initial design to optimized design by fine-grained design tuning

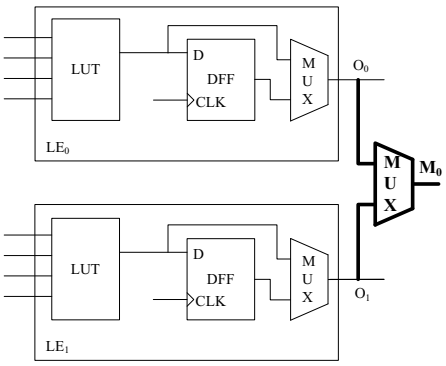


Fig. 4. Modified logic block composition

only swaps LEs between a critical and non-critical path, no resource congestion is introduced. Furthermore, intra-CLB design tuning does not affect global routing, meaning that optimized designs can be generated via simple transforms rather than re-routing.

The process of LE swapping can be further divided into three steps.

1) For a given design, the critical path is extracted. In figure 3(a), the solid line and dotted lines respectively represent critical path and non-critical paths.
2) In figure 3(b), by removing non-critical paths, each LE along a critical path can be duplicated with its neighboring LE, followed by a multiplexor. Suppose the critical path has S LEs, there are 2^S combinations of selection signals. However, we do not need to try every combination to determine the optimal path. In figure 3(b), the relative speed of LE0 and LE1 can be determined by simply switching the multiplexor selection signal while leaving other selection signals unchanged. By doing this for all neighboring LEs, only $2 \times S$ trials are required to obtain the optimal critical path.
3) Assuming LE1 is faster than LE0, a swap is conducted to generate the optimized design shown in figure 3(c). Although the nominal timing performance of the two designs is the same, due to process variation, the swapped design is faster than the original.

V. Simulation Flow

The design flow used in this work is shown in figure 5. Given the critical path contains S tunable LEs, the fine-grained design tuning process is performed on each primary configuration. A vector of multiplexor selection signals, denoted as $Conf$ in figure 5, contains the optimized critical path, based on which the final implementation is generated.

Due to resource congestion, some LEs on critical paths can not be swapped with their neighbors. If 6 out of 8 LEs in a cluster are occupied by the target path, only 2 LEs can be swapped. We define coverage rate R_c as

$$R_c = P_l \times \frac{N_t}{N_a}$$

to denote the proportion of swappable LEs, where P_l, N_t and N_a respectively denote the percentage of logic delay, number of swappable LEs and total number of LEs.

In the presence of process variation, any near-critical path may turn out to be critical and need to be considered.

VI. Experimental Results

We use the process variation model described in Section III to constrcut 1000 FPGA variation distributions from 10 wafers. Twenty MCNC benchmark circuits are placed and routed by VPR. Since in the presence of variation, multiple paths could be the real critical path, we modify VPR to find near critical paths, defined as those whose delays are larger than 90% of the critical path delay. The real timing performance is determined by applying the variation model to the post-layout netlists.

With consideration of process variation, the critical path delays for a design over 1000 FPGAs are described by a distribution rather than a single value. We use the mean and standard deviation to present timing performance. Improvements can be identified as reductions of the mean and standard deviation, denoted as Δ_μ and Δ_σ. In table II, we

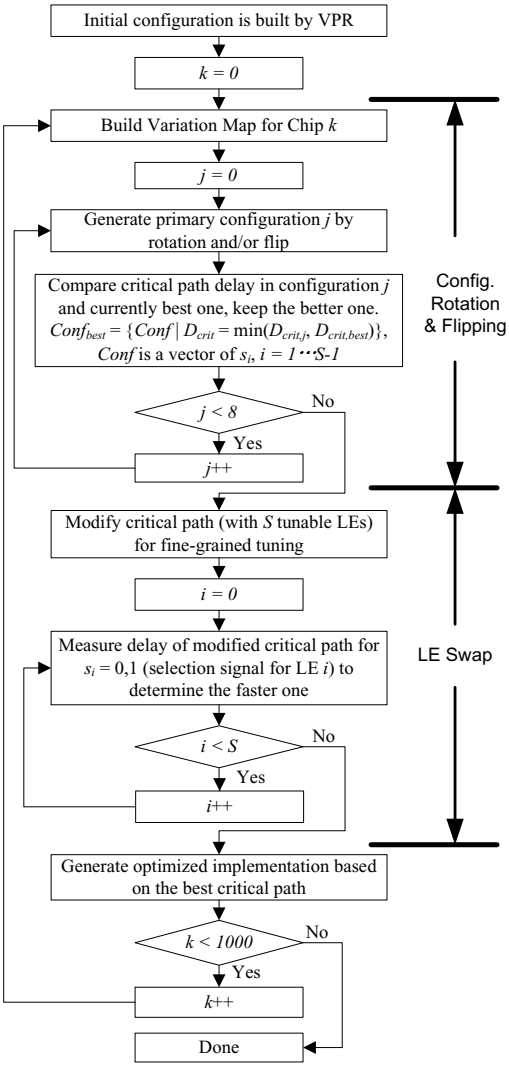


Fig. 5. Simulation flow

compare the critical path delay distribution of a standard design with an optimized one. The values in the second and third columns denote reduction of mean and standard deviation.

After applying LE swapping to the optimized design from configuration's rotation and flipping, the further reduction of mean and standard deviation are presented in the fourth and fifth columns. This is not as large as configuration rotation and flipping partially because LE swapping only deals with random variation. Moreover, since only LEs are swapped, routing delay cannot be improved. As mentioned in section V, not all LEs along the critical path can be swapped, which further limits the potential improvement. On average, coverage rate R_c of LEs is 47.4%

Compared to MECPCs, the proposed method achieves a much larger reduction of the mean because configuration

TABLE II
EXPERIMENTAL RESULTS

	Config. R. & F.		LE Swapping			
	$\Delta_\mu\%$	$\Delta_\sigma\%$	$\Delta_\mu\%$	$\Delta_\sigma\%$	$\Delta_{D_t}\%$	$\Delta_Y\%$
alu4	-10.9	-20.4	-3.81	-0.9	-15.3	89.5
apex2	-9.99	-13.9	-4.88	-2.4	-14.6	88.9
apex4	-10.7	-16.9	-2.60	-1.5	-13.8	87.4
bigkey	-7.07	-10.7	-4.11	0.8	-9.10	74.7
clma	-8.04	-16.0	-3.68	-3.6	-12.5	89.8
des	-7.95	-4.68	-5.22	-15.4	-10.6	81.8
diffeq	-10.6	-16.9	-5.64	-8.0	-16.9	93.7
dsip	-7.93	-14.7	-4.44	3.4	-10.8	78.5
elliptic	-8.81	-13.7	-5.54	-12.0	-15.3	94.3
ex1010	-11.0	-20.5	-2.94	-2.4	-12.6	92.6
ex5p	-7.21	-8.10	-4.13	1.0	-15.4	78.4
frisc	-10.6	-18.8	-6.10	-13.1	-16.6	97.9
misex3	-10.2	-14.4	-3.28	-4.6	-13.9	85.1
pdc	-6.91	-16.2	-2.95	4.0	-9.85	75.5
s298	-10.3	-18.2	-4.01	-0.4	-14.6	89.4
s38417	-5.87	-10.2	-5.51	-8.6	-11.9	89.7
s38584.1	-8.15	-9.53	-4.84	-10.7	-13.5	84.0
seq	-10.4	-25.6	-2.94	-0.5	-14.9	90.5
spla	-7.16	-10.9	-2.91	-0.2	-10.0	75.1
tseng	-11.2	-23.0	-5.32	-8.3	-17.9	96.8
Average	-9.05	-15.2	-4.24	-4.2	-13.5	86.7

rotation and flipping better utilizes variations in spatial correlation. However, reduction of the standard deviation is less than that of MECPCs. This is mainly because in MECPCs, only within-die random variation is considered and die-to-die variation is ignored.

Timing yield can be evaluated from two equivalent perspectives. (1) Given a timing performance target, evaluate the proportion of chips that satisfy the timing constraint. (2) Given a yield target, evaluate how fast a design can operate over a number of circuits and meet a yield constraint.

We define the timing performance target (D_t) as the 95% percentile of the distribution. This means that 95% of chips can satisfy the timing constraint D_t. We can then compare two D_t's from our post-layout optimization and traditional design flow. The percentage of D_t reduction is shown in the last column. On average, configuration rotation and flipping can reduce the mean of the critical path delay distribution by 9.05%. LE swapping can further reduce the value by 4.24%. The combined optimization technique achieves a 13.5% reduction in D_t.

Taking benchmark circuit "*clma*" as an example, if we assume the mean of critical path delay distribution of the original design as the yield target, approximately half of the chips can satisfy this timing constraint as shown in figure 6. After optimization, as stated before, both mean and standard deviation of the distribution are reduced, making the distribution tighter and moving it to left. A statistical analysis shows that 94.9% chips using optimized design can now satisfy the timing constraint. Yield is improved by 89.8%.

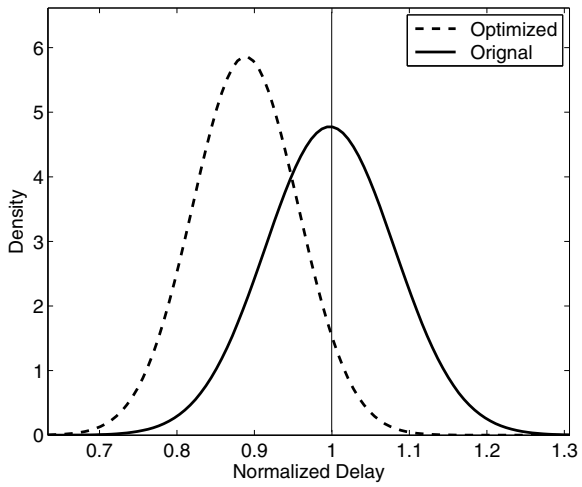


Fig. 6. "The distributions of critical path delay for original design and optimized design (after both configuration rotation and flipping + LE swapping) for benchmark circuits *clma*"

VII. Conclusion and Future Work

In this work, we first presented a theoretical analysis on how different kinds of redundancy schemes can contribute to timing yield improvement in the presence of random variation. Based on this, we proposed a novel FPGA design methodology combining configuration-level redundancy and fine-grained design tuning. Experimental results show that the proposed technique reduces the mean and standard deviation of critical path delay over 20 MCNC benchmark circuits and hence significantly improves timing yield.

To make the proposed design methodology more practical, we plan to address the following issues in our future work.

- Currently, the 8 different configurations have 8 different I/O pin assignments. We plan to investigate virtual I/O schemes to isolate functional circuits from physical I/O pins, while keeping the timing performances of all configurations identical.
- FPGA devices are often not perfectly symmetrical as assumed in this work. We plan to evaluate the impact of our technique on non-symmetric architectures.

VIII. Acknowledgement

This work was supported in part by National Science Foundation of China (NSFC) under grant No. 60876029, in part by the General Research Fund CUHK417807 and CUHK418708 from Hong Kong SAR Research Grants Council (RGC), and in part by a grant N_CUHK417/08 from the NSFC/RGC Joint Research Scheme.

References

[1] H.-Y. Wong, L. Cheng, Y. Lin, and L. He, "FPGA device and architecture evaluation considering process variations," in *IEEE/ACM International Conference on Computer-Aided Design*, Nov 2005, pp. 19– 24.

[2] V. Betz and J. Rose, "VPR: A new packing, placement and routing tool for FPGA research," in *Proceedings of the 7th International Workshop on Field-Programmable Logic and Applications*. Springer-Verlag, 1997, pp. 213–222.

[3] Y. Lin, M. Hutton, and L. He, "Placement and timing for FPGAs considering variations," in *International Conference on Field Programmable Logic and Applications (FPL)*, Sep 2006, pp. 1–7.

[4] S. Sivaswamy and K. Bazargan, "Variation-aware routing for FPGAs," in *Proceedings of ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA)*, 2007, pp. 71–79.

[5] K. Katsuki, M. Kotani, K. Kobayashi, and H. Onodera, "A yield and speed enhancement scheme under within-die variations on 90nm LUT array," in *Proceedings of the IEEE Custom Integrated Circuits Conference (CICC)*, 2005, pp. 601–604.

[6] S. Srinivasan and V. Narayanan, "Variation aware placement for FPGAs," in *Proceedings of IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures*, vol. 00, Mar 2006, p. 2 pp.

[7] L. Cheng, J. Xiong, L. He, and M. Hutton, "FPGA performance optimization via chipwise placement considering process variations," in *Proceedings of International Conference on Field Programmable Logic and Applications (FPL)*, Aug 2006, pp. 1–6.

[8] Y. Matsumoto, M. Hioki, T. Kawanami, T. Tsutsumi, T. Nakagawa, T. Sekigawa, and H. Koike, "Performance and yield enhancement of FPGAs with within-die variation using multiple configurations," in *Proceedings of ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA)*, 2007, pp. 169–177.

[9] J. Cong, M. Romesis, and M. Xie, "Optimality and stability study of timing-driven placement algorithms," in *Proceedings of IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, Nov. 2003, pp. 472–478.

[10] H. Chang and S. Sapatnekar, "Statistical timing analysis under spatial correlations," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 9, pp. 1467–1482, Sep 2005.

[11] P. Friedberg, Y. Cao, J. Cain, R. Wang, J. Rabaey, and C. Spanos, "Modeling within-die spatial correlation effects for process-design co-optimization," in *Proceedings of IEEE International Symposium on Quality of Electronic Design (ISQED)*, pp. 516–521, 2005.

[12] A. Asenov, A. Brown, J. Davies, S. Kaya, and G. Slavcheva, "Simulation of intrinsic parameter fluctuations in decananometer and nanometer-scale mosfets," *IEEE Transactions on Electron Devices*, vol. 50, no. 9, pp. 1837–1852, Sep 2003.

[13] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45nm design exploration," in *Proceedings of International Symposium on Quality Electronic Design (ISQED)*, pp. 585–590, 2006.

[14] K. Compton, Z. Li, J. Cooley, S. Knol, and S. Hauck, "Configuration relocation and defragmentation for run-time reconfigurable computing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 10, no. 3, pp. 209–220, Jun 2002.

[15] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deep-submicron FPGA performance and density," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 3, pp. 288–298, Mar 2004.

[16] Xilinx, "Using dedicated multiplexers in spartan-3 generation FPGAs."