ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.11, No.3, 1–9

Classification on variation maps: a new placement strategy to alleviate process variation on FPGA

Zhenyu Guan[a)], **Justin S. J. Wong, Sumanta Chaudhuri, George Constantinides, and Peter Y. K. Cheung**

Department of Electrical and Electronic, Imperial College London, South Kensington Campus, London SW7 2AZ, UK

a) *zhenyu.guan@googlemail.com*

Abstract: This paper proposes a 2-stage variation-aware placement method that benefits from the optimality of a full-chipwise (chip-by-chip) placement to alleviate the impact of process variation. By classifying FPGAs into a small number of classes based on their variation maps and performing placement optimisation specifically for each class instead of each chip, two-stage placement can greatly reduce the execution time with similar timing improvement as achieved by full chipwise optimal placement. Our proposed method is implemented in a modified version of VPR 5.0 and verified using variation maps measured from 129 DE0 boards equipped with Cyclone III FPGAs. The results are compared with variation-blind, Statistical static timing analysis (SSTA) and full chipwise placement. The timing gain of 7.5% is observed in 20 MCNC benchmarks with 16 classes for 95% timing yield, while reducing execution time by a factor of 8 compared to full-chipwise placement.

Keywords: FPGA, process variation, placement, variation maps

Classification: Integrated circuits

References

[1] Y. Lin, L. He and M. Hutton: IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **16** (2008) 124.

[2] Y. Matsumoto, M. Hioki, T. Kawanami, H. Koike, T. Tsutsumi, T. Nakagawa and T. Sekigawa: ACM Trans. Reconfigurable Tech. Syst. **1** [1] (2008) 3:1.

[3] P. Sedcole and P. Y. K. Cheung: ACM Trans. Reconfigurable Tech. Syst. **1** [2] (2008) 1936.

[4] E. Stott, Z. Guan, J. M. Levine, J. S. J. Wong and P. Y. K. Cheung: IEEE Design Test (2013) 1.

[5] P. Sedcole and P. Y. K. Cheung: ACM/SIGDA FPGA (2007) 178.

[6] D. Aghamirzaie, S. A. Razavi, M. S. Zamani and M. Nabiyouni: IEEE/IFIP VLSI System on Chip Conference (2010) 85.

[7] J. S. J. Wong, P. Sedcole and P. Y. K. Cheung: IEEE FPT (2008) 105.

[8] Z. Guan, J. S. J. Wong, S. Chaudhuri, G. Constantinides and P. Y. K. Cheung: IEEE FPL (2012) 519.

[9] C. M. Bishop: *Pattern Recongnition and Machine Learning* (Springer

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014

Science+Business Media LLC, 2006).
[10] Z. Guan: Ph.D thesis Imperial College London, London (2013).

1 Introduction

Field-Programmable Gate Arrays (FPGAs) have become one of the most popular technology for implementing digital circuits, but are expected to face new challenges posted by the future increases in process variability [5]. To guarantee the physical timing performance and reliability of FPGA products, either the physical yield of FPGAs chips would be compromised through a more vigorous manufacturing test, or a more conservative timing model that takes variation into account is used. However, it is impossible to have one universal model that precisely describes the timing of all components under process variation for a big number of FPGAs because each of them may have unique variation pattern. Therefore, existing commercial FPGA design tools rely on the worst-case delay model of FPGA components, such that all possible patterns of process variation are taken into account for all FPGAs. Nonetheless, this approach may cause sub-optimal operating speed because certain components may be able to operate at faster speeds.

Compared with manufacturing level process control, variation-aware design technology only involving post-silicon optimisation is more cost-effective and significantly easier to apply. Several high level variation-aware optimization based on Computer-Aided Design (CAD) tools are exploited in recent years.

2 Previous work

2.1 Full chipwise variation-aware placement

Theoretically, performing variation-aware placement on individual device based on its delay variation, called full chipwise placement, will yield optimal timing performance for that device. To apply full chipwise placement and routing, each component is assigned an unique delay value based on the variation map. The algorithms of placement is modified to use the delay value from the variation map to evaluate the timing performance. For each FPGA, the configuration of placement is unique and generated on a chip-by-chip basis. However, this "per chip" strategy is impractical because the execution time required is $O(N)$ where N is the total number of devices used, and can be very large.

2.2 SSTA placement

Statistical model of variation and SSTA is used by published researches in variation-aware placement. By replacing the static and deterministic timing of LUT and interconnects with probability distribution, the delay of end-to-end signal paths is calculated by SSTA placement algorithm as distribution. SSTA method would outperform worse-case timing with a negligible quantity of chips which fail timing requirement. In addition, SSTA enables a trade-off between product parametric yield and speed. The results of SSTA-driven optimisation reported so far are promising [1].

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014

Nevertheless, these works assume that the same timing variability model is representative for all devices of a given type. This "one configuration fits all" approach, though efficient, assumes that the approximated timing model matches the physical timing of specific FPGA chips.

2.3 Multiple configurations

For one native implementation of a circuit, there is a probability that it can meet the timing requirement without any knowledge of process variation. Therefore, by selecting one appropriated solution from a set of functionally equivalent configurations, the delay of critical path can be reduced under large stochastic and spatial correlated process variation. Comparing with the "one configuration fits all" approach, the multiple configuration method can potentially improve timing yield at the expense of needing to compute and store extra configurations [2, 6]. However, there are some limitations with this methodology. Firstly, each configuration must first be generated with a set of valid placement and routing, and maybe stored in some form of memory in an embedded system. Secondly, it is hard to make sure that multiple configurations can cover all possible optimized solution for specific variation map [3].

3 Experiement and results

3.1 FPGA characterisation and classification

The transition probability method is adopted in this experiment to collect variation map for each FPGA. One of the variation maps measured in this experiment is shown in Fig. 1. The unit of the finest measurable component in the Cyclone III FPGAs using the transition possibility method is two adjacent Look Up Tables (LUTs) and the interconnect between them inside one Logic Array Block (LAB) [7]. However, to avoid overhead of runtime for full chipwise variation-aware placement, the region-based variation maps are used in this project which means the resources share

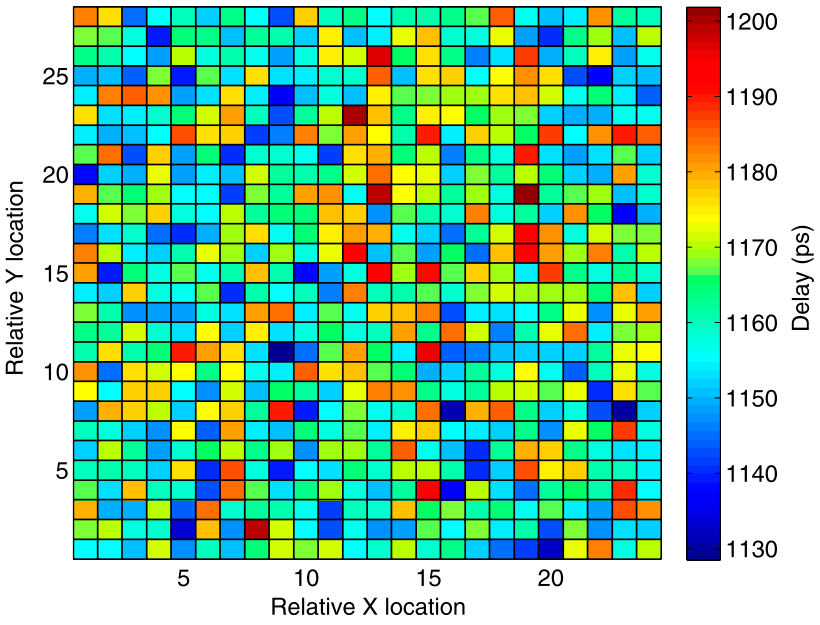

Fig. 1. One variation map measured on Cyclone III FPGA [4].

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014

same variation parameters in one region.

The k-means method [9] is used to classify all measured variation maps into a finite number of classes based on least-square error. The number of classes is flexible and adaptive to the design requirement but 16 is chosen in our experiment as the initial class number for the 129 variation maps. The impact of the number of classes will be discussed in later sections. The results of classification based on the filtered variation maps are shown in Fig. 2 where the median maps of 16 classes are illustrated. From this figure, it can be seen clearly that the median map of different classes differ significantly from each other.

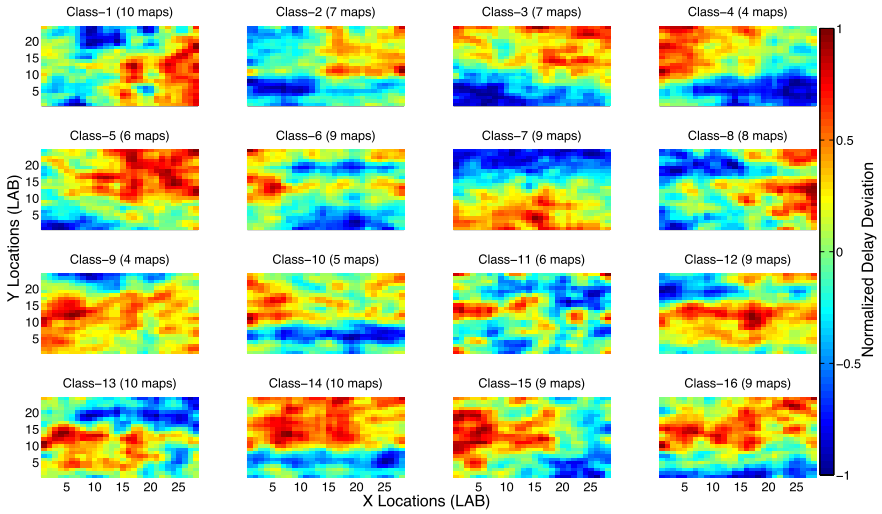

Fig. 2. Collection and classification on variation maps [10].

3.2 Proposed two-stage placement

A two-stage variation-aware placement is proposed in this paper to save execution time and alleviate the impact of process variation. The key idea of two-stage variation-aware placement is to classify variation maps into finite number of classes in the first stage, and then apply full chipwise variation-aware placement only to the median map of each class. In the second stage, the placement produced in the first stage can be loaded for the other FPGAs with variation maps that fall into the same class, thus reducing the execution time required.

The advantage of our two-stage variation-aware and adaptive placement is that it only requires $O(k)$ time where k is the number of classes used by classification, and k can be chosen to be significantly smaller than N. A tradeoff between timing performance and execution time can be made according to the end-users' requirement by choosing a suitable k [10].

3.3 Experiment flow

Four placement methods, variation-blind, SSTA, full chipwise and two-stage placement are tested as shown in Fig. 3. For variation-blind method, an original placement is performed to generate one placement configuration without considering the effect of process variation; for SSTA, collected variation maps are analysed and modeled [1] at first. One placement configuration is produced by SSTA-driven algorithm based on the variation

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014

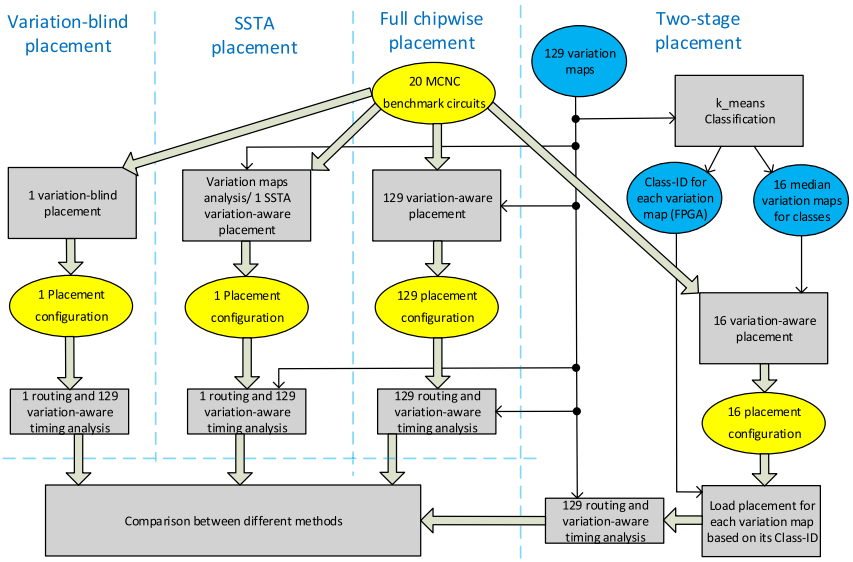

Fig. 3. Work flows for variation-blind, SSTA, two-stage placement and full chipwise placement [8].

model; for full chipwise placement, the variation maps are used during the variation-aware placement process. In this case, 129 placement configurations are produced with corresponding variation maps; for two-stage placement method, all variation maps are classified into 16 classes in the first stage and each variation map is assigned a Class-ID. The median map of each class is used to perform a variation-aware placement. In the second stage, the results of variation-aware placement are loaded for each variation map according to its Class-ID. The noise-reduced routing and variation-aware timing analysis is performed for after placement for each strategy. At the end, the results provided by different methods are collected and compared in terms of delay of critical paths for 95% timing yield [8].

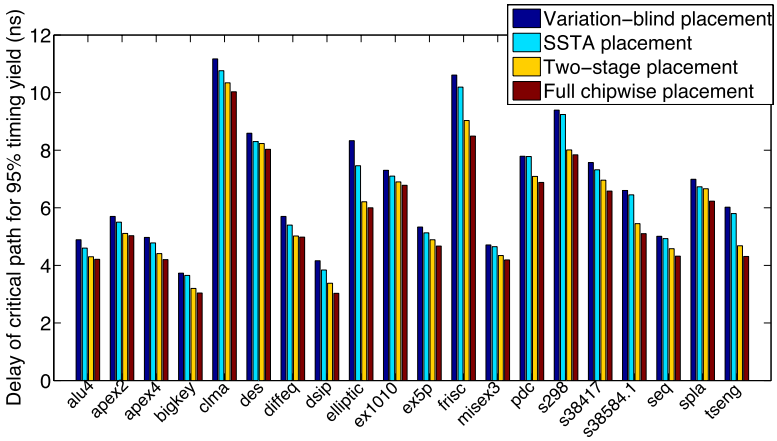

Fig. 4. Under process variation, timing performance in a set of benchmark FPGA applications is improved by tailoring the configuration to fit the measured delay map. A bespoke resource placement for each device gives the greatest benefit, while the classified approach delivers moderate gains at a far lower computational cost [4].

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014

3.4 Results

The results between variation-blind, SSTA, two-stage and full chipwise placement for 20 MCNC benchmarks are shown in Fig. 4. The variation blind represents the results produced by the unmodified VPR, where timing data from the variation maps are only used at the end for obtaining critical path delay. The timing gain with the two-stage, SSTA and full chipwise methods are calculated using the variation-blind results as references. For 95% timing yield, the improvements of the two-stage placement with 16 classes, SSTA and full chipwise placement are 7.5%, 4.3%, 11% respectively. As we expected, the two-stage placement can achieve better performance than SSTA but less than full chipwise.

The improvement of variation-aware placement varies between benchmarks because of variations in the length of critical path, the ratio of logic delay to interconnect delay and the number of near critical paths.

4 Analysis and enhancement

To explore the efficiency of the proposed two-stage placement, one of benchmarks, *frisc*, is chosen to be tested with more experiments, whose logic delay is twice as much as its interconnect delay, allowing it to benefit more from the placement optimisation, and hence shows the highest timing improvement.

4.1 Comparison of run time cost

It is not easy to measure the absolute run time in our experiment. Therefore, the execution time is estimated and normalized to show the difference between different placement optimization methods in Fig. 5 for 129 FPGAs. For SSTA method, only one SSTA placement is applied for all FPGAs. For full chipwise, the run time is proportional to the number of FPGAs because it treats every FPGA differently. For the two-stage placement, instead of the number of FPGAs, the run time is related to the number of classes we chosen. In this case, 16 full chipwise placement are required based on the median map of each class. For other FPGA in each class, the execution time cost to load the placement is negligible compared with the time used by full chipwise placement. The run time will increase proportionally with higher number of classes, but may also result in better

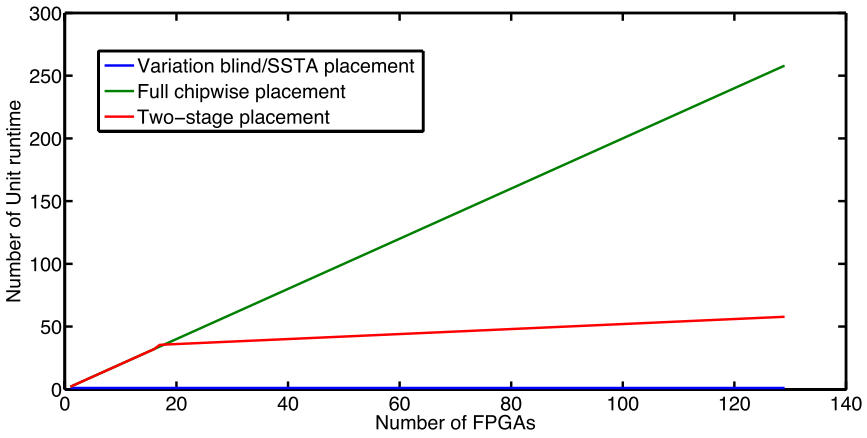

Fig. 5. Run time cost of SSTA chipwise and two-stage placement.

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014

timing performance in terms of critical path delay due to better matching of the actual variation map on each FPGA.

Although the improvement achieved by two-stage placement is less than that of full chipwise variation-aware placement, its main advantage over full chipwise is the much reduced execution time. In this case, the two-stage method only requires 16 variation-aware placements based on the median map of each class, whereas the full chipwise method has to do 129 individual placements for all variation maps. The execution time of the classification process using Matlab (about 30 seconds) is negligible compared with VPR placement (more than 30 minutes). Therefore, the amount of computation is reduced by a factor of 129/16 which translates into about 8 times speedup.

4.2 Choosing the number of classes

The potential timing improvement is a function of the number of classes (k). What we are interested in is how the critical path timing scales with k which in turn scales the total execution time of the placement. To find out the relationship between them, we scaled k from 1 to 129 and observed how the critical path delay changes for the *frisc* benchmark circuit. If k is 1, the two-stage placement is similar to the "one configuration fits all" method which provided a lower-bound solution with one configuration. While k equals N gives the upper-bound solution identical to full chipwise placement. The results are shown in Fig. 6. As expected, the critical path delay decreases with higher number of classes. Therefore, FPGA users can select a desirable number of classes according to their design specifications and placement execution time constrains by trading-off between timing performance and execution time.

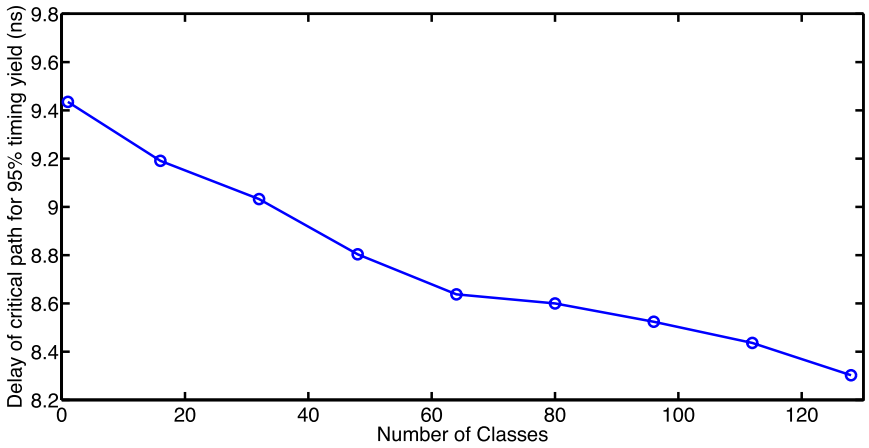

Fig. 6. Number of clusters (k) against critical path delay for *frisc*.

4.3 The effect of FPGA utilisation on timing improvement

How FPGA resource utilisation ratio affects timing gain with the two-stage method is also of interest, since more unused resources should imply more freedom in the placement process to achieve better critical path timing. Utilisation ratio (Ur) in this experiment is set according to:

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014

$$Ur = \frac{used\_blocks}{total\_blocks} \quad (1)$$

The results of the *frisc* benchmark with different Ur are shown in Fig. 7. With variation blind and SSTA, there is a clear increase in critical path delay as the FPGA Utilisation ratio (Ur) approaches 100%. However, the negative effect of increasing Ur is much less apparent with the full chipwise approach, and our two-stage method lies slightly above it with better overall timing than the variation blind approach. One explanation of the observations is that *frisc* has critical and near critical paths that fit well within the fast region on the FPGAs in most cases and thus reducing the size of FPGA to increase Ur had relatively small impact on the critical path delay.

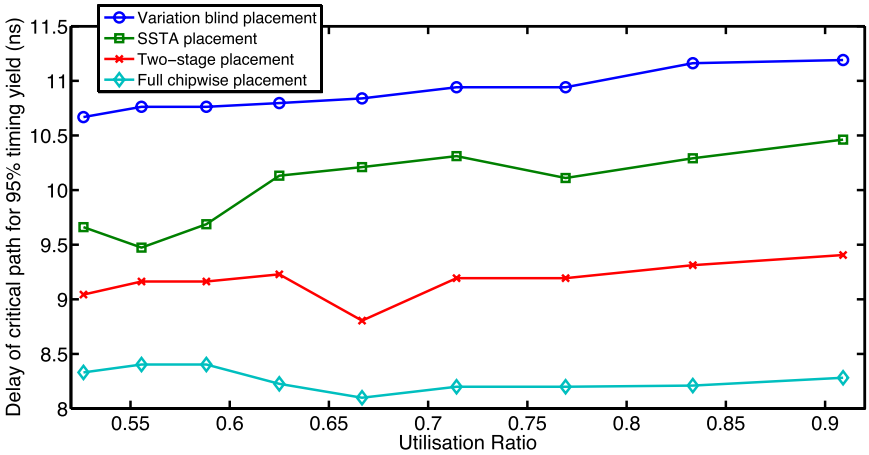

Fig. 7. FPGA Utilisation Ratio (Ur) against critical path delay for *frisc* [10].

4.4 Classification enhancement

The k-means method used in the experiment is low in complexity and execution time, and easy to apply. To determine if enhancing the classification with other algorithms, such as PCA, is beneficial, we combined PCA with the original k-means method and repeated our

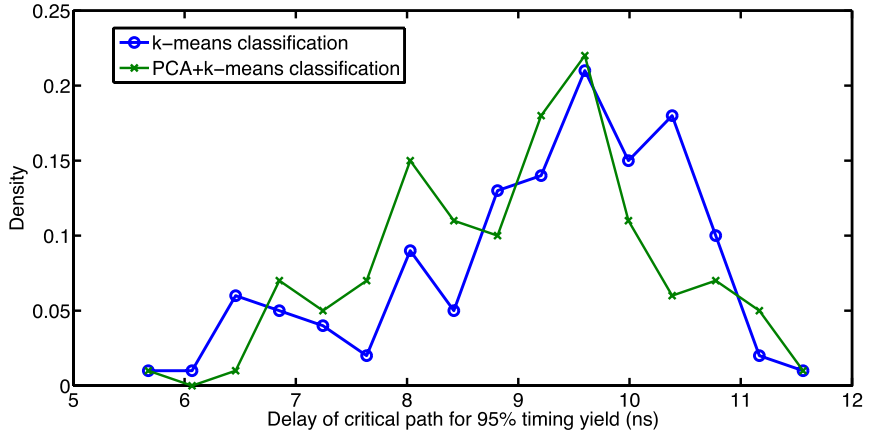

Fig. 8. Density of critical path produced by k-means classification against PCA with k-means for *frisc* [10].

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014

experiment with the *frisc* benchmark. The results are shown in Fig. 8 where the difference between the distributions of k-means and PCA with k-means in terms of critical path over 129 variation maps is within the uncertainty (i.e. noise) in the placement, and is not significant. Therefore, we conclude that k-means method alone is sufficient and should be used.

5 Conclusion

This paper employed detailed delay variation measurements of 129 Cyclone III FPGA chips to demonstrate the potential timing improvement of designs using variation-aware timing-driving placement in VPR. With full-chipswise and SSTA, about 11% and 4.3% improvement in terms of critical path delay for 95% timing yield were observed, while the two-stage optimization method achieved a 7.5% improvement but showed an 8 times speedup in total placement execution time compared with the full chipwise placement. For N FPGAs and k classes, the speedup relative to full chipwise is expected to be N/k. The observed timing improvement and reduction in execution time with the two-stage method clearly demonstrate its effectiveness and practicality against delay variability in FPGAs. While variability will inevitably impact design timing yield on FPGAs in the future, our method provides one promising solution that can be easily employed to alleviate the problem caused by severe processes variation.

DOI: 10.1587/elex.10.20130912
Received November 19, 2013
Accepted November 27, 2013
Publicized December 05, 2013
Copyedited February 10, 2014