Layout-aware Signal Selection in Reconfigurable Architectures

Prateek Thakyal
ECE, University of Florida, USA
Email: thakyal@ufl.edu

Prabhat Mishra
CISE, University of Florida, USA
Email: prabhat@cise.ufl.edu

Abstract—**Post-silicon validation is an important and increasingly complex task in SoC design methodology. One of the major challenges in post-silicon debug is the limited observability of internal signals. Existing signal selection techniques try to maximize observability by selecting a small set of profitable trace signals. Unfortunately, these techniques do not consider design constraints such as routing congestion in reconfigurable architectures. In this paper, we propose a layout-aware signal selection algorithm that takes into account both observability and routing congestion in field-programmable gate array (FPGA). Our experimental results demonstrate that our approach can trade-off between observability and wire-length reduction in FPGA-based designs.**

Keywords-**Post-Silicon Debug, Signal Selection, Layout, FPGA**

I. Introduction

Signal selection is a crucial step during design time (pre-silicon stage) since it affects the effectiveness of post-silicon validation. A key challenge during post-silicon debug is limited observability of the internal nodes. Due to design overhead constraints, only a small set of trace signals is used, for example, 64-128 signals in a design with millions of signals. Due to the limited coverage of design by the signals, it may not be possible to root cause all the bugs. The increase in complexity exacerbates this problem in SoCs and hence there is a great need for validation through reconfigurable architecture based prototyping. Unlike simulation, FPGAs can run with real interfaces and thereby accurately model and catch bugs much faster.

Signal selection differs in the FPGA domain compared to the SoC domain in terms of usage. While for SoCs the selected signals cannot be altered once fabricated (unless a larger set is used with dynamic selection), in FPGAs the validation engineer may change the signals for observation through reconfiguration based on debug requirements. The ability to select a different set of signals would be beneficial in many scenarios. For example, a verification engineer may like to focus on a specific set of components (functional regions) during debug. Some components can be ignored in a certain duration during execution due to clock gating and other considerations. Similarly, certain regions may be well verified datapath and less likely to have errors compared to other control-intensive regions.

Debugging software and post-silicon debug of SoC designs are different in terms of the ability to observe internal signals. While debugging some software code, it is possible to observe all the internal variables or signals. However, during post-silicon debug of integrated circuits, visibility of the internal signals is very limited due to constraints in terms of simulation duration and the number of signals that can be observed. Debugging in FPGAs bridges this gap.

Although the FPGAs may not supply the freedom to choose any number of signals of IDE (integrated development environment), the power to alter the trace signals through reconfiguration, without any area or delay penalty, is a huge advantage.

One can argue that since FPGAs can be reconfigured, there is little room for signal selection algorithms. However, it is not the case. Manual signal selection is based on trial-and-error can result in large number of iterations. Moreover during manual selection it is difficult to prevent closely correlated signals. Thus, signal selection algorithms can play a significant role in reducing debug iterations. The initial runs may be able to partition the susceptible regions. Signal selection may then be further applied on the susceptible region, thereby narrowing down the hunt.

With the increasing need for higher bandwidth and device usage in reconfigurable architectures, routing congestion can become very severe. Although, configurable logic blocks (CLBs) are flexible, certain applications may use a lot of routing resources around some CLBs. The routing problem aggravates with the high number of control signals or the high fanout nets. Hence, it is imperative to select signals considering the layout.

While early work on layout-aware signal selection [5] has proposed several promising ideas, layout-aware signal selection has not been explored for reconfigurable architectures. Therefore, in practice, it may not be possible to route a set of observation-friendly trace signals in FPGA-based designs. This paper makes two important contributions: i) it investigates signal selection problem in reconfigurable architectures, and ii) proposes a layout-aware signal selection framework for FPGA-based designs. The rest of the paper is organized as follows. Section II provides details of the related work. Section III describes our layout-aware signal selection algorithm. Section VI presents the experimental results. Finally, Section VII concludes the paper.

II. Related Work

We survey the existing approaches in three broad categories: i) signal selection for post-silicon validation of SoC designs, ii) early work on signal selection in FPGA, and iii) layout-awareness for various design automation problems.

A. Signal Selection for Post-Silicon Debug

Existing signal selection techniques can be broadly divided into three categories: metric-based selection, simulation-based selection and hybrid approaches. Metric-based methods [1], [2], [9] select signals by iterative addition of beneficial signals till the trace buffer is full. Although, metric-based algorithms have an advantage of being extremely fast compared to the simulation-based approach, their restoration performance can be inferior. Simulation-based trace signal selection starts with all the signals as observable, and then iteratively eliminates signals which have minimum impact on the restoration ratio [1] on removal [3], [4]. Simulation based methods provide higher state restoration ratio, but requires a longer runtime. Li and Davoodi [12] developed a hybrid of the metric and simulation-based methods to select trace signals. Hybrid approach first identifies top candidates using metric evaluation and then uses simulation to accurately evaluate the state restoration ratio for each candidate. These approaches [12], [13] did not consider routing congestion or reconfigurability constraints.

B. Signal selection in FPGAs

Hung and Wilton [7] suggested a new metric, "post-silicon debug difficulty" for signal selection in FPGA. For a set of selected signals, the metric indicates the number of circuit states that can be activated. The reasoning used here is that the designers do not go bit-by-bit to hunt down a bug. Rather, if the designer knows the state closely, then he may be able to brainstorm on the possible reasons of the bug. Hung and Wilton [8] also presented graph centrality as one of the metrics for faster selection of the signals. However, quantitative comparison between "post-silicon debug difficulty" and restorability based algorithms has not been studied. The restorability based algorithms have not been applied on FPGA. To the best of our knowledge, our work is the first attempt on applying the restorability based signal selection algorithms on FPGA-based designs.

C. Layout-aware Approaches in SoCs

Due to ever-increasing complexity of SoC designs, layout friendliness has been investigated by various researchers. Layout-awareness has been used as a key criteria in scan-chain reordering [6], fault pattern generation [11] and memory BIST synthesis [10]. Layout-aware signal selection for SoCs has been studied in [5]. However, layout has not been considered in the context of signal selection in reconfigurable architectures.

III. Overview

Figure 1 provides an overview of our proposed layout-aware signal selection. Layout of design is first evaluated to get distances of signals from the trace buffer. Signal selection module takes two inputs – the design (netlist) and layout (signal distance values) – and returns the selected signals. The following sections describe these two important steps: Manhattan distance calculation using the layout, and layout-aware signal selection.

[1]**Restoration ratio** is defined as the ratio between the number of signal states restored (including the traced states) and the number of states traced (observed). For example, if 4 signals are traced for 2 cycles (8 observed states) and the number of unknown signal states restored is 16, then the restoration ratio is $\frac{8+16}{8} = 3$. Higher restoration ratio implies better signal selection.

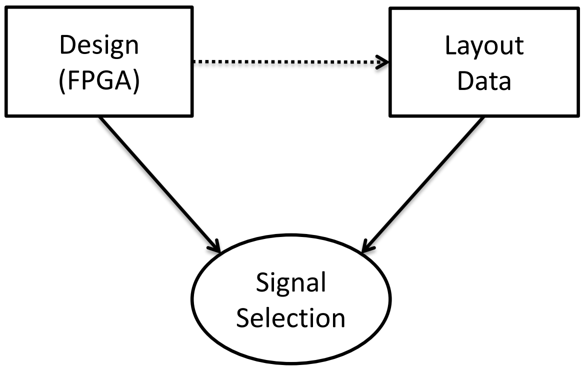

Figure 1. Layout-aware signal selection flow

IV. Manhattan Distance Calculation

One of the major challenges during placement and routing in a design is the routing congestion. Routing congestion is defined as the percentage of tracks blocked of the total tracks available for routing. Many metrics provide an evaluation criteria for layout congestion. Euclidean distance, Manhattan distance, and total wire-length may be used as a representative of the congestion in the design. Collection and interpretation of congestion information is non-trivial with the present tools. Using exact wire-length may be compute intensive.

Thus wire-length estimation techniques such as half-perimeter wire-length, squared-Euclidean distance, minimum Steiner-tree wire-length, minimum spanning tree wire-length or complete-graph wire-length may be used. We use wire-length estimate as the congestion criteria. All prospective selected flip-flops need to connect to the trace buffer in a star fashion, with the trace buffer at the center. Hence, half-perimeter wire-length is best suited for this purpose. Half-perimeter wire-length of any two connected nodes is equal to the Manhattan distance between them.

Manhattan distance is defined as the sum of absolute difference with respect to X and Y coordinate values of any two points. For example, the following equation provides the Manhattan distance between trace buffer (x_{tb},y_{tb}) and a signal (x_i,y_i).

$ManhattanDistance = (|x_{tb} - x_i| + |y_{tb} - y_i|)$.

For layout-awareness, the total Manhattan distance of all the selected flip-flops to the trace buffer needs to be minimized. Total Manhattan Distance (TMD) is given by following equation:

$\sum_{i=1}^{TraceBufferSize} (|x_{tb} - x_i| + |y_{tb} - y_i|)$

We use normalized Manhattan distance as a layout-awareness metric for different signals. Manhattan-distance (from the trace buffer) to all the prospective signals is calculated and normalized with respect to the farthest prospective signal.

Maximum Manhattan distance, among all prospective flip-flops from the trace buffer can be computed as:

$MD_{max} = max(MD_i)$

Similarly, normalized Manhattan distance is computed as:

$MD_I = MD_i/MD_{max}$

Normalized Manhattan distance is used in signal selection algorithms to prioritize signals based on the proximity to the trace buffer.

V. Layout-aware Signal Selection

The basic idea of our algorithm is that the normalized Manhattan distance values (computed in the previous section) are used with signal selection parameters like restorability and visibility to either eliminate or add a flip-flop into the set of flip-flops selected for the trace buffer. Algorithm 1 shows the major steps during layout-aware signal selection. It is important to note that our algorithm can be used on top of any existing signal selection procedure by invoking the specific procedure in step 6. In other words, the $signalSelection()$ subroutine in the algorithm can be replaced by any of the existing signal selection algorithms.

In the algorithm, the first step is to identify the trace buffer in the layout and get its coordinates. In step 2, coordinates of all the flip-flops are obtained and their Manhattan distance from the trace buffer is calculated. Step 3 computes the maximum value among the Manhattan distances. In step 4, the Manhattan distance of all the signals is normalized with respect to the maximum Manhattan distance calculated in step 3. Step 5 identifies the restoration ratio offered by all the signals. Restoration and Manhattan distance values of all the signals are used to select the signals in step 6. Finally, the algorithm returns the selected signals.

In the remainder of this section, we describe how step 6 can invoke two specific signal selection algorithms - simulation-based and metric-based signal selection.

A. Simulation-based Signal Selection

Simulation-based signal selection uses iterative elimination of less beneficial signals. In every iteration, simulations are used to determine the impact of eliminating a signal from the remaining signals. Signal with minimal impact on the restoration ratio is eliminated every cycle. Elimination continues till the number of elements in the observable set is equal to the trace buffer capacity. Layout awareness is added by either scaling the visibility (restorability) of the flip-flop (based on the normalized Manhattan distance), or by eliminating flip-flops based on the normalized Manhattan distance threshold. Signal with the minimum impact on visibility and minimum reduction in Manhattan distance is eliminated.

Figure 2 illustrates the normalized Manhattan distance cut-off (threshold) for signal selection. The rectangle in the middle of the rhombus represents a trace buffer location. The rhombus represents a region enclosed by a constant Manhattan distance. When normalized Manhattan distance threshold is applied, only the signals within the rhombus are taken into account. The two rhombi represent boundaries of two different threshold values of normalized Manhattan distance within which the signals are to be selected. RegionA and RegionB represent areas within normalized Manhattan distances of 0.2 and 0.4, respectively. All the signals in the design are considered when the threshold value is 1.

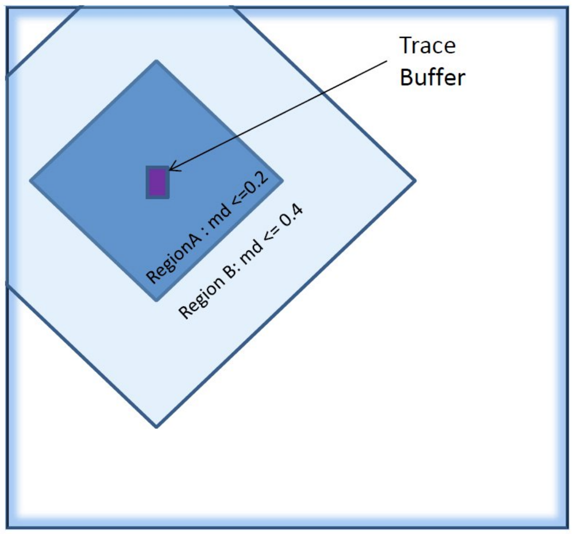

Figure 2. Manhattan distance based cut-off

There can be three approaches to layout-aware signal selection. One can be to push each eliminated signal on a stack, and continue eliminating the signals till all the signals are eliminated. Then, the selection may be done by popping the signals from the stack and selecting the signals which are within the normalized Manhattan distance threshold. The second approach can be to eliminate all the signals which do not meet the normalized Manhattan distance threshold in the first iteration itself, and subsequently run the default simulation-based algorithm on remaining signals. The third approach can be to use the normalized Manhattan distance as weight to prioritize elimination of signals which are far from the trace buffer. The results described in the next section used the first approach. The normalized Manhattan distance threshold is swept from 0.1 to 1 at steps of 0.1. Restoration ratio is evaluated for each set of signals. The set with the maximum reduction in Manhattan distance and minimum impact to the restoration ratio is selected.

Algorithm 1: Layout-aware Signal Selection

```
Inputs : design, layout, traceBuffer
Output: signals
  /* Determine (x,y) coordinates of
     the traceBuffer.                    */
1 $(x_{tb}, y_{tb}) = getLocation(traceBuffer, layout)$;
  /* Find Manhattan distance of all
     FFs from the traceBuffer            */
2 $\bigvee_{i=1}^{Signals} dist_i = |x_{tb} - x_i| + |y_{tb} - y_i|$ ;
  /* Find maximum Manhattan
     Distance.                           */
3 $max_{dis} = max(\bigvee dist_i)$ ;
  /* Normalize Manhattan distance        */
4 $\bigvee_{i=1}^{Signals} ndist_i = dist_i / max_{dist}$ ;
  /* Compute restoration ratio           */
5 $\bigvee_{i=1}^{Signals} restorability_i =$
  $getRestorability(signal, netlist)$;
6 $signals = signalSelection(ndist, restorability)$
  ;
Return: signals
```

B. Metric-based Signal Selection

Metric-based signal selection tries to maximize restoration ratio, while adding new signals to trace

Table I
COMPARISON WITH METRIC-BASED SIGNAL SELECTION FOR FPGAS

Benchmark	Restoration Ratio			Manhattan Distance		
	Basu & Mishra [1]	Layout-aware	% change	Basu & Mishra [1]	Layout-aware	% change
s9234*	2.66	2.97	11.65	6062	2063	-65.97
s13207*	8.30	9.58	15.42	6528	3133	-52.01
s35932	35.00	24.73	-29.34	8980	5778	-35.66
Average	15.32	12.43	-18.89	7190	3658	-49.12

*Restoration ratio not provided in [1] and had to be generated

buffer, until the trace buffer gets full. It can be modified to evaluate the combined impact of restoration ratio and normalized Manhattan distance. Separate weights are used for the restoration ratio and normalized Manhattan distance (ndist). The weight is then varied from 0 to 1 at a step size of 0.1.

Three possible modifications can be explored. One may be to iterative select a large number of signals, and put them on a FIFO queue in the order of selection. The final signals selected are the top elements in the queue which meet normalized Manhattan distance threshold. The second approach maybe to eliminate all the elements which do not meet the normalized Manhattan distance threshold at the beginning, and then run the default metric-based signal selection on the remaining signals. Another approach may be to give relative weight to the proximity of a signal to the trace buffer, and the restoration ratio offered by the signal. All the approaches have their own trade-offs. The first approach may end up selecting closely correlated signals and thereby giving lower restoration ratio. The second approach provides better results than the first, but it may still be desirable to have a few signals in the selection which are very good (although beyond the threshold). The third approach balances proximity of the signal and the restoration ratio offered by the signals. The results in the next section use the last approach.

VI. EXPERIMENTS

A. Experimental Setup

We have used ISCAS'89 benchmarks were used for the evaluation. We generated a BRAM using the *coregen* application of ISE and used it as the trace buffer. The design was then synthesized in the Xilinx framework and a $user\ constraints\ file$ (.ucf) was dumped from the floorplan editor. The constraints file has the locations of all the flip-flops and the BRAM cells used in the design. Using the coordinates in the constraints file, Manhattan distance of each and every flip-flop to the trace buffer was calculated and normalized. The signal selection was then run giving more precedence to the signals near the trace buffer. We compared the total Manhattan distance of all the selected signals from the trace buffer using existing algorithms and our proposed layout-aware signal selection. We used trace buffer width of 32.

B. Results

Signal selection evaluation was done using both metric-based and simulation-based approaches. Table I compares our layout-aware signal selection with metric-based [1] approach. Results show that on an average 49% reduction in the Manhattan distance is achieved with a restoration ratio penalty of 18.89%. Added routing-constraints caused perturbations in the evaluation, resulting in a higher restoration ratio in smaller benchmarks.

Table II presents data for simulation-based approach. The results show that on an average improvement of 23% in the Manhattan distance can be obtained with 17.6% degradation in restoration ratio.

VII. CONCLUSION

Post-silicon validation and debug are critical components of the SoC design methodology. FPGA provide a vital platform for validation at the prototyping stage. The challenge is to select beneficial signals for debug while considering the design constraints like routability. Existing approaches, though efficient at identifying good signals, overlook the design constraints, thereby selecting some signals which may not be routable. We

Table II
COMPARISON WITH SIMULATION-BASED SIGNAL SELECTION FOR FPGAS

Benchmark	Restoration Ratio			Manhattan Distance		
	Chatterjee et al. [3]	Layout-aware	% change	Chatterjee et al. [3]	Layout-aware	% change
s13207*	9.47	8.54	-9.82	5217	2320	-55.53
s35932	43.13	34.39	-20.26	7327	6261	-14.55
s9234	4.18	3.87	-7.42	6080	5744	-5.53
Average	18.93	15.60	-17.58	6208	4775	-23.08

*Restoration ratio not provided in [3] and had to be generated

developed techniques to incorporate layout-awareness in the existing set of algorithms towards identification of signals which are not only beneficial from the debug perspective but also from a routing perspective. Our technique further gives designer freedom to customize the weight for layout-awareness to suit the design needs. Our approach can be applied on top of the existing signal selection techniques such as metric-based [1] and simulation-based approaches [3]. Experimental results demonstrated that our approach can select layout-friendly signals with minor impact on restorability.

VIII. ACKNOWLEDGMENT

This work was partially supported by National Science Foundation (NSF) grant CCF-1218629.

REFERENCES

[1] K. Basu and P. Mishra. RATS: Restoration-aware trace signal selection for post-silicon validation. *IEEE Transactions on VLSI*, 21(4):605–613, April 2013.

[2] K. Rahmani and P. Mishra. Efficient signal selection using fine-grained combination of scan and trace buffers. *International Conference on VLSI Design*, pages 308-313, 2013.

[3] D. Chatterjee, C. McCarter, and V. Bertacco. Simulation-based signal selection for state restoration in silicon debug. In *ICCAD*, pages 595–601, 2011.

[4] K. Rahmani, P. Mishra and S. Ray. Scalable trace signal selection using machine learning. *International Conference on Computer Design*, 384-389, 2013.

[5] P. Thakyal and P. Mishra. Layout-aware Selection of Trace Signals for Post-Silicon Debug In *IEEE International Symposium on VLSI (ISVLSI)*, July 2014.

[6] P. Gupta, A.B. Kahng, I.I. Mandoiut, and P. Sharma. Layout-aware scan chain synthesis for improved path delay fault coverage. *IEEE Transactions on CAD*, 24(7):1104–1114, July 2005.

[7] E. Hung and S. Wilton. On evaluating signal selection algorithms for post-silicon debug. In *International Symposium on Quality Electronic Design*, 1–7, March 2011.

[8] E. Hung and S. Wilton. Scalable signal selection for post-silicon debug. *IEEE Transactions on VLSI*, 21(6):1103–1115, June 2013.

[9] H. Ko and N. Nicolici. Automated trace signals identification and state restoration for improving observability in post-silicon validation. In *Design, Automation and Test in Europe*, pages 1298–1303, March 2008.

[10] A. Kokrady, C. P. Ravikumar, and N. Chandrachoodan. Layout-aware and programmable memory BIST synthesis for nanoscale system-on-chip designs. In *Asian Test Symposium*, pages 351–356, 2008.

[11] J. Lee, S. Narayan, M. Kapralos, and M Tehranipoor. Layout-aware, ir-drop tolerant transition fault pattern generation. In *Design, Automation and Test in Europe*, pages 1172–1177, March 2008.

[12] M. Li and A. Davoodi. A hybrid approach for fast and accurate trace signal selection for post-silicon debug. In *Design, Automation Test in Europe*, pages 485–490, 2013.

[13] K. Rahmani, P. Mishra and S. Ray. Efficient trace signal selection using augmentation and ILP techniques. *IEEE International Symposium on Quality Electronic Design*, pages 148-155, 2014.