Efficient Trace Signal Selection using Augmentation and ILP Techniques *

Kamran Rahmani, Prabhat Mishra
Dept. of Computer and Information Sc. & Eng.
University of Florida, USA
{kamran, prabhat}@cise.ufl.edu

Sandip Ray
Strategic CAD Labs
Intel Corporation, USA
sandip.ray@intel.com

ABSTRACT

A key problem in post-silicon validation is to identify a small set of traceable signals that are effective for debug during silicon execution. Most signal selection techniques rely on a metric based on circuit structure. Simulation-based signal selection is promising but have major drawbacks in computation overhead and restoration quality. In this paper, we propose an efficient simulation-based signal selection technique to address these bottlenecks. Our approach uses (1) bounded mock simulations to determine state restoration effectiveness, and (2) an ILP-based algorithm for refining selected signals over different simulation runs. Experimental results demonstrate that our algorithm can provide significantly better restoration ratio (up to 515%, 51% on average) compared to the state-of-the-art techniques.

1. INTRODUCTION

The goal of post-silicon validation of an Integrated Circuit (IC) is to ensure that fabricated, pre-production silicon operates correctly under actual operating conditions with real application. It is a complex activity performed under aggressive schedules, representing more than 50% of the overall validation cost [10]. A fundamental challenge in post-silicon validation is limited observability and controllability. Due to limitations in the number of output pins and area and power overheads of internal trace buffers, only a few hundreds of the millions of internal signals in the design can be observed during silicon execution. Furthermore, in order for a signal to be observed, the design must be instrumented *a priori* with appropriate control hardware that routes the signal to an observation point. It is therefore crucial to identify trace signals that maximize design visibility under the constraints imposed by the observability restrictions.

Signal selection in current industrial practice is primarily manual, guided by the designer's experience and intuition: *e.g.*, more trace signals are selected from hardware blocks that route high message traffic, exhibit more bugs during pre-silicon validation, etc. In the absence of objective techniques for qualifying observability value, inadequacy of selected signals often manifest themselves only during silicon debug, typically in the form of observability holes that make it difficult to identify, diagnose, and root-cause an observed failure. However, this stage is too late for redesign of the debug infrastructure or selection of new trace signals (with associated routing hardware). Inability to adequately observe, validate, and debug at this stage results in costly escapes, complex work-arounds, or silicon respins.

Research in post-silicon validation has attempted to address this issue by developing algorithms to select trace signals through automatic analysis of RTL or gate-level designs. The idea is to select a set of signals S that maximizes *state restorability*, *i.e.*, the set of states that can be reconstructed based on the observation of the signals in S. Most existing signal selection approaches [9, 2, 6, 14] involve defining a metric based on the circuit structure, which is then used in a (typically greedy) selection process to evaluate a candidate signal set. More recently, Chatterjee *et al.* [4] have developed a simulation-based selection approach that performs better than pure structural analysis. However, their approach has drawbacks in computation overhead and restoration quality. Li *et al.* [8] proposed a hybrid approach combining metric-based and simulation-based techniques. However, restoration performance in this case depends on the input vector. Consequently, in order to use it as a selection metric, evaluation of several input sequences is necessary, which is not handled in any of the existing approaches [4, 8].

In this paper, we develop an approach that preserves the quality of simulation-based signal selection while ameliorating the computational bottlenecks. We achieve restoration quality better than simulation-based selection techniques while significantly improving runtime performance. Our experiments demonstrate improvement in restoration ratio as high as 515% (51% on average) over existing techniques.

Our approach has two components: (1) an iterative approach to signal selection based on mock simulations, and (2) a filtering scheme based on Integer-Linear Programming (ILP) to refine the selected set. The use of ILP for constraining a selection function is, of course, a well-known technique with applications to a number of applications in constraint-based optimizations including verification. Our key contributions in this paper are (1) the formulation of ILP as a filter mechanism on mock simulations given the objective of optimizing restoration ratio, and (2) a complete overall framework for signal selection based on this formulation. Our results demonstrate that the framework is viable as a practical signal selection strategy; we know of no other approach that achieves comparable restoration ratio under similar run-time performance.

The remainder of the paper is organized as follows. Section 2 provides the relevant background. Section 3 motivates the need for our approach using illustrative examples. We

*This work was partially supported by NSF grant CCF-1218629.

describe our approach in Section 4 followed by experimental results in Section 5. We discuss related work in Section 6 and conclude the paper in Section 7.

2. BACKGROUND

2.1 Post-silicon Validation Overview

Figure 1 provides an overview of post-silicon validation and debug process focusing on the role of signal selection. A modern IC design includes debug mechanisms such as *embedded logic analyzers* (ELA) to record values of internal design signals during silicon execution. An ELA consists of trigger and sampling units; trigger units are used to specify events that trigger recording initiation, and sampling units then record a small set of signals to the trace buffer for a specified number of cycles. The sampled signals can then be transferred from the trace buffer for off-chip analysis. In particular, the off-chip analysis can apply restoration algorithms on sampled signals to infer the values of other design signals and reconstruct internal states. The traced and restored signal values can be used together to detect design errors. To make this possible, the set of signals to be sampled is selected *a priori* by pre-silicon analysis of the design. Note that the number of sampled signals is restricted by the width of the trace buffer and typically represent a very small fraction of the internal signals in the design. Thus ideally one would like to choose the set of signals that permit maximum reconstruction of design states. Unfortunately, exhaustive exploration of all signal subsets to determine the most profitable signals is computationally intractable; most signal selection approaches [9, 2, 6, 4] involve developing heuristics that are efficient in practice while still yielding signals with good restoration.

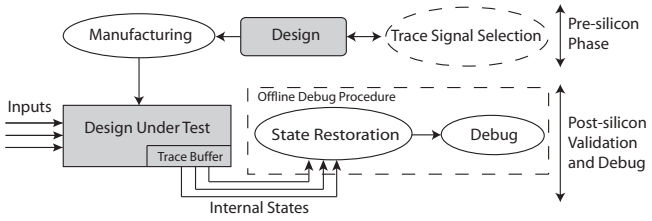

Figure 1: Simplified overview of post-silicon validation flow and role of trace signal selection. Signals selected through pre-silicon analysis are funneled to trace buffer from which silicon states are restored offline to assist in debug.

2.2 Signal Restoration

Restoration entails inferring values of untraced signal states from a sequence of traced signals sampled over a period of time. This is achieved using forward and backward propagation of signal values of circuit elements (*e.g.*, gates, latches, etc.). Figure 2 illustrates forward and backward restoration for common logical gates. Forward propagation involves reconstructing the output of a circuit element from traced inputs. For example, if one of the inputs of the *OR* gate is '1', the output value would be '1'. If all the inputs are known, the unknown output can be definitely determined. On the other hand, backward propagation involves inferring input values from the observed output. For example, if the output of the *OR* gate is '0', both of the inputs would be '0'. Backward reconstruction might fail in certain scenarios. For example, if the output of a 2-input *OR* gate is '1' and one of the input has a known value of '1', the other input still cannot be reconstructed. During signal value reconstruction, forward and backward restoration are repeated for all the gates in the circuit until no more states can be restored. Restoration Ratio (RR), defined below, is a popular metric for measuring the quality of a set of selected trace signals.

$$\text{Restoration Ratio} = \frac{\text{No. of traced and restored signals}}{\text{No. of traced signals}}$$

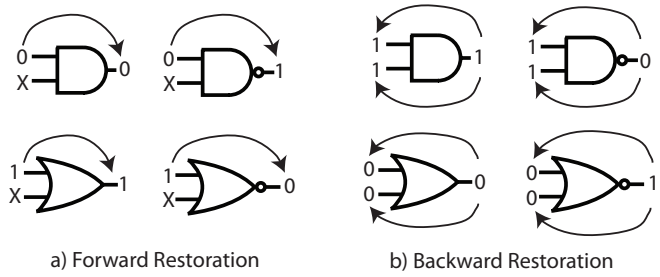

Figure 2: Basic restoration rules for common logic gates in a) forward restoration: knowledge of inputs can reconstruct the output; and b) backward restoration: knowing the output can restore the unknown inputs

Consider the simple circuit shown in Figure 3. Suppose that the width of the trace buffer is 2 (*i.e.*, only two signals can be traced at any clock cycle), and the trace buffer depth is 8 (*i.e.*, selected signals are traced for 8 cycles). Suppose that A and C are selected as trace signals. Table 1 shows the signal states that can be restored: 32 signal values can be restored while 16 are traced, yielding a restoration ratio of $(32 + 16)/16 = 3$.

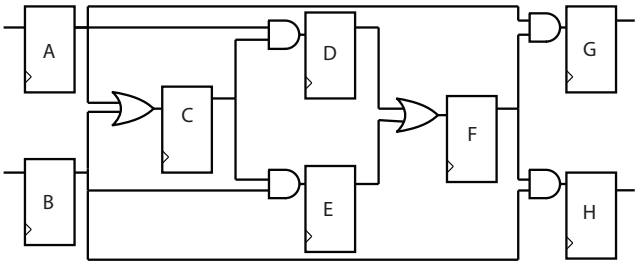

Figure 3: A simple circuit to illustrate restorability.

2.3 Signal Selection

The notion of restorability is based on the execution of the circuit on an input sequence; when the input sequence represents an on-field execution scenario for the circuit during post-silicon validation. However, signals must be selected *a priori* based on the circuit structure. Heuristics for selecting

Table 1: Illustration of restored signals for the simple circuit shown in Figure 3. The traced signals A and C are shaded. An X indicates that the signal value cannot be restored at that cycle using the known signal states.

Signal/Cycle	1	2	3	4	5	6	7	8
A	0	1	0	0	0	1	1	1
B	0	X	1	1	1	X	X	X
C	0	0	1	1	1	1	1	1
D	X	0	0	0	0	0	1	1
E	X	0	0	1	1	1	X	X
F	X	X	0	0	1	1	1	1
G	X	0	X	0	0	0	1	1
H	X	0	X	0	0	1	X	X

signals must take care to comprehend and encapsulate overlaps and interactions between different signals, and anticipate how such interactions might affect restorability on-field — an intrinsically difficult task.

Existing signal selection approaches can be classified in two categories, *structural* and *simulation-based*. Approaches in the first category use greedy heuristic to iteratively select signals optimizing a metric based on the circuit structure [6, 9, 2]. They are relatively efficient in computation speed, but have poor restoration quality compared to simulation-based algorithms. Simulation-based algorithms are based on the intuition that if a set of signals works well for some random input vectors then it is also likely to provide high state reconstruction on other inputs and therefore a high restorability ratio. In particular, Chatterjee *et al.* [4] showed that mock simulations are more effective in identifying trace signals than metrics based on the circuit structure. Their approach involves an iterative removal process. They start with a set of candidate signals which is initialized with all flip-flops. In each iteration, their algorithm attempts to remove one of the signals which appears least important based on simulation results. The process continues until the number of remaining candidates equals to the trace buffer width. Figure 4 illustrates the approach for a sample circuit with a total of 4 flip-flops and a trace buffer of width 2.

There are three key problems with the above approach. First, it may eliminate beneficial signals early. For example, in the first iteration, elimination of any signal can lead to the same outcome (100% restoration) since all the signals are present except one; it is possible that a set of beneficial signals may be eliminated in the first few iterations. Second, as restoration quality depends on the input vector, multiple simulation/restoration processes are needed to reduce the error variance in selection. Finally, their bottom-up approach starts with all the flip-flops and eliminates one or multiple flip-flops at each iteration. This increases the number of simulation/restoration processes significantly which makes their approach computationally expensive. In this paper, we present a top-down simulation-based selection approach to address these challenges.

3. ILLUSTRATION OF OUR APPROACH

Our approach is inspired by simulation-based signal selection techniques, but includes a refinement technique to address the weaknesses of previous simulation-based approaches described above. Before presenting the technical details of our approach, we motivate it by comparing its results using illustrative examples with state-of-the-art metric-based and simulation-based approaches, *viz.*, Basu *et al.* [2] and Chatterjee *et al.* [4]; these experiments expose some key features of our approach which we then discuss.

For the circuit in Figure 3, Basu *et al.* select signals A and C, thus yielding the restoration ratio of 3 as shown in Table 1. On the other hand, both our approach and the simulation-based approach of Chatterjee *et al.* selects signals A and B. The corresponding restorability calculations are shown in Table 2. From the table, 40 states are restored from tracing 16 states, yielding a restoration ratio of 3.5.

Table 2: Restored signals for circuits in Figure 3 when signals A and B are traced. The signals A and B are selected by our signal selection algorithm when applied to the circuit design.

Signal/Cycle	1	2	3	4	5	6	7	8
A	0	1	0	0	0	1	1	1
B	0	0	1	1	1	0	0	0
C	X	0	1	1	1	1	1	1
D	X	0	0	0	0	0	1	1
E	X	0	0	1	1	1	0	0
F	X	X	0	0	1	1	1	1
G	X	0	X	0	0	0	1	1
H	X	0	0	0	0	1	0	0

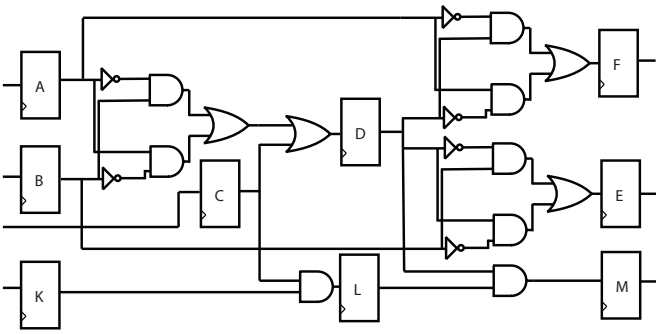

Figure 5: Example circuit to compare our approach with Chatterjee *et al.* [4]

On the other hand, to illustrate the distinction between our approach and Chatterjee *et al.* consider the circuit in Figure 5. For a trace buffer width of 2, Chatterjee *et al.* produce signals B and C. From Table 3, this leads to a restoration of 13 states from a total of 16 traced states, yielding a restoration ratio of 1.81. On the other hand, our approach selects signals C and K. From Table 4, this allows restoration of 18 states from 16 traced states, resulting in a restoration ratio of 2.13.

It is illuminating to understand the source of the differences between the different approaches on these simple examples. The high restoration ratio achieved by both our

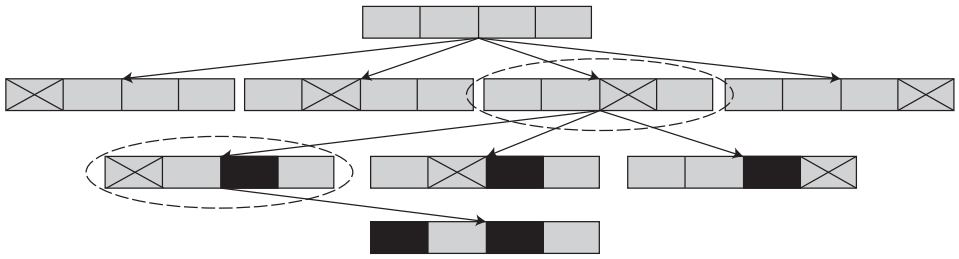

Figure 4: Trace signal selection of Chatterjee *et al.* [4] for a sample circuit with 4 flip-flops and a trace buffer of width 2. Each row illustrates an iteration of the algorithm. In each step, a flip-flop whose elimination results in minimum impact on restoration performance is removed. The black boxes show the eliminated flip-flops in previous iterations, while the crosses illustrate the flip-flop being evaluated in current iteration.

Table 3: Restored signals from [4] for the circuit in Figure 5.

Signal/Cycle	1	2	3	4	5	6	7	8
A	X	X	X	X	X	X	X	X
B	0	0	1	1	0	0	1	0
C	0	1	1	1	0	1	1	0
D	X	X	1	1	1	X	1	1
E	X	X	X	0	0	1	X	0
F	X	X	X	X	X	X	X	X
K	X	X	X	X	X	X	X	X
L	X	0	X	X	X	0	X	X
M	X	X	0	X	X	X	0	X

Table 4: Restored signals using our method for the circuit in Figure 5.

Signal/Cycle	1	2	3	4	5	6	7	8
A	X	X	X	X	X	X	X	X
B	X	X	X	X	X	X	X	X
C	0	1	1	1	0	1	1	0
D	X	X	1	1	1	X	1	1
E	X	X	X	X	X	X	X	X
F	X	X	X	X	X	X	X	X
K	0	0	0	0	0	0	1	1
L	X	0	0	0	0	0	0	1
M	X	X	0	0	0	0	0	0

approach and that of Chatterjee *et al.* for the circuit in Figure 3 represents a general trend of superior signal quality achieved by simulation-based selection techniques; our observations here match the conclusions of Chatterjee *et al.* as well. The comparison with Chatterjee *et al.* for the circuit in Figure 5 is more interesting. Their approach is based on *greedy elimination*: starting with the set of all signals, they iteratively remove signals one at a time. In each iteration the objective is to select a candidate signal whose elimination minimizes the number of states which become unrestorable as a result; this signal is then eliminated and the algorithm iterates. The problem with this approach is that the candidate computation assumes that all the remaining signals are available for state restoration, an assumption that is flawed precisely by virtue of the iterative elimination algorithm itself. Thus it is possible that a profitable signal s is eliminated in an early iteration when the states reconstructable from s can also be restored by other signals available at that iteration; however, these states can no longer be reconstructed when a subsequent iteration eliminates other signals. In the example, the signal K is eliminated in an early iteration since the states restorable from K can be restored without K as long as the signal L is available; however, when a subsequent iteration eliminates L as well, the set of states that can be restored gets drastically reduced.

4. AUGMENTATION-BASED SELECTION

Our algorithm exploits the advantages of simulation-based signal selection while avoiding the drawbacks discussed above. Figure 6 illustrates the framework. We apply an iterative approach based on augmentation rather than elimination. In particular, we maintain a set S of signal candidates (initially empty), which we "grow" at each iteration by identifying the most promising signal based on mock simulations; the objective is to maximize the set of states that can be restored from the signals in the candidate set. The key observation here is that in this approach restorability of the candidate set is never over-estimated at any iteration since each member of S is guaranteed to be in the final trace selection set. Furthermore, note that the number of iterations in this approach is bounded by buffer size, which is very small precisely because of the observability limitation in post-silicon validation. On the other hand, the number of iterations in the elimination-based selection is bounded by the number of signals which can be large. Thus our approach achieves much better run-time performance compared to the elimination-based selection.

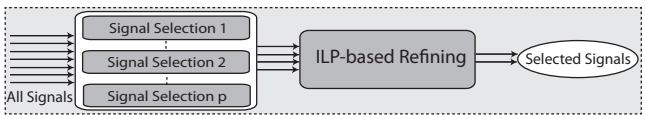

Figure 6: Proposed signal selection process. Simulation-based signal selection is applied p times to the circuit. The result of all the runs are combined and is refined using an ILP-based method. The output of the ILP optimization is the set of selected signals.

Our second observation is that any selection algorithm

based on random simulation is susceptible to perturbations based on the randomness in the input vectors. To eliminate the influence of randomness, our approach makes use of multiple simulation runs using an ILP-based refinement algorithm to consolidate the results from these runs.

4.1 Augmentation-based Signal Selection

We first describe our augmentation-based selection algorithm; we will discuss the ILP-based refinement in the next subsection. Algorithm 1 outlines the major steps of the signal selection process. The inputs of the algorithm are the *circuit, trace buffer width (w),* and *the number of cycles in mock simulations (c)*. To understand the workings of the algorithm we need two key concepts: *restoration influence* and *restoration difference*.

Given a set of candidate signals s, an input vector I, and the number of simulation cycles c, we define the *Restoration Influence* $RI(s, I, c)$, as the total number of states that can be restored if we do a mock simulation over c cycles using input vector I and the signals set S. The *restoration difference* between two candidates s_1 and s_2 with respect to I and c, denoted by $RD(s_1, s_2, I, c)$, is then given by the following formula:

$$RD(s_1, s_2, I, c) = RI(s_1, I, c) - RI(s_2, I, c)$$

Algorithm 1 Signal selection

```
1: procedure SelectSignals(circuit, w, c)
2:     Create list of selected signals S        ▷ initially empty
3:     while |S| < w do
4:         Generate a random input vector I
5:         for each flip-flop f that is not in the S do
6:             Calculate RD(S ∪ {f}, S, I, c)
7:         end for
8:         Find flip-flop f with maximum RD. If two or more
       flip-flops have same RD, find the one with higher
       connectivity
9:         Add f to the list S
10:    end while
11:    return S
12: end procedure
```

Informally, for a given c-cycle mock simulation I, the restoration difference between two candidate signal sets s_1 and s_2 measures the observability improvement achieved by selecting s_2 over s_1. In particular, if $s_2 = s_1 \cup \{f\}$ for some design signal f, then it measures the observability improvement achieved by augmenting s_1 with f. Algorithm 1 is a greedy algorithm that uses this metric to iteratively grow the set S of currently selected signals. At each iteration, it (1) performs a new simulation for c cycles using a random input vector I, (2) computes the restoration difference between S and $S \cup \{f\}$ for each design signal f, and (3) augments S with the signal that maximizes the restoration difference. If two or more signals have identical restoration difference, then the tie is broken in favor of the signal that has the highest connectivity.[1] The process is continued until w signals have been selected.

[1]The connectivity of a flip-flop is the number of flip-flops connected to it through other combinational gates in both backward and forward directions.

4.2 ILP Optimization

Experiments show that most of the selected trace signals are identical in different runs of our signal selection. However, in any simulation-based signal selection approach, signals may be different in different runs depending on generated random input vector seed. The goal of our refinement algorithm is to eliminate the influence of randomness and also to cover more states of the circuit through selected signals. To do so, we use multiple runs of the signal selection algorithm which are then processed by ILP to select the best signal set among all outcomes.

Algorithm 2 ILP Matrices Initialization

```
1: procedure InitializeMatrices(circuit, w, c, p)
2:     Create S[1..p][1..w] and R[1..p][1..w]
3:     Create k and j, initialize to 1
4:     Create list of all selected signals A ▷ initially empty
5:     while k <= p do
6:         T = Signal selection algorithm with (circuit, w,
       c)
7:         Generate a random input vector I
8:         j = 1
9:         for each flip-flop f in the T do
10:            S[k][j] = f
11:            A = A ∪ {f}
12:            RD_f = RD(T, T − {f}, I, c)
13:            R[k][j] = RD_f
14:            j++
15:        end for
16:        k++
17:    end while
18:    return A, S, and R
19: end procedure
```

To perform the refinement, we first create ILP formulation matrices from the signal selection algorithm. Algorithm 2 outlines the steps involved. The inputs of the algorithm are the *circuit*, *trace buffer width (w)*, *the number of cycles in mock simulations (c)*, and *refinement precision (p)*. The refinement precision specifies the number of runs of the signal selection algorithm used in the refinement process. The algorithm returns two matrices S and R, and a set A, which are then used as the basis of the ILP optimization. A is the set of all flip-flops selected in the p runs of our selection algorithm. The matrices S and R record the importance of the selected flip-flops in state reconstruction: $S[k][j]$ records the j-th selected flip-flop in the k-th run of our selection algorithm; $R[k][j]$ records the number of states that is lost in the mock simulation corresponding to the k-th run if $S[k][j]$ is removed from the final selected set. The algorithm executes p runs of our selection algorithm, filling out the entries $S[k][j]$ and $R[k][j]$ at the k-th run. Recall that the perturbation caused to the selection set is typically small. Thus, for the set T of flip-flops computed in the k-th run and any $f \in T$, most of the signals in $T - \{f\}$ end up in the final selected signal set; thus, the value of $R[k][j]$ is a reliable estimate of the importance of flip-flop $S[k][j]$.

Once the required matrices are initialized, we can model our refinement process as an ILP optimization problem in a fairly standard manner. For each flip-flop in A, we create a variable which can be 0 or 1. $A_i = 1$ indicates that A_i

is eliminated; $A_i = 0$ indicates that it is not removed and therefore exists in final trace signals set. Note that since A is a cumulative superset of all selected flip-flops during p runs, for each $1 \leq i \leq p$ and $1 \leq j \leq w$, we have $S[i][j] \in A$. Equation 1 shows the objective function which should be minimized. L_i is the number of states that is lost in i_{th} run, based on signal assignments in A. The aim is to minimize the total number of lost states in all the runs.

$$min : \sum_{i=1}^{p} L_i \tag{1}$$

Equation 2 shows how L_i is calculated. Recall that $S[i][j]$ is the assignment of signal j in A (which is 0 or 1), and $R[i][j]$ is the number of states that is lost in i-th run if j-th signal is removed (i.e., is equal to 1). Therefore, L_i is the total number of states that is lost due to removed flip-flops of i-th run.

$$L_1 = \sum_{i=1}^{w} S[1][i] * R[1][i],\ ...,\ L_p = \sum_{i=1}^{w} S[p][i] * R[p][i] \tag{2}$$

The constraints of ILP optimization problem are shown in Equation 3. Recall that A is the superset of all selected signals in different runs. However, $|A|$ may be larger than w as some selected signals may be different during signal selection runs. It means $|A| - w$ signals must be removed from A. These signals are removed in such a way that the total number of lost states in all runs is minimized. The remaining w flip-flops in A which are assigned to 0 correspond to the final trace signals set.

$$\sum_{i=1}^{|A|} A_i = |A| - w$$

$$A_1, A_2, ..., A_{|A|} \in \{1, 0\} \tag{3}$$

4.3 Complexity and Scalability

Simulation of large industrial designs incurs high cost in running time. Indeed, simulation time is the primary bottleneck in the usability of simulation-based signal selection on large-scale designs. Therefore, a good metric of the complexity of such algorithms is the number of mock simulations required in the computation. Note that although our approach involves ILP-based optimization, the running time for solving the ILP in practice is still negligible compared to the time for mock simulations. The reason is that the perturbation caused by randomization in simulations in practice to the selected set of signals is small, so that there is a large overlap between the signals selected at different runs. Thus, the selected set A of flip-flops over all the different runs in our ILP-based refinement is of the order of the width of the trace buffer, independent of the number p of iterations of the selection algorithm actually performed. Consequently, we compute the complexity of our algorithm in terms of the number of required mock simulations.

Assume that there are N flip-flops in the circuit and the trace buffer width is w. Number of needed simulation in each run of signal selection algorithm is $N + (N-1) + ... + (N - w + 1)$. Note that $N >> w$ for large circuits, since the trace buffer size is bounded by the observability limitations. The complexity of Algorithm 1 is thus $\theta(Nw)$. Algorithm 2 consists of a main loop which runs signal selection algorithm followed by w additional simulations to fill in matrix R. Consequently, each iteration needs $\theta(Nw) + \theta(w) = \theta(Nw)$ simulations. Therefore, the complexity of our algorithm is $p * \theta(Nw) = \theta(Npw)$. However, our experiments show that in practice $p << N$ is enough to cover most of the input vectors. Consequently, in most cases, our algorithm requires fewer simulations than the previous simulation-based approach of Chatterjee *et al.* [4], which has a complexity of $O(N^2)$ — with the lower bound of $\Omega(N^2/d_{step})$ which is still computationally expensive since $N >> d_{step}$ in large industry-scale circuits ($d_{step} = 50$ in their experiments). On the other hand, the hybrid approach [8] uses simulation/restoration computation only for top k% of the candidate signals, (where $k = 5$ in their experiments). The complexity of their approach is $O(kwN)$ where w is the trace buffer width. Note that once the parameters are fixed, both our approach and the hybrid approach have the same asymptotic complexity $\theta(N)$), with different constant coefficients.

In addition, not only all the simulations in each iteration of our selection algorithm are independent, but the iterations of initialization algorithm are also independent tasks. This makes our approach scalable for very large industry-level circuits by running them in parallel in a multi-processor environment.

5. EXPERIMENTS

5.1 Experimental Setup

In order to investigate the effectiveness of our proposed approach, we have developed a cycle-accurate simulator for ISCAS'89 benchmarks using C++. Our simulator also conducts restoration in both forward and backward directions. The simulator iterates on the unknown signals queue and attempts to restore them leveraging both forward and backward techniques. This process terminates when it is not possible to restore any more states. In addition, we checked the correctness of our simulator by comparing its output with the output of Verilog simulation of the identical circuits using *Icarus Verilog* [15]. We also used *lp_solve* 5.5 [1] to solve the ILP optimization part of our approach.

In the results reported below, the comparisons with related work [4, 8] are based on our implementation of their results. The reason is that their reported results used their own synthesized/optimized version of the ISCAS'89 benchmarks, while we used the standard, publicly available versions. Moreover to make the comparison fair for comparing restorability, identical input vectors should be used in all the approaches. We used the same parameters $c = 64$ and $PT = 95\%$ as reported in Chatterjee *et al.* [4]. In addition, we used the same parameters $M = 64$, $k = 5\%$, and an initialization simulation of 10K cycles as reported in Li *et al.* [8]. We also used $c = 32$ and $p = 6$ for our approach in our experiments. Our experiments demonstrate that restoration ratio shows no improvement for $p > 6$ in the set of used benchmarks. After signal selection and for reporting the restoration ratios, we fed the simulator with 100 sets of random input vectors and noted the average restoration ratios for the selected set of signals. However, we forced the circuits to operate in their normal mode by fixing the relevant control (reset) signals, while assigning random values to all the other inputs. The control signals include active low reset signals *RESET* in *s35932* and *g35* in *s38584* which was set to 1 in our experiments. To make the comparison fair,

Table 5: Restoration ratios using our approach compared with existing selection approaches

Circuit	#Flip-flops	Buffer Width	Simulation-based [4]	Hybrid [8]	Our Approach	Improvement over the best
s5378	179	8	**13.41**	13.32	14.63	9.1%
		16	**7.35**	7.26	9.26	26.0%
		32	**4.47**	4.27	5.11	14.3%
s9234	228	8	13.98	**14.58**	15.97	9.5%
		16	8.30	**8.55**	9.32	9.0%
		32	4.46	4.46	5.53	24.0%
s15850	597	8	26.33	**27.38**	45.89	67.6%
		16	19.89	**20.65**	25.82	25.0%
		32	13.19	13.19	13.97	5.9%
s13207	669	8	35.52	**39.21**	52.22	33.2%
		16	20.13	**22.47**	34.89	55.3%
		32	11.25	**12.52**	16.37	30.8%
s38584	1452	8	19.73	**25.87**	159.1	515.0%
		16	28.39	**29.01**	48.39	66.8%
		32	32.45	**34.62**	44.46	28.4%
s38417	1636	8	29.23	**51.01**	53.47	4.8%
		16	17.02	**19.22**	26.87	39.8%
		32	**15.14**	13.25	17.22	13.7%
s35932	1728	8	132.00	**139.52**	185.1	32.7%
		16	67.45	**71.36**	93.2	30.6%
		32	34.63	**35.08**	47.13	34.4%

these random input vectors are different from those which are used in signal selection process.

5.2 Results

5.2.1 Restoration Quality

Table 5 presents the restoration ratios of our approach compared with previous techniques [4, 8] using the ISCAS'89 benchmarks. The trace buffer sizes used in our experiment are $8 \times 4k$, $16 \times 4k$, and $32 \times 4k$. The corresponding restoration ratio for each technique is reported. The last column indicates the percentage of improvement using our approach compared with the best (shown in bold) result provided by existing approaches. The results indicate that our approach performs significantly better in most cases; in particular we achieve improvement in restoration performance is up to 515% (in *s38584*). Note however that the restoration ratio is heavily dependent on the circuit structure, and such high restoration in isolated cases may be an anomaly. Nevertheless, our approach performs better on most cases, with an improvement of 51.23% in restoration quality. Compared to original simulation-based signal selection [4], our fine-grained pruning reduces the chance of removing effective flip-flops prior to selection itself. On the other hand hybrid selection [8] incorporate simulations for only top 5% of the candidate flip-flops, which sacrifices the precision of the selection process; our approach performs better by addressing this weakness through refinement.

5.2.2 Signal Selection Time

In addition to restoration ratio, we compared the runtime between our approach and Chatterjee *et al.* Figure 7 illustrates the selection time of our approach compared and normalized to [4] using different ISCAS'89 benchmarks. Since selection complexity of [4] is $O(N^2)$ ($\Omega(N^2/50)$ in the best case) and ours is $\theta(Npw)$, as expected, for smaller benchmarks where pw is comparable to or larger than N our approach takes comparable time or longer than [4] (for example *s5378* benchmark and buffer width of 16 and 32 respectively). However, our approach demonstrates consistent speed-up for larger benchmarks (*s15850, s13207, s38584, s38417, and s35932*). The reason is that even after pruning phase of [4], number of conducted simulations in [4] is significantly larger than our approach. In fact, once p and w are fixed, our approach grows linearly with respect to number of flip-flops in circuit. In short, our approach not only produces better restoration quality, but also it is more feasible in terms of selection runtime in large circuits. This makes our approach a better fit for large-scale industry circuits where $N >> pw$. Our signal selection time speed-up is up to 127.6X (in *s38417* with buffer width of 8) and 12.9X on average. Note however that the hybrid approach of Li *et al.* [8] also reports significant speed-up over simulation-based techniques. However, their runtime results are reported for a multithreaded implementation running on a specific quad-core machine, and are difficult to reproduce in our framework to provide a fair comparison.

6. RELATED WORK

Various signal selection techniques [6, 9, 2] used restorability calculations to determine the profitable signals. Prabhakar *et al.* [11] proposed a logic implication based trace signals selection technique that uses the primary inputs in restoration process. The use of scan chains in post-silicon debug has been extensively studied in [16, 5]. Various approaches [7, 3, 12] divided trace buffer bandwidth into two parts, one for the trace signals and the other one for the scan signals.

Chatterjee *et al.* [4] demonstrated that simulation-based signal selection is a promising approach. However, their approach requires $O(N^2)$ simulations (where N is the number

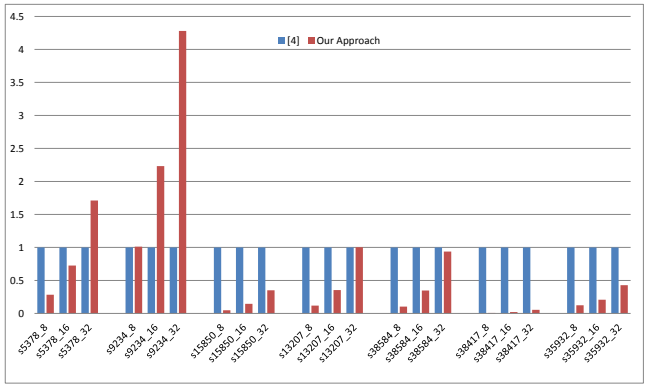

Figure 7: Selection times of our approaches compared and normalized to [4]

of flip-flops), making their approach computationally infeasible for large circuits. To address this issue, they propose a signal pruning phase as a pre-processing process. The pruning can be viewed as a faster but less precise run of the algorithm itself. It reduces the initial candidate flip-flops set but still requires long signal selection time. In addition, it may sacrifice the signal selection quality. Li *et al.* [8] proposed a hybrid (metric-based and simulation-based) signal selection technique; however, this approach uses simulation for a small fraction of the signals and thereby sacrifices restoration performance. Finally, very recently, we show how to make use of machine learning techniques to ameliorate the cost of simulations [13].

7. CONCLUSION

Post-silicon validation is an expensive phase in the production of integrated circuits, and crucially depends on signal selection to effective use of the limited available observability. Thus it is critical to develop signal selection techniques that provide high state reconstruction and can scale to large industrial designs. Existing metric-based signal selection techniques are computationally efficient, but often yield signals with poor restorability; simulation-based techniques, while superior in restoration quality suffer from major computational drawbacks.

We presented a simulation-based signal selection technique that yields signals with higher restorability than current approaches while still being computationally efficient. Our key contribution is the observation that simulation-based signal selection can be significantly improved by augmentation through ILP-based refinement, together with the insights to smoothly integrate the augmentation phase into the selection framework resulting in a unified scalable infrastructure. Our experiments demonstrate that our approach provides up to 515% (51.23% on average) improvement in restoration ratio compared to existing signal selection techniques.

8. REFERENCES

[1] lp_solver. http://lpsolve.sourceforge.net/5.5.

[2] K. Basu and P. Mishra. Rats: Restoration-aware trace signal selection for post-silicon validation. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 21(4):605–613, 2013.

[3] K. Basu, P. Mishra, and P. Patra. Efficient combination of trace and scan signals for post silicon validation and debug. In *ITC*, pages 1–8, 2011.

[4] D. Chatterjee, C. McCarter, and V. Bertacco. Simulation-based signal selection for state restoration in silicon debug. In *Computer-Aided Design (ICCAD), 2011 IEEE/ACM International Conference on*, pages 595 –601, nov. 2011.

[5] R. Datta, A. Sebastine, and J. Abraham. Delay fault testing and silicon debug using scan chains. In *Test Symposium, 2004. ETS 2004. Proceedings. Ninth IEEE European*, pages 46 – 51, may 2004.

[6] H. F. Ko and N. Nicolici. Algorithms for state restoration and trace-signal selection for data acquisition in silicon debug. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 28(2):285 –297, feb. 2009.

[7] H. F. Ko and N. Nicolici. Combining scan and trace buffers for enhancing real-time observability in post-silicon debugging. In *Test Symposium (ETS), 2010 15th IEEE European*, pages 62 –67, may 2010.

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

[9] X. Liu and Q. Xu. Trace signal selection for visibility enhancement in post-silicon validation. In *Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09.*, pages 1338 –1343, april 2009.

[10] A. Nahir, A. Ziv, R. Galivanche, A. J. Hu, M. Abramovici, A. Camilleri, B. Bentley, H. Foster, V. Bertacco, and S. Kapoor. Bridging pre-silicon verification and post-silicon validation. In *DAC*, pages 94–95, 2010.

[11] S. Prabhakar and M. Hsiao. Using non-trivial logic implications for trace buffer-based silicon debug. In *Asian Test Symposium, 2009. ATS '09.*, pages 131 –136, nov. 2009.

[12] K. Rahmani and P. Mishra. Efficient signal selection using fine-grained combination of scan and trace buffers. In *VLSI Design (VLSI Design), 2013 26th International Conference on*, jan. 2013.

[13] K. Rahmani, P. Mishra, and S. Ray. Scalable Trace Signal Selection Using Machine Learning. In *Proceedings of the 31st IEEE International Conference on Computer Design (ICCD 2013)*, pages 384–389. IEEE, 2013.

[14] H. Shojaei and A. Davoodi. Trace signal selection to enhance timing and logic visibility in post-silicon validation. In *Proceedings of the International Conference on Computer-Aided Design*, ICCAD '10, pages 168–172, Piscataway, NJ, USA, 2010. IEEE Press.

[15] Stephen Williams. Icarus Verilog. http://iverilog.icarus.com/.

[16] G. Van Rootselaar and B. Vermeulen. Silicon debug: scan chains alone are not enough. In *Test Conference, 1999. Proceedings. International*, pages 892 –902, 1999.