Efficient Signal Selection using Fine-grained Combination of Scan and Trace Buffers

Kamran Rahmani and Prabhat Mishra
Computer and Information Science and Engineering
University of Florida, Gainesville FL 32611-6120, USA
Email: {kamran, prabhat}@cise.ufl.edu

***Abstract*—Post-silicon validation is a critical part of integrated circuit design methodology. The primary objective is to detect and eliminate the bugs that has escaped pre-silicon validation phase. One of the key challenges in post-silicon validation is the limited observability of internal signals in manufactured chips. Leveraging on-chip buffers addresses this issue by storing some of the internal signal states during runtime. A promising direction to improve observability is to combine trace and scan signals - a small set of trace signals are stored every cycle, whereas a large set of scan signals are dumped across multiple cycles. Existing techniques are not very effective since they explore a coarse-grained combination of trace and scan signals. In this paper, we propose a fine-grained architecture that addresses this issue by using various scan chains with different dumping periods. We also propose an efficient algorithm to select beneficial signals based on this architecture. Our experimental results demonstrate that our signal selection algorithm can improve restoration ratio up to 91% (32.3% on average) compared to existing trace only techniques. Our approach also shows up to 116% improvement (54.7% on average) compared to techniques that allows combination of trace and scan signals.**

I. Introduction

Pre-silicon validation techniques are used to verify the functionality of a design before the manufacturing phase. Due to drastic growth of design complexity and also shrinkage of time-to-market window, it is not always possible to capture all the design bugs using these techniques. Post-silicon validation is used to capture these escaped bugs.

Figure 1 illustrates an overview of post-silicon validation and debug process. Signal selection and trace buffer design are done in pre-silicon phase. If an error occurred during post-silicon validation phase, the traced values of internal signals are dumped. During the debug process both dumped signals and restored signals are used to locate the error. The number of signals that can be traced is limited to trace buffer width. Therefore, the primary objective is to select a small set of profitable signals that can maximize restoration performance. A major challenge in optimal signal selection is that even for small circuits, there are numerous signal combinations. For example, s35932 circuit of ISCAS'89 benchmarks suite has 1728 flip-flops. If the trace buffer width is 32, we need to choose 32 signals out of the total 1728 flip-flops. It is easy to observe that there are more than 10^{69} such combinations. This makes exhaustive exploration computationally intensive and quite impractical[1]. Several trace signal selection techniques were proposed over the years [1], [2], [3], [4]. These algorithms attempt to select a set of promising trace signals to maximize the number of states that can be restored.

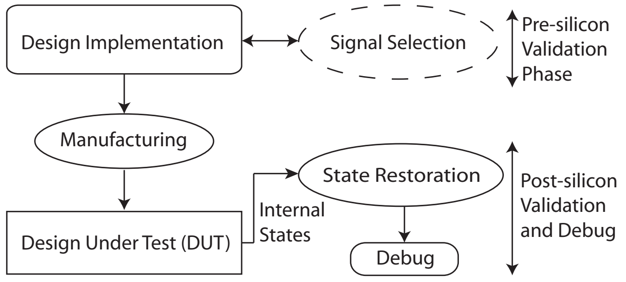

Fig. 1. Overview of post-silicon validation

To improve the observability further, various approaches [5], [6] explored a profitable combination of trace and scan signals. The basic idea is to divide the trace buffer (width) into two parts. The first part stores the trace signals and the second part stores the scan signals. The idea is to select a very small set of important control signals that would be traced every cycle. The remaining slots of the trace buffer will be filled with a portion of a large set of scan signals that would be dumped across several cycles. Existing approaches divides signals into two extreme categories - very important and less important. They lose opportunity from scenarios where some other partitioning is useful such as very important, important, less important, and so on.

It would be beneficial if we divide the signals in a large number of categories in terms of dumping period. This enables us to select a promising signal with a profitable dumping period. In this paper, we propose an efficient fine-grained architecture that shares the trace buffer bandwidth between several scan chains with different dumping periods. Our signal selection algorithm assigns the signals to different scan chains

[1]If each simulation for evaluating one combination takes only 1 second, more than 10^{60} years is needed in order to find the best combination in s35932 circuit.

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

based on our new restoration power (RP) metric in order to maximize the number of states that can be restored.

The rest of the paper is organized as follows. Section II describes related works in post-silicon debug and signal selection. Section III presents the background and motivation for our approach. Section IV describes our architecture and proposed signal selection algorithm. Section V presents our experimental results. Finally, Section VI concludes the paper.

II. Related Work

Limited observability of internal signals is the primary issue in post-silicon validation. Once the values of internal signals are determined, they can be analyzed using various proposed algorithms. Caty et al. [7] proposed failure propagation tracing technique to locate the errors in the circuit. De Paula et al. [8] proposed a formal method for post-silicon debug. Nataraj et al. [9] proposed physical probing techniques for post-silicon debug. Several design for debug techniques like embedded logic analyzer [10] and shadow flip-flops [11] have been proposed over the years.

Recently, trace buffers have been widely used in post-silicon debug. Trace buffers are used to store the state of some selected internal signals. The rest of the signals are obtained using restoration algorithms for traced signals. The primary problem is which of the signals need to be traced to maximize the number of states that can be restored. Ko et al. [3] and Liu et al. [1] have proposed efficient signal selection algorithms based on partial restorability. Basu et al. [2] improved their methods by proposing an efficient algorithm that selects signals based on their total restorability. Recently, Chatterjee et al. [4] proposed a simulation-based signal selection algorithm to further improve restoration performance. Prabhakar et al. [12] proposed a logic implication based trace signals selection method.

The use of scan chains in post-silicon debug has been extensively studied in [13], [14]. Ko et al. [5] proposed an architecture that divides trace buffer bandwidth into two parts, one for the trace signals and the other one for the scan signals. In order to find the most beneficial partitioning they proposed an exhaustive exploration. However, exhaustive exploration is not practical in real designs with large number of flip-flops. Basu et al. [6] addressed these issues by proposing an efficient algorithm that chooses trace and scan signals based on connectivity graph of flip-flops. However, both of these techniques divides the signals in two extreme categories. One set of signals are traced every cycle. The other signals are dumped in a relatively large period. They do not consider other profitable fine-grained scenarios. In this paper, we propose a promising fine-grained architecture that shares the trace buffer bandwidth between several scan chains with different lengths.

III. Background and Motivation

In post-silicon debug, unknown signals can be restored from the traced signal states using forward and backward restoration. Forward restoration deals with reconstructing the output from the input. In other words, known inputs can provide the output. On the other hand, backward restoration deals with restoring the inputs from the output. If all the inputs are known the unknown output can be definitely determined while backward restoration might fail in certain scenarios [3].

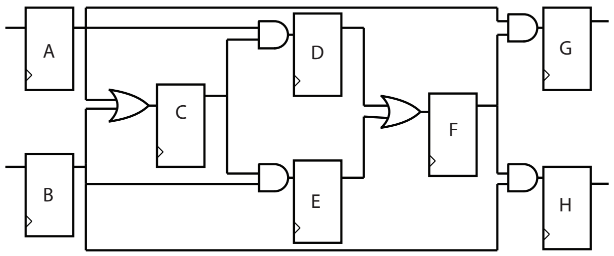

Fig. 2. Example circuit

Figure 2 shows a simple circuit with 8 flip-flops used in [6]. We use this example to show the benefit of using different fine-grained scan chains. Restoration ratio (RR) which is a popular metric for measuring restorability is defined as follows.

$$Restoration\ Ratio = \frac{No.\ of\ tarced\ and\ restored\ states}{No.\ of\ traced\ states}$$

TABLE I
Restored signals using [6]

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

Let us assume that the trace buffer width is 2 which means state of only two signals can be stored in each clock cycle. Table I shows the signal states that can be restored using the selected signals in [6]. Traced signals are shown in shades. Signal C is traced every cycle whereas A and F are dumped in alternate cycles using a scan chain with one shadow flip-flop. Although scan signals are dumped in alternate cycles, the table shows states for both A and F in cycle 1, cycle 3, and so on. This is because in cycle 1 the state of signal A is dumped whereas in cycle 2 the state of signal F is dumped. However, the scan chain (i.e., A and F using shadow flip-flops) holds the state for the same cycle, although different parts were dumped in different cycles. In other words, the signal state of F captured at cycle 1 is dumped in cycle 2. The symbol ‘X‘ represents the state that cannot be restored using known signal states. It can be seen that in this case a total number of 36 states can be restored and a total number of 16 states are traced. Therefore the restoration ratio is 2.25.

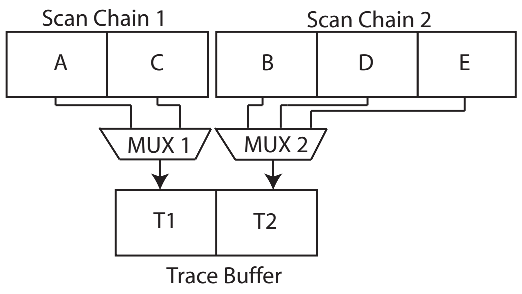

Fig. 3. Proposed debug architecture for example circuit in Figure 2

We now show how using different scan chains can help in signal restoration using the same circuit. Figure 3 shows an illustrative example of our proposed partitioning of trace buffer of width 2 for the same circuit. It can be observed that trace buffer width is shared between two different scan chains of length 2 and length 3. In this case there are no trace signals and the buffer width is partitioned between two scan chains. We apply our method to select efficient signals of the sample circuit for this debug architecture. Consequently, we assign signals A and C to the first scan chain while signals B, D, and E to the second scan chain. Table II shows the values of trace buffer in each cycle. The subscript indicates the value in that clock cycle. For example, A_3 implies the value of flip-flop A in cycle 3. It can be observed that signals A and C are stored in trace buffer in alternate cycles whereas B, D, and E are dumped in every third cycle. In other words, signals A and C are dumped with period (T) equals to 2 and frequency (f) equals to $1/2$. On the other hand, dumping period for signals B, D, and E is equal to 3, while dumping frequency for these signals is equal to $1/3$.

TABLE II
TRACE BUFFER VALUES IN OUR DEBUG ARCHITECTURE

Buffer/Cycle	1	2	3	4	5	6	7	8
T1	A_1	C_1	A_3	C_3	A_5	C_5	A_7	C_7
T2	B_1	D_1	E_1	B_4	D_4	E_4	B_7	D_7

Table III shows the signal states that can be restored using the signals chosen by our method. It can be seen that a total number of 55 states can be restored and a total number of 17 states are traced. The restoration ratio in this case is 3.24 which is higher than 2.25 [6]. Thus, more signal states give a more detailed view of the internal state of the circuit.

The primary problem of using different scan chains is to determine what signals to select for each scan chain. Signals should be chosen such that more important signals are assigned to smaller scan chains with small dumping period. These signals cover significant parts of the circuit. Less important signals on the other hand, should be distributed among the scan chains with larger dumping periods. In addition, minimizing the overlaps between the states that can be restored by different scan chains should be considered in selection algorithm in order to maximize the restoration ratio in a debug scenario. In this paper, we have developed an algorithm to select efficient set of signals for each scan chain.

TABLE III
RESTORED SIGNALS USING OUR METHOD

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

IV. FINE-GRAINED COMBINATIONS

In this section, we first propose our fine-grained debug architecture. Next, we present our signal selection algorithm.

A. Debug Architecture

Our fine-grained architecture is motivated by the design of [6], [5]. They proposed an architecture that divides the trace buffer into two parts, one for trace signals and the other one for scan signals. However, this partitioning is coarse-grained. One extreme is important signals that are traced every cycle whereas the other one is less important signals that are assigned to a scan chain. The dumping frequency for scan signals depends on trace buffer width and number of signals in the scan chain. Putting more signals in scan chain decreases the dumping frequency for signals. On the other hand, putting less signals may not be desirable as it decreases the coverage of the circuit. As discussed earlier, the limitation of the existing approaches is to consider only two extremes and losing the opportunity for not considering in-between scenarios. We consider a fine-grained approach by allowing multiple partitions of trace buffer width.

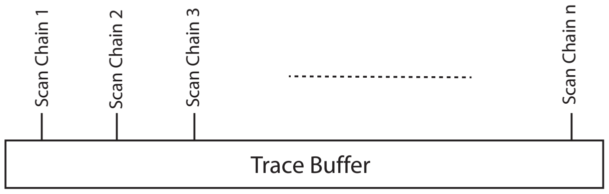

Fig. 4. Width n of the trace buffer is shared by n scan chains

As illustrated in Figure 4 the width n of the trace buffer is partitioned for n different scan chains. These partitions are shown to be numbered from 1 to n. Each of these scan chains comprises of different number of signals which determines the dumping frequency for those signals. It has to be noted that if a particular scan chain contains only one signal it is essentially a trace signal that is traced every cycle. These different signal chains provide more fine-grained dumping frequencies. Thus, each signal can be assigned to the appropriate scan chain based on its importance. These fine-grained scan chains enable us to dump larger number of signals which improves the observability in the circuit compared with coarse-grained scan signals. Next, we describe our proposed algorithm which tries to select the best signal in each iteration considering all the signals that have been selected before.

B. Fine-grained Signal Selection Algorithm

In this section, we describe our greedy algorithm which determines the signals that need to be assigned to each scan chain in order to maximize the observability. We define $P_0(f)$ and $P_1(f)$ for flip-flop f in the circuit that define the probability of its value being 0 and 1, respectively. These values can be calculated by feeding the simulator with circuit graph and random input vectors and running it for numerous times. We also use *connectivity* information similar to [6]. The connectivity of a flip-flop is the number of flip-flops connected with it through other combinational gates in both backward and forward directions.

In order to partition the trace buffer in different scan chains we define *buffer width (bw)*, *minimum trace signals* (ω), *partition factor* (α), and *step function* (ϕ). The buffer is divided into two parts. First part consists of ω trace signals that are dumped every cycle. The remaining $bw - \omega$ buffers are further divided into α partitions. Each partition consists of $(bw-\omega)/\alpha$ scan chains with identical length. The step function determines the length of scan chains in each partition. In other words, assume l_i and l_{i+1} are the lengths of scan chains in two successive partition, then we would have: $l_{i+1} = \phi(l_{i+1})$. It has to be noted that l_0 is equal to initial value of 1.

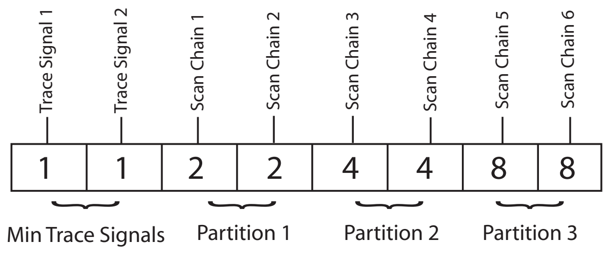

Fig. 5. An example of trace buffer partitioning in a debug architecture with $bw = 8, \omega = 2, \alpha = 3$, and $\phi(i) = 2 * \phi(i-1)$

Figure 5 illustrates an example of trace buffer partitioning in a debug architecture with $bw = 8, \omega = 2, \alpha = 3$, and $\phi(i) = 2 * \phi(i-1)$. It can be seen that there are two trace signals that are dumped every cycles. The rest of the trace buffer is shared between fine-grained scan chains. For example, first partition consists of two scan chains each of them with identical length of 2. In other words, two signals that are assigned to scan chain 1 will be dumped in alternate cycles. We also define *dumping period (T)* for each scan chain. Clearly, dumping period for a particular scan chain is equal to its length. For example, four signals are assigned to scan chain 3 in Figure 5. Each of these signals will be dumped every four cycles. Fine-grained partitioning enables us to assign signals to different dumping periods based on their importance. For example in Figure 5, important (control) signals are assigned to trace slots (T=1). On the other hand, less important signals are assigned to scan chains with larger lengths (T=2, T=4, and T=8), based on their impact on the restoration performance.

We use *restoration power (RP)* as a selection metric in our algorithm. Assume *S* is the current set of assigned flip-flops to scan chains. In addition, f_T implies that flip-flop *f* is dumped every *T* cycles. We show the dumped values of *f* using *v* that can be either 0 or 1. We define $\delta(S \cup \{f_T\}, v)$ as the number of additional states that can be restored using $S \cup \{f_T\}$ (compared to restored stated using only *S*) over T_c cycles when we dump *f* each *T* cycles with the assumption that *f* values are *v* over the dumping cycles. This is used in the selection metric of our algorithm. Clearly, larger T_c is more desirable as it yields more precise result. However, large T_c in real scenarios is not practical as there are numerous number of flip-flops that make the restoration process computationally expensive. Our experimental results demonstrate that $T_c = LCM(l_0, l_1, ...l_{n-1})$ is large enough where l_i is the length of scan chain i and LCM is least common multiple. The reason is that restoration pattern in whole circuit is repeated over each $T_c = LCM(l_0, l_1, ...l_{n-1})$ cycles. For example, in Figure 5 $T_c = LCM(1, 2, 4, 8) = 8$ is used in our algorithm. For a particular flip-flop *f* with a dumping period of T (f_T), *RP* is defined as follows.

$$RP(f_T) = T*(P_0(f)*\delta(S\cup\{f_T\},0)+P_1(f)*\delta(S\cup\{f_T\},1))$$

In other words, *RP* is the number of probable additional states that can be restored by adding f_T to the *S*. It is multiplied by T because we would like to take into account the resources that f_T uses for this additional restored states. Larger *T* means smaller dumping frequency and smaller resource usage. In fact, the intuition of *RP* is that in each iteration we try to choose a flip-flop that makes the best trade-off between the maximum newly restored states and minimum resource usage.

Algorithm 1 outlines the major steps in our signal selection algorithm. First, we create a graph of flip-flops using the same methodology described in [2]. This graph is used to compute the connectivity of each flip-flop. Next, we calculate P_0 and P_1 for each flip-flop, and partition the trace buffer using input parameters. We create an empty list S to hold the list of selected flip-flops. In each iteration, RP is calculated for all the not chosen flip-flops (flip-flops that are not in S). If two or more flip-flops have equal RP we choose the one with higher connectivity. This increases the chance of restoring more states during the real debug scenario as it is connected to more flip-flops. We continue assigning one flip-flop in each iteration until all the scan chains get full.

We now show how our algorithm works for the example circuit in Figure 2. Assume that trace buffer width is 2. We

Algorithm 1 Signal selection algorithm

```
1: procedure SELECTSIGNALS($circuit, bw, \omega, \alpha, \phi$)
2:     Create flip-flops graph
3:     Calculate $P_0$ and $P_1$ for all the flip-flops
4:     Create list of selected signals S /* initially empty */
5:     Partition trace buffer using ($bw, \omega, \alpha, \phi$)
6:     while there is an available scan chain do
7:         for each flip-flop F that is not in the S do
8:             for each available scan chain of length T do
9:                 Calculate RP for $f_T$ using $S \cup \{f_T\}$
10:            end for
11:        end for
12:        Find flip-flop f with dump period T ($f_T$) that has the maximum RP. If two or more flip-flops have same RP, find the one with higher connectivity
13:        Assign f to a scan chain with length T
14:        Add $f_T$ to the list S
15:    end while
16:    return $S$
17: end procedure
```

partition the trace buffer using $bw = 2, \omega = 0, \alpha = 2$, and $\phi(i) = 1 + \phi(i-1)$. In other words, we have two scan chains of length 2 and 3, respectively. Hence, from our algorithm we would have $T_c = LCM(2,3) = 6$, which is used in restoration power calculation. Table IV summarizes intermediate results of our algorithm in each iteration. First column is the candidate flip-flops from the example circuit. Second and third columns are P_0 and P_1 of each flip-flop which are calculated by feeding our simulator with 100 different random inputs. The rest of the columns are the restoration power of flip-flops in each iteration. Each cell in these columns contains two rows which are the RP values if we assign the flip-flop to the scan chain of length 2 (T=2) or length 3 (T=3), respectively. In the first iteration, signal C has the highest RP for T=2 and is assigned to the first scan chain of length 2 (the RP value for the signal in each iteration is shown in bold). In the second iteration, both signals A and B yield the maximum RP when they are assigned to scan chain of length 2. In addition, since both of them have same connectivity (3), our algorithm selects one of them (signal A) randomly. Till now, signals A and C are assigned to first scan chain of length 2. Using the same procedure, signals B, E, and D are assigned to second scan chain of length 3 in the remaining iterations.

From Table IV, it can be observed that our algorithm covers a large part of the circuit by assigning more resources to important signals. It continues this procedure by assigning less resources to signals that can cover other parts of the circuit. As a result, it gets benefit of both spatial and temporal observability of fine-grained sets of signals.

TABLE IV
DIFFERENT RESTORATION POWER VALUES OF OUR ALGORITHM IN EXAMPLE CIRCUIT OF FIGURE 2

Signal	P_0	P_1	RP (I1)	RP (I2)	RP (I3)	RP (I4)	RP (I5)
A	0.5	0.5	15.00 15.00	**10.00** 6.00	- -	- -	- -
B	0.5	0.5	15.00 15.00	10.00 6.00	- **13.50**	- -	- -
C	0.29	0.71	**21.19** 19.15	- -	- -	- -	- -
D	0.6	0.4	14.70 11.93	4.37 8.37	- 10.20	- 4.20	- **6.00**
E	0.6	0.4	14.68 11.92	4.37 8.37	- 13.18	- **7.18**	- -
F	0.45	0.55	14.99 14.09	6.49 5.70	- 10.35	- 3.00	- 3.00
G	0.68	0.32	9.80 8.85	2.63 4.90	- 9.00	- 3.00	- 3.00
H	0.68	0.32	9.81 8.86	2.63 4.90	- 9.95	- 3.95	- 3.00

V. EXPERIMENTS

A. 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 conducts simulation in both forward and backward directions following the mechanism outlined in [1]. The simulator iterates on the unknown signals queue and tries to restore them using 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 same circuits.

We fed the simulator with 100 sets of random values and noted the average restoration ratios. 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. Table V shows the set of parameters that we used for each benchmark and different trace buffer widths.

B. Results

Table VI compares the restoration ratios of our approach with several previous trace only techniques [3], [2], [4] using different ISCAS'89 benchmarks. The trace buffer used in our experiment are $8 \times 4k$, $16 \times 4k$, and $32 \times 4k$. The corresponding restoration ratio for each technique (if available) is reported. Last column indicates the percentage of improvement using our approach compared with the best (shown in bold) result provided by existing techniques. It can be observed that our approach performed mostly better than previous techniques. The improvement in restoration performance is up to 91% in *s9234* (32.3% on average). However, our approach shows performance degradation (or comparable results) in *s5378* which is the smallest circuit. The reason is that as there are

TABLE V
DIFFERENT PARAMETERS FOR OUR APPROACH

Circuit	Buffer Width	ω	α	ϕ
s5378	8	4	1	$\phi(i) = 1 + \phi(i-1)$
	16	8	4	$\phi(i) = 2 * \phi(i-1)$
	32	8	3	$\phi(i) = 1 + \phi(i-1)$
s9234	8	4	4	$\phi(i) = 2 * \phi(i-1)$
	16	8	4	$\phi(i) = 2 * \phi(i-1)$
	32	12	4	$\phi(i) = 2 + \phi(i-1)$
s15850	8	2	3	$\phi(i) = 2 * \phi(i-1)$
	16	2	7	$\phi(i) = 1 + \phi(i-1)$
	32	8	6	$\phi(i) = 1 + \phi(i-1)$
s38584	8	2	3	$\phi(i) = 2 + \phi(i-1)$
	16	4	3	$\phi(i) = 2 + \phi(i-1)$
	32	8	3	$\phi(i) = 1 + \phi(i-1)$
s38417	8	2	3	$\phi(i) = 2 * \phi(i-1)$
	16	8	4	$\phi(i) = 2 * \phi(i-1)$
	32	16	4	$\phi(i) = 2 + \phi(i-1)$
s35932	8	4	1	$\phi(i) = 1 + \phi(i-1)$
	16	8	1	$\phi(i) = 1 + \phi(i-1)$
	32	16	1	$\phi(i) = 1 + \phi(i-1)$

less number of flip-flops in this circuit, most of the dominating signals can fit in the trace buffer. In other words, the trace buffer width is enough to cover most parts of the circuit just using trace only signals. In this case, partitioning and putting more signals in trace buffer does not help much. Our approach performs significantly better in majority of the cases as existing trace only approach only takes advantage of temporal observability of a small set of signals while missed the opportunity of both spatial and temporal observability of a large set of signals.

TABLE VI
RESTORATION RATIOS USING OUR APPROACH COMPARED WITH DIFFERENT TRACE ONLY APPROACHES

Circuit	Buffer Width	[3]	[2]	[4]	Ours	Imprv.(%) over best
s5378	8	**14.67**	-	13.24	14.65	0
	16	**8.99**	-	7.83	8.64	-4
	32	4.72	-	**4.89**	5.00	2
s9234	8	4.76	-	**10.68**	20.43	91
	16	**7.18**	-	7.16	12.31	71
	32	**4.67**	-	4.18	6.78	45
s15850	8	19.93	-	**39.54**	47.35	20
	16	24.22	-	**24.85**	26.00	5
	32	13.30	-	**13.60**	14.71	8
s38584	8	19.23	78.00	**84.10**	146.64	74
	16	13.96	40.00	**47.04**	80.85	72
	32	8.68	20.00	**26.97**	43.22	42
s38417	8	18.63	**55.00**	45.21	55.40	1
	16	18.62	29.00	**30.77**	33.41	9
	32	14.20	16.00	**20.25**	21.33	5
s35932	8	64.00	95.00	**96.12**	178.51	86
	16	38.13	60.00	**67.45**	89.25	32
	32	21.06	35.00	**43.23**	45.01	4

We also compared with the existing trace+scan approach proposed by Basu et al. [6] in Table VII. It is important to note that we did not compare with other trace+scan approaches (such as [5]) since [6] has shown to perform better than other approaches. It can be observed that our approach outperforms [6] consistently and produced up to 116% improvement in *s38584* (54.7% on average). The reason for significant improvement is that their approach is limited by coarse-grained partitioning (two fixed partitions) of signals.

TABLE VII
RESTORATION RATIOS USING OUR APPROACH COMPARED WITH [6]

Circuit	Buffer Width	Basu et al. [6]	Our Approach	Improvement (%)
s38584	32	20.00	43.22	116
s38417	32	18.00	21.33	19
s35932	32	35.00	45.01	29

VI. CONCLUSIONS

Signal selection is an important part of post-silicon debug. Existing techniques mainly focused on trace only signal selection. Recent techniques employed coarse-grained combination of trace and scan signals and showed limited effectiveness. In this paper, we presented a debug architecture consisting of fine-grained combination of trace and scan signals. We developed an efficient algorithm to select most profitable signals based on the proposed architecture. Our experimental results using ISCAS'89 benchmarks demonstrated that our approach shows up to 91% (32.3% on average) higher restoration ratio compared to existing trace only approaches. Our approach produces up to 116% improvement (54.7% on average) compared with the state-of-the-art approach that considers a combination of trace and scan signals.

REFERENCES

[1] X. Liu and Q. Xu, "Trace signal selection for visibility enhancement in post-silicon validation," in *DAC*, 2009.

[2] K. Basu and P. Mishra, "Efficient trace signal selection for post silicon validation and debug," in *VLSI Design*, 2011.

[3] H. F. Ko and N. Nicolici, "Algorithms for state restoration and trace-signal selection for data acquisition in silicon debug," *CAD*, 2009.

[4] D. Chatterjee et al., "Simulation-based signal selection for state restoration in silicon debug," in *ICCAD*, 2011.

[5] H. F. Ko and N. Nicolici, "Combining scan and trace buffers for enhancing real-time observability in post-silicon debugging," in *ETS*, 2010.

[6] K. Basu etal., "Efficient combination of trace and scan signals for post silicon validation and debug," in *ITC*, 2011.

[7] O. Caty et al., "Microprocessor silicon debug based on failure propagation tracing," in *ITC*, 2005.

[8] F. De Paula et al., "Backspace: Formal analysis for post-silicon debug," in *FMCAD*, 2008.

[9] N. Nataraj et al., "Fault localization using time resolved photon emission and stil waveforms," in *ITC*, 2003.

[10] M. Abramovici et al., "A reconfigurable design-for-debug infrastructure for socs," in *DAC*, 2006.

[11] D. Josephson and B. Gottlieb, "The crazy mixed up world of silicon debug [ic validation]," in *CICC*, 2004.

[12] S. Prabhakar and M. Hsiao, "Using non-trivial logic implications for trace buffer-based silicon debug," in *ATS*, 2009.

[13] G. Van Rootselaar and B. Vermeulen, "Silicon debug: scan chains alone are not enough," in *ITC*, 1999.

[14] R. Datta et al., "Delay fault testing and silicon debug using scan chains," in *ETS*, 2004.