2009 27th IEEE VLSI Test Symposium

Automated Selection of Signals to Observe for Efficient Silicon Debug

Joon-Sung Yang and Nur A. Touba

Computer Engineering Research Center
Department of Electrical and Computer Engineering
University of Texas, Austin, TX 78712

Abstract

Internal signals of a circuit are observed to analyze, understand, and debug nonconforming chip behavior. The number of signals that can be observed is limited by bandwidth and storage requirements. This paper presents an automated procedure to select which signals to observe to facilitate early detection of circuit malfunction to help find the root cause of a bug. This paper exploits the nature of error propagation in sequential circuits by observing signals which are most often sensitized to possible errors. Given a functional input vector set, an error transmission matrix is generated by analyzing which flip-flops are sensitized to other flip-flops. Signal observability is enhanced by merging data from relatively independent flip-flops. The final set of signals to observe is determined through integer linear programming (ILP) which provides a set of locations that maximally cover the possible error sites within given constraints. Experimental results indicate that the cycle in which a bug first appears can be more rapidly and precisely found with the proposed approach thereby speeding up the post-silicon debug process.

1. Introduction

The advance of technology allows sophisticated designs with millions of transistors. Due to inaccuracies in modeling integrated circuits (ICs) along with process variations during the manufacturing process, identifying and resolving problems in ICs after first silicon is a very time consuming task [Josephson 04], [Ko 08], [Yang 08a, 08b]. Unlike during pre-silicon verification, the accessibility and visibility of internal signals are very limited in post-silicon debug and hence this is the major challenge in the validation and debug of first silicon. The narrow observability of internal signals makes silicon debug costly and time consuming.

Techniques have been proposed to enhance the observability of internal signals via complete, but non-real time observation, using scan chains and selective, but real time observation, such as using trace buffers or direct access via dedicated pins. Scan-based debug [Hopkins 06], [Vermeulen 02] gives high observability of internal signals by re-using scan chains, however, it requires halting the system to scan out responses from the circuit-under-debug (CUD). Trace buffer based debug [Abramovici 06], [Anis 07a, 07b], [Yang 08a] provides at-speed signal capture capability over a limited number of clock cycles which enhances the observability of the internal signals. The amount of data that can be observed with a trace buffer is limited by its on-chip storage space. Compression techniques can be applied to further improve the observability provided by a trace buffer [Anis 07a, 07b], [Yang 08a]. In [Vermeulen 01], a set of signals required for debugging was connected to a multiplexer module, called *SPY*, for real-time observation, and then captured in a register or monitored via chip pins.

Increased internal signal observability helps to discover erroneous behavior closer to the source of the problem, both in space and time. Some previous research has been done on ways to enhance internal observability. In [Abramovici 05] and [Hsu 06], techniques are proposed for constructing the values of more signals than are captured each clock cycle in a trace buffer. The captured silicon data is mapped to Boolean equations and non-visible values in combinational logic are expanded by a dependency and approximation method. This method provides some improvement in observing localized signals. [Park 08] shows an architectural level approach for post-silicon bug localization. It records the history of the program executed and identifies the bug location-time information at the system level. Experimental results show that its method can effectively locate bugs with high accuracy.

In [Ko 08], an automated data reconstruction method for sequential circuits is investigated. The restorability of signals is calculated to determine the signals to be traced. Results in [Ko 08] for ISCAS benchmark circuits show that this approach can restore signals up to 130 times better. However, if the logic depth between internal state

1093-0167/09 $25.00 © 2009 IEEE
DOI 10.1109/VTS.2009.51

elements is deep, the amount of restorability may be very limited. If the combinational logic depth is shallow, this approach can greatly help post-silicon debug with a number of internal signals implied by captured data.

In [Yang 08b], a signal monitoring technique based on non-destructive scan chains is investigated. In non-destructive scan, shadow scan latches are used to retain the internal state during scan out. Conventional scan chains that have non-destructive scan capability are configured to operate as multiple MISRs during normal system operation. Internal signal observability is increased by observing the compressed internal system states without halting the system. Information from the MISRs is periodically monitored to identify erroneous behavior. Results show that only a small number of scan dumps are needed to zero in the first erroneous clock cycle. However, this technique can only be applied to designs which have non-destructive scan chains.

In this paper, a method to maximize the effectiveness of limited internal signal observability is proposed based on carefully selecting which signals to observe. An automated procedure is described for selecting the signals to observe to maximize early error detection during silicon debug. By detecting circuit misbehavior soon after it occurs, the search space for zeroing in on the root cause of the misbehavior is greatly reduced thereby speeding up the debug process. The proposed method exploits the nature of error propagation in sequential circuits by observing signals which are most often sensitized to possible error sites. The set of signals to observe is determined by using an error transmission matrix that is generated by analyzing which flip-flops are sensitized to other flip-flops. Signal observability is enhanced by merging data from relatively independent flip-flops. The final set of signals to observe is determined through integer linear programming (ILP) which provides a set of locations that maximally cover the possible errors with a given condition.

The paper is organized as follows. Sec. 2 gives an overview of the proposed scheme. Sec. 3 discusses the three procedures to determine the signals to observe in detail. Experimental results are shown in Sec. 4 and conclusions are given in Sec. 5.

2. Overview of Proposed Scheme

The proposed method provides information on which limited number of signals to observe in a circuit to maximize the efficiency of the post-silicon debug process. In a sequential circuit, it may take many cycles for an error to propagate to a primary output where it can be observed [Yang 08b]. In the proposed method, signals are observed along the paths where error propagation is most likely.

In debug mode, functional vectors are applied and the responses are analyzed to validate a chip. Using the functional vectors and treating the flip-flops as sources of errors, fault simulation can be performed to study the error transmission between flip-flops. The error transmission information can be represented as a matrix which will be referred to here as the *"error transmission matrix"*. Based on this information, the flip-flops that are most often sensitized to other flip-flops can be identified and used as candidates for the set of signals to observe.

Flip-flops are *relatively independent* if a single error in a circuit will not influence them simultaneously. Relatively independent flip-flops can be XORed together to increase the overall observability of the internal signals. The error transmission matrix can be updated by forming signal groups by combining (XORing) the relatively independent flip-flops in the matrix.

Because there is limited storage space provided by DFD (design for debug) hardware, e.g., a trace buffer, it is important to efficiently choose the set of signals to observe which will detect as many errors as possible. For this purpose, integer linear programming (ILP) is used to select the signal groups from the error transmission matrix

3. Details of Signals to Observe Selection

The following subsections describe each of the steps in the proposed procedure for selecting the signals to observe.

3.1 Generating Error Transmission Matrix

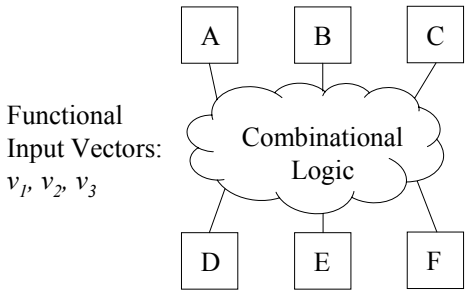

Figure 1. Example of a Simple Logic

Fig.1 shows an example of some simple logic that has 6 sequential elements represented by rectangles named *A* to *F* and combinational logic illustrated as a cloud. For simplicity, assume three functional vectors (v_1, v_2 and v_3) are applied to this logic. When the vectors are applied, if there is a bug, the erroneous response could be captured in some flip-flops at some time. That faulty response would likely keep propagating in a sequential circuit over multiple cycles.

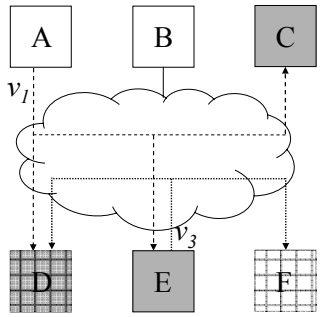

Figure 2. Error (in A and E) Propagation

The error transmission matrix is generated by injecting errors at each flip-flop for each vector in the vector set and performing fault simulation for one cycle to see where the error propagates. For example, simulation can be done to see which flip-flops are corrupted by an error in *A* when input vector v_1 is applied. Next, we make *B* faulty and see which flops are sensitized to the error with v_1. To illustrate this, Fig. 2 shows where an error at *A* and *E* propagates. The error at *A* is transmitted to flip-flops *C*, *D* and *E* for input vector v_1 (highlighted in gray color), and an error at *E* is transmitted to *D* and *F* for v_3 (highlighted in dashed line), respectively.

	A	B	C	D	E	F
(A, v_1)	0	0	1	1	1	0
(B, v_1)	1	0	0	0	1	0
(C, v_1)	1	1	0	0	0	0
(D, v_1)	0	0	0	0	0	0
(E, v_1)	0	0	0	0	1	0
(F, v_1)	0	1	1	0	1	0
(A, v_2)	0	0	0	0	0	1
(B, v_2)	1	0	0	1	1	0
(C, v_2)	0	1	0	0	0	0
(D, v_2)	1	0	0	0	0	0
(E, v_2)	0	1	1	0	0	0
(F, v_2)	0	0	0	0	1	0
(A, v_3)	0	1	0	0	1	0
(B, v_3)	0	0	0	0	0	1
(C, v_3)	1	1	0	0	1	0
(D, v_3)	0	0	0	0	0	0
(E, v_3)	0	0	0	1	0	1
(F, v_3)	0	0	0	0	1	0

Figure 3. Error Transmission Matrix

Error transmission information corresponding to input vectors and error locations is represented in the error transmission matrix. This is illustrated in Fig. 3. Each column represents a flip-flop in the circuit, and each row shows the error information. *(A, v_1)* in the first row indicates that an error is located in *A* and for vector v_1 it propagates to *C*, *D* and *E* which each have an *'1'* in the first row of Fig. 3. Once the error transmission matrix is generated, the flip-flops that are most often sensitized to possible errors can be identified assuming bugs in silicon are modeled as occurring evenly distributed in time and space. Note that an error will likely propagate for multiple clock cycles and need not necessarily be detected in the first cycle in which it occurs. The columns in the error transmission matrix with the most 1's are probabilistically more likely to capture errors over time since errors are transmitted to them most frequently. Hence, they are candidates for signals to observe for the better observability. Moreover, if a limited set of signals to observe is to be selected, then columns that are most non-overlapping and cover as many rows as possible are also more likely to cover more errors. This will be discussed in more detail later.

3.2 Merging Relatively Independent Flip-Flops

Relatively independent flip-flops in a circuit are identified and merged to achieve better observation capability. The overall goal is to find misbehavior as early as possible, so observing more signals helps silicon debug by providing more internal signal information.

If two flip-flops are relatively independent, the erroneous response for one error will not be simultaneously transmitted to both flip-flops. For the example in Fig. 2, since an error in *A* is transmitted to *C*, *D,* and *E* for vector v_1, flip-flops *A, B,* and *F* are relatively independent to the possible error *(A, v_1)*. Therefore, *A*, *B*, and *F* can be merged together in this case and *E* can be combined with *A*, *B*, *C* in the *(E, v_3)* case.

In the error transmission matrix, if there are multiple *1's* in a row, the corresponding flip-flops are relatively dependent for a possible error. By finding the columns in the matrix which are not sensitized to the same errors simultaneously, relatively independent flip-flops can be identified.

Relatively independent flip-flops can be XORed together without losing error observation for single flip-flop errors. Note, however, that flip-flops are relatively independent only with respect to single errors, so it is still possible for multiple errors to cancel. However, this serves as a good heuristic for increasing overall error coverage. In Fig. 3, relative independence is checked among flip-flops *(A ~ F)* and three relatively independent signal groups can be found. Flip-flop *A*, *C* and *F* are not sensitized to the same errors, and any error set does not influence flip-flop *B* and *D* simultaneously. Therefore, the first signal group *(S_0)*, the second group *(S_1)* and the last group *(S_2)* can be expressed respectively as follows:

<u>*Signal Groups (S_0, S_1, S_2)*</u>

$S_0 = A \oplus C \oplus F$

$S_1 = B \oplus D$

$S_2 = E$

Then, the error transmission matrix is updated based on the signal groups as shown in Fig. 4. Error index ($R_0 \sim R_{17}$) is used to identify the errors from (A, v_1) to (F, v_3).

	A	B	C	D	E	F		S_0	S_1	S_2
(A, v_1) : R_0	0	0	1	1	1	0		1	1	1
(B, v_1) : R_1	1	0	0	0	1	0		1	0	1
(C, v_1) : R_2	1	1	0	0	0	0		1	1	0
(D, v_1) : R_3	0	0	0	0	0	0		0	0	0
(E, v_1) : R_4	0	0	0	0	1	0		0	0	1
(F, v_1) : R_5	0	1	1	0	1	0	→	1	1	1
(A, v_2) : R_6	0	0	0	0	0	1		1	0	0
(B, v_2) : R_7	1	0	0	1	1	0		1	1	1
(C, v_2) : R_8	0	1	0	0	0	0		0	1	0
(D, v_2) : R_9	1	0	0	0	0	0		1	0	0
(E, v_2) : R_{10}	0	1	1	0	0	0		1	1	0
(F, v_2) : R_{11}	0	0	0	0	1	0		0	0	1
(A, v_3) : R_{12}	0	1	0	0	1	0	→	0	1	1
(B, v_3) : R_{13}	0	0	0	0	0	1		1	0	0
(C, v_3) : R_{14}	1	1	0	0	1	0		1	1	1
(D, v_3) : R_{15}	0	0	0	0	0	0		0	0	0
(E, v_3) : R_{16}	0	0	0	1	0	1		1	1	0
(F, v_3) : R_{17}	0	0	0	0	1	0		0	0	1

Figure 4. Updated Error Transmission Matrix

A limit can be placed on the number of signals which are XORed together when the error transmission matrix is updated to minimize delay and/or routing. Results are shown in Sec. 4 with different limits on the maximum number of signals XORed together.

3.3 Determining the Set of Signals to Observe

Because there are limitations on storage, bandwidth, and overhead for observing signals, it is very important to choose the best set of signal groups to observe for the most efficient debugging. With given conditions in the error transmission matrix, debugging capability can be maximized by finding a set of signal groups that are sensitive to the broadest set of errors.

Integer linear programming (ILP) is employed to select an optimal set of signals based on the error transmission matrix. The updated error transmission matrix in Fig. 4 is formulated as the set of equations in Fig. 5.

In Fig. 5, R_0 denotes 0^{th} row in the updated error transmission matrix and S_0 represents 0^{th} column. Because the objective in solving ILP is to maximize the number of errors covered by a set of signal groups, the objective equation *("maximize the covered errors")* is expressed as the summation of the entire error sets *(1)*. If an error is covered by any signal group, then the value *'1'* is assigned to a corresponding error variable (R_k). And if an error is not covered, *'0'* will be assigned *(4)*. When a signal group is selected for observation, S_i has *'1' (5)*. For example, if S_0 is selected and S_1 is not selected, S_0 is *1* and S_1 is *0*. Because the number of signal groups to observe is always larger than *1* (i.e. $S_0 + S_1 + S_2 \geq 1$) *(2)*, the row constraints can be represented as equations *(3)*. In the first row, R_0 is always detected by $S_{0\text{-}0}$, $S_{0\text{-}1}$ or $S_{0\text{-}2}$ where $S_{k\text{-}i}$ denotes a signal group in k^{th} row and i^{th} column. Therefore, the ILP formulation from the first row can be expressed as $S_{0\text{-}0} + S_{0\text{-}1} + S_{0\text{-}2} \geq R_0$. From the second row, we can derive a constraint as $S_{1\text{-}0} + S_{1\text{-}2} \geq R_1$. This implies that if a signal selection of either S_0 or S_2 $(S_{1\text{-}0} + S_{1\text{-}2} = 1)$, or selection of both S_0 and S_2 $(S_{1\text{-}0} + S_{1\text{-}2} = 2)$ will cover R_1 $(R_1 = 1)$, and if none of S_0 and S_1 $(S_{1\text{-}0} + S_{1\text{-}2} = 0)$ is selected, R_1 would not be covered $(R_1 = 0)$. Therefore, a summation of $S_{1\text{-}0}$ and $S_{1\text{-}2}$ is always greater than or equal to R_1. In the same manner, a total of 18 ILP formulations are generated in *(3)*.

$$max : R_0 + R_1 + R_2 + \ldots + R_{15} + R_{16} + R_{17} \quad (1)$$

$$s.t : S_0 + S_1 + S_2 = num.\ of\ signal\ groups\ to\ observe \quad (2)$$

$$\left.\begin{aligned} S_{0\_0} + S_{0\_1} + S_{0\_2} &\geq R_0 \\ S_{1\_0} + S_{1\_2} &\geq R_1 \\ &\vdots \\ S_{16\_0} + S_{16\_1} &\geq R_{16} \\ S_{17\_2} &\geq R_{17} \end{aligned}\right\} \quad (3)$$

$$R_0, R_1, R_2, \ldots, R_{15}, R_{16}, R_{17} \in \{0, 1\} \quad (4)$$

$$S_0, S_1, S_2 \in \{0, 1\} \quad (5)$$

Figure 5. ILP Formulation for Updated Error Transmission Matrix

If two signal groups are to be chosen in Fig 4, the ILP solver selects S_0 and S_2. 12 errors are covered by the S_0 and S_1 combination, S_0 and S_2 gives 15 covered errors, and S_1 and S_2 can cover 13 errors. Therefore, S_0 and S_2 cover the maximum number of possible errors with the given constraint in Fig. 5. The signals corresponding to S_0 and S_2 are selected for observation.

A general ILP formulation for locating the set of signals to observe is shown in Fig. 6. When there are n rows and m columns in the updated error transmission matrix, the constraints are expressed using R and S where R and S denote the error list and the signal group respectively. The formulation of the objective is shown in *(1)* in Fig. 6 to maximize the number of covered errors

that satisfies *(2)*. $S_{k\text{-}i}$ in *(3)* is a signal list in i^{th} column with k^{th} row error list in the updated error transmission matrix. And $x_{k\text{-}i}$ is an element in the intersection of k^{th} row and i^{th} column of the matrix. The solution space for R and S is *{0, 1}* in *(4)* and *(5)*.

$$max: \sum_{k=0}^{n-1} R_k \tag{1}$$

$$s.t: \sum_{i=0}^{m-1} S_i = num.\ of\ \ signal\ groups\ to\ observe \tag{2}$$

$$\sum_{i=0}^{m-1} x_{k\_i} S_{k\_i} \geq R_k \quad \begin{matrix} for\ k \in \{0, 1, \cdots, n-1\} \\ for\ x_{k\_i} \in \{0, 1\} \end{matrix} \tag{3}$$

$$R_k \in \{0, 1\} \quad for\ k \in \{0, 1, \cdots, n-1\} \tag{4}$$

$$S_i \in \{0, 1\} \quad for\ i \in \{0, 1, \cdots, n-1\} \tag{5}$$

Figure 6. General ILP Formulation for Updated Error Transmission Matrix

The final set of signals to observe is determined through ILP which provides a set of signal groups that maximally cover the possible errors with the constraints in Fig. 6.

The size of the error transmission matrix increases with the number of functional vectors. The matrix can be partitioned for scalability. For example, if there are n patterns, we can generate two error transmission matrices using $n/2$ patterns each. The final signal groups can be determined from the two matrices by counting the number of possible errors detected.

4. Experimental Results

Experimental results are presented for ISCAS-89 benchmark circuits [Brglez 89] and an NOC (network-on-chip) design [Jang 08]. Random faults were injected in circuits to generate erroneous data. Random input patterns were applied to the ISCAS-89 benchmark circuits and deterministic functional verification vectors were applied to the NOC design. Fault simulation was conducted to generate the error transmission matrix. As discussed in Sec. 3.2, the number of signals which can be merged for the signal groups can be limited to avoid issues related to the physical design such as timing and wiring. We use three threshold values: 8, 12 and ∞ , to limit the number of signals included in a single group. The error transmission matrix was updated using threshold values. To find the final set of signals to observe, GNU Linear Programming Kit (GLPK) 4.32 [GLPK] was used as the ILP solver.

Table 1 shows how many flip flops are observed with each of the three threshold values. The first column shows the circuit name. The number of flip-flops in *s9234, s38584, and NOC* are 211, 1426 and 1991, respectively. The second column shows the maximum number of signals that can be merged into one signal group when updating the error transmission matrix. In the third column, the number of flip-flops observed is shown when 8, 12, 16 and 32 signal groups are chosen by ILP in the updated error transmission matrix. As can be seen from the results except for *s9234*, more flip-flops can be observed as the maximum value in the second column is increased. Since *s9234* is a relatively small benchmark circuit, the number of relatively independent flip-flops that are found is limited by its circuit size and not by the three threshold values.

Table 1. Number of Flip-Flops Observed by Proposed Method

Circuit	Max. Val	Num. of Signal Groups			
		8	12	16	32
s9234	8	23	25	35	53
	12	23	25	35	53
	∞	23	25	35	53
s38584	8	61	87	112	230
	12	86	127	173	380
	∞	893	1037	1205	1402
NOC	8	64	93	126	256
	12	92	134	184	379
	∞	367	921	1169	1543

Table 2 shows comparisons in terms of the average latency to detect bugs using the proposed method compared with two other techniques. The latency is measured by the number of clock cycles after the error is injected until it is observed. The measured latency is averaged over 300 different random error injections. The first column in Table 2 shows the name of benchmark circuit and the second column shows the type of technique used. For comparison purposes, two different ways of selecting the signals to observe were used in addition to the proposed method. In one way, signals are randomly chosen and observed to detect circuit misbehavior. In the other way, signals are chosen using structural information in the following way. The size of each logic cone is sorted and the flip-flops that are fed by the largest logic cones are selected. Debug was performed with the three methods: random, structure-based, and the proposed method to compare the efficiency of the silicon debug. For the proposed method, three different threshold values for the maximum number of signals to merge is used which are shown in the third column. As can be seen, the proposed method detects the erroneous data more rapidly in all cases. It can also be seen that the more signals that are merged, the faster debug process is achieved. These results show that careful signal selection can be used to increase the efficiency and speed of silicon debug.

Table 2. Average Erroneous Response Detection Latency Results for 300 Different Random Error Injections

Circuit	Type of Technique	Max. Val	Average Detection Latency with Different Number of Signal Groups			
			8	12	16	32
s9234	Random	N/A	320.81	212.18	186.82	173.73
	Structure	N/A	244.28	197.71	176.82	173.01
	Proposed	8	49.36	41.84	41.06	20.52
		12	49.36	41.84	41.06	20.52
		∞	49.36	41.84	41.06	20.52
s38584	Random	N/A	197.56	184.46	131.85	109.87
	Structure	N/A	178.08	146.73	127.32	115.86
	Proposed	8	67.35	61.44	51.87	31.41
		12	59.42	54.13	49.04	19.67
		∞	10.24	6.62	0.58	0.34
NOC	Random	N/A	594.42	571.87	574.52	504.65
	Structure	N/A	643.26	551.66	541.23	492.24
	Proposed	8	201.56	153.68	124.08	68.37
		12	148.97	117.79	109.80	45.21
		∞	47.63	21.79	16.37	9.12

5. Conclusions

In this paper, an automated procedure for selecting which signals to observe is proposed for more efficient silicon debug. The set of signals selected by the proposed method are most often sensitized to possible errors and they maximally cover the errors within given constraints. The result shows that the proposed method can detect the faulty response rapidly and can increase the effectiveness of DFD hardware.

It should also be noted that the proposed technique could be universally applied to any designs including those which do not have scan chains with non-destructive scan out capability.

Acknowledgement

The authors would like to thank Wooyoung Jang at the University of Texas at Austin for providing the NOC design used for the experiments. This research was supported by the National Science Foundation under Grant No. CCR-0426608.

References

[Abramovici 05] Abramovici, M., and Y.-C. Hsu, "A New Approach to Silicon Debug," *Proc. of Int. Silicon Debug and Diagnosis Workshop (SDD),* 2005.

[Abramovici 06] Abramovici, M., P. Bradley, K. Dwarakanath, P. Levin, G. Memmi, and D. Miller, "A Reconfigurable Design-for-Debug Infrastructure for SoCs," *Proc. of Design Automation Conference,* pp. 7-12, 2006.

[Anis 07a] Anis, E., and N. Nicolici, "On Using Lossless Compression of Debug Data in Embedded Logic Analysis," *Proc. of Int. Test Conference,* Paper 18.3, 2007.

[Anis 07b] Anis, E., and N. Nicolici, "Low Cost Debug Architecture using Lossy Compression for Silicon Debug," *Proc. of Design, Automation, and Test in Europe*, pp. 1-6, 2007.

[Brglez 89] Brglez, F., D. Bryan, and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," *Proc. of International Symposium on Circuits and Systems*, pp. 1929-1934, 1989.

[GLPK] http://www.gnu.org/software/glpk/glpk.html

[Hopkins 06] Hopkins, A., and K. McDonald-Maier, "Debug Support for Complex Systems on-Chip: A Review," *Proc. on Computers and Digital Techniques*, Vol 153, No. 4, pp. 197-207, Jul. 2006.

[Hsu 06] Hsu, Y.-C., F. Tsai, W. Jong and Y.-T. Chang, "Visibility Enhancement for Silicon Debug," *Proc. of Design Automation Conference*, pp. 13-18, 2006.

[Jang 08] Jang, W., Ding, D. and Pan., D., "A Voltage-Frequency Island Aware Energy Optimization Framework for Networks-on-Chip", *Proc. of Int. Conf. on Computer-Aided Design*, 2008.

[Josephson 04] Josephson, D. and Gottlieb, B., "The Crazy Mixed up World of Silicon Debug," *Proc. of Custom Integrated Circuits Conference*, pp. 665-670, 2004

[Ko 08] Ko, H. F., and Nicolici, N., "Automated Trace Signals Identification and State Restoration for Improving Observability in Post-Silicon Validation", *Proc. of Design, Automation, and Test in Europe*, pp. 1298-1303, 2008.

[Park 08] Park, S.-B., and Mitra, S., "IFRA: Instruction Footprint Recording and Analysis for Post-Silicon Bug Localization in Processors", *Proc. of Design Automation Conf.*, pp. 373-378, 2008.

[Vermeulen 01] Vermeulen, B., Oostdijk, S. and Bouwman, F., "Test and Debug Strategy of the PNX8525 Nexperia™ Digital Video Platform System Chip", *Proc. of Int. Test Conference*, pp. 121-130, 2001.

[Vermeulen 02] Vermeulen, B., Waayers, T. and Goel, S.K., "Core-Based Scan Architecture for Silicon Debug", *Proc. of Int. Test Conference*, pp. 638-647, 2002.

[Yang 08a] Yang, J.-S., and Touba, N. A., "Expanding Trace Buffer Observation Window for In-System Silicon Debug through Selective Capture", *Proc. of VLSI Test Symp.*, pp. 345-351, 2008.

[Yang 08b] Yang, J.-S., and Touba, N. A., "Enhancing Silicon Debug via Periodic Monitoring", *Proc. of Symposium on Defect and Fault Tolerance*, pp. 125-133, 2008.