1993

Clock Partitioning for Testability

Kent L. Einspahr
Concordia College, Seward, NE, eins@seward.ccsn.edu

Sharad C. Seth
University of Nebraska-Lincoln, seth@cse.unl.edu

Vishwani D. Agrawal
AT&T Bell Laboratories, Murray Hill, NJ

Proceedings, Third Great Lakes Symposium on VLSI, 1993. 'Design Automation of High Performance VLSI Systems'.
doi: 0.1109/GLSV.1993.224484

Clock Partitioning for Testability

Kent L. Einspahr
Concordia College
Seward, NE 68434

Sharad C. Seth
University of Nebraska
Lincoln, NE 68588

Vishwani D. Agrawal
AT&T Bell Laboratories
Murray Hill, NJ 07974

Abstract

An implementation of a design for testability model for sequential circuits is presented. The flip-flops in a sequential circuit are partitioned to reduce the number of cycles and the path lengths in each partition, thereby reducing the complexity of test generation. The implementation includes a Podem-based test generator. Preliminary results using the Contest sequential test generator are presented.

Keywords: design for testability, sequential circuit, test generation, partition

Introduction

Testability continues to be a major concern of VLSI designers because of the ever increasing demand for quality in the manufactured products. This paper presents current research towards an implementation that partitions sequential circuits to improve testability. The partitioning of the circuit lends itself to adapting the multiple clock scheme of Agrawal, Seth, and Deogun [2]. In that model, no flip-flops need to be scanned. We provide a more in-depth discussion of the unique partitioning problem involved and present some partitioning characteristics of the ISCAS-89 benchmark circuits. We propose a model for partitioning the circuit for the two-clock scheme which can be extended to a multiple clock model. The paper also discusses the modifications to the Podem algorithm which are necessary to implement the corresponding test generator. A simple example of the model is illustrated and preliminary results applying the two-clock scheme to the Contest sequential test generator [1] are presented.

Supported by NSF Grant MIP-9015115.

Partitioning of Circuit

It is known [3] that the complexity of test generation in a sequential circuit is due to the depth of the circuit. We are able to reduce the depth of the circuit by partitioning the flip-flips in order to minimize the number of cycles and the path lengths in each partition. The flip-flops in each partition are clocked identically, independent of other partitions, thus allowing for multiple clocks. Only one partition (i.e. one clock) may be active at a time. For discussion purposes, we will limit the description of the partitioning problem to two partitions.

The partitioning problem is related to the well-known feedback vertex set problem. Our problem is different and more involved since we are also concerned with the vertex set that is deleted. Ideally, we would like both sets to result in acyclic graphs. Practically, few sequential circuits can be partitioned into two acyclic vertex sets hence heuristics must be used to obtain a good partitioning. Partitioning data for the ISCAS-89 circuits is illustrated in Table 1. (Self-loops refer to the number of self-loops in an S-graph [3]).

The partitioning algorithm that we are currently using involves finding the feedback vertex set (FVS) of the S-graph and balancing the partitions. The FVS is first identified by a process similar to Lee and Reddy [6]. If the FVS is smaller than the complementary set (i.e. the non-feedback vertex set), the partitions are balanced by adding nodes from the non-feedback vertex set to the FVS until the size of the sets are more nearly the same. The two heuristics we have used to move nodes from one set to the other are that 1) no additional cycle can be formed by adding a node to the FVS and 2) an attempt is made to keep the length of the chains of flip-flops in each partition small. We obtained test generation results for benchmark circuits partitioned according to our algorithm as well as for the algorithm presented in [2]. The results obtained using our algorithm have shown improvements over those obtained using the algorithm of [2]. Our results are presented later in the paper.

1.1 Circuit	# Flip-Flops	# Self-Loops	# Acyclic Partitions (after self-loops removed)
s27	3	3	2
s208_1	8	8	1
s298	14	14	2
s344	15	15	3
s349	15	15	3
s382	21	15	4
s386	6	6	6
s400	21	15	4
s420_1	16	16	1
s444	21	15	4
s510	6	6	6
s526	21	21	2
s641	19	15	5
s713	19	15	5
s820	5	5	5

Circuit	# Flip-Flops	# Self-Loops	# Acyclic Partitions (after self-loops removed)
s832	5	5	5
s953	6	6	6
s1196	17	0	1
s1238	17	0	1
s1423	74	71	10
s1488	6	6	6
s1494	6	6	6
s5378	179	0	3
s13207	669	310	11
s13207_1	663	285	11
s15850	597	438	17
s15850_1	533	376	17
s35932	1440	288	2
s38584	1452	1098	15
s38584_1	1426	1072	15

Table 1: Partitioning characteristics of ISCAS-89 circuits.

The flip-flops that are removed from the original sequential circuit to form a partition will be referred to as the *external* flip-flops of the partition while the flip-flops that remain in the partition will be referred to as the *internal* flip-flops. Given a copy of the original circuit, a specific partition is obtained by removing all external flip-flops and replacing each deleted flip-flop by a *dummy output*, (DO), and a *pseudo input*, (SI). A *pseudo output*, (SO) is added to the output of each internal flip-flop of the partition. An example of a partitioned sequential circuit is illustrated in Figures 1-3. The pseudo output signals during CLK_1 (CLK_2) become the pseudo input signals when CLK_2 (CLK_1) becomes active.

A dummy output is placed on any line or fanout that is an input to an external flip-flop. As its name implies, the dummy output serves no function in the respective partition. Furthermore, a portion of the circuit that feeds the dummy output may also be functionally useless within the partition and will be disabled. Dummy outputs and disabled circuitry are labeled in Figures 2 and 3.

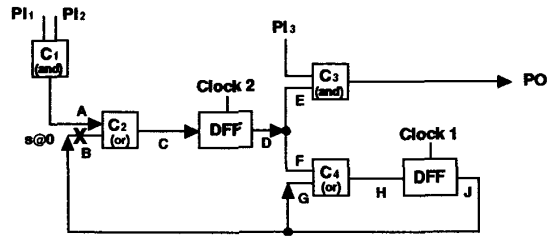

Figure 1: Complete sequential circuit C.

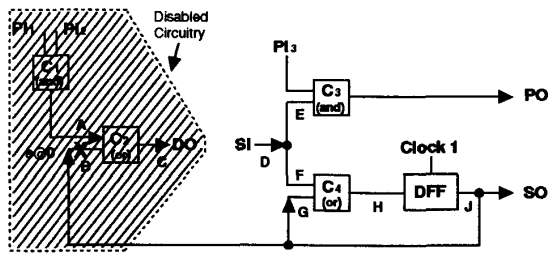

Figure 2: Partition 1. Active during Clock 1.

Test Generation

The signals at the flip-flop outputs will be determined by the state of the primary and pseudo inputs of the partition as well as by the previous states of any internal feedback signals.

Test generation using the partitioned sequential circuit is being implemented by a modified Podem algorithm that processes each partition independently in different phases. It must provide for the inter-partition communication of pseudo-output/pseudo-input states during the clock transitions. Time frame

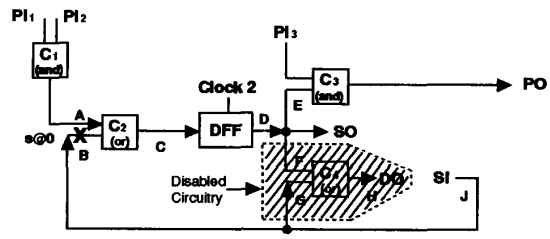

Figure 3: Partition 2. Active during Clock 2.

```
While faults remain in fault list do

    Get fault from fault list
    Initialize: Activate CLK1, De-activate CLK2

    While test for fault not found and test still possible
        /* Processing of active partition with active clock */
        Perform modified PODEM on active partition

        If fault effect propagated to PO then Return TestSuccessful

        If fault effect propagated to SO or the other partition has not been tested then
            Exchange activation of clocks and partitions
            Propagate SO of previous partition to SI of newly active partition
        Else Return NoTestPossible
        Endif

    End While
End While
```

Figure 4: High-Level Modified PODEM Algorithm

and vector processing must be integrated into the Podem framework with the ability to perform some forward and reverse time processing. The high-level description of the algorithm is given in Figure 4.

Partitioning of a sequential circuit is implemented by first replacing all flip-flops by unit-delay buffers in contrast to the zero-delay buffers used in the partial scan design of Min and Rogers [7]. Unit-delay buffers will allow the test sequence to be composed of differing test vector patterns as opposed to the identical test vectors generated by Min and Rogers' method.

Since the flip-flops are being modeled by unit-delay buffers, the input and the output of a flip-flop are separated in time. All lines in the circuit will be maintained using a vector to store the time-separated states on each line. When a unit-delay buffer is encountered during backtrace or during simulation, the time is adjusted either backward (backtrace) or forward (simulation) by appropriately referencing the previous vector element or the next vector element, respectively. The current state of the partition is given by the current element across all line vectors in the partition.

A successful test can only be obtained by eventually propagating the fault effect to a primary output. If unable to propagate the fault effect to a PO, the modified Podem algorithm attempts to drive the fault effect to a pseudo output causing a clock (partition) transition. Similarly, during backtrace, modified Podem will try to assign the primary inputs first and to assign the pseudo inputs only if necessary.

Example

In the following example, refer to Figures 1, 2, 3, and 5. Assume a stuck-at-0 fault on line B as shown. The fault lies in the disabled circuitry of partition 1 and cannot be excited in that partition. Clock 1 is set to inactive and clock 2 becomes the active clock. The initial objective is to set line B to D (i.e. good/faulty = 1/0). The initial table of vectors is shown in the bottom-leftmost box of Figure 5. Vector elements that are left blank are considered to be at X.

During the processing that follows, standard Podem techniques are followed to propagate the fault effect to a primary output in partition 2 as shown along the bottom horizontal row in Figure 5. A new time frame is entered when the D-frontier propagates through a unit-delay buffer, indicated by the assignment of values in the next element of the state vectors.

After successfully driving the fault effect to a primary output in partition 2, reverse time processing must be performed to justify the values on the pseudo inputs. In this example two previous time frames are necessary to justify the pseudo inputs and are illustrated in the (upward) vertical direction where each new row indicates a clock (partition) transition - from partition 2 to partition 1 and then from partition 1 back to partition 2.

Only a single value on each line can be propagated between partitions during the clock transition and is indicated by the vertical dashed lines between partitions.

Circuit	# FFs	# Faults	Original Test Generation				Two-Clock Test Generation			
			# Faults Detected	Coverage (%)	CPU Time (sec)	# Test Vectors	# Faults Detected	Coverage (%)	CPU Time (sec)	# Test Vectors
s208.1	8	50	33	66	245.7	382	35	70	182.2	314
s298	14	50	41	82	68.2	282	46	92	46.3	182
s344	15	50	42	84	81.8	206	43	86	59.3	146
s386	6	50	41	82	118.2	212	44	88	103.8	182
s444	21	50	40	80	159.2	346	40	80	174.3	282

Table 2: Comparison of Contest results. Original mode vs. Two-clock mode.

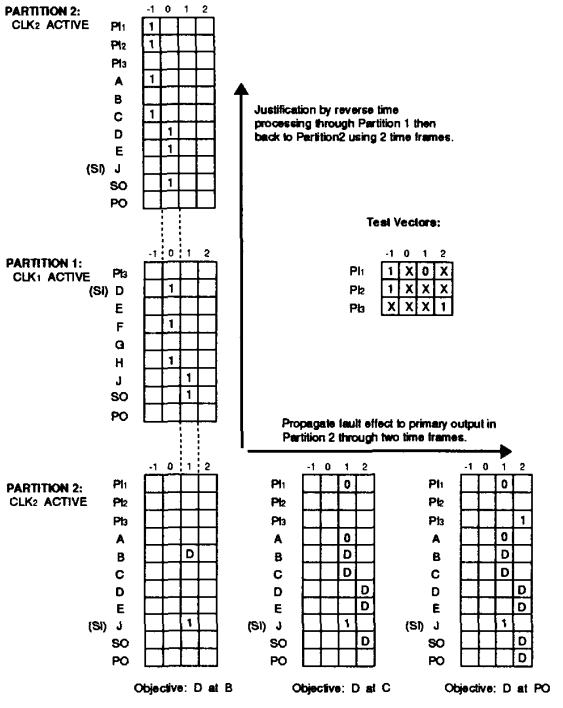

Figure 5: Vector illustration for example.

Preliminary Results

Experimental results have not yet been obtained for our modified Podem implementation. However, we have obtained some preliminary results using the Contest sequential test generator. Our purpose in conducting the experiment with Contest was to evaluate the two-clock model quickly in the context of an existing sequential test generator.

Preliminary results for some of the smaller ISCAS-89 benchmark circuits are presented in Table 2. The results were obtained using a modified implementation of the Contest sequential test generator. Contest was modified to include two extra primary inputs to select the mode for test generation. Depending upon the values assigned to the mode input bits, the test generator can execute in *normal* mode or as a two-clock test generator allowing execution in *clock-1* mode or *clock-2* mode. To obtain the results, Contest was allowed to execute in normal mode until the successful test rate fell below a certain threshold. At that point, Contest began using the two-clock mode. For the results shown in the table above, the threshold was set low enough that most of the test vectors were generated during the two-clock mode.

The results were obtained for 50 faults selected randomly from the complete fault list for each circuit. For each test circuit, the number of test vectors generated in the two-clock mode was 14%-35% smaller than the corresponding results in normal mode. Furthermore, for every circuit (except s444) the fault coverage of the two-clock test generator was greater while requiring less CPU time. These results are for the smaller benchmark circuits. We believe the improvement would be even greater for some of the larger benchmark circuits. Table 3 shows the number of flip-flops in the circuit that need to be removed to eliminate all the cycles in both the original (unpartitioned) circuit and in a partitioned circuit. It is conjectured that circuits for which the difference between the number of FF's required to produce an acyclic circuit is greater between the original circuit and the partitioned circuit, the improvement due to the two-clock scheme will be even more significant.

We are in the process of obtaining additional test results with both the modified Contest and modified Podem test generators.

Further Research

As shown in Table 1, the restriction that each partition be acyclic limits the two-clock DFT scheme to a small number of sequential circuits. Partial scan has been used to improve testability in sequential circuits [2] [4] [5] [6]. We are extending the multiple

Circuit	Total FFs	FFs Needed to Break All Cycles	
		Original Circuit	Partitioned Circuit
s27	3	1	0
s208_1	8	0	0
s298	14	1	0
s344	15	5	2
s349	15	5	2
s382	21	9	6
s386	6	5	4
s400	21	9	6
s420_1	16	0	0
s444	21	9	6
s510	6	5	4
s526	21	3	0
s641	19	7	3
s713	19	7	3
s820	5	4	3

Circuit	Total FFs	FFs Needed to Break All Cycles	
		Original Circuit	Partitioned Circuit
s832	5	4	3
s953	6	5	4
s1196	17	0	0
s1238	17	0	0
s1423	74	39	27
s1488	6	5	4
s1494	6	5	4
s5378	179	31	2
s13207	669	70	30
s13207_1	663	69	29
s15850	597	118	80
s15850_1	533	118	76
s35932	1440	297	0
s38584	1452	374	162
s38584_1	1426	374	160

Table 3: Number of FF's required to break all cycles for the ISCAS-89 circuits.

clock DFT scheme to use partial scan. The data presented in Table 3 may be a good measure of the benefits afforded by a two-clock scheme that uses partial scan. The table shows that the number of partial scan flip-flops required with the two-clock scheme is significantly reduced for circuits with many flip-flops. The number of scan FF's continues to decrease if the table is carried out to more than two partitions.

We also plan to investigate an optimal partitioning algorithm further. Most of the current work has limited the partitioning to two clocks. Further research will extend the two-clock scheme to multiple clocks.

Conclusion

We have described current research involving the implementation of a test generator using a unique partitioning of the sequential circuit. The partitioning offers a low overhead alternative to scan-based design that reduces the test generation complexity and allows a new design-for-testability scheme to be applied. The algorithm which has been implemented using a combinational test generator overcomes limitations of some recent partial scan methods. We have tested the current implementation on real circuits through the (a) construction of an automatic tool for insertion of extra logic for testability and (b) demonstration of its effectiveness on benchmark sequential circuits. Results using the two-clock DFT scheme with a sequential test generator show an improvement in the reduction of test vectors and CPU time with an increase in fault coverage.

References

[1] V.D. Agrawal, K-T. Cheng, and P. Agrawal. "A directed search method for test generation using a concurrent simulator". *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 8(2):131–138, Feb. 1989.

[2] V.D. Agrawal, S.C. Seth, and J.S. Deogun. "Design for Testability and Test Generation With Two Clocks". *Proc. CSI/IEEE International Symposium on VLSI Design*, pages 112–117, Jan. 1991.

[3] K.T. Cheng and V.D. Agrawal. "A Partial Scan Method for Circuits with Feedbacks". *IEEE Trans. Comput.*, C-39(4):544–548, April 1990.

[4] R. Gupta, R. Gupta, and M.A. Breuer. "The BALLAST Methodology for Structured Partial Scan Design". *IEEE Trans. Comput.*, C-39(4):538–544, April 1990.

[5] A. Kunzmann and H. Wunderlich. "An Analyical Approach to the Partial Scan Problem". *Journal of Electronic Testing: Theory and Applications (JETTA)*, 1(2):163–174, May 1990.

[6] D.H. Lee and S.M. Reddy. "On Determining Scan Flip-Flops in Partial Scan Designs". *Proc. of the International Conference on Computer Aided Design*, pages 322–325, Nov 1990.

[7] H.B. Min and W.A. Rogers. "A Test Methodology for Finite State Machines Using Partial Scan Design". *Journal of Electronic Testing: Theory and Applications (JETTA)*, 3(2):127–137, May 1992.