REFERENCES

[1] A. Chandrakasan, S. Sheng, and R. Broderson, "Low-power CMOS digital design," *IEEE J. Solid-State Circuits*, vol. 27, pp. 473–484, 1992.

[2] C. Nagendra, R. M. Owens and M. J. Irwin, "Power-delay characteristics of CMOS adders," *IEEE Trans. VLSI Syst.*, vol. 2, pp. 377–381, 1994.

[3] T. Callaway and E. Swartzlander, "Estimating the power consumption of CMOS adders," in *Proc. IEEE Symp. Comput. Arith.*, 1993, pp. 210–219.

[4] Preliminary Product Information, "MIPS technology R4200 microprocessor," 1993.

[5] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*. Reading, MA: Addison-Wesley, 1993.

[6] K. Yano, T. Yamanaka, T. Nishida, M. Saito, K. Shimohigashi, and A. Shimizu, "A 3.8-ns CMOS 16 × 16-bit multiplier using complementary pass-transistor Logic," *IEEE J. Solid-State Circuits*, vol. 25, pp. 388–395, 1990.

[7] M. Suzuki, K. Shinbo, T. Yamanaka, A. Shimizu, K. Sasaki, and Y. Nakagome, "A 1.5-ns 32-b CMOS ALU in double pass-transistor logic," *IEEE J. Solid-State Circuits*, vol. 28, pp. 1145–1151, 1993.

[8] R. H. Krambeck, C. M. Lee and H. S. Law, "High-speed compact circuits with CMOS," *IEEE J. Solid-State Circuits*, vol. SC-17, no. 3, pp. 614–619, 1982.

[9] J. Yetter, B. Miller, W. Jaffe, and E. DeLano, "A 100 MHz superscalar PA-RISC CPU/coprocessor chip," in *Proc. Symp. VLSI Circ. Dig. Tech. Papers*, 1992, pp. 12–13.

[10] I. Hwang and A. Fisher, "Ultrafast compact 32 b CMOS adder in multiple-output domino logic," *IEEE J. Solid-State Circuits*, vol. 24, pp. 358–369, 1989.

[11] E. Hokennek, R. Montoye and P. Cook, "Second-generation RISC floating point with multiply-add fused," *IEEE J. Solid-State Circuits*, vol. 25, pp. 1207–1212, 1990.

[12] R. Brent and H. Kung, "A regular layout for parallel adders," *IEEE Trans. Comput.*, vol. C-31, pp. 260–264, 1982.

[13] U. Ko and P. T. Balsara, "Short-circuit power driven gate sizing technique for reducing power dissipation," *IEEE Trans. VLSI Syst.*, to be published.

[14] T. Sakurai and A. R. Newton, "Delay analysis of series-connected MOSFET circuits," *IEEE J. Solid-State Circuits*, vol. 26, pp. 122–131, 1991.

Sequential Circuit Testability Enhancement Using a Nonscan Approach

Elizabeth M. Rudnick, Vivek Chickermane, Prithviraj Banerjee, and Janak H. Patel

***Abstract*— Recent studies show that a stuck-at test applied at the operational speed of the circuit identifies more defective chips than a test having the same fault coverage but applied at a lower speed. Design-for-testability approaches based on full scan, partial scan, or silicon-based solutions such as CrossCheck achieve very high stuck-at fault coverage. However, in all these cases, the tests have to be applied at speeds lower than the operation speed. In this work, we investigate various design-for-testability (DFT) techniques for sequential circuits that permit *at-speed* application of tests while providing for very high fault coverage. The method involves parallel loading of flip-flops in test mode for enhanced controllability combined with probe point insertion for enhanced observability. Fault coverage and ATG effectiveness improved to greater than 96% and 99.7%, respectively, for the ISCAS89 sequential benchmark circuits studied when these nonscan DFT techniques were used. The average area overhead for the nonscan DFT enhancements was 9.9% for standard cell implementations of three circuits synthesized from high-level descriptions, compared to 20.2% for full scan. ATG effectiveness improved to greater than 99.3% for all three circuits with the nonscan DFT enhancements.**

***Index Terms*— At-speed testing, design for testability, sequential circuits, testable chip implementation, test point insertion.**

I. INTRODUCTION

High quality is of critical importance to manufacturers of integrated circuits. Test sets having very high fault coverage are needed to ensure high quality. However, generating test sets with high fault coverage is a difficult problem, especially in highly sequential circuits. With deterministic test generators, generation of a test sequence to detect a fault typically involves fault excitation, fault effect propagation, and state justification. Good observability of the fault site is required for propagating the fault effects, and good controllability of the flip-flops is required for justifying the state. Typical design-for-testability (DFT) techniques involve improving the controllability and observability of internal circuit nodes, thereby reducing the complexity of test generation [1]. With full scan design, complete controllability and observability are provided at the flip-flops. As a result, the state justification phase is avoided, and the fault effect propagation phase is simplified. Partial scan design is a cost-effective alternative in which only a subset of the flip-flops are placed in the scan chain; state justification and fault effect propagation are simplified, and delay and area overheads are reduced relative to full scan. The disadvantages of full scan and partial scan are that vectors cannot be applied at the clock speed, and test application time is higher than in a nonscan design due to shifting of test vectors through scan chains.

Manuscript received June 10, 1993; revised March 16, 1994. This work was supported by the Semiconductor Research Corporation under Contract SRC 92-DP-109.

E. M. Rudnick was with the Center for Reliable and High-Performance Computing, University of Illinois, Urbana, IL 68101 USA. She is now with Motorola Incorporated, Austin, TX 78730 USA.

V. Chickermane was with the Center for Reliable and High-Performance Computing, University of Illinois, Urbana, IL 68101 USA. He is now with IBM Corporation, Endicott, NY 13760 USA.

P. Banerjee and J. H. Patel are with the Center for Reliable and High-Performance Computing, University of Illinois, Urbana, IL 68101 USA.

IEEE Log Number 9410846.

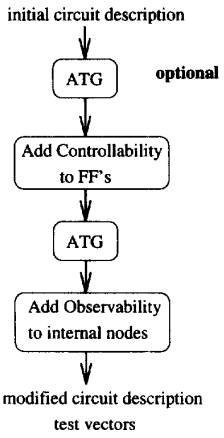

Fig. 1. DFT methodology.

Controllability and observability points are also provided in silicon-based solutions such as CrossCheck [2], [3]. With this method, test circuitry is embedded into the basic cells, making them observable; controllable flip-flop cells can also be used. The drawback of this approach is that test application must be slowed down due to scanning of observation and control points and transmission delays on long diffusion lines used to select observation and control points.

It is well known that not all defects are modeled by the stuck-at fault model. Studies have shown that applying parallel vectors (broadside vectors) at the operational speed of the circuit is more effective in identifying defective chips than applying tests with the same stuck-at fault coverage in a scan mode [4]. In our DFT methods, we provide separate controllability and observability points, allowing for high fault coverage, but more importantly, our techniques allow for parallel application of tests at the operational speed of the circuit. In this sense, our DFT methodology can be called an *at-speed* DFT method. This is in contrast to the slow test application speed in CrossCheck and the serial test application in scan mode. Our method is more closely related to parallel scan techniques [5]–[8]. Our tool OPUS-NS first selects a small set of flip-flops to be loaded in parallel from the primary inputs (PI's) when a test input line is asserted. Tests are then generated to detect faults in the modified circuit and to propagate the effects of undetected faults to internal nodes, using the HITEC sequential circuit test generator [9]. Finally, OPUS-NS selects a set of probe points to detect all faults that were excited but not detected and adds the appropriate hardware so that the probe points can be observed at the primary outputs (PO's).

II. DFT Methodology Overview

Our overall DFT methodology is illustrated in Fig. 1. The ATG tool is first run for the initial circuit. Selection of flip-flops for improved controllability can then be guided by the locations of undetected faults [10]. An incremental test generation phase follows, in which only faults not previously detected are targeted. Finally, the circuit is modified to improve the observability of internal nodes. Alternatively the first ATG run can be skipped and controllability points inserted based on testability measures. We have chosen the latter DFT methodology in the present work since it allows for modifications to be made earlier in the design phase. In addition, test generation time is reduced, since the ATG tool is used on a more testable version of the circuit only.

III. Controllability Enhancement

Several flip-flops are selected for controllability enhancement, and multiplexors are automatically inserted between the selected flip-flops and the PI's. Controllability enhancements are not necessarily restricted to the flip-flops; nodes in the combinational logic portion of the circuit could be selected as well. However, the area overhead for adding controllability at the flip-flops may be smaller if a loadable flip-flop cell is available. Furthermore, if controllability enhancements are restricted to the flip-flops, then flip-flop selection algorithms developed for partial scan can be used.

In this work, the optimization-based partial scan algorithm OPUS [11] is used to select flip-flops for controllability enhancement. OPUS-NS (OPUS for nonscan DFT) uses two testability criteria for flip-flop selection, subject to the constraint that the number of flip-flops selected is less than or equal to the number of PI's. OPUS can also be constrained to avoid selection of flip-flops that are on the critical path. The first objective is to minimize the number of cycles in the circuit graph, since cycles increase the complexity of state justification. With a careful selection of flip-flops, all cycles (except self-loops) can be broken, and the justification problem is simplified. Due to design constraints, we may not be able to break all cycles, in which case we would like to minimize the number of cycles. If the circuit is initially acyclic or can be made acyclic by selecting fewer flip-flops than the number of PI's, a second testability criterion is used. The SCOAP sequential testability analysis is used to calculate the zero-controllability, $\mathrm{SC}^0[l]$, and the one-controllability, $\mathrm{SC}^1[l]$, of each line l in the circuit. The second objective is then to select a subset of flip-flops for controllability enhancement that maximizes the reduction in SCOAP measures over all lines. Stated formally, let Θ be a circuit and let Λ be the circuit after a set S of flip-flops are made fully controllable. Let $\mathrm{Profit}(S)$ be

$$\mathrm{Profit}(S) = \sum_l (\mathrm{SC}^0[l] + \mathrm{SC}^1[l])_\Theta - \sum_l (\mathrm{SC}^0[l] + \mathrm{SC}^1[l])_\Lambda$$

Then our objective is to select a set S of flip-flops such that $\mathrm{Profit}(S)$ is maximized, while $|S| \leq \mathrm{MAX}$, where MAX is the number of PI's minus the number of flip-flops chosen with the first objective. More details are available in [11].

Fig. 2 shows how the circuit is modified to improve controllability and observability. Each PI is used to control one flip-flop in the test mode. The next state lines to the selected flip-flops are set by the PI's in the test mode, while in the normal mode, they are set by the original next-state logic. A 2-to-1 multiplexor determines this selection. Tests are generated to detect faults in the modified circuit using the HITEC sequential circuit test generator [9]. No modifications are needed in the test generator, and the time frames at which flip-flops are loaded are determined by the test generator during the normal backtracing procedure. An alternative to our loadable flip-flop arrangement can be used in which flip-flops not selected are forced into a hold mode. The hardware overhead is the same as that of full scan, however, and no significant improvements in fault coverage were found.

Other minor variations on our nonscan DFT scheme can be used to increase the number of flip-flops selected. Three such arrangements are shown in Fig. 3, where **pred**[FFi] is the predecessor of flip-flop i in the original circuit. In Fig. 3(a), multiple flip-flops are grouped together and connected to the same PI. As a result, all flip-flops in a group will load the same logic value when the test line is asserted. In Fig. 3(b), multiple flip-flops are placed in small scan chains of length 2 or 3, with the first flip-flop in each chain connected to a PI. This is different from scan-based schemes [5], [6] in that not all flip-flops are necessarily selected, and flip-flop values are not scanned out. It is also different from scan-based techniques in that neither the ATG nor the tester has to consider any scan operations. In Fig. 3(c), three flip-flops

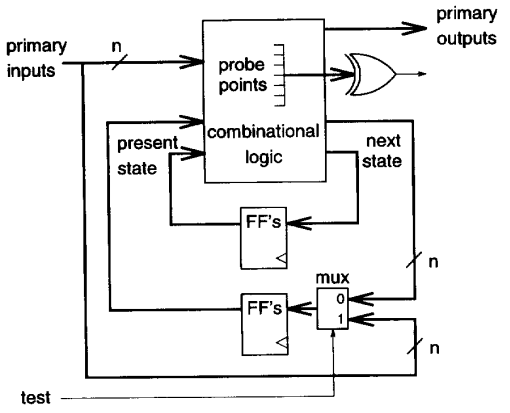

Fig. 2. Loadable flip-flops and XOR-tree for enhanced testability.

are grouped together and connected to the same PI, but at most, one of the three flip-flops is loaded in any given time frame. Two test lines and a decoder are used to generate the required load signals. In all the schemes described, the number of flip-flops selected is dependent on the number of PI's. This limitation can be easily removed by using highly-controllable internal nodes rather than PI's as control points.

IV. Observability Enhancement

The testability of a circuit can also be improved by inserting probe points. Several internal nodes are selected and connected to the appropriate hardware so that they can be observed at the PO's. A minimal set of probe points is selected that enable detection of faults previously aborted or found to be untestable due to observability problems. A binary tree of exclusive-OR (XOR) gates is used to compress the probe points into a single output; only one additional pin is required on the chip. If the number of probe points is very small, then they can be multiplexed with the existing output pins. As shown in Fig. 2, the PO's are not affected, and the probe points can be placed at any internal circuit nodes. Furthermore, any number of probe points can be selected. In contrast, scan design enhances the observability at state lines only. Any arrangement of XOR gates in the tree generates the same output value; therefore, the actual implementation can be determined at the time of placement and routing. Exclusive-OR trees were also used in [12], [13] for observability enhancement. However, full scan design was used in addition, making the hardware overhead significant.

Aliasing can occur with the XOR-tree if the fault effects propagate to an even number of probe points. This problem can be avoided by using disjoint XOR-trees, in which the probe points are partitioned so that the fault effects can propagate to an output pin [14]. If the effects of a fault f propagate to probe points p_i and p_j, then p_i and p_j should be placed in disjoint XOR-trees, each feeding a separate output. Hence, aliasing can be prevented at the cost of extra outputs. In practice, aliasing is not a serious problem in our application. For very deep XOR-trees, flip-flops can be inserted in the tree to break long paths so that the propagation delay of the XOR-tree does not exceed the clock cycle time. Then the output is delayed by the number of clock cycles equal to the number of flip-flops in the longest path.

To select the probe points, fault simulation is done using the undetected fault list, and a list of nodes is maintained to which the fault effects propagate. A minimal set of nodes is then selected that *cover* the corresponding faults, and probe points are placed at the selected nodes. The set covering problem is NP-complete, but a greedy algorithm can be used to find a near-optimal solution

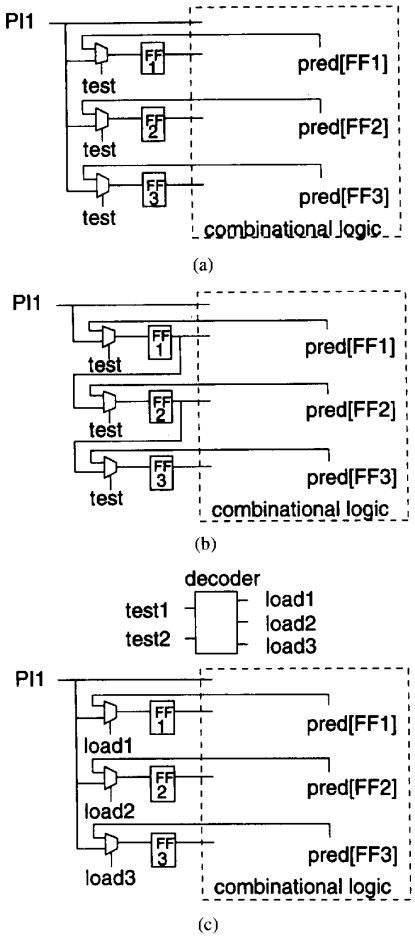

Fig. 3. Variations in nonscan controllability enhancement schemes.

in polynomial time. With this algorithm, the circuit node covering the most undetected faults is selected, and faults that it covers are removed from the fault list. Additional nodes are selected until all detectable faults are covered or the number of probe points selected is equal to the maximum number allowed. More details are available in [14].

V. Overall Flow in OPUS-NS

OPUS-NS enhances controllability and observability in the following five steps:

1) A set of flip-flops is selected to be loaded in parallel, and multiplexors are inserted connecting the selected flip-flops to the PI's. A new netlist of the modified circuit is provided.
2) Test sequences are generated using the collapsed fault list for the modified circuit.
3) Test sequences are generated to propagate fault effects to internal circuit nodes for faults not detected in step 2.
4) Fault simulation is performed with test vector sequences from steps 2 and 3 combined, and probe points are selected to cover the undetected faults.
5) An XOR-tree is inserted with connections to the selected probe points and a new PO. Alternatively, the probe points can be multiplexed with the PO's, but test generation must be repeated for the modified circuit.

TABLE I
HITEC TEST GENERATION RESULTS

Circuit	Vectors	Faults				Fault
		Total	Det	Unt	Abt	Cov
s298	259	308	265	26	17	86.0
s382	4931	399	363	8	28	90.9
s832	967	870	816	48	6	93.8
s953	14	1079	89	990	0	8.2
s1238	478	1355	1283	72	0	94.7
s1488	1192	1486	1444	40	2	97.2
s1494	1285	1506	1453	53	0	96.5
s5378	2179	4603	3233	175	1195	70.2
s35932	240	39094	34902	3984	208	89.3

TABLE II
FAULT COVERAGE IMPROVEMENT WITH TESTABILITY ENHANCEMENTS

Circuit	Fault Coverage %			
	init	con	con+obs	scan
s298	86.0	94.9	96.1	100
s382	90.9	95.3	96.5	100
s832	93.8	98.4	100	98.4
s953	8.2	100	NA	100
s1238	94.7	94.8	99.1	94.9
s1488	97.2	99.9	100	100
s1494	96.5	99.2	100	100
s5378	70.2	96.0	99.1	99.1
s35932	89.3	89.4	99.9	89.8

VI. RESULTS FOR ISCAS89 BENCHMARK CIRCUITS

A selected subset of the ISCAS89 sequential benchmark circuits having controllability and observability problems [15] was used to evaluate our nonscan DFT techniques. The circuits not used were easily testable without any DFT enhancements. HITEC test generation results for the selected circuits are shown in Table I [9], giving numbers for Total, Detected, Untestable, and Aborted faults. Fault coverage improvements with the DFT enhancements are summarized in Table II.

Fault coverage is the percentage of faults detected. Results are given for circuits with the controllability enhancements (**con**) and both the controllability and observability enhancements (**con+obs**). Results for the initial unmodified circuits (**init**) and also for full scan (**scan**) implementations are shown for comparison. For s953, the fault coverage obtained using the loadable flip-flops was 100%; therefore, no probe points were selected. For s832, s1488, and s1494, the number of PI's exceeded the number of flip-flops, so all flip-flops could be loaded. Addition of observability makes several combinationally redundant faults detectable. This accounts for higher fault coverage than for full scan in s832, s1238, and s35932. ATG effectiveness improved to nearly 100% for all circuits, where ATG effectiveness is the percentage of faults either detected or proven to be untestable within the gate-level logic simulation constraints.

The flip-flop selection, probe point selection, and test generation results are shown in Table III for circuits modified with the various testability enhancements. The execution times listed are for a SUN SPARCstation IPC and include both the test generation and fault simulation times. Columns marked *con* correspond to test generation after the hardware for improving controllability is inserted. Columns marked *prop* correspond to generation of tests to propagate the effects of undetected faults to internal nodes. Tests to propagate fault effects are required for selecting probe points. In some cases, the number of test vectors generated for fault propagation is excessive, since a separate test sequence is generated to propagate the effects of each undetected fault. A smaller test set can be obtained by repeating test generation for a modified circuit in which the selected probe points are converted to PO's. For the s35932 circuit, the number of test vectors was reduced to 1160 with this procedure, and the test generation time was 4 hours.

TABLE III
FLIP-FLOP SELECTION, PROBE POINT SELECTION, AND ATG RESULTS

Circuit	Gates	Probe Points	Flip-Flops		Testgen Time		Vectors	
			Total	Selected	con	prop	con	prop
s298	119	3	14	3	3.25m	2.0s	265	14
s382	158	3	21	3	12.3h	2.15h	2563	39
s832	287	8	5	5	2.07m	1.9s	467	28
s953	395	0	29	16	30s	NA	485	NA
s1238	508	22	18	6	40s	9.0s	488	120
s1488	653	1	6	6	2.20m	2.0s	376	2
s1494	647	11	6	6	2.42m	3.4s	376	26
s5378	2779	54	179	35	66.6h	34.1h	2170	461
s35932	17,793	276	1728	35	35.7h	24.6h	569	32,849

In addition, the nonscan schemes for improving controllability shown in Fig. 3 were investigated. First, 2 or 3 flip-flops were grouped together and connected to each PI (**grp2** or **grp3**), as shown in Fig. 3(a). Second, 2 or 3 flip-flops were chained together, with the first flip-flop connected to a PI (**chn2** or **chn3**), as shown in Fig. 3(b). Third, 3 flip-flops were connected to each PI, but at most one was loaded in any given time frame (**select**), as shown in Fig. 3(c). For these schemes, the number of flip-flops selected was equal to two or three times the number of PI's. The selected flip-flops were grouped according to the order in which they appeared in the netlist; in an actual implementation, flip-flops would be grouped according to location in order to minimize routing overhead. Fault coverage and ATG effectiveness results are shown in Table IV for circuits having sufficient numbers of selectable flip-flops. The best overall fault coverage for each circuit is highlighted in the table. Only the **select** arrangement had consistently good results as compared to the original loadable flip-flop scheme in which a single flip-flop is loaded from each PI. For the s35932 circuit, the fault coverage with the original loadable flip-flop scheme was almost as high as the fault coverage obtained for full scan; therefore, only minor improvements in fault coverage are possible for the alternative schemes, and they have a relatively high cost in hardware overhead. For the s5378 circuit, the original loadable flip-flop scheme gave the best results, showing that the best nonscan scheme depends on the design. If limiting the hardware overhead is a major goal, then the original loadable flip-flop scheme should be used.

VII. OPUS-NS: CASE STUDIES

The impact of the nonscan DFT techniques on physical design was evaluated using three circuits synthesized from high-level descriptions: an arithmetic circuit (**mult**), a 12 b microprogram sequencer (**Am2910**), and a digital signal processing (DSP) circuit (**filter**). The microprogram sequencer is control dominant, while the DSP circuit is data dominant. The arithmetic circuit falls between these two extremes.

TABLE IV
FAULT COVERAGE AND ATG EFFECTIVENESS FOR ALTERNATIVE NONSCAN SCHEMES

	Fault Coverage %					ATG Effectiveness %				
Circuit	grp2	grp3	chn2	chn3	select	grp2	grp3	chn2	chn3	select
s298	92.6	90.3	97.2	96.5	*99.7*	98.0	95.1	99.2	99.5	*100*
s382	93.7	90.9	91.2	93.5	*98.1*	94.6	92.4	92.8	97.0	*99.2*
s5378	87.3	84.9	77.1	73.0	*95.0*	89.6	86.9	79.3	75.4	*97.1*
s35932	89.9	*90.0*	89.5	89.5	89.6	*100.0*	*100.0*	99.6	99.5	99.6

The arithmetic circuit, mult, is a 16 b two's complement multiplier that uses a shift-and-add algorithm. The circuit contains an adder, several multiplexors, and registers for the accumulator, the multiplier, and the multiplicand, as well as a control unit designed using two counters, one counting the control state and the other counting the shift operations. The Am2910 12 b microprogram sequencer is similar to the one described in [16]. The sequencer has a stack of depth 5, a stack pointer, a microprogram counter, and a register counter (down counter). A 4-to-1 multiplexor selects the next microprogram address from either the input data bus, the stack, the register counter, or the microprogram counter. Sixteen instructions are accepted, with different modes used to decide the next microprogram address. The filter circuit is an 8-point infinite impulse response filter for DSP applications. It has a very regular structure with seven adders, eight fixed-point multipliers, seven 8 b registers, and seven 2-to-1 multiplexors. Each multiplier has one input fixed, and the data input and output buses are eight bits wide. The circuit has no control unit, and a global reset line is the only control signal.

VHDL descriptions of the circuits were used, and the correctness of the logic descriptions was verified at the high level. The hierarchical descriptions were then flattened to the gate level, and many of the redundancies were removed through constant propagation. Additional redundancies were removed by identifying "can't happen don't cares" at the interface of internal modules and by simplifying the logic. After some datapath resynthesis, the gate-level descriptions were verified with functional vectors. Verification of the gate-level descriptions was done hierarchically, first for each functional block and then for an entire design. Then the OCT/VEM [17] tools were used to produce physical layouts.

Several versions of each circuit were designed:

- Version 1 (*init*): Initial version of the circuit after redundancy removal and datapath resynthesis.
- Version 2 (*con*): Circuit after parallel load-based controllability enhancements are added.
- Version 3 (*con + obs*): Circuit after observability enhancements with XOR-tree are added.
- Version 4 (*scan*): Full-scan implementation.

Circuit descriptions are given in Table V, including the numbers of gates, flip-flops, PI's, PO's, and logic levels for each version. Results for test generation and physical layout are shown in Table VI. Results for full scan implementations are given for comparison. The gate counts in Table V include flip-flops. Note that the OCT/VEM design library does not have multiplexor cells; therefore, each multiplexor was implemented using four logic gates.

For the mult circuit, the redundancy removal and resynthesis phases reduced the logic gate overheads by 16.6%, and the resulting fault coverage was 96.0%. Fault coverage improved to 99.3% with the controllability enhancements alone. The area overhead was 8.1%, compared to 11.2% for full scan. Faults that were aborted or proven untestable are those that prevent initialization of the circuit. All of these faults were found to be potentially detectable with the test vectors generated. Thus, no probe points were added. Furthermore, the multiplexor insertion did not cause any delay overheads, since none of the controllable flip-flops chosen lay on the critical path. The area overhead of both the nonscan and scan designs can be reduced by designing a more compact controllable flip-flop.

TABLE V
DESCRIPTIONS OF SYNTHESIZED CIRCUITS

Circuit	Version	Gates	Flip-Flops	PIs	POs	Levels
mult	init	702	55	18	33	49
mult	con	763	55	19	33	49
Am2910	init	1056	87	20	16	43
Am2910	con	1136	87	21	16	45
Am2910	con+obs	1181	87	21	17	45
filter	init	8894	56	9	8	328
filter	con	8930	56	10	8	330
filter	con+obs	9057	56	10	9	330

TABLE VI
TEST GENERATION AND PHYSICAL LAYOUT RESULTS FOR SYNTHESIZED CIRCUITS

Circuit	Version	Vectors	Time	Faults	Det	Unt	Abt	Cov	Eff	Overhead
mult	init	273	3.65h	1708	1640	21	47	96.0	97.2	-
mult	con	279	26.7m	1800	1788	6	6	99.3	99.7	8.1%
mult	scan	70	12.0s	1708	1705	3	0	99.8	100	11.2%
Am2910	init	874	5.19h	2391	2164	171	56	90.5	97.7	-
Am2910	con	522	1.96h	2453	2262	164	27	92.2	98.9	8.5%
Am2910	con+obs	820	1.97h	2453	2395	42	16	98.0	99.3	14.5%
Am2910	scan	230	6.18m	2391	2270	159	0	93.4	100	33.3%
filter	init	347	60.2h	19,936	14,221	4827	888	71.3	95.5	-
filter	con	358	11.6h	19,992	14,342	4818	832	71.7	95.8	6.0%
filter	con+obs	924	11.9h	19,992	15,276	4716	0	76.4	100	7.0%
filter	scan	274	7.85h	19,936	15,105	4787	44	75.8	99.8	16.2%

Time: CPU time on SUN SPARCstation IPC
Abt: number of faults aborted
Faults: total number of faults
Cov: fault coverage = Det/Faults
Det: number of faults detected
Eff: ATG effectiveness = (Det + Unt)/Faults
Unt: number of untestable faults
Overhead: area overhead

For the Am2910 circuit, flip-flop selection was done using a high-level DFT tool [18], since the high-level circuit description was available. Fault coverage improved from 90.5% to 98.0% with the nonscan controllability and observability enhancements. The area overhead was 14.5%, compared to 33.3% for full scan. The overhead of the full scan implementation is much higher since all 87 flip-flops are modified, while in the nonscan implementation, only 10 flip-flops are modified. Also, the circuit with the nonscan testability enhancements has significantly higher fault coverage than the full scan circuit, since addition of the probe points makes several combinationally redundant faults detectable.

The *filter* circuit is very difficult to test because of its large sequential depth and a large number of combinational redundancies, due to each multiplier having one input tied to a fixed value. In selecting probe points, we restricted the target faults to those that were not combinationally redundant. Fault coverage improved from 71.3% to 76.4% with the nonscan controllability and observability enhancements, compared to 75.8% for full scan. The higher fault coverage indicates that some of the combinational redundancies were removed by adding the probe points. The area overhead was 7.0%, compared to 16.2% for the full scan implementation. If the combinationally redundant faults are included in the target fault list and 64 probe points are selected, then the fault coverage improves to 83.7%, and the area overhead increases to 10.2%.

REFERENCES

[1] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digit. Syst. Test., Testable Des.*, New York: Computer Science Press, 1990.
[2] T. Gheewala, "CrossCheck: A cell based VLSI testability solution," in *Proc. Des. Automa. Conf.*, 1989, pp. 706–709.

[3] S. J. Chandra, T. Ferry, T. Gheewala, and K. Pierce, "ATPG based on a novel grid-addressable latch element," in *Proc. Des. Automa. Conf.*, 1991, pp. 282–286.
[4] P. C. Maxwell, R. C. Aitken, V. Johnson, and I. Chiang, "The effect of different test sets on quality level prediction: When is 80% better than 90%?" in *Proc. Int. Test Conf.*, 1991, pp. 358–364.
[5] S. M. Reddy and R. Dandapani, "Scan design using standard flip-flops," *IEEE Des. Test*, pp. 52–54, Feb. 1987.
[6] B. Vinnakota and N. K. Jha, "Synthesis of sequential circuits for parallel scan," in *Proc. Europ. Conf. Des. Automa.*, 1992, pp. 366–370.
[7] S. Lee and K. G. Shin, "Design for test using partial parallel scan," *IEEE Trans. Comput.-Aid. Des.*, pp. 203–211, Feb. 1990.
[8] C.-H. Chen and D. G. Saab, "Behavioral synthesis for testability," in *Proc. Int. Conf. Comput.-Aid. Des.*, 1992, pp. 612–615.
[9] T. M. Niermann, "Techniques for sequential circuit automatic test generation," Univ. Illinois, Urbana, IL, Tech. Rep. CRHC-91-8, Mar. 1991.
[10] V. Chickermane and J. H. Patel, "A fault oriented partial scan design approach," in *Proc. Int. Conf. Comput.-Aid. Des.*, 1991, pp. 400–403.
[11] ——, "An optimization based approach to the partial scan design problem," in *Proc. Int. Test Conf.*, 1990, pp. 377–386.
[12] V. S. Iyengar and D. Brand, "Synthesis of pseudo-random pattern testable designs," in *Proc. Int. Test Conf.*, 1989, pp. 501–508.
[13] H. Fujiwara and A. Yamamoto, "Parity-scan design to reduce the cost of test application," in *Proc. Int. Test Conf.*, 1992, pp. 283–292.
[14] E. M. Rudnick, V. Chickermane, and J. H. Patel, "An observability enhancement approach for improved testability and at-speed test," *IEEE Trans. Comput.-Aid. Des.*, vol. 13, pp. 1051–1056, Aug. 1994.
[15] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits," in *Proc. Int. Symp. Circ. Syst.*, 1989, pp. 1929–1934.
[16] Advanced Micro Devices, "The AM2910, a complete 12-bit microprogram sequence controller," in *AMD Data Book*, AMD Inc., Sunnyvale, CA, 1978.
[17] A. Casotto, Ed., *OCTTOOLS-5.1 Part 1: User Guide*, Electron. Res. Lab., Univ. California, Berkeley, CA, Oct. 1991.
[18] V. Chickermane, J. Lee, and J. H. Patel, "Design for testability using architectural descriptions," in *Proc. Int. Test Conf.*, 1992, pp. 752–761.

Implementation of Micropipelines in Enable/Disable CMOS Differential Logic

Shih-Lien Lu

***Abstract*—This paper examines an alternative implementation of Micropipelines' logic/data processing structures. To satisfy the timing requirements of the Micropipeline, currently a delay element needs to be introduced in each of its stages. The alternative approach presented here eliminates this by using a differential CMOS logic family—Enable/Disable CMOS differential logic, ECDL instead of the conventional static CMOS. This will ease the process of synthesizing Micropipeline stages. The effectiveness of this technique in eliminating the delay requirement has been exemplified by presenting an adder implemented using ECDL.**

I. Introduction

Timing management is the key to the effective design of any large scale digital system. Though other approaches have been intriguing, a globally clocked timing discipline has been the dominant design philosophy in most of these systems. However, recently, there is a renewed interests in designing self-timed digital systems. In [1], Sutherland has convincingly pointed out that this clocked-logic conceptual framework has been pushed to its limit. He introduced a new timing discipline called transition-signalling framework, using which he described a new design methodology called micropipelines. Micropipelines use the two phase bundled data convention as proposed by Seitz [2]. For the successful operation of micropipelines using the two phase bundled data convention, the delays in data transmission across stages should be less than the delays encountered in transmitting control signals. Work in [1] uses static CMOS to implement the data processing blocks of the pipeline. This suggests that generally a delay element needs to be introduced in the control wires of the bundle to satisfy the timing requirements. Jacobs, Meng and Broderson [3]–[4] have described another approach to design using the transition signalling framework and a four phase signalling scheme [2]. They use dynamic cascoded voltage switch logic (DCVSL) [5] to implement the logic processing blocks.

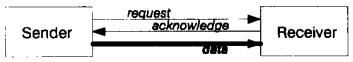

Fig. 1. Bundled data convention signals.

In this paper we describe an approach which uses the two phase signalling scheme and uses a static differential CMOS logic family—enable/disable CMOS differential logic (ECDL) [6] to implement the logic processing blocks. ECDL produces a completion signal automatically after the logic processing is completed. This completion signal when directly used follows the four phase signalling convention. This is converted to the two phase signalling framework by using an edge triggered flip-flop. The two phase signals thus generated are used as event control signals for the succeeding stages thereby eliminating the need to introduce a delay element. This delay element elimination will reduce the complexity involved in the automatic synthesis of self-timed designs. The following section describes the micropipeline as conceived by Sutherland, its equivalent circuit structure in ECDL, and the possible advantages of using this method. Section III describes an adder implemented using ECDL. Section IV presents the conclusions.

II. Micropipelines in ECDL

A. Micropipelines

The bundled data convention [2] says that the control, and data lines should be treated as a single bundle and data should always be available before the control signal. The micropipeline as conceived in [1] satisfies this condition by providing two extra lines namely the request and acknowledge lines in addition to the data lines (refer to Fig. 1). The request signal issued by the sender informs the receiver that valid data is available in the data lines. The acceptance of this data by the receiver is acknowledged by the acknowledge signal. A two phase transition signalling scheme is used, which means that both the rising and falling edges of the signals described above can flag an action. In contrast the four phase scheme [3]–[4] uses only the rising or the falling transition to flag actions.

Micropipelines both store and process data; the storage elements and the processing elements alternate along their length (Fig. 2). Note that the delay element that has been introduced is necessary to ensure that data is valid before the control signal hits, that is the delay

Manuscript received April 1, 1994; revised October 31, 1994. This work was supported by the National Science Foundation under Grant MIP-9211510.
The author is with the Department of Electrical and Computer Engineering, Oregon State University, Corvallis, OR 97331 USA.
IEEE Log Number 9410847.