VLSI DESIGN
1997, Vol. 5, No. 3, pp. 299–311
Reprints available directly from the publisher
Photocopying permitted by license only

VLSI Testing for High Reliability: Mixing I_{DDQ} Testing with Logic Testing

S. HWANG[a] and R. RAJSUMAN[b,*]

[a]*96-3 Yongun-dong, Dong-gu, Taejon, Dept. of Information and Communication Eng. Taejon University, Seoul, Korea;* [b]*1501 McCarthy Blvd. LSI Logic, MS E-171 Milpitas, Ca-95035, USA*

In this paper, we examine the effectiveness of combined logic and I_{DDQ} testing to detect stuck-at and bridging faults. The stuck-at faults are detected by the logic test and I_{DDQ} testing detects bridging faults.

Near minimal stuck-at test sets are used for this combined logic and I_{DDQ} test environment. These near minimal stuck-at test sets are generated using standard test programs, while using collapsed fault lists. We examined ISCAS '85 and ISCAS '89 benchmark circuits under this combined test environment. A comparison is given for the fault coverage obtained under this combined test environment with other studies based on pure logic test and I_{DDQ} test. Also, the results of I_{DDQ} based test sets (vectors generated specifically for I_{DDQ} testing) are compared with that of stuck-at test sets. Finally, we present a case study on a microprogrammed processor using a functional test set to detect bridging faults in I_{DDQ} testing.

Keywords: I_{DDQ} testing, bridging faults, logic testing, current testing, stuck-at testing, functional testing

1. INTRODUCTION

Nearly three decades ago, the stuck-at fault model was introduced torepresent faults in digital circuits [1]. In general, the stuck-at-1/0 fault model has worked satisfactorily for TTL and nMOS circuits. Various techniques have been developed for test generation using this classical fault model. Although, it cannot represent some important failure modes directly (including bridging and open faults) [2–4], the model is convenient and widely used.

In the recent years, CMOS has emerged as the dominant technology. It isgenerally recognized that the stuck-at fault model has serious problemsin representing CMOS failure modes; these problems include bridging faultsand transistor stuck-on faults, which may account for upto 40% of all defects [4–5]. With shrinking geometries, the significance of these faults is increasing. In CMOS, one logic value does not always dominate in case of a bridge [6]. In CMOS circuits, many bridging faults cause an output node to be connected to both Vdd and Gnd through low resistance paths, resulting in an indeterminate logic value [6–7]. The potential divider rule dictates that the output in such situations will be in-between high and low voltages. Such fault cannot be detected by conventional logic testing methods, regardless of how the test vector has been obtained.

*Corresponding author.

Researchers have discovered an effective way to detect such faults in CMOS. This method is based upon the measurement of power supply current in CMOS circuits and has proven an extremely powerful technique to detect CMOS bridging faults [8–15]. This method is generally known as *Current Testing* or I_{DDQ} Testing. The basic philosophy behind this technique is the fact that steady state power supply current in a CMOS circuit is extremely small (may be on the order of neno-amperes). In the presence of a bridging fault, when the fault is sensitized, it provides a low resistance path from power supply (Vdd) to ground (GND). Thus, there is a large steady state power supply current (I_{DDQ}) in the circuit (may be on the order of mA). When one monitors the power supply current, a faulty circuit can be identified if the circuit exhibits large I_{DDQ}.

Some of the motivations for I_{DDQ} testing are [8-15]:

1. Power supply pin provide additional accessibility— fault propagation is not required.
2. Large current results when two bridged nodes have opposite logic value.
3. Leaky transistors have conductivity values a few orders of magnitude higher than the conductivity of a fault-free transistor.
4. Minimal stuck-at test sets have a close relationship with minimal current test tests.
5. I_{DDQ} testing is also effective for detecting parametric drifts that may eventually cause a device to fail.

The main advantage of I_{DDQ} testing is that it can detect faults (such as bridging and transistor stuck-on) which cause an indeterminate logic output in CMOS circuits and hence cannot be detected by conventional logic testing. Another advantage is that I_{DDQ} testing requires only fault sensitization, fault propagation is not required. In this method, fault effect is automatically observed at the power supply pin.

It is important to realize that I_{DDQ} testing does not verify the functionality. Logic testing is still necessary to verify the correct circuit's operation. As logic testing is still needed, an important question is how effective a stuck-at test set would be if used in I_{DDQ} environment. In Ref. [13–14], this question has been addressed. However, the test quality was not examined for combined logic and I_{DDQ} testing. It is presented in this paper. We generated test vectors under stuck-at fault model covering all line stuck-at faults at the gate level. In this test generation, fault collapsing process is used to obtain near-minimal test sets. These near-minimal test sets were applied while both logic output and powersupply current were monitored.

In the following section, our test procedure is given. Section 3 includes the test results of stuck-at and bridging faults for combinational circuits and section 4 presents the results on sequential circuits. These sections also provide a comparison of bridging fault coverage by stuck-at test sets and I_{DDQ} test sets. Section 5 includes the case study on a microprogrammed processor using a functional test set to detect bridging faults in I_{DDQ} testing. The conclusions are given in section 6.

2. TEST PROCEDURE

We generated stuck-at test sets for ISCAS '85 and ISCAS '89 benchmark circuits using standard ATPGs. During this test generation process, collapsed fault list were used and hence, we obtained near minimal stuck-at test sets. I_{DDQ} testing is done with these near-minimal stuck-at test sets.

In our simulation, two external input files were required: one is a netlist file describing the circuit and other is a test vector file for the given circuit. For example, the netlist and test vector files of the C17 ISCAS '85 circuit are given in figure 1:

In the netlist file, the first line shows the number of delay elements. The second line indicates the primary input variables in order. The other lines show the type of gates, sequence number, inputs to the gates and outputs of the gates.

The I_{DDQ} simulation consists of three phases. In the first phase, the circuit netlist file is converted into

```
0;
G0,G1,G2,G3,G4;
NAND:D0,(G0,G2),(G5);          11111
NAND:D1,(G2,G3),(G6);          00101
NAND:D2,(G1,G6),(G7);          10000
NAND:D3,(G6,G4),(G8);          11011
NAND:D4,(G5,G7),(G9);          00001
NAND:D5,(G7,G8),(G10);         01100
```

(a) Netlist (b) Vectors

FIGURE 1 Netlist and test vectors of c17 circuit.

a format readable by our simulation program. In the second phase, the total number of intra-transistor bridging faults in each circuit is computed based on the netlist description of the circuit. Also, the sets of unique input values to individual gates are generated by each test vector. In this phase, the total number of bridging faults and the number of detected bridging faults between any two logical nodes are also computed for all the test vectors. Finally, in the third phase, the total number of detected intra-transistor bridging faults is computed using a table look-up scheme. In this table look-up scheme, all the bridging faults covered by each vest vector are specified together with the test vectors, types of gates and sizes of gates. Thus, the fault coverage of intra-transistor and gate-level bridging faults in I_{DDQ} testing are obtained for every combinational and sequential circuit.

3. ISCAS '85 COMBINATIONAL CIRCUITS

The MCNC has provided the netlists of digital combinational circuits. These combinational circuits are also described at the gate level in a high level language. Table I is the description of eleven circuits used in this study. Among these circuits, c432 contains 18 XOR gates, and c499 and c1355 are functionally equivalent: all XOR gates (104) of c499 have been expanded into equivalent 4-NAND gates in c1355.

Stuck-at Test Generation

The stuck-at test sets are generated for these eleven circuits. To obtain the collapsed stuck-at fault sets

TABLE I Profiles of combinational benchmark circuits

Circuit Number	Circuit Function	No. of Inputs	No. of Outputs	No. of Fanout Stems	No. of Total Lines	No. of Total Gates	No. of Cmplx. Gates
c17	?	5	2	3	17	6	0
c432	priority decoder	36	7	89	432	160	18
c499	ECAT	41	32	59	499	202	104
c880	ALU & control	60	26	125	880	383	0
c1355	ECAT	41	32	259	1355	546	0
c1908	ECAT	33	25	385	1908	880	0
c2670	ALU & control	233	140	454	2670	1193	0
c3540	ALU & control	50	22	579	3540	1669	0
c5315	ALU & selector	178	123	806	5315	2307	0
c6288	16-bit multiplier	32	32	1456	6288	2416	0
c7552	ALU & control	207	108	1300	7522	3512	0

and test sets, the automatic test pattern generator in SIS CAD tool has been used [16]. First of all, the fault collapsing is done across gates to reduce the target faults. This fault collapsing is done by removing the equivalent faults and noting the fault dominance in the fault list. For this collapsed fault list, random test generation is implemented using parallel fault simulation. This random test pattern generation is run until 320 consecutive vectors are found that do not detect a new fault. After the random patterns are simulated, the ATPG performs a deterministic search to find tests for the remaining faults based upon the algorithm described in [17]. In this algorithm, a set of equations are constructed to calculate Boolean Difference between faulty and fault-free circuit for a particular fault. Boolean satisfiability is used to find the inputs satisfying the equations and hence detect the fault. If the equations cannot be satisfied, the fault is declared redundant.

In most of the circuits as shown in Table II, 100% real stuck-at fault coverage was obtained. The pure stuck-at fault coverage is calculated as:

$$\textit{Pure Fault Coverage} = \frac{\textit{Number of detected faults}}{\textit{Total number of faults}} \quad (1)$$

The redundant faults are defined as faults which cannot be detected, the real stuck-at fault coverage is computed counting the redundant faults, and given as:

$$\frac{\textit{Number of detected faults} + \textit{Number of redundant faults}}{\textit{Total number of faults}} \quad (2)$$

Bridging Fault Coverage

Table III contains the simulation results on intra-transistor bridging faults. The intra-transistor bridging faults include source-drain, gate-drain and source-drain bridging for every transistor. The fault coverage is calculated by equation (1).

Table IV shows the simulation results of gate-level bridging faults. The gate-level bridging faults includes bridging between any two logical nodes. For example, in circuit c1355 there are 32 buffers, the 32 input-output bridging within these buffers are not detected in I_{DDQ} test. Besides these 32 faults, another 440 bridging faults between two logical nodes are also not covered.

Overall Test Results

Table V shows the coverage of line stuck-at faults, intra-transistor and gate-level bridging faults. In this

TABLE II Test results of stuck-at faults of ISCAS '85 benchmark circuits

Circuit Number	No. of Vectors	No. of Collapsed Stuck-at Faults	No. of Detected Faults	Pure Fault Coverage*	No. of Redundant Stuck-at Faults	Real Fault Coverage**
c17	6	17	17	100.00%	0	100.00%
c432	82	484	480	99.17%	4	100.00%
c499	70	928	920	99.14%	8	100.00%
c880	120	772	772	100.00%	0	100.00%
c1355	115	1240	1232	99.35%	8	100.00%
c1908	168	1783	1776	99.61%	7	100.00%
c2670	200	2519	2407	95.55%	104	99.68%
c3540	265	3281	3146	95.89%	135	100.00%
c5315	213	4956	4896	98.79%	59	99.98%
c6288	60	5840	5806	99.42%	34	100.00%
c7552	338	7009	6876	98.10%	133	100.00%

(*) Fault coverage when not counting redundant faults.

(**) Real fault coverage is $\frac{\text{\# of detected faults + \# of redundant faults}}{\text{\# of all possible collapsed faults}}$.

TABLE III Test results of intra-transistor bridging faults of ISCAS '85 benchmarks

Circuit Number	No. of Vectors	No. of Intra-Tr. Bridging Faults	No. of Detected Faults	Fault Coverage
c17	6	72	72	100.00%
c432	82	2472	2417	97.78%
c499	70	5292	5192	98.11%
c880	120	5406	5406	100.00%
c1355	115	6924	6924	100.00%
c1908	168	10338	10336	99.98%
c2670	200	16092	16003	99.45%
c3540	265	22512	22453	99.74%
c5315	213	33786	33779	99.98%
c6288	60	30336	30250	99.72%
c7552	338	46188	46152	99.92%

TABLE IV Test results of gate-level bridging faults of ISCAS '85 benchmarks

Circuit Number	No. of Vectors	No. of Gate-level Bridging Faults*	No. of Detected Faults	Fault Coverage
c17	6	55	53	96.36%
c432	82	19110	19103	99.96%
c499	70	29403	29323	99.73%
c880	120	97903	97829	99.92%
c1355	115	171991	171519	99.73%
c1908	168	416328	415381	99.77%
c2670	200	1016025	1015105	99.91%
c3540	265	1476621	1474425	99.85%
c5315	213	3086370	3084909	99.95%
c6288	60	2995128	2994937	99.99%
c7552	338	6913621	6912340	99.98%

(*) Bridging faults between any two logical nodes.

TABLE V Stuck-at and bridging faults detection of ISCAS '85 benchmarks

Circuit Number	No. of Vectors	Pure Stuck-at Fault Coverage*	Real Stuck-at Fault Coverage**	Intra-Tr. Bridging Fault Coverage	Gate-level Bridging Fault Coverage
c17	6	100.00%	100.00%	100.00%	96.36%
c432	82	99.17%	100.00%	97.78%	99.96%
c499	70	99.14%	100.00%	98.11%	99.73%
c880	120	100.00%	100.00%	100.00%	99.92%
c1355	115	99.35%	100.00%	100.00%	99.73%
c1908	168	99.61%	100.00%	99.98%	99.77%
c2670	200	95.55%	99.68%	99.45%	99.91%
c3540	265	95.89%	100.00%	99.74%	99.85%
c5315	213	98.79%	99.98%	99.98%	99.95%
c6288	60	99.42%	100.00%	99.72%	99.99%
c7552	338	98.10%	100.00%	99.92%	99.98%

(*) Fault coverage when not counting redundant faults.
(**) Fault coverage when counting redundant faults.

table, c432 and c499 show 97.78% and 98.11% coverage of intra-transistor bridging fault coverage respectively. These two circuits provide relatively low coverages comparing to other circuits due to the presence of complex XOR gates. Other circuits show very high fault coverages close to 100%. For example, c499 contains 104 complex XOR gates and in c1355 these XOR gates are expanded into 4-NAND equivalents. Therefore, c1355 has no complex gate and show better fault coverage than c499. The reason that 100% intra-transistor bridging fault coverage is not obtained is due to the presence of redundant faults.

The coverage of bridging faults between two logical nodes is shown in the last column of Table V. The circuit c17 shows 96.36% coverage of gate-level bridging faults, but all other circuits show very high fault coverage. The input-output shorts within the individual buffers in these circuits are not detectable by I_{DDQ} testing. This type of bridging faults are not counted in the fault coverage given in Table IV. From Table IV, the results indicate that these stuck-at test sets may not provide N_j^0 and N_j^1 for each logical node N_j and N_i, where N_j^0 mean '0' at node j and N_j^1 means '1' at node i. Therefore, few gate-level bridging faults remain undetected in the I_{DDQ} testing.

From Table V, it is clear that a test set which provides more than 99% real fault coverage of all single stuck-at faults in combinational circuits will provide about 98%–100% coverage of intra-transistor bridging faults, and about 99%–100% coverage of gate-level bridging faults in most cases in I_{DDQ} test for these circuits.

The coverage of bridging faults for ISCAS-85 circuits has been examined using I_{DDQ} test patterns [18]. Table VI includes the results of this study as well as our study. The results or our study and those of [18] show a close relationship between the stuck-at test set and the coverage of bridging faults in I_{DDQ} testing. *It shows that there is no need to generate specific test patterns for I_{DDQ} testing, instead stuck-at test sets can be used very effectively.*

4. ISCAS '89 SEQUENTIAL CIRCUITS

The Microelectronics Center of North Carolina (MCNC) has announced netlists of digital sequential circuits in ISCAS '89. These sequential circuits are described at the gate level netlist using a high level language. All of these sequential circuits are synchro-

TABLE VI Bridging fault detection in Iddq by stuck-at test sets vs. Iddq test sets

	Our Test Results (By Stuck-at Test Sets)*				Results From Ref. (By Iddq Test Sets)+			
Circuit Number	No. of Vectors	No. of Gate-level Bridging Faults	No. of Detected Faults	Fault Coverage	No. of Vectors	Total Faults	Covered Faults	Fault Coverage
c17	6	55	53	96.36%	.	.	.	.
c432	82	19110	19103	99.96%	17	1546	1543	99.81%
c499	70	29403	29323	99.73%	20	2747	2747	100.00%
c880	120	97903	97829	99.92%	17	3227	3227	100.00%
c1355	115	171991	171519	99.73%	18	4356	4353	99.93%
c1908	168	416328	415381	99.77%	25	4669	4663	99.87%
c2670	200	1016025	1015105	99.91%	17	13589	13563	99.81%
c3540	265	1476621	1474425	99.85%	29	16316	16295	99.88%
c5315	213	3086370	3084909	99.95%	23	40143	40117	99.94%
c6288	60	2995128	2994937	99.99%	29	21475	21468	99.97%
c7552	338	6913621	6912340	99.98%	30	53439	53388	99.90%

(*) Bridging faults between any two logical nodes.
(+) Realistic bridging faults between two logical nodes. -Proc. ITC, 1991, by Ferguson et al.

TABLE VII Profiles of sequential benchmark circuits

Circuit Number	No. of Inputs	No. of Outputs	No. of D-flipflops	No. of Total Lines	No. of Total Gates
s27	4	1	3	27	10
s298	3	6	14	298	119
s344	9	11	15	344	160
s349	9	11	15	349	161
s382	3	6	21	382	158
s386	7	7	6	386	159
s400	3	6	21	400	162
s641	35	24	19	641	379
s1196	14	14	18	1196	529
s1238	14	14	18	1238	508

nous and have D-type flip-flops. Table VII describes the nature of these circuits.

Stuck-at Test Generation

We have examined ISCAS-89 sequential benchmark circuits using minimal test sets obtained by GenTest [19]. GenTest is an automatic test pattern generator developed at AT&T Bell Labs. It combines the automatic test generator STG3 and the differential fault simulator DSIM [20]. It generates tests for a circuit described at the flip-flop and gate-level. The test generator STG3 generates a fault list for the circuitafter fault collapsing across individual gates during preprocessing. This fault collapsing is done by removing equivalent faults. Table VIII is the stuck-at test results for ten sequential circuits used in this study.

In many cases in this study 100% real stuck-at fault coverage is not obtained because of the abandoned faults which are due to limited number of backtracks or specified CPU time limit in test generation. Again pure and real stuck-at fault coverages were obtained using equation (1) and (2) respectively.

Bridging Fault Coverage

Table IX, show the statistical results of intra-transistor bridging fault coverages. In most of the cases, the overall intra-transistor bridging fault coverage is more than 98% except for s641 which has 94.55% coverage. The bridging fault coverage is calculated by equation (1).

Table X illustrates the results on gate-level bridging fault coverage. In this case, fault coverage is always more than 99%.

Overall Test Results

Table XI shows the coverage of line stuck-at, intra-transistor and gate-level bridging faults for ISCAS-89 sequential circuits. In Table XI, the coverage of intra-transistor bridging faults in I_{DDQ} is much higher than the pure stuck-at fault coverage. The main reason could be that the initialization vectors of D-type flip-

TABLE VIII Test results of stuck-at faults of ISCAS '89 benchmark circuits

Circuit Number	No. of Vectors	No. of Collapsed Stuck-at Faults	No. of Detected Faults	Pure Fault Coverage*	No. of Redundant Stuck-at Faults	Real Fault Coverage**
s27	18	32	32	100.00%	0	100.00%
s298	217	308	263	85.39%	20	91.32%
s344	149	342	329	96.20%	7	98.21%
s349	91	350	323	92.29%	22	98.48%
s382	3474	399	364	91.23%	10	93.57%
s386	247	384	314	81.77%	70	100.00%
s400	1418	424	382	90.09%	14	93.17%
s641	129	467	403	86.30%	63	99.75%
s1196	348	1242	1239	99.76%	3	100.00%
s1238	353	1355	1283	94.69%	72	100.00%

(*) Fault coverage when not counting redundant faults.

(**) Real fault coverage is $\frac{\text{\# of detected faults + \# of redundant faults}}{\text{\# of all possible collapsed faults}}$.

TABLE IX Test results of intra-transistor bridging faults of ISCAS '89 benchmarks

Circuit Number	No. of Vectors	No. of Intra-Tr. Bridging Faults	No. of Detected Faults	Fault Coverage
s27	18	288	288	100.00%
s298	217	2502	2454	98.08%
s344	149	2742	2729	99.53%
s349	91	2772	2714	97.91%
s382	3474	3180	3167	99.59%
s386	247	3114	3064	98.39%
s400	1418	3282	3250	99.02%
s641	129	4878	4612	94.55%
s1196	348	8340	8340	100.00%
s1238	353	8694	8682	99.86%

flops as well as the redundant stuck-at test vectors for certain gates provide additional coverage of intra-transistor bridging faults in I_{DDQ} testing. However, the reason that 100% intra-transistor bridging faults are not detected in most cases is due to the redundant faults. It cannot be identified that a fault was not sensitized or the fault effect was not propagated in the logic testing.

The last column in Table XI shows the coverage of bridging faults between two logical nodes. The s27 circuit shows 100% fault coverage and others show very close to 100% coverage of gate-level bridging faults. The results indicate that the reduced stuck-at test may not provide N_j^0 and N_i^1 for all logical nodes, and hence not all the gate-level bridging faults are sensitized for I_{DDQ} testing. Consequently, the results show that a test set which provides about 91% or more real stuck-at fault coverage will provide about 98%–99% coverage of intra-transistor bridging faults and about 99%–100% coverage of gate-level bridging faults in I_{DDQ} testing.

TABLE X Test results of gate-level bridging faults of ISCAS '89 Benchmarks

Circuit Number	No. of Vectors	No. of Gate-level Bridging Faults*	No. of Detected Faults	Fault Coverage
s27	18	136	136	100.00%
s298	217	9180	9144	99.61%
s344	149	16836	16792	99.74%
s349	91	17020	16945	99.56%
s382	3474	16471	16439	99.81%
s386	247	14706	14675	99.79%
s400	1418	17766	17727	99.78%
s641	129	93528	92493	98.89%
s1196	348	157080	157007	99.95%
s1238	353	145530	145519	99.99%

(*) Bridging faults between any two logical nodes.

The stuck-at fault coverage of ISCAS-89 circuits has been examined in I_{DDQ} environment using I_{DDQ} test patterns, and compared with the logic test results of GenTest [21]. Table XII includes the results of this study as well as GenTest, our results of intra-transistor and gate-level bridging fault are also included in Table XII. The second column in Table XII shows the number of collapsed stuck-at faults. The real stuck-at fault coverage of GenTest can be compared with the I_{DDQ} based fault coverage of [21]. In this comparison, the I_{DDQ} based fault coverage is higher. The results of our study and that of [21] clearly show very high coverage of bridging and stuck-at faults in I_{DDQ} testing.

5. CASE STUDY ON A MICROPROGRAMMED PROCESSOR

A functional test set for a microprogrammed processor was developed to detect logical faults [22]. Since our test generation scheme is developed for a 1-bit processor and extension is done based upon n-cascaded copies, the test set of n-bit processor is relatively small. For 1-bit processor, we generate test vectors for each blocks using its functional description. After test generation is completed, all test vectors arecarefully reviewed and combined together to form a complete test set for one functional module. During this review and compaction process, redundancy and duplicate test vectors are deleted to obtain a minimal test set.

A microprogrammed processor contains a microsequencer and an arithmetic logic unit (ALU) as shown in figure 2. The generalized ALU module is designed to implement arithmetic and logic functions defined for a particular system. It contains a multiplexer or a decoder, a combinational logic block which actually implements the functions and a register to hold the results. For simplicity, we assume that the combinational logic block can perform only four different operations. The truth-tables of the mux and

TABLE XI Stuck-at and bridging faults detection of ISCAS '89 benchmarks

Circuit Number	No. of Vectors	Pure Stuck-at Fault Coverage*	Real Stuck-at Fault Coverage**	Intra-Tr. Bridging Fault Coverage	Gate-level Bridging Fault Coverage
s27	18	100.00%	100.00%	100.00%	100.00%
s298	217	85.39%	91.32%	98.08%	99.61%
s344	149	96.20%	98.21%	99.53%	99.74%
s349	91	92.29%	98.48%	97.91%	99.56%
s382	3474	91.23%	93.57%	99.59%	99.81%
s386	247	81.77%	100.00%	98.39%	99.79%
s400	1418	90.09%	93.17%	99.02%	99.78%
s641	129	86.30%	99.75%	94.55%	98.89%
s1196	348	99.76%	100.00%	100.00%	99.95%
s1238	353	94.69%	100.00%	99.86%	99.99%

(*) Fault coverage when not counting redundant faults.
(**) Fault coverage when counting redundant faults.

combinational logic block are given in Table XIII and Table XIV respectively.

The micro-sequencer generates the addresses of microinstructions stored in micro controlmemory. The input decoder of micro memory decodes this address and thus, the microinstruction is fetched and executed. The micro-sequencer has four main blocks: an instruction register (IR), and address register (AR), a multiplexer or a decoder and an incrementer. The mux selects on of three incoming inputs depending upon the values of C_0 and C_1. These values are available from the micro control memory or from an external interrupt circuit. The functionality of the micro-sequencer mux is described in Table XV.

A functional fault is defined as a fault under which the circuit does not carry out a predefined operation or it carries out an operation which was not defined. If a fault does not cause a change in functionality, we consider the fault to be outside the domain of our consideration. The effect of the fault can only be observed at the primary output lines in logic testing. Our model covers the following faults:

1. Any single bridging fault between two lines at at the functional block without feedback effect. We assume only low resistance bridging or hard short. Such a fault is assumed to cause a logical OR or logical AND effect at the faulty lines.

TABLE XII Fault detection by stuck-at test sets vs. I_{DDQ} test sets

			Gen Test Results						Our Test Results+		Results From Reference*
			Stuck-at Faults in Logic Test				Bridging Faults in Iddq		Stuck-at Faults in Iddq Test		
Circuit Number	No. of Col. S-a Faults	# Vectors	# Detected	Pure Fault Coverage	# Redundant	Real Fault Coverage	Intra-Tr. Fault Cov.	Gate-level Fault Cov.	# Vectors	# Detected	Fault Coverage
s27	32	18	32	100.00%	0	100.00%	100.00%	100.00%	7	32	100.00%
s298	308	217	263	85.39%	20	91.32%	98.08%	99.61%	63	305	99.03%
s344	342	149	329	96.20%	7	98.21%	99.53%	99.74%	46	338	98.83%
s349	350	91	323	92.29%	22	98.48%	97.91%	99.56%	48	345	98.57%
s382	399	3474	364	91.23%	10	93.57%	99.59%	99.81%	192	399	100.00%
s386	384	247	314	81.77%	70	100.00%	98.39%	99.79%	41	381	99.22%
s400	424	1418	382	90.09%	14	93.17%	99.02%	99.78%	199	421	99.29%
s641	467	129	403	86.30%	63	99.75%	94.55%	98.89%	51	440	94.22%
s1196	1242	348	1239	99.76%	3	100.00%	100.00%	99.95%	60	1214	97.75%
s1238	1355	353	1283	94.69%	72	100.00%	99.86%	99.99%	59	1327	97.93%

(*) Iddq-based test patterns are used. -Proc. ITC, 1990, by Fritzemeier et al.
(+) Stuck-at test patterns are used.

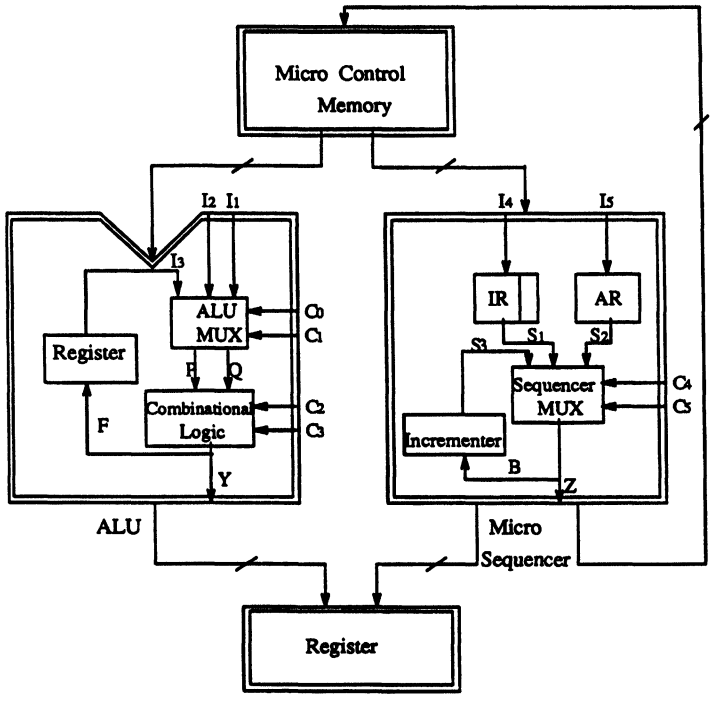

FIGURE 2 Diagram of a generalized microprogrammed processor.

TABLE XIII Truth table of ALU MUX

C_0	C_1	Function of the MUX
0	0	$P = I_1$, $Q = I_2$
0	1	$P = I_2$, $Q = I_3$
1	0	$P = I_3$, $Q = I_1$
1	1	P = O, Q = 0

2. Any single stuck-at fault at any line. In this situation, the faulty line changes the functionality for at least one set of inputs, and hence the fault is detected.
3. Any wrong selection of operations. This implies the fault at the control lines or bad design of circuits. As the functionality of the block changes for at least one set of inputs, the fault is detectable.

TABLE XIV Truth table of combinational logic

C_2	C_3	Function of the block
0	0	P AND Q
0	1	P OR Q
1	0	P XNOR Q
1	1	P XOR Q

This model is expanded to include intra-transistor and gate level bridging faults in I_{DDQ} testing.

5.1 Test Results on 1-bit Processor

In our logic test set, we have minimal number of test vectorsto detect any functional fault caused by either stuck-at or bridging in any basic block. We generated two test sets to test a 1-bit slice microprocessor: one set for ALU and the other set for micro-sequencer. The test set for ALU contains 11 vectors and the test set for micro-sequencer contains 8 vectors. We evaluated the fault coverage of each functional block with its own test set. When I_{DDQ} testing is done with these logic test vectors, the coverage of intra-transistor

TABLE XV Truth table of micro-sequencer MUX

C_4	C_5	Operation
0	0	$Z = S_1$
0	1	$Z = S_2$
1	0	$Z = S_3$
1	1	Z = 0

bridging faults of 1-bit slice 3-to-2 mux is found to be 98.04%. In the same way, 1-bit slice 3-to-1 mux shows 100% fault coverage, and combinational logic shows 91.87% fault coverage. The 1-bit slice register and incrementer also show 100% fault coverage each.

Overall, the ALU shows 96.64% coverage of intra-transistor bridging faults and micro-sequencer shows 100% coverage. Consequently, the overall bridging fault coverage of 1-bit slice processor is about 98.56%, which covers most of the intra-transistor bridging faults. It should be noted that this overall fault coverage is the minimal coverage because the test vectors generated for each functional block in the same module can cover the intra-transistor bridging faults in other blocks in I_{DDQ} testing.

5.2 Test Procedure for n-bit Processor

The test scheme for n-bit processor is an extension of 1-bit slice processor. Each n-bit functional block is simply the cascade connection of n 1-bit slices. These n-bit blocks are tested by the extended test sets of 1-bit blocks. We use the following assumptions:

Assumption 1: While generating tests, the bridging faults between non-adjacent input lines, between non-adjacent output lines and between non-adjacent control lines are not considered. In literature, such faults are reported as unrealistic. Thus, this assumption does not provide exhaustive fault coverage, but the results are still applicable in any general situation.

Assumption 2: To test a bridging fault, two lines should carry opposite logic values. Bridging between two lines may also cause oscillations in the circuit, such faults are not considered here.

The n-bit combinational logic needs 7 test vectors and the n-bit register needs 6 vectors. The n-bit 3-to-1 and 3-to-2 mux's also need 8 and 7 vectors respectively. The n-bit incrementer needs variable number of vectors depending on n because of the carry propagation, the required number of vectors is n + 5.

For combinational logic, the coverage of intra-transistor bridging faults in I_{DDQ} testing is found to be 91.87% (same as 1-bit slice). The register and 3-to-1 mux showed 100% fault coverage, while 3-to-2 mux showed 98.04% coverage of intra-transistor bridging faults. The n-bit incrementer with (n + 5) vectors showed the following coverage of intra-transistor bridging faults:

$$100\left(\frac{Q - \sum_{i=0}^{n-2} i}{Q}\right)\% \tag{3}$$

$$\text{where } Q = 24 + 60n + 6\sum_{i=2}^{n}(i+1) \tag{4}$$

For example, for n = 8, the 8-bit incrementer provides 97.22% coverage of intra-transistor bridging faults.

Since the processor contains ALU and micro-sequencer, the coverages of intra-transistor bridging faults for these two n-bit modules are: 96.64% for ALU and $(100 - 25\sum_{i=0}^{n-2} i/\mathrm{Q})$ % for micro-sequencer. These coverages are minimal with the same reason as the 1-bit slice. For example, for n = 8, the 8-bit micro-sequencer shows 99.31% fault coverage in I_{DDQ} testing.

6. CONCLUSIONS

Logic testing has been used for many years to detect logical faults in the circuits. Specially, stuck-at test scheme is still popular because of its simplicity. Many bridging faults in CMOS circuits result in neither wired-OR nor wired-AND, but rather cause an indeterminate voltage at the fault-site. Hence, logic testing for CMOS bridging faults is ineffective and only I_{DDQ} testing can overcome this problem.

In this study, we examined near-minimal stuck-at test sets for combinational and sequential circuits. These test sets detect stuck-at faults in the logic test, and intra-transistor and gate-level bridging faults in I_{DDQ} test. We noted that these stuck-at test sets provide very high bridging fault coverages in I_{DDQ} testing. We also noted in a particular case study on a bit-slice microprocessor that a small functional test set provides very high coverage of bridging faults in I_{DDQ} testing.

Based upon these studies, we conclude that the stuck-at tests can be reviewed to obtain better quality by detecting bridging faults in I_{DDQ} testing. However, in the test generation, the stuck-at fault coverage should not be lowered for logic testing. We noticed that even though fault propagation toward primary outputs is not possible for certain stuck-at faults in the logic testing (which means redundant fault), it is still desirable to find-out input stimulus which sensitizes the fault. This input may provide necessary value to individual gates to possible detect some more bridging faults in I_{DDQ} testing.

We suggest that during stuck-at test generation, fault sensitization function should be implemented such that its results are available separately. The results of fault sensitization procedure can be used very effectively in the I_{DDQ} testing.

Acknowledgement

The authors would like to express their sincere thanks to Scott Davidson of AT&T Bell Labs for providing test vectors for sequential circuits using GenTest.

References

[1] Roth, J. P. (1966). "Diagnosis of automata failures: A calculus and method", *IBM J. Res. and Dev.*, vol. 13.

[2] Wadsack, R. L. (1978). "Fault modeling and logic simulation of CMOS and MOS integrated circuits", *Bell System Tech. J.*, (May–June) pp. 1449–1474.

[3] Jain, S. K. and Agrawal, V. D. (1983). "Test generation for MOS circuits using D-algorithm", *Proc. 20th Design Auto. Conf.*, pp. 64–70.

[4] Galiay, J., Crouzet Y. and Vergniault, M. (1980). "Physical versus logical fault models MOS LSI circuits, impact on their testability", *IEEE Trans. Comp.*, vol. C-27, (June), pp. 527–531.

[5] Maly, W., Ferguson, F. J. and Shen, J. P. (1985). "Inductive fault analysis of MOS integrated circuits", *IEEE Design and Test of Comp.*, (December), pp. 13–26.

[6] Malaiya, Y. K., Jayasumana, A. P. and Rajsuman, R. (1986). "A detailed examination of bridging faults", *Proc. Int. Conf.*, Computer Design, pp. 78–81.

[7] Rajsuman, R., Malaiya Y. K. and Jayasumana, A. P. (1989). "Limitations of switch level analysis for bridging faults", *IEEE Trans. CAD*, vol. 8(7), pp. 807–811.

[8] Malaiya, Y. K. and Su, S. Y. H. (1982). "A new fault model and testing technique for CMOS devices", *Proc. Int. Test Conf.*, pp. 25–34.

[9] Hawkins, C. F., Soden, J. M., Fritzemeier, R. R. and Horning, L. K. (1989). "The use of quiescent power supply current measurement in detection of defects CMOS ICs", *IEEE Trans. Indust. Electronics*, vol. 36(2), (May), pp. 211–218.

[10] Lee, K. J. and Breuer, M. A. (1992). "Design and test rules for CMOS circuits to facilitate IDDQ testing of bridging faults", *IEEE Trans. CAD*, vol. 11(5), (May), pp. 659–670.

[11] Rajsuman, R. (1992). Digital hardware testing, chapter 11, "Current Testing", Artech House Inc..

[12] Malaiya, Y. K. and Rajsuman, R. (1992). "Bridging faults and I_{DDQ} testing", *IEEE Computer Society Press*, Technological Series.

[13] Hwang, S., Rajsuman, R. and Davidson, S. (1994). "Detect efficiency of stuck-at test sets on bridging faults in I_{DDQ} environment", *Proc. Int. Conf. on VLSI Design*, Calcutta, India.

[14] Hwang, S. and Rajsuman, R. (1993). "Effectiveness of stuck-at test sets to detect bridging faults in I_{DDQ} environment", *Proc. IEEE Asian Test Symp.*, Beijing, China.

[15] Keating, M. and Meyer, D. "A new approach to dynamic Idd testing", *Proc. Int. Test Conf.*, pp. 316–321, 1987.

[16] SIS: A system for sequential circuit synthesis, Memo No. UCB/ERL/M92/41, Electronics Research Lab, *Dept. of Elect. Eng. and Computer Sc., UCB*, (May) (1992).

[17] Larrabee, T. (1989). "Efficient generation of test patterns using boolean difference" *Proc. Int. Test Conf.*, pp. 795–801.

[18] Ferguson, F. J. and Larrabee, T. (1991). "Test pattern generation for realistic bridging faults in CMOS ICs", *Proc. Int. Test Conf.*, pp. 492–499.

[19] Cheng, W. T. and Chakraborty, T. (1989). "GenTest: An automatic test-generation system for sequential circuits", *IEEE Computer*, (April), pp. 43–49.

[20] Cheng, W. T. and Yu, M. L. (1989). "Differential fault simulation - a fast method using minimal memory", *Proc. Design Auto. Conf.*, pp. 424–428.

[21] Fritzemeier, R. R., Soden, J. M., Treece, R. K. and Hawkins, C. F. (1990). "Increased CMOS IC stuck-at fault coverage with reduced I_{DDQ} *test sets",* Proc. Int. Test Conf., *pp. 427–435.*

[22] Hwang, S., Rajsuman R. and Malaiya, Y. K. (1990). "On the testing of microprogrammed processors", *Proc. Int. Symp. on Microprogramming and Microarchitecture*, pp. 260–266.

Authors' Biographies

Suntae Hwang is a faculty member of the Information and Communications Engineering Department at Taejon University in Korea. Previously, he was a research scientist of Korea Institute of Science and Technology from 1979 to 1982. He also was a lecturer of EE Department at Cleveland State University, OH in 1988 and 1989. From 1993 to 1995, he worked at Hyundai Electronics Research Center in Korea as an ASIC designer.

His research interests include VLSI design, testing and parallel architecture. Hwang received his M.S.

and Ph.D. degrees in Computer Engineering and Science from Case Western Reserve University. Prof. Suntae Hwang Dept. of Information & Communication Eng., TAEJON University, 96-3 Yongun-dong, Dong-gu, Taejon, KOREA Tel) 82-42-280-2554 fax) 82-42-284-0109 e-mail) hwang@ice.taejon.ac.kr

Rochit Rajsuman received Ph.D. EE from Colorado State University in 1988. From 1988 to 1995 he was Assistant Professor at Case Western Reserve University in the department of Computer Engineering and Science. He also held a secondary appointment in the department of Electrical Engineering. Since 1995, he is with LSI Logic Corporation.

He has published more than 50 papers and authored two monographs, Digital Hardware Testing and IDDQ Testing for CMOS VLSI, both published by Artech House Publishers, Norwood, MA. He has also authored four patents.

He has served on program committees on various conferences including Steering committee chair for IEEE Int. Workshop on Iddq Testing. He was founder and General Chair for the IEEE Int. Workshop on Memory Technology, Design and Testing. Since 1995, he has served as steering committee chair for the same workshop. Rochit Rajsuman 1501 McCarthy Blvd., LSI Logic, MS E-171, Milpitas, CA 95035 (408) 433-8789 rajsuman@lsil.com