Resistive Bridging Fault Simulation of Industrial Circuits*

Piet Engelke* Ilia Polian* Juergen Schloeffel** Bernd Becker*

*Albert-Ludwigs-University
Institute for Computer Science
Georges-Köhler-Allee 51
79110 Freiburg im Breisgau, Germany
{engelke|polian|becker}@informatik.uni-freiburg.de

**NXP Semiconductors GmbH
Design Technology Center
Georg-Heyken-Str. 1
D-21147 Hamburg, Germany
juergen.schloeffel@nxp.com

Abstract

We report the successful application of a resistive bridging fault (RBF) simulator to industrial benchmark circuits. Despite the slowdown due to the consideration of the sophisticated RBF model, the run times of the simulator were within an order of magnitude of the run times for pattern-parallel complete-circuit stuck-at fault simulation. Industrial-size circuits, including a multi-million-gates design, could be simulated in reasonable time despite a significantly higher number of faults to be simulated compared with stuck-at fault simulation.

Keywords: Resistive bridging faults, bridging fault simulation, case study

1 Introduction

Defect-based test (DBT) [1, 2] is a methodology to improve the quality of micro- and nanoelectronic products by employing accurate models of actual defects showing up in the silicon. DBT is used to complement standard test methods which are based on the stuck-at fault model. It has long been known that many defects are not adequately represented by the stuck-at fault model [1, 3–6]. Although a large share of defect population is detected by stuck-at tests, the product quality level is often inadequate when only the stuck-at fault model is used. DBT helps to increase the quality level by explicitly targeting and detecting defects not covered by the stuck-at fault model.

The conventional approach to testing micro- and nanoelectronic circuits includes test generation using the stuck-at fault model and application of the generated test patterns to the circuit by the automatic test equipment. The list of faults to be considered is derived directly from the gate-level netlist of the circuit. In contrast, pattern generation for DBT may require additional information, such as circuit layout or technology parameters [7–9].

Fault simulation is a key element of a DBT flow. Although test generation methods for non-standard fault models do exist [3, 10–13], they do not always scale for industrial-size circuits. Hence, it is important to determine the coverage of realistic defects by the existing test pattern sets (where the test pattern sets might have been created using conventional fault models). Then, defects missed by that sets could be addressed pinpointedly using expensive defect-based test generation approaches. Alternatively, techniques such as n-detection [14] or its extensions [15, 16] could be employed. These techniques increase the accidental detections of non-targeted defects by targeting the same stuck-at fault $n > 1$ times. Defect-based fault simulation is useful in determining the value of n which leads to an adequate defect coverage, as overestimating n would lead to large test patterns and increased cost of test application.

Resistive short defects have been an important defect class in the past [17], and their relevance continues to grow. Within a DBT framework, short defects are represented by bridging faults. Simple bridging fault models ignore the resistance of the defect [3, 18–23]. Resistive bridging faults (RBF) are modeling this aspect with a higher degree of accuracy [16, 24–35]. Short defect resistance R_{sh} is a random parameter not known in advance. Hence, an RBF simulator calculates for a given fault the range of resistances in which a given test pattern set detects the fault. This range is called *analogue detectability interval* or ADI [24, 25].

An ADI is often of the shape $[0, R_{\max}]$, where $R_{\max}$ is the maximal R_{sh} value for which the fault is detected. However, an ADI can also be a union of disjoint intervals [33]. Once the ADI is known for each fault in the fault list, the fault coverage is obtained by relating C-ADI, the range of resistances detected by any test pattern from the test pattern set, to G-ADI, the range of resistances detectable by any possible

*Parts of this work are supported by the German Research Foundation under grant BE 1176/14-1.

test pattern [33, 34]. This approach is different from the conventional bridging fault simulation, where any fault is either detected by the test pattern set or not and the fault coverage is the fraction of the detected faults. Calculation of G-ADI is NP complete [34]; it can be obtained by exhaustive simulation or an ATPG procedure [36], or approximated [30, 34].

For fault simulator prototypes from the literature, no experimental data for industrial-size circuits have been reported [25, 30, 34, 37]. In this paper, we demonstrate that RBF simulation can be performed for multi-million gates designs. By employing the sectioning technique [37], a resistive bridging fault is reduced to a number of multiple stuck-at faults. The actual simulation is performed by a multiple stuck-at fault simulation engine which implements the usual speed-ups such as parallel-pattern simulation. The ADI is calculated from the detection status of the multiple-stuck-at faults rather than by interval operations as done in [30, 34]. The accuracy of the result is not affected by the sectioning technique, the obtained ADI is neither over- nor underapproximated.

We report results for large industrial circuits provided by NXP. While the simulation data reported earlier assumed 10,000 faults or less in the fault list, we employ fault lists containing $10 \times S$ faults where S is the number of gates in the circuit. (This appears to be a realistic number of faults when inductive fault analysis [7] with no fault list truncation is done.) We also present results for ISCAS85, ISCAS89 and ITC99 circuits using identical setup. For comparison, we report results for stuck-at fault simulation of the same circuits using the same simulation engine. It turns out that, while the complexity of the RBF simulation is higher than the complexity of the stuck-at fault simulation, the overhead is limited and does not differ significantly for different classes and sizes of circuits.

The remainder of the paper is organized as follows. In Section 2 we briefly reiterate the techniques used in the simulation tool [38]. Experimental results are reported in Section 3. Section 4 concludes the paper.

2 Resistive Bridging Fault Simulation

In a fault-free circuit, the voltage level V_a at the output a of the logic gate A is either $V_a = V_{DD}$ if the logical value at a is 1 or $V_a = 0$V if the logical value at a is 0. If two lines (say, outputs a and b of gates A and B) are connected by a resistive short defect with a resistance R_{sh} and have opposing logic values (say, logic-1 at a and logic-0 at b), then V_a will assume some value below V_{DD} and V_b will assume some value above 0V. The exact values of V_a and V_b depend on the parameters of the transistors in gates A and B, the number of driving transistors (i.e., the logical values applied to the inputs of gates A and B) and R_{sh} [24].

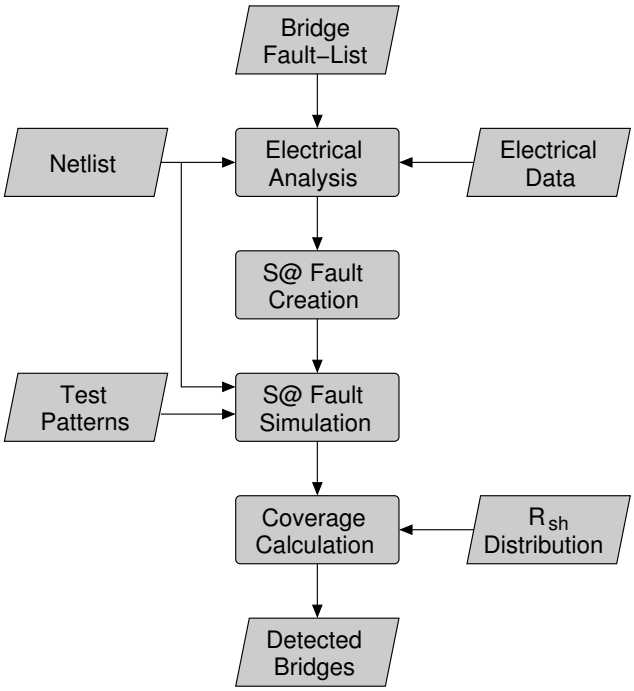


Figure 1: Resistive bridging fault simulation flow

The intermediate voltages V_a and V_b will be interpreted by the gates succeeding lines a and b as either logic-1 or logic-0, depending on the logic threshold of the gates. The logic threshold depends on the parameters of the transistors within the succeeding gate; different inputs of a gate generally have different logic thresholds. The defect is detected if at least one succeeding gate interpreted the voltage on a line involved in the bridge as a faulty logical value and this value propagated to an output through a sensitized path. Multiple fault effects could be propagated through reconverging paths, resulting in fault effect cancellation for certain values of R_{sh} [33]. Further pattern-dependent effects including the multiple strengths problem are described in [34].

A critical resistance R^{crit} is a value of R_{sh} such that at least one succeeding gate interprets the voltage at its output as logic-1 for $R_{sh} < R^{crit}$ and logic-0 for $R_{sh} > R^{crit}$, or vice versa. For instance, if the logic threshold of a gate succeeding line a is Th, R^{crit} corresponds to R_{sh} for which $V_a = Th$. The fault could be detected for $R_{sh} < R^{crit}$ (provided that there is a sensitized path to an output) and is not detected for $R_{sh} > R^{crit}$. This holds only for a given assignment I of logic values to the inputs of gates A and B; critical resistances for other input assignments of A and B than I may differ.

Let the number of different critical resistances for a given fault (pair of bridged nodes) be n. Let $R_1^{crit} < R_2^{crit} < \cdots < R_n^{crit}$ be the sorted list of the critical resistances and let R_0^{crit} be set to 0Ω. A section is an interval $[R_{i-1}^{crit}, R_i^{crit}]$

Circuit	Cells	PI	PO	Resistive bridging fault simulation					Stuck-at fault simulation		
				Faults	FC	Preproc. [s]	Sim. [s]	T/BV [ms]	Faults	FC	Time [s]
p35k	48,927	2,912	2,229	489,270	82.44	3.51	974.34	0.00020	67,733	58.68	12.88
p45k	46,075	3,739	2,550	460,750	97.74	3.68	381.85	0.00008	68,760	93.04	9.27
p77k	75,033	3,487	3,400	750,330	78.50	6.13	29,236.90	0.00390	120,348	59.51	2158.37
p78k	80,875	3,148	3,484	808,750	97.85	6.76	551.16	0.00007	163,310	100.00	3.59
p81k	96,722	4,029	3,952	967,220	87.17	12.15	1,788.86	0.00018	204,174	69.27	19.15
p89k	92,706	4,683	4,557	927,060	92.00	8.34	1,688.96	0.00018	150,538	75.89	28.66
p100k	102,443	5,902	5,829	1,024,430	98.28	8.54	868.72	0.00008	162,129	93.57	25.37
p141k	185,360	11,290	10,502	1,853,600	98.02	15.45	1,178.11	0.00006	282,428	91.50	25.63
p267k	296,404	17,332	16,621	2,964,040	97.10	20.63	2,189.27	0.00007	366,871	90.41	45.22
p269k	297,497	17,333	16,621	2,974,970	97.12	21.82	2,428.70	0.00008	369,055	90.59	45.60
p295k	311,901	18,508	18,521	3,119,010	90.80	26.44	5,024.21	0.00016	472,022	77.63	73.36
p330k	365,492	18,010	17,468	3,654,920	96.12	28.98	2,842.00	0.00008	540,758	86.66	64.87
p378k	404,367	15,732	17,420	4,043,670	97.96	36.00	2,913.98	0.00007	816,534	100.00	25.48
p388k	506,034	25,005	24,065	5,060,340	98.87	46.88	2,784.74	0.00006	881,417	96.06	75.76
p469k	49,771	635	403	497,710	98.43	9.24	27,704.30	0.00557	142,751	98.53	2951.41
p951k	1,147,491	92,027	104,747	11,474,910	99.01	95.52	4,783.05	0.00004	1,557,914	95.32	136.67
p1522k	1,193,824	71,414	68,035	11,938,240	93.26	106.87	17,884.31	0.00015	1,697,662	80.91	301.88
p2927k	2,539,052	101,844	95,143	25,390,520	96.57	183.24	29,328.41	0.00012	3,527,607	88.56	1146.49
Average					94.29	35.57	7,475.10	0.00062		85.90	397.20

Table 1: Experimental results for combinational cores of NXP circuits, 10,000 test vectors

bounded by two critical resistances which does not contain any further critical resistance ($1 \leq i \leq n$) [37]. The logical behavior of the circuit for any two values of R_{sh} from the same section $[R_{i-1}^{crit}, R_i^{crit}]$ is identical.

For section $[R_{i-1}^{crit}, R_i^{crit}]$ and an input assignment I, it is possible to determine a multiple stuck-at fault f_i^I such that the circuit behavior under f_i^I exactly corresponds to the behavior of the circuit under the bridging fault with $R_{sh} \in [R_{i-1}^{crit}, R_i^{crit}]$ provided that the input values assignment to the gates driving the bridged nodes is I. By simulating the multiple stuck-at faults f_1^I through f_n^I, the ADI is obtained as the union of all sections $[R_{i-1}^{crit}, R_i^{crit}]$ for which the corresponding fault f_i^I has been detected while I was applied to the inputs of the gates driving the bridged nodes at the same time.

Figure 1 shows the simulation flow. First, the electrical analysis is performed to obtain the critical resistances for all faults in the fault list [24, 25]. Electrical data such as transistor parameters and logic gate thresholds is required for this. Then, the mapping of sections to multiple stuck-at faults is performed. For every section $[R_{i-1}^{crit}, R_i^{crit}]$, a hash table mapping possible input values of the gates driving the bridge I to the multiple stuck-at fault f_i^I is generated. The hash tables are kept compact by excluding the values of I for which no faulty effect is observed.

Then, 64-bit parallel-pattern multiple-stuck-at fault simulation is performed. The masks to perform fault injection are obtained from the good-value simulation by hash-table look-up. In general, the masks contain different values for different bit positions. This is in contrast to conventional parallel-pattern stuck-at fault simulation where the masks are fixed before good-value simulation and contain identical values for all bit positions.

After the simulation has yielded the detection status of each section, the ADI is calculated and aggregated to C-ADI. If G-ADI is available, it is read in and the exact fault coverage G-FC is calculated. Otherwise, G-ADI is over-approximated as the union of all sections and the approximated fault coverage E-FC (see [34] for data on the accuracy loss due to this approximation). In fault coverage calculation, every value of R_{sh} is weighted by the probability density function $\rho(r)$ of the short resistance r which can be derived from manufacturing data. This R_{sh} distribution is a further input to the tool.

3 Experimental Results

We applied 10,000 random test vectors on the combinational cores of industrial circuits by NXP. Columns 1 through 4 of Table 1 contain the name of the circuit and the number of cells, inputs and outputs of its combinational core. The number of simulated resistive bridging faults is given in column 5. It equals the number of cells in the circuit multiplied by 10. We selected the faults randomly. The faults could also be selected based on layout analysis [7]. The number of faults was chosen to be close to typical numbers of faults obtained by layout-based selection procedures.

The subsequent columns report the obtained RBF cover-

Circuit	Cells	PI	PO	Resistive bridging fault simulation					Stuck-at fault simulation		
				Faults	FC	Preproc. [s]	Sim. [s]	T/BV [ms]	Faults	FC	Time [s]
b01	54	7	7	472	96.80	0.02	0.31	0.00007	122	100.00	0.00
b02	31	5	5	103	92.79	0.01	0.1	0.00010	62	100.00	0.00
b03	183	34	34	1,830	98.57	0.02	0.63	0.00003	394	100.00	0.01
b04	694	77	74	6,940	98.68	0.08	3.56	0.00005	1,540	99.35	0.02
b05	608	35	70	6,080	98.52	0.08	4.42	0.00007	1,554	98.78	0.01
b06	64	11	15	364	96.30	0.01	0.31	0.00009	140	100.00	0.00
b07	476	50	57	4,760	98.38	0.07	3.06	0.00006	1,129	96.90	0.01
b08	192	30	25	1,920	96.19	0.03	2.58	0.00013	417	99.76	0.01
b09	188	29	29	1,880	94.11	0.04	1.33	0.00007	414	100.00	0.00
b10	197	28	23	1,970	97.32	0.04	1.91	0.00010	486	100.00	0.01
b11	579	38	37	5,790	98.21	0.08	3.81	0.00007	1,436	99.30	0.02
b12	1,127	126	127	11,270	98.31	0.15	7.67	0.00007	2,827	93.92	0.04
b13	370	63	63	3,700	99.32	0.05	1	0.00003	801	100.00	0.01
b14_1	4,624	277	299	46,240	97.75	0.71	51.71	0.00011	12,475	92.63	0.39
b14	5,923	277	299	59,230	96.85	0.96	87.98	0.00015	16,167	88.87	0.91
b15_1	8,422	484	518	84,220	77.92	1.31	323.04	0.00038	22,060	60.83	5.27
b15	8,026	485	519	80,260	89.56	1.19	247.96	0.00031	21,282	73.64	6.14
b17_1	25,983	1,449	1,509	259,830	79.58	3.99	1133.84	0.00044	67,861	62.08	20.76
b17	25,719	1,451	1,511	257,190	87.30	4.21	1034.35	0.00040	68,207	71.10	26.01
b18_1	74,881	3,307	3,293	748,810	89.24	12.12	3991.49	0.00053	202,888	75.81	139.75
b18	76,513	3,307	3,293	765,130	89.14	12.26	4084.34	0.00053	206,812	75.80	142.37
b20_1	11,199	522	512	111,990	97.93	1.73	136.28	0.00012	30,813	92.92	1.43
b20	12,991	522	512	129,910	97.67	2.08	175	0.00013	35,731	91.65	2.45
b21_1	10,696	522	512	106,960	98.49	1.63	109.82	0.00010	29,155	94.39	1.26
b21	13,168	522	512	131,680	97.51	2.04	179.11	0.00014	36,058	90.48	2.52
b22_1	16,416	735	725	164,160	98.21	2.35	180.64	0.00011	44,835	93.48	2.33
b22	18,789	735	725	187,890	97.89	2.94	239.38	0.00013	51,341	92.01	3.48
Average					94.76	1.86	444.65	0.00017		90.51	13.16

Table 2: Experimental results for combinational cores of ITC99 circuits, 10,000 test vectors

Circuit	Cells	PI	PO	Resistive bridging fault simulation					Stuck-at fault simulation		
				Faults	FC	Preproc. [s]	Sim. [s]	T/BV [ms]	Faults	FC	Time [s]
cs00027	21	7	4	2	100.00	0.01	0.00	0.00000	32	100.00	0.00
cs00208	131	18	9	1,310	95.70	0.02	0.77	0.00006	217	100.00	0.00
cs00298	156	17	20	1,560	97.65	0.03	0.71	0.00005	308	100.00	0.01
cs00344	210	24	26	2,100	94.01	0.02	1.46	0.00007	342	100.00	0.00
cs00349	211	24	26	2,110	94.13	0.02	1.56	0.00007	350	99.43	0.00
cs00382	209	24	27	2,090	98.60	0.04	0.80	0.00004	399	100.00	0.00
cs00386	185	13	13	1,850	96.77	0.02	0.68	0.00004	384	100.00	0.00
cs00400	213	24	27	2,130	98.30	0.04	1.02	0.00005	424	98.58	0.00
cs00420	269	34	17	2,690	92.46	0.03	2.44	0.00009	455	86.59	0.01
cs00444	232	24	27	2,320	98.07	0.03	1.30	0.00006	474	97.05	0.01
cs00510	249	25	13	2,490	95.04	0.03	2.29	0.00009	564	100.00	0.01
cs00526	245	24	27	2,450	97.46	0.04	1.50	0.00006	553	99.64	0.00
cs00641	476	54	43	4,760	99.43	0.03	0.44	0.00001	467	98.29	0.01
cs00713	489	54	42	4,890	98.41	0.04	1.43	0.00003	581	92.08	0.01
cs00820	336	23	24	3,360	94.48	0.09	4.24	0.00013	850	98.47	0.01
cs00832	334	23	24	3,340	94.70	0.10	4.37	0.00013	870	96.67	0.01
cs00838	545	66	33	5,450	76.68	0.05	10.14	0.00019	931	64.34	0.03
cs00953	492	45	52	4,920	94.36	0.08	6.83	0.00014	1,079	97.68	0.02
cs01196	593	32	32	5,930	97.13	0.08	3.56	0.00006	1,242	97.58	0.02
cs01238	572	32	32	5,720	97.00	0.07	3.83	0.00007	1,355	92.47	0.02
cs01423	827	91	79	8,270	97.07	0.06	3.84	0.00005	1,515	98.75	0.02
cs01488	692	14	25	6,920	97.94	0.08	2.20	0.00003	1,486	99.80	0.02
cs01494	686	14	25	6,860	98.12	0.08	2.42	0.00004	1,506	99.00	0.02
cs05378	3,221	214	228	32,210	99.17	0.24	9.22	0.00003	4,603	98.18	0.07
cs09234	6,094	247	250	60,940	95.56	0.36	39.21	0.00006	6,927	84.55	0.52
cs13207	9,441	700	790	94,410	98.09	0.60	39.68	0.00004	9,815	90.92	0.49
cs15850	11,067	611	684	110,670	98.26	0.65	37.17	0.00003	11,725	91.57	0.66
cs35932	19,876	1,763	2,048	198,760	96.40	1.48	140.19	0.00007	39,094	89.81	2.19
cs38417	25,585	1,664	1,742	255,850	96.00	1.66	168.88	0.00007	31,180	88.75	1.84
cs38584	22,447	1,464	1,730	224,470	95.87	1.99	136.25	0.00006	36,303	94.52	1.31
Average					96.10	0.27	20.95	0.00006		95.16	0.24

Table 3: Experimental results for combinational cores of ISCAS 89 circuits, 10,000 test vectors

Circuit	Cells	PI	PO	Resistive bridging fault simulation					Stuck-at fault simulation		
				Faults	FC	Preproc. [s]	Sim. [s]	T/BV [ms]	Faults	FC	Time [s]
c0017	13	5	2	2	98.59	0.01	0.00	0.00000	22	100.00	0.00
c0095	39	5	7	77	95.90	0.01	0.22	0.00029	110	95.45	0.00
c0432	203	36	7	2,030	98.59	0.05	0.55	0.00003	524	99.24	0.01
c0499	275	41	32	2,750	98.14	0.04	0.28	0.00001	758	98.94	0.01
c0880	469	60	26	4,690	97.78	0.05	2.01	0.00004	942	99.79	0.02
c1355	619	41	32	6,190	97.50	0.06	4.87	0.00008	1,574	99.49	0.02
c1908	938	33	25	9,380	99.35	0.15	2.88	0.00003	1,879	99.52	0.03
c2670	1,566	233	140	15,660	96.65	0.15	7.94	0.00005	2,747	88.31	0.08
c3540	1,741	50	22	17,410	98.81	0.33	6.13	0.00004	3,428	95.83	0.05
c5315	2,608	178	123	26,080	99.65	0.30	4.15	0.00002	5,350	98.90	0.05
c6288	2,480	32	32	24,800	91.65	0.20	42.54	0.00017	7,744	99.56	0.17
c7552	3,827	207	108	38,270	99.04	0.39	13.92	0.00004	7,550	94.29	0.16
Average					97.64	0.15	7.12	0.00007		97.44	0.05

Table 4: Experimental results for ISCAS 85 circuits, 10,000 test vectors

age (E-FC in the terminology of [34]), the time for preprocessing, the simulation time and the time per bridging fault and vector (the last value is given in milliseconds). For comparison, stuck-at fault simulation has been performed for the same 10,000 random vectors using the same simulation engine. The final three columns of Table 1 contain the number of stuck-at faults, the achieved stuck-at fault coverage and the simulation time. The last row of the table details average numbers. The measurement was performed on a 2 GHz AMD Opteron Linux machine with 4 GB RAM.

Tables 2, 3 and 4 report the same data for ITC99, ISCAS 89 and ISCAS 85 circuits, respectively. For some of the smaller circuits, the number of faults which can be treated by the simulator was smaller than the number of cells multiplied by 10.

The RBF coverage achieved by 10,000 random vectors tends to be lower for larger circuits, although there appear to be smaller circuits with many random-pattern resistant faults such as p77k. The coverage is typically higher than the stuck-at fault coverage, but it does not track for all circuits. For instance, the RBF coverage of p100k significantly exceeds its stuck-at coverage while for p78k the opposite is the case.

The largest simulation time of approximately 8 hours is required for the 2,5 million gate design p2927k. The average simulation time of the RBF simulation is approximately 19 times larger than that of the stuck-at fault simulation (this ratio is higher for smaller circuits). Given that the number of faults is approximately five times larger, the average simulation speed is competitive. Note that the actual number of multiple stuck-at faults simulated during RBF simulation is still higher because one RBF consists of multiple sections and every section is mapped to one pattern-dependent multiple stuck-at fault.

Figure 2 (a) shows the simulation time per bridging fault as a function of the circuit size. It can be seen that, with exception of two outliers, the time per fault is almost independent of the circuit size. Figure 2 (b) plots the RBF simulation time against the stuck-at simulation time. The dependency is almost perfectly linear, with exception of the smallest circuits.

4 Conclusions

We demonstrated the applicability of an accurate resistive bridging fault simulator to large industrial circuits with fault lists of realistic size. The simulation effort is not far away from the effort for stuck-at fault simulation, and the gap decreases for larger circuits. Key for this efficiency is the mapping of an RBF to a number of multiple stuck-at faults which allow to employ pattern parallelism and other speed-ups known for stuck-at fault simulation.

5 References

[1] W. Maly. Realistic fault modeling for VLSI testing. In *Design Automation Conf.*, pages 173–180, 1987.

[2] S. Sengupta, S. Kundu, S. Chakravarty, P. Paravathala, R. Galivanche, G. Kosonocky, M. Rodgers, and TM Mak. Defect-based test: A key enabler for successful migration to structural test. *Intel Technology Journal*, 1, 1999.

[3] F.J. Ferguson and T. Larrabee. Test pattern generation for realistic bridge fault in CMOS ICs. In *Int'l Test Conf.*, pages 492–499, 1991.

[4] C. Hawkins, J. Soden, A. Righter, and F. Joel Ferguson. Defect classes - an overdue paradigm for CMOS IC testing. In *Int'l Test Conf.*, pages 413–425, 1994.

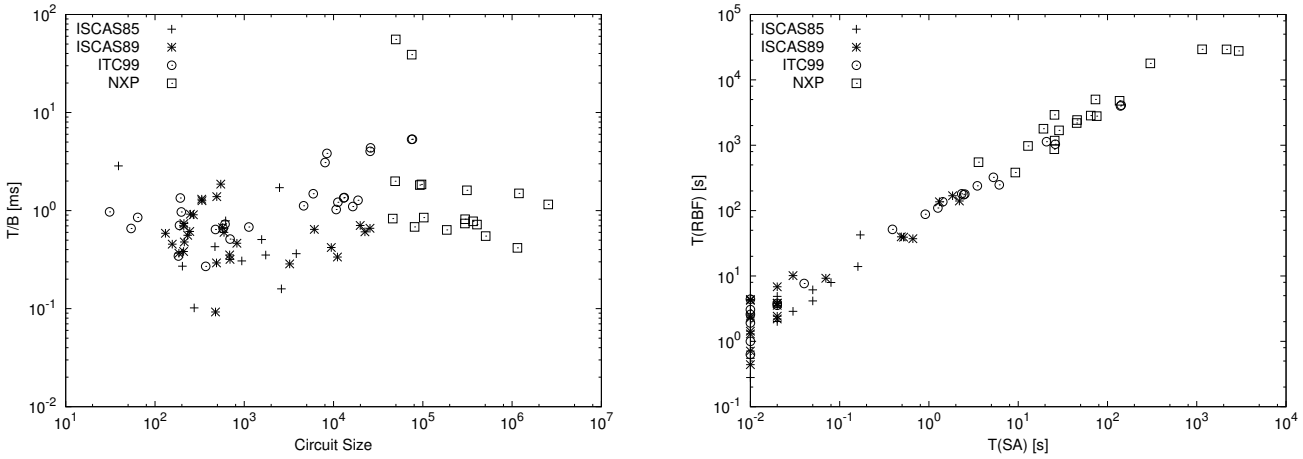
[5] R.C. Aitken. Finding defects with fault models. In *Int'l Test Conf.*, pages 498–505, 1995.

[6] S. Zachariah and S. Chakravarty. A scalable and efficient methodology to extract two node bridges from large industrial circuits. In *Int'l Test Conf.*, pages 750–759, 2000.

[7] F. Joel Ferguson and J. Shen. Extraction and simulation of realistic CMOS faults using inductive fault analysis. In *Int'l Test Conf.*, pages 475–484, 1988.

[8] H. Konuk, F. Joel Ferguson, and T. Larrabie. Accurate and efficient fault simulation of realistic CMOS network breaks. In *Design Automation Conf.*, pages 345–351, 1995.

[9] J. Khare and W. Maly. *From contamination to defects, faults and yield loss*. Kluwer Academic Publisher, 1996.



(a) (b)

Figure 2: Simulation time per resistive bridging fault as function of circuit size (a); simulation time for resistive bridging faults vs. stuck-at faults (b)

[10] Y.-S. Chang, S.K. Gupta, and M.A. Breuer. Test generation for ground bounce in internal logic circuitry. In *VLSI Test Symp.*, pages 95–104, 1999.

[11] A. Krstic, Y.-M. Jiang, and K.-T. Cheng. Pattern generation for delay testing and dynamic timing analysis considering power-supply noise effects. *IEEE Trans. on CAD*, 20(3):416–425, 2001.

[12] M. Nourani, M. Tehranipoor, and N. Ahmed. Pattern generation and estimation for power-supply noise analysis. In *VLSI Test Symp.*, pages 439–444, 2005.

[13] D. Mitra, S. Bhattacharjee, S. Sur-Kolay, B.B. Bhattacharya, S.T. Zachariah, and S. Kundu. Test pattern generation for power supply droop faults. In *VLSI Design Conf.*, 2006.

[14] S.C. Ma, P. Franco, and E.J. McCluskey. An experimental chip to evaluate test techniques experimental results. In *Int'l Test Conf.*, pages 663–672, 1995.

[15] M. R. Grimaila, S. Lee, J. Dvorak, K. M. Butler, B. Stewart, H. Balachandran, B. Houchins, V. Mathur, J. Park, L. C. Wang, and M. Ray Mercer. REDO – Random excitation and deterministic observation - first commercial experiment. In *VLSI Test Symp.*, pages 268–274, 1999.

[16] I. Polian, I. Pomeranz, S. Reddy, and B. Becker. On the use of maximally dominating faults in n-detection test generation. *IEE Proceedings Computers and Digital Techniques*, 151(3):235–244, 5 2004.

[17] R. Rodríguez-Montañés, E.M.J.G. Bruls, and J. Figueras. Bridging defects resistance measurements in a CMOS process. In *Int'l Test Conf.*, pages 892–899, 1992.

[18] K.C.Y. Mei. Bridging and stuck-at faults. *IEEE Trans. on Comp.*, C-23(7):720–727, July 1974.

[19] P. Banerjee and J.A. Abraham. A multivalued algebra for modeling physical failures in mos vlsi circuits. *IEEE Trans. on CAD*, 4(5):312–321, 1985.

[20] S.D. Millman and Sir J.P. Garvey. An accurate bridging fault test pattern generator. In *Int'l Test Conf.*, pages 411–418, 1991.

[21] P.C. Maxwell and R.C. Aitken. Biased voting: A method for simulating CMOS bridging faults in the presence of variable gate logic thresholds. In *Int'l Test Conf.*, pages 63–72, 1993.

[22] G.S. Greenstein and J.H. Patel. E-PROOFS: a CMOS bridging fault simulator. In *Int'l Conf. on CAD*, pages 268–271, 1992.

[23] J. Rearick and J.H. Patel. Fast and Accurate CMOS Bridging Fault Simulation. In *Int'l Test Conf.*, pages 54–62, 1993.

[24] M. Renovell, P. Huc, and Y. Bertrand. CMOS bridge fault modeling. In *VLSI Test Symp.*, pages 392–397, 1994.

[25] M. Renovell, P. Huc, and Y. Bertrand. The concept of resistance interval: A new parametric model for resistive bridging fault. In *VLSI Test Symp.*, pages 184–189, 1995.

[26] H. Hao and E.J. McCluskey. Resistive shorts within CMOS gates. In *Int'l Test Conf.*, pages 292–301, 1991.

[27] M. Favalli, M. Dalpasso, P. Olivo, and B. Ricco. Analysis of dynamic effects of resistive bridging faults in CMOS and BiCMOS digital ICs. In *Int'l Test Conf.*, pages 865–873, 1993.

[28] H. Vierhaus, W. Meyer, and U. Glaser. CMOS bridges and resistive transistor faults: IDDQ versus delay effects. In *Int'l Test Conf.*, pages 83–91, 1993.

[29] Y. Liao and D.M.H. Walker. Fault coverage analysis for physically-based CMOS bridging faults at different power supply voltages. In *Int'l Test Conf.*, pages 767–775, 1996.

[30] C. Lee and D. M. H. Walker. PROBE: A PPSFP simulator for resistive bridging faults. In *VLSI Test Symp.*, pages 105–110, 2000.

[31] V. Sar-Dessai and D.M.H. Walker. Accurate fault modeling and fault simulation of resistive bridges. In *Int. Symp. Defect and Fault Tolerance in VLSI Systems*, pages 102–107, 1998.

[32] V. Sar-Dessai and D.M.H. Walker. Resistive Bridge Fault Modeling, Simulation and Test Generation. In *Int'l Test Conf.*, pages 596–605, 1999.

[33] M. Renovell, F. Azaïs, and Y. Bertrand. Detection of defects using fault model oriented test sequences. *Jour. of Electronic Testing: Theory and Applications*, 14:13–22, 1999.

[34] P. Engelke, I. Polian, M. Renovell, and B. Becker. Simulating resistive bridging and stuck-at faults. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 25(10):2181–2192, Oct. 2006.

[35] Z. Li, X. Lu, W. Qiu, W. Shi, and D.M.H. Walker. A circuit level fault model for resistive bridges. *ACM Trans. on Design Automation of Electronic Systems*, 8(4):546–559, 10 2003.

[36] P. Engelke, I. Polian, M. Renovell, and B. Becker. Automatic test pattern generation for resistive bridging faults. *Jour. of Electronic Testing: Theory and Applications*, 22(1):61–69, 2006.

[37] T. Shinogi, T. Kanbayashi, T. Yoshikawa, S. Tsuruoka, and T. Hayashi. Faulty resistance sectioning technique for resistive bridging fault ATPG systems. In *Asian Test Symp.*, pages 76–81, 2001.

[38] P. Engelke, B. Braitling, I. Polian, M. Renovell, and B. Becker. SUPERB: Simulator utilizing parallel evaluation of resistive bridges. In *Asian Test Symp.*, pages 433–438, 2007.