An Efficient Test Pattern Generator -Mersenne Twister-

Hiroshi Iwata† Sayaka Satonaka‡ Ken'ichi Yamaguchi†

†Department of Information Engineering, ‡Faculty of Advanced Engineering
Nara National College of Technology
22, Yatacho, Yamato-Kooriyama, Nara, Japan. 639-1080
{ iwata, sayaka, yamaguti } @info.nara-k.ac.jp

Abstract— Built-in self test (BIST) is an answer for a high reliable manufacturing test with a reasonable cost. In this paper, we supposed that the Mersenne Twister is used as the test pattern generator instead of the LFSR to implement BIST into VLSIs. Experimental results show that the test patterns generated through the Mersenne Twister are efficient with respect to the fault coverage and it is implemented with a comparable cost to the LFSR.

I. Introduction

It is an imperative phase for not only "design" but also "manufacturing test" to ship a high performance and reliable computer. The manufacturing test phase is required to guarantee no faulty production shipping with an acceptable cost (tester, time to market, and so on.). In manufacturing test, the automated test equipment (ATE) applies test patterns to the circuit under test, and compares the output responses of the circuit under test to the expected values. Usually, the test patterns are automatically generated based on an automatic test pattern generator (ATPG) algorithm. The ATPG algorithm generates a test pattern detecting a modeled logical/delay fault which is well represented some physical defect (short, open, shrink, and so on). On the other hand, the expected values are given by a good machine simulator with the test patterns. Therefore, the ATE is desired to have the high capacity memories (to store the test patterns and the expected values) and the high operation speed (detecting the delay fault, time to market for the product and occupation time of the ATE).

To reduce the cost of the ATE, built-in self test (BIST) techniques[1, 2] are used for the manufacturing test. Figure 1 shows a BIST architecture. The functions of ATE, which are the test controller, the test pattern generator (TPG) and the response analyzer (RA), are implemented in the produced VLSIs. The test controller continuously applies the test patterns generated in the TPG to the circuit under test, and the output responses are stored into the RA. After the preplanned test schedule (a sufficient number of the test patterns are supplied), the comparison result (Go or No Go) or the compressed response (signature) can be observed with a cheap tester (FPGA/LED).

There are two methods to implement TPG and RA, one uses ROM/memory, and the other is that a pseudo random pattern generator and a response compressor are used as the TPG and the RA, respectively. Since the pre-computed test patterns and the expected values are stored in ROM/memory, it is able to achieve a high reliable test with a short test time. However, the area overhead of the ROM/memory is considerable to built-in it into the VLSIs. On the other hand, the linear feedback shift register (LFSR) based pseudo random pattern generator and response compressor are able to be implemented with a low area overhead[1]. The pseudo random patterns generated through the LFSR are applied to the circuit under test, and the output responses are compressed into the response compressor. After the preplanned test clocks, the compressed value (signature) is scanned out to a primary output and compared to the expected value.

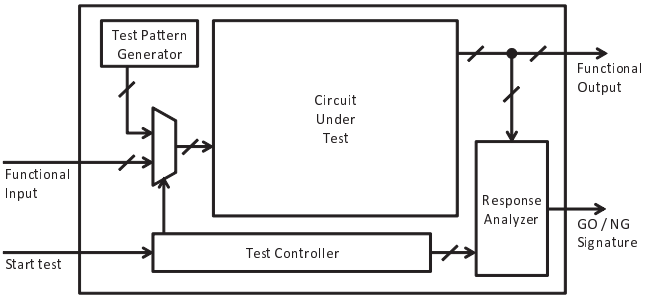

Fig. 1. Built-in self test (BIST) architecture.

From the advantage of a reasonable area, the LFSR is used generally as the TPG and the response compressor. Moreover, the LFSR based pseudo random patterns are able to achieve high reliable test for combinational circuits[3], and the probability of the "aliasing" (erroneous values are missing by the compression) is extremely rare[4]. However, the LFSR based pseudo random patterns are not suitable for sequential circuits and the delay fault detection. To ensure the performance of the circuit, the delay fault degrading the performance should be detected. Therefore, we propose that a new pseudo random pattern generation algorithm, "Mersenne Twister"[5], is implemented to the TPG instead of the LFSR. The advantages of the Mersenne Twister are the following. First, the period of the Mersenne Twister is very long. For example,

[1]The area overhead of n-bit LFSR is estimated with n D-FFs and XOR gates at most.

the period of MT19937 (a kind of the Mersenne Twister) is $2^{19937} - 1 \simeq 4.3 \times 10^{6001}$. Then, the relation between consecutive two patterns is negligible small. Therefore, the pseudo random patterns generated through the Mersenne Twister algorithm has been used for a large scale simulation like a Monte Carlo simulation.

In this paper, we showed the trade-offs between the reliabilities of the manufacturing test and the cost of the implementation. From the perspective of reliabilities, the pseudo random patterns generated through the Mersenne Twister and the LFSR are evaluated by the fault detection abilities with the logical and delay fault model. From the design point of view, design areas required for implementing the Mersenne Twister and the LFSR are evaluated.

The rest of the paper is organized as follows. Evaluation of reliabilities and area overheads are reported in Section II and III, respectively. Section IV concludes the paper.

TABLE I
BENCHMARK CIRCUIT SPECIFICATION.

	#PIs	#POs	#FFs	Area	#Faults
GCD	32	16	48	1,002	2,138
$mult_b$	32	16	39	831	2,390
$mult_s$	32	32	105	1,583	4,544

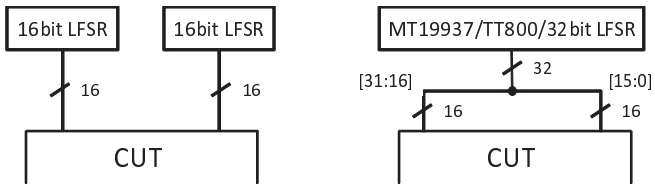

Fig. 2. The structure of the test pattern generator and CUT

II. Evaluation of reliabilities

In this section, the pseudo random test patterns generated through the Mersenne Twister and the LFSR algorithm are compared with the fault coverage to evaluate the reliability of the manufacturing test.

A. Fault coverage

To evaluate the reliability of the manufacturing test, the **fault coverage** is widely used as a metric with respect to the quality of the test pattern. For an effective simulation on the computer, there exist some suitable fault modeling methods representing physical defects. In this paper, we used the single stuck-at fault model and single transition fault model representing the logical fault and the delay fault, respectively. Therefore, the fault coverage is given by the equation 1. In the equation 1, $\#TF$ denotes the total number of all the possible faults assumed in the fault model, and $\#DF$ denotes the number of the faults detected by applying some test pattern set to the circuit. Therefore, the fault coverage means the ratio of the detected faults to the all possible faults, i.e., the high reliability of the manufacturing test is able to be given by a test pattern set achieving the high fault coverage.

$$\frac{\#DF}{\#TF} \times 100[\%] \tag{1}$$

B. Experimental setup

In the following experiments, we used Synopsys Design Compiler to perform logic synthesis, and Synopsys TetraMAX to evaluate the fault coverage on Dell PowerEdge T410 (Intel Xeon X5650(2.67GHz), 4.0GiB). Three benchmark circuits (GCD, $mult_b$ and $mult_s$) are used to evaluate the Mersenne Twister and the LFSR by the fault coverage. Table I shows the specification of these circuits. Columns "#PIs", "#POs", "#FFs", "Area" and "#Faults" mean that the bit number of the primary input, primary output, the number of flip flops, area of the two-input nand gate conversion, and the number of a modeled fault, respectively. The number of faults equals to double the number of the signal lines since the single stuck-at and transition fault models are assumed in the experiment. Rows "GCD", "$mult_b$" and "$mult_s$" mean the circuit giving the greatest common divisor, the Booth's multiplier, and a sequential multiplier. The test pattern set is applied from 32 bit versions of the Mersenne Twister and the LFSR since the bit numbers of the primary input for the benchmark circuits are 32 bit.

C. Experimental results

Fig.2 shows the structure of the test pattern generator and CUT. In this experiment, four types of the pseudo random pattern generators (MT19937, TT800, 32bit LFSR and 16bit LFSR) are used for evaluating the fault coverage. The Mersenne twister has several variations with regarding to the its period. We use the following two types of the Mersenne twister, MT19937 and TT800. Moreover, 32 bit LFSR is able to generate the 32bit pseudo random patterns of which the period is $2^{32} - 1$. However, 16bit LFSR generates only the 16 bit pseudo random patterns of which the period is $2^{16} - 1$. In this experiment, two independent 16bit LFSRs are used as a 32 bit LFSR. These period of MT19937, TT800, 32bit LFSR and 16bit LFSR are $2^{19937} - 1$, $2^{800} - 1$, $2^{32} - 1$ and $2^{16} - 1$, respectively.

Table II shows the fault coverage comparisons of the Mersenne Twister and the LFSR with 500,000 test patterns and 20 different initial seeds. Columns Stuck-at fault and Transition fault mean the fault coverage based on the single stuck-at fault model, and the fault coverage based on the single transition fault model, respectively. Rows "Best", "Worst" and "Average" mean the best fault coverage with a selected initial seed, the worst fault coverage with a selected initial seed, and the average of 20 fault simulation results, respectively.

From the experimental results, there exists distinct differ-

TABLE II
FAULT COVERAGE COMPARISONS OF MERSENNE TWISTER AND LFSR.

		Stuck-at fault			Transition fault		
		GCD	$mult_b$	$mult_s$	GCD	$mult_b$	$mult_s$
Mersenne Twister (MT19937)	Best	100.00[%]	99.33[%]	96.04[%]	99.80[%]	81.62[%]	83.57[%]
	Worst	99.25[%]	95.86[%]	92.25[%]	98.92[%]	75.90[%]	77.81[%]
	Average	99.80[%]	97.30[%]	94.40[%]	99.40[%]	77.60[%]	80.10[%]
Mersenne Twister (TT800)	Best	99.95[%]	98.91[%]	95.16[%]	99.61[%]	79.17[%]	81.69[%]
	Worst	99.02[%]	95.86[%]	89.92[%]	98.92[%]	76.20[%]	77.32[%]
	Average	99.68[%]	97.64[%]	93.44[%]	99.28[%]	77.49[%]	79.46[%]
32bit LFSR	Best	99.91[%]	85.02[%]	70.97[%]	99.26[%]	67.21[%]	44.17[%]
	Worst	98.60[%]	82.59[%]	70.20[%]	86.96[%]	55.81[%]	36.47[%]
	Average	99.59[%]	84.36[%]	70.61[%]	94.79[%]	62.41[%]	37.49[%]
16bit LFSR	Best	99.95[%]	85.15[%]	70.58[%]	99.12[%]	63.67[%]	37.84[%]
	Worst	98.64[%]	83.35[%]	69.83[%]	86.76[%]	61.00[%]	36.58[%]
	Average	99.35[%]	84.10[%]	70.24[%]	94.62[%]	62.24[%]	37.25[%]

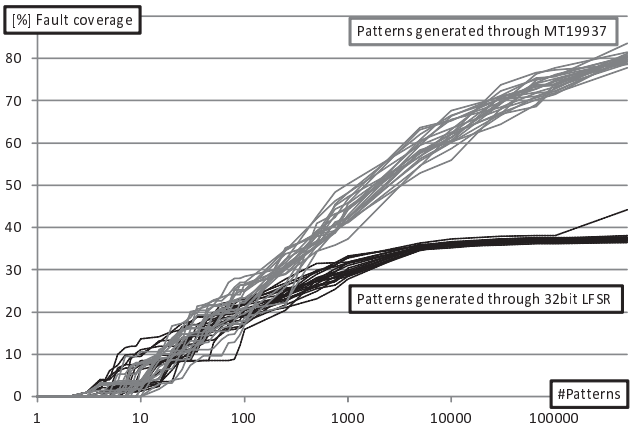

Fig. 3. Transition fault coverage for the sequential multiplier, $mult_s$.

ences of the fault coverage based on the transition fault model for the circuits, $mult_b$ and $mult_s$. The pseudo random patterns generated through the Mersenne Twister achieved the 10 percent to 30 percent high fault coverages comparing to the LFSR's ones. On the other hand, for the circuit GCD, the LFSR based pseudo random test patterns achieved a comparable fault coverage based on both fault models. It should be noted that there exists 10 percent to 20 percent difference of the fault coverage between the Mersenne Twister and the LFSR for $mult_b$ and $mult_s$ even if the stuck-at fault model is assumed.

Figure 3 shows that the fault coverage result for $mult_s$ based on the transition fault model. Horizontal axis of the graph represents the number of test patterns, and this scale is logarithmic. Then, vertical axis of the graph represents the fault coverage. There exists 40 experimental results of the fault simulation. The solid gray lines and dashed black lines mean the fault simulation results applying test patterns generated through MT19937 and 32bit LFSR with 20 differential initial seeds, respectively. Figures 4 to 9 show that the best case (It is the test pattern set giving the highest fault coverage with 500,000 test patterns) of each test pattern generation algorithm. These experimental results show that the test pattern sets generated through the Mersenne Twister are able to achieve higher fault coverage than the LFSR's ones.

Table III shows the fault coverage comparisons of the pseudo random patterns of the Mersenne Twister (Best case, MT19937) and TetraMax ATPG patterns. The ATPG is applied to each circuit and those abort limits are 100 seconds, 100 seconds and 10 seconds per a fault for GCD, $mult_b$ and $mult_s$, respectively. Since the ATPG was not able to run on the 100 seconds abort limit setting for $mult_s$, we use 10 seconds abort limit for the circuit. Rows CPU time[s] and #Pattern mean the CPU time for the ATPG and the number of the generated test patterns from the ATPG. Rows Fault coverage[%] and Best fault coverage[%] mean the fault coverage reported by the ATPG and the fault simulator with the best 500,000 pseudo random pattern, respectively. As the result, the pseudo random patterns generated with the Mersenne Twister achieved higher fault coverages for 5 fault models with no CPU times than the ATPG patterns, and moreover, these higher fault coverages were not able to be achieved with the LFSRs. Therefore, the high reliable manufacturing test is able to be performed by implementing the Mersenne Twister as the test pattern generator.

III. EVALUATION OF AREAS

The area of the test pattern generator (TPG) is discussed in this section. The reliable and practical manufacturing test is achieved if the Mersenne Twister giving the high fault coverage is implemented to the TPG with an acceptable cost. In this paper, the Mersenne Twister, MT19937, TT800, and 32bit LFSR ($\simeq$ two 16bit LFSRs) are used to evaluate these areas.

A. Implementation of Mersenne Twister

A circuit structure of MT19937 is proposed in [6]. Figure 10 shows the structure of MT19937. The circuit is constructed

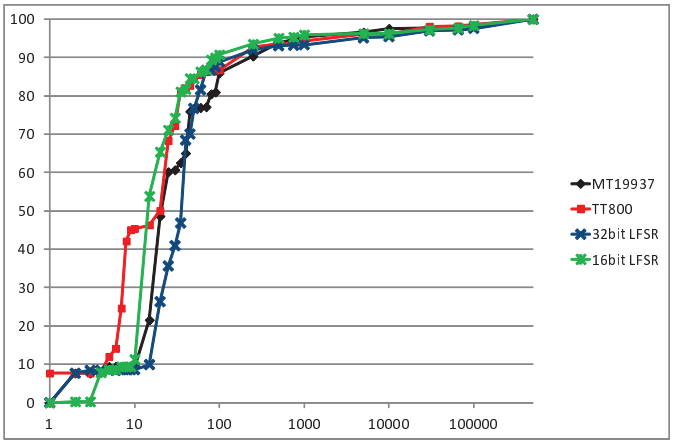

Fig. 4. Fault simulation result for GCD based on the stuck-at fault model

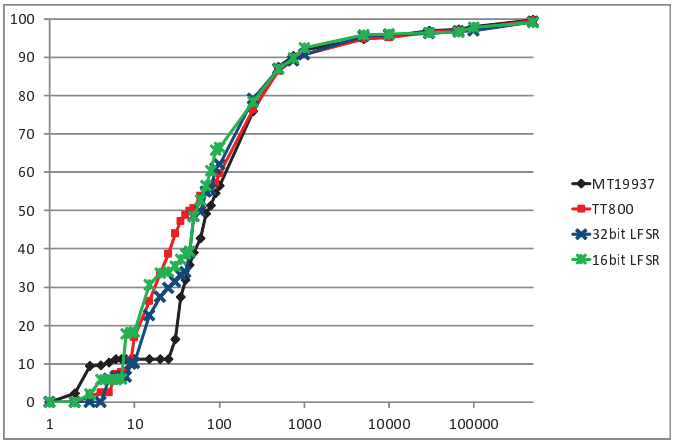

Fig. 5. Fault simulation result for GCD based on the transition fault model

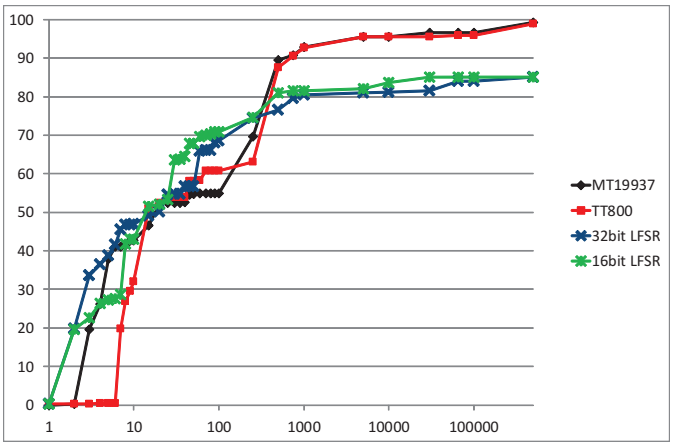

Fig. 6. Fault simulation result for $mult_b$ based on the stuck-at fault model

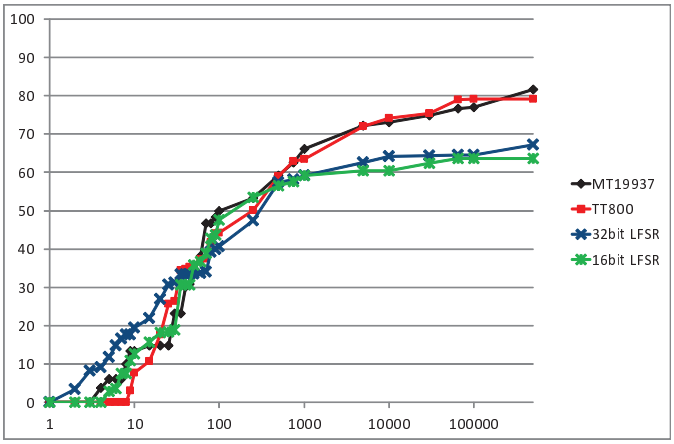

Fig. 7. Fault simulation result for $mult_b$ based on the transition fault model

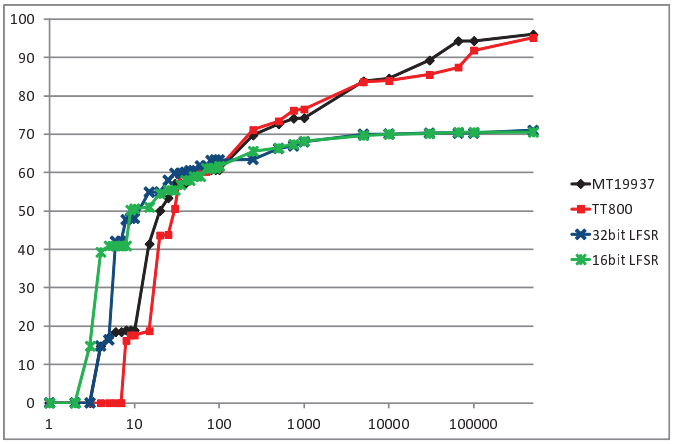

Fig. 8. Fault simulation result for $mult_s$ based on the stuck-at fault model

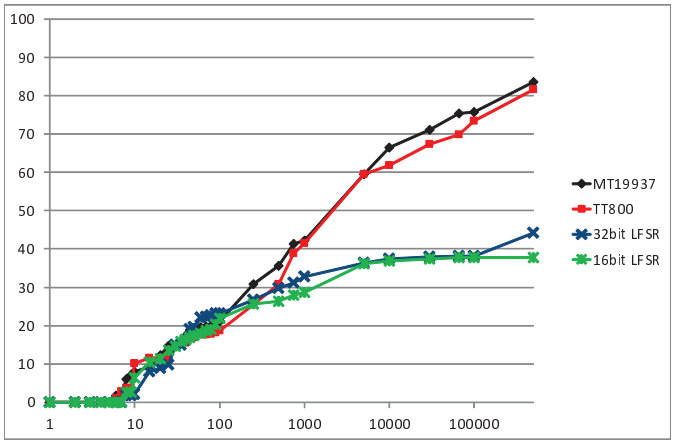

Fig. 9. Fault simulation result for $mult_s$ based on the transition fault model

TABLE III
FAULT COVERAGE COMPARISON OF MERSENNE TWISTER AND ATPG.

		Stuck-at fault			Transition fault		
		GCD	$mult_b$	$mult_s$	GCD	$mult_b$	$mult_s$
ATPG	Abort limit[s]	100	100	10	100	100	10
	CPU time[s]	9,884.03	10,559.18	15,169.00	8,255.19	21,302.45	21,443.61
	#Pattern	276	148	40	468	222	103
	Fault coverage[%]	98.11	98.24	79.61	97.75	96.53	55.49
MT19937	Best fault coverage[%]	100.00	99.33	96.04	99.80	81.62	83.57
32bit LFSR	Best fault coverage[%]	99.91	85.02	70.97	99.26	67.21	44.17

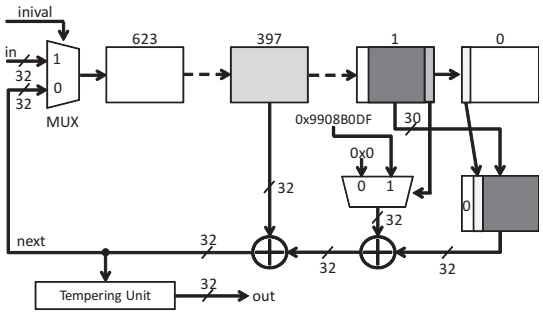

Fig. 10. An example of Hardware structure of MT19937

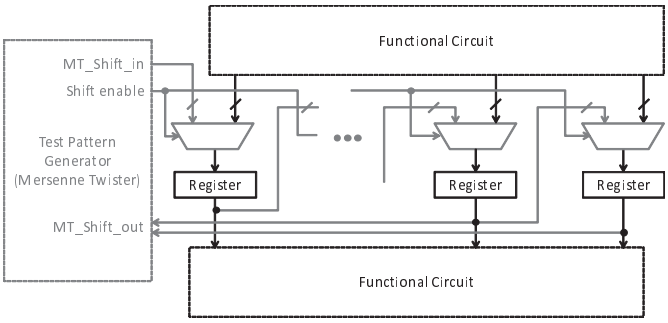

Fig. 11. Sharing register files as Mersenne Twister

TABLE IV
AREA COMPARISONS OF MERSENNE TWISTER AND LFSR.

	MT19937	TT800	LFSR
#FF	19,969	801	32
Combinational area	546	479	51
Sequential area	139,783	5,607	288

TABLE V
AREA OVERHEAD FOR THE PIPELINE PROCESSOR[7]

	Area with TPG	Area overhead [%]
LFSR	877,838	0.02
MT19937	938,080	6.44
TT800	880,809	0.33

from three units, "Register Unit", "Next Generator Unit" and "Tempering Unit". TT800 is implemented by the circuit structure with some parameters shown in [5].

Table IV shows logic optimization results for MT19937, TT800 and 32bit LFSR. The number of the flip flops in MT19937 is larger than the others since the "Register Unit" consits of 624 32bit registers. The area of TT800 is less than the MT19937's one, however, larger than the 32bit LFSR's one.

B. *Sharing register unit*

To minimize the area impact of the "Register Unit", we propose that registers in the "Register Unit" are shared to registers in the functional circuit. There exists the sufficient number of registers in the practical design as represented by a register file. Figure 11 shows the idea of sharing the functional register file as the Mersenne Twister. The registers in the functional circuit are connected in series to implement a shift register since the registers in the "Register Unit" are used as the huge shift register. To implement the shift register, multiplexers are inserted between the functional circuit and registers (see the dashed gray lines in Figure 11).

Table V shows the area comparison results of sharing the register files as the "Register Unit" for the pipeline processor introduced in [7]. The pipeline processor is written in RTL with Verilog HDL and it is optimized with Synopsys Design Compiler. After the logic optimization, the area of the original pipeline processor is 877,627. There exist a sufficient number of registers mainly used as the register file in the pipeline processor. In this experiment, the numerous number of registers are shared to the "Register Unit" of the Mersenne Twisters and the flip-flops of the LFSR. The cost of implementing the TPG into VLSIs is evalu-

ated as the area overhead. The area overhead is calculated in `( TPG area + original area ) / original area`. Experimental result says that implementing MT19937 is useful for the TPG if the cost of 6.44% area overhead is acceptable, and TT800 is able to be implemented to the TPG with a comparable cost to the 32bit LFSR.

IV. Conclusion

The manufacturing test is an imperative phase to ship a high reliable, performance and reasonable VLSIs. In this paper, reliability (fault coverage) and implementing cost (area overhead) of the Mersenne Twister and the LFSR are evaluated to achieve high reliable, performance and reasonable manufacturing test. From the perspective of the fault coverage, the Mersenne Twister based pseudo random patterns have a huge impact to the fault coverage in the delay fault model. On the other hand, MT19937 can be implemented with 6.44% area overhead by sharing the registers in the pipeline processor. The cost of the area overhead will be neglected if the TPG is implemented with built-out self test (BOST) techniques. The BOST techniques can be performed by using some FPGA or some application specific IC as the ATE.

Our future works are that considerations for the experimental results which there exists distinct differences in the fault coverages of $mult_b$, $mult_s$ and GCD on the transition fault model.

Acknowledgments

This work is supported by VLSI Design and Education Center (VDEC), the University of Tokyo in collaboration with Synopsys, Inc.

References

[1] E.C. Archambeau and E.J. McCluskey, “Fault coverage of Pseudoexhaustive Testing,” Digest of Papers 14th Annual International Fault-Tolerant Computing Symposium, pp.141-145, Jun., 1984.

[2] Paul H. Bardell, William H. McAnney and Jacob Savir, “Built-in Test fo VLSI: Pseudorandom Techniques,” John Wiley and Sons, New York, 1987.

[3] Indradeep Ghosh, Nirajk Jha and Sudipta Bhawmik, “A BIST scheme for RTL circuits based on symbolic testability analysis”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol.19, pp.111-128, Jan., 2000.

[4] Miron Abramovici, Melvin A.Breuer and Arthur D.Friedman, “Digital Systems Testing and Testable Design”, Wiley-IEEE Press, Jan., 1993.

[5] Makoto Matsumoto and Takuji Nishimura, “Mersenne Twister: A 623-dimensionally equidistributed uniform pseudo random number generator”, ACM Transactions on Modeling and Computer Simulation, Vol.8, No.1, pp.3-30, Jan., 1998.

[6] Shingo Watanabe and Koki Abe, “A VLSI Design of Mersenne Twister”, IPSJ SIG Technical Reports, Vol.2005, No.41(CSEC-29), pp.13-18, 2005.

[7] David A. Patterson and John L. Hennessy, “Computer Organization and Design, Fourth Edition: The Hardware/Software Interface”, Morgan Kaufmann, Nov., 2008.