Weighted Random Patterns for BIST Generated in Cellular Automata

Ondrej Novak

Technical University Liberec, Hálkova 6, 461 17 Liberec I, Czech Republic
e-mail: ondrej.novak@vslib.cz

Abstract

The paper presents a design method for Built-In Self Test (BIST) that uses a cellular automaton (CA) for test pattern generation. We have extensively studied the quality of generated patterns and we have found several interesting properties of them. The proposed CA can generate weighted random patterns which can be used instead of linear feedback shift register (LFSR) sequences, the fault coverage is higher. There is no need of reseeding the CA in order to generate patterns with different weights. The CA is formed by T flip-flops and does not contain any additional logic in the feedback. We proposed a new scheme of test pattern generation for BIST where the CA serves as a test pattern generator. It is formed by a modified scan chain flip-flops. A number of experiments were done with ISCAS 85 and 89 benchmark circuits. We compared the quality of the generated test patterns with the quality of the patterns generated in an LFSR and with the quality of modified test patterns with several different global test pattern weights. The proposed CA can be advantageously used in BIST.

1. Introduction

Built-in self-test (BIST) is a concept useful for testing the VLSI circuits, where it solves the problem of limited access to the circuit-under-test (CUT), offers on-line and in-line applicability of the test sets and very radically reduces the amount of output information - see e.g. **[8**]. To implement BIST, we must embed both the test pattern generator (TPG) and output data compactor into the structure of the CUT which naturally imposes limits on their size, complexity and level of control which may lead to a loss of fault coverage. Our task was to find BIST structures and their function algorithms, which will guarantee the required level of fault coverage when observing the simplicity requirements.

The techniques for hardware test pattern generation can be classified in the following groups: pseudoexhaustive testing **[9]**, pseudorandom testing **[1],** weighted random testing **[10],** , deterministic tests **[3]** and mixed mode pattern generation **[4], [5].** The mixed mode pattern generation consists in generation of a given number of pseudorandom patterns and after it in exercising the CUT with deterministic test vectors. The deterministic vectors have to detect random resistant faults.

It was shown that for a substantial part of designed circuits it is practically impossible to detect all faults by a pseudorandom test set. One way how to improve the fault coverage and or to reduce the number of generated test patterns is to generate weighted random patterns (WRP). There exist a relatively large number of proposals **[10]** how an optimized set of weights can be determined such that the required number of random patterns is minimized. Usually the input-oriented weight computation is used, the resulting test set has different probabilities of zeros and ones for each CUT input. This approach is very efficient but in the case of hardware test pattern generation it demands quite a lot of additional hardware .

In **[7]** it was shown that it could be very efficient to calculate and apply different global weights for random testing. It provides lower hardware overhead and shorter computational time than the previous method but it is more hardware consuming than using the pseudorandom patterns. The method uses single LFSR for pattern generation with the probability of ones equal to 0.5, weight computational block for deriving patterns with modified weights and a multiplexer which switches between patterns with different weights. The multiplexer is controlled by a counter which enables to generate in one test set patterns with different weights. The output of the multiplexer feeds the scan chain of a CUT.

In this paper we introduce a new scheme of weighted random pattern generation. Instead of the external LFSR we modify the scan chain in such a way that it forms a CA. The CA is used for generating code words of chosen code with greater minimal code distance of its dual code and for generation of non code words. The CA can generate whole code sequences after seeding with one seed only. We have verified that the quality of generated test patterns is better than the quality of the patterns generated in the LFSR with primitive polynomial, the hardware overhead is low.

2. Design of a CA

Let us suppose, that we have a binary linear code (n,k). Let us suppose that a code word bits are contained in a shift register with local feedback loops as it is given in Fig.1. From the theory of linear codes follows that if we add mod 2 two different code words we obtain another code word. As the codes are cyclic, another code word could be obtained by shifting a code word to the right or to the left. The scheme given in Fig. 1 performs simultaneously adding mod 2 of two different code words (One code word correspond to the present state of the shift register, the second correspond to the code word shifted to the right, these two code words are added with the help of the XORs). If the characteristic polynomial of the (n,k) code is non primitive we obtain code word length shorter than 2^{n-k}-1. If the characteristic polynomial is irreducible there must exist a primitive element of the corresponding field. If the polynomial x+1 is a primitive element of the corresponding field the automaton from Fig.1 generates after seeding with one code word all the code words of the (n,k) code.

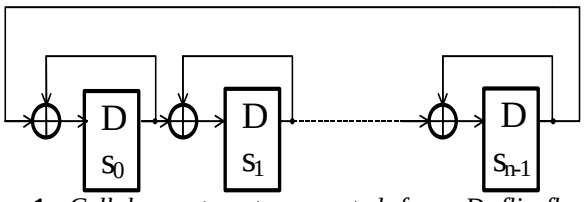


Fig. 1. *Cellular automaton created from D flip-flops performing multiplication of the polynomials corresponding to code words by the polynomial x+1.*

The automaton is a cyclic additive cellular automaton (CA) with the rule 60 for each cell **[2]**, having a regular linear structure.

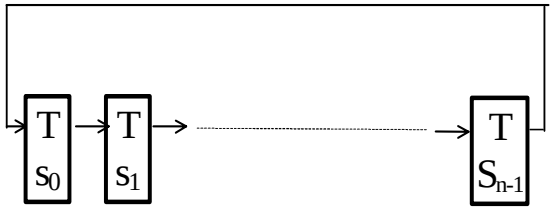
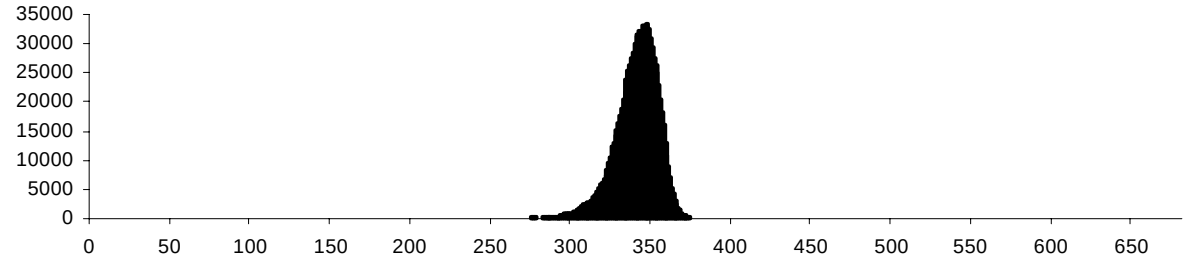


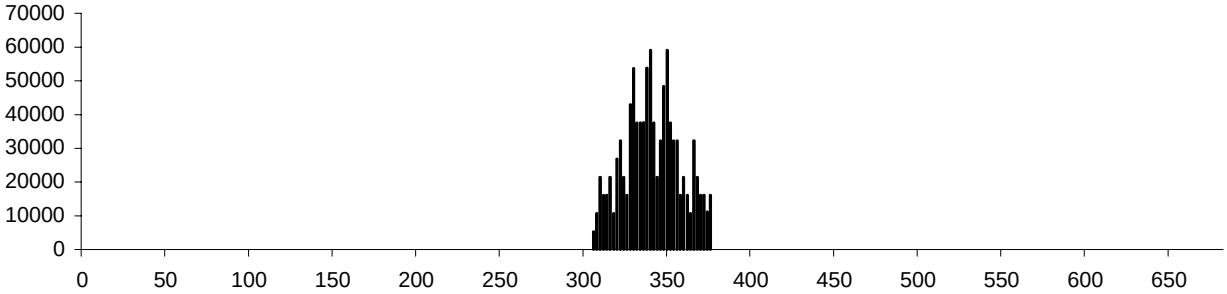
Fig. 2. *Cellular automaton created from T flip-flops performing multiplication of the polynomials corresponding to code words by the polynomial x+1.*

This CA can be further simplified. Instead of the D flip-flops and local feedback taps with XORs we can use T flip-flops because a T flip-flop performs in each clock period a XOR function of its own state and the input signal, the result is stored as a new internal state **[6]** . Thus the CA given in Fig.2 has the same function as the CA in Fig. 1, the hardware realization is substantially simpler.
As far as we know, the proposed CA is the simplest possible automaton which can generate all code words of codes with non primitive polynomials with one seed only. It is universal in the sense that it can generate all possible code sequences of the given length without any hardware modification, we can also generate non code patterns which correspond to XOR of two or more code words of different codes, the period is equal to the case of seeding the CA with a code word. It is also possible to find seeds for which the CA has non periodic behavior.

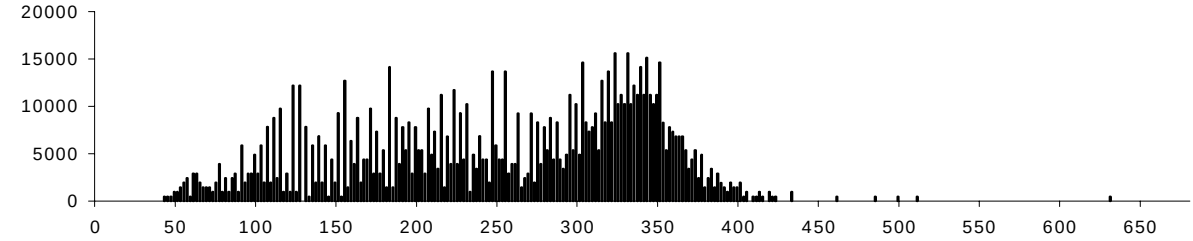
We have studied the properties of the patterns generated in CA. There are substantial differences between the properties of the patterns generated in the LFSR and CA. The weights of the patterns obtained from a LFSR with a primitive characteristic polynomial are very closed to the value n/2, where n is equal to the scan chain length. An example is given in the Graph 1. We have compared the weight rates of the LFSR patterns with the weight rates of CA patterns . The seed in the CA was chosen in such a way that the number of corresponding code information bits was the same as in the case of the LFSR. In the Graph 2 we can see a rate of pattern weights of code words generated in the CA. The polynomial of the code was non primitive and irreducible. In the Graphs 3 and 4 we can see the rates of pattern weights which we obtained after seeding the CA with different non code seeds. We have found that for different seeds we can generate patterns with different weight rates.
An interesting question is: What is the probability of finding a seed which causes generating patterns with higher or lower weights than n/2? We have studied this problem on a simple example of 17 bit CA. We have simulated all different sequences with different seeds. The resulting numbers of sequences with the given weights are shown in the Graph 5.
If we want to use the CA in CUT testing we have to do several practical steps:
1) Choose an (n,k) code, $n \geq$ number of CUT inputs, $2^{n-k} - 1 \geq$ minimal test length. In order to keep the period of CA maximal it is necessary to choose n equal to the length of some code with irreducible polynomial. For our experiments we have chosen the codes with lengths 17, 23, 25, 41, 47, 55, 69, 267, 683, 765, 1687, ...
2) Verify whether the polynomial x+1 is a primitive element of the field. Otherwise we have to chose another characteristic polynomial of the (n,k) code .
3) Verify whether the selected code or non code sequence is suitable for testing the CUT. We can either to compute the optimal weight distribution **[7]** or to simulate the fault coverage for randomly chosen test sequences and to select the best one.
We can use the CA as a TPG in several ways. One possibility is shown in Fig. 3.



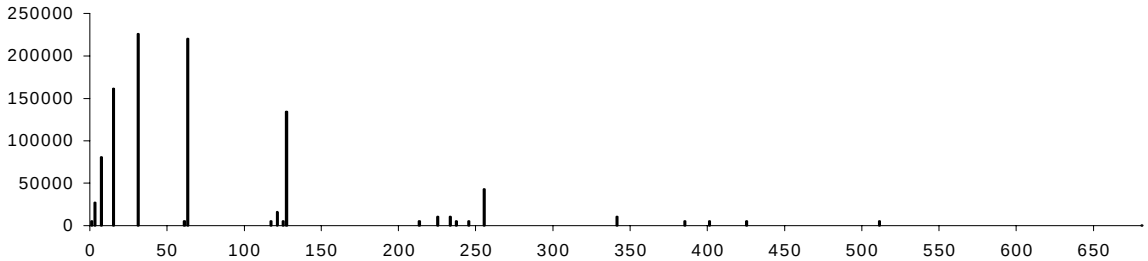
Graph 1: *Weight rates of 23 bit LFSR with a primitive characteristic polynomial. The LFSR output was serially fed to the scan chain of the length 683 bits.*



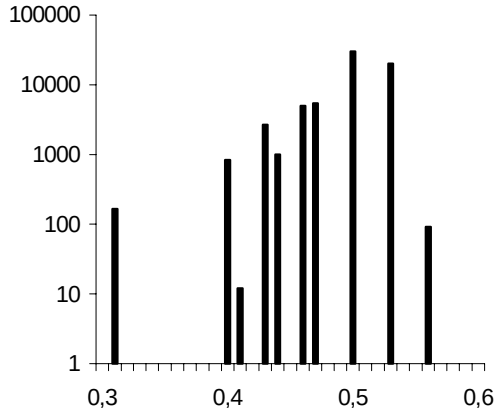
Graph 2: *Weight rates of the 683 bit CA which generates code patterns.*



Graph 3. *Weight rates of the 683 bit CA which generated non code patterns.*



Graph 4. *Weight rates of the 683 bit CA which generated non code patterns. The seed was different from that one in the Graph 3.*



Graph 5: *17 bit CA. Number of CA sequences (Y axis) with given weight (X axis).*

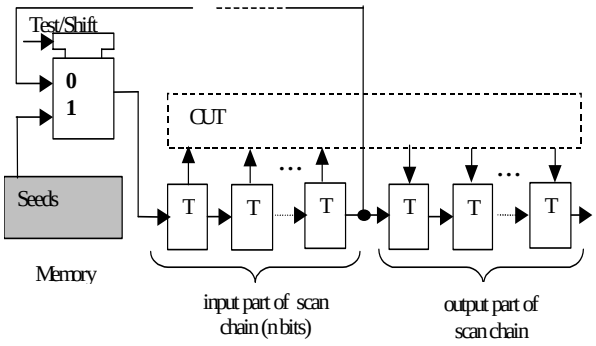


Fig. 3.

Let us suppose that the CUT is designed in accordance with the Scan Design methodology with modified Boundary Scan cells in such a way that D flip-flops in the scan chain are replaced by T flip-flops. The testing starts after reset and seeding the CA from the memory. At each clock cycle the CA generates a new pattern, and the CUT responds are compacted in the output part of the chain. After performing a given number of test steps the deterministic test patterns are shifted into the chain and the CUT responses are compacted in the output part of the chain. We have to keep in mind that the seeds which are stored in the memory have to be modified for both parts of testing in such a way that after shifting into the scan chain the desired test cube would be present at the CUT inputs.

3. Hardware overhead

We have designed the CA in MIETEC 2.4 μm CMOS technology. In this technology we have the following sizes of the flip-flops:

D flip-flop with set or reset – 180μm x 106 μm
T flip-flop with set – 170μm x 106 μm
T flip-flop with reset – 190μm x 106 μm
XOR gate – 80μm x 106 μm.

If we compare the solution of T flip-flop CA with asynchronous set from Fig. 2 with the simple chain formed by D flip-flops with set or reset we can see that the used chip area will be similar. If we compare the solution of T flip-flop CA with the D flip flop CA we can see that we use only about 65 % of silicon for the flip-flops and gates, we also spare some area because of simpler routing.

4. Results Obtained

We have chosen ISCAS 85 and ISCAS 89 benchmark circuits with a significant number of random pattern resistant faults. The internal flip-flops were considered to be CUT inputs. We checked the number of non detected faults both for a 32 bit LFSR with a primitive polynomial and for the CA. The polynomial of the 32 bit LFSR was randomly chosen from the list of primitive polynomials. The seed of the CA was chosen in such a way that the weight rates of randomly chosen code and non code sequences were estimated. After getting pattern sequences with different weight rates the fault coverage of the "promising" test sequences was verified. For every circuit we have done 8 experiments with different LFSR sequences and 8 experiments with CA sequences. The best results are given in Tab. 1.

circuit	cube size	32 bit LFSR	WRP [7]	CA
s 641	54	12	1	7
s 713	54	11	0	7
s 820	23	9	4	6
s 832	23	17	4	5
s 953	45	10	1	3
s 1238	32	10	13	11
s 5378	214	38	28	27
s1196	32	17	18	11
s 13207	700	624	196	472
s 9234	247	648	193	602
c 2670	157	304	6	292
c 7552	206	324	93	130

Table 1. *Comparison of the numbers of faults which remain undetected after 10 000 test patterns generated in 32-bit LFSR, by WRP method and by a CA .*

In the table we compare the numbers of undetected faults with WRP fault coverage **[7]**.The experiments done with the ISCAS benchmark circuits showed that in some cases we obtain better fault coverage for CA then for LFSR, in some cases we obtain similar results. Because of the lower hardware overhead the CA solution is advantageous. If we compare the proposed solution with the weighted random patterns with global weights **[7]** we can see that the CA gives worse results. The hardware overhead which is necessary for generating test patterns with a priori given global weights is much higher than for the CA solution and thus the CA solution could be used in the cases in which the size of hardware overhead is crucial.

4. Conclusion

A linear cellular automaton which can be used as a TPG for BIST was designed. Its main features are its simple structure and high fault coverage. The simplicity of the structure is due to the existence of only one feedback with no XOR. The CA can be implemented in the CUT by replacing the D flip flops in the scan chain with the T flip/flops and by adding a feedback from the last to the first flip-flop. The properties of the generated patterns depend on the seed of the CA. No additional hardware changes have to be done when we want to change the rate of weights in generated patterns.

We compared the properties of the CA with the properties of a 32 bit LFSR which is usually used for pseudorandom test pattern generation.
The CA has the following advantages:

- no hardware overhead, the whole TPG can be built by converting the existing scan chain
- depending on the seed the CA can generate weighted pseudorandom test sets with different rates of weights
- the fault coverage for ISCAS benchmark circuits is better than it is for an external LFSR

Disadvantages:

- all flip-flops have to be set or reset-able, a seed for the CA has to be calculated in more complex manner than for the LFSR
- the patterns are generated in the scan chain and thus the CUT responses must not influence the flip/flops in the input part of the chain during the test.

The TPG can be advantageously used in mixed mode testing where we generate a given number of patterns and after it we exercise the circuit with deterministic test patterns which are stored in a memory.

Acknowledgment

The author wishes to thank H.-J. Wunderlich and S. Hellebrand for their help with the estimation of fault coverage of the benchmark circuits. The research was in part supported by the research grants of the Czech Grant Agency GACR 102/98/1003 and of the Ministry of Education, Youth and Sport VS 96006.

References:

[1] Bardell, P. – McAnney, W. H.- SAVIR, J.: Built-In Test for VLSI. New York: Wiley-Interscience, 1987

[2] CHAUDHURI, P.P. et al.: Additive Cellular Automata Theory and Applications Volume I. IEEE Computer Society Press, 1997, 340 pp.

[3] DAEHN, W. – MUCHA, J.: Hardware test pattern generators for built-in test. Proc. of IEEE ITC,1981, pp. 110-113

[4] KOENEMANN, B.: LFSR – coded test patterns for scan designs. Proc. Europ. Test Conf., Munich , Germany, 1991, pp. 237-242

[5] HELLEBRAND, S. - RAJSKI, J.- TARNICK, S.- VENKATARAMAN, S. - COURTOIS, B.: Built-In Test for Circuits with Scan Based on Reseeding of Multiple-Polynomial Linear Feedback Shift Registers. IEEE Trans. on Comp.,vol. 44, No. 2, February 1995, pp. 223-233

[6] Hlawiczka, A.: D or T Flip-Flop Based Linear Registers. Archives of Control Sciences, vol. 4., XXXVII, 1992, pp. 578-587

[7] KUNZMANN, A. : Efficient Random Testing with Global Weights. Proc. of IEEE EURO-DAC `96

[8] McCLUSKEY, E.J.: Built-in self-test techniques. IEEE Design & Test of Comput., Vol. 2., April 1985, pp. 21-28

[9] WANG, L.T. - McCLUSKEY, E.J.: Condensed linear feedback shift register (LFSR) testing - A pseudoexhaustive test technique. IEEE Trans. on Comp. Vol. C-35, Apr. 1986, pp. 367-370

[10] WUNDERLICH, H. J. : Self Test Using Unequiprobable Random Patterns. Proc. IEEE 17 FTCS, Pittsburgh 1987, pp.236-244