www.ijiarec.com

ISSN:2348-2079

Volume-7 Issue-1

International Journal of Intellectual Advancements and Research in Engineering Computations

Improving fault coverage and reduce the number of test data in test cube by using dynamic LFSR reseeding

D. Sathishkumar[1], P.Thilagavathi[2]

[1]PG Student, Department of ECE

[2] Associate Professor, Department of ECE

KSR College of Engineering (Autonomous), Thiruchengode, 637215, TamilNadu, India.

ABSTRACT

In test data compression the methods that are based on linear-feedback shift register (LFSR), a seed that generates a test for a target fault is computed based on a test cube for the fault. With a given LFSR, a seed may not exist for a given test cube, even though a seed may exist for a different test cube (non-test cube) that detects the same fault. Therefore a seed required to compute a test cube is maximum. This issue is addressed by using dynamic LFSR reseeding to reduce the number of seeds needed to compute test cube. The metric described in this brief (output deviation metric) used to achieve high fault coverage rather than classical and static LFSR reseeding. We verify the results on ISCAS'89 circuits, and a subset of the IWLS'05 circuits Experimental results demonstrate the advantages of dynamic LFSR reseeding for fault coverage improvement.

Keywords: LFSR (Linear Feedback Shift Register), dynamic reseeding, linear de-compressors, test cubes, test data compression.

INTRODUCTION

Generally test data compression methods are basically depends on linear de-compressors. A favorite compression approach is that of statically reseeding a Linear Feedback Shift Register (LFSR). The main concept of static LFSR reseeding is to capitalize the minimum volume of prescribed bits in the test cubes (i.e., test patterns with 0's and 1's values) in order to reduce the volume of test cubes into LFSR seeds. A seed is generated by solving a system of linear equations, where the original state of each LFSR cell is measured to be a binary variable. The main disadvantage of this approach is that it offers limited compression. Many LFSR reseeding methods have been implemented which offer better compression. Even though all the methods are very efficient in compressing test data, they do not cover un-modeled defects that are not explicitly covered by the test data that are being compressed).

The first approach is interposed to merge increased un-modeled defect coverage and LFSR reseeding was proposed in [15]. In this method the variables that remain free after encoding the test cubes into LFSR seeds are properly filled in order to accelerate the output deviations of the test vector generated by each seed. Output deviations suggest an efficient probabilistic means to estimate test vectors based on their potential for detecting arbitrary defects [16]. The method introduced in [15] affects from many serious drawbacks. At first it achieves only a limited improvement in defect coverage compared to classical reseeding [4]. The reason is twofold: first, output deviations are not effectively capitalized in [15]

Due to the inefficacy of the metric used. Second, [15] capitalize only the free variables of each seed, which offers only a moderate improvement in the output deviations of the resulting test vectors.

Author for correspondence:
Department of ECE, KSR College of Engineering (Autonomous), Thiruchengode, 637215, TamilNadu, India.

In addition, as a classical reseeding method, [15] affects from limited compression. Finally, the method of [15] cannot be introduced to the highly effective (with respect to compression) dynamic LFSR reseeding methods, which generally consumes most of the free variables. In this paper, we introduce a new dynamic LFSR reseeding method that offers high compression and improve un-modeled defect coverage. Maximum un-modeled defect coverage is achieved: (1) by using a new output- deviation-based metric that is more efficient than that of [15] for detecting faults, and (2) by safely encoding the test cubes into seeds using test data compression as well as defect-oriented criteria. Using this method, the introduced encoding method capitalize all seed variables individually for achieving high compression; thus the defect-detection potential of the generated seeds is enriched without compromising compression. Simulation results explains that the proposed defect-aware dynamic reseeding method hold the high compression of dynamic reseeding and at the same time offers improved defect coverage than both the original defect-unaware dynamic reseeding method and [15]. In addition, fast coverage ramp-up is achieved; decrease that way the test-application time in an abort-at-first-fail environment.

EXISTING METHODS

Computation of non-test cube

When test data compression is depends on the use of a linear-feedback shift register (LFSR), test cubes are used to generating seeds for the LFSR [1-10]. Given a test cube ci for a respective fault fi, a seed si for the LFSR is acquired by solving a set of linear equations that relates seed si with the specified values (o’s and 1’s) of test cube ci [1]. When seed si is filled into the LFSR, and the LFSR is clocked for the suitable number of clock cycles, the scan chains of the circuit are loaded with a test ti. The test ti has all the specified values of test cube ci. Therefore, test ti is promised to detect fault fi. When an LFSR is combined with a given set of test cubes, a seed may not exist for one or more of the test cubes ci [2-9]. Nevertheless, even if a seed does not exist for a test cube ci that find out a fault fi, it is feasible that a seed exists for a different test cube ci for fault fi.

To negotiate this problem it is possible to generate many test cubes to replace which seeds do not exist. Alternatively, a procedure is implemented earlier uses a test cube ci for a target fault fi only as counsel for the generation of a seed si. The procedure allows the seed si to generate a test ti that disunity with test cube ci as long as test ti detects fault fi. However, this procedure still depends on the use of specific test cubes (0’s and1’s). Hence, even with a partial match, it may not be able to determine a seed si for a fault fi depends on a test cube ci. The procedure explained in [11] combines with the circuit an XOR network that models the constraints of the test data decompression logic. By achieving test generation for the extended circuit, the procedure from [11] determines seeds for an LFSR straightway, without first generating test cubes.

The objective of this brief is to expose that it is possible to generate seeds for LFSR-based test generation without using test cubes and without expanding the circuit. This mitigates the constraints that the use of test cube is able to find out target faults without the need to function test generation for a more hard circuit. A non-test cube ui for a fault fi prevents fault fi from being detected. In order to find the fault, it is need to prevent non-test cube ui from appearing in a test set. This can be applied to every test and test cube for detecting fault. Hence, the primary use of non-test cubes for generating seeds does not degrade the ability of the procedure to find seeds when they exist for a suitable LFSR.

The procedure for generating seeds based on non-test cubes uses a low-complexity procedure that is depends on logic simulation of the LFSR to generate the test TI that a given seed SI produces. Fault simulation of the fault fi under test set ti is used for determining whether test ti finds fault fi. To generate a seed si for a given suitable fault fi, the procedure uses a set of non-test cubes Ui for fault fi. It starts from a random assignment to seed si. It changes seed si to avoid the presence of non-test cubes from Ui in test ti. The alternation of seed si is expected to solve the detection of fault fi when a seed for fi exists. The merits of this

Copyrights © International Journal of Intellectual Advancements and Research in Engineering Computations, www.ijiarec.com

procedure is that it is not restricted by a given test cube.

Hence, a seed for a given LFSR may be found even if one cannot be found based on a test cube. Its demerit is it takes more time to search seed, since it is advised only by values that need to be avoided. To address this problem, it is possible to use non-test cubes only for faults that cannot be found based on test cubes. Since only hard-to-detect faults are covered. In addition, single stuck-at fault and bridging fault has been covered by dynamic LFSR reseeding. If Single stuck-at faults are used as target faults. In a single stuck-at fault where line gi is stuck at the value ai is denoted by fi = gi/ai. The procedure can be extended to other fault models**.**

Static LFSR reseeding

In static reseeding every new seed flushes the de-compressor and thus any variables left unspecified (free) during the seed-computation process are wasted. In the case of classical reseeding [4], many variables are wasted because each seed is used for encoding a single test cube. Since the LFSR size is determined by the cube with the maximum number of specified bits, and usually, many cubes have fewer defined bits than, a lot of variables remain free when the cubes are encoded.

PROPOSED METHOD

Dynamic LFSR Reseeding

In this paper, we propose efficient dynamic LFSR reseeding method that increases the effectiveness of the seeds with respect to the detection of both timing-dependent and timing-independent faults at the same time. This is a new facet compared to classical reseeding and statistical reseeding.

DEVIATION BASED METRIC

In this section we introduce output deviation-based metric for estimating a dynamic seeds. Each seed is used to compute one test vector, which is applied using two capture cycles. The Maximum Expected Deviation, $MED(j,\ r_k,v)$, is an evaluation of the maximum deviation value expected around the seed generation process at output$j$, when its fault-free response is $v(v=0.1)$at capture cycle $r_k(k=1,\ 2)$. It is determined as follows: initially, for every test cube, a predestined number of temporal seeds are computed by randomly displacing free variables. For each output j, the computed test vectors are divided into four groups: those generating fault-free responses 0, 1 at capture cycle's $r1,r2$. The output-deviation values of all computed test vectors are determined and the highest value for every output j and for each fault-free response v at capture cycle r_k composes $MED(j,r_k,v)$.

Let $D(s,\ j,r_k,v)$ be the deviation value at output j for the test vector computed by candidate seed s, which generates fault-free response v at that output at capture cycler_k. The value $D(s,\ j,r_k,v)$ is esteemed to be near-maximum if

$$D(s,i,r_k,v) \geq F_1 \cdot MED(i,r_k,v).$$

F1 is a real-valued parameter close to 1 for choosing seeds with output deviation values that are very near to the *MED* values. We experimentally validated that a value of F1 in the range [0.99, 0.995] gives high-quality seeds in all cases.

The second assignment is to rank all circuit outputs according to their potential of conforming errors. Every output j is assigned four weights $w0(j,r_k,v)$ for k=1, 2 and v=0, 1, which are initially set equal to the number of lines in the logic cone of the related output. These weights are denotative of the volume of undetected defects that can be solved for both fault-free responses 0 and 1 at output j during both capture cycles. Let sets $MS(s,r_k,v)$ consist of all circuit outputs j, for which the deviation value $D(s,\ j,\ r_k,\ v)$ satisfy inequality (1). Then, the weight of each candidate seed is calculated using the formula

$$WS(s) = \sum_{k=1,2} \sum_{v=0,1} \sum_{i \in MS[s,r_k,v]} wo(i,r_k,v).$$

The above mentioned formula explains that, for either fault-free response 0 or 1, only the weights of the outputs that get close maximum deviation

Copyrights © International Journal of Intellectual Advancements and Research in Engineering Computations, www.ijiarec.com

values for capture cycles, [i.e., those belonging to $MS(s,r_k,0),MS(s,r_k,1)$] involve into the final weights sum. Note that the first response $(k=1)$ covers the timing-independent faults, while the second response $(k=2)$ covers timing-dependent faults. The seed with the largest value is picked up as the one with the best potential to solve both timing-independent and timing-dependent un-modeled faults.

The weight enables the collection of selective seeds that compute vectors with the maximum deviation values at the outputs of large cones of the CUT. However, maximizing the deviations only at a portion of outputs may result in low defect coverage, even when this portion contains outputs of the largest logic cones. To this end, for every selected seed, every output which satisfies (1) is measured, and the corresponding weight is divided by a constant factor. This is explained by the fact that if seed gives a maximum deviation at output for fault-free response at capture cycle, then it is likely that many faults at the fan-in cone of will be solvable at output when it is applied. In that way, the reduced weight at the output has much smallerimpact on the selection of the next seeds. We verified that a value of in the range [2, 10] is enough to maximize the deviations at all outputs.

Finally, we note that responses often contain X-values which reduce the ability of test vectors to detect faults. Such outputs are not taken for generating weight for seed. Many new methods have been proposed for handling the X's so as not to corrupt the compacted responses, like [10]. Such methods can be merged with the proposed method to provide a unified solution. However, during output response compaction, some non-X response values can be eliminated, which degrade the un-modeled fault coverage of both the proposed and the original dynamic reseeding method

GENERATION OF DEFECT AWARE SEEDS

In the proposed method, we concentrate two objectives: the effective compression of test cubes, and the maximum fault coverage of the resulting test vectors. Maximum fault coverage is achieved by computing candidate seeds which encode collection of different test cubes. High compression is achieved by carefully computing the candidate seeds using a new encoding criterion, which guarantees that all candidate seeds give nearly the same high level of compression. T candidate seeds (T is a user-defined parameter) are computed and valued using the output deviation-based metric presented in Section III. The best candidate seed based on this metric is choosing each time. The computation of the T candidate seeds is done as follows: we start by encoding in each of the T candidate seeds the T most specified test cubes which have not yet been encoded. Then, for each candidate seed which still has free variables, we continue by encoding additional test cubes.

These test cubes are the most specified ones that also require the displacement of the fewest variables. code any more test cubes. Then, the T computed candidate seeds are valued using (2), the one with the maximum weight is selected, and the test cubes encoded by the selected seed are dropped. This process is continuously applied for selecting all seeds. This encoding method covers two goals simultaneously: first, as mentioned above, it tries to increase the number of highly specified test cubes encoded by every seed using the minimum number of variables, which tends to reduce the overall seed volume [3]. Second, it attempts different encodings early on in the encoding process (i.e., during the selection of the first cube for every seed) and deals that the T test vectors computed by the T candidate seeds will be sufficiently different (and Therefore they will potentially provide sufficiently different fault coverage). In that way, it maximizes the likelihood of computing high-quality candidate seeds as early as possible, which can potentially offer steep coverage ramp-up.

This encoding process is completely different than the existing encoding process of [15]. Especially, in [15] all candidate seeds for all test cubes are computed before the selection process begins (this is possible as each seed encodes a single test cube). Thus, the selection is very efficient as it considers all candidate seeds at the same time. Moreover, the selected candidate seeds can be applied in any order to target high fault

Copyrights © International Journal of Intellectual Advancements and Research in Engineering Computations,
www.ijiarec.com

coverage ramp-up. However, in dynamic LFSR reseeding, the encoding of test cubes into seeds is more complicated because every seed depends on all the previous seeds (the LFSR is never flushed). Thus, all candidate seeds cannot be present at the starting of the selection process (every seed can be generated only after the generation of the previous seeds) and also seed ordering is not an option. The proposed Dynamic reseeding overcomes these limitations as follows: 1) it computes candidate seeds which offer both diverse encoding and high variable exploitation (by encoding early on in the encoding process different, highly specified test cubes) and 2) it uses the *MED* values to value the candidates, which offer a good estimate of the fault potential of all seeds expected to be computed throughout the seed computation process. Thus, at each step, it guarantees that the most effective seeds with respect to the compression efficiency, fault coverage and coverage ramp-up are selected. Note that we cannot ensure that every selected seed increases the fault coverage. However, among the candidate seeds, there are

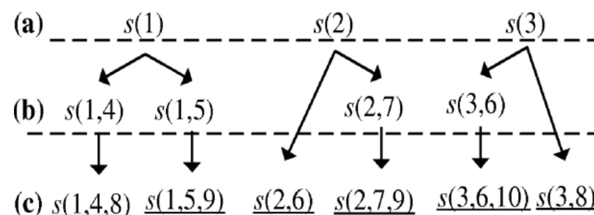

Figure. I. generation of candidate seeds for dynamic reseeding

Usually seeds that maximize fault coverage which is efficiently identified by the proposed metric. We also note that even a small value of test data can provide huge increase in the fault coverage offered by the resulting seeds, and thus the encoding process is effective for large circuits. Very often during the candidate seed computation process, there are test cubes that include the same high number of specified bits and, at the same time, their encoding requires the displacement of the same minimum number of variables. Although the encoding process helps the selection of such cubes in order to reduce the seed count, usually only one of them can be encoded in a single seed, because, after it's encoding, the remaining variables do not sufficient to encode any of the rest test data. We exploit this property to further maximize the quality of the candidate seeds. Especially, during the computation of every candidate seed, the first time that a set of test cubes, say ST, is found with the above property, we select m of them (m is a predetermined parameter) and we individually encode them in the candidate seed. Thus, the candidate seed is replaced by m new ones, and each one of them embeds all the test cubes of the initial candidate seed (i.e., the one that we replace with the m new ones), as well as one of the test cubes of set ST. Note that this is done only for the first (and consequently most-specified) m test cubes found for each one of the initially computed candidate seeds, in order to keep the candidate-seeds' volume very low. For the same reason, we set the maximum value of m equal to 2. Thus, the volume of computed candidate seeds cannot exceed, which is relatively very small.

EXPERIMENTAL RESULTS

In this section, we value the effectiveness of the proposed method. We verify the results on cadence took using test sets targeting complete coverage of stuck-at faults and bridging faults for the largest ISCAS'89 circuits and a subset of the IWLS'05 circuits. The number of scan chains was set equal to 25 for the ISCAS circuits, 40 for the medium sized IWLS circuits, and 150 for the large *Ethernet* IWLS circuit.

To establish the advantage of dynamic reseeding compared to the classical reseeding-based method of [15], which uses a different output deviation-oriented metric, we have proposed the method of [15] as well as the defect-unaware method [4]. We have combined these two

methods for experiments using compacted stuck-at test sets in order to minimize the number of seeds need as well as achieve high fault coverage.

To determine the effectiveness of the proposed defect-aware reseeding method, we consider the fault coverage of un-modeled faults, namely transition and bridging faults, obtained by applying seeds to the circuit under test. We use the launch-on-capture (LOC) scheme, to apply test-vector pairs. For calculating the bridging fault coverage, we randomly selected 100 K pairs of lines for each circuit. For each pair, four bridging faults were simulated. Note that none of these two fault models were targeted by the test sets. Finally, as the metric proposed in section III used, for generating results.

Table 1.fault coverage improvement with test cubes (iscas-89)

			Test cubes		
Circuit	Inp	B	f.c	seeds	time
S1423	89	7	95.72	49	987.89
S1423	89	8	95.98	51	987.90
S1423	89	9	96.11	53	987.91
S1423	89	10	97.34	54	987.92
S1423	89	11	97.89	57	987.93
S1423	89	12	98.00	59	987.95
S5378	205	18	98.76	212	988.00
S5378	205	28	99.02	234	988.12
S9234	224	32	99.23	320	988.32
S9234	224	35	99.54	324	988.33
S9234	224	37	99.76	332	988.34

CONCLUSION

We have implemented a defect-oriented dynamic LFSR reseeding technique for detection of un-modeled defects such as transition fault and bridging fault. This technique uses the output-deviations metric for grading the test patterns produced by LFSR seeds. We calculated un-modeled fault coverage using transition faults and bridging faults as surrogate fault models. Compared to classical and static LFSR reseeding, dynamic LFSR reseeding offers higher fault coverage, without any loss of compression.

REFERENCES

[1]. Pomeranz, ''Computing Seeds for LFSR-Based Test Generation from Nontest Cubes,'' IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEM, 43, 2016, 36-42.

[2]. B. Koenemann, "LFSR-coded test patterns for scan designs," in Proc. Eur. Test Conf., 1991, 237–242.

[3]. S. Hellebrand, S. Tarnick, J. Rajski, and B. Courtois, "Generation of vector patterns through reseeding of multiple-polynomial linear feedback shift registers," in Proc. Int. Test Conf., 1992, 120–129.

[4]. C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, B. Keller, and B. Koenemann, "OPMISR: The foundation for compressed ATPG vec- tors," in Proc. Int. Test Conf., Oct. 2001, 748–757.

[5]. J. Rajski et al., "Embedded deterministic test for low cost manufacturing test," in Proc. Int. Test Conf., 2002, 301–310.

[6]. N. A. Touba, "Survey of test vector compression techniques," IEEE Des. Test Comput., 23(4), 2006, 294–303.

[7]. S. Alampally, R. T. Venkatesh, P. Shanmugasundaram, R. A. Parekhji, and V. D. Agrawal, "An efficient test data reduction technique through dynamic pattern mixing across multiple fault models," in Proc. IEEE 29th VLSI Test Symp.,2011, 285–290.

Copyrights © International Journal of Intellectual Advancements and Research in Engineering Computations,
www.ijiarec.com

[8]. D. Czysz, G. Mrugalski, N. Mukherjee, J. Rajski, P. Szczerbicki, and J. Tyszer, "Deterministic clustering of incompatible test cubes for higher power-aware EDT compression," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., 30(8), 2011, 1225–1238.
[9]. Chandra, J. Saikia, and R. Kapur, "Breaking the test application time barriers in compression: Adaptive scan-cyclical (AS-C)," in Proc. Asian Test Symp., 2011, 432–437.
[10]. O. Acevedo and D. Kagaris, "Using the Berlekamp–Massey algorithm to obtain LFSR characteristic polynomials for TPG," in Proc. Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst., 2012, 233–238.
[11]. X. Lin and J. Rajski, "On utilizing test cube properties to reduce test data volume further," in Proc. IEEE 21st Asian Test Symp., 2012, 83–88.
[12]. T. Moriyasu and S. Ohtake, "A method of one-pass seed generation for LFSR-based deterministic/pseudo-random testing of static faults," in Proc. Latin-Amer. Test Symp., 2015, 1–6.
[13]. Pomeranz 2013, "Non-test cubes for test generation targeting hard-to-detect faults," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., 32(12), 2013, 1957–1965.
[14]. X. Lin and J. Rajski , "On utilizing test cube properties to reduce test data volume further," Proc. IEEE 21st Asian Test Symp.,2012, 83–88.
[15]. M. J. Shulte and J. E. Stine, "The symmetric table addition method for accurate function approximation," J. VLSI Signal Process. 11, 1999, 1–11.
[16]. V. G. Oklobdzija, "An algorithmic and novel design of a leading zero detector circuit. Comparison with logic synthesis," IEEE Trans. Very Large Scale Integr. VLSI) Syst., 2(1), 1994, 124–128.

Copyrights © International Journal of Intellectual Advancements and Research in Engineering Computations, www.ijiarec.com