

Care Bit Density and Test Cube Clusters: Multi-Level Compression Opportunities

Bernd Koenemann
Cadence Design Systems, Inc.,
San Jose, CA
e-mail: berndk@cadence.com

Abstract:

Most of the recently discussed and commercially introduced test stimulus data compression techniques are based on low care bit densities found in typical scan test vectors. Data volume and test times are reduced primarily by compressing the don't-care bit information. The original care bit density, hence, dominates the theoretical compression limits. Further compression can be achieved by focusing on opportunities to compress care bit information in addition to the don't-care bit information.

This paper discusses at a conceptual level how data compression based on test cube clustering effects, as used in Weighted Random Pattern methods, could be combined with care bit oriented methods to achieve multi-level test stimulus compression.

Introduction and Background

Test data compression using simple on-chip decoding hardware has become a very hot topic in research as well as in commercial scan test generation tools. The recently introduced commercial scan test data compression tools take advantage of the fact that typical scan test vectors resulting from Automatic Test Pattern Generation (ATPG) contain relatively few care bits which must be at specific logic values for target fault detection. The vast majority of bit values in the test vectors are don't-care bit values that are generated by more or less arbitrary fill algorithms. How these two scan test properties - the relatively low density of care bits and the freedom to choose don't care bit values - can be utilized for test data compression was first described in [1]. The so called LFSR-Coding algorithm introduced in that paper shows how an LFSR (Linear Feedback Shift Register) seed can be calculated, such that cycling the LFSR with that seed will produce the correct care bit values while filling the don't-care bits with pseudo-random values as a by-product. Only the care bit values contribute to the seed calculation. It was shown that under certain statistical assumptions, a size of the seed vector needed to represent the care bit values essentially is somewhat larger than the

number of care bits. In other words, the original care bit density more or less determines the achievable compression ratio. For example, if the original care bit density in a test vector is 2.5%, then a close to 40x stimulus data compression ratio can theoretically be achieved by using LFSR seeds to represent the care bits.

Subsequent technical papers by different researchers have introduced various extensions and improvements to the original LFSR-Coding method and have shown empirically that the theoretical compression limit can indeed be approached [e.g., 2, 3, 4].

The seed calculation used in LFSR-Coding takes advantage of the fact that the stimulus values generated by cycling the LFSR are linear Boolean sums (XOR sums) of the seed values. The care bit values in a test vector, thus, create a set of linear equations that is solved to determine the appropriate seed vector for the test. We therefore refer to the encoding algorithm as linear Boolean encoding. The linear Boolean encoding concept can be generalized to work in conjunction with linear decoding circuits other than LFSRs. In fact, any combinational or sequential linear network could be used. The simplest form of a linear decoding network is to fan out from each scan input to multiple scan chains. This approach is known as broadcast scan, and has been shown to produce very good compression results also [5, 6, 7].

The on-chip decoding methods can be complemented by equally simple and effective tester-resident methods. One concept that is used in practice is to algorithmically generate the fill data for the don't-care bits on the tester. The RLE (Run Length Encoding) approach described in [7, 8] utilizes the repeat features of certain testers for this purpose. The ATPG fill algorithm is changed to repeat the last care bit value rather than using the more common pseudo-random fill. This simple change creates runs of repeating scan-in vectors that can be represented by a single broadcast vector and a repeat op-code rather than loading the vectors directly into buffer memory. It has been shown [7] that the tester-resident RLE method can be combined with moderate on-chip broadcast scan (e.g., 10x or 20x fan-out) for very dramatic data volume reductions (100x).

All techniques described so far focus on the compressing the don't-care bit values. However, there exists a long practical history of test data compression

using a fundamentally different approach. Weighted Random Pattern (WRP) test exploits the fact that test cubes (the care bits in a test vector) for multiple fault tests tend to form clusters with small Hamming Distance between the respective cubes in each cluster [9]. That is, the cubes in a cluster differ from each other in very few bit positions. For example, the stuck-at fault test cubes for multi-input AND gate differ from each other in at most 2 bit positions irrespective of the width of the AND gate. Each stuck-at fault test for a 20-input AND gate, for instance, has 20 care bits. There are $20+1=21$ different stuck-at fault tests associated with the 20-input AND gate. The total number of care bits for all tests, hence, is 420 bits. From an information content point of view, recognizing the clustering effect, each cube could be represented as the XOR sum of one common base cube for each cluster and a difference vector for each additional unique test cube within the respective cluster. The difference vectors for the AND gate example will have at most 2 non-zero positions, meaning that the difference vectors are very amenable to data compression using sparse vector techniques. In other words, instead of storing 21 full test cubes for the AND gate, the tests would be represented by some form of data for a single base cube and by highly compressed data for the 21 difference vectors.

In the WRP approach, the base cube information is implicitly expressed by so called weight value sets [9, 10] that more or less strongly bias the output values from a Pseudo Random Pattern Generator (PRPG) towards the base cube values. Each test vector bit position is associated with a 4-bit weight value, meaning that representing the base vector in this way consumes 4 test vectors' worth of data volume. The care bit values of the vectors produced by cycling the weighted PRPG structure will, due to the biasing, be within a close Hamming distance of the base cube. Groups of "trial" vectors with different PRPG seeds are fault-simulated to determine which seeds actually produce useful test cubes. The resulting effective seeds can be interpreted as being a compact data representation for difference vectors that complement the base cube information encoded in the weight sets. Storage-efficient hardware/software methods for jumping from one effective PRPG seed to the next one are described in [11]. For large circuits with many scan cells, the data volume for the seed information is negligible compared to the weight set information. Many years of experience with WRP suggest that the combined storage for the weight value sets and effective PRPG seeds tends to be 5x to 20x less than the storage required for a full stored pattern scan test set with equivalent fault coverage. WRP, often combined with PRPG seed jumping, has been and still is in production use for some of the industry's most complex chips [12, 13].

The concepts discussed in the following show how methods utilizing low care bit density and methods taking advantage of test cube clustering could be used in concert to achieve higher compression ratios than can be achieved with either method by itself.

Some WRP Basics

To better appreciate the opportunities and problems arising from a combination of care bit based and cluster based compression techniques, it is useful to first review some WRP basics.

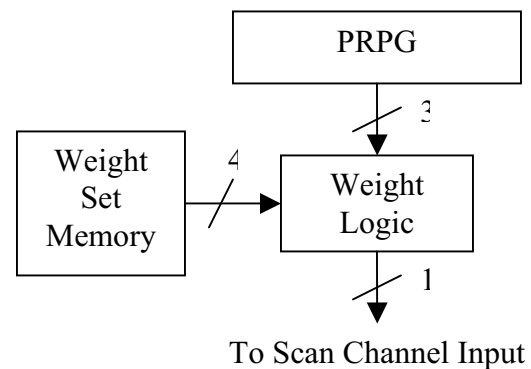


Figure 1: WRP Decoding Logic for One Scan Channel

Figure 1 shows a high level diagram of the WRP decoding logic for one scan channel. A multi-bit weight value determines the strength and direction of the bias. Simple combinational weight logic [12] receives the multi-bit weight value and several bits from a Pseudo-Random Pattern Generator (PRPG) source to produce a single test bit value to be loaded into the associated on-chip scan channel. The weight values typically are stored in a weight set memory. Each scan cell on the chip is associated with a corresponding multi-bit weight value for each weight set. The total amount of storage for the weight information is determined by the number of scan cells times the number of bits per weight value times the number of weight value sets. The memory advantage over conventional stored pattern testing is that the same weight value set is reused multiple times to produce multiple test vectors. Despite this storage advantage, the weight value set data volume is still too high to be efficiently stored on-chip. Hence, the weight value memory and weighting logic tend to be part of special WRP-capable ATE.

The weight values for each scan cell are derived from ATPG. The basic concept is to let the ATPG tool generate one or more than one test vector at the time. The tests are

translated into weight values by assigning a strong weight bias towards all required care bit values in the vector while assigning, while the don't-care values are left unbiased (which results in un-weighted pseudo-random values). It has been shown, that unlike in conventional ATPG, test cubes with a small number of conflicting care bit values can still be combined into a weight set. The conflicting bit positions are set to unbiased weighting (i.e., pseudo-random). Likewise, bit positions that remain don't cares after the merging of test cubes are set to unbiased weighting. After a weight set has been calculated, groups of test vectors are generated by emulating the decoding hardware with the given weight value set and multiple different PRPG seeds. The resulting test vectors are fault simulated to determine which, if any, new faults are detected. The effective test vectors (the ones that detect new faults), as mentioned before, are fully characterized by their associated weight value set and a PRPG seed.

Hybrid and Biased Random Patterns

The original WRP implementation used 4-bit weight values. Hybrid Random Pattern (HRP) testing uses 2-bit weight values [14]. Figure 2 shows a possible implementation of the weighting logic for HRP.

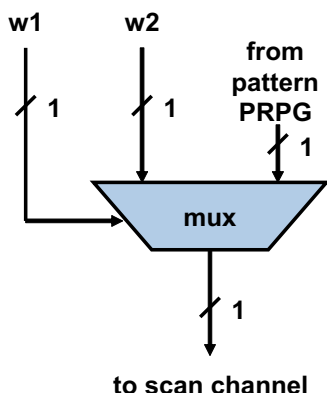


Figure 2: Weighting Logic for HRP

One of the weight value bits (labeled w1 in the picture) selects between a PRPG and the second weight value bit (w2) as the source of data for the scan channel input. If the PRPG is selected, then the value of w2 doesn't matter. If w2 is selected, then the value of w2 determines whether a logic 0 or a logic 1 is scanned into the scan channel for the respective scan cycle.

The weight values are derived from ATPG. During compaction, a small number of conflicts are allowed between the care bits of test cubes that are merged into a test vector. The compaction process is illustrated in Figure 3.

x	x	1	x	0	0	x	u	u	1	0	x	x	x	x	0	x	1	x	x
x	x	x	x	x	1	x	x	1	1	x	x	0	x	x	0	x	x	x	x
x	x	1	x	0	u	x	u	u	1	0	x	0	x	x	0	x	1	x	x

Figure 3: Test Cube Merging for HRP

The top row depicts a vector already in the merge buffer. The 'u' values indicate bit positions with conflicting care bit values. Don't care bit positions are denoted by an 'x'. Non-conflicting care bits are assigned their respective logic value '0' or '1'. The middle row shows a new test cube to be merged and the bottom row illustrates how the new test cube can be merged into the test vector. After the merging is completed, any bit positions with 'x' or 'u' are set to unbiased pseudo-random values, while bit positions with '0' and '1' are assigned the respective fixed logic value. As in WRP, groups of "trial" vectors are derived by modifying the PRPG seed and the resulting vectors are fault-simulated. Additional trial vectors with the same weight values and different PRPG seeds are generated and simulated until some cut-off criterion is reached (e.g., no new faults detected in last n vectors). Then, the ATPG tool is invoked again targeting the remaining untested faults in the faults list to determine a new weight value set.

The maximum number of conflicts ('u' values) allowed during merging is determined such that with a high degree of probability the pseudo-random values assigned to those bit positions will resolve to the required test cube values within a certain number of trial vectors. For example, there is an over 98.5% likelihood that 3 'u' bits in will resolve correctly within 32 pseudo-random patterns. If the number of trials is increased to 64, then the likelihood increases to over 99.98%.

Biased Random Patterns (BRP) use single bit weight values [15] that modulate the polarity of a uniformly weighted pseudo-random pattern stream towards the respective care bit base values of test cube clusters. A possible implementation of the BRP weighting logic for a single scan channel is shown in Figure 4.

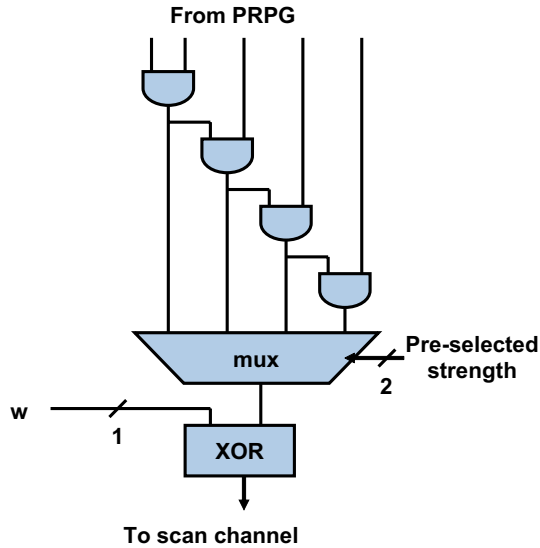


Figure 4: Weighting Logic for BRP

The logic shown in Figure 4 allows for pre-selecting a uniform 0-probability of 1/4, 7/8, 15/16, or 31/32. In other words the output from the mux will statistically contain more 0s than 1s. The single-bit weight value w selectively inverts the uniformly weighted data so that certain scan cells will receive predominantly 1s while the others receive predominantly 0s.

The strength of the uniform weighting determines how similar the resulting vectors will be to the base cubes. Stronger weighting effectively reduces the effective Hamming distance between the generated test vectors. For example, if a 1-probability of 1/32 is selected, then on average 1 out of 32 bit values will be different.

The algorithm for merging multiple test cubes into a BRP weight set is slightly different than the algorithm for HRP or WRP. Conflicts between care bit values in BRP cannot be resolved to unbiased pseudo-random values but have to be resolved to one of the conflicting values as is illustrated in Figure 5.

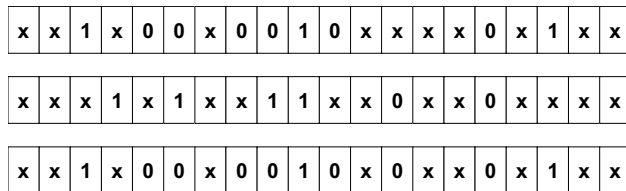


Figure 5: Test Cube Merging for BRP

The simple example shown in Figure 5 assumes that the care bit value already in the merge buffer (top row) wins over a conflicting care bit value in a new test cube (middle row). The means the merged test as is biased away from the “losing” conflict value. As a result it is

more difficult to recover the “losing” care bit value and the number of allowed conflicts must be limited accordingly.

In HRP non-conflicting care bit positions are fixed to the correct value and only conflicting bit positions are subject to random modification. In BRP, by contrast, due to the uniform weighting, all bit positions are equally subject to modification. How well different types of test cubes and associated clusters are covered depends on the size of the cube (t), the number of conflicts relative to the base cube encoded in the weight values (c), the uniform weighting strength (expressed by 0-probability, p), and the number of trial vectors. Some example probabilities are shown in Figure 6.

c	1				2			
t \ p	0.750	0.875	0.938	0.969	0.750	0.875	0.938	0.969
4	1.000	1.000	1.000	1.000	1.000	1.000	0.968	0.600
8	1.000	1.000	1.000	1.000	1.000	0.999	0.930	0.554
16	0.965	1.000	1.000	1.000	0.672	0.910	0.795	0.465
32	0.033	0.864	1.000	1.000	0.011	0.248	0.431	0.314

Figure 6: BRP Coverage Probabilities (1000 trials)

The results show, for example, that for strength of 17/16 and 31/32 test cubes of size 32 with 1 conflict are covered with high probability, while 2 conflicts are not less likely to be covered. In general, weaker weighting strength is better for tolerating more conflicts (allowing for larger Hamming distances between the cubes in the cluster) for smaller test cubes. Stronger weighting is needed to cover larger test cubes at the expense of less tolerance to conflicts. Since test cube sizes and clustering distances vary, it is advantageous to use different weight strengths. This effect was also seen in a relatively recent practical application of what amounts to biased random patterns in an industrial high-end custom processor product [16]. The overall concept and effectiveness of biased patterns has also been investigated in earlier experimental work [17].

Combining Clustering Effects and Care Bits

The test cube merging algorithms for HRP and BRP as discussed above, allow for some limited number of conflicts between merged test cubes but otherwise work very much like normal merging for stored pattern tests. Although the tolerance of conflicts enables somewhat better merging, the overwhelming majority of scan cells in large designs still remain don't cares. In other words, only the weights for a small subset of care bits must be at specific values to cover the target faults for the cluster with high probability. The weight values for the don't

care bits could be assigned arbitrarily without affecting the target fault coverage probabilities. Hence, it should be possible to apply sparse care bit techniques similar to those already used for test vector compression in conjunction with weighting.

A generic architecture for one possible on-chip implementation is illustrated in Figure 7.

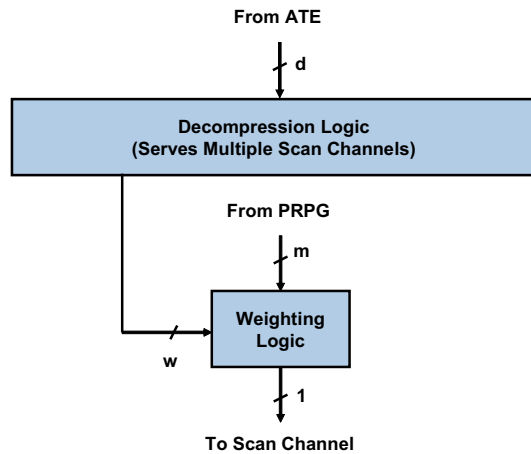


Figure 7: Generic Decompression Architecture

The Decompression Logic receives compressed weight information from the ATE and it sends the decompressed weight values to the weighting logic. A first level of data compression, thus, is achieved by encoding the care bits in the weight sets into a compact form. The second level of compaction results from repeatedly looping over the same weight value set multiple times with different PRPG seeds. The data savings come from the fact that the weight information for each set has to be stored only once. The PRPG seed information for the derived test vectors is negligible by comparison.

Estimation of Data Compression Efficiency

Experience with large scan designs indicates that it takes 6K scan tests or so to achieve high stuck-at fault coverage. If we assume a design with 1M scan cells, then it takes 6B bits to the test full stimulus information. The average care bit density even with good ATPG compaction for such designs can be assumed to be in the 2% range. The compression limit of the typical care bit based techniques is around 50x for a care bit density of 2%. The compressed stimulus data volume hence is around 12M bits.

Using WRP, it takes about 200 4-bit weight value sets to achieve comparable test coverage. Each weight value set for the design with 1M scan cells consumes 4M bits and the total data volume is 800M bits. This is about 7.5x

better than the original uncompressed test data volume, but worse than the compressed data.

The original HRP experiments on small designs showed only little difference in the number of weight sets compared to 4-bit weights [14]. A later developed improved WRP algorithm, by contrast, showed that the lower granularity in the 2-bit approach could result in an increased number of weight sets [10]. It therefore is assumed for the purpose of this paper that HRP requires 300 2-bit weight value sets compared to the 200 4-bit sets for WRP. The total stimulus data volume for HRP, hence, would be 600M bits, which is 10x better than the original data volume.

WRP and HRP allow for some conflicting values during the merging of test cubes. Hence, it must be expected that the resultant care-bit density in the weight value sets is higher than the care bit density in test vectors. For the sake of simplicity it is assumed that the care bit density in compacted WRP and HRP weight value sets is 50% higher than in compacted test vectors. For the example circuit the care bit density for the weight value would be around 3%, meaning that the compression limit for care bit based compression of the weight value sets should be around 33x. This compression is on top of the data reduction achieved from the utilizing the clustering effects. The conceptually achievable data volume compression using the HRP approach, hence, would be 330x, which is 6.6 times better than expected from care bit based compression alone.

Test time, however, is negatively impacted by the combined effects of having to supply several bits of weight data per scan cell and the fact that the number of effective vectors extracted from WRP/HRP is higher than the achievable vector count with normal compression.

The proposed method, hence, primarily addresses the need for data volume reduction.

Not enough data exists yet to make an educated guess for the BRP approach.

Summary and Future Work

This paper has focused on introducing some of the basic concepts of a possible multi-level data compression technology that exploits the complementary properties of sparse care bits and test cube clustering effects. Initial estimates suggest the possibility of tangible data volume reduction on top of the compression possible with care bit based techniques alone.

The next step is to explore different hardware implementation options and trade-offs. Particular areas of interest are refinements to the interface between the ATE and the on-chip resources to mitigate at least some of the test time impact, as well as the investigation of pragmatic trade-offs between data compression and test time.

References

1. B. Koenemann, "LFSR-Coded Test Patterns for Scan Designs", ETC '91, pp.237-242, 1991
2. Rajski
3. Pieter M. Trouborst, "LFSR Reseeding as a Component of Board Level Test", ITC '96, pp. 58-96, 1996
4. N. Zacharia et al. "Two-Dimensional Test Data Decompressor for Multiple Scan Designs, ITC '96, pp. 186-194, 1996□
5. K. Lee, J. Chen, and C. Huang, "Using a Single Input to Support Multiple Scan Chains," Proc. Int'l Conf. Computer-Aided Design (ICCAD 98), ACM Press, New York, 1998, pp. 74-78.
6. F. Hsu, K. Butler, and J. Patel, "A Case Study on the Implementation of the Illinois Scan Architecture," Proc. Int'l Test Conf. (ITC 01), IEEE Press, Piscataway, N.J., 2001, pp. 538-547
7. C. Barnhart et al., "Extending OPMISR beyond 10× Scan Test Efficiency", IEEE Design & Test Magazine, Sept.-Oct. 2002, pp. 65-73
8. C. Barnhart et al., "OPMISR: The Foundation for Compressed ATPG Vectors," Proc. Int'l Test Conf. (ITC 01), IEEE Press, Piscataway, N.J., 2001, pp. 748-757
9. J.A.Waicukaski et al., "A Method for Generating Weighted Random Test Patterns", IBM Journal of R&D, vol. 3 no.2, 1989, pp. 149-161
10. R. Kapur et al., "Design of an Efficient Weighted Random Pattern Generation System", Proc. International Test Conference, 1994, pp. 491-500
11. B. Koenemann, "A Pattern Skipping Method for Weighted Random Pattern Testing", Proc. European Test Conference, 1993, pp. 418-425
12. P. Gillis et al., "Test Methodologies and Design Automation for IBM ASICs", IBM Journal of R&D, vol. 40 no. 4, 1996, pp. 461-474
13. T. Foote et al., "Testing the 400MHz IBM Generatio-4 CMOS Chip", Proc. International Test Conference, 1997, pp. 106-114
14. B. Koenemann et al., "Hybrid Random Pattern Self-Testing of Integrated Circuits", US Patent 5612963, 1997
15. M. Kusko et al., "99% AC Test Coverage Using Only LBIST on the 1GHz IBM S/390 zSeries 900 Microprocessor", Proc. International Test Conference 2001, pp.587-592
16. M. Gruetzner and C.W. Starke, "Experience with Biased Random Pattern Generation to Meet the Demands of a High Quality BIST", Proc. European Test Conference 1993, pp. 408-417