World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:11, No:1, 2017

A Modified Run Length Coding Technique for Test Data Compression Based on Multi-Level Selective Huffman Coding

C. Kalamani, K. Paramasivam

Open Science Index, Electronics and Communication Engineering Vol:11, No:1, 2017 publications.waset.org/10007222.pdf

***Abstract*—**Test data compression is an efficient method for reducing the test application cost. The problem of reducing test data has been addressed by researchers in three different aspects: Test Data Compression, Built-in-Self-Test (BIST) and Test set compaction. The latter two methods are capable of enhancing fault coverage with cost of hardware overhead. The drawback of the conventional methods is that they are capable of reducing the test storage and test power but when test data have redundant length of runs, no additional compression method is followed. This paper presents a modified Run Length Coding (RLC) technique with Multilevel Selective Huffman Coding (MLSHC) technique to reduce test data volume, test pattern delivery time and power dissipation in scan test applications where redundant length of runs is encountered then the preceding run symbol is replaced with tiny codeword. Experimental results show that the presented method not only improves the test data compression but also reduces the overall test data volume compared to recent schemes. Experiments for the six largest ISCAS-98 benchmarks show that our method outperforms most known techniques**.**

***Keywords*—**Modified run length coding, multilevel selective Huffman coding, built-in-self-test modified selective Huffman coding, automatic test equipment.

I. Introduction

IN advanced VLSI technology, the system design contains huge number of transistors on a single chip, which leads to enlarge in volume of needed test pattern to test the circuits. The inefficient functionality and increasing the integration size of VLSI chips make testing for these chips more troublesome. The most two essential sources of the test cost are test data volume and test compression ratio.

If the size of test sets circuits increases, both the size of the VLSI architecture and its complexity also increases. The standard testing systems of advanced digital circuits need test patterns application which is done by a test pattern generator (TPG) to the circuit under test (CUT) then it is compared the results with known correct results. The increase in test data volume increases time to transport data from Automatic Test Equipment (ATE) to CUT. To conquer this issue, a few techniques have been proposed [1], [2]. Test data compression based on coding techniques are mostly used because it is independent of structural information

The coding techniques can be separated into five sorts [3]-[5]. Dictionary based coding [6] and LFSR-based reseeding coding [7] use fixed-length (FL) codeword length to code the FL symbol in the original test data. Huffman-based coding [8] uses variable length (VL) codeword to code the FL symbol in the original test data. Traditional run-length-based coding [9] uses FL codeword to code the VL symbol in the original test data. VIHC coding [10], Golomb coding [11], Frequency-Directed Run-length (FDR) coding [12], Alternating Run-length coding [13], [14], Extended Frequency-Directed Run-length (EFDR) coding [15] use VL codeword to code the VL symbol in the original test data. Equal Run Length Coding (ERLC) [18] is based on runs of 0's and 1's and discovers the relationship between two consecutive runs. A shorter codeword is used to represent the whole second run of two equal length consecutive runs. The benefit of these classifications of compression systems is that they can productively misuse correlations in the predefined bits and are usable on any set of test data. Consequently, they are powerful for IP cores for which no auxiliary data is available. Conversely, they are not as viable in misusing don't-care bits as linear methods and they necessitate more complex control logic.

The presented method is a combination of Modified Run Length coding (MRLC) [22] and Modified Selective Huffman Coding (MSHC) [17] with Multilevel Huffman Coding (MHC) [16]. To reduce test data volume by redundant length of runs is encountered then the preceding run symbol is replaced with tiny code word and gives better compression results techniques and necessitate less complex control logic.

The sections are organized as follows: Section II examines a modified RLC with the MLSHC. Section III portrays encoding algorithm and decoding architecture of the presented method. Section IV discusses implementation results. Finally, Section V summarizes the conclusion of the paper.

II. Proposed Technique

A. MRL Based MLSHC

The presented technique has done a few modifications in run length encoding scheme, which is extraordinarily designed in counter. The MRL gives a better compression ratio than conventional run length encoding. The MRL technique is experienced to excess the redundant run length then the preceding run symbol is replaced with tiny codeword. MRL

Ms. C. Kalamani is with the ECE Department, Dr.Mahalingam College of Engineering and Technology, Coimbatore, Tamilnadu, India (e-mail:kalamec18@gmail.com).

Dr. K. Paramasivam is with the ECE Department, Kumara guru College of Technology, Coimbatore, Tamilnadu, India (kp_sivam@yahoo.com).

Open Science Index, Electronics and Communication Engineering Vol:11, No:1, 2017 publications.waset.org/10007222.pdf

technique is very similar to the EFDR technique except that the MRLC code for run-length i, the EFDR code for run-length $i+1$.

Let a test set comprising of three 12 bit lengths of the test bits, which has the pattern block size $b_h = 4$. Hence forth, the three 12bit length test data are split into a set of nine 4-bit blocks,

B = {10X1; XX10; 1XXX; X011; 10X1; 10X1; 0X10; 101X; 1XXX}

Every unspecified block can contain from 1 to $2^4 = 16$ conceivable minterms binary combinations.

In the MRL technique, the most regularly occurred unspecified block is distinguished. It is then compared with the following most regularly occurred unspecified block to check whether there is a conflict in any bit position such as, one has a 1 and the other has a 0. If there is no conflict, it is the length of the Huffman codeword of the each encoded distinct block $b_i, i \in [1,m]$ and l_b is the length of the original unencoded data block.

Note that the weights of the neighbor nodes p_A and p_B can be written conflict, then by identifying all bit positions whether block has a predetermined value, they are merged.

The Multilevel Huffman is defined in Two Main Points as:

1. The utilized Huffman code is specified by taking into account the occurrence frequency of multiple kinds of information and not only that of the data blocks. As a consequence, the compression scheme is more sophisticated and hence much more effective.
2. The unencoded data blocks are indicated by using a separate Huffman codeword instead of appending an extra bit to every block either encoded or not. This separate codeword precedes only each unencoded block.

The modified selective Huffman code is generated as follows. Consider that D_1 and D_2 are two datasets.

The number of distinct blocks (m) is same for both the set. For the first set $D_1 : p_1, p_2, \ldots, p_m$, are the occurrence frequencies of m distinct blocks $b_1, b_2, \ldots, b_m$. $(p_1 \geq p_2 \geq \ldots \geq p_m)$. $l_i, i \in [1,m]$ is the length of the Huffman codeword of the each encoded distinct block $b_i, i \in [1,m]$ for set D_1. For the another set $D_2 : p'_1, p'_2, \ldots, p'_m, p'_{m+1}$, are the occurrence frequencies of $m+1$ distinct blocks $b'_1, b'_2, \ldots, b'_m, b'_{m+1}$. $l'_i, i \varepsilon [1, m+1]$ is the length of the Huffman codeword of the each encoded distinct block $b'_i, i \in [1, m+1]$ for set D_2. It is observed that the length of first $m-1$ codewords for both the data set is same irrespective of their frequency of occurrence.

$$l_i = l'_i \; for \; i < m$$

For m^{th} codeword, the length is different by 1 bit for both the data set.

$$l_i + 1 = l'_i \; for \; i = m$$

The $m+1^{th}$ codeword length in set D_2 is same as m^{th} codeword length in set D_2.

$$l'_m = l'_{m+1}$$

The average block length of the presented encoding, which is the average length of the codeword and the unencoded data blocks in the compressed test set is given by the relation

$$W_{MS-Huffman} = [l_1 p_1 + l_2 p_2 + \cdots + l_{m-1} p_{m-1}]_{l_m p_m} + I_b p_u$$

where $l_i, i \in [1.m]$as

$$p_A + p_B + p_u = 1$$
$$p_A + p_B = 1 - p_u = p_1 + p_2 + \cdots + p_m$$

If the encoding is equivalent if and only if $p_u \geq p_A$ and $p_u \geq p_B$.

III. Proposed Algorithm

A. Encoder Algorithm

Input: Test vector tv[I]with $0 \leq I < L$, L length of the vector and pattern block size k.

Output: encoded output sequence enc_{seq}

1. $\text{enc}_{\text{seq}} \leftarrow$ m − bit binary expansion of b_h ,
$\text{B}_0 =$ initial specified bit
2. tv ← tv ‖ N Where the number of N is k − length(tv)mod k
3. for i ← 1, …. N
for j ← $\text{b}_{\text{min}}$to b_{max}
if ($\text{tv}_\text{j} = \text{B}_0$ or $\text{tv}_\text{j} = \text{'X'}$)
cp ← tv[1]tv[2] … tv[N]
ind_v ← NULL
else
j ← j + $\lfloor \log_2(\text{N}) \rfloor + 1$
$\text{B}_0 = \overline{\text{B}_0}$
i = 1
end if
end for
end for
4. Encode the elements of each block
for 1 ≤ n ≤ (length(tv)mod k)/k
puc ← tv[Nk]tv[Nk − 1] … tv[Nk + k − 1]

for 1 ≤ i ≤ k
if cp[i] = puc[i] ≠ X
ind[i] = 0
else if (cp[i] ≠ puc[i] and cp[i] ≠ X and puc[i] ≠ X)
ind[i] = 1
$\text{enc}_{\text{seq}} = \text{enc}_{\text{seq}} \| 0$
else
ind[i] = X
end if
end for
if(ind[i_1] and ind[i_2] ≠ X and ind[i_1] ≠ ind[i_2])
for $1 \leq \text{i}_1 \neq \text{i}_2 \leq \text{k}$
$\text{enc}_{\text{seq}} \leftarrow \text{enc}_{\text{seq}} \| \text{cp} \| \text{ind\_v} \| 0$
cp ← puc
ind_v ← NULL
else if(ind[i_0] ≠ X)
cp[i] ← puc[i] ⊕ ind[i_0]
else cp[i] ← puc[i]
puc[i] ≠ X
ind_v ← ind_v ‖ 1 ‖ ind[i_0]

Open Science Index, Electronics and Communication Engineering Vol:11, No:1, 2017 publications.waset.org/10007222.pdf

```
    end if
      end if
end for
```

IV. Decoding Architecture

The compressed test data should be decoded utilizing on-chip hardware before being connected to the scan-chain. Fig. 1 demonstrates the decompression architecture utilized to decompress the encoded data. It comprises of multilevel selective Huffman finite state machine (MSH-FSM), ATE, look up table (LUT), counter, two multiplexers, and decoder controller. In this decoder architecture, MSH-FSM can be obtained simply by adding a counter indicating the effect of parameter, decoder controller block, counter, multiplexers (MUXes), and LUT.

Fig 1 represents the block diagram of the test decoder architecture. The technique involves measurable coding for compression of data to be stored on ATE. The compressed data from the ATE will be serially transferred to chip utilizing a single scan-in input. The tester channel D_{in} is equivalent to encoded blocks of length in serial fashion. E/N shows the status of the bit-stream on D_{in} channel. The loaded data D_{in} is applied to a multiplexer. The selection line of this multiplexer is connected to E/N signal.

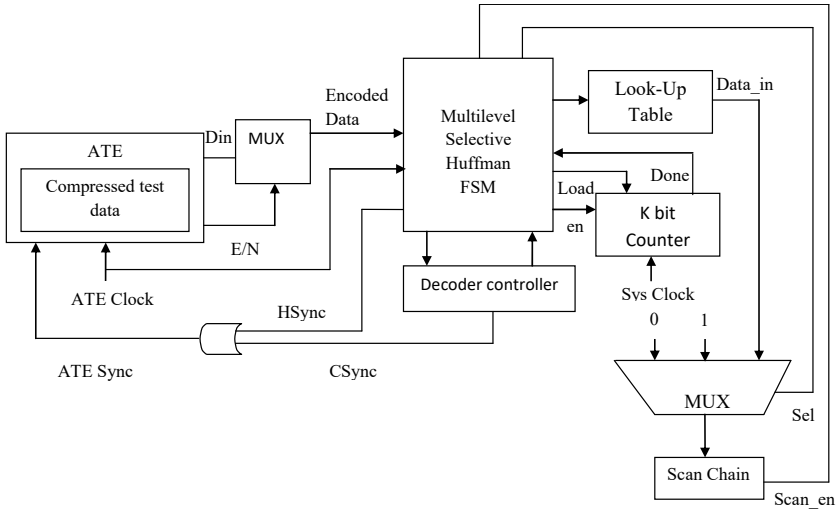

Fig. 1 Decoder architecture

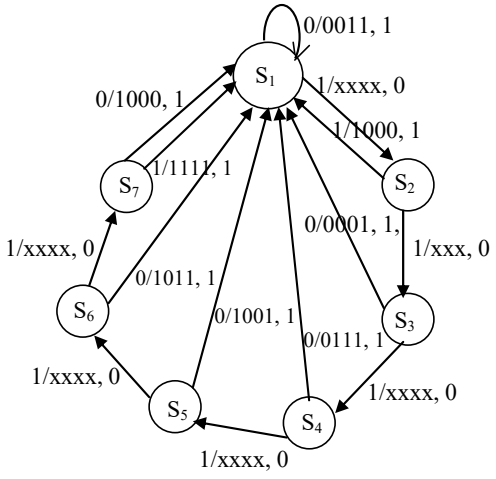

Fig. 2 FSM

If the data bits of D_{in}channel match to codewords of relating blocks, E/N is active high. If the data bits on D_{in} match to unencoded data blocks, E/N is active low. If E/N is low, the data on D_{in} are applied directly to serializer bypassing the decoder. If the E/N is high, the data on D_{in} are coded and connected to decoder that decodes the data bits. The MSH-FSM receives the new codeword for decoding. At the point when the value of the counter and the output of the lookup table will be equivalent, sel value is stopped, and offers sign to the MSH-FSM for next process in LUT. The output of the LUT selects the data depending sel value of the MSH-FSM. The state diagram utilized for proposed decoder is appeared in Fig. 2. The number of states is identical to the total number of branches in the multilevel selective Huffman tree minus one. The MSH-FSM starts from stateS_1, and changes its state based on D_{in} bit from ATE. Subsequent to detecting a codeword, decoding starts at the frequency of system clock and MSH-FSM back to its default state i.e. S1 state.

V. Results and Discussion

The compression and decompression techniques were utilized to compress test sets for the ISCAS'89 benchmark circuits. The compression technique is implemented utilizing MATLAB7.0 language, and Verilog code. Most of the parts on test data compression utilize the same circuits for experiments.

The compression ratio is computed as

$$Compressed\ ratio = \frac{Length\ of\ Original\ data - Length\ of\ Compressed\ data}{Length\ of\ Original\ data} \times 100$$

Block size used in this algorithm is one of the important parameter which decides the efficiency of compression. This

Open Science Index, Electronics and Communication Engineering Vol:11, No:1, 2017 publications.waset.org/10007222.pdf

algorithm is experimented with various block size as shown in Table I.

TABLE I
COMPRESSION RATIO FOR DIFFERENT BLOCK SIZES IN A MRLMHC ALGORITHM

Benchmark Circuit	No. of bits	Compressed bits				
		Block Size (m_h)				
		4	5	6	7	8
s5378	20758	5507	5055	5810	5959	5150
s9234	25935	8750	8160	8327	8224	8020
s13207	163100	21388	20746	21424	21438	21170
s15850	57434	12382	12975	12096	12368	12681
s38417	113152	24386	24589	24546	24089	23966
s38584	161040	39987	40561	40634	38730	39084

Table II compares the compression ratio of the proposed method with other compression techniques with Huffman as in [20], Selected Huffman, RLHC, Optimal Huffman as in [21], Multilevel Huffman, Modified Selected Huffman techniques for ISCAS'89 benchmark circuits.

The presented method gets better compression ratio as compared with other compression techniques. For s5378 benchmark circuits, the technique gets better (75.19%) compression ratio, and multilevel huffman gets (28.8%) compression ratio as compared with other techniques. For s9234, s13207, s15850, s38417, s38584 benchmark circuits, the presented method gets better (69.04%, 87.02%, 77.9%, 78.83%, and 75.73%) compression ratio, and multilevel Huffman gets less compression ratio as compared with proposed compression techniques.

TABLE II
PERCENTAGES OF COMPRESSED-DATA REDUCTION OF THE PRESENTED METHOD WITH OTHER COMPRESSED TECHNIQUES

Benchmark Circuit	Huffman [20]	S H [8]	RLHC [19]	O H [21]	MH [16]	MSH [17]	**Proposed**
s5378	54.6	37.5	71.9	37.8	28.8	54.2	***75.19***
s9234	42.9	26.7	66.5	24.9	15.1	55.5	***69.08***
s13207	60	40.6	**86.3**	17.8	22.7	70.3	***87.02***
s15850	54.2	32.6	**77.1**	28.7	6.9	64.2	***77.92***
s38417	63.6	36.2	**73.8**	33.1	26.7	59.9	***78.82***
s38584	57.2	33.7	**76.2**	31.2	14.1	61.9	***75.73***

VI. CONCLUSION

MRL with multilevel selective Huffman compression technique reduced the test data volume and better compression ratio in this paper. It was demonstrated that the technique gives better compression compared to existing methods like Huffman, selective Huffman, RLHC, optimal Huffman, multilevel selective Huffman compression techniques. It was verified and established with experimental results that the proposed technique not only enhances the test data compression but also reduces the general test data volume compared to recently implemented techniques. Tests for the six biggest ISCAS-98 benchmarks demonstrate that presented technique outperforms well known techniques.

REFERENCES

[1] V. Iyengar, K. Chakrabarty and B. T Murray, "Built-in Self-Testing of Sequential Circuits Using Precomputed Test Sets", Proc. VTS., pp.418-423, 1998.
[2] Jas, J. Ghosh-Dastidar and N. A. Touba, "Scan Vector Compression/Decompression Using Statistical Coding", Proc. VTS, pp.114-120, 1999.
[3] N.A. Touba, "Survey of test vector compression techniques", IEEE Design and Test of Computers 23 (4) (2006) 294–303.
[4] Kalamani, C & Paramasivam, K, "Survey of Low Power Testing Using Compression Techniques", International Journal of Electronics & Communication Technology, vol. 4, no. 4(2013), pp. 13-18.
[5] Wenfa Zhan, Huaguo Liang, Feng Shi, "Test data compression scheme based on variable-to-fixed-plus-variable-length coding", Journal of Systems Architecture 53 (11) (2007) 877–887.
[6] L. Lei, K. Chakrabarty, "Test data compression using dictionaries with fixed-length indices", in: Proceedings of the VLSI Test Symposium, 2003, pp. 219–224.
[7] Al-Yamani, E. McCluskey, "Seed encoding for LFSRs and cellular automata", in: Proceedings of the Design Automation Conference, 2003, pp. 560–565.
[8] Jas, J. Ghosh-Dastidar, N.A. Touba, "An efficient test vector compression scheme using selective Huffman coding", IEEE Transactions on Computer- Aided Design 23 (6) (2003) 797–806.
[9] Jas, N.A. Touba, "Test vector decompression via cyclical scan chains and its application to testing core-based designs", in: Proceedings of the International Test Conference, 1998, pp. 458–464.
[10] T. Paul, B. Al-Hashimi, N. Nicolici, "Variable-Length input huffman coding for system-on-a-chip test", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 22 (6) (2003) 783–796.
[11] Chandra, K. Chakrabarty, "System-on-a-chip test data compression and decom- pression architectures based on Golomb codes", IEEE Transitions on Computer-Aided Design of Integrated Circuits and System 20 (3) (2001) 355–368.
[12] Chandra, K. Chakrabarty, "Test data compression and test resource partitioning for system-on-a-chip using frequency-directed run-length (FDR) codes", IEEE Transactions on Computers 52 (8) (2003) 1076–1088.
[13] Chandra, K. Chakrabarty, "Reduction of SOC test data volume, scan power and testing time using alternating run-length codes", in: Proceedings of the IEEE/ACM Design Automation Conference, 2002, pp. 673–678.
[14] Wuertenberger, C.S. Tautermann, S. Hellebrand, "A hybrid coding strategy for optimized test data compression", in: Proceedings of the International Test Conference, 2003, pp. 451–459.
[15] Aiman El-Maleh, "Test data compression for system-on-a-chip using extended frequency-directed run-length code", IET Computers and Digital Techniques 2 (3) (2008) 155–163.
[16] Kavousianos, Xrysovalantis, Emmanouil Kalligeros, and Dimitris Nikolos. "Multilevel Huffman coding: an efficient test-data compression method for IP cores", Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 26.6 (2007): 1070-1083.
[17] Mehta, Usha Sandeep, Kankar S. Dasgupta, and Nirnjan M. Devashrayee. "Modified selective Huffman coding for optimization of test data compression, test application time and area overhead", Journal of Electronic Testing 26.6 (2010): 679-688.
[18] W. Zhan, A. El-Maleh, "A new scheme of test data compression based on equal- run-length coding (ERLC)",, Integr. VLSI Journal. 45(1) (2012) 91–98.
[19] M. Nourani, M. H. Tehranipour, "RL-Huffman encoding for test compression and Power reduction in scan applications", ACM Trans. Design Autom. Electron.Syst. 10(1)(2005)91–115.
[20] D. Huffman, "A method for the construction of minimum-redundancy codes", Proc. IRE40 (9) (1952) 1098–1101.
[21] X. Kavousianos, E. Kalligeros, D. Nikolos, "Optimal selective Huffman coding for test-data compression", IEEETrans.Comput.56 (8)(2007) 1146–1152.
[22] M. VidyaSagar, J.S. Rose Victor, "Modified Run Length Encoding Scheme for High Data Compression Rate'', International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) 2(12), (2013) 3238-3241.