ADVANCED FREQUENCY-DIRECTED RUN-LENTH BASED CODING SCHEME ON TEST DATA COMPRESSION FOR SYSTEM-ON-CHIP

Zhang Ying, *Wu Ning*, *Ge Fen*

(College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, P. R. China)

Abstract: Test data compression and test resource partitioning (TRP) are essential to reduce the amount of test data in system-on-chip testing. A novel variable-to-variable-length compression codes is designed as advanced frequency-directed run-length (AFDR) codes. Different from frequency-directed run-length (FDR) codes, AFDR encodes both 0- and 1-runs and uses the same codes to the equal length runs. It also modifies the codes for 00 and 11 to improve the compression performance. Experimental results for ISCAS 89 benchmark circuits show that AFDR codes achieve higher compression ratio than FDR and other compression codes.

Key words: test data compression; FDR codes; test resource partitioning; system-on-chip

CLC number: TP302 **Document code**: A **Article ID**: 1005-1120(2012)01-0077-07

INTRODUCTION

With the improvement of integrated circuit design and manufacturing technology, the complexity of test vector sets has increased followed by the increase of testing time and cost[1]. Conventional testing methods store the test vector and test responses in automatic test equipment (ATE) with limited test equipment speed, I/O channels and storage space. The bandwidth of ATE is the bottleneck of high speed testing and will increasingly impact chip testing in complex system. One solution classified as test resource partitioning (TRP) methods is to compress test vectors for reducing storage requirements and test time. The scheme requires additional decompression module set in the original chip, so relatively simple realization circuit and the compression effect become key issues for compression coding methods.

Compression coding methods for test vectors are classified into three kinds: run-length based codes, dictionary based codes and statistical codes[2]. The run-length based codes adopts different codewords based on the sequence's length (runs of 0s or 1s) distribution without the constraint of coding length. It has better compression effectiveness and relatively simple realization circuits compared with dictionary based codes and statistical codes. The classical run-length coding methods, including Golomb codes[3] and frequency-directed run-length(FDR) codes[4], feature the prefix and the tail constituting the codeword. The prefix shows group characteristics and the information of the source codes length. The tail is assigned to corresponding binary codes according to

Foundation items: Supported by the National Natural Science Foundation of China(61076019, 61106018); the Aeronautical Science Foundation of China (20115552031); the China Postdoctoral Science Foundation (20100481134); the Jiangsu Province Key Technology R&D Program (BE2010003); the Nanjing University of Aeronautics and Astronautics Research Funding(NS2010115); the Nanjing University of Aeronatics and Astronautics Initial Funding for Talented Faculty(1004-YAH10027).

Received date: 2011-04-25; **revision received date**: 2011-06-30

E-mail: tracy403@nuaa.edu.cn

specific encoding methods. There are two common drawbacks for Golomb codes and FDR codes: one is that they only concentrate on the runs of 0s without consideration for runs of 1s. The other is that the coding sources are differential test vectors. The differential signal requires cyclical scan register (CSR) module for the on-chip decoder, which increases the hardware overhead.

Many new coding schemes are proposed in related compression codes research, including alternate variable length codes[5], the variable-length input Huffman coding[6] and selected variable-length input coding (SVIC)[7] based on statistical characteristic analysis, the 9-C code[8] and modified frequency-directed run-length (MFDR) codes combining the statistical properties and characteristics of FDR codes[9], etc. Among them, coding schemes based on the statistical characteristic analysis[6-7] have higher compression ratio, while the encoding and decoding processes are relatively complex and the hardware overhead is even larger than Golomb and FDR codes.

A novel compression codes called advanced frequency-directed run-length (AFDR) codes is proposed with relatively lower hardware overhead and higher compression efficiency by improving FDR coding manner. AFDR codes considers both the runs of 0s and 1s simultaneously, and optimizes the codes for 00 and 11 to further improve the compression efficiency. Furthermore, its decompression circuit is simpler than FDR codes without need for CSR circuit.

1 AFDR CODES

The proposed AFDR codes is the improved coding scheme based on FDR. FDR only encodes the consecutive 0-sequence, the sequence which ends with 1 such as 00001 is encoded according to the length of 0-run. And shorter 0-runs are mapped shorter codewords, while single 1 is regarded as the sequence whose run-length is zero. AFDR encodes both the 0-runs and 1-runs with almost the same codeword as FDR codes and also applies the shorter codewords to the shorter runs.

Fig. 1 shows the distribution of the 0- and 1-runs for test vectors of s9234 circuit. The s9234 circuit is a typical sequence circuit and one of the largest ISCAS benchmark circuits. It is obvious that the shorter runs occur more frequently, so mapping them to short codewords will increase the compression ratio greatly. This is also the reason for the primary encoding manner of AFDR and FDR codes.

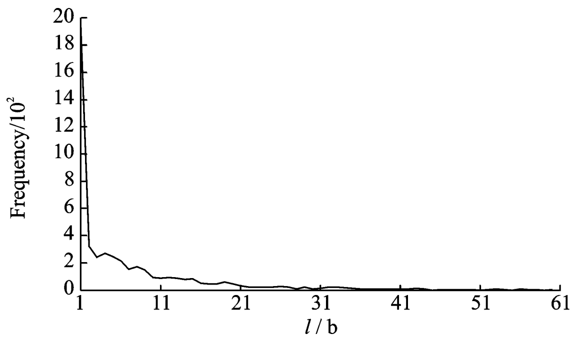


Fig. 1 Distribution of run-length for circuit s9234

The AFDR codes is constructed as follows: the 0- and 1-runs are divided into groups A_1, A_2, A_3, …, A_k, where k is determined by the maximum length $l_{\max}$ ($2^k-2\leqslant l_{\max}\leqslant 2^{k+1}-3$). A run with the length l is mapped to group A_j based on

$$j = \lceil \log_2(l+3)-1 \rceil \tag{1}$$

Each codeword of AFDR consists of two parts with same length—a group prefix and a tail. The group prefix is used to identify the group which the run belongs to. For example, the run length is 6, then $j=\lceil \log_2(6+3)-1 \rceil=3$, so it belongs to group A_3. The tail is used to identify the members within the group. The size of codeword increases by 2 b (1 b for the prefix and 1 b for the tail) as the run's length moves from group A_j to group A_{j+1}.

The AFDR encoding procedure is shown in Table 1. FDR codes has two codewords in group A_1: 0 b and 1 b run legnth respectively, while AFDR coding has only one codeword for the 1 b length of runs (01 or 10) for AFDR handles 0 and 1 strings simultaneously. However, AFDR codes assumes 0-runs and 1-runs appear alternately. When they become consecutive, additional codeword, 00, is chosen as the separator.

The 2 b length in Table 1 indicates the special adoption to improve the compression efficiency. According to the original coding rules, the codeword for 0- and 1-runs with 2 b length (001 or 110) is 1000, obviously wider than the original code, thus impacting on compression ratio. Furthermore, it can be proved that the quantity of 2 b long runs is quite considerable. Fig. 2 shows the run-length distributions of standard circuit s9234, s13207, s38417 and s38584. The numbers of 2 b long runs is larger than that of other longer ones. Therefore, reducing the 2 b long codeword will influence the efficiency of compression. AFDR coding applies 000 for 2 b long runs instead of original 1000.

Table 1 Application of AFDR coding

Group	Run length	Prefix	Tail	Codeword
A_1	1	0	1	01
A_2	2 *	10	00	1000(000 *)
	3		01	1001
	4		10	1010
	5		11	1011
A_3	6	110	000	110000
	7		001	110001
	8		010	110010
	9		011	110011
	10		100	110100
	11		101	110101
	12		110	110110
	13		111	110111
⋮	⋮	⋮	⋮	⋮

* —2 b long run

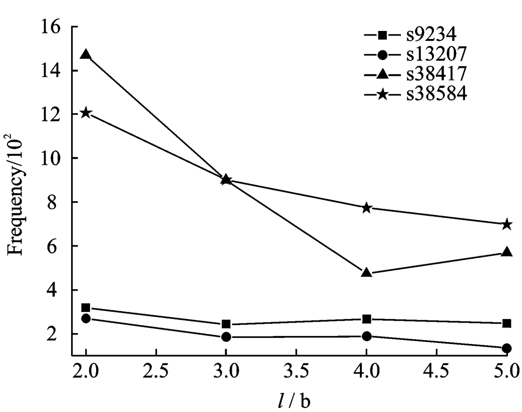


Fig. 2 Comparison of length frequency of runs for ISCAS benchmark circuits

An example is presented to illustrate AFDR coding realization and compared with FDR coding. Assume that there is a 32 b test sequence T_D= { 00000111011111000000000000011111 }. Apply FDR coding and the encoded sequence T_{FDR} = {1011000001000000001101110000000}, 32 b long. While apply AFDR coding, the corresponding sequence T_{AFDR}={10110000010111101101010} is 23 b long. It is obvious that the compression effectiveness of AFDR is better than FDR coding. Fig. 3 shows the encoding procedure of AFDR coding.

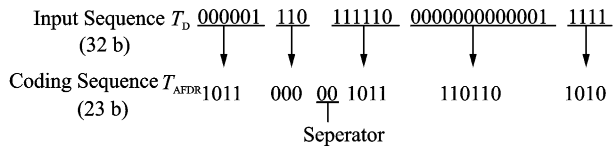


Fig. 3 AFDR encoding procedure of test sequence T_D

2 ANALYSIS ON AFDR CODES

The probability of 0 is defined as p and the probability of 1 as $(1-p)$. $H(p)$ is the entropy indicating the value of information. As far as data compression is concerned, the entropy is the amount of information required for encoding which is relevant to the theory limit of compression ratio. $H(p)$ of the test vector is proposed by the following equation[5]

$$H(p) = - p\log_2 p - (1-p)\log_2(1-p) \quad (2)$$

The upper limit of the compression gain β_{max} is obtained by

$$\beta_{max} = \frac{1}{H(p)} \quad (3)$$

For the run-length coding methods, the compression gain β is defined as

$$\beta = \frac{\lambda}{L_{avg}} \quad (4)$$

where λ is the average number of bits in any run generated by the data source and L_{avg} the average codeword length.

According to the AFDR coding table, λ is defined as

$$\lambda = 1 + \sum_{i=1}^{\infty} i(p^i(1-p) + (1-p)^i p) = \frac{p^2 - p + 1}{p(1-p)} \quad (5)$$

The run lengths within the limit $2^k-2\leqslant l\leqslant 2^{k+1}-3$ belong to group A_k, so the probability $P(i,k)$ of an arbitrarily chosen run with the length i within group A_k is given as

$$P(i,k)=\sum_{i=2^k-2}^{2^{k+1}-3}(p^i(1-p)+p(1-p)^i)= p^{2^k-2}(1-p^{2^k})+(1-p)^{2^k-2}\cdot (1-(1-p)^{2^k}) \tag{6}$$

Codeword in group A_k consists of 2 kb, and the average codeword length L_{avg} is given as

$$L_{avg}=\sum_{k=1}^{\infty}2kP(i,k)=2\sum_{k=1}^{\infty}(p^{2^k-2}+(1-p)^{2^k-2}) \tag{7}$$

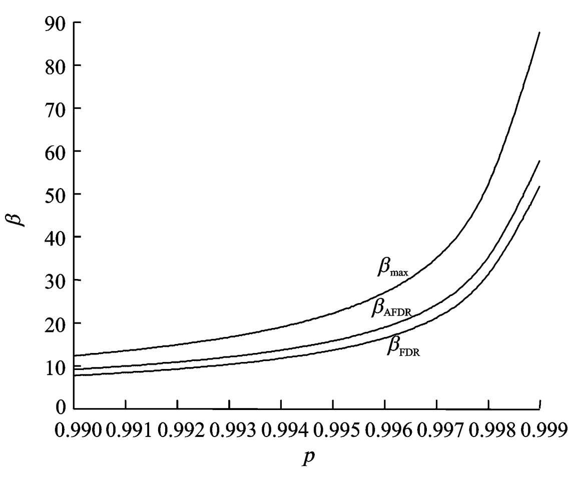
Therefore, the compression gain β_{AFDR} of AFDR coding is given as

$$\beta_{AFDR}=\frac{\lambda}{L_{avg}}=\frac{p^2-p+1}{2p(1-p)\sum_{k=1}^{\infty}(p^{2^k-2}+(1-p)^{2^k-2})} \tag{8}$$

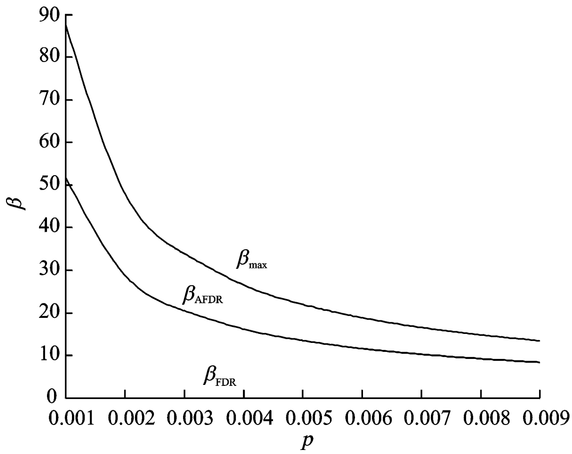
While the compression gain β_{FDR} is expressed as follows[5]

$$\beta_{FDR}=\frac{1}{2(1-p)\sum_{k=1}^{\infty}p^{2^k-2}} \tag{9}$$

Fig. 4 shows the comparison of β_{AFDR}, β_{FDR} and β_{max} with different probability distributions. It is concluded that the compression gain of AFDR is superior to that of the FDR. The compression gain of AFDR is close to the upper bound when $0.990\leqslant p\leqslant 0.999$ as well as $0.001\leqslant p\leqslant 0.009$, while that of FDR stays at 0.5.



(a) $0.990\leqslant p\leqslant 0.999$



(b) $0.001\leqslant p\leqslant 0.009$

Fig. 4 Comparison of compression gain between AFDR, FDR and uper limit

3 DATA PREPROCESSING AND TEST DATA DECOMPRESSION

Testing vectors often contain a large number of unspecified bits (x) to be determined as 0 or 1 before being compressed. How to fill the unspecified bits will affect the run-length distribution and the maximum compression ratio. The optimization of data preprocessing is essential to test vectors compression. The schemes for data preprocessing belong to non-deterministic polynomial complete (NPC) problems and a compatible preprocessing method for AFDR codes is adopted to balance the process complexity and related compression effect. The realization process is as follows:

(1) For vectors as $0\cdots0x\cdots x0\cdots0$ or $1\cdots1x\cdots x1\cdots1$, the unspecified bits are filled with their adjacent bits (0 or 1).

(2) For $0\cdots0x\cdots x1\cdots1$ or $1\cdots1x\cdots x0\cdots0$, name the first char of the vector as the previous char and the end char as the next char, then fill all the x with the previous char if the run length of the char string does not exceed the maximum length of its group, otherwise fill the x with the previous char until the run length is the maximum length and fill the remaining x with the next

char.

The decoder of AFDR is composed of a finite state machine (FSM), a shift counter (sf_counter), a read counter (rd_counter) and a T flip-flop. Fig. 5 depicts the structure of this decoder. The signal b_in is the input of FSM and en is sent to the input data to show the decoder is ready. The signal out is the output of FSM and transmitted through the T flip-flop to generate final decompressed signal f_out. va indicates that the output is valid. c_in is the signal sent from FSM to shift the prefix or tail into the sf_counter, the signals shift and, dec1 are respectively applied to shift the data in and to decrease the number of sf_counter. inc and dec2 are used to increase and decrease the number of rd_counter, respectively. rs1 and rs2 indicate the reset states of the corresponding counter.

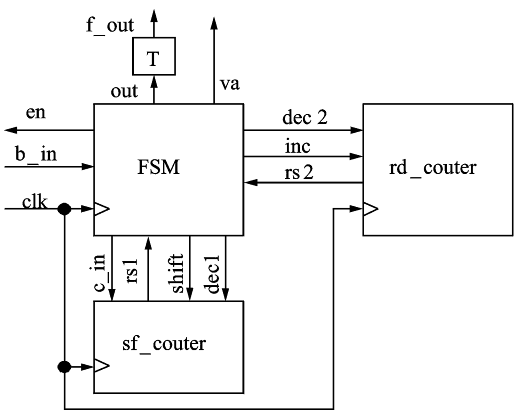


Fig. 5 Block diagram of decoder for AFDR

The operation process of the decoder is as follows:

Step 1 FSM uses b_in to send the group prefix with the end of 0 to sf_counter. en, shift and inc are set high in this period.

Step 2 out remains low to keep T flip-flop outputting the previous state, meawhile va is high to show the output is effective. dec1 is high. sf_counter continues decreasing until rs1 is set high (the number of sf_counter is 0).

Step 3 The tail is sent to sf_counter from b_in and the length is counted by rd_counter. dec2 decreases rd_counter until rs2 is set high (the input of tail ends).

Step 4 out is high until sf_counter is 0 and then trigger T flip-flop changes state.

If the group prefix is 0 (Step 1), then en, shift and inc are high. The process of the next clock cycle is as follows:

(1) If b_in is 1 (the run length is 1 b), dec1, dec2 are high and the out is low simultaneously while va gains high level to make T flip-flop to output the previous state, then moves to Step 4.

(2) If b_in is 0, en, shift and inc are high and acquire the next b_in. If the new b_in is 0 (the run length is 2 b), let T flip-flop to output the pervious state twice, otherwise (the separate codes) dec2 and out are high and va is low to maintain the state of T flip-flop and prepares for the next codeword.

The state diagram of FSM for decompression is shown in Fig. 6. S_0—S_5 are the states for decompressing the strings with run-length longer than 2 b, while S_6—S_9 are for decompressing the strings with run-length of 1 b or 2 b length and the separator. FSM is synthesized using Synopsys design compiler and the logic synthesized circuit containing only 4 flip-flops and 43 gates. It is obvious that the additional hardware overhead that the decompression circuit requires is very small.

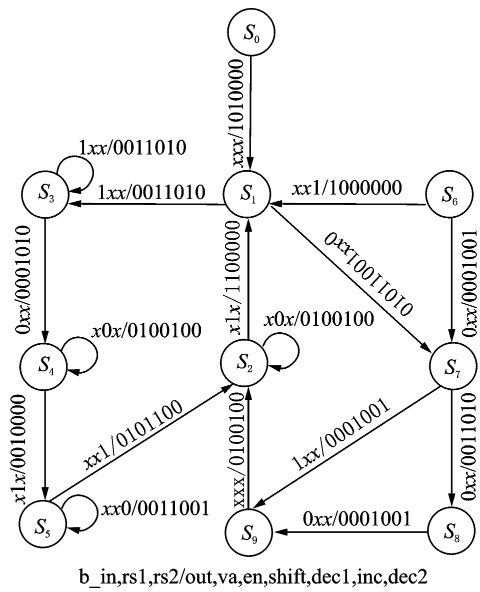


Fig. 6 State diagram of FSM for decompression circuit

4 EXPERIMENTAL RESULTS

The experimental results of the test vector compression are presented for some large-scale circuits in the ISCAS89 Benchmark. The original test vectors are generated by Mintest[10] ATPG tool from Duke University. Table 2 shows the compression ratios of AFDR coding and other schemes in Refs. [3,4,7,9]. The average (AVG) compression ratio is given in the last line.

The compression ratio is computed as follows

$$r = \frac{S_{T_D} - S_{T_E}}{S_{T_D}} \times 100\% \tag{10}$$

where S_{T_D} is the size of the source test set T_D and S_{T_E} the size of the encoded test set T_E.

Table 2 Compression ratios of different schemes

Circuit	S_{T_D}	Compression ratio/%				
		Golomb[3]	FDR[4]	MFDR[9]	SVIC[7]	AFDR
S9234	39 273	43.34	44.88	57.74	60.83	47.71
S13207	165 200	74.78	78.67	83.42	82.21	81.76
S15850	76 986	47.11	52.87	66.93	65.84	67.5
S35932	28 208		10.19	10.27		80.7
S38417	164 736	44.12	54.53	57.95	57.82	61.94
S38584	199 104	47.71	52.85	59.32	59.52	63.32
AVG		51.41	49.00	55.94	65.24	67.16

Concluded from Table 2, the compression ratios of AFDR coding are higher than those of other coding schemes except MFDR and SVIC codes of s9234 and s13207. Considering both 0- and 1-runs, AFDR codes has better compression effectiveness for most of the benchmark circuits by modifying the codewords of 00 and 11 strings. Because of the statistic characteristic of test vectors, MFDR and SVIC coding have higher compression ratios for some circuits with specific statistical distribution such as s9234 and s13207 circuits. Moreover, AFDR is obviously superior to other coding schemes on the average compression ratio.

5 CONCLUSION

An effective compression scheme for test vector of system-on-chip is proposed as AFDR coding. The AFDR coding improves the FDR codes by considering the 0- runs and 1-runs simultaneously. The CSR circuit is never needed to decrease the hardware and encoding time consumption. The compression effectiveness is improved by the optimization of specific run lengths. The probabilistic analysis of AFDR codes is also presented to demonstrate the intrinsic superiority. The experimental results on ISCAS89 benchmark circuits validate the compression effectiveness of AFDR coding. The following research will focus on adoption of other preprocessing methods for higher compression ratio and the construction of practical tester with AFDR coding.

References:

[1] Semiconductor industry association. International technology roadmap for semiconductors 2009 edn [EB/OL]. http://www.itrs.net/Links/2009ITRS/Home2009.htm,2005-01-10/2011-05-23.

[2] Mehta U, Dasgupta K S, Devashrayee N M. Survey of test data compression technique emphasizing code based schemes [C]// 12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools, DSD′09. Patras, Greece: IEEE, 2009:617-620.

[3] Chandra A, Chakrabarty K. System-on-a-chip test data compression and decompression architectures based on Golomb codes [J]. IEEE Transaction Computer-Aided Design, 2001,20:355-368.

[4] Chandra A, Chakrabarty K. Frequency-directed Run-length (FDR) codes with application to System-on-a-chip test data compression [C]// 19th IEEE VLST Test Symposium. Marina Del Reg, USA: IEEE, 2001:42-47.

[5] Chandra A, Chakrabarty K. A unified approach to reduce SoC test data volume, scan power and testing time [J]. IEEE Transaction on Computer-aided Design of Integrated Circuits and system, 2003,22(3):352-363.

[6] Gonciari P T, Al-Hashimi B M, Nicolici N. Variable-length input Huffman coding for system-on-a-chip test [J]. IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, 2003, 22(6):783-796.

[7] Hu B, Chen G, Xie Y. System on chip test data compression based on SVIC coding [J]. Journal of Electronic Measurement and Instrument, 2006, 20 (1): 73-78. (in Chinese)

[8] Tehranipoor M, Nourani M, Chakrabarty K. Nine-coded compression technique for testing embedded cores in SoCs [J]. IEEE Transaction on Verg Large Scale Integration (VLSI) Systems, 2005, 13(6): 719-731.

[9] Feng J, Li G. A test data compression method for system-on-a-chip electronic design [C]// 4th IEEE International Symposium on Test and Applications DELTA 2008. Hong Kong, China: IEEE, 2008: 270-273.

[10] Hamzaoglu I, Patel J H. Test set compaction algorithms for combinational circuits [C]// Procssing of IEEE/ACM International Conference on CAD. San Jose, USA: IEEE, 1998: 283-289.

片上系统测试数据压缩的优化型FDR编码机制

张 颖 吴 宁 葛 芬

（南京航空航天大学电子信息工程学院，南京，210016，中国）

摘要：测试数据压缩是片上系统(System-on-chip，SoC)测试中的关键问题之一，用于有效地减少测试数据总量。本文提出了一种新颖的变长-变长压缩编码，称为AFDR(Advanced frequency-directed run-length)编码。它同时对0游程和1游程进行编码，并对等长游程赋以相同的编码，优化了仅仅考虑0游程的FDR(Frequency-directed run-length)码。此外，对游程长度为2的数据(即00和11)进行特殊处理，进一步地提高了压缩比。ISCAS 89标准电路下的实验结果表明，AFDR编码的压缩效果明显优于FDR编码以及同类型的其他编码。

关键词：测试数据压缩；FDR编码；测试源划分；片上系统

中图分类号：TP302

(Excecutive editor: Zhang Bei)

基金项目：国家自然科学基金(61076019，61106018)资助项目；航空科学基金(20115552031)资助项目；中国博士后科学基金(20100481134)资助项目；江苏省科技支撑计划(BE2010003)资助项目；南京航空航天大学基本科研业务经费(NS2010115)资助项目；南京航空航天大学引进人才科研启动基金(1004-YAH10027)资助项目。