SOC TEST DATA VOLUME MINIMIZATION WITH LOW POWER CONSTRAINT

Thesis report submitted in the partial fulfillment of the

Requirement for the award of the degree of

MASTER OF TECHNOLOGY

IN

VLSI DESIGN & CAD

Submitted by

Aditya Chopra

Roll No.: 600961001

Under the Guidance of

Ms. Harpreet Vohra

Assistant Professor, ECED

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

THAPAR UNIVERSITY, PATIALA (PUNJAB) - 147004

JULY - 2011

CERTIFICATE

I hereby declare that the work which is being presented in the thesis entitled," **SOC TEST DATA VOLUME MINIMIZATION WITH LOW POWER CONSTRAINT"** in partial fulfillment of the requirement for the award of degree of M.Tech. (VLSI Design & CAD) at Electronics and Communication Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Ms. Harpreet Vohra, Assistant Professor, ECED.

The matter presented in this thesis has not been submitted in any other University/Institute for the award of my degree.

Date: 5/7/2011

Aditya Chopra
Roll.No.600961001

It is certified that the above statement made by the student is correct to the best of my knowledge and belief.

Ms.Harpreet Vohra
Assistant Professor
ECED,Thapar University

Counter signed by:

Dr. A. K. Chatterjee
Professor & Head
ECED, Thapar University
Patiala-147004

Dr. S. K. Mohapatra
Dean of Academic Affairs
Thapar University
Patiala-147004

ACKNOWLEDGEMENT

I take this opportunity to express my profound sense of gratitude and respect to all those who helped me through the duration of this thesis. I would have never succeeded in completing my task without the cooperation, encouragement and help provided to me by various people. Words are often too less to reveals one's deep regards. I acknowledge with gratitude and humility my indebtedness to **Ms. Harpreet Vohra, Assistant Professor,** Electronics and Communication Engineering Department, Thapar University, Patiala, under whose guidance I had the privilege to complete this thesis. I wish to express my deep gratitude towards her for providing individual guidance and support throughout the thesis work.

I convey my sincere thanks to **HEAD OF THE DEPARTMENT, Dr. A. K. Chatterjee** as well as **PG Coordinator, Dr. Alpana Agrawal, Associate Professor,** Electronics and Communication Engineering Department, entire faculty and staff of Electronics and Communication Engineering Department for their encouragement and cooperation.

My greatest thanks are to all who wished me success especially my parents and other family members and friends without whom I would not have been able to complete my thesis work.
I thank and owe my deepest regards to all of them and all others who have helped me directly or indirectly.

Aditya Chopra

ABSTRACT

Test Data Compression is one of the useful techniques for reducing the volume of test data for System-On-Chip (SOC) testing. In this thesis i propose a multi code compression to compress test data of SOC's. The multi code compression scheme is based on different compression techniques (e.g. Golomb, Frequency-directed run-length (FDR), Alternating run-length (AR), Extended Frequency-directed run-length (EFDR) and IFDR). To achieve higher compression ratio, i have made an attempt to combine the best properties of all the above techniques so as to utilize all the properties in a single multi code compression scheme.

Considering the large amount of power required in testing i have also proposed a Double Hamming Distance Reordering scheme. Analysis of different Don't Care filling techniques along with above scheme has been made and then using Difference vector technique along with the various don't care filling technique has been made so as to increase compression ratio and reduce power simultaneously has also been discussed.

We have also compared test vectors generated from two ATPG engine's i.e. TETRAMAX(SYNOPSYS) and MILEF and have concluded that MTFILLDRDIFF technique in most cases is best for saving Peak and Average power by filling up the don't care bits in the uncompacted test set for various ISCAS89 circuits.

Experimental results show that the proposed Multi Code Compression Scheme along with double reordered scheme and difference vector can achieve about 40-200% increment in Compression Ratio and a 30-50% decrease in average and peak power with most of the ISCAS89 Benchmark circuits compared with a single code compression scheme using original Test data generated for both the ATPG engines namely TETRAMAX and MILEF.

Finally an Decoder design for Multi Code Compression Scheme has also been implemented using (tools) which can work for 6 different schemes with an area overhead of 5262.163086um^2 involving 37 Flip Flops.

TABLE OF CONTENTS

CERTIFICATE………………………………………………………..…….. i

ACKNOWLEDGEMENT………………………...……..…………… ii

ABSTRACT……………………………………...………….……… iii

LIST OF FIGURES…………………………………………….….viii

LIST OF TABLES………………………………………..……....….x

ABBREVIATIONS……………………………………................... xi

CHAPTER 1 INTRODUCTION…………………………….………… 1

1.1 MOTIVATION…………………………………………………..………. 1

1.2 TESTING METHODOLOGIES..…………………………………..…………. 3

1.2.1 SCAN BASED TESTING…………………….……………………..…..4

1.2.2 MAKING FLIP FLOPS DIRECTLY CONTROLLABLE FROM PRIMARY INPUTS……………………..………………………..……..5

1.3 THESIS ORGANIZATION…………………..……………………………...6

CHAPTER 2 LITERATURE REVIEW……….………………………… 7

2.1 WEIGHTED TRANSITIONS METRIC MODEL………..………………………7

2.2 DON'T CARE FILLING TECHNIQUES…………………..……….………...9

2.2.1 RANDOM FILL(R-FILL)……………………..………..………9

2.2.2 MINIMUM TRANSITION FILL (MTFILL)……………………….........9

2.2.3 ZERO FILL…………………………………………………..............…10

2.2.4 COLUMN WISE BIT STUFFING (CBS)……………………………...…10

2.3 HAMMING DISTANCE BASED REORDERING…………………………...…11

2.3.1 INTRODUCTION……………………………………….................11

2.3.2 HAMMING DISTANCE REORDERING……………….……..…..12

2.3.3 HAMMING DISTANCE BASED REORDER SCHEME FOR DON'T CARE SET……………………………………………………………….…...14

2.3.3.1 SELECTION OF FIRST TEST PATTERN OF REORDERED TEST SET…………………………………………………….……15

2.2.3.2 REORDERING OF REMAINING PATTERNS………………16

2.4 TEST DATA COMPRESSION………………………………………...….17

2.4.1 RUN-LENGTH CODING……………………………………............18

2.4.1.1 GOLOMB CODE…………………………………………...19

2.4.1.2 FREQUENCY DIRECTED RUN-LENGTH (FDR) CODE……………………………………………………..…...20

2.4.1.3 EXTENDED FREQUENCY DIRECTED RUN-LENGTH (EFDR) CODE……..…………………………………………...……...23

2.4.1.4 ALTERNATIVE FREQUENCY DIRECTED RUN-LENGTH (AFDR) CODE……………...……………………………..……..…24

2.5 CYCLICAL SCAN REGISTER DECOMPRESSION ARCHITECTURE………26

CHAPTER 3 DESIGN AND IMPLEMENTATION OF MCC ENCODER AND DECODER WITH DOUBLE REORDERING SCHEME……………………………………………………………..…..…29

3.1 DOUBLE HAMMING DISTANCE BASED REORDER SCHEME…………..29

3.2 COMPRESSIONS ANALYSIS………………………………………………30

3.3 ANALYSIS OF MULTI CODE COMPRESSION…………………………...31

3.4 MULTICODE COMPRESSION……………………………………...…35

3.5 TEST DATA DECOMPRESSION (DECODER)……………………..........…36

3.5.1 DECOMPRESSION IIP……………………………….........................36

3.6 TOOLS USED…………………………………………………….……..43

CHAPTER 4 RESULTS AND ANALYSIS………………………….........44

4.1 ENCODER RESULTS AND ANALYSIS…………………………….……...44

4.1.1 MATLAB RESULTS AND ANALYSIS FOR MILEF ATPG…..........45

4.1.1.1 Circuit s298……………………………..…………..…45

4.1.1.2 Circuit s400…………………………………….........47

4.1.1.3 Circuit s1196……………………………………….....49

4.1.1.4 Circuit s1494……………………………………………51

4.1.2 DESIGN VISION SYNTHESIZED AND DFT INSERTED CIRCUITS…………………………………………………………...53

4.1.3 MATLAB RESULTS AND ANALYSIS FOR TETRAMAX DATA……………………………………………………..……....54

4.1.3.1 Circuit s298.t………………………………………….…..54

4.1.3.2 Circuit s400t……………………………………………………….56

4.1.3.3 Circuit s1196t……………………………………………………...58

4.1.3.4 Circuit s1494t……………………………………………………..60

4.2 DECOMPRESSION DECODER RESULTS…………………………………....61

4.2.1 DESIGN VISION RESULTS…………………………………………….62

4.2.2 XILNIX RESULTS………………………………………………………66

CHAPTER 5 CONCLUSION AND FUTURE WORK…………………....69

REFERENCES………………………………………………………..........70

LIST OF FIGURES

Figure No.	Title of Figure	Page No
Figure 1.1	Test data transportation and an SOC testing mechanism	2
Figure 1.2	Formation of Scan Chain	4
Figure 1.3	Conversion of Flip Flop to Scan Flip Flop	5
Figure1.4	A master slave scan Flip Flop	5
Figure 1.5	Circuit utilizing Testing Logic	6
Figure 2.1	Conceptual architecture for the SOC testing with using on chip decompression	17
Figure 2.2	Three bit variable to block encoding	18
Figure 2.3	An example of run length encoding	18
Figure 2.4	Block Diagram of GOLOMB Decoder	19
Figure 2.5	Block diagram of FDR decoder	22
Figure 2.6	Block Diagram of EFDR	24
Figure 2.7	Compare of FDR coding and AR coding	25
Figure 2.8	Block diagram of AR decoder	26
Figure 2.9	The architecture of CSR decompression	27
Figure 2.10	The decompression architecture of using boundary scan register to replace CSR	27
Figure 2.11	The decompression architecture for using internal scan chain and user defined scan element to replace CSR	28
Figure 3.1	Double Hamming Distance Based Reordering Scheme	29
Figure 3.2	Compared on number of codeword in each run length Encoding	30
Figure 3.3	An example for weight of one test vector	34
Figure 3.4	Structural drawing of decompression IIP	37
Figure 3.5	Block diagram of decompression IIP	40
Figure 3.6	State diagram of the FSM	42
Figure 4.1	Summary of s298 results	46

Figure 4.2	Summary of s400 results	48
Figure 4.3	Summary of s1196 results	50
Figure 4.4	Summary of s1494 results	52
Figure 4.5	A simple D Flip flop	53
Figure 4.6	A DFT inserted scan Flip flop	53
Figure 4.7	Summary of s298t results	55
Figure 4.8	Summary of s400t results	57
Figure 4.9	Summary of s1196t results	59
Figure 4.10	Summary of s1494t results	61
Figure 4.11	RTL for MMC FSM used in MCC Decoder	62
Figure 4.12	RTL for Synthesized MMC FSM used in MCC Decoder	62
Figure 4.13	Symbol for MMC FSM used in MCC Decoder	63
Figure 4.14	Symbol for MCC Decoder	64
Figure 4.15	RTL for MCC Decoder	65
Figure 4.16	Synthesized RTL for MCC Decoder	65
Figure 4.17	Symbol for MMC FSM used in MCC Decoder	66
Figure 4.18	Synthesized MMC FSM used in MCC Decoder in XILNIX	66
Figure 4.19	Symbol for MCC Decoder in XILNIX	67
Figure 4.20	Synthesized RTL for MCC Decoder in XILNIX	67

LIST OF TABLES

Table No.	Title of Table	Page No
Table 2.1	Test vectors for full adder circuit	13
Table 2.2	Example of Golomb coding with m= 4	20
Table 2.3	An example for FDR code	21
Table 2.4	An example for EFDR code	23
Table 2.5	An example for AR coding	25
Table 3.1	Compared on lengths of codeword in each run length encoding	32
Table 3.2	Compared on weight on each run-length encoding	33
Table 3.3	A Multi Code Compression scheme	36
Table 3.4	The truth table of control signals of decompression IIP.	38
Table 3.5	The truth table of control signals of 4-16 decoder	41
Table 4.1	Results for ISCAS 89 s298 circuit for MILEF ATPG patterns	45
Table 4.2	Results for ISCAS 89 s400 circuit for MILEF ATPG patterns	47
Table 4.3	Results for ISCAS 89 s1196 circuit for MILEF ATPG patterns	49
Table 4.4	Results for ISCAS 89 s1494 circuit for MILEF ATPG patterns	51
Table 4.5	Results for ISCAS 89 s298t circuit for TETRAMAX patterns	54
Table 4.6	Results for ISCAS 89 s400t circuit for TETRAMAX patterns	56
Table 4.7	Results for ISCAS 89 s1196t circuit for TETRAMAX patterns	58
Table 7.8	Results for ISCAS 89 s1494t circuit for TETRAMAX patterns	60

ABBREVIATIONS

VLSI	Very Large Scale Integration
CMOS	Complementary Metal Oxide Semiconductor
SOC	System On Chip
IP	Intellectual property
ATE	Automatic Test Equipment
I/O	Input/output
TAM	Test Access Mechanism
BW	Burrows wheeler
FSM	Finite State Machine
HC	Huffman Code
FDR	Frequency Directed Run-Length
EFDR	Extended Frequency Directed Run-Length
AFDR	Alternate Frequency Directed Run-Length
IFDR	Inverse Frequency Directed Run-Length
MCC	Multi Compression Code
DR	Double Hamming Distance based Reorder scheme
WTM	Weighted Transition Metric
DFF	D-Flip Flop
SD	Scan Data
TC	Test Control
ATPG	Automatic Test Pattern Generator
P_{avg}	Average Power
P_{peak}	Peak Power
CUT	Circuit under Test
DUT	Design under Test
DFT	Design for Testability
RFILL	Random Fill
MTFILL	Minimum Transition Filling

CBS	Column Wise Bit Stuffing
THD	Total Hamming Distance
CSR	Cyclical Scan Register
TD	Test Data
CBSTD	Column wise Bit Stuffing for Test Data
CBSTDDIFF	Column wise Bit Stuffing for Test Data with Difference vector
CBSTDDR	Column wise Bit Stuffing for Test Data with Double reordered scheme
CBSTDDRDIFF	Column wise Bit Stuffing for Test Data with Double reordered scheme and difference vector
CBS	Column wise Bit Stuffing Hamming distance reordered scheme
CBSDIFF	Column wise Bit Stuffing Hamming distance reordered scheme with difference vector
CBSDR	Column wise Bit Stuffing with double reordered scheme
CBSDRDIFF	Column wise Bit Stuffing with double reordered scheme and difference vector.
MTFILLTD	Minimum transition filling for Test Data
MTFILLTDDIFF	Minimum transition filling for Test Data with Difference vector
MTFILLTDDR	Minimum transition filling for Test Data with Double reordered scheme
MTFILLTDDRDIFF	Minimum transition filling for Test Data with Double reordered scheme and difference vector
MTFILL	Minimum transition filling Hamming distance reordered scheme
MTFILLDIFF	Minimum transition filling Hamming distance reordered scheme with difference vector
MTFILLDR	Minimum transition filling with double reordered scheme
MTFILLDRDIFF	Minimum transition filling with double reordered scheme and difference vector
ZEROFILLTD	Zero filling for Test Data
ZEROFILLTDDIFF	Zero filling for Test Data with Difference vector
ZEROFILLTDDR	Zero filling for Test Data with Double reordered scheme

ZEROFILLTDDRDIFF	Zero filling for Test Data with Double reordered scheme and difference vector
ZEROFILL	Zero filling Hamming distance reordered scheme
ZEROFILLDIFF	Zero filling Hamming distance reordered scheme with difference vector
ZEROFILLDR	Zero filling with double reordered scheme
ZEROFILLDRDIFF	Zero filling with double reordered scheme and difference vector

CHAPTER 1
INTRODUCTION

1.1 Motivation

With the advent of the CMOS process, core-based design methodology is widely used to design a system-on-chip (SOC). However, SOCs present many test challenges, such as huge test data, test power, etc [1, 2]. A typical SOC consists of different types of intellectual property (IP) cores. Due to the growth of complexity in an SOC, the volume of test data increases quickly. A huge amount of test data not only consumes the memory and I/O channel bandwidth of automatic test equipment (ATEs) but also increases testing time.

The testing time of an SOC depends on the test-data volume, the time required to transfer the data to the cores, the rate at which the test data is transferred (measured by the cores test-data bandwidth and ATE channel capacity), and the maximum scan chain length. The total test time can be reduced by either reducing the test-data volume or by shortening and reorganizing the scan chains. While test-data volume reduction techniques can be applied to both hard and soft cores, scan chains cannot be modified in hard cores. Lower testing time will increase production capacity as well as reduce test cost and time-to-market for SOCs. Therefore, new techniques are needed for decreasing test-data volume in order to overcome memory bottlenecks and to reduce testing time.

Built-in self test (BIST) has emerged as a useful approach for alleviating the above problems [29]. BIST reduces dependencies on expensive ATEs and it allows precomputed test sets to be embedded in test sequences generated by BIST hardware. However, BIST can be applied directly to SOC designs only if the embedded cores are BIST-ready. Since most IP cores that are currently available from core vendors are not BIST ready, considerable redesign is necessary for incorporating BIST. This increases time-to-market and, therefore, defeats the very purpose of using IP cores.

Test-data compression offers a promising solution to the problem of reducing the test-data volume for SOCs, especially if the IP cores in the system are not BIST-ready [30, 31]. In this approach, a precomputed test set for an IP core is compressed (encoded) to a much smaller test set, which is stored in ATE memory. An on-chip decoder is used for pattern decompression to obtain T_D from T_E during testing.

The testing time of an SOC is related to the test data volume, the timing requirement of transferring the test data to the core, the ATE channel bandwidth, and the length of maximum scan chain in the chip. As Figure 1.1 shows, the test patterns of each core in an SOC are transferred by ATE I/O channel and Test Access Mechanism (TAM). However, the bandwidth and speed of the I/O channel and the ATE memory are limited and those limitations actually increase the testing time. Efficient test data reduction techniques can simultaneously decrease testing time and ATE memory requirement.

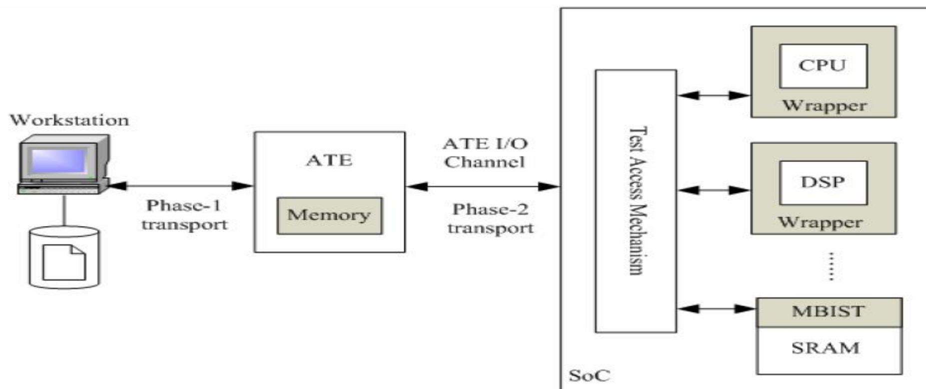


Figure 1.1: Test data transportation and an SOC testing mechanism [1]

Test data transportation usually includes two phases [1], as shown in Figure 1-1. The Phase-1 transports test patterns from workstation to the ATE. The Phase-2 transport is between the ATE and the chip under test. The compression technique for Phase-1 can be possibly complex, but decompression for Phase-2 must be simple [9]. Reducing the time needed to download test pattern from workstation to ATE is proposed in [10]. This technique applies a combination of Burrows-Wheeler (BW) transformation and run-length coding. Both the encoding and decoding algorithm are totally practiced in software. The BW decoder is too complex to implement in on-chip hardware. Test data reduction for Phase-2 transport can be achieved by test compaction [4-5] and compression methods [13-22]. Test data compression techniques can roughly be divided into statistical coding and run-length coding.

These methods will consume lower circuit area to achieve the purpose of test data reduction in single scan chain and it can extend to multiple scan chain by using virtual circuit to broadcast test vector to multiple cores [28]. In [13], the input test data is analyzed and compressed by Huffman code (HC). The problem of more area overhead in decoder was solved

via scaling FSM states down. Some HC-like coding techniques also have been described in [14-15]. In [19], a Golomb code was proposed to encode the difference test patterns, which shows higher compression ratio can be obtained for the test patterns with longer 0's rum length. A Frequency-Directed Run-length (FDR) coding was proposed in [20]. It gets higher compression ratio by reducing the number of codeword in shorter 0's run length. In [21], an Extended Frequency-Directed Run-length (EFDR) was proposed to improve the disadvantage of FDR. It adds one additional bit in the codeword to code the 1's run length for boosting the compression ratio. In [22], an Alternating Run-length (AR) was proposed to compress test data. It can code the input test data without increasing the length of codeword by alternating the 0's and 1's run lengths. This method not only can increase compression ratio but also can decrease power consumption of scan cell in test mode. However, there is not a compression technique can efficiently compress the test data of all cores in an SOC.

In this Thesis a Multi Code Compression Scheme is presented. It can achieve very good compression ratio as it combines the properties of various run length codes in a single compression scheme.

A Double Hamming distance based reorder scheme has also been proposed which reduces the average and peak testing power required for testing the circuit. Which when incorporated with difference vector and don't care filling techniques reduces testing power as well as increases the compression ratio when applied with our proposed Multi Code Compression (MCC) schemes.

A decompression circuitry for the proposed MCC scheme is also proposed. Experimental Results are shown in the section 4.1 Furthermore, Decoder for Decompression has been synthesized using both DESIGN VISION (SYNOPSYS) and XILNIX.

1.2 TESTING METHODOLOGIES

There are many techniques for testing a circuit, in this chapter we shall be concentrating on the two techniques that we have used in our work.

1.2.1 SCAN BASED TESTING

I have used Design Vision for replacing flip flops with scan cells and making a scan chain. Using this synthesized scan inserted netlist, i have generated test vectors using TETRAMAX (SYNOPSYS).Now we shall discuss in brief about scan based testing [23].

As the Figure 1.2 suggests that in normal mode the circuit under test functions as a normal circuit with gates and flip flops, while in test mode all the flip flops are connected together and formed a scan chain so as to increase the controllability and observability of the circuit during testing.

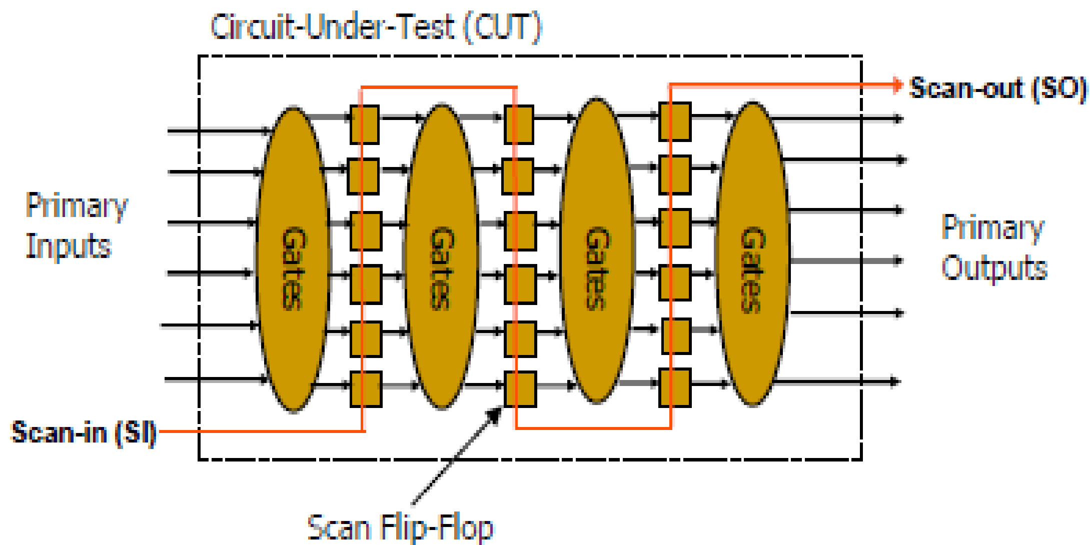


Figure 1.2 Formation of Scan Chain [23]

A simple multiplexed flip flop is presented in Figure 1.3 which is a basic scan cell which replaces the simple flip flop in the circuit.

Figure 1.4 shows a Master Slave Scan Flip Flop which is basically a multiplexed Flip Flop. Here a multiplexer and two new signals, scan-data SD and test control TC, are added to the D flip flop (DFF). The original data input D is stored in the flip flop when TC is 1 and SD is stored when TC is 0.

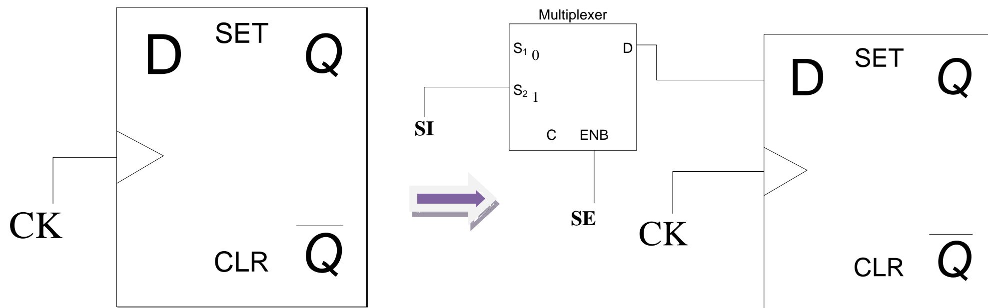


Figure 1.3 Conversion of Flip Flop to Scan Flip Flop [23]

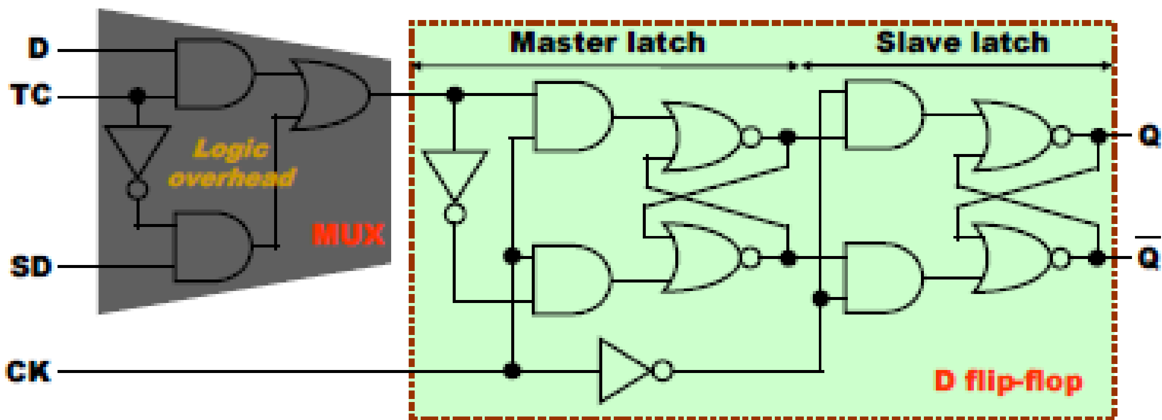


Figure 1.4 A master slave scan Flip Flop [23]

1.1

1.2 1.2.2 MAKING FLIP FLOPS DIRECTLY CONTROLLABLE FROM PRIMARY INPUTS

This scheme is utilized by MILEF ATPG which uses MIXED FAN algorithm for test generation.

In this all the flip flops are made directly controllable by making them as primary inputs in the testing mode. A brief description is given under.As we can see in Figure 1.5 that all the Flip Flops in the circuit are made primary inputs so that they can be made directly controllable and thus the circuit can be tested efficiently.

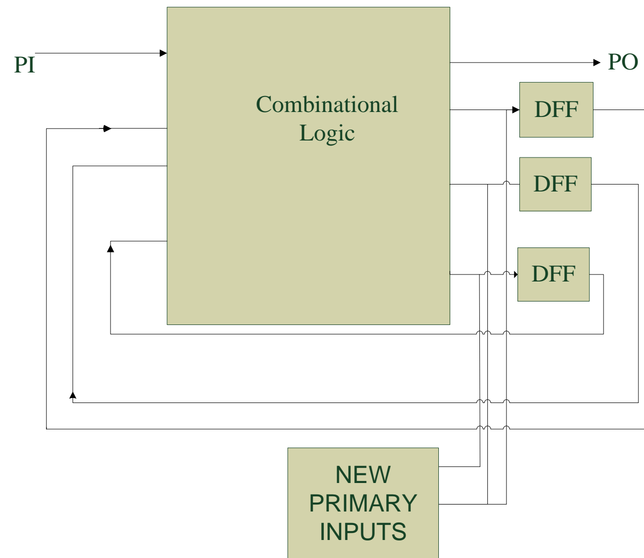


Figure 1.5 Circuit utilizing Testing Logic

1.3 Thesis Organization

This report is organized into five chapters briefed as under:

Chapter 1: Introduces the motivation for our Thesis, outlines the problem, various testing methodologies and states the contributions of our work done.

Chapter 2: This is the literature review which constitutes of Description of various don't care filling techniques, presents Weighted Transition Metric (WTM) model to calculate Average and Peak power consumption during Testing, Presents Hamming Distance based Reordering scheme for with and without don't care test set and backgrounds of test data compression and decompression are given for various compression schemes.

Chapter 3: Proposed Multi codes Compression Scheme and Decompression decoder are presented with Double Reordered Hamming Distance based reordering scheme and Tools used.

Chapter 4: Comprises of a detailed comparison of various compression schemes and our MCC scheme implemented in MATLAB 7.6b for four ISCAS 89 Benchmark circuits with test data set generated for un-compacted data using two tools namely TETRAMAX and MILEF ATPG and also presents the synthesis results for the decoder which is implemented in the two synthesis tools namely Design Vision and Xilinx.

Chapter 5: Conclusion and Future Work that can be done is presented in this thesis.

CHAPTER 2
LITERATURE REVIEW

2.1 WEIGHTED TRANSITIONS METRIC MODEL

In this section an examination of the impact of test set encoding on power consumption during scan testing is done. We then show how power consumption can be minimized by appropriately assigning binary values to the don't care bits in T_D and then applying various run length encoding for test data compression.

For a CMOS circuit, power consumption can be classified as either static or dynamic. Static power consumption, which is caused by leakage current, is usually negligible and therefore ignored. Dynamic power is consumed hence the outputs of circuit elements make high to low and low to high transitions, this constitutes the predominant fraction of CMOS power consumption.

For scan vectors, the dynamic power consumption during testing depends on the number of transitions that occur in the scan chain as well as on the number of circuit elements that switch during the scan in and scan out operations. Power estimation models based on the switching activity of circuits have been presented in the literature [6], [11]. We use the weighted transitions metric (WTM) introduced in [11] to estimate the power consumption due to scan vectors. This model was validated in [11], hence we do not report on its accuracy. The WTM metric models the fact that the scan in power for a given vector depends not only on the number of transitions in it but also on their relative positions. For example, consider a scan vector v1v2v3v4v5=01000, where v_1 is first loaded into the scan chain. The 0 to 1 transition between v_1 and v_2 causes more switching activity in the scan chain than the 1 to 0 transition between v_2 and v_3. We use the same model to estimate the power consumption during scan out operation.

The weighted transitions count metric is also strongly correlated to the switching activity in the internal nodes of the core under test during the scan in operation. It was shown experimentally in [11] that scan vectors that have higher weighted transition metric dissipate more power in the core under test.

Consider a scan chain of length l and a scan vector $t_j = t^*_{j,1}, t^*_{j,2} \ldots\ldots t^*_{j,l}$, with $t^*_{j,1}$ scanned in before $t^*_{j,2}$, and so on. The weighted transitions metric for t_j denoted WTM_j is given by

$$WTM_j = \sum_{i=1}^{l=1} (l-i).(t^*j \oplus t^*j, i+1) \tag{2.1}$$

If the test set T_D contains n vectors $t_1, t_2 \ldots., t_n$, then the average scan in power P_{avg} and peak scan in power P_{peak} are estimated as follows:-

$$P_{avg} \quad \frac{\sum_{j=1 i=1}^{n} \sum_{i=1}^{l-1} (l-i).(t^*j, i \oplus t^*j, i+1)}{n} \tag{2.2}$$

$$P_{peak} = \frac{\max}{j \in \{1,2,\ldots n\}} \left\{ \sum_{i=1}^{l-1} (l-i).(t^*j, i \oplus t^*j, i+1) \right\} \tag{2.3}$$

If the peak power exceeds a threshold value, it can cause structural damage to the silicon or to the package. Likewise, elevated average power can also cause structural damage to the silicon, bonding wires, or the package. It also adds to the thermal load that must be transported away from the device under test.

Scan in power is influenced by the manner in which the don't cares in T_D are mapped to binary values. While P_{avg} and P_{peak} can be minimized by choosing an appropriate mapping, such a mapping is not guaranteed to provide high test data compression as well as reduced transitions during scan in. Even though we do not directly address scan out power, our experiments with benchmark circuits show that this approach reduces the number of transitions and the resulting WTM during scan out. This is an added advantage of using encoded test sets for scan testing.

Scan out vectors are dependent on the Scan in vectors and the relationship between the two depends on the function implemented by the circuit under test (CUT). Determining the effect of filling don't cares cubes on the Scan out vectors require circuit simulation. We cannot afford the time required to compute the effect of each decision on the scan out vectors, hence we just focus on minimizing the scan-in power.

2.2 Don’t care Filling Techniques

An Automatic Test Pattern Generator (ATPG) tool generates test vectors for a given circuit with assuming circuit inputs don’t care values i.e.an un-compacted test set contains various inputs with don’t care bits specified by ‘X’.

A brief description of some of the approaches for filling the don’t care bits is as follows-

2.2.1 Random Fill (RFILL)

Random Fill [11] is the conventional approach for filling the X’s in the test cubes. R-Fill involves replacing the X’s randomly with 1’s and 0’s.The idea behind R-fill is that it increases the chance of detecting additional faults with a single test cube to hopefully eliminate the need for other test cubes so that they will be dropped from the test set when it is reverse fault simulated.

However R-fill is very bad in terms of Power, however random filling the X’s may result in a lot of transitions when scanning the vectors into the scan chain, which may result in a lot of switching activity in the circuit.

An option for R-Fill is given in all the available ATPG which generates an Randomly specified compacted test data vectors for any given circuit.

When power is a consideration it is generally better to use Minimum Transition Fill (MT Fill) approach.

2.2.2 Minimum Transition Fill (MT Fill)

Minimum Transition Fill [12] involves filling strings of X’s with the same value to minimize the number of transitions, for example, when filling the test cube 01XX10, it would be best to fill the strings of X’s with 1’s i.e. 011110.

For each string of X’s in a test cube, if the specified bits on either side of the string have the same value, then the string of X’s should be filled with that value to minimize the number of transitions. If they have opposite values, then it doesn’t matter which value the string of X’s is filled with. For example, when filling 0XX01X1X0, the first two X’s should be filled with 0’s, the third X should be filled with a 1, and the last X could be filled randomly with either 0 or 1. While MT-fill minimizes power, the drawback is that it may not be as effective as R-fill [11] for detecting additional faults.

Consequently, the reverse fault simulation step may not drop as many test vectors from the test set when MT-fill is used compared to when R-fill [11] is used. If the initial test cubes are not statically compacted, but simply filled with MT-fill, then the resulting test set will provide the minimum number of transitions per test vector for that initial set of test cubes. When two test cubes a and b are merged, the number of transitions in the resulting test cube c is never less than the maximum number of transitions in either a or b after MT-fill. This is because test cube c can be formed by specifying X's in either test cube a or b. There is no way to specify X's in either test cube a or b so that there are fewer transition than MT-fill. Hence, the peak power is minimum for the initial test cubes, and it monotonically increases as static compaction is performed, i.e., it can never decrease.

2.2.3 ZERO FILL

When we need to perform compression using run length codes and we need strings of 0's for better compression results we can fill all the X's bits with 0 bit.

For example, when filling the test cube 01XX10, it would result in 010010.

This increases the compression ratio, however this may impact a negative response on the switching activity which may depend upon the nature of the test set generated by the ATPG.

2.2.4 COLOUMN WISE BIT STUFFING

Column Wise Bit Stuffing (CBS) [6] is an approach which when used with difference vector increases the compression ratio to a greater extent compared to other X's filling approaches discussed earlier for most of the circuits.

Consider an example-

10110X00XXX010

111X0X0X10X0XX

1X100XX01X00X1

0XX0XX10XXX0XX

101X1X1X10X00X

The bit stuffing of first vector will be done in such a way that it generates maximum zeroes to give maximum compression with run length based codes like Golomb, FDR. So for such cases, the don't care bits will be replaced by zeroes. For run length based codes like EFDR and AFDR, the better compression can be achieved by increasing the run length of 1s as well as

0s. For such case, the don’t care bit will be replaced by the same value of its prior bit i.e. if the prior bit is 1 then the don’t care will be replaced by 1 and if the prior bit is 0 then the don’t care will be replaced by 0.

For the second test patterns and onwards, the don’t care bit will be replaced by the same value which its upper vector has at the same position. The goal here is to get the maximum zeroes in difference vector. In given example, the first vector is 10110X00XXX010. So here all don’t care bits are replaced by zeroes and new bit stuffed vector will be 10110**0**00**000**010. Now for the second test pattern, the bit stuffing will be like this.

1st vector 1 0 1 1 0 0 0 0 0 0 0 0 1 0

2nd Vector 1 1 1 x 0 x 0 x 1 0 x 0 x x

Bit Stuffed 2nd Vector 1 1 1 **1** 0 **0** 0 **0** 1 0 **0** 0 **1 0**

The bit stuffed vectors for our example are as follow:

10110000000010

11110000100010

11100000100011

01100010100011

10101010100001

2.3 HAMMING DISTANCE BASED REORDERING

2.3.1 INTRODUCTION

The power dissipation during testing [7] is minimized by reducing the number of transition in the circuit. This is achieved by reducing the hamming distance between successive test vectors. Usually test vectors are generated randomly and hence it is necessary to rearrange the order of occurrence of test vectors so that the hamming distance between successive test vectors is minimum. In general the total switching power in the whole circuit is proportional to the hamming distance of input test vectors. Therefore the reordered test vector set with minimum hamming distance is used for testing the CUT to reduce the switching power. The problem of minimizing switching power is solved by graph theory using Hamiltonian path [3] technique. Graph G (V, E) is defined with V nodes and E edges. The problem is formulated by considering the test vector as node and hamming distance between them as edge cost of the graph. Here the

Hamiltonian path is a path with all nodes and minimum total edge cost. Graph Theory based Reordering algorithm [8] is used to construct the Hamiltonian path, which is resultant reordered test vector set whose total hamming distance is minimum. Now the path developed by the algorithm is reordered test vector sequence which offers less number of transitions at the input which in turn results in reduced power dissipation in the circuit under test during testing [3]. Heuristic approach is used in the algorithm to find more suboptimal sequences.

2.3.2 HAMMING DISTANCE REORDERING

Total Hamming Distance (THD) [24]: Total hamming distance is defined as Sum of hamming distance between successive test vectors in the sequence. Let hamming distance d $[t_i, t_j]$ be the total number of changes between i^{th} and j^{th} test vector. The Total Hamming Distance (THD) for the whole test vector set is calculated by the following relation

$$\text{THD} = \sum_{i=1}^{n-1} d[t_i, t_{i+1}] \qquad (2.4)$$

Where n represents total number of test vectors in the whole set.

The overall procedure to minimize the switching activity during testing is as follows:

1. Consider a digital circuit with p inputs and q outputs.
2. Generate all the test vectors to detect all the single stuck at faults of the circuit. Let the number of test vectors be n.
3. Find the hamming distance between each and every test vector and load the same in array hd of size n x n. Let hd[i][j] be the array elements which gives hamming distance between i^{th} and j^{th} test vectors.
4. Apply reordering algorithm to find the reordered test vector sequence with minimum total hamming distance.
5. Perform fault simulation with reordered test vector set which gives minimum number of transition and hence less power dissipation.
6. Since Heuristic based algorithm generates more sub-optional sequences, select the best sequence with least total switching activity.

The reordering algorithm used in step 4 is discussed in the next sub-section.

A. Reordering Algorithm:

The various parameters used in the algorithms are as follows:

$t_1, t_2, \dots t_n$ be n test vectors with m bits each.

T= {1, 2… k … n} where k represents k_{th} position in the vector set generated by ATPG.

R is a set to store ordered test vector sequence and Q is a set to store T-R.

Step 1: Select a test vector x such that swa_init[x] is minimum in the array swa_init [ ]. add x to set R.

Step 2: Select a test vector y such that hd[x][y_{min}] is minimum in the array

Step 3: Add y_{min} to R; Q=T-R; x_{min}=y_{min}

Step 4: From the array hd[x_{min}][j] when j varies as in Q, find y_{min} so that hd[x_{min}][y_{min}] is the smallest value. Go to step 3.

Step 5: In the step 4, if hd [x_{min}][j] has more than one smallest value, then such number of reordered sequence will be generated for every x_{min}.

These sequences are called as sub-optimal sequences. Finally the set R will have reordered test vector sequence with minimum hamming distance which results in minimum switching activity during testing.

The above procedure is illustrated with simple full adder circuit with 3 inputs and 2 outputs. The test vector set that used to detect the entire single stuck at faults is given in Table 2.1. The set consists of 7 vectors with total hamming distance as 15. The vectors are represented by the order of occurrence for the sake of convenience.

Table 2.1 Test vectors for full adder circuit

Test Vector set n=7	No
000	T_1
111	T_2
001	T_3
010	T_4
101	T_5
011	T_6
111	T_7

The hamming distance array hd[ ][ ] of order n x n is constructed. This is given as in (2). It is known that the number of bit changes between t1 and t2 test vectors in the table is three.

Similarly between t2 and t3 test vectors is two. This is shown in the array as hd [1][2]=3 and hd[2][3]=2 respectively. The hamming distance array hd [ ][ ] is developed in this method.On application of reordering algorithm to this matrix hd [ ] [ ], the reordered test vectors are generated with minimum hamming distance.

The solution and the THD are given as follows:

Unordered sequence: t1 – t2 – t3 – t4 – t5 – t6- t7 THD: 15

Ordered sequence: t3 – t1 – t4 – t6 – t2 – t5– t7 THD: 6

The total hamming distance for the resultant ordered sequence is 6 which show that 60 % of total hamming distance [24] is reduced when compared with that of unordered sequence. This reduces the switching activity and hence the power dissipation in the circuit. The above approach is experimented with various benchmark circuits and results are appreciable and improvement was achieved when compared to existing methods.

$$Hd[i][j] = \begin{matrix} 0 & 3 & 1 & 1 & 2 & 2 & 1 \\ 3 & 0 & 2 & 2 & 1 & 1 & 2 \\ 1 & 2 & 0 & 2 & 2 & 1 & 2 \\ 1 & 2 & 2 & 0 & 3 & 1 & 2 \\ 2 & 1 & 2 & 3 & 0 & 2 & 1 \\ 2 & 1 & 1 & 1 & 2 & 0 & 3 \\ 1 & 2 & 2 & 2 & 1 & 3 & 0 \end{matrix}$$

2.3.3 HAMMING DISTANCE BASED REORDER SCHEME FOR DON'T CARE TEST SET

This scheme has been proposed in [6] and has been done with the following assumptions:

1. The ATPG derived test set for given DUT contains the test patterns with a large amount of don't care values.
2. Stuck at faults based test patterns can be reordered without any loss of fault coverage [26].

Here only thing that needs to be taken care is: The corresponding fault free outputs that are stored in ATE as golden references must be also reordered in the same sequence. Here we will adopt the following definition from the run based reordering scheme [25].

Incompatible & Compatible - Given two bits i, j = {O, 1, X}, i and j are incompatible if i = 0 and j =1 or vice versa. All other circumstances are compatible.

Distance - The distance between two scan frames is equal to the number of corresponding incompatible bits. This definition is similar to Hamming distance with extension of don't-care bits. For example, given two frames F1 = (10XX01) and F2 = (001X11), the distance d (F1, F2) is 2 because the first and the fifth corresponding bits in the frames are incompatible.

2.3.3.1 Selection of First Test pattern of Reordered Test Set

For the selection of first test pattern, the heuristic that is applied in this scheme is "**Hardest Path First**". The test pattern with minimum don't cares will be selected as the first test pattern of reorder list. The reason for selecting the test pattern with minimum don't care bits is that there is a minimum flexibility to stuff the bits later. If more than one test pattern have minimum don't care values than anyone can be selected. Following example illustrates the procedure. The test set T contains five test patterns of 14 bits each.

1X100XX01X00X1;

111X0X0X1010XX;

101X0X00XXX010;

0XX0XX10XXX0XX;

101X1X1X10X00X;

Here the number of don't care in each vector are

O1: 1X100XX01X00X1 = 5

O2: 111X0X0X10X0XX = 6

O3: 10110X00XXX010 = 4

O4: 0XX0XX10XXX0XX = 9

O5: 101X1X1X10X00X = 5

So here the third test pattern from orignal list (O3) is selected as first test pattern of reordered new list (N1).

N1: 10110X00XXX010 = 4

In future, the bit stuffing of first vector will be done in such a way that it generates maximum zeroes to give Maximum compression with run length based codes like Golomb, FDR.

The scheme can be further improved for EFDR, AFDR. For such cases, the bit stuffing will be done to increase the run length of 0s as well as 1s.

2.3.3.2 Reordering for Remaining patterns

From remaining test patterns, the patterns with minimum Hamming distance from first pattern will be placed next to first pattern. It is decided to take the next vector with minimum Hamming distance because when the further bit stuffing and difference vector will be done, this new sequence will generate maximum zeroes so run length will in turn increase and hence the compression will increase. If we take each test pattern as a vertex in a complete undirected graph G, and the distance between two patterns as the weight of an edge. Then this problem is similar to Hamilton problem, which is NP-hard and solved by various greedy algorithms.

The simplest pure greedy algorithm is: choosing as the next pattern in a path the one that is closest to the current pattern, provided it hasn't been visited yet. It seems that the Hamilton path of G is the solution to our reordering problem.

Now from the first pattern of new list (N1) the Hamming distances for remaining all vectors (O1, O2, O4, O5) are calculated. The Hamming distance is calculated bit position wise. We will define here the Hamming distance between the two patterns as the number of positions for which both the patterns have different values. Let’s find out the Hamming distance between N1 and O1

N1: 1 0 1 1 0 X 0 0 X X X 0 1 0

O1: 1 X 1 0 0 X X 0 1 X 0 0 X 1

Hamming Distance Hd: 0 0 0 1 0 0 0 0 0 0 0 0 0 1

Here we will compare bit from N1 with the bit with same position from O1. If N1(i) = O1(i) or N1(i) = X or O1(i) = X then Hd(i) = 0 otherwise Hd(i) = 1. Total Hamming Distance between N1 and O1 is sum of Hd (i). As the further bit stuffing is done comparing bits Column wise, so the Hd (i) is kept 0 when any of N1 (i) or O1 (i) is X.

For given example, let’s calculate the Hamming distance

Total distance between N1 and O1 = 2

Total distance between N1 and O2 = 1

Total distance between N1 and O4 = 3

Total distance between N1 and O5 = 2

As the O2 has minimum Hamming distance from N1, we will put O2 at second position of new set. i.e. as N2. Now remaining vectors will be compared with N2 and the N3 will be placed based on Hamming distance and so on. The reordered test set is as follow:

10110X00XXX010

111X0X0X10X0XX

1X100XX01X00X1

0XX0XX10XXX0XX

101X1X1X10X00X

2.4 TEST DATA COMPRESSION

Figure 2.1 shows a conceptual architecture for SOC testing with compressed test. As the figure shows, the compressed test (T_E) is transported to the SOC through the ATE I/O channel. Subsequently, the TD is transported to the core under test through the TAM. Finally the T_E is decompressed into the original test by the corresponding decompression circuitry, and the cores under test can receive the original test.

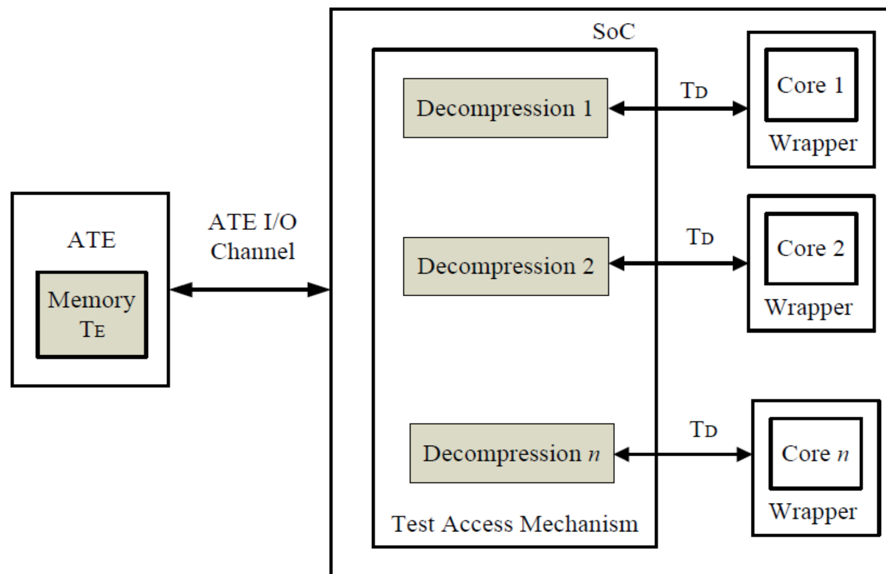


Figure 2.1 Conceptual architecture for the SOC testing with using on chip decompression [1]

2.4.1 Run-Length Coding

In this paragraph, we offer an easy concept to explain Run-length Coding, in Figure 2.2, an example of 3 bits variable-to-block, it is presented with fixed-length codeword, instead of using length original code. In addition, the test set T_D in Figure 2.3, if it follow Figure 2.2 coding rule, can be divided into few segments by applying the method of 0's run-length coding. When the runs of 0's are less than six, using one codeword to represent a segment that starts with 0, ends up with 1. On the contrary, when runs of 0's are over than six, segment is represented by the several codeword. The number for the bit of encoded T_E and the original T_D is 27 and 42. Thus, this method can efficiently reduce test data. The compression ratio of run-length coding will be changed along with different definitions between codeword and original code. In the next section, we will introduce several encoding methods with using run-length coding concept.

Original data		Codeword
1		000
01		001
001	⇨	010
0001		011
00001		100
000001		101
0000001		110
00000000		111

Figure 2.2 Three bit variable to block encoding [27].

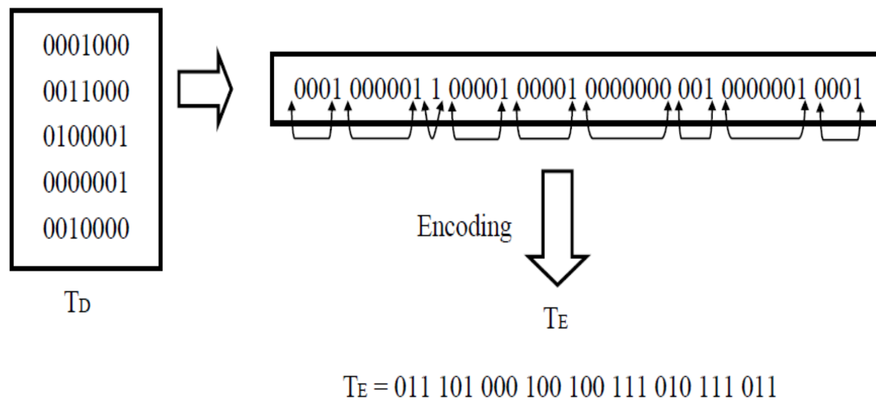


Figure 2.3 An example of run length encoding [27].

2.4.1.1 Golomb Code

Before introducing Golomb coding, the group size m must be decided, this parameter has powerful influence on compression efficiency. In [19], we can determine the value of m, when $m=2^N$, $N \geq 1$ is appropriate for testing data. In most of cases, it finds out to have the best compression when N=2, the encoding process for m= 4 is shown in Table 2.2, few characteristics have been summarized as following:

- A Codeword consists of group prefix and tail.
- The number of bit of group prefix is given by L+1, this value also represents group number, L is a run of length, ex. when L=5, the group prefix and tail are 2 bits.
- The tail is a sequence of $\log_2 m$ bits.

To find out the optimum value of m, first of all the probability p of runs of 0's is found out in the test data set. Then m is given by-

$$m=ceil\left((1/\log_2(p))*\log_2(1/(1+p))\right) \qquad (2.5)$$

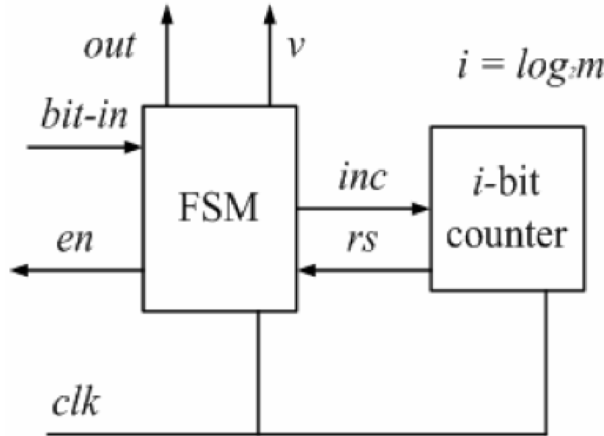


Figure 2.4 Block Diagram of GOLOMB Decoder [19]

The block diagram of the Golomb decoder is shown in Figure 2.4, it consists of a 9-state finite state machine (FSM) and the $\log_2 m$ bit counter. The bit in is primary input, the encoded test data is sent via bit-in to FSM when the signal en is high. The inc is a count-up signal to notice counter, as it finishes counting, the rs signal/line will show high condition. The out

represents the output decoded data. When ‘v’ is high which means the output data is valid. The movement of decoder is that the counter counts up to m cycles. to enable the next input data is ready to be sent. Moreover, the v is high as the outputs m 0’s while the counter is doing count-up. On the other hand, when bit-in is 0, the counter stops, thus, FSM follows the tail, the part of codeword, and sends out few 0s and signal 1 but out, then the movement of decompression would be competed in orders.

Table 2.2: Example of Golomb coding with m= 4 [19]

Group	Run-length of 0s	Group prefix	Tail	Codeword
A1	0	0	00	000
	1		01	001
	2		10	010
	3		11	011
A2	4	10	00	1000
	5		01	1001
	6		10	1010
	7		11	1011
A3	8	110	00	11000
	9		01	11001
	10		10	11010
	11		11	11011
………..	…………	……..	……	…………….

2.4.1.2 Frequency Directed Run-length (FDR) Code

In [20], it illustrates that the FDR coding has been proposed according to the frequency of 0’s run length shown in most of test pattern. The compression technique is given in Table 2.3. We can summarize several characteristics as following:

- The codeword consists of group prefix and tail.
- The group prefix and tail have the same number of bits in each codeword.

- If the group number is Aj, the value of j is equal to the number of bit of prefix, and there will be 2^j member in a group, e.g., the two bits of prefix and the four members in the A2, the eight members and three bits of group prefix in A3.
- The value of binary in prefix's content and the value of run 0s length in the first member are the same, e.g., the binary value of prefix is 6 as well as the run-length of 0s is also 6 in first member.
- There is two-bit gap in codeword between groups.
- If A_k represent the last group L max as the longest rum-length, then relation is $(2^k-3) < L_{max} \leq (2^{k+1}-3)$. For example: if L_{max} is 15, thus the k must be 4 to match the relation above, which means the longest length, 15, locates on A 4 group and the longest length accepted is A4 is between 14 to 29.

Table 2.3: An example for FDR code [20]

Group	Run-length of 0s	Group prefix	Tail	Codeword
	0		0	00
A1	1	0	1	01
	2		00	1000
	3		01	1001
A2	4		10	1010
	5	10	11	1011
	6		000	110000
	7		001	110001
	8		010	11000
A3	9	110	011	110011
	10		100	110100
	11		101	110101
	12		110	110110
	13		111	110111
………..	…………	……..	……	…………….

The block diagram of FDR which consists of three parts: 9-state FSM, k-bit serial input counter, and $\log_2$ k-bit counter shown in Figure 2.5, The bit_ in is inputting encoded data when the en is high, and the out outputting decoded data is valid while the v is high. The counter-in is a path which sends data to k-bit counter, and the shift is a signal that takes control of the counter-in. the dec1 is used to notice k-bit counter starting count-down, and the rs1 is high as the counting finishes. The dec2 tells $\log_2$ k-bit counter start doing count-down, inc, on the other hand, tells it starts counting up. The rs2 will tell when it finishes counting. The operation of circuit is shown as Figure 2.5:

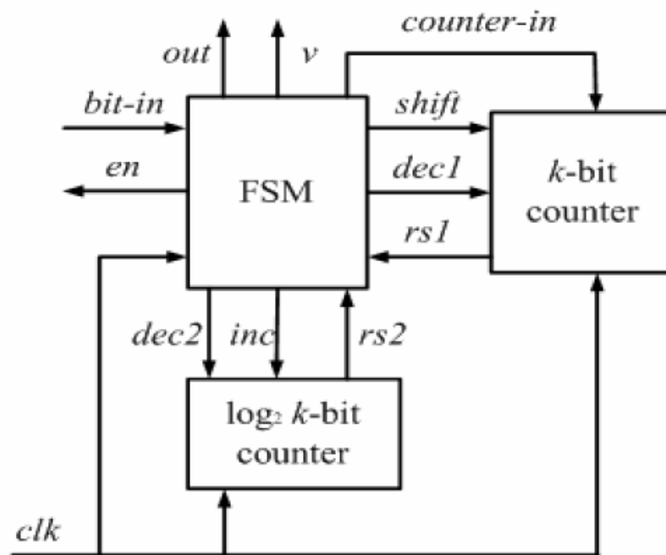


Figure 2.5 Block diagram of FDR decoder [20]

1. The FSM sends the data of prefix into the K-bit counter, signal en, shift and inc keep at high till it receipts the last bit, 0 to divide the prefix from tail.
2. After receiving the data from prefix, FSM enable *dec1* and v high and output several 0s till the k-bit counter finishes counting down.
3. The $\log_2$ k-bit counter also counts the number of bit in prefix during the and the $\log_2$ k bit counter starts counting down the number of bit of tail which received by k-bit counter.
4. When the tail data in k-bit counter counts down, the out outputs a few of 0's and output a 1 after it finishes counting to achieve the decoding for a codeword.

2.4.1.3 Extended Frequency Directed Run-length (EFDR) Code

This method extends the function of encoding run 0s length in FDR to reach encoding 1's run length. The advantages are proposed in [21], Table 2.4 list the process of encoding which adds extra bits on the length of codeword as the boldfaces in the Figure to tell which rum it is decoding in present. Moreover, the rest of codeword stay the same. The extra bit must be located in MSB of codeword to inform the decoder whether it should invert the output data, shown in

Table 2.4: An example for EFDR code [21]

Group	Run-length of 0s	Group prefix	Tail	Codeword Run of 0's	Codeword Run of 1's
A1	0	0	0	000	100
	1		1	001	101
A2	2	10	00	01000	11000
	3		01	01001	11001
	4		10	01010	11010
	5		11	01011	11011
A3	6	110	000	0110000	1110000
	7		001	0110001	1110001
	8		010	0110010	1110010
	9		011	0110011	1110011
	10		100	0110100	1110100
	11		101	0110101	1110101
	12		110	0110110	1110110
	13		111	0110111	1110111

Figure 2.6, before inputting prefix data, the extra bit will be input first to enable XOR gate by FSM via the mux signal, if the mux is 1, the out is inverse the fout data, if it's 0, the out is equal fout, through the signal mux can decode codeword which encode by run 0s run-length or run 1s run-length. The operation of whole signals and is the same in section 2.1., except mux and XOR gate.

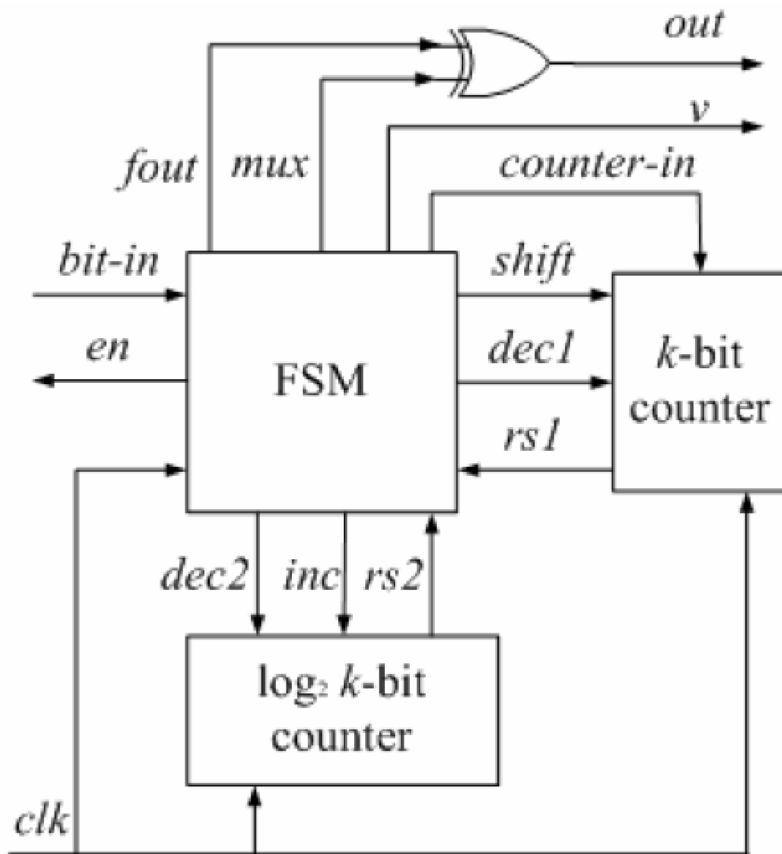


Figure 2.6 Block Diagram of EFDR [21]

2.4.1.4 Alternative Run-length (AR) code

The AR coding also improves FDR coding to encode input test data by alternating 0's run length and 1's run lengths without adding length of codeword. Table 2.5 illustrates the process of AD coding, as we can see that the codeword of AD coding is the same as FDR coding whether the input test data is encoded by run-length of 0's or 1's.

Table 2.5: An example for AR coding [22].

Group	Run-length of 1s	Run-length of 0s	Group prefix	Tail	Codeword
	0	0		0	00
A1	1	1	0	1	01
	2	2		00	1000
	3	3		01	1001
A2	4	4	10	10	1010
	5	5		11	1011
	6	6		000	110000
	7	7	110	001	110001
	8	8		010	110010
A3	9	9		011	110011
	10	10		100	110100
	11	11		101	110101
	12	12		110	110110
	13	13		111	110111

Figure 2.7 is an example to explain which type can encode by run-length of 0's or 1's. Figure 2.7 is an example, there are 26 bits in input stream, nevertheless, the encoding result of FDR consists of 30 bits, over 4 bits, but the result of AR is 18 bits, thus AR coding is a efficient method in this case.

Input data stream: 00000001111111111000000001 (26- bits)

FDR encoded data: 110001000000000000000000110010(30-bits)

AR encoded data: 110001110011110010 (18-bits)

|run 0 | |run 1| |run 01|

Figure 2.7: Compare of FDR coding and AR coding [22]

When encoding run 0s length it always fetch segment which starts with 0, ends with 1. 1's run length, on the opposites, starts with 1, and ends with 0. On other hand, the detail description

that the efficacy of power consumption and benefit of compression ratio present is discussed in [22].

The circuit diagram block of AR decoder is shown in Figure 2.8, this is adding a T Flip-Flop with negative trigger and a XOR gate in FDR decoder. The operation of addition part is using signal feedback and negative trigger to control the point t, when t=1, the output data of out is invert fout, when t=0, the output data of *out* and fout is equal. The operation of other part is the same of FDR decoder.

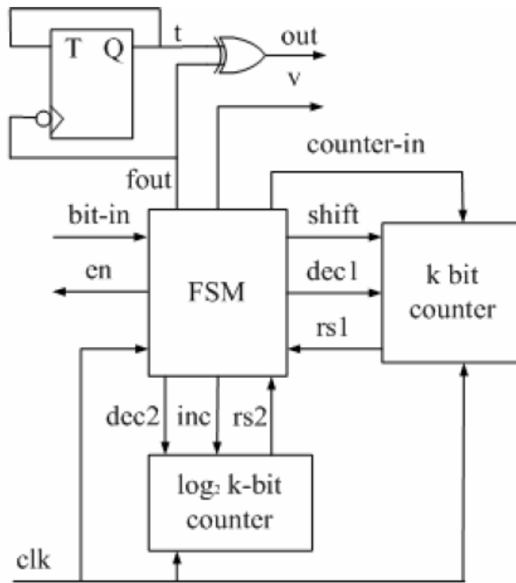


Figure 2.8 Block diagram of AR decoder [22]

2.5 Cyclical Scan Register Decompression Architecture

The result can be discovered by analyzing the majority of test set, there will be a few different from one test pattern to others, which is the similarity between test patterns. In Figure 2.9, it also illustrates the characteristic in most test set, thus using this similarity of test pattern to change compressing target from the original test pattern (T_D) to the different vector (T_{diff}), and the T_{diff} is generated by the T_D , if $T_D = \{ t1, t2, t3, \ldots, tn\}$ and $T_{diff} = \{t1, t1 \oplus t2 \oplus t3, \ldots, tn\text{-}1 \oplus tn\}$, and using various run-length coding to compress T_{diff} into the T_E . At the process of decoding, through architecture shown in Figure 2.9, the T_E will be decoded into T_{diff} by on chip decoder with run length coding, and using XOR gate and feedback from the cyclical scan register

(CSR) to decompress the T_{diff} into the T_{diff} into the T_D . The compression ratio of run length coding will be growing, because the process that transforms the T_D into the T_{diff} is increasing the length of run of 0's. Therefore, this encoding method and decoding architecture are certainly increasing compression ratio of run length coding in most of the test set.

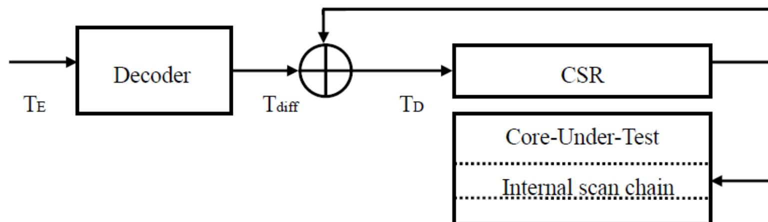


Figure 2.9 The architecture of CSR decompression [27]

The CSR decompression architecture can be changed by following the different circuit condition's in Figure 2.10, it is using boundary scan register to replace CSR, if the length of boundary scan register is longer than test pattern, then we can feedback the needed length of test pattern to XOR operation with the next test pattern. Thus, this technique can simultaneously save extra area of CSR to reach the purpose. Besides, the internal scan chain of the other cores is also can be used if the core clock is different from the core under test and controlled. The architecture is given by Figure.2.11 and used when the internal scan chain's length is equal to the test pattern or shorter than its length. If it is shorter, the user defined scan elements can be added to complete the test pattern length.

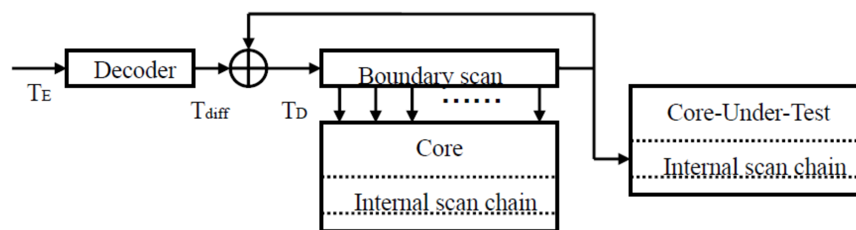


Figure 2.10 The decompression architecture of using boundary scan register to replace CSR [27]

The SCR decompression architecture can be changed by following the different circuit conditions, in Figure 2.11, it is using boundary scan register to replace CSR, if the length of boundary scan register is longer than test pattern, then we can feedback the needed length of test pattern to XOR operation with the next test pattern. Thos, this technique can simultaneously save extra area of CSR to reach the purpose. Besides, the internal scan chain of the other cores is Laos

can be used if the core clock is different from the core under test and controlled. The architecture is given by Figure 2.11 and used when the internal scan chains length is equal to the test pattern or shorter than its length. If it is shorter, the user defined scan elements can be added to complete the test pattern length.

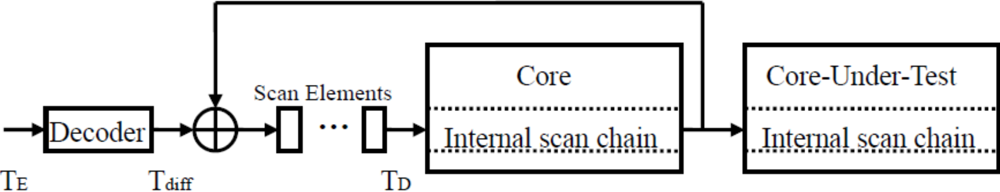


Figure 2.11 The decompression architecture for using internal scan chain and user defined scan element to replace CSR[27]

CHAPTER 3

DESIGN AND IMPLEMENTATON OF MCC ENCODER AND DECODER WITH DOUBLE REORDERING SCHEME

3.1 Double Hamming Distance Based Reorder Scheme

This is based on the above mentioned reordering schemes [6], [24] discussed in section 2.3. As the name suggests we have incorporated both the schemes so as to have a better saving of testing power and a better compression ratio when the proposed scheme is implemented with proposed Multi Compression code (MCC) after taking the difference vector.

A flow chart depicting the scheme is given below in Figure 3.1

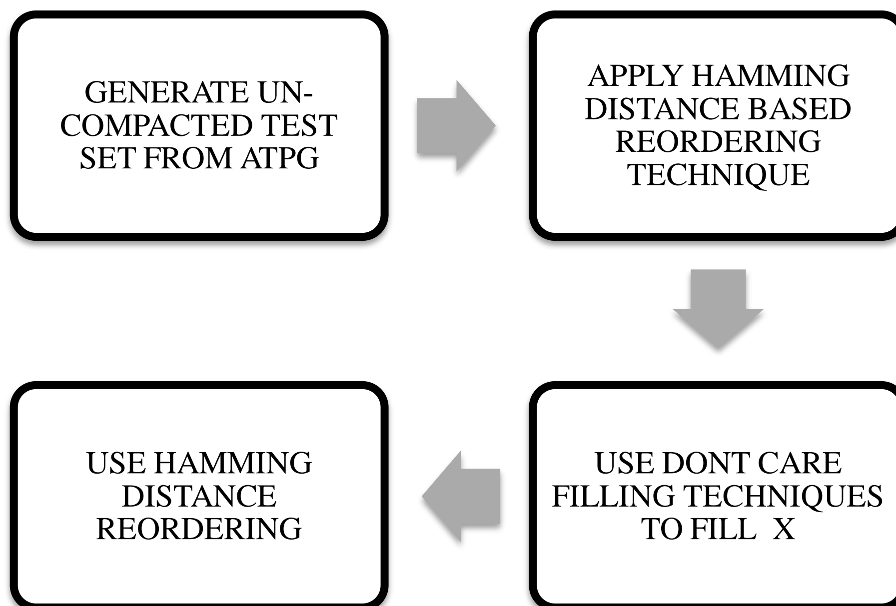


Figure 3.1 Double Hamming Distance Based Reordering Scheme

As described in Figure 3.1 we use uncompacted test set with don't care bits generated by an ATPG in this scheme.

Next step is to apply the double hamming distance scheme proposed in [6] which reorders the original test set. The next step is to fill the don't care bits with don't care filling techniques mentioned in

[11], [12], [6] ,and the last and final step is to apply [24] to again reorder the test set with minimum hamming distance between the successive patterns.

This scheme when apply with proposed MCC technique after taking difference vector gives better compression result along with an added advantage of low power being utilized in doing testing.

3.2 Compressions Analysis

Each test data compression method has the highest compression ratio for different circuits design with the same ATPG tool. Second, as we know, if the number of bit of codeword is fewer the compression efficiency is more excellent. Thus we can use less bit of codeword to represent longer original test data.

For example consider the length of codeword in these four methods as shown in Figure 3.2. When the run length is ≤ 5, the FDR has the shortest codeword. On the other hand, when the number of run length is > 6, Golomb code (group size m = 16 or 32) has the shortest codeword. Moreover, EFDR and AR are extended from FDR, under each different status of 1's run lengths in test set. EFDR which adds on bits length in codeword to stand for the encoding of 1's run length or 0's run length as well as AR, which interchanges 1's run length with 0's run length without any increase in length as its encoding method.

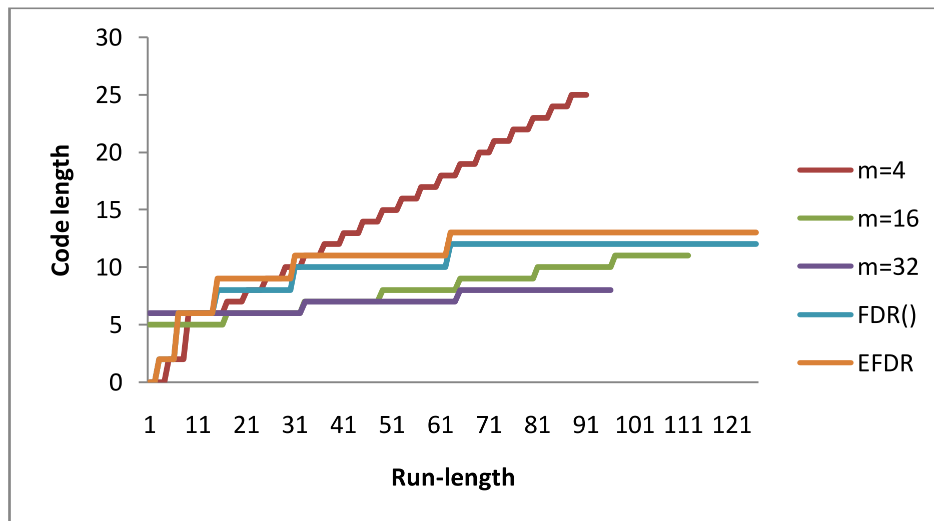


Figure 3.2 Compared on number of codeword in each run length encoding

Thus, we provide combining decoder which includes the decoding of these four methods to accord with the requirement of the highest compression ratio in test pattern. Besides, we

propose Multi code compression (MCC) is implemented to increase the compression ratio with each encoding methods advantages (FDR is applied hence 0's run length is < 3 to 5, Golomb is applied when 0's run length is > 6. EFDR or AR is provided for 1's run length.

3.3 Analysis of Multi Code Compression

Detailed analysis of multi code compression method is described in this section. Let the precompouted test set be T_D= {td1, td2, td3…. tdn} which has equal numbers of bit as r from td1 to tdn, and let the compressed test data be T_E = {te1, te2, te3…. ten}. Compressed te1 to ten has different size due to run 1's or 0's distribution and the adequate run length encoding. Let us size due to run 1's or 0's distribution and the adequate run length encoding. Let us assume Golomb, FDR, EFDR and AR as G, F, E and A. The number of bit of tdn as c, therefore, Gc1 stands for the number bits in the 1st compressed test vector by Golomb. Next, with four kinds of run length method, the comparison on each number of bits in code word will be illustrated, as Table 3.1 below. The first column Is the number of run length while the rest of them represent the number of bits. Each run length methods have its longer or shorter decoding status compared to the original data. Thus, it is assumed that in every test vector, k sub runs are contained. However, AR, due to the decoding method which alternates 0 or 1's, so the value of k is different from the other three methods. L here used as the number of bit in every sub run, so td1 = {L1, L2, L3… Lk}, L1+ L2+ L3…+ Lk= r, and the first compressed test vector represented as te1 = {u1, u2, u3 ...uk}, u as the number of bit of every compressed sub runs. So that, u1+u2+u3… +uk = c1 But, the number of c1 changes with different encoding methods. With every test vector as a unit calculates the encoding efficiency by Weight (w). With the variety of run length, different codeword represents it s corresponding method. We assume the original in lengths number of bit is os. csX represents the number of bit compressed by compression method X. The wX=os–csX. When the run length is y, wyX stands for the weight in every method, shown as in Table 3.2.

Obviously, the weight of each method in different run length can be easily told. For example, when run length is 4, Golomb (m =32) is 6 bit codeword. Os4 = 5, cs4G (m=32) = 6, we obtain w4G= -1 as result. Thus, the value of weight is w5F =2 when the FDR run length is 5. If the value of weight is positive, it shows this encoding method is effective. When it is 9 or negative, t shows the compression is ineffective. If the value of w is bigger, the encoding efficiency is more outstanding.

Table 3.1 Compared on lengths of codeword in each run length encoding.

Original		No. Bits of Codeword (cs)				
Run length	No. bits of original(os)	Golomb m=16 (csG)	Golomb m=32 (csG)	FDR (csF)	EFDR (csE)	AR (csA)
0.	1.	5	6	2	3	2
1.	2.	5	6	2	3	2
2.	3.	5	6	4	5	4
3.	4.	5	6	4	5	4
4.	5.	5	6	4	5	4
5.	6.	5	6	4	5	4
6.	7.	5	6	6	7	6
7.	8.	5	6	6	7	6
8.	9.	5	6	6	7	6
9.	10.	5	6	6	7	6
10.	11.	5	6	6	7	6
11.	12.	5	6	6	7	6
12.	13.	5	6	6	7	6
….	….	….				

Let a test vector be td, shown in Figure 3.3. When td5 is encode by Golomb, FDE and EFDR, this vector is separated into 9 sub runs, so k =9, on the other hand, when using AR as encoding method, k =17. However, the total number of test vectors weight can be obtained. In this case, FDR has positive value while others have negative ones. Therefore, FDR is the only method that compresses effectively and other methods are ineffectual compression. These four kinds of methods encoding result are easily and effectively shown by calculating the weights. We can use every single test vectors weights to acquire the method that has higher compression ratios.

Table 3.2 Compared on weight on each run-length encoding

Original		No. Bits of Codeword (cs)				
Run length	No. bits of original	Golomb m=16	Golomb m=32	FDR	EFDR	AR
0.	1.	-4	-5	-1	-2	-1
1.	2.	-3	-4	0	-1	0
2.	3.	-2	-3	-1	-2	-1
3.	4.	-1	-2	0	-1	0
4.	5.	0	-1	1	0	1
5.	6.	1	0	2	1	2
6.	7.	2	1	1	0	1
7.	8.	3	2	2	1	2
8.	9.	4	3	3	2	3
9.	10.	5	4	4	3	4
10.	11.	6	5	5	4	5
11.	12.	7	6	6	5	6
12.	13.	8	7	7	6	7
….	….	….				

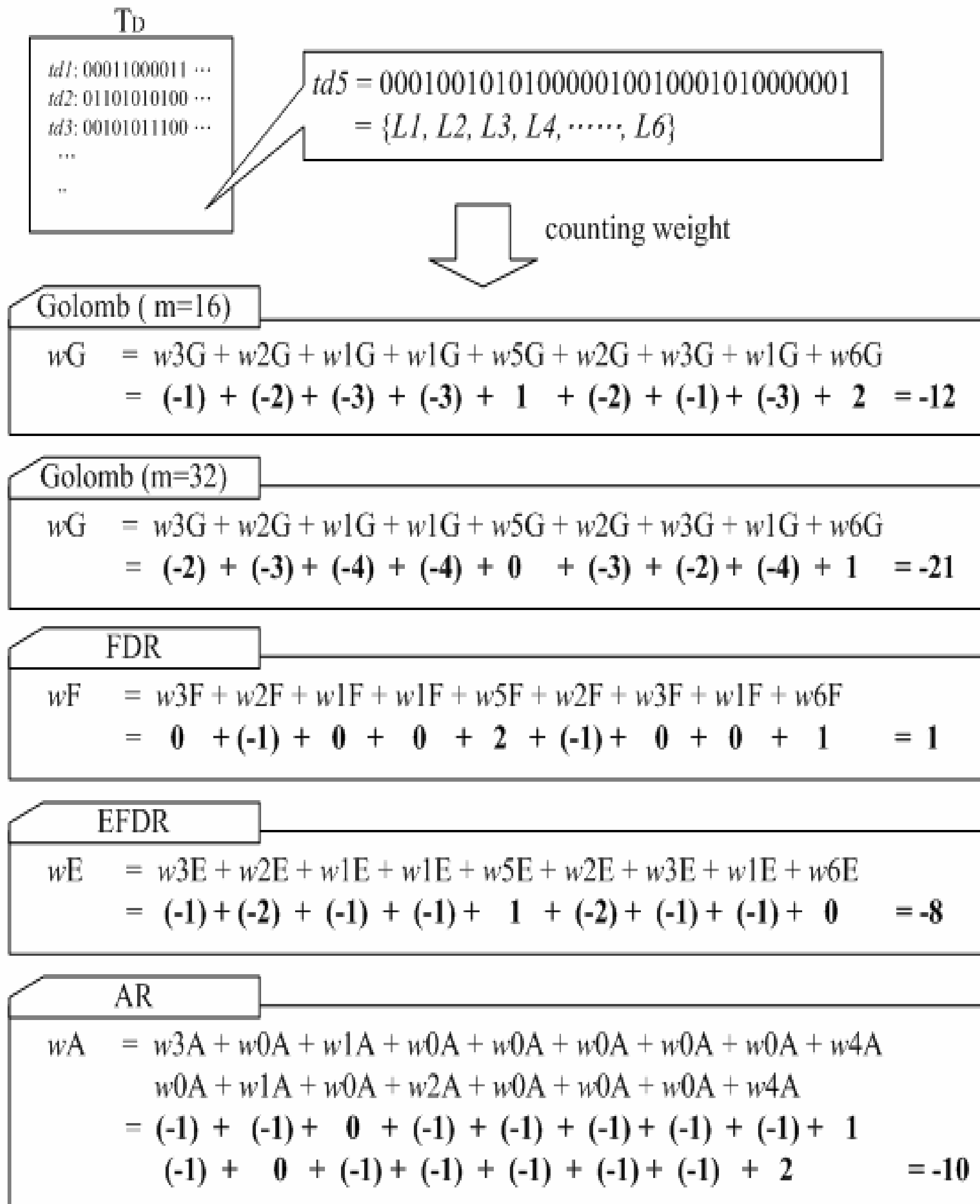


Figure 3.3 An example for weight of one test vector

3.4 MULTI CODE COMPRESSION

In the last section, we calculated each test vector's compression result in different encoding methods by weight. However, if we encode all the test pattern with this concept, it would soon occur serious negative effect while extra hardware overhead rapidly increases but compression ratio does not proportionately becomes higher.

During compression, some conditions should be considered as well, such as the frequency and distribution of 1's and 0's run length in test set, the required hardware cost for decompression and etc.

Therefore we generate difference vector test set (T_{diff}) with the test pattern reordering in the concept mentioned in section 2.5 along with the double Hamming distance based Reorder scheme presented in section 3.1 so as to increase compression and make the process low power for both average and Peak power which are required in testing.

With a careful observation of all the test data compression variable to variable run length codes we have managed to conclude on certain properties of each encoding which are listed as under.

- For l=0 to 2 no compression technique provides a positive compression for both run length of either 0's or 1's.
- For l=3 to 5 FDR and IFDR provides highest compression for run length of 0's and 1's respectively.
- For l>5 GOLOMB provides highest compression for a run length of 0's.
- For l>5 for sequence of 1's we can compare AR, EFDR and IFDR for highest compression and utilize with maximum compression.

So it's better to leave small run length of 0's or 1's as it is without compressing to get an higher percentage of compression ,rather an positive compression in various cases and thus a state called BYPASS MODE is introduced in the compression scheme.

Based on above conclusions we can deduce an MULTI CODE COMPRESSION (MCC) scheme given in Table 3.3

Table 3.3 A Multi Code Compression scheme

Run length of 0's	Compression scheme used	Run length of 1's	Compression scheme used
0	NO scheme used BYPASS mode	0	NO scheme used BYPASS mode
1	NO scheme used BYPASS mode	1	NO scheme used BYPASS mode
2	NO scheme used BYPASS mode	2	NO scheme used BYPASS mode
3	FDR	3	IFDR or EFDR or AR
4	FDR	4	IFDR or EFDR or AR
5	FDR	5	IFDR or EFDR or AR
6 or Higher	GOLOMB	6 or Higher	IFDR or EFDR or AR

3.5 TEST DATA DECOMPRESSION (DECODER)

In this Section, we first introduce decompression IIP Decoder architecture which integrates the five kinds of run length decoders. This IIP Decoder not only conform to requirement of different cores via switching control signal, but it also decrease area overhead that cause of different run length decoders needed by hardware sharing. Moreover, we also describe that implementation of decompression IIP for easy describing; we use FDR similar to represent FDR, EFDR, AR, and IFDR in the later section.

3.5.1 Decompression IIP

The decompression IIP decompresses the encode test set T_E and outputs difference vectors T_{diff}. The XOR gate and the CSR are used to generate the test pattern from the T_{diff}. The decompression IIP. Shown in Figure 3.4 totally has 7 primary input pins, 1 clock pins, 1 synchronous reset pin and 3 primary output pins. The pin bit_in is used to receive encoded data. The IIP proceeding in Golomb or FDR similar is separated by signal select. The mode signal is used to determine decoding or bypass mode. The value of Gsize [2:0] signals can assign the parameter m or be used to recognize which decoding methods while IIP operating in Golomb or

FDR similar, respectively. The signal en_FSM use to enable the IIP and the pin out transfer decoded test data to core under test. The en pin is a commutation path between ATE and decoder to notice external tester can send next bit encoded data. The truth table of mode, select, and Gsize [2:0] is clearly defined the relationship between operation methodology of IIP and the value of these pins, shown in Table 3.4. Since signal mode=0, the decompression IIP proceed with bypass mode, the spins en and v are kept in high. In other hand, when mode=1, the decoder is in the decoding mode that depend of the value of signals select and Gsize [2:0] to determine the used decoding method. If signal select = 0 , the Golomb decoding , the parameter m is assigned by signals Gsize[2:0]. Since select=1, the FDR similar decoding, the signals Gsize [2:0] can decide the IIP to use FDR, EFDR, AR, or IFDR to decode test data.

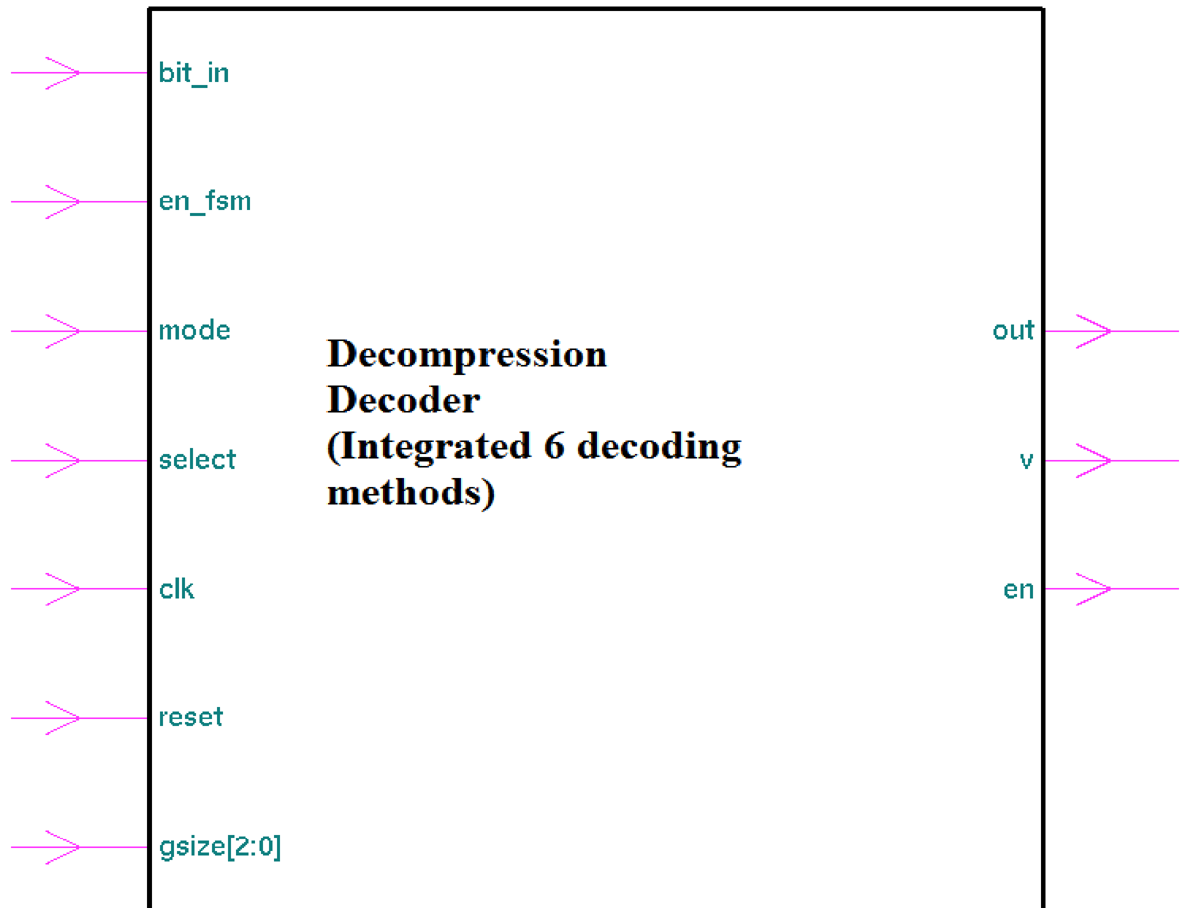


Figure 3.4 Structural drawing of decompression IIP

Table 3.4: The truth table of control signals of decompression IIP**.**

Mode	Select	Gsize[2]	Gsize[1]	Gsize [0]	State
0	0	X	X	X	Bypass
1	0	0	0	0	Goloumb m=2
1	0	0	0	1	Goloumb m=4
1	0	0	1	0	Goloumb m=8
1	0	0	1	1	Goloumb m=16
1	0	1	0	0	Goloumb m=32
1	1	0	0	0	FDR
1	1	0	0	1	EFDR
1	1	1	1	0	IFDR
1	1	1	1	1	AR

Before we introduce the architecture of decompression IIP, the parameters p and q are needed to be defined first. These parameters both are related to parameters k and m. For the FDR-similar decoding mode, we assume that the l_{max} is the longest run of 0's or 1's in test set T_D (or T_{diff}) and let k= ceil [$\log_2 l_{max}$]. The value of k represents the number of bit of prefix (or tail) in codeword which the l_{max} has been encoded by FDR-similar methodology. It means that the size o f k is changed with different requirements of test set. For the Golomb decoding mode, we know that parameter m is refereed to group size and the number of bit of tail in any group is $\log_2$ m, thus the value of m must be decided when we use Golomb code. However, we define p=MAX ($\log_2$ m, $\log_2$k) and q=MAX ($\log_2$ m, k), it means that the value of p is the bigger one between $\log_2$ K and $\log_2$ m, and the q equal the bigger one between $\log_2$ m and k.

The block diagram of decompression IIP is shown in Figure 3.5. It is majorly comprised of a 4-16 decoder, a p-bit counter, a q-bit counter, a Dout block and a FSM block. The 4-16 decoder has two purposes, one is to depend on signal of Gsize [2:0] pin to decode the data to send to p-bit counter, other is to decode control signals to blocks FSM and Dout. The truth table of the 4-16 decoder is shown in Table 3.5. While select = 0 and Gsize [2:0] = 000, the Golomb decoding mode with m =2, the signals dout [4:0] send the data that is the binary value of m-1 to p-bit counter for preparing to count down, and the other signals are low. If select =1, FDR

similar decoding mode, the signals select 1, en_mux, and en_i output control signals with different decoding methods. The p-bit counter is not only used to count the prefix and tail length of codeword in FDR similar decoding, it but also parallel accept the binary value of m-1 from the 4-16 decoder and serial shift m-1 data into q-bit counter via path shift 2 and the multiplexer MUX#2. Moreover, the q-bit counter also accepts the prefix or tail from path counter_ in. The Dout block accept control signals from the 4-16 decoder and the FSM to output correct 0's or 1's run length. The FSM block consists of nine states to control the other block. The signals dec1 and dec2 are used to decrement, the signals rs1 and rs2 indicate the reset state of the counter, and the shift 1 and enshift are the enable singles to indicate to shift data. The detail operation of the decompression IIP is as follows:

A. In mode=1 and select= 0 , the Golomb decoding, while signal bit_in accept a 1, the FSM enable signal inc/load to become high and to notify p-bit counter to load the value of m-1 from 4-16 decoder through path data [4:0]. The p-bit counter counts down to zero. The signal en is low when the p-bit counter is busy with counting and enables the input at the end of m cycle to accept another bit. During this operation, the decoder outputs m zeros via out and make the valid signal v to become high.
B. When the input is zero, the FSM starts decoding the tail of the input codeword. At the same time, the signal inc/load becomes high to enable p-bit counter for parallel loading the value of m -1. In the next cycle, the m-1 serial shift into q-bit counter via path shift2. This operation is a transformation of data [4:0] that from data type to control signal type. If m =4, the number of bit of tail is 2 bits , see Table 5-1, this mean q-bit counter needs two enable signals to fetch two bits data of tail in two clock cycles. This, data [4:0] = 00011 provide two bits of 1 to enable q bit counter.

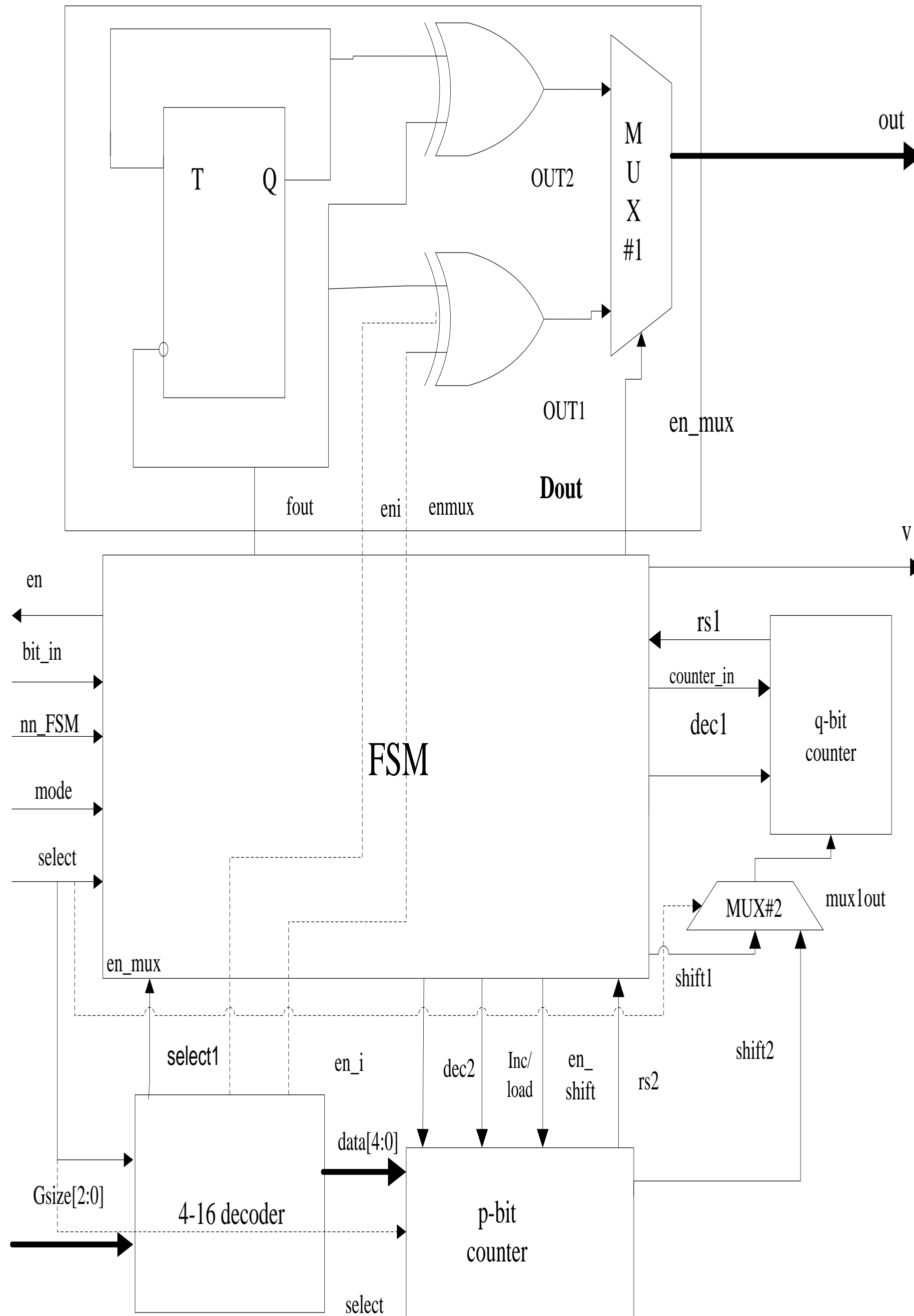


Figure 3.5 Block diagram of decompression IIP

D. When mode = 1, and select=1, the FDR similar decoding, the FSM supply the q-bit counter with the prefix. The end of the prefix is separated by the zero. The signals en, shift 1 and inc/load are high until the 0 is received. In this operation, the q-bit counter is used to store the data of prefix, and the p-bit counter uses for counting the number of bit of prefix.

E. The FSM outputs zeros, decrements the q-bit counter, and makes the signal dec1 high. It continues to output zeros throughout till rs1 become high, p-bit counter is decremented, and the signal rs2 indicates while it is in the zero throughout till rs1 becomes high. The signal v is used to identify a valid output.

F. The tail part shifted in q– bit counter through path counter in unit the p-bit counter counts down to zero. The signal dec2 maintains while it is in the zero state.

G. The FSM output 0s depending on the tail followed by a 1 at the end of tail decoding.

Table 3-5 The truth table of control signals of 4-16 decoder

Input Signals				**State**	**Output signals**			
Select	GSize[2]	GSize [1]	GSize [0]		Select 1	en_mux	en_i	Data [4:0]
0	0	0	0	Golomb m =2	0	0	0	00001
0	0	0	1	Golomb m =4	0	0	0	00011
0	0	1	0	Golomb m =8	0	0	0	00111
0	0	1	1	Golomb m=16	0	0	0	01111
0	1	0	0	Golomb m=32	0	0	0	11111
1	0	0	0	FDR	1	0	0	0000
1	0	0	1	EFDR	0	0	0	0000
1	0	1	0	IFDR	1	0	1	0000
1	0	1	1	AR	1	1	0	0000

The state diagram for the FSM used for decompression is shown in Figure 3.6 Note that the state diagram consists of only nine states after combining five kinds of decoding methodologies and one bypass mode. The S0 is initial state for defining output signals of FSM in the begging. The S1 is bypass state. The states S2-S4 and S3-S5 are correspond to the Golomb and FDR similar in prefix decoding. The states S6-S8 are tail decoding in any method. Synthesized FSM takes only 9 Flip Flops and decoder takes 37 Flip Flops.

bit_in,rs1,rs2,select,select1,en_FSM,mode/fout,v,en,counter_in,shift1,dec1,inc,dec2,enshift,enmux

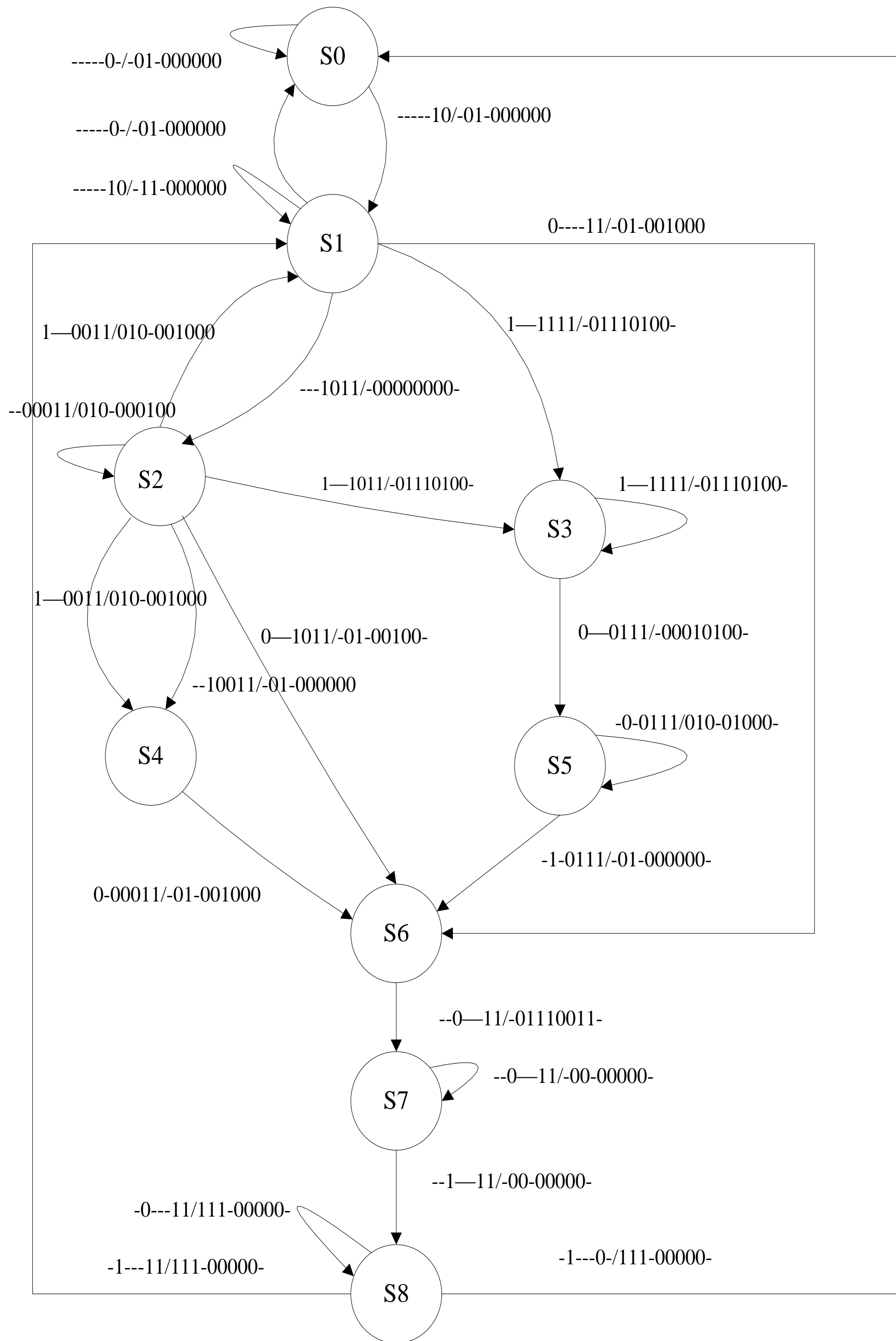


Figure 3.6 State diagram of the FSM

3.6 TOOLS USED

1. **MATLAB 7.5b**: Matlab has been utilized for implementing various compression schemes, WTM model for power estimation and for implementing Double Hamming Distance Based Reordering Technique.
2. **DESIGN VISION**: It has been used for implementing Scan chain insertion and DFT and for synthesizing the Decoder.
3. **TETRAMAX**: It has been used as an ATPG for generating Test vectors for four ISCAS 89 Benchmark circuits.
4. **XILNIX**: For synthesizing the decoder.
5. **MILEF ATPG**: It has been used as an ATPG for generating Test vectors for four ISCAS 89 Benchmark circuits

Chapter 4
RESULTS AND ANALYSIS

4.1 ENCODER RESULTS AND ANALYSIS

We have used MATLAB 7.5b for implementing various compression schemes, Double Hamming Based Reordering technique, various don't care filling techniques and WTM model for Power Estimation for various ISCAS89 benchmark circuits.

Matlab results for test set generated by two ATPG engines TETRAMAX and MILEF are presented and compared for efficiency and power.

We have used SCAN based technique in generating test vectors from TETRAMAX and MILEF is based on MIXED FAN algorithm which works on the fact of making each flip flop in the circuit directly controllable by making it primary input during testing.

Also results from DESIGN VISION which is used for SCAN insertion and DFT for ISCAS benchmark circuits are also presented for Four ISCAS 89 circuits which show how the circuits are synthesized and scan chain is inserted in the circuit.

An complete analysis so as to why proposed scheme is a better approach for test data compression is shown graphically and is also explained thereafter.

Synthesis Results on Design Compiler

Synopsys DC tool version D-2010.03-SP1 is used for synthesis. The Working Environment is :
Operating Condition Name : WCCOM

Library : fsd0c_a_generic_core_ss1p08v125c

Process : 1.00

Temperature : 125.00 ^{0}C

Voltage : 1.08 V

Interconnect Model : Wire load (worst case tree)

Technology: 90 nm

Timing Analysis Effort : Medium

Power Analysis Effort : Low

Wire Load Model : G10K

4.1.1 MATLAB RESULTS AND ANALYSIS FOR MILEF ATPG

4.1.1.1 Circuit s298

Results are presented for ISCAS s298 Benchmark circuit for Power and Compression for different compression and filling techniques in Table 4.1.

Table 4.1 Matlab results for ISCAS 89 s298 circuit for MILEF ATPG patterns

CIRCUIT									
S298	POWER		COMPRESSION						
	AVG	PEAK	GOLOMB	M	FDR	EFDR	AFDR	IFDR	MCC
CBSTD	57.4419	96	-4.56	2	-16.37	-34.93	-14.64	-39.44	6.79
CBSTDDIFF	20.9612	57	46.97	8	43	32.92	34.34	-81.21	55.59
MTFILLTD	14.907	53	-14.09	2	-11.81	28.09	48.02	-28.04	37.53
MTFILLTDDIFF	19.6822	51	-2.74	2	-1.23	21.39	39.99	-41.18	33.42
ZEROFILLTD	25.5039	57	38.39	8	37.8	25.4	25.76	-77.38	49.7
ZEROFILLTDDIFF	30.3178	71	35.57	4	30.87	17.42	19.65	-73.64	41.91
CBS	57.3411	96	-4.74	2	-16.55	-34.79	-14.46	-39.26	6.93
CBSDIFF	20.8915	57	46.79	8	42.91	32.69	33.97	-81.12	55.49
MTFILL	14.907	53	-14.04	2	-11.81	27.95	48.02	-28.04	37.44
MTFILLDIFF	19.6124	51	-1.46	2	0.32	21.16	39.63	-42.82	32.92
ZEROFILL	25.5039	57	38.39	8	37.8	25.4	25.76	-77.38	49.66
ZEROFILLDIFF	30.1705	71	35.48	4	30.78	17.33	19.47	-73.55	41.95
CBSTDDR	57.4419	96	-4.61	2	-16.19	-34.75	-14.36	-39.44	6.84
CBSTDDRDIFF	18.8295	79	54.35	8	49.38	40.77	39.9	-83.77	60.24
MTFILLTDDR	14.907	53	-14.14	2	-11.72	26.77	47.74	-28.23	36.48
MTFILLTDDRDIFF	10.6357	58	60.78	16	61.97	58.28	58.41	-88.14	71.77
ZEROFILLTDDR	25.5039	57	38.26	8	37.8	25.31	25.67	-77.38	49.48
ZEROFILLTDDRDIFF	11.2171	60	67.99	16	63.98	58.87	57.41	-89.97	72.14
CBSDR	57.3411	96	-4.88	2	-16.37	-34.56	-14	-39.08	6.84
CBSDRDIFF	18.4341	75	55.13	8	49.93	41.36	41.09	-84.13	60.51
MTFILLDR	14.907	53	-14.14	2	-11.72	26.77	47.74	-28.23	36.48
MTFILLDRDIFF	10.6357	58	60.78	16	61.97	58.28	58.41	-88.14	71.77
ZEROFILLDR	25.5039	57	38.26	8	37.8	25.31	25.67	-77.38	49.48
ZEROFILLDRDIFF	11.2171	60	67.99	16	63.98	58.87	57.41	-89.97	72.14

As you can see in Figure 4.1 which is a summary of the s298 results in Table 4.1 for scheme with highest CR and minimum average and peak power, we can conclude as under-

Maximum Compression ratio is for ZEROFILLDRDIFF=72.14, however it has higher average and peak power than MTFILLDRDIFF.

With only 0.5% decrease in CR than ZEROFILLDRDIFF and a decrease of 5.183 %, 3.33% for average and peak power respectively for MTFILLDRDIFF we can say that MTFILLDRDIFF is the most optimum scheme for a low power compression scheme.

Also there is an increase of 15.81% from maximum compression obtained for any run length code obtained for the same MTFILLDRDIFF scheme and an 49.45% increase in CR and 28.65% decrease, 9.43% increase in average and peak power respectively if compared with simple MTFILLTD scheme.

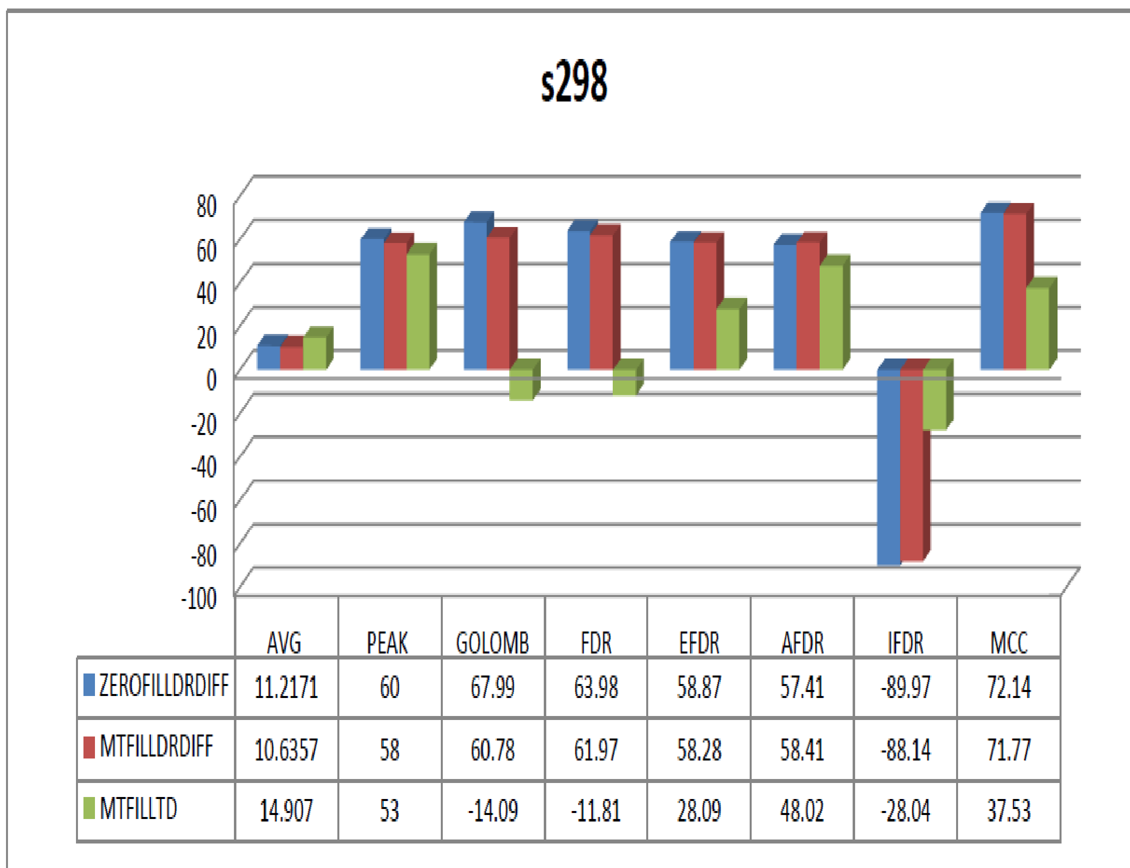


Figure 4.1 Summary of s298 results

4.1.1.2 Circuit s400

Results are presented for ISCAS s298 Benchmark circuit for Power and Compression for different compression and filling techniques in Table 4.2.

Table 4.2 Matlab results for ISCAS 89 s400 circuit for MILEF ATPG patterns

CIRCUIT									
s400	POWER		COMPRESSION						
	AVG	PEAK	GOLOMB	M	FDR	EFDR	AFDR	IFDR	MCC
CBSTD	136.7603	224	1.14	2	-3.94	-15.72	-2.23	-43.84	16.58
CBSTDDIFF	38.5822	111	59.22	16	58.9	52.03	51.88	-87.73	67.78
MTFILLTD	33.1575	95	1.66	2	12.33	45.06	59.02	-45.03	43.46
MTFILLTDDIFF	42.3699	120	-9.08	2	-2.45	38.76	54.68	-32.02	42.52
ZEROFILLTD	47.6986	113	56.36	8	54.51	46.83	47.6	-85.67	63.5
ZEROFILLTDDIFF	53.9795	153	48.23	8	47.6	38.47	40.01	-82.08	58.82
CBS	138.4247	224	0.54	2	-5.25	-17.12	-3.14	-43.26	16.12
CBSDIFF	39.1575	111	58.39	16	58.5	51.4	51.48	-87.5	67.47
MTFILL	33.1575	95	1.63	2	12.39	45.35	59.19	-44.98	43.55
MTFILLDIFF	42.0959	120	-10.64	2	-4.28	38.93	55.25	-30.01	42.78
ZEROFILL	47.6986	113	56.36	8	54.51	46.83	47.6	-85.67	63.53
ZEROFILLDIFF	53.8973	153	47.32	8	47.03	37.93	40.01	-81.91	58.53
CBSTDDR	136.7603	224	1.14	2	-3.71	-15.53	-1.94	-43.89	16.67
CBSTDDRDIFF	35.5137	182	63.07	16	61.64	55.85	54.85	-88.64	70.18
MTFILLTDDR	33.1575	95	2	2	13.76	48.4	61.19	-44.01	45.18
MTFILLTDDRDIFF	25.637	105	69.04	16	69.86	66.55	66.78	-91.1	76.80
ZEROFILLTDDR	47.6989	113	56.76	8	54.74	46.95	48	-85.67	63.81
ZEROFILLTDDRDIFF	25.8562	156	74.97	16	71.35	67.04	65.18	-92.35	77.37
CBSDR	138.4247	224	0.51	2	-5.14	-17.12	-3.2	-43.32	16.10
CBSDRDIFF	35.8699	130	63.5	16	61.82	56.31	54.85	-88.53	70.55
MTFILLDR	33.1575	95	1.86	2	13.76	48.4	61.13	-44.01	45.15
MTFILLDRDIFF	25.0342	105	68.58	16	69.63	66.12	66.67	-91.15	76.51
ZEROFILLDR	47.6989	113	56.76	8	54.85	47.06	48	-85.67	63.81
ZEROFILLDRDIFF	25.911	113	74.89	16	71.46	67.15	65.24	-92.35	77.31

As you can see in Figure 4.2 which is a summary of the s400 results in Table 4.2 for scheme with highest CR and minimum average and peak power, we can conclude as under-

Maximum Compression ratio is for ZEROFILLTDDRDIFF=77.37, however it has higher average and peak power than MTFILLDRDIFF.

With only 1.11% decrease in CR than ZEROFILLTDDRDIFF and a decrease of 3.179 %, 32.69% for average and peak power respectively for MTFILLDRDIFF we can say that MTFILLDRDIFF is the most optimum scheme for a low power compression scheme.

Also there is an increase of 9.40% from maximum compression obtained for any run length code obtained for the same MTFILLDRDIFF scheme and an 29.63% increase in CR and 24.49% decrease, 10.52% increase in average and peak power respectively if compared with simple MTFILLTD scheme.

	AVG	PEAK	GOLOMB	FDR	EFDR	AFDR	IFDR	MCC
ZEROFILLTDDRDIFF	25.8562	156	74.97	71.35	67.04	65.18	-92.35	77.37
MTFILLDRDIFF	25.0342	105	68.58	69.63	66.12	66.67	-91.15	76.51
MTFILLTD	33.1575	95	1.66	12.33	45.06	59.02	-45.03	43.46

Figure 4.2 Summary of s400 results

4.1.1.3 Circuit s1196

Results are presented for ISCAS s1196 Benchmark circuit for Power and Compression for different compression and filling techniques in Table 4.3

Table 4.3 Matlab results for ISCAS 89 s1196 circuit for MILEF ATPG patterns

CIRCUIT									
s1196	POWER		COMPRESSION						
	AVG	PEAK	GOLOMB	M	FDR	EFDR	AFDR	IFDR	MCC
CBSTD	203.2919	343	-16.44	2	-24.04	-15.86	7.82	-29.84	16.07
CBSTDDIFF	125.1012	302	40.9	8	45.97	36.67	38.49	-79.57	56.31
MTFILLTD	103.8902	238	-17.58	2	-13.71	31.06	52.55	-24.44	39.46
MTFILLTDDIFF	127.0231	285	-15.55	2	-12.54	22.59	44	-27.8	34.55
ZEROFILLTD	150.3988	298	31.19	4	34.38	26.35	30.18	-71.73	45.59
ZEROFILLTDDIFF	164.9277	330	23.01	4	27.46	16.37	21.88	-67.83	39.67
CBS	202.8121	325	-17.78	2	-25.23	-16.39	8.11	-28.07	15.8
CBSDIFF	125.0607	302	40.82	8	45.97	36.74	38.58	-79.55	56.3
MTFILL	103.8902	238	-17.49	2	-13.73	30.99	52.55	-24.46	39.44
MTFILLDIFF	127.6561	285	-15.42	2	-12.52	22.25	43.79	-27.93	34.37
ZEROFILL	150.3988	298	31.25	4	34.38	26.32	30.2	-71.73	45.58
ZEROFILLDIFF	165.7081	330	23	4	27.35	16.1	21.64	-67.77	39.5
CBSTDDR	203.2919	343	-16.43	2	-24.06	-15.96	7.71	-29.95	16.03
CBSTDDRDIFF	106.6561	286	49.07	8	48.28	38.76	37.5	-81.61	57.7
MTFILLTDDR	103.8902	238	-17.58	2	-13.69	31.09	52.66	-24.51	39.43
MTFILLTDDRDIFF	66.6387	247	64.22	16	64.79	59.7	60.4	-89.6	73.29
ZEROFILLTDDR	150.3988	298	31.08	4	34.18	26.23	29.93	-71.73	45.75
ZEROFILLTDDRDIFF	66.6647	264	69.21	16	65.41	60.37	58.62	-90.23	73.83
CBSDR	202.8121	325	-17.75	2	-25.2	-16.46	8	-28.16	15.78
CBSDRDIFF	107.5694	284	49.49	8	48.14	38.73	37.84	-81.76	57.71
MTFILLDR	103.8902	238	-17.68	2	-13.67	31.15	52.64	-24.53	39.89
MTFILLDRDIFF	66.3179	247	64.55	16	65.16	60.27	60.71	-89.7	73.57
ZEROFILLDR	150.3988	298	31.1	4	34.28	26.33	30.02	-71.73	45.76
ZEROFILLDRDIFF	66.7688	275	68.8	16	65.39	60.4	58.65	-90.12	73.74

As you can see in Figure 4.3 which is a summary of the s1196 results in Table 4.3 for scheme with highest CR and minimum average and peak power, we can conclude as under-

Maximum Compression ratio, is for ZEROFILLTDDRDIFF=73.83, however it has higher average and peak power than MTFILLDRDIFF.

With only 0.35% decrease in CR than ZEROFILLTDDRDIFF and a decrease of 0.493 %, 6.43% for average and peak power respectively for MTFILLDRDIFF we can say that MTFILLDRDIFF is the most optimum scheme for a low power compression scheme.

Also there is an increase of 12.90% from maximum compression obtained for any run length code obtained for the same MTFILLDRDIFF scheme and a 40% increase in CR and 36.16% decrease, 3.78% increase in average and peak power respectively if compared with simple MTFILLTD scheme.

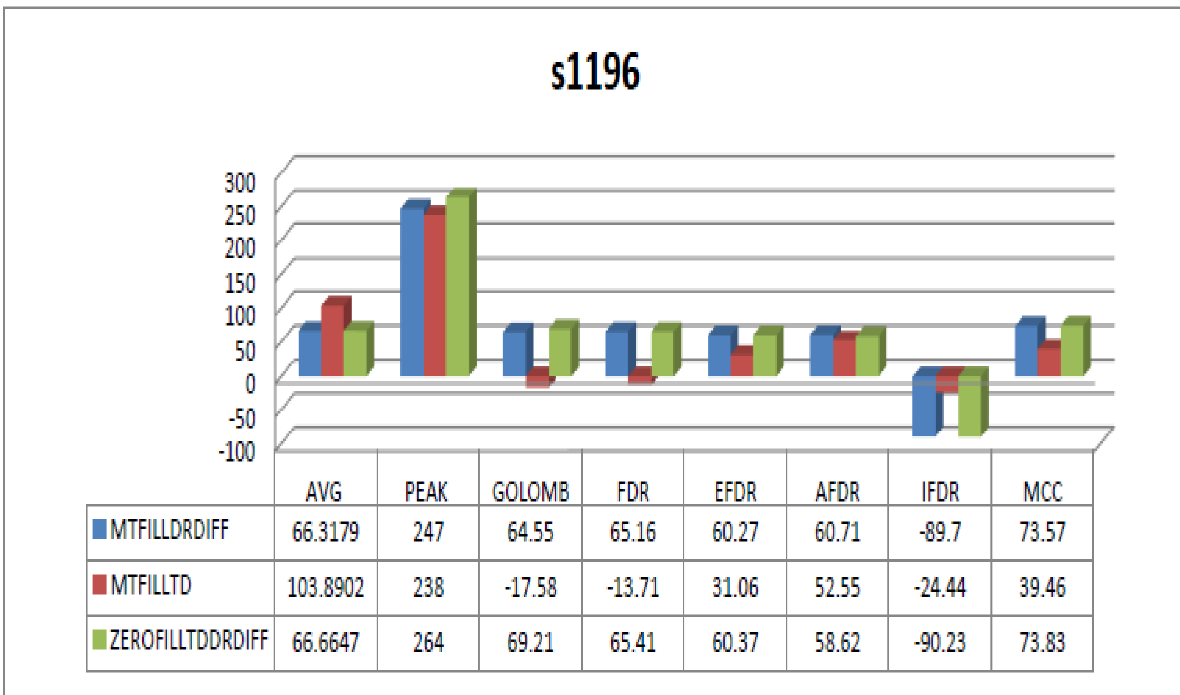


Figure 4.3 Summary of s1196 results

4.1.1.4 Circuit s1494

Results are presented for ISCAS s1494 Benchmark circuit for Power and Compression for different compression and filling techniques in Table 4.4

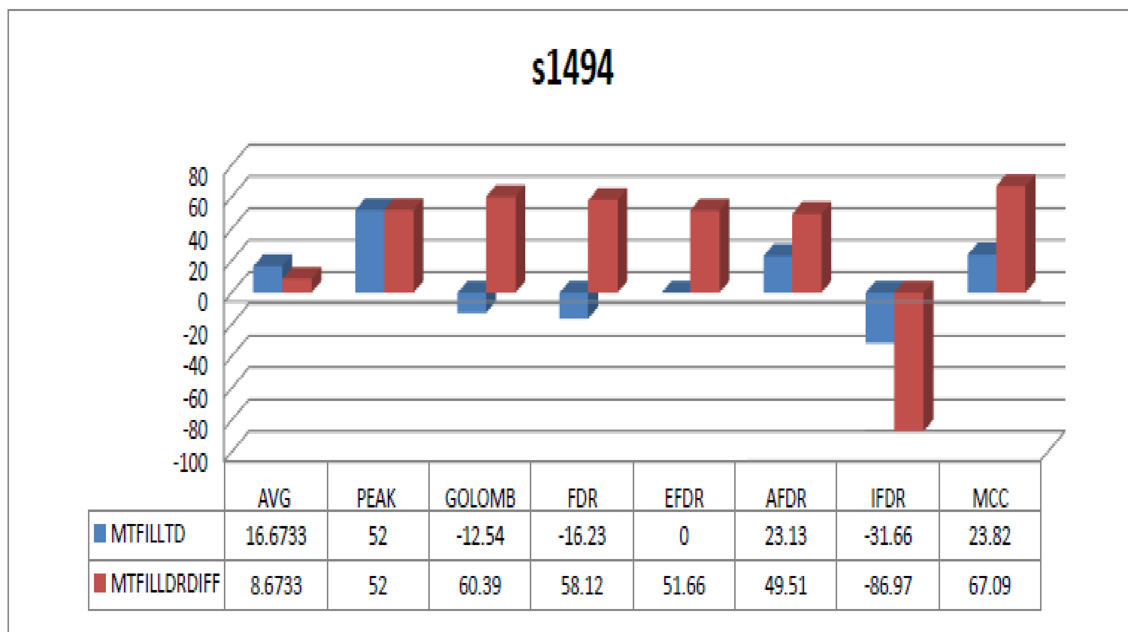
Table 4.4 Matlab results for ISCAS 89 s1494 circuit for MILEF ATPG patterns

CIRCUIT								
S1494	POWER		COMPRESSION					
	AVG	PEAK	GOLOMB	M	FDR	EFDR	AFDR	IFDR
CBSTD	42.9403	88	-24.15	2	-32.6	-32.8	-9.13	-17.9
CBSTDDIFF	21.4233	62	19.32	4	19.76	3.55	9.74	-65.22
MTFILLTD	16.6733	52	-12.54	2	-16.2	0	23.13	-31.66
MTFILLTDDIFF	22.5824	60	-6.35	2	-9.9	-5.19	17	-38.8
ZEROFILLTD	28.517	66	12.24	2	6.74	-10.2	-0.32	-55.8
ZEROFILLTDDIFF	33.3466	75	9.11	2	2.84	-15.7	-3.98	-52.52
CBS	43.2926	88	-24.94	2	-33.5	-33.4	-9.38	-17
CBSDIFF	21.4631	62	19.18	4	19.76	3.84	9.9	-65.18
MTFILL	16.6733	52	-12.54	2	-16.2	0.06	23.13	-31.66
MTFILLDIFF	22.5369	60	-6.11	2	-9.62	-4.85	16.96	-39.04
ZEROFILL	28.517	66	12.22	2	6.7	-10.3	-0.28	-55.8
ZEROFILLDIFF	33.5369	75	8.73	2	2.39	-15.8	-4.22	-52.07
CBSTDDR	42.9403	88	-24.21	2	-32.6	-32.6	9.01	-17.98
CBSTDDRDIFF	18.8381	71	45.11	8	39.81	28.69	25.08	-77.92
MTFILLTDDR	16.6733	52	-12.4	2	-16	-0.22	22.85	-31.57
MTFILLTDDRDIFF	8.4744	58	60.82	16	58.16	51.46	49.03	-86.81
ZEROFILLTDDR	28.517	66	12.44	2	7.06	-10.4	-0.61	-55.48
ZEROFILLTDDRDIFF	11	48	59.76	8	54.38	47.16	44.2	-85.35
CBSDR	43.2926	88	-24.98	2	-33.6	-33.2	-9.21	-17.13
CBSDRDIFF	18.7926	63	45.5	8	40.63	29.71	26.14	-78.45
MTFILLDR	16.6733	52	-12.36	2	-16.2	-0.34	22.81	-31.57
MTFILLDRDIFF	8.6733	52	60.39	16	58.12	51.66	49.51	-86.97
ZEROFILLDR	28.517	66	12.4	2	7.06	-10.5	-0.28	-55.64
ZEROFILLDRDIFF	11.6875	66	59.84	8	53.98	46.45	43.51	-85.19

As you can see in Figure 4.4 which is a summary of the s1494 results in Table 4.4 for scheme with highest CR and minimum average and peak power, we can conclude as under-

Maximum Compression ratio, is for MTFILLDRDIFF=67.09, also it has higher average and less peak power than other schemes.

Also there is an increase of 11.09% from maximum compression obtained for any run length code obtained for the same MTFILLDRDIFF scheme and a 190.05% increase in CR and 47.18% decrease, no change in average and peak power respectively if compared with simple MTFILLTD scheme.



	AVG	PEAK	GOLOMB	FDR	EFDR	AFDR	IFDR	MCC
MTFILLTD	16.6733	52	-12.54	-16.23	0	23.13	-31.66	23.82
MTFILLDRDIFF	8.6733	52	60.39	58.12	51.66	49.51	-86.97	67.09

Figure 4.4 Summary of s1494 results

After analyzing various ISCAS 89 circuits with test set generated by MILEF ATPG we can say that MTFILLDRDIFF i.e. we have first reordered the original un-compacted test set with don’t care bits and then don’t care bits are filled using Minimum Transition Filling approach and then again it is reordered and at last difference vector is taken before being compressed by Multi Compression Code scheme that is proposed in this thesis.

4.1.2 DESIGN VISION SYNTHESISED & DFT INSERTED CIRCUITS

First of all we are presenting a simple D flip flop in Figure 4.5 and a DFT scan cell inserted Flip Flop in Figure 4.6 as this will remain same for all the circuits.

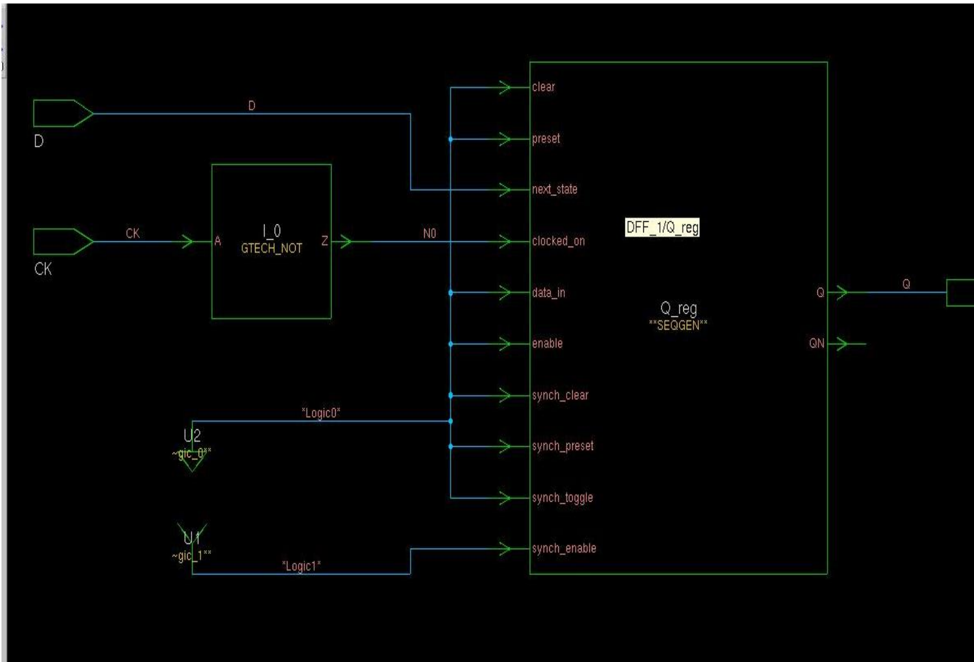


Figure 4.5 A simple D Flip flop

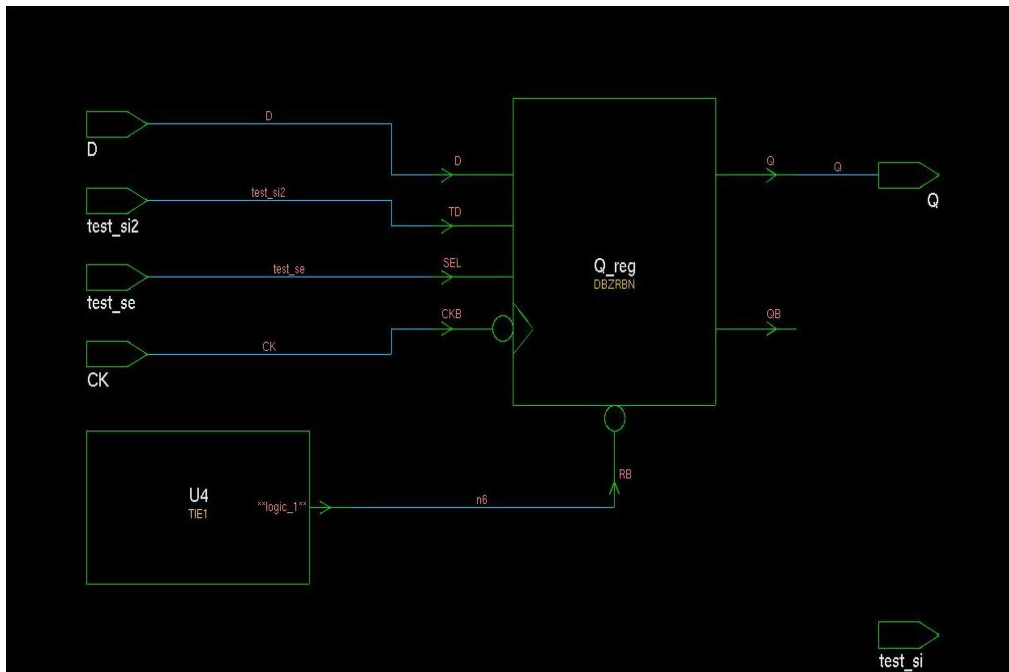


Figure 4.6 A DFT inserted scan Flip flop

4.1.3 MATLAB RESULTS AND ANALYSIS FOR TETRAMAX DATA

The next set of tables describes the results for TETRAMAX generated un-compacted test set for ISCAS 89 circuits. Here we have used word‘t’ along with the circuit name so as to distinguish it from the patterns generated by MILEF ATPG.

4.1.3.1. Circuit s298t

Results are presented for ISCAS s298t Benchmark circuit for Power and Compression for different compression and filling techniques in Table 4.5

Table 4.5 Matlab results for ISCAS 89 s298t circuit for TETRAMAX ATPG patterns

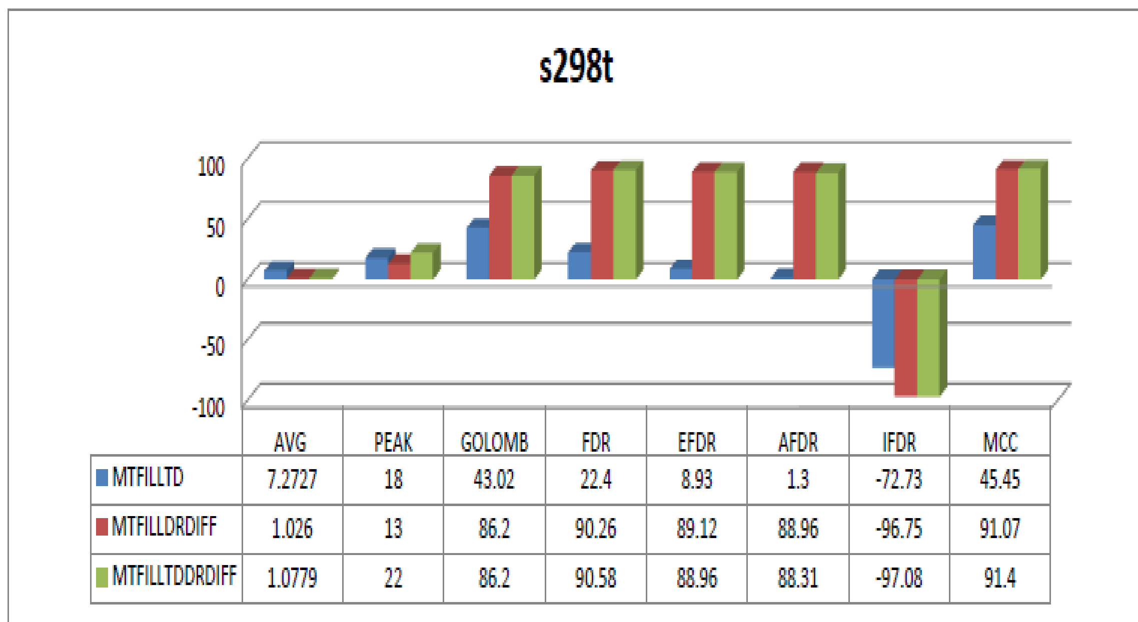
CIRCUIT									
s298t	POWER		COMPRESSION						
	AVG	PEAK	GOLOMB	M	FDR	EFDR	AFDR	IFDR	MCC
CBSTD	14.1169	18	-3.73	2	-9.74	-36.85	-33.44	-40.26	10.88
CBSTDDIFF	2.2857	22	78.41	32	81.82	77.27	78.25	-94.81	82.95
MTFILLTD	7.2727	18	43.02	4	22.4	8.93	1.3	-72.73	45.45
MTFILLTDDIFF	2.4026	22	74.84	32	82.14	76.14	78.9	-93.51	81.98
ZEROFILLTD	7.4156	18	43.51	4	22.73	8.77	0.65	-73.05	45.45
ZEROFILLTDDIFF	2.5325	24	76.79	32	82.79	76.46	78.25	-93.83	82.14
CBS	16.6364	18	-11.85	2	-14.29	-48.05	-39.61	-33.77	2.6
CBSDIFF	2.2597	17	80.36	32	82.14	77.6	77.92	-94.81	83.77
MTFILL	7.2727	18	43.02	4	22.4	8.93	1.3	-72.73	45.45
MTFILLDIFF	2.4026	17	76.14	16	80.84	75.16	79.22	-93.51	80.36
ZEROFILL	7.4156	18	43.51	4	22.73	8.77	0.65	-73.05	45.45
ZEROFILLDIFF	2.5325	24	74.84	32	81.49	75.49	78.57	-93.83	81.49
CBSTDDR	14.1169	18	-3.73	2	-9.74	-37.18	-33.44	-40.26	10.71
CBSTDDRDIFF	1.4935	21	84.74	32	86.36	83.77	83.12	-96.1	87.66
MTFILLTDDR	7.2727	18	42.86	4	21.75	8.93	0.65	-72.73	45.45
MTFILLTDDRDIFF	1.0779	22	86.2	64	90.58	88.96	88.31	-97.08	91.4
ZEROFILLTDDR	7.4156	18	43.34	4	22.08	8.6	0	-73.05	45.29
ZEROFILLTDDRDIFF	1.1558	24	86.2	64	90.26	88.8	88.64	-97.08	91.23
CBSDR	16.6364	18	-11.85	2	-14.29	-48.21	-39.61	-33.77	2.6
CBSDRDIFF	1.0779	17	87.01	64	89.61	85.23	87.34	-97.08	89.94
MTFILLDR	7.2727	18	42.86	4	21.75	8.77	0.65	-72.73	45.29
MTFILLDRDIFF	1.026	13	86.2	64	90.26	89.12	88.96	-96.75	91.07
ZEROFILLDR	7.4156	18	43.34	4	22.08	8.6	0	-73.05	45.29
ZEROFILLDRDIFF	1.1558	24	86.2	64	90.26	88.8	88.64	-97.08	91.23

As you can see in Figure 4.7 which is a summary of the s298t results in Table 4.5 for scheme with highest CR and minimum average and peak power, we can conclude as under-

Maximum Compression ratio, is for MTFILLTDDRDIFF=91.4, however it has higher average and peak power than MTFILLDRDIFF.

With only 0.36% decrease in CR than MTFILLTDDRDIFF and a decrease of 4.8 %, 40.90% for average and peak power respectively for MTFILLDRDIFF we can say that MTFILLDRDIFF is the most optimum scheme for a low power compression scheme.

Also there is an increase of 0.89% from maximum compression obtained for any run length code obtained for the same MTFILLDRDIFF scheme and a 111.69% increase in CR and 85.89% decrease, 27.77% decrease in average and peak power respectively if compared with simple MTFILLTD scheme.



	AVG	PEAK	GOLOMB	FDR	EFDR	AFDR	IFDR	MCC
MTFILLTD	7.2727	18	43.02	22.4	8.93	1.3	-72.73	45.45
MTFILLDRDIFF	1.026	13	86.2	90.26	89.12	88.96	-96.75	91.07
MTFILLTDDRDIFF	1.0779	22	86.2	90.58	88.96	88.31	-97.08	91.4

Figure 4.7 Summary of s298t results

4.1.3.2 Circuit s400t

Results are presented for ISCAS s400t Benchmark circuit for Power and Compression for different compression and filling techniques in Table 4.6

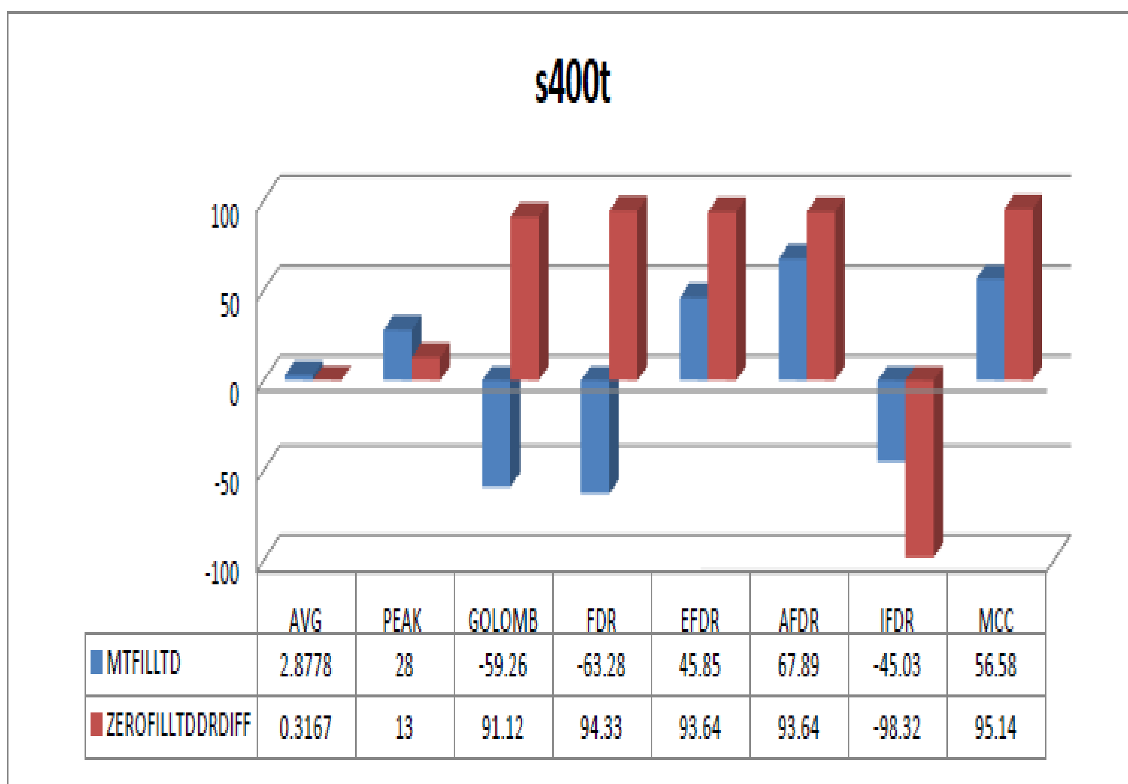
Table 4.6 Matlab results for ISCAS 89 s400t circuit for TETRAMAX ATPG patterns

CIRCUIT									
s400t	POWER		COMPRESSION						
	AVG	PEAK	GOLOMB	M	FDR	EFDR	AFDR	IFDR	MCC
CBSTD	14.3566	28	-32.82	2	-42.27	-29.99	-3.62	-43.84	7.79
CBSTDDIFF	2.8005	28	60.54	8	56.3	49.22	45.82	-87.73	64.56
MTFILLTD	2.8778	28	-59.26	2	-63.28	45.85	67.89	-45.03	56.58
MTFILLTDDIFF	3.9052	20	-0.94	2	9.85	34.2	55.92	-32.02	34.13
ZEROFILLTD	8.3067	28	15.8	4	5.36	-4.18	15.46	-85.67	28.43
ZEROFILLTDDIFF	3.7631	21	54.71	8	50.62	42.18	38.28	-82.08	60.88
CBS	11.818	28	-55.99	2	-62.34	-13.15	24.94	-43.26	16.18
CBSDIFF	2.7556	22	58.04	16	56.42	49.84	46.26	-87.5	65.55
MTFILL	2.8778	28	-59.16	2	-63.15	46.13	68.08	-44.98	56.80
MTFILLDIFF	3.8878	20	-0.72	2	10.1	34.41	56.05	-30.01	34.23
ZEROFILL	8.3067	28	15.8	4	5.36	-4.15	15.46	-85.67	28.40
ZEROFILLDIFF	3.7581	21	54.61	8	50.69	42.8	38.72	-81.91	61.28
CBSTDDR	14.3566	28	-32.82	2	-42.27	-28.93	-3.68	-8.98	8.23
CBSTDDRDIFF	0.7406	18	86.97	64	86.41	85.01	84.23	-96.95	89.78
MTFILLTDDR	2.8778	28	-59.26	2	-63.28	64.31	67.89	42.21	73.82
MTFILLTDDRDIFF	0.4963	25	91.02	64	94.01	93.11	93.02	-98.13	94.48
ZEROFILLTDDR	8.3067	28	15.77	4	5.24	-0.87	15.27	-70.26	28.3
ZEROFILLTDDRDIFF	0.3167	13	91.12	128	94.33	93.64	93.64	-98.32	95.14
CBSDR	11.818	28	-55.95	2	-62.28	-11.85	24.94	7.61	17.27
CBSDRDIFF	0.5511	19	88.53	64	90.02	89.37	89.28	-97.51	92.55
MTFILLDR	2.8778	28	-59.26	2	-63.28	64.31	67.89	42.21	73.78
MTFILLDRDIFF	0.4663	25	91.02	64	94.01	93.14	93.27	-98.19	94.45
ZEROFILLDR	8.3067	28	15.77	4	5.24	-0.81	15.27	-70.39	28.27
ZEROFILLDRDIFF	0.3815	19	91.12	128	94.01	93.27	93.39	-98.25	94.54

As you can see in Figure 4.8 which is a summary of the s400t results in Table 4.6 for scheme with highest CR and minimum average and peak power, we can conclude as under-

Maximum Compression ratio, is for ZEROFILLTDDRDIFF=95.14, however it has lowest average and peak power than any scheme.

Also there is an increase of 0.85% from maximum compression obtained for any run length code obtained for the same ZEROFILLTDDRDIFF scheme and a 40.13% increase in CR and 88.99% decrease, 53.57% decrease in average and peak power respectively if compared with simple MTFILLTD scheme.



	AVG	PEAK	GOLOMB	FDR	EFDR	AFDR	IFDR	MCC
MTFILLTD	2.8778	28	-59.26	-63.28	45.85	67.89	-45.03	56.58
ZEROFILLTDDRDIFF	0.3167	13	91.12	94.33	93.64	93.64	-98.32	95.14

Figure 4.8 Summary of s400t results

4.1.3.3 Circuit s1196t

Results are presented for ISCAS s1196t Benchmark circuit for Power and Compression for different compression and filling techniques in Table 4.7

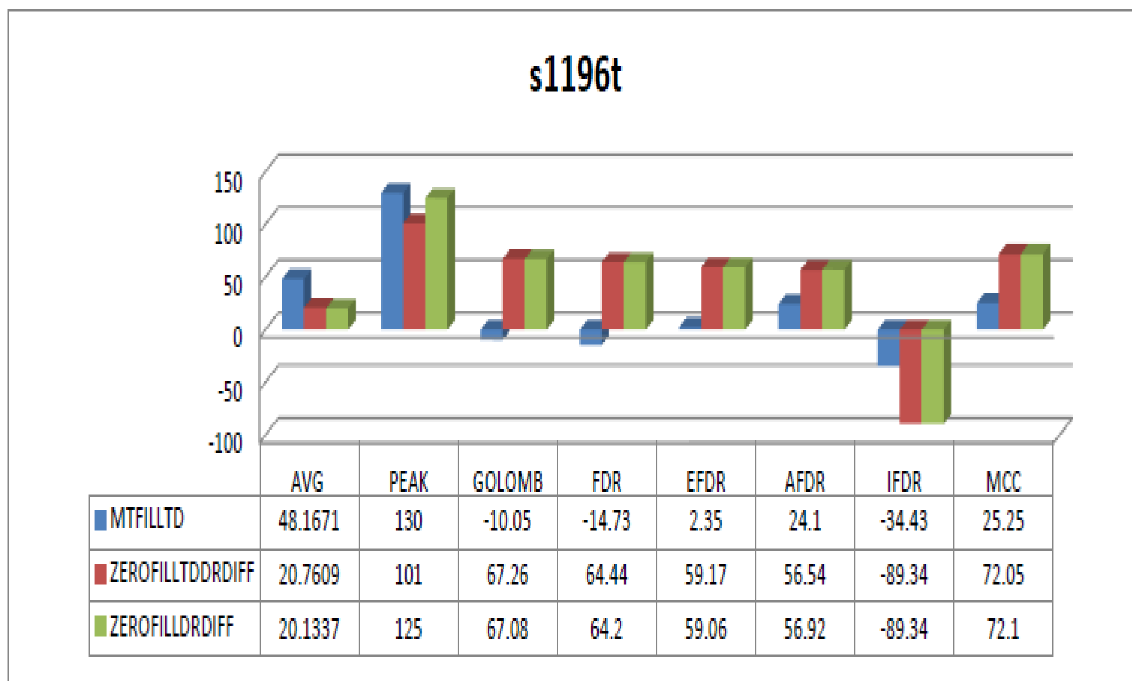
Table 4.7 Matlab results for ISCAS 89 s1196t circuit for TETRAMAX ATPG patterns

CIRCUIT									
s1196t	POWER		COMPRESSION						
	AVG	PEAK	GOLOMB	M	FDR	EFDR	AFDR	IFDR	MCC
CBSTD	87.8046	151	-20.2	2	-29.94	-40.33	-20.69	-23.8	4.83
CBSTDDIFF	43.2622	118	36.19	4	35.33	23.84	26.64	-75.7	45.27
MTFILLTD	48.1671	130	-10.05	2	-14.73	2.35	24.1	-34.43	25.25
MTFILLTDDIFF	50.0437	123	2.29	2	2.56	7.18	24.88	-48.37	25.77
ZEROFILLTD	69.6452	131	12.8	2	6.24	-9.31	-1.23	-56.11	18.83
ZEROFILLTDDIFF	56.5398	153	17.06	4	19.79	7.29	14.6	-64.8	36.04
CBS	88.0308	151	-20.47	2	-30.46	-41.01	-21.15	-23.42	4.55
CBSDIFF	43.0129	120	36.31	4	35.52	24.22	26.91	-75.73	45.42
MTFILL	48.1671	130	-10.05	2	-14.73	2.37	24.01	-34.46	25.27
MTFILLDIFF	50.2519	123	2.31	2	2.5	6.99	24.67	-48.45	25.64
ZEROFILL	69.6452	131	12.76	2	6.21	-9.27	-1.26	-56.11	18.83
ZEROFILLDIFF	56.6838	153	16.72	4	19.52	7.04	14.41	-64.55	35.84
CBSTDDR	87.8046	151	-20.17	2	-29.91	-40.32	-20.66	-23.8	4.82
CBSTDDRDIFF	34.3882	139	52.62	8	48.75	39.58	36.82	-82.36	58.67
MTFILLTDDR	48.1671	130	-10.05	2	-14.73	2.16	24.1	-34.51	25
MTFILLTDDRDIFF	21.6298	123	62.56	16	61.74	56.01	54.51	-88.23	70.07
ZEROFILLTDDR	69.6452	131	12.73	2	6.18	-9.31	-1.31	-56.08	18.83
ZEROFILLTDDRDIFF	20.7609	101	67.26	16	64.44	59.17	56.54	-89.34	72.05
CBSDR	88.0308	151	-20.47	2	-30.46	-41.21	-21.17	-23.39	4.45
CBSDRDIFF	33.9897	110	52.44	8	48.88	39.78	37.52	-82.57	58.52
MTFILLDR	48.1671	130	-10.05	2	-14.73	2.16	24.1	-34.51	25
MTFILLDRDIFF	21.6298	123	62.56	16	61.74	56.01	54.51	-88.23	70.07
ZEROFILLDR	69.6452	131	12.73	2	6.18	-9.31	-1.31	-56.08	18.83
ZEROFILLDRDIFF	20.1337	125	67.08	16	64.2	59.06	56.92	-89.34	72.1

As you can see in Figure 4.9 which is a summary of the s1196t results in Table 4.7 for scheme with highest CR and minimum average and peak power, we can conclude as under-

Maximum Compression ratio, is for ZEROFILLDRDIFF=72.10 , however it has lower average and higher peak power than ZEROFILLTDDRDIFF which has an 19.2% decrease in peak power and an 3.02% increase in average power, so ZEROFILLDRDIFF is for optimum when requirement is to decrease average power and ZEROFILLTDDRDIFF when we want lower peak power ,as such CR of ZEROFILLTDDRDIFF an 0.069% decrease in CR than ZEROFILLDRDIFF..

Also there is an increase of 7.12% from maximum compression obtained for any run length code obtained for the same ZEROFILLTDDRDIFF scheme and a 198.96% increase in CR and 56.89% decrease, 22.30% decrease in average and peak power respectively if compared with simple MTFILLTD scheme.



	AVG	PEAK	GOLOMB	FDR	EFDR	AFDR	IFDR	MCC
MTFILLTD	48.1671	130	-10.05	-14.73	2.35	24.1	-34.43	25.25
ZEROFILLTDDRDIFF	20.7609	101	67.26	64.44	59.17	56.54	-89.34	72.05
ZEROFILLDRDIFF	20.1337	125	67.08	64.2	59.06	56.92	-89.34	72.1

Figure 4.9 Summary of s1196t results

4.1.3.4 Circuit s1494t

Results are presented for ISCAS s1494t Benchmark circuit for Power and Compression for different compression and filling techniques in Table 4.8

Table 4.8 Matlab results for ISCAS 89 s1494t circuit for TETRAMAX ATPG patterns

CIRCUIT									
s1494t	POWER		COMPRESSION						
	AVG	PEAK	GOLOMB	M	FDR	EFDR	AFDR	IFDR	MCC
CBSTD	33.8952	54	-15.38	2	-26.12	-42.65	-16.5	-32.82	4.75
CBSTDDIFF	8.0403	36	69.14	16	63.9	58.37	55.46	-89.58	71.68
MTFILLTD	8.5242	24	-18.8	2	-17.8	25.96	50.93	-25.56	37.53
MTFILLTDDIFF	11.0887	39	0.78	2	9.55	30.49	52.85	-48.45	33.96
ZEROFILLTD	19.2016	44	28.88	4	23.26	6.36	15.38	-73.2	34.68
ZEROFILLTDDIFF	13.9315	54	48.57	8	47.27	38.03	37.47	-81.58	56.64
CBS	33.9194	53	-16.44	2	-27.61	-43.18	-16.38	-31.45	4.5
CBSDIFF	7.9597	42	69.32	16	64.14	58.75	55.89	-89.76	71.99
MTFILL	8.5242	24	-18.73	2	-17.74	26.15	51.05	-25.5	37.66
MTFILLDIFF	10.996	39	1.27	2	10.11	30.68	52.92	-49.01	34.06
ZEROFILL	19.2016	44	28.88	4	23.2	6.36	15.38	-73.2	34.68
ZEROFILLDIFF	13.9234	51	48.36	8	47.08	37.87	37.47	-81.51	56.67
CBSTDDR	33.8952	54	-15.38	2	-26.12	-42.71	-16.5	-32.82	4.71
CBSTDDRDIFF	4.6008	37	78.1	32	75.81	72.77	70.91	-93.67	81.11
MTFILLTDDR	8.5242	24	-18.77	2	-17.74	29.65	50.99	-22.58	41
MTFILLTDDRDIFF	1.371	36	88.18	64	89.95	88.83	89.27	-97.64	91.97
ZEROFILLTDDR	19.2016	44	28.85	4	23.14	6.36	15.32	-73.2	34.68
ZEROFILLTDDRDIFF	2.496	47	87.19	64	87.72	86.2	85.55	-97.08	89.67
CBSDR	33.9194	53	-16.44	2	-27.61	-43.21	-16.38	-31.45	4.5
CBSDRDIFF	4.6169	53	77.7	32	75.43	72.24	71.03	-93.73	81.2
MTFILLDR	8.5242	24	-18.77	2	-17.74	29.65	50.99	-22.58	41
MTFILLDRDIFF	1.371	36	88.18	64	89.95	88.83	89.27	-97.64	91.97
ZEROFILLDR	19.2016	44	28.85	4	23.14	6.36	15.32	-73.2	34.68
ZEROFILLDRDIFF	2.625	53	87.19	64	87.78	86.2	85.48	-97.02	89.8

As you can see in Figure 4.10 which is a summary of the s1494t results in Table 4.8 for scheme with highest CR and minimum average and peak power, we can conclude as under-

Maximum Compression ratio, is for MTFILLTDDRDIFF=91.97, however it has lower average and higher peak power than MTFILLTD.

Also there is an increase of 2.24% from maximum compression obtained for any run length code obtained for the same MTFILLDRDIFF scheme and a 80.58% increase in CR and 83.91% decrease, 50% increase in average and peak power respectively if compared with simple MTFILLTD scheme.

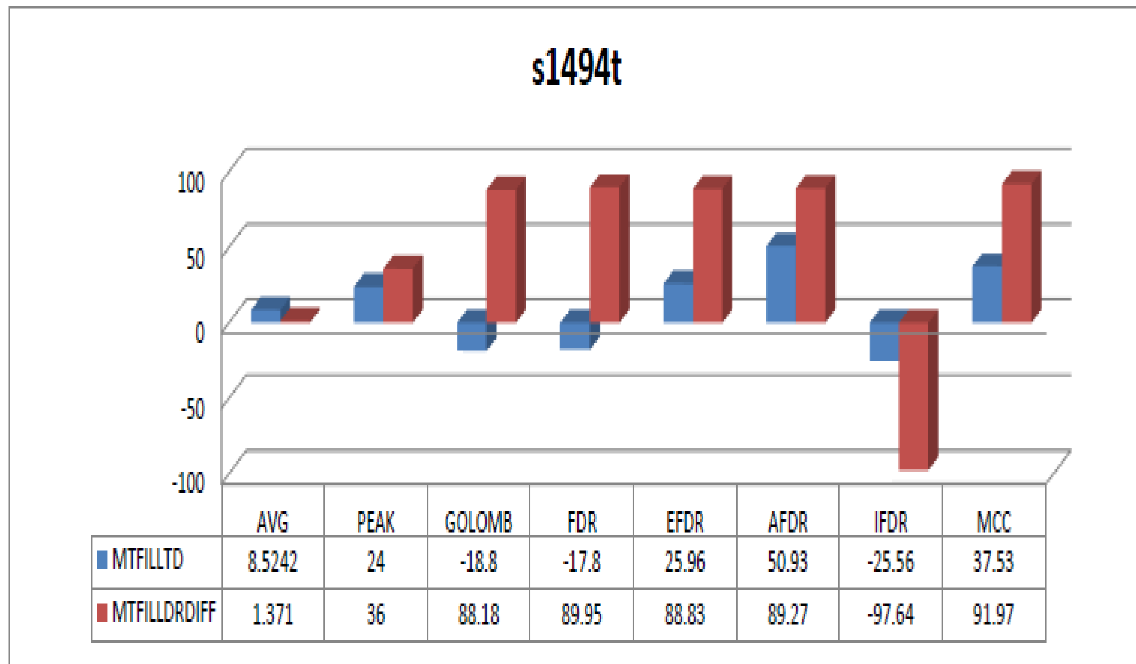


Figure 4.10 Summary of s1494t results

4.2 DECOMPRESSION DECODER RESULTS

We have implemented the decoder using VERILOG HDL (Hardware Description Language) and simulated it on MODELSIM (MENTOR GRAPHICS) and synthesized on DESIGN VISION (SYNOPSYS) and as well as on XILNIX.

The results for area, power and cell count are compared for the two tools namely Design vision and Xilinx.

4.2.1 Design Vision Results

Next we present results showing synthesized MMC FSM used in MCC Decoder.

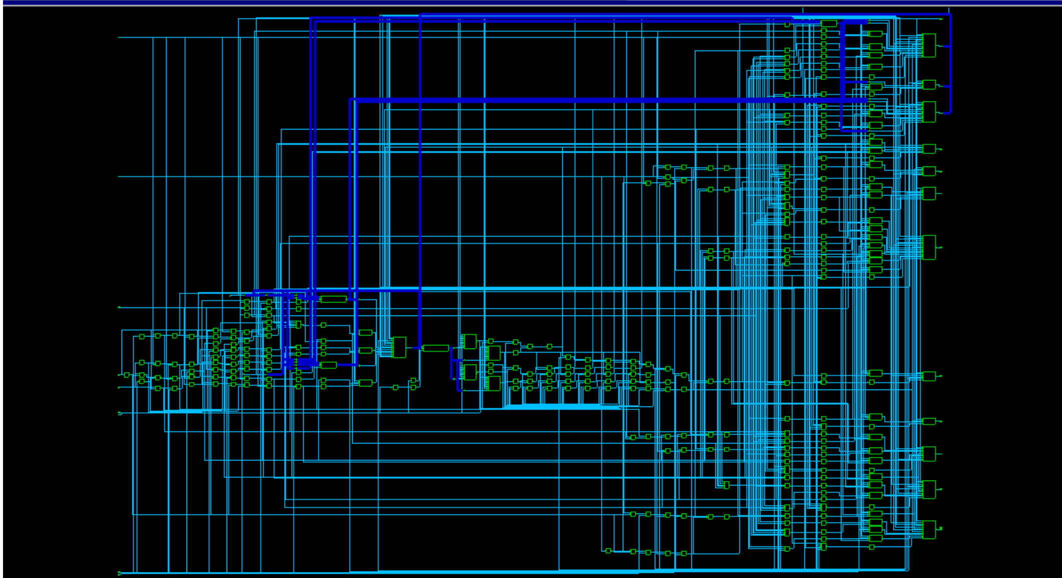


Figure 4.11 RTL for MMC FSM used in MCC Decoder

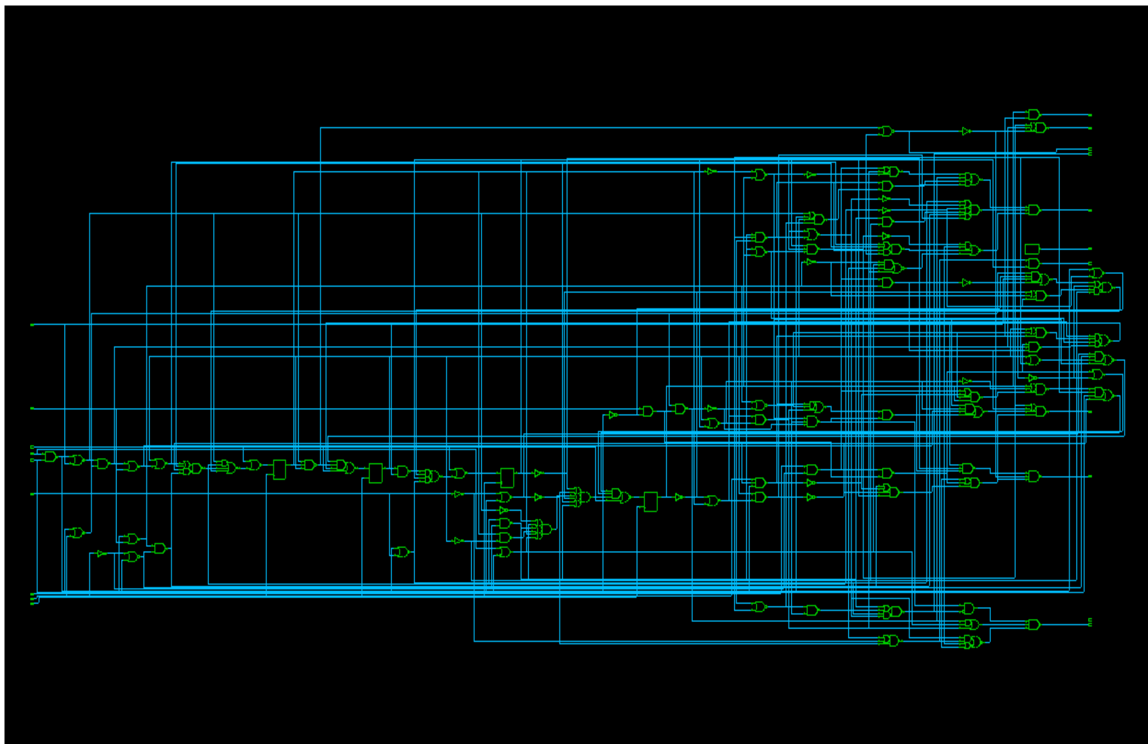


Figure 4.12 RTL for Synthesized MMC FSM used in MCC Decoder

- Net combinational area=1199.922729 um^2
- Non-combinational area=203.112000 um^2
- Total cell area=1403.035156 um^2
- Cell internal power =225.8517 uw (39.1%)
- Net switching power=348.5653 uw (61%)
- Total dynamic power =574.171 uw
- Cell leakage power=121.3074 nw

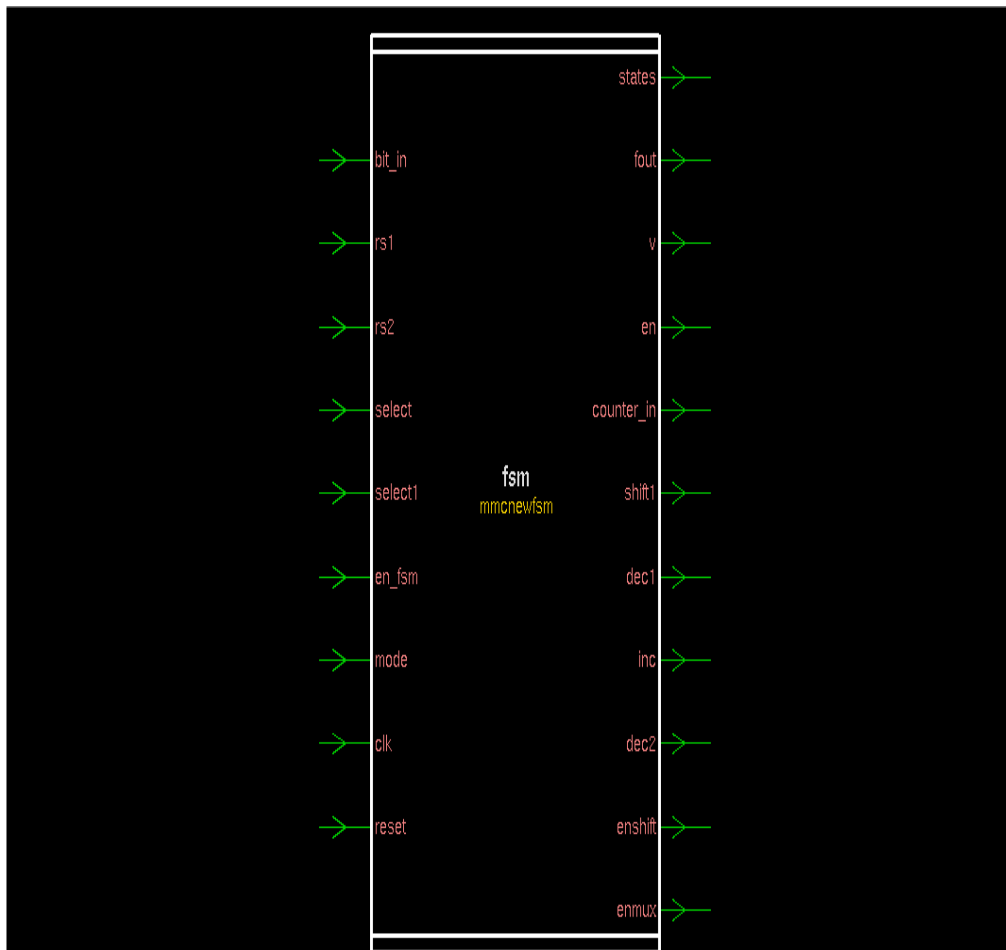


Figure 4.13 Symbol for MMC FSM used in MCC Decoder

Next we present results for synthesized MCC Decoder.

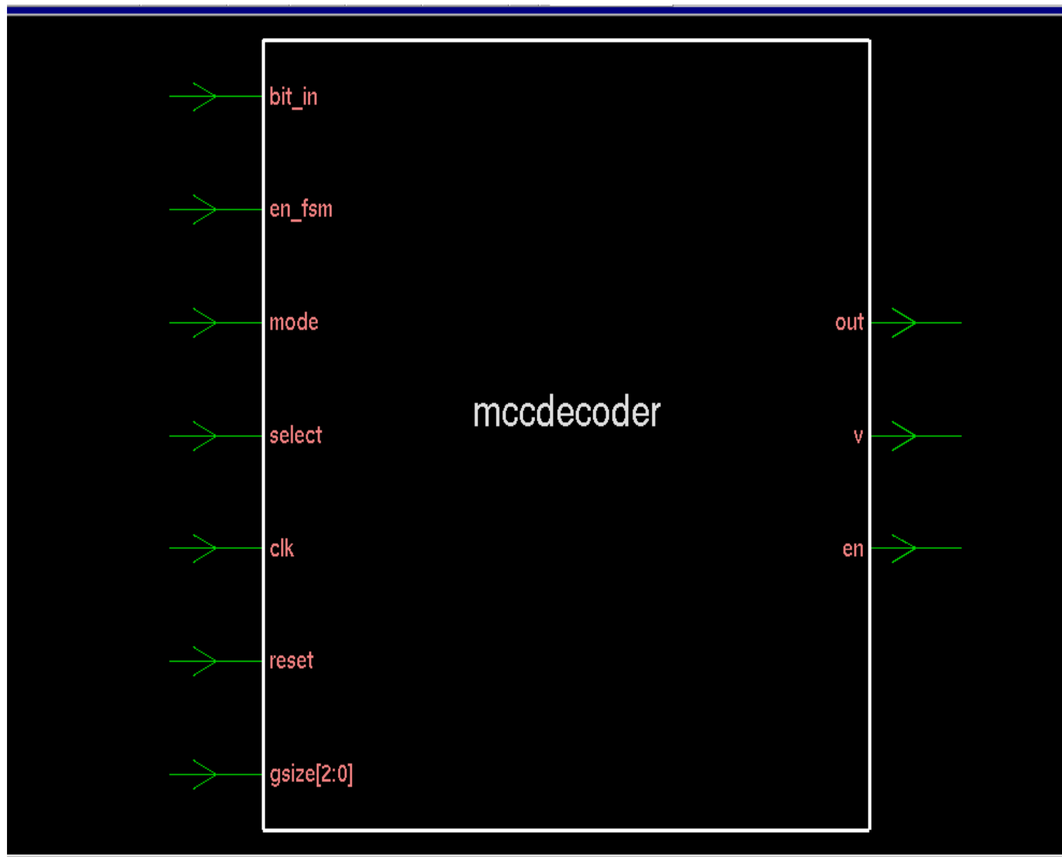


Figure 4.14 Symbol for MCC Decoder

- Net combinational area=3622.266113 um^2
- Non-combinational area=1599.898071 um^2
- Total cell area=5262.163086 um^2
- Cell internal power =97.4477 uw (69.1%)
- Net switching power=43.3866 uw (31%)
- Total dynamic power =140.8344 uw (100%)
- Cell leakage power=572.0903 nw

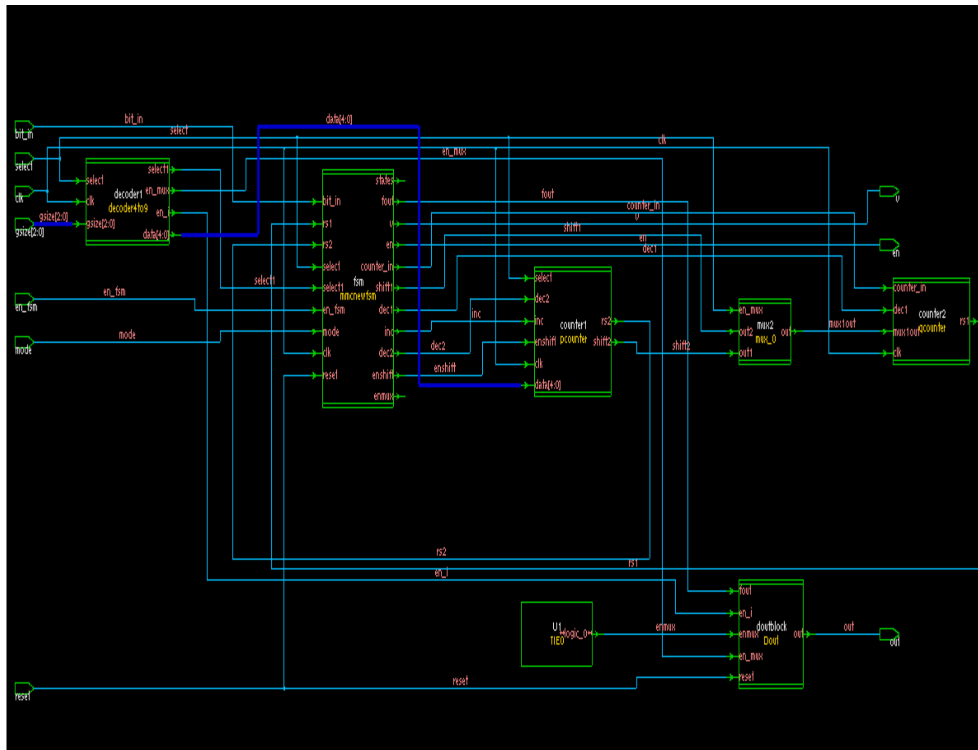


Figure 4.15 RTL for MCC Decoder

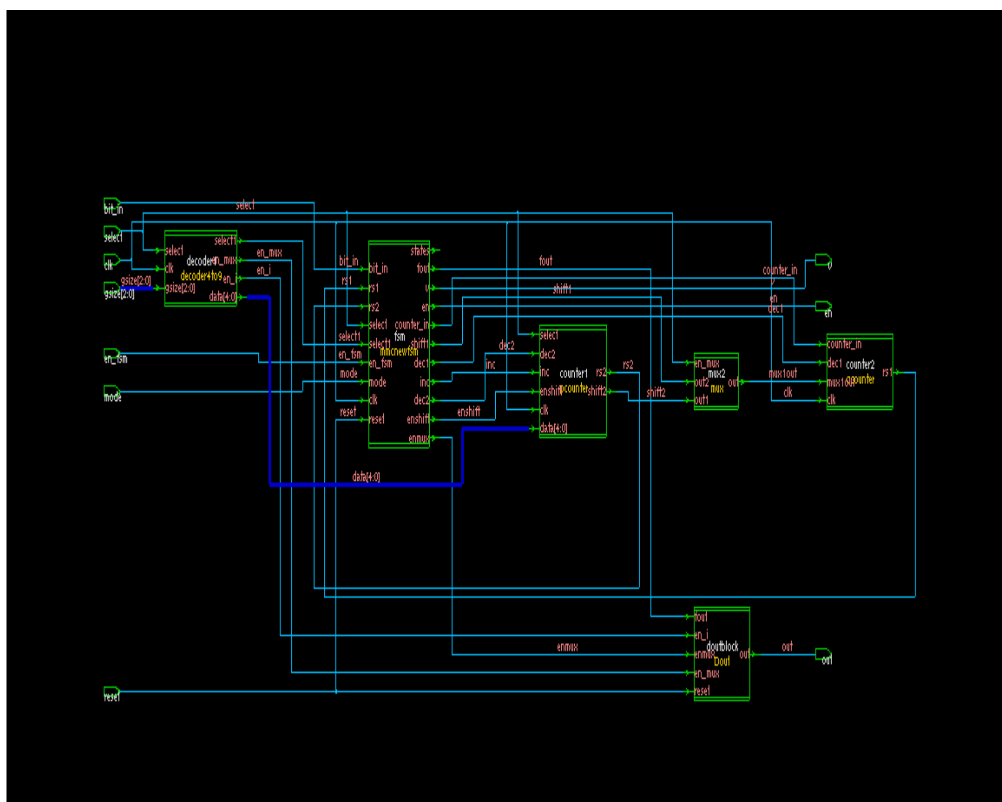


Figure 4.16 Synthesized RTL for MCC Decoder

4.2.2 XILNIX RESULTS

We have synthesized MCC Decoder and MMC FSM on XILNIX for Spartan 3e Kit. Results for the same are presented in this section.

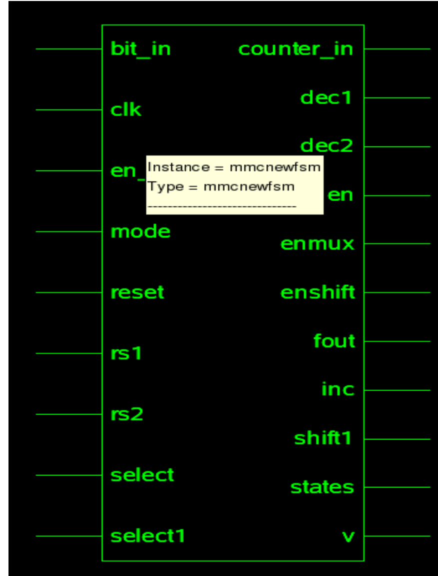


Figure 4.17 Symbol for MMC FSM used in MCC Decoder

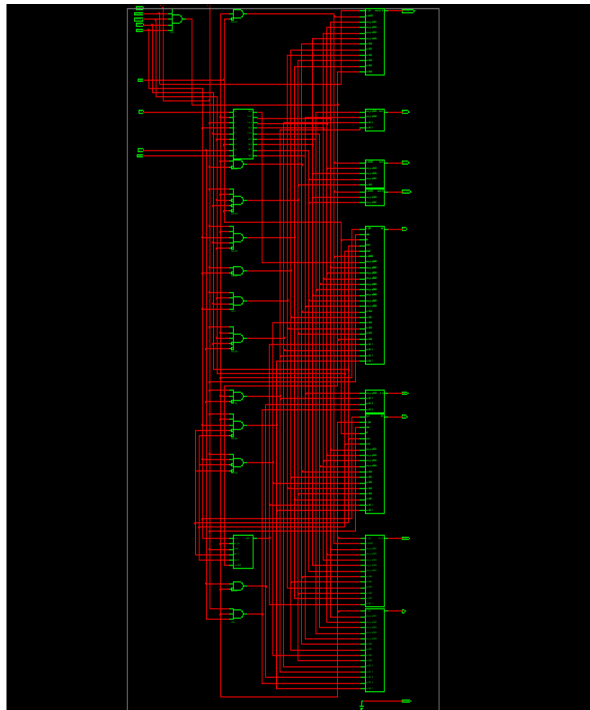


Figure 4.18 Synthesized MMC FSM used in MCC Decoder in XILNIX

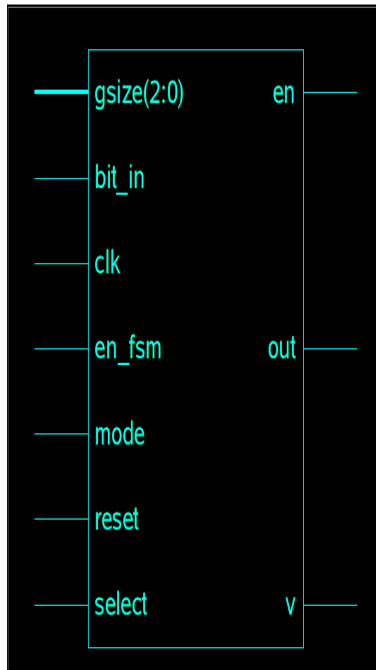


Figure 4.19 Symbol for MCC Decoder in XILNIX

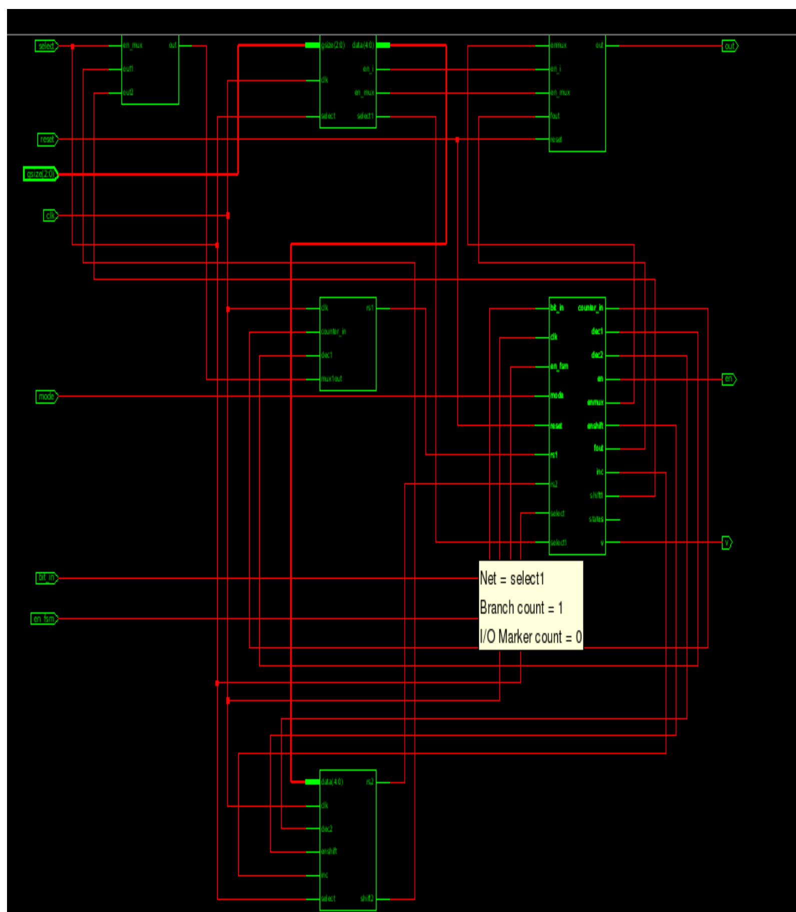


Figure 4.20 Synthesized RTL for MCC Decoder in XILNIX

Synthesize report generated by XILNIX for MCC DECODER

No of Flip Flops used=37

No of slices=75

Synthesis report generated by XILNIX for MMC FSM used in MCC DECODER

No of states=9

No of transitions =24

No of Inputs=7

No of outputs=9

No of Flip Flops used=9

No of slices=29

Synthesis Results working environment

The Xilinx ISE 12.4 is used for implementation of all the circuits. The Working Environment for the design is :
Target Device : XC3S500E-4FG320
Technology : 90 nm
Tool Version : ISE 12.4
FSM Style : LUT
Optimization Goal : Speed
Design Strategy : Balanced

Chapter 5

CONCLUSION AND FUTURE WORK

I have presented a test set compression method and a decompression architecture for testing embedded cores in an SOC. The proposed method is MCC codes which integrates Golomb, FDR, EFDR, AR and IFDR codes with double Hamming Distance Reordering technique to save both testing power and have a better compression efficiency so as to compressing test cubes having a negative weight.

It is shown that MCC codes with Double Hamming Distance Reordering technique can be used for efficient compression of test data for SOC's and to save ATE memory and testing time. The on chip decompression decoder is small and easy to implement. In addition it is scalable and independent of the core under test and utilizes only 37 Flip Flops with total cell area of 5262.163086 um^2.

I have also compared test vectors generated from two ATPG engine's i.e. TETRAMAX(SYNOPSYS) and MILEF and have concluded that MTFILL technique using my Multi Code Compression Scheme along with double reordered scheme and difference vector can achieve about 40-200% increment in Compression Ratio and a 30-50% decrease in average and peak power with most of the ISCAS89 Benchmark circuits compared with a single code compression scheme using original Test data generated for both the ATPG engines namely TETRAMAX and MILEF.

Experimental results for ISCAS89 benchmarks show that the compression scheme along with don't care filling and double hamming distance reordering technique and difference vector give better results than any other run length codes. Also decoders applied in the decompression engine have area over head that is not of much a problem.

I have implemented Decoders on both XILNIX and Design Vision for and their results for area, power, and cell count have been compared.

Our future work is to integrate Heterogeneous codes to obtain higher compression ratio and to study the timing specifications for this circuit.

REFERENCES

[1] C.-W. Wu, J.-F. Li and C.-T. Huang, "Core-Based System-on-Chip Testing: Challenges and Opportunities", in Journal of The Chinese Institute of Electrical Engineering, vol. 8, no. 4, pp. 335 – 353, 2001.

[2] Y. Zorian, E.-J. Marinissen, and S. Dey, "Testing Embedded-Core-Based System Chips", in Proc. International Test Conference, pp. 130 – 143, 1998.

[3] P.K.Lala, Digital Circuit Testing and Testability, Academic Press, 1997.

[4] M. Schulz, E. Trischhler, and T.Sarfert, "SOCRATES: A Highly Efficient Automatic Test Pattern Generation System," in IEEE Transactions on Computer-Aided Design, pp. 126 – 137, Jan. 1988.

[5] I.Pomeranz, L. Reddy, and S. Reddy, "COMPACTEST: A Method to Generate Compact Test Sets for Combinational Circuits", in Proc. Test International Conference, pp. 194 – 203, 1991.

[6] Usha S. Mehta, Kankar S. Dasgupta, Niranjan M. Devashrayee "Hamming Distance Based Reordering and Column wise Bit Stuffing with Difference Vector: A Better Scheme for Test Data Compression with Run Length Based Codes" , 23rd International Conference on VLSI Design, 2010.

[7] M.Abromovici, M.A.Breuer and A.D. Friedman, Digital System Testing and Testable Design, New York, Computer science press, 1990.

[8] Alan Gibbson, Algorithmic Graph theory, Cambridge University press, 1985

[9] T. J. Yamaguchi, M. Tilger, M. Ishida, and D. S. Ha, "An Efficient Method for Compressing Test Data", in Proc. International Test Conference, pp. 79 - 88, Nov.1997.

[10] T. J. Yamaguchi, D. S. Ha, M. Ishida, and T. Ohmi, "A Method for Compressing Test Data Based on Burrows-Wheeler Transformation", in IEEE Transactions on Computers, vol. 51, no. 5, pp. 486 – 497, May 2002.

[11] Ranganathan Sankaralingam, Rama Rao Oruganti, and Nur A. Touba “Static Compaction Techniques to Control Scan Vector Power Dissipation” VTS 2000.

[12] W.-D. Tseng” Scan chain ordering technique for switching activity reduction during scan test” IEE Proceedings online no. 20045139.

[13] A. Jas, J. Ghosh-Dastidar, and N.-A. Touba, “Scan Vector Compression / Decompression Using Statistical Coding”, in Proc. 17th IEEE VLSI Test Symposium, pp. 114 – 120, April 1999.

[14] P. T. Gonciari, B. M. Al-Hashimi, and N. Nicolici, “Improving Compression Ratio, Area Overhead, and Test Application Time for System-on-a-Chip Test Data Compression/Decompression”, in Proc. Design, Automation and Test in Europe Conference and Exhibition, pp. 604 – 611, March 2002.

[15] A. Jas, J. Ghosh-Dastidar, M.-E. Ng, and N.-A. Touba, “An Efficient Test Vector Compression Scheme Using Selective Huffman Coding”, in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 22 no. 6 June 2003.

[16] D. Das and N. A. Touba, “Reducing Test Data Volume Using External/LBIST Hybrid Test Patterns” in Proc. Test International Conference, pp. 115 -122, 2000.

[17] A. Jas, C. V. Krishna, and N. A, Touba, “Hybrid BIST Based on Weighted Pseudo-Random Testing: A new Test Resource Partitioning Scheme”, in IEEE VLSI Test Symposium, pp. 114 – 120, 2001.

[18] C. V. Krishna, A. Jas, and N. A. Touba, “Test Vector Encoding Using Partial LFSR reseeding”, in Proc. Test International Conference, pp. 885 – 893, 2001.

[19] A. Chandra, and K. Chakrabarty, “System-On-A-Chip Test-Data Compression and Decompression Architectures Based on Golomb Codes”, in IEEE Transactions, Computer-Aided Design of Integrated Circuits and Systems, vol. 20, no. 3, pp. 355 – 368, March 2001.

[20] A. Chandra, and K. Chakrabarty, “Frequency-Directed Run-Length (FDR) Codes with Application to System-on-a-Chip Test Data Compression”, in Proc. 19th IEEE, VLSI Test Symposium. pp. 42 – 47, May 2001.

[21] A.-H. El-Maleh, and R.-H. Al-Abaji,” Extended Frequency-Directed Run-Length Code With Improved Application to System-on-A-Chip Test Data Compression”, in 9th

International Conference, Electronics, Circuits and Systems, vol. 2, pp. 449 – 452, Sept. 2002.

[22] A. Chandra, and K. Chakrabarty, “A Unified Approach to Reduce SOC Test Data Volume, Scan Power and Testing Time”, in IEEE Transactions, Computer-Aided Design of Integrated Circuits and Systems, vol. 22, no. 3, pp. 352 - 363, March 2003.

[23] M.L.Bushnell and V.D. Agrawal, Essentials of Electronic Testing for Digital Memory and Mixed Signal VLSI Circuits, Kluwer Academic Publishers, London, 2000.

[24] K.Paramasivam Dr.K.Gunavathi “Reordering Algorithm for Minimizing Test Power in VLSI Circuits” Engineering Letters, 14:1, EL_14_1_15 (Advance online publication: 12 February 2007)

[25] H. Fang, C. Tong and X. Cheng, “RunBasedReordering: A Novel Approach for Test Data Compression and Scan Power” ASP-DAC '07: Proceedings of the 2007 conference on Asia South Pacific design, automation, January 2007.

[26] A. Chandra and K. Chakrabarty, “Efficient test data compression and decompression for system-on-a-chip using internal scan chains and Golomb coding”, DATE '01: Proceedings of the conference on Design, automation and test in Europe, March 2001

[27] A. Jas, J. and N.-A. Touba, “Test Vector Decompression via Cyclical Scan Chains and Its Application to Testing Core-Based Designs”, in Proc. Test International Conference, pp. 458 – 464, 1998.

[28] K.-J. Lee, J.-J. Chen, C.-H. Huang, “Using a Single Input to Support Multiple Scan Chains”, Proceedings of ICCAD, pp. 74 – 78, November 1998.

[29] B. T. Murray and J. P. Hayes, “Testing ICs: Getting to the core of the problem,” Computer, vol. 29, pp. 32–38, Nov. 1996.

[30] T. Yamaguchi, M. Tilgner, M. Ishida, and D. S. Ha, “An efficient method for compressing test data,” in Proc. Int. Test Conference, Nov. 1997, pp. 79–88.

[31] M. Ishida, D. S. Ha, and T. Yamaguchi, “COMPACT: A hybrid method for compressing test data,” in Proc. IEEE VLSI Test Symposium.