Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

FAULT DETECTION WITH OPTIMUM MARCH TEST ALGORITHM

[1,2]NOR AZURA ZAKARIA, [3]W.Z.W. HASSAN, [4]I.A. HALIN, [5]R.M. SIDEK, , [6]XIAOQING WEN

[1]Master of Science student, Department of Electrical and Electronic Engineering, Universiti Putra Malaysia, Serdang Selangor, Malaysia
[2]Integrated Circuit Development, PDSI, MIMOS Berhad, Kuala Lumpur, Malaysia
[3]Dr, Senior Lecturer, Department of Electrical and Electronic Engineering, Universiti Putra Malaysia, Serdang Selangor, Malaysia
[4]Dr, Senior Lecturer, Department of Electrical and Electronic Engineering, Universiti Putra Malaysia, Serdang Selangor, Malaysia
[5]Associate Professor, Dr, Senior Lecturer, Department of Electrical and Electronic Engineering, Universiti Putra Malaysia, Serdang Selangor, Malaysia
[6]Professor, Department of Computer Systems and Engineering, Kyushu Institute of Technology Iizuka, Fukuoka, Japan

E-mail: [1]norazura.z@gmail.com, [2]norazura@mimos.my, [3]wan@eng.upm.edu.my , [4]izhal@eng.upm.edu.my, [5]roslina@eng.upm.edu.my, [6]wen@cse.kyutech.ac.jp

ABSTRACT

This paper presents a research work aimed to detect previously-undetected faults, either Write Disturb Faults (WDFs) or Deceptive Read Destructive Faults (DRDFs) or both in March Algorithm such as MATS++(6N), March C-(10N), March SR(14N), and March CL(12N). The main focus of this research is to improve fault coverage on Single Cell Faults as well as Static Double Cell Faults detection, using specified test algorithm. Transition Coupling Faults (CFtrs), Write Destructive Coupling Faults (CFwds) and Deceptive Read Destructive Faults (CFdrds) are types of faults mainly used in this research. The experiment result published in [1] shows BIST (Built-In-Self-Test) implementation with the new algorithm. It provides the same test length but with bigger area overhead, we therefore proposed a new 14N March Test Algorithm with fault coverage of more than 95% using solid 0s and 1s Data Background (DB). This paper reveals the design methodology to generate DB covers all memories function by applying non-transition data, transition data, and single read and double read data. The automation hardware was designed to give the flexibility to the user to generate other new March Algorithm prior to the selected algorithm and analyzed the performance in terms of fault detection and power consumption.

Keywords: *Deceptive Read Destructive Faults; Write Disturb Faults; March Test Algorithm*

1. INTRODUCTION

Nowadays Static Random Access Memory (SRAM) has become an indispensable component of digital systems, which can be used as a standalone product or embedded memories in a System on Chip (SoC) product. The number of SRAM cores in SoC is increasing dramatically because of the design requirement of facilitating multiple applications especially in communication field. This leads to ever-higher density and ever-larger die sizes. On the other hand, technology scaling leads to smaller feature sizes while enabling a huge number of transistors to be fabricated into a single chip. However, such technology scaling also leads to higher risk of unknown defects that randomly occur in such a huge number of memory cells. Therefore, fault diagnosis and debugging of SRAM are complicated and required efficient test algorithms.

In the industry, test algorithms with 10N to 14N test operations, such as March C- (10N), March2 (14N) [17], March SR (14N), and March CL (12N), are usually used for testing memories in SoC. It has been shown that March SS with 22N [2] operations can detect all Static Single Cell Faults (SSCFs) and Static Double Cell Faults (SDCFs). Similarly, March MSS with 18N operations [7] can detect all unlinked faults of SSCFs and SDCFs. However, the

© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

number of test operations of these algorithms is too large for SoC since memories occupy around 70% to 90% of the chip area and thus the area available for Built-in-Self-Test (BIST) is limited. The motivation is to work on developing the new Algorithm with more than 95% fault coverage. Hence, our BIST hardware can achieve memory test algorithms of up to 14N test operations.

Referring to the discussion in Section 2.B and based on our analysis towards Functional Fault Primitive (FFP) and studies on SSCFs detection by well-known algorithms such as MATS++(6N), March C-(10N), March SR (14N) and March CL(12N), the following were concluded: March C- and MATS++ algorithms cannot detect DRDFs and WDFs, both March SR and March CL cannot detect WDFs and March CL algorithm can only detect DRDF0. This occurrence are due to the test sequence of the test algorithm inability to fulfill their Functional Fault Primitives (FFPs) for both or one of them, referring to FFP for WDF = ***(<0w0/1/->, <1w1/0/->)*** which represent WDF0 and WDF1 respectively and FFPs for DRDF= ***(<r0/↑/1>, <r1/↓/0***>), which represent DRDF0 and DRDF1 respectively [5][6]. Previous works only addressed the testing of DCFs with shorter time, but did not discuss how to improve the testing of DRDFs and WDFs [5-9].

To improve fault detection, an automation program was developed to generate a modified March-Test algorithm based on the sequence operation (SQ) generation rule scheme. Proposed SQ generation rule indicate the behavior of missing sequence at the existing Algorithm to detect the fault. Firstly, DB generator programmable was designed and based on a new list of DBs, the new 12N and 14N March Algorithm was generated. The new March Algorithm was modified by replacing the DB value inside the Algorithm. Compared to the previous work, the development of generating DB is optimized at 30N in March SAM with DBs sequence of 00,11,01 and 10. But in this current work, the focus of the proposition solution is to cover undetected fault at the SSCFs at optimum length of Algorithm. Since the rule step criteria is to have transition and non-transition writing operation, the outcome at SDCFs detection, particularly at CFwd, CFdrd and CFtr need to be evaluated. In [9], the detection of CFdr whereby it requires two times read operations; first read operation sensitizes the fault, while the second read operation will detect the fault. During CFdrd detection, other fault CFst, CFir and CFrd can also be detected, however these faults only requires one read operation. Data Background generator reported in [9], although designed to program for different sizes and different kinds of data background for testing WOMs, only covers conventional fault.

This paper is organized as follows: Section 2 shows the FFP list for all faults and its notation. This section also elaborates the analysis of undetectable SSCFs of selected March Test Algorithms. In Section 3, the methodology steps taken in this research work was explained. It covers the proposed solution and the development of Data Background Generator hardware based on SQ generation rule. The development of new March Test algorithm from DB SQ list is discussed in detail. Section 4 elaborates the evaluation of new March Test Algorithm towards fault detection and power performance estimation. Section 5 discusses the fault coverage compare to its original March Test algorithm. Section VI concludes this paper.

2. BACKGROUND

The SRAM testing can be classified into two categories, namely the testing of Single Cell Faults (SCFs) and the testing of Double Cell Faults (DCFs), which can be further sub-divided into dynamic faults and static faults, respectively as shown in Table 1. There are six types of static SCFs (SSCFs), which are Transition Faults (TFs), State Faults (SFs), Write Disturb Faults (WDFs), Read Disturb Faults (RDFs), Incorrect Read Faults (IRFs), and Deceptive Read Disturb Faults (DRDFs) [2-6]. For static DCFs (SDCFs), our focus is on Write Destructive Coupling Faults (CFwds), Deceptive Read Destructive Faults (CFdrds) and Transition Coupling Faults (CFtrs).

Table 1: Functional Fault Primitives

Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Single Cell Faults	
Functional Fault Model	**Fault Primitives**
SF	<1/0/->, <0/1/->
TF	<0w1/0/->, <1w0/1/->
WDF	<0w0/↑/->, <1w1/↓/->
RDF	<r0/↑/1>, r1/↓/0>
DRDF	<r0/↑/0>, r1/↓/1>
IRF	<r0/0/1>, r1/1/0>

Double Cell Faults	
Functional Fault Model	**Fault Primitives**
CFwd	<0;0w0/↑/->, <1;0w0/↑/->, <0;1w1/↓/->, <1;1w1/↓/->
CFdrd	<0;r0/↑/0>, <1;r0/↑/0>, <0;r1/↓/1>,1; r1/↓/1>
CFtr	<0;0w1/0/->, <1;0w1/0/->, <0;1w0/1/->, <1;1w0/1/->

2.1. Fault Notation

In order to describe these faults, a compact notation is referred as *Fault Primitive (FP)* as shown in Table I. All the notations and fault behaviours are discussed in [4]. The notation of FP is explained below:

1) <S/F/R > or < S/F/R >$_v$) denotes a Fault Primitive definition involving a *single cell;* the cell c_v (victim cell) used for sensitizing a fault is the same as where the fault appears. *S* describes the *sensitizing* operation or state; *S* {0,1, w0, w1, w, w r0, r1} whereby 0 denotes a 0 value, 1 denotes a 1 value, w0 (w1) denotes write 0 (1) operation, w ↑(w↓**)** denotes an up (down) transition write operation, and *r0* (r1) denotes a read 0 (1) operation.

2) <S_a;S_v,/F/R> (or <S_a;S_v,/F/R>$_{a,v}$): denotes a FP involving *two cells;* S_a denotes the sensitizing operation or state of the *aggressor cell (a-cell);* while S_v denotes the sensitizing operation or state of the *victim cell (v-cell).* The a-cell *(c_a)* is the cell sensitizing a fault in another cell called the v-cell (c_v). The set of S_i is defined as: S_i ∈{ *0,1, X, w0, wl, w↑, w↓, r0, rl}(i* ∈ *{a,v))*, whereby X is the 'don't care' value *X* ∈ {0,1}.

3) In both notations, *F* denotes the *faulty* value of the victim cell (v-cell); The faulty value in test given as *F* ∈{0,l,↑,↓,?}, consists of ↑(↓), which denotes an up (down) transition and "?", which denotes an undefined logical value. R denotes the logical value which appears at the output of the SRAM if the sensitizing operation applied to the v-cell is a *read* operation; its test value given as *R* ∈ {0,l,?,-}, consists of 0, 1, and "?" denotes an undefined or random logical value. . A '-' in *R* means that the output data is not applicable. An undefined logical value can occur if the voltage difference between the bit lines (used by the sense amplifier) is very small. In that case; e.g., if *S = w0,* then no data will appear at the memory output, and therefore *R* is replaced by a '-'.

2.2 UNDETECTABLE FAULT ANALYSIS ON WDFS AND DRDFS

As mentioned in Section I, the fault analysis will focus on March Algorithm with a maximum of 14N test operations. Table II shows four types of well-known March Test Algorithms in this case study, namely MATS++, March C-, March SR and March CL.

Table I1: Well Known March Test Algorithm

March Algorithm	Operation Sequence
MATS++ :6N [10,11,16]	*{⇳ (w(0)); ⇑ r(0),w(1)); ⇓ (r(1) , w(0), r(0)) }.*
March C- :10N [9,10,11,15]	*{⇳(w0); ⇑(r0,w1); ⇑(r1,w0); ⇓(r0,w1); ⇓ (r1,w0);⇳(r0) }*
March SR: 14N [4]	*{⇳(w0) ;⇑(r0,w1,r1,w0) ⇑(r0,r0,); ⇑(w1); ⇓ (r1,w0,r0, w1); ⇓(r1,r1) }*
March CL 12N [3,8,11]	*{⇳(w0); ⇑(r0,w1); ⇑(r1 , r1, w0); ⇓(r,w1,r1,); ⇓(r1,w0); ⇳(r0) }*

Table III: Test Operation Sequence Of Wdf And Drdf Detection

Fault Type	FFM	Test Operation Sequence
Write Disturb Fault (WDF)	WDF0 = < 0w0/↑/-> WDF1 = < 1w1/↓/->	⇳(wx,wx,**rx**) or (..wx) ;⇳(wx**rx**) or ⇳(..wx..wx) ;⇳(**rx**..) or ⇳(wx, rx;) ⇳(wx,**rx**) ⇳ - arbitrary addressing order, it can be opposite addressing or same addressing order, **rx** ,the bold font denotes the detection, if the fault occur, **x** is read inverted value. If x is 0, WDF0 is detected and if x is 1, WDF1 is detected
Deceptive Read Destructive Fault (DRDF)	DRDF0= < r0/ ↑/0> DRDF1 = < r1/ ↓/1>	⇳(rx,**rx**) or ⇳(..rx) ;⇳(**rx**..) or ⇳(rx) ⇳(**rx**) or ⇳(..rx) ⇳(**rx**..) ⇳ - arbitrary addressing order, it can be opposite addressing or same addressing order, **rx** ,the bold font denotes the detection, if the fault occur, **x** is read inverted value. If x is 0, DRDF0 is detected and if x is 1, DRDF1 is detected

Table IV: March Algorithm And Its Detection Analysis

© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

March Algorithm	Fault detection towards FFMs and its test sequence operation			
Algorithm\Faults	WDF0	WDF1	DRDF0	DRDF1
MATS++ [6N]	X	X	X	X
MARCH C- [10]	X	X	X	X
MARCH SR [14N]	X	X	√	√
MARCH CL [12N]	X	X	√	X

Table III shows the functional fault primitives and the test sequence operation to detect WDFs and DRDFs. As shown in the third column of Table III, if any of the test sequence operation is matched with the test operation listed in the elements of the March Algorithm, the specified fault can be detected. Based on the analysis on selected March Algorithm, a list of faults that can be detected and undetected is shown in Table IV. It shows that the MATS++ and March C- cannot detect WDFs and DRDFs. But March SR can detect DRDFs. Both March SR and March CL cannot detect WDFs however March CL can only detect DRDF0. Therefore, this is a motivation to detect the undetected faults by modifying the sequence of the transition values but still maintaining the addressing order and its element.

3. METHODOLOGY

3.1 Proposed Solution

To improve the fault detection of SRAM testing, some rules have been set to generate a new sequence of transition and non-transition values. This is to ensure the undetected faults such as Deceptive Read Destructive Faults (DRDFs) and Write Disturb Faults (WDFs) can be recovered. The March-test sequences should also consider the detection for Static Double Cell Faults (SDCFs) as well. The proposed steps are as follows:

Proposition 1: Allowing non-transition and transition operations and two consecutive double read operations in order to sensitize and desensitize faults such WDFs, TFs and DRDFs. There will be a customized March Test by modifying well-known algorithm with the automation program based on the sequence operation (SQ) generation scheme.

*Proposition 2: In the specified March algorithm, there is a need to have a double read operation whereby one of these operations is listed in the specified March Algorithm, which follows one of the test operation sequence specified in Table III. The operation rule must operate both condition values. If the March algorithm only allows FFP (<**r1**/ ↓ /**0**>) of the DRDF1 to be detected, the March-test sequences should also facilitate to satisfy FFP (<**r0**/ ↑/**1**) of the DRDF0 detection and vice versa.*

Proposition 3: If the test sequence operation does not consist of operations to detect WDFs specified in Table III, this sequence need to be added in the defined test Algorithm and if the sequence fulfilled only one condition, the operation rule of the non- transition values need to be added.

Proposition 4: If the test sequence operation does not consists of test operations to detect TF↓ by given "w1w0r0" operations and to detect TF↑ by given "w0w1r1", this sequence operation needs to be added in the defined test Algorithm.

Proposition 5: The March-test sequences must also consider the detection for Static Double Cell Faults (SDCFs) as well. As explained in Proposition 2, the double read operation is included in the test Algorithm, this operation allows CFdr to be detected. In order to detect the fault, all states of arbitrary cell should be generated and double read operation need to be run on victim cell. In Proposition 3, the test operation allows detecting CFwd, and as per described in Proposition 4, the test operation also will cover CFtr. All coupling fault will happen at two arbitrary cells where one cell as aggressor and another cell as victim cell, each type of fault is referring to the cell location address as per describe in Table I.

3.2 Data Background Generation

First, the automation program called DB generator is written in Verilog Hardware Design Language (HDL). As explained in Section 3.1, the proposition steps from Proposition 1 to Proposition 4 are followed to ensure the undetected faults such as Deceptive Read Destructive Faults (DRDFs) and Write Disturb Faults (WDFs) can be detected. Proposition 5 covers the evaluation of detecting coupling fault. The DB generator hardware is designed to generate list of bits patterns which indicates the sequence of write operation, if DB [bit] data value is "0", the write operation will be writing 0s and if DB [bit] data value is "1", the write operation will be writing 1s. In developing the new sequence of writing operations referred to as SQ, one rule was set and this is referred to as SQ generation rule. The rules are as follows:

a) Count Number of Write Operation in the test algorithm, given W_N where $\{W\}_N$ = W1, W2, W3, W4,…WN.

b) Writing 0 is a must for initialization. So it

Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

always starts with writing a 0, and all possible sequences that start with writing 1 are removed. Therefore bits pattern, will always start with 0 e.g. 01101 where the MSB (most significant bit value is 0). With this condition, the total of possible sequence, total SQ list = $2^{(op\_num\ -1)}$ -1.

c) Writing a non-transition operation, *w0→w0* and *w1→w1,* must occur as the next sequence where this is specified as a static scheme operation. Therefore the possible bit pattern will have any arbitrary of "00" and "11".

d) Writing transition operation, *w0→w1* and *w1→w0* must occur, otherwise, the sequence operation is omitted. This scheme is denoted as a dynamic scheme. Therefore the possible bit pattern will have any arbitrary of "01" and "10".

e) In detecting fault of memories functional defect, the writing operation only execute those operation; w0→w0, write 0 operation to a cell which contains a 0, w1→w1, write 1 operation to a cell which contains a 1, w0→w1, write 0 operation

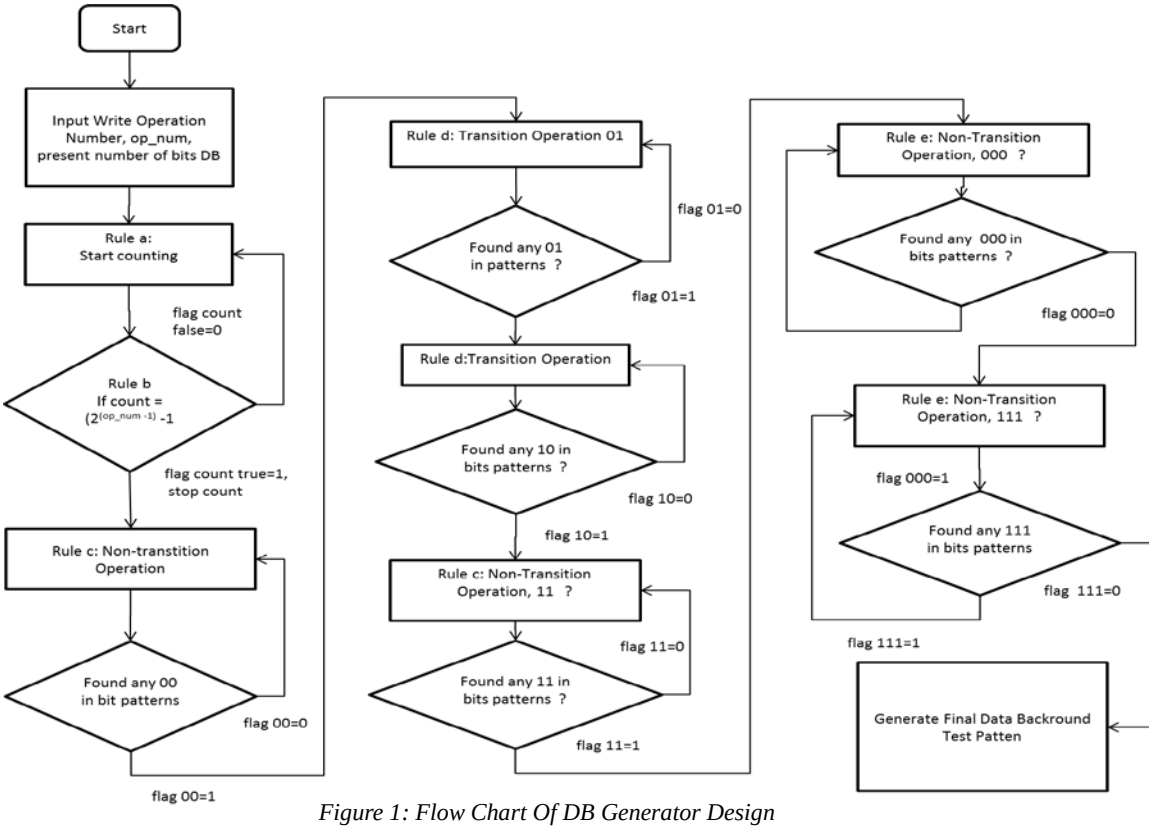

Figure 1: Flow Chart Of DB Generator Design

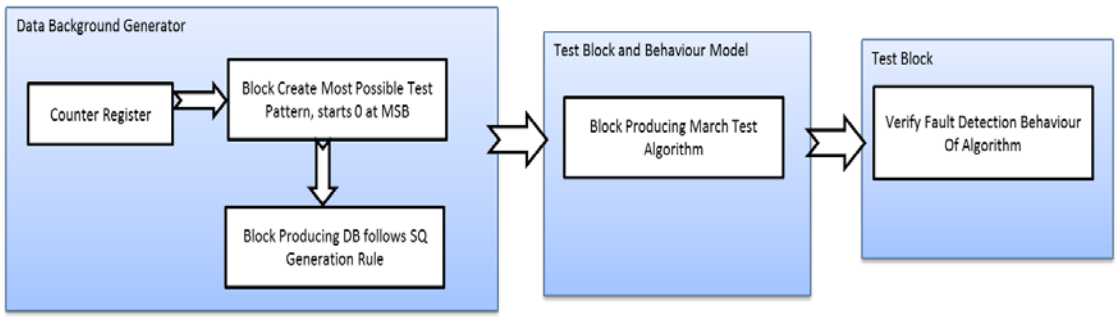

Figure 2: Overall Architecture

to a cell which contains a 1 and w1→w0 whereby write a 1 operation to a cell which contains a 0 and analysis from fault primitives list, the sequence detection or sensitization does not has triplet operation such as w0→w0→w0 or w1→w1→w1, therefore any bits pattern either "000" and "111"

© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

will be ignored.

f) The hardware program will search for other possible sequence bit patterns which obeys rule (a) to rule (e).

Figure 1 shows the flow chart of DB (Data Background) Generator design that follows SQ generation rule based on rule (a) to rule(e). Each control signal with prefix name "*flag<ruledonate>*" will filter all the possible data that matches with the specified rule. Figure 2 shows the overall architecture of the design hardware consisting three blocks; First block is Data Background Generator, second block is a test block to receive DB bits pattern and producing March Test Algorithm and finally third block is the test block verifying fault detection of the Algorithm. The design implementation of March Algorithm generation block is explained in detail in Section 3.3

3.3 March Algorithm Generation Steps

After generating DB bits pattern, customized Algorithm will be produced by replacing each DB bits value with all write operations and re-union with other operations that follows in original March Test algorithm or producing new March Algorithm with new DB list. The process steps are described below:

a) Collect all the combination of DB final to be used to generate new Algorithm or to modify existing Algorithm.

b) For producing new Algorithm, each DB[n] value append the "w" present write operation and "r present read operation e.g w{DB[n]} and r{DB[n]}. Some of the rules need to be added to ensure it gives the high fault coverage. An example of new test Algorithm generation with possible sequence operation for a specified test algorithm consisting of five write operations, W_N =5, is illustrated in Table V. The generated SQs follows SQ rule (a) and SQ rule (b). For SQ rule (c), SQ4 is omitted as there is no w0→w0. For SQ rule (d), SQ3 is omitted as there is no transition w1→w0, thus TF1 cannot be detected.

Table V: Example Of The Possible Writing Sequence In Producing New Algorithm

	SQ1	SQ2	SQ3	SQ4
w1	w0	w0	w0	w0
w2	w0	w1	w0	w1
w3	w1	w1	w0	w1
w4	w1	w0	w1	w1
w5	w0	w0	w1	w0

c) Double read operation position needs to be identified by ensuring both *r0,r0* and *r1,r1* occurs between it. If the original algorithm does not contain a double read operation, the test Algorithm needs to be removed. If the read operation happened after w2 and w4, SQ4 is omitted because DRDF0 cannot be detected.

c) In this paper, the work is focus on modifying the existing Algorithm. Generating customized test Algorithm with the new SQ list operation will produce a new test algorithm called new-March-test Algorithm. All possible SQs will be inserted into the March Algorithm until the optimum number of SQs is fulfilled to obtain the highest coverage.

d) As an example, March SR referred from Table III, contains 6 write operations. The DB bits pattern will contain 6 bit value, namely, DB[5], DB[4],DB[3], DB[2],DB[1] and DB[0]. Each March element is divided by ";" and the first March element, called M0, is the initialization operation. Each March element will contain the testing operation that follows the sequence of addressing order consists of ⇑, *which* denotes ascending address order, ⇓ denotes descending address order and ⇕ denotes an arbitrary address order. The original Algorithm is *{⇕ (w0) ;⇑ (r0,w1,r1,w0) ⇑ (r0,r0,); ⇑ (w1); ⇓ (r1,w0,r0, w1); ⇓ (r1,r1) }*, by replacing the bits pattern the new Algorithm will be produced*: {⇕(w*{DB[5]}*) ;*

⇑(r{DB[5]}*, ,w*{DB[4]}*,r*{DB[4]}*,w*{DB[3]}*);*

⇑(r{DB[3]}*,r*{DB[3]}*,); ⇑(w*{DB[2]}*);*

⇓(r{DB[4]}*, w*{DB[1]}*,r* {DB[1]}*, w*{DB[0]}*);*

⇓(r{DB[0]}*,r*{DB[0]}*) }*

e) In implementing a method as referred to in (c), any double read operation must has both r0r0 and r1r1 at its March Test Algorithm. Otherwise, the new Algorithm will be neglected.

f) Finally each test Algorithm is evaluated based on its fault coverage and power consumption estimation.

4. NEW MARCH TEST ALGORITHM AND ITS PERFORMANCE EVALUATION

The test algorithm with the new DB bits patterns will produce a new test algorithm called new-March-test Algorithm. All possible DBs which obeys SQ generation rule stated in Section 3.2 and

Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

chronology process steps stated in Section 3.3 will be inserted into the March Algorithm until the optimum number of DBs is fulfilled to obtain the highest coverage.

1) Modification of MATS++ and March C-

MATS++ consists of 3 write operations that is not sufficient to meet SQ rule (c) and SQ rule (d). For March C- the implementation only can be done up to SQ rule (d) and since the test algorithm cannot comply with SQ rule (e), the implementation work for Mod March C- is not discussed. Note that with SQ lists that are generated according to SQ rule (d), the detection improvement only covers WDFs. Therefore the development of Mod March C- and Mod MATS++ can be ignored.

2) Modification of March CL with W_n,=5

The Modified March Algorithm for March CL has 2 types of March-test sequences for the detection of SSCFs which obeys SQ generation rule. Both March CL-1 and March CL-2 are tabulated in Table VI.

Table VI: New Db And New March Cl

`	DB bits pattern	New March Algorithm	Rules Obey
March CL-1	00110	{⇳(w(0)); ⇑(r(0),w(0)); ⇳(r(0)); ⇑(r(0)w(1)); ⇓(r(1),w(1)); ⇳(r(1)); ⇓ (r(1),w(0)); ⇳(r(0)) }	Yes
March CL-2	01100	{⇳(w(0)); ⇑ (r(0),w(1)); ⇳ (r(1)); ⇑ (r(1),w(1)); ⇓ (r(1),w(0)); ⇳ (r(0)); ⇓ (r(0),w(0)); ⇳ (r(0)) }	Yes

Fault detection on both March CL-1 and March CL-2 covers all Static Single Cell Faults FFMs.

1) All SFs, RDFs and IRFs are detected since from each cell a 0 and a 1 is read.

2) All TFs are detected because each cell is read after an up and a down transition write operation. For example in March CL-1, the transition fault, TF↓ < 0w1/0/- > is sensitized by M3,2 and detected by M4,1 while the transition fault, TF↑ < 1w0/1/- > is sensitized by M6,2 and detected by M7,1.

3) WDF and DRDF detection are already solved by following the SQ generation rule and the detection details are tabulated in Table VII and Table VIII.

Fault detection on both new March CL, March CL-1 and March CL-2 covers CFtr and CFwd as tabulated in Table VII and Table VIII. Both March CL-1 and March CL-2 cannot detect CFdrd because there is no consecutive double read operation in the March element to detect the fault.

TABLE VII: FAULT DETECTION BY MARCH CL-1

Static Single Cell Fault Detection by March CL-1			
Fault Type	Fault Primitive	Sensitized By	Detected By
DRDF	< r0/↑/0 >	M2,1	M3,1
	< r1/↓/1 >	M5,2	M6,1
WDF	< 0w0/↑/->	M1,2	M2,1
	< 1w1/↓/->	M4,2	M5,1

Double Cell Static Fault Detection March CL-1				
Fault Type	Fault Primitive	Address	Sensitized By	Detected By
CFwd	CFwd <0;0w0 /↑/->	v < a	M1,2	M2,1
	CFwd <0;0w0 /↑/->	v > a	M1,2	M2,1
	CFwd <1;1w1 /↓/->	v > a	M4,2	M5,1
	CFwd <1;1w1 /↓/->	v < a	M4,2	M5,1
Cftr	CFtr< 0;0w1/↑/->	v < a	M3,2	M4,1
	CFtr< 1;0w1 /↑/->	v > a	M3,2	M4,1
	CFtr< 0;1w0 /↓/->	v > a	M6,2	M7,1
	CFtr< 1;1w0 /↓/->	v < a	M6,2	M7,1

Table VIII: Fault Detection By March Cl-2

Static Single Cell Fault Detection by March CL-2			
Fault Type	Fault Primitive	Sensitized By	Detected By
DRDF	< r0/↑/0 >	M5,1	M6,1
	< r1/↓/1 >	M2,1	M3,1
WDF	< 0w0/↑/->	M6,2	M7,1
	< 1w1/↓/->	M3,2	M4,1

Double Cell Static Fault Detection March CL-2				
Fault Type	Fault Primitive	Address	Sensitized By	Detected By
CFwd	CFwd <0;0w0 /↑/->	v < a	M1,2	M2,1
	CFwd <0;0w0 /↑/->	v > a	M1,2	M2,1
	CFwd <1;1w1 /↓/->	v > a	M3,2	M4,1
	CFwd <1;1w1 /↓/->	v < a	M3,2	M4,1
Cftr	CFtr< 0;0w1/↑/->	v < a	M1,2	M2,1
	CFtr< 1;0w1 /↑/->	v > a	M1,2	M2,1
	CFtr< 0;1w0 /↓/->	v > a	M4,2	M5,1
	CFtr< 1;1w0 /↓/->	v < a	M4,2	M5,1

Power Estimation: State transitions of the transistors are the toggling of the output values from 1 to 0 or from 0 to 1 which indicate the switching power. The generated test Algorithm shows that there are two times transition state write operations compared to the original test Algorithm that has 3 times transition state write operations, thus reducing test power consumption.

3) Modification of March SR with W_n,=6

There are four possible test Algorithms with generated DB tabulated in the Table IX. The Modified March SR Algorithm has two sets of March-test sequences which obeys both SQ generation rule and chronology process step that covers all detection of SSCFs. Another two test algorithms are neglected as they do not contain r0r0 in the test Algorithm.

Journal of Theoretical and Applied Information Technology
10th January 2013. Vol. 47 No.1
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table IX: New Db And New March Sr

`	DB bits pattern	New March Algorithm	Rules Obey
March-test-SR-1	001011	{ ⇳ (w0) ; ⇑ (r0,w0,r0,w1) ⇑ (r1,r1,); ⇑ (w0); ⇓ (r0,w1,r1, w1); ⇓ (r1,r1)	No
March-test-SR-2	001100	{ ⇳ (w0) ; ⇑ (r0,w0,r0,w1) ⇑ (r1,r1,); ⇑ (w1); ⇓ (r1,w0,r0, w0); ⇓ (r0,r0)	Yes
March-test-SR-3	001101	{ ⇳ (w0) ; ⇑ (r0,w0,r0,w1) ⇑ (r1,r1,); ⇑ (w1); ⇓ (r1,w0,r0, w1); ⇓ (r1,r1)	No
March-test-SR-4	010011	{ ⇳ (w0) ; ⇑ (r0,w1,r1,w0) ⇑ (r0,r0,); ⇑ (w0); ⇓ (r0,w1,r1, w1); ⇓ (r1,r1)	Yes
March-test-SR-5	011001	{ ⇳ (w0) ; ⇑ (r0,w1,r1,w1) ⇑ (r1,r1,); ⇑ (w0); ⇓ (r0,w0,r0, w1); ⇓ (r1,r1)	No

Fault detection by March SR-1 is tabulated in the Table X and fault detection by March SR-2 is tabulated in the Table XI. March SR-1 and March SR-2 covers CFwd, CFdrd and CFtr.

Table X: Fault Detection By March Sr-1

Static Single Cell Fault Detection by March SR-1			
Fault Type	Fault Primitive	Sensitized By	Detected By
DRDF	< r0/↑/0 >	M5,1	M5,2
	< r1/↓/1 >	M2,1	M2,2
WDF	< 0w0/↑/->	M1,2	M1,3
	< 1w1/↓/->	M3,2	M4,1

Double Cell Static Fault Detection of March SR-1				
Fault Type	Fault Primitive	Address	Sensitized By	Detected By
CFdrd	CFdrd<0;r1/↓/1>	v>a	M2,1	M2,2
	CFdrd<1;r1/↓/1>	v<a	M2,1	M2,2
	CFdrd<0;r0/↑/0>	v>a	M5,1	M5,2
	CFdrd<1;r0/↑/0>	v<a	M5,1	M5,2
CFwd	CFwd <0;0w0 /↑/->	v < a	M1,2	M1,3
	CFwd <1;0w0 /↑/->	v > a	M1,2	M1,3
	CFwd <0;1w1 /↓/->	v < a	M3,2	M4,1
	CFwd <1;1w1 /↓/->	v > a	M3,2	M4,1
	CFwd <0;0w0 /↑/->	v > a	M4,4	M5,1
	CFwd <1;0w0 /↑/->	v < a	M4,4	M5,1
Cftr	CFtr< 0;0w1/↑/->	v<a	M1,4	M1,1
	CFtr< 1;0w1 /↑/->	v>a	M1,4	M1,1
	CFtr< 0;1w0 /↓/->	v < a	M3,1	M4,1
	CFtr< 1;1w0 /↓/->	v > a	M3,1	M4,1

Table XI: Fault Detection By March Sr-2

Static Single Cell Fault Detection by March SR-2			
Fault Type	Fault Primitive	Sensitized By	Detected By
DRDF	< r0/↑/0 >	M2,1	M2,2
	< r1/↓/1 >	M5,1	M5,2
WDF	< 0w0/↑/->	M3,2	M4,1
	< 1w1/↓/->	M4,4	M5,1

Double Cell Static Fault Detection March SR2				
Fault Type	Fault Primitive	Address	Sensitized By	Detected By
CFdrd	CFdrd<0;r0/↑/0>	v>a	M2,1	M2,2
	CFdrd<0;r0/↑/0>	v<a	M2,1	M2,2
	CFdrd<1;r1/↓/1>	v>a	M5,1	M5,2
	CFdrd<1;r1/↓/1>	v<a	M5,1	M5,2
CFwd	CFwd <0;0w0 /↑/->	v < a	M3,1	M4,1
	CFwd <1;0w0 /↑/->	v > a	M3,1	M4,1
	CFwd <0;1w1 /↓/->	v > a	M4,1	M5,1
	CFwd <1;1w1 /↓/->	v < a	M4,1	M5,1
Cftr	CFtr< 0;0w1/↑/->	v<a	M1,2	M1,3
	CFtr< 1;0w1 /↑/->	v>a	M1,2	M1,3
	CFtr< 0;1w0 /↓/->	v < a	M2,4	M3,1
	CFtr< 1;1w0 /↓/->	v > a	M2,4	M3,1
	CFtr< 0;0w1/↑/->	v>a	M4,2	M4,3
	CFtr< 1;0w1 /↑/->	v < a	M4,2	M4,3

Power Estimation: The generated March Algorithm shows less number of transition states. March SR-4 has three states of transition write operations and March SR-2 has two states of transition write operations compared to the original March SR that has four states of transition write operations. Thus in terms of test power performance, power consumption for generated Algorithm is reduced.

5. FAULT COVERAGE AND DISCUSSION

Refer to Section 4, , the generated test algorithms achieved 100% fault coverage for SSCFs by using the SQ generation rule scheme. The undetected fault, WDFs and DRDFs is definitely improved by the proposed solution. The summarized result of the fault detection shows that after modifying March CL and March SR with its new DB, the detection on SDCFs are also improved. The total detection of each type CFs denotes 8 detections depending on the condition of the position of the aggressor cell and the location of victim cell. For the first four faults, where victim cell address is smaller than the aggressor cell ($c_v < c_a$) and another four faults where the address of the victim cell is higher than the aggressor cell ($c_v > c_a$). If the FP condition is 4, the total condition detection will be 8.

The result of fault coverage and fault detection summary as compared to their original March Algorithm is tabulated in Table XII. Result showed hat both new March CLs are unable to detect CFdrd, but to detect the fault, a consecutive double read operation at its March element operation is required. However, in terms of SDCFs detection, March SR-1 and March SR-2 are able to detect CFtr, CFwds and CFdrds. Detection performance on the detection of CFwd shows an improvement as per generated Algorithm compared to its original March Algorithm. However for CFtr detection, March SR-1 is able to detect six of eight faults compared to its original Algorithm which covers all condition of CFtr. It is because the state transition is less compared to the original Algorithm but on the other hand, the lower the state transition, the less power it consume. As described in Section 4, the generated test algorithm still leads the test power performance even though the performance of CFs does not achieve 100% fault coverage.

The evaluation data collected in Table XII shows both March SR-1 and March SR-2 gives the highest fault coverage which is up to 97% compared to its original Algorithm that can only achieve 86% fault coverage.

© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table XII: Fault Coverage

	March CL	March CL-1	March CL-2	March SR	March SR-1	March SR-2
Fault	Static Single Cell Faults					
SF	2/2	2/2	2/2	2/2	2/2	2/2
STAF	2/2	2/2	2/2	2/2	2/2	2/2
DRDF	1/2	2/2	2/2	2/2	2/2	2/2
RDF	2/2	2/2	2/2	2/2	2/2	2/2
IRF	2/2	2/2	2/2	2/2	2/2	2/2
WDF	0	2/2	2/2	0/2	2/2	2/2
TF	2/2	2/2	2/2	2/2	2/2	2/2
Fault	Static Double Cell Faults					
CFdrd	0/8	0/8	0/8	6/8	4/8	4/8
CFwd	0/8	4/8	4/8	0/8	4/8	6/8
CFtr	6/8	4/8	4/8	8/8	6/8	4/8
% coverage	63%	80%	80%	86%	97%	97%

The detection of CFs in this work is limited because the program maintains the element of the operation in the algorithm and only focuses to change the data values to achieve 100% of SSCFs compare to the original Algorithm. However, March SR-1 and March SR-2 can be used for some designs in industry since the fault coverage achieves 97%. In order to achieve 100% fault coverage, another 4N test operation is required. Longer test operation and bigger hardware area, will increase the test cost.

6. CONCLUSION

In this paper, we have shown the undetectable SSCFs faults can be detected by generating customized March Algorithms from a well-known March algorithms, with a maximum of 14N operations and achieved more than 95% fault coverage. The customized March-test algorithm is programmed based on the SQ generation rule as it dominate rules in writing and reading sequences to fulfill FP of SSCFs. In addition, the generated test algorithm also considers the SDCF detection. The customized March SR-1 and March SR-2 algorithm can be used for some design application with 97% yield performance which leads to less power consumption. For future work, some features in the proposed sequence operation will be enhanced later to improve linked DCFs detection.

ACKNOWLEDGMENT

The authors would like to give high appreciation to Ministry of Science and Technology to sponsor the grant supporting this research.

REFRENCES:

[1] Zakaria, N.A, W.Z.W Hassan, I.A. Halin, R.M. Sidek, Xiaoqing Wen, "Testing Static Single Cell Faults Using Static and Dynamic Data Background" *Proceedings of 2011 IEEE Student Conference on Research and Development*, SCOReD 2011, pp. 1–6, 2011

[2] S. Hamdioui, A. J. van de Goor, and M. Rodgers, "March SS: A Test for All Static Simple RAM faults," *Proceeding of IEEE International Workshop Memory Technology, Design and Testing*, Bendor, France, pp. 95–100, July 2002.

[3] J. V.A Vardanian, Y. Zorian, "A March-based fault location algorithm for static random access memories" *Proceedings of the Eighth IEEE International in On-Line Testing Workshop*, pp. 256 - 261, 8-10 July 2002.

[4] S. Hamidoui, A.J. van de Goor, "An Experiment Analysis of Spot Defects in SRAMs: Realistic Fault Models and Tests", *Proceedings of the Ninth Asian Test Symposium, 2000*, pp. 131 – 138, 4-6 Dec. 2000.

[5] S. Hamdioui, 'Testing Static Random Access Memories: Defects, Fault Models and Test Patterns', *Kluwer Academic Publishers*, Boston, MA; ISBN 1-4020-7752-1, 2004.

[6] A.J. van de Goor and Z. Al-Ars, "Functional Memory Faults: A Formal Notation and a Taxonomy", *Proceeding of 18th VLSI Test Symposium 2000*, pp. 281-289, 6 August 2002.

[7] G. Harutunvan, V.A Vardanian, Y. Zorian, "Minimal March Tests for Unlinked Static Faults in Random Access Memories" *Proceeding of 23rd VLSI Test Symposium, 2005*, pp. 53 – 59, 2005.

[8] C.-F. Wu et al., ''Fault Simulation and Test Algorithm Generation for Random Access Memories,'' *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 4, 2002, pp. 480-490.

[9] Wei-Lun Wang, Kuen-Jong Lee "A programmable data background generator for march based memory testing" *Proceeding of IEEE Asia-Pacific Conference on Digital 2002*, APASIC 2002, pp 347-350, 2002.

[10] Jin-Fu Li and Cheng-Wen Wu, "Memory Fault Diagnosis by Syndrome Compression", *Design, Automation and Test in Europe, 2001*. Conference and Exhibition 2001, pp. 97–101, 13-16 March 2001.

© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[11] Wan Hassan, W.Z, Othman, M., Suparjo, B.S., "A Realistic March-12N test And Diagnosis Algorithm for SRAM Memories" *IEEE International Conference on Semiconductor Electronics*, pp. 919 – 923, 2 July 2007.

[12] Bosio, A. Di Carlo, S. Di Natale, G. Prinetto, P. "March AB, a state-of-the-art march test for realistic static linked fault and dynamic faults in SRAMs*" Computers & Digital Techniques, IET*, vol.1 No. 3, pp. 237-245, 21 May 2007.

[13] L.-T. Wang, C.-W. Wu, and X. Wen, (Editors), Chapter 8, *VLSI Test Principles and Architectures: Design for Testability*, San Francisco: Elsevier, 2006.

[14] S. Hamdioui, J.E.Q.D. Reyes "New data-background sequences and their industrial evaluation for word-oriented random-access memories" *IEEE Transaction On Computer Aided Design Of Integrated Circuits and System*, Vol. 24, No. 6, pp. 892-904, 2005.

[15] A.J. van de Goor, I.B.S. Tlili, and S. Hamdioui, "Converting March Tests for Bit-Oriented Memories into Tests for Word-Oriented Memories", *IEEE Int. Workshop on Memory Technology, Design*, pp 46-52, 1998.

[16] A.J. van de Goor, Testing semiconductor memories: Theory and Practice, *John Wiley & Sons, Chichester*, England, 1991.

[17] R. Dean Adams, High Performance Memory Testing: Design Principles, Fault Modeling and Self Test, *Kluwer Academic Publishers*, pp. 138, 2002.

[18] Sultan M. Al-Harbi, Fadel Noor, Fadi M. Al Turjman, "March DSS: A New diagnostic March Test For All Memory Simple Static Fault" *IEEE Transaction On Computer Aided Design Of Integrated Circuits and System*, Vol. 26, No. 9, September 2007.

[19] Mentor's Product ,Tessent™ MemoryBIST Usage Guide and Reference, November 2010.

[21] Wan Hasan Wan Zuha, Abdul Halin Izhal, Roslina Mohd Sidek, & Othman Masuri. 2009. An Efficient Fault Syndromes Simulator for SRAM Memories. *IEICE Trans, Eelectron* 92(5): pp.639-646.

[20] International Technology Roadmap For Semiconductors 2010 Update Overview.