文章编号:1674-2974(2019)04-0097-05 DOI:10.16339/j.cnki.hdxbzkb.2019.04.014

一种改进的 SRAM 故障内建自检测算法

曾健平[1],王振宇[1†],袁甲[2],彭伟[1],曾云[1]
(1.湖南大学 物理与微电子科学学院,湖南 长沙 410000;
2.中国科学院微电子研究所,北京 100029)

摘 要:面向 March C+算法故障覆盖率的问题,本文提出一种改进的 March CS 算法来完成存储器 SRAM 的内建自测试. 通过增加原算法元素的读写操作来敏化存储单元的故障,检测原算法不能敏化的静态故障和动态故障,从而提高故障覆盖率.最后,通过对 1 024*32 位静态随机存储器进行故障仿真验证,以及 FPGA 对 SRAM 芯片的应用性测试,March CS 算法检测静态故障和动态故障的覆盖率分别达到 91.67%和 76.93%.

关键词: March CS 算法;静态故障; 动态故障; 故障覆盖率;
中图分类号: TN47 **文献标志码:**A

An Improved SRAM Fault Built-in-self-test Algorithm

ZENG Jianping[1], WANG Zhenyu[1†], YUAN Jia[2], PENG Wei[1], ZENG Yun[1]
(1.School of Physics and Microelectronics Science, Hunan University, Changsha 410000, China;
2. Institute of Microelectronics, Chinese Academy of Sciences, Beijing 100029, China)

Abstract: This paper proposed an improved March CS algorithm to complete the built-in self-test of SRAM memory due to the problem of March C + algorithm´s fault coverage. The static and dynamic fault of memory cell were sensitized by the original algorithm increasing the read-write operation of the algorithm element, so that the fault coverage was enhanced. Finally, the March CS algorithm achievesd 91.67% and 76.93% coverage of static and dynamic faults respectively through the simulation experiments of the 1 024*32-size fault static Randon-Access memory and the measurement of FPGA to SRAM chips.

Key words: March CS algorithm; static fault; dynamic fault; fault coverage

集成电路的发展一直遵循着摩尔定律, 片上系统 SoC (System-on-Chip) 已经成为了集成电路 IC (Integrated Circuit Design)设计的主流之一[1].随着 IC 设计的发展与更新,集成电路的测试也成为一个备受关注的焦点问题.在如今的存储器测试方法中,内建自测试 BIST(Built-in-Self-Test)成为了一种主流测试方法[2-3].存储器内建自测试 MBIST 不仅能简化测试中的步骤,而且可以缩短测试所需时间,对故障的覆盖比较全面[4].

存储器内建自测试 MBIST 的研究集中在测试

* **收稿日期:**2018-04-26
基金项目:国家自然科学基金资助项目(61705065),National Natural Science Foundation of China(61705065);湖南省自然科学基金资助项目(2017JJ3034), Natural Science Foundation of Hunan Province(2017JJ3034);长沙市科技计划项目(kq1804001),Technology Program of Changsha(kq1804001)
作者简介:曾健平(1966—),男,湖南祁东人,湖南大学副教授,博士
† 通讯联系人,E-mail:1042729376@qq.com

算法中,通过采用一种或多种测试算法对静态随机存储器 SRAM 进行故障测试. 目前比较常用的算法包括:MSCAN 算法、棋盘算法、GALPAT 算法以及 March 算法等.其中 March 算法[5]在当今存储器测试应用较多,比较常用的有 March C 算法、March C+算法等,它们具有较高的故障覆盖率,且测试时间短,但仍然有许多故障不能被覆盖[6].

随着集成电路内建自测试技术的发展,国外芯片公司已经设计出比较成熟的内建自测试的工具,比如 Mentor 公司的 MBIST Architect 软件以及 Synopsys 开发出的 DFT Compiler 软件.本文的算法是通过 MBIST Architect 软件实现内建自测试[7],并且是对于 memory compile 生成的 1024X32 的 SRAM 进行功能仿真与验证.

1 存储器的故障概念

存储器的故障一般指逻辑与功能上故障,其中主要包括固定故障(SAF)、转换故障(TF)、读破坏故障(RDF)、写干扰故障(WDF)以及耦合故障(CF)等[8].故障原语 FP(Fault Primitive)是用来表示存储器的故障行为,其通常的表现形式有两种,分别为〈S/F/R〉与〈S_a;S_v/F/R〉,其中,〈S/F/R〉通常表示单个单元的故障行为,而〈S_a;S_v/F/R〉通常表示两个单元的故障敏化序列. 其中 S 表示存储器故障行为的操作和状态,F 表示故障单元的状态值,R 表示存储器的读操作的结果.而对于两个单元的故障原语,S_a 表示施主单位的敏化序列,a 表示施主单元的地址.S_v 表示故障单位的敏化序列,v 表示故障单元的地址[9].

存储器的故障类型一般有固定故障、转换故障、写干扰故障、读破坏故障以及耦合故障等类型.其中,耦合故障(CF)是发生在多个存储单元之间,是指某个单元的变化而影响到其他存储单元发生变化.耦合故障的类型主要是以下几种:反相耦合故障(CF_{in})是指某个存储单元跳变为 0 或 1 时,而导致其它一个的存储单元的值进行翻转. 定值耦合故障(CF_{id})是指某个存储单元跳变为 0 或 1 时,而导致其它的存储单元的值变为了一个固定值. 状态耦合故障(CF_{st})是指存储单元在某一个确定状态时,而导致了另一个存储单元跳变到了错误状态.

2 March C+算法

在对 SRAM 存储器进行内建自测试(MBIST)时,March C+算法是最常用的一种.它具有较高的故障覆盖率,能检查固定故障 SAF、转换故障 TF、读破坏故障 RDF 以及部分的耦合故障 CF,并且它的测试时间较短,对静态故障与动态故障的覆盖率分别达到 75%与 50.13%.其中 March C+算法如下图 1 所示[10-11].

⇕ W0; ⇑(r0,w1,r1); ⇑(r1,w0,r0);
M0 M1 M2
⇓(r0,w1,r1); ⇓(r1,w0,r0); ⇕(r0);
M3 M4 M5

图 1 March C+算法

Fig.1 March C+ Algorithm

虽然使用 March C+算法对存储器进行故障测试时,其覆盖率是比较全面,但是仍存在一些静态故障与动态故障没有被覆盖.其中,静态故障包括单个单元的静态写干扰故障(WDF),单元静态干扰耦合故障(CFd_{sxwx})与单元静态写干扰耦合故障(CFd_{wd})共 10 种故障.同样,在 March C+算法进行 SRAM 的动态故障测试时,仍然有大量的故障不能被检测出来,其中包括单个单元的动态写干扰故障(d_{WDF})、动态读破坏故障(d_{RDF}),以及两个单元的动态写干扰耦合故障(dCF_{wd})、动态干扰耦合故障(dCF_{dwxwxw})和动态读破坏耦合故障(dCF_{rdwxrx})共 20 种故障.

3 March CS 算法的提出

在前述 March C+算法中提到,算法对许多的故障检测是没有覆盖的. 本节在原算法的基础上提出了新算法 March CS,来提高存储单元的静态故障与动态故障的覆盖检测率.

用 March C+算法进行芯片存储器测试时, 仍存在一些静态故障没有被覆盖, 其中包括单个单元的静态写干扰故障(WDF)、静态干扰耦合故障(CFd_{sxwx})等.这类故障不能被检测出来的关键是:在将存储单元的状态初始化为 0(或者 1)状态时,而不能继续进行 w0(或者 w1)操作,从而不能将存储单元敏化为 0w0(或者 1w1)的状态.故在 March C+算法的基础上,第 M1 和 M3 的元素中均加入 w0 操作,可以敏化存储单元的 0w0 操作.同理,在 M2 与 M4 的元素中加入 w1 操作, 可以敏化存储单元的 1w1 操作.因此,得到改进的 March CS* 新算法如下图 2 所示.在改进 March CS* 算法下, 能够检测出原算法不能检测的静态故障.

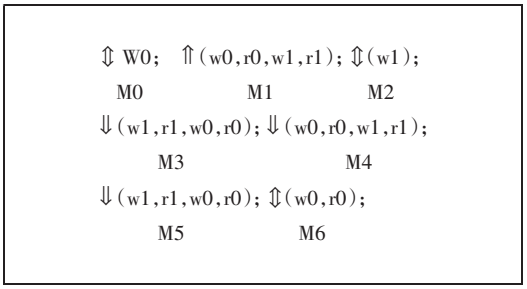

图 2　March CS* 算法

Fig.2　March CS* Algorithm

在使用 March C+算法检测动态故障时,大量的动态故障不能被检测出来,其中包括部分单元的动态写干扰故障(d_{WDF})、两个单元的动态写干扰耦合故障(dCF_{wd})等.在前述改进的 March CS* 算法能对静态故障以及部分动态故障检测,但是对于动态写干扰故障并没有进行覆盖. 动态写干扰故障的故障原语为<0*w*0*w*0/↑/–>和<1*w*1*w*1/↓/–>, 单元动态写干扰耦合故障的故障原语为<0;0*w*0*w*0/↑/–>,<1;0*w*0*w*0/↑/–>,<0;1*w*1*w*1/↓/–>和<1;1*w*1*w*1/↓/–>.要敏化这些故障, 关键要实现两步 w0 或者 w1 操作, 在 March CS* 算法的 M1 与 M4 元素中加入 w0 操作, 从而敏化该故障单元 0w0w0 状态. 同理,在 March CS* 算法的 M3 与 M5 元素中加入 w1 操作,从而敏化故障单元 1w1w1 状态.因此,得到的新算法 March CS 如图 3 所示.March CS 算法加大了对动态故障的检测.

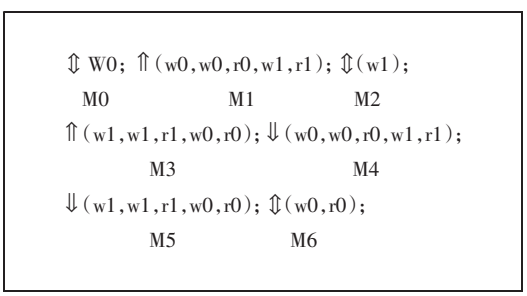

图 3　March CS 算法

Fig.3　March CS Algorithm

March CS 算法对 SRAM 进行故障检测时,故障覆盖率有了很大的提高.其一,March CS 算法能检测 March C+算法所不能覆盖的 10 种静态故障; 其二, March CS 算法能检测出原算法不能敏化的 18 种动态故障. 新算法对这些静态故障和动态故障的敏化序列和检测序列如表 1 和表 2 所示(动态故障检测只列出施主单元地址大于故障单元地址,即 $a>v$,同理,$a<v$ 也能实现).其中,"/"的左边表示对该故障的敏化序列,"/" 的右边表示对该故障的检测序列.M_i 表示算法中的第 i 个 March 元素, 符号 $M_{i,j}$ 表示算法中第 i 个 March 元素的第 j 步操作.

表 1　March CS 静态故障检测

Tab.1　Static fault detection of March CS algorithm

故障模型	故障原语	March CS 算法	
		$a > v$	$a < v$
写干扰故障	<0w0/↑/–>	$M_{1,2}/M_{1,3}$	$M_{4,2}/M_{4,3}$
	<1w1/↓/–>	$M_{3,2}/M_{3,3}$	$M_{5,2}/M_{5,3}$
干扰耦合故障	<0w0;0/↑/–>	$M_{1,1}/M_{1,3}$	$M_{4,1}/M_{4,3}$
	<0w0;1/↓/–>	$M_{1,2}/M_{1,5}$	$M_{4,2}/M_{4,5}$
	<1w1;0/↑/–>	$M_{3,2}/M_{3,5}$	$M_{5,2}/M_{5,5}$
	<1w1;1/↓/–>	$M_{3,1}/M_{3,3}$	$M_{5,1}/M_{5,3}$
写干扰耦合故障	<0;0w0/↑/–>	$M_{1,2}/M_{1,3}$	$M_{4,2}/M_{4,3}$
	<1;0w0/↑/–>	$M_{4,2}/M_{4,3}$	$M_{1,2}/M_{1,3}$
	<1;1w1/↓/–>	$M_{3,2}/M_{3,3}$	$M_{5,2}/M_{5,3}$
	<0;1w1/↓/–>	$M_{5,2}/M_{5,3}$	$M_{3,2}/M_{3,3}$

表 2　March CS 动态故障检测

Tab.2　Dynamic fault detection of March CS algorithm

故障模型	故障原语	March CS 算法
		$a > v$
写干扰故障	<0w0w0/↑/–>	$M_{1,1}, M_{1,2}/M_{1,3}$
	<1w1w1/↓/–>	$M_{3,1}, M_{3,2}/M_{3,3}$
读破坏故障	<0w0r0/↑/1>	$M_{1,2}, M_{1,3}/M_{1,3}$
	<1w1r1/↓/0>	$M_{3,1}, M_{3,3}/M_{3,3}$
写干扰耦合故障	<0;0w0w0/↑/–>	$M_{1,1}, M_{1,2}/M_{1,3}$
	<1;0w0w0/↑/–>	$M_{4,1}, M_{4,2}, M_{4,4}/M_{4,3}$
	<0;1w1w1/↓/–>	$M_{5,1}, M_{5,2}, M_{5,4}/M_{5,3}$
	<1;1w1w1/↓/–>	$M_{3,1}, M_{3,2}/M_{3,3}$
干扰耦合故障	<0w0w0;0/↑/–>	$M_{4,1}, M_{4,2}/M_{4,3}$
	<0w0w0;1/↓/–>	$M_{1,1}, M_{1,2}, M_{1,4}/M_{1,5}$
	<1w1w1;0/↑/–>	$M_{3,1}, M_{3,2}, M_{3,4}/M_{3,5}$
	<1w1w1;1/↓/–>	$M_{5,1}, M_{5,2}/M_{5,3}$
	<0w0r0;1/↓/–>	$M_{1,2}, M_{1,3}, M_{1,4}/M_{1,5}$
	<1w1r1;0/↑/–>	$M_{3,2}, M_{3,3}, M_{3,4}/M_{3,5}$
读破坏耦合故障	<0;0w0r0/↑/1>	$M_{1,2}, M_{1,3}/M_{1,3}$
	<1;0w0r0/↑/1>	$M_{4,2}, M_{4,3}, M_{4,4}/M_{4,5}$
	<0;1w1r1/↓/0>	$M_{5,2}, M_{5,3}, M_{5,4}/M_{5,5}$
	<1;1w1r1/↓/0>	$M_{3,2}, M_{3,3}/M_{3,3}$

4 March CS 算法性能仿真与验证

March CS 算法内建自测试的实现是通过 MBIST Architect 工具进行操作，对新算法进行自定义的设计，对 memory compile 所生成的 1 024*32 位 SRAM 存储器进行仿真. 通过 Verilog 代码对 SRAM 进行故障设置，然后敏化存储器的各种故障状态，用 ModelSim 对新算法进行仿真验证，检测其覆盖率.最后，通过 FPGA 来对一款低功耗的 1024X32 的 SRAM 芯片进行应用性测试.

March CS 新算法对静态故障的检测有了很大的提升，如表 3 所示，March C+算法对静态故障的覆盖率是 75%，而新算法 March CS 对静态故障的覆盖率提升为 91.67%. 图 4 是 March CS 算法对静态写干扰故障<0w0/↑/->的检测仿真，当检测到地址为 16 的存储单位的状态值为全 0，并且对该位进行 w(00000000)操作.由于在此操作下，该单元会发生故障，因此在进行 r0 操作时，该单元的状态值会由 0 转变为故障状态 1，则读出的值为(55555555). 其信号 int_failure 为高时，表明检测出该故障.

表 3 静态故障覆盖率对比

Tab.3 The comparison of static fault coverage

故障模型(FFM)	March C+	March CS
状态故障(SAF)	2/2	2/2
转换故障(TF)	2/2	2/2
写干扰故障(WDF)	0/2	2/2
读破坏故障(RDF)	4/4	4/4
伪读破坏故障(DRDF)	2/2	2/2
状态耦合故障(CFst)	4/4	4/4
干扰耦合故障(CFds)	8/12	12/12
转换耦合故障(CFtr)	4/4	4/4
读破坏耦合故障(CFrd)	6/8	8/8
写破坏耦合故障(CFwd)	0/4	4/4
伪读破坏耦合故障(CFdrd)	4/4	0/4
故障覆盖率总计	75%	91.67%

March CS 新算法对动态故障的覆盖更为全面，如表 4 所示，原 March C+算法对动态故障的覆盖率为 50.13%，而 March CS 算法对动态故障的覆盖率提升为 76.93%. 图 5 是 March CS 算法对动态干扰耦合故障<1w1w1;0/↑/->的仿真验证，在图中存储地址为 24 的单元状态为 1，March CS 算法对存储单元的状态敏化为(1w1w1,0)，由于在该操作下存储单元会发生故障，则进行 r0 操作时，所读到的状态值由 0 转变故障状态 1. 读出数据 SRAM_Q 为 e0000000，信号 int_failure 拉高表明检测出故障.

表 4 动态故障覆盖率对比

Tab.4 The comparison of dynamic fault coverage

故障模型(FFM)	March C+	March CS
写干扰故障(dWDF)	2/4	4/4
读破坏故障(dRDF)	2/4	4/4
伪读破坏故障(dRDF)	2/4	2/4
写破坏耦合故障(dCFwd)	4/8	8/8
干扰耦合故障(dCFdswxwx)	0/4	4/4
干扰耦合故障(dCFdsrxwy)	4/8	4/8
干扰耦合故障(dCFdswxrx)	4/8	6/8
读破坏耦合故障(dCFrdwxrx)	4/8	8/8
伪读破坏耦合故障(dCFdrd)	4/4	0/4
故障覆盖率总计	50.13%	76.93%

为了验证 March CS 算法的功能应用性，使用 FPGA 对一款真实的低电压的 1024X32 的 SRAM 芯片进行 March CS 算法测试. 通过使用 quartus 工具中的 SignalTap 来记录接收与发送的数据，图 6 是使用 March CS 算法对低功耗 SRAM 芯片的故障测试图，在算法 March 元素 M3 的操作中（即 tstate 为 0100 状态)，在图中地址为 71 位时，则读出来的数据 sram_output_Q 的最低位出现故障，使 int_failure 信号拉高.

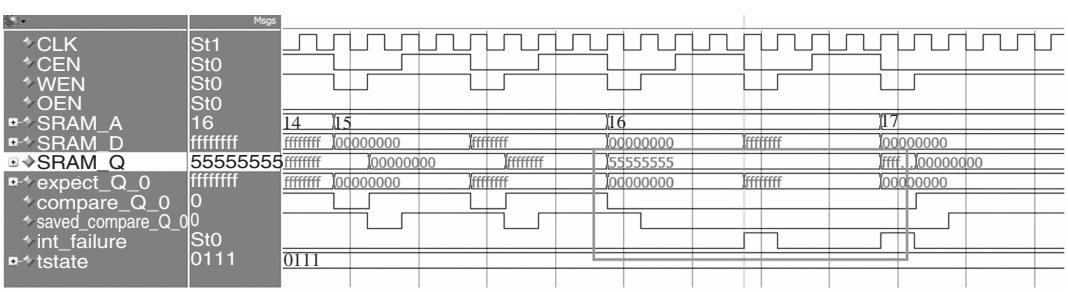

图 4 March CS 算法对静态写干扰故障检测仿真图

Fig.4 Simulation diagram of static interference fault detection in March CS algorithm

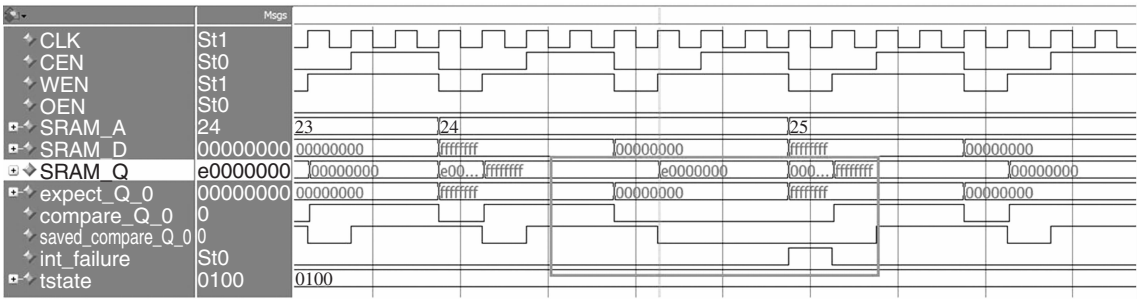

图 5　March CS 算法对动态干扰耦合故障检测仿真图

Fig.5　Simulation diagram of dynamic interference coupling fault detection in March CS algorithm

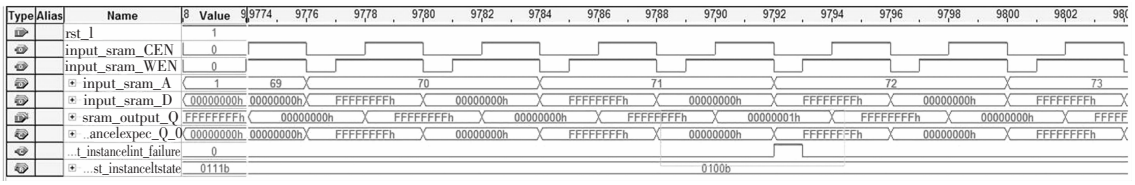

图 6　March CS 算法对 SRAM 芯片的故障测试图

Fig.6　The failure assessment diagram of March CS algorithm to SRAM chip

5　结　论

本文对 March C+算法进行改进,从而提出一种新算法 March CS 来覆盖原算法不能检测的故障.首先,改进的 March CS 算法能覆盖单个单元写干扰静态故障、干扰耦合静态故障等故障.其次,March CS 算法对存储单元动态故障检测得到了很大的优化,对于 March C+算法所不能检测动态写干扰故障、动态读破坏故障等故障均已覆盖.最后,通过对各故障模型仿真和 FPGA 对 SRAM 芯片测试来验证 March CS 的故障覆盖率.

参考文献

[1] MASNITA M I,ZUHA W H W,SIDEK R M,*et al*. The data and read/write controller for March-based SRAM diagnostic algorithm MBIST [C]//Research and Development.Serdarg:IEEE,2009:296—299.

[2] BUI T Q, PHAM L D, NGUYEN H M,*et al*. An effective architecture of memory built-In self-test for wide range of SRAM[C]//International Conference on Advanced Computing and Applications. Can Tha:IEEE,2017:121—124.

[3] 朱小莉,陈迪平,王镇道. SRAM 的一种可测性设计[J]. 湖南大学学报(自然科学版),2003,30(6):22—25.
ZHU X L,CHEN D P,WANG Z D. A kind of design-for-test for SRAM[J]. Journal of Hunan University(Natural Sciences),2003,30(6):22—25.(In Chinese)

[4] HARUTYUNYAN G,ZORIAN Y. An effective embedded test & diagnosis solution for external memories[C]//On-Line Testing Symposium.Halkidiki:IEEE,2015:168—170.

[5] WANG Y W,ZHENG Q B,YUAN Y. The improvement of march C+ algorithm for embedded memory test [C]//Computer Engineering and Technology:19th CCF Conference,NCCET 2015.Hefei:Springer,2015:31—37.

[6] SHIRUR Y J M,LAKSHMI H R,CHAKRAVARTHI V S. Implementation of area efficient hybrid MBIST for memory clusters in asynchronous SoC[C]//Fifth International Symposium on Electronic System Design. Surathkal:IEEE,2014:226—227.

[7] LU Y,ZHU Y,LI M. A MBIST controller based on JTAG interface applied in power line chip [C]//IEEE International Conference on Solid-State and Integrated Circuit Technology.Hangzhou:IEEE,2017:1404—1406.

[8] HAMDIOUR S,VAND G A J,RODGERS M. March SS: A test for all static simple RAM faults[C]// IEEE International Workshop on Memory Technology,Design and Testing. Isle of Bendor,France:IEEE,2002:95—100.

[9] 张显敞. 存储器测试算法研究及应用实现[D]. 成都:电子科技大学自动化工程学院,2013:25—39.
ZHANG X C. The design and study of memory test algorithm[D]. Chengdu:School of Automation Engineering,University of Electronic Science and Technology,2013:25—39.(In Chinese)

[10] XU Z M,SU Y P,YU Z G. SRAM BIST circuit design based on the march C-algorithm [J]. Semiconductor Technology,2007,32(3):245—247.

[11] SIVANANTHAM S,TRESA T. Built-in self-test methodology for system-on-a-chip testing [J]. Journal of Scientific & Industrial Research,2017,76(3):149—153.