

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/3943092>

Flash memory built-in self-test using March-like algorithms

Conference Paper · February 2002
DOI: 10.1109/DELTA.2002.994602 · Source: IEEE Xplore

CITATIONS
51

READS
3,065

6 authors, including:



Jen-Chieh Yeh

Industrial Technology Research Institute

52 PUBLICATIONS 711 CITATIONS

SEE PROFILE

Flash Memory Built-In Self-Test Using March-Like Algorithms

Jen-Chieh Yeh, Chi-Feng Wu, Kuo-Liang Cheng, Yung-Fa Chou,
Chih-Tsun Huang, and Cheng-Wen Wu
Department of Electrical Engineering
National Tsing Hua University
Hsinchu, Taiwan 30013

Abstract

Flash memories are a type of non-volatile memory based on floating-gate transistors. The use of commodity and embedded flash memories are growing rapidly as we enter the system-on-chip (SOC) era. Conventional tests for flash memories are usually ad hoc—the test procedure is developed for a specific design. We propose improved March-like algorithms (i.e., March FT) for both bit-oriented and word-oriented flash memory, to cover the disturbance faults derived from the IEEE 1005 Standard, as well as conventional faults. A novel flash memory fault simulator is used to analyze and generate the test algorithms. In addition, we present BIST designs for two industrial flash memories. The area overhead is only about 3% for a medium-sized flash memory.

1. Introduction

The rapid-growing market of portable electronic devices such as mobile phones and digital camera has created a growing demand for flash memories. Flash memories are a type of non-volatile memory based on floating-gate transistors. They can store charge into or remove charge from the floating gate (FG) electrically. Their in-field programmability and low-power consumption make flash memories widely used for portable devices. New generations of flash memory have higher capacity and lower access time than their predecessors. Various charge mechanisms, cell structures, and array architectures have been developed in the past few years [1]. Furthermore, flash memories can be embedded in logic systems to allow software updates. Embedded flash memory cores thus play an important role in the system-on-chip (SOC) environment.

There are many challenges for embedded flash designs. Reliability has been considered as the primary test issue for flash memories [2]. They are commonly tested for disturbance problems, including read-disturb fault, program-disturb fault, and erase-disturb fault. Conventionally, test procedure development for these disturb faults is usually ad hoc. Additionally, although many March tests have been developed in the past to test random access memories (RAMs) [3], these tests are not suitable for flash memories due to quite different memory operations. Most flash memories can do random read and random program (write 0), but cannot do random erase (write 1). Instead, they do flash erase, i.e., block erase or chip erase. Therefore, March tests for RAMs are in general not applicable to flash memories. Recently, a systematic approach for testing flash memories was proposed [4, 5], in which fault models

that capture the characteristics of disturbances and test algorithms to detect these faults are proposed. Furthermore, the Flash March algorithm proposed in [5] is March-like, which facilitates the coverage analysis and the test pattern generation.

In this paper, the IEEE 1005 Standard Definitions and Characterization of Floating Gate Semiconductor Arrays [6] is used to derive realistic fault models, then to develop March-like test algorithms for these fault models. Certain functional fault models suitable for non-volatile memories are also included. The major advantage of March-like test algorithms is their regularity, enabling easy built-in self-test (BIST) and built-in self-diagnosis (BISD) circuit design [7].

A memory fault simulator (called RAMSES-FT) has also been developed to verify the test algorithms. The fault simulator is an extension to our previous work [8]. With RAMSES-FT, we have enhanced our test algorithm generator (called TAGS [9]) to automatically generate test algorithms for flash memories with various array architectures (e.g. NOR and NAND types) and various cell structures (e.g., split-gate and stacked-gate cells). In addition, we present BIST implementations for commercial flash memories. Experimental result shows that the hardware overhead is about 3.2% for a 4Mb embedded flash core, and 2.28% for a 1Mb commodity one.

2. The Target Fault Models

Disturb mechanisms that occur in the flash memories include read-disturb, program-disturb, and erase-disturb ones, which cause abnormal data change in the victim memory cells. In Fig. 1, e.g., programming cell C11 of the NOR type flash memory (assuming that C01 and C10 are already programmed) can cause C01, C10, C12, or C21 to change its cell value, i.e., from un-programmed to programmed, or from programmed to un-programmed.

We follow the IEEE 1005 Standard definitions on disturb conditions and charge transfer parameters [6]. For ease of discussion, we consider only the NOR-type architecture with stacked-gate cells, and assume the erase operation is mass erase (i.e., whole-chip erase). Other types of flash memory can be dealt with in a similar way.

The possible disturb mechanisms for NOR-type stacked-gate flash memories include *gate program disturbance* (GPD), *gate erase disturbance* (GED), *drain program disturbance* (DPD), *drain erase disturbance* (DED), and *read disturbance* (RD). A GPD fault occurs when a cell under program

cell) causes another unprogrammed cell (affected cell) on the same word line to be programmed. The disturb mechanism is that during the programming operation on the selected cell, the control gate of the affected cell has a positive high voltage. Therefore, e.g., if the affected cell has a defective gate oxide between its channel and floating gate (FG), then hot electrons are injected from the substrate to the floating gate, changing the state of the affected cell to 0 (programmed). The electron transfer path is from the substrate to FG. In the Fig. 1, Assume that cell C12 is the affected cell and it has not been programmed. When we program cell C11, a GPD fault will cause C12 to be programmed.

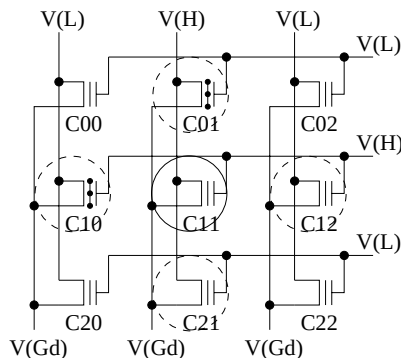


Figure 1. Disturbances in flash memory.

A GED fault occurs when a cell (selected cell) under program causes another programmed cell (affected cell) on the same word line to be erased. The disturb mechanism is different from GPD, because the affected cell has a defective tunnel oxide between the FG and control gate (CG). Fig. 1, the C10 is the affected cell and it has been programmed. When we program cell C11, a GED fault will cause cell C10 to be erased.

The DED fault occurs when a cell (selected cell) under program causes another programmed cell (affected cell) on the same bit line to be erased. The disturb mechanism is that during the programming operation on the selected cell, the bit line is at a positive high drain bias. Therefore, e.g., if the affected cell has a poor gate oxide, the electrons stored in its FG will tunnel through the gate oxide to the drain. In Fig. 1, cell C01 is the affected cell and has been programmed. When we program C11, a DED fault will cause cell C01 to be erased.

A DPD fault occurs when a cell (selected cell) under program causes another unprogrammed cell (affected cell) on the same bit-line to be programmed. The disturb mechanism is that during the programming operation on the selected cell, there will be a fairly deep depletion region formed under the affected cell. Electrons entering this depletion region can be accelerated by the electric field and be injected over the gate oxide potential barrier to the FG. In effect, the state of the affected cell will be changed to 0 (programmed). The electron transfer path is from the substrate to FG. In the Fig. 1, cell C21 is the affected cell and has not been programmed. When we program C11, a DPD fault will cause cell C21 to be programmed.

An RD fault occurs on the selected cell. Because the bias conditions for reading are the same as for programming, hot

electrons can be injected from the channel into the FG even if it is at a low gate voltage. This is commonly known as the *soft-program* problem.

Besides the disturbances, *over erase* (OE) is also a common fault for flash memories. It occurs when a cell is overly erased so that its threshold voltage is low enough to turn the cell into a depletion-mode transistor. This fault results in the following faulty behavior: 1) the faulty cell remains unprogrammed after a program operation; and 2) it causes the read operations on the same bit-line to induce a leakage current, resulting in erroneous reads. Description of this fault can also be found in [6].

Several functional fault models commonly used for testing RAMs are also considered useful for testing non-volatile memories [10], including stuck-at fault (SAF), transition fault (TF), stuck-open fault (SOF), address decoder fault (AF), and state-coupling fault (CFst). The fault models introduced in this section are adopted for testing NOR-type stacked-gate flash memories. When testing flash memories with different array/cell architectures, applicable fault models can vary. However, the fault simulation and test generation methodology to be described next can still apply.

3. Test Algorithms

Conventional March algorithms, such as MATS++ and March C- [3], are not suitable for testing flash memories. Recently, there have been some efforts to test flash memory disturbances. The March EF proposed in [4] detects flash disturbances with low test complexity. Moreover, a March-like algorithm for flash memory is proposed in [5]. In addition to flash disturbances, we target at all conventional function faults including stuck-at fault, transition fault, stuck-open fault, address decoder fault and state-coupling fault.

The March-like algorithm facilitates the coverage analysis and the test pattern generation, as well as BIST implementation. To derive the March-like algorithm, a flash fault simulator, RAMSES-FlashTest (RAMSES-FT), is developed based on RAMSES ram simulator [8].

The extended simulation engine supports flash-specific operations such as chip-erase, block-erase and program, etc. In the extensible nature, the improved RAMSES can simulate flash disturbances with proper fault descriptors [8]. Only different fault descriptors are necessary for various flash architectures. The calculation of fault coverage for different disturb faults is dependent on the layout topology of the flash memory. Accurate simulation needs detailed description of the arrangement of bit lines and word lines, and the address/data scrambling schemes. When one or more of the specifications are missing, the simulation engine will consider all possible conditions, therefore it may lead to pessimistic results, but more general tests.

The algorithm generation proposed in [9] can be extended for flash memories. The difference is the flash-erase (f) operation replaces a write-one (w1) operation, resulting in a new stand-alone operation without addressing order. Each algorithm, therefore, have to begin with flash-erase for the initialization. Following the conventional March definition

our additional rules for NOR-type stacked-gate flash memories, March FT can be presented as follows:

$$(f); \downarrow (r1, w0, r0); \uparrow (r0); (f); \uparrow (r1, w0, r0); \uparrow (r0)$$

March FT is an improvement over the Flash March algorithm proposed in [5] with two additional read operations, to the second and fifth March elements. Table 1 compares the fault coverage of our improvement with the Flash March. With two additional read operations, the coverage of SOF and CFst reaches 100%. The test complexities are also listed in Table 1, where F , P and R are the number of flash-erase program and read operations, respectively, and N denotes the number of addresses. Typically, a flash-erase operation costs much more than a program operation, and a program operation costs much more than a read operation. The cost of F and P will dominate the test time, therefore, the penalty of our extra two reads is very small.

Table 1. Simulation results and comparison.

Flash March [5]	GPD	GED	DPD	DED	OE	RD
	100%	100%	100%	100%	100%	100%
	SAF	TF	SOF	AF	CFst	
	100%	100%	50%	100%	75%	
Test length						
$2F + 2NP + 4NR$						
March FT	GPD	GED	DPD	DED	OE	RD
	100%	100%	100%	100%	100%	100%
	SAF	TF	SOF	AF	CFst	
	100%	100%	100%	100%	100%	
Test length						
$2F + 2NP + 6NR$						

In general, ad hoc tests or non-March tests such as Algorithm EF [4] have the advantage of shorter testing time, however, the geometry topology is necessary to explicitly perform row and column operations, therefore the algorithms are more complex to implement in practice. March FT shows the advantage of March-like algorithms: they are more regular, easier to generate, and covering more functional faults. March algorithms, additionally, do not rely on the array geometry or layout topology, which are more general and applicable to more types of memory.

Most memories in the modern applications are word-oriented memories. To cover intra-word coupling fault and intra-word GPD, the Cocktail-March presented in [8, 9] can be applied for the flash testing for efficient and fast test generation. Based on March FT and TAGS techniques in [9], the resultant word-oriented March FT is listed as follows for a 4-bit word-oriented flash memory.

$$\left\{ \begin{array}{l} (f); \downarrow (r1111, w0000, r0000); \uparrow (r0000); \\ (f); \uparrow (r1111, w0000, r0000); \downarrow (r0000); \\ (f); \downarrow (w0011); \downarrow (r0011); (f); \downarrow (w1100); \downarrow (r1100); \\ (f); \downarrow (w0101); \downarrow (r0101); (f); \downarrow (w1010); \downarrow (r1010) \end{array} \right\}$$

Table 2 shows the test coverage for a 4-bit word-oriented flash memory with NOR type and stack gate architecture. The test length is $2F + 2NP + 6NR + \log(m)[2F + 2NP + 2NR]$, where N is the number of memory words, and m is the word width.

4. The Built-In Self-Test Designs

In this section, we focus on the built-in self-test (BIST) design of the asynchronous flash memories with NOR-array

Table 2. Simulation results ($m = 4$).

Word-oriented March FT	GPD	GED	DPD	DED	OE	RD
	100%	100%	100%	100%	100%	100%
	SAF	TF	SOF	AF	CFst	
Test length						
$6F + 6NP + 10NR$						

and stacked-gate architecture, using Fowler-Nordheim (FN) tunneling erasing and electron injection programming mechanisms.

Fig. 2(a) shows the timing diagram of the chip erase operation. In the chip erase cycle, the chip erase signal has to be asserted before the chip enable (CEB) and the write enable (WEB). No address information is required since the memory block will be erased one by one automatically. The chip erase mode is a time-consuming process in millisecond order. In addition, there is block or section erase mode to erase one memory block at a time. The block erase requires address input to indicate the target block to erase.

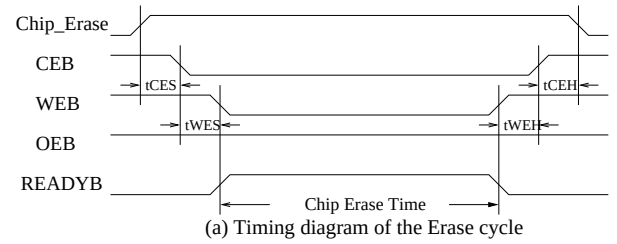
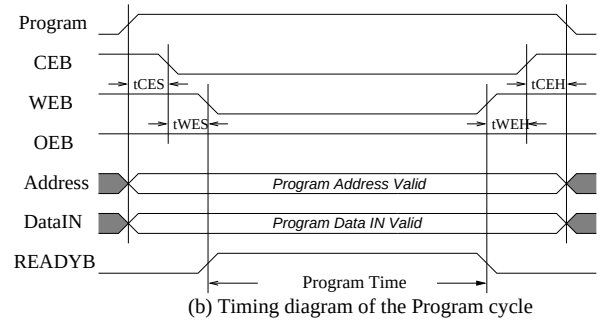
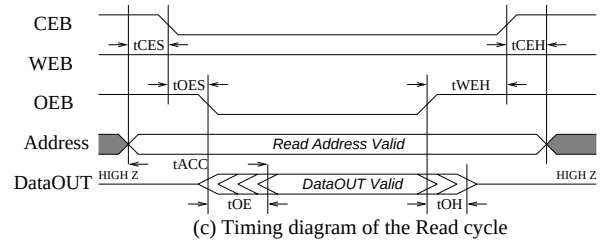


Figure 2. Typical flash memory operations.

Moreover, the timing diagram of program mode is shown in Fig. 2(b). After the program signal, CEB and WEB are asserted, the selected cells are programmed. The time of the program process is in micron-second order. After the program process, the floating gate of the programmed cell has injected electrons, resulting in logic 0 value stored in the cell. Therefore, the program operation has "AND" effect, e.g., to program "1010" into a word of "0101" will result in the content of "0000". To flip a word of "0101" to "1010", hc

erase is required before the program. Fig. 2(c) shows the timing diagram of read process. In a read cycle, the CEB and the output enable (OEB) should be asserted, then address should be applied. The output data will be valid after a output delay (tACC).

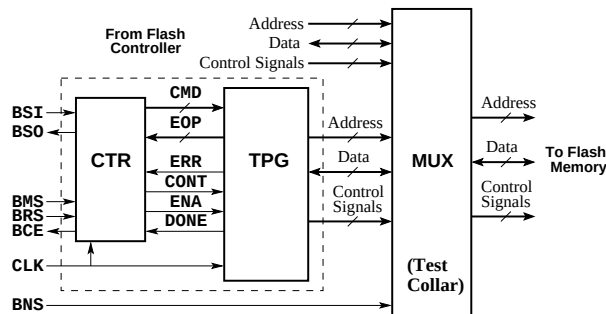


Figure 3. The proposed BIST architecture.

To implement a BIST with the chip erase, program and read operations as mentioned, the block diagram is shown in Fig. 3. The BIST design consists of three blocks: the controller (CTR), the test pattern generator (TPG) and the test collar (or multiplexer, MUX). The CTR manages test commands that are sent to the TPG for generating timing sequences and iterating through the address space. The CTR also has serial interface, i.e., BSI (BIST serial in) and BSO (BIST serial out), to communicate with the test access mechanism. Furthermore, the BMS (BIST mode select), BRS (BIST reset), BCE (BIST command end), and BNS (BIST normal select) are control signals based on [7].

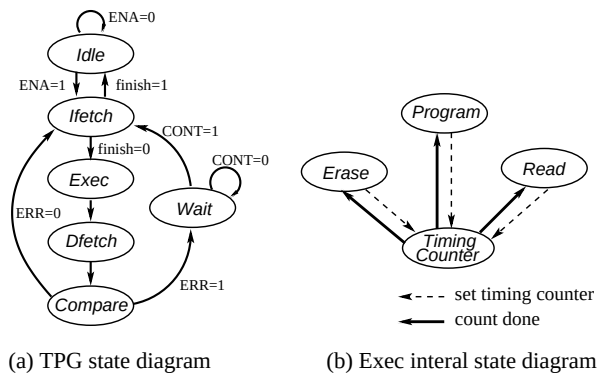


Figure 4. The state diagrams of the TPG.

The TPG generates address sequences according to the user-specific test command. Fig. 4(a) shows the state diagram of the TPG. Initially, the TPG stays in the Idle state after the reset. With proper ENA signal, the TPG goes through the Ifetch state to issue and decode the test command; the Exec state to generate the timing sequence for the required operations; the Dfetch state to read the data from the flash memory; and the Compare state to compare the data output with the fault-free one. When detecting a faulty cell, the Wait state allows the BIST to scan out the syndrome serially. Different from the conventional memory BIST, the Exec state shown in Fig. 4(b) consists of three operation, namely, erase, program and read. A large counter is required to meet the timing specification of flash memories, such as erase and program operations.

5. Experimental results and discussions

The proposed flash BIST designs facilitates the test for both the embedded flash cores and commodity flash chips, to reduce the test time and the control complexity when loaded on the external tester. In addition, BIST can support engineer test mode with little modification, helping the designer to the analysis and diagnosis. Two industrial cases are discussed in the following, including a embedded flash core and a commodity flash design, with different design-specific test mode.

Fig. 5 shows the block diagram of a typical 4Mb (512K × 8) embedded flash memory core with BIST circuitry. The flash core is a split-gate architecture with 0.25μm 2P3M CMOS technology, supporting mass erase, page erase, and byte program operations.

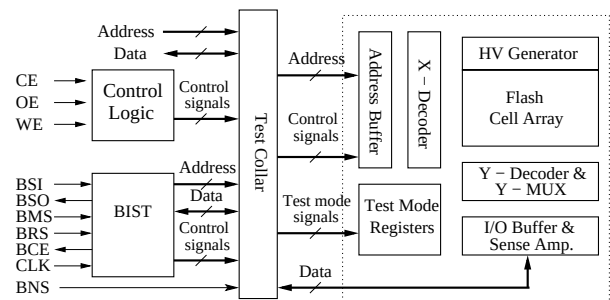


Figure 5. BIST architecture for an embedded flash memory.

During the test, the BIST accesses the memory array via the test collar, in place of the flash controller. Furthermore, an additional engineer mode controller is implemented to generate the specified controls for Test Mode Registers. With appropriate settings, the designer can assert the internal test mode easily. The engineer mode controller can decode the proper control signals to reset the test circuitry, configure the test mode, and enable the engineer test. The engineer test can be programmed in the field through the serial command interface of the BIST to retain the flexibility.

In addition, the data scramble scheme of the target flash core results in that each bit of a data word is contributed by separate memory block. Testing the word-oriented memory core is identical to testing bit-oriented memory blocks in a parallel fashion, under the assumption that intra-block coupling fault does not exist. As a result, bit-oriented test algorithm, i.e., March-FT, is sufficient for the case. Consequently, the area overhead of the flash BIST is 3.2% (see Table 3). The built-in March-FT algorithm takes about 44.6 seconds for a complete test because of the large time penalty for erase and program. However, the requirement of tester can be minimized when using the BIST, reducing the test cost further.

The other test case is the BIST for a commodity flash memory with single 3V power, which supports chip erase, block erase and word program operations. The flash memory uses stacked-gate architecture with boot sector. It is a 1Mb (128K × 8) flash memory chip using 0.4μm flash technology and 0.5μm three-metal CMOS process. The flash

electron injection for programming and Fowler-Nordheim tunneling for erase. Fig. 6 shows the block diagram of the target flash memory with BIST circuitry.

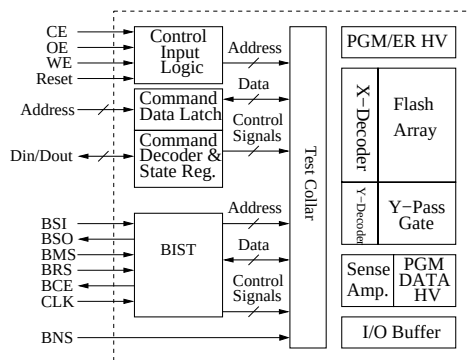


Figure 6. BIST architecture for a commodity flash memory.

Similar to the embedded case, there are design-specific considerations for the commodity chip. The memory has built-in verification circuit for program and erase operations. The verification circuit can insure correct program and erase operations in the commodity flash chip. Because of the timing uncertainty for the verification, Exec state uses a hand-shaking mechanism to determine the completeness of the program and erase operations, by detecting the assertion of the signal READYB instead of counting predefined cycles.

Unlike the previous test case, the flash chip uses address scrambling. The word-oriented test algorithm discussed in Section 2 is necessary to cover all target faults. The BIST overhead is about 2.28%, and the test time is about 13 seconds, instead of the typical test time of tens of seconds for commodity flash chip (see the last column of Table 3). Moreover, low end logic tester is sufficient to assert the BIST, with lower test cost.

Table 3. Experimental results of our BISTs.

	Embedded Flash Core	Commodity Flash Chip
Memory Size	512K × 8	128K × 8
Memory Area	5.807mm ²	8.81mm ²
Mass Erase Time	200ms	190ms
Byte program Time	20us	8us
Erase Penalty	2.5ms	1us
Program Penalty	21us	1us
BIST Frequency	100MHz	100MHz
BIST Area	0.1863mm ²	0.2017mm ²
Hardware Overhead	3.2%	2.28%
Scrambling Type	Data	Address
Test Algorithm	March FT	Word-oriented March FT
Testing Time	44.612sec	13sec

As mentioned previously, there are different flash memories with various charge mechanisms, cell structures, and array architectures [1, 6]. Instead of developing ad hoc tests that are specific to a special type of memory, we have developed a more general, March-like test methodology that can be used and reused for various flash memories. The regularity of March-like tests results in ease of test generation, built-in self-test, and built-in self-diagnosis.

For flash memories, a fault can be activated by a large number of operation. For example, the read disturbance (RD) may need hundreds, or even thousands of read operations to invert the cell from the unprogrammed state to the programmed one. The number of required operations depends on the flash memory architecture and the process technology. Our March-like algorithms can be easily modified to meet this requirement. Instead of executing operations in a test element only once for each address, operations in the test element can be repeatedly applied on the address, according to an empirical value. The value can be measured during the design verification or characterization.

6. Conclusions

Bit-oriented and word-oriented March-like algorithms—the March FT algorithms—are proposed, which cover all kinds of disturb faults defined in the IEEE 1005 Standard. The inherent regularity of March-like tests makes them especially suitable for BIST and fault diagnosis. We have implemented the BIST circuit for different flash memories. The hardware overhead is 3.2% for a 4Mb embedded flash core, and 2.28% for a 1Mb commodity one. In addition, a flash memory simulator is developed to facilitate the analysis and generation of the test algorithms. Our future work is to support more memory types and realistic fault models, and to develop a memory error catch and analysis (MECA) system for flash memories.

References

- [1] P. Pavan, R. Bez, P. Olivo, and E. Zanoni, "Flash memory cells—an overview", *Proc. of the IEEE*, vol. 85, no. 8, pp. 1248–1271, Aug. 1997.
- [2] S. Aritome, R. Shirota, G. Hemink, T. Endoh, and F. Mausouka, "Reliability issues of flash memory cells", *Proc. of the IEEE*, vol. 81, no. 5, pp. 776–787, May 1993.
- [3] A. J. van de Goor, *Testing Semiconductor Memories: Theory and Practice*, John Wiley & Sons, Chichester, England, 1991.
- [4] M. G. Mohammad, K. K. Saluja, and A. Yap, "Testing flash memories", in *Proc. 13th Int. Conf. VLSI Design*, Jan. 2000, pp. 406–411.
- [5] M. Mohammad and K. K. Saluja, "Flash memory disturbances: modeling and test", in *Proc. IEEE VLSI Test Symp. (VTS)*, Marina Del Rey, California, Apr. 2001, pp. 218–224.
- [6] IEEE, *IEEE 1005 Standard Definitions and Characterization of Floating Gate Semiconductor Arrays*, IEEE Standards Department, Piscataway, 1999.
- [7] C.-W. Wang, C.-F. Wu, J.-F. Li, C.-W. Wu, T. Teng, K. Chiu, and H.-P. Lin, "A built-in self-test and self-diagnosis scheme for embedded SRAM", in *Proc. Ninth IEEE Asian Test Symp. (ATS)*, Taipei, Dec. 2000, pp. 45–50.
- [8] C.-F. Wu, C.-T. Huang, and C.-W. Wu, "RAMSES: a fast memory fault simulator", in *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems (DFT)*, Albuquerque, Nov. 1999, pp. 165–173.
- [9] C.-F. Wu, C.-T. Huang, K.-L. Cheng, and C.-W. Wu, "Simulation-based test algorithm generation for random access memories", in *Proc. IEEE VLSI Test Symp. (VTS)*, Montreal, Apr. 2000, pp. 291–296.
- [10] A. K. Sharma, *Semiconductor Memories: Technology and Reliability*, IEEE Press, Piscataway, 1997.