2014년 3월 전자공학회 논문지 제 51권 제 3 호
Journal of The Institute of Electronics and Information Engineers Vol. 51, NO. 3, March 2014 http://dx.doi.org/10.5573/ieie.2014.51.3.061

논문 2014-51-3-7

MLC NAND-형 Flash Memory 내장 자체 테스트에 대한 연구

(MLC NAND-type Flash Memory Built-In Self Test for research)

김 진 완*, 김 태 환*, 장 훈**

(Jin-Wan Kim©, Tae-Hwan Kim, and Hoon Chang)

요 약

임베디드 시스템의 저장매체 시장의 플래시 메모리의 점유율이 증가되고 반도체 산업이 성장함에 따라 플래시 메모리의 수요와 공급이 큰 폭으로 증가하고 있다. 특히 스마트폰, 테블릿 PC, SSD등 SoC(System on Chip)산업에 많이 사용되고 있다. 플래시 메모리는 셀 배열 구조에 따라 NOR-형과 NAND-형으로 나뉘고 NAND-형은 다시 Cell당 저장 가능한 bit수에 따라서 SLC(Single Level Cell)과 MLC(Multi Level Cell)로 구분된다. NOR-형은 BIST(Bulit-In Self Test), BIRA(Bulit-In Redundancy Analysis)등의 많은 연구가 진행되었지만 NAND-형의 경우 BIST 연구가 적다. 기존의 BIST의 경우 고가의 ATE 등의 외부 장비를 사용하여 테스트를 진행해야한다. 하지만 본 논문은 MLC NAND-형 플래시 메모리를 위해 제안되었던 MLC NAND March(x)알고리즘과 패턴을 사용하며 내부에 필요한 패턴을 내장하여 외부 장비 없이 패턴 테스트가 가능한 유한상태머신(Finite State Machine) 기반구조의 MLC NAND-형 플래시 메모리를 위한 BIST를 제안하여 시스템의 신뢰도 향상과 수율향상을 위한 시도이다.

Abstract

As the occupancy rate of the flash memory increases in the storage media market for the embedded system and the semi-conductor industry grows, the demand and supply of flash memory is increasing by a big margin. They are especially used in large quantity in the smart phones, tablets, PC, SSD and Soc(System on Chip) etc. The flash memory is divided into the NOR type and NAND type according to the cell arrangement structure and the NAND type is divided into the SLC(Single Level Cell) and MLC(Multi Level Cell) according to the number of bits that can be stored in each cell. Many tests have been performed on NOR type such as BIST(Bulit-In Self Test) and BIRA(Bulit-In Redundancy Analysis) etc, but there is little study on the NAND type. For the case of the existing BIST, the test can be proceeded using external equipments like ATE of high price. However, this paper is an attempt for the improvement of credibility and harvest rate of the system by proposing the BIST for the MLC NAND type flash memory of Finite State Machine structure on which the pattern test can be performed without external equipment since the necessary patterns are embedded in the interior and which uses the MLC NAND March(x) algorithm and pattern which had been proposed for the MLC NAND type flash memory.

Keywords : Memory Test, MLC NAND Flash memory, BIST(Built-In Self Test), Fault Test

Ⅰ. 서 론

플래시 메모리의 역사는 1971년 FAMOS(Floating gate Avalanche injection MOS)라 불리는 부유 게이트 구조의 비휘발성 메모리(Nonvolatile Memory)인 EPROM(Electrically Programmable ROM)이 처음 Intel

* 학생회원, ** 정회원, 숭실대학교 컴퓨터학부
(Department of Computer, Soongsil University)
© Corresponding Author (E-mail: brodmea@ssu.ac.kr)
※ 이 논문은 2013년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (NRF-2011-0010065)
접수일자: 2013년11월19일, 수정완료일: 2014년2월27일

사에서 발표로 시작되었다. FAMOS EPROM은 자외선을 쪼여 축적된 데이터를 소거하는 방식을 취하였기 때문에 경제성 등의 문제가 있어 이러한 문제점들을 해결하고자 탄생한 것이 전기적으로 소거하는 EEPROM (Electrically Erase & Programmable ROM)이다. 통상적으로 플래시 메모리는 Flash EEPROM을 가리키며 전기적으로 데이터를 변경할 수 있는 읽기 전용의 메모리로 전기적으로 고쳐 쓰기(rewrite)가 가능하다는 점에서 ROM(Read Only Memory)과 다르다. 그리고 데이터를 고쳐 쓰기 전에 소거(Flashing)동작이 필요하고 데이터가 지워지지 않는 점에서 휘발성 메모리(Volatile Memory)인 RAM(Random Access Memory)과 비휘발성 메모리(Nonvolatile Memory)인 ROM의 중간적 성향에 해당한다. 플래시 메모리는 빠른 임의 접근속도, 작은 크기, 적은 전력 소모의 장점으로 인해 최근 일반 PC환경을 넘어서 서버타입 스토리지환경과 스마트 폰과 태블릿 PC 등 다양한 분야에서 공급과 수요가 크게 성장하고 있다. 플래시 메모리의 구조와 동작방식에 따라서 RAM과 같은 셀 배열 구조를 가져 셀별 접근이 자유로운 NOR-형과 페이지(Page)단위로 접근이 가능한 NAND-형으로 크게 구분된다.

NAND-형 플래시 메모리는 하드디스크에 비해 빠른 속도와 대용량화와 적은비용으로 인해 SSD(Solid State Drive)와 스마트 폰 등에 주로 내장되고 있다. 메모리 영역이 시스템 내에서 차지하는 비율이 증가함에 따라 신뢰도에 주는 영향이 커지기 때문에 NAND-형 플래시 메모리의 테스트의 중요성이 높아지고 있다.

[그림 1]은 2013년 까지 스마트 폰에 탑재된 NAND-형 플래시 메모리 시장규모를 나타낸다. 2008년부터 스

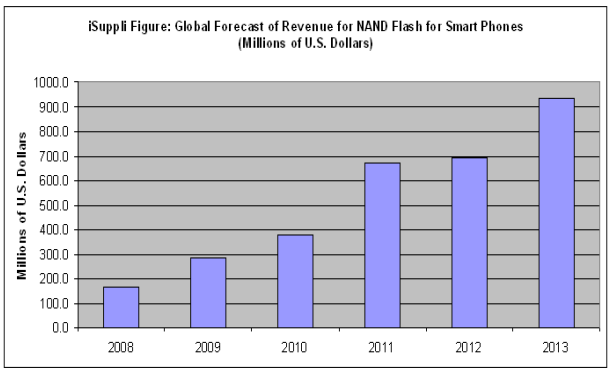


그림 1. 스마트 폰용 NAND 플래시 매출의 세계적 예측
Fig. 1. Global Forecast of Revenue for NAND Flash for Smart Phone.

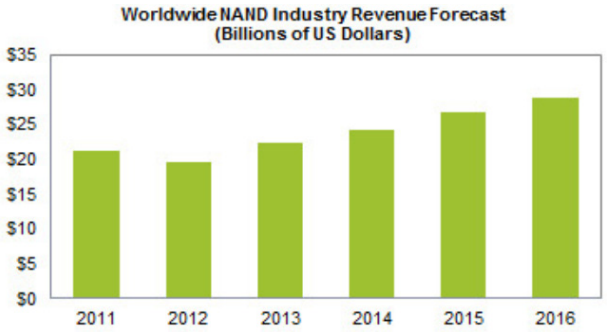


그림 2. 전 세계 NAND 산업 수익 예측
Fig. 2. Worldwide NAND Industry Revenue Forecast

마트폰의 보급률이 증가함에 따라 꾸준하게 시장규모가 커지고 있는 것을 볼 수 있다.

[그림 2]는 NAND-형 플래시 메모리의 전 세계 시장 규모 수익 예측치를 그래프로 나타낸 그림이다. 위 그림에서 보이는 바와 같이 2011년도부터 꾸준한 수요증가로 인한 시장 규모가 증가되는 것을 볼 수 있다. 최근 NAND-형 플래시 메모리의 가격 하락으로 SSD의 대중화 등의 영향으로 인해 예측 되고 있는 시장 규모보다 더 큰 시장을 형성하게 될 것으로 예상된다.

본 논문은 다음과 같이 구성되어 있다. Ⅱ장에서는 NAND-형 플래시메모리의 셀 구조와 배열 구조에 따른 특성을 분석하고, Ⅲ장에서 NAND-형 플래시 메모리의 고장유형을 살펴본다. Ⅳ장에서 MLC NAND-형 플래시 메모리의 셀 구조와 논문에 사용될 MLC NAND-March(x) 알고리즘에 대해 설명한다. Ⅴ장에서는 MLC NAND-March(x)알고리즘을 적용한 BIST구조를 제안하고 MBIST에 대한 구조의 설명과 동작 구조를 보여준다. Ⅵ장에서는 제안한 MBIST구조에 대한 실험 시뮬레이션에 대한 설명과 실험 결과를 설명한다. 마지막으로 Ⅶ장에서 제안한 MLC NAND-형 플래시 메모리를 위한 BIST에 대한 연구결과와 향후 방안에 대한 결론을 맺는다.

Ⅱ. NAND-형 플래시 메모리 구조

플로팅 게이트 셀(Floating Gate Cell)은 NAND-형 플래시 메모리를 구성하는 가장 작은 기본 단위로 [그림 3]은 플로팅 게이트 셀 구조의 단면도를 보여준다. 전통적인 MOSFET(Metal Oxide Semiconductor Field-effect Transistor)구조에 플로팅 게이트(Floating

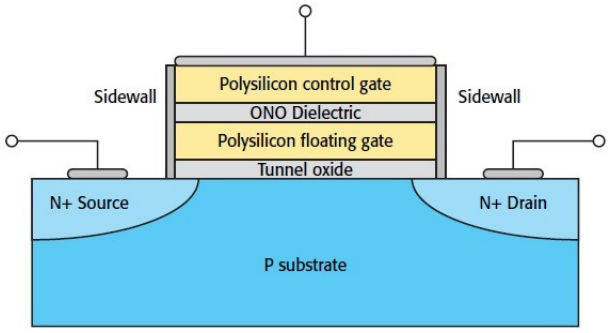


그림 3. 플로팅 게이트 셀의 구조
Fig. 3. Structure of Floating Gate Cell.

Gate)가 추가된 형태로 구성되어 있다. 플로팅 게이트는 절연체로 구성된 터널 옥사이드(Tunnel Oxide)로 불리는 옥사이드 층(Oxide Layer)에 의해 실리콘 기판(Silicon Substrate)과 절연되어 있고 컨트롤 게이트와는 ONO(Oxide Nitride Oxide)로 구성된 층에 의해 절연되어 있다.

이처럼 NAND-형 플래시 메모리는 플로팅 게이트 셀에 전하를 충전 및 방전하는 방식으로 데이터를 저장하는데 전하의 충전 양에 따른 전압 값을 가지고 데이터의 값을 나타내기 때문에 데이터의 추가 기록을 위해서는 전하들의 위치를 실리콘 기판으로 되돌려 놓아야 하는 특징이 있는데 이를 Write Once라고 한다. 이런 특징 때문에 플래시 메모리는 셀에 값을 한번만 기록을 할 수 있고 다시 기록을 하려고 하면 셀의 소거 작업을 해야만 가능하다[1~2].

[그림 4]는 NAND-형 플래시 메모리의 기본 셀 배열 구조를 나타내고 있다.

NAND-형 플래시 메모리의 셀 배열구성은 NOR-형과는 다르게 Select Line, Word Line, Bit Line으로 구

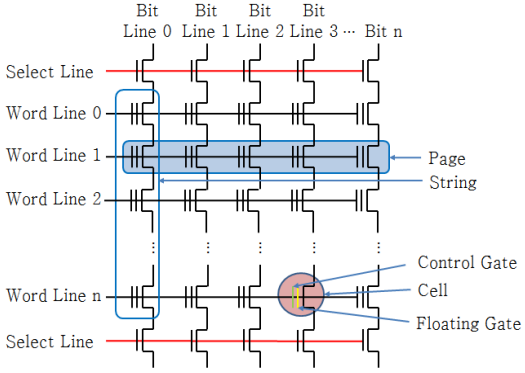


그림 4. NAND-형 플래시 메모리 셀 배열
Fig. 4. Cell Array of NAND-type Flash memory.

성 되어 있으며 각각의 Word Line을 묶어서 Page라고 부르고 각각의 Bit Line은 String이라고 부르며 배열에서 Word Line이 Select Line사이에 존재함을 볼 수 있는데 이는 Select Line을 통해 Block 단위로 메모리를 구분하기 위함이다.NAND-형의 경우 NOR-형과는 달리 메모리 셀에 데이터를 읽거나 기록(cell값을 “0”으로 기록)할 때 페이지 단위로 수행되게 되고 앞서 언급된 Write Once특성으로 인해 한번 기록된 페이지는 덮어쓰기 작업이 불가능하게 된다. 플래시 메모리의 데이터를 변경하는 동작으로는 Flash(Erase), Program, Read이 3가지 동작이 있다. Program동작은 플래시 메모리 셀에 “0”을 기록하는 동작이며 한번 Program된 셀은 재기록이 불가능하기 때문에 Flash동작을 하게 되는데 Flash동작은 소거동작으로 한번 기록된 페이지를 재사용하기 위해 Block단위로 모든 셀의 값을 “1”로 기록하는 작업을 수행한다. Block단위로 Flash를 하는 이유는 플로팅 게이트 셀이 Block내에서 하나의 실리콘 기판을 공유하고 있기 때문이다. 따라서 Flash동작을 수행할 때 실리콘 기판에 높은 고전압을 인가해야하는데 이런 경우 실리콘 기판이 공유 되어 있기 때문에 블록내의 모든 플로팅게이트 셀이 영향을 받게 된다.

Ⅲ. NAND-형 플래시 메모리 고장유형

NAND-형 플래시 메모리의 유형에 따라 알고리즘들이 제안되었다[3~6]. 그중 MLC NAND-형 플래시 메모리의 특성을 고려하고 효과적으로 오류를 검출할 수 있는 MLC NAND-March(x)알고리즘과 MLC NAND Pattern이 제안되었다[7].

[표 1]은 NAND-형 플래시 메모리에서 발생할 수 있는 고장 유형을 나타내고 있다. NAND-형 플래시 메모리에서 발생하는 교란(Disturb)고장과 기존의 RAM Style에서 사용되는 고장모델이 있다[3, 8~9]. 각 고장모델에 대한 설명은 다음과 같다.

[그림 5]의 교란 고장은 플래시 메모리에서만 발생하는 고장으로 결함 셀은 같은 워드라인이나 비트라인의 특정 셀에 접근될 때 교란을 받을 수 있게 된다. 같은 워드라인의 특정 셀에 접근할 때 결함 셀이 예기치 않게 소거(Erase)되거나 프로그램(Program) 되는 고장을 WED(Word-Line Erase Disturb)와 WPD(Word-Line Program Disturb)라고 하고, 마찬가지로 비트라인의 경

표 1. Disturb 고장과 RAM-Style 고장
Table 1. Disturb Faults and RAM-Style Faults.

Disturb 고장	
Word-Line Erase Disturb(WED)	0 → 1
Word-Line Program Disturb(WPD)	1 → 0
Bit-Line Erase Disturb(BED)	0 → 1
Bit-Line Program Disturb(BPD)	1 → 0
RAM-Style 고장	
Stuck-At Fault(SAF)	셀이 0이나 1로 고정됨
Transition Fault(TF)	셀이 0(1) → 1(0)로 변화 실패
Stuck-Open Fault(SOF)	셀에 접근 불가
Address Decoder Fault(AF)	셀과 주소가 1대 1 매칭 안됨

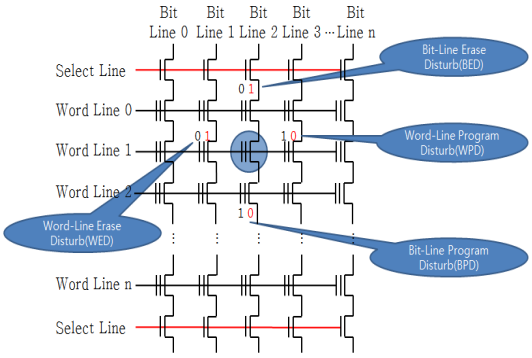


그림 5. 교란 고장
Fig. 5. Disturb Faults.

우에도 같은 비트라인의 특정 셀에 접근할 때 결함 셀이 예기치 않게 소거되거나 프로그램 되는 고장을 BED(Bit-Line Erase Disturb)와 BPD(Bit-Line Program Disturb) 고장이라고 한다[3, 8].

그리고 NAND-형 플래시메모리에서도 기존 RAM 스타일 고장이 각각의 셀에서 발생할 수 있다. 예를 들면 램 스타일 고장으로 SAF(Stuck-at-Fault), TF(Transition-Fault), SOF(Stuck-Open-Fault) 그리고 AF(Address-Fault)를 포함하고 있다.

[그림 6]에서와 같이 SAF를 갖는 셀은 셀의 값이 "0"에서 "1"로 또는 "1"에서 "0"으로 쓰기를 했을 때 하나의 값으로 고정된다.

[그림 7]과 같이 TF를 가진 셀은 자신이 가지고 있는 현재 값 "0"또는"1"에서 "1"또는"0"로 변환하지 못하고 한 가지 상태로 고정이 된다.

SOF를 가진 셀은 관련된 라인들의 전체적 인 영향을 끼쳐서 SOF고장으로 인해 접근할 수 없게 되서 감

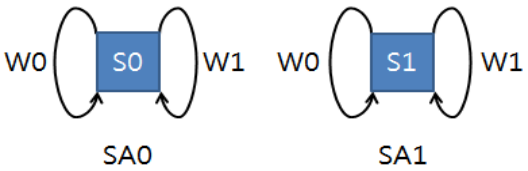


그림 6. RAM Style 고장 : 고착고장
Fig. 6. RAM Style Fault : SAF.

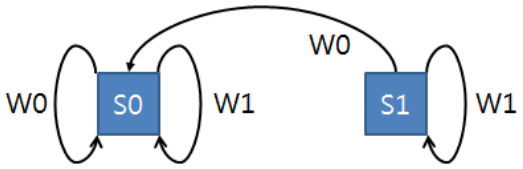


그림 7. RAM Style 고장 : 천이고장
Fig. 7. RAM Style Fault : TF.

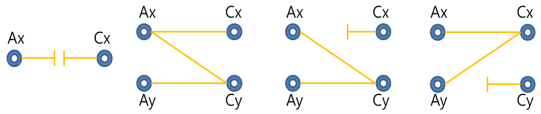


그림 8. RAM Style 고장 : 주소 고장
Fig. 8. RAM Style Fault : AF.

지증폭기에 의해 반환된 값이 이전 페이지의 값으로 출력된다.

[그림 8]은 AF(Address Decoder Fault)를 나타내는 그림이다. AF는 셀이 주소가 1대 1 Matching을 하지 못하는 것으로 4가지 형태를 띠고 있다.

Ⅳ. MLC NAND-March(x) 알고리즘

SLC NAND-형 플래시는 하나의 셀에 1bit가 저장되는 반면 MLC NAND-형 플래시는 하나의 셀에 2bit가 저장된다[10].

[그림 9]와 같이 MLC NAND-형 플래시 메모리는 3개의 Reference 전압을 기준으로 00, 01, 10, 11의 상태를 갖는다.

[표 2]는 MLC NAND-형 플래시 메모리의 4가지 전

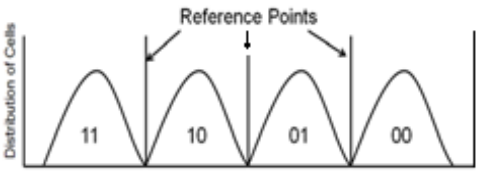


그림 9. MLC 기준 전압
Fig. 9. MLC Reference Points.

표 2. MLC 셀 상태와 값의 표기
Table 2. MLC Cell State and Notation of Value.

Value	Programmed	Notation
00	Fully Programmed	A
01	Partially Programmed	B
10	Partially Erased	C
11	Fully Erased	D

표 3. MLC NAND-March(x) 알고리즘
Table 3. MLC NAND-March(x) Algorithm.

알고리즘	Element
March(x)	{(e); ↓(r11, $_wMLC_{NAND(x)}$, $_rMLC_{NAND(x)}$); ↕($_rMLC_{NAND(x)}$); (e); ↓(r11, $_w\overline{MLC_{NAND(x)}}$, $_r\overline{MLC_{NAND(x)}}$); ↕($_r\overline{MLC_{NAND(x)}}$); (e); ↓(r11, w00, r00)}

표 4. MLC NAND-March(x) 패턴 조합
Table 4. MLC NAND-March(x) Pattern combination.

	A	B	C	D
A		A-B	A-C	A-D
B	B-A		B-C	B-D
C	C-A	C-B		C-D
D	D-A	D-B	D-C	

압 상태를 기준으로 생기는 플로팅 게이트의 저장 비트 값을 나타내고, 해당 값에 따른 상태를 부르는 명칭과 다음에 나와 있는 MLC NAND-March(x)패턴을 보기 편하게 나타낸 표기법을 함께 나타내고 있다. 예를 들면 Fully Programmed와 Partially Programmed에 대한 패턴을 표기할 때 00, 01이 아닌 A와 B로 표기한다.

본 논문에서는 기존에 제안된 MLC NAND-March(x) 알고리즘[7]과 MLC NAND-형 플래시메모리 패턴[7]을 ROM에 저장해두고 ROM에 저장된 패턴을 이용하여 테스트를 진행하게 된다.

MLC NAND-March(x)알고리즘의 각각의 Element는 FSM으로 구성하여 동작된다. 다음 [표 3]은 MLC NAND -March(x)알고리즘의 구성요소를 보여준다.

MLC NAND-March(x)알고리즘의 경우 셀의 값에 따라 가지는 MLC NAND-형 플래시 메모리의 특징을 적용한 패턴을 가지게 되며 이러한 패턴의 반전을 $\overline{MLC_{NAND}}$로 표기했다.

[표 4]는 MLC NAND-March(x) 알고리즘에 사용할 패턴의 조합을 나타내는 표이다.

다음 [그림 10]과 [그림 11]은 앞에 표에서 언급한 MLC NAND-형에 사용될 패턴 중 A-B형 패턴과 반전

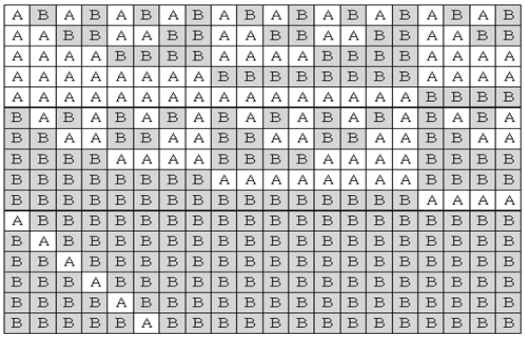


그림 10. MLC NAND(A-B) 패턴
Fig. 10. MLC NAND(A-B) Pattern.

그림 11. 반전 MLC NAND(A-B) 패턴
Fig. 11. Inverse MLC NAND(A-B) Pattern.

된 패턴의 예이다.

위와 같이 MLC NAND(x) 패턴과 Inverse MLC NAND(x) 패턴을 사용하는 이유는 AF, SAF, SOF, TF 검출 뿐만 아니라 플래시 메모리에서 발생 가능한 Disturb 고장까지 검출하기 위해 반전된 패턴이 필요하다.

Ⅴ. 제안하는 MBIST 구조

본 논문의 제안하는 MBIST 구조는 MLC NAND-March(x) 알고리즘을 사용하기 위해 FSM기반의 MBIST 구조[11~12]을 사용한다.

다음 [그림 12]는 제안하는 MBIST 구조의 블록 다이어그램을 보여준다. 제안하는 MBIST는 크게 BIST Logic Controller, Address Generator, ME FSM, Comparator, ROM으로 구성 되어 있다.

BIST Logic Controller에 MBIST를 사용하겠다는 BIST Enable신호가 가해지면 MLC NAND- March(x) 테스트 알고리즘의 March Elements를 수행하는 ME Start 신호를 ME FSM모듈로 보내면서 MLC

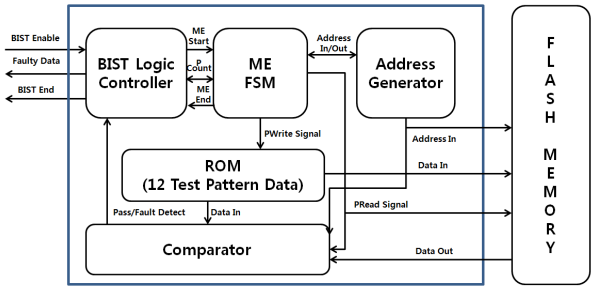


그림 12. FSM 기반 MBIST 구조
Fig. 12. FSM Based MBIST Architecture.

NAND-March(x) 테스트 알고리즘 각각의 March Elements에 대한 State를 수행하게 된다. 그리고 BIST Logic Controller는 ME FSM 모듈로부터 ME End 신호를 받을 때 마다 MLC NAND -March(x) 테스트 알고리즘이 수행했던 테스트 패턴을 제외한 다른 테스트 패턴들도 수행하게 된다. 예를 들어 [표 5]의 첫 번째 테스트 패턴으로 A-B를 사용해 MLC NAND-March(x) 테스트 알고리즘을 수행하고 ME FSM모듈로부터 ME End 신호를 받게 되면 그 다음으로 A-C조합 테스트 패턴을 사용해서 다시 MLC NAND-March(x) 테스트 알고리즘을 수행하게 된다.

[표 5]에 보여 지는 테스트 패턴을 모두 수행하면서 테스트 패턴 카운트인 P Count가 “12”가 되면 테스트를 수행하는 동안 결함이 검출되었는지 Comparator 모듈을 통해 플래시 메모리에 Write한 테스트 패턴의 Data와 ROM의 테스트 패턴을 비교해서 확인을 한다. 만약 Pass/Fault Detect 신호를 통해 결함이 검출 되지 않으면 정상적으로 테스트가 완료되어 MBIST를 종료하는 BIST End신호를 내보내면서 MBIST가 종료하게 된다. 만약 결함이 검출된다면 Faulty Data로 결함 데이터를 함께 출력하면서 BIST End신호를 내보내고 MBIST를 종료한다.

ME FSM 모듈은 MLC NAND-March(x) 테스트 알고리즘 전체의 동작을 수행하기 위한 FSM을 가지고 있으며 테스트를 수행하고 플래시 메모리의 페이지 단위로 동작을 반복 수행하기 위한 MLC NAND-March(x) 테스트 알고리즘의 March Element로 구성되어 있다.

[그림 13]은 MLC NAND-March(x)알고리즘 수행에 대한 ME FSM의 동작 구조를 나타낸다. ME FSM모듈 안에 Test FSM에 State는 BIST Logic Controller에서

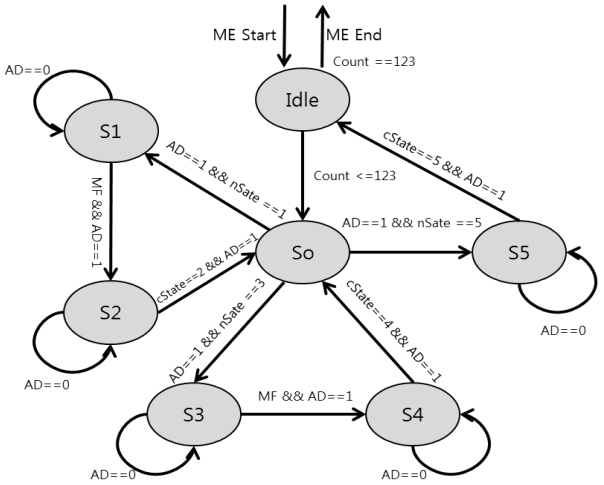


그림 13. Test FSM 구조
Fig. 13. Test FSM Structure.

보내는 ME Start신호를 받는 Idle State, 메모리 Erase 동작과 Erase된 메모리 Data를 Read동작을 수행하는 S0 State, 주소를 감소시키면서 테스트 패턴을 Write하고 Read동작을 수행하는 S1 State, 주소 증감에 상관없이 S1의 Read동작을 한 번 더 수행하기 위한 S2 State, 주소를 감소시키면서 반전된 테스트 패턴을 Write하고 Read동작을 수행하는 S3 State, 주소증감에 상관없이 S3의 Read 동작을 한 번 더 수행하는 S4 State 그리고 “00”을 Write하는 동작과 “00”을 Read하는 동작을 수행하는 S5 State로 총 7개의 State로 구성되어 있다.

먼저 ME Start신호가 들어오면 그 신호를 받아 ME FSM은 Idle상태에서 S0상태로 전이가 되고 메모리 Erase를 실행한 후 정상적으로 Erase가 되었는지 주소를 감소하면서 “11” Read 동작을 수행한다. 메모리 주소가 끝날 때까지 수행했다는 AD=1 신호와 nState=1 신호가 동시에 충족할 때까지 S0의 모든 동작을 수행하게 되고 두 신호가 만족하게 되면 다음 State인 S1로 천이되게 된다.

S1에서는 주소를 감소시키면서 MLC NAND(x) 테스트 패턴을 AD 신호가 “1”이 될 때까지 Write Pattern NAND(x)를 먼저 수행하고 Read Pattern NAND(x)를 수행한 후 MF신호가 “1”이 되면 AD 신호와 MF 신호를 통해 S1이 종료되고 S2로 천이된다.

S2에서는 Read Pattern NAND(x)를 주소 증감에 상관없이 수행하고 cState=2와 AD=1이 성립되면 S0로 상태가 천이되게 된다. 다시 S0상태로 천이되면 앞에서

설명한 Erase동작과 “11”Read동작을 수행 하고 이전에 들어온 cState신호의 값에 따라 다음 천이할 State를 정하게 된다. 예를 들면 cState=2와 AD=1신호를 통해 S2에서 S0로 천이가 되었고 S0의 State에서 동작을 모두 수행했다면 S3로 천이가 되게 된다.

S3에서는 S1과 유사하나 반전된 테스트 패턴을 이용하여 결함을 검출을 확인 하게 된다. 주소를 감소하면서 MLC NAND(x) 패턴을 AD신호와 MF신호가 “1”이 될 때까지 Write를 수행하고 AD신호와 MF신호가 “1”이 되면 다시 주소를 감소하면서 MLC NAND(x)패턴을 Read 동작을 수행하고 마찬가지로 AD신호와 MF신호가 “1”이 되면 다음 상태인 S4로 천이되게 된다.

S4에서는 주소 증감에 상관없이 S3의 동작이 정상적으로 수행되었는지 확인하기 위해 한 번 더 반전된 MLC NAND(x)패턴을 사용하여 Read동작을 수행한다. 그리고 cState=4와 AD=1 신호가 만족하게 되면 S4를 종료하고 다시 S0의 상태로 천이되게 된다.

마지막으로 S0에서 S5로 상태가 천이가 되면 최종적으로 모든 셀 영역에 주소를 감소하면서 AD신호가 “1”이 될 때까지 Write “00”을 수행하고 AD신호와 MF신호가 “1”이 되면 같은 방법으로 Read “00”을 수행하게 된다. 모든 동작이 종료 되어 cState=5와 AD=1이 만족될 때 Idle상태로 상태가 다시 천이되면서 한 Block에 대한 테스트를 종료하게 되고 Block Count가 증가하게 된다. 설정된 Block Count를 충족할 때까지 같은 동작을 반복 수행하게 되며 Count를 충족하게 되면 Idle상태에서 더 이상 진행되지 않고 BIST Logic Controller으로 ME End신호를 내보내게 된다. P Count가 증가하게 되면 다른 테스트 Pattern을 이용하여 FSM을 다시 수행하게 된다.

다음 [표 5]는 ME FSM의 각 상태에 대한 수행하는 동작에 대해 보여주고 있다.

[표 5]와 같이 Idle상태에서는 FSM이 동작하게 되면 Block단위의 MLC NAND-March(x) 테스트 알고리즘의 수행을 위해 Count를 하고 일반적으로 대기를 하고 있다가 ME Start신호를 받으면 S0상태로 천이 되면서 위에서 설명한 각각의 State를 동작함으로써 ME FSM이 동작하게 된다. 또한 테스트 대상 메모리의 크기에 따라 Block Count의 수는 사용자가 설정할 수 있다.

[그림 14]는 S1 State에서 두 개의 MLC NAND-March(x) Elements 수행을 위한 S1의 FSM을 보여주고 있다.

표 5. Test FSM 상태 동작
Table 5. Test FSM State Operation.

State	Operation
Idle	ME End, Count
S0	(e); ↓(r11);
S1	↓($_{w}MLC_{NAND(x)}$, $_{r}MLC_{NAND(x)}$);
S2	↕($_{r}MLC_{NAND(x)}$);
S3	↓($_{w}\overline{MLC_{NAND(x)}}$, $_{r}\overline{MLC_{NAND(x)}}$);
S4	↕$_{r}\overline{MLC_{NAND(x)}}$
S5	↓(w00, r00);

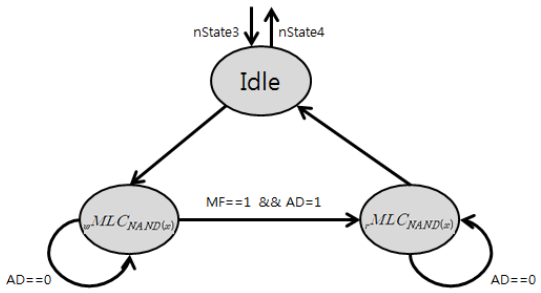


그림 14. FSM S1 상태
Fig. 14. FSM S1 State.

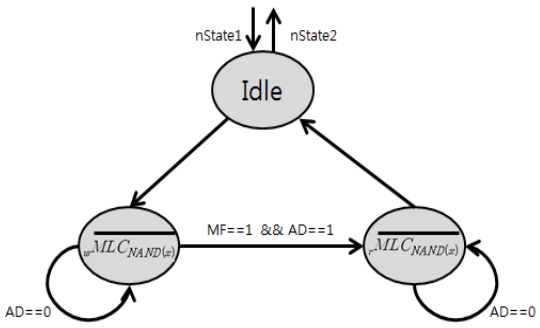


그림 15. FSM S3 상태
Fig. 15. FSM S3 State.

S1은 주소를 감소시키면서 MLC NAND- March(x) Elements 동작을 수행하는데 먼저 테스트 패턴 Write 동작을 수행하고 AD신호가 “1”이 되면 테스트 패턴을 Read하면서 다시 AD신호가 “1”이 되면 Idle상태로 이동되어 다음 State인 S2로 천이되게 된다.

[그림 15]는 [그림 14]와 같은 S3 State의 FSM을 보여준다.

S3는 S1과 비슷하게 동작하는데 MLC NAND-March(x) Elements 동작을 위한 반전된 테스트 패턴을

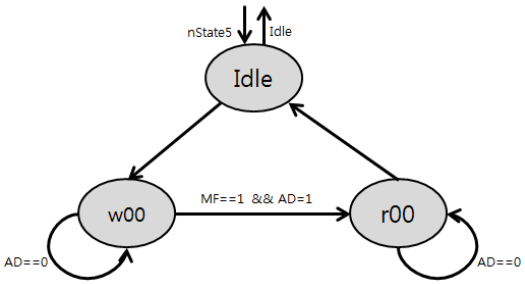


그림 16. FSM S5 상태
Fig. 16. FSM S5 State.

사용하여 반전된 테스트 패턴을 Write하면서 AD신호가 “1”이 되면 반전된 테스트 패턴을 Read하게 된다. 그리고 Read동작을 수행하면서 AD신호가 “1”이 되면 Idle상태로 이동되어 다음 State인 S4로 천이되게 된다.

[그림 16]도 [그림 14]와 [그림 15]와 비슷하게 두 개의 동작을 수행하기 위한 S5의 FSM을 보여준다.

S5는 각 셀에 “00”을 Write하는 동작을 수행하다가 AD신호가 “1”이 되면 r00 State로 천이되어 “00”을 Read하는 동작을 수행한다. 마찬가지로 AD신호가 “1”이 되면 Idle상태를 통해 Test FSM의 Idle 상태로 천이가 되게 된다.

위의 제안하는 FSM과 ROM에 저장해둔 테스트 패턴을 사용하여 MLC NAND-March(x)알고리즘을 수행하면서 NAND-형 플래시 메모리에서 발생할 수 있는 SAF, TF, SOF, CF와 플래시 메모리의 고유 고장인 Disturbance고장을 검출할 수 있는 Memory Built-In Self Test 구조를 구성하였다.

VI. 실 험

제안하는 FSM 기반의 MBIST 구조는 VerilogHDL로 기술하여 구현하였으며, 시뮬레이션은 Xilinx사 ISE에 포함된 ISim 시뮬레이터를 사용하여 RTL시뮬레이션을 하였다.

시뮬레이션은 [표 5]에 나타난 MLC NAND-March(x) 패턴 조합 중 (A-B)패턴을 가지고 진행한 결과 이며 [그림 17]은 FSM중 [표 5]의 S0에 대한 소거(Flash)동작에 대한 시뮬레이션 결과이다. 시뮬레이션 결과와 같이 ME Start신호가 들어오게 되면 Idle상태에서 S0상태로 천이되면서 모든 메모리의 값들이 “11”로 Erase되는 것을 확인할 수 있다. 그리고 주소 끝까지 실행했다

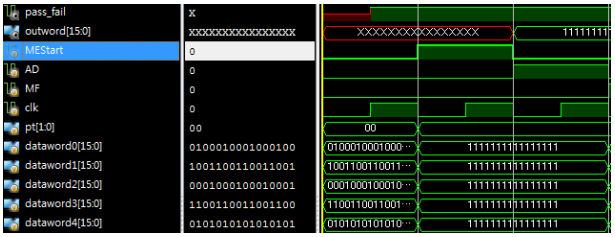


그림 17. FSM S0 시뮬레이션 결과
Fig. 17. FSM S0 Simulation Result.

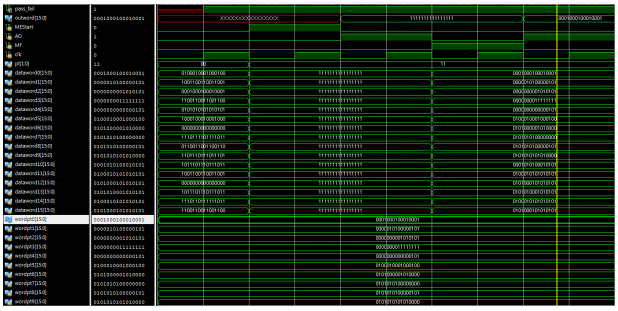


그림 18. FSM S1/S2 시뮬레이션 결과
Fig. 18. FSM S1/S2 Simulation Result.

는 AD신호가 “1”이 되면 Erase동작이 정상적으로 수행했는지 확인하기 위해 Read를 수행하게 되며 시뮬레이션의 outword의 값이 “1111111111111111”로 Read되며 정상 동작하는 것을 확인할 수 있다.

그리고 [그림 18]은 S0 이후에 S1과 S2의 시뮬레이션 결과를 보여준다. 먼저 S0 이후에 Erase된 메모리의 셀 값에 (A-B)조합의 테스트 패턴을 S1에서 Write하게 된다.

위의 시뮬레이션에서 보이는 것처럼 S0에서 “11”값을 Read가 완료되었다는 MF신호가 “1”이 되면 다음 State인 S1으로 천이되게 되고 (A-B) 테스트 패턴의 값들이 메모리로 Write되는 것을 확인할 수 있다. 그리고 (A-B) 테스트 패턴의 값이 메모리의 주소 끝까지 정상적으로 Write되면 AD신호가 “1”이 되고 메모리의 값을 Read하게 된다. 시뮬레이션의 outword의 값이 “0001000100010001“로 결함 없이 정상적으로 (A-B) 테스트 패턴이 메모리에 Write된 것을 확인 할 수 있다.

[그림 19]는 S0 이후에 S3과 S4의 시뮬레이션 결과를 보여준다. 앞에서 설명한 S1과 S2의 시뮬레이션 결과와 마찬가지로 S0의 Erase된 메모리의 셀에 반전된 (A-B)조합의 테스트 패턴을 S3에서 메모리로 Write하게 된다. [그림 19]에서 보이는 것처럼 S0에서 “11”값으로 Read가 완료되었다는 MF신호가 “1”이면 다음 상태

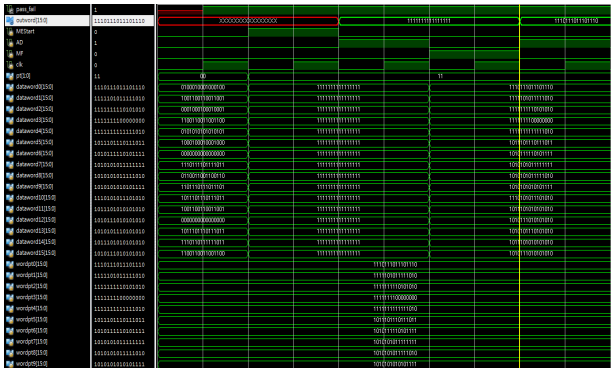


그림 19. FSM S3/S4 시뮬레이션 결과
Fig. 19. FSM S3/S4 Simulation Result.

인 S3상태로 천이되게 된다. 그리고 S3 상태에서는 반전된 (A-B) 테스트 패턴을 메모리에 Write하게 된다. 그리고 반전된 (A-B) 테스트 패턴의 값이 메모리의 주소 끝까지 정상적으로 Write하게 되면 AD 신호가 "1"이 되고 메모리의 값을 Read하게 된다.

위의 [그림 19]에서 보이는 시뮬레이션처럼 Read되는 outword의 값이 "1110111011101110"으로 S1에서 사용했던 (A-B)테스트 패턴의 반전된 값인 것을 확인할 수 있다.

마지막으로 [그림 20]의 시뮬레이션은 S0 에서 S5로 천이되어 Write "00"과 Read "00"을 수행하는 시뮬레이션을 보여주고 있다. S5도 다른 State와 마찬가지로 S0에서 Erase동작을 먼저 수행하고 정상적으로 Erase되었는지 확인을 위해 값들을 "11"을 Read한 다음 동작하게 된다. S0의 동작이 완료되면 MF 신호가 "1"이 되고 S5로 천이되면서 모든 메모리에 "00"값을 Write하게 된다. 그리고 메모리의 주소 끝까지 "00"을 Write하게 되면 AD신호가 "1"이 되고 모든 메모리 셀의 값이 정상

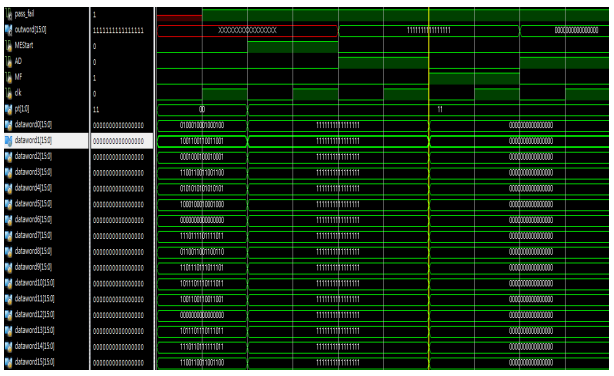


그림 20. FSM S5 시뮬레이션 결과
Fig. 20. FSM S5 Simulation Result.

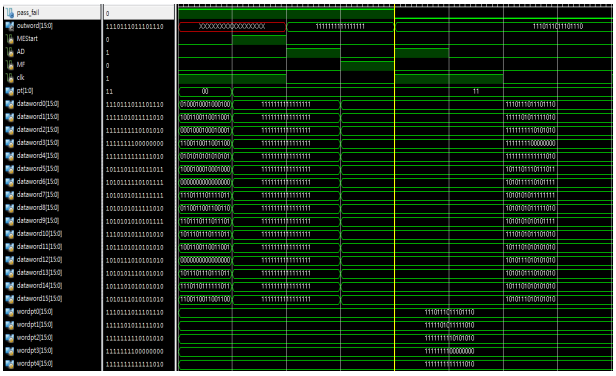


그림 21. FSM 시뮬레이션의 결함 검출
Fig. 21. Fault Detection of FSM Simulation.

적으로 "00"이 되었는지 확인을 위해 셀들의 값을 "00"으로 Read하게 된다. [그림 20]의 시뮬레이션을 보면 출력되는 outword의 값이 "0000000000000000"으로 정상 출력이 되는 것을 확인할 수 있다.

앞에서 보여주는 시뮬레이션들은 결함이 발생하지 않은 것으로 가정하고 시뮬레이션을 동작시킨 것이다. 결함의 발생여부를 확인하기 위한 신호는 pass_fail 신호로 결함이 발생하지 않으면 "1"이 출력되고 결함이 발생하면 "0"을 출력한다.

[그림 21]은 고장이 발생했을 때의 시뮬레이션 결과를 보여준다.

[그림 21]의 결과를 설명하면 먼저 S0에서 Erase를 수행하고 메모리의 값들이 정상적으로 Erase되었는지 확인하기 위해 "11"을 Read한다. MF 신호를 통해 S0에서 S3 상태로 천이되게 되고 S3에서 반전된 (A-B) 테스트 패턴을 메모리에 Write하게 된다. 그리고 메모리의 주소 끝까지 수행하게 되면 AD신호가 "1"이 되어 메모리의 값이 정상적으로 반전된 (A-B) 테스트 패턴이 Write되었는지 확인하기 위해 반전된 (A-B) 테스트 패턴의 값으로 Read 동작을 수행하게 된다. 그런데 메모리의 첫 번째 워드에서 반전된 (A-B) 테스트 패턴의 값으로 "1110111011101110"을 Write했는데 Read해서 출력된 값은 "1110111011001101"로 잘못된 Date를 Read하는 것을 확인할 수 있다. 제안하는 MBIST에서 Comparator는 ROM의 패턴과 출력된 메모리의 패턴을 비교하는데 이렇게 결함이 발생하게 되면 pass_fail신호로 "0"을 출력하게 된다. 그러면 MBIST는 정지하게 되고 전체 테스트를 종료하게 된다.

위의 실험 결과로 얻은 Fault Coverage를 이전 연구와 비교하면 다음과 [표 6]과 같다.

표 6. 제안한 Memory BIST의 Fault coverage
Table 6. Memory BIST Fault coverage of the proposed.

	previous[3]	previous[4]	proposed
SAF	100 %	100 %	100 %
SOF	50 %	100 %	100 %
TF	100 %	100 %	100 %
AF	100 %	75 %	75 %
DF	100 %	100 %	100 %
Test length	2e+3NR+2NP	3e+9NR+3NP	3e+8NR+3NP

[표 6]의 결과를 보면 이전에 제시된 [3]과 비교하면 물리적 결함 고장인 AF 검출이 다소 낮지만 SOF의 검출이 훨씬 더 높다. 또한 [4]에비해서 Test Length는 한번의 Read동작이 줄어들었지만 동등한 Fault 검출률을 얻을 수 있기 때문에 Test 시간이 감소하는 이득을 얻게 된다. 또한 기존에 제시된 [3]은 NOR-형 플래시 메모리를 테스트하는 알고리즘이고, [4]는 NAND-형 플래시 메모리에 관한 알고리즘이지만 SLC 플래시 메모리를 위한 방법임을 감안할 때 MCL 적용하기가 어렵지만 제안하는 방법은 MLC 구조에 대한 Test가 가능하다는 강점이 있을 수 있다.

Ⅶ. 결 론

MLC NAND-형 플래시 메모리는 기존 저장장치에 비해 빠른 속도와 적은 전력 등의 이유로 품질향상을 위한 연구가 계속되고 있다. 본 논문에서는 대용량화가 용이하고 생산단가가 저렴해 많은 보급이 늘어나고 있으며 신뢰도 측면에서 중요한 역할을 차지하는 MLC NAND-형 플래시 메모리의 구조적 특성에 맞춰 고장을 검출 할 수 있도록, 기존에 제안된 March(x)알고리즘을 이용하여 Memory Built-In Self Test를 구성하였다.

본 논문에서 제안한 연구는 MLC NAND-형 플래시 메모리에 MBIST를 적용하기 위한 시도로 MLC NAND-March(x) 패턴 저장 공간으로 인해 오버헤드가 발생하지만 MLC NAND-형 플래시를 테스트할 수 있고 높은 Fault 검출에 따른 신뢰도 향상이라는 장점이 있을 수 있으며 MLC NAND-형 플래시 메모리의 수율 향상에 기여 할 수 있을 것이다.

지속적인 연구로 향후에는 테스트 패턴을 저장하지 않고 필요한 패턴을 생성할 수 있는 패턴생성기를 이용한 고장검출로 조금 더 빠르고 많은 종류의 Fault를 검출하고 오버헤드를 줄일 수 있는 연구가 필요하다.

REFERENCES

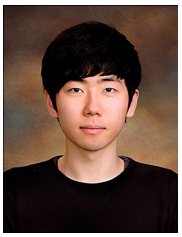
[1] Joe E. Brewer, Manzur Gill, “Nonvolatile MEmory Technologies with Emphasis on Flash“ Institute of Electrical and Engineers, 2008.
[2] Pavan P, Bez R, Olivo P, Zanoni E. “Flash memory cells－.an overview,” Proc IEEE 1997;85(8):1248 - .71
[3] M. G. Mohammed and K. K. Saluja, “Flash Memory Disturbances : Modeling and Test,” Proceedings of 19th VLSI Test Symposium, 2001, pp 218-224, April 2001.
[4] Tei-Wei Kuo, Po-Chun Huang, Yuan-Hao Chang, Chia-Ling Ko, Chih-Wen Hsueh, “An efficient fault detection algorithm for NAND flash memory,” Proc ACM SIGAPP Applied Computing Review, Vol 11, Issue 2, 2011.
[5] S.Kchiu, J.C. Yeh, C.H. Huang, and C.W.Wu, "Diagonal Test and Diagnostic Schemes for Flash Memories," In Proceedings of International Test Conference, pp 37-46, 2002.
[6] M. G. Mohammad, K. K. Saluja, and A. Yap, “Testing Flash Memories,” In Proceedings of Thirteenth Int'l Conference on VLSI Design, pp 406-411, 2000.
[7] Gi-Ung Jang, Phil-Joo Hwang, and Hoon Chang, “Fault Test Algorithm for MLC NAND-type Flash Memory” Journal of The Institute of Electronics Engineers of Korea Vol. 49-SD, NO. 4, April 2012.
[8] Stefano Di Carlo, Michele Fabiano, Roberto Piazza, Paolo Prinetto, “Exploring Modeling and Testing of NAND Flash memories”, IEEE 2010.
[9] A. van de Goor. Testing Semiconductor Memories: Theory and Practice. John Wiley & Sons, Inc., 1991.
[10] Advantech, Comparing SLC and MLC Flash Technologies and Structure, September, 2009.
[11] Swati Singh, Chandrawat, “Built-In-Self Test for Embedded Memories by Finite State Machine” International Journal of Digital Application & Contemporary research, Volume2, Issu2, September 2013.
[12] S. Sharma, V. Moyal, “Programmable FSM based MBIST Architecture”, International Journal

of Digital Application & Contemporary research, Volume 1, Issue 7, February 2013.

저 자 소 개

김 진 완(학생회원)
2010년 숭실대학교 전자계산원 학사 졸업.
2012년～현재 숭실대학교 대학원 컴퓨터학과 석사과정

<주관심분야 : 컴퓨터구조, 메모리테스트, VLSI 설계 및 테스트, 임베디드 시스템, 메모리 에러정정>



김 태 환(학생회원)
2010년 숭실대학교 전자계산원 학사 졸업.
2014년 숭실대학교 대학원 컴퓨터학과 석사 졸업

<주관심분야 : 컴퓨터구조, 메모리테스트, VLSI 설계 및 테스트, 임베디드 시스템, 메모리 에러정정>

장 훈(정회원)
1987년 서울대학교 공대 전자공학과 학사 졸업.
1989년 서울대학교 공대 전자공학과 석사 졸업.
1993년 University of Texas at Austin 박사 졸업.
1991년 IBM Inc. Senior Member of Technical Staff.
1993년 Motorola Inc. Senior Member of Technical Staff.
1994년～현재 숭실대학교 컴퓨터학부 교수.
<주관심분야 : 컴퓨터구조, 메모리 테스트, VLSI 설계 및 테스트, 임베디드 시스템>