論文98-35C-8-1

Content Addressable Memory의 이웃패턴감응고장 테스트를 위한 내장된 자체 테스트 기법

(Built-In Self Test for Testing Neighborhood Pattern Sensitive Faults in Content Addressable Memories)

康 容 碩 *, 李 鍾 哲 *, 姜 成 昊 *

(Yong-Seok Kang, Jong-Cheol Lee, and Sungho Kang)

요 약

Content Addressable Memory(CAM)의 이웃패턴감응고장들을 효과적으로 테스트하기 위한 새로운 병렬 테스트 알고리듬과 내장된 자체 테스트(Built-In Self Test) 아키텍처를 개발하였다. 테스트 모드에서 읽기 동작은 CAM 고유의 기능인 검색 기능에 의해 한 번의 동작으로 대체할 수 있으며 쓰기 동작은 약간의 주변회로를 수정으로 병렬로 수행할 수 있도록 하였다. 결과에서 알 수 있듯이 작은 복잡도와 면적 오버헤드로 효과적이고 실용적인 테스팅이 가능하다.

Abstract

A new parallel test algorithm and a Built-in Self Test(BIST) architecture are developed to test various types of functional faults efficiently in Content Addressable Memories(CAMs) . In test mode, the read operation is replaced by one parallel content addressable search operation and the writing operation is performed parallely with small peripheral circuit modifications. The results show that an efficient and practical testing with very low complexity and area overhead can be achieved.

I. 서 론

고집적 회로의 많은 수의 기능적 소자들이 외부에 핀을 갖지 않은 형태의 내장된 환경에서 동작하고 있다. 따라서 내장된 환경에서 발생하는 테스트 상의 어려움을 해결하기 위한 새로운 테스트를 고려한 설계 방법의 개발이 필요하다. 메모리 소자는 고집적 회로의 다양한 응용 분야에서 사용되기 때문에 내장된 RAM(Random Access Memory)를 보다 쉽게 테스트하기 위한 많은 연구[1,2,3,4,5,6,7]가 있었다. 하지만 이러한 여러 시도들은 Content Addressable Memory(CAM)에는 적당한 방법들이 아니다. CAM은 빠른 데이터 검색 기능을 필요로 하는 영역에서 주로 이용되어 왔으며 현재 집적 기술의 급속한 발달에 의해 일반적인 목적의 CAM의 사용빈도도 급격히 증가하고 있다. 최근, 내장된 자체 테스트 기법(Built-In Self Test; BIST)은 내장된 메모리뿐 아니라 개별 소자로 사용되는 메모리의 테스트를 하기 위한 효과적인 방법으로 각광받고 있다. 그러나 CAM을 테스트하기 위한 연구[8,9,10]는 얼마 되지 않고 이를 BIST로 구현한 연구[8]는 더욱 적다. 따라서 내장된 CAM을 위한 BIST 방법의 개발은 CAM의 고속 병렬 검색 기능을 필요로 하는 많은 시스템에 큰 이점을 제공할 수 있다.

* 正會員, 延世大學校 電氣工學科

(Dept. of Electrical Eng. Yonsei University)

※ 본 연구는 한국전자통신연구소의 연구비 지원에 의한 결과임(수탁과제명 : 고집적 메모리를 위한 BIST 자동합성기에 관한 연구)

接受日字:1997年8月29日, 수정완료일:1998年6月30日

RAM의 기능적 테스트를 위해서는 어드레스 디코더, 메모리 셀 어레이, 그리고 읽기/쓰기 회로에 존재하는 다양한 물리적 결함(defect)을 몇 개의 고장 모델로 나타내어야 한다. 다음에서 기술하는 4가지 고장 모델들이 RAM의 품질을 보장하기 위해 사용된다[11]. 이들은 고착고장(Stuck-at fault; SAF), 전이고장(Transition fault; TF), 결합고장(Coupling fault: CF), 그리고 이웃패턴감응고장(Neighborhood pattern sensitive fault; NPSF)[12]이다. NPSF는 동적이웃패턴감응고장(Active NPSF; ANPSF), 수동이웃패턴감응고장(Passive NPSF; PNPSF), 그리고 정적이웃패턴감응고장(Static NPSF; SNPSF)으로 나눌 수 있다.

본 논문에서는 CAM 테스트를 위한 효과적인 새로운 테스트 방법에 관하여 논한다. CAM에서 읽기 동작은 한 번의 검색 동작으로 대체할 수 있으며 병렬 쓰기가 가능하도록 약간의 주변회로를 수정하였다. SAF, TF, 그리고 가장 복잡한 알고리듬과 하드웨어를 필요로 하는 2형(type-2) NPSF를 위한 새로운 테스트 알고리듬과 BIST 아키텍처를 개발하였다.

Ⅱ. CAM 아키텍처

CAM은 자체의 검색 기능을 갖고 있기 때문에 충분한 메모리 공간이 사용될 수만 있으면 데이터 검색은 특별한 검색 알고리듬 없이도 한 번의 검색 동작을 통해 수행할 수 있다. 그림 1에 나타난 기본적인 CAM의 셀은 병렬 검색 동작을 위한 핵심부로 일반적으로 9개의 트랜지스터로 구성되어 있다. SRAM 셀과 비교하여 셀에 저장된 데이터와 검색하려는 데이터를 비교하기 위해 M1m, M2m 그리고 M3m과 같은 부가적인 트랜지스터를 필요로 한다. XOR 회로와 같은 기능을 수행하는 비교회로는 비트선(BIT0과 BIT1)과 메모리 셀을 입력으로 사용하고 매치선(match line) MT를 출력으로 각각 사용한다.

일반적으로 워드선(WL)과 비트선 쌍의 값에 따라 CAM 셀은 4가지의 모드로 동작한다. 읽기와 쓰기 모드는 일반 RAM에서의 동작과 동일하고 이들 모드는 CAM의 특징적인 모드인 검색 모드와 결합하여 검색 후 읽기 모드와 검색후 쓰기 모드를 사용할 수 있다. 쓰기 모드에서 WL은 1의 값을 갖는다. 따라서 트랜지스터 M1a와 M2a는 ON 상태가 되어 쓰기 레지스터에 의해 구동되는 서로 보수값을 갖는 BIT0과 BIT1의 값은 메모리 셀에 저장된다. 읽기 모드에서도 WL은 1의 값을 갖지만 비트선 BIT0과 BIT1은 미리 논리값 1로 충전되어 있다. 셀에 저장된 값에 따라 BIT1 또는 BIT2가 방전하게 된다. 따라서 감지증폭기는 논리값 0 또는 1을 읽을 수 있다. 검색 모드는 표1에 나타내었다. 이 모드에서는 WL은 논리값 0을 갖는다. 검색 레지스터의 데이터는 BIT1과 BIT2선을 구동하고 MT는 미리 논리값 1로 충전시켜둔다. 이들 비트선 쌍의 값은 CAM 셀의 저장된 값과 비교된다. 만약 BIT1과 BIT0의 값이 각각 논리값 1과 0이고 셀의 노드 Z0는 0 그리고 Z1은 1이라고 하면 M2m은 ON 상태가 되고 M1m은 계속 OFF 상태로 있을 것이다. 하지만 BIT0의 값이 0이기 때문에 M3m은 OFF 상태가 되어 MT선을 통한 접지와의 전류 경로가 형성되지 않아 MT는 계속 논리값 1을 유지할 것이다. 이렇게 MT가 1의 값을 갖는 경우가 검색하는 데이터와 저장된 데이터가 일치하는 경우이다. 만약 일치하지 않는 경우에는, 즉 BIT1=0, BIT0=1, Z0=0, 그리고 Z1=1인 경우나 BIT1=1, BIT0=0, Z0=1, 그리고 Z1=0인 경우에는 M3m을 통해 MT와 접지 사이의 전류 경로가 생기게 되어 MT는 논리값 0을 갖게 된다. 워드 본위(word oriented)인 경우에는 같은 워드선을 갖는 셀들이 한 개의 MT에 연결되어 wired-AND와 같이 동작하여 한 비트라도 불일치가 발생하면 MT는 0이된다.

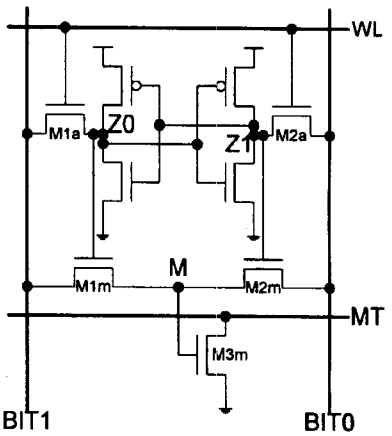


그림 1. 전형적인 9 트랜지스터 CAM 셀
Fig. 1. Typical Nine Transistor CAM Cell.

만약 BIT1과 BIT0의 값을 0으로 하면 검색모드에

서 CAM 셀은 마스킹될 수 있다. 이러한 경우 M1m 또는 M2m이 ON 상태가 되더라도 M3m이 절대로 ON 상태가 될 수 없게 되므로 저장된 값과 상관없이 항상 MT는 1이 되어 일치한 것과 같은 결과를 출력한다.

표 1. CAM 셀의 검색 동작
Table 1. CAM Cell Search Operation.

비교 결과	BIT1	BIT0	Z0	Z1	M1m	M2m	M3m	MT
일치	1	0	0	1	OFF	ON	OFF	1
	0	1	1	0	ON	OFF	OFF	
불일치	1	0	1	0	ON	OFF	ON	0
	0	1	0	1	OFF	ON	ON	
항상 일치	0	0	1	0	ON	OFF	OFF	1
			0	1	OFF	ON	OFF	
항상 불일치	1	1	1	0	ON	OFF	ON	0
			0	1	OFF	ON	ON	

Ⅲ. 테스트 알고리듬

본 논문에서 제안하는 알고리듬은 고착고장, 전이고장, 그리고 이웃패턴감응고장 모델을 고려할 수 있다. 이들 고장 모델들은 다음과 같이 정의할 수 있다. 고착고장은 셀이나 선이 항상 논리값 0이나 1에 고착된 고장으로 반대의 상태로 변화가 불가능한 고장이다. 고착고장의 특수한 경우로 전이고장이 있다. 한 셀이나 선이 0에서 1 또는 1에서 0으로의 전이가 불가능한 고장을 말한다. 패턴감응고장은 다음과 같이 정의된다. 한 셀의 값이나 값의 전이가 다른 모든 셀의 값에 따라 영향을 받는 것을 말한다. 실제 테스트에서 모든 가능한 메모리 셀의 패턴을 고려하는 것은 불가능하다. 따라서 패턴감응고장을 축소한 이웃패턴감응고장 모델을 사용한다. 이웃셀은 특정고장 모델에 관련된 셀의 수를 말하는 것이고 기준셀은 테스트 대상이 되는 셀을 말한다. 기준셀을 제외한 이웃셀을 제외 이웃셀(deleted neighborhood)라고 한다. 제외 이웃셀로 4개를 고려하는 고장 모델을 1형(type-1)이라고 하고 8개의 이웃셀을 고려하는 것을 2형 이웃패턴감응고장 모델이라고 한다. 이웃패턴감응고장은 3가지의 경우로 나눌 수 있는데 제외 이웃셀에서의 변화로 인해 기준셀의 내용이 변하는 고장을 동적이웃패턴감응고장이라고 한다. 수동이웃패턴감응고장을 갖는 기준셀은 제외 이웃셀의 패턴에 의해 전이가 불가능하다. 마지막으로 정적이웃패턴감응고장은 기준셀의 값이 특정한 제외 이웃셀의 패턴에 의해 어떤 한 상태로 강제되는 고장을 말한다.

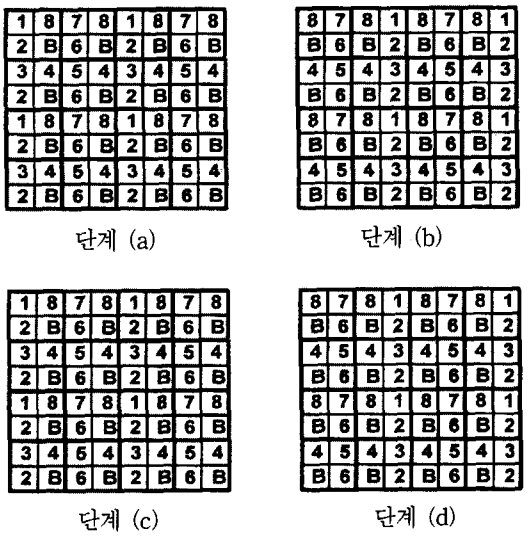


그림 2. 셀 할당
Fig. 2. Cell Assignments.

고착고장을 검출하기 위해서는 각각 셀에서 0과 1의 값을 읽어야한다. 전이고장을 검출하기 위해서는 각 셀은 반드시 0에서 1, 그리고 1에서 0의 전이를 거친 후에 다음 전이 동작을 수행하기 전에 읽는 동작을 수행하여 값을 확인하여야 한다. 이러한 고착고장과 전이고장을 검출하기 위한 조건들은 제안된 알고리듬을 사용하여 이웃패턴감응고장을 테스트하는 과정을 통하여 만족된다. 이웃패턴감응고장은 특정한 이웃셀의 패턴이나 이웃셀중의 한 셀에서의 전이동작에 의해 발생하게 되므로 셀 할당이 고장 검출에서 중요한 역할을 한다. 병렬 테스트를 위한 셀 할당 방식을 그림 2에 나타내었다. 같은 숫자를 갖는 셀들은 다음 장에서 제안하는 아키텍처를 통해 동시에 쓸 수 있고 기준셀 주위의 8개의 제외 이웃셀을 고려하는 2형 이웃패턴감응고장 테스트를 위한 모든 가능한 패턴을 쓸 수 있다.

모든 2형 정적이웃패턴감응고장을 테스트하기 위해서는 각 기준셀은 반드시 제외 이웃셀의 모든 패턴에서 0과 1의 상태에서 읽혀야 한다. 동적이웃패턴감응고장은 제외된 이웃셀의 모든 가능한 변화를 거치면서 0과 1의 상태에서 읽혀지면 검출할 수 있다. 수동패턴

감응고장을 테스트하기 위해서는 각 기준셀을 제외 이웃셀의 모든 패턴에서 0과 1의 상태에서 쓰기와 읽기를 수행하여야 한다. 이러한 2형 이웃패턴감응고장의 3가지 모델을 테스트할 수 있는 모든 조건을 만족하는 알고리듬을 그림 3에 나타내었다. 그림 3의 알고리듬에서 사용하는 오일러리안(eulerian) 시퀀스는 8개의 제외 이웃셀에서 발생할 수 있는 모든 변화를 갖는 시퀀스이다. 그림 2의 각 단계에서 이 알고리듬에 따른 테스트 시퀀스를 가하게 된다. 그림 3에 나타낸 테스트 시퀀스의 길이보다 짧은 알고리듬을 사용할 수 있으나 하드웨어 오버헤드가 크게 된다. 따라서 적은 하드웨어 오버헤드로 테스트 시퀀스를 생성하기 위해서 알고리듬을 대칭되게(symmetrized) 하였다.

```
초기화;                    // 모든 셀에 0의 값을 쓴다. //
loop
        기준셀에 1을 쓴다;
        기준셀을 읽는다;
        제외 이웃셀에 8비트 오일러리안 시퀀스를 가한다;
        기준셀을 읽는다.
end-loop
초기화;                    //모든 셀에 1의 값을 쓴다. //
loop
        기준셀에 0을 쓴다;
        기준셀을 읽는다;
        제외 이웃셀에 8비트 오일러리안 시퀀스를 가한다;
        기준셀을 읽는다.
end-loop
```

그림 3. 이웃패턴감응고장 테스트를 위한 알고리듬
Fig. 3. Test Algorithm for NPSFs.

Ⅳ. 내장된 자체 테스트 아키텍쳐

병렬 쓰기를 위해서는 일반적으로 사용하는 어드레스 디코더와 매치(match) 레지스터 비교기를 수정하여야 한다. CAM에선 병렬 검색을 수행할 수 있고 이를 테스트 모드에서는 읽기 동작으로 대체할 수 있으나 쓰기를 병렬로 할 수는 없다. 어드레스 디코더를 수정하면 쓰기도 병렬로 동작할 수 있게 된다. 수정된 어드레스 디코더는 일반 어드레스 입력뿐 아니라 별도의 마스크(mask) 어드레스를 입력으로 사용한다. 각 어드레스 워드(word)에 대하여 그에 대응하는 같은 길이를 갖는 마스크 어드레스가 사용된다. 마스크 어드레스에서 1의 값을 갖는 비트는 그에 대응하는 일반 어드레스의 비트를 마스크하게 된다. 따라서 마스크된 비트의 값이 0일 때와 1일 때 접근할 수 있는 셀들을 동시에 접근할 수 있다. 구현된 마스크 디코더 회로를 그림 4에 나타내었다. 그림 4에서 일반 어드레스는 A_n으로 나타내었고 O_m과 O_{m+1}는 어드레스선과 반전된 어드레스선을 입력으로 사용하는 기존의 디코더의 입력으로 사용된다. 마스크 어드레스선($\overline{M_n}$)은 그 값에 따라 병렬 쓰기가 가능하도록 한다. 만약 $\overline{M_n}$의 값이 0이면 O_m과 O_{m+1}는 모두 1의 값을 갖는다. 따라서 어드레스 디코더는 일반 동작 모드에서 A_n의 값이 1 또는 0일 때 디코딩되는 어드레스를 모두 출력할 수 있게 된다.

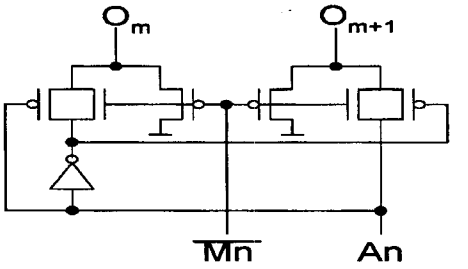


그림 4. 수정된 마스크 디코더
Fig. 4. Modified Masking Decoder.

수정된 매치 레지스터는 모든 셀들이 동일한 값을 갖고 있는 가를 결정하기 위해 사용된다. CAM에서 병렬로 검색된 모든 결과는 매치 레지스터에 저장되고 일치된 워드의 위치를 가리킨다. 테스트 모드에서 검색되어야 할 기준셀들은 짝수나 홀수 어드레스에 위치하고 워드안에서도 짝수나 홀수 비트에 위치한다. 따라서 매치 레지스터의 짝수 위치와 홀수 위치를 구별하기 위한 하드웨어가 필요하고 검색하는 과정에서도 검색 레지스터를 마스킹하여 기준셀만의 검색이 가능하도록 한다. 만약 매치 레지스터의 값의 결과가 모두 1의 값이 나오지 않으면 이는 CAM에 고장이 존재함을 의미한다. 따라서 그림 2의 각 단계에 따라 짝수나 홀수 어드레스에 모두 1의 값을 갖고 있는가를 확인하기 위한 그림 5와 같은 매치 레지스터 비교기가 필요하다. 그림 5에서 고장 표시기(error flag)는 비교 동작을 수행하기 전에 미리 논리값 1로 충전되고 상태 표시기(state indicator)에 의해 홀수 또는 짝수선을 선택하게 된다. 고장이 없는 상태에선 이 고장 표시기는 계속 1의 값을 유지하고 있게 된다. 만약 이 표시기가 0으로 방전되면 CAM에 고장이 있는 것으로 판

명된다. 이러한 형태의 비교기를 사용하면 하나의 고장뿐 아니라 여러 개의 고장이 발생하는 경우에도 검출이 가능하다. 이웃패턴감응고장을 테스트하기 위한 제안된 알고리듬의 일반 어드레스와 마스크 어드레스 시퀀스를 표 2에 나타내었다. 그림 3에서 기준셀의 읽기 동작은 검색 동작으로 대체되기 때문에 읽기 동작을 위한 어드레스는 필요하지 않다.

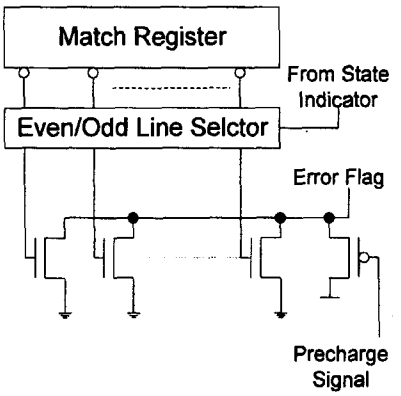


그림 5. 매치 레지스터 비교기
Fig. 5. Match Register Comparator.

고착고장과 전이고장뿐 아니라 이웃패턴감응고장을 테스트하기 위한 일반 어드레스 생성기와 마스크 어드레스 생성기는 두 개의 9비트 쉬프트(shift) 레지스터와 한 개의 3비트 계수기로 구현된다. 알고리듬에서 마스크 어드레스중 Mx1과 My1보다 상위의 비트는 모두 1의 값을 갖게 되기 때문에 일반 어드레스중 Ax1과 Ay1보다 상위 비트는 고려할 필요가 없다. 표 2에서 나타낸 것과 같이 Ax0와 Mx1은 같은 시퀀스를 갖고 이들 시퀀스는 9비트 쉬프트 레지스터와 인버터를 사용하여 그림 2의 4개의 단계에 따라 생성할 수 있다. Ay0와 My1도 같은 방식으로 생성될 수 있다. Ax1과 Ay1을 위한 시퀀스는 3비트 계수기를 이용하여 생성할 수 있다. 비록 계수기의 총 상태가 8가지뿐이지만 리셋 신호를 사용하여 하나의 상태를 추가하여 9비트 쉬프트 레지스터와 동기하여 사용할 수 있다. 표 2의 시퀀스는 그림 2의 (a) 단계를 위한 시퀀스이기 때문에 (a) 단계를 위해 생성된 시퀀스와 그의 반전된 시퀀스를 조합하여 다른 3개의 단계를 위한 시퀀스도 생성한다. 어드레스 생성을 위해 구현된 회로는 그림 6에 나타내었다. 전체 내장된 자체 테스트를 위한 하드웨어는 w개의 워드선을 갖는 CAM에서 $6\log_2 w+3w+C$개의 트랜지스터를 필요로 한다. 여기서 C는 CAM의 크기에 비례하지 않는 고정된 비트의 레지스터나 계수기 그리고 클럭 생성기 등을 위해 필요한 트랜지스터의 수를 표시한다.

표 2. 일반 어드레스 시퀀스와 마스크 어드레스 시퀀스
Table 2. Address Sequence and Mask Address Sequence.

상 태	숫 자	A_{X1}	A_{X0}	A_{Y1}	A_{Y0}	M_{X1}	M_{X0}	M_{Y1}	M_{Y0}
(a)	B	X	1	X	1	1	0	1	0
	2	0	0	X	1	0	0	1	0
	6	1	0	X	1	0	0	1	0
	4	X	1	1	0	1	0	0	0
	8	X	1	0	0	1	0	0	0
	3	0	0	1	0	0	0	0	0
	7	1	0	0	0	0	0	0	0
	5	1	0	1	0	0	0	0	0
	1	0	0	0	0	0	0	0	0

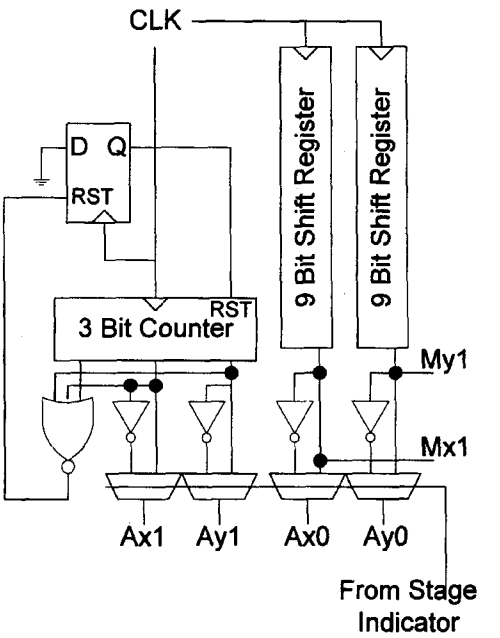


그림 6. 일반 어드레스 및 마스크 어드레스 생성기
Fig. 6. Address and Mask Address Generators.

V. 결 과

그림 5의 매치 레지스터 비교기를 SPICE3을 이용해 시뮬레이션을 수행한 것을 그림 7에 나타내었다. 그림에서 0ns에서 5V의 값을 갖는 신호는 반전된 precharge 신호이다. prechare가 끝나고 매치 레지스터부터 읽혀진 값이 비교기로 입력되었을 때 고장이 없는 경우에는 Error Flag 신호선의 값이 5V로 상승

하는 것을 알 수 있다(a). 그러나 고장이 발생하여 비교기로 입력된 값이 모두 동일하지 않을 경우에는 그림의 (b)와 같이 Error Flag의 값이 1.5V 수준으로 하강하게 됨을 알 수 있다. 실험에서 볼 수 있듯이 비교기로 입력되는 신호중 고장인 셀의 수가 많을수록 더욱 깨끗하고 분명한 신호를 얻을 수 있었으나 매치 레지스터 출력중 한 개의 값이 0인 경우에도 충분히 그 결과를 알 수 있는 신호가 발생되었다. 따라서 개발된 비교기를 사용하면 단일 고장뿐 아니라 다중 고장도 검출이 가능함을 알 수 있다. 또한 보다 효율적으로 precharge 하기 위해 precharge 신호가 들어오는 곳의 PMOS의 채널은 그 폭을 다른 트랜지스터의 채널보다 더 넓게 설정해 주었다. 상호 비교기의 precharge 입력은 쓰기 동작을 나타내는 신호를 이용하여 precharge 시켜주고 검색 신호가 발생하면 precharge를 멈추고 매치 레지스터의 값들을 비교하면 된다. 또한 그림 5의 선택기의 제어 신호도 어드레스 및 데이터 생성기에 있는 기준셀의 데이터를 저장하고 있는 플립플롭의 값에 의해 발생될 수 있다. 따라서 본 비교기를 위해서는 별도의 제어신호를 생성시켜줄 필요가 없고 어드레스 및 데이터 생성기에서 발생된 신호를 그대로 이용할 수 있으며 또한 게이트로 설계했을 때보다 많이 오버헤드를 줄일 수 있는 장점이 있다.

그림 8은 그림 4에서 보여주는 마스크 어드레스 디코더를 SPICE3을 사용하여 검증한 결과이다. 그림 8의 (a)는 마스크 어드레스 디코더의 입력을 보여주고 있다. 20ns에서 30ns 까지의 부분에서 1이된 신호가 마스크 어드레스이며 30ns에서 1로 전이되는 신호가 일반 어드레스의 값이다. 이와 같은 입력이 주어졌을 때 결과는 그림 8의 (b)와 같이 나타났다. 그림 (b)의 0ns에서 1인 신호는 Om 신호이다. 0ns에서 20ns사이에서 마스크 어드레스는 0을 갖고 보통 어드레스도 0을 갖는다. 이때는 마스크 어드레스 디코더는 보통의 디코더와 같이 동작하게 되고 수정된 디코더 두 개의 출력중 하나만 1의 값을 갖게 된다. 일반 어드레스의 입력값이 0이므로 이때에는 수정된 회로의 Om 신호가 1이된다. 20ns에서 30ns에서는 마스크 어드레스가 1이되도록 하였다. 이때에는 BIST 회로가 테스트 모드에 있는 경우로 마스크 어드레스 디코더의 수정부의 출력이 모두 1이 되어야 한다. 그림 8의 (b)로부터 설계된 회로가 정상적으로 동작하고 있음을 알 수 있다. 30ns 이후에는 다시 마스크 어드레스가 0이 되고 보통 어드레스는 1을 유지하고 있다. 이 경우에는 수정된 회로의 Om+1만이 1이됨을 실험 결과로부터 알 수 있다.

그림 6의 회로를 이용하여 표 2의 어드레스 시퀀스를 생성하는 회로 합성 및 시뮬레이션 결과를 그림 9에 나타내었다. Synopsys의 VHDL 시뮬레이터와 합성기를 이용하였다. 그림 9의 오른쪽 두 개의 세로선 사이에 그림 2의 상태 (a)에서 사용하는 어드레스 시퀀스중 한 개를 나타낸다.

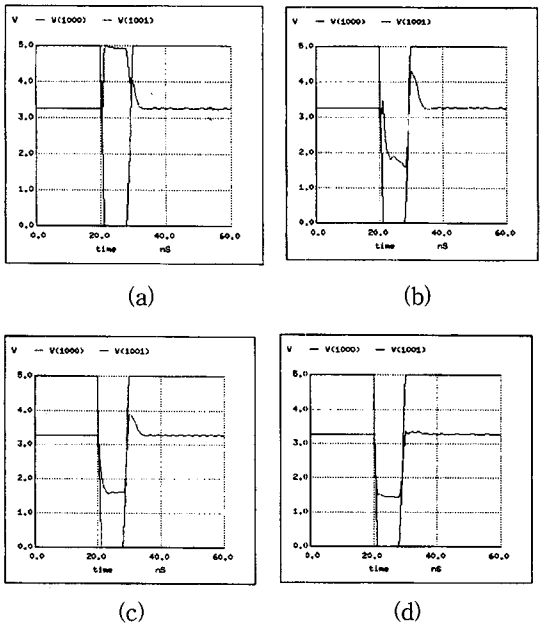
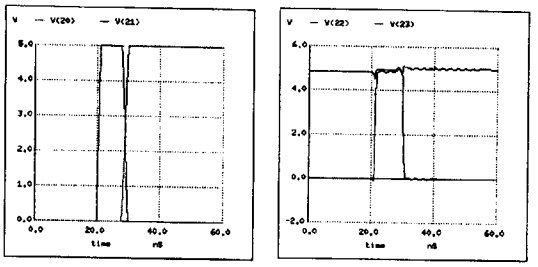


그림 7. 매치 레지스터 비교기의 시뮬레이션 결과
(a) 고장이 없는 경우 (b) 하나의 입력이 0인 경우 (c) 세 개의 입력이 0인 경우 (d) 아홉 개의 입력이 0인 경우

Fig. 7. Simulation Results of Match Register Comparator.
(a) Fault free (b) One bit in logic 0 (c) Three bits in logic 0 (d)Nine bits in logic 0

그림 10에 전체적인 시뮬레이션을 위한 블록 다이어그램을 나타내었다. 그림 11에는 전체 시뮬레이션 결과를 나타내었다. CAM에 고장을 삽입한 후 본 논문에서 제시한 알고리듬을 사용하여 테스트 수행하여 고장이 발생한 경우 TEST_FAIL 신호가 1이 됨을 알 수 있다. CAM의 셀과 그 밖의 주변회로는 테스트 시뮬레이션을 위해 VHDL를 사용한 기능적 모델을 이용하였다.



(a) (b)

그림 8. 마스크 어드레스 디코더의 시뮬레이션 결과
(a) 마스크 어드레스 디코더의 입력
(b) 수정된 디코더의 출력

Fig. 8. Simulation Results for Mask Address Decoder.
(a) Input for Mask Address decoder
(b) Output of Modified decoder

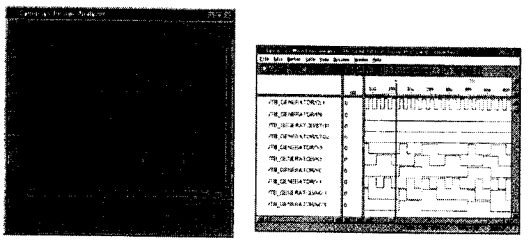


그림 9. 일반 어드레스 및 마스크 어드레스 생성기의 합성 및 시뮬레이션 결과

Fig. 9. Synthesis and Simulation Results of Address Generator.

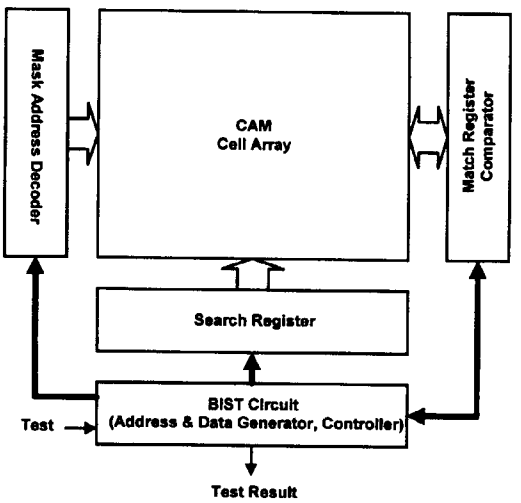


그림 10. 시뮬레이션을 위한 블록 다이어그램

Fig. 10. Block Diagram for Test Simulation.

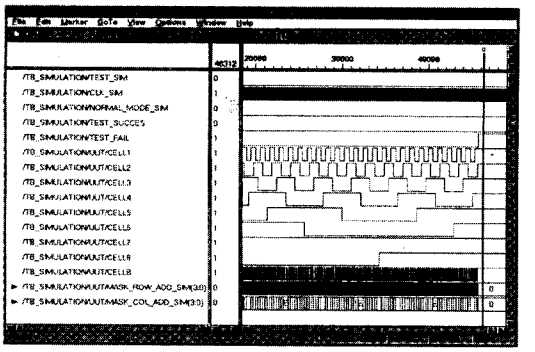


그림 11. 테스트 시뮬레이션 결과

Fig. 11. Test Simulation Result.

그림 3에서 8비트 오일러리안 시퀀스는 $(8\times2^8+1)$개의 패턴을 갖는다. 따라서 그림 3에 나타낸 알고리듬을 위한 테스트 길이는 $2\times4\times(8\times2^8+1)$가 된다. 고착고장, 전이고장, 그리고 이웃패턴감응고장을 테스트하기 위한 알고리듬은 그림 2에 나타낸 4개의 다른 단계에 각각 가해야 하기 때문에 전체 테스트 시퀀스의 길이는 4×16392로 CAM의 크기와 상관없게 된다. 이웃패턴감응고장을 테스트하기 위한 이전의 연구들과의 비교를 표3에 나타내었다. 표 3에서 비교된 연구[3]은 본 논문에서 접근한 방식과 다른 방법으로 2형 이웃패턴감응고장을 테스트하기 위한 제외 이웃셀에 패턴을 가하였다. 우선 본 논문에서는 제외 이웃셀에서 가능한 모든 패턴을 고려하였으나 [3]에서는 8개의 제외 이웃셀들을 상하, 좌우, 대각선에 위치하는 셀들로 구분하여 같은 값을 할당함으로써 결국 3비트의 오일러리안 시퀀스를 가하게 되어 전체 가능한 제외 이웃셀의 패턴을 전부 고려하지 않았다. 이에 따라 3가지 종류의 이웃패턴감응고장을 테스트하기 위한 조건을 만족시키지 못했다. 그럼에도 불구하고 표 3에서 알 수 있는 바와 같이 본 논문과의 동작수의 비교에서 많은 차이를 갖고 본 논문이 훨씬 효과적임을 알 수 있다. 하드웨어 오버헤드 면에서 비교하면 본 논문의 병렬 쓰기와 매치 레지스터 비교기를 위해 총 $6\log_2 w+3w+1$개의 트랜지스터와 다른 내장된 자체 테스트 회로의 구성을 위해 CAM의 크기와 상관없이 약 1000여개의 트랜지스터를 필요로 한다. 1K×8의 CAM인 경우 본 논문의 경우 약 3100개의 트랜지스터를 필요로 하고 이러한 면적 오버헤드는 약 1% 정도이다. 이는 이전의 연구[8]에 비해 약간 많은 수준이나 이전의 연구[8]가 CAM에서 사용하는 셀 어레

이도 다시 설계하여야 하는 어려움을 갖는 것에 비해 외부 회로의 수정만으로 가능한 것이기에 훨씬 실용성이 높다. 따라서 결과적으로 새로 제안된 접근 방법을 사용하면 경제적인 하드웨어 오버헤드로 이웃패턴감응고장 및 고착고장, 전이고장을 위한 테스트 시간을 상당히 줄일 수 있다.

표 3. 이전 연구와의 비교
Table 3. Comparison with the Previous Approach.

크기	동작수	
	본 논문	Mazumder [8]
8K × 8	65.6K	270K
16K × 8	65.6K	540K
64K × 8	65.6K	2.2M
256K × 8	65.6K	8.7M

Ⅵ. 결 론

CAM은 내장된 환경에서 고속 데이터 검색을 위해 다양한 응용 분야에서 사용되고 있다. 따라서 CAM을 위해 새로운 내장된 자체테스트 기법의 개발은 테스트 패턴의 생성과 시뮬레이션을 위한 새로운 대안으로써 테스트에 소용되는 시간과 복잡도를 줄일 수 있을 것이다. 제안된 내장된 자체 테스트 회로를 사용한 새로운 병렬 테스트 알고리듬은 적은 하드웨어 오버헤드로 테스트 시간을 대폭 줄일 수 있기 때문에 다양한 CAM 응용 분야에서 효과적이고 실제적인 테스트 방법이 될 것이다.

참 고 문 헌

[1] M. Franklin and K. Saluja, "Embedded RAM Testing," in IEEE Int. Workshop on Memory Technology, Design and Testing, 1995, pp. 29-33.

[2] S. Jain and C. Stroud, "Built-in Self Testing of Embedded Memories," in IEEE Design and Test of Computers, October 1986, pp. 27-37.

[3] M. Sachdev and M. Verstraelen, "Development of a Fault Model and Test Algorithms for Embbeded DRAMs," in Proc. of IEEE International Test Conference, 1993, pp. 815-824.

[4] A. J. Goor, "Using March Tests to Test SRAMs," in IEEE Design and Test of Computers, March 1993, pp. 8-14.

[5] M. Nicolaidis, "An Efficient Built-In Self Test for Functional Test of Embedded RAMs," in Proc. of IEEE 15th International Synposium on Fault-Tolerant Computer, June 1985, pp. 118-123.

[6] R. Dekker, F. Beenker and L. Thijssen, "A Realistic Self-Test Machine for Static Random Access Memories," in Proc. of IEEE International Test Conference, 1988, pp. 353-361.

[7] K. Kinoshita and K. Saluja, "Built-In Testing of Memory Using On-chip Compact Testing Scheme," in Proc. of IEEE International Test conference, 1984, pp. 271-281.

[8] P. Mazumder, J. Patel and W. Fuchs, "Methodology for Testing Embedded Content Addressable Memories," in IEEE Trans. on CAD, January 1988, pp. 11-20.

[9] G. M. Blair, "A Content Addressable Memory with a Fault-Tolerance Mechanism," in IEEE Journal of Solid-State Circuits, 1987, pp. 614-616.

[10] G. Giles and C. Hunter, "A Methodology for Testing Content Addressable Memories," in Proc. of IEEE International Test Conference, 1985, pp. 471-474.

[11] S. Thatte and J. Abraham, "Testing of Semiconductor Random Access Memories," in Proc of Internatinal Conference on Fault-Tolerant Computers, 1977, pp. 81-87.

[12] A. J. Goor, Testing Semiconductor Memories; Theory and Practice. Singapore: John Wiley and Sons, 1995.

저 자 소 개

康 容 碩(正會員) 第 33卷 A編 第 9號 參照
1995년 2월 연세대학교 전기공학과 졸업(공학사). 1997년 8월 연세대학교 전기공학과 졸업(공학석사). 현재 연세대학교 전기공학과 박사과정. 주관심분야는 테스팅, DFT와 VLSI & CAD 등임

姜 成 昊(正會員) 第 33卷 A編 第 9號 參照
1986년 2월 서울대학교 제어계측공학과 졸업(공학사). 1998년 5월 The Univ. of Texas at Austin Electrical and Computer Eng. 졸업(공학석사). 1992년 5월 The Univ. of Texas at Austin Electrical and Computer Eng. 졸업(공학박사). 1989년 11월 ~ 1992년 8월 미국 Schlumberger Inc. Research Scientist. 1992년 9월 ~ 1992년 10월 미국 The Univ. of Texas at Austin, Post Doctoral Fellow. 1992년 8월 ~ 1994년 6월 미국 Motorola Inc., Senior Staff Engineer. 1994년 9월 ~ 현재 연세대학교 기계전자공학부 조교수. 주관심분야는 테스팅, DFT와 VLSI & CAD, Design Verification 등임

李 鍾 哲(正會員)
1996년 2월 연세대학교 전기공학과 졸업(공학사). 1998년 2월 현세대학교 전기공학과 졸업(공학석사). 현재 삼성전자 SRAM 사업부. 주관심분야는 테스팅, DFT와 VLSI & CAD 등임