논문 2007-44SD-2-10

이중 포트 메모리의 실제적인 고장을 고려한 효율적인 테스트 알고리즘

(An Efficiency Testing Algorithm for Realistic Faults in Dual-Port Memories)

박 영 규*, 양 명 훈*, 김 용 준*, 이 대 열*, 강 성 호**

(Youngkyu Park, Myung-Hoon Yang, Yongjoon Kim, Daeyeal Lee, and Sungho Kang)

요 약

메모리 설계 기술과 공정기술의 발달은 고집적 메모리의 생산을 가능하게 하였다. 그러나 이는 메모리의 복잡도를 증가시켜 메모리 테스트를 더욱 복잡하게 하여, 결과적으로 메모리 테스트 비용의 증가를 가져왔다. 효과적인 메모리 테스트 알고리즘은 짧은 테스트 시간동안 다양한 종류의 고장을 검출하여야 하며, 특히 이중 포트 메모리 테스트 알고리즘의 경우에는 단일 포트 메모리의 고장과 이중 포트 메모리 고장을 모두 검출할 수 있어야 한다. 본 논문에서 제안하는 March A2PF 알고리즘은 18N의 짧은 테스트 패턴을 통해 이중 포트 및 단일 포트 메모리와 관련된 모든 종류의 고장을 검출하는 효과적인 테스트 알고리즘이다.

Abstract

The development of memory design and process technology enabled the production of high density memory. However, this increased the complexity of the memory making memory testing more complicated, and as a result, it brought about an increase in memory testing costs. Effective memory test algorithm must detect various types of defects within a short testing time, and especially in the case of port memory test algorithm, it must be able to detect single port memory defects, and all the defects in the dual port memory. The March A2PF algorithm proposed in this paper is an effective test algorithm that detects all types of defects relating to the duel port and single port memory through the short 18N test pattern.

Keywords: Dual Port Memory, Test Algorithm, Fault Model

I. 서 론

메모리 설계 기술과 미세 공정 기술의 발달로 회로의 복잡도가 증가하고 회로의 집적도가 크게 향상되었다. 메모리의 고집적화와 동작속도의 증가는 메모리의 테스트를 점점 어렵게 하고 테스트 비용을 증가시킨다[1]. 효과적인 테스트를 위해 변경된 마치 테스트를 고려한 BIST 회로를 이용한 테스트 방법이 제안되었다[2][3]. 높은 신뢰도를 유지하면서 빠르고, 보다 효과적인 메모리 테스트를 위해서는 효율적인 테스트를 위한 설계와 현실적인 고장모델과 테스트 알고리즘의 복잡도를 줄이는 것이 중요하다.

이중 포트 메모리는 동시에 두 개의 동작(읽기 또는/그리고 쓰기)을 수행하는 두 개의 포트로부터 동시에 데이터의 입출력이 가능한 특징을 가지고 있다. 이런 이중 포트 메모리는 많은 양의 데이터를 빠른 시간에 처리할 수 있는 장점을 가진다. 이중 포트 메모리의 경우 단일 포트 메모리의 고장 테스트와 두 포트를 통해 동시 접근을 통해 발생하는 고장 테스트를 할 수 있는 특별한 테스트가 필요하다. 이중 포트 메모리 테스트는 단일 포트 메모리에 대한 고장 모델과 이중 포트 메모

* 학생회원, ** 평생회원, 연세대학교 전기전자공학과
(Department of Electrical and Electronic Engineering, Yonsei University)
접수일자: 2006년7월3일, 수정완료일자: 2007년1월25일

리에서만 발생되는 고장 모델까지 모두 고려한 효율적인 테스트 알고리즘을 사용하여 테스트 시간을 줄이고 고장 검출률을 높여 신뢰도를 높이는 것이 중요하다. 테스트 시간과 고장 검출률은 알고리즘의 복잡도와 알고리즘이 고려하는 고장 모델에 의해 결정된다.

본 논문에서는 이중 포트 메모리에서 발생할 수 있는 모든 결함에 대하여 IFA(Inductive Fault Analysis)를 통하여 다양한 고장모델을 제시한 [4]의 고장모델을 바탕으로 이중 포트 메모리 관련 고장모델을 검출하는 새로운 테스트 알고리즘을 제안한다. 기존 알고리즘은 이중 포트 메모리에서 발생하는 모든 고장모델을 각 고장모델마다 독립적인 알고리즘을 사용하여 검출하지만[8~10], 제안하는 알고리즘은 하나의 알고리즘을 사용하여 이중 포트 메모리에서 발생하는 모든 고장모델을 검출할 수 있는 효율적인 알고리즘이다.

Ⅱ. 이중 포트 메모리의 고장모델

이중 포트 메모리 테스트는 고장모델에 따라서 가해지는 테스트 패턴이 달라지기 때문에 효과적인 테스트 알고리즘을 만들기 위해서는 적절한 고장모델을 결정하는 것이 중요하다. 이중 포트 메모리 테스트에서 사용하는 고장모델은 단일 포트 관련 고장모델과 이중 포트 관련 고장모델로 나누어진다. 단일 포트 관련 고장모델은 기존에 많은 연구를 통해 고장모델이 정립되어졌고, 이중 포트 관련 고장모델은 단일 포트 관련 고장모델처럼 아직 정립이 되지 못하였다. 본 논문에서 사용하는 이중 포트 관련 고장모델은 [4]에서 제안한 고장모델을 사용한다. [4]에서는 메모리 셀 안에서 발생할 수 있는 개방(open), 단락(short), 연결(bridge) 등의 모든 결함(spot defect)을 바탕으로 시뮬레이션을 통해 Fault Primitives(FPs)를 정하고, 다시 이들을 분류하여 고장모델을 결정하였다.

이중 포트 메모리의 기능적 고장모델은 단일 포트에 관련된 고장(1PFs : single port faults)과 두 개의 포트에 관련된 고장(2PFs : two port faults)으로 나누어진다. 1PFs는 단일 셀에 관련된 고장 1PF1과 두 개의 셀에 관련된 고장 1PF2로 나누어지며, 2PFs도 단일 셀에 관련된 고장 2PF1과 두 개의 셀에 관련된 고장 2PF2로 나누어진다.

1. 단일 포트 관련 고장 모델

단일 포트 관련 고장(1PFs)은 크게 하나의 셀과 관련된 고장(1PF1)과 두 개의 셀과 관련된 고장(1PF2)으로 나눌 수 있다[5~7]. 하나의 셀과 관련된 고장은 셀 내부의 노드들 간의 잘못된 연결이나 끊어짐 등으로 인해 생기는 고장이다. 두 개의 셀과 관련된 고장은 두 개의 셀 내부의 노드들 간의 잘못된 연결이나 간섭에 의해 발생하게 되는 고장으로 고장을 유발시키는 결합셀과 고장이 유발되는 피결합셀이 존재하게 된다.

가. 1PF1 관련 고장 모델

단일 포트 관련 고장(1PFs) 중 하나의 셀과 관련된 고장(1PF1)의 종류는 표 1과 같다. 표 1은 1PF1 관련 고장모델과 각 고장모델의 FPs를 보여주며, FPs가 나타내는 의미는 다음과 같다. <S/F/R>에서 S는 고장을 활성화시키는 값 또는 동작을 나타내고, F는 고장 셀의 값을 나타낸다. R은 만약 FPs에 읽기 동작이 있다면 출력에 나타나는 논리 값을 나타낸다. 각 고장모델을 간단히 살펴보면 다음과 같다.

1) Stuck-at Fault(SAF) : 메모리 셀이나 신호선의 값이 항상 논리적으로 0 또는 1로 고정되는 고장이다.
2) Transition Fault(TF)
3) Read Disturb Fault(RDF)
4) Deceptive Read Disturb Fault(DRDF)
5) Incorrect Read Fault(IRF)
6) Random Read Fault(RRF)

표 1. 1PF1 고장모델 x∈{0,1}
Table 1. Fault models of 1PF1. x∈{0,1}.

고장모델		Fault Primitives
1PF1	SAF	<∀/0/->,<∀/1/->
	TF	<w↑/0/->,<w↓/1/->
	RDF	<r0/↑/1>,<r1/↓/0>
	DRDF	<r0/↑/1>,<r1/↓/0>
	IRF	<r0/0/1>,<r1/1/0>
	RRF	<r0/0/?>,<r1/1/?>

나. 1PF2 관련 고장 모델

단일 포트 관련 고장(1PFs) 중 두 개의 셀과 관련된 고장(1PF2)의 종류는 표 2와 같다. 1PF2는 두 셀간의 결합 고장(Coupling Fault)을 말하며, 결합셀이 피결합셀에 고장을 유발시키는 경우이다. 표 2는 1PF2 관련 고장 모델들과 각 고장 모델의 FPs를 보여주며, FPs가 나타내는 의미는 다음과 같다. <Sa;Sv/F/R>에서 Sa는 결합셀의 상태 또는 활성화시키는 동작 나타내고, Sv는 피결합셀의 상태 또는 활성화시키는 동작을 나타낸다. 각 고장모델을 간단히 살펴보면 다음과 같다.

표 2. 1PF2의 고장모델 x∈{0,1}
Table 2. Fault models of 1PF2. x∈{0,1}.

고장모델		Fault Primitives
1PF2	CFds	<wx;1/↓/->,<rx;0/↑/->,<rx;1/↓/->,<wx;0/↑/->
	CFst	<1;1/0/->,<1;0/1/->,<0;1/0/->,<0;0/1/->
	CFir	<0;r0/0/1>,<0;r1/1/0>,<1;r0/0/1>,<1;r1/1/0>
	CFrr	<0;r0/0/?>,<0;r1/1/?>,<1;r0/0/?>,<1;r1/1/?>
	CFdr	<0;r0/↑/0>,<0;r1/↓/1>,<1;r0/↑/0>,<1;r1/↓/1>
	CFrd	<0;r0/↑/1>,<0;r1/↓/0>,<1;r0/↑/1>,<1;r1/↓/0>
	CFtr	<0;w↓/1/->,<0;w↑/0/->,<1;w↓/1/->,<1;w↑/0/->

1) State Coupling Fault(CFst) : 피결합셀에 어떤 동작의 적용 없이 결합셀이 특정한 값을 갖고 있다면, 피결합셀이 강제적인 특정한 논리 값(0 또는 1)으로 변하는 고장이다.
2) Disturb Coupling Fault(CFds)
3) Incorrect Read Coupling Fault(CFir)
4) Random Read Coupling Fault(CFrr)
5) Deceptive Read Destructive Coupling Fault(CFdr)
6) Read Destructive Coupling Fault(CFrd)
7) Transition Coupling Fault(CFtr)

2. 이중 포트 관련 고장 모델

이중 포트 관련 고장 모델은 두 개의 포트를 통해 하의 셀 또는 두 개의 셀에 동시에 접근하게 될 때 발생할 수 있는 고장으로 두 가지 또는 그 이상의 약고장(weak fault)이 동시에 발생하여 하나의 고장으로 나타나게 되는 경우를 말한다. 이중 포트 관련 고장모델은 그림 1과 같이 하나의 셀과 관련된 고장(2PF1)과 두 개의 셀에 관련된 고장(2PF2)으로 분류된다. 또한 두 개의 셀에 관련된 고장모델은 피결합셀과 결합셀의 동작 영향에 따라 3개의 고장으로 분류되며 다음과 같다.

1) 2PF2a : 두 포트를 통하여 동시에 결합셀에 접근하게 될 때 피결합셀에 고장이 유발되는 경우이다.
2) 2PF2v : 결합셀이 특정 데이터 상태에 있고 두 포트를 통해 동시에 피결합셀에 접근하게 될 때, 발생하게 되는 고장이다.
3) 2PF2av : 두 개의 포트를 통해 서로 다른 셀인 결합셀과 피결합셀에 동시에 두 개의 동작을 적용할 때, 발생하는 고장이다.

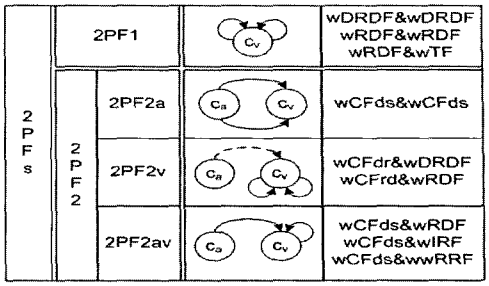


그림 1. 이중포트 메모리관련 고장모델 분류
Fig. 1. Classification of Fault models for Dual-Port Memories.

표 3. 2PFs의 고장모델
Table 3. Fault models of 2PFs.

고장 모델		Fault Primitives
2PF1	wDRDF&wDRDF	<r0;r0/↑/0>,<r1:r1/↓/1>
	wRDF&wRDF	<r1:r1/↓/0>,<r0:r0/↑/1>
	wRDF&wTF	<r0:w↑/0/->,<r1:w↓/1/->
2PF2a	wCFds&wCFds	<w0:rd;0/↑/->,<w0:rd;1/↓/-> <w1:rd;0/↑/->,<w1:rd;1/↓/->
		<rx:rx;0/↑/->, <rx:rx;1/↓/->
2PF2v	wCFdr&wDRDF	<0;r0:r0/↑/0>,<1;r0:r0/↑/0> <0;r1:r1/↓/1>,<1;r1:r1/↓/1>
	wCFrd&wRDF	<0;r1:r1/↓/0>,<1;r1:r1/↓/0> <1;r0:r0/↑/1>,<0;r0:r0/↑/1>
2PF2av	wCFds&wRDF	<w0:r0/↑/1>,<w0:r1/↓/0> <w1:r0/↑/1>,<w1:r1/↓/0>
	wCFds&wIRF	<w0:r0/0/1>,<w0:r1/1/0> <w1:r0/0/1>,<w1:r1/1/0>
	wCFds&wRRF	<w0:r0/0/?>,<w0:r1/1/?> <w1:r0/0/?>,<w1:r1/1/?>

표 3을 보면 고장모델 앞에 w가 붙는데, 이는 약고장(weak fault)을 의미한다. 약고장은 단일 포트의 읽기 또는 쓰기 동작이나 이중 포트에서의 동시에 읽기 동작에 의해 셀의 노드에 작은 전압의 방해로 인해 만들어질 수 있는 부분적인 고장을 말한다. 이 고장은 그 심각도에 따라 전적으로 발생할 수도 있고 부분적으로 발생할 수도 있다. 즉, 여러 개의 약고장이 동시적으로 발생해 그 효과가 가산적일 때 발생할 수 있고, 감산적일 때는 발생하지 않을 수도 있다.

가. 2PF1관련 고장모델

이중 포트 관련 고장(2PFs) 중 하나의 셀과 관련된 고장인 2PF1은 표 3에서와 같이 3개의 고장 모델이 존재한다. 2PF1은 두 포트를 통해서 하나의 셀에 동시에 접근할 때, 접근한 셀에 오동작이 일어나는 경우이다.

표 3에서 2PF1의 FPs가 나타내는 의미는 다음과 같다. <S1:S2/F/R>에서 ":"는 S1과 S2가 동시에 두 개의 포트를 통해 동작하는 것을 나타낸다. S1과 s2는 셀의

상태 또는 동작을 나타내고, F는 피결합셀에 값을 나타내다. R은 만약 FPs에 읽기 동작이 있다면 S1이나 S2의 읽기 결과를 나타낸다. 2PF1의 각 고장모델을 간단히 설명하면 다음과 같다.

- wDRDF&wDRDF : 동시에 하나의 셀에 두 개의 읽기 동작을 적용하였을 때, 그 셀의 값이 바뀌는 고장이다. 이 고장은 처음엔 정확한 값을 반환하지만 셀의 값이 바뀌는 것이 천천히 발생한다.
- wRDF&wRDF : 동시에 하나의 셀에 두 개의 읽기 동작을 적용하였을 때, 그 셀의 값이 바뀌고 틀린 값을 반환하는 고장이다.
- wRDF&wTF : 같은 셀에 동시에 읽기 동작과 쓰는 동작이 적용될 때 발생하는 고장이다. 이 고장은 셀의 같은 열이나 서로 다른 포트의 비트 라인의 잘못된 연결로 발생한다. 이 고장모델은 이중 포트 메모리에서 똑같은 위치에 동시에 읽기와 쓰기 동작이 가능한 경우에만 존재한다.

나. 2PF2관련 고장모델

두 개의 셀에 관련된 고장인 2PF2는 결합셀과 피결합셀의 동작 영향에 따라 2PF2a, 2PF2v, 2PF2av 세 가지의 고장으로 분류된다. 표 3에서 2PF2의 각 고장모델별로 FPs가 나타내는 의미는 다음과 같다.

- 2PF2a의 FPs <Sa;Sa;Sv/F/R>에서 Sa는 동시에 결합셀에 적용되는 동작을 말하며, Sv는 피결합셀의 상태를 나타낸다. F는 피결합셀에 고장 값을 나타내며, R은 읽기 동작이 있을 때 출력으로 나오는 논리값을 나타낸다.
- 2PF2v의 FPs <Sa;Sv:Sv/F/R>에서 Sa는 결합셀의 상태를 나타내며, Sv는 동시에 피결합셀에 적용되는 동작 상태를 나타낸다.
- 2PF2av의 FPs <Sa:Sv/F/R>에서 Sa는 결합셀에 적용되는 동작을 말하며 Sv는 피결합셀의 적용되는 동작을 나타낸다. 여기서 Sa과 Sv는 동시에 동작한다.

이중 포트 메모리에서 2PF2의 세 가지 고장 2PF2a, 2PF2v, 2PF2av가 발생하는 원인과 각 고장의 고장모델에 대해 간단히 설명하면 다음과 같다.

(1) 2PF2a

이 고장은 같은 행(row), 열(column) 또는 대각선의 인접한 셀의 노드 사이 연결에 의해 발생한다. 또한 같은 행에서 인접한 셀의 비트 라인과 셀의 노드 사이 연결에 의해 발생한다.

- wCFds&wCFds는 두 포트를 통하여 동시에 두 개의 동작이 결합셀에 적용될 때, 피결합셀 값이 바뀌는 고장이다. FPs 중에 일부는 같은 위치에 동시에 읽기와 쓰기 동작이 가능한 경우에만 존재한다.

(2) 2PF2v

이 고장은 같은 행, 열 또는 대각선에 속하는 인접한 셀의 노드 사이 연결에 의해 발생한다.

- wCFdr&wDRDF : 결합셀이 특정 상태에 있고 동시에 두 개의 읽기 동작이 피결합셀에 적용될 때, 피결합셀이 바뀌는 고장이다. 읽기 동작을 하면 정확한 값이 반환된다.
- wCFrd&wRDF : 결합셀이 특정 상태에 있고, 동시에 두 개의 읽기 동작이 피결합셀에 적용될 때, 피결합셀이 바뀌는 고장이다. 읽기 동작을 하면 틀린 값을 반환한다.

(3) 2PF2av

이 고장은 같은 또는 인접한 열들에 속하는 다른 포트들의 비트 라인 사이 연결에 의해 발생한다. 그림 2의 (a)의 경우는 같은 열의 비트 라인이 연결되어 발생한 고장이고, (b)는 같이 이웃한 열의 비트 라인 간 연결로 고장이 발생했을 경우이다.

- wCFds&wRDF : 결합셀에 쓰기 동작을 적용하고 동시에 피결합셀에 읽기 동작을 적용할 때, 피결합셀의 값이 바뀌고 틀린 값이 반환된다.
- wCFds&wIRF : 결합셀에 쓰기 동작을 적용하고 동시에 피결합셀에 읽기 동작을 적용할 때, 틀린 값이 반환된다. 피결합셀의 상태는 변하지 않는다는 것을 주목해야 한다.
- wCFds&wRRF : 결합셀에 쓰기 동작을 적용하고 동시에 피결합셀에 읽기 동작을 적용할 때, 임의의 값이 반환된다.

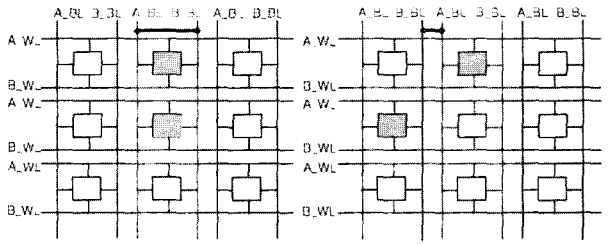


그림 2. 2PF2av가 나타나게 되는 예
Fig. 2. Examples of 2PF2av.

Ⅲ. 제안하는 테스트 알고리즘

이중 포트 메모리에서 발생하는 고장모델은 크게 단일 포트에 관련된 고장모델과 두 개의 포트에 관련된 고장모델이 있다. 단일 포트에 관련된 고장모델을 검출하는 테스트 알고리즘은 MATS+, March C-, PMOVI 등 많이 제안되었다. 하지만 이중 포트 관련 고장모델을 검출하는 테스트 알고리즘은 아직 단일 포트 고장모델은 검출하는 알고리즘만큼 많이 제안되고 있지 못하다. 기존에 제안된 이중 포트 테스트 알고리즘은 특정한 이중 포트 관련 고장모델만을 검출할 수 있는 제한적인 알고리즘이다. 그래서 이중 포트 메모리에서 발생하는 모든 고장모델을 검출하기 위해서는 몇 개의 알고리즘을 같이 사용해야 하는 단점을 가지고 있다. 본 논문에서 이중 포트 메모리에서 발생하는 모든 고장을 검출할 수 있는 효율적인 테스트 알고리즘인 March A2PF를 제안한다.

그림 3은 본 논문이 제안하는 알고리즘 March A2PF로 이중 포트 메모리에서 발생하는 모든 고장을 검출한다. March A2PF를 보면 M0, M1, M2, M3, M4, M5로 6개의 마치요소로 구성되어 있으며, 18N 테스트 패턴 길이를 가진다. N은 메모리 셀의 수를 말한다.

이 장에서는 Ⅱ장에서 설명한 이중 포트 관련 고장(2PFs)의 각 고장모델별 검출하는 조건 설명하고, 제안한 알고리즘이 각 고장모델 검출 조건을 적용하는 방법을 간단히 설명한다.

각 고장모델의 검출 조건을 설명하는데 있어 마치 테스트 표기를 위해 다음과 같은 표기법을 사용한다.

- ':'는 여러 개의 포트들에 동시에 인가되는 작업을 구분하는데 사용한다.

$$\begin{aligned}
&\Updownarrow(w0:-);\\
&\quad M_0\\
&\Uparrow(r0:r0,w1:r0,w1:r1,r1:r1);\\
&\quad M1\\
&\Uparrow(r1:r1,w0:r1,w0:r0,r0:r0);\\
&\quad M2\\
&\Downarrow(r0:r0,w1:r0,w1:r1,r1:r1);\\
&\quad M3\\
&\Downarrow(r1:r1,w0:r1,w0:r0,r0:r0);\\
&\quad M4\\
&\Updownarrow(r0:-);\\
&\quad M5
\end{aligned}$$

그림 3. 새로운 알고리즘 March A2PF
Fig. 3. A new test algorithm March A2PF.

- 'n'은 no operation을, '-'은 가능한 아무 동작을 의미한다.
- 동작이 일어나는 셀의 행과 열은 아래 첨자를 사용하여 표시한다.
- $\Updownarrow_{a=0}^{n-1}$는 주소의 증가($\Uparrow_{a=0}^{n-1}$) 또는 감소($\Downarrow_{0}^{a=n-1}$)를 나타낸다. 그리고 $\Uparrow_{r=0}^{n-1}\Uparrow_{n-1}^{a=r+1}$는 r이 0에서 n-1까지 증가하고 각각의 r값에 대해서 a가 r+1에서 n-1까지 증가함을 나타낸다.

1. 고장모델 검출조건

이중 포트 메모리의 고장모델은 크게 단일 포트에 관련된 고장(1PFs : single port faults)과 두 개의 포트에 관련된 고장(2PFs : two port faults)으로 나누어진다.

가. 1PFs 고장모델 검출조건

1PFs는 하나의 셀과 관련된 고장(1PF1)과 두 개의 셀에 관련된 고장(1PF2)으로 분류되고, 각 고장모델을 검출하는 조건은 다음과 같다.

(1) 1PF1 관련 고장모델 검출 조건

각 고장모델 검출 조건을 살펴보면 다음과 같다.

- 모든 SAFs, RDFs, IRFs 그리고 RRFs는 각 셀을 '1'과 '0'을 읽음으로써 검출한다.
- 모든 TFs는 각 셀에 w↑과 w↓ 동작을 수행한 후에 읽기 동작을 수행함으로써 검출한다.
- 모든 DRDFs는 각 셀에 읽기 동작을 두 번 연속 수행함으로써 검출한다. 첫 번째 읽기 동작은 고장을 활성화시키며, 두 번째 동작은 고장을 검출한다.

(2) 1PF2 관련 고장모델 검출 조건

1PF2 관련 고장모델을 검출하기 위해서는 두 셀의 모든 상태를 고려하여야 한다. 즉, 결합셀과 피결합셀이 00, 01, 10, 11인 네 개의 상태를 고려한다. 또한 결합셀의 주소가 피결합셀의 주소보다 위에 있는 경우와 반대인 경우를 고려하여야 한다. 각각의 고장 검출 조건을 살펴보면 다음과 같다.

- 모든 CFst의 FPs는 두 셀의 네 가지 상태를 읽기 동작에 의해서 발생되고 확인된다.
- 모든 CFds의 FPs는 읽기, 쓰기, 천이 쓰기 동작에 의해서 활성화되고 검출된다.
- 모든 CFrd, CFdr, CFir, CFrr 중에 CFrd, CFir, CFrr의 FPs는 두 셀의 네 가지 상태에서 읽기 동작

에 의해서 검출된다. 그리고 CFdr의 FPs는 한번 더 읽기 동작을 수행함으로 검출된다.

- 모든 CFtr의 FPs는 두 셀의 네 가지 상태에서 천이 쓰기 동작의 적용에 의해서 고장이 활성화되고, 읽기 동작에 의해서 검출된다.

나. 2PFs 고장모델 검출조건

2PFs는 하나의 셀과 관련된 고장(2PF1)과 두 개의 셀에 관련된 고장(2PF2)으로 분류되고, 각 고장모델을 검출하는 조건은 FPs를 바탕으로 설명한다.

(1) 2PF1 관련 고장모델 검출 조건

2PF1의 고장모델은 wDRDF&wDRDF, wRDF&wTF, wRDF&wRDF로 3개가 있다. 각각의 고장 검출 조건을 살펴보면 다음과 같다.

- wDRDF&wDRDF : wDRDF&wDRDF는 <r0;r0/↑/0>, <r1:r1/↓/1>와 같이 2개의 FPs로 구성되어 있고, 이 고장의 검출은 Case A와 Case B의 각각 두 개의 마치 요소로 마치 테스트를 통해 가능하다.

Case A (to detect <r0;r0/↑/0>)
⇕(...,r0:r0); ⇕(r0:-,...);
Case B (to detect <r1:r1/↓/1>)
⇕(...,r1:r1); ⇕(r1:-,...);

첫 번째 마치 요소로 두 포트를 통해 동시에 읽기 동작을 통해 고장이 활성화되고 두 번째 마치 요소인 하나의 읽기 동작에 의해 고장이 검출된다.

- wRDF&wRDF : wRDF&wRDF는 <r0:r0/↑/1>, <r1:r1/↓/0>와 같이 2개의 FPs로 구성되어 있고, 이 고장의 검출은 Case A와 Case B의 각각 두 개의 마치 요소로 마치 테스트를 통해 가능하다.

Case A (to detect <r0:r0/↑/1>)
⇕(...,r0:r0,...);
Case B (to detect <r1:r1/↓/0>)
⇕(...,r1:r1,...);

두 포트를 통해 동시에 읽기 동작을 하는 마치 요소는 고장을 활성화시켜 검출을 한다. wRDF&wRDF는 wDRDF&wDRDF를 검출하면 동시에 검출이 된다.

- wRDF&wTF : wRDF&wTF 는 <r0:w↑/0/->, <r1:w↓/1/->와 같이 2개의 FPs로 구성되어 있고, 이 고장의 검출은 Case A와 Case B의 각각 두 개의 마치 요소로 마치 테스트를 통해 가능하다. 이 고장은 동시에 똑같은 위치에 읽기와 쓰기 동작을 하는데, 이런 경우 읽기 데이터는 버린다.

Case A (to detect <r0:w↑/0/->)
⇕(...,w1:r0); ⇕(r1:-,...);
Case B (to detect <r1:w↓/1/->)
⇕(...,w0:r1); ⇕(r0:-,...);

첫 번째 마치 요소로 두 포트를 통해 동시에 동작을 적용해 고장을 활성화시켜 두 번째 마치 요소인 하나의 읽기 동작에 의해 고장이 검출된다. 이 고장모델은 똑같은 위치에 동시에 읽기와 쓰기 동작이 지원되는 이중 포트 메모리가 존재하지 않는다면 존재하지 않는다.

- 모든 2PF1관련 고장모델 검출 조건

앞에서는 2PF1의 3개 고장모델에 대한 각각의 고장 검출 조건을 설명했다. 2PF1의 3개 고장모델을 검출하기 위해 검출 조건을 각각 다르게 제시하였다. 하지만 다음의 Case A와 Case B의 각각 세 개의 마치 요소로 마치 테스트를 하면, 2PF1의 모든 고장모델의 고장 검출이 가능하다.

Case A (to detect <r0;r0/↑/0>, <r0:r0/↑/1>, <r1:w↓/1/->)
⇕(...,w0:r1); ⇕(...,r0:r0); ⇕(r0:-,...);
Case B (to detect <r1:r1/↓/1>, <r1:r1/↓/0>, <r0:w↑/0/->)
⇕(...,w1:r0); ⇕(...,r1:r1); ⇕(r1:-,...);

(2) 2PF2 관련 고장모델 검출 조건

2PF2는 피결합셀과 결합셀의 동작 영향에 따라 2PF2a, 2PF2v, 2PF2av로 분류된다. 2PF2의 각 고장모델을 검출하는 조건을 살펴보면 다음과 같다.

(가) 2PF2a 고장모델 검출 조건

2PF2a의 고장모델은 wCFds&wCFds로 1개가 있고, 고장 검출 조건은 다음과 같다.

- wCFds&wCFds : wCFds&wCFds는 <w0:rd;0/↑/->, <w0:rd;1/↓/->, <w1:rd;0/↑/->, <w1:rd;1/↓/->, <rx:rx;0/↑/->, <rx:rx;1/↓/->와 같이 8개의 FPs로 구성되어 있고, 이 고장의 검출은 Case A와 Case B의 마치 요소로 마치 테스트를 통해 가능하다. (x∈{0,1}이고 d는 don't care 값)

Case A
$\Updownarrow_{a=0}^{n-1}$(r0:r0,...,w1:rd,...,r1:r1,...,w0:rd);
$\Updownarrow_{a=0}^{n-1}$(r0:-,...);
Case B
$\Updownarrow_{a=0}^{n-1}$(r1:r1,...,w0:rd,...,r0:r0,...,w1:rd);
$\Updownarrow_{a=0}^{n-1}$(r1:-,...);

고장의 결과 값이 1일 때, Case A의 첫 번째 마치 요소로 2PF2a의 모든 고장을 활성화시킨다. 여기서 a의

주소 순서가 증가하면, Case A는 피결합셀이 결합셀보다 위에 있을 경우(v>a)에 첫 번째 마치 요소 "r0:r0"에 의해 검출된다. 반대인 경우(v<a)는 두 번째 마치 요소 "r0:-"에 의해 검출된다. 그리고 a의 주소 순서가 감소하면, Case A는 결합셀이 피결합셀보다 위에 있을 경우(v<a)에 첫 번째 마치 요소 "r0:r0"에 의해 검출되고 반대인 경우(v>a)는 두 번째 마치 요소 "r0:-"에 의해 검출된다. Case B는 고장의 결과 값이 0일 때 고장이 활성화되어 고장을 검출한다.

이 고장모델은 똑같은 위치에 동시에 읽기와 쓰기 동작이 지원되는 이중 포트 메모리가 존재하지 않는다면 <w0:rd;0/↑/->, <w0:rd;1/↓/->, <w1:rd;0/↑/->, <w1:rd;1/↓/-> FPs는 존재하지 않고, 동시에 읽기 동작에 의한 FPs인 <rx:rx;0/↑/->, <rx:rx;1/↓/->만 존재한다.

(나) 2PF2v 고장모델 검출 조건

2PF2v의 고장모델은 wCFdr&wDRDF, wCFrd&wRDF로 2개가 있다. 고장모델의 고장을 검출하기 위해서는 결합셀과 피결합셀의 모든 쌍(c_a,c_v)을 선택하고, 각 쌍을 00, 01, 10, 11의 네 가지 상태 검사를 고려한 마치 테스트에 의해 검출된다.(a∈{0,1,...,v-1,v+1,...,n-1}) 고장모델 wCFdr&wDRDF, wCFrd&wRDF의 검출 조건은 다음과 같다.

- wCFdr&wDRDF : wCFdr&wDRDF는 <0;r0:r0/↑/0>, <1;r0:r0/↑/0>, <0;r1:r1/↓/1>, <1;r1:r1/↓/1>와 같이 4개의 FPs로 구성되어 있고, 이 고장의 검출은 Case A와 Case B의 두 개의 마치 요소로 마치 테스트를 통해 가능하다. 이 고장은 결합셀이 "0" 또는 "1"의 특정 값을 유지하고 동시에 두 개의 읽기 동작이 피결합셀에 적용될 때, 피결합셀이 바뀌는 고장이다. 00, 01, 10, 11의 네 가지 상태가 결합셀이 "0" 또는 "1"일 때, 피결합셀에서 발생하는 고장을 모두 고려한 것이다.

Case A (to detect <0;r0:r0/↑/0>, <1;r0:r0/↑/0>)
⇕(...,r0:r0); ⇕(r0:-,...);
Case B (to detect <0;r1:r1/↓/1>, <1;r1:r1/↓/1>)
⇕(...,r1:r1); ⇕(r1:-,...);

첫 번째 마치 요소로 두 포트를 통해 동시에 읽기 동작을 통해 고장이 활성화되고, 두 번째 마치 요소인 하나의 읽기 동작에 의해 고장이 검출된다. 여기서 Case A로 결합셀이 "0" 과 "1"일 때의 피결합셀에서 발생하는 고장을 검출하고, Case B로 결합셀이 "0" 과 "1"일 때의 피결합셀에서 발생하는 고장을 검출한다. 그 결과 wCFdr&wDRDF와 wCFrd&wRDF의 각 4개씩의 FPs를 모두 검출한다.

- wCFrd&wRDF : wCFrd&wRDF는 <0;r1:r1/↓/0>, <1;r1:r1/↓/0>, <1;r0:r0/↑/1>, <0;r0:r0/↑/1>와 같이 4개의 FPs로 구성되어 있다. wCFrd&wRDF는 wCFdr&wDRDF를 검출하기 위해서 읽기 동작 마치 요소를 두 번 적용하는데, 읽기 동작 마치 요소를 처음 적용했을 때 검출이 된다.

(다) 2PF2av 고장모델 검출 조건

2PF2av의 고장모델은 wCFds&wRDF, wCFds&wIRF, wCFds&wRRF로 3개로 구성된다. 각 고장 모델은 4개의 FPs로 각각 구성되며 다음과 같다.

- wCFds&wRDF : <w0:r0/↑/1>, <w0:r1/↓/0>, <w1:r0/↑/1>, <w1:r1/↓/0>
- wCFds&wIRF : <w0:r0/0/1>, <w0:r1/1/0>, <w1:r0/0/1>, <w1:r1/1/0>
- wCFds&wRRF : <w0:r0/0/?>, <w0:r1/1/?>, <w1:r0/0/?>, <w1:r1/1/?>

고장모델의 고장을 검출하기 위해서는 결합셀과 피결합셀의 모든 쌍(c_a,c_v)을 선택해 두 셀에 동시에 활성화시키는 동작을 적용하고 피결합셀을 읽어 검출한다. 2PF2av 고장 모델은 그림 2에서 보이는 것과 같이 인접한 또는 같은 열들의 다른 포트들 간 비트 라인이 잘못 연결되어 발생한다. 그러므로 2PF2av 고장을 검출하기 위해서는 결합셀이 있을 때, 같은 열의 인접한 피결합셀과 같은 행의 인접한 피결합셀을 테스트하여 검출한다. 그래서 고장을 검출하기 위한 결합셀과 피결합셀의 공간 배치를 보면 그림 4과 같다. 결합셀이 $c=c_{r,c}$ 일 경우, 피결합셀의 공간배치를 보면 다음과 같다.

a. $c_{r+1,c}$과 $c_{r,c+1}$ b. $c_{r-1,c}$과 $c_{r,c+1}$
c. $c_{r+1,c}$과 $c_{r,c-1}$ d. $c_{r-1,c}$과 $c_{r,c-1}$

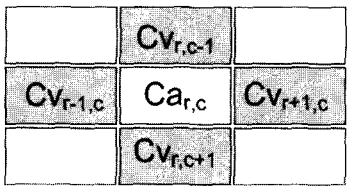


그림 4. 결합셀과 피결합셀의 공간배치
Fig. 4. Space disposition of Aggressor cell and Victim cell.

여기서 셀을 c=$c_{r,c}$로 표기하는데, 이는 셀c의 행과 열의 위치를 말한다. 위의 4개의 경우 중에 하나의 방법을 사용하여 결합셀과 피결합셀 간의 마치 테스트를 통해 2PF2av의 고장을 검출할 수 있다.

2PF2av의 모든 고장 모델 검출 조건은 다음과 같다. wCFds&wRDF, wCFds&wIRF, wCFds&wRRF는 Case A, Case B, Case C, Case D 각각의 마치 요소들 중에 각 그룹에서 한 개씩의 마치 요소를 가지고 마치 테스트를 통해 고장 검출이 가능하다. Case A를 보면, Case A의 마치 요소들 중에 Group A에서 1개와 Group B에서 1개를 선택해 총 2개의 마치요소로 테스트를 하는 것이다. Case B, Case C, Case D도 똑같이 적용된다.

Case A (to detect <w0:r0/↑/1>, <w0:r0/0/1>, <w0:r0/0/?>)

Group A 1. $\Uparrow_{c=0}^{C-1}\Uparrow_{r=0}^{R-1}$(...,w0r,c:r0r+1,c,...);

2. $\Uparrow_{c=0}^{C-1}\Downarrow_{0}^{r=R-1}$(...,w0r,c:r0r-1,c,...);

3. $\Downarrow_{0}^{c=C-1}\Uparrow_{r=0}^{R-1}$(...,w0r,c:r0r+1,c,...);

4. $\Downarrow_{0}^{c=C-1}\Downarrow_{0}^{r=R-1}$(...,w0r,c:r0r-1,c,...);

Group B 1. $\Uparrow_{c=0}^{C-1}\Uparrow_{r=0}^{R-1}$(...,w0r,c:r0r,c+1,...);

2. $\Uparrow_{c=0}^{C-1}\Downarrow_{0}^{r=R-1}$(...,w0r,c:r0r1,c+1,...);

3. $\Downarrow_{0}^{c=C-1}\Uparrow_{r=0}^{R-1}$(...,w0r,c:r0r,c-1,...);

4. $\Downarrow_{0}^{c=C-1}\Downarrow_{0}^{r=R-1}$(...,w0r,c:r0r,c-1,...);

Case B (to detect <w0:r1/↓/0>, <w0:r1/1/0>, <w0:r1/1/?>)

Group A 1. $\Uparrow_{c=0}^{C-1}\Uparrow_{r=0}^{R-1}$(...,w0r,c:r1r+1,c,...);

2. $\Uparrow_{c=0}^{C-1}\Downarrow_{0}^{r=R-1}$(...,w0r,c:r1r-1,c,...);

3. $\Downarrow_{0}^{c=C-1}\Uparrow_{r=0}^{R-1}$(...,w0r,c:r1r+1,c,...);

4. $\Downarrow_{0}^{c=C-1}\Downarrow_{0}^{r=R-1}$(...,w0r,c:r1r-1,c,...);

Group B 1. $\Uparrow_{c=0}^{C-1}\Uparrow_{r=0}^{R-1}$(...,w0r,c:r1r,c+1,...);

2. $\Uparrow_{c=0}^{C-1}\Downarrow_{0}^{r=R-1}$(...,w0r,c:r1r1,c+1,...);

3. $\Downarrow_{0}^{c=C-1}\Uparrow_{r=0}^{R-1}$(...,w0r,c:r1r,c-1,...);

4. $\Downarrow_{0}^{c=C-1}\Downarrow_{0}^{r=R-1}$(...,w0r,c:r1r,c-1,...);

Case C (to detect <w1:r0/↑/1>, <w1:r0/0/1>, <w1:r0/0/?>)

Group A 1. $\Uparrow_{c=0}^{C-1}\Uparrow_{r=0}^{R-1}$(...,w1r,c:r0r+1,c,...);

2. $\Uparrow_{c=0}^{C-1}\Downarrow_{0}^{r=R-1}$(...,w1r,c:r0r-1,c,...);

3. $\Downarrow_{0}^{c=C-1}\Uparrow_{r=0}^{R-1}$(...,w1r,c:r0r+1,c,...);

4. $\Downarrow_{0}^{c=C-1}\Downarrow_{0}^{r=R-1}$(...,w1r,c:r0r-1,c,...);

Group B 1. $\Uparrow_{c=0}^{C-1}\Uparrow_{r=0}^{R-1}$(...,w1r,c:r0r,c+1,...);

2. $\Uparrow_{c=0}^{C-1}\Downarrow_{0}^{r=R-1}$(...,w1r,c:r0r1,c+1,...);

3. $\Downarrow_{0}^{c=C-1}\Uparrow_{r=0}^{R-1}$(...,w1r,c:r0r,c-1,...);

4. $\Downarrow_{0}^{c=C-1}\Downarrow_{0}^{r=R-1}$(...,w1r,c:r0r,c-1,...);

Case D (to detect <w1:r1/↓/0>, <w1:r1/1/0>, <w1:r1/1/?>)

Group A 1. $\Uparrow_{c=0}^{C-1}\Uparrow_{r=0}^{R-1}$(...,w1r,c:r1r+1,c,...);

2. $\Uparrow_{c=0}^{C-1}\Downarrow_{0}^{r=R-1}$(...,w1r,c:r1r-1,c,...);

3. $\Downarrow_{0}^{c=C-1}\Uparrow_{r=0}^{R-1}$(...,w1r,c:r1r+1,c,...);

4. $\Downarrow_{0}^{c=C-1}\Downarrow_{0}^{r=R-1}$(...,w1r,c:r1r-1,c,...);

Group B 1. $\Uparrow_{c=0}^{C-1}\Uparrow_{r=0}^{R-1}$(...,w1r,c:r1r,c+1,...);

2. $\Uparrow_{c=0}^{C-1}\Downarrow_{0}^{r=R-1}$(...,w1r,c:r1r1,c+1,...);

3. $\Downarrow_{0}^{c=C-1}\Uparrow_{r=0}^{R-1}$(...,w1r,c:r1r,c-1,...);

4. $\Downarrow_{0}^{c=C-1}\Downarrow_{0}^{r=R-1}$(...,w1r,c:r1r,c-1,...);

2PF2av의 FPs <w0:r0/↑/1>, <w0:r0/0/1>, <w0:r0/0/?>을 검출하기 위해서는 Case A의 두 그룹에서 각 1개씩 마치 요소를 선택하여야 한다. 예로 Group A.1과 Group B.1을 사용하여 고장을 검출하면, Group A.1의 마치 요소인 "$w0_{r,c}$:$r0_{r+1,c}$"로 두 포트를 통해 동시에 동작을 적용해 고장을 활성화시킨다. 그리고 Group B.1의 마치 요소인 "$w0_{r,c}$:$r0_{r,c+1}$"로 두 포트를 통해 동시에 동작을 적용해 고장을 활성화시킨다. 그리고 Group A.1과 Group B.1의 두 번째 마치 요소인 하나의 읽기 동작에 의해 고장이 검출된다. 여기서 Group A.1은 결합셀($c_{r,c}$)과 피결합셀($c_{r+1,c}$)이 인접한 똑같은 행에 있는 경우이고, Group B.1는 결합셀($c_{r,c}$)과 피결합셀($c_{r,c+1}$)이 인접한 똑같은 열에 있는 경우이다. 이와 같은 방법으로 Case B, Case C, Case D도 고장을 검출한다.

2. March A2PF의 고장 검출 방법

그림 3에서 제안한 알고리즘 March A2PF이 이중 포트 메모리의 각각의 고장모델을 검출하는 과정을 간단히 설명하면 다음과 같다.

가. 1PFs 관련 고장모델 검출

1PFs 관련 고장모델은 그림 3에서 제안한 March A2PF의 알고리즘에서 단일 포트에 해당하는 마치 요소를 가지고 검출한다. March A2PF에서 단일 포트의 마치요소만 표시한 것이 그림 5이다. 그림 5를 사용하여 1PFs의 모든 고장을 3.1장에서 제시한 검출 조건으로 검출한다.

$$\Updownarrow(w0);\quad M_0$$
$$\Uparrow(r0,w1,w1,r1);\quad M1$$
$$\Uparrow(r1,w0,w0,r0);\quad M2$$
$$\Downarrow(r0,w1,w1,r1);\quad M3$$
$$\Downarrow(r1,w0,w0,r0);\quad M4$$
$$\Updownarrow(r0);\quad M5$$

그림 5. 1PFs를 검출하기 위한 March A2PF의 변형
Fig. 5. Modification of March A2PF for detecting 1PFs.

(1) 1PF1 관련 고장모델 검출

그림 5의 알고리즘은 1PF1 관련 고장모델을 모두 검출한다. 1PF1의 모든 고장은 Ⅲ.1장에서 제시한 검출 조건으로 검출한다. 그림 5의 M1, M2, M3를 사용하여 모든 SAFs, RDFs, IRFs, RRFs, TFs 그리고 DRDFs를 검출한다. DRDF는 연속으로 두 번의 읽기 동작을 통해 검출한다.

(2) 1PF2 관련 고장모델 검출

그림 5의 알고리즘은 1PF2 관련 고장모델을 모두 검출한다. 그림 5의 M0, M1, M2, M3, M4, M5 마치 요소에 의해서 두 셀의 00, 01, 10, 11 네 가지 상태를 고려하여, 모든 CFst, CFds, CFrd, CFdr, CFir, CFrr 그리고 CFtr의 FPs를 읽기, 쓰기, 천이 쓰기 동작에 의해 고장을 활성화시키고 검출한다. 예를 들어, CFds의 고장 결과 값이 1일 때, 피결합셀이 결합셀보다 상위 주소에 있으면 M1의 r0로 검출하고, 피결합셀이 결합셀보다 하위 주소에 있으면 M3의 r0로 검출한다. 또한 고장 결과 값이 0이면, M2와 M4가 피결합셀과 결합셀의 위치에 따라 고장을 검출한다.

가. 2PF1 관련 고장모델 검출

그림 3에서 제안한 알고리즘 March A2PF는 2PF1 관련 고장모델을 모두 검출한다. 2PF1의 모든 고장은 3.1장에서 제시한 검출 조건으로 검출한다. March A2PF에서 M3, M4, M5의 마치 요소가 2PF1의 고장 검출 조건을 모두 포함하고 있다. Case A의 마치 요소들을 M4와 M5가 포함하고 있으며, Case B의 마치 요소들은 M3와 M4가 포함하고 있다.

고장 검출 방법을 간단히 설명하면, Case A의 고장 검출 조건을 만족하는 M4는 동시에 두 포트로 읽는 동작 "r0:r0"을 수행하고 M5의 한 포트로 읽는 동작 "r0:-"을 같은 셀에 수행함으로 wDRDF&wDRDF와 wRDF&wRDF의 FPs인 <r0;r0/↑/0>와 <r0:r0/↑/1>를 검출한다. M4에서 "w1:r0"의 동작을 수행하고 "r0:r0"을 수행함으로써 wRDF&wTF의 FPs인 <r1:w↓/1/->를 검출한다. Case B의 고장 검출 조건을 만족하는 M3와 M4의 마치요소로 나머지 FPs <r1:r1/↓/1>, <r1:r1/↓/0>, <r0:w↑/0/->을 검출할 수 있다.

나. 2PF2 관련 고장모델 검출

그림 3의 March A2PF가 2PF2a, 2PF2v, 2PF2av의 각 고장모델을 검출하는 과정을 살펴보면 다음과 같다.

(1) 2PF2a 고장모델 검출

March A2PF는 2PF2a 관련 모든 고장모델을 검출한다. 2PF2a의 모든 고장모델은 Ⅲ.1장에서 제시한 검출 조건으로 검출이 가능하다. March A2PF에서 M1, M2, M3, M4 마치 요소가 2PF2a의 고장 모델 wCFds& wCFds의 8개의 FPs를 검출하기 위한 모든 조건을 포함하고 있다. March A2PF를 사용하여 고장 검출 방법을 간단히 설명하면 다음과 같다.

- <r0:r0;0/↑/->와 <w1:rd;0/↑/->는 피결합셀(v)이 결합셀(a)보다 주소가 더 클 때, M1에 의해 활성화되고 검출된다. 반대로 v<a일 때는 M3에 의해 활성화되고 검출된다.
- <r1:r1;1/↓/->와 <w0:rd;1/↓/->는 v>a일 때, M2에 의해 활성화되고 검출된다. 반대로 v<a일 때는 M4에 의해 활성화되고 검출된다.
- <r0:r0;1/↓/->와 <w1:rd;1/↓/->는 v>a일 때, M3에 의해 활성화되고 M4에 의해 검출된다. 반대로 v<a일 때는 M1에 의해 활성화되고 M2에 의해 검출된다.
- <r1:r1;0/↑/->와 <w0:rd;0/↑/->는 v>a일 때, M4에 의해 활성화되고 검출된다. 반대로 v<a일 때는 M2에 의해 활성화되고 검출된다.

(2) 2PF2v 고장모델 검출

March A2PF는 2PF2v 관련 고장모델을 모두 검출한다. 2PF2v의 모든 고장은 3.1장에서 제시한 검출 조건으로 검출이 가능하다. March A2PF에서 M0, M1, M2, M3, M4, M5 마치 요소가 2PF2v의 고장모델 wCFdr& wDRDF와 wCFrd&wRDF의 FPs를 검출하기 위한 모든 조건을 포함하고 있다. March A2PF를 사용한 고장 검출 방법을 간단히 설명하면 다음과 같다.

a) (ca,cv)가 "00"이 될 때, 마치요소 M0의 쓰기 동작으로 모든 셀에 "0"이 쓰인다. 즉, 결합셀(ca)과 피결합셀(cv)은 "00"이 되는 것이다. 그리고 M1의 두 포트를 통해 동시에 읽기 동작인 "r0:r0"에 의해서 피결합셀에 접근하여 wCFrd&wRDF의 FPs인 <0;r0:r0/↑/1>이 검출되고, wCFdr&wDRDF의 FPs인 <0;r0:r0/↑/0>이 활성화된다.

b) (ca,cv)가 "01"이 될 때, M1의 쓰기 동작 "w1:r1"에 의해 피결합셀에 "1"을 쓴다. 이때 (ca,cv)는 "01"이 된다. M1의 두 포트를 통해 동시에 읽기 동작인 "r1:r1"에 의해서 피결합셀에 접근하여 wCFrd& wRDF의 FPs인 <0;r1:r1/↓/0>이 검출되고, wCFdr

&wDRDF의 FPs인 <0;r1:r1/↓/1>이 활성화된다. 그리고 M2의 읽기 동작 "r1:r1"에 의해 <0;r1:r1/↓/1>이 검출된다.

c) (ca,cv)가 "01"이 일 때, M2의 쓰기 동작 "w0:r0"에 의해 결합셀에 "0"을 쓴다. 이때는 (ca,cv)는 "01"이 유지된다. M2의 두 포트를 통해 동시에 읽기 동작인 "r0:r0"에 의해서 피결합셀에 접근하여 wCFrd&wRDF의 FPs인 <0;r0:r0/↑/1>이 검출되고, wCFdr&wDRDF의 FPs인 <0;r0:r0/↑/0>이 활성화된다. 그리고 M3의 읽기 동작 "r0:r0"에 의해 <0;r0:r0/↑/0>이 검출된다.

d) (ca,cv)가 "11"이 될 때, M3의 쓰기 동작 "w1:r1"에 의해 결합셀에 "1"을 쓴다. 이때 (ca,cv)는 "11"이 된다. M3의 두 포트를 통해 동시에 읽기 동작인 "r0:r0"에 의해서 피결합셀에 접근하여 wCFrd&wRDF의 FPs인 <1;r1:r1/↓/0>이 검출되고, wCFdr&wDRDF의 FPs인 <1;r1:r1/↓/1>이 활성화된다. 그리고 M5의 읽기 동작 "r0:-"에 의해 <1;r1:r1/↓/1>이 검출된다.

e) (ca,cv)가 "10"이 될 때, M4의 쓰기 동작 "w0:r0"에 의해 피결합셀에 "0"을 쓴다. 이때 (ca,cv)는 "10"이 된다. M4의 두 포트를 통해 동시에 읽기 동작인 "r0:r0"에 의해서 피결합셀에 접근하여 wCFrd&wRDF의 FPs인 <1;r0:r0/↑/1>이 검출되고, wCFdr&wDRDF의 FPs인 <1;r0:r0/↑/0>이 활성화된다. 그리고 M5의 읽기 동작 "r0:-"에 의해 <1;r0:r0/↑/0>이 검출된다.

표 4. 2PF2v의 고장검출 상태표
Table 4. State table for detecting 2PF2v.

단계	마치요소	이전상태	동작	다음상태
1	M_0	--	$c_{a,v}$에 대한 'w0:-'	00
2	M_1	00	c_a에 대한 'r0:r0'	00
3		00	c_v에 대한 'w1:-'	01
4		01	c_v에 대한 'r1:r1'	01
5	M_2	01	c_v에 대한 'r1:r1'	01
6		01	c_a에 대한 'w1:-'	11
7		11	c_v에 대한 'r0:r0'	11
8	M_3	11	c_v에 대한 'r0:r0'	11
9		11	c_v에 대한 'w0:-'	10
10		10	c_v에 대한 'r1:r1'	10
11	M_4	10	c_v에 대한 'r1:r1'	10
12		10	c_a에 대한 'w0:-'	00
13		00	c_v에 대한 'r0:r0'	00
14	M_5	00	c_v에 대한 'r0:r0'	00

wCFrd&wRDF 고장은 wCFdr&wDRDF의 고장을 검출하는 과정에서 첫 번째 읽기 동작에서 검출이 된다. 그리고 같은 셀에 두 번째 읽기 동작을 가하면 wCFdr&wDRDF이 검출된다. March A2PF가 2PF2v의 모든 고장을 검출하는 방법을 간단한 상태표로 나타낸 것이 표 4이다.

(3) 2PF2av 고장 모델 검출

March A2PF는 2PF2av 관련 고장 모델을 모두 검출한다. 2PF2av의 모든 고장은 3.1장에서 제시한 검출 조건으로 검출이 가능하다. March A2PF에서 M1, M2, M3, M4 마치 요소가 2PF2av의 고장 모델 wCFds&wRDF, wCFds&wIRF의 모든 고장과 일어날 수 있을 만한 wCFds&wRRF의 모든 고장을 검출하기 위한 모든 조건을 포함하고 있다. 2PF2av 관련 고장을 검출하기 위해서는 결합셀과 같은 열의 인접한 피결합셀을 테스트하고, 같은 행의 인접한 피결합셀을 테스트하여 검출을 한다. 그림 4를 보면 같은 열 그리고 행과 인접한 셀은 4개가 있고, 4개의 인접한 셀의 공간 배치를 보면 4개의 조합이 나오는 것을 3.1장에서 확인할 수 있다. March A2PF는 공간배치 요소 중에 'a. $c_{r+1,c}$과 $c_{r,c\ 1}$'을 사용하여 테스트를 한다. 2PF2av를 검출하기 위해서는 메모리 셀의 행과 열의 증가와 감소가 중요한 요소가 된다. 그림 6은 그림 3에서 제안한 알고리즘 March A2PF에 메모리 셀의 행과 열의 증감 요소를 넣은 알고리즘이다. March A2PF의 2PF2av 관련 각 고장모델 검출요소 포함 관계는 다음과 같다.

a) <w0:r0/↑/1>, <w0:r0/0/1>, <w0:r0/0/?>의 FPs을 검출하기 위해 Case A의 Group A.1과 Group B.4를 사용하는데, March A2PF의 마치 요소 M1이 인접한 열을 테스트하는 $w0_{r,c}$:$r0_{r+1,c}$과 인접한 행을 테스트하는 $w0_{r,c}$:$r0_{r,c\ 1}$을 포함하고 있다.

b) <w0:r1/↓/0>, <w0:r1/1/0>, <w0:r1/1/?>의 FPs을 검출하기 위해 Case B의 Group A.1과 Group B.4를 사용하는데, March A2PF의 마치 요소 M2가 $w0_{r,c}$:$r1_{r+1,c}$과 $w0_{r,c}$:$r1_{r,c-1}$을 포함하고 있다.

c) <w1:r0/↑/1>, <w1:r0/0/1>, <w1:r0/0/?>의 FPs을

표 5. 이중 포트 관련 알고리즘별 성능 비교
Table 5. Comparisons of performances for Dual-Port Memories.

테스트 알고리즘		[8]				[9]				[10]					March A2PF
		2PF1	2PF 2.1.1	2PF 2.1.2	2PF2	r2 PF1	r2 $PF2_{aa}$	r2 $PF2_{vv}$	r2 $PF2_{av}$	2PF1	$2PF2_a$	$2PF2_v$	$2PF2_{av}$	s2PF	
Fault Model \ Test Length		6N	15N	12N	10N	7N	10N	10N	14N	7N	12N	14N	10N	16N	18N
1PF1	SAF, TF, RDF, DRDF, IRF, RRF	단일 포트 메모리의 고장모델은 기존의 단일 포트 메모리를 위한 테스트 알고리즘 이용하여 따로 검출한다.													○
1PF2	CFds, CFst, CFir, CFrr, CFdr, CFrd, CFtr														○
2PF1	wDRDF&wDRDF	○	×	×	×	○	×	×	×	○	×	○	×	○	○
	wRDF&wRDF	○	×	△	○	○	○	○	×	○	○	○	×	○	○
	wRDF&wTF	×	○	×	×	○	○	×	×	○	○	×	×	○	○
2PF2a	wCFds&wCFds	×	×	×	○	×	△	×	×	×	○	△	×	○	○
2PF2v	wCFdr&wDRDF	×	×	×	×	×	×	×	×	×	×	○	×	○	○
	wCFrd&wRDF	×	×	×	△	×	×	○	×	×	×	○	×	○	○
2PF2av	wCFds&wRDF	×	×	○	×	×	×	×	△	×	×	×	○	×	○
	wCFds&wIRF	×	×	○	×	×	×	×	△	×	×	×	○	×	○
	wCFds&wRRF	×	×	○	×	×	×	×	△	×	×	×	○	×	○

○ : 검출 가능. △ : 부분 검출 가능, × : 검출 불가능

검출하기 위해 Case C의 Group A.1과 Group B.4를 사용하는데, March A2PF의 마치 요소 M3가 $w1_{r,c}$:$r0_{r+1,c}$과 $w1_{r,c}$:$r0_{r,c-1}$을 포함하고 있다.

d) <w1:r1/↓/0>, <w1:r1/1/0>, <w1:r1/1/?>의 FPs을 검출하기 위해 Case A의 Group A.1과 Group B.4를 사용하는데, March A2PF의 마치 요소 M4가 $w1_{r,c}$:$r1_{r+1,c}$과 $w1_{r,c}$:$r1_{r,c-1}$을 포함하고 있다.

March A2PF를 사용하여 결합셀과 피결합셀의 모든 쌍(c_a,c_v)을 선택해 두 셀에 동시에 활성화시키는 동작을 적용하고 피결합셀을 읽어 검출한다. 고장 검출 방법을 간단히 설명하면 다음과 같다. 위의 고장모델 검출요소 포함 관계 a)를 보면, 결합셀과 피결합셀의 쌍에서 $w0_{r,c}$:$r0_{r+1,c}$은 결합셀에 "0"을 쓰고 동시에 인접한 열에 "0"을 읽음으로 고장을 활성화시켜 검출하고, $w0_{r,c}$:$r0_{r,c-1}$은 결합셀에 "0"을 쓰고 인접한 행에 0을 읽음으로 고장을 검출한다. 나머지 2, 3, 4도 결합셀에 "0" 또는 "1"을 쓰고 동시에 인접한 열과 행에 "0" 그리고 "1"을 읽음으로 고장을 활성화시켜 검출한다.

$$\Updownarrow_{c=0}^{C-1}(\Updownarrow_{r=0}^{R-1}(w0_{r,c}:-)); \quad M_0$$
$$\Uparrow_{c=0}^{C-1}(\Uparrow_{r=0}^{R-1}(r0_{r,c}:r0_{r,c},w1_{r,c}:r0_{r+1,c},w1_{r,c}:r1_{r+1,c},r1_{r,c}:r1_{r,c})); \quad M_1$$
$$\Uparrow_{c=0}^{C-1}(\Uparrow_{r=0}^{R-1}(r1_{r,c}:r1_{r,c},w0_{r,c}:r1_{r+1,c},w0_{r,c}:r0_{r+1,c},r0_{r,c}:r0_{r,c})); \quad M_2$$
$$\Downarrow_{0}^{c=C-1}(\Downarrow_{0}^{r=R-1}(r0_{r,c}:r0_{r,c},w1_{r,c}:r0_{r,c-1},w1_{r,c}:r1_{r,c-1},r1_{r,c}:r1_{r,c} \quad M_3$$
$$\Downarrow_{0}^{c=C-1}(\Downarrow_{0}^{r=R-1}(r1_{r,c}:r1_{r,c},w0_{r,c}:r1_{r,c-1},w0_{r,c}:r0_{r,c-1},r0_{r,c}:r0_{r,c}. \quad M_4$$
$$\Updownarrow_{c=0}^{C-1}(\Updownarrow_{r=0}^{R-1}(r0_{r,c}:-)): \quad M_5$$

그림 5. 행과 열의 증감 요소를 포함한 March A2PF
Fig. 5. March A2PF including variation element of row and column.

본 논문이 제안하는 알고리즘은 18N의 테스트 패턴의 길이로 이중 포트 메모리에서 발생하는 1PF1, 1PF2, 2PF1, 2PF2a, 2PF2v, 2PF2av의 모든 고장모델을 검출한다. 기존의 알고리즘은 각 고장모델을 별도의 알고리즘을 적용하여 검출한다. 기존의 알고리즘은 각 고장모델을 별도의 알고리즘을 각각 적용하여 검출하는 비효율적인 방법이지만, March A2PF는 하나의 알고리즘으로 이중 포트 메모리에서 발생하는 모든 고장을 검출하는 효율적인 알고리즘이다.

IV. 성능평가

기존의 제안된 이중 포트 메모리 테스트 알고리즘은 단일 포트 관련 고장모델은 기존의 단일포트 메모리 테스트 알고리즘을 사용하여 검출을 하고, 이중 포트 관련 고장모델은 한 알고리즘으로 모두 검출이 안 되는 단점을 가지고 있다. 표 5는 이중 포트 관련 고장모델을 바탕으로 본 논문이 제안한 March A2PF와 기존의 알고리즘의 고장 검출 유무와 알고리즘의 길이를 비교한 것이다. 기존에 제안된 이중 포트 메모리를 테스트

하기 위한 알고리즘은 다음과 같다.

먼저 [8]에서 제안한 알고리즘을 살펴보면, 고장모델 wCFds, wCFid, wRDF 및 wTF를 대상으로 발생하는 고장을 검출하는 각 알고리즘을 제시하였다. 이 알고리즘은 각각 6~15N의 테스트 패턴의 길이로 2PFs 관련 고장의 일부를 검출한다. 이 알고리즘은 단일 포트 관련 고장은 별도의 단일 포트 메모리 테스트 알고리즘을 사용하여 검출한다. 또한 이중 포트 관련 모든 고장을 검출할 수 없지만, 가장 많은 고장을 검출하기 위해서 43N의 테스트 패턴 길이가 필요하다.

다음으로 [9]에서 제안한 알고리즘을 살펴보면 다음과 같다. 단일 포트 관련 고장은 별도의 단일 포트 메모리 테스트 알고리즘을 사용하여 검출하고, 이중 포트 메모리에서 발생하는 2PF1과 2PF2의 고장모델을 검출하는 각각의 알고리즘을 제안하였다. 이 알고리즘은 각각 7~14N의 테스트 패턴의 길이를 가지고 있다. [9]에서 제안한 알고리즘은 4개의 알고리즘을 사용하여 이중 포트 메모리의 고장을 검출하는데 2PF2v 관련 고장모델 중 wCFrd&wRDF는 검출이 불가능하다. 그래서 2PF2v의 고장모델 wCFrd&wRDF를 제외한 모든 이중 포트 관련 고장을 검출하는데 4개의 알고리즘을 사용하여 7+10+10+14=41N의 테스트 패턴의 길이가 필요하다.

마지막으로 [10]에서 제안한 알고리즘은 이중 포트 메모리에서 발생하는 2PF1과 2PF2의 모든 고장모델을 각각의 알고리즘을 사용하여 검출할 수 있다. 이 알고리즘은 각각 7~16N의 테스트 패턴의 길이를 가지고 있다. [10]에서 제안한 알고리즘은 이중 포트 관련 고장모델인 2PF1, 2PF2a, 2PF2v, 2PF2av를 검출하기 위하여 각각 알고리즘을 제안하였다. 모든 2PF1 관련 고장을 검출하는 7N의 알고리즘 March 2PF1, 모든 2PF2a 관련 고장을 검출하는 12N의 March 2PF2a, 2PF2v 관련 모든 고장을 검출하는 14N의 March 2PF2v와 모든 2PF2av 관련 고장을 검출하는 10N의 March 2PF2av를 제안하였다. 기존의 제안된 알고리즘들과는 달리 [10]에서는 2PF1, 2PF2a, 2PF2v의 모든 고장을 검출하는 알고리즘이 March s2PF를 제안하였다. March s2PF는 16n의 테스트 패턴의 길이로 2PF2av 관련 고장을 제외한 모든 고장을 검출한다. 그래서 [10]에서 제안한 알고리즘을 사용하여 모든 이중 포트 관련 고장을 검출하는데 2개의 알고리즘을 사용하면 검출이 가능하다. March s2PF와 March 2PF2av를 사용하면 10+16=26N의 테스트 길이로 모든 고장이 검출된다. 이 알고리즘의 경우도 단일 포트 관련 고장은 별도의 단일 포트 메모리 테스트 알고리즘을 사용하여 검출한다.

기존에 제안한 알고리즘에서는 단일 포트 관련 고장 검출을 위해 별도의 단일 포트 메모리 테스트 알고리즘을 사용하여 하는 단점을 가지고 있다. 그리고 이중 포트 메모리에서 발생하는 이중 포트 관련 모든 고장모델을 검출하기 위해서 각 고장모델별로 독립적인 테스트 알고리즘을 적용하여야 한다. 그래서 모든 고장을 검출하기 위해서 여러 개의 테스트 알고리즘을 사용하여야 함으로 상당히 긴 테스트 알고리즘을 적용되고, 테스트 시간도 길어진다는 단점이 있다. 기존의 여러 개의 테스트 알고리즘을 사용하여 테스트를 할 경우, 테스트 패턴에 중복되는 동작이 많으므로, 이들을 효과적으로 배치하고 테스트 패턴을 줄여 모든 고장을 검출할 수 있는 알고리즘이 필요하다.

기존에 제시된 알고리즘들을 사용하여 이중 포트 메모리의 모든 고장을 검출하려면 최소 26N의 테스트 패턴이 필요하다. 하지만 본 논문에서 제안하는 테스트 알고리즘 March A2PF는 18N으로 이중 포트 메모리에 관련된 모든 고장을 검출할 수 있다. 결과적으로 March A2PF는 이중 포트 메모리의 모든 고장을 하나의 알고리즘으로 테스트하는 것이 가능하고, 보다 다양한 고장을 검출할 수 있는 매우 효과적인 알고리즘이다.

V. 결 론

본 논문에서는 이중 포트 메모리에서 동시에 두 포트의 사용으로 발생하는 모든 고장모델과 기존의 단일 포트에서 발생하는 고장모델을 살펴보고, 그 고장을 잡을 수 있는 효율적인 새로운 알고리즘 March A2PF를 제안하였다. 기존의 이중 포트 메모리 알고리즘은 모든 이중 포트 관련 고장모델을 한 알고리즘으로 검출하지 못하고, 각 고장모델별로 알고리즘을 따로 적용하여 검출하여야 하는 단점을 가지고 있었다. 그리고 단일 포트 관련 고장모델 검출을 위해서는 기존의 단일 포트 메모리 테스트 알고리즘을 사용하였다. 하지만 March A2PF는 하나의 알고리즘으로 이중 포트 관련 2PF1과 2PF2 고장을 모두 검출할 수 있다. 또한 March A2PF는 단일 포트 메모리 알고리즘을 사용하지 않고 단일 포트 관련 고장도 검출할 수 있는 장점을 가진다. 즉, March A2PF는 18N의 짧은 테스트 패턴의 길이로 이중 포트 메모리의 모든 고장을 검출할 수 있는 매우 효율적인 알고리즘이다.

참 고 문 헌

[1] M. Inoue, et. al, "A New Test Evaluation Chip for Lower Cost Memory Tests," Proceedings of IEEE Design and Test of Computers, pp. 15-19, 1993.

[2] T. Matsumura, "An efficient Test Methode for Embedded Multi-Port RAM With BIST circuitry," Proceedings of IEEE International Workshop on Memory Technology, Design and Testing, pp. 62-67, 1995.

[3] A.A. Amin, M.Y. Osman, R.E. Abdel-Aal and H. Al-Muhtaseb, "New fault models and efficient BIST algorithms for dual-port memories," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume 16, pp. 987-1000, 1997.

[4] S. Hamdioui and A.J. Van de Goor, "Thorough testing of any multiport memory with linear tests," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume 21, pp. 217-231, 2002.

[5] A.J. Van de Goor and Z. Al-Ars, "Functional memory faults: a formal notation and a taxonomy," Proceedings of IEEE VLSI Test Symposium, pp. 281-289, 2000.

[6] S. Hamdioui and A.J. Van De Goor, "An experimental analysis of spot defects in SRAMs: realistic fault models and tests," Proceedings of IEEE VLSI Test Symposium, pp. 131-138, 2000.

[7] S. Hamdioui, A.J. Van de Goor and M. Rodgers, "March SS: a test for all static simple RAM lts," Proceedings of IEEE Memory Technology, Design and Testing Workshop, pp. 95-100, 2002.

[8] A.J. Van de Goor and S. Hamdioui, "Fault models and tests for two-port memories," Proceedings of IEEE VLSI Test Symposium, pp. 401-410, 1998.

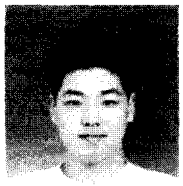
[9] S. Hamdioui, M. Rodgers, A.J. Van de Goor and D. Eastwick, "March tests for realistic faults in two-port memories," Proceedings of IEEE International Workshop, pp. 73-78, 2000.

[10] S. Hamdioui and A.J. Van de Goor, "Efficient tests for realistic faults in dual-port SRAMs," IEEE Transactions on Computers, Volume 51, pp. 460-473, 2002.

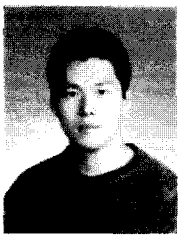
저 자 소 개

박 영 규(학생회원)
2004년 호서대학교 전자공학과 학사 졸업.
2004년 연세대학교 전기전자 공학과 석사 과정.
<주관심분야 : Memory test, DFT, BIST>

양 명 훈(학생회원)
1996년 연세대학교 전기공학과 학사 졸업.
1998년 연세대학교 전기공학과 석사 졸업.
2004년 삼성전자 System LSI 사업부 선임연구원.
2005년 현재 연세대학교 전기전자공학과 박사 과정.
<주관심분야 : DFT, BIST, SoC 설계>



김 용 준(학생회원)
2002년 연세대학교 전기공학과 학사 졸업.
2004년 연세대학교 전기전자 공학과 석사 졸업.
2005년 현재 연세대학교 전기전자 공학과 박사 과정.
<주관심분야 : CAD, DFT, Testing>



이 대 열(학생회원)
2006년 연세대학교 전기전자 공학과 학사 졸업.
2006년 연세대학교 전기전자 공학과 석사 과정.
<주관심분야 : DFT, SoC 테스트>

강 성 호(평생회원)
1986년 서울대학교 제어계측공학과 학사 졸업.
1988년 The University of Texas, Austin 전기 및 컴퓨터공학과 석사 졸업.
1992년 The University of Texas, Austin 전기 및 컴퓨터공학과 박사 졸업.
1992년 미국 Schlumberger Inc. 연구원.
1994년 Motorola Inc. 선임연구원.
2007년 현재 연세대학교 전기전자공학과 교수.
<주관심분야 : SoC 설계 및 응용, DFT, SoC 테스트>