論 文

RAM의 병렬 테스팅을 위한 알고리듬개발 및 테스트회로 설계에 관한 연구

準會員 趙 鉉 默* 正會員 白 京 甲* 正會員 白 寅 天* 正會員 車 均 鉉*

A Study on the Test Circuit Design and Development / of Algorithm for Parallel RAM Testing

Hyeon Mook Cho,* Kyeong Kap Paek,* In Cheon Paik,* Kyun Hyon Tchah* *Regular Members*

要 約

본 논문에서는 RAM에서 발생하는 모든 PSF(Pattern Sensitive Fault)[1][2][3]를 검사하기 위한 알고리듬과 테스트회로를 제안하였다. 기존의 테스트회로와 사용된 알고리듬은 RAM셀들을 연속적으로 테스트하기나 메모리의 2차원적 구조를 사용하지 못했기 때문에 많은 테스트 시간이 소요되었다. 본 논문에서는 기존의 RAM회로에 테스트를 위한 부가적인 회로를 첨가하여 병렬적으로 RAM을 테스트하는 방법을 제안하였다. 부가적으로 첨가된 회로로는 병렬 비교기와 오류 검출기, 그룹 선택회로이고 병렬테스팅을 위해서 수정된 디코더를 사용하였다. 또한, 효과적인 테스트패턴을 구하기 위해 Eulerian경로[2][3][4]의 구성방법에 대해서도 연구를 수행하였다. 결과적으로, 본 논문에서 사용한 알고리즘을 사용하면 b×w=n(b : bit line 수, w : word line 수)의 매트릭스 형태로 표현되는 RAM을 테스트하는데 325×워드라인 수 만큼의 동작이 필요하게 된다. 구현한 각 회로에 대해서 회로 시뮬레이션을 수행한 후 10 bit×32 word Testable RAM을 설계하였다.

ABSTRACT

In this paper, algorithm and testable circuit to find all PSF(Pattern Sensitive Fault) occured in RAM were proposed. Conventional test circuit and algorithm took much time in testing because consecutive test for RAM cells or 2-dimensional memory structure was not employed. In this paper, methodology for parallel RAM-testing was proposed by compensating additional circuit for test to conventional RAM circuit. Additional circuits are parallel comparator, error detector, group selector circuit and a modified decoder used for parallel testing. And also, the constructive method of Eulerian path to obtain efficient test pattern was performed. Consequently, if algorithm proposed in this paper is used, the same operations as 325×word lines will be needed to test b×w=n matrix RAM. Circuit simulation was performed, and 10 bits×32 words testable RAM was designed.

I. 서 론

반도체 기술의 발달로 메모리 분야에서의 집적도 증가가 가속화됨에 따라 짧은 시간내에 메모리를 테

*高麗大學校 電子工學科
Dept. of Electronic Engineering, Korea Univ.
論文番號 : 92-67(接受1991. 8. 8)

스트할 수 있는 기술이 요구되고 있다. 높은 집적도의 메모리 칩에서 인접한 메모리 셀들간의 누설전류에 의해서 많은 오류가 발생하게 되어 이웃하는 셀의 내용을 변하게 하는데 이러한 기능적 결함의 형태를 PSF(Pattern Sensitive Fault)라고 한다. 이러한 PSF는 기존의 stuck-at fault와 더불어 메모리 칩에서 중요한 오류형태를 구성하게 된다.

이러한 오류를 짧은 시간에 최소의 부가회로를 이용하여 외부의 장비에 의존하지 않고 테스트하는 내장 테스트 회로설계와 알고리듬이 필요하게 되었다. 기존의 테스트회로와 사용된 알고리듬은 RAM 셀들을 연속적으로 테스트하거나 메모리의 2차원적 구조를 사용하지 못했기 때문에 많은 테스트 시간이 소요되었다. 본 연구에서는 기존 RAM 회로에 테스트를 위한 부가적인 회로를 첨가하여 병렬적으로 RAM을 테스트하는 방법을 제안한다. On-chip상에서 테스트가 가능하도록 부가적인 회로를 첨가한 RAM의 전체구성은 메모리 셀, 수정된 디코더, 병렬 비교기와 오류 검출기, 감지 증폭기, 그룹 선택기 회로등으로 이루어진다. 본 연구에서는 Eulerian경로와 병렬 테스팅 알고리듬을 이용하여 발생가능한 모든 PSF에 대해서 테스트 시간을 최대한으로 줄이는 방법을 소개한다. 결과적으로, 본 연구에서 제안한 알고리듬을 이용하면 b×w=n(b : bit line w : word line)의 매트릭스 형태로 표현되는 RAM을 완전하게 테스트하는데 325×워드라인 수만큼의 동작이 필요하게 된다.

Ⅱ. Testable RAM의 구조와 설계

테스트를 위한 부가회로를 갖는 testable RAM의 전체 구성은 그림 2.1과 같다. 그림 2.1의 testable RAM은 b×w=n 매트릭스 형태를 이룬다. 여기에서 b는 비트라인의 수이고, w는 워드라인의 수이다. 테스트를 위해서 부가적으로 첨가되거나 수정된 회로는 수정된 디코더, 병렬 비교기와 오류 검출기, 그룹 선택기회로이다.

2.1 메모리 셀의 구조

그림 2.2는 세 개의 트랜지스터로 이루어진 다이내믹 메모리 셀과 제어신호들을 나타낸다. 셀의 상태는 커패시터 C_S에 저장된다. 메모리 셀의 동작중 판독 동작을 예로들어 설명하면, 열의 X-enable 신호는

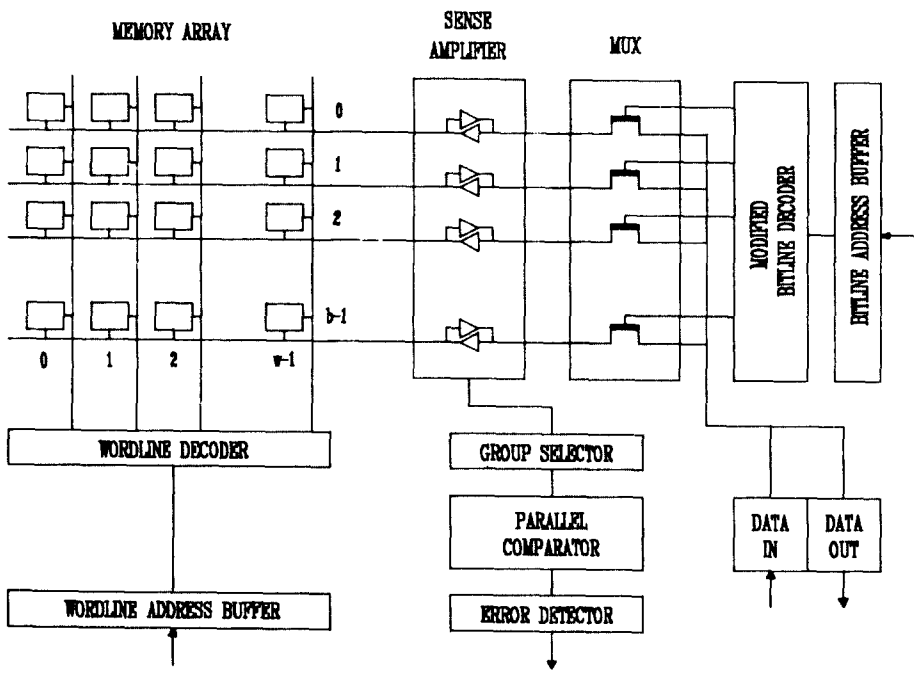


그림 2.1 Testable RAM의 전체 구성도

판독신호와 결합하여 T_3를 on시킨다. T_3는 Column Data Out라인에 대한 클럭이 있는 pull-up부하 역할을 하고 Read신호가 high일 때 on된다. 따라서 커패시터 C_s에 저장된 역전압이 Column Data Out라인에 나타나게 된다. Y-enable신호는 행을 선택해서 T_9을 on시켜서 데이타를 출력 레지스터에 나타나게 한다.

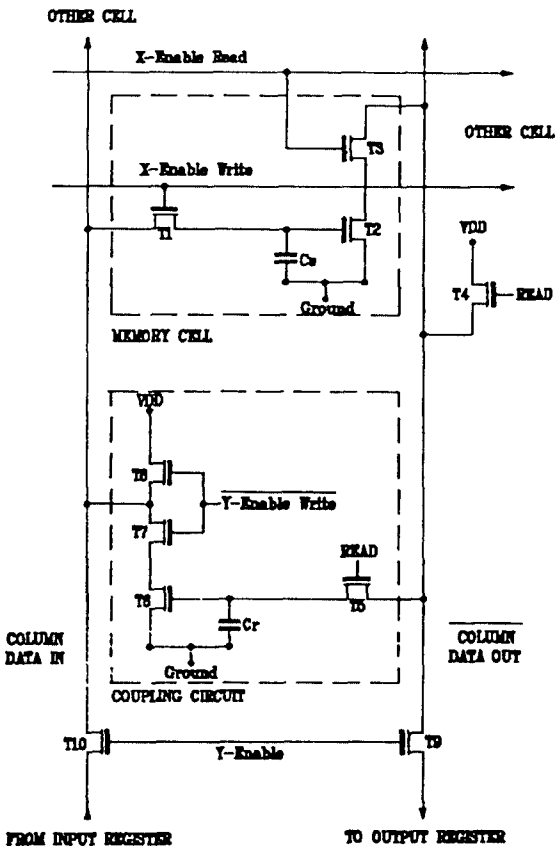


그림 2.2 메모리 셀과 제어신호

2.2 병렬 비교기와 오류 검출기

병렬 비교기와 오류 검출기의 회로[3]를 그림 2.3에 나타내었다. 병렬 비교기와 오류 검출기는 병렬로 선택된 비트선과 연결되어, 선택된 하나의 워드에 대해서 j=mod 5에 의해서 선택된 셀들의 내용을 갖는 그룹 선택기로부터 입력된 신호들이 0이나 1로 같은 값을 갖는지를 검사한다. 만일, 선택된 신호중 하나라도 서로다른 값을 갖는다면 오류 검출기가 오류신호인 ERROR=1을 출력시킨다. N-채널 트랜지스터들인 $T_1, T_2 \cdots T_m$은 그룹 선택기로부터 온 신호들이 모두 동시에 1인지를 감지한다. 반면에, p-채널 트랜지스터들인 $P_1, P_2 \cdots P_m$은 이러한 신호들이 모두 0인지를 검사하게 된다. 트랜지스터 T_0와 P_0는 충전 트랜지스터이고, T_d는 방전 트랜지스터로서 충전 주기 동안에는 단락되고 방전 출력 주기 ϕ_2동안에 도통된다.

2.3 수정된 디코더

수정된 디코더회로는 그림 2.4에 나타내었다. SELECT신호에 의해 구동되는 Q_5, Q_6을 부가함으로서 수정된 디코더회로가 구성된다. 수정된 디코더는 비트 라인을 j=i mod5에 의해서 5개의 집합으로 나누어 선택을 하게 된다. 테스트동작시에는 SELECT=0이 되고, 출력은 입력 어드레스에 관계없이 트랜지스터 Q_5, Q_6에 의해 로직 1이 되므로 비트 라인에 연결된 MUX가 모두 on된다. 정상적인 동작시에는 SELECT=1이 되어 디코더출력은 입력 어드레스 A_1, A_2, A_3, A_4에 의해서 선택된다.

2.4 그룹 선택기

그룹 선택기는 테스트 동작시에, 감지 증폭기의 출력을 j=mod 5에 의해서 5개의 집합으로 나누어서 병렬 비교기로 보내주는 기능을 수행한다. 그림 2.5는 본 연구에서 사용한 그룹 선택기를 나타낸다.

그림 2.5에서는 트랜지스터 Q_1, $Q_2, \cdots Q_{b-1}$는 감지 증폭기와 연결이 되고, 제어신호 S_0, S_1, S_2, S_3, S_4에 의해서 선택이 된다. 정상적인 동작을 수행할 경우에는, $S_0=S_1=S_2=S_3=S_4=1$이 되어 모든 P-트랜지스터가 off되므로 병렬 비교기와 감지 증폭기가 분리된다. 테스트 모드에서는 S_0, S_1, S_2, S_3, S_4중 하나의 신호가 선택되어 그룹의 수만큼 한번에 선택된다.

Ⅲ. RAM의 병렬 테스팅을 위한 알고리듬

3.1 테스트 알고리듬의 기본 이론

C_n으로 정의되는 n-bit RAM은 b개의 비트선과 w개의 워드선으로 n=w×b 형태의 매트릭스로 표현된다. B={0,1,2,…b-1}는 C_n에서의 비트선의 집합이며, W={0,1,2,…w-1}은 워드선의 집합이다. Pair(i,j)는 i번째 비트선과 j번째 워드선의 교점에서의 셀 주소를 나타낸다. 이러한 셀을 C_{ij}로 표시하고 셀 C_{ij}의 상태는 S_{ij}로 표시한다. C_{ij}에서 정상

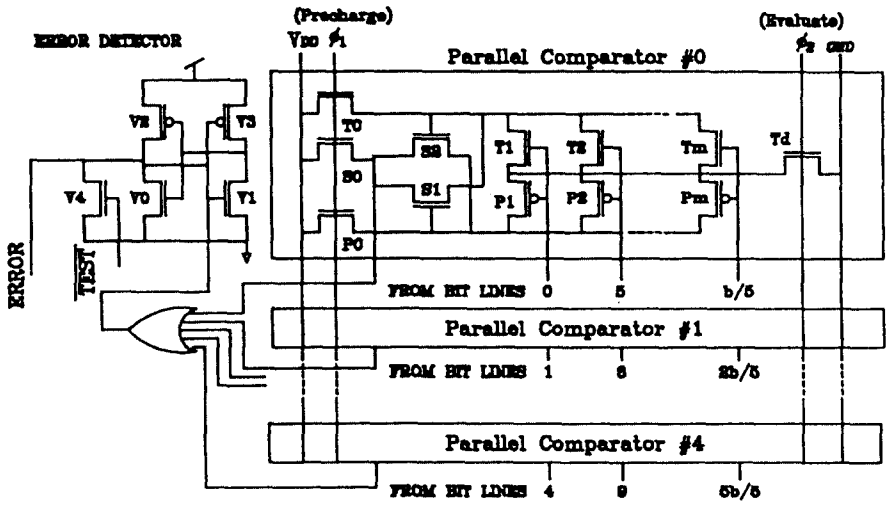


그림 2.3 병렬 비교기와 오류 검출기의 회로도

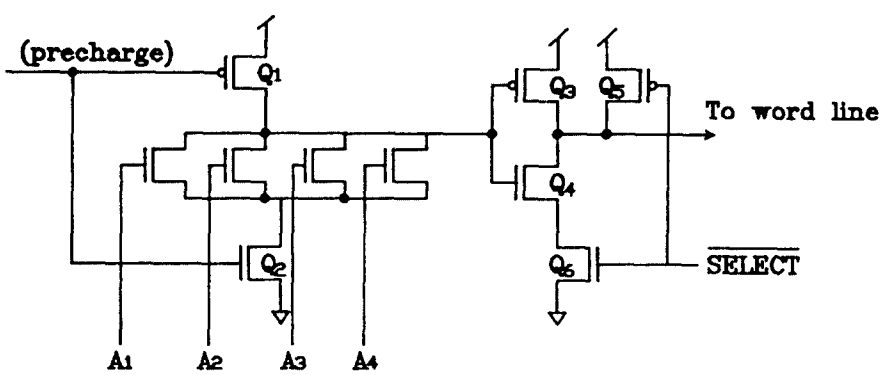


그림 2.4 수정된 디코더의 회로도

적인 동작은 0이나 1을 저장하거나, 또는 그 내용을 판독하는 것이다. 이런 동작들은 $W_0(C_{ij})$, $W_1(C_{ij})$, $R(C_{ij})$로 나타낸다. 테스트 알고리듬의 설명을 위해서 다음을 정의한다.

정의1 : 인접한 4개의 셀들에 둘러싸인 C_{ij}를 중심 셀이라고 하고, 인접한 셀들과 $C_{i\pm1}$와 $C_{i,j\pm1}$을 포함해서 이러한 형태의 셀의 구조를 Type 1 neighborhood[2][3]라고 정의한다. 중심 셀을 없앤 인접한 cell들의 집합을 N_1으로 정의한다.

정의2 : C_{ij}의 값이 N_1의 고정된 패턴에 의해서 결함이 생기면 그러한 결함을 Static Pattern Sensitive Fault(SPSF)라고 정의한다.

정의3 : N_1중의 어떤 한 셀의 값이 0에서 1, 1에서 0으로 변함으로써 C_{ij}의 값이 변하게 될때 그런 결함을 Dynamic Pattern Sensitive Fault(DPSF)라고 정의한다.

각 메모리 셀 C_{ij}는 양의 정수 k로 할당된다. Type 1 neighborhood에 대해서는 k=5이다. 각 neighborhood의 내용은 마진 상태 벡터 $\langle S_{k-1}\ S_{k-2}\cdots S_j\cdots S_0\rangle$로 표현되는데, 여기에서 S_j는 $j=\{0,1,2,\cdots k-1\}$중의 하나로 할당된 셀의 상태를 나타낸다.

크기가 k인 neighborhood의 상태 벡터는 neig-

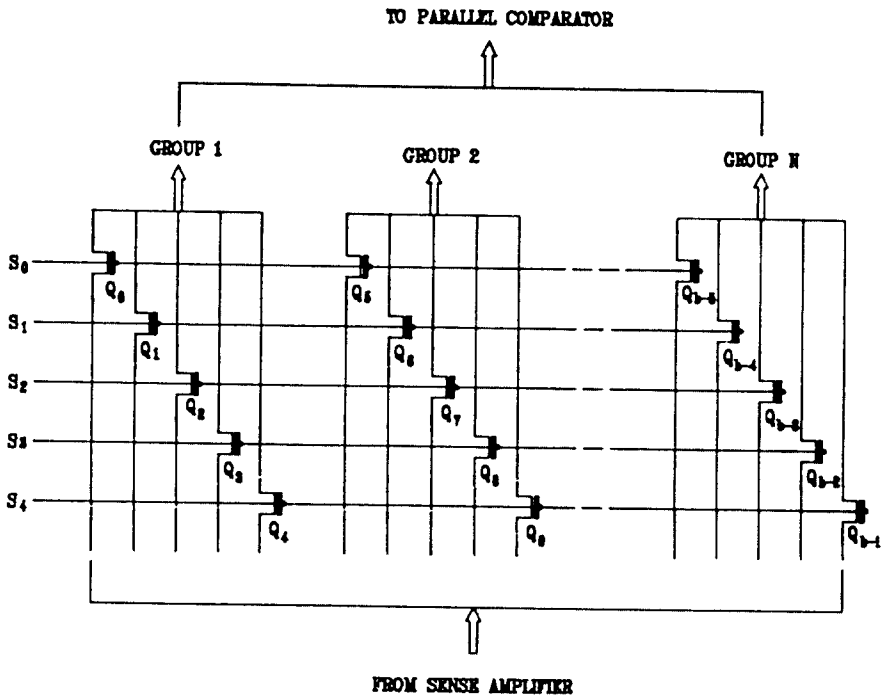


그림 2.5 그룹 선택기의 회로도

hborhood에 저장될 수 있는 모든 가능한 패턴들을 나타내는 node로 구성된 state space graph로 기술될 수 있다. K=3을 갖는 neighborhood에 대한 state space graph를 그림 3.1에 나타내었다. K=3을 갖는 neighborhood에 대한 state space graph에서 노드는 m으로 숫자화 되는데 여기에서, $m=\sum s_i \cdot 2^i$이고, $0 \le m \le 2^k-1$이 된다. 인접한 셀에서의 0이나 1을 쓰는 transition write operation W_i에 의해서 야기되는 상태의 변화는 edge로 나타낸다. 노드의 label들은 비트 패턴을 나타내고, 화살표가 있는 edge는 1 bit transition write를 나타낸다. 두개의 화살표로 된 edge는 SPSF[3]를 검사하고, 하나의 화살표로 된 edge는 DPSF[3]를 검사하는 transition write이다. (여기에서, i=2가 중심 셀이라고 가정한다.) 테스트를 완전하게 하기 위해서는 각각의 edge를 모두 경유하면서 테스트길이를 가장 짧게 하는 경로를 택해야 하는데 이러한 경로를 Eulerian 경로[2][3][4]라고 한다.

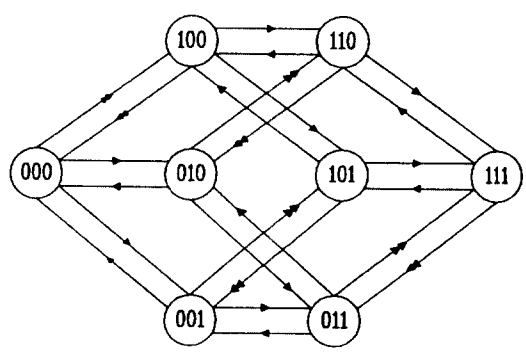


그림 3.1 K=3을 갖는 Neighborhood의 State Space Graph

3.2 RAM에서 PSF를 테스트하기 위한 알고리듬

각 메모리 셀 C_{ij}는 양의 정수 k로 할당된다. 여기에서, $k=(2j+i)(\mathrm{mod}5)$이다. 즉, k={0, 1, 2, 3, 4} 중의 하나의 값이 된다. k값에 의해서 할당된 각 중심 셀 C_{ij}는 할당된 값들이 k값이 아닌 서로 다른 값을 갖는 4개의 셀(N_i)에 의해 둘러싸이게 된다. 그림 3.2

는 1로 할당된 모든 셀이 2, 3, 4, 0으로 할당된 셀들에 의해 둘러싸이게 됨을 보여준다(Type 1 neighborhood의 경우).

j \ i	0	1	2	3	4	5	6
0	0	1	2	3	4	0	1
1	2	3	4	0	1	2	3
2	4	0	1	2	3	4	0
3	1	2	3	4	0	1	2
4	3	4	0	1	2	3	4
5	0	1	2	3	4	0	1

그림 3.2 Cell number assignment

이러한 할당은 셀의 값들이 5개의 연속적인 워드선 후에 반복되는 주기적 패턴을 가짐을 알 수 있다. Table 1은 Type 1 neighborhood에 대해서 다음장에서 설명한 Eulerian 경로로 구한 연속적인 비트 패턴을 나타낸다. Table 1은 모든 SPSF와 DPSF를 감지하는데 필요한 transition write를 나타낸다.

그림 3.3은 RAM에서 Type 1 neighborhood에 대한 모든 PSF를 테스트하는 알고리듬을 설명한다.

Transition write 동작을 수행 (그림 3.3의 Ⓔ－Ⓖ) 할때, P로 할당된 모든 셀들은 #m에 의해서 내용이 바뀌며, 워드라인을 순차적으로 증가시키면서 모든 셀에 대해서 #m 동작을 수행한다. P로 할당된 셀에서 #m에의한 오류의 발생을 감지하기 위해서 메모리의 모든 셀들은 다섯개의 병렬 비교기에 의해서 검사된다. 한번의 판독동작으로, 하나의 워드라인에 해당하는 모든 셀들은 동시에 판독되고 검사가 된다. 따라서, 그림 3.3의 알고리듬에서 Ⓘ를 전체 메모리에 대해서 수행하기 위해서는 워드라인 만큼의 동작이 필요하게 된다.

특정한 비트패턴 m에 대해서 Ⓔ－Ⓚ를 수행하는 것은, P로 할당된 셀에 대해 비트패턴 m에 대한 하나의 SPSF를 감지하는 것이고, P로 할당된 셀 이외의 셀에 대해서는 DPSF를 감지함을 의미한다. Table 1의 모든 비트패턴에 대해서 Ⓔ－Ⓚ를 160번 수행함으로써 각 메모리 셀과 관련된 32개의 SPSF와 128개의 DPSF를 모두 테스트할 수 있다. 그림 3.3의 알고리듬을 정리하면 크게 다음의 3가지 동작으로 모든 PSF를 검사하게 된다.

1)RAM셀의 내용을 모두 0으로 초기화하는 동작

2)160개의 각 패턴을 한번 저장한 후 판독하기 위해서 320번의 저장, 판독동작을 수행

3)모든 워드라인에 대해서 위의 동작을 수행

결과적으로 위의 동작을 모두 수행하기 위해서는 325×워드라인수 만큼의 동작이 필요하게 된다.

Ⅳ. 테스트를 위한 Eulerian 경로

Eulerian 경로는 앞장에서 설명한 state space graph[2]에서 모든 edge를 한번씩 지나는 경로를 의미한다. State space graph에서 각 노드가 이진 N-tuple이고, 노드 n_i와 n_j가 서로 하나의 비트만이 다른값을 가지면서 이 노드들을 연결하는 edge가 존재할 때, 이러한 방향성 그래프를 G_N으로 표시하고 Eulerian graph라고도 한다. 이 그래프는 각 노드로 들어오는 edge와 나가는 edge가 각각 N개이고, G_N에는 $N \cdot 2^N$의 edge가 존재한다. 이러한 Eulerian graph에서 효과적인 테스트 프로시듀어를 구성하기 위해서 Eulerian 경로를 구성해야 한다. 본 연구에서 사용한 방법은 G_{N-1}에서의 Eulerian 경로로 부터 G_N의 Eulerian 경로를 유도해 내는 방법이다. N=2에 대한 Eulerian 경로를 그림 4.1에 나타내었다.

G_N에 대한 Eulerian 경로를 구하기 위해서 G_{N-1}에 대한 Eulerian 경로의 두개의 동일한 경로를 사용한다. 그러한 경로중 하나의 경로에 대한 노드들의 N번째 위치는 0으로 고정되고, 이를 0－경로라 한다. 또한, 다른 경로에 대한 노드들은 N번째 위치가 1로 고정되며 이를 1－경로라고 한다. 각 경로에서 2^{N-1}개의 서로 다른 (N－1) tuple에 대해 초기노드를 포함한 2^{N-1}노드를 crossover 노드로 정한다. G_N에 대한 Eulerian 경로를 구성하기 위해서는 0－경로의 초기노드에서 시작해서 1－경로로 crossover한 후, 이 경로에서 다음 crossover 노드에 도달할 때까지 traverse한 다음 crossover노드에 도달하면 0－경로로 crossover한다. 그런다음 crossover노드를 만날때까지 계속 0－경로를 따라서 traverse하다가 1－경로로 crossover한다. 이런 방법으로 0－경로와 1－경로간에 crossover가 이루어지면서, 또 부분적인 traverse를 수행하면서 0－경로의 마지막 crossover노드까지 도달하게 된다. 같은 방법으로 1－경로의 마지막

Table 1. Eulerian 경로를 이용한 Type 1 neighborhood에서의 바른 패턴

Op #	$S_4S_3S_2S_1S_0$	Op #	$S_4S_3S_2S_1S_0$	Op #	$S_4S_3S_2S_1S_0$	Op #	$S_4S_3S_2S_1S_0$
0	00000	40	01001	80	10010	120	01001
1	01000	41	01000	81	10110	121	11001
2	11000	42	01010	82	11110	122	11000
3	10000	43	01011	83	11010	123	01000
4	10001	44	01111	84	01010	124	01010
5	11001	45	01110	85	01110	125	11010
6	01001	46	01100	86	00110	126	11011
7	00001	47	01101	87	00010	127	01011
8	00011	48	11101	88	00011	128	01001
9	01011	49	11100	89	00111	129	01011
10	11011	50	11110	90	01111	130	01111
11	10011	51	11111	91	01011	131	01101
12	10010	52	11011	92	11011	132	00101
13	11010	53	11011	93	11111	133	00111
14	01010	54	11000	94	10111	134	00011
15	00010	55	11001	95	10011	135	00001
16	00110	56	10001	96	00011	136	10001
17	01110	57	10000	97	10011	137	10011
18	11110	58	10010	98	10010	138	10111
19	10110	59	10011	99	00010	139	10101
20	10111	60	10111	100	00000	140	11101
21	11111	61	10110	101	10000	141	11101
22	01111	62	10110	102	10001	142	11011
23	00111	63	10101	103	00001	143	11001
24	00101	64	10001	104	00101	144	11000
25	01101	65	10101	105	10101	145	11010
26	11101	66	11101	106	10100	146	11110
27	10101	67	11001	107	00100	147	11100
28	10100	68	01001	108	00110	148	10100
29	11100	69	01101	109	10110	149	10110
30	01100	70	00101	110	10111	150	10010
31	00100	71	00001	111	00111	151	10000
32	00101	72	00000	112	01111	152	00000
33	00100	73	00100	113	11111	153	00010
34	00110	74	01100	114	11110	154	00110
35	00111	75	01000	115	01110	155	00100
36	00011	76	11000	116	01100	156	01100
37	00010	77	11100	117	11100	157	01110
38	00000	78	10100	118	11101	158	01010
39	00001	79	10000	119	01101	159	01000
						160	00000

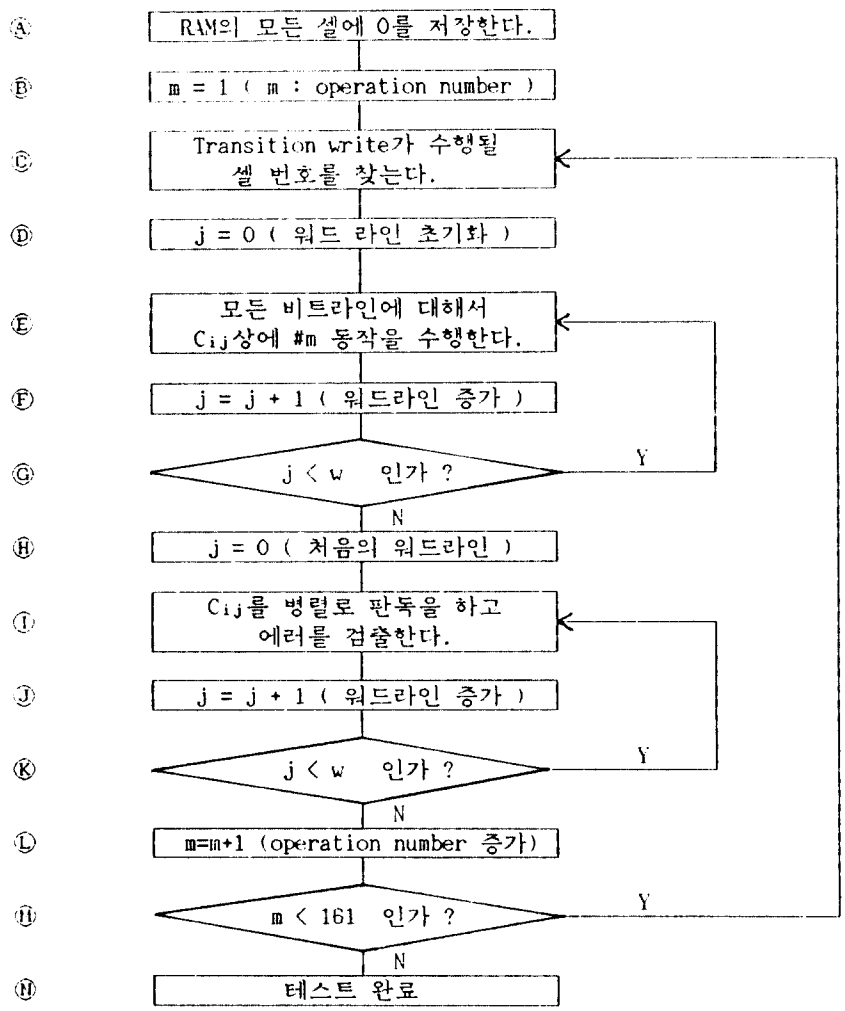


그림 3.3 Type 1 neighborhood에 대한 PSF 검출 알고리듬

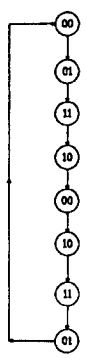
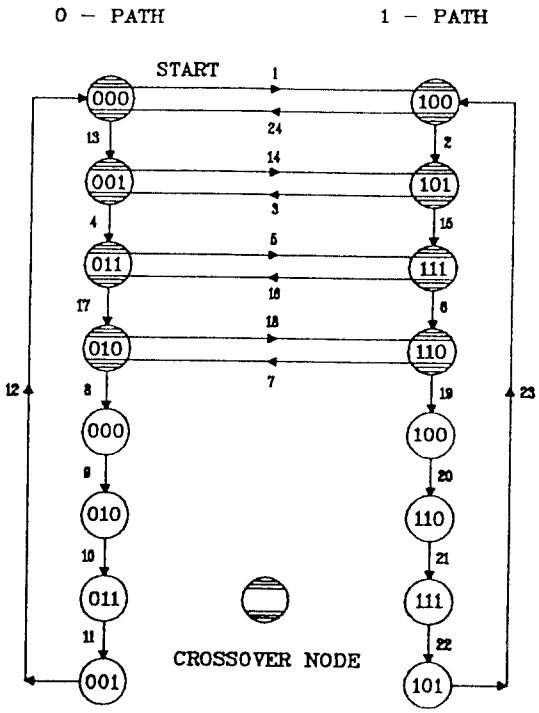


그림 4.1 N=2에 대한 Eulerian 경로

crossover노드를 거친 후 0-경로에 도달하면 원하는 Eulerian 경로를 구할 수 있다. G_N에 대한 Eulerian 경로에서 edge들의 시이퀀스를 EP_N이라고 정의한다. EP_N중 하나의 위치는 ↓ 또는 ↑로 표시하고, 나머지는 0와 1을 가지는 N-tuple로 구성된다. EP_N중 ↓ 또는 ↑의 위치는 N-tuple에서 그것에 해당하는 edge를 traverse할 때의 변화를 나타낸다. 그림 4.2(a)는 N=3에 대한 Eulerian 경로를 나타내고, 그림 4.2(b)는 이러한 Eulerian 경로에 대한 EP_3 시이퀀스를 나타낸다. 일반적인 N값이 대해서 Eulerian 경로를 구하는 컴퓨터 프로그램을 작성하였다.



(a)

1. ↑ 0 0	9. 0 ↑ 0	17. 0 1 ↓
2. 1 0 ↑	10. 0 1 ↑	18. ↑ 1 0
3. ↓ 0 1	11. 0 ↓ 1	19. 1 ↓ 0
4. 0 ↑ 1	12. 0 0 ↓	20. 1 ↑ 0
5. ↑ 1 1	13. 0 0 ↑	21. 1 1 ↑
6. 1 1 ↓	14. ↑ 0 1	22. 1 ↓ 1
7. ↓ 1 0	15. 1 ↑ 1	23. 1 0 ↓
8. 0 ↓ 0	16. ↓ 1 1	24. ↓ 0 0

(b)

그림 4.2 (a)N=3에 대한 Eulerian 경로
(b)N=3의 Eulerian 경로에 대한 EPS 시이퀀스

V. 테스트 방법

Testable RAM은 테스트 모드 선택을 위한 핀, ERROR 검출 PIN, 그룹 별로 비트선을 선택하기 위한 비트선 address 핀(B0, B1,…,B4)이 첨가되어 있다. 테스트 모드가 아닌 정상 동작에서는 일반 RAM과 같이 동작하고 테스트 동작시에는 TEST PIN에 신호가 가해지고 각 word line에 대해 5개의 그룹으로 묶인 비트선에 해당되는 셀에 테스트 패턴이 입력된다. 모든 입력을 완료한 후에 다시 각 워드 선에 해당하는 비트선 정보에 따라 그룹선택기에 의해 병렬 비교기에서 서로 다른 정보, 즉 오류가 있는 지를 판정하게 된다. 테스트 동작중 데이터 저장의 clocking scheme의 한 예를 그림 5.1에 나타내었다.

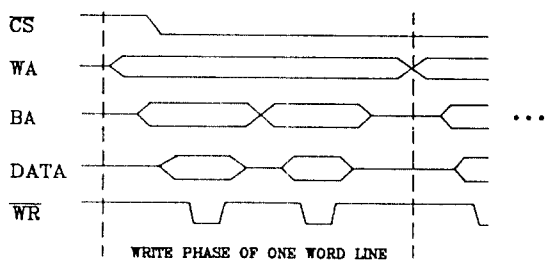


그림 5.1 데이터 저장을 위한 clocking 예

VI. 결 론

본 연구에서는 RAM의 모든 PSF을 검사하는 알고리즘을 설명하고, 칩 상에서 테스트가 가능하도록 기존의 RAM에 부가적인 회로인 병렬 비교기와 오류 검출기를 첨가하고 디코더를 수정하여 10 bit×32 word Testable RAM을 설계하였다. 본 연구에서 사용한 알고리즘을 이용하면, 325×워드수 만큼의 동작으로 모든 PSF을 검사하게 된다. 또한, 본 연구에서는 효과적인 테스트를 위해, 최적의 테스트 패턴을 생성하는 Eulerian 경로의 구성법에 대해서도 소개를 하였다. 본 연구에서 설계한 각 회로에 대해서 회로 시뮬레이션을 수행한 후, 레이아웃을 수행하였다. 전체 레이아웃은 그림 6.1에 나타내었다.

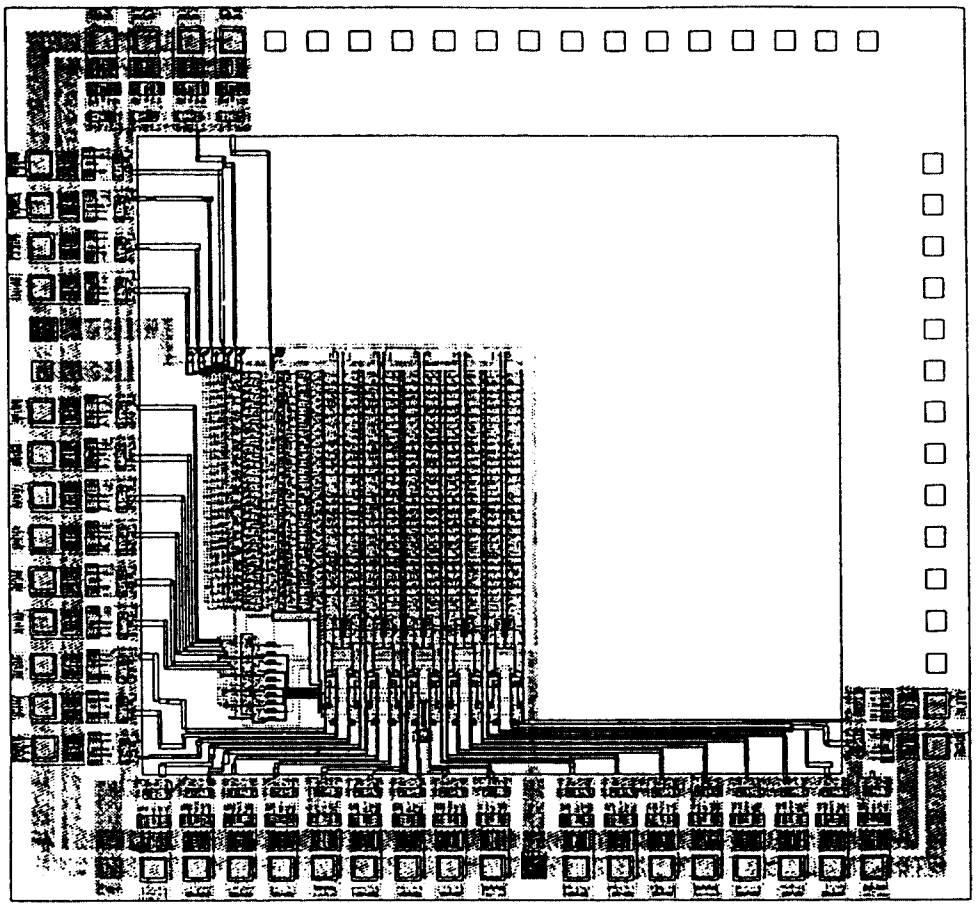


그림 6.1 전체 레이아웃

참 고 문 헌

1. J.P. Hayes, "Detection of pattern-sensitive faults in random access memories," IEEE *Trans. Comput.*, vol. C-24, pp.150-157, Feb. 1975.
2. D. S. Suk and S. M. Reddy, "Test Procedures for a Class of Pattern Sensitive Faults in Semiconductor Random Access Memories," *IEEE Trans on Comput.*, vol. C-29, pp.419-429, June. 1980.
3. P. Mazumder, J.M.Patel, and W.K.Fuchs, "Design and algorithms for parallel testing of random-access and content-addressable memories," in *Proc. Design Automat. Conf.*, vol.24, July 1987, pp.688-694.
4. P. Mazumder, "Parallel testing of parametric faults in three dimensional DRAM's," *IEEE Journal of Solid-State Circuits*, vol.23, pp.933-941, Aug. 1988.
5. Linda E.H.Brackenbury, *Design of VLSI systems*, MACMILLAN EDUCATION LTD 1987.

趙 鉉 默(Hyeon Mook Cho) 準會員

1965年 8月 25日生

1989年 2月 : 高麗大學校 電子工學科 卒業(工學士)

1991年 2月 : 高麗大學校 大學院 電子工學科 卒業(工學碩士)

1992年 現在 : 高麗大學校 大學院 電子工學科 博士課程

※ 主關心分野 : VLSI 設計 및 DSP

白 寅 天(In Cheon Paik) 正會員

1962年 12月 19日生

1985年 2月 : 高麗大學校 電子工學科 卒業(工學士)

1987年 2月 : 高麗大學校 大學院 電子工學科 卒業(工學碩士)

1992年 8月 : 高麗大學校 大學院 電子工學科 卒業(工學博士)

※ 主關心分野 : CAD 및 컴퓨터 네트워크

白 京 甲(Kyeong Kap Paek) 正會員

1965年 11月 8日生

1987年 2月 : 高麗大學校 電子工學科 卒業(工學士)

1990年 2月 : 高麗大學校 大學院 電子工學科 卒業(工學碩士)

1992年 現在 : 高麗大學校 大學院 電子工學科 博士課程

※ 主關心分野 : VLSI 設計 및 通信 시스템

車 均 鉉(Kyun Hyon Tchah) 正會員

1939年 3月 26日生

1965年 : 서울大學校 工學士

1967年 : 美國 일리노이大學校 工學碩士

1976年 : 서울大學校 工學博士

1977年~現在 : 高麗大學校 工科大學 電子工學科 教授

※ 主關心分野 : CAD 및 通信시스템 等