論 文

神經 回路網을 利用한 2비트 에러 檢證 및 修正 回路 設計

正會員 權 健 兌* 正會員 鄭 鎬 宣**

A Design of 2-bit Error Checking and Correction Circuit Using Neural Network

Geoun Tae KWON* Ho Sun CHUNG** *Regular Members*

要 約 본 논문에서는 단층 구조 퍼셉트론 신경 회로망 모델을 사용하여 입력 데이타에서 발생한 2비트의 에러를 검증 및 수정하는 회로를 설계하였다. 순회 해밍 부호를 응용하여 6비트의 데이타 비트와 8비트의 체크 비트를 갖는 (14, 6) 블럭 부호를 사용하였다. 모든 회로들은 이중 메탈 CMOS 2㎛ 설계 규칙에 따라 설계되었다. 회로를 시뮬레이션한 결과, 2비트 에러 검증 및 수정 회로는 최대 67MHz의 입력주파수에서 동작함을 확인하였다.

ABSTRACT In this paper we designed 2 bit ECC(Error Checking and Correction) circuit using Single Layer Perceptron type neural networks. We used (14, 6) block codes having 6 data bits and 8 check bits with appling cyclic hamming codes. All of the circuits are layouted by CMOS 2㎛ double metal design rules. In the result of circuit simulaton, 2-bit ECC circuit operates at 67MHz of input frequency.

I. 서 론

최근 공정 기술이 발달함에 따라 집적 회로 소자의 고집적화가 가능하게 되어 시스템의 가격은 하락했지만, 시스템의 복잡성등이 문제점으로 제기되고 있다. 시스템이 보다 커지고 복잡해짐에 따라 시스템의 각 블록(block)이나 부시스템(sub-system) 간의 데이타 전송시 에러가 발생되는 경우가 있다. 따라서 시스템의 신뢰도(reliability)를 높이기 위해 에러 검증 및 수정 기술이 필요하게 되었다[1]. 기존의 에러 검증 및 수정 회로들은 디지틀 로직(digital logic) 기술[2]을 사용하였기 때문에, 데이타 비트수가 늘어남에 따라 회로가 매우 커지고 데이타의 전송속도도 느려지는 단점이 있다.

본 논문에서는 이런 단점들을 해결하기 위해 Rosenblatt[3]가 제안한 단층 구조 퍼셉트론(

*現代電子産業(株) 半導体研究所 SRAM設計室 勤務
**慶北大學校 電子工學科
Dept. of Electronics, Kyungpook Nat'l Univ.
論文番號 : 91-2 (接受1990. 8. 31)

Single Layer Perceptron) 방식 신경 회로망을 에러 검증 및 수정 회로에 적용함으로써 계산 속도의 향상과 회로의 간략화를 이룰 수 있었다. 순회 해밍 부호(cyclic hamming codes)[4]를 응용하여 6비트의 입력 데이타에서 발생한 2비트의 에러를 검증하고 수정하는 (14, 6) 부호 에러 검증 및 수정 회로를 설계하였다. 여기서 설계된 회로들은 CMOS 이중 배선 2㎛ 설계 규칙으로 레이아웃되었다. 설계 규칙 검사(Design Rule Check)와 전기적 연결 검사(Electrical Rule Check)는 본 연구실에서 개발한 KUIC DRC (Kyungpook national University Intelligent Cad Design Rule Checker)[12]와 KUIC ERC(KUIC Electrical Rule Checker)[13]를 이용하여 수행하였다. 레이아웃 에디터로는 KUIC LED (KUIC Layout EDitor)[15]를 사용하였다.

Ⅱ. 2비트 에러 검증 및 수정 회로 설계

1. 코드 생성

2비트 에러 검증 및 수정 회로의 데이타 비트를 6비트로 잡고 거기서 2비트의 에러를 검증하고, 수정하는 회로를 설계하였다.

$$Df \geq 2t+1 \qquad (1)$$

t : 수정하려는 비트 수

Df : 최소 해밍 거리(minimum hamming distance)

식(1)에 의해 코드 상호간의 최소 거리 즉, 비트의 차이가 적어도 5이상이 되어야만 2비트의 에러를 수정할 수 있다. 이 식에 의해 체크 비트의 수는 최소 8비트가 되었다. (15, 7)코드를 구하여 거기서 단축시켜 (14, 6)코드를 만들었다. $(X^{15}+1)$의 인수를 구하여, 이것들 중 하나인 $(X^8+X^7+X^6+X^4+1)$을 전체 코드 발생 다항식인 G(x)로 잡았다. 이 값과 6비트의 데이타 비트에서 생긴 데이타 다항식과 곱하면 전체 코드가 발생된다. 데이타 다항식은 십진수를 6비트의 2진수 데이타로 나타낸 것으로서 "000000"부터 "111111"까지 즉, 십진수 0부터 63

표 1. 64개의 코드.
Table 1. 64 codewords.

	데이타 비트						체크 비트							
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	1	0	0	0	1	0	1	1	1
3	0	0	0	0	1	0	0	0	1	0	1	1	1	0
4	0	0	0	0	1	1	0	0	1	1	1	0	0	1
5	0	0	0	1	0	0	0	1	0	1	1	1	0	0
6	0	0	0	1	0	1	0	1	0	0	1	0	1	1
7	0	0	0	1	1	0	0	1	1	1	0	0	1	0
8	0	0	0	1	1	1	0	1	1	0	0	1	0	1
9	0	0	1	0	0	0	1	0	1	1	1	0	0	0
10	0	0	1	0	0	1	1	0	1	0	1	1	1	1
11	0	0	1	0	1	0	1	0	0	1	0	1	1	0
12	0	0	1	0	1	1	1	0	0	0	0	0	0	1
13	0	0	1	1	0	0	1	1	1	0	0	1	0	0
14	0	0	1	1	0	1	1	1	1	1	0	0	1	1
15	0	0	1	1	1	0	1	1	0	0	1	0	1	0
16	0	0	1	1	1	1	1	1	0	1	1	1	0	1
17	0	1	0	0	0	0	0	1	1	1	0	0	0	0
18	0	1	0	0	0	1	0	1	1	0	0	1	1	1
19	0	1	0	0	1	0	0	1	0	1	1	1	1	0
20	0	1	0	0	1	1	0	1	0	0	1	0	0	1
21	0	1	0	1	0	0	0	0	1	0	1	1	0	0
22	0	1	0	1	0	1	0	0	1	1	1	0	1	1
23	0	1	0	1	1	0	0	0	0	0	0	0	1	0
24	0	1	0	1	1	1	0	0	0	1	0	1	0	1
25	0	1	1	0	0	0	1	1	0	0	1	0	0	0
26	0	1	1	0	0	1	1	1	0	1	1	1	1	1
27	0	1	1	0	1	0	1	1	1	0	0	1	1	0
28	0	1	1	0	1	1	1	1	1	0	0	0	1	1
29	0	1	1	1	0	0	1	0	0	1	0	1	0	0
30	0	1	1	1	0	1	1	0	0	0	0	0	1	1
31	0	1	1	1	1	0	1	0	1	1	1	0	1	0
32	0	1	1	1	1	1	1	0	1	0	1	1	0	1
33	1	0	0	0	0	0	0	1	0	1	0	0	1	0
34	1	0	0	0	0	1	0	1	0	0	0	1	0	1
35	1	0	0	0	1	0	0	1	1	1	1	1	0	0
36	1	0	0	0	1	1	0	1	1	0	1	0	1	1
37	1	0	0	1	0	0	0	0	0	0	1	1	1	0
38	1	0	0	1	0	1	0	0	0	1	1	0	0	1
39	1	0	0	1	1	0	0	0	1	0	0	0	0	0
40	1	0	0	1	1	1	0	0	1	1	0	1	1	1

41	1 0 1 0 0 0	1 1 1 0 1 0 1 0
42	1 0 1 0 0 1	1 1 1 1 1 1 0 1
43	1 0 1 0 1 0	1 1 0 0 0 1 0 0
44	1 0 1 0 1 1	1 1 0 1 0 0 1 1
45	1 0 1 1 0 0	1 0 1 1 0 1 1 0
46	1 0 1 1 0 1	1 0 1 0 0 0 0 1
47	1 0 1 1 1 0	1 0 0 1 1 0 0 0
48	1 0 1 1 1 1	1 0 0 0 1 1 1 1
49	1 1 0 0 0 0	0 0 1 0 0 0 1 0
50	1 1 0 0 0 1	0 0 1 1 0 1 0 1
51	1 1 0 0 1 0	0 0 0 0 1 1 0 0
52	1 1 0 0 1 1	0 0 0 1 1 0 1 1
53	1 1 0 1 0 0	0 1 1 1 1 1 1 0
54	1 1 0 1 0 1	0 1 1 0 1 0 0 1
55	1 1 0 1 1 0	0 1 0 1 1 0 0 0
56	1 1 0 1 1 1	0 1 0 0 0 1 1 1
57	1 1 1 0 0 0	1 0 0 1 1 0 1 0
58	1 1 1 0 0 1	1 0 0 0 1 1 0 1
59	1 1 1 0 1 0	1 0 1 1 0 1 0 0
60	1 1 1 0 1 1	1 0 1 0 0 0 1 1
61	1 1 1 1 0 0	1 1 0 0 0 1 1 0
62	1 1 1 1 0 1	1 1 0 1 0 0 0 1
63	1 1 1 1 1 0	1 1 1 0 1 0 0 0
64	1 1 1 1 1 1	1 1 1 1 1 1 1 1

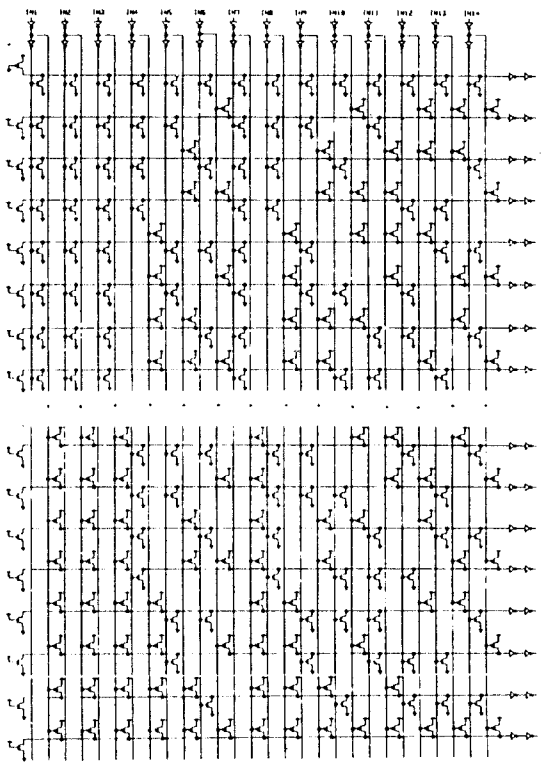


그림 1. 단층 구조 퍼셉트론 방식 2비트 에러 검증 회로
Fig. 1. 2 bit Error Checking circuit using SLP model.

까지의 값이다. 이 값들을 다항식화한 후 G(x)와 곱한다.

예를 들면, 십진수 9의 값, "001001"을 다항식화 하면 X^5+X^2가 된다. 이 식과 G(x)를 곱하면 $X^{13}+X^{12}+X^{11}+X^{10}+X^9+X^9+X^8+X^6+X^5+X^2$이 된다. 여기서 더하기 해 줄 때 승수가 같은 항끼리는 서로 삭제하면 된다. 그러므로, 최종값은 $X^{13}+X^{12}+X^{11}+X^{10}+X^8+X^6+X^5+X^2$이 된다. 이 값을 다시 디지틀화 하면 "00100110101111"이 된다. 이런 방법으로 나머지 항들도 구하여 64개의 코드를 구성하면 다음과 같다.

2. 에러 검증 회로 설계

그림 1은 2비트 에러 검증 회로이다. 14개의 입력단자를 가지며 각 입력단자에서 2개의 인버터를 통해 신호가 서로 상반되게 들어온다. 가운데의 PMOS와 NMOS로 구성된 부분이 시냅스 부분이며, 왼편에 바이어스 시냅스가 붙어 있다. 표 1의 코드에 의해 각 비트의 값이 1이면 PMOS, 0이면 NMOS로 회로를 구성한다. 여기서 PMOS의 W/L값은 6/2, NMOS는 2/2의 값을 가진다. 이 값들을 각각의 MOS의 기준값으로 사용한다. 그리고 좌측의 바이어스 시냅스들이 입력 상태를 검증하여 에러의 발생 여부를 결정하는 회로이다. 출력단에 있는 버퍼(buffer) 즉, 뉴런 부분이 전압을 증폭하여 출력을 "HIGH" 또는 "LOW" 상태로 만들어 준다. 회로 왼편에 바이어스 시냅스는 항상 "ON" 되는 NMOS를 사용한다. 예외로, 전부 0인 "00000 0 00000000" 코드에서는 항상 "ON" 되는 PMOS를 사용한다. NMOS 바이어스 시냅스의 컨덕턴스 비는 표 10의 코드에서 각 단의 1의 갯수에서 체크 하려는 비트수 2를 빼면 된다. PMOS 바이어스는 체크하는 비트수만큼의 값을

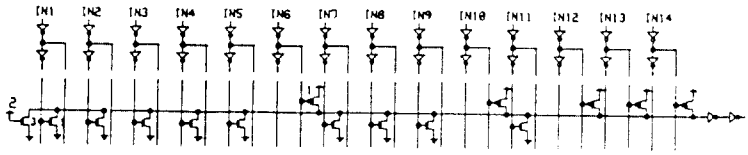


그림 2. "000001 00010111" 코드 검출을 위한 에러 검출 회로
Fig. 2. Error checking circuit for "000001 00010111" code.

컨덕턴스 값으로 갖는다. 즉 2의 값, 12/2의 W/L 값을 갖는다. 예를 들면 "000001 00010111"의 패턴인 경우 2의 값을 가지므로 NMOS 바이어스는 2/2의 3배인 6/2의 W/L 값을 가지게 된다. PMOS는 0V의 입력이 들어올 때 "ON" 되며 NMOS는 5V의 입력이 들어올 때 "ON" 된다.

임의의 입력이 들어오면 거기에 따라 이 PMOS, NMOS가 ON, OFF 되고 그 값에 바이어스 시냅스의 값이 더해져서 뉴런부분을 지치게 된다. 이 바이어스 시냅스들이, 들어온 입력이 64개 코드중에서 자기와 가장 비슷한 패턴을 가진 코드를 거친 뉴런의 출력만 5V의 값 즉, "HIGH" 상태가 되고 나머지 63개의 출력은 "LOW"의 상태가 되는 역할을 한다.

그림 2는 1의 값을 갖는 코드에 대해 간단히 그린 에러 검증 회로이다. PMOS와 NMOS의 시냅스는 각각 1의 컨덕턴스 값을 가지고, 왼쪽의 바이어스 시냅스는 3의 값을 가진다.

이런 회로에 입력으로 다음의 7가지 상태가 들어왔다고 가정할 때 출력 상태를 살펴보겠다.

1) 자기 자신의 입력 "000001 00010111"이 들어왔을 때
   PMOS : 다섯개 모두 "ON"
   ..... 컨덕턴스의 합은 5
   NMOS : 모두 "OFF", 바이어스 "ON"
   ..... 컨덕턴스의 합은 3
   출력값은 컨덕턴스 차에 의해 "HIGH"
   ..... 코드 검증
2) 1비트 빠진 입력 "000000 00010111"이 들어왔을 때
   PMOS : 네개만 "ON"
   ..... 컨덕턴스의 합은 4
   NMOS : 모두 "OFF", 바이어스 "ON"
   ..... 컨덕턴스의 합은 3
   출력값은 컨덕턴스 차에 의해 "HIGH"
   ..... 에러 검증
3) 2비트 빠진 입력 "000000 00000111"이 들어왔을 때
   PMOS : 세개만 "ON"
   ..... 컨덕턴스의 합은 3
   NMOS : 모두 "OFF", 바이어스 "ON"
   ..... 컨덕턴스의 합은 3
   출력값은 컨덕턴스 차에 의해 "HIGH"
   ..... 에러 검증
4) 1비트 빠지고, 1비트 더해진 입력 "000000 00011111"이 들어왔을 때
   PMOS : 네개만 "ON"
   ..... 컨덕턴스의 합은 4
   NMOS : 한개 "ON", 바이어스 "ON"
   ..... 컨덕턴스의 합은 4
   출력값은 컨덕턴스 차에 의해 "HIGH"
   ..... 에러 검증
5) 1비트 더해진 입력 "000001 00011111"이 들어왔을 때
   PMOS : 모두 "ON"
   ..... 컨덕턴스의 합은 5
   NMOS : 한개만 "ON", 바이어스 "ON"
   ..... 컨덕턴스의 합은 4
   출력값은 컨덕턴스 차에 의해 "HIGH"
   ..... 에러 검증
6) 2비트 더해진 입력 "000011 00011111"이 들어왔을 때
   PMOS : 모두 "ON"
   ..... 컨덕턴스의 합은 5
   NMOS : 두 개 "ON", 바이어스 "ON"

..... 컨덕턴스의 합은 5
출력값은 컨덕턴스 차에 의해 "HIGH"
..... 에러 검증

7) 3비트 더해진 입력 "000111000111111"이 들어왔을 때
PMOS : 모두 "ON"
..... 컨덕턴스의 합은 5
NMOS : 세개 "ON", 바이어스 "ON"
..... 컨덕턴스의 합은 6
출력값은 컨덕턴스 차에 의해 "LOW"
..... 다른 코드 검증

이 바이어스 시냅스는 들어온 입력이 64개 코드중에서 자기와 가장 비슷한 패턴을 가진 코드를 거친 뉴런의 출력값만 5V의 값 즉, "HIGH" 상태가 되고, 나머지 63개 출력은 "LOW"의 상태가 되는 역할을 한다. 예를 들면 표 1에서 두번째 코드 "000001 00010111"에서 에러가 발생했을 때 나타날 수 있는 모든 경우의 수는 105가지이다. 1비트의 에러가 발생했을 때, 2비트의 에러가 발생했을 경우이다. 각각의 경우는 다음과 같다.

표 2. "000001 00010111"에서 1비트 더해진 에러 상태
Table 2. 1-bit added error state of "000001 00010111" code.

100001 00010111	000001 10010111
010001 00010111	000001 01010111
001001 00010111	000001 00110111
000101 00010111	000001 00011111
000011 00010111	

표 3. "000001 00010111"에서 1비트 빠진 에러 상태
Table 3. 1-bit subtracted error state.

000000 00010111
000001 00000111
000001 00010011
000001 00010101
000001 00010110

표 4. "000001 00010111"에서 1비트 더해지고 1비트 빠진 상태.
Table 4. 1 bit added and 1 bit subtracted error state.

100000 00010111	010000 00010111	001000 00010111
100001 00000111	010001 00000111	001001 00000111
100001 00010011	010001 00010011	001001 00010011
100001 00010101	010001 00010101	001001 00010101
100001 00010110	010001 00010110	001001 00010110
000100 00010111	000010 00010111	000000 10010111
000101 00000111	000011 00000111	000001 10000111
000101 00010011	000011 00010011	000001 10010011
000101 00010101	000011 00010101	000001 10010101
000101 00010110	000011 00010110	000001 10010110
000000 01010111	000000 00110111	000000 00011111
000001 01000111	000001 00100111	000001 00001111
000001 01010011	000001 00110011	000001 00011011
000001 01010101	000001 00110101	000001 00011101
000001 01010110	000001 00110110	000001 00011110

표 5. "000001 00010111"에서 2비트 더해진 경우
Table 5. 2-bit added error state.

110001 00010111	011001 00010111	001101 00010111
101001 00010111	010101 00010111	001011 00010111
100101 00010111	010011 00010111	001001 10010111
100011 00010111	010001 10010111	001001 01010111
100001 10010111	010001 01010111	001001 00110111
100001 01010111	010001 00110111	001001 00011111
100001 00110111	010001 00011111	100001 00011111
000111 00010111	000011 10010111	000001 11010111
000101 10010111	000011 01010111	000001 10110111
000101 01010111	000011 00110111	000001 10011111
000101 00110111	000011 00011111	000101 00011111
000001 01110111	000001 00111111	000001 01011111

표 6. "000001 00010111"에서 2비트 빼진 경우
Table 6. 2-bit subtracted error state.

000000 00000111	000001 00000011	000001 00010001
000000 00010011	000001 00000101	000001 00010010
000000 00010101	000001 00000110	000001 00010100
000000 00010110		

위에서와 같은 105가지의 입력이 들어왔을 때 두번째 코드의 출력값이 "HIGH"가 되고 나머지 63개의 출력값은 "LOW"가 된다. 이렇게 하여 에러가 발생함을 검출하게 된다.

3. 에러 수정 회로 설계

그림 3이 에러를 검출한 결과값에 의해 에러를 수정하는 회로이다. 5V의 신호값에 의해 "ON" 될 수 있게 NMOS를 사용하였고, 각각의 64개의 코드에 의해 "1"의 위치에 NMOS를 구성한다. "0"의 위치는 비워 두는 대신에 제일 윗단에 5/2의 PMOS로써 바이어스값을 걸어주었다. NMOS는 2/2의 W/L값을 가진다. "000001 0010111" 코드를 가지는 에러 검출 회로에서 에러를 검출한 결과값이 "HIGH"가 나올 때 이 값에 해당하는 층의 NMOS가 모두 "ON" 될 것이다. 그러하여, 0V의 출력 신호가 나오고, 나머지 자리에는 바이어스 저항소자에 의해 5V의 값이 나오게 된다. 두 개의 MOS가 같이 "ON" 되는 자리에는 PMOS가 5/2의 W/L 값을 가지므로 NMOS의 출력 값에 영향을 미치지 못한다.

최종적으로, 이 값들을 원하는 데이타 값으로 바꾸어 주기 위해 인버터를 사용하였다. 이 인버터들의 W/L값은 PMOS는 12/2, NMOS는 5/2의 값을 기준값으로 사용하였다. 입력 인버터나 출력 인버터나의 인버터들은 "3"의 팬더턴스 값 즉, PMOS는 36/2, NMOS는 15/2의 값을 사용하였다. 그 이유는 팬아웃(fanout) 때문이다.

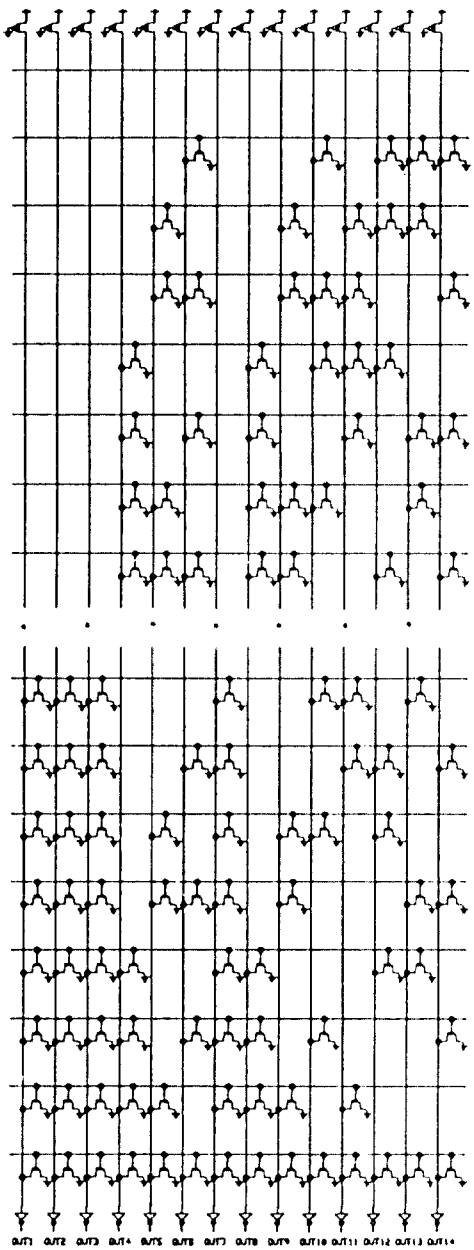


그림 3. 2비트 에러 수정 회로
Fig. 3. 2 bit error correction circuit.

Ⅲ. 회로 시뮬레이션 및 레이아웃

1. 회로 시뮬레이션

"000001 0010111"과 "000000 0000000"의 에러 입력 상태에 대해서 2 비트 에러 검증 및 수정 회로를 PSPICE 시뮬레이션하였다. 그림 4는 입력 패턴으로서 입력을 초기치는 0으로 고정시킨 후 각각 5 nano second의 시간마다 1 더해진 경우, 2 더해진 경우, "000001 0010111"의 패턴, 거기서 1 빠진 경우, 2빠진 경우, 1 더해진 경우, 2 더해진 경우를 시뮬레이션 하였다. 각각의 입력 형태에 대한 시뮬레이션 결과가 그림 5이다. "000000 00000000"와 가까운 입력이 들어왔을 때는 "000000 00000000"가 나오고, "000001 0010111"과 가까운 입력이 들어왔을 때는 "000001 0010111"의 패턴이 나왔다. 회로를 시뮬레이션한 결과, 2비트 ECC회로는 최대 67MHz

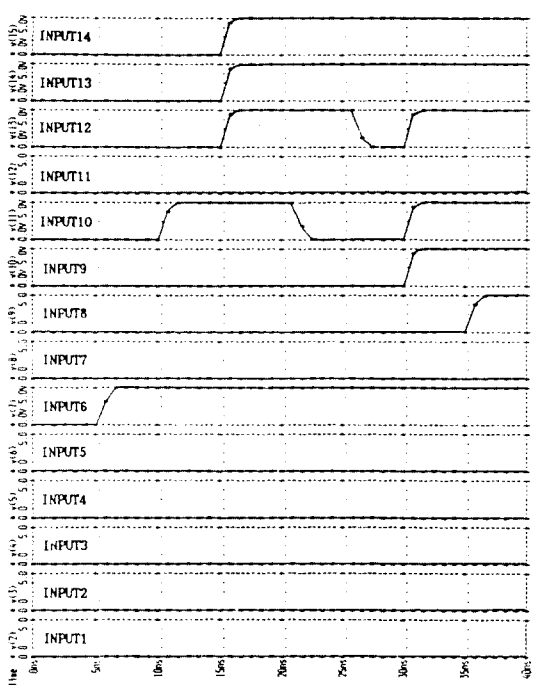


그림 4. PSPICE 시뮬레이션의 입력 패턴
Fig. 4. Input waveform of PSPICE simulation.

의 입력단 동작 주파수에서 동작함을 확인할 수 있었다.

2. 레이아웃

단층 구조 퍼셉트론 방식 2비트 에러 검증 및 수정 회로를 칩(chip)화 하기 위해 설계된 뉴럴 표준 셀을 이용하여 마스크 도면을 만들었다. 이 도면들은 CMOS 이중 배선 2㎛ 설계 규칙에 의해 레이아웃 되었다. 그림 6은 단층 구조 퍼셉트론 형태의 2비트 에러 검증 및 수정 회로의 마스크 도면이다. 설계 규칙 검사와 전기적 연결 검사는 본 연구실에서 개발한 KUIC-DRC (Kyungpook national University Intelligent Cad-Desing Rule Checker)와 KUIC ERC (KUIC Electrical Rule Checker)를 이용하여 수행하였다. 레이아웃 에디터로는 KUIC LED (KUIC- Layout EDitor)를 사용하였다.

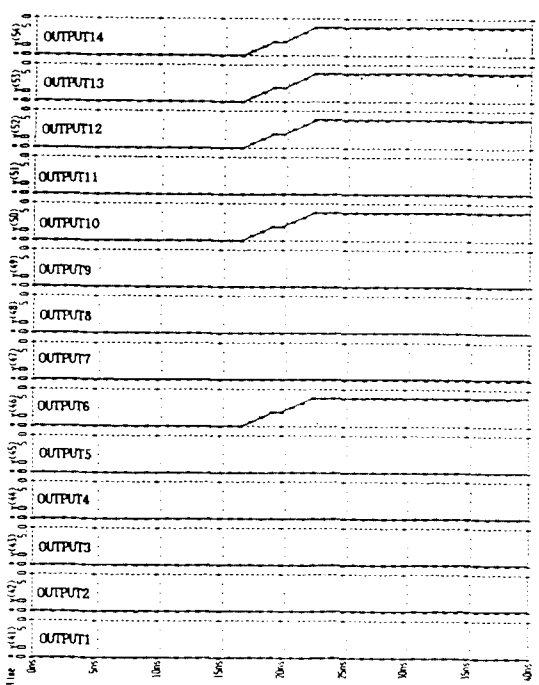


그림 5. PSPICE 시뮬레이션의 출력 파형
Fig. 5. Output waveform of PSPICE simulation.

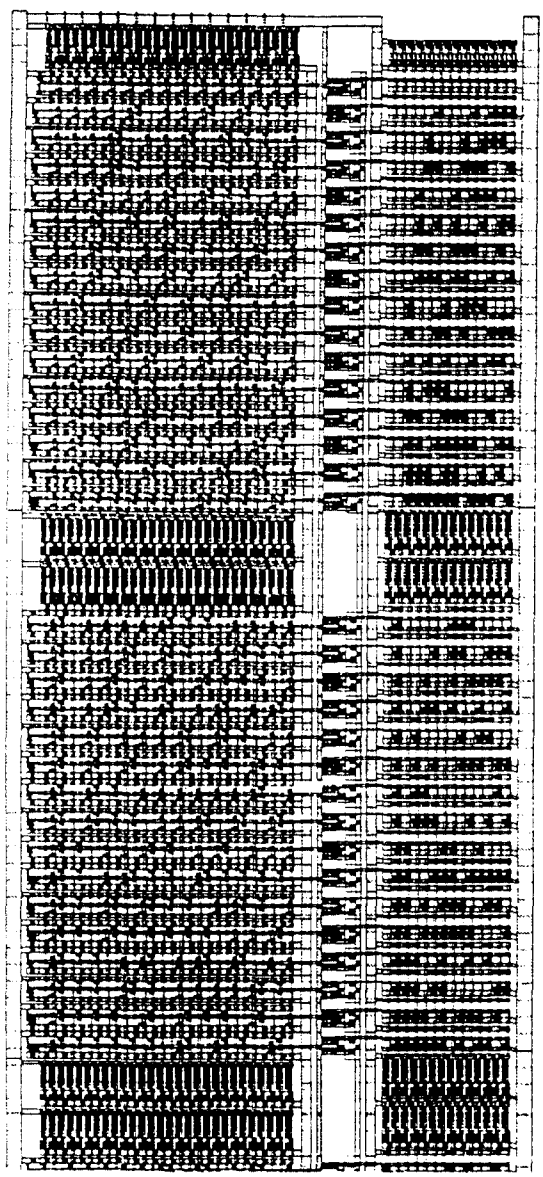


그림 6. 2비트 에러 검증 및 수정 회로의 마스크 도면
Fig. 6. Layout of 2 bit ECC circuit.

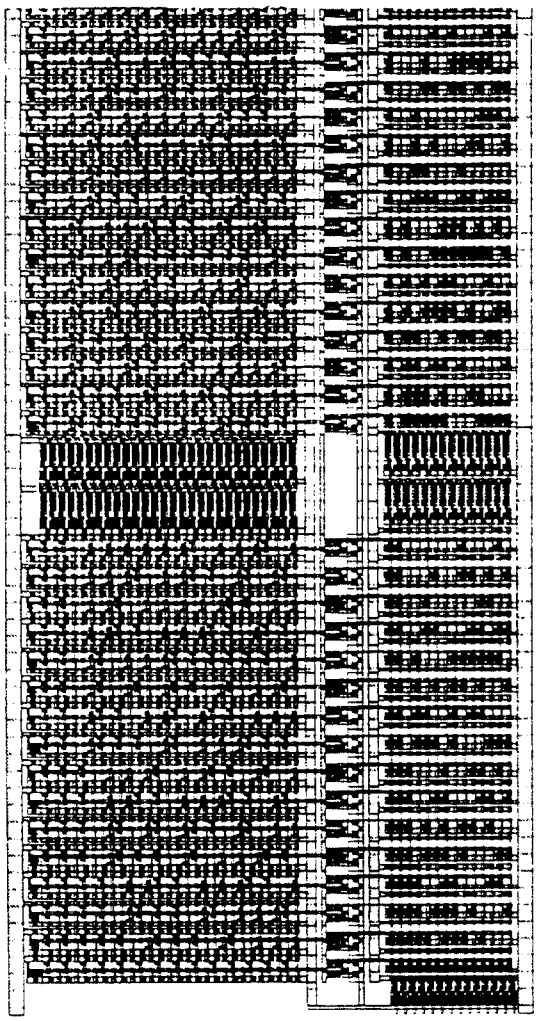


그림 7. 2비트 에러 검증 및 수정 회로의 마스크 도면(계속)
Fig. 7. Layout of 2-bit ECC circuit. (continue)

Ⅳ. 결 론

단층 구조 퍼셉트론(single layer perceptron) 방식 신경 회로망을 사용하여 2비트의 에러를 검증하고 수정하는 (error checking and correction) 회로를 설계하였다. 2비트 에러 검증 및 수정 회로의 설계에는 데이타 비트가 6비트이고 체크 비트가 8비트인 (14, 6) 부호를 적용하였다. 회로를 시뮬레이션한 결과 2비트 에러 검증 및 수정 회로는 최대 67MHz의 입력단 동작 주파수를 가짐을 확인할 수 있었다. 이 회로는 이미 설계된 뉴럴 표준 셀을 이용하여 설계되었고, 마스크 도면은 CMOS 이중 배선 2μm 설계 규칙에 의해 KUIC-LED로 레이아웃 하였다. 이 논문에서 제안된 신경 회로망 개념을 이용한 에러 검증 및 수정 회로는 개념상으로는 아주 단순하기 때문에 데이타 비트의 확장이 쉬울 것이다. 하지만 회로의 기하학적인 크기는 코드 길이가 커짐에 따라 PMOS 및 NMOS의 갯수와 채널 넓이가 늘어나므로 코드길이에 의존적이다. 이것은 PMOS 및 NMOS를 구동시키는 인버터의 팬아웃 문제이기도 하므로 코드의 길이는 한정적이다. 이러한 문제는 인버터 대신에 차동 증폭기를 사용함으로써 해결할 수 있을 것이다.

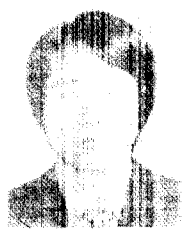
參 考 文 獻

1. W.W. Peterson and E.J. Weldon, *Error Correcting Codes*, 2d ed. The M.I.T. Press, Cambridge, Mass., 1972.
2. T.R.N. Rao and E. Fujiwara, *Error Control Coding for Computer Systems*, Englewood Cliffs, New Jersey, Prentice Hall, 1989.
3. R. Rosenblatt, *Principles of Neurodynamics : Perceptrons and the Theory of Brain Mechanisms*, Washington, DC : Spartan Books, 1962.
4. 이만영, 부호이론, 희중당, 1985.
5. D.W. Tank and J.J. Hopfield, "Simple 'Neural' optimization networks : An A / D converter, signal decision circuit, and a linear programming circuit", *IEEE Transaction and Circuit and Systems*, pp. 533~541, vol. CAS-33, No. 5, May. 1986.
6. J.J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities", *Proceeding of the National Academy of Science. U.S.A.*, vol. 79, pp. 2,554~2,558, April 1982.
7. 김태경, "신경 회로망의 VLSI 구현", 경북대학교 전자공학과 석사학위 논문, 1989. 2.
8. Y. Takefuji, P. Hollis, Y.P. Foo and Y.B. Cho, "Error

correcting system based on neural circuits", *IEEE Proceedings of the First Annual International Conference on Neural Networks*, vol. 3, pp. 293~300, June 1987

9. 남호원, 고희진, 권건대, 정호선, 이우일, "신경 회로망을 이용한 에러 수정 회로", 대한전자공학회 하계종합학술대회 논문집, pp. 574~577. 1989.
10. C. Mead and L. Conway, *Introduction to VLSI System*. Addison Wesley, 1980.
11. N.H.E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*. Addison Wesley, 1985.
12. 서인환, "집적회로의 계층적 설계 규칙 검증 시스템 : KUIC-DRC", 경북대학교 석사학위논문, 1989. 2.
13. 김홍락, "CMOS 레이아웃 도면으로부터 회로 추출 및 검증", 경북대학교 석사학위 논문, 1989. 12.
14. 권건대, 정호선, 이우일, "신경회로망을 이용한 2비트 ECC 회로 설계", 전자계산, 반도체, 재료 및 부품, 씨에이디 합동 학술발표회 논문집. pp. 9~100. 1990.
15. 이원, "개선된 레이아웃 에디터 KUIC-LED III의 개발", 경북대학교 석사학위 논문, 1989. 12.

權 健 兌(Geoun Tae KWON) 正會員
1964年 9月 1日生
1988年 2月 : 慶北大學校 電子工學科 卒業
1990年 8月 : 慶北大學校 産業大學院 電子材料專攻 卒業(工學碩士)
1990年 9月~現在 : 現代電子産業(株) 半導體研究所 SRAM 設計室 勤務



鄭 鎬 宣(Ho Sun CHUNG) 正會員
1943年 1月29日生
1969年 2月 : 仁荷大學校 電氣工學科 卒業
1975年 2月 : 서울大學校 大學院 電子工學科 碩士學位 取得
1980年10月 : 프랑스 ENSEEIHT 電子工學科 博士學位 取得.
1976年 5月~現在 : 慶北大學校 工科大學 電子工學科 副教授.
※ 主關心分野는 CAD 시스템開發, 神經回路網의 VLSI 구현 및 神經컴퓨터 開發 等임.