Regular paper
Journal of the KIECS. pp. 721-728, vol. 13, no. 4, Aug. 31 2018, t. 90, pISSN 1975-8170 | eISSN 2288-2189
http://dx.doi.org/10.13067/JKIECS.2018.13.4.721

선형 블록 오류정정코드의 구조와 원리에 대한 연구

문현찬* · 갈홍주** · 이원영***

Study on Structure and Principle of Linear Block Error Correction Code

Hyun-Chan Moon* · Hong-Ju Kal** · Won-Young Lee***

요 약

본 논문은 다양한 구조의 선형 블록 오류정정코드를 소개하고, 이를 회로로 구현하여 비교 분석한 결과를 보여주고 있다. 메모리 시스템에서는 잡음 전력으로 인한 비트 오류를 방지하기 위해 ECC(: Error Correction Code)가 사용되어 왔다. ECC의 종류에는 SEC-DED(: Single Error Correction Double Error Detection)와 SEC-DED-DAEC(: Double Adjacent Error Correction)가 있다. SEC-DED인 Hsiao 코드와 SEC-DED-DAEC인 Dutta, Pedro 코드를 각각 Verilog HDL을 이용해 설계 후 0.35μm CMOS 공정을 사용해 회로로 합성하였다. 시뮬레이션에 의하면 SEC-DED회로는 인접한 두 개의 비트 오류를 정정하지 못하지만 적은 회로 사용 면적과 빠른 지연 시간의 장점이 있으며, SEC-DED-DAEC 회로의 경우 Pedro 코드와 Dutta 코드 간에는 면적, 지연 시간의 차이가 없으므로 오류 정정률이 개선된 Pedro 코드를 사용하는 것이 더 효율적임을 알 수 있다.

ABSTRACT

This paper introduces various linear block error correction code and compares performances of the correction circuits. As the risk of errors due to power noise has increased, ECC(: Error Correction Code) has been introduced to prevent the bit error. There are two representatives of ECC structures which are SEC-DED(: Single Error Correction Double Error Detection) and SEC-DED-DAEC(: Double Adjacent Error Correction). According to simulation results, the SEC-DED circuit has advantages of small area and short delay time compared to SEC-DED-DAEC circuits. In case of SED-DED-DAEC, there is no big difference between Dutta's and Pedro's from performance point of view. Therefore, Pedro's code is more efficient than Dutta' code since the correction rate of Pedro's code is higher than that of Dutta's code.

키워드

Double Adjacent Error, ECC, Memory Soft Error
이중 근접 오류, 오류 정정 부호, 메모리 소프트 오류

Ⅰ. 서 론

DRAM(: Dynamic Random Access Memory)은 트랜지스터와 커패시터를 이용한 구조로 이뤄져 있으며 커패시터에 전하를 채워 비트 0과 1을 저장하는 메모리이다. 성능을 증가시키기 위해 DRAM의 구조가 점점 축소되었고 각각의 칩마다 많은 메모리셀이 집적되면서 용량이 처음 대비 수배로 늘어났다. 결과적으로

*,** 서울과학기술대학교 전자IT미디어공학과
(mhcqwe92@naver.com*, qawsed9342@gmail.com**)
*** 교신저자 : 서울과학기술대학교 전자IT미디어공학과
· 접 수 일 : 2018. 06. 14
· 수정완료일 : 2018. 07. 15
· 게재확정일 : 2018. 08. 15
· Received : Jun. 14, 2018, Revised : Jul. 15, 2018, Accepted : Aug. 15, 2018
· Corresponding Author : Won-Young Lee
Dept. Electronic and IT Media Engineering, Seoul National University of Science and Technology,
Email : wylee@seoultech.ac.kr

메모리의 용량은 늘어났지만 전력소모의 문제에 부딪혀 저전력 IC소자가 필수적으로 사용되었다. 최근 DRAM의 전압은 2.5V에서부터 1.2V까지 점차 낮아졌으며, 이로 인해 SNR(: Signal to Noise Ratio)의 감소하여 저장된 데이터에 오류 발생 확률이 증가한다.

메모리에서 발생하는 오류는 크게 하드 오류(hard error)와 소프트 오류(soft error)로 분류 할 수 있다. 메모리 공정과정에서 물리적 결함이 발생하거나 메모리 내부에 직접적인 이상이 생긴 경우 즉, 메모리 셀이 영구적으로 손상되는 상황을 하드 오류라고 하며 이 경우에는 데이터 값이 0이나 1로 고정되기도 한다. 따라서, 하드 오류는 새로운 메모리로 교체함으로써 해결할 수 있다.

반면 소프트 오류는 메모리 내부 전기적 신호의 간섭 혹은 비트 셀이 작아짐으로써 커진 알파입자 혹은 우주선(ray)등의 외부적 요인으로 그림 1과 같이 비트 셀의 데이터가 일시적으로 뒤집어 질 때 발생하는 오류이다. 소프트 오류는 일시적이므로 데이터를 다시 쓰고 읽는 과정 등으로 정정이 가능하지만 개인이 사용하는 용도와 달리 대용량 서버와 같이 수많은 데이터가 처리되는 시스템에서 오류가 발생하면 일시적 오류일지라도 시스템에서 충돌이 발생하거나 프로그램이 예상과 다른 값을 내보내 치명적인 피해와 손실을 야기 할 수 있다. 이러한 소프트 오류 문제를 해결하기 위한 방법으로써 데이터 통신에서 사용하고 있는 다양한 오류 비트 정정 방식을 사용할 수 있다 [1-6].

본 논문에서는 대표적 선형 블록 코드인 Hamming 코드와 Hsiao SEC-DED 코드를 2.1장에서 설명하고 2.2장에서 기존 SEC-DED-DAEC 회로의 구조와 최적화 코드인 Pedro 코드를 소개한다. 3장에서는 앞에서 확인해본 코드들을 Verilog HDL을 이용해 직접 구현하고 회로를 synthesis tool을 통해 합성한 후, 면적과 지연 시간을 비교한다.

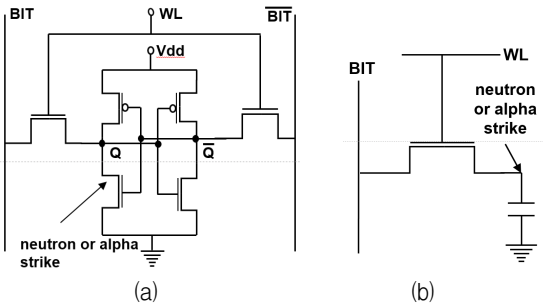

그림 1. (a)SRAM에서 Soft Error 발생과 (b)DRAM에서의 Soft Error 발생
Fig. 1 (a)Soft error on SRAM and (b)soft error on DRAM

II. 본 론(제목 수정)

2.1 Hamming Code and SEC-DED

SEC-DED(: Single Error Correction Double Error Detection)는 1비트 오류 정정과 2비트 오류 검출을 할 수 있는 선형 블록 오류정정 코드 이다. 이에 대해서 살펴보려면 서론에서 언급했던 Hamming 코드를 살펴볼 필요가 있다. Hamming 코드는 1950년 R. W. Hamming에 의해서 발명된 오류 검출 및 정정 코드이며 이는 다음과 같은 원리로 구성된다. 최종 전송할 부호화된 비트가 n개라고 할 때, 전송할 정보 비트(k)에 대한 각각 벡터 값에 n비트 부호 벡터를 대응시켜 전송하는 방식이다. 이러한 과정을 거치면 k개의 정보 비트로부터 r개의 패리티 비트가 생성되며 총 $n=k+r$개의 비트가 부호화 되어 전송 된다[7]. Hamming 코드에서 패리티 비트의 개수는 표 1에서 정보비트 개수에 따라 확인 할 수 있다.

부호 벡터 n=7, 정보 비트 k=4, 패리티 비트 r=3인 (7, 4) Hamming 코드를 예로 들어 설명할 때 정보 비트 k의 벡터가 $k=[k_1, k_2, k_3, k_4]$이고 이에 따른 부호 벡터 $n=[n_1, n_2, \ldots, n_7]$일 때 각 벡터 값을 연산하여 식 (1)과 같이 패리티 비트(p)가 생성된다.

$$n=[n_1,\ldots,n_7]=[k_1, k_2, k_3, k_4, p_1,p_2,p_3] \quad (1)$$

표 1. Hamming 코드의 비트 구성
Table 1. Bit configuration of Hamming code

Encoded bit (n)	Information bit (k)	Parity bit (r)
1	0	1
2	0	2
3	1	2
4	1	3
5	2	3
6	3	3
7	4	3
8	4	4
9	5	4
10	6	4
11	7	4
12	8	4
13	9	4
14	10	4
15	11	4
16	11	5

여기서 p는 $p_1 = k_1 \oplus k_2 \oplus k_3$, $p_2 = k_2 \oplus k_3 \oplus k_4$, $p_3 = k_1 \oplus k_2 \oplus k_3$와 같이 정보 비트의 XOR연산을 통해 생성된다.

이러한 과정은 행렬을 이용해 수식화 될 수 있다. 정보 비트의 조합에 따라 패리티 비트를 생성하는 행렬을 G(generator)-행렬이라 하고 G-행렬은 앞선 n 비트의 벡터와 비교했을 때 식 (2)와 같이 $G = [I_k \mid P]$(k×n행렬)로 표현 된다(I_k는 k 비트에 따른 단위행렬).

$$G = [I_k \mid P] = \begin{bmatrix} 1\,0\,0\,0 \mid 1\,0\,1 \\ 0\,1\,0\,0 \mid 1\,1\,1 \\ 0\,0\,1\,0 \mid 1\,1\,0 \\ 0\,0\,0\,1 \mid 0\,1\,1 \end{bmatrix} \quad (2)$$

k개의 정보 비트는 $[k] \times [I_k \mid P] = (1 \times k * k \times n)$ 행렬 연산을 통해 오류 검출 및 정정이 가능한 부호화된 데이터를 생성한다.

n 비트로 부호화된 데이터는 복호화과정을 통해 원래 데이터로 복원된다. 패리티 비트를 제거 하고 비트의 오류를 판별하는 과정으로 이어지기 때문에 부호화과정 보다 복잡하게 이뤄진다. 복호화에 사용되는 행렬은 H-행렬로 정의 된다. H-행렬은 식 (3)과 같이 $H = [P^T \mid I_r]\,(n-k=r)$로 표현 된다($P^T$는 G-행렬의 P부분을 전치(transpose)한 행렬).

$$H = [P^T \mid I_r] = \begin{bmatrix} 1\,1\,1\,0 \mid 1\,0\,0 \\ 0\,1\,1\,1 \mid 0\,1\,0 \\ 1\,1\,0\,1 \mid 0\,0\,1 \end{bmatrix} \quad (3)$$

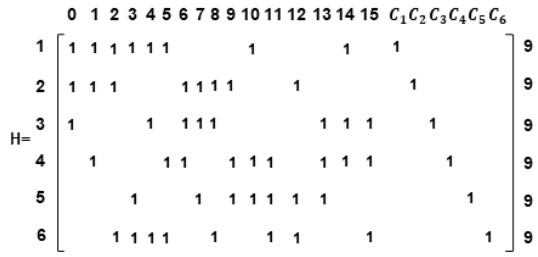

그림 2. SEC-DED을 위한 (22, 16) H-행렬
Fig. 2 (22, 16) H-matrix for SEC-DED

복호화과정은 아래와 같은 순서로 이루어진다.

1) H-행렬$[r \times n]$은 n 비트의 부호화된 데이터와 (H-행렬 × $[n \times 1]$)연산을 함으로써 r' bit를 생성한다. 이는 n 비트 부호화 데이터의 k 비트에서 다시 패리티 비트 r'을 생성하는 것이며 기존의 패리티 r 비트와 생성된 r' 비트를 비교 연산하여 동일한 지 검사한다. 비교 연산 후 생성 되는 것을 신드롬(Syndrome) 이라고 하며 오류의 위치를 찾는데 사용된다.

2) 신드롬 복호기를 구성한다. 만약 1개의 오류가 발생하면 신드롬 복호기에서 나온 값은 오류가 발생한 비트의 위치를 알려준다.

Hamming 코드 이후 연구된 SEC-DED 데이터 부호화와 복호화 과정은 거의 같은 원리로 이어져 왔기 때문에 데이터 비트의 조합에 따라 서로 다른 H-행렬이 구현 가능하다. 실제로 ECC에는 한정된 H-행렬가 아닌 다양한 H-행렬이 사용된다. H-행렬의 구조를 변화시키며 ECC 구조의 최적화와 오류 정정률을 개선시켰다. 이중 대표적인 코드 중 하나가 Hsiao 코드이다[8]. Hsiao가 H-행렬의 구조를 새롭게 하면서 이후 SEC-DED 연구가 본격화되었다. Hsiao의 SEC-DED는 Hamming 코드를 최적화한 구조이며, Hamming 코드와 다르게 몇 가지 규칙이 있다.

1) H-행렬의 각 열은 모두 0인 열은 없다.

2) H-행렬의 모든 열은 달라야한다.

3) 각 열은 홀수개의 1을 가진다.

4) H-행렬의 행의 1 개수는 같아야한다.

그림 2의 Hsiao SEC-DED의 (22, 16) H-행렬은 위의 규칙을 따라 구성되어 있다. 16개의 정보 비트와 6

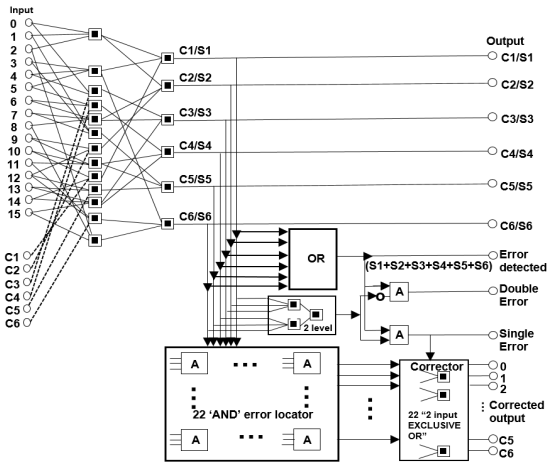

그림 3. Hsiao SEC-DED 부호기와 복호기
Fig. 3 Hsiao SEC-DED encoder and decoder

개의 패리티 비트를 생성하는 행렬이고 1의 개수에 따라 구현에 필요한 논리 게이트의 수가 결정된다.

복호화 과정 또한 Hsiao SEC-DED와 Hamming 코드는 차이점을 갖는다. 패리티를 검사하는 과정에서 생성된 신드롬은 오류가 발생하면 H-행렬의 열과 같은 값으로 나오게 된다. 그렇기 때문에 신드롬 복호기 또한 H-행렬의 열의 1과 0 값에 따라 논리 게이트를 이용하여 복호기를 구성할 수 있다.

그림 3은 Hsiao (22, 16) 코드에 대한 부호기와 복호기의 구조를 보여주고 있다. 복호 과정을 3-input AND 게이트만 사용하여 구현한 것을 알 수 있는데 이는 H행렬 열의 1의 개수가 3개로 모두 같기 때문이다. 따라서, 열에서 0의 값을 갖는 부분에 대해 NOT게이트 연산이 필요하지 않고 3-input AND게이트만을 이용해 복호화가 가능함을 알 수 있다. Hsiao (39, 32) 코드 또한 3개의 가중치를 가지므로 3-input AND게이트만으로 복호기의 구현이 가능하다. 식 (3)과 그림 2를 통해 확인한 Hamming코드와 Hsiao코드의 H-행렬 중 I-행렬 부분을 제외한 행 부분이 G-행렬의 열과 같은 것을 알 수 있다. 때문에 G-행렬을 따로 사용하지 않고 H-행렬만으로 부호화와 복호화를 할 수 있음을 알 수 있다.

여기까지 SEC-DED의 원리와 부호기와 복호기의 구조를 확인하였다. SEC-DED이후 다수의 비트 오류를 해결하는 방법에 대한 필요성이 대두되었다. 물론 2비트 오류를 정정하는 BCH(: Bose Chaudhuri Hocquenghem) 코드가 존재 하지만 BCH 코드의 경우 알고리즘이 복잡하여 SEC-DED에 비해 칩 사용 면적이 클 뿐 아니라 처리 시간 또한 긴 단점이 있다. 따라서, 2비트 오류를 정정 하는 것에 초점을 맞추기 보단 인접한 오류를 해결 할 수 있는 DAEC(: Double Adjacent Error Correction) 코드 연구가 진행 되었다.

2.2 SEC-DED-DAEC

Dutta 코드는 대표적인 SEC-DED-DAEC로서 Hsiao H-행렬의 구조를 바꿔 인접한 데이터의 오류를 정정하는 방법이기 때문에 추가적인 패리티 비트가 필요하지 않다[9]. 그러므로 이는 기존 Hsiao H-행렬의 구성 시 필요한 몇 가지 규칙과 더불어 추가적인 조건을 만족해야 한다. DAEC 코드의 신드롬 검사용 H-행렬은 기존 신드롬 검사용 H-행렬과 2 비트 오류 정정을 위한 추가적인 행렬로 구성된다. 새로운 신드롬 검사 행렬은 1의 개수가 짝수 개인 조건을 만족해야한다. 만약 단일 비트 오류가 발생하면 생성된 신드롬이 H-행렬의 한 열과 일치되어 오류가 정정되고 인접한 두 비트에 오류가 발생하면 이때 생성된 신드롬은 새로운 행렬의 한 열과 일치되어 신드롬 복호화를 통해 오류가 정정 된다.

1) H-행렬의 각 열은 모두 0인 열은 없다.
2) H-행렬의 모든 열은 달라야한다.
3) 각 열은 홀수개의 1을 가진다.
4) SEC-DED가 가능해야한다.
5) 각각의 인접한 열 XOR연산 값은 모두 다르고 0 이 아니다.
6) XOR 연산을 통한 인접 신드롬 행렬은 짝수 가중치를 갖는다.

Dutta 코드는 위와 같은 조건을 만족해야 하며, 이를 통해 구성된 Dutta 코드의 (16, 22) H-행렬 예시는 식 (4)와 같다.

$$H=\begin{bmatrix}1100110001110100100000\\0001010011011001010000\\1010100111100011001000\\1001001110010110000100\\0110101100001101000010\\0111011000101 0\end{bmatrix} \quad (4)$$

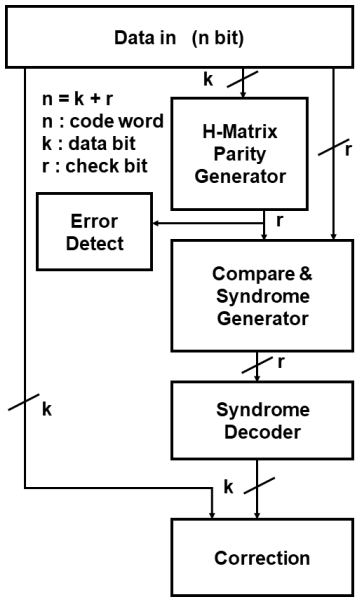

그림 4. ECC 회로의 구조
Fig. 4 Block diagram of an ECC circuit

SEC-DED-DAEC 부호기와 복호기를 설계 하는 방법은 행렬을 바탕으로 연산회로를 구성하는 것이며, 이는 SEC-DED와 동일하다. 그림 4는 ECC의 전체구조를 보여주고 있다. 구조는 대부분 동일하지만 DAEC의 경우 신드롬 복호기 부분에서 SEC-DED의 신드롬 복호기보다 구성이 확연하게 복잡해진다. 이는 H-행렬의 인접한 열끼리의 연산을 통한 복호 기능이 추가되었기 때문이다. 다시 말해 기존 SEC-DED에 DAEC 기능을 추가하기 위해서는 각각의 열을 XOR 연산을 하는 과정이 새롭게 추가되어야 한다. 연산 과정은 1열⊕2열, 2열⊕3열, 3열⊕4열.....16열⊕17열 총 [6 X 16] 행렬을 생성하는 것이다. 이 방법을 1열과 3열을 예시로 살펴보자. 먼저 H-행렬 1열 $C1$은 단 하나의 열인 2열 $C2$와 인접하므로 식 (5)와 같이 $C1$과 $C2$를 XOR연산한다.

$$C1 \oplus C2 = [101100] \oplus [100011] = [001111] \quad (5)$$

이 값이 새로운 인접 신드롬 검사 행렬의 첫 번째 열에 해당하는 값이다. 다음으로 3열 $C3$은 $C2$와 4열 $C4$ 총 두 개의 열과 인접한다. 같은 방법으로 각각 XOR연산하면 식 (6)과 같다.

$$\begin{aligned} C2 \oplus C3 &= [100011] \oplus [001011] = [101000] \\ C3 \oplus C4 &= [001011] \oplus [010101] = [011110] \end{aligned} \quad (6)$$

이러한 연산 방식을 통해 식 (7)과 같이 새로운 [6 X 16] 인접 신드롬 검사 행렬을 생성할 수 있다.

$$[6\times 16]\,Dutta\ Adjacent\ Syndrome\ Check\ Matrix = \begin{bmatrix} 0101010010011101 \\ 0011110101101011 \\ 1111101000100101 \\ 1011010010111010 \\ 1011110100010111 \\ 1001101001111110 \end{bmatrix} \quad (7)$$

Hsiao SEC-DED 복호기와 Dutta SEC-DED-DAEC 복호기를 비교했을 때 SEC-DED-DAEC는 인접한 비트의 오류를 정정하는 장점이 있는 반면 더 많은 XOR가 필요한 단점이 있다. 이러한 단점을 보완하기 위해서 DAEC 코드의 최적화 연구가 진행되었다. 이 중 대표적인 Pedro SEC-DED-DAEC 코드에 대해 확인하고 직접 설계 후 Dutta 코드와 비교했다.

Pedro는 SEC-DED-DAEC의 복호기 회로를 최적화 하였다[10]. 이는 새로운 열의 구조 또는 완전히 다른 방식으로 복호화하는 것이 아닌 비교적 간단한 방법으로 이루어져있다. 식 (4)의 H-행렬의 XOR연산을 통한 식 (7)의 인접 신드롬 검사 행렬이 생성 될 때 열의 1의 개수가 일정하지 않기 때문에 복호기 구현 시 부가적인 소자가 필요하다는 문제가 있다. 이를 해결하기 위해서 H행렬과 동일하게 식 (7) 행렬의 1의 수 또한 일정하게 맞춤으로써 추가 논리 게이트 수를 줄이는 방법을 고안하였다. H-행렬의 경우는 1열당 1의 개수가 최소 3개, 인접 신드롬 검사 행렬의 경우는 1열당 1의 개수가 최소 4개를 만족하도록 열을 배치하면 논리 게이트 수를 줄일 수 있다. 기존 16비트 DAEC 코드의 경우 6개의 패리티 비트를 가지므로 각각 ${}_6C_3$ 과 ${}_6C_4$ 안에서 서로 다른 열을 만족시키는 배열이 필요하다. H-행렬의 경우 ${}_6C_3$ = 20개 중 서로 다른 16개의 열을 각각 인접한 열의 연산으로 1의 개수가 4개가 되도록 배열하면 되지만, 인접 신드롬 검사 행렬의 경우는 ${}_6C_4$ = 15이므로 1의 수가 4개인 서로 다른 16열을 생성하지 못함을 알 수 있다. 따라서, Pedro 코드는 16비트와 64비트의 경우에서 기존 SEC-DED-DAEC의 패리티보다 1비트 많은 패

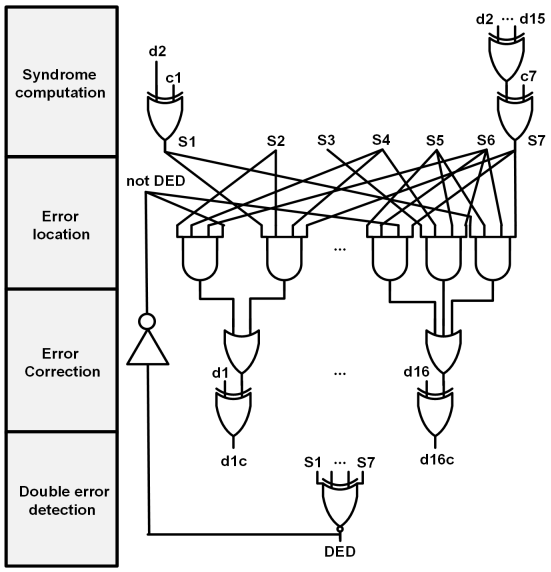

그림 5. Pedro SEC-DED-DAEC 부호기와 복호기
Fig. 5 Pedro SEC-DED-DAEC encoder and decoder

리티 비트가 필요하다. 그러나 32비트 에서는 추가 패리티 없이 동일하게 최적화를 할 수 있다. 즉, 16비트의 경우 7비트 패리티, 64비트의 경우 9비트 패리티, 32비트는 동일하게 7비트 패리티가 생성된다.

식 (8)은 7개의 패리티를 생성하는 최적화 된 Pedro (23, 16) H-행렬이다.

$$H=\begin{bmatrix}0100000000000000001000000\\1001010010100000000100000\\0010101000011010001000 0\\1011000101010110000010 00\\0000101110110101000010 0\\1110011 01\end{bmatrix} \quad (8)$$

식 (8)은 H-행렬의 1열 당 1의 최소 개수 3개를 만족하며 이를 통해 생성되는 인접 신드롬 검사 행렬의 각 열의 1개수가 모두 4개가 된다. 이 결과 디코더를 구현하기 위해 NOT게이트를 추가로 사용하지 않고 AND게이트만을 이용한다. 다만, 그림 5에서 확인 할 수 있듯이 not DED라는 신호가 필요한데 이는 Double Error Detect 신호에 NOT을 취한 것 이다. 2비트 오류가 발생하면 not DED는 항상 0의 값을 가지게 되므로 한 개의 오류를 정정하는 복호화를 하지 않고 인접한 열의 신드롬을 확인하여 인접 비트 오류를 정정한다. 반대로 1비트 오류의 경우에는 신드롬과 H-행렬에 해당하는 홀수개의 1을 맞춰 주는 역할을 한다. 하지만 not DED는 신드롬을 모두 XNOR하여 그 결과를 반전 시킨 후 다시 피드백하는 구조이므로 무시할 수 없는 신호 전달 지연이 발생하고 면적에도 적지 않은 영향을 미칠 수 있다.

표 2. Hsiao SEC-DEC 부호기와 복호기의 회로 면적
Table 2. Circuit areas of Hsiao SEC-DEC encoder and decoder.

Data bit	Circuit Area (μm^2)
16 bit	105.039
32 bit	212.846
64 bit	422.274

표 3. SEC-DEC-DAEC 회로간의 회로 면적 비교
Table 3. Comparison on circuit areas of SEC-DEC-DAEC circuits

Data bit	Circuit Area (μm^2)		
	Dutta code	Pedro code	Pedro/Dutta(%)
16 bit	232.949	224.608	96.42 (-3.58)
32 bit	460.641	440.243	95.97 (-4.43)
64 bit	864.342	871.377	100.81 (+0.81)

III. 실험 결과

앞에서 살펴본 Hsiao 코드와 Dutta와 Pedro의 SEC-DED-DAEC 코드를 Verilog HDL을 이용해 설계 하고, 이를 Synopsys design tool과 CMOS 350nm 공정을 사용하여 회로 합성하였다. 합성 결과를 토대로 각 코드 별 면적과 신호 지연 시간을 시뮬레이션 함으로써 최적화 된 코드를 비교 분석 하였다.

표 2는 Hsiao SEC-DED 부호기와 복호기를 CMOS 350nm 공정으로 설계 및 합성한 결과이다. 데이터의 길이가 2배씩 증가할 때 마다 필요한 면적 또한 약 2배씩 증가하는 것을 알 수 있다. 16비트의 데이터에 대해 Hsiao SEC-DED 부호기와 복호기를 구현하는 경우 105.0396 μm^2 의 면적이 사용된 반면, 64비트의 데이터의 경우는 422.274 μm^2 의 면적이 사용되었다.

표 4. SEC-DEC-DAEC 회로간의 신호전달 시간 비교
Table 4. Comparison on propagation delay of SEC-DEC-DAEC circuits

Data bit	Propagation time (ns)		
	Dutta code	Pedro code	Pedro/Dutta(%)
16 bit	4.08	4.11	100.74 (+0.74)
32 bit	5.12	5.12	100.00 (+0)
64 bit	6.15	6.15	100.00 (+0)

표 3은 Dutta와 Pedro의 SEC-DED-DAEC 부호기와 복호기를 CMOS 350nm 공정으로 설계 및 합성한 결과를 보여주고 있다. Hsiao SEC-DED회로와 비교했을 때 두 종류의 회로 모두 인접한 두 개의 비트 오류를 정정하는 기능이 추가됨에 따라 Hsiao SEC-DEC 회로 대비 사용 면적이 약 2배 증가했음을 알 수 있다. 이를 통해 인접한 두 개의 비트 오류를 정정하기 위해서는 기존 대비 2배의 회로 면적이 요구됨을 알 수 있다. Dutta와 Pedro SEC-DED-DAEC 회로간의 사용 면적을 비교하면 Pedro 회로를 사용하는 경우 16비트와 32비트에서 약 3.58~4.43 % 이득을 보지만, 64비트의 경우는 사용 면적의 차이가 없음을 알 수 있다. Pedro 코드는 16비트와 64비트의 경우 Dutta 코드 대비 패리티 비트가 1비트 추가 된다. 16비트 같이 적은 비트 연산에서는 패리티 비트가 추가된 영향이 미비하여 면적 이득이 나타나는데 반해 64비트 이상 즉, 처리 비트 수가 증가할수록 패리티 비트 연산에 필요한 로직이 복잡해져 면적 이득이 없어지는 것을 알 수 있다.

그러나 Pedro 코드는 Dutta 코드 대비 오류 정정률이 개선된 코드이며, 표 4에서 볼 수 있듯이 복호기 회로의 신호 전달 시간 또한 서로 유사 수준이다. 따라서 16비트에서부터 64비트의 입력에 대해서 회로 사용 면적이 거의 같은 경우라면 오류 정정률 측면에서 충분히 사용할만한 가치가 있음을 알 수 있다.

Ⅳ. 결 론

본 논문에서는 메모리에서 발생하는 오류를 정정하고 탐지하는 선형 블록 오류정정코드 중 하나인 1 비트 오류 정정과 2 비트 오류 탐지가 가능한SEC-DED, 그리고 2 비트 오류 정정까지 가능한 SEC-DED-DAEC ECC의 구조와 원리를 설명한다. 전체적으로 살펴 볼 때 Hamming 코드의 최적화라고 할 수 있는 Hsiao SEC-DED는 이 구조를 이용한 Dutta SEC-DED-DAEC와 비교할 때 DAEC는 불가능 하지만 집적도와 지연 시간 측면에서 크게 앞서므로 다중 오류가 없으며 빠른 데이터 처리가 요구되는 메모리에서 필요할 것이다. SEC-DED-DAEC 회로의 경우, Dutta 코드와 Pedro 코드를 비교했을 때 면적과 지연시간이 비등했다. 따라서, Dutta 코드 대비 오류 정정률이 높은 Pedro 코드가 효과적인 방안이라고 할 수 있다.

감사의 글

이 연구는 서울과학기술대학교 교내연구비의 지원으로 수행되었습니다. 또한 본 연구는 IDEC에서 EDA Tool를 지원받아 수행하였습니다.

References

[1] E. Jang, “LDPC Coding for image data and FPGA Implementation of LDPC Decoder," *J. of the Korea institute of Electronic Communication Science*, vol. 12, no. 4, 2017, pp. 569-574.

[2] J. Jung, Y. Lee, and H. Shin, “A Study on Forward Error Correction of Long-Distance Submarine Optical Communication Systems," *J. of the Korea institute of Electronic Communication Science*, vol. 3, no. 3, 2008, pp. 170-176.

[3] A. Doniyor and H. Suh, “An Efficient Algorithm for finding Optimal Spans to determine R=1/2 Rate Systematic Convolutional Self-Doubly Orthogonal Codes," *J. of the Korea institute of Electronic Communication Science*, vol. 10, no. 11, 2015, pp. 1239-1244.

[4] H. Kal, H. Moon, and W. Lee, “Design of BCH Code Decoder using Parallel CRC Generation,” *J. of the Korea institute of*

Electronic Communication Science, vol. 13, no. 2, 2018, pp. 333-340.

[5] W. Zhang and H. Suh, "Analysis of Coarse Acquisition Code Generation Algorithm in GPS System," *J. of the Korea institute of Electronic Communication Science*, vol. 12, no. 1, 2017, pp. 61-68.

[6] S. Chen and H. Suh, "An Effective Decoding Algorithm of LDPC Codes with Lowering Error Floors," *J. of the Korea institute of Electronic Communication Science*, vol. 9, no. 10, 2014, pp. 1111-1116.

[7] R. Hamming, "Error Correcting and Error Detecting Codes", *The Bell Sys. Tech. Journal,* vol. 29, no. 2, Apr. 1950, pp. 147-160.

[8] M. Hsiao, "A Class of Optimal Minimum Odd-weightcolumn SEC-DED codes", *IBM Journal of Research and Development,* vol. 14, no. 4, July 1970, pp. 395-401.

[9] A. Dutta and N. Touba, "Multiple bit upset tolerant memory using a selective cycle avoidance based SEC-DED-DAEC code," In *Proc. 25th IEEE VLSI Test Symp.*, Berkeley, USA, 2007.

[10] P. Reviriego, J. Martínez, and J. Maestro, "A method to design SEC-DED-DAEC codes with optimized decoding," *IEEE Transactions on Device and Materials Reliability*, vol. 14, no. 3, Sept. 2014, pp. 884-889.

저자 소개

문현찬(Hyun-Chan Moon)

2012년~현재 서울과학기술대학교 전자IT미디어공학과 학사과정

※ 관심분야 : Error correction code, Hardware networking

갈홍주(Hong-Ju Kal)

2018년 서울과학기술대학교 전자IT미디어공학과 졸업(공학사)

2018년~현재 연세대학원 전기전자공학과 재학

※ 관심분야 : 컴퓨터 구조, 메모리 구조

이원영(Won-Young Lee)

2006년 KAIST 전기 및 전자공학과 졸업(공학사)

2008년 KAIST 대학원 전기 및 전자공학과 졸업(공학석사)

2012년 KAIST 대학원 전기 및 전자공학과 졸업(공학박사)

2012~2015년 삼성전자 메모리사업부 책임연구원

2015년~현재 서울과학기술대학교 전자IT미디어공학과 조교수

※ 관심분야 : VLSI, High-speed Serial Interface