

효율적인 오류검출 방식의 낸드 플래시 컨트롤러

백청택* · 이용환*

*금오공과대학교

A NAND Flash Controller with Efficient Error Detection Unit

Chung-taek Baik* · Yong-hwan Lee*

*Kumoh National Institute of Technology

E-mail : ctbaik@kumoh.ac.kr

요 약

낸드 플래시 메모리는 최근 많은 디지털 기기에서 사용되고 있으며 그 용량과 성능면에서의 발전이 급격이 이루어지고 있다. 낸드 플래시 메모리는 읽고 쓰기 회수에 제한이 있어 이 수명이 다하면 데이터의 신뢰성을 보장하기 어렵다. 이 때문에 낸드 플래시 데이터의 오류를 검출하는 ECC(Error Correction Code) 알고리즘의 적용이 필수적이다. 기존에는 ECC 알고리즘을 논리 게이트로 구현하였으나 본 논문에서는 룩업 테이블 방식을 사용하여 신뢰성과 데이터 처리 시간을 향상시키고자 한다.

ABSTRACT

Recently, Nand flash memory is widely used for digital equipments and its capacity and performance are rapidly improving. The limit on the number of writings and readings to/from Nand flash memory does not guarantee the integrity of its data. Therefore, ECC algorithm should be applied to the Nand flash controller. To reduce the access time, we use the look-up table to implement the ECC algorithm instead of the conventional logic gates.

키워드

Mobile, NAND Flash controller, SRAM interface

1. 서 론

최근 낸드 플래시 메모리는 휴대폰, 캠코더, 아이팟, 디지털카메라, USB 플래시드라이브 등에 폭 넓게 사용이 확대되면서 수요가 급증하고 있다. 이러한 원인은 모바일 기기의 요건인 크기가 작고 가벼우며 외부충격에도 강하고 가격이 싼 조건들을 낸드 플래시 메모리가 모두 만족하기 때문이다. 반도체산업협회(SIA)의 분석에 따르면 낸드 플래시 메모리 판매는 계속성장하며 2009년에는 350달러까지 증가할 것으로 보고 있다.

또한 사용자의 데이터 저장 요구량의 증가로 메모리 용량 확대에 대한 연구가 활발히 진행 중이다. 2002년 국제반도체회로 학술회의(ISSCC)에서 삼성전자의 황창규 사장이 메모리 신성장론을 발표하였다. 그 내용은 반도체의 집적도가 2배로 증가하는 시간이 1년으로 단축되었으며 무어의 법칙을 뛰어 넘고 있다는 것이다. 2006년에는

DVD급 영화 20편을 저장 할 수 있는 기술 공정 40 나노의 32 기가바이트 낸드 플래시 메모리를 개발하였다. 따라서 본 논문에서는 대용량의 낸드 플래시 메모리의 신뢰성과 데이터 처리 속도 향상을 위한 낸드 플래시 컨트롤러의 ECC(Error Correcting Code)를 설계한다.

논문의 구성은 다음과 같다. 2장에서는 낸드 플래시 메모리의 특성에 대해 알아보고 3장에서는 낸드 플래시 메모리 컨트롤러 구조에 관해 설명한다. 4장에서는 해밍코드 알고리즘을 5장에서는 룩업 테이블 구조의 ECC에 대해 알아본다. 마지막으로 성능 분석 및 결론으로 구성된다.

II. 낸드 플래시 메모리 특성

낸드 플래시 메모리는 전원이 꺼져도 기록된 내용이 지워지지 않는 비휘발성 저장장치이며 제

조 단가가 싸고 대용량의 저장이 가능하여 휴대형 전자기기의 저장매체로 주로 사용된다.

반면 쓰기 횟수가 100,000회로 제한되어 있으며 제자리 덮어쓰기가 불가능하며, 하나의 블록 내에서 데이터를 순차적으로만 써야하는 제약이 있고, 런타임 배드블록(Run-Time Bad Block)이 발생할 경우 이를 처리해 주어야하며, 지우는 단위와 프로그램의 단위가 다르다는 특성이 있다. 이러한 낸드 플래시 메모리를 사용하기 위해서는 호스트 컨트롤러(Flash Translation Layer)가 필요하며 이 컨트롤러는 각각의 플래시 메모리 소자 특성에 따라 다르다.

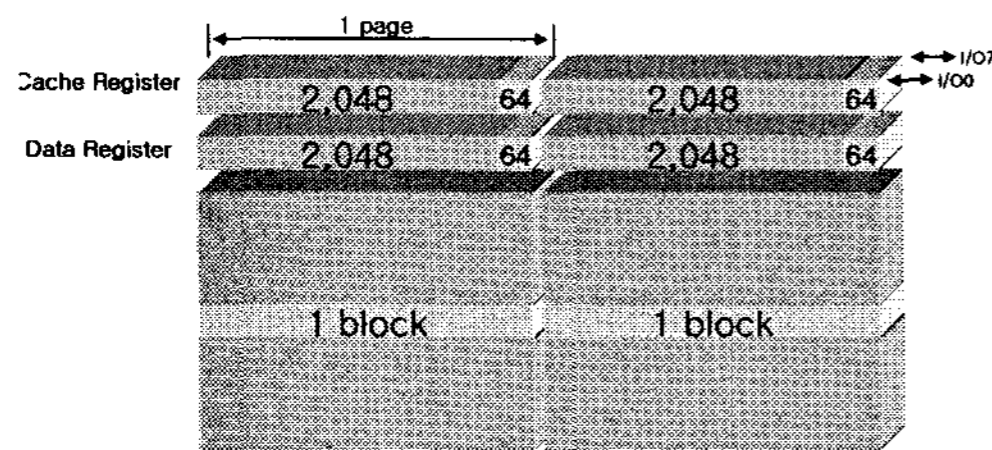


그림 4. 낸드 플래시 메모리 구조

그림 1. 은 낸드 플래시 메모리의 구조를 나타낸다. 낸드 플래시 메모리의 한 페이지는 2,112 바이트이며 한 블록은 64 페이지로 구성된다. 또한 읽기와 쓰기 명령을 받으면 캐시 레지스터와 데이터 레지스터를 통해서 페이지 단위로 실행하며 지우기 명령은 블록단위로 실행된다. 쓰기 명령을 받으면 우선 블록의 데이터를 모두 지운 후에 쓰기 연산을 실행하므로 다소 시간이 걸린다.

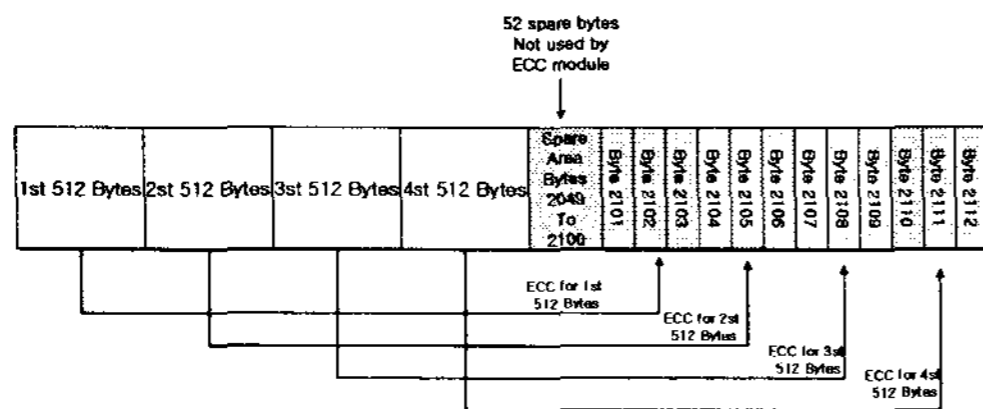


그림 5. 낸드 플래시 메모리의 한 페이지 구조

2,048 바이트의 한 페이지 데이터 영역은 네 개의 512 바이트의 블록으로 나뉜다. 각각의 블록 데이터는 오류 검출을 위한 정보로 인코딩되어 낸드 플래시의 2,101 - 2,112 영역의 3 바이트에 저장된다. 그리고 나머지 52 바이트 영역(Spare Area)은 컨트롤러 영역으로 사용한다.

III. 낸드 플래시 메모리 컨트롤러 구조

본 논문에서 제공하는 낸드 플래시 컨트롤러는

호스트 디바이스와 인터페이스에서 일반 RAM 처럼 간단한 메모리 맵핑이 가능하도록 지원한다 [2]. 그림 3. 은 낸드 플래시 컨트롤러 전체 블록도를 나타낸다[1]. 내부 구조는 데이터 패스 모듈과 에러 정정 모듈이 통합된 형태로 구성된다. 낸드 플래시 메모리에 데이터를 쓰고 읽을 때 페이지 단위로 동작하기 때문에 임시로 데이터를 저장하는 버퍼가 필요하다. 데이터 패스 모듈에는 2,112 바이트 버퍼가 2개가 있어서 호스트 시스템과 플래시 컨트롤러가 동시에 버퍼를 사용할 수 있다.

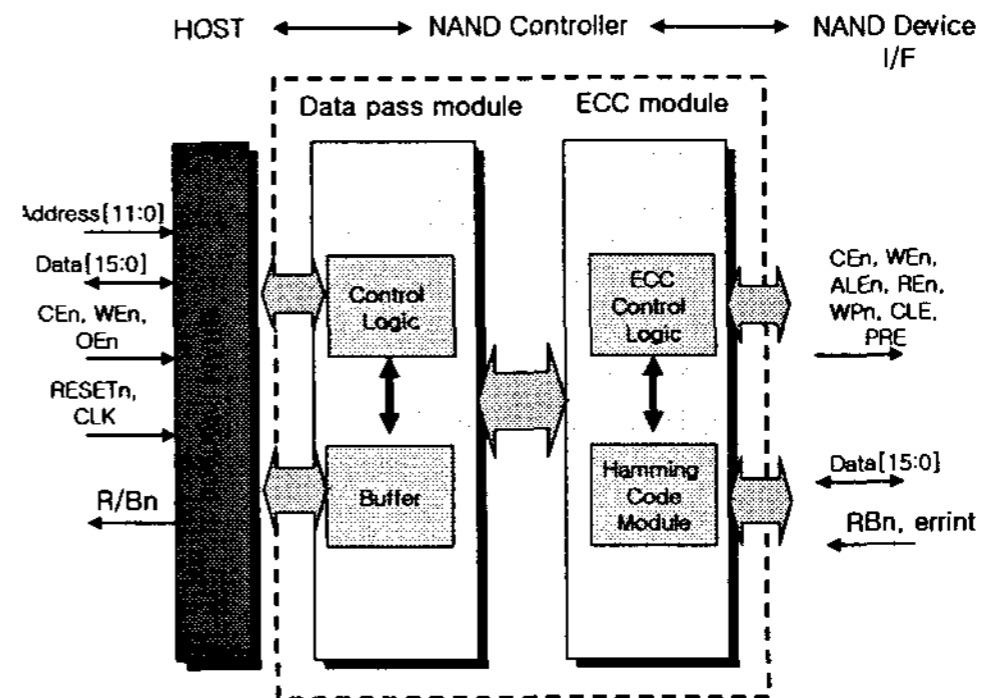


그림 6. 낸드 플래시 컨트롤러 블록도

호스트에서 데이터를 쓸 때 컨트롤러 내부에 있는 버퍼에 페이지 단위의 데이터를 쓴다. 컨트롤러는 낸드 플래시 메모리의 쓸 주소에 있는 데이터를 지우거나 다른 페이지로 옮기고 메모리에 쓸 유효한 데이터가 있는 버퍼를 액세스한다. 버퍼의 데이터는 ECC 모듈을 통하여 에러 검출을 위해 인코딩 정보를 생성한 후 데이터와 함께 낸드 플래시 메모리에 페이지 단위로 저장된다.

반대로 데이터를 읽을 때는 낸드 플래시 내부의 데이터와 인코딩된 정보를 컨트롤러로 읽어온다. 데이터는 ECC 모듈에서 패리티 비트를 생성하며 그 패리티 정보는 기존에 쓰기 실행에서 인코딩된 정보와 비교하여 에러를 검출한다. 그래서 에러가 있으면 정정하거나 데이터를 다시 요청하며 에러가 없으면 호스트가 사용하지 않는 버퍼에 데이터 값을 저장한 후 SRAM 인터페이스 컨트롤러 신호에 따라 출력하게 된다.

IV. 낸드 플래시 메모리를 위한 ECC 해밍 코드 알고리즘

낸드 플래시 메모리에서 한 페이지의 내부 512 바이트의 블록에 적용되는 해밍코드 정보는 24 비트(3바이트)에 저장된다. 그림 4. 는 블록 패리티 비트 테이블을 보여준다.

	7	6	5	4	3	2	1	0			
0	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	P8'	P 16'	P 32'
1	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	P8	P 16'	
2	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	P8'	P 16'	
3	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	P8	P 16'	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
510	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	P8'	P 16'	P 32'
511	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	P8	P 16'	
	P1	P1'	P1	P1'	P1	P1'	P1	P1'			
	P2		P2'		P2		P2'				
	P4				P4'						

그림 7. 블록 페리티 비트 테이블

P1은 512 바이트에서 모든 1, 3, 5, 7 비트에 대한 Column 페리티 비트이고 P1'은 0, 2, 4, 6 비트에 대한 Column 페리티 비트를 나타낸다.

$$\begin{aligned}
 p1' &= \text{bit6} \text{ xor } \text{bit4} \text{ xor } \text{bit2} \text{ xor } \text{bit0} \\
 p1 &= \text{bit7} \text{ xor } \text{bit5} \text{ xor } \text{bit3} \text{ xor } \text{bit1} \\
 p2' &= \text{bit5} \text{ xor } \text{bit4} \text{ xor } \text{bit1} \text{ xor } \text{bit0} \\
 p2 &= \text{bit7} \text{ xor } \text{bit6} \text{ xor } \text{bit3} \text{ xor } \text{bit2}
 \end{aligned}$$

P8'은 512 바이트에서 모든 짝수 열의 XOR 연산을 통하여 계산된다. Row 페리티 비트 (rowparity) 계산은 다음과 같다[2].

$$\begin{aligned}
 \text{rowparity} &= \text{bit7} \text{ xor } \text{bit6} \text{ xor } \text{bit5} \text{ xor } \text{bit4} \\
 &\quad \text{xor } \text{bit3} \text{ xor } \text{bit2} \text{ xor } \text{bit1} \text{ xor } \text{bit0} \\
 P8' &= P8' \text{ xor } (\text{rowparity} \& \text{!(rownumber[0])})
 \end{aligned}$$



그림 8. ECC 인코딩 레지스터

그림 5. 는 각각의 인코딩 정보에 대한 페리티 비트를 계산하기 위한 24비트 레지스터이다.

V. 록업 테이블 ECC 설계

그림 6. 은 ECC 블록 다이어그램을 나타낸다 [3]. Encoding data generator 단계에서는 메모리의 데이터를 이용해서 해밍 코드로 인코딩 또는 인코딩된 데이터를 메모리에 쓴다. Check generator 단계에서는 이전에 메모리에 저장된 인코딩된 데이터와 현재 Encoding data generator 에서 생성된 인코딩 데이터를 비교한다. Error locator 단계에서는 에러의 상태 또는 위치를 확인한다. 마지막으로 Error corrector 에서 에러를 수정하여 출력한다.

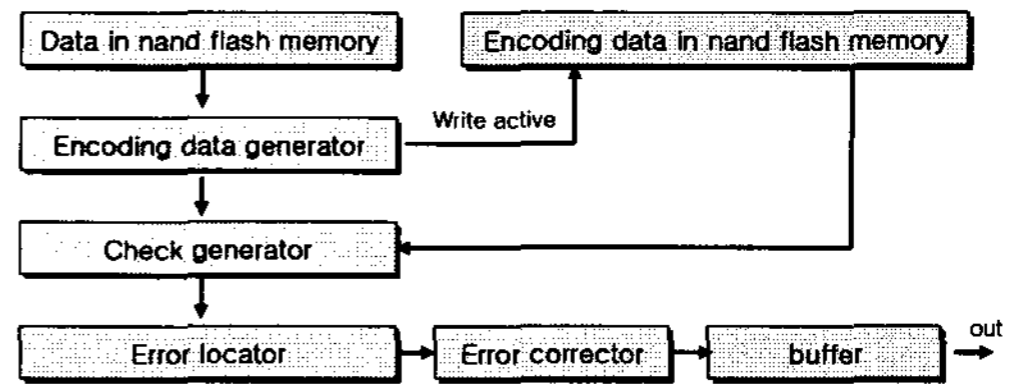


그림 9. ECC 블록 다이어그램

기존의 encoding data generator와 check generator 는 XOR 트리 구조로 구현되었다. XOR 트리 구조는 지연에 민감한 특성을 가진다. 그래서 지연을 줄이기 위해서 XOR 트리의 수를 효과적으로 줄이는 방법들이 제시되었다[4][5].

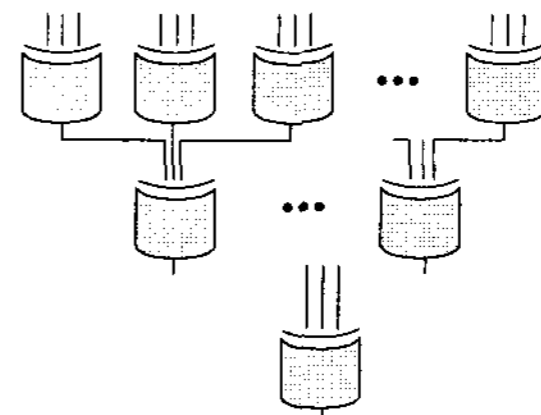


그림 10. 입력 3개의 XOR 연산 구조

본 논문에서는 XOR 트리보다 지연을 줄이기 위한 ROM 기반의 록업 테이블을 이용한 encoding data generator 블록을 설계하였다. 그림 8. 은 낸드 플래시 메모리의 Column(p1, p2, p3, p4) 페리티 비트 생성을 위한 블록도이다.

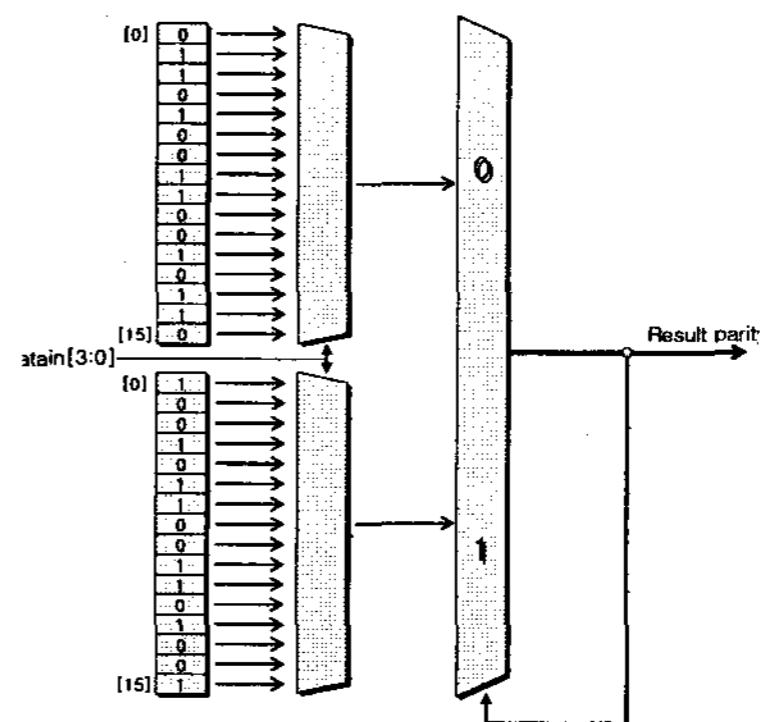


그림 11. 록업 테이블을 이용한 Column encoding Generator 블록

결과 페리티(Result parity)를 생성할 때 XOR 트리를 사용하지 않고 Datin의 4비트를 사용하여 록업 테이블 데이터를 출력하였다. 록업 테이블은 16비트 두 개로 구성되며 서로 반전된 형태이다. 그래서 결과 페리티 값에 따라 록업 테이블이 선택된다. 또한 Row의 록업 테이블은 8비트에 대한 256 비트의 데이터를 선택적으로 사용하여

출력하였다. 룩업 테이블을 사용하여 설계한 Encoding Data Generator 블록은 Verilog HDL로 기술되었으며 Modelsim을 이용하여 기능 검증을 하고 Synopsys 와 0.35um CMOS 표준 셀 라이브러리를 이용하여 합성하였다. XOR 트리 구조로 설계된 것과 룩업 테이블로 설계한 블록의 셀 면적과 동작 속도를 비교하였다. 그 결과는 아래 표 1. 과 같다.

표 1. XOR 트리 구조와 LUT 합성결과 비교

	XOR	룩업 테이블	성능
셀 면적	1033.61	1744.53	
동작속도	(7.6ns) data arrival time(4.6)	(5.6ns) data arrival time(4.64)	
합성조건	Perating Condition Name : V300WTP0850 Process :1.30 Temperature : 85.00 Voltage : 3.00		

[4]M.Y.Hsiao,"A class of optimal minimum odd-weight-column SEC_DED codes,"IBM j.Res. Develop., vol.14, pp. 395-401, July 1970

[5]J.A.Fifield, and C.H.Stapper,"High-speed on-chip ECC for synergistic fault-tolerant memory chips,"IEEE Journal of Solid-State Circuits, vol.26,no.10,pp. 1449-1452,Oct.1991

※ 2007년도 IT-SoC 핵심설계인력양성 사업의 SoC 전공실습프로젝트 지원에 의한 연구결과의 일부임.

VI. 결 론

본 논문에서 휴대형, 디지털카메라, USB 플래시 드라이브 등과 같은 대용량의 메모리를 요구하는 모바일 기기의 특성을 고려하여 동작 속도 증가를 위한 낸드 플래시 컨트롤러 ECC 모듈을 룩업 테이블로 설계하였다. 그리고 기존의 XOR 트리 구조와 지연시간, 면적을 비교한 결과 지연시간은 작아지고 면적은 증가하였다. 따라서 대용량의 낸드 플래시 메모리의 고속처리를 위한 낸드 플래시 메모리 컨트롤러로 사용이 적합하다.

참고문헌

[1] 백청택, 이용환, "휴대형 장치를 위한 낸드 플래시 컨트롤러", 한국정보기술학회, pp. 667-670

[2]www.micron.com/products/nand/technotes "Micron NAND Flash Controller via Xilinx Spartan-3 FPGA"

[3]W.Gao, S.Simmons, "A Study on the VLSI Implementation of ECC for Embedded DRAM",IEEE 2003,pp. 203-206