



**(19) 대한민국특허청(KR)**  
**(12) 등록특허공보(B1)**

(45) 공고일자 2016년12월07일  
 (11) 등록번호 10-1684045  
 (24) 등록일자 2016년12월01일

(51) 국제특허분류(Int. Cl.)  
 G11C 29/42 (2015.01) G06F 11/10 (2006.01)  
 G11C 29/04 (2006.01) G11C 29/44 (2006.01)  
 (21) 출원번호 10-2014-7030518  
 (22) 출원일자(국제) 2012년05월31일  
 심사청구일자 2014년10월30일  
 (85) 번역문제출일자 2014년10월30일  
 (65) 공개번호 10-2014-0140632  
 (43) 공개일자 2014년12월09일  
 (86) 국제출원번호 PCT/US2012/040108  
 (87) 국제공개번호 WO 2013/180714  
 국제공개일자 2013년12월05일  
 (56) 선행기술조사문헌  
 US06584595 B2\*  
 US20100325522 A1\*  
 \*는 심사관에 의하여 인용된 문헌

(73) 특허권자  
**휴렛 팩커드 엔터프라이즈 디벨롭먼트 엘피**  
 미국 텍사스주 (우편번호: 77070) 휴스턴 콤팩트  
 센터 드라이브 웨스트 11445  
 (72) 발명자  
**우디피 아니루드하 나겐드란**  
 미국 텍사스주 78745 오스틴 아파트먼트 324 브로  
 디 레인 6607  
**무랄리마노하르 나빈**  
 미국 캘리포니아주 94304-1100 팔로 알토 엠에스  
 1183 빌딩 3 페이지 밀 로드 1501  
 (뒷면에 계속)  
 (74) 대리인  
**제일특허법인**

전체 청구항 수 : 총 15 항

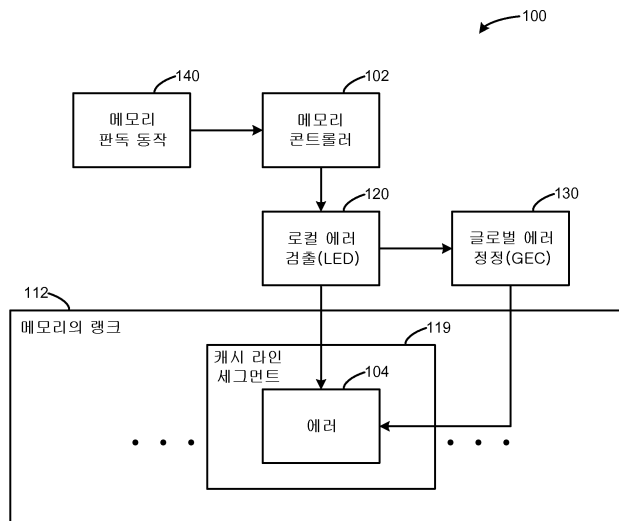
심사관 : 조성찬

(54) 발명의 명칭 로컬 에러 검출 및 글로벌 에러 정정

**(57) 요약**

시스템은 데이터 충실도를 체크하고 에러를 정정하기 위해 로컬 에러 검출(LED) 및 글로벌 에러 정정(GEC) 정보를 사용할 수 있다. LED는 메모리의 랭크와 연관된 데이터의 캐시 라인 세그먼트마다 계산될 수 있다. 데이터 충실도는 랭크의 캐시 라인 세그먼트 사이의 에러의 존재 및 에러의 위치를 식별하기 위해, LED 정보에 기초하여, 메모리 판독 동작에 응답하여 체크될 수 있다. 에러를 갖는 캐시 라인 세그먼트는 에러의 식별에 응답하여 GEC에 기초하여 정정될 수 있다.

**대표도 - 도1**



(72) 발명자

**조우피 노먼 폴**

미국 캘리포니아주 94304-1100 팔로 알토 페이지  
밀 로드 1501

**데이비스 앨런 린**

미국 유타주 84112 솔트 레이크 시티 알엠 3190 사  
우스 센트럴 캠퍼스 드라이브 50

**발라수브라마니안 라지브**

미국 유타주 84070 샌디 오헨리 로드 11147

---

## 명세서

### 청구범위

#### 청구항 1

에러 검출 코드에 기초하여, 메모리의 랭크와 연관된 데이터의 캐시 라인 세그먼트마다 로컬 에러 검출(LED) 정보를 컴퓨팅하는 단계와,

에러 정정 코드에 기초하여 상기 캐시 라인 세그먼트에 대한 글로벌 에러 정정(GEC)을 생성하는 단계와,

상기 랭크의 캐시 라인 세그먼트들 사이의 에러의 존재 및 에러의 위치를 식별하기 위해, 메모리 판독 동작에 응답하여 상기 LED 정보에 기초하여 데이터 충실도(data fidelity)를 확인하는 단계와,

상기 에러의 식별에 응답하여, 상기 GEC에 기초하여 상기 에러를 가진 캐시 라인 세그먼트를 정정하는 단계를 포함하는

방법.

#### 청구항 2

제 1 항에 있어서,

함께 송신될 인접한 캐시 라인과 연관된 복수의 GEC 업데이트를 병합(coalescing)하는 단계를 더 포함하는

방법.

#### 청구항 3

제 1 항에 있어서,

상기 에러 검출 코드는 상기 랭크의 캐시 라인 세그먼트 내의 에러의 존재 및 에러의 위치를 식별하는

방법.

#### 청구항 4

제 1 항에 있어서,

상기 데이터와 연관된 캐시 라인 세그먼트에서 캐시 라인 세그먼트에 대해 컴퓨팅된 LED 정보 및 GEC를 저장하는 단계를 더 포함하는

방법.

#### 청구항 5

제 1 항에 있어서,

상기 데이터와 연관된 메모리 로우(row)와는 상이한 메모리 로우 내의 캐시 라인 세그먼트에 대해 컴퓨팅된 LED 정보 및 GEC를 저장하는 단계를 더 포함하는

방법.

**청구항 6**

랭크 내의 N개의 데이터 칩 사이의 칩-입도(a chip-granularity)에서 에러의 위치를 검출하기 위해, 캐시 라인 세그먼트에 걸쳐 컴퓨팅된 에러 검출 코드에 기초하여, 메모리 판독 동작에 응답하여 로컬 에러 검출(LED)을 수행하는 단계와,

상기 에러의 검출에 응답하여 상기 랭크 내의 N개의 데이터 칩 상에서 상기 캐시 라인 세그먼트에 걸쳐 글로벌 에러 정정(GEC)을 수행하는 단계 - 상기 GEC는 GEC 정보를 생성하기 위해 에러 정정 코드에 기초함 - 와,

무에러(error-free) 세그먼트 및 상기 GEC 정보에 기초하여, 상기 에러를 가진 데이터 세그먼트를 재구성하는 단계를 포함하는

방법.

**청구항 7**

제 6 항에 있어서,

인접한 캐시 라인들로의 복수의 GEC 업데이트가 병합되고 함께 송신되는

방법.

**청구항 8**

제 6 항에 있어서,

상기 에러 정정 코드는 N개의 캐시 라인 세그먼트에 기초하는

방법.

**청구항 9**

제 6 항에 있어서,

기록 동작에 응답하여 상기 GEC 정보를 업데이트하는 단계를 더 포함하는

방법.

**청구항 10**

제 6 항에 있어서,

상기 N개의 칩의 각각의 예비할당 영역(a reserved region)에서, 대응하는 캐시 라인의 로우 버퍼 내에 상기 GEC 정보를 저장하는 단계를 더 포함하는

방법.

**청구항 11**

제 6 항에 있어서,

물리적 데이터 맵핑 정책에 기초하여, 상기 랭크의 각각의 칩 상에 데이터 및 대응하는 LED 정보를 저장하는 단계와,

캐시 라인 액세스 요청에 응답하여 상기 데이터 및 LED 정보를 제공하는 단계를 더 포함하는

방법.

**청구항 12**

제 6 항에 있어서,

상기 GEC 정보를 보호하기 위해 계층화된 에러 정정 코드를 생성하는 단계를 더 포함하되,

상기 계층화된 에러 정정 코드는 N번째 칩 상에 저장되고, 복수의 칩으로부터 GEC 정보 세그먼트에 기초하여 상기 GEC 정보를 복구하는 데 사용되는

방법.

**청구항 13**

제 12 항에 있어서,

GEC 단계 동안에, 상기 계층화된 에러 정정 코드에 기초하여 관심 GEC 로우 내의 에러를 검출하는 것에 기초하여, 정정 불가능한 더블-칩 고장을 식별하는 단계를 더 포함하는

방법.

**청구항 14**

메모리 컨트롤러로서,

상기 메모리 컨트롤러는,

관독 동작에 응답하여 메모리의 랭크와 연관된 데이터의 캐시 라인 세그먼트에 대한 로컬 에러 검출(LED) 정보에 기초하여 데이터 충실도를 확인하고,

상기 LED 정보에 따라 상기 랭크의 캐시 라인 세그먼트 사이의 에러의 존재 및 위치를 식별하며,

에러 정정 코드에 기초하여 상기 캐시 라인 세그먼트에 대한 글로벌 에러 정정(GEC)을 생성하고,

상기 에러의 식별에 응답하여, 상기 GEC에 기초하여 상기 에러를 가진 캐시 라인 세그먼트를 정정하는

메모리 컨트롤러.

**청구항 15**

제 14 항에 있어서,

상기 LED 및 GEC 정보는 상기 메모리 컨트롤러와 연관된 펌웨어 정보에 따라 맵핑되는

메모리 컨트롤러.

**발명의 설명**

**배경 기술**

[0001] 메모리 시스템 신뢰성은 현대의 서버 및 블레이드에서 심각하고 증가하는 문제점이다. 현존하는 메모리 보호 메커니즘은, 모든 메모리 액세스 상의 다수의 칩의 활성화, 증가된 액세스 입도(granularity), 및 저장 장치 오버헤드(overhead)의 증가 중 하나 이상을 필요로 한다. 이들은 증가된 동적 랜덤 액세스 메모리(DRAM) 액세스 시간, 감소된 시스템 성능, 및 실질적으로 더 높은 에너지 소비를 유도한다. 현재의 상업적 칩킬(chipkill)-레벨 신뢰성 메커니즘은 리드-솔로몬(Reed-Solomon: RS)-코드, 심벌 기반 코덱 등과 같은 통상의 에러-정정 코드(Error-Correcting Code: ECC)에 기초할 수도 있다. 그러나, 현재의 ECC 코드는 x4 DRAM의 사용을 위해 메모리

리 시스템 디자인을 제한한다. 또한, 주어진 용량에 대해, 좁은 칩(즉, I/O DRAM x4 칩)을 갖는 듀얼 인라인 메모리 모듈(dual in-line memory module: DIMM)은 더 넓은 I/O 칩(즉, x8, x16 또는 x32 칩)을 갖는 칩들보다 더 많은 에너지를 소비한다.

[0002] 이러한 효율적인 칩킬 메커니즘의 이용 불가능성은 이들이 제공하는 장점에 도 불구하고, 넓은 입출력(I/O) DRAM이 채택의 결여의 일 이유이다. 둘째로, 현재 ECC 코드는 코딩 효율을 증가시키기 위해 큰 데이터 워드(word)에 걸쳐 컴퓨팅된다. 이 ECC 코드 핸들링은 모든 메모리 동작에 대해 많은 수의 칩 또는 심지어 랭크(rank)를 활성화하는 큰 액세스 입도 및 증가된 에너지 소비를 야기한다. 면적, 밀도 및 비용 제약은 칩의 랭크 내에 다소의 오버페치(overfetch)를 야기할 수 있지만, 내고장성(fault tolerance)을 제공하기 위해 부가의 비효율성을 부여하는 것이 회피되어야 한다. 핸들링은 뱅크-레벨 및 랭크-레벨 평형화를 잠재적으로 감소시킬 수 있는데, 이는 광학 채널(photonic channel)과 같은 고대역폭 I/O로 데이터를 공급하는 DRAM의 능력을 감소시킨다. 마지막으로, 통상의 ECC 코드는 지연 시간 및 회로 면적의 모두의 견지에서 비효율적인 복소 갈루아 필드 산술(complex Galois field arithmetic)을 이용한다.

**도면의 간단한 설명**

[0003] 도 1은 예에 따른 메모리 컨트롤러를 포함하는 시스템의 블록 다이어그램이다.  
 도 2는 예에 따른 메모리 모듈의 블록 다이어그램이다.  
 도 3은 예에 따른 메모리 모듈 랭크의 블록 다이어그램이다.  
 도 4는 예에 따른 잉여 비트(surplus bit)를 포함하는 캐시 라인의 블록 다이어그램이다.  
 도 5는 예에 따른 데이터 충실도의 체크에 기초하는 흐름도이다.  
 도 6은 예에 따른 에러 검출 및/또는 정정의 수행에 기초하는 흐름도이다.

**발명을 실시하기 위한 구체적인 내용**

[0004] 본 명세서에 설명된 예는 에러 검출과 에러 정정 기능을 분리하는 2-티어 보호 방안을 사용할 수 있다. 체크섬(checksum) 및 패리티(parity)에 기초하는 것들과 같은 코드는 오버헤드를 거의 또는 전혀 갖지 않고 강한 내고장성을 제공하는데 효과적으로 사용될 수 있다. 시스템 펌웨어와 같은 저장 장치가 DRAM 데이터 메모리 내에 몇몇 정정 코드를 저장하도록 메모리 컨트롤러에 지시하는데 사용될 수 있다. 메모리 컨트롤러는 데이터 맵핑, 에러 검출 및 정정을 핸들링하도록 수정될 수 있다. 에러 검출/정정 코드의 신규한 적용 및 메모리의 신규한 물리적 데이터 맵핑은 고성능 메모리 모듈(예를 들어, ECC DIMM x4, x8, x16, x32 등)이 페치폭 및/또는 저장장치 오버헤드를 증가시키지 않고 칩킬 가능성을 제공하는 것을 가능하게 할 수 있다. 또한, 로컬성(locality) 및 DRAM 로우 버퍼(row buffer) 히트 레이트(hit rate)는 데이터 및 ECC 코드를 동일한 로우 버퍼 내에 배치함으로써 더 향상될 수 있다. 따라서, 효과적인 내고장성 메커니즘이 제공되어, 강한 신뢰성 보장을 가능하게 하고, 에너지를 보존하고 성능을 향상시키기 위해 가능한 한 적은 칩을 활성화하고, 회로 복잡성을 감소시키고, x8, x16 또는 x32와 같은 넓은 I/O DRAM 칩과 함께 동작한다.

[0005] 도 1은 예에 따른 메모리 컨트롤러(102)를 포함하는 시스템(100)의 블록 다이어그램이다. 시스템(100)은 메모리 관독 동작(140)에 응답하여, 메모리의 랭크(112)의 캐시 라인 세그먼트(119)의 에러(104)를 검출하고 그리고/또는 정정하기 위해 로컬 에러 검출(120) 및/또는 글로벌 에러 정정(130)을 적용한다.

[0006] 예에서, 시스템(100)은 데이터의 캐시 라인 세그먼트(119)마다 로컬 에러 검출(local error detection: LED)(120) 정보를 컴퓨팅한다. 캐시 라인 세그먼트(119)는 메모리의 랭크(112)와 연관된다. LED(120)는 에러 검출 코드에 기초하여 컴퓨팅된다. 시스템(100)은 글로벌 패리티에 기초하여 캐시 라인 세그먼트를 위한 글로벌 에러 정정(global error correction: GEC)을 생성한다. 시스템(100)은 랭크(112)의 캐시 라인 세그먼트(119) 사이의 에러(104)의 존재 및 에러(104)의 위치를 식별하기 위해 LED(120) 정보에 기초하여, 메모리 관독 동작(140)에 응답하여 데이터 충실도를 체크한다. 시스템(100)은 에러(104)의 식별에 응답하여, GEC에 기초하여, 에러(104)를 갖는 캐시 라인 세그먼트(119)를 정정한다.

[0007] 대안예에서, 시스템(100)은 랭크(112) 내의 N개의 데이터 칩 사이의 칩-입도에서 에러(104)의 위치를 검출하기 위해, 캐시 라인 세그먼트(119)에 걸쳐 컴퓨팅된 체크섬에 기초하여, 메모리 관독 동작(140)에 응답하여 로컬

에러 검출(LED)(120)을 수행한다. 시스템(100)은 에러(104)의 검출에 응답하여 랭크(112) 내의 N개의 데이터 칩 상의 캐시 라인 세그먼트(119)에 걸쳐 글로벌 에러 정정(GEC)(130)을 수행한다. 시스템(100)은 무에러 세그먼트 및 GEC 정보에 기초하여, 글로벌 패리티를 사용하여 GEC(130)를 수행하여 GEC 정보를 생성하고 에러(104)를 갖는 데이터 세그먼트(119)를 재구성한다.

[0008] 따라서, 시스템(100)은 세그먼트(119)로 낮아진 입도의 레벨에서, 2-계층 내고장성 메커니즘을 구축하기 위해 간단한 체크섬 및 패리티 동작을 사용할 수 있다. 제 1 보호 계층은 데이터 충실도를 검증하기 위한 확인(예를 들어, 판독 동작(140) 직후의 확인)인 로컬 에러 검출(LED)(120)이다. LED(120)는 메모리 모듈 내에 복수의 칩을 가로질러 LED 정보(120)를 분배함으로써, 칩-레벨 에러 검출(칩킬, 즉 전체 DRAM 칩의 고장을 견디는 능력)을 제공할 수 있다. 따라서, LED 정보(120)는 각각의 캐시 라인과 전체로서, 뿐만 아니라 모든 캐시 라인 "세그먼트", 즉 랭크 내의 단일 칩 내에 존재하는 라인의 부분과 연관될 수 있다.

[0009] 비교적 짧은 체크섬(1의 여수(complement), 플레처의 합(Fletcher's sum) 등)이 에러 검출 코드로서 사용될 수 있고, 세그먼트에 걸쳐 컴퓨팅되고 데이터에 첨부될 수 있다. 에러 검출 코드는 주기적 중복 검사(cyclic redundancy check: CRC), Bose, Ray-Chaudhuri, 및 Hocquenghem(BCH) 코드 등과 같은 다른 유형의 에러 검출 및/또는 에러 보호 코드에 기초할 수 있다. 이 에러 검출 코드는 다수의 캐시 라인을 위한 이러한 LED 정보를 포함하기 위해 동일한 메모리 로우 또는 상이한 메모리 로우 내에 저장될 수 있다. 계층-1 보호(LED(120))는 에러의 존재를 검출할 뿐만 아니라, 에러의 위치를 핀포인팅(pinpoint)하는데, 즉 에러(104)와 연관된 칩 또는 다른 위치 정보를 로케이팅할 수 있다.

[0010] LED(120)가 에러를 검출하면, 제 2 보호 계층, 글로벌 에러 정정(GEC)(130)이 적용될 수 있다. GEC(130)는 랭크(112) 내의 N개의 데이터 칩 상의 데이터 세그먼트(119)를 가로지르는 XOR-기반 글로벌 패리티와 같은 패리티에 기초할 수 있다. GEC(130)는 또한 CRC, BCH 등과 같은 다른 에러 검출 및/또는 에러 보호 코드에 기초할 수 있다. GEC 결과는 데이터 세그먼트와 동일한 로우에 또는 다수의 데이터 로우를 위한 GEC 정보를 포함하는 개별 로우에 저장될 수 있다. 데이터는 무고장 세그먼트 및 GEC 세그먼트의 판독, 및 위치 정보(예를 들어, LED(120)에 기초하는 고장 칩의 식별)에 기초하여 재구성될 수 있다.

[0011] LED(120) 및 GEC(130)는 단일 캐시 라인 내의 데이터 워드에 걸쳐 컴퓨팅될 수 있다. 따라서, 더티 라인(dirty line)이 프로세서로부터 메모리로 재기록될 때, "기록-전-판독(read-before-write)"을 수행할 필요가 없고, 양 코드는 직접 컴퓨팅되어, 이에 의해 기록 성능에 대한 영향을 회피한다. 더욱이, LED(120) 및/또는 GEC(130)는 에러 정정 코드(ECC) 목적으로 제한된 중복 저장을 제공할 수 있는 기성품 메모리 시스템의 견지에서, 정규 데이터 메모리 내에 저장될 수 있다. 부가의 판독/기록 동작이 프로세서-요청된 판독/기록과 함께 이 정보에 액세스하는데 사용될 수 있다. 각각의 로우 내의 제공된 저장 공간에 LED 정보를 저장하는 것은 데이터 라인과 직렬로 판독 및 기록되는 것을 가능하게 할 수 있다. GEC 정보는 에러가 있는 데이터 판독의 매우 드문 경우에 액세스될 수 있기 때문에 개별 캐시 라인에서 데이터 메모리 내에 저장될 수 있다. 적절한 데이터 맵핑은 데이터와 동일한 로우 버퍼에 이를 로케이팅하여 로컬성 및 히트 레이트를 증가시킬 수 있다.

[0012] 메모리 컨트롤러(102)는 데이터 맵핑, LED(120)/GEC(130) 연산 및 검증을 제공하고, 요구된다면 부가의 판독을 수행할 수도 있는 등이다. 따라서, 시스템(100)은 운영 체제(Operating System: OS) 또는 다른 컴퓨팅 시스템 부품에 통지하고 그리고/또는 수정할 필요 없이, 전체 기능성을 투명하게 제공할 수 있다. LED(120)/GEC(130)를 저장하기 위해 몇몇 데이터 메모리를 분리하는 것은 시스템 펌웨어와 연관된 최소 수정을 통해 핸들링될 수도 있어, 예를 들어 OS 및 애플리케이션 관점으로부터 투명하게 저장된 LED(120)/GEC(130)를 수용하기 위해 이용 가능한 메모리 저장 장치의 보고된 양을 감소시킨다.

[0013] 도 2는 예에 따른 메모리 모듈(210)의 블록 다이어그램이다. 메모리 모듈(210)은 메모리 컨트롤러(202)와 인터페이싱할 수 있다. 메모리 모듈(210)은 듀얼 인라인 메모리 모듈(DIMM)로서 구성된 동적 랜덤 액세스 메모리(SDRAM)와 같은 국제 반도체 표준 협의 기구(Joint Electron Device Engineering Council: JEDEC)-스타일 더블 데이터 레이트(DDR<sub>x</sub>, 여기서 x=1, 2, 3, ...) 메모리 모듈일 수도 있다. 각각의 DIMM은 적어도 하나의 랭크(212)를 포함할 수 있고, 랭크(212)는 복수의 DRAM 칩(216)을 포함할 수 있다. 2개의 랭크(212)가 도시되어 있고, 각각의 랭크(212)는 9개의 칩(216)을 포함한다. 랭크(212)는 다수의 뱅크(214)로 분할될 수 있고, 각각의 뱅크는 랭크(212) 내의 칩(216)을 가로질러 분배된다. 하나의 뱅크(214)가 랭크 내의 칩에 걸쳐 도시되어 있지만, 랭크는 예를 들어 4 내지 16개의 뱅크로 분할될 수도 있다. 칩(216) 내의 각각의 랭크(212)/뱅크(214)의 부분은 세그먼트(219)이다. 메모리 컨트롤러(202)가 캐시 라인을 위한 요청을 발행할 때, 랭크(212) 내의 칩(216)이 활성화되고, 각각의 세그먼트(219)는 요청된 캐시 라인의 부분에 기여한다. 따라서, 캐시 라인은 다

수의 칩(216)을 가로질러 스트라이핑된다.

- [0014] 64 비트의 데이터 버스폭 및 64 바이트의 캐시 라인을 갖는 예에서, 캐시 라인 전송은 8 데이터 전송의 버스트에 기초하여 실현될 수 있다. 칩은  $xN$  부분, 예를 들어,  $x4$ ,  $x8$ ,  $x16$ ,  $x32$  등일 수 있다. 뱅크(214)의 각각의 세그먼트는  $N$ 개의 어레이(218)(4개가 도시되어 있음)로 분할될 수 있다. 각각의 어레이(218)는 그 칩(216)에 대해 데이터 I/O 핀 상의 단일 비트 내지  $N$ -비트 전송에 기여할 수 있다. 따라서, 소정의 DIMM 능력에 대해,  $x8$ ,  $x16$  또는  $x32$ 와 같은 더 넓은 I/O DRAM 부분을 이용하는 것은 소정의 데이터 버스폭을 성취하도록 요구되는 DRAM 칩(216)의 수를 감소시켜, 더 많은 칩(216)에 대한 DIMM 상에 초과 공간을 생성하고, 이에 의해 이용 가능한 독립적인 뱅크(214)의 수를 증가시킨다.
- [0015] 각각의 칩(216)은 데이터(211), LED(220)로부터의 정보, 및 GEC(230)로부터의 정보를 저장하는데 사용될 수 있다. 이에 따라, 각각의 칩(216)은 데이터(211)의 세그먼트(219), LED(220) 및 GEC(230) 정보를 포함할 수 있다. 이는 어떻게 9-칩 ECC 메모리 모듈이 사용되는지에 대조적이고, 여기서 8개의 칩이 데이터를 위해 사용되고 9번째 칩이 ECC 정보를 위해 사용된다. 이에 따라, 본 명세서에 설명된 예시적인 용도는 각각의 칩이 에러를 식별하고 정정하기 위해 데이터(211), LED(220) 및 GEC(230)를 포함할 수 있기 때문에, 강인한 칩킬 보호를 제공한다. 도 2의 예는 액세스 입도, 에너지 소비 및 성능을 향상시키는 9개의  $x8$  칩의 단일의 랭크를 갖는 가능성을 도시하고 있다. 또한, 예는 신뢰성 보장을 상당히 증가시키는(예를 들어 36개 중 1개의 데드 칩의 통상적인 지원에 대조적으로) 9개 중 1개의 데드칩을 취급하는 능력과 같은 매우 높은 비에서 칩킬 보호를 지원할 수 있다.
- [0016] 본 명세서에 설명된 예는 메모리 액세스마다 활성화된 칩의 수의 다수배 감소를 허용할 수 있다. 이는 적어도 그 정도로 오버헤드를 배제함으로써 동적 에너지 소비를 감소시키는 것을 돕고, 미사용 칩이 저전력 모드로 배치되게 함으로써 정적 에너지 소비를 감소시키는 것을 돕는다. 에너지 장점에 추가하여, 액세스 입도를 감소시키는 것은 랭크-레벨 및 뱅크-레벨 평행화를 증가시킨다. 이는 실질적인 성능 이득을 가능하게 한다. 본 명세서에 설명된 예는 DRAM 부분, DIMM 레이아웃, DDR 프로토콜, 버스트 길이 등에 대한 제약을 부여하지 않고,  $x8$ ,  $x16$  또는  $x32$  DRAM에 적용될 수 있어, 서버가 이들 메모리 구성의 장점을 활용하게 한다. 예들은 예시적인 메모리 컨트롤러 및 소정 정도로 메모리 펌웨어가 예들을 위한 지원을 제공하도록 수정될 수 있기 때문에, 시스템 디자인에 대한 비침입적 수정을 갖고 성취될 수 있다. 예들은 16-비트 또는 32-비트 심벌 또는 다른 증가된 복잡성, 지연 시간 및 에너지 소비에 걸쳐 갈루아 필드 산술의 필요 없이 강한 내고장성을 제공하기 위해 체크섬 및 패리티 계산을 이용할 수 있다.
- [0017] 따라서, 본 명세서의 예는 특장하게 설계된 DRAM 칩 마이크로아키텍처, 예를 들어 ECC 정보를 저장하기 위해 프로비저닝된 특정 영역을 갖고 칩킬-레벨 신뢰성이 가능하면 기록 성능에 대해 상당한 페널티를 부여할 것인 로컬화된 데이터 맵핑 아키텍처를 이용하는 DRAM에 대한 요구 없이 이익을 제공한다. 또한, DIMM 및 랭크 편성에 대해 제약을 갖는 통상의 심벌-기반 ECC 코드를 사용할 필요성이 없다. 예들은 컴퓨팅 시스템의 운영 체제, 메모리 관리 유닛, 캐시 등에 투명할 수 있기 때문에, 예들은 컴퓨팅 시스템의 다수의 부품에 대한 수정의 필요 없이 친화적인 구현예이다.
- [0018] 도 3은 예에 따른 메모리 모듈 랭크(312)의 블록 다이어그램이다. 랭크(312)는  $N$ 개의 칩, 예를 들어, 9개의  $x8$  DRAM 칩(3160(칩 0 ... 칩 8) 및 8의 버스트 길이를 포함할 수 있다. 대안예에서,  $N$ 개의 칩의 다른 수/조합이  $xN$ 의 다양한 레벨 및 버스트 길이에서 사용될 수 있다. 데이터(311), LED(320) 및 GEC(330)는 랭크(312)의 칩(316) 전체에 걸쳐 분배될 수 있다.
- [0019] LED(320)는 모든 판독 동작 직후에 데이터 충실도를 확인할 수 있다. 부가적으로, LED(320)는 랭크(312) 내의 칩-입도에서 고장의 위치를 식별할 수 있다. 이러한 칩-레벨 검출(칩킬을 위해 사용 가능함)을 보장하기 위해, LED(320)는 랭크(312) 내의 단일 칩(316) 내에 존재하는 라인의 부분인 모든 캐시 라인 "세그먼트"에서, 전체로서(심벌-기반 ECC 코드에서와 같이) 전체 캐시 라인보다 더 많은 특이성과 연관된 칩 레벨에서 유지될 수 있다. 캐시 라인(A)은 연관된 로컬 에러 검출 코드( $L_{A0}$  내지  $L_{A8}$ )를 갖고, 세그먼트  $A_0$  내지  $A_8$ 로 분할된다.
- [0020] 캐시 라인이 메모리 요청과 같은 데이터 동작과 연관된 64 바이트의 데이터 또는 512 데이터 비트와 연관될 수 있다. 총 512 데이터 비트(하나의 캐시 라인)가 요구되기 때문에, 각각의 칩은 캐시 라인을 향해 57 비트를 제공한다. 8의 버스트 길이를 갖는  $x8$  칩이 57 비트의 데이터(예를 들어, 도 3에서  $A_0$ ) 및 이들 57 비트와 연관된 7 비트의 LED 정보(320)( $L_{A0}$ )로서 해석되는 액세스당 64 비트를 공급한다. 물리적 데이터 맵핑 정책은 이들이 보호하는 LED 비트(320)와 데이터 세그먼트(311)가 동일한 칩(316) 상에 위치되는 것을 보장하는데 사용될



수 있다. 9개의 칩으로 곁해진 57 비트의 데이터는 513 비트이고 단지 512 비트가 캐시 라인을 저장하는데 요구되기 때문에, 하나의 비트의 메모리는 모든 576 비트에 대해 미사용인 채로 남아 있는 것으로 나타난다. 그러나, 이 "잉여 비트"는 그 상세가 도 4를 참조하여 설명되는 제 2 보호 계층(예를 들어, GEC)의 부분으로서 사용된다.

[0021] LED 코드(320)에 기인하는 판독 또는 기록에 어떠한 성능 페널티도 존재하지 않는다. 모든 캐시 라인 액세스는 또한 그 대응 LED 정보를 판독/기록한다. LED(320)는 "내장형"이기 때문에, 즉 정확하게 하나의 캐시 라인에 속하는 비트로부터 구성되기 때문에, 어떠한 기록-전-판독도 요구되지 않는다 - 코드를 구성하는데 사용된 모든 비트는 기록전에 모든 메모리 컨트롤러에 미리 있음. LED(320)를 위한 에러 검출 코드의 선택은 예측된 고장 모드에 의존할 수 있다. 예를 들어, 간단한 1의 여수 가산 체크섬은 가장 통상적인/빈번한 메모리 고장의 모드를 포함하는 예측된 고장 모드의 범위를 위해 사용될 수 있다.

[0022] 계층 2 글로벌 에러 정정 코드라 또한 칭하는 GEC(330)는 일단 LED(320)(계층 1 코드)가 에러를 검출하고 에러의 위치를 지시하면 손실된 데이터의 복구를 보조한다. 계층 2 GEC(330)는 3개의 티어로 구성될 수도 있다. GEC(330) 코드는 57-비트 엔티티일 수 있고, 각각 데이터 영역으로부터 57-비트 필드인 9개의 캐시 라인 세그먼트의 칼럼방향 XOR 패리티로서 제공될 수 있다. 캐시 라인 A에 대해, 예를 들어 그 GEC(330)는 데이터 세그먼트(A<sub>0</sub>, A<sub>1</sub>, ..., A<sub>8</sub>)가 XOR인 패리티(PA)와 같은 패리티일 수 있다. GEC(330) 코드로부터의 데이터 재구성은 에러가 있는 칩(316)이 LED(320)에 의해 플래그될 수 있기 때문에, 비-리소스 집약적 동작일 수 있다(예를 들어, 무에러 세그먼트 및 GEC(330) 코드의 XOR). 부가의 전용 ECC 칩(메모리 모듈 랭크(312) 상에 ECC 칩으로서 일반적으로 사용되는 것은 대신에 데이터 + LED(320)를 저장하는데 사용됨)에 대한 필요성이 존재하지 않기 때문에, GEC 코드는 전용 ECC 칩을 사용하는 것에 대조적으로, 데이터 메모리 자체에 저장될 수 있다. 이용 가능한 메모리는 펌웨어 수정 또는 다른 기술을 거쳐, 운영 체제의 관점으로부터 물리적으로 있는 것보다 작게 나타나게 될 수도 있다(예를 들어, LED(320) 및/또는 GEC(330)를 저장하기 위한 12.5% 오버헤드만큼). 메모리 컨트롤러는 또한 LED(320) 및/또는 GEC(330)를 수용하기 위해 변화를 인식할 수 있고, 이에 따라 데이터를 맵핑할 수 있다(OS, 애플리케이션 등에 투명하게 하도록 LED(320) 및/또는 GEC(330)에 맵핑하는 것과 같은).

[0023] 칩킬을 위해 9개 중 하나의 데드칩(316)의 강한 내고장성을 제공하고, 각각이 액세스시에 터치된 칩(316)의 수를 최소화하기 위해, GEC(330) 코드는 그 대응 캐시 라인과 동일한 랭크에 배치될 수 있다. 랭크(312) 내의 9개의 칩(316)의 각각 내의 특정-예비할당 영역(도 3에 열거 음영된 GEC(330))은 이 목적으로 확보될 수도 있다. 특정-예비할당 영역은 모든 DRAM 페이지(로우) 내의 캐시 라인의 서브세트일 수 있지만, 이는 명료화를 위해 도 3에 별개의 로우의 세트로서 도시되어 있다. 이 코로케이션은 GEC(330) 정보로의 임의의 판독 또는 기록이 실제 데이터 캐시 라인으로의 판독 또는 기록과 함께 행해질 때 로우-버퍼 히트를 생성하도록 보장될 수 있어, 따라서 성능에 대한 임의의 잠재적인 영향을 감소시키는 것을 보장할 수도 있다.

[0024] 도 4는 예에 따른 잉여 비트(436)를 포함하는 캐시 라인(413)의 블록 다이어그램이다. GEC(430) 정보는 도 3에 또한 도시된 캐시 라인(A)으로서 예에 대해, N개의 칩(예를 들어, 칩 0...8)을 가로질러 예비할당 영역에 레이아웃될 수 있다. 캐시 라인(413)은 또한 패리티(432), 계층화된 패리티(434), 및 잉여 비트(436)를 포함할 수 있다.

[0025] 도 3에 도시된 바와 같은 데이터 비트에 유사하게, 57-비트 GEC(430)는 모든 N개(즉, 9개)의 칩(419) 사이에 분배될 수 있다. PA 필드의 제 1 7개의 비트(PA<sub>0-6</sub>)는 제 1 칩(416)(칩 0)에 저장될 수 있고, 다음의 7개의 비트(PA<sub>7-13</sub>)는 제 2 칩(칩 1)에 저장될 수 있는 등이다. 비트 PA<sub>49-56</sub>은 제 8 칩(칩 7) 상에 저장될 수 있다. 마지막 비트(PA<sub>56</sub>)는 잉여 비트(436) 내에서 제 9 칩(칩 8) 상에 저장될 수 있다. 잉여 비트(436)는 캐시 라인을 저장하기 위해 이용 가능한 513 비트(57 비트 x 9 칩) 중 단지 512 비트만을 사용하는 것에 관련하여 전술된 바와 같이, N번째 칩(칩 8)의 데이터 + LED 영역으로부터 차용될 수도 있다.

[0026] 칩(416)의 고장은 또한 그 칩 내에 저장된 GEC(430) 내의 대응 비트의 손실을 야기한다. 따라서, GEC(430) 코드(PA) 자체는 제 3 티어(PP<sub>A</sub>)라 또한 칭하는 부가의 패리티(432)에 의해 보호된다. 도시된 예에서 PP<sub>A</sub>는 7-비트 필드이고, N-1개의 다른 7-비트 필드, PA<sub>0-6</sub>, PA<sub>7-13</sub>, ..., PA<sub>49-55</sub>의 XOR이다. 패리티(432)(PP<sub>A</sub> 필드)는 N번째(제 9) 칩(칩 8) 상에 저장된 것으로 도시되어 있다. 전체 칩(416)이 고장이면, GEC(430)는 먼저 다른 칩으로부터 비오손된 GEC 세그먼트와 조합된 패리티(432)를 사용하여 복구된다. 비오손된 칩(416)은 에러의 위치의 지시를 포함할 수 있는(즉, 고장난 칩을 로케이팅함) LED에 기초하여 결정될 수 있다. 전체 GEC(430)는 이어서

원본 데이터를 재구성하는데 사용된다.

- [0027] 완전히 고장난 칩 에러에 추가하여, 제 2 칩 내에 에러가 존재할 수도 있다. 본 명세서에 설명된 예는 정정이 아니면, 다양한 고장 모델 하에서 이러한 고장의 검출을 가능하게 한다. 제 2 에러가 또한 전체-칩 고장이면, 초기 데이터 판독과 함께 LED에 의해 검출되고 더블칩 고장으로서 플래그될 것이다. 그러나, 제 2 에러가 단지 관심 GEC(430) 로우에서 발생하면, 이는 GEC 페이지 중에 검출될 수 있다.
- [0028] 예시적인 고장 시나리오에서, 제 2 칩이 완전히 고장된 것으로 -  $A_1$  및  $PA_{7-13}$ 이 손실될 것으로 가정한다. 게다가, 제 1 칩의 GEC 영역에 에러가 존재하면, 비트  $PA_{0-6}$  중 하나 이상이 오손되는 가능성이 존재한다.  $PP_A(432)$ 로부터 손실 비트  $PA_{7-13}$  및  $PA_{0-6}$ ,  $PA_{14-20}$ ,  $PA_{21-27}$ , ...,  $PA_{56}$ 의 재구성은 자체로 부정확할 수도 있다. 이 문제 점을 처리하기 위해, 계층화된 패리티(434)가 사용되는데, 예를 들어 잔여 9 비트의 9개의 칩(416)(도 4에 이어-4에 대해, T4로 표기됨)이  $PA_{55}$  및  $PP_A$ 를 통해 GEC 비트  $PA_0$ 를 가로질러 에러 검출 코드를 구성하는데 사용된다. 신뢰성 타겟이 이를 반드시 정정하는 것이 아니라 단지 제 2 에러를 검출하는 것이기 때문에, 어떠한 정확한 에러 로케이션 정보 또한 정정 능력도 이 스테이지에서 요구되지 않는다는 것을 주목하라. 따라서, 코드는 각각의 T4 비트(434)를 형성하기 위해 상이한 칩으로부터 비트의 다양한 치환을 사용하여 구성될 수도 있다. 이는 동일한 칩(416)으로부터 다수의 비트, 및 칩(416)을 가로지르는 상이한 칼럼으로부터의 비트를 포함하여 검출의 확률을 최대화할 수 있다.
- [0029] 다른 예에서, 단일의 캐시 라인(A)을 고려한다. 칩 0 내지 7(보편성의 손실 없이, 예를 들어, N-1개의 칩)은 데이터 영역에 57 비트의 데이터 플러스 7 비트의 LED를, GEC 영역에 7 비트의 GEC(430) 패리티 플러스 1 비트의 T4 정보(계층화된 패리티(434))를 포함할 수 있다. 칩-8(N번째 칩)은 데이터 영역에 56 비트의 데이터 플러스 7 비트의 LED 플러스 1개의 잉여 비트(436)를, GEC 영역에 8 비트의 패리티(데이터 영역으로부터 차용한 잉여 비트를 포함함) 플러스 1 비트의 T4 정보를 포함할 수 있다.
- [0030] 첫번째 8개의 칩 중 하나, 예를 들어 제 2 칩이 고장나면, GEC 패리티 정보( $PA_{1-13}$ )에 추가하여, 57 비트의 데이터( $A_1$ )가 손실된다. 손실 정보는  $A_0 - A_8$ 을 판독함으로써 복구될 수 있고,  $A_1(L_{A1})$ 과 연관된 LED는 칩 에러를 지시한다. GEC 세그먼트  $PA_{0-6}$ ,  $PA_{14-20}$ ,  $PA_{21-27}$ , ...,  $PA_{49-55}$ ,  $PA_{56}$  및  $PP_A$ 를 판독하여 손실 GEC 비트  $PA_{7-13}$ 을 복구하여, 이에 의해 GEC 패리티 PA를 재구성한다.  $A_0$ 와  $A_2 - A_7$ 의 값을 조합하여, 데이터값  $A_1$ 이 재구성될 수 있고, 따라서 전체 원래 캐시 라인을 복구한다. 그러나, 제 9 칩이 고장나면, 단지 56 비트의 데이터만이  $PP_A$  및 잉여 비트( $PA_{56}$ )에 추가하여 손실될 것이다( $A_8$ ). 손실된 56 비트는 첫번째 8개의 칩( $PA_{0-55}$ ) 내에 저장된 패리티의 56개의 칼럼으로부터 복구될 수 있고, 따라서 전체 원래 캐시 라인을 복구한다. 잉여 비트( $PA_{56}$ )의 손실은 비실질적이다. 이들 경우를 가로질러, GEC 패리티 비트 자체의 정확성은 계층화된 패리티(434)(T4)에 의해 보장된다.
- [0031] 판독 동작은 에러가 검출되지 않으면 GEC(430) 정보에 액세스할 필요가 없는데, 이는 드문 이벤트이다. 따라서, GEC(430)는 판독에 대해 상당한 영향을 갖지 않는다. 기록 동작에 대해, GEC(430)는 데이터가 수정될 때 업데이트될 수 있다(이는  $P_x$ ,  $PP_x$ , 및 T4를 포함함). 기준선 구현예에서, 각각의 캐시 라인 기록은 2개의 기록 - 데이터 로케이션에 대해 하나(전체 576 비트의 데이터 + LED + 잉여 비트) 및 그 대응 GEC 로케이션에 대해 다른 하나(72-비트) - 로 변환될 수 있다. 72 비트의 GEC+T4 코드는 기록당 업데이트될 수 있지만, 다른 제약(예를 들어, DDR3 프로토콜)이 액세스당 8의 버스트의 완료와 연관될 수 있다(예를 들어, 데이터의 전체 72-바이트의 "캐시 라인" 크기). 따라서, 몇몇 성능 영향을 감소시키기 위해, 단일의 기록 명령 내로의 8개의 상이한 GEC 업데이트와 같이, 업데이트가 조합될 수 있다. 이는 기록들이 미리 버퍼링되고 메모리 컨트롤러로부터 간헐적으로 스트리밍되기 때문에 낮은 오버헤드이고, 부가의 로직이 가능한 한 다수의 GEC 기록을 병합(coalescing)하기 위해 이 스테이지에서 용이하게 구현될 수 있다. 데이터 맵핑이 일단 데이터 라인이 기록되면 GEC 기록이 로우-버퍼 히트인 것을 보장하기 때문에, 성능 영향은 더 감소된다. 부가적으로, GEC 코드에 기여하는 비트들은 컨트롤러에서 미리 이용 가능한 단일의 캐시 라인으로부터 오기 때문에, 데이터 캐시 라인 자체의 기록-전-판독에 대한 요구가 존재하지 않는다는 것을 주목하라. 이는 성능 영향을 더 최소화한다. 완전한 병합이 구현되지 않으면(기록되고 있는 어드레스에 기초하여), 데이터 마스킹이 적절한 비트를 메모리 내로 기록하는데 이용될 수 있다. 8의 완전한 버스트는 그럼에도 불구하고 수행될 수도 있다 - 몇몇 데이터 단편이 DRAM에 실제로 기록되면서 마스킹됨.

- [0032] 모든 이들 고려 사항에 의해, 모든 기록은 애플리케이션의 액세스 특성에 따라 몇몇 부분  $\delta < 1$ 에 대해,  $1 + \delta$  기록으로 변환된다. 8개의 GEC 워드가 단일의 "캐시 라인" 내에 피팅되기 때문에 비병합 기준선 구현예에서  $\delta = 1$  및 오러클러 디자인에서  $\delta = 0.125$ 이고, 단일의 기록 내로 잠재적으로 병합될 수 있다는 것을 주목하라.
- [0033] 9개의 칩(N=9)에 대한 예시적인 구현예에서, 9개의 x8 칩을 갖는 랭크 내의 각각의 64-바이트(512-비트) 캐시 라인에 대해, 이하의 비트가 사용될 수 있다: 63 비트의 LED 정보, 칩당 7 비트; 9개의 칩을 가로질러 확산된 57 비트의 GEC 패리티; 7 비트의 제 3-레벨 패리티, PPx; 및 칩당 1 비트인 9 비트의 T4 보호. 이는 캐시 라인의 512 비트로부터 총 136 비트에 가산하여 26.5%의 저장 장치 오버헤드이다. 이 26.5%로부터, 12.5%가 표준 ECC DIMM에 가산된 9번째 칩에 의해 제공될 수 있고(예를 들어, 표준 ECC-전용 동작에 대해 이를 예비할당하는 대신에, 일반적인 사용을 위해 9번째 칩을 이용 가능하게 함), 다른 14%는 GEC 영역 내의 데이터 메모리 내에 저장된다.
- [0034] 본 명세서에 설명된 예는 더 큰 전력 효율 및 더 큰 랭크-레벨 평균화와 연관된 더 넓은 I/O DRAM 부분에 적용될 수 있다. 특정예가 x16 DRAM에 대해 제공될 것이고, 유사한 기술이 x32 DRAM 이상으로 개념을 확장하기 위해 사용될 수 있다.
- [0035] 9개의 x16 DRAM의 랭크를 고려한다. 8의 버스트에서 x16 DRAM에 의해 공급된 128 비트는 x8 DRAM을 사용하는 것에 유사하게 저장 장치 오버헤드를 갖는, 114 데이터 비트 및 14 체크섬 LED 비트로서 해석될 수 있다. GEC 동작은 불변 유지될 수 있다. 액세스 입도 및 오버페치의 증가가 존재할 수 있지만, 저장 장치 오버헤드는 대략 25%(LED + GEC)로 일정하게 유지될 수 있다.
- [0036] 액세스 입도가 정확하게 1개의 캐시 라인(즉, 64-비트 버스)으로 고정되면, x16 칩을 갖는 최소 랭크 크기는 5개의 칩(4 데이터 플러스 1 ECC)이다. 각각의 칩은 103 데이터 비트( $103 * 4$  칩 = 512-비트 캐시 라인)로서 해석되는, 8의 버스트당 128 비트를 제공한다. 이는 LED 코드를 저장하기 위해 칩당 25 비트를 남겨두는데, 이러한 것은 매우 강한 에러 보호를 제공하지만 저장 영역의 낭비일 수도 있다(오버헤드가 24%일 것임). GEC 오버헤드는, 글로벌 패리티가 대략 50%의 총 오버헤드를 갖고, 25%의 저장 장치 오버헤드로, 4개의 103-비트 데이터 세그먼트에 걸쳐 컴퓨팅된 103-비트 엔티티이기 때문에, 마찬가지로 증가할 수 있다.
- [0037] 저장 장치 오버헤드가 우선순위화되면, 이 오버헤드는 액세스 입도의 증가의 절충에 의해 약 12.5%로 고정될 수 있다. 예를 들어 x16 칩 및 5-칩 랭크에 의해, 9회 판독이 연속으로 발생할 수 있어, 총 사이클당 8 비트 \* 8 사이클의 버스트 \* 9 액세스 = 5,760 비트를 판독한다. 이는 12.5%의 저장 장치 오버헤드를 갖는, 10 캐시 라인(5120 비트) 플러스 이들의 LED 코드의 매우 큰 액세스 입도를 야기한다. GEC 오버헤드는 37.5%의 전체 ECC 저장 장치 오버헤드에 대해 대략 25%(4개의 칩 중 1개)를 유지한다.
- [0038] 액세스 입도도 저장 장치 오버헤드도 손상되지 않지만, 통상의 DIMM을 구현하는데 자유도가 존재하면, 단일 DIMM 내의 이중 DRAM의 사용이 사용될 수 있다. 이 경우에, 최소 액세스 입도는 12.5% 저장 장치 오버헤드를 여전히 유지하면서 유지될 수 있다. x16 부분에 의해, 예를 들어, 최소 크기 랭크는 4개의 x16 DRAM 플러스 하나의 x8 DRAM(DRAM은 여전히 기성품이지만, DIMM은 아니라는 것을 주목하라)이어서, 72 비트의 DIMM 폭을 제공한다. 8의 버스트 길이에 의해, 각각의 x16 DRAM은 128 비트를 공급하고, x8 DRAM은 64 비트를 공급한다. 이들은 각각 (114 데이터 + 14 LED) 및 (56 데이터 + 8 LED)로서 해석되어야 한다. GEC 오버헤드 또는 동작으로의 변화가 존재하지 않을 것이다.
- [0039] 따라서, 저장 장치 오버헤드, 액세스 입도의 중요도(통상적으로, 작업부하의 액세스 로컬성의 함수) 및 동종 DIMM을 구축하기 위한 자발성(willingness)을 포함하는 다수의 옵션이 변경되어야 한다 - x16 또는 x32와 같은 넓은 I/O 부분이 이들의 감소된 전력 소비에 기인하여 메인스트림이 되기 때문에 -. 본 명세서에 설명된 예는 이들 옵션이 다양한 조합 및 변형을 갖는 디자인에 유효하도록 충분히 융통성이 있다.
- [0040] 동적 및 정적 전력의 모두의 감소를 통해, 전통적인 칩킬 메커니즘에 비교하여, 실질적인 전력 절약이 실현될 수 있다. 예를 들어 표준 8-버스트 액세스에서 정확하게 하나의 64-바이트 캐시 라인을 판독/기록하는 예를 들어 단지 9개의 x8 칩과 같은 요청을 서비스하는데 요구되는 절대 최소 개수의 칩을 활성화하는 것이 가능하다. 이는 강제 프리페칭을 유발하고 동적 전력 소비를 증가시킬 수 있는 종래의 칩킬 해결책에 대조적이다(예를 들어, 판독/기록마다 부가의 칩을 활성화하고, 표준 8-버스트 액세스마다 다수의 캐시 라인을 액세스함으로써). 본 명세서에 제공된 예는 또한 칩당 로우 버퍼의 크기가 일정하지만 더 적은 칩이 활성화되기 때문에, 활성화 전력의 감소를 가능하게 할 수 있다. 활성화 전력은 또한 더 적은 개수의 칩이 랭크를 구성하기 때문에 x4 칩

으로부터 x8 칩으로 진행하여 감소될 수 있다. 각각의 활성화의 푸트프린트는 또한 감소될 수 있어, 미사용 램크/뱅크가 신속하게 진입되고 진출될 수 있는 얇은 저전력 모드와 같은 저전력 모드로의 전이를 허용한다.

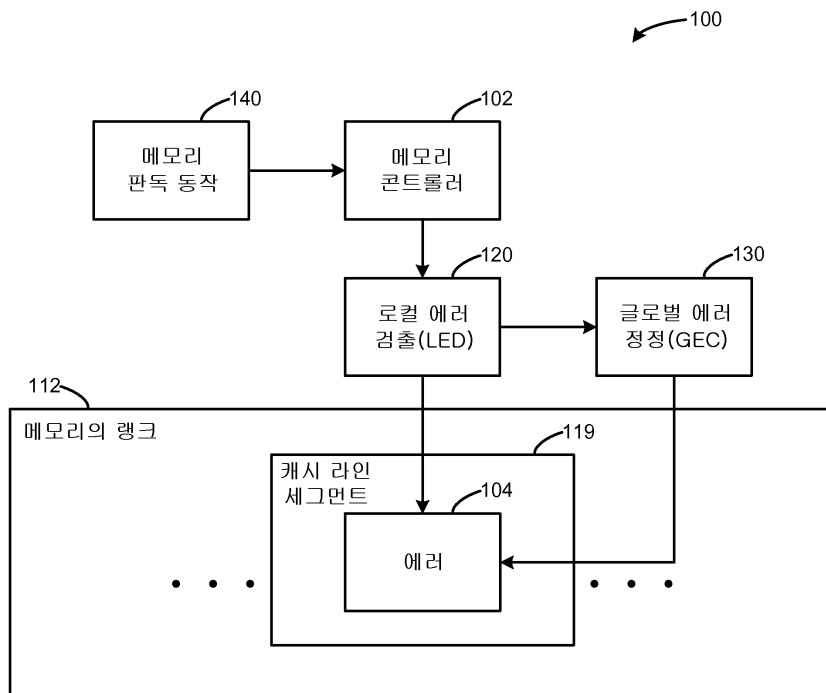
[0041] 큰 에너지 장점에 추가하여, 액세스 밀도를 감소시키는 것은 또한 성능에 긍정적인 영향을 갖는다. 시스템 내의 소정의 칩의 총 개수에 대해, 증가된 램크-레벨 및 뱅크-레벨 평행화가 존재한다. 이는 뱅크 상층 및 전체 평균 메모리 액세스 지연 시간을 감소시킬 수 있다. 이 이득의 부분은 정규 기록과 함께 요구된 GEC 라인에 대한 여분의 기록에 기인하여 손실될 수 있다. 이 오버헤드에 불구하고, 예들은 병합 없이도, 여전히 이익이 있다.

[0042] 도 5는 예에 따른 데이터 충실도의 체크에 기초하는 흐름도(500)이다. 블록 510에서, 로컬 에러 검출(LED) 정보가 에러 검출 코드에 기초하여, 메모리의 램크와 연관된 데이터의 캐시 라인 세그먼트마다 컴퓨팅된다. 블록 520에서, 글로벌 에러 정정(GEC)은 글로벌 패리티에 기초하여 캐시 라인 세그먼트에 대해 생성된다. 블록 530에서, 데이터 충실도는 램크의 캐시 라인 세그먼트 사이의 에러의 존재 및 에러의 위치를 식별하기 위해, LED 정보에 기초하여, 메모리 판독 동작에 응답하여 체크된다. 블록 540에서, 에러를 갖는 캐시 라인 세그먼트는 에러의 식별에 응답하여, GEC에 기초하여 정정된다.

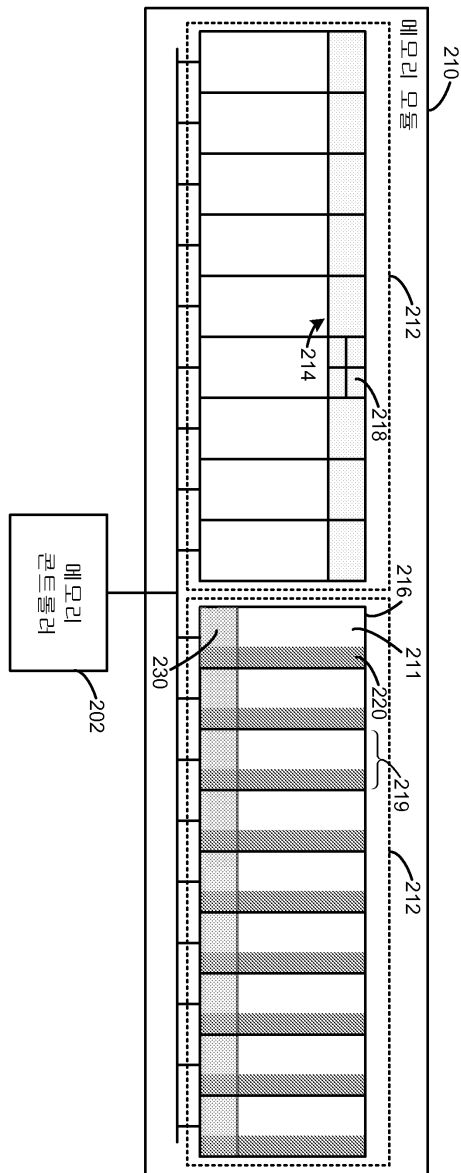
[0043] 도 6은 예에 따른 에러 검출 및/또는 정정을 수행하는 것에 기초하는 흐름도(600)이다. 블록 610에서, 로컬 에러 검출(LED)은 램크 내의 N개의 데이터 칩 사이의 칩-입도에서 에러의 위치를 검출하기 위해, 캐시 라인 세그먼트에 걸쳐 컴퓨팅된 체크섬에 기초하여, 메모리 판독 동작에 응답하여 수행된다. 블록 620에서, 글로벌 에러 정정(GEC)은 에러를 검출하는 것에 응답하여, 램크 내의 N개의 데이터 칩 상의 캐시 라인 세그먼트에 걸쳐 수행되고, GEC는 GEC 정보를 생성하기 위해 글로벌 패리티를 사용하여 수행된다. 블록 630에서, 에러를 갖는 데이터 세그먼트는 무에러 세그먼트 및 GEC 정보에 기초하여 재구성된다. 블록 640에서, GEC 정보는 기록 동작에 응답하여 업데이트된다. 블록 650에서, 계층화된 패리티가 GEC 정보를 보호하도록 생성되고, 계층화된 패리티는 N번째 칩 상에 저장되고, 복수의 칩으로부터 GEC 정보 세그먼트에 기초하여 GEC 정보를 복구하는데 사용된다.

도면

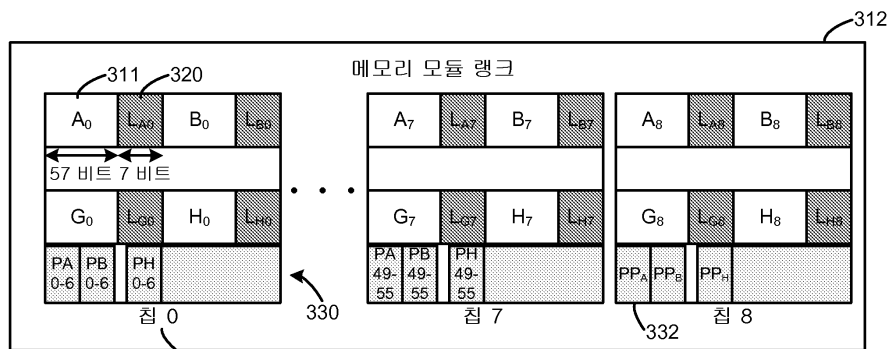
도면1



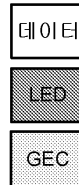
도면2



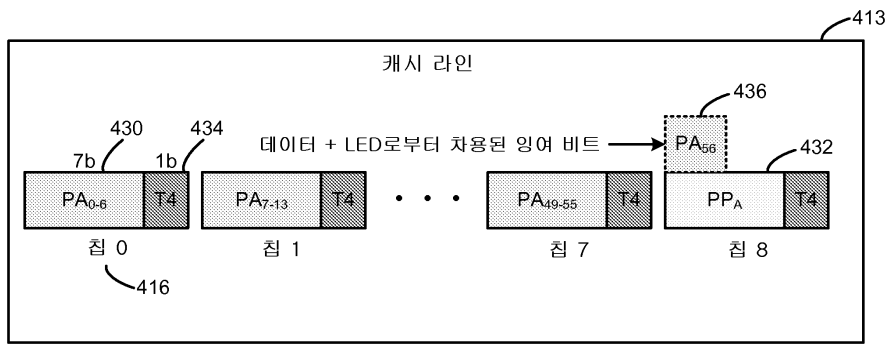
도면3



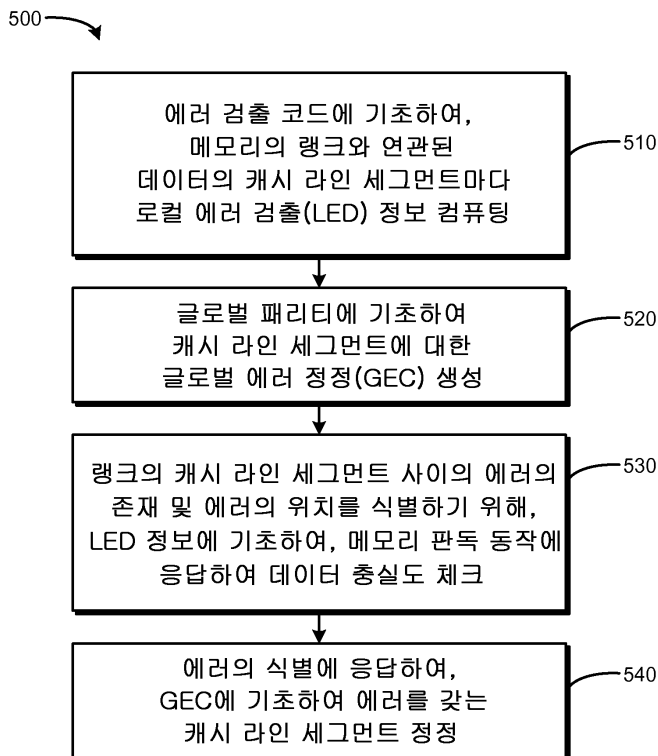
A, B, C, D, E, F, G, H - 세그먼트  $X_0$  내지  $X_8$ 로 각각 구성된 캐시 라인  
 $L_{XN}$  - 캐시 라인 X, 세그먼트 N에 대한 L1 로컬 에러 검출(LED)  
 $[PX_0:PX_N]$  - 세그먼트  $X_0$  내지  $X_8$ 를 가로지르는 L2 글로벌 에러 정정(GEC)  
 $PP_X$  - GEC 세그먼트  $PX_{0-6}$  내지  $PX_{49-55}$ 를 가로지르는 패리티



도면4



도면5



도면6

