(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2014년07월10일
(11) 등록번호 10-1418466
(24) 등록일자 2014년07월04일

(51) 국제특허분류(Int. Cl.)
H03M 13/00 (2006.01)
(21) 출원번호 10-2010-7006847
(22) 출원일자(국제) 2008년12월12일
심사청구일자 2013년12월12일
(85) 번역문제출일자 2010년03월29일
(65) 공개번호 10-2011-0052530
(43) 공개일자 2011년05월18일
(86) 국제출원번호 PCT/US2008/086537
(87) 국제공개번호 WO 2010/019169
국제공개일자 2010년02월18일
(30) 우선권주장
61/089,297 2008년08월15일 미국(US)
(56) 선행기술조사문헌
E.Cavus 외 1인, "A Performance Improvement and Error Floor Avoidance Technigue for Belief Propagation Decoding of LDPC Codes", 2005 IEEE 16th Int. Symp. PIMRC, pp.2386-2390, 2005.09.11
전체 청구항 수 : 총 10 항

(73) 특허권자
엘에스아이 코포레이션
미국 캘리포니아 95131, 새너제이, 라이더 파크 드라이브 1320
(72) 발명자
군남, 키란
미국 캘리포니아 95134 산 호세 에이피티 226 엘 란 빌리지 레인 371
(74) 대리인
장훈

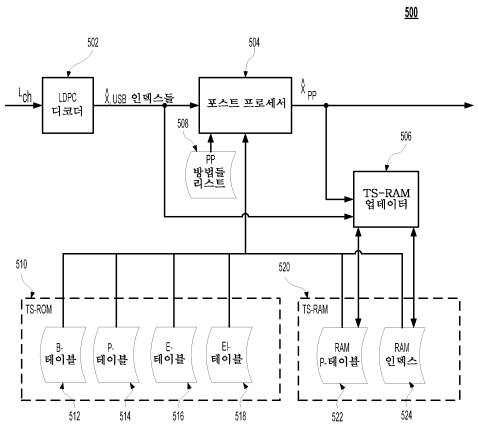
심사관 : 성경아

(54) 발명의 명칭 **니어 코드워드들의 ROM 리스트 디코딩**

(57) 요 약

본 발명의 일부 실시예들은 (LDPC) 리스트 디코딩 동안, ROM에서 트랩핑 세트 프로파일들의 구성 및 그 프로파일들을 검색하기 위한 방법들을 제공한다. 프로파일들은 우세도, 즉, 디코더의 에러-플로어 특성들에 대한 그들의 영향에 의해 순위화된다. 더 우세한 트래핑 세트 프로파일들은 불만족된 검사 노드들(USCs) 및 잘못 만족된 검사 노드들(MSCs) 모두에 관한 정보를 포함하고, 덜 지배적인 트래핑 세트 프로파일들은 단지 USC들에 관한 정보만을 포함한다. 트래핑-세트 프로파일 정보는 포인터-추적 검색을 이용하여 가장 지배적인 매칭 트래핑 세트 프로파일들의 빠른 위치 지정 및 검색을 허용하는 링크된 계층화된 데이터 테이블의 수로 조직화된다.

대 표 도 - 도5



특허청구의 범위

청구항 1

그래프-기반 코드를 이용하여 인코딩되는 인코딩된 데이터를 디코딩하는 방법에 있어서:

(a) 후보 디코딩된 코드워드(codeword)를 생성하기 위해 상기 인코딩된 데이터를 디코딩하는 단계; 및

(b) 상기 후보 디코딩된 코드워드가 디코딩된 정확한 코드워드가 아닌 경우에, 상기 디코딩된 정확한 코드워드를 생성하기를 시도하기 위하여 트래핑-세트(trapping-set)-ROM 리스트 디코딩 방법을 수행하는 단계를 포함하며,

상기 후보 디코딩된 코드워드는 적어도 하나의 불만족된 검사 노드(unsatisfied check node)를 갖고, 불만족된 검사 노드는 패리티 검사를 실패한 검사 노드이고;

상기 트래핑 세트-ROM 리스트 디코딩 방법은:

(b1) 상기 적어도 하나의 불만족된 검사 노드에 대응하는 트래핑-세트 프로파일을 식별하기 위해 ROM 메모리에 저장된 하나 이상의 트래핑 세트 프로파일들을 액세스하는 단계로서;

상기 ROM 메모리에 저장된 제 1 저장된 트래핑 세트 프로파일은 적어도 하나의 불만족된 검사 노드에 대한 저장된 정보 및 적어도 하나의 잘못 만족된 검사(mis-satisfied check) 노드에 대한 저장된 정보를 포함하고, 잘못 만족된 검사 노드는 (i) 에러 있는 비트 노드들과 관련되고 (ii) 상기 패리티 검사를 만족하는 검사 노드이고;

상기 ROM 메모리에 저장된 제 2 저장된 트래핑 세트 프로파일은 하나 이상의 불만족된 검사 노드들 및 하나 이상의 잘못 만족된 검사 노드들을 갖는 트래핑 세트와 관련되고, 상기 제 2 저장된 트래핑 세트 프로파일은 상기 하나 이상의 불만족된 검사 노드들에 대한 저장된 정보를 포함하지만 상기 하나 이상의 잘못 만족된 검사 노드들에 대한 정보는 포함하지 않는, 상기 하나 이상의 트래핑 세트 프로파일들을 액세스하는 단계;

(b2) 식별된 상기 트래핑-세트 프로파일과 관련된 하나 이상의 에러 있는 비트 노드들을 식별하는 단계;

(b3) 변경된 후보 디코딩된 코드워드를 생성하기 위해 상기 식별된 하나 이상의 에러 있는 비트 노드들을 변경하는 단계

를 포함하는, 인코딩된 데이터를 디코딩하는 방법.

청구항 2

제 1 항에 있어서,

상기 제 1 저장된 트래핑 세트 프로파일은, 상기 제 1 저장된 트래핑-세트 프로파일과 관련된 상기 트래핑 세트가 상기 제 2 저장된 트래핑-세트 프로파일과 관련된 상기 트래핑 세트보다 상기 디코딩의 에러-플로어 특성들(erro-floor characteristics)에 대해 더 큰 영향을 가지는 것을 의미하는, 상기 제 2 저장된 트래핑 세트 프로파일의 트래핑 세트보다 더 우세한(dominant) 트래핑 세트와 관련되는, 인코딩된 데이터를 디코딩하는 방법.

청구항 3

제 1 항에 있어서,

단계 (b2)는:

(1) 상기 후보 디코딩된 코드워드에서 상기 적어도 하나의 불만족된 검사 노드와 관련된 하나 이상의 에러 있는 비트 노드들을 식별하는 단계; 및

(2) (i) 상기 후보 디코딩된 코드워드가 하나 이상의 잘못 만족된 검사 노드들을 갖고 (ii) 상기 ROM 메모리가 상기 하나 이상의 잘못 만족된 검사 노드들에 관한 정보를 포함하는 경우, 상기 하나 이상의 잘못 만족된 검사 노드들에 관련된 하나 이상의 에러 있는 비트 노드들을 식별하는 단계;

를 포함하고,

상기 트래핑-세트-ROM 리스트 디코딩 방법은:

(b4) 추가 처리를 수행하는 단계로서:

상기 (b2) 단계가 하나 이상의 에러 있는 비트 노드들을 식별하는 경우, 상기 추가 처리는 변경된 후보 디코딩된 코드워드에 신드롬 검사를 수행하는 단계를 포함하고;

상기 (b2) 단계가 하나 이상의 에러 있는 비트 노드들을 식별하지 않는 경우, 상기 추가 처리는 변경된 인코딩된 데이터를 디코딩하는 단계를 포함하는, 상기 추가 처리 수행 단계를 더 포함하는, 인코딩된 데이터를 디코딩하는 방법.

청구항 4

제 1 항에 있어서,

트래핑 세트 프로파일에서 각각의 불만족된 검사 노드에 대하여, 상기 트래핑 세트 프로파일은:

상기 불만족된 검사 노드가 위치된 디코딩 층;

상기 디코딩 층 내 상기 불만족된 검사 노드에 대한 인덱스; 및

상기 불만족된 검사 노드에 관련된 하나 이상의 에러 있는 비트 노드들에 대한 하나 이상의 인덱스들을 포함하고;

트래핑 세트 프로파일에서 각각의 잘못 만족된 검사 노드에 대하여, 상기 트래핑 세트 프로파일은 상기 잘못 만족된 검사 노드와 관련된 하나 이상의 에러 있는 비트 노드들에 대한 위치 정보를 포함하는, 인코딩된 데이터를 디코딩하는 방법.

청구항 5

제 4 항에 있어서,

상기 ROM 메모리는 복수의 저장된 트래핑 세트 프로파일들을 포함하고;

상기 복수의 저장된 트래핑 세트 프로파일들에 대한 에러 있는 비트 노드 인덱스들, 상기 디코딩 층들, 및 불만족된 검사 인덱스들이 제 1 테이블에 저장되고;

잘못 만족된 검사 노드와 관련된 상기 에러 있는 비트 노드들에 대한 상기 위치 정보는 제 2 테이블에 저장되는, 인코딩된 데이터를 디코딩하는 방법.

청구항 6

제 5 항에 있어서,

상기 저장된 트래핑 세트 프로파일들은 상기 저장된 트래핑 세트 프로파일들에서 불만족된 검사 노드들의 수에 기초하여 상기 제 1 테이블에서 그룹화되고;

동일한 수의 불만족된 검사 노드들을 갖는 저장된 트래핑 세트 프로파일들의 각그룹의 상기 저장된 트래핑 세트 프로파일들은 트래핑 세트 프로파일 우세도(dominance)에 의해 순위화되고, 트래핑 세트 프로파일의 우세도는 관련된 트래핑 세트가 상기 단계 (a)의 디코딩에 대한 에러-플로어 특성들(error-floor characteristics)에 갖는 효과에 의존하는, 인코딩된 데이터를 디코딩하는 방법.

청구항 7

제 6 항에 있어서,

다수의 불만족된 검사 노드들을 갖는 각각의 트래핑 세트 프로파일에 대하여:

상기 불만족된 검사 노드들은 디코딩 층에 의해 상기 트래핑 세트 프로파일에서 그룹화되고;

동일한 디코딩 층을 갖는 불만족된 검사 노드들의 그룹의 상기 불만족된 검사 노드들은 불만족된 검사 인덱스에 의해 정렬되는, 인코딩된 데이터를 디코딩하는 방법.

청구항 8

제 6 항에 있어서,

상기 ROM 메모리는 제 3 테이블 및 제 4 테이블을 추가로 포함하고,

상기 제 3 테이블은 매칭된 트래핑 세트 프로파일에 대한 잘못 만족된 검사 노드들에 관련된 상기 에러 있는 비트 노드들에 대한 상기 위치 정보의 상기 제 2 테이블에서 어드레스들을 식별하고,

상기 제 1 테이블에서 동일한 수의 불만족된 검사 노드들을 갖는 저장된 트래핑 세트 프로파일들의 각각의 그룹에 대하여, 상기 제 4 테이블은:

(i) 상기 저장된 트래핑 세트 프로파일들의 그룹에 대한 상기 제 1 테이블의 시작 어드레스;

(ii) 상기 잘못 만족된 검사 노드들에 관련된 위치 정보를 갖는 그룹에서 트래핑 세트 프로파일들의 수; 및

(iii) 상기 저장된 트래핑 세트 프로파일들의 그룹에 대한 상기 제 3 테이블의 시작 어드레스를 식별하는, 인코딩된 데이터를 디코딩하는 방법.

청구항 9

제 1 항에 있어서,

상기 그래프-기반 코드는 저-밀도 패리티 검사(LDPC) 코드인, 인코딩된 데이터를 디코딩하는 방법.

청구항 10

그래프-기반 코드를 이용하여 인코딩되는 인코딩된 데이터를 디코딩하는 장치에 있어서:

후보 디코딩된 코드워드를 생성하기 위하여 상기 인코딩된 데이터를 디코딩하도록 적응된 디코더; 및

상기 후보 디코딩된 코드워드가 디코딩된 정확한 코드워드가 아닌 경우, 상기 디코딩된 정확한 코드워드를 생성하기를 시도하기 위해 트래핑-세트-ROM 리스트 디코딩 방법을 수행하도록 적응된 포스트-프로세서(post-processer)를 포함하고:

상기 후보 디코딩된 코드워드는 적어도 하나의 불만족된 검사 노드를 갖고, 불만족된 검사 노드는 패리티 검사를 실패한 검사 노드이고;

상기 트래핑 세트-ROM 리스트 디코딩 방법은:

(b1) 상기 적어도 하나의 불만족된 검사 노드에 대응하는 트래핑-세트 프로파일을 식별하기 위해 ROM 메모리에 저장된 하나 이상의 트래핑 세트 프로파일들을 액세스하는 단계로서;

상기 ROM 메모리에 저장된 제 1 저장된 트래핑 세트 프로파일은 적어도 하나의 불만족된 검사 노드에 대한 저장된 정보 및 적어도 하나의 잘못 만족된 검사(mis-satisfied check) 노드에 대한 저장된 정보를 포함하고, 잘못 만족된 검사 노드는 (i) 에러 있는 비트 노드들과 관련되고 (ii) 상기 패리티 검사를 만족하는 검사 노드이고;

상기 ROM 메모리에 저장된 제 2 저장된 트래핑 세트 프로파일은 하나 이상의 불만족된 검사 노드들 및 하나 이상의 잘못 만족된 검사 노드들을 갖는 트래핑 세트와 관련되고, 상기 제 2 저장된 트래핑 세트 프로파일은 상기 하나 이상의 불만족된 검사 노드들에 대한 저장된 정보를 포함하지만 상기 하나 이상의 잘못 만족된 검사 노드들에 대한 정보는 포함하지 않는, 상기 하나 이상의 트래핑 세트 프로파일들을 액세스하는 단계;

(b2) 식별된 상기 트래핑-세트 프로파일과 관련된 하나 이상의 에러 있는 비트 노드들을 식별하는 단계;

(b3) 변경된 후보 디코딩된 코드워드를 생성하기 위해 상기 식별된 하나 이상의 에러 있는 비트 노드들을 변경하는 단계

를 포함하는, 인코딩된 데이터를 디코딩하는 장치.

청구항 11

삭제

청구항 12

삭제

청구항 13

삭제

청구항 14

삭제

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

명 세 서

기 술 분 야

[0001] 관련 출원들과의 상호-참조

[0002] 본 출원은 그 내용들이 전체적으로 본원에 참조되어 있는 대리인 문서 번호 제08-0241-PR 호로서 2008년 8월 15일자로 출원된 미국 가출원 번호 제61/089,297호의 출원일의 이점을 주장한다.

[0003] 본 출원의 주제는 그 내용들이 전체적으로 본원에 참조되어 있는 대리인 문서 번호 제08-0241호로서 본 출원과 동일자로 출원된 미국 특허 출원 번호 제XX/xxx,xxx호와 관련된다.

[0004] 본 발명은 디지털 신호 프로세싱(digital signal processing)에 관한 것이며, 특히, 저-밀도 패리티 검사(Low-Density Parity Check: LDPC) 코딩으로서 공지된 데이터-인코딩(data-encoding) 방법에 관한 것이다.

배 경 기 술

[0005] 통신은 송신기에 의한 통신 채널을 통한 수신기로의 정보의 송신이다. 실제 세상에서, 통신 채널은 송신기로부터 수신된 정보의 왜곡된 버전(distorted version)을 수신기로 출력하는 잡음이 있는 채널이다. 하드 디스크(Hard Disk: HD) 드라이브는 송신기로부터 정보를 수용하고, 상기 정보를 저장하고 나서, 아마도, 상기 정보를 다소 왜곡된 카피(copy)를 수신기로 송신하는 하나의 이와 같은 잡음이 있는 채널이다.

[0006] HD 드라이브와 같은 통신 채널에 의해 도입된 왜곡은 채널 에러(channel error), 즉, 채널 입력 신호가 0이었을 때 수신기가 채널 출력 신호를 1로서 해석하는 경우, 및 그 반대의 경우를 초래할 만큼 충분히 클 수 있다. 채널 에러들은 처리량을 감소시키므로, 바람직하지 않다. 그러므로, 채널 에러들을 검출 및/또는 수정하는 툴(tool)들이 계속 필요하다. 저-밀도 패리티 검사(LDPC) 코딩이 채널 에러들의 검출 및 수정을 위한 하나의 방법이다. LDPC 코드들은 공지된 니어-섀넌-리미트(near-Shannon-limit) 코드들 중에서 낮은 신호-대-잡음 비(SNR) 애플리케이션들에 대한 매우 낮은 비트-에러 레이트(bit-error rate)들을 성취할 수 있는 코드들이다. LDPC 디코딩은 병렬화(parallelization), 낮은 구현 복잡성, 낮은 디코딩 지연, 뿐만 아니라, 높은 SNR들에서의 덜-심각한 에러-플로어(error-floor)들에 대한 가능성에 의해 두드러지게 된다. LDPC 코드들은 가상으로 모든 차세대 통신 표준들에 대해 고려된다.

발명의 내용

해결하려는 과제

[0007] 본 발명의 목적은 그래프-기반 코드(graph-based code)로 인코딩되는 인코딩된 데이터를 디코딩하는 방법 및 장치를 제공하는 것이다.

과제의 해결 수단

[0008] 일부 실시예들에서, 본 발명은 그래프-기반 코드를 이용하여 인코딩되는 인코딩된 데이터를 디코딩하는 방법들을 포함한다. 상기 방법은 (a) 상기 인코딩된 데이터에 디코딩을 수행하여 후보 디코딩된 코드워드(candidate decoded codeword)를 생성하는 단계 및 (b) 상기 후보 디코딩된 코드워드가 디코딩된 정확한 코드워드가 아닌 경우에, 디코딩된 정확한 코드워드를 생성하도록 시도하기 위하여 트래핑-세트(Trapping-Set: TS)-ROM 리스트 디코딩 방법을 수행하는 단계를 포함한다. 후보 디코딩된 코드워드는 적어도 하나의 불만족된 검사 노드를 가지고, 불만족된 검사 노드는 패리티 검사를 실패한 검사 노드이다. 상기 TS-ROM 리스트 디코딩 방법은 ROM 메모리에 저장된 하나 이상의 TS 프로파일(profile)들을 액세스한다. 제 1 저장된 TS 프로파일은 적어도 하나의 불만족된 검사(unsatisfied check; USC) 노드에 대한 저장된 정보 및 적어도 하나의 잘못 만족된 검사(mis-satisfied check; MSC)에 대한 저장된 정보를 포함한다. MSC 노드는 (1) 에러 있는 비트 노드들(EBNs)에 관련되고 (2) 패리티 검사를 만족하는 검사 노드이다. 제 2 저장된 TS 프로파일은 하나 이상의 USC 노드들 및 하나 이상의 MSC 노드들을 갖는 트래핑 세트에 관련되고, 제 2 저장된 TS 프로파일은 하나 이상의 USC 노드들에 대한 저장된 정보를 포함하지만, 하나 이상의 MSC 노드들에 대한 정보를 포함하지 않는다.

[0009] 다른 실시예들에서, 본 발명은 그래프-기반 코드를 이용하여 인코딩되는 인코딩된 데이터를 디코딩하는 장치이다. 상기 장치는 (a) 상기 인코딩된 데이터에 대한 디코딩을 적용하여 후보 디코딩된 코드워드를 생성하는 디코더, 및 (b) 상기 후보 디코딩된 코드워드가 디코딩된 정확한 코드워드가 아닌 경우에, 디코딩된 정확한 코드워드를 생성하도록 시도하기 위하여 트래핑-세트(TS)-ROM 리스트 디코딩 방법을 수행하도록 적응되는 포스트 프로세서를 포함한다. 후보 디코딩된 코드워드는 적어도 하나의 불만족된 검사 노드를 가지고, 불만족된 검사 노드는 패리티 검사를 실패한 검사 노드이다. TS-ROM 리스트 디코딩 방법은 ROM 메모리에 저장된 하나 이상의 TS 프로파일들을 액세스한다. 제 1 저장된 TS 프로파일은 적어도 하나의 불만족된 검사(USC) 노드에 대한 저장된 정보 및 적어도 하나의 잘못 만족된 검사(MSC)에 대한 저장된 정보를 포함한다. MSC 노드는 (1) 에러 있는 비트 노드들(EBNs)에 관련되고 (2) 패리티 검사를 만족하는 검사 노드이다. 제 2 저장된 TS 프로파일은 하나 이상의 USC 노드들 및 하나 이상의 MSC 노드들을 갖는 트래핑 세트에 관련되고, 제 2 저장된 TS 프로파일은 하나 이상의 USC 노드들에 대한 저장된 정보를 포함하지만, 하나 이상의 MSC 노드들에 대한 정보를 포함하지 않는다.

발명의 효과

[0010] 본 발명에 의하면, 그래프-기반 코드로 인코딩되는 인코딩된 데이터를 디코딩하는 방법 및 장치가 제공된다.

도면의 간단한 설명

[0011] 도 1은 LDPC 코딩을 사용하는 전형적인 하드 디스크(HD) 드라이브(100)의 일부의 블록도.

도 2(a)는 LDPC H 매트릭스(200)를 도시한 도면이며, 도 2(b)는 **H** 매트릭스(200)의 태너 그래프(Tanner graph).

도 3은 디코더(112)에 의해 사용된 전형적인 LDPC 디코딩 방법(300)의 흐름도.

도 4는 트래핑 세트들을 식별하고 이러한 트래핑 세트들에 관한 다양한 정보를 기록하는 오프-라인 트래핑-세트(TS) 시뮬레이션 툴(off-line trapping-set simulation tool)(400)의 블록도.

도 5는 본 발명의 일 실시예에 따른 LDPC 디코딩 시스템(500)의 블록도.

도 6은 도 5의 ROM P-테이블(514)의 예시적인 배치도.

도 7은 도 5의 B-테이블(512)의 예시적인 배치도.

도 8은 도 5의 E-테이블(516)의 예시적인 배치도.

도 9는 도 5의 EI-테이블(518)의 예시적인 배치도.

도 10은 도 5의 RAM P-테이블(522)의 예시적인 배치도.

도 11은 도 5의 RAM 인덱스 테이블(524)의 예시적인 배치도.

도 12는 도 5의 LDPC 디코딩 시스템(500)에 의해 사용된 예시적인 프로세스(1200)의 흐름도.

도 13은 도 5의 포스트-프로세서(post-processor)(504)에 의해 구현되는 도 12의 예시적인 TS-ROM 리스트 디코딩 프로세스(1206)의 흐름도.

도 14는 도 13의 예시적인 TS-ROM 탐색 프로세스(1314)의 흐름도.

도 15는 도 12의 예시적인 TS-RAM 리스트 디코딩 프로세스(1208)의 흐름도.

도 16은 도 12의 예시적인 TS-RAM 업데이트 프로세스(1216)의 흐름도.

발명을 실시하기 위한 구체적인 내용

[0012] 본 발명의 다른 양태들, 특징들 및 장점들은 다음의 상세한 설명, 첨부된 청구항들, 및 유사하거나 동일한 요소들에는 동일한 참조 번호들이 병기되어 있는 첨부 도면들로부터 더 충분히 명백해질 것이다.

[0013] 도 1은 LDPC 코딩을 사용하는 전형적인 하드 디스크(HD) 드라이브(100)의 일부의 블록도이다. HD 드라이브(100)는 플래터(platter)들(102) 및 판독 채널(104)을 포함한다. 판독 채널(104)은 LDPC 인코더(106), 기록 프로세서(108), 판독 프로세서(110), 및 LDPC 디코더(112)를 포함한다. 경로(114)는 LDPC 인코더(106) 및 LDPC 디코더(112) 사이의 잡음이 있는 채널이다.

[0014] 플래터들(102)로 기록될 정보 워드들은 LDPC 인코더(106)에 의해 프로세싱되어, LDPC 코드워드들을 산출한다. LDPC 코드워드들은 다수의 모듈(module)들, 예를 들어, BPSK(이진 위상-시프트 키잉) 인코더, 디지털-대-아날로그 변환기 등을 포함하는 기록 프로세서(108)로 송신된다. 기록 프로세스(108)의 출력(116)이 플래터들(102)로 기록된다.

[0015] 플래터들(102)로부터 판독된 신호들(118)은 다수의 모듈들, 예를 들어, 전치-증폭기, 연속-시간 필터(continuous-time filter), 고정된-임펄스 응답 필터(fixed-impulse response filter), 검출기, 아날로그-대-디지털 변환기 등을 포함하는 판독 프로세서(110)로 송신된다. 판독 프로세서(110)는 LDPC 디코더(106)에 로그-우도 비(Log-Likelihood Ratio: LLR) 값들(L_{ch})을 출력하고, 상기 LDPC 디코더는 이어서 디코딩된 정보 워드들을 출력한다. 추가적으로, LDPC 디코더(106)는 판독 프로세서(110)로 E_{LDPC} 값들을 역송신한다. E_{LDPC}는 아래의 식 6에 의해 정의되며, 중간의 계산된 LLR 값들을 나타낸다. 판독 프로세서(110)는 자신의 성능을 튜닝(tuning)하는데 E_{LDPC} 값들을 사용하는데, 이는 터보-디코딩(turbo-decoding)으로서 공지된 프로세스이다.

[0016] **LDPC 인코딩**

[0017] LDPC 인코더(106)는 코드워드를 산출하기 위하여, LDPC 코드에 의해 지정된 다수의 패리티 비트들을 정보 워드의 비트들에 부가한다. 정보 워드 내의 비트들은 가변 비트들로서 공지되어 있고, 이러한 가변 비트들의 수는 K로 표시된다. LDPC 코드워드 내의 비트들의 총 수는 N으로 표시된다. 따라서, 패리티 비트들의 수는 N-K에 의해 제공된다. 특정 LDPC 코드의 레이트는 K/N, 즉, 정보 워드 길이 대 코드워드 길이의 비이다. 따라서, 9-비트 코드워드를 산출하기 위하여 각각의 3-비트 정보 워드에 6 패리티 비트들을 부가하는 LDPC 코드는 1/3의 레이트를 갖는다. 전형적인 HD 드라이브의 경우에, 4506 비트의 코드워드 길이 및 0.9의 레이트에 대하여, 워드 길이 K는 4096 비트(전형적인 HD 드라이브 섹터의 길이)이며, 패리티 비트들의 수는 대략 410 비트이다.

[0018] LDPC 코드워드 내의 각각의 패리티 비트는 특정 LDPC 코드에 의해 지정된 바와 같은 특정한 방식으로 상기 코드워드 내의 하나 이상의 다른(가변 또는 패리티) 비트들과 관련되며, 패리티 비트에 할당된 값은 LDPC 코드를 만족시키도록 설정된다. 전형적인 LDPC 코드들은 관련된 비트들이 패리티 검사 제약들, 예를 들어, 관련된 비트들의 합이 짝수인 것, 예를 들어, 합 모듈로 2(sum modulo 2) = 0인 것을 만족시킨다는 것을 지정한다.

[0019] **LDPC 코드**

[0020] 특정 LDPC 코드가 패리티 검사 매트릭스, 또는 H 매트릭스, 또는 간단히 **H**로서 공지된 1들 및 0들의 2차원 매트릭스에 의해 정의된다. **H**는 LDPC 인코더 및 디코더 둘 모두에 의해 연역적으로 공지된다. **H**는 N개의 컬럼

(column)들 및 N - K개의 로우(row)들, 즉, 코드워드의 모든 비트에 대한 컬럼 및 모든 패리티 비트에 대한 로우를 포함한다. H 내의 각각의 1은 컬럼의 코드워드 비트 및 로우의 패리티 비트 사이의 관련성을 나타낸다. 예를 들어, H의 제 3 로우, 제 7 컬럼에서의 1은 제 3 패리티 검사 비트가 코드워드의 제 7 비트와 관련된다는 것을 의미한다. 검사 비트 및 상기 검사 비트와 관련된 모든 가변 비트들의 값의 모듈로 2 합은 0이어야 한다.

[0021] H의 컬럼 내의 1들의 수는 상기 컬럼의 가중치(w_c)로서 공지된다. 유사하게, H의 로우 내의 1들의 수는 상기 로우의 가중치(w_r)로서 공지된다. 모든 컬럼들이 동일한 w_c를 가지며 모든 로우들이 동일한 w_r을 가지는 H에 의해 정의된 LDPC 코드는 규칙적인 LDPC 코드로서 공지된다. w_c 및/또는 w_r이 모든 컬럼들 및/또는 로우들에 걸쳐 동일하지 않은 H에 의해 정의된 LDPC 코드는 불규칙적 LDPC 코드로서 공지된다.

[0022] 전형적인 LDPC 코드들의 정의 특성은 H가 "희박하다(sparse)"는 것, 즉, H의 요소들이 주로 0들이고 1들이 아주 적다는 것이다. H 매트릭스들이 전형적으로 양호하게 동작하기 위하여 $w_c \geq 3$을 필요로 하고, 불규칙적 LDPC 코드들이 규칙적 LDPC 코드들을 능가한다는 것이 연구를 통해 제시되었다.

[0023] 도 2(a)는 LDPC H 매트릭스(200)를 도시한다. H 매트릭스(200)는 N=9개의 컬럼들 및 N-K=6개의 로우들을 포함한다. 따라서, H 매트릭스(200)는 3-비트 정보 워드를 수용하고, 6 패리티 비트를 부가하고, 9-비트 코드워드를 출력하는 LDPC 코드를 정의한다. 따라서, 이 특정 LDPC 코드의 레이트는 3/9 또는 1/3이다. H 매트릭스(200)에 의해 정의된 LDPC 코드는 2개의 w_c 및 3개의 w_r로 규칙적이다.

[0024] **채널 출력: 로그 우도 비들**

[0025] 도 1을 참조하면, LDPC 인코더(106) 및 LDPC 디코더(112) 사이의 경로(114)는 잡음이 있는 채널이며, 이와 같이, 디코더(112)는 LDPC 인코더(106)에 의해 출력된 코드워드들의 완전한 카피를 수신하지 못한다. 그 대신에, 판독 프로세서(110)는 하나 이상의 L_{ch} 값들을 출력하는데, 여기서 각각의 L_{ch} 값은 채널 입력 코드워드 내의 비트에 대응한다.

[0026] 각각의 L_{ch} 값은 로그-우도 비(LLR)이다. LLR은 비트들의 수를 포함하는 데이터 구조이며, 여기서 단일 부호 비트(single sign bit)는 경판정(hard decision)(즉, 원래 비트가 1이었는지 또는 0이었는지에 관한 판독 프로세서(110)의 가장 양호한 추측)을 표시하고, 나머지 매그니튜드 비트(magnitude bit)들은 상기 경판정에서의 판독 프로세서(110)의 신뢰도를 표시한다. 더 정확하게는, LLR은 $\log \frac{p_0}{p_1}$ 을 나타내고, 여기서 p_0는 샘플이 0을 나타낼 확률이고, p_1은 샘플이 1을 나타낼 확률이다.

[0027] 예를 들어, 판독 프로세서(110)는 각각의 L_{ch} 값을 5-비트 데이터 구조로서 출력할 수 있고, 여기서, 최상위 비트는 경-판정 값을 표시하는 부호 비트이며, 4 매그니튜드 비트들의 16개의 값들은 경 판정의 신뢰도를 표시한다. 따라서, 예를 들어, 일 전형적인 방식에서, 이진 00000의 LLR 값은 최소 신뢰도를 갖는 0의 경-판정 값을 표시할 것이고, 이진 01111의 값은 최대 신뢰도를 갖는 0의 경-판정 값을 표시할 것이고, 이진 10000은 사용되지 않을 것이고, 이진 10001은 최소 신뢰도를 갖는 1의 경-판정 값을 표시할 것이고, 이진 11111의 값은 최대 신뢰도를 갖는 1의 경-판정 값을 표시할 것이다.

[0028] **LDPC 디코딩: 신뢰 전파(Belief propagation)**

[0029] 도 3은 디코더(112)에 의해 사용된 전형적인 LDPC 디코딩 방법(300)의 흐름도이다. LDPC 디코더(112)는 N개의 수의 L_{ch} 값들을 수신하고, 디코딩된 정보 워드를 출력한다. 디코딩 방법(300)의 핵심은 신뢰 전파라고 칭해지는 반복적인 2-단계 메시지-통과 알고리즘이다. 신뢰 전파는 태너 그래프라고 칭하는 시각화의 사용으로 가장 양호하게 설명된다.

[0030] 도 2(b)는 H 매트릭스(200)의 태너 그래프이다. 일반적으로, 태너 그래프는 1) H 내의 컬럼들의 수와 동일한(따라서, 가변 비트들의 수와 동일한) 다수의 비트 노드들(n), 2) H 내의 로우들의 수와 동일한(따라서, 패리티 비트들의 수와 동일한) 다수의 검사 노드들(m), 3) 각각이 단일 비트 노드를 단일 검사 노드에 연결하는, 에지(edge)들이라고도 칭해지는 라인들, 4) 각각의 비트 노드(n)에 대하여, 수신기로부터 수신되는 원래 L_{ch} 값, 및 5) 각각의 비트 노드(n)에 대하여, 계산된 경-판정 출력 값($\hat{x}_n$)을 포함한다. 태너 그래프(2(b))는 9개의 비트

노드들(n_0 내지 n_8), 6개의 검사 노드들(m_0 내지 m_5), 비트 노드들을 검사 노드들에 연결하는 18개의 에지들(202), 9개의 L_{ch} 값들, 및 9개의 $\hat{x}_n$ 값들을 포함한다.

[0031] 태너 그래프에서의 에지들은 (즉, 가변) 비트 노드들(n) 및 검사 노드들(m) 사이의 관계들을 나타내는데, 즉, 에지들은 **H** 내의 1들을 나타낸다. 예를 들어, 도 2(b)에서, 에지(202)는 제 1 비트 노드 n_0를 제 4 검사 노드 m_3에 연결하는데, 이는 도 2(a)의 **H** 매트릭스(200)의 제 1 컬럼, 제 4 로우에 1이 존재한다는 것을 의미한다.

[0032] 태너 그래프는 이분 그래프(bipartite graph)인데, 즉, 엣지가 비트 노드를 하나의 검사 노드에 연결시킬 수 있고, 비트 노드를 또 다른 비트 노드에 연결시키거나 검사 노드를 또 다른 검사 노드에 연결시킬 수는 없다. 에지들에 의해 특정 검사 노드(m)에 연결된 모든 비트 노드들(n)의 세트는 N(m)으로 표시된다. 에지들에 의해 특정 비트 노드(n)에 연결된 모든 검사 노드들(m)의 세트는 M(n)으로 표시된다.

[0033] 특정 (비트 또는 검사) 노드의 인덱스(index)는 그래프에서 자신의 순서적 시퀀스(ordinal sequence)이다. (비트 또는 검사) 노드의 정도는 상기 노드에 연결된 에지들의 수이다. 따라서, 태너 그래프에서의 비트 노드(n)의 정도는 대응하는 H 매트릭스 내의 컬럼(n)의 가중치(w_c)와 동일하고, 태너 그래프에서의 검사 노드(m)의 정도는 대응하는 H 매트릭스 내의 로우(m)의 가중치(w_r)와 동일하다.

[0034] 도 3으로 리턴하면, 프로세싱은 단계(302)에서 시작하고, 디코더 초기화 단계(304)로 진행한다. 디코더 초기화(304)는 비트 노드(n)에 연결된 모든 에지들(예를 들어, 도 2(b)의 202)을 비트 노드(n)와 관련된 대응하는 L_{ch} 값으로 설정하는 것, 및 비트 노드의 $\hat{x}_n$ 값을 비트 노드 n의 L_{ch}의 경-판정 값으로 설정하는 것을 포함한다. 따라서, 예를 들어, 도 2(b)에서, 비트 노드 n_0와 관련된 L_{ch} 값이 +5인 경우에, 단계(304)에서, 비트 노드 n_0를 검사 노드들 m_0 및 m_3에 연결시키는 2개의 에지들(202)은 +5로 설정되고, 비트 노드 n의 $\hat{x}_n$ 값은 1로 설정된다. 이 단계의 제 1 부분을 표현하는 대안적인 방식은 비트 노드 n_0가 세트 $M(n_0)$ 내의 모든 검사 노드 m에 +5의 메시지를 송신하는 것이다. 비트 노드(n)로부터 검사 노드(m)로 송신된 메시지는 Q_{nm}으로 표시되고, 여기서 Q_{nm}은 LLR의 형태이다. 이제 막 초기화되었던 디코더의 상태는 상태 0이라고 칭해진다.

[0035] 그 후, 단계(304)는 N개의 $\hat{x}_n$ 값들을 포함하는 벡터($\hat{x}$)를 신드롬 검사 단계(syndrome check step)(306)에 송신한다. 벡터($\hat{x}$)는 코드워드 후보이다. 신드롬 검사 단계(306)는 다음 식 1을 사용하여 신드롬 벡터(z)를 계산하며:

[0036] $$z = \hat{x}H^T \qquad (1)$$

[0037] 여기서, H^T는 H 매트릭스의 전치(transpose)이다. z가 0 벡터인 경우에, 벡터($\hat{x}$)는 **H**에 의해 정의된 모든 패리티 검사 제약들을 만족시키는데, 즉, $\hat{x}$는 유효 코드워드이다. 상기 경우에, 프로세싱은 주기적 리던던시 검사(Cyclic-Redundancy Check: CRC) 단계(318)로 진행한다.

[0038] 대신에, z가 0 벡터가 아닌 경우에, 벡터($\hat{x}$)는 패리티 검사 제약들 중 하나 이상을 만족시키지 못하는데, 이는 전형적으로 불만족된 검사 노드들 또는 USC들이라고 칭해진다. 0 스칼라 값(scalar value)들이 아닌 신드롬 벡터(z) 내의 요소들의 수는 벡터($\hat{x}$) 내의 USC들의 수(b)이다. 또한, 신드롬 벡터(z)의 비-제로 스칼라 요소들의 인덱스들은 벡터($\hat{x}$) 내의 USC들의 인덱스들이다.

[0039] 벡터($\hat{x}$)가 신드롬 검사(306)에 실패한 경우에, 프로세싱은 하나 이상의 디코딩 반복들(308) 중 제 1 디코딩 반복으로 진행한다. 디코딩 반복(308)은 3개의 단계들: 1) 신뢰-전파 검사-노드 업데이트 단계(310), 2) 신뢰-전파 비트-노드 업데이트 단계(312), 및 3) 단계 306과 동일한 신드롬 검사 단계(314)를 포함한다.

[0040] 신뢰-전파 검사-노드 업데이트 단계(310)에서, 각각의 검사 노드(m)는 다음 식들 2, 3 및 4에 따라 R_{mn}으로 표시된 메시지들을 계산하기 위하여 세트 N(m) 내의 모든 비트 노드들 n으로부터 수신된 Q_{nm} 메시지들을 사용하며:

[0041] $$R_{mn}^{(i)} = \delta_{mn}^{(i)} \max\left(\kappa_{mn}^{(i)} - \beta, 0\right) \quad (2)$$

[0042] $$\kappa_{mn}^{(i)} = \left|R_{mn}^{(i)}\right| = \min_{n' \in N(m) \backslash n} \left|Q_{n'm}^{(i-1)}\right| \quad (3)$$

[0043] $$\delta_{mn}^{(i)} = \left(\prod_{n' \in N(m) \backslash n} \operatorname{sgn}\left(Q_{n'm}^{(i-1)}\right) \right) \quad (4)$$

[0044] 여기서, i는 디코딩 반복이고, N(m)\n은 비트 노드(n)를 제외한 세트(N(m))이며, β는 그 값이 코드 파라미터들에 따르는 양의 상수이다. 그 후, 계산된 R_{mn} 메시지들이 동일한 에지들을 따라 세트(N(m)) 내의 모든 비트 노드들 n으로 역송신된다. Q_{nm} 메시지들과 마찬가지로, R_{mn} 메시지들은 LLR들이다.

[0045] 다음으로, 신뢰-전파 비트-노드 업데이트 단계(312)에서, 각각의 비트 노드(n)는 다음 식 5에 따라 Q_{nm} 메시지들을 계산하며:

[0046] $$Q_{nm}^{(i)} = L_n^{(0)} + \sum_{m' \in M(n) \backslash m} R_{m'n}^{(i)} \quad (5)$$

[0047] 여기서, $L_n^{(0)}$ 은 비트 노드(n)에 대한 L_{ch} 값이고, M(n)\m은 검사 노드(m)를 배제한 세트(M(n))이다. 그 후, 비트 노드(n)가 계산된 Q_{nm} 메시지를 세트(M(n)) 내의 모든 검사 노드들(m)로 송신한다.

[0048] 비트-노드 업데이트 단계(312) 동안에도, 각각의 비트 노드(n)는 다음 식들 6 및 7에 따라 자신의 $\hat{x}_n$ 값을 업데이트한다:

[0049] $$E_n^{(i)} = \sum_{m' \in M(n)} R_{mn}^{(i)} \quad (6)$$

[0050] $$P_n = L_n^{(0)} + E_n^{(i)} \quad (7)$$

[0051] $P_n \geq 0$인 경우에, $\hat{x}_n = 0$이고, $P_n < 0$인 경우에, $\hat{x}_n = 1$이다. 식 6에 의해 발생된 값들은 E-값들 또는 E_{LDPC} 값들이라고도 칭해진다. 전형적으로, E_{LDPC} 값들은 터보-디코딩으로서 공지된 튜닝 프로세스의 부분으로서 판독 프로세서(예를 들어, 도 1의 판독 프로세서(110))로 역송신된다. 식 7에 의해 발생된 값들은 P-값들이라고 칭해진다. 식들 2 내지 7에 의해 표현된 특정 신뢰-전파 알고리즘은 최소-합 알고리즘으로서 공지되어 있다.

[0052] $\hat{x}_n$가 각각의 디코딩 반복(308) 동안 업데이트되고 디코딩 프로세스(300)에 의해 최종적으로 출력된다는 점을 주의하라. 원래 LLR 값들(L_{ch})은 디코딩 프로세스(300) 동안 변화되지 않은 채로 유지된다. 즉, 각각의 디코딩 반복(308) 동안, 각각의 비트 노드(n)는 자신이 검사 노드(m)를 통하여 관련되는 모든 다른 비트 노드들(n)의 적절한 값으로 자신의 표(vote)를 행사한다. 예를 들어, 도 2(b)에서, 비트 노드 n_0는 검사 노드들 m_0 및 m_3와 관련된다. 그러므로, n_0는 검사 노드들 m_0 및 m_3와 관련된 비트 노드들, 즉, n_3, n_5, n_6, 및 n_7의 적절한 값들로 자신의 표를 행사한다. 비트 노드(n)의 L_{ch} 값의 매그니튜드 값이 크면 클수록(즉, 신뢰도가 크면 클수록), 더 많은 비트 노드(n)의 표가 카운팅된다. 이 표-행사의 순 효과(net effect)는 낮은 L_{ch} 매그니튜드 값(즉, 신뢰도)을 갖는 비트 노드의 $\hat{x}_n$ 값이 변화하고 상기 비트 노드가 관련되는 높은-신뢰도 비트 노드의 신뢰성들에 따를 것이라는 점이다. 즉, 비트 노드의 L_{ch} 값이 에러 있는 경-판정 값 및 낮은 매그니튜드를 갖는 경우에, 다른 비트 노드들의 결합된 표들이 하나 이상의 반복들 이후에, 상기 에러 있는 경-판정 값을 수정하는 경향이

있을 것이다.

[0053] 비트-노드 업데이트 단계(312)는 디코더의 현재 $\hat{x}_n$ 값들로 이루어지는 벡터($\hat{\boldsymbol{x}}$)를 신드롬 검사 단계(314)로 송신한다. 단계(314)의 신드롬 검사는 상술된 단계(306)의 신드롬 검사와 동일하다. 벡터($\hat{\boldsymbol{x}}$)가 신드롬 검사(314)를 통과하는 경우에, 상기 벡터($\hat{\boldsymbol{x}}$)는 CRC 단계(318)로 송신된다.

[0054] **LDPC 디코딩: 주기적 리던던시 검사 및 잘못 만족된 검사 노드들**

[0055] 신드롬 검사(306 또는 314)를 통과한다는 것은 단지 벡터($\hat{\boldsymbol{x}}$)가 유효 코드워드이지만, 반드시 디코딩된 정확한 코드워드(DCCW)는 아니라는 것을 의미한다. LDPC 디코더가 DCCW가 아닌 유효 코드워드를 생성하는 것이 가능하다. 그 경우에, 벡터($\hat{\boldsymbol{x}}$) 내에 USC들이 존재하는 것이 아니라, 잘못 만족된 검사 노드들(MSC)들이 존재한다. 따라서, 유효 벡터($\hat{\boldsymbol{x}}$)가 DCCW라는 것을 보장하기 위하여, 프로세스(300)는 벡터($\hat{\boldsymbol{x}}$)를 주기적 리던던시 검사(CRC)(318)로 통과시킨다. CRC 검사는 송신 또는 저장 동안 데이터의 변경을 검출할 수 있는 검사합 동작(checksum operation)이다.

[0056] 벡터($\hat{\boldsymbol{x}}$)가 CRC 검사를 통과하는 경우에, 상기 벡터($\hat{\boldsymbol{x}}$)는 DCCW이고, 프로세스(300)는 전역 변수(global variable) DCCW를 참으로 설정하고, 벡터($\hat{\boldsymbol{x}}$)를 출력하고, 단계(320)에서 종료된다. 그렇지 않은 경우에, 벡터($\hat{\boldsymbol{x}}$)는 DCCW가 아니며, 프로세스(300)는 전역 변수 DCCW를 거짓으로 설정하고, 벡터($\hat{\boldsymbol{x}}$)를 출력하고, 단계(320)에서 종료된다. 전역 변수 DCCW는 DCCW가 발생되었는지의 여부를 다른 디코딩 프로세스들(예를 들어, 후술되는 도 12의 TS-ROM 리스트 디코딩 프로세스(1206))에 통지한다.

[0057] 단계(314)로 리턴하면, 벡터($\hat{\boldsymbol{x}}$)가 신드롬 검사에 실패한 경우에, 상기 벡터($\hat{\boldsymbol{x}}$)는 여전히 하나 이상의 USC들을 포함한다. USC들을 분석하는 전형적인 방법은 또 다른 디코딩 반복(308)을 수행하는 것이다. 그러나, 적당한 시간 량에서 결코 불만족될 하나 이상의 USC들이 특정 벡터($\hat{\boldsymbol{x}}$) 내에 존재할 수 있다. 따라서, LDPC 디코더들은 전형적으로 자신들이 특정 벡터($\hat{\boldsymbol{x}}$) 상에서 얼마나 많은 디코딩 반복들을 수행할 수 있는지에서 제한된다. 최대 반복들의 수에 대한 전형적인 값들은 50으로부터 200까지의 범위이다.

[0058] 도 3에서, 단계(316)는 최대 반복 수에 도달하였는지의 여부를 결정한다. 그렇지 않은 경우에, 또 다른 디코딩 방법(308)이 수행된다. 그 대신에, 최대 반복 수에 도달한 경우에, 디코더 프로세스(300)는 실패하는데, 즉, 디코더는 "실패된 디코더"이다. 그 경우에, 프로세스(300)는 전역 변수 DCCW를 거짓으로 설정하고, 벡터($\hat{\boldsymbol{x}}$)를 출력하고, 단계(320)에서 종료된다.

[0059] 실패된 디코더의 벡터($\hat{\boldsymbol{x}}$)가 적은 수(예를 들어, 16개보다 더 적은)의 USC들을 포함하는 경우에, 상기 벡터($\hat{\boldsymbol{x}}$)는 니어 코드워드(Near CodeWord: NCW)라고 칭해진다. 실패된 디코더의 벡터($\hat{\boldsymbol{x}}$)가 적은 수(예를 들어, 15개보다 더 많은)의 USC들을 포함하는 경우에, 상기 벡터($\hat{\boldsymbol{x}}$)는 무효 코드워드(Invalid CodeWord: ICW)라고 칭해진다.

[0060] 실패된 디코딩 프로세스를 핸들링(handling)하는 2개의 전형적인 방법들은 1) 대응하는 데이터의 재송신을 요구하거나 또는 2) 하나 이상의 포스트-프로세싱(Post-Processing: PP) 방법들로 벡터($\hat{\boldsymbol{x}}$)를 통과시키는 것이다. 전형적으로, 벡터($\hat{\boldsymbol{x}}$) 내의 USC들의 수(b)는 이러한 2개의 방법들 중 어느 것이 사용될지를 나타낸다. 더 큰(예를 들어, 16보다 더 큰) b는 전형적으로 재송신 또는 다른 포스트-프로세싱 방법에 의해 핸들링되는 반면, 작은 b 값들은 에러-플로어 완화 포스트-프로세싱 방법들에 의해 핸들링된다.

[0061] **BER, SNR, 및 에러 플로어들**

[0062] LDPC 디코더의 비트-에러 레이트(BER)는 얼마나 많은 에러 있는 디코딩된 비트들이 x개의 프로세싱된 비트들에 대해 발생될 것인지를 나타내는 비율이다. 따라서, 예를 들어, 10^{-9}의 BER을 갖는 디코더는 평균적으로 프로세싱된 10억 비트마다 하나의 에러 있는 비트를 발생시킬 것이다. BER이 작으면 작을수록, 디코더가 더 양호해진다. LDPC 디코더의 BER은 디코더가 실패할 때 증가(악화)되는데, 즉, 디코딩된 정확한 코드워드(DCCW)에 수렴함이

없이 종료된다.

[0063] LDPC 디코더의 BER은 디코더의 입력 신호의 신호-대-잡음 비(SNR)에 의해 강하게 영향을 받는다. SNR의 함수로서의 BER의 그래프는 전형적으로 2개의 다른 영역들: SNR에서의 유닛 증가가 제공되면 BER이 신속하게 개선(감소)되는 초기 "워터폴(waterfall)" 영역, 및 SNR에서의 유닛 증가들이 단지 BER에서의 적당한 개선들을 발생시키는 다음의 "에러 플로어" 영역을 포함한다. 따라서, 에러 플로어 영역에서의 상당한 BER 개선들을 성취하는 것은 SNR 증가 이외의 방법들을 필요로 한다.

[0064] LDPC 디코딩의 에러-플로어 특성들을 개선시키는 하나의 방법은 코드워드 길이를 증가시키는 것이다. 그러나, 코드워드 길이를 증가시키는 것은 또한 LDPC 디코딩에 필요한 메모리 및 다른 계산 자원들을 증가시킨다. 따라서, 전형적으로 HD 드라이브들 상에 판독-채널 디바이스들을 갖는 경우와 같이, 이와 같은 자원들이 엄격하게 제한되는 경우에, 필요한 에러-플로어 개선을 발생시키기 위하여 다른 방법들이 발견되어야 한다.

[0065] 또 다른 부족한 자원은 프로세싱 사이클들이다. 전형적으로, 지정된 처리량을 성취하기 위하여, HD 드라이브는 코드워드를 디코딩하는데 고정된 수의 판독-채널 프로세싱 사이클들의 예산을 세운다. 상기 예산을 초과하는 방법들(즉, 오프-더-플라이 방법(off-the-fly method)들)은 처리량을 감소시킨다. 클록-사이클 할당 내에서 DCCW를 복구하므로 처리량을 감소시키지 않는 온-더-플라이 방법(on-the-fly method)들이 더 바람직하다.

[0066] **트래핑 세트들 및 지배적인 트래핑 세트들(dominant trapping sets)**

[0067] (a, b) 트래핑 세트는 디코더가 최대 수의 반복들을 내에서 만족시킬 수 없는 b개의 USC들의 세트이며, a개의 에러 있는 비트 노드(Erroneous Bit Node: EBN)들은 이러한 USC들과 관련된다. 트래핑 세트들의 대부분은 5개보다 더 적은 USC들 또는 10개보다 더 적은 EBN들을 포함한다. 트래핑 세트들은 LDPC 디코더의 에러-플로어 특성들에 상당한 영향을 미치는데, 즉, LDPC 디코더가 DCCW에 수렴하지 못할 때, 이것은 종종 트래핑 세트 때문이다.

[0068] LDPC 디코더의 에러-플로어 특성들을 개선시키는 하나의 방식은 (i) 실패된 디코더의 $\hat{x}$ 벡터 내의 USC들을 조사하고, (만약 있다면) 트래핑 세트들을 식별하고, (ii) 그러한 USC들과 관련된 EBN들을 식별하고, (iii) 그러한 트래핑 세트들과 관련된 하나 이상의 EBN들을 플립(flip)하고, (iv) 디코더를 재시작하는 것이다. 하나의 가능한 구현예에서, LDPC 디코더가 막 초기화된 경우에, 즉 디코더가 상태 0인 경우에, EBN을 플립하는 것은 (i) 상기 EBN의 L_{ch} 값의 경-판정 값을 반전하고, 즉, 1이 0이 되고, 역도 또한 마찬가지이며, (ii) 상기 동일한 L_{ch} 값의 매그니튜드 비트들, 즉, 신뢰도를 최대, 예를 들어, 모두 1들로 설정하는 것을 포함한다. 디코더가 상태 0 이외의 어떤 상태인 경우에, EBN을 플립하는 것은 (i) (상기 식 7에 의해 정의된) EBN의 P값의 경-판정 값을 결정하고, (ii) 상기 EBN의 L_{ch} 값, P-값 및 모든 관련된 Q_{nm} LLR들의 경-판정 값들을 단계(i)의 경-판정 값의 반대로 설정하고, (iii) 상기 EBN의 L_{ch} 값, P-값 및 모든 관련된 Q_{nm} LLR들의 매그니튜드 비트들을 최대로 설정하는 것을 포함한다. 종종, 하나 또는 2개의 EBN들을 플립하는 것은 트래핑 세트를 "파괴할" 것이며, 재시작된 디코더가 DCCW에 수렴할 것이다.

[0069] 상이한 트래핑 세트들은 파괴될 때, 에러-플로어 특성들에서의 상이한 개선들을 발생시킬 것이다. 지배적인 트래핑 세트(DTS)들은 트래핑 세트들의 최소 세트를 나타내며, 상기 세트의 파괴는 BER/에러-플로어 특성들에서의 지정된 개선을 발생시킨다. 예를 들어, DTS-1은 BER에서의 1차 매그니튜드 개선, 예를 들어, 10^{-9}으로부터 10^{-10}으로의 개선을 발생시키는 트래핑 세트들의 최소 세트를 나타내는 반면, DTS-3은 BER에서의 3차 매그니튜드 개선, 예를 들어, 10^{-10}으로부터 10^{-13}으로의 개선을 발생시킬 것이다.

[0070] **니어 코드워드들의 리스트 디코딩**

[0071] 리스트 디코딩은 트래핑 세트들을 검출 및 파괴하는 하나의 포스트-프로세싱 방법이다. 리스트 디코딩에서, 벡터($\hat{x}$) 내의 관측된 트래핑 세트는 공지된 트래핑 세트들의 리스트 또는 리스트들에 대해 매칭된다. 트래핑-세트 리스트는 전형적으로 상기 리스트 내의 각각의 트래핑 세트 내의 모든 USC들의 인덱스들 및 그러한 USC들과 관련된 하나 이상의 EBN들의 인덱스들을 포함한다. 관측된 트래핑 세트와 매칭하는 트래핑 세트가 리스트에서 발견되는 경우에, EBN 인덱스 값(들)이 상기 리스트로부터 검색된다. 그 후, 그러한 비트 노드들이 플립되고, 도 3의 디코딩 프로세스(300)는 재시작된다.

[0072] **트래핑-세트 시뮬레이션**

[0073] 리스트 디코딩에 필요한 트래핑-세트 리스트는 전형적으로 소프트웨어 및 하드웨어 시뮬레이션 툴들을 사용하여 오프라인으로 생성된다. 도 4는 트래핑 세트들을 식별하고 이러한 트래핑 세트들에 관한 다양한 정보를 기록하는 오프-라인 트래핑-세트(TS) 시뮬레이션 툴(400)의 블록도이다. 툴(400)은 예를 들어, 필드-프로그래밍 가능한 게이트 어레이(Field-Programmable Gate Array: FPGA)에서 구현될 수 있다. LDPC 정확한 코드워드(CCW)(402)가 도 1의 잡음이 있는 채널(114)의 행동(behavior)을 에뮬레이팅(emulating)하는 채널 및 신호 모델(404)로 송신된다. 채널 및 신호 모델(404)은 L_{ch} 값들(406)을 LDPC 디코더(408)로 출력한다. LDPC 디코더(408)가 니어 코드워드(NCW)(410)를 발생시키는 경우에, NCW(410)가 신드롬 검사 모듈(412) 및 미스매칭 위치 레코더(mismatch location recorder)(414)로 송신된다. 신드롬 검사 모듈(412)은 NCW(410) 내의 모든 USC들의 인덱스들(416)을 출력한다. 미스매칭 위치 레코더(414)는 NCW(410)를 CCW(402)와 비교하고, NCW(410) 내의 모든 EBN들의 인덱스들(418)을 출력한다. USC 인덱스들(416) 플러스 DBN 인덱스들(418)은 트래핑-세트(TS) 정보(420)를 구성한다.

[0074] 소정의 LDPC 구현예에 대하여, 모든 가능한 트래핑 세트들은 수가 몇 백만에 달할 수 있다. 그러나, 상기 구현예의 에러 플로어의 상당한(즉, 1차 매그니튜드 이상의) 개선을 성취하는 것은 전형적으로 모든 가능한 트래핑 세트들의 서브셋, 즉, 지배적인 트래핑 세트(DTS)들만을 필요로 한다. 따라서, 오프-라인 TS 시뮬레이션 툴(400)은 입력으로서 TS 정보(420)를 취하고 DTS-N 정보(424)를 발생시키는 DTS-N 컴파일러(comfiler)(422)를 포함한다.

[0075] DTS-N 컴파일러(422)는 3-단계 프로세스: 수집, 순위화(ranking), 및 평가를 사용한다. 트래핑 세트 수집 방법은 트래핑 세트들을 검출하기 위하여 코드 그래프의 구조에 기초하여 결정론적인 잡음 임펄스들을 사용한다. 그 후, 수집된 트래핑 세트들은 거리-대-에러 경계(Distance-to-Error Boundary: DEB) 값들에 의해 순위화되고, 여기서 낮은 DEB 값들을 갖는 트래핑 세트들은 에러 플로어에 더 많이 기여한다. 그 후, 트래핑 세트들을 평가하고 예측된 순위들을 확인하기 위하여 중요도 샘플링(importance sampling)이 사용된다.

[0076] 실제로, 이와 같은 FPGA-기반 오프라인 시뮬레이션들은 수행하는데 1년까지 걸릴 수 있다. 예를 들어, 4Gb/s HD 드라이브에 대해 10^{-15}의 BER을 산출하는 트래핑 세트들을 식별하는 것은 대략 289일 동안 오프라인-시뮬레이션 툴(예를 들어, 도 4의 툴(400))을 실행하는 것을 필요로 한다. 전형적으로, 이 시간 제약은 HD 드라이브 판독 채널의 최종적인 설계 및 칩들의 대량-제조 사이에 종종 1 내지 2년의 지연이 존재하기 때문에, 문제가 되지 않는다.

[0077] **트래핑 세트 판독-전용 메모리(TS-ROM)**

[0078] 따라서, 도 4의 오프-라인 TS 시뮬레이션 툴을 사용하면, 특정 LDPC 구현예에 대한 에러-플로어 특성들에서 개선들을 산출할 하나 이상의 트래핑 세트들 또는 지배적인 트래핑 세트들을 식별하는 것이 연역적으로 가능하다. 런-타임 환경(run-time environment)에서 리스트 디코딩을 구현하는 하나의 방식은 트래핑-세트 판독-전용 메모리(TS-ROM)에 도 4의 오프라인으로 발생된 트래핑-세트 정보(420)를 저장하고, 상기 TS-ROM을 리스트 디코더 프로그램에 결합하는 것이다. TS-ROM 리스트 디코더 프로그램은 $\hat{x}$ 내의 관측된 USC들을 TS-ROM에 저장된 트래핑 세트들과 비교한다. 매칭이 발견되는 경우에, TS-ROM 리스트 디코더 프로그램은 LDPC 디코더 내의 적절한 비트-노드 값들을 플립하고, 디코더를 재시작한다.

[0079] 전형적으로, TS-ROM 정보는 단독으로- 또는 이중으로-링크(link)된 리스트 내에 랜덤으로 저장되며, 상기 리스트는 무작정 순차적 탐색(brute-force sequential search)을 사용하여 탐색된다. 전형적으로, 각각의 (a, b) 트래핑 세트는 TS-ROM 리스트에서 (2+a+b) 레코드들을 점유한다. 따라서, (4,4) 트래핑-세트 프로파일(즉, 4개의 USC들 및 4개의 EBN들)에 대하여, 10개의 레코드들이 존재할 것이다: 4개의 USC들이 존재한다는 것을 나타내는 하나의 레코드, 그 다음의 4개의 개별적인 USC 레코드들, 4개의 EBN들이 존재한다는 것을 나타내는 그 다음 레코드, 그 다음의 4개의 개별적인 EBN 레코드들. 전형적인 TS-ROM 리스트 구현예는 100개의 트래핑 세트들을 저장하고, 잘못 만족된 검사 노드들에 관한 임의의 정보를 저장하지 않는다.

[0080] 경제적으로 실용적인 이와 같은 TS-ROM 구현예에 대하여, 단일 TS-ROM이 다수의 구현예들에서 필요한 에러-플로어 개선을 성취할 수 있어야 한다. 그러나, 트래핑 세트들은 동일한 LDPC 코드가 구현될 때에도 구현예마다 변화한다. 예를 들어, 2개의 HD 드라이브들 상에서 사용된 LDPC 코드가 동일할지라도, HD 드라이브들과 관련된 트래핑 세트들은 상이할 수 있다. 구체적으로는, 트래핑 세트들이 HD 드라이브의 지터 프로파일(jitter profile), 심볼간 간섭 특성들, 및 펄스-쉐이핑(pulse-shaping) 방식들에 의해 영향을 받는다는 것이 연구를 통해 제시되었다. 이러한 팩터들은 상이한 제조자들의 HD 드라이브들 사이 뿐만 아니라, 동일한 제조자로부터의 상이한 HD

드라이브 모델들 사이에서 변화할 수 있고, 심지어 동일한 모델의 상이한 실운용(production run)들 사이에서도 변화할 수 있다. 따라서, 트래핑 세트들은 2개의 동일한-모델 하드 드라이브들 사이에서도 변화할 수 있다. 그렇게 많은 상이한 HD 드라이브들의 LDPC 트래핑 세트들을 시뮬레이팅하는 것은 비실용적이다. 그러나, 많은 종류의 HD 드라이브들에 공통인 그러한 트래핑 세트들로만 로딩된 TS-ROM은 특정 HD 드라이브와 쌍을 이룰 때 필요한 레벨의 에러-플로어 개선을 발생시키지 못할 수 있다.

[0081] TS-ROM의 성능을 개선시키는 하나의 방법은 FPGA-기반 오프라인-시뮬레이션 툴(예를 들어, 도 4의 툴(400))에 의해 발생된 정보를 제조된 디바이스의 테스트 모델들로부터 획득된 결과들로 보충하는 것이다. 전형적으로, 일단 회로 설계가 완성되었다면, 대량 생산이 개시되기 전에 상기 설계의 제한된 수의 테스트 모델들이 테스트를 위해 제조 및 분배될 것이다. 특정 HD 드라이브 구현예에 대해 10^{-15}의 BER을 산출할 트래핑 세트들을 결정하는 데 1년이 걸릴 수 있지만, 10^{-12}의 BER을 산출할 트래핑 세트들을 결정하는데에는 단지 하루가 걸린다. 따라서, 테스트 모델들은 제한된 시간 기간 동안 LDPC 테스트 모드에서 실행되고, 임의의 발견된 트래핑 세트들이 저장된다. TS-ROM 내에 이미 존재하지 않는 임의의 발견된 트래핑 세트들이 TS-ROM에 추가된다. 소비자들에게 분배될 실제 디바이스를 사용함으로써, 이 방법은 FPGA-기반 오프라인-시뮬레이션 툴(도 4의 툴(400))을 피할 수 있는 트래핑 세트들을 캡처(capture)한다.

[0082] **트래핑 세트 랜덤-액세스 메모리(TS-RAM)**

[0083] TS-ROM의 정적 트래핑-세트 리스트에 대한 하나의 런-타임 대안은 트래핑-세트 랜덤-액세스 메모리(TS-RAM)에 트래핑-세트 정보를 저장하고, 도 4의 오프-라인 트래핑-세트 시뮬레이션 툴(400)을 실제의 개별적인 디바이스(예를 들어, HD 드라이브) 상에서 실행되는 런-타임 트래핑-세트 집합 및 분석 툴로 변화시키는 것이다. 채널 및 신호 모델(예를 들어, 도 4의 모델(404))로부터 초기 값들을 수신하는 대신에, 런-타임 툴은 상기 특정 디바이스의 실제 신호를 프로세싱할 것이다. 런-타임 툴은 리스트-디코더 기능을 포함할 것인데; 즉, 런-타임 툴은 관측된 USC들을 TS-RAM 내의 저장된 트래핑-세트 정보와 매칭하도록 시도하고, 매칭이 발견되는 경우에, 저장된 정보를 사용하여 디코더 비트-노드 값들을 변화시키고 디코더를 재시작할 것이다. 매칭이 발견되지 않는 경우에, 런-타임 툴은 관측된 트래핑 세트들을 분석할 것인데, 즉, USC들과 관련된 EBN들을 식별하고, 관측된 트래핑 세트가 TS-RAM 내의 저장에 대한 임계값 요건들(예를 들어, DTS-N 내의 멤버쉽(membership)을 만족시키는지를 결정한다.

[0084] 이론적으로, 상술된 TS-RAM 툴은 임의의 구현예의 트래핑-세트 프로파일에 적응할 수 있다. 실제로, 도 4의 오프-라인 시뮬레이션 툴(400)에 의해 수행된 트래핑 세트/지배적인 트래핑 세트 시뮬레이션은 계산적으로 복잡하다. 특히, 아마도 수 백만 개의 트래핑 세트들로부터 지배적인 트래핑 세트들을 구성하는 것이 특히 복잡하다. 이 복잡성은 상술된 TS-RAM 툴이 대부분의 HD 드라이브들에 적합하지 않도록 한다. 전형적으로, HD 드라이브들은 높은 레이트들(예를 들어, 초 당 4 기가바이트)로 데이터를 출력하며, 매우 낮은 BER/에러-플로어 레이트들(예를 들어, 10^{-13} 내지 10^{-15})을 요구하지만, 자신들의 펌웨어에서 적당한 계산 자원들만을 제공한다.

[0085] 더욱이, TS-RAM 툴은 도 4의 오프-라인 시뮬레이션 툴(400)과 마찬가지로, EBN 인덱스들을 생성하기 위하여 정확한 코드워드(CCW)를 필요로 한다. CCW들은 오프-라인 시뮬레이션 환경에서 용이하게 이용 가능하지만, 런-타임 환경에서는 그렇지 않다.

[0086] 본 발명의 일부 실시예들에 따르면, ROM 내의 저장된 트래핑-세트의 구성을 위해 방법들이 수행된다. 트래핑-세트 프로파일들은 우세도(dominance), 즉 LDPC 디코더의 에러-플로어 특성들에 대한 자신들의 영향에 의해 순위화된다. 더-지배적인 트래핑-세트 프로파일들은 불만족된 검사 노드(USC)들 및 잘못 만족된 검사 노드(MSC)들 둘 모두에 관한 정보를 포함하지만, 덜-지배적인 트래핑-세트 프로파일들은 USC들에 관한 정보만을 포함한다. 그 후, 트래핑-세트 프로파일 정보는 포인터-추적 탐색(pointer-chase search)을 사용하여 가장-지배적인 매칭하는 트래핑-세트 프로파일들의 고속 위치지정 및 검색을 허용하는 다수의 링크된 계층적 데이터 테이블들로 조직된다.

[0087] 본 발명의 일부 실시예들에 따르면, RAM 내의 지배적인 트래핑-세트들의 수집 및 식별을 위해 효율적인 런-타임 방법들이 수행된다. 새롭게-발견된 트래핑 세트들은 가능하다면 RAM에 저장되고 나서, 다음 팩터들: 트래핑 세트들이 최종적으로 매칭되었던 이래로 RAM이 탐색되었던 횟수, 트래핑 세트가 RAM에 추가되었던 이래로 트래핑 세트가 매칭되었던 총 횟수, 불만족된 검사 노드들의 수; 및 에러 있는 비트 노드들의 수 중 어느 하나 이상에 기초하여 분류 또는 순위화된다. 낮은-순위의 트래핑-세트 프로파일들은 새롭게-발견된 트래핑-세트 프로파일들에 대한 공간을 만들기 위하여 RAM으로부터 삭제된다. 따라서, 도 4의 DTS-N 컴파일러(422)에서 사용된 것과 같

이, 지배적인 트래핑 세트들의 연역적 식별을 위해 높은-계산적-복잡성의 오프라인 방법을 사용하는 것 이외에, 또는 그 대신에, 본 발명의 이러한 실시예들은 가능한 한 많은 새롭게-발견된 트래핑 세트들이 저장되고 비-지배적인 트래핑-세트 프로파일들이 주기적인 순위화 및 삭제에 의해 윈도우 아웃(window out)되는 낮은-계산적 복잡성의 귀납적 방법들을 수행한다.

[0088] 본 발명의 실시예들은 전형적으로, 온-더-플라이 방법들인데, 즉, 본 발명의 실시예들은 LDPC 디코딩에 대해 예산을 세운 클록-사이클들 내에서 DCCW를 복구할 수 있으므로, 시스템 처리량에 부정적인 영향을 미치지 않는다.

[0089] 도 5는 본 발명의 일 실시예에 따른 LDPC 디코딩 시스템(500)의 블록도이다. 도 1의 종래 기술의 HD 드라이브(100)와 유사한 본 발명의 HD 드라이브에서, 도 5의 LDPC 디코딩 시스템(500)은 도 1의 LDPC 디코더(112)와 유사한 LDPC 디코더의 부분으로서 구현될 것이다. 그 정도로, 도 5의 입력 L_{ch} 값들은 도 1의 디코더 입력 L_{ch} 값들과 유사하고, 도 5의 출력 $\hat{x}_{pp}$ 벡터는 도 1의 디코딩된 정보 워드와 유사하다.

[0090] LDPC 디코더(502)는 L_{ch} 값들을 수신하고, 도 3의 LDPC 디코딩 프로세스(300)를 수행하고, 포스트-프로세서(504) 및 TS-RAM 업데이터(506)로 벡터($\hat{x}$)를 출력한다. 포스트-프로세서(504)는 포스트-프로세싱 방법들, 예를 들어, TS-ROM 디코딩, TS-RAM 리스트 디코딩 등을 나타내는 하나 이상의 실행 가능한 프로그램들을 포함하는 메모리인 포스트-프로세싱(PP) 방법 리스트(508)에 연결된다. 포스트-프로세서(504)가 특정 PP 방법을 수행할 필요가 있는 경우에, 포스트-프로세서(504)는 PP 모듈 리스트(508)로부터 실행 가능한 프로그램을 판독하고, 상기 프로그램을 실행한다. 포스트-프로세서(504)는 임의의 수의 이러한 PP 방법들을 병렬로 또는 직렬로 수행할 수 있다. 포스트-프로세서(504)는 LDPC 디코딩 시스템(500)으로부터의 출력인 것 이외에, TS-RAM 업데이터(506)에도 송신되는 벡터($\hat{x}_{pp}$)를 출력한다.

[0091] **데이터 테이블들**

[0092] 실행 동안, 특정 PP 방법은 상기 PP 방법 실행 가능 프로그램 코드로부터 분리된 데이터 구조들에 액세스할 필요가 있다. 특히, TS-ROM 및 TS-RAM 리스트 디코딩 방법들은 TS-ROM(510) 및 TS-RAM(520)에 저장된 트래핑-세트 정보의 하나 이상의 리스트들에 액세스한다.

[0093] 도 5의 예시적인 실시예에서, TS-ROM은 4개의 테이블들: B-테이블(512), P-테이블(514), E-테이블(516), 및 EI-테이블(518)을 포함한다. TS-RAM(520)은 2개의 테이블: RAM P-테이블(522) 및 RAM 인덱스(524)를 포함한다. 테이블은 하나 이상의 동등한 크기의 로우들(레코드들) 및 하나 이상의 동등한 크기의 컬럼들(필드들)로 조직된 디지털 데이터의 2-차원 매트릭스이다. 테이블의 레코드들은 제로에서 시작하여 상부로부터 하부로 순서대로 넘버링된다. 이 넘버는 레코드 넘버이다.

[0094] P-테이블들(514 및 522)은 USC들 및 이들의 관련된 EBN들에 관한 정보를 포함한다. B-테이블(512)은 ROM-P 테이블(514)에 대한 포인터 정보를 포함한다. EI-테이블(518)은 MSC들에 관한 정보를 포함하고, E-테이블(516)은 EI-테이블(518)에 대한 포인터 정보를 포함한다. RAM 인덱스 테이블(524)은 RAM B-테이블(522)에 대한 포인터 정보를 포함한다.

[0095] 도 6은 도 5의 ROM P-테이블(514)의 예시적인 배치도이다. ROM P-테이블(514)은 트래핑-세트 프로파일 정보, 즉, USC 및 EBN 인덱스들을 포함한다. ROM-P 테이블(514)은 각각의 저장된 트래핑 세트의 각각의 USC에 대해 하나씩인 다수의 레코드들(로우들)을 포함한다. 레코드 넘버(602)는 0에서 시작하는, P-테이블(514) 내의 레코드의 순서적 위치이다.

[0096] ROM P-테이블(514) 내의 각각의 레코드는 3개의 필드들: LAYER(604), US_INDEX(606), 및 EBN_INDEX(608)를 포함한다. 일부 LDPC 디코더들은 달리는 계층으로서 공지된 한 세트의 업데이트 동작들을 병렬로 실행하도록 구성된다. LAYER(604)는 USC를 포함하는 디코딩 계층의 수를 표시한다. US_INDEX(606)는 USC의 인덱스를 포함한다. EBN_INDEX(608)는 USC와 관련된 하나 또는 2개의 EBN들의 인덱스들을 포함한다.

[0097] ROM P-테이블(514)은 우선 b(즉, $\hat{x}$ 내의 USC들의 수) 상에 분류되는데, 예를 들어, b = 2인 모든 트래핑 세트들이 먼저 오고, 모든 b = 3 트래핑 세트들이 다음에 오는 등이 된다. 따라서, b = 2 범위 내에 각각의 트래핑 세트에 대한 2개의 레코드들(예를 들어, 610, 612)이 존재할 것이며, 궁극적으로, b = 3인 트래핑 세트들에 대한 3-레코드 세트들(예를 들어, 614, 616), b = 4인 트래핑 세트들에 대한 4-레코드 세트들(618, 620) 등이 다음에 온다.

[0098] 각각의 b 범위 내에서, 트래핑 세트들은 우세도, 즉, 에러-플로어 특성들에 대해 트래핑 세트가 미치는 영향에 의해 분류된다. 에러-플로어 특성들에 상당한 영향을 미치는 그러한 트래핑 세트들은 b 범위의 초기에 발생하고, 더 적은 영향을 미치는 그러한 트래핑 세트들은 종단 부근에서 발생한다. 그 후, 특정 트래핑 세트에 대한 레코드들이 USC_INDEX(606)에 의해 분류된다.

[0099] 도 7은 도 5의 B-테이블(512)의 예시적인 배치도이다. B-테이블(512)은 ROM P-테이블(514) 내의 각각의 b 값에 대하여, ROM P-테이블(514) 내 및 E-테이블(516) 내의 상기 b 값의 첫 번째 발생들에 대한 포인터들 및 상기 b 값에 대한 E-테이블(516) 내의 레코드들의 수를 포함한다. 따라서, 각각의 b 값에 대해 B-테이블(512) 내에 단일 레코드가 존재하고, 여기서 b 값은 레코드 넘버(702)에 의해 지정된다. 그러나, 레코드 넘버들(702)은 0에서 시작하지만, b 값들은 전형적으로 2 또는 그 이상에서 시작한다. 따라서, 본 발명의 이 예시적인 실시예에서, 대응하는 b 값을 산출하기 위하여 레코드 넘버(702)에 오프셋(offset)이 추가된다. 예를 들어, $b \geq 2$인 트래핑 세트들만이 저장되는 경우에, 대응하는 b 값에 도달하기 위하여 각각의 레코드 넘버에 2의 오프셋이 추가될 것이다.

[0100] 필드 PTABLE_START_OFFSET(704)은 ROM P-테이블(514) 내의 특정 b 값의 첫 번째 발생의 위치를 포함한다. 필드 ETABLE_START_OFFSET(706)은 E-테이블(516) 내의 특정 b 값의 첫 번째 발생의 위치를 포함한다. 필드 NUM_ETABLE_ENTRIES(708)는 이 특정 b 값에 대한 E-테이블(516) 내의 레코드들의 수를 포함한다.

[0101] 도 8은 도 5의 E-테이블(516)의 예시적인 배치도이다. E-테이블(516)은 EI-테이블(518) 내의 MSC 레코드들에 대한 포인터들을 포함한다. E-테이블(516) 내의 각각의 레코드는 레코드 넘버(802), EITABLE_START_ADDRESS 필드(804), 및 EITABLE_END_ADDRESS 필드(806)를 갖는다. EITABLE_START_ADDRESS 필드(804)는 EI-테이블(518) 내의 대응하는 데이터의 첫 번째 발생에 대한 포인터를 포함하고, EITABLE_END_ADDRESS 필드(806)는 EI-테이블(518) 내의 대응하는 데이터의 최종적인 발생에 대한 포인터를 포함한다.

[0102] 도 9는 도 5의 EI-테이블(518)의 예시적인 배치도이다. EI-테이블(518)은 MSC들과 관련된 EBN들의 인덱스들을 저장한다. EI-테이블(518) 내의 각각의 레코드는 레코드 넘버(902) 및 2개의 필드들: BLOCK_COLUMN 필드(904) 및 B_INDEX 필드(906)를 포함한다. BLOCK_COLUMN 필드(904)는 EBN이 위치되는 블록 컬럼을 표시하고, B_INDEX 필드(906)는 EBN의 인덱스이다.

[0103] 도 10은 도 5의 RAM P-테이블(522)의 예시적인 배치도이다. RAM P-테이블(522)은 ROM P-테이블(514)에서 발견되지 않은 새롭게-식별된 트래핑 세트들의 프로파일들을 저장한다. RAM P-테이블(522) 내의 각각의 로우(즉, 레코드)는 레코드 넘버(1002) 및 2개의 필드들: 2-비트 TAG 필드(1004) 및 R_WORD 필드(1006)를 포함한다. TAG 필드(1004)의 4개의 가능한 값들은 R_WORD 필드(1006) 내의 데이터의 구조 및 레코드 유형을 표시한다. TAG 필드(1004)가 11의 값을 갖는 경우에, 레코드는 전체 트래핑 세트에 속하는 정보를 포함하는 1차 레코드이다. TAG 필드(1004)가 10의 값을 갖는 경우에, 레코드는 2차 레코드이고, 트래핑-세트 프로파일 내의 특정 USC에 속하는 정보를 포함한다. TAG 필드(1004)가 00 또는 01의 값을 갖는 경우에, R_WORD 필드(1006)는 비어 있는데, 즉, 이 레코드는 새롭게-식별된 트래핑 세트들에 대한 프로파일 정보를 저장하는데 이용 가능하다. 트래핑-세트 프로파일은 전형적으로 b 2차 레코드들보다 앞서는 단일 1차 레코드를 포함한다.

[0104] 레코드가 1차 레코드인 경우에, R_WORD 필드(1006)는 서브-필드들: (i) b-값 서브필드(1008), (ii) 트래핑 세트 EBN들의 수를 기록하는 a-값 서브필드(1010), (iii) 이 트래핑 세트와 최종적으로 매칭되는 TS-RAM 탐색의 수를 표시하는 LAST_HIT_NUM 서브필드(1012), 및 (iv) 이 특정 트래핑-세트 프로파일이 이 트래핑 세트가 TS-RAM에 저장되었던 이래로 관측된 트래핑 세트와 얼마나 많은 횟수로 매칭되었는지를 기록하는 HIT_COUNTER 서브필드(1014)를 포함한다.

[0105] 레코드가 2차 레코드인 경우에, R_WORD 필드(1006)는 트래핑 세트 내의 단일 USC의 계층(LAYER 필드 1016) 및 인덱스(USC_INDEX 필드 1018), 및 상기 USC와 관련된 하나 이상의 EBN들의 인덱스들(EBN_INDEX 필드 1020)을 포함한다.

[0106] 도 11은 도 5의 RAM 인덱스 테이블(524)의 예시적인 배치도이다. RAM P-테이블(522) 내의 각각의 트래핑 세트 프로파일에 대해 RAM 인덱스 테이블(524) 내의 레코드가 존재한다. RAM 인덱스 테이블(524) 내의 각각의 레코드는 단일 필드인 RAM_PTABLE_OFFSET(1102)를 포함한다. RAM_PTABLE_OFFSET(1102)는 RAM P-테이블 내의 특정 트래핑-세트 프로파일의 시작에 대한 포인터인데, 즉, RAM_PTABLE_OFFSET(1102)는 트래핑-세트 프로파일 1차 레코드의 도 10의 레코드 넘버(1002)를 포함한다. RAM 인덱스 테이블(524) 내의 레코드들은 우세도에 의해 분류되어, RAM 인덱스 테이블(524) 내의 제 1 레코드가 RAM P-테이블(522) 내의 가장-지배적인 트래핑 세트를

가리키고, RAM 인덱스 테이블(524) 내의 최종적인 레코드가 RAM P-테이블(522) 내의 최소-지배적인 레코드를 가리키게 된다.

[0107] 도 12는 도 5의 LDPC 디코딩 시스템(500)에 의해 사용된 예시적인 프로세스(1200)의 흐름도이다. 프로세싱은 단계(1202)에서 시작하고, 도 5의 LDPC 디코더(502)에 의해 L_{ch} 값들을 LDPC 디코딩하는 단계인 단계(1204)로 진행한다. 단계(1204)에서의 LDPC 디코딩이 DCCW를 산출하는 경우에, 프로세스(1200)는 단계(1218)에서 종료된다. 그렇지 않은 경우에, 프로세싱은 (도 5의 포스트-프로세서(504)에 의해 수행되는) TS-RAM 리스트 디코딩 단계(1206)로 진행한다.

[0108] 단계(1206)가 DCCW를 산출하는 경우에, 프로세스(1200)는 단계(1218)에서 종료된다. 그렇지 않은 경우에, 프로세싱은 (도 5의 포스트-프로세서(504)에 의해 수행되는) TS-RAM 리스트 디코딩 단계(1208)로 진행한다. 단계(1208)가 DCCW를 발생시키는 경우에, 프로세스(1200)는 단계(1218)에서 종료된다. 그렇지 않은 경우에, 프로세싱은 유사한 방식으로 동작하는 (도 5의 포스트-프로세서(504)에 의해 수행되는) 하나 이상의 추가적인 포스트-프로세싱 방법들(1210, 1212,...,1214)로 진행한다.

[0109] TS-RAM 리스트 디코딩(1208) 또는 추가적인 포스트-프로세싱 방법들(1210, 1212,...,1214) 중 어느 하나가 DCCW를 발생시키는 경우에, 프로세싱은 도 5의 TS-RAM(520)이 아마도 TS-RAM 업데이터(508)에 의해 업데이트되는 단계(1216)로 진행한다. 그 후, 프로세싱은 단계(1218)에서 종료된다.

[0110] 도 12는 포스트-프로세싱 방법들(1207 내지 1214)의 하나의 가능한 시퀀싱(sequencing)을 디스플레이한다. 포스트-프로세싱 방법들의 여러 시퀀스들이 다른 것들보다 더 실용적일지라도, 포스트-프로세싱 방법들의 거의 어느 한 시퀀스가 사용될 수 있다. 예를 들어, ROM에 이미 저장된 트래핑 세트들이 RAM 내에서 복제되지 않는 것을 보장하기 위하여 TS-RAM 리스트 디코딩 이전에 TS-ROM 리스트 디코딩을 시퀀싱하는 것이 바람직하다.

[0111] **TS-ROM 리스트 디코딩**

[0112] 도 13은 도 5의 포스트-프로세서(504)에 의해 구현되는 도 12의 예시적인 TS-ROM 리스트 디코딩 프로세스(1206)의 흐름도이다. 프로세싱은 단계(1302)에서 시작하고, 프로세스(1206)가 도 12의 LDPC 디코더(1204)로부터 수신된 벡터($\hat{x}$)에서 관측된 USC들의 수($b_{observed}$)가 프로세스(1206)에 의해 효율적으로 핸들링될 수 있는 USC들의 최대 수(b_{max})보다 더 큰지를 결정하는 단계(1304)로 진행한다. $b_{observed}$ = 0인 경우에, 벡터($\hat{x}$) 내에 USC들이 존재하지 않으므로(즉, $\hat{x}$는 니어-코드워드 오수정(near-codeword mis-correction)이다), 매칭하는 트래핑 세트가 존재하지 않는다. 단계(1304)가 거짓을 평가하는 경우에, 프로세스(1206)는 종료된다. 그렇지 않은 경우에, 프로세싱은 단계(1306)로 지속된다.

[0113] 단계(1306)에서, 디코더의 현재 상태가 저장되고, 상태 1로 라벨링(labelling)된다. 그 후, 프로세싱은 관측된 USC들이 우선 디코딩 계층에 의해 분류되고 나서, 인덱스에 의해 분류되는 단계(1308)로 지속된다. 다음으로, 단계(1310)에서, 다음의 4개의 값들이 도 5의 B-테이블(512)로부터 페칭(fetching)되어 저장된다:

[0114] (1) b = $b_{observed}$에 대한 도 7의 PTABLE_START_OFFSET 필드(704);

[0115] (2) b = $b_{observed}$+1에 대한 도 7의 PTABLE_START_OFFSET 필드(704);

[0116] (3) b = $b_{observed}$에 대한 도 7의 ETABLE_START_OFFSET 필드(706); 및

[0117] (4) b = $b_{observed}$에 대한 도 7의 NUM_ETABLE_ENTRIES 필드(708).

[0118] 제 1 값은 도 5의 P-테이블(514) 내의 매칭하는 트래핑-세트 정보(즉, USC 및 EBN 인덱스들)에 대한 이의 탐색을 시작하는 장소를 프로세스(1206)에 지시하고, 제 2 값은 이의 탐색을 종료하는 시간을 프로세스(1206)에 지시한다. 유사하게, 제 3 및 제 4 값들은 확장된 정보(즉, MSC 인덱스들)에 대한 이의 탐색을 시작 및 종료하는 장소를 프로세스(1206)에 지시한다.

[0119] 따라서, 예를 들어, $b_{observed}$ = 5인 경우에, 프로세스(1206)는 b = 5 및 b = 6에 대한 도 7의 PTABLE_START_OFFSET 필드(704)의 값들, 및 b = 5에 대한 도 7의 ETABLE_START_OFFSET 필드(706)와 도 7의 NUM_ETABLE_ENTRIES 필드(708)의 값들을 페칭한다.

[0120] 다음으로, 단계(1312)에서, 프로세스(1206)는 도 5의 P-테이블(514)을 선택하고, b = $b_{observed}$에 대한

PARTABLE_START_OFFSET의 저장된 값에 의해 표시된 어드레스로 진행한다. 다음으로, 단계(1314)에서, 관측된 USC들과 매칭하는 트래핑 세트에 대해 TS-ROM이 탐색된다.

[0121] 도 14는 도 13의 예시적인 TS-ROM 탐색 프로세스(1314)의 흐름도이다. 프로세스(1314)는 단계(1402)에서 시작되고, 단계(1404)에서, 관측된 USC들에 대한 동형 매칭(isomorphic match)인 도 5의 P-테이블(514) 내의 다음 레코드에 대해 탐색한다. 관측된 USC들의 특정 세트에 대한 동형 매칭은 USC들의 수 및 상기 USC들 사이의 거리들이 관측된 USC들과 동일한 트래핑 세트일 것이다. 따라서, 관측된 USC들이 [1,3,10]인 경우에, [1,3,10]은 매칭이고, [2,4,11]은 동형 매칭이며, [3,5,12], [4,6,13], 등도 마찬가지이다. 매칭이 발견되지 않는 경우에, 프로세스(1314)는 매칭(no match) 없는 상태(1406)로 종료된다.

[0122] 그 대신에, 단계(1404)에서 매칭이 발견되는 경우에, 단계(1408)에서, 매칭하는 P-테이블 레코드의 도 6의 EBN_INDEX 필드(608)의 값이 저장된다. EBN_INDEX 필드는 이 매칭된 트래핑 세트와 관련된 하나 및 아마도 2개의 에러 있는 비트 노드들의 인덱스들을 포함한다.

[0123] 다음으로, 프로세스(1314)는 임의의 확장된 정보, 즉, 이 매칭 트래핑 세트 내의 잘못 만족된 검사 노드(MSC)들과 관련된 EBN들의 인덱스들을 위치지정하도록 시도한다. 확장된 정보는 EI-테이블(518) 내에 유지된다. 그러나, 확장된 정보는 P-테이블(514)에 저장된 모든 트래핑 세트들에 대해 유지되는 것이 아니라, 각각의 b 범위 내의 트래핑 세트들의 서브셋들에 대해서만 유지된다. 상기 서브셋은 특정 b 범위 내의 더-지배적인 트래핑 세트들, 즉, 에러-플로어 특성들에 더-상당한 영향을 미치는 그러한 트래핑 세트들에 대응한다. 상술된 바와 같이, P-테이블(514)에서, 특정 b 범위 내의 트래핑 세트들은 우세도에 의해 분류되므로; 확장된 정보는 상기 b 범위 내의 제 1 x 레코드들에 대해서만 유지된다. EI-테이블(518) 내의 각각의 b 범위의 시작 및 종단은 B-테이블(512) 내의 필드들 ETABLE_START_OFFSET(706) 및 NUM_ETABLE_ENTRIES(708)에 의해 표시된다.

[0124] 프로세스(1314)는 자신이 ROM P-테이블(514)의 레코드들을 통하여 탐색할 때 트래핑 세트들의 내부 카운트를 유지한다. 따라서, 예를 들어, 프로세스(1314)가 도 6의 P-테이블(514) 내의 b = 2 트래핑 세트들을 통해 탐색하고 있는 경우에, 프로세스(1314)는 레코드들 0 및 1을 트래핑 세트 0(예를 들어, 도 6의 트래핑 세트(610))로서 식별하고, 레코드들 2 및 3을 트래핑 세트 1(예를 들어, 도 6의 트래핑 세트(612))로서 식별하고, 이런 형태로 진행된다. 이 트래핑-세트 넘버는 TSNUM이라고 칭해진다.

[0125] 단계(1410)에서, TSNUM은 도 13의 단계(1310)에 저장되었던 NUM_ETABLE_ENTRIES 필드의 값과 비교된다. TSNUM이 NUM_ETABLE_ENTRIES의 값보다 더 큰 경우에, 확장된 정보는 이용 가능하지 않고, 프로세스(1314)는 확장된 정보가 없는 매칭의 상태(1412)로 종료된다.

[0126] 그 대신에, 단계(1410)에서, TSNUM이 NUM_ETABLE_ENTRIES의 저장된 값보다 더 작거나 이와 동일하다는 것이 발견된 경우에, 이 매칭된 트래핑 세트에 대해 확장된 정보가 존재한다. 단계(1414)에서, 변수 ETABLE_ENTRY_ADDRESS를 발생시키기 위하여 TSNUM이 ETABLE_START_OFFSET의 저장된 값에 추가된다.

[0127] 다음으로, 단계(1416)에서, 프로세스(1314)는 도 5의 E-테이블(516)을 선택하고, 변수 ETABLE_ENTRY_ADDRESS의 값과 동일한 어드레스를 갖는 레코드로 진행하고, 도 8의 EITABLE_START_ADDRESS 필드(804) 및 도 8의 EITABLE_END_ADDRESS 필드(806)의 값들을 저장한다.

[0128] 다음으로, 단계(1418)에서, 프로세스(1314)는 도 5의 EI-테이블(518)을 선택하고, 저장된 EITABLE_START_ADDRESS 및 EITABLE_END_ADDRESS 값들 사이의 모든 레코드의 도 9의 BLOCK_COLUMN 필드들(904) 및 도 9의 B_INDEX 필드들(906)의 값들을 저장한다. 최종적으로, 프로세스(1314)는 확장된 정보를 갖는 매칭의 상태(1420)로 종료한다.

[0129] 도 13으로 리턴하면, 단계(1314)가 매칭 없는 상태 종료되는 경우에, 프로세스(1206)는 단계(1316)에서 종료된다.

[0130] 단계(1314)가 확장된 정보가 없는 매칭의 상태로 종료되는 경우에, 프로세스(1206)는 USC 인덱스들 및 이 트래핑 세트와 관련된 EBN 인덱스들 중 일부를 소유하지만, 확장된 정보(즉, MSC들과 관련된 EBN들의 인덱스들)을 소유하지 않는다. 이 경우에, 단계(1318)는 그러한 EBN 인덱스들에서 비트 노드들을 플립하고, 반복적인 LDPC 디코딩이 단계(1320)에서 수행된다. 단계(1320)의 프로세스는 디코더 초기화 단계(304) 및 초기 신드롬 검사 단계(306)가 스킵(skip)된다는 점을 제외하면, 도 3의 프로세스(300)와 동일하다. 단계(1320)가 DCCW에 수렴하는 경우에, 프로세스(1206)는 단계(1316)에서 종료된다. 그렇지 않은 경우에, 단계(1322)에서, 디코더는 상태 1로 복구되고, 단계(1314)에서, 다음의 매칭 트래핑 세트가 탐색된다.

[0131] 단계(1314)가 확장된 정보를 갖는 매칭 상태로 종료되는 경우에, 프로세스(1206)는 매칭된 트래핑 세트와 관련된 모든 EBN들의 인덱스들을 소유한다. 이 경우에, 신뢰 전파(도 3의 단계들(310 및 312))를 수행하는 것은 필요하지 않다. 그 대신에, 단계(1324)에서, EBN들이 플립되고, 결과적인 벡터($\hat{x}$)가 신드롬 검사(1326)를 받게 된다. 벡터($\hat{x}$)가 단계(1326)에서 신드롬 검사에 실패한 경우에, 프로세스(1206)는 단계(1322)로 진행한다. 그 대신에, 벡터($\hat{x}$)가 신드롬 검사를 통과한 경우에(즉, 벡터($\hat{x}$)가 유효 코드워드인 경우에), 단계(1328)에서, 상기 벡터 $\hat{x}$가 실제로 정확한 코드워드인지를 결정하기 위하여 벡터($\hat{x}$)에 대해 CRC 검사가 수행된다. 벡터($\hat{x}$)가 CRC 검사(1328)를 통과하는 경우에(즉, 벡터($\hat{x}$))가 DCCW인 경우에), 프로세스(1206)는 단계(1316)에서 종료된다. 벡터($\hat{x}$)가 CRC 검사(1328)에 실패한 경우에, 프로세스(1206)는 단계(1322)로 진행한다.

[0132] **TS-RAM 리스트 디코딩**

[0133] 도 5의 포스트-프로세서(504)에 의해 사용되는 또 다른 PP 방법은 도 12의 TS-RAM 리스트 디코딩(1208), 즉, 랜덤-액세스 메모리와 같은 휘발성 메모리에 저장된 트래핑-세트 정보를 사용한 트래핑 세트들의 리스트 디코딩이다. TS-RAM 리스트 디코딩은 도 5의 RAM P-테이블(522)이 선택된 트래핑 세트들에 대한 USC 및 EBN 정보를 저장한다는 점에서 TS-ROM 리스트 디코딩과 유사하다. 그러나, ROM P-테이블(514)과 달리, RAM P-테이블(522)은 도 5의 TS-RAM 업데이터(506)에 의해 런-타임 동안 변경된다. 그러므로, 단지 가장-중요한 정보, 예를 들어, USC 및 EBN 인덱스들이 저장된다. 확장된 정보(예를 들어, 도 5의 EI-테이블(518))는 TS-RAM 내에 유지되지 않는다.

[0134] RAM P-테이블(522) 내의 프로파일들도 임의의 방식으로 분류되지 않는다. 그 대신에, 도 16의 별도의 RAM 인덱스 테이블(524)이 우세도에 의해 분류된, RAM P-테이블(522)에 저장된 트래핑-세트 프로파일들의 어드레스들의 리스트를 유지한다.

[0135] 도 15는 도 12의 예시적인 TS-RAM 리스트 디코딩 프로세스(1208)의 흐름도이다. 프로세싱은 단계(1502)에서 시작되고, 도 13의 단계(1304)와 목적 및 동작이 동일한 단계(1504)로 진행한다. 단계(1504)가 거짓을 평가하는 경우에, 프로세스(1208)는 단계(1506)에서 종료된다; 그렇지 않은 경우에, 프로세싱은 현재 디코더 상태가 기록되고 상태 1로 라벨링되는 단계(1508)로 지속된다. 그 후, 프로세싱은 단계(1510)로 지속된다.

[0136] 단계(1510)에서, 프로세스(1208)는 도 5의 RAM P-테이블(522) 내의 가장-지배적인 트래핑-세트 프로파일로 진행한다. 구체적으로는, RAM 인덱스 테이블(524)이 우세도에 의해 RAM P-테이블(522) 내에 프로파일들을 순위화하기 때문에, 프로세스(1208)는 RAM 인덱스 테이블(524) 내의 제 1 레코드로 진행하고, 도 11의 RAM_PTABLE_OFFSET 필드(1102)의 값을 검색한다. 그 후, 프로세스(1208)는 RAM P-테이블(522) 내의 포인터를 상기 저장된 오프셋 값으로 이동시킨다.

[0137] 단계(1512)에서, 프로세스(1208)는 수행된 TS-RAM 탐색들의 총수, 즉, 프로세스(1208)가 실행되었던 총 횟수를 추적하는 전역 변수 RAM_SEARCH_COUNTER를 증분시킨다. 또한, 단계(1512)에서, 프로세스(1208)는 관측된 USC들에 대한 동형 정합에 대하여 RAM 인덱스 테이블(524)에 의해 표시된 순서로, 즉, 감소하는 우세도의 순서로 RAM P-테이블(522) 내의 프로파일들을 조사한다. 매칭이 발견되지 않는 경우에, 프로세싱은 단계(1514)로 지속된다.

[0138] 그 대신에, 단계(1512)에서, 매칭이 발견되는 경우에, 단계(1516)에서, 매칭된 프로파일의 도 10의 LAST_HIT_NUM 필드(1012)가 전역 변수 RAM_SEARCH_COUNT의 값으로 설정되고, 도 10의 HIT_COUNTER 필드(1014)가 1만큼 증분된다. 그 후, 단계(1518)에서, 도 10의 EBN_INDEX 필드들(1020)의 값들이 저장된다. 단계(1520)는 EBN_INDEX 값들에 위치된 비트 노드들을 플립하고, 단계(1522)에서, LDPC 디코딩이 수행된다. 단계(1522)는 도 13의 단계(1320)와 동일하다. 디코딩 프로세스(1522)가 DCCW에 수렴하는 경우에, 프로세싱은 단계(1524)로 지속되며; 그렇지 않은 경우에, 단계(1524)에서, 디코더가 상태 1로 리셋되고 나서, 프로세싱은 또 다른 동형 매칭이 P-테이블(522)에서 탐색되는 단계(1512)로 지속된다.

[0139] 단계(1514)에서, 프로세스(1208)는 도 11의 RAM 인덱스 테이블(524)을 업데이트한다. 구체적으로는, 단계(1514)는 RAM P-테이블(522) 내의 필드들, 예를 들어, LAST_HIT_NUM(1012), HIT_COUNTER 필드(1014), USC 노드 필드(1008)의 수, 및 EBN 필드(1010)의 수의 임의의 조합을 기반으로 하여 RAM P-테이블(522) 내의 모든 TS-RAM 프로파일들을 분류한다. 그 후, 모든 저장된 프로파일들의 어드레스들, 즉, 모든 1차 레코드들의 레코드 넘버들(1002)이 도 11의 RAM 인덱스 테이블(524) 내에 레코드들로서 저장된다. 프로파일 어드레스들은 (예를 들어, 가장 지배적인 것으로부터 최소 지배적인 것으로) 자신들이 단계(1514)에서 분류되었던 것과 동일한 순서로 RAM 인덱스 테이블(524)에 저장된다.

[0140] 일단 단계(1514)가 완료되면, 프로세싱은 단계(1506)에서 종료된다.

[0141] **TS-RAM 업데이터**

[0142] 도 12의 프로세스(1200)의 논의에서 설명된 바와 같이, TS-ROM 리스트 디코딩(1206) 이외의 임의의 포스트-프로세싱 방법이 DCCW에 도달하는 경우에, 이것은 새로운 트래핑 세트가 발견되었다는 것을 의미할 수 있다. 그러한 경우에, 단계(1216)는 상기 새로운 트래핑 세트를 도 5의 RAM P-테이블(522)에 추가하도록 시도할 수 있다.

[0143] 일 실시예에서, 단계(1216)는 TS-RAM(520) 내에 지배적인 트래핑 세트들을 유지하는 낮은-복잡성 프로세스이다. 구체적으로는, 이 실시예에서, 단계(1216)는 도 4의 DTS-N 컴파일러(422)에 의해 수행되는 계산들과 같이, 지배적인 트래핑 세트들을 연역적으로 결정하기 위한 소모적인 계산들을 수행하지 않고, 그 대신에, (i) 하나 이상의 팩터들, 예를 들어, 트래핑 세트가 최종적으로 매칭되었던 이래로 TS-RAM이 얼마나 많은 횟수로 탐색되었는지, 트래핑 세트가 매칭된 총 횟수, USC들의 수, EBN들의 수 등의 임의의 조합에 기초하여 TS-RAM 트래핑 세트들을 순위화하고 나서, (ii) 새롭게-발견된 트래핑 세트들을 위한 공간을 만들기 위하여 가장 낮은-순위의 트래핑 세트들을 제거한다. 이러한 팩터들을 사용하여 TS-RAM(520) 내의 트래핑 세트들을 순위화하는 것은 전형적으로 오프-라인 시뮬레이션 툴들(예를 들어, 도 4의 컴파일러(422))에 의해 수행된 분석보다 상당히 덜 복잡하다.

[0144] 도 16은 도 12의 예시적인 TS-RAM 업데이트 프로세스(1216)의 흐름도이다. 프로세싱은 단계(1602)에서 시작하고, DCCW가 도 12의 TS-RAM 리스트 디코딩 프로세스(1208)에 의해 발생되었는지가 결정되는 단계(1604)로 진행한다. 그러한 경우에, 트래핑-세트 프로파일은 TS-RAM에 추가될 필요가 없고, 프로세스(1216)는 단계(1608)에서 종료된다.

[0145] 그 대신에, 단계(1604)가 아니오/거짓을 평가하는 경우에, 이것은 TS_ROM 또는 TS-RAM 리스트 디코딩 이외의 어떤 포스트-프로세싱 방법이 DCCW에 도달하였으므로, 새로운 트래핑 세트가 발견되었고 RAM에 부가되어야 한다는 것을 의미한다. 단계(1610)에서, 트래핑-세트 프로파일이 생성된다. 트래핑-세트 프로파일은 트래핑-세트 USB들의 인덱스들, 및 이러한 USB들과 관련된 EBN들의 인덱스들을 포함한다. USB 인덱스들은 도 5의 LDPC 디코더(502)에 의해 이미 발생되었다. EBN 인덱스들을 생성하기 위하여, 단계(1610)는 포스트-프로세서(504)에 의해 발생된 DCCW($\hat{x}_{PP}$)를 LDPC 디코더(502)에 의해 생성된 벡터($\hat{x}$)와 비교한다.

[0146] 단계(1612)에서, 새로운 트래핑-세트 프로파일을 부가하는데 충분한 자유 공간이 존재하는지가 결정된다. 그러한 경우에, 단계(1614)에서, 새로운 트래핑-세트 프로파일이 RAM P-테이블(522)에 부가되고, 프로세스(1216)는 단계(1616)로 진행한다.

[0147] 그러나, 단계(1612)에서, 새로운 트래핑-세트 프로파일을 부가하는데 충분한 자유 공간이 RAM P-테이블(522) 내에 존재하지 않는 경우에, 단계(1618)에서, 가장 낮은-순위의 제거-적격 트래핑-세트 프로파일이 제거된다. 이 예에서, 프로파일의 제거-적격성(purge-eligibility)은 상기 프로파일이 최종적으로 매칭되었던 이래로 RAM이 탐색되었던 횟수, 즉, 프로파일의 LAST_HIT_NUM 필드의 값보다 더 적은 전역 변수 RAM_SEARCH_COUNT의 값에 의해 결정된다. 개재된 탐색들의 수가 지정된 임계값보다 더 큰 경우에, 프로파일은 제거-적격이다. 모든 프로파일들이 우선 제거-적격성에 의해 순위화된다고 가정하면, 모든 제거-적격 레코드들은 도 11의 RAM 인덱스 테이블(524)의 종단에 있을 것이다.

[0148] 따라서, 단계(1618)에서, RAM 인덱스 테이블(524) 내의 최종적인 레코드가 선택되고, RAM_PTABLE_OFFSET 필드(1102)의 값이 검색된다. (검색된 RAM_PTABLE_OFFSET 값에 의해 표시된) 저장된 오프셋 값에서의 RAM P-테이블(522) 내에 위치된 프로파일이 제거-적격인 경우에, RAM P-테이블(522) 내의 상기 저장된 오프셋 값에 위치된 1차 레코드 및 관련된 2차 레코드가 삭제된다. 단계(1620)에서, RAM 인덱스 테이블 MBS가 업데이트되고 나서, 제어는 새로운 트래핑-세트 프로파일을 부가하는데 충분한 자유 공간이 RAM P-테이블(522) 내에 존재하는지가 결정되는 단계(1612)로 루프 백(loop back)된다. 단계(1620)의 프로세싱은 도 15의 단계(1514)의 프로세싱과 동일하다.

[0149] 그 대신에, 단계(1618)에서, 가장 낮은-순위의 프로파일이 제거-적격이 아닌 경우에, 프로세싱은 단계(1616)로 지속된다. 단계(1616)에서, RAM 인덱스 테이블(523)이 업데이트되고, 프로세싱은 단계(1608)에서 지속된다. 단계(1616)의 프로세싱은 도 15의 단계(1514)의 프로세싱과 동일하다.

[0150] 본 발명이 LDPC 코딩 및 디코딩을 구현하는 하드 디스크들의 콘텍스트에서 설명되었을지라도, 본 발명의 그렇게 제한되지 않는다. 일반적으로, 본 발명은 LDPC 코딩 및 디코딩을 포함하는 임의의 적절한 통신 경로에서 구현될

수 있다.

[0151] 또한, 상기에 사용된 예시적인 신뢰-전파 알고리즘이 오프셋 최소-합 알고리즘(Offset Min-sum Algorithm: OMS)일지라도, 본 발명은 그렇게 제한되지 않고, 임의의 신뢰-전파 변형, 예를 들어, 섬-프로덕트 알고리즘(Sum-Product Algorithm: SPA) 또는 Bahl-Cocke-Jelinek-Raviv(BCJR) 알고리즘과 함께 사용될 수 있다.

[0152] 또한, 상기에 사용된 신뢰-전파 예가 단일 검사-노드 업데이트 단계 동안 모든 검사 노드들이 업데이트되고 나서, 모든 비트 노드들이 단일 비트-노드 업데이트 단계에서 업데이트되는 특정 디코딩 스케줄(플러딩 스케줄(flooding schedule))을 사용하였을지라도, 본 발명은 그렇게 제한되지 않고, 임의의 디코딩 스케줄, 예를 들어, 로우-시리얼 스케줄(row-serial schedule), 컬럼-시리얼 스케줄, 및 로우-컬럼 시리얼 시리얼 스케줄과 함께 사용될 수 있다.

[0153] 또한, 상기에 사용된 예시적인 LDPC 디코더가 비-계층화된 디코더였을지라도, 본 발명은 그렇게 제한되지 않고, 계층화된 디코더 및 비-계층화된 디코더 둘 모두와 함께 사용될 수 있다.

[0154] 또한, 상기에 제공된 예시적인 TS-RAM 구현예가 RAM 내의 트래핑-세트 프로파일들을 HD 드라이브의 판독 채널 내에 저장한다는 것을 가정하였을지라도, 본 발명은 그렇게 제한되지 않는다. RAM P-테이블(예를 들어, 도 5의 522)은 또한 HD 드라이브의 플래터들 상에 저장되거나 플래시 메모리와 같은 별도의 메모리에 저장될 수 있다.

[0155] 또한, 상기에 제공된 예시적 TS_ROM 구현예가 판독-전용 메모리의 콘텍스트에서 설명되어 있을지라도, 본 발명은 그렇게 제한되지 않는다. 일반적으로, 명세서 및 청구항들 둘 모두에서 사용되는 바와 같은 용어 "ROM"은 임의의 데이터-저장 디바이스 내의 데이터가 변경될 수 있든지 또는 없든지 간에, 정적 TS-프로파일 데이터를 저장하는 임의의 데이터-저장 디바이스를 칭하는 것으로 해석되어야 한다.

[0156] 또한, 본 발명의 실시예들이 LDPC 코드들의 콘텍스트에서 설명되었을지라도, 본 발명은 그렇게 제한되지 않는다. 본 발명의 실시예들은 그래프에 의해 정의될 수 있는 임의의 코드, 예를 들어, 토네이도 코드(tornado code)들, 구조화된 IRA 코드들에 대해 구현될 수 있는데, 그 이유는 상기 코드가 트래핑 세트들을 경험하는 그래프-정의된 코드들이기 때문이다.

[0157] 본 발명이 로그-우도 비들을 수신하는 것에 관하여 설명되었지만, 본 발명은 그렇게 제한되지 않는다. 우도 비들 또는 경-판정 값들과 같은 다른 소프트 값(soft value)들이 프로세싱되는 본 발명의 실시예들이 구상될 수 있다.

[0158] 본 발명은 이러한 방법들을 실행하는 방법들 및 장치들의 형태로 구현될 수 있다. 본 발명은 또한 자기 기록 매체들, 광 기록 매체들, 고체 상태 메모리, 플로피 디스켓들, CD-ROM들, 하드 드라이브들, 또는 임의의 다른 기계-판독 가능한 저장 매체와 같은 유형의 매체들 내에 구현되는 프로그램 코드의 형태로 구현될 수 있고, 상기 프로그램 코드가 컴퓨터와 같은 기계 내로 로딩되어 상기 기계에 의해 실행될 때, 상기 기계는 본 발명을 실행하는 장치가 된다. 본 발명은 또한 저장 매체에 저장되고, 기계 내로 로딩되고/되거나 기계에 의해 실행되든지, 또는 전기 배선 또는 케이블링(cabling), 광섬유들, 또는 전자기 방사와 같은 어떤 송신 매체 또는 캐리어를 통하여 송신되든지 간에 예를 들어, 프로그램 코드의 형태로 구현될 수 있고, 상기 프로그램 코드가 컴퓨터와 같은 기계 내로 로딩되어 상기 기계에 의해 실행될 때, 상기 기계는 본 발명을 실행하는 장치가 된다. 범용 프로세스 상에 구현될 때, 프로그램 코드 세그먼트(program code segment)들은 특정 논리 회로들과 유사하게 동작하는 특정한 디바이스를 제공하기 위하여 프로세스와 결합한다.

[0159] 명시적으로 다르게 진술되지 않는다면, 단어 "약" 또는 "대략"이 각각의 수적인 값 및 범위는 상기 값 또는 범위의 값 앞에 있는 것처럼 대략적인 것으로 해석되어야 한다.

[0160] 본 발명의 특징을 설명하기 위하여 설명 및 도시되었던 부분들의 세부사항들, 재료들 및 정렬들의 다양한 변화들이 다음의 청구항들에서 표현된 바와 같은 본 발명의 범위로부터 벗어남이 없이 당업자들에 의해 행해질 수 있다는 점이 또한 이해될 것이다.

[0161] 청구항들에서의 도면 번호들 및/또는 도면 참조 라벨들의 용도는 청구항들의 해석을 용이하게 하기 위하여 청구된 주제의 하나 이상의 가능한 실시예들을 식별하고자 하는 것이다. 이와 같은 용도는 이러한 청구항들의 범위를 반드시 대응하는 도면들에 도시된 실시예들로 제한하는 것으로 해석되어서는 안된다.

[0162] 본원에 설명된 예시적인 방법들의 단계들이 반드시 설명된 순서로 수행될 필요는 없다는 점이 이해되어야 하고, 이와 같은 방법들의 단계들의 순서가 단지 예시적이라는 점이 이해되어야 한다. 마찬가지로, 추가적인 단계들이 이와 같은 방법들에 포함될 수 있고, 일부 단계들이 본 발명의 다양한 실시예들에 따른 방법들에서 생략 또는

결합될 수 있다.

[0163] 만약 있다면, 다음의 방법 청구항들에서의 요소들이 대응하는 라벨링을 갖는 특정 시퀀스에서 인용될지라도, 청구항 인용들이 그러한 요소들 중 일부 또는 모두를 구현하는 특정 시퀀스를 의미하지 않는다면, 그러한 요소들은 반드시 상기 특정 시퀀스로 구현되는 것으로 제한되지는 않게 된다.

[0164] "하나의 실시예" 또는 "실시예"에 대한 본원의 언급은 상기 실시예와 관련하여 설명된 특정 특징, 구조 또는 특성이 본 발명의 적어도 하나의 실시예에 포함될 수 있다는 것을 의미한다. 명세서의 다양한 장소들에서의 "하나의 실시예에서" 구문의 출현들은 반드시 모두 동일한 실시예를 칭하지는 않고, 별도 또는 대안 실시예들이 반드시 다른 실시예들을 서로 배제하지도 않는다. 이것은 용어 "구현예"에도 적용된다.

부호의 설명

[0165] 500 : LDPC 디코딩 시스템 502 : LDPC 디코더

504 : 포스트-프로세서 506 : TS-RAM 업데이터

508 : 포스트-프로세싱(PP) 방법 리스트 510 : TS-ROM

512 : B-테이블 514 : P-테이블

516 : E-테이블 518 : EI-테이블

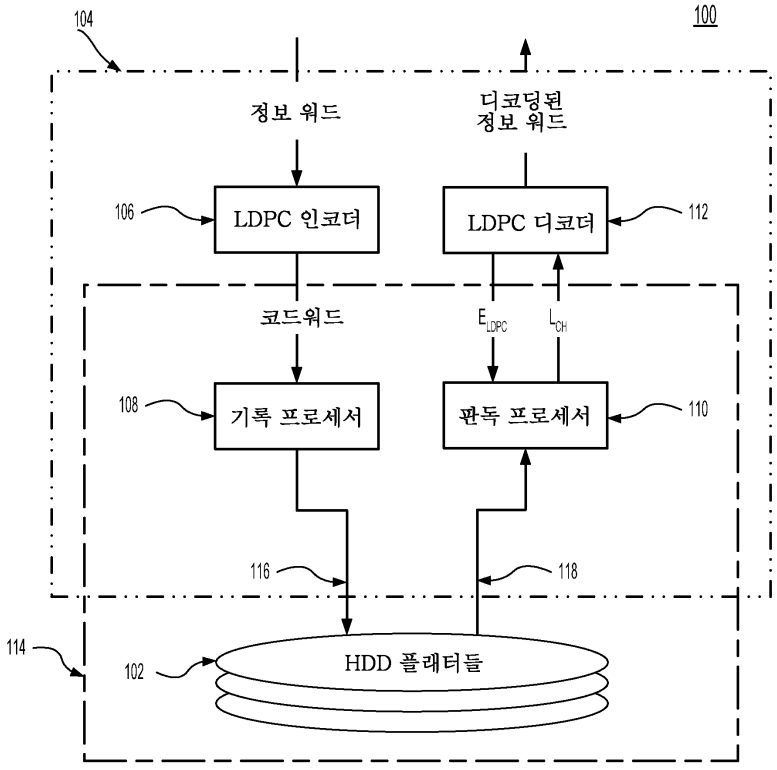
520 : TS-RAM 522 : RAM P-테이블

524 : RAM 인덱스 테이블

도면

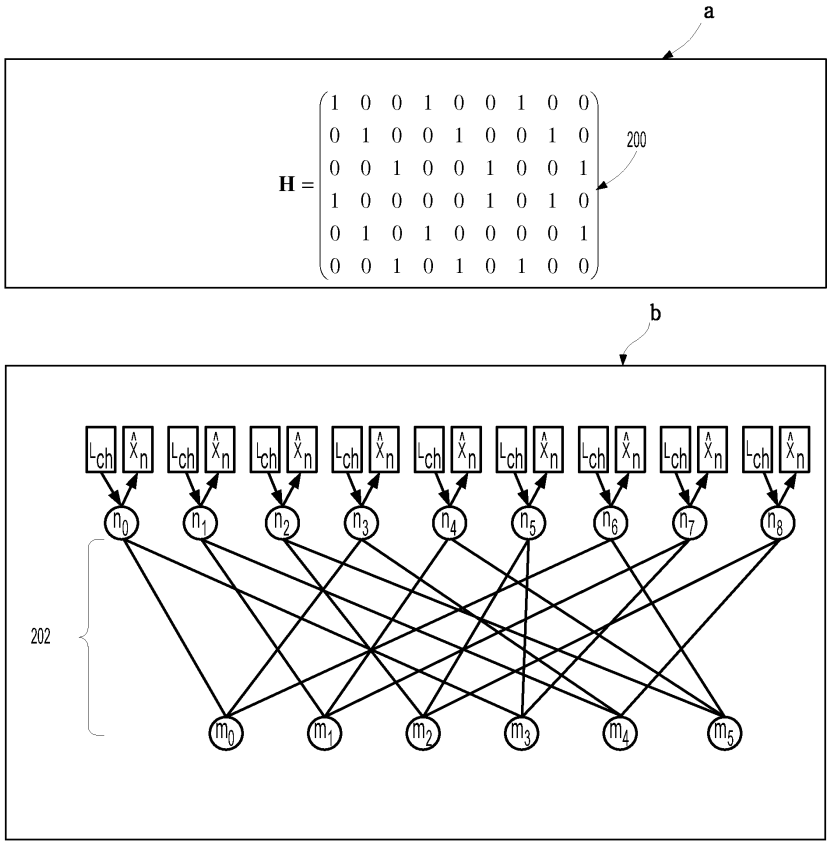
도면1

종래기술



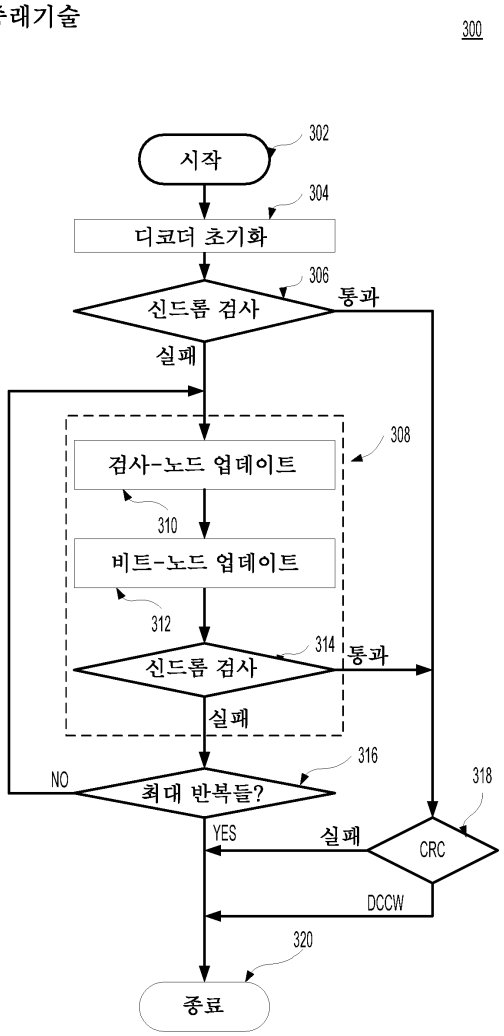
도면2

종래기술



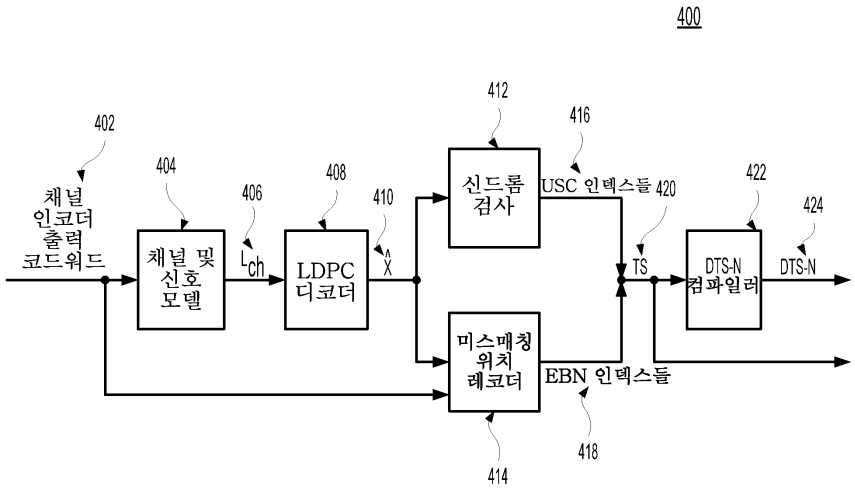
도면3

종래기술

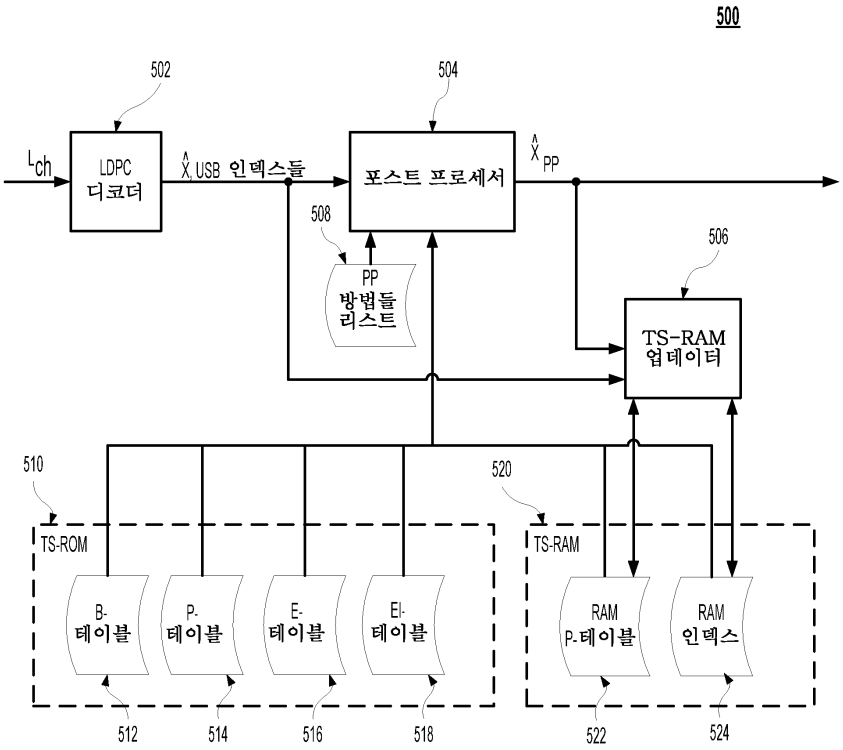


도면4

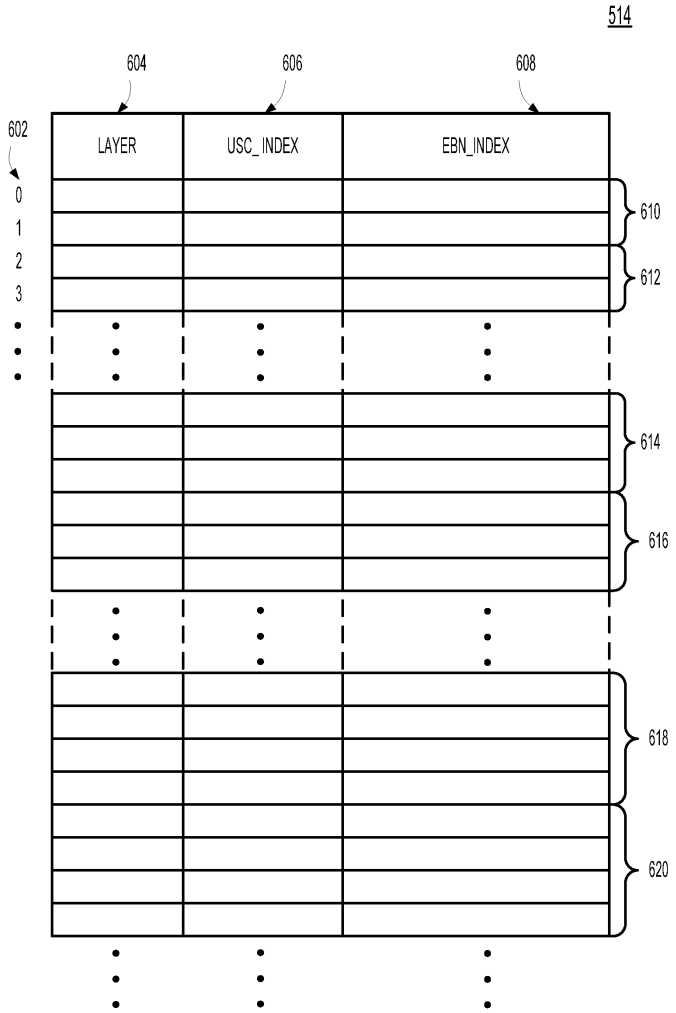
종래기술



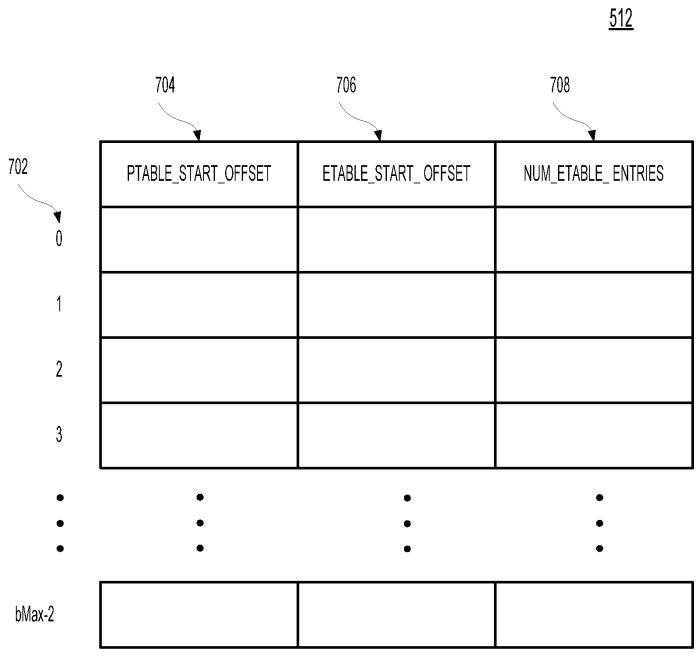
도면5


500
502
504
Lch
LDPC
디코더
X̂, USB 인덱스들
포스트 프로세서
X̂ PP
508
PP
방법들
리스트
506
TS-RAM
업데이터
510
520
TS-ROM
B-
테이블
P-
테이블
E-
테이블
EI-
테이블
TS-RAM
RAM
P-테이블
RAM
인덱스
512
514
516
518
522
524

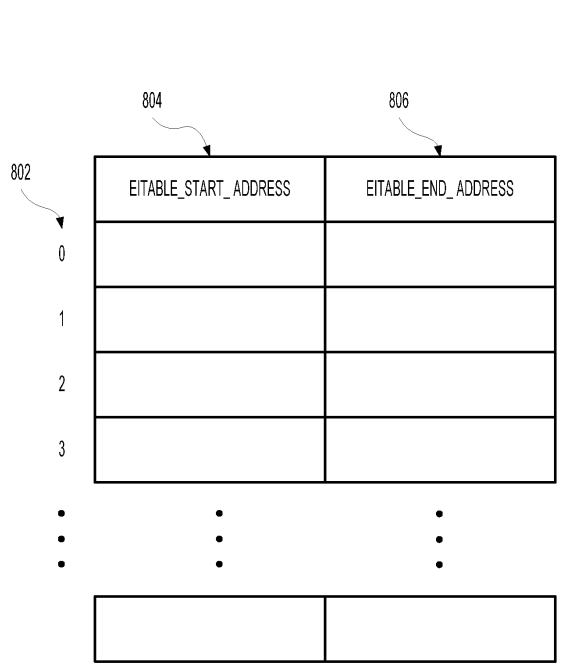
도면6


514
604
606
608
602
LAYER
USC_INDEX
EBN_INDEX
0
1
2
3
610
612
614
616
618
620

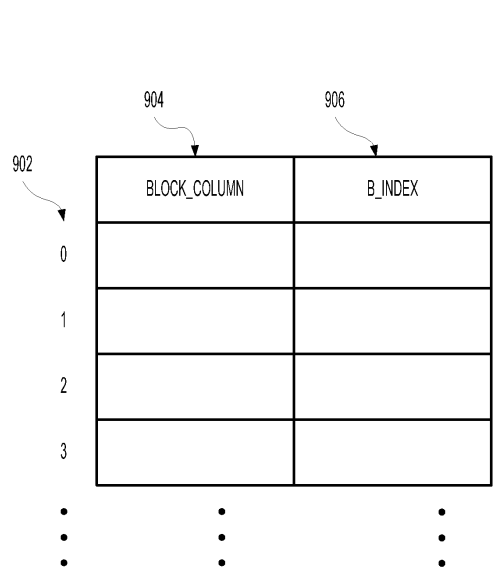
도면7


512
704
706
708
702
PTABLE_START_OFFSET
ETABLE_START_OFFSET
NUM_ETABLE_ENTRIES
0
1
2
3
bMax-2

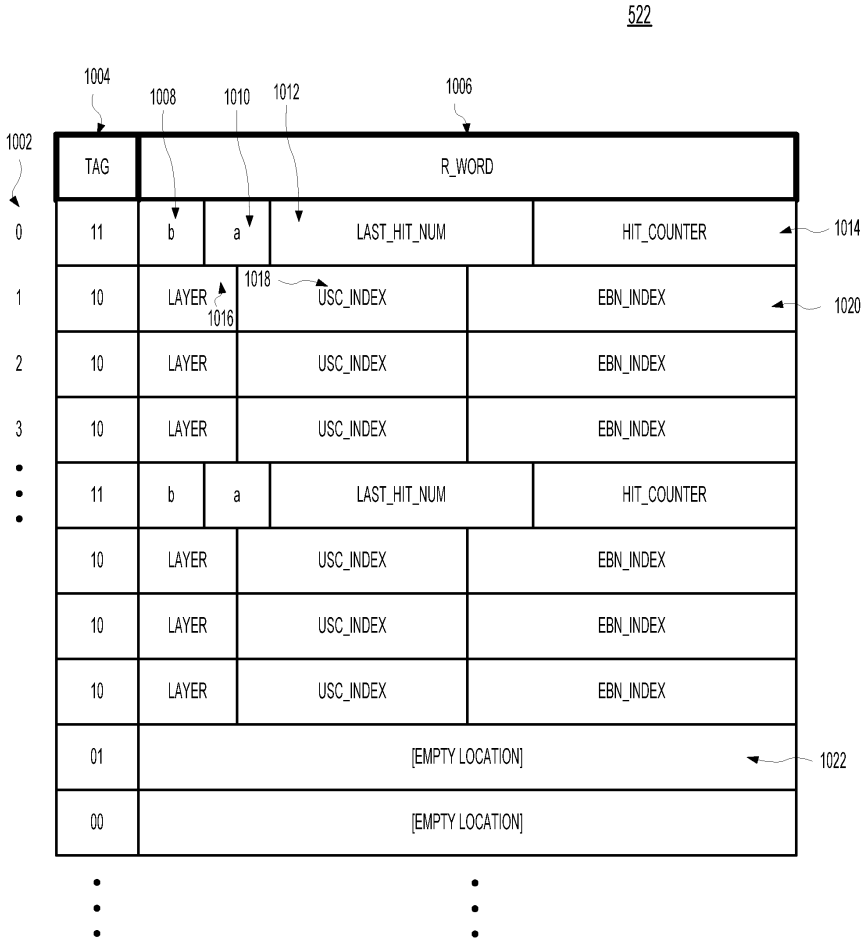
도면8


516
804
806
802
EITABLE_START_ADDRESS
EITABLE_END_ADDRESS
0
1
2
3

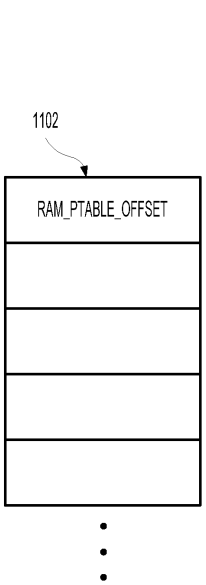
도면9


518
904
906
902
BLOCK_COLUMN
B_INDEX
0
1
2
3

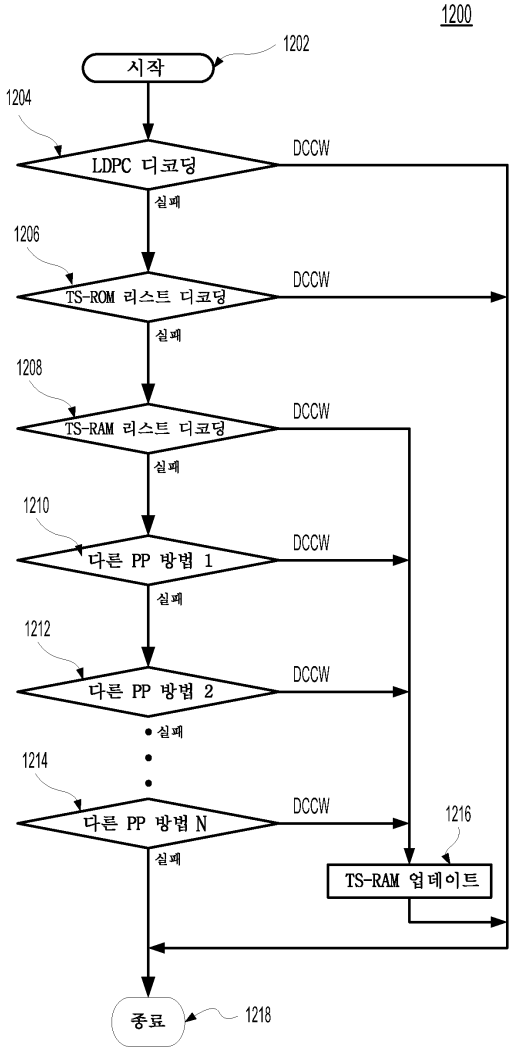
도면10


522
1004
1008
1010
1012
1006
1002
TAG
R_WORD
0
11
b
a
LAST_HIT_NUM
HIT_COUNTER
1014
1
10
LAYER
1016
1018
USC_INDEX
EBN_INDEX
1020
2
10
LAYER
USC_INDEX
EBN_INDEX
3
10
LAYER
USC_INDEX
EBN_INDEX
11
b
a
LAST_HIT_NUM
HIT_COUNTER
10
LAYER
USC_INDEX
EBN_INDEX
10
LAYER
USC_INDEX
EBN_INDEX
10
LAYER
USC_INDEX
EBN_INDEX
01
[EMPTY LOCATION]
1022
00
[EMPTY LOCATION]

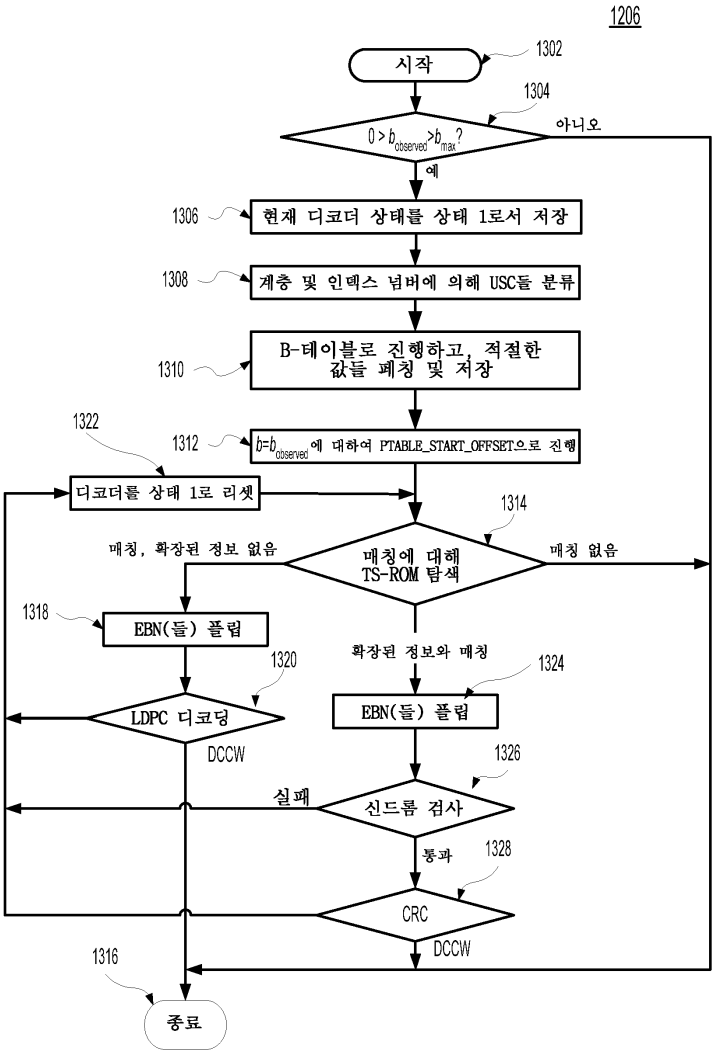
도면11


524
1102
RAM_PTABLE_OFFSET

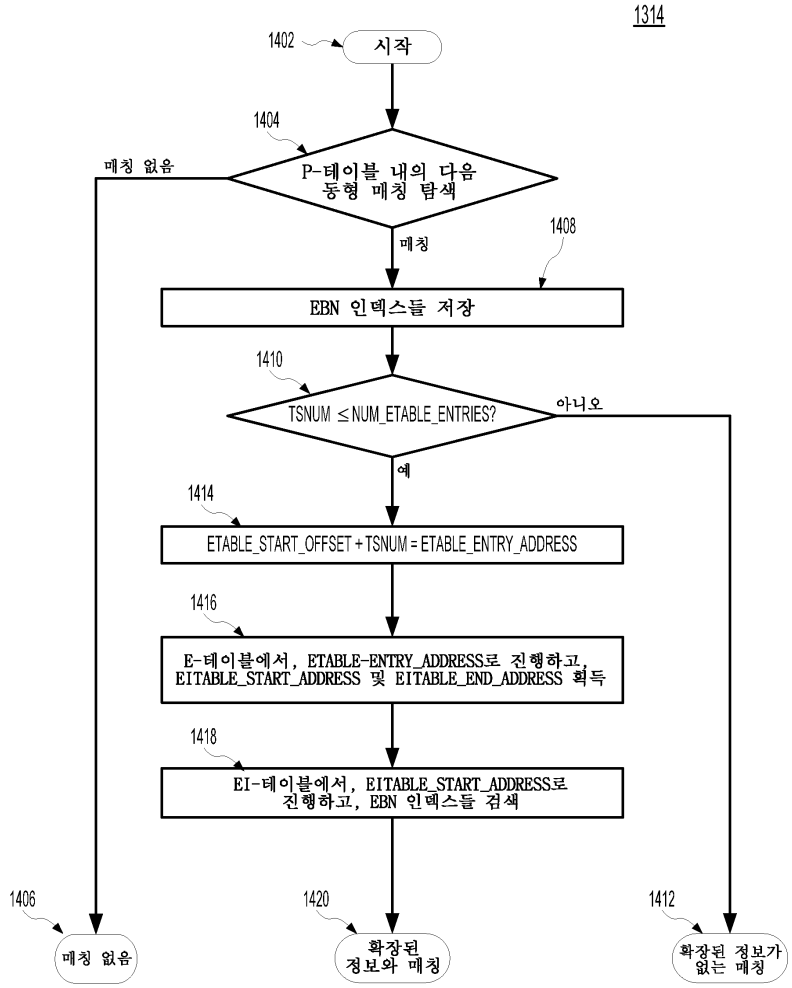
도면12


1200
1202
시작
1204
LDPC 디코딩
DCCW
실패
1206
TS-ROM 리스트 디코딩
DCCW
실패
1208
TS-RAM 리스트 디코딩
DCCW
실패
1210
다른 PP 방법 1
DCCW
실패
1212
다른 PP 방법 2
DCCW
실패
1214
다른 PP 방법 N
DCCW
실패
1216
TS-RAM 업데이트
종료
1218

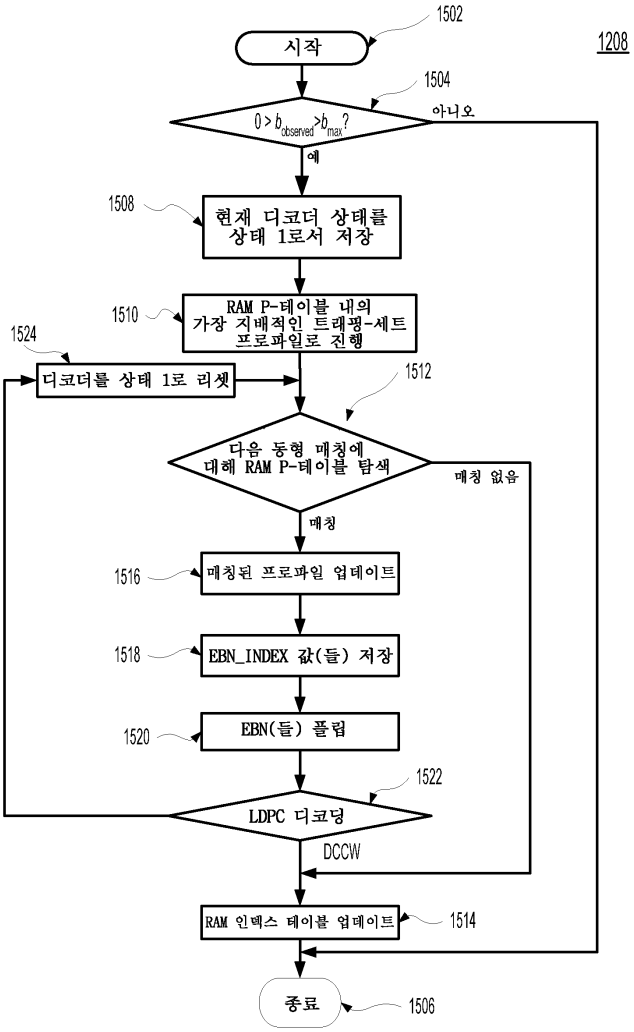
도면13


1206
시작
1302
1304
$0 > b_{observed} > b_{max}$?
아니오
예
1306
현재 디코더 상태를 상태 1로서 저장
1308
계층 및 인덱스 넘버에 의해 USC들 분류
1310
B-테이블로 진행하고, 적절한 값들 페칭 및 저장
1322
1312
$b=b_{observed}$에 대하여 PTABLE_START_OFFSET으로 진행
디코더를 상태 1로 리셋
1314
매칭에 대해 TS-ROM 탐색
매칭, 확장된 정보 없음
매칭 없음
1318
EBN(들) 플립
확장된 정보와 매칭
1320
1324
LDPC 디코딩
EBN(들) 플립
DCCW
1326
실패
신드롬 검사
통과
1328
CRC
DCCW
1316
종료

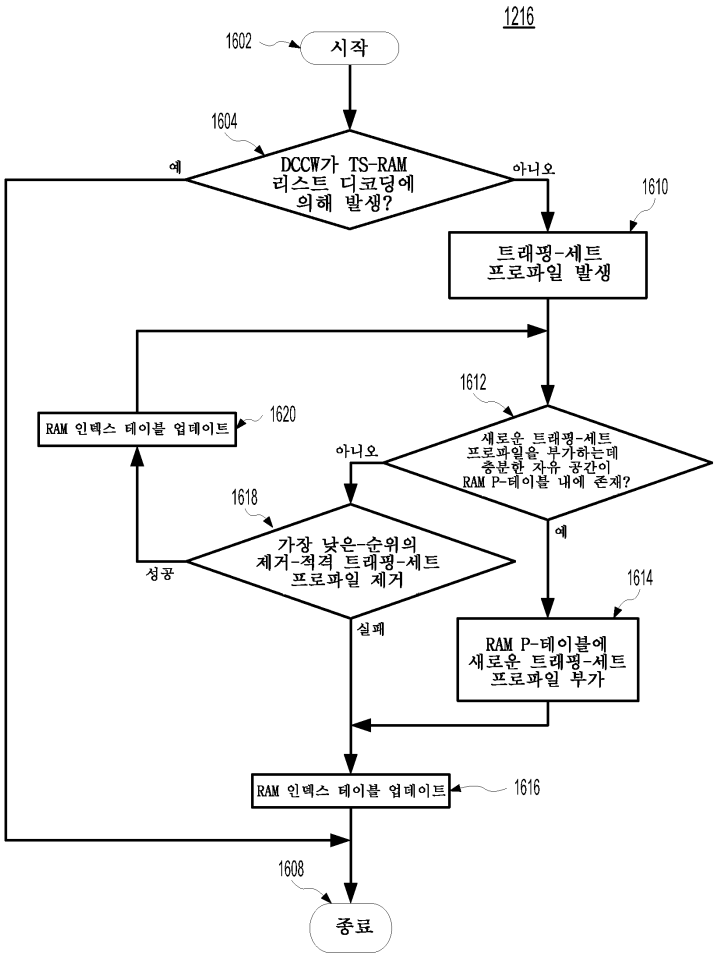
도면14


1314
1402
시작
1404
P-테이블 내의 다음 동형 매칭 탐색
매칭 없음
매칭
1408
EBN 인덱스들 저장
1410
TSNUM ≤NUM_ETABLE_ENTRIES?
아니오
예
1414
ETABLE_START_OFFSET + TSNUM = ETABLE_ENTRY_ADDRESS
1416
E-테이블에서, ETABLE-ENTRY_ADDRESS로 진행하고, EITABLE_START_ADDRESS 및 EITABLE_END_ADDRESS 획득
1418
EI-테이블에서, EITABLE_START_ADDRESS로 진행하고, EBN 인덱스들 검색
1406
매칭 없음
1420
확장된 정보와 매칭
1412
확장된 정보가 없는 매칭

도면15


1208
1502
시작
1504
$0 > b_{observed} > b_{max}$?
아니오
예
1508
현재 디코더 상태를 상태 1로서 저장
1510
RAM P-테이블 내의 가장 지배적인 트래핑-세트 프로파일로 진행
1524
디코더를 상태 1로 리셋
1512
다음 동형 매칭에 대해 RAM P-테이블 탐색
매칭 없음
매칭
1516
매칭된 프로파일 업데이트
1518
EBN_INDEX 값(들) 저장
1520
EBN(들) 플립
1522
LDPC 디코딩
DCCW
RAM 인덱스 테이블 업데이트
1514
종료
1506

도면16


1216
1602
시작
1604
DCCW가 TS-RAM 리스트 디코딩에 의해 발생?
예
아니오
1610
트래핑-세트 프로파일 발생
1612
새로운 트래핑-세트 프로파일을 부가하는데 충분한 자유 공간이 RAM P-테이블 내에 존재?
아니오
예
1614
RAM P-테이블에 새로운 트래핑-세트 프로파일 부가
1618
가장 낮은-순위의 제거-적격 트래핑-세트 프로파일 제거
성공
실패
1620
RAM 인덱스 테이블 업데이트
1616
RAM 인덱스 테이블 업데이트
1608
종료