(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2020년03월23일
(11) 등록번호 10-2092476
(24) 등록일자 2020년03월17일

(51) 국제특허분류(Int. Cl.)
H03M 13/29 (2006.01)
(52) CPC특허분류
H03M 13/2963 (2013.01)
H03M 13/2975 (2019.01)
(21) 출원번호 10-2018-0128883
(22) 출원일자 2018년10월26일
심사청구일자 2018년10월26일
(56) 선행기술조사문헌
KR1020150094365 A
US20020162072 A1
US20110194641 A1

(73) 특허권자
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
(72) 발명자
허준
서울특별시 강남구 광평로10길 6, 206동 103호(일원동, 한솔마을아파트)
안병규
서울특별시 성북구 인촌로24가길 4-9(안암동5가, 세진주택)
윤성식
서울특별시 송파구 백제고분로 348(석촌동, 잠실한솔아파트)
(74) 대리인
송인호, 윤형근, 최영중, 최관락

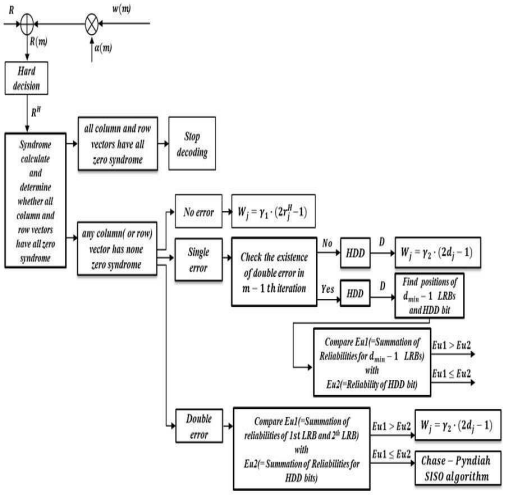
전체 청구항 수 : 총 8 항 심사관 : 조춘근

(54) 발명의 명칭 **블록 터보 부호의 신드롬 기반 복호 방법 및 장치**

(57) 요 약

본 발명은 블록 터보 부호의 신드롬 기반 복호 방법 및 장치를 개시한다. 본 발명에 따르면, 확장 해밍 부호를 구성 부호로 하는 블록 터보 부호의 신드롬 기반 복호 방법으로서, (a) 채널 통과 정보, 직전 절반 반복 복호의 외부 정보(extrinsic information)과 신뢰도 인자를 이용하여 다음 절반 반복 복호의 입력 정보값을 생성하는 단계; (b) 상기 입력 정보값의 경판정을 통해 경판정 워드를 생성하는 단계; (c) 상기 경판정 워드를 이용하여 상기 블록 터보 부호의 열 또는 행의 개수에 상응하는 n개의 1비트 신드롬을 계산하는 단계; 및 (d) 상기 계산된 n개의 1비트 신드롬을 이용하여 다음 절반 복호의 진행 여부를 판단하는 단계를 포함하는 신드롬 기반 블록 터보 부호의 복호 방법이 제공된다.

대 표 도 - 도1



이 발명을 지원한 국가연구개발사업

과제고유번호	20150003850041001
부처명	과학기술정보통신부
연구관리전문기관	정보통신기술진흥센터
연구사업명	대학ICT연구센터지원사업
연구과제명	무결점 보안을 위한 지상/위성 양자통신 기술개발
기 여 율	1/1
주관기관	고려대학교산학협력단
연구기간	2018.01.01 ~ 2018.12.31

명 세 서

청구범위

청구항 1

복호기에서 확장 해밍 부호를 구성 부호로 하는 블록 터보 부호의 신드롬 기반으로 복호하는 방법으로서,

(a) m-1번째 절반 반복 복호 결과에 따른 입력 정보값을 입력 받는 단계;

(b) 상기 입력 정보값의 경판정을 통해 경판정 워드를 생성하는 단계;

(c) 상기 경판정 워드를 이용하여 상기 블록 터보 부호의 열 또는 행의 개수에 상응하는 n개의 1비트 신드롬을 계산하는 단계; 및

(d) 상기 계산된 n개의 1비트 신드롬을 이용하여 m번째 절반 반복 복호의 진행 여부를 판단하는 단계를 포함하는 신드롬 기반 블록 터보 부호의 복호 방법.

청구항 2

제1항에 있어서,

상기 경판정 워드는 1 by n 행렬이며, 상기 1비트 신드롬은 상기 경판정 워드에 모두 1의 값을 갖는 n by 1 행렬을 곱하여 계산되는 신드롬 기반 블록 터보 부호의 복호 방법.

청구항 3

제1항에 있어서,

상기 (d) 단계는,

상기 n개의 1비트 신드롬이 모두 0이면 패리티 체크 행렬과 상기 입력 정보값을 곱하여 n-k-1(k는 정보 비트의 개수임) 비트의 신드롬을 계산하는 단계; 및

상기 계산된 n-k-1 비트의 신드롬이 모두 0인지 여부를 판단하는 단계를 포함하되,

상기 n-k-1 비트의 신드롬 중 하나라도 0이 아니면 상기 m번째 절반 반복 복호가 진행되는 신드롬 기반 블록 터보 부호의 복호 방법.

청구항 4

제3항에 있어서,

상기 (c) 단계 내지 (d) 단계는 상기 블록 터보 부호의 열 또는 행 모두에 대해 순차적으로 수행되는 신드롬 기반 블록 터보 부호의 복호 방법.

청구항 5

제1항에 있어서,

(e) 상기 m번째 절반 반복 복호에서 상기 입력 정보값의 경판정을 통해 경판정 워드를 생성하는 경입력 기반 복호의 적용 여부를 판단하는 단계를 더 포함하되,

상기 (c) 단계에서 계산된 상기 n개의 1비트 신드롬이 모두 1인 경우, 상기 (e) 단계는 상기 m-1번째 절반 반복 복호 과정에서 2개의 오류가 있는지 여부를 통해 상기 m번째 절반 반복 복호 과정에서 경입력 기반의 복호 기법을 적용할지 여부를 판단하는 신드롬 기반 블록 터보 부호의 복호 방법.

청구항 6

제5항에 있어서,

상기 (e) 단계는, 상기 m-1 절반 반복 복호 과정에서 2개의 오류가 포함된 경우, 상기 블록 터보 부호의 열 또

는 행의 정보 비트 중 가장 신뢰도가 낮은 3개의 비트와 경입력 기반의 복호를 통해 정정되는 비트의 위치를 파악한 후, 상기 가장 신뢰도가 낮은 3개의 비트의 신뢰도 합과 상기 경입력 기반의 복호를 통해 정정되는 비트의 신뢰도의 대소를 비교하여 경입력 기반의 복호를 수행할 것인지 여부를 결정하는 신드롬 기반 블록 터보 부호의 복호 방법.

청구항 7

제5항에 있어서,

상기 (e) 단계는 상기 입력 정보값의 신드롬 계산 결과가 2개의 오류를 포함하는 경우, 상기 블록 터보 부호의 열 또는 행의 정보 비트 중 가장 신뢰도가 낮은 2개의 비트와 경입력 기반의 복호를 통해 정정되는 비트의 신뢰도 값이 일치하는지 여부를 통해 경입력 기반의 복호를 수행할 것인지 여부를 결정하는 신드롬 기반 블록 터보 부호의 복호 방법.

청구항 8

확장 해밍 부호를 구성 부호로 하는 블록 터보 부호의 신드롬 기반 복호 장치로서,

프로세서; 및

상기 프로세서에 연결된 메모리를 포함하되,

상기 메모리는,

m-1번째 절반 반복 복호 결과에 따른 입력 정보값을 입력 받고,

상기 입력 정보값의 경판정을 통해 경판정 워드를 생성하고,

상기 경판정 워드를 이용하여 상기 블록 터보 부호의 열 또는 행의 개수에 상응하는 n개의 1비트 신드롬을 계산하고,

상기 계산된 n개의 1비트 신드롬을 이용하여 m번째 절반 반복 복호의 진행 여부를 판단하도록,

상기 프로세서에 의해 실행 가능한 프로그램 명령어들을 저장하는 신드롬 기반 블록 터보 부호의 복호 장치.

발명의 설명

기 술 분 야

[0001] 본 발명은 블록 터보 부호의 신드롬 기반 복호 방법 및 장치에 관한 것으로서, 보다 상세하게는 저 복잡도 조기 정지 기법과 혼합 복호 기법을 적용한 블록 터보 부호의 신드롬 기반 복호 방법 및 장치에 관한 것이다.

배 경 기 술

[0002] Berrou 등에 의해 제안된 블록 터보 부호는 2개의 recursive systematic convolutional code로 연결된 parallel concatenated convolutional codes에 대해 반복 복호를 수행하여 Shannon’s limit에 근접한 성능을 보인다.

[0003] 반복 복호는 블록 터보 부호의 가장 중요한 특징으로, 각 복호기가 생성하는 연출력(soft output)을 다른 복호기에 넘겨주어 반복 복호를 수행한다.

[0004] 반복 복호가 진행됨에 따라 비트오율(BER) 성능은 개선되지만, BER 성능의 개선 정도는 점차 줄어들어 점차 포화(saturation_되고, 반면에 복호 연산과 복호 지연은 선형적으로 증가하게 된다.

[0005] 이는 채널 특정에 따라 올바르게 복호되기 위한 반복 복호의 요구 회수가 프레임마다 다르기 때문에, 반복 복호 횟수를 고정하는 것은 비효율적이다. 따라서, 불필요한 연산량과 복호 지연을 줄이기 위해 블록 터보 부호의 반복 복호를 효율적으로 조기에 정지시키는 것이 필요하다.

[0006] 종래의 조기 정지 기법은, 매번 입력 정보 열과 행의 절반의 복호 과정을 수행한 결과 열과 행의 신드롬 연산을 수행하여 모든 열과 행의 신드롬이 모두 0(zero 벡터)이면 오류가 없는 것으로 판정한다.

[0007] 그러나, 이는 매번 반복 복호마다 신드롬 연산 과정을 수행해야 하기 때문에 연산량이 증가하는 문제점이 있다.

[0008] 또한, 블록 터보 부호의 복호를 위해 복잡도를 낮추기 위한 다양한 방법이 제시되고 있다.

[0009] R, Pyndiah, "Near-optimum decoding of product codes: block turbo codes," IEEE Trans. Commun., vol 46, No. 8, pp. 1003-1010, Aug. 1998.(선행문헌 1)는 부호의 hard input hard output 복호 알고리즘을 연입력연출력(soft input soft output)으로 확장하는 방법인 Chase-Pyndiah 복호 알고리즘을 제시한다.

[0010] 기존 기법에서는 경입력 복호 기법의 적용 비율을 증가시키기 위해 단일 오류에서 경입력과 연입력 기반의 복호 기법을 구분하는데, 복잡도를 크게 낮추기 못하였다.

[0011] Pen-Yao Lu, Erl-Huei Lu, Tso-Cho Chen, "An Efficient Hybrid decoder forBlock Turbo Codes," IEEE Commun. Letters, vol. 18, no. 12, pp. 2077?2080, Oct. 2014.(선행문헌 2)는 입력 연출력 복호 기법과 경입력 경출력 복호 기법을 복호기의 입력 값이 포함하고 있는 오류의 개수에 따라 선택적으로 사용하여 복호 복잡도를 낮추는 방법을 제시한다.

[0012] E.H. Lu and P.Y. Lu, "A syndrome-based hybrid decoder for turbo product codes," in Proc. IEEE 3CA Tainan, Taiwan , May 2010, pp. 280?282.(선행문헌 3)는 복호기 입력 벡터의 신드롬을 계산하여 그 값이 error free를 의미하는 0이면, 저 복잡도의 경입력 연출력 복호 기법을 적용하고 그 외의 신드롬 값을 가질 경우에는 연입력 연출력 복호 기법을 적용하여 복잡도를 크게 낮출 수 있는 알고리즘을 제시한다.

[0013] Guo Tai Chen, Lei Cao, Lun Yu, Chang Wen Chen, "An Efficient Stopping Criterion for Turbo Product Codes," IEEE Commun. Letters, vol. 11, no. 6, pp. 525 - 527, July 2007.(선행문헌 4)는 Block Turbo code의 반복 복호 과정에서 불필요한 반복 복호를 방지하기 위한 방법을 제시한다. 이를 위해 매 반복 복호 과정이 종료된 후 모든 복호 출력 벡터에 신드롬 연산을 수행하여 모든 출력 벡터가 열과 행 방향 모두 0을 가지면 더 이상 오류가 존재하지 않는다는 것을 보장할 수 있으므로 이 경우 복호 과정을 종료하는 방법을 제시하였다.

[0014] Byungkyu Ahn, Sungsik Yoon, Jun Heo, "Low Complexity Syndrome-Based Decoding Algorithm Applied to Block Turbo Codes," IEEE Access., vol. 6, pp. 2169-3536, April 2018.(선행문헌 5)는 신드롬 기반 Block Turbo code의 복호 복잡도를 줄일 수 있는 hybrid 복호 기법을 제안한다. 이를 통하여 블록 터보 부호의 구성부호로 사용되는 선형 블록부호의 오류정정 능력까지는 최대한 복잡도가 낮은 경입력 연출력 복호 기법을 적용하고 적용할 수 없는 부분에만 높은 복잡도의 연입력 연출력 복호 기법을 적용하는 방법을 제시하였다.

발명의 내용

해결하려는 과제

[0015] 상기한 종래기술의 문제점을 해결하기 위해, 성능 열화 없이 연산량과 복잡도를 낮출 수 있는 저 복잡도 조기 정지 기법과 혼합 복호 기법을 적용한 블록 터보 부호의 신드롬 기반 복호 방법 및 장치를 제안하고자 한다.

과제의 해결 수단

[0016] 상기한 바와 같은 목적을 달성하기 위하여, 본 발명의 일 실시예에 따르면, 확장 해밍 부호를 구성 부호로 하는 블록 터보 부호의 신드롬 기반 복호 방법으로서, (a) 채널 통과 정보, 직전 절반 반복 복호의 외부 정보(extrinsic information)과 신뢰도 인자를 이용하여 다음 절반 반복 복호의 입력 정보값을 생성하는 단계; (b) 상기 입력 정보값의 경판정을 통해 경판정 워드를 생성하는 단계; (c) 상기 경판정 워드를 이용하여 상기 블록 터보 부호의 열 또는 행의 개수에 상응하는 n개의 1비트 신드롬을 계산하는 단계; 및 (d) 상기 계산된 n개의 1비트 신드롬을 이용하여 다음 절반 복호의 진행 여부를 판단하는 단계를 포함하는 신드롬 기반 블록 터보 부호의 복호 방법이 제공된다.

[0017] 상기 경판정 워드는 1 by n 행렬이며, 상기 1비트 신드롬은 상기 경판정 워드에 모두 1의 값을 갖는 n by 1 행렬을 곱하여 계산될 수 있다.

[0018] 상기 (d) 단계는, 상기 n개의 1비트 신드롬이 모두 0이면 상기 입력 정보값과 패리티 체크 행렬을 곱하여 상기 블록 터보 부호의 행 또는 열의 n-k-1 비트의 신드롬을 계산하는 단계; 및 상기 계산된 n-k-1 비트의 신드롬이 모두 0인지 여부를 판단하는 단계를 포함하되, 상기 n-k-1 비트의 신드롬 중 하나라도 0이 아니면 다음 절반 복

호가 진행될 수 있다.

[0019] 상기 (c) 단계 내지 (d) 단계는 상기 행 또는 열 모두에 대해 순차적으로 수행될 수 있다.

[0020] (e) 경입력 기반 복호의 적용 여부를 판단하는 단계를 더 포함하되, 상기 (c) 단계에서 계산된 상기 n개의 1비트 신드롬이 모두 1인 경우, 상기 (e) 단계는 상기 직전 절반 반복 복호 과정에서 2개의 오류가 있는지 여부를 통해 다음 절반 반복 복호 과정에서 경입력 기반의 복호 기법을 적용할지 여부를 판단할 수 있다.

[0021] 상기 (e) 단계는, 상기 직전 절반 반복 복호 과정에서 2개의 오류가 포함된 경우, 가장 신뢰도가 낮은 3개의 비트와 경입력 기반의 복호를 통해 정정되는 비트의 위치를 파악한 후, 상기 가장 신뢰도가 낮은 3개의 비트의 신뢰도 합과 상기 경입력 기반의 복호를 통해 정정되는 비트의 신뢰도의 대소를 비교하여 경입력 기반의 복호를 수행할 것인지 여부를 결정할 수 있다.

[0022] 상기 (e) 단계는 상기 입력 정보값의 신드롬 계산 결과가 2개의 오류를 포함하는 경우, 가장 신뢰도가 낮은 2개의 비트와 경입력 기반의 복호를 통해 정정되는 비트의 신뢰도 값이 일치하는지 여부를 통해 경입력 기반의 복호를 수행할 것인지 여부를 결정할 수 있다.

[0023] 본 발명의 다른 측면에 따르면, 해밍 부호를 구성 부호로 하는 블록 터보 부호의 신드롬 기반 복호 장치로서, 프로세서; 및 상기 프로세서에 연결된 메모리를 포함하되, 상기 메모리는, 채널 통과 정보, 직전 절반 반복 복호의 외부 정보(extrinsic information)과 신뢰도 인자를 이용하여 다음 절반 반복 복호의 입력 정보값을 생성하고, 상기 입력 정보값의 경판정을 통해 경판정 워드를 생성하고, 상기 경판정 워드를 이용하여 상기 블록 터보 부호의 열 또는 행의 개수에 상응하는 n개의 1비트 신드롬을 계산하고, 상기 계산된 n개의 1비트 신드롬을 이용하여 다음 절반 복호의 진행 여부를 판단하도록, 상기 프로세서에 의해 실행 가능한 프로그램 명령어들을 저장하는 신드롬 기반 블록 터보 부호의 복호 장치가 제공된다.

발명의 효과

[0024] 본 발명에 따르면, 낮은 복잡도로 다음 절반 복호의 진행 여부를 빠르게 판단하고, 경입력 기반의 복호 기법의 적용 비율을 높여 복호 효율을 높일 수 있는 장점이 있다.

도면의 간단한 설명

[0025] 도 1은 본 발명의 일 실시예에 따른 블록 터보 부호의 복호 과정을 도시한 도면이다.

도 2는 종래기술에 따른 조기 복호 정지 기법을 설명하기 위한 도면이다.

도 3은 본 발명의 일 실시예에 따른 조기 복호 정지 기법을 설명하기 위한 도면이다.

도 4는 연입력 기반의 복호 기법인 Chase-Pyndiah 복호 기법을 나타낸 도면이다.

도 5는 연입력 기반의 Chase II 기법에서의 부호어 생성 과정을 도시한 도면이다.

도 6은 본 발명의 바람직한 일 실시예에 따른 블록 터보 부호의 복호 장치의 구성을 도시한 도면이다.

도 7은 본 실시예에 따른 블록 터보 부호의 신드롬 기반 복호 과정의 복잡도 감소 효과를 나타낸 것이다.

도 8은 본 실시예에 따른 블록 터보 부호의 신드롬 기반 복호 과정의 BER 성능을 나타낸 것이다.

도 9는 (32,26,4)/(64,57,4)/ (128,120,4)/(256,247,4) 부호를 구성 부호로 사용하는 블록 터보 부호에 적용했을 때의 BER 성능을 나타낸 것이다.

도 10은 단일 오류로 판정되는 것 중 저 복잡도의 경입력 기반 복호로 처리되는 비율의 변화를 비교한 것이다.

도 11은 2개의 오류를 가진 것으로 판정되는 것 중 저복잡도의 경입력 복호 기법으로 처리되는 비율 측정 결과를 나타낸 것이다.

발명을 실시하기 위한 구체적인 내용

[0026] 본 발명은 다양한 변경을 가할 수 있고 여러 가지 실시예를 가질 수 있는 바, 특정 실시예들을 도면에 예시하고 상세하게 설명하고자 한다.

[0027] 그러나, 이는 본 발명을 특정한 실시 형태에 대해 한정하려는 것이 아니며, 본 발명의 사상 및 기술 범위에 포

함되는 모든 변경, 균등물 내지 대체물을 포함하는 것으로 이해되어야 한다.

[0028] 본 발명은 블록 터보 부호의 복호 복잡도를 성능 열화 없이 낮추기 위해 다음의 기법을 적용한다.

[0029] (1) 저 복잡도 조기 반복 복호 정지 기법

[0030] 기존의 블록 터보 부호에서 사용되는 조기 반복 복호 정지 기법은 모든 열(또는 행)의 절반의 반복 복호 과정이 완료된 이후에 모든 열(또는 행 벡터)와 패리티 체크 행렬의 곱을 통해 얻어지는 신드롬이 모두 0인지 아닌지를 계산한다.

[0031] 모든 열과 행의 신드롬이 0이면 반복 복호를 멈추지만 매번 반복 복호 과정마다 신드롬 판정을 수행하면 연산량이 증가하게 된다.

[0032] 이를 해결하기 위해, 본 발명은 블록 터보 부호에서 확장 해밍 부호가 가지는 신드롬 중 특정한 한 비트의 값을 알면 단일 오류를 가지는 것과 그 외의 오류를 가지는 것을 구분할 수 있다는 특성을 이용한다.

[0033] 본 실시예에 따르면, 1차적으로 한 비트만을 이용하여 오류가 있는지 여부를 판정하여 모든 열과 행 중 단 한곳이라도 단일 오류를 가지는 것으로 판정되면 다음 절반의 반복 복호를 수행하게 함으로써 오류 추정에 소모되는 연산량을 크게 줄일 수 있다.

[0034] (2) 단일 오류로 판정된 경우의 경입력 기반 복호 기법의 적용 비율 증가

[0035] 확장 해밍 부호를 구성 부호로 가지는 블록 터보 부호에서는 신드롬 기반의 복호 과정에서 입력 정보값의 신드롬 연산을 통해 입력 정보값이 가지는 오류의 형태를 오류가 없는 것, 단일 오류를 가지는 것, 그 외의 오류를 가지는 것으로 나눌 수 있다.

[0036] 기존 연구에서는 오류가 없는 경우와 단일 오류가 발생한 경우 중 일부의 경우에 복잡한 연입력 기반의 복호 기법 대신 낮은 복잡도의 경입력 복호 기법을 적용해도 성능 열화 없이 저 복잡도 복호가 가능함을 보였다.

[0037] 기존 연입력 기반의 복호 기법으로 사용되는 Chase-Pyndiah 복호 기법에서 ML 복호 기법으로 부호어를 찾는 과정을 수행하는데 이 부호어와 본 발명에서 제시한 단일 오류로 판정된 경우의 경입력 기반의 복호를 통해 정정된 비트 위치와 신뢰도가 가장 신뢰도가 낮은 3비트의 신뢰도의 합보다 작을 경우 동일함을 증명하고 이를 통해 기존보다 단일 오류로 판정된 경우 중 저 복잡도의 경입력 기반 복호 기법을 적용할 수 있는 비율을 증가시킬 수 있고 이를 통해 복호 복잡도를 크게 낮춘다.

[0038] (3) 멀티 오류가 판정된 경우의 경입력 기반 복호 기법의 적용 비율 증가

[0039] 확장 해밍 부호를 구성 부호로 가지는 블록 터보 부호에서 입력 정보값의 신드롬 연산을 통해 2개의 오류를 가진 것으로 판정된 경우 중 확장 해밍 부호를 확장하면서 추가된 패리티 비트에 오류가 발생하고 나머지 비트 중 한 비트에 오류가 발생한 경우에는 2개의 비트 오류를 정정한다.

[0040] 입력 정보값이 2개의 오류를 가진 것으로 판정될 경우 2비트의 오류 정정이 일어난다. 이러한 오류 패턴들을 경입력 기반 복호 기법으로 정정하기 위해, 본 발명에서는 2비트의 신뢰도 합이 가장 신뢰도가 낮은 비트와 2번째로 신뢰도가 낮은 비트의 신뢰도 합보다 낮은 경우에는 기존의 연입력 기반 복호 기법의 ML 부호어와 같은 부호어를 경입력 정보를 바탕으로 한 복호 방법을 통해 얻을 수 있음을 증명하고 이를 적용한 비율만큼 복호 복잡도를 낮출 수 있다.

[0041] 이하, 도면을 참조하여 블록 터보 부호의 복호 방법을 상세하게 설명한다.

[0042] 도 1은 본 발명의 일 실시예에 따른 블록 터보 부호의 복호 과정을 도시한 도면이다.

[0043] 도 1을 참조하면, 채널을 통과한 정보 R과 m-1번째 절반 반복(half iteration)의 extrinsic 정보 W와 신뢰도 인자 α(m-1)의 곱의 합을 통하여 다음 절반 반복의 입력 정보값 R(m)을 생성한다.

[0044] 입력 정보값 R(m)의 경판정을 통해 경판정 워드(Hard decision word) RH를 생성한다.

[0045] 도 2는 종래기술에 따른 조기 복호 정지 기법을 설명하기 위한 도면이다.

[0046] 도 2에 도시된 바와 같이, RH와 패리티 체크 행렬을 곱하여 일반적으로는 신드롬을 계산한다.

[0047] 그러나, 본 발명에서는 도 3에 도시된 바와 같이, 복잡도를 낮추기 위하여 패리티 체크 행렬 대신 모두 1의 값을 가지는 n by 1 행렬을 1 by n의 길이를 가지는 RH와 곱하여 1비트의 신드롬 판정 결과를 얻는다.

[0048] 이 때 판정 결과가 1이면 단일 오류를 포함하는 것이고 0이면 오류가 없거나 2개의 오류를 포함하는 것을 의미한다.

[0049] 일반적으로 반복 복호 과정에서 모든 n개의 열이나 행 중에 오류가 포함되면 2개의 오류 보다는 단일 오류로 판정될 확률이 훨씬 높다.

[0050] 본 실시예에 따르면, 블록 터보 부호의 열 또는 행의 개수에 상응하는 n개의 1비트 신드롬을 계산하고, 계산된 n개의 1비트 신드롬을 이용하여 다음 절반 복호의 진행 여부를 판단한다.

[0051] 본 발명에서는 신드롬 판정을 단일 비트로 하되 모든 n개의 열 또는 행이 0으로 판정된다면 단일 오류를 포함하지 않고 있으므로 거의 대부분의 경우 2개의 오류는 포함되지 않을 것이라는 확률적 추정을 근거로 제시되었다.

[0052] 따라서 n개의 1비트 신드롬 판정의 결과가 모두 0이 아니면 다음 절반 복호 과정을 진행하고 그 외의 경우에는 아주 낮은 확률로 2개의 오류가 존재할 가능성도 있기 때문에 모든 n개의 열(또는 행)의 1비트 신드롬이 모두 0일 때만 나머지 n-k-1비트의 신드롬을 패리티 체크 행렬로부터 구해서 나머지 신드롬 비트의 값들도 모두 0임을 확인한다. 여기서, k는 정보 비트의 개수이다.

[0053] 이때 하나의 열(또는 행)의 신드롬이 0아닌 값을 가지면 조기 복호 정지 기법은 중단된다.

[0054] 그러나 n개의 1비트 신드롬이 0라면 이번에는 패리티 비트를 제외한 정보 비트들을 포함하고 있는 행(또는 열)의 1비트 신드롬을 구하고 그 값이 모두 0이면 나머지 n-k-1비트의 신드롬을 모두 확인하여 모두 0이면 조기 복호 정지 기법을 적용하여 더 이상의 반복 복호 과정을 멈춘다. 만약 그렇지 않으면 다음 절반 반복 복호를 계속 수행한다.

[0055] 오류가 하나라도 존재하는 경우에는 각 열 또는 행 벡터의 신드롬을 개별적으로 구한 후 도 1과 같이 오류가 없다고 판정된 열 또는 행 벡터는 R^H의 값을 부호어로 사용하고 그 값에 적절한 가중치를 곱하여 j번째 비트 위치의 extrinsic information을 다음식을 이용하여 구한다.

수학식 1

[0056] $$w_j = \gamma_1 \cdot (2r_j^H - 1)$$

[0057] 여기서, γ_1 은 선행문헌1에서 사용되는 β 의 2배의 값을 사용함

[0058] 또한, w_j 는 j번째 비트의 외부 정보, r_j^H 는 입력 정보의 경판정 벡터 R^H 의 j번째 비트 값, γ_1 은 보정 인자 값을 의미한다.

[0059] 단일 오류가 존재하는 경우에는 직전 절반 반복 복호 과정에서 2개의 오류가 하나도 없다고 판정된 경우 다음 절반 반복 복호 과정에서부터는 경입력 기반의 복호 기법을 적용하고, 2개의 오류가 하나라도 존재하면 가장 신뢰도가 낮은 3개의 비트와 경입력 기반의 복호를 통해 정정되는 비트의 위치를 파악한 후, 가장 신뢰도가 낮은 3개의 비트의 신뢰도 합과 경입력 기반의 복호를 통해 정정되는 비트의 신뢰도의 대소를 비교하여 경입력 기반의 복호를 할 것인지 기존의 연입력 기반 Chase-Pyndiah 복호를 할 것인지를 결정한다.

[0060] 본 실시예에 따르면, 이때 경입력 기반의 복호를 하는 부분과 연입력 기반의 복호를 하는 부분의 구분은 다음과 같은 근거로 인해 제시되었다.

[0061] 우선 기존 연입력 기반의 복호 기법인 Chase-Pyndiah 복호 기법은 도 4와 같은 순서로 복호 과정이 진행된다. 이때 2번 과정에서 생성되는 여러 개의 부호어 후보군 중에 가장 유클리드 거리가 짧은 것을 골라서 3번에서 부호어로 결정하게 된다. 하지만 이 과정은 도 5에 도시된 바와 같이, 여러 부호어를 생성한 후에 그 중에서 수신 정보와의 유클리드 거리가 가장 가까운 것을 선정하게 된다. 그런데 수신 정보와 가장 유클리드 거리가 가까운 벡터는 수신 정보를 경판정한 벡터이다. 따라서 경판정한 벡터와 2^p개의 부호어 중 하나의 부호어를 비교했을 때 서로 다른 값을 나타내는 비트의 위치가 존재한다면 그 비트들의 신뢰도(reliability)값의 4배만큼 R과의 유클리드 거리가 경판정 벡터에 비해서 멀어진다.

[0062] 또한 입력 정보값의 신드롬 계산 결과가 단일 오류로 판정된다면 경판정 벡터를 확장 해밍 부호로 복호하면 1비트의 값이 경판정 벡터에 비해서 유클리드 거리 증폭된다. 그런데 이렇게 경입력 복호 과정을 통해 구한 부호어가 최소 유클리드 거리를 구하기 위해서는 나머지 경쟁 부호어들보다 유클리드 거리가 짧다는 것이 보장 되어야 한다.

[0063] 따라서 본 실시예에서는 경쟁 부호어 중에서 가장 유클리드 거리가 짧은 경우를 다음과 같이 설정할 수 있다.

[0064] 경쟁 부호어는 경입력 기반 복호로 구한 부호어와 부호어 집합을 이루고 있으므로 최소 4비트의 값이 달라야 한다. 따라서 최소 4비트가 다른 부호어 중 입력 정보값으로부터 가장 유클리드 거리가 짧은 경우는 앞서 경입력 기반 복호에 의해 정정된 비트값이 경판정 비트와 동일하고 나머지 비트 중 3비트의 값이 플립된 경우이다.

[0065] 이 경우 유클리드 거리가 가장 최소화 되기 위해서는 상기한 3비트가 가장 신뢰도 값이 작아야 한다. 따라서 3비트의 위치에 가장 신뢰도가 낮은 값이 존재할 경우를 가정하여 3비트의 신뢰도 값의 합과 기존 경입력 기반 복호 기법에서 오류 정정된 1비트의 신뢰도 값을 비교하여 경입력 기반 복호 기법의 신뢰도가 더 작을 경우 경입력 기반 복호 기법을 적용한다.

[0066] 이때 j번째 비트의 외부 정보는 다음식을 통해 구한다.

수학식 2

[0067] $$w_j=\gamma_2 \cdot (2d_j^{HDD}-1)(j=1,2,\ldots,n)$$

[0068] 또한, w_j는 j번째 비트의 외부 정보, d_j^{HDD}는 입력 정보의 경판정 복호로 생성된 벡터의 j번째 비트 값, γ_2은 보정 인자 값을 의미한다.

[0069] 여기서 γ_2는 β의 3배로 했을 때가 가장 성능이 좋으므로 이값을 적용하였다.

[0070] 만약 입력 정보값의 신드롬 계산 결과가 2개의 오류를 의미할 경우에는 기존에는 경입력 복호로는 오류 정정이 불가능하므로 항상 연입력 기반의 Chase-Pyndiah 복호 기법을 적용한다.

[0071] 그러나 확장 해밍 부호의 경우 해밍 부호를 먼저 복호한 후에 나머지 확장 비트의 값을 해밍 부호의 각 비트값으로 mod 2 연산한 값이 0인지 1인지에 따라 0과 1로 판정하기 때문에 해밍 부호에 해당하는 비트 중 1비트에서 오류가 발생하고 해밍 부호의 확장 과정에서 추가되는 패리티 비트에서 오류가 1비트 발생할 경우에는 오류 정정이 가능하다.

[0072] 이런 경우를 감안하여 본 실시예에서는 2개의 오류가 발생하더라도 일부 오류 패턴의 경우 경입력 기반의 저 복잡도 복호 기법을 적용한다.

[0073] 경입력 기반의 복호를 수행할 경우와 연입력 기반의 복호를 수행할 경우를 구분하기 위하여 다음과 같은 분류 기준을 사용하였다.

[0074] 2개의 오류로 판정될 경우 경입력 정보를 이용한 복호를 수행할 경우 오류 정정 과정에서 추가로 2비트의 플립이 발생한다. 반전이 일어난 2비트의 신뢰도 값의 4배만큼 경판정 벡터와 채널 정보의 유클리드 거리에서 멀어

진다.

[0075] 그런데 경입력 정보 기반 복호 후의 부호어가 부호어 후보군 중 최소 유클리드 거리를 가지기 위해서는 부호어와 최소 4비트 다른 부호어 중에서 더 유클리드 거리가 짧은 것이 없어야 한다.

[0076] 경입력 기반의 복호를 통해 오류 정정된 비트의 신뢰도 값이 가장 신뢰도가 낮은 2개의 비트의 값과 일치한다면 이때는 어떤 경쟁 부호어도 이보다 거리가 짧은 경우는 생길 수가 없기 때문에 이를 만족하는 경우 경입력 복호로 구한 값을 부호어로 결정하고 j번째 비트의 외부 정보는 상기한 수학식을 통해 구한다.

[0077] 그렇지 않은 경우 연입력 기반의 복호 기법을 적용하여 오류 정정 과정을 수행한다.

[0078] 도 6은 본 발명의 바람직한 일 실시예에 따른 블록 터보 부호의 복호 장치의 구성을 도시한 도면이다.

[0079] 도 6에 도시된 바와 같이, 본 실시예에 따른 장치는 프로세서(600) 및 메모리(602)를 포함할 수 있다.

[0080] 프로세서(600)는 컴퓨터 프로그램을 실행할 수 있는 CPU(central processing unit)나 그밖에 가상 머신 등을 포함할 수 있다.

[0081] 메모리(602)는 고정식 하드 드라이브나 착탈식 저장 장치와 같은 불휘발성 저장 장치를 포함할 수 있다. 착탈식 저장 장치는 컴팩트 플래시 유닛, USB 메모리 스틱 등을 포함할 수 있다. 메모리(202)는 각종 랜덤 액세스 메모리와 같은 휘발성 메모리도 포함할 수 있다.

[0082] 이와 같은 메모리(602)에는 프로세서(600)에 의해 실행 가능한 프로그램 명령어들이 저장된다.

[0083] 저 복잡도의 블록 터보 부호의 복호를 위해, 메모리(602)에는 채널 통과 정보, 직전 절반 반복 복호의 외부 정보(extrinsic information)과 신뢰도 인자를 이용하여 다음 절반 반복 복호의 입력 정보값을 생성하고, 입력 정보값의 경판정을 통해 경판정 워드를 생성하고, 경판정 워드를 이용하여 상기 블록 터보 부호의 열 또는 행의 개수에 상응하는 n개의 1비트 신드롬을 계산하며, 계산된 n개의 1비트 신드롬을 이용하여 다음 절반 복호의 진행 여부를 판단하도록 프로세서(600)에 의해 실행되는 프로그램 명령어들이 저장될 수 있다.

[0084] 여기서, 경판정 워드는 1 by n 행렬이며, 상기한 1비트 신드롬은 경판정 워드에 모두 1의 값을 갖는 n by 1 행렬을 곱하여 계산된다.

[0085] 또한, 다음 절반 복호의 진행 여부를 판단하기 위해, 메모리(602)에는 상기한 n개의 1비트 신드롬이 모두 0이면 상기 입력 정보값과 패리티 체크 행렬을 곱하여 블록 터보 부호의 행 또는 열의 n-k-1 비트(여기서, k는 정보비트수)의 신드롬을 계산하고, 계산된 n-k-1 비트의 신드롬이 모두 0인지 여부를 판단하고 n-k-1 비트의 신드롬 중 하나라도 0이 아니면 다음 절반 복호가 진행되도록 결정하는 프로그램 명령어들이 저장된다.

[0086] 또한, 메모리(600)에는 경입력 기반 복호의 적용 여부를 판단하되, 계산된 상기 n개의 1비트 신드롬이 모두 1인 경우, 직전 절반 반복 복호 과정에서 2개의 오류가 있는지 여부를 통해 다음 절반 반복 복호 과정에서 경입력 기반의 복호 기법을 적용할지 여부를 판단하는 프로그램 명령어들이 저장된다.

[0087] 나아가, 상기한 프로그램 명령어들은 직전 절반 반복 복호 과정에서 2개의 오류가 포함된 경우, 가장 신뢰도가 낮은 3개의 비트와 경입력 기반의 복호를 통해 정정되는 비트의 위치를 파악한 후, 가장 신뢰도가 낮은 3개의 비트의 신뢰도 합과 경입력 기반의 복호를 통해 정정되는 비트의 신뢰도의 대소를 비교하여 경입력 기반의 복호를 수행할 것인지 여부를 결정한다.

[0088] 그리고, 프로그램 명령어들은 입력 정보값의 신드롬 계산 결과가 2개의 오류를 포함하는 경우, 가장 신뢰도가 낮은 2개의 비트와 경입력 기반의 복호를 통해 정정되는 비트의 신뢰도 값이 일치하는지 여부를 통해 경입력 기반의 복호를 수행할 것인지 여부를 결정한다.

[0089] 도 7은 본 실시예에 따른 블록 터보 부호의 신드롬 기반 복호 과정의 복잡도 감소 효과를 나타낸 것이다.

[0090] 여기서, 종래기술(Conventional)은 상기한 선행문헌 4에 기재된 기법이다.

[0091] 도 7을 참조하면, (32,26,4)와 (64,47,4) 확장 해밍 부호를 구성 부호로 갖는 블록 터보 부호를 복호하는 경우 선행문헌 4의 조기 복호 정지 기법과 본 발명에서 제안한 방법을 동일한 Chase-Pyndiah 복호 기법에 적용했을 때의 SNR별 반복 복호 감소 효과는 일치함을 알 수 있다.

[0092] 도 8은 본 실시예에 따른 블록 터보 부호의 신드롬 기반 복호 과정의 BER 성능을 나타낸 것이다.

[0093] 도 8에 도시된 바와 같이, (32,26,4)와 (64,47,4) 확장 해밍 부호를 구성 부호로 갖는 블록 터보 부호를 복호하

는 경우 선행문헌 4의 조기 복호 정지 기법과 본 발명에 따른 방법을 동일한 Chase-Pyndiah 복호 기법에 적용했을 때 BER 성능을 비교해 보면 (32,26,4)와 (64,47,4) 확장 해밍 부호를 구성 부호로 가지는 경우 모두 기존과 제안 기법의 성능이 일치함을 알 수 있다.

[0094] 도 9는 (32,26,4)/(64,57,4)/ (128,120,4)/(256,247,4) 부호를 구성 부호로 사용하는 블록 터보 부호에 적용했을 때의 BER 성능을 나타낸 것이다.

[0095] 도 9를 참조하면, 전체적으로 본 발명에 따른 기법과 기존 2가지 기법인 선행 문헌 1과 5의 BER성능을 비교해보면 거의 유사함을 알 수 있으며 구성 부호의 길이가 길어질수록 점점 기존 기법에 비해 본 발명에 따른 기법의 성능이 조금 더 개선됨을 알 수 있다.

[0096] 도 10은 선행문헌 5 대비 단일 오류로 판정되는 것 중 저 복잡도의 경입력 기반 복호로 처리되는 비율의 변화를 비교한 것이다. 이는 10000개의 샘플 비교 결과를 나타낸다.

[0097] 도 10은 부호율 약 0.6~0.9사이의 4개의 서로 다른 부호를 구성 부호로 사용하는 블록 터보 부호에서 기존 논문 중 단일 오류 중 많은 부분을 경입력 기반 복호 기법으로 정정하는 기법[선행문헌 5]과 비교하였을 때 본 실시예에 따른 경입력 기반 복호 기법으로 정정하는 비율 변화를 나타낸다.

[0098] 결과에서 알 수 있듯이 반복 복호 횟수에 따라 증가율이 다르며 평균적으로 최소 1.4배에서 5.6배 저 복잡도 복호 기법이 많이 사용되고 있음을 알 수 있다.

[0099] 따라서 이 비율의 증가만큼 기존의 높은 복잡도를 가지는 Chase-Pyndiah 복호 기법대신 저 복잡도를 가지는 경입력 복호 기법을 사용함으로써 전체 복호 복잡도를 크게 낮출 수 있다.

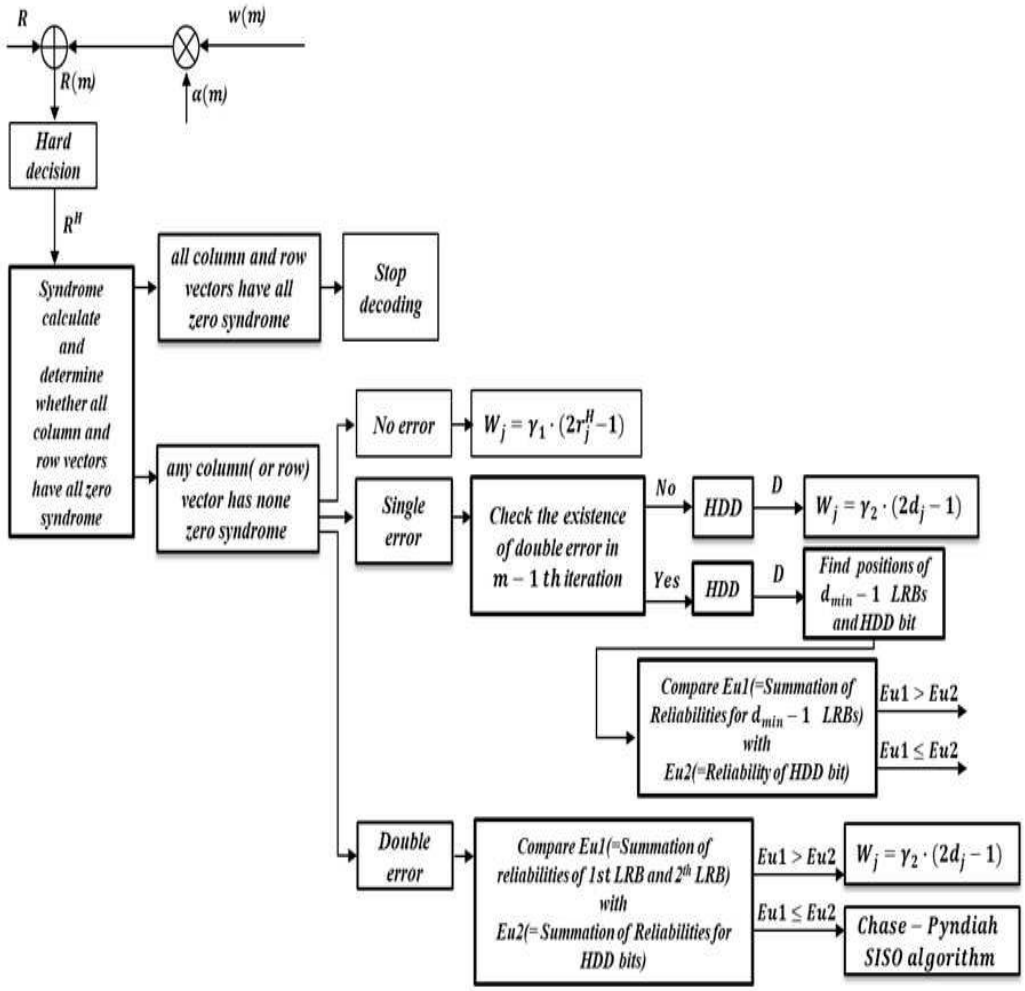
[0100] 도 11은 2개의 오류를 가진 것으로 판정되는 것 중 저복잡도의 경입력 복호 기법으로 처리되는 비율 측정 결과를 나타낸 것이다.

[0101] 도 11은 10000개의 샘플 비교 결과를 나타낸 것이다. 도 11을 참조하면, 2개의 오류를 가진 것 중 제안 기법을 적용하면 일부의 오류는 경입력 정보를 기반으로 한 확장 해밍 부호의 복호 기법만으로도 오류 정정이 가능하며 그 비율만큼 기존 기법[선행문헌 5] 대비 복잡도를 크게 개선할 수 있다.

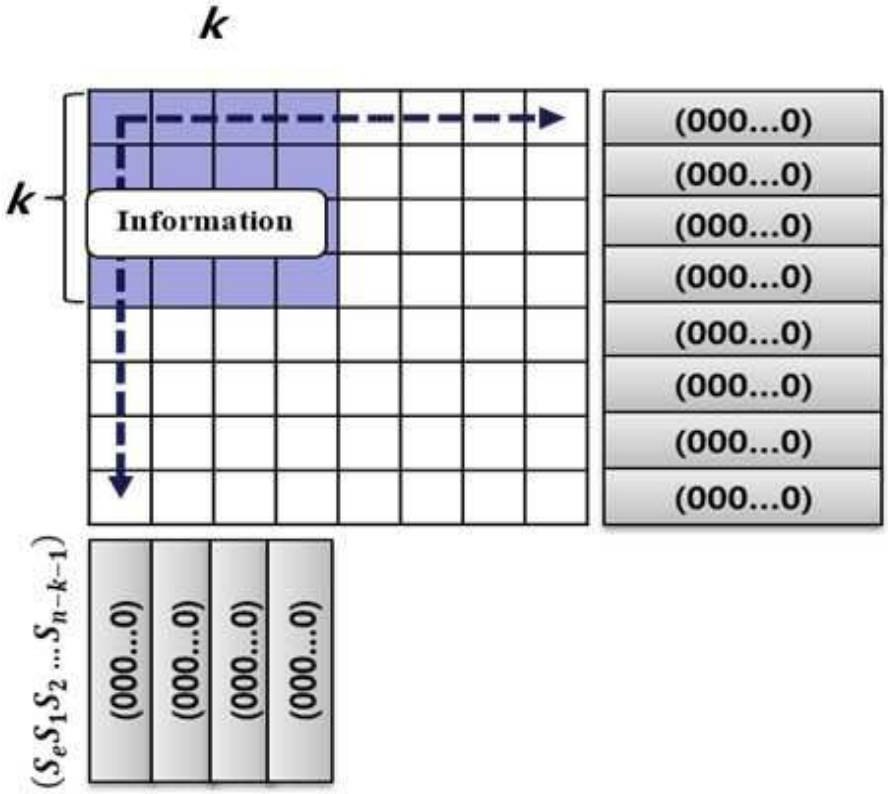
[0102] 상기한 본 발명의 실시예는 예시의 목적을 위해 개시된 것이고, 본 발명에 대한 통상의 지식을 가지는 당업자라면 본 발명의 사상과 범위 안에서 다양한 정정, 변경, 부가가 가능할 것이며, 이러한 정정, 변경 및 부가는 하기의 특허청구범위에 속하는 것으로 보아야 할 것이다.

도면

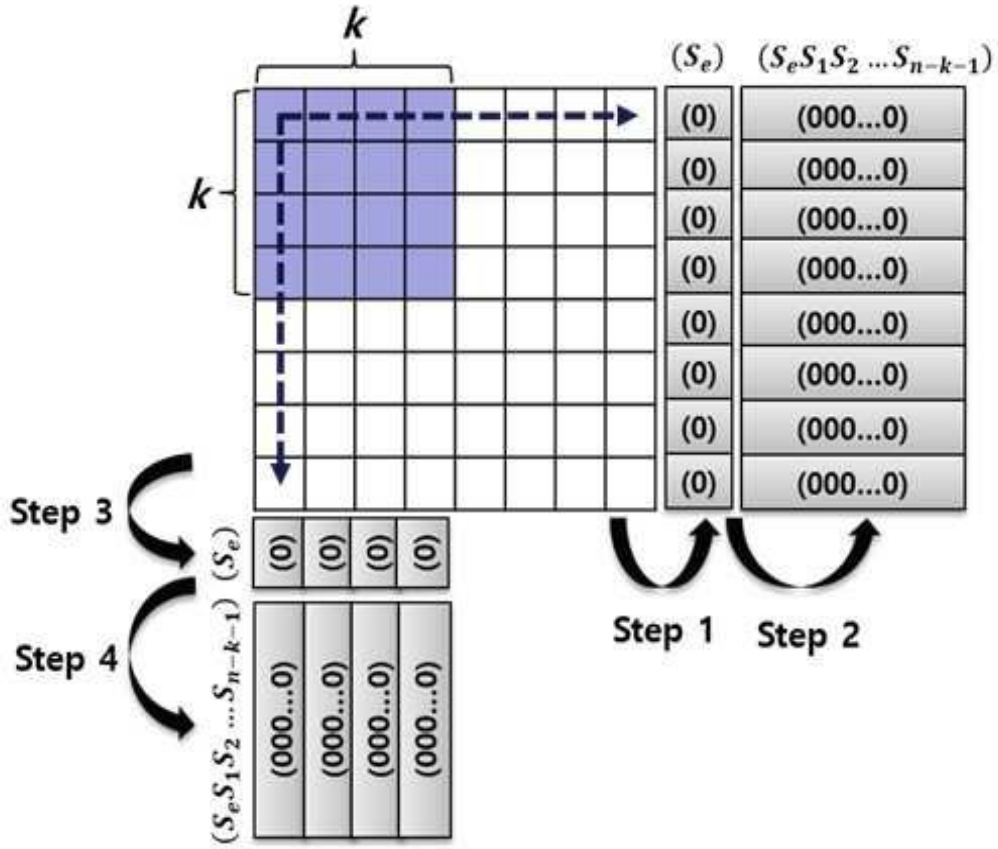
도면1


R
w(m)
R(m)
α(m)
Hard decision
R^H
Syndrome calculate and determine whether all column and row vectors have all zero syndrome
all column and row vectors have all zero syndrome
Stop decoding
any column(or row) vector has none zero syndrome
No error
$W_j = \gamma_1 \cdot (2r_j^H - 1)$
Single error
Check the existence of double error in m − 1 th iteration
No
HDD
D
$W_j = \gamma_2 \cdot (2d_j - 1)$
Yes
HDD
D
Find positions of $d_{min} - 1$ LRBs and HDD bit
Compare Eu1(=Summation of Reliabilities for $d_{min} - 1$ LRBs) with Eu2(=Reliability of HDD bit)
Eu1 > Eu2
Eu1 ≤ Eu2
Double error
Compare Eu1(=Summation of reliabilities of 1st LRB and 2th LRB) with Eu2(= Summation of Reliabilities for HDD bits)
Eu1 > Eu2
$W_j = \gamma_2 \cdot (2d_j - 1)$
Eu1 ≤ Eu2
Chase − Pyndiah SISO algorithm

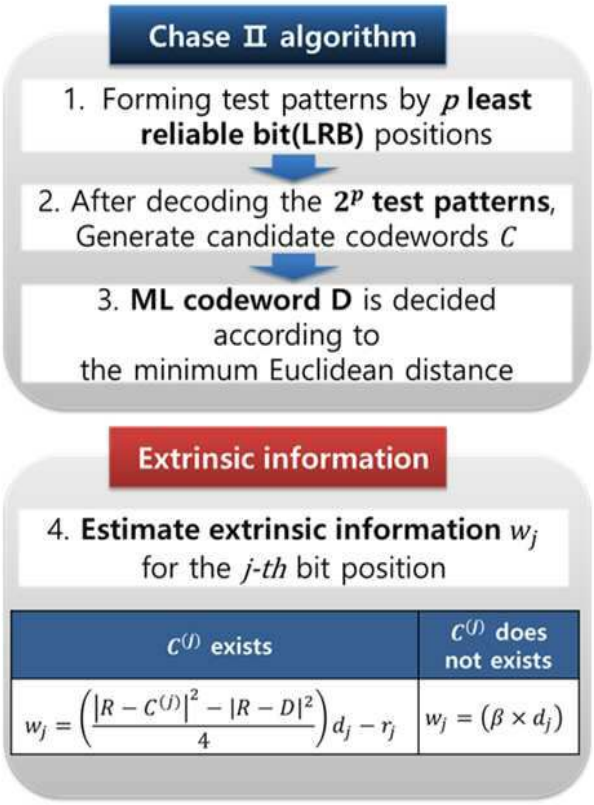
도면2


k
k
Information
(000...0)
(000...0)
(000...0)
(000...0)
(000...0)
(000...0)
(000...0)
(000...0)
$(S_e S_1 S_2 \ldots S_{n-k-1})$
(000...0)
(000...0)
(000...0)
(000...0)

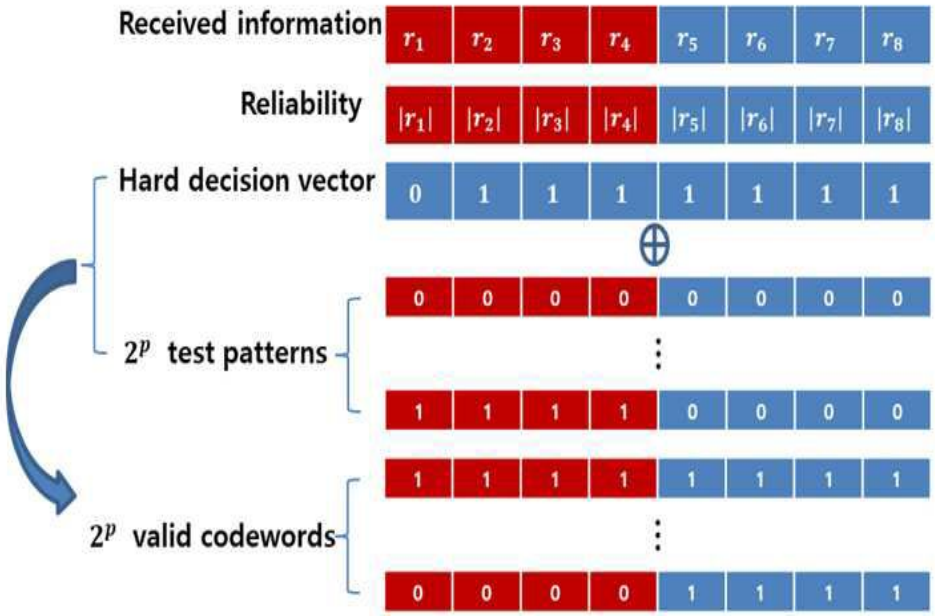
도면3


k
k
(S_e)
$(S_eS_1S_2 \dots S_{n-k-1})$
(0)
(000...0)
Step 1
Step 2
Step 3
Step 4
(S_e)
$(S_eS_1S_2 \dots S_{n-k-1})$

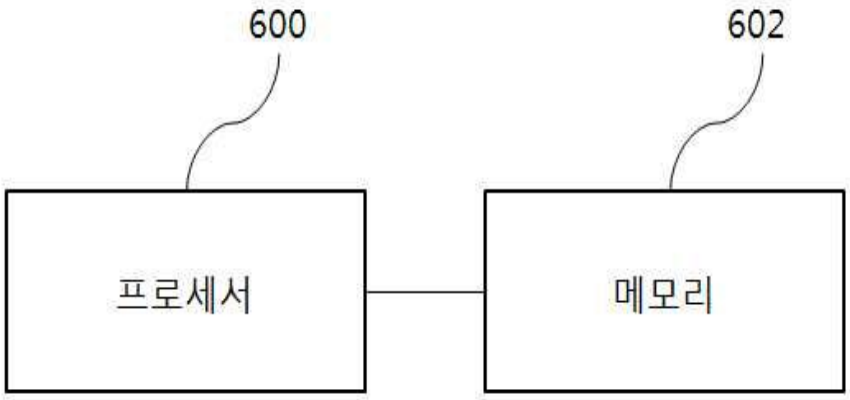
도면4


Chase II algorithm
1. Forming test patterns by p least reliable bit(LRB) positions
2. After decoding the 2^p test patterns, Generate candidate codewords C
3. ML codeword D is decided according to the minimum Euclidean distance
Extrinsic information
4. Estimate extrinsic information w_j for the j-th bit position
C^(j) exists
C^(j) does not exists
w_j = ((|R − C^(j)|^2 − |R − D|^2)/4) d_j − r_j
w_j = (β × d_j)

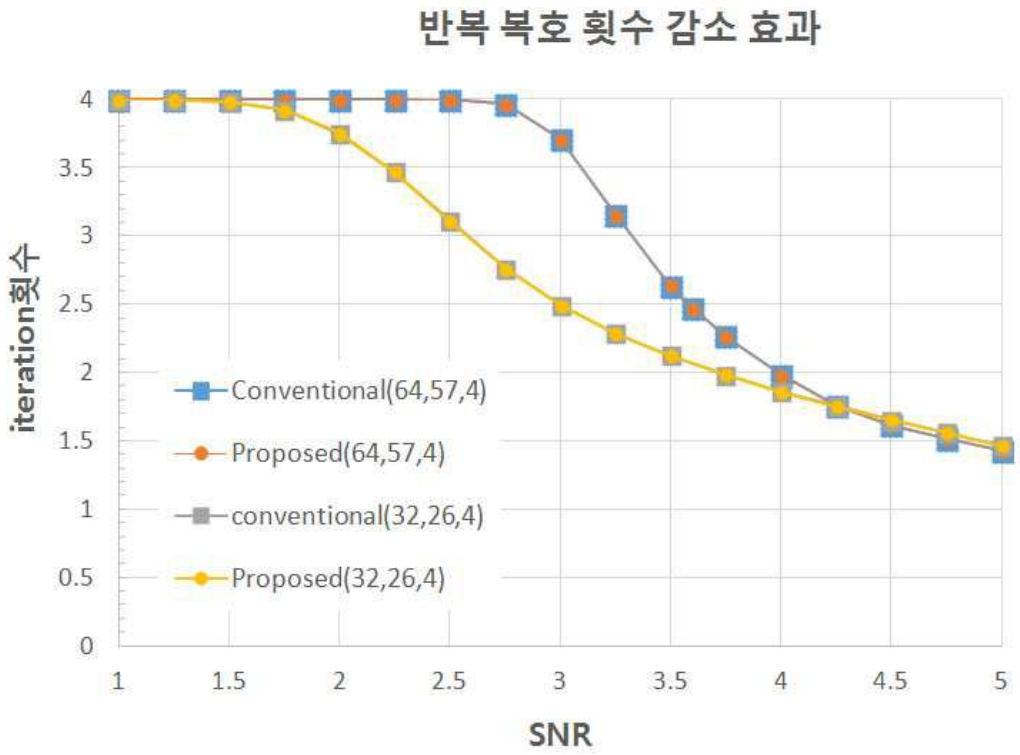
도면5


Received information
r1 r2 r3 r4 r5 r6 r7 r8
Reliability
|r1| |r2| |r3| |r4| |r5| |r6| |r7| |r8|
Hard decision vector
0 1 1 1 1 1 1 1
2^p test patterns
0 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0
2^p valid codewords
1 1 1 1 1 1 1 1
0 0 0 0 1 1 1 1

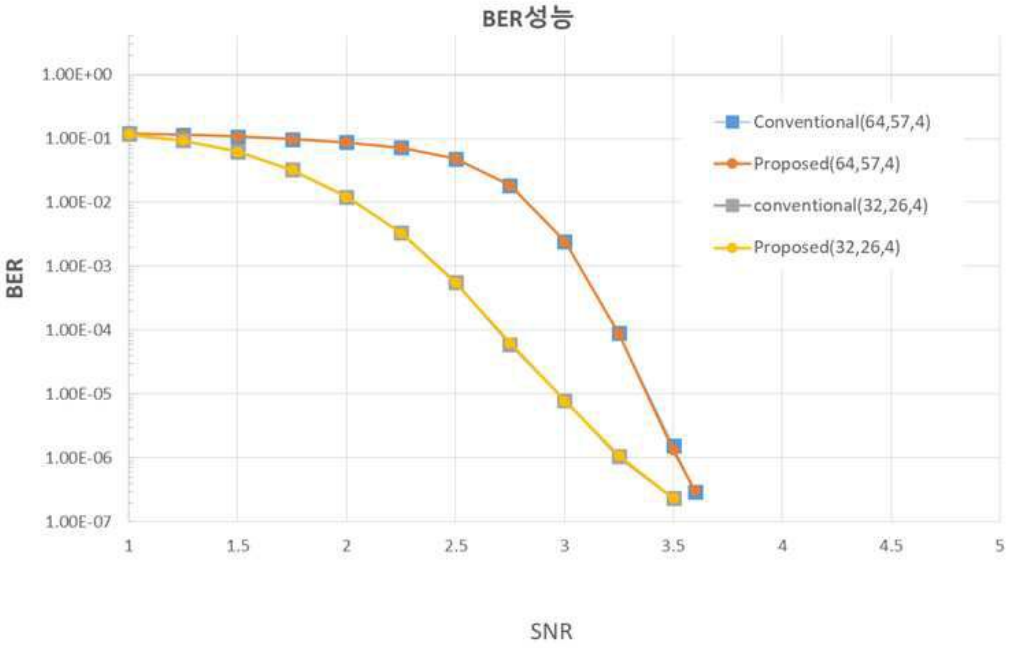
도면6


600
602
프로세서
메모리

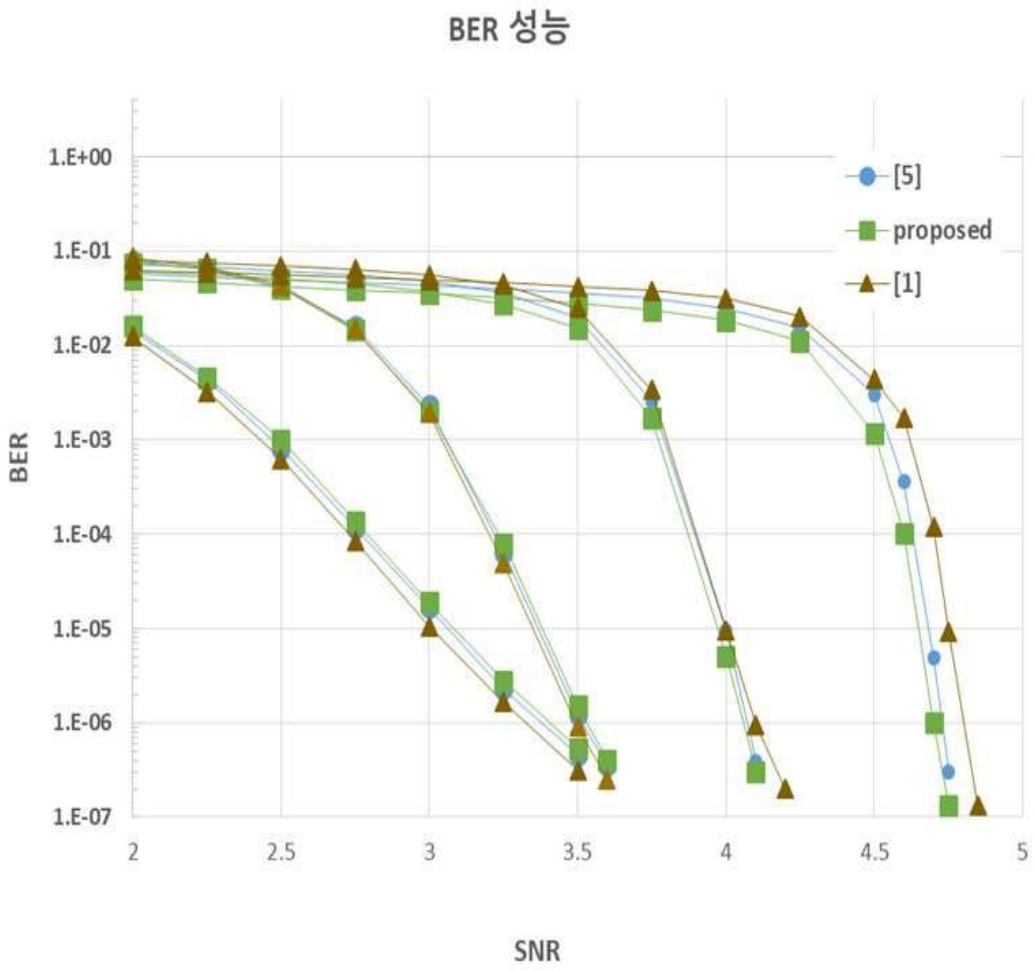
도면7


반복 복호 횟수 감소 효과
iteration횟수
SNR
Conventional(64,57,4)
Proposed(64,57,4)
conventional(32,26,4)
Proposed(32,26,4)
0
0.5
1
1.5
2
2.5
3
3.5
4
1
1.5
2
2.5
3
3.5
4
4.5
5

도면8


BER성능
1.00E+00
1.00E-01
1.00E-02
1.00E-03
1.00E-04
1.00E-05
1.00E-06
1.00E-07
BER
1
1.5
2
2.5
3
3.5
4
4.5
5
SNR
Conventional(64,57,4)
Proposed(64,57,4)
conventional(32,26,4)
Proposed(32,26,4)

도면9


BER 성능
1.E+00
1.E-01
1.E-02
1.E-03
1.E-04
1.E-05
1.E-06
1.E-07
BER
2
2.5
3
3.5
4
4.5
5
SNR
[5]
proposed
[1]

도면10

[1].(32,26,4) 터보 곱 부호

SNR	1	2	3	4	5	6	7	8
1	170%	182%	155%	158%	153%	157%	166%	185%
1.25	166%	172%	152%	151%	144%	145%	155%	169%
1.5	163%	167%	143%	146%	137%	138%	142%	156%
1.75	158%	158%	142%	133%	129%	128%	138%	151%
2	152%	153%	134%	129%	122%	124%	134%	155%
2.25	147%	144%	126%	122%	117%	133%	169%	229%
2.5	143%	136%	119%	116%	117%	150%	238%	378%
2.75	140%	133%	116%	112%	115%	182%	312%	513%
3	137%	129%	112%	107%	119%	217%	395%	624%
3.25	134%	127%	109%	102%	121%	227%	374%	633%
3.5	130%	125%	107%	95%	121%	219%	303%	541%
avg	145%	152%	143%	148%	158%	201%	269%	378%

[2].(64,57,4) 터보 곱 부호

SNR	1	2	3	4	5	6	7	8
1	196%	244%	155%	176%	163%	177%	200%	247%
1.25	187%	244%	157%	176%	163%	175%	196%	246%
1.5	183%	243%	153%	174%	159%	174%	194%	235%
1.75	178%	229%	155%	167%	157%	168%	189%	226%
2	170%	217%	156%	166%	152%	160%	177%	208%
2.25	166%	202%	153%	158%	145%	150%	161%	182%
2.5	160%	190%	149%	151%	135%	134%	137%	147%
2.75	156%	178%	140%	137%	124%	118%	120%	124%
3	150%	162%	131%	125%	114%	109%	129%	177%
3.25	145%	152%	124%	115%	105%	129%	257%	557%
3.5	140%	143%	117%	106%	102%	196%	580%	1928%
3.6	139%	141%	115%	101%	102%	230%	739%	2554%
avg	164%	195%	142%	146%	135%	160%	257%	569%

[3].(128,120,4) 터보 곱 부호

SNR	1	2	3	4	5	6	7	8
1	200%	372%	160%	184%	165%	184%	219%	297%
1.25	199%	362%	159%	184%	163%	183%	220%	302%
1.5	210%	355%	157%	188%	165%	185%	217%	308%
1.75	200%	354%	157%	187%	166%	185%	221%	310%
2	198%	345%	158%	188%	165%	185%	219%	307%
2.25	186%	336%	156%	187%	163%	184%	216%	302%
2.5	183%	317%	156%	186%	163%	180%	214%	302%
2.75	180%	291%	154%	181%	158%	178%	202%	275%
3	171%	266%	153%	172%	153%	163%	186%	239%
3.25	161%	237%	152%	161%	143%	149%	159%	189%
3.5	155%	214%	146%	148%	127%	126%	128%	132%
3.75	151%	190%	137%	133%	112%	101%	96%	99%
4	145%	169%	127%	116%	98%	97%	186%	595%
4.1	143%	163%	124%	110%	92%	120%	383%	2239%
avg	177%	284%	150%	166%	145%	159%	205%	421%

[4].(256,247,4) 터보 곱 부호

SNR	1	2	3	4	5	6	7	8
1	201%	529%	161%	195%	174%	215%	245%	345%
1.25	166%	629%	152%	178%	163%	167%	218%	291%
1.5	223%	545%	161%	181%	157%	174%	209%	313%
1.75	140%	477%	183%	185%	172%	177%	227%	345%
2	224%	516%	160%	168%	176%	206%	224%	329%
2.25	187%	688%	150%	176%	170%	183%	234%	352%
2.5	195%	516%	150%	191%	177%	194%	244%	443%
2.75	131%	476%	158%	184%	163%	193%	243%	339%
3	202%	511%	169%	189%	177%	198%	267%	486%
3.25	208%	380%	151%	171%	159%	173%	221%	380%
3.5	176%	490%	146%	179%	162%	186%	241%	425%
3.75	173%	535%	152%	169%	154%	171%	213%	307%
4	165%	281%	136%	177%	149%	170%	210%	268%
4.25	178%	244%	129%	155%	137%	140%	134%	162%
4.5	151%	174%	135%	128%	113%	105%	97%	103%
4.6	147%	204%	130%	129%	104%	89%	76%	84%
4.7	146%	187%	130%	121%	95%	73%	94%	276%
4.75	145%	182%	129%	117%	90%	72%	155%	858%
avg	176%	420%	149%	166%	149%	160%	197%	339%

도면11

[1].(32,26,4) 터보 곱 부호

SNR	1	2	3	4	5	6	7	8
1	0.79%	0.75%	1.03%	1.05%	1.23%	1.50%	1.87%	1.87%
1.25	0.83%	1.03%	1.14%	1.17%	1.69%	1.74%	1.70%	2.03%
1.5	0.87%	1.15%	1.34%	1.73%	1.61%	1.94%	1.85%	2.62%
1.75	1.19%	1.15%	1.39%	1.85%	1.63%	1.97%	2.35%	1.95%
2	1.14%	1.34%	1.88%	2.33%	2.32%	2.67%	2.79%	2.11%
2.25	1.26%	1.53%	2.10%	2.58%	2.70%	3.00%	2.91%	3.72%
2.5	1.43%	1.74%	2.17%	2.66%	3.02%	3.01%	2.90%	1.66%
2.75	1.54%	2.02%	2.61%	3.18%	3.73%	3.27%	4.01%	1.95%
3	1.72%	2.32%	2.97%	3.54%	3.88%	3.47%	3.80%	4.00%
3.25	1.91%	2.65%	3.33%	4.00%	4.05%	3.32%	3.24%	3.42%
3.5	2.11%	2.92%	3.71%	4.28%	4.16%	3.78%	3.41%	5.10%
avg	1.34%	1.69%	2.15%	2.58%	2.73%	2.70%	2.80%	2.77%

[2].(64,57,4) 터보 곱 부호

SNR	1	2	3	4	5	6	7	8
1	0.15%	0.15%	0.25%	0.41%	0.47%	0.62%	0.72%	0.83%
1.25	0.14%	0.17%	0.31%	0.37%	0.53%	0.62%	0.71%	0.81%
1.5	0.18%	0.14%	0.30%	0.35%	0.47%	0.63%	0.76%	0.77%
1.75	0.21%	0.18%	0.34%	0.44%	0.53%	0.59%	0.72%	0.88%
2	0.21%	0.22%	0.30%	0.39%	0.48%	0.68%	0.74%	0.78%
2.25	0.37%	0.27%	0.35%	0.51%	0.68%	0.71%	0.91%	0.87%
2.5	0.33%	0.35%	0.52%	0.51%	0.69%	0.76%	1.01%	1.05%
2.75	0.44%	0.47%	0.63%	0.86%	0.97%	0.87%	0.87%	1.05%
3	0.39%	0.60%	0.79%	1.03%	0.74%	0.89%	1.00%	0.70%
3.25	0.60%	0.76%	1.04%	1.29%	1.33%	0.93%	1.97%	2.08%
3.5	0.66%	0.92%	1.28%	1.53%	1.65%	1.58%	1.79%	1.03%
3.6	0.71%	1.00%	1.37%	1.65%	1.71%	1.59%	1.62%	1.25%
avg	0.36%	0.44%	0.62%	0.78%	0.85%	0.87%	1.07%	1.01%

[3].(128,120,4) 터보 곱 부호

SNR	1	2	3	4	5	6	7	8
1	0.04%	0.03%	0.15%	0.18%	0.19%	0.34%	0.35%	0.36%
1.25	0.03%	0.03%	0.16%	0.17%	0.25%	0.29%	0.35%	0.46%
1.5	0.03%	0.03%	0.14%	0.19%	0.21%	0.25%	0.32%	0.36%
1.75	0.03%	0.03%	0.18%	0.18%	0.24%	0.28%	0.33%	0.37%
2	0.03%	0.04%	0.16%	0.19%	0.23%	0.29%	0.32%	0.32%
2.25	0.05%	0.04%	0.13%	0.18%	0.20%	0.28%	0.33%	0.42%
2.5	0.05%	0.05%	0.19%	0.18%	0.26%	0.31%	0.32%	0.41%
2.75	0.05%	0.05%	0.16%	0.15%	0.20%	0.32%	0.34%	0.41%
3	0.08%	0.07%	0.14%	0.16%	0.23%	0.27%	0.41%	0.40%
3.25	0.10%	0.08%	0.17%	0.21%	0.22%	0.34%	0.44%	0.48%
3.5	0.15%	0.13%	0.23%	0.25%	0.38%	0.33%	0.41%	0.54%
3.75	0.15%	0.24%	0.36%	0.41%	0.43%	0.39%	0.51%	0.70%
4	0.22%	0.29%	0.45%	0.57%	0.62%	0.65%	0.88%	0.84%
4.1	0.26%	0.35%	0.51%	0.61%	0.62%	0.69%	0.71%	0.16%
avg	0.09%	0.10%	0.22%	0.26%	0.31%	0.36%	0.48%	0.44%

[4].(256,247,4) 터보 곱 부호

SNR	1	2	3	4	5	6	7	8
1	0.00%	0.01%	0.16%	0.10%	0.09%	0.17%	0.14%	0.19%
1.25	0.00%	0.00%	0.09%	0.04%	0.15%	0.13%	0.16%	0.16%
1.5	0.00%	0.02%	0.13%	0.05%	0.18%	0.13%	0.15%	0.15%
1.75	0.02%	0.02%	0.12%	0.07%	0.06%	0.19%	0.17%	0.26%
2	0.01%	0.02%	0.12%	0.09%	0.09%	0.14%	0.22%	0.17%
2.25	0.02%	0.00%	0.17%	0.13%	0.13%	0.14%	0.13%	0.19%
2.5	0.01%	0.01%	0.10%	0.07%	0.10%	0.12%	0.20%	0.23%
2.75	0.01%	0.02%	0.12%	0.07%	0.10%	0.14%	0.18%	0.18%
3	0.01%	0.02%	0.13%	0.09%	0.13%	0.22%	0.18%	0.21%
3.25	0.00%	0.00%	0.16%	0.11%	0.10%	0.09%	0.09%	0.22%
3.5	0.01%	0.02%	0.10%	0.05%	0.07%	0.14%	0.13%	0.21%
3.75	0.00%	0.02%	0.07%	0.07%	0.10%	0.16%	0.19%	0.19%
4	0.07%	0.02%	0.13%	0.05%	0.11%	0.15%	0.13%	0.26%
4.25	0.03%	0.08%	0.09%	0.10%	0.19%	0.10%	0.18%	0.25%
4.5	0.01%	0.09%	0.10%	0.17%	0.11%	0.08%	0.19%	0.16%
4.6	0.05%	0.11%	0.14%	0.22%	0.20%	0.13%	0.13%	0.39%
4.7	0.10%	0.13%	0.21%	0.20%	0.21%	0.20%	0.17%	1.01%
4.75	0.10%	0.14%	0.21%	0.24%	0.22%	0.24%	0.22%	0.33%
avg	0.03%	0.05%	0.13%	0.12%	0.13%	0.15%	0.17%	0.29%