(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2020-0025059
(43) 공개일자 2020년03월10일

(51) 국제특허분류(Int. Cl.)
H03M 13/39 (2006.01) *H03M 13/00* (2017.01)
H03M 13/19 (2006.01) *H03M 13/31* (2006.01)
(52) CPC특허분류
H03M 13/39 (2013.01)
H03M 13/19 (2013.01)
(21) 출원번호 10-2018-0101851
(22) 출원일자 2018년08월29일
심사청구일자 2018년08월29일

(71) 출원인
남서울대학교 산학협력단
충청남도 천안시 서북구 성환읍 대학로 91 , 남서울대학교내
(72) 발명자
정호영
경기도 평택시 소사2길 9, 소사SK뷰 아파트 103동 1303호
(74) 대리인
김견수

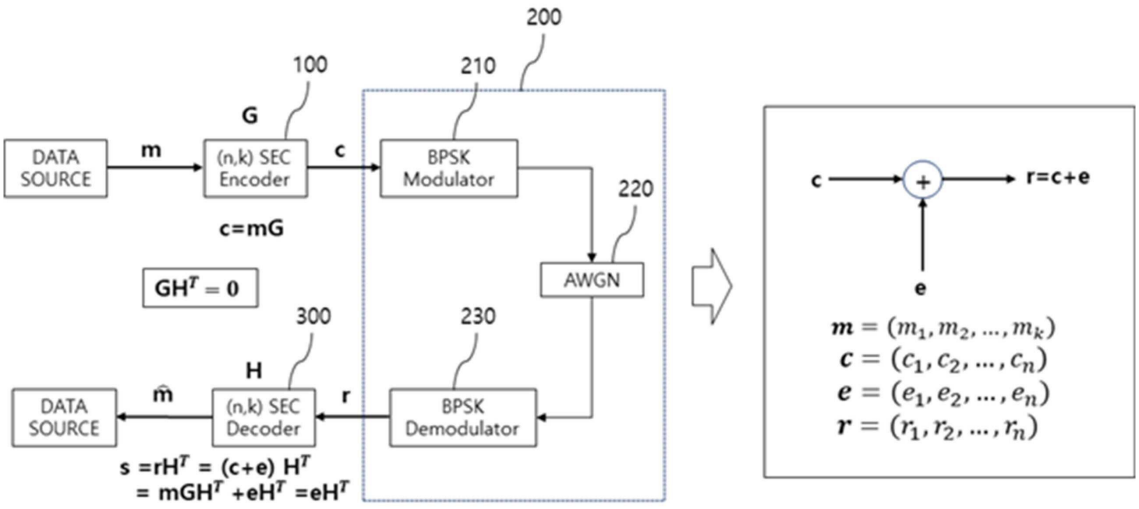
전체 청구항 수 : 총 12 항

(54) 발명의 명칭 **SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법**

(57) 요 약

본 발명은 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법에 관한 것으로, SEC(Single Error Correction)부호, 즉 단일오류정정부호로 부호화된 데이터에 복수의 오류가 발생한 경우 단일 오류 이상을 정정할 수 없었던 문제를 극복하여, 복수의 오류에 대해서도 정정이 가능하도록 하는 복호기 및 그 방법을 제시하고자 한다.

대 표 도 - 도1



(52) CPC특허분류
H03M 13/31 (2013.01)
H03M 13/6502 (2013.01)
H04L 1/0061 (2013.01)
H04L 1/203 (2013.01)

명 세 서

청구범위

청구항 1

복조기로부터 복조된 소프트 값을 수신하는 SEC부호 벡터 수신부;

상기 수신한 SEC부호 벡터로부터 신드롬 벡터를 추출하는 신드롬 추출부;

상기 추출한 신드롬 벡터를 이용하여 오류를 검출하는 오류 검출부; 및

상기 오류가 검출된 경우, 다수의 오류벡터에서 오류패턴을 결정하는 오류패턴 결정부;를 포함하며,

상기 오류패턴은 단일 오류와 2개 이상의 오류를 정정하도록 구성되는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호기.

청구항 2

청구항 1에 있어서,

상기 오류 검출부는,

상기 신드롬 벡터가 제로 벡터이면 오류가 없는 것으로 판정하고, 상기 제로 벡터가 아니면 적어도 하나 이상의 오류가 존재하는 것으로 판정하며, 상기 오류가 존재하는 것으로 판정이 되면, 단일 오류는 물론 2개 이상의 오류를 정정할 수 있는 오류패턴을 결정하도록 요청하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호기.

청구항 3

청구항 1에 있어서,

상기 오류패턴 결정부는,

상기 오류가 검출된 경우, 특정 신드롬 벡터에 해당하는 적어도 하나 이상의 각 오류패턴 중에서 오류가 존재하는 것으로 인식된 비트에 해당하는 위치의 상기 소프트 값을 제곱하여 더한 다음 그 결과가 가장 작은 값을 가지는 것을 오류패턴으로 결정하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호기.

청구항 4

청구항 3에 있어서,

상기 오류패턴에서, 단일오류만 검출되는 오류패턴인 경우, 상기 검출한 단일오류에 해당하는 소프트 값을 곱하고, 그 결과에 상기 정정할 2개 이상의 오류에 대한 개수만큼 더 곱하여 다른 오류패턴들과 그 크기를 비교함으로써, 상기 오류패턴을 결정하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호기.

청구항 5

청구항 1에 있어서,

상기 SEC부호는 해밍부호를 포함한 패리티검사부호 또는 선형블록부호를 포함하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호기.

청구항 6

청구항 1에 있어서,

상기 결정한 오류패턴과 상기 수신한 SEC부호 벡터를 서로 배타적논리합(Exclusive-OR)으로 연산하는 회로를 통해서 상기 단일 오류와 2개 이상의 오류를 정정하는 오류 정정부;를 더 포함하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호기.

청구항 7

복조기로부터 복조된 소프트 값을 수신하는 SEC부호 벡터의 수신 단계;

상기 수신한 SEC부호 벡터로부터 신드롬 벡터를 추출하는 신드롬 추출 단계;

상기 추출한 신드롬 벡터를 이용하여 오류를 검출하는 오류 검출 단계; 및

상기 오류가 검출된 경우, 다수의 오류벡터에서 오류패턴을 결정하는 오류패턴 결정 단계;를 포함하며,

상기 오류패턴은 단일 오류와 2개 이상의 오류를 정정하도록 구성되는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호 방법.

청구항 8

청구항 7에 있어서,

상기 오류 검출 단계는,

상기 신드롬 벡터가 제로 벡터이면 오류가 없는 것으로 판정하고, 상기 제로 벡터가 아니면 적어도 하나 이상의 오류가 존재하는 것으로 판정하며, 상기 오류가 존재하는 것으로 판정이 되면, 단일 오류는 물론 2개 이상의 오류를 정정할 수 있는 오류패턴을 결정하도록 요청하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호 방법.

청구항 9

청구항 7에 있어서,

상기 오류패턴 결정 단계는,

상기 오류가 검출된 경우, 특정 신드롬 벡터에 해당하는 적어도 하나 이상의 각 오류패턴 중에서 오류가 존재하는 것으로 인식된 비트에 해당하는 위치의 상기 소프트 값을 제곱하여 더한 다음 그 결과가 가장 작은 값을 가지는 것을 오류패턴으로 결정하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호 방법.

청구항 10

청구항 9에 있어서,

상기 오류패턴에서, 단일오류만 검출되는 오류패턴인 경우, 상기 검출한 단일오류에 해당하는 소프트 값을 곱하고, 그 결과에 상기 정정할 2개 이상의 오류에 대한 개수만큼 더 곱하여 다른 오류패턴들과 그 크기를 비교함으로써, 상기 오류패턴을 결정하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호 방법.

청구항 11

청구항 7에 있어서,

상기 SEC부호는 해밍부호를 포함한 패리티검사부호 또는 선형블록부호를 포함하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호 방법.

청구항 12

청구항 7에 있어서,

상기 결정한 오류패턴와 상기 수신한 SEC부호 벡터를 서로 배타적논리합으로 연산하는 회로를 통해서 상기 단일오류와 2개 이상의 오류를 정정하는 오류 정정 단계;를 더 포함하는 것을 특징으로 하는 SEC부호의 멀티오류정정을 위한 복호 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법에 관한 것으로, 더욱 상세하게는 SEC(Single Error Correction)부호, 즉 단일오류정정부호로 부호화된 데이터에 복수의 오류가 발생한 경우 단일

오류 그 이상을 정정할 수 없었던 문제를 극복하여, 복호 복잡도를 증가시키지 않으면서도 복수의 오류에 대해서도 정정이 가능하도록 하는 복호기 및 그 방법을 제시하고자 한다.

배 경 기 술

[0002] 최근의 IoT(Internet of Things) 환경은 스마트폰(Smart Phone), 랩탑컴퓨터(Laptop Computer) 등과 같은 모바일 기기는 물론 TV, 냉장고 등과 같은 일상적인 생활 기기가 인터넷에 연결되어 인간은 물론 기기들 간의 통신을 통해 서로 연결되는 유비쿼터스 사회로 진화하고 있다.

[0003] 상기 IoT 환경에서 사용하는 기기는 대부분 에너지가 제한된 경우가 많아 데이터를 송수신하는데 많은 에너지를 소비할 수 없고, 센서와 같은 노드의 경우 길이가 짧은 데이터를 끊임없이 전송하여야 하는 특성이 있다.

[0004] 에너지 관점에서 효율적인 통신을 위해 일반적으로 사용되는 방법이 SEC부호를 사용하는 것이다. 상기 SEC부호는 복잡도가 낮고 짧은 길이를 가지면서도 적정한 신뢰도를 보인다는 점에서 IoT 환경에 적합한 부호라고 할 수 있다.

[0005] 해밍(Hamming)부호와 같은 패리티 검사 부호(Parity Check Codes)는 조합 논리 회로 형태로 복호기를 구성할 수 있을 정도로 복호 알고리즘이 간단하여 요구되는 메모리 용량도 작을 뿐만 아니라 복호 지연도 거의 없다.

[0006] 패리티 검사 부호의 복호 알고리즘은 반복 복호 형태가 아니므로 복호 지연이 거의 없고 데이터 처리를 위한 프로세서 성능도 높은 사양을 요구하지 않는다. 따라서 에너지 자원이 많지 않은 IoT기기의 통신에 적합하다고 할 수 있다.

[0007] 그러나 해밍(Hamming)부호와 같은 패리티 검사 부호(Parity Check Codes)는 SEC부호의 일종으로, 터보 부호(Turbo Code)나 LDPC(Low Density Parity Check) 부호에 비해 낮은 비트 오율 성능을 보이는 단점을 갖는데, 이는 패리티 검사 부호 대부분이 1개의 오류를 정정하고, 많아야 2개의 오류까지 검출할 수 있는 부호이기 때문이다.

[0008] 따라서 본 발명은 송신기에서 SEC부호를 전송한 경우 수신기에서 복수의 오류가 발생하면 오류를 정정할 수 없었던 문제를 극복하여 복수의 오류에 대해서도 정정이 가능하도록 하는 멀티오류 정정이 가능한 복호기 및 그 방법을 제시한다.

[0009] 다음으로 본 발명의 기술분야에 존재하는 선행기술에 대하여 간단하게 설명하고, 이어서 본 발명이 상기 선행기술에 비해서 차별적으로 이루고자 하는 기술적 사항에 대해서 기술하고자 한다.

[0010] Salah Abdulghani Alabady 등이 제안한(Salah Abdulghani Alabady, and Fadi Al-Turjman,"Low Complexity Parity Check Code for Futuristic Wireless Networks Applications," IEEE Access, Vol. 6, 2018, pp. 18398-18407.) (n,k) SEC부호는 패리티 비트 수 (n-k) 값을 늘려 신드롬(syndrome) 벡터 수를 늘린 후 2개의 오류벡터 중 일부를 정정할 수 있도록 하여 비트 오율 성능을 늘리고자 제안하였다. 그러나 이는 늘어난 리던던시(redundancy)를 고려하면 기존의 패리티 검사 부호에 비해 SNR 대비 오율 성능은 크게 개선되었다고 볼 수 없다.

[0011] 그러나 본 발명에서는 복조기 출력단의 소프트 값을 이용하여 2개의 오류를 모두 정정할 수 있는 복호 알고리즘을 제시하고자 한다. 복조기 출력단의 소프트 값은 복호 과정 중에서 신드롬을 계산한 후 이에 해당하는 다수의 2-오류 패턴 중 최소의 제곱 합을 가지는 하나의 패턴을 구별할 때만 사용되므로 복호기 복잡도는 거의 변화가 없다.

발명의 내용

해결하려는 과제

[0012] 본 발명은 상기와 같은 문제점을 해결하기 위해 창작 된 것으로서, 송신측에서 SEC부호를 전송한 경우 수신측에서 복수의 오류가 발생하면 오류를 정정할 수 없었던 문제를 극복하여 복수의 오류에 대해서도 정정이 가능하도록 하는 복호기를 제시하는 것을 목적으로 한다.

[0013] 또한 본 발명은 SEC부호에서 멀티오류정정을 위하여 복조기의 출력 소프트 값을 활용하여 SEC부호의 BER 성능을 개선할 수 있는 복호 방식 및 복호기를 제시하는 것을 또 다른 목적으로 한다.

[0014] 또한 본 발명은 SEC부호에서 멀티오류정정을 위하여 연산량이나 메모리 용량의 증가를 가능한 줄인 상태에서

BER 성능을 개선할 수 있는 복호 방식 및 복호기를 제시하는 것을 또 다른 목적으로 한다.

과제의 해결 수단

[0015] 본 발명의 일 실시예에 따른 SEC부호의 멀티오류정정을 위한 복호기는, 복조기로부터 복조된 소프트 값을 수신하는 SEC부호 벡터 수신부; 상기 수신한 SEC부호 벡터로부터 신드롬 벡터를 추출하는 신드롬 추출부; 상기 추출한 신드롬 벡터를 이용하여 오류를 검출하는 오류 검출부; 및 상기 오류가 검출된 경우, 다수의 오류벡터에서 오류패턴을 결정하는 오류패턴 결정부;를 포함하며, 상기 오류패턴은 단일 오류는 물론 2개 이상의 오류를 정정하도록 구성되는 것을 특징으로 한다.

[0016] 상기 오류 검출부는, 상기 신드롬 벡터가 제로 벡터이면 오류가 없는 것으로 판정하고, 상기 제로 벡터가 아니면 적어도 하나 이상의 오류가 존재하는 것으로 판정하며, 상기 오류가 존재하는 것으로 판정이 되면, 단일 오류는 물론 2 개 이상의 오류를 정정할 수 있는 오류패턴을 결정하도록 요청하는 것을 특징으로 한다.

[0017] 또한 상기 오류패턴 결정부는, 상기 오류가 검출된 경우, 특정 신드롬 벡터에 해당하는 적어도 하나 이상의 각 오류패턴 중에서 오류가 존재하는 것으로 인식된 비트에 해당하는 위치의 상기 소프트 값을 제곱하여 더한 다음 그 결과가 가장 작은 값을 가지는 것을 오류패턴으로 결정하는 것을 특징으로 한다.

[0018] 상기 오류패턴에서, 단일오류만 검출되는 오류패턴인 경우, 상기 검출한 단일오류에 해당하는 소프트 값을 곱하고, 그 결과에 상기 정정할 2개 이상의 오류에 대한 개수만큼 더 곱하여 다른 오류패턴들과 그 크기를 비교함으로써, 상기 오류패턴을 결정하며, 상기 SEC부호는 해밍부호를 포함한 패리티검사부호 또는 선형블록부호를 포함하는 것을 특징으로 한다.

[0019] 또한 상기 SEC부호의 멀티오류정정을 위한 복호기는, 상기 결정한 오류패턴과 상기 수신한 SEC부호 벡터를 서로 배타적논리합(Exclusive-OR)으로 연산하는 회로를 통해서 상기 단일 오류는 물론 2개 이상의 오류를 정정하는 오류 정정부;를 더 포함하는 것을 특징으로 한다.

[0020] 한편, 본 발명의 또 다른 일 실시예에 따른 SEC부호의 멀티오류정정을 위한 복호 방법은, 복조기로부터 복조된 소프트 값을 수신하는 SEC부호 벡터의 수신 단계; 상기 수신한 SEC부호 벡터로부터 신드롬 벡터를 추출하는 신드롬 추출 단계; 상기 추출한 신드롬 벡터를 이용하여 오류를 검출하는 오류 검출 단계; 및 상기 오류가 검출된 경우, 다수의 오류벡터에서 오류패턴을 결정하는 오류패턴 결정 단계;를 포함하며, 상기 오류패턴은 단일 오류는 물론 2개 이상의 오류를 정정하도록 구성되는 것을 특징으로 한다.

[0021] 여기서 상기 오류 검출 단계는, 상기 신드롬 벡터가 제로 벡터이면 오류가 없는 것으로 판정하고, 상기 제로 벡터가 아니면 적어도 하나 이상의 오류가 존재하는 것으로 판정하며, 상기 오류가 존재하는 것으로 판정이 되면, 단일 오류는 물론 2개 이상의 오류를 정정할 수 있는 오류패턴을 결정하도록 요청하는 것이며, 상기 오류패턴 결정 단계는, 상기 오류가 검출된 경우, 특정 신드롬 벡터에 해당하는 적어도 하나 이상의 각 오류패턴 중에서 오류가 존재하는 것으로 인식된 비트에 해당하는 위치의 상기 소프트 값을 제곱하여 더한 다음 그 결과가 가장 작은 값을 가지는 것을 오류패턴으로 결정하는 것을 특징으로 한다.

[0022] 또한 상기 SEC부호의 멀티오류정정을 위한 복호 방법은, 상기 결정한 오류패턴과 상기 수신한 SEC부호 벡터를 서로 배타적논리합으로 연산하는 회로를 통해서 상기 단일 오류는 물론 2개 이상의 오류를 정정하는 오류 정정 단계;를 더 포함하는 것을 특징으로 한다.

발명의 효과

[0023] 이상에서와 같이 본 발명의 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법은 SEC부호로 부호화된 데이터에 복수의 오류가 발생한 경우 단일 오류 이상을 정정할 수 없었던 문제를 극복하여, 복수의 오류에 대해서도 정정이 가능하도록 하는 효과가 있다. 특히 본 발명에 따른 복호기와 그 복호 방법은 복수의 오류정정을 위해 연산량이나 메모리 용량의 증가를 가능한 적게 한 상태에서 BER 성능 개선을 얻을 수 있으므로 에너지와 연산 자원이 부족한 IoT 디바이스에서 데이터 송수신에 활용될 수 있는 장점이 있다.

도면의 간단한 설명

[0024] 도 1은 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법을 설명하기 위한 개념도이다.

도 2는 본 발명에서 채용하는 단일 오류 정정을 위한 SEC부호의 복호 기법을 설명하기 위한 도면이다.

도 3은 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호에서 복조기 출력단의 신호에 대한 전력분포를 조사한 결과를 나타낸 도면이다.

도 4는 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호에서 복조기 출력단의 소프트 값에 대한 특성을 조사한 결과를 나타낸 도면이다.

도 5는 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법을 나타낸 도면이다.

도 6은 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법에 해밍코드를 적용한 경우 AWGN 채널에서 BER 성능을 시뮬레이션한 결과이다.

도 7은 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법에 선형블록코드를 적용한 경우 BER 성능을 시뮬레이션한 결과이다.

도 8은 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법에 3개의 오류패턴까지 적용한 경우 BER 성능의 개선여부를 시뮬레이션한 결과이다.

발명을 실시하기 위한 구체적인 내용

[0025] 이하, 첨부된 도면을 참조하여 본 발명의 다양한 실시예를 상세히 설명하기로 한다. 본 발명의 명세서 또는 출원에 개시되어 있는 일 실시예들에 대해서 특정한 구조적 내지 기능적 설명들은 단지 본 발명에 따른 실시예를 설명하기 위한 목적으로 예시된 것으로, 다르게 정의 되어 있지 않는 한, 기술적이거나 과학적인 용어를 포함해서 본 명세서에서 사용되는 모든 용어들은 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가지고 있다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥상 가지는 의미와 일치하는 의미를 가지는 것으로 해석되어야 하며, 본 명세서에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 아니한다.

[0026] 먼저 종래기술에 따라 1개의 오류를 정정하기 위한 SEC부호의 복호에 대해서 설명하고자 한다. (n,k) 선형 블록 부호(LBC, linear block code)의 오류 정정 능력 t는 선형 블록 부호의 최소 해밍 거리(Minimum Hamming Distance) d_{min}에 의해 [수학식 1]과 같이 정해진다.

[0027] $$t \leq \left\lfloor \frac{(d_{min}-1)}{2} \right\rfloor$$ [수학식 1]

[0028] 이때 $\lfloor x \rfloor$ 는 가우스 기호를 의미하며 x보다 크지 않은 최대 정수를 말한다. 따라서 $d_{min} \leq 4$인 선형 블록 부호의 오류 정정 능력은 $t \leq 1$이므로 부호 길이 n-비트 내에 1개의 오류 까지만 확실하게 정정할 수 있고, 2개 이상의 오류는 정정을 보장하지 않는다. 이렇게 1개의 단일 오류를 정정할 수 있는($d_{min} \leq 4$ 인) 선형 블록 부호를 단일오류정정(SEC, single error correction)부호라고 한다. SEC부호는 비록 오류 정정 능력이 작지만 부/복호기의 복잡도가 낮아 오류 발생 확률이 작은 유선 통신이나 시스템의 복잡도가 낮아야 하는 통신 분야에 널리 사용되고 있다. 가장 대표적인 SEC부호의 예로 $d_{min}=3$인 해밍부호를 들 수 있으며, 해밍부호는 유선 통신에서 ARQ의 오류 검출 부호로 사용되거나 GPS(Global Positioning System) 전송 데이터의 오류 정정 부호로 사용되고 있다.

[0029] 이어서 본 발명의 일 실시예에 따른 멀티 오류정정을 위한 SEC부호의 부호화 및 복호화에 대한 개념에 대해서 설명하고자 한다.

[0030] 도 1은 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법을 설명하기 위한 개념도이다.

[0031] 도 1에 도시한 바와 같이, 통신에서 전송오류나 저장매체의 결함으로 인해 손상된 데이터를 복원하는 것은, 특정 소스 메시지(m)를 채널 코딩(c)한 다음 변조하고, 변조한 데이터를 채널로 전송하거나 저장매체에 저장하면, 수신측에서 채널에서 생긴 오류나 저장매체의 결함으로 인해 손상된 데이터를 수신하게 되고, 수신된 데이터는 복조를 통해 채널 코딩된 데이터를 추출하고, 복호기는 상기 추출한 데이터를 채널 디코딩하여 오류를 정정한 다음 소스 메시지를 복원하게 된다.

[0032] 상기 (n,k) SEC부호기(100)는 (n,k) 선형 블록 코드(Linear Block Code)에 대해서, k-비트 블록을 의미하고, n은 k-비트 블록을 n-비트 블록으로 매핑하여 출력하는 채널코딩의 결과에 대한 비트폭을 의미한다. 즉, 입력 소스 메시지(m)를 k-비트 단위로 잘라서 n-비트의 블록으로 매핑하여 부호화함으로써 오류정정부호화 한다. 종래에는 SEC부호를 이용한 단일오류정정을 위해서, k-비트의 메시지에 추가적인 비트를 더하여 n-비트로 부호화하여 데이터를 전송하거나 저장하였다. 채널코딩 과정에서 k-비트로부터 n-비트를 생성하기 위한 생성행렬(G)이 사용된다. 즉, (n,k) SEC인코더(100)는 생성행렬(G)를 사용하여 k-비트의 입력을 n-비트 출력으로 매핑함으로써 채널코딩을 수행하는 역할을 한다(c = mG).

[0033] 상기 채널코딩된 데이터(c)는 다시 채널전송부(200)에서 BPSK(binary phase shift keying) 변조기(210)로 변조된 다음 통신 채널(AWGN, additive white Gaussian noise)(220)이나 저장매체를 통해서 전송하거나 저장하게 된다. 이 경우, 전송해야할 데이터나 저장할 데이터가 소스 메시지에 비해서 늘어나게 된다. 그러나 오류로 인한 데이터의 손실에 대응할 수 있다. 이렇게 전송되거나 저장된 데이터를 수신측에서 수신한 다음 BPSK 복조기(230)를 통해서 복조하여 수신 데이터(r)를 추출한다. 상기 추출한 수신 데이터(r)에 대해서 패리티 체크 매트릭스(H)를 적용하여 채널 복호화를 수행하면 소스 메시지($\hat{m}$)가 추출된다. 복호화한 메시지가 소스 메시지와 동일하면 전송과정이나 저장관리하는 과정에서 오류가 발생하지 않은 것이고, 만약 오류가 1개 발생하더라도 이는 정정을 통해서 감지 및 수정이 가능하다.

[0034] 여기서 생성행렬(G)과 패리티 검사 매트릭스(H)의 전치행렬을 곱하면 0이 되는 관계가 있다($GH^T=0$).

[0035] (n,k) SEC복호기(300)에서는 신드롬(s)을 먼저 계산하고 다시 신드롬을 디코딩하여 최종적으로 소스 메시지를 복원한다. 즉, $s=rH^T=(c+e)H^T=mGH^T+eH^T=eH^T$가 된다. 여기서 e는 오류이다. 따라서 신드롬(s)은 오류(e)에 의해 결정된다. 만일 오류(e)가 없다면(e=0), s=0이 되고, s≠0이면, 오류(e)가 존재(e≠0)함을 알 수 있다. 이 과정이 오류를 검출하는 과정이다.

[0036] 상기 신드롬을 디코딩하는 것은 먼저 오류패턴을 찾고, 이로부터 오류를 제거함으로써 수행된다. 즉, $\hat{c}=r-e=r\oplus e$가 된다. 이러한 관계식으로부터 (n,k) SEC복호기(300) 회로를 그린다고 생각하면, (n,k) SEC복호기(300)로 입력되는 수신데이터 r=c+e에 H^T를 곱하여 신드롬(s)을 구하고, 상기 신드롬을 디코딩하여 오류를 제거할 때, 오류 패턴에 상기 수신데이터를 배타적논리합(Exclusive-OR)으로 연산하여 소스 메시지($\hat{m}$)를 추출하는 것으로 회로를 구성하면 된다.

[0037] 도 1의 우측에서 보인 바와 같이, 통신채널(AWGN, additive white Gaussian noise)(220), BPSK변조기(210) 및 BPSK복조기(230)에서 발생하는 현상은 SEC부호(100)의 출력이 채널을 통해서 전송될 때 오류(e)가 발생하였으며, BPSK복조기(230)에서 출력되는 신호는 (r=c+e)인 관계가 있다.

[0038] 이하에서는 (n,k) SEC부호에 대한 복호의 개념을 좀 더 자세하게 설명하고자 한다.

[0039] 도 2는 본 발명에서 채용하는 단일 오류 정정을 위한 SEC부호의 복호 기법을 설명하기 위한 도면이다.

[0040] (n,k) SEC부호는 선형 블록 부호로 [수학식 2]와 같은 생성행렬 G를 이용하여 k-비트의 메시지 벡터 $m=(m_1,m_2,\ldots,m_k)$로부터 n-비트의 부호벡터 $c=(c_1,c_2,\ldots,c_n)$를 얻을 수 있다. 부호벡터 c는 행렬 연산 c=mG를 통해 얻을 수 있는데 c의 처음 k-비트는 메시지 비트가 되며 나머지 (n-k) 비트는 c의 뒷부분에 위치하는 체계적 형태를 갖게 된다. [수학식 2]는 (7, 4) SEC부호의 생성행렬을 나타낸 것이다.

$$G=[P\,I_k]=\begin{bmatrix}1&1&0&1&0&0&0\\0&1&1&0&1&0&0\\1&1&1&0&0&1&0\\1&0&1&0&0&0&1\end{bmatrix}$$

[0041] [수학식 2]

[0042] 부호벡터 c는 변조 과정을 거쳐 채널로 전송되고 수신단에서 다시 복조 과정을 거쳐 복조기 출력부(310)를 통해 출력하고, 수신 부호 벡터 $r=(r_1,r_2,\ldots,r_n)$이 복호기로 입력된다. 복조기 출력 값은 n-개의 아날로그 값을 갖게 되며 본 발명에서는 이를 소프트 벡터 $q=(q_1,q_2,\ldots,q_n)$라고 한다. n-개의 소프트 값은 실수 값이며 경판정(hard decision)을 통해 이진 값 {0,1}으로 변환되어 수신 부호 벡터 r을 형성한다. 이러한 과정은 부호 벡터 수신부(320)에서 수행한다. 경판정 과정에서 오류가 발생하게 되며 이 때 발생한 오류 벡터를 $e=(e_1,e_2,\ldots,e_n)$으로 표

현하기로 한다. 여기에서 e_i의 값은 0 혹은 1의 값을 가지며 0이면 오류가 발생하지 않은 것이고 1이면 오류가 발생한 것을 의미한다. 이제 수신 부호 벡터 r은 부호 벡터 c와 오류벡터 e를 이용해 [수학식 3]과 같이 나타낼 수 있다.

[0043] $r = c + e$ [수학식 3]

[0044] (n,k) SEC부호의 복호(decoding) 과정은 패리티 검사 행렬 H를 통해 이루어지는데 생성행렬 G와 H가 항상 [수학식 4]가 성립하는 관계를 갖도록 H를 구성한다. [수학식 5]는 [수학식 4]의 관계가 성립하는 (7, 4) SEC부호의 패리티 검사 행렬을 나타낸 것이다.

[0045] $GH^T=0$ [수학식 4]

[0046] $$\boldsymbol{H}=[\boldsymbol{I}_{n-k}\boldsymbol{P}^T]=\begin{bmatrix}1&0&0&1&0&1&1\\0&1&0&1&1&1&0\\0&0&1&0&1&1&1\end{bmatrix}$$ [수학식 5]

[0047] 수신 부호 벡터 r이 복호기에 입력되면 복호기는 [수학식 6]의 연산을 통해 (n-k)-비트로 구성되는 신드롬 벡터 $s=(s_1,s_2,\ldots,s_{n-k})$를 얻는다. 이러한 과정은 신드롬 추출부(330)에서 수행된다.

[0048] $s=rH^T$ [수학식 6]

[0049] [수학식 6]에 [수학식 3]을 대입하고, [수학식 4]를 이용하면, [수학식 7]을 얻을 수 있으며, $c=e\cdot H^T$를 이용해 각각의 신드롬에 해당하는 m-오류 패턴을 생성할 수 있다. 이 과정은 오류검출 및 오류패턴 결정부(340)에서 수행된다.

[0050] $$\begin{aligned}\boldsymbol{s}=\boldsymbol{rH}^T&=(\boldsymbol{c}+\boldsymbol{e})\boldsymbol{H}^T=\boldsymbol{cH}^T+\boldsymbol{eH}^T\\&=\boldsymbol{mGH}^T+\boldsymbol{eH}^T=\boldsymbol{eH}^T\end{aligned}$$ [수학식 7]

[0051] [수학식 7]에서 나타낸 바와 같이, 신드롬(s)은 오류벡터(e)와 패리티 검사 행렬(H)의 전치행렬을 곱해서 산출되므로, 조합논리회로를 이용하여 구현할 수 있다.

[0052] 또한 신드롬(s)으로부터 오류를 검출하고 단일오류에 대해서 룩업-테이블을 이용하여 오류패턴을 결정한 다음 수신한 부호벡터와 배타적논리합을 통해서 송신한 소스 메시지($\hat{m}$)를 복원할 수 있다. 이 과정은 오류 정정부(350)에서 수행된다.

[0053] 이어서 종래의 단일 오류 정정을 위한 SEC부호에 대해서 복수의 오류가 발생할 경우 오류를 정정하지 못하는 문제에 대해서 논의하고, 이에 대한 해결방안을 제시하고자 한다.

[0054] [수학식 7]에서 오류가 발생하지 않으면 e=0이므로 s=0인 벡터가 될 것이다. 따라서 $s\neq0$이면 오류가 발생한 경우이며 [표 1]과 같은 룩업-테이블(look-up table)을 이용하여 오류를 정정하게 된다. [표 1]은 단일 오류패턴에 대한 룩업-테이블인데, 만약 복수의 오류가 생기면 [표 1]로는 오류를 정정할 수 없게 되며 단지 오류 발생 유·무만을 알 수 있다. 이 과정이 오류검출 과정이다.

[0055] 따라서 [표 2]의 2-오류패턴에 대해서 고려해 볼 필요가 있다. 예를 들어, (7, 4) 해밍부호의 경우 m-오류 패턴($m\leq2$)과 이에 대응되는 신드롬을 [수학식 7]을 이용해 구해 보면 [표 1] 및 [표 2]와 같다. 앞서 살펴본 바와 같이 [표 1]은 (7, 4) 해밍코드의 신드롬에 단일 오류벡터 만을 매핑한 룩업-테이블이다. 각 신드롬 별로 대응되는 [표 1]의 단일 오류 패턴과 [표 2]의 2-오류 패턴을 정리해 보면 [표 3]과 같다. 예를 들어 신드롬 s=(100)에 해당하는 m-오류 패턴($m\leq2$)은 [0000100], [0110000], [1000010], [0001001]의 4개임을 알 수 있다. 따라서 신드롬 s=(100)가 추출되었다고 해서 오류 패턴이 단일 오류인 [0000100]이라고 단정 지을 수 없다. 한 가지 특이한 점은 (7, 4) 해밍부호의 경우 신드롬 (000)에 대응되는 오류 패턴을 (0000000)의 한 가지 뿐임을 알 수 있다.

[0056] [표 1] 1-오류 패턴과 해당 신드롬

오류 패턴 e	신드롬 s
[0000000]	[000]
[1000000]	[110]
[0100000]	[011]
[0010000]	[111]
[0001000]	[101]
[0000100]	**[100]**
[0000010]	[010]
[0000001]	[001]

[0057]

[0058] [표 2] 2-오류 패턴과 해당 신드롬

오류 패턴 e	신드롬 s
[1100000]	[101]
[1010000]	[001]
[1001000]	[011]
[1000100]	[010]
[1000010]	**[100]**
[1000001]	[111]
[0110000]	**[100]**
[0101000]	[110]
⋮	⋮
[0000011]	[011]

[0059]

[0060] [표 3] 신드롬에 대응되는 m-오류 패턴(m≤2)

신드롬 s	오류 패턴 e
[000]	**[0000000]**
[001]	[1010000] [0001100] [0100010] **[0000001]**
[010]	[0011000] [1000100] **[0000010]** [0100001]
[011]	**[0100000]** [1001000] [0010100] [0000011]
[100]	[0110000] **[0000100]** [1000010] [0001001]
[101]	[1100000] **[0001000]** [0010010] [0000101]
[110]	**[1000000]** [0101000] [0000110] [0010001]
[111]	**[0010000]** [0100100] [0001010] [1000001]

[0061]

[0062] [표 3]은 m-오류 패턴(m≤2)에 대한 해밍 (7, 4) 코드의 룩업-테이블로 사용할 수 있다. [표 3]을 참조하면, (7, 4) SEC코드에서 신드롬 벡터 s가 0벡터가 아닐 경우 [표 3]의 룩업-테이블을 이용하여 단일 오류 전부 및 2-오류 패턴을 복원할 수 있다.

[0063] [수학식 7]에 의해 계산된 신드롬 벡터 s가 0벡터가 아닐 경우 [표 3]의 룩업-테이블을 이용하여 단일 오류와 2-오류 패턴을 복원할 수 있다. 단일 오류 패턴의 경우에는 대응되는 신드롬 벡터가 유일(unique)하므로 단일 오류 전부를 정정할 수 있다. 그러나 2-오류 패턴의 경우에는 신드롬에 해당하는 오류 패턴이 다수 존재하므로 정확히 선별하지 못하고 오류 정정을 할 경우 오히려 오류를 더하는 결과를 초래할 수도 있다.

[0064] [표 3]을 보면 알 수 있듯이, 하나의 신드롬 벡터에 두 개 이상의 2-오류 패턴이 존재하는 경우에 대하여도 본 발명은 이를 해결하여 오류 정정 능력을 개선시킬 수 있는 복호 방식을 제시한다.

[0065] 이어서 본 발명에서 기존의 단일 오류 정정이 가지는 한계를 극복하고 복수의 오류에 대해서도 복잡도가 증가하지 않는 선에서 오류정정이 가능하도록 하는 방법에 대한 착안점에 대해서 설명하고자 한다.

[0066] (n,k) SEC부호에 대해서, SEC복호기로 입력되는 수신 부호 벡터 r의 비트 값들은 복조기(demodulator)에 의해 복원된 아날로그 신호 값을 경판정(hard decision) 과정을 통해 이진 값으로 변환하여 얻는다. 경판정 과정에서 많은 채널 정보가 사라지게 되며 이는 복호기의 성능 열화로 이어진다.

[0067] 이를 보완하기 위해 터보 부호나 LDPC 부호의 복호에서는 연 판정(soft decision) 값을 이용하여 복호 성능을 개선하고 있다. 하지만 블록 부호의 패리티 체크 부호의 경우에는 신드롬 계산이나 복호를 위한 룩업-테이블을 이용한 오류 정정 과정이 경 판정 값을 기초로 이루어지므로 연 판정 값을 복호 과정에 사용할 수 없다. 더욱이 터보 부호나 LDPC 부호에서 연 판정 값을 이용하여 복호 연산을 하는 것은 과도한 연산 능력이 요구되기 때문에 에너지가 제한되어 있고 연산 능력이 부족한 IoT 기기의 통신에 적합하지도 않다.

[0068] 따라서 본 발명에서는 복조기의 아날로그 출력 값(이하, 소프트 값이라고 함)에 대해 경 판정을 한 후 이를 버리지 않고 복호기(decoder)에 수신 부호 벡터 r과 함께 전해 주어 복호 성능을 개선할 수 있는 복호 알고리즘을 제시한다. 여기서 편의 상 n개의 소프트 값으로 이루어진 벡터를 $q=(q_1,q_2,\ldots,q_n)$으로 표시한다.

[0069] 도 3은 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법을 위해서 복조기 출력단의 신호에 대한 전력분포를 설명하기 위한 도면이다.

[0070] 도 3에 도시된 바와 같이, $E_b/N_0=8$[dB] 및 $E_b/N_0=10$[dB]의 AWGN 채널에서 BPSK 시스템의 복조기 출력 중 오류 비트와 올바른 비트에 해당하는 소프트 값의 전력 분포가 나타나 있다. 도 3에서 오류 비트에 해당하는 소프트 값의 전력은 거의 0에 근접한 값을 보이며 올바른 비트에 해당하는 소프트 값의 전력은 오류 비트의 소프트 값 전력에 비해 월등히 높은 값을 가진다. 특히 $E_b/N_0=10$[dB]의 경우 더욱 그렇다. 이는 오류 비트에 해당하는 소프트 값의 경우 임계 치(BPSK의 경우 0)를 넘어야 오류가 되기 때문에 오류 비트에 해당하는 소프트 전력 값이 작을 확률이 크다.

[0071] 도 4는 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법을 위해서 복조기 출력단의 소프트 값에 대한 특성을 설명하기 위한 도면이다.

[0072] 도 4에 도시된 바와 같이, SNR(혹은 E_b/N_0) 변화에 따른 오류 비트와 올바른 비트의 소프트 값의 평균 전력 변화를 파악할 수 있다. 전 영역에 걸쳐 두 평균 전력 값 차이는 줄어들지 않음을 볼 수 있다. SNR이 증가할수록 오류 비트에 해당하는 소프트 값의 전력은 0을 향해 수렴하며 올바른 비트에 해당하는 소프트 값의 전력은 1에 수렴함을 알 수 있다. 따라서 소프트 값의 전력은 해당 비트가 오류 비트인지 올바른 비트인지 구분할 수 있는 보조 정보를 담고 있으며, 이를 SEC부호의 복호에 이용할 수 있다.

[0073] 이하에서는 이러한 복조기 출력단의 소프트값에 대한 특성을 이용하여, 본 발명에서 제시하는 복조기 및 복조방법에 대해서 설명하고자 한다.

[0074] 도 5는 본 발명의 일 실시예에 따른 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법을 설명하기 위한 도면이다.

[0075] 먼저 (n,k) SEC부호는 선형 블록 부호로 생성행렬 G를 이용하여 k-비트의 메시지 벡터 $m=(m_1,m_2,\ldots,m_k)$로부터 n-비트의 부호벡터 $c=(c_1,c_2,\ldots,c_n)$를 얻는다. 부호벡터 c는 행렬 연산 $c=mG$를 통해 얻는다.

[0076] 부호벡터 c는 변조 과정을 거쳐 채널로 전송되고 수신단에서 다시 복조 과정을 거쳐 수신 부호 벡터 $r=(r_1,r_2,\ldots,r_n)$이 복호기로 입력된다. 이러한 과정은 복조기 출력부(410)를 통해서 수행된다. 복조기 출력 값은 n-개의 아날로그 값을 갖게 되며 본 발명에서는 이를 소프트 벡터 $q=(q_1,q_2,\ldots,q_n)$라고 한다. n-개의 소프트 값은 실수 값이며 경판정(hard decision)을 통해 이진 값 {0,1}으로 변환되어 수신 부호 벡터 r을 형성한다. 이러한 부호벡터의 수신은 부호 벡터 수신부(420)에서 수행된다. 경판정 과정에서 오류가 발생하게 되며 이때 발생한 오류 벡터를 $e=(e_1,e_2,\ldots,e_n)$으로 표현하기로 한다. 여기에서 e_i의 값은 0 혹은 1의 값을 가지며 0이면 오류가 발생하지 않은 것이고 1이면 오류가 발생한 것을 의미한다. 수신 부호 벡터 r은 부호 벡터 c와 오류벡터 e를 이용해 $r=c+e$와 같이 표현할 수 있다.

[0077] (n,k) SEC부호의 복호(decoding) 과정은 패리티 검사 행렬 H를 통해 이루어지는데 생성행렬 G와 H가 항상 $GH^T=0$가 성립하는 관계를 갖도록 H를 구성한다.

[0078] 수신 부호 벡터 r이 복호기에 입력되면 복호기는 $s=rH^T$의 연산을 통해 (n-k)-비트로 구성되는 신드롬 벡터 $s=(s_1,s_2,\ldots,s_{n-k})$를 얻는다. 이 과정은 신드롬 추출부(430)에서 수행된다.

[0079] 이 과정에서 오류가 발생하지 않으면 e=0이므로 s=0인 벡터가 될 것이다. 따라서 $s\neq0$이면 오류가 발생한 경우이며 [표 3]과 같은 룩업-테이블(look-up table)을 이용하여 오류를 정정하게 된다. [표 3]은 m-오류 패턴($m\leq2$)에 대한 룩업-테이블이며, 2-오류가 생긴 경우에도 오류를 정정할 수 있다. 여기서 오류를 검출하는 과정은 오류 검출부(440)에서 수행된다.

[0080] 우선 s=(010)인 경우, $s\neq0$이므로, $e\neq0$임을 알 수 있다. 이 경우 s=(010)에 해당하는 오류벡터의 룩업-테이블에서 [수학식 8]을 계산한다. 여기에서 최소값을 가지는 항에 대한 오류패턴을 선택한다. 상기 선택된 오류벡터

에 대해서 오류를 정정한다.

[0081] $$\mathrm{Min}\{(q_{13}^2+q_{14}^2),(q_{21}^2+q_{25}^2),(q_{36}^2\times 2),(q_{42}^2+q_{47}^2)\}$$ [수학식 8]

[0082] 상기 설명한 2-오류 정정 방법은 신드롬 벡터에 해당하는 오류벡터에 대해서 각 오류패턴에 해당하는 수신부호의 아날로그 값을 제곱하여 더한 값이 가장 작은 오류벡터를 최종적인 오류패턴으로 결정하여 오류를 정정하는 것이다. 여기서 오류패턴을 결정하는 것은 오류패턴 결정부(450)에서 수행하고, 결정한 오류패턴과 수신한 부호벡터를 배타적논리합으로 연산하여 오류를 정정할 수 있다. 이러한 오류정정은 오류 정정부(350)와 같이 별도로 구비하여 수행할 수 있다.

[0083] 신드롬 벡터 '에 해당하는 2-오류 벡터가 3개 이상 존재하여도 소프트 값의 전력이 가장 작은 오류 패턴을 선정하면 된다. 즉, [수학식 8]의 계산은 하나의 신드롬 벡터가 0 벡터가 아닌 경우에만 실행되며 계산량도 극히 적으므로 이로 인해 증가되는 복잡도는 거의 없다고 볼 수 있다. SEC부호의 부호 길이 n이 작으므로 소프트 값을 저장해야 하는 메모리 용량도 n개 이하이므로 복잡도에 미치는 영향은 거의 없다고 할 수 있다.

[0084] 이하에서는 본 발명에 따른 멀티 오류 정정을 통한 복호기의 성능에 대해서 설명하고자 한다.

[0085] 도 6은 시뮬레이션을 통해 얻은 (7, 4) 해밍부호, (15, 11) 해밍부호에 대한 기존 복호 방식과 본 발명에 따른 2-오류 정정을 위한 복호 방식의 BER 성능을 나타낸 것이다. 도 6에서 (7, 4) 해밍부호와 (15, 11) 해밍부호에 관계없이 본 발명에서 제시한 복호 방식은 종래의 복호 방식에 비해 10^{-4}의 비트 오율에서 약 1.1[dB] ~ 1.2[dB]의 성능 개선을 확인할 수 있다.

[0086] 도 7은 선형 블록 코드에 대해서 시뮬레이션한 것으로, (6, 3) 선형 블록 코드의 경우에도 복호 방식이 동일하므로 동일한 결과를 나타낸다.

[0087] 다만, 도 8에 도시한 바와 같이, 3-오류 패턴에 대한 오류정정을 위한 BER 성능은 2-오류정정의 경우에 비해서 성능의 개선이 거의 이루어지지 않았다. 이는 (7, 4) 해밍부호와 같이 부호의 길이가 짧은 경우에 2-오류 이상이 발생할 확률이 높지 않고, 따라서 그 성능도 제한적이라고 판단된다. 그러나 부호의 길이가 길어질 경우 3-오류 이상의 오류 정정에 대한 성능이 높아질 것으로 예측된다.

[0088] 이상에서 설명한 바와 같이 2-오류 정정을 위한 복호기의 경우, 복잡도나 메모리의 증가가 거의 없이 얻은 것이므로 향 후 낮은 복잡도와 복호 지연이 없어야 하는 IoT 기기의 오류 정정 부호 방식으로 유용할 것으로 판단된다.

[0089] 이상에서 설명한 바와 같이, 본 발명의 SEC부호에서 멀티오류정정을 위한 복호기 및 그 복호 방법은 SEC부호로 부호화된 데이터에 복수의 오류가 발생한 경우 단일 오류 이상을 정정할 수 없었던 문제를 극복하여, 복수의 오류에 대해서도 정정이 가능하도록 하는 효과가 있다.

[0090] 특히 본 발명에 따른 복호기와 그 복호 방법은 복수의 오류정정을 위해 연산량이나 메모리 용량의 증가를 가능한 적게 한 상태에서 BER 성능 개선을 얻을 수 있으므로 에너지와 연산 자원이 부족한 IoT 디바이스에서 데이터 송수신에 활용될 수 있다.

[0091] 상기에서는 본 발명에 따른 바람직한 실시예를 위주로 상술하였으나, 본 발명의 기술적 사상은 이에 한정되는 것은 아니며 본 발명의 각 구성요소는 동일한 목적 및 효과의 달성을 위하여 본 발명의 기술적 범위 내에서 변경 또는 수정될 수 있을 것이다.

[0092] 아울러 이상에서는 본 발명의 바람직한 실시예에 대하여 도시하고 설명하였지만, 본 발명은 상술한 특정의 실시예에 한정되지 아니하며, 청구범위에서 청구하는 본 발명의 요지를 벗어남이 없이 당해 발명이 속하는 기술분야에서 통상의 지식을 가진 자에 의해 다양한 변형 실시가 가능한 것은 물론이고, 이러한 변형 실시들은 본 발명의 기술적 사상이나 전망으로부터 개별적으로 이해되어서는 안 될 것이다.

부호의 설명

[0093] 100: (n, k) SEC부호기 200: 채널송수신부

210: BPSK 변조기 220: AWGN 채널

230: BPSK 복조기

300: (n, k) SEC복호기

310, 410: 복조기 출력부

320, 420: 부호 벡터 수신부

330, 430: 신드롬 추출부

340: 오류검출 및 오류패턴 결정부

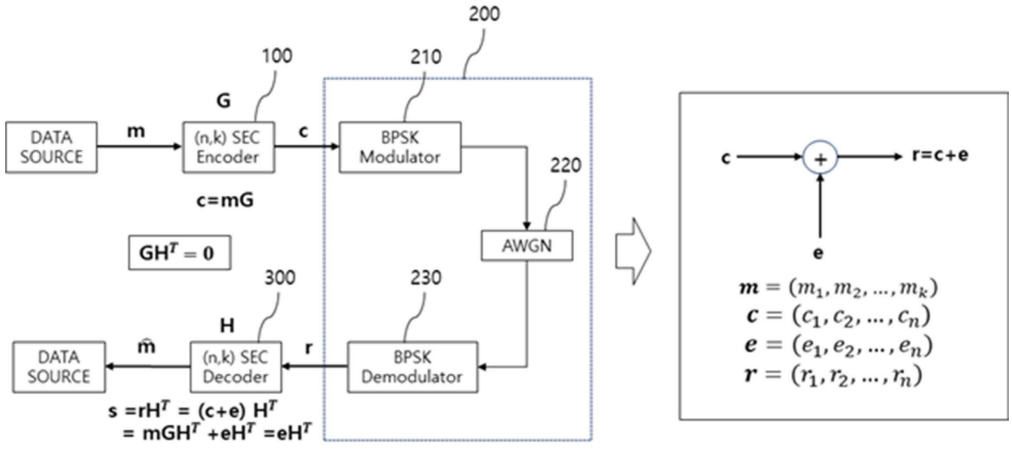
350: 오류 정정부

440: 오류 검출부

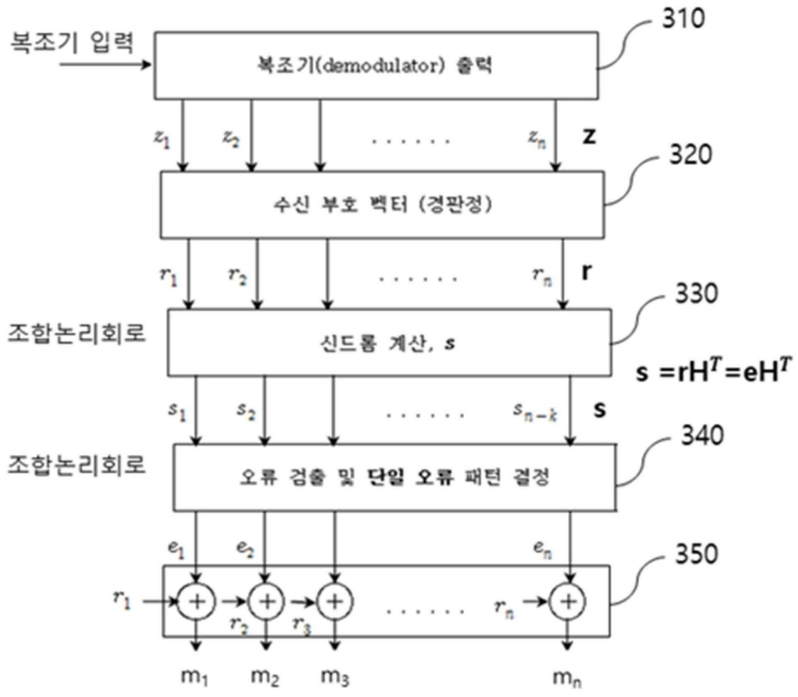
450: 오류패턴 결정부

도면

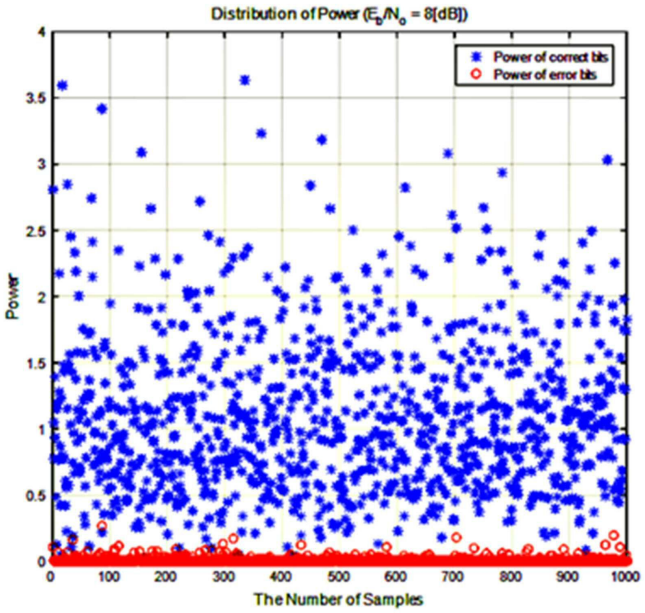
도면1



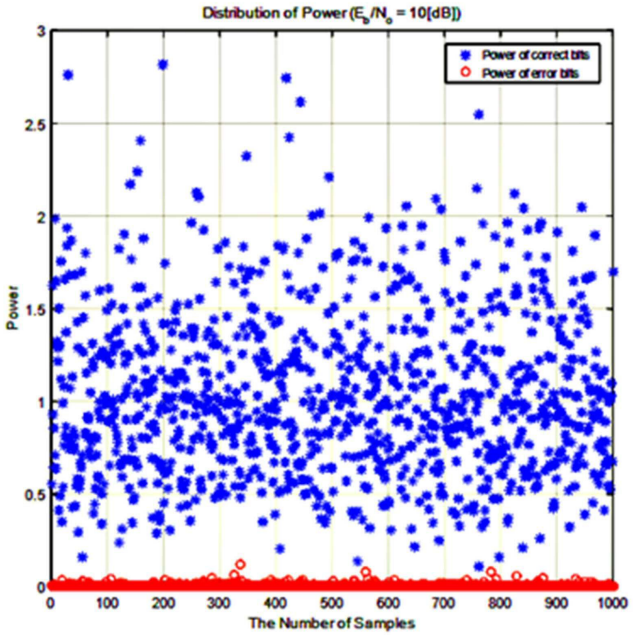
도면2



도면3

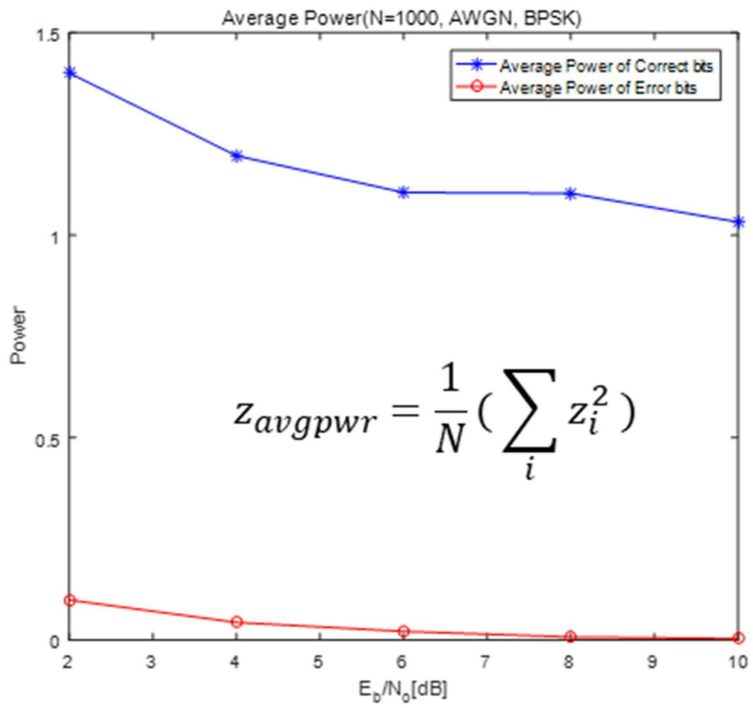


$$\frac{E_b}{N_0} = 8\ [dB], \qquad z_i^2$$

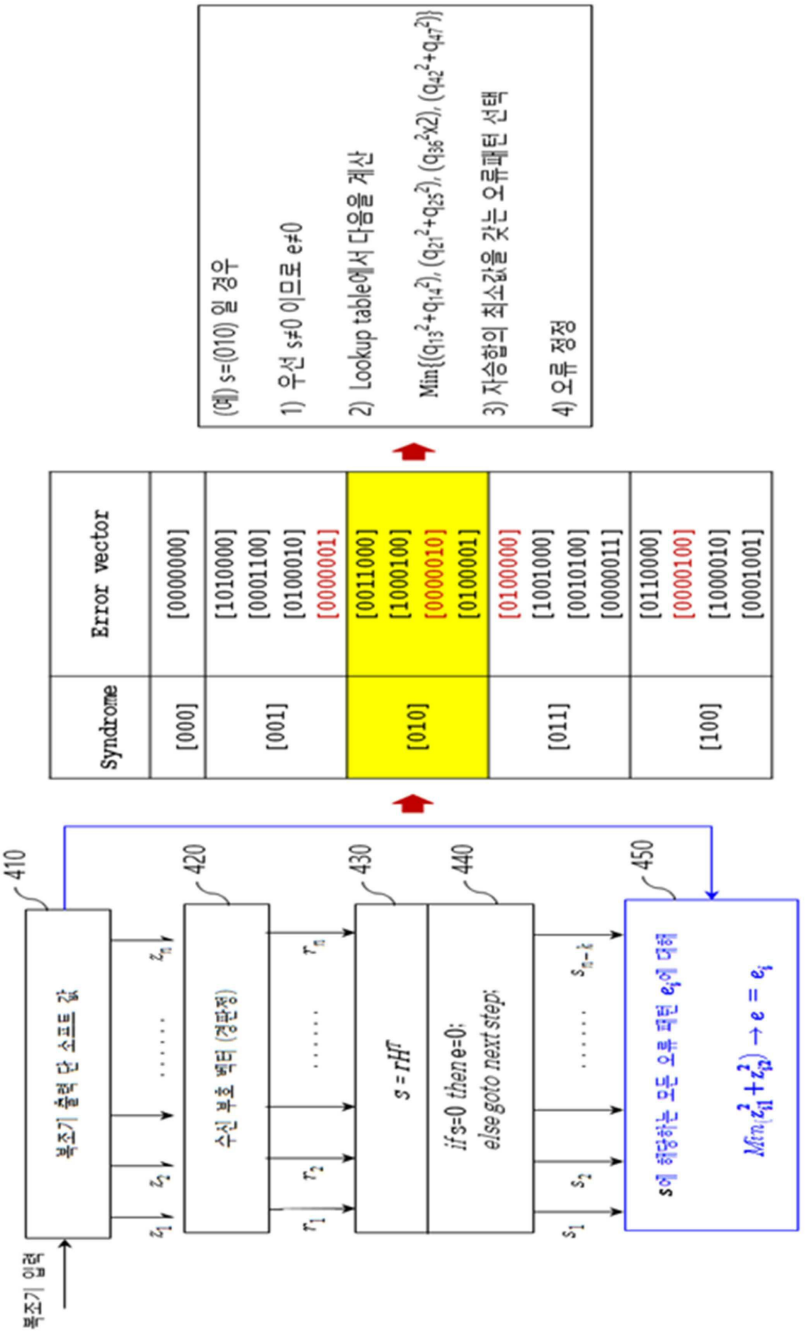


$$\frac{E_b}{N_0} = 10\ [dB], \qquad z_i^2$$

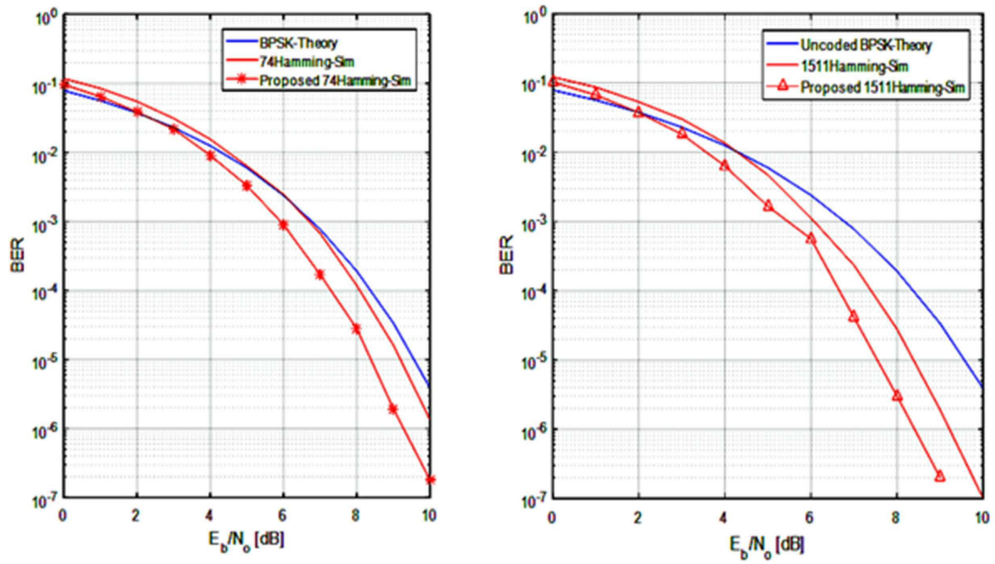
도면4


Average Power(N=1000, AWGN, BPSK)
Average Power of Correct bits
Average Power of Error bits
Power
1.5
1
0.5
0
2
3
4
5
6
7
8
9
10
Eb/N0[dB]
$z_{avgpwr} = \frac{1}{N}(\sum_i z_i^2)$

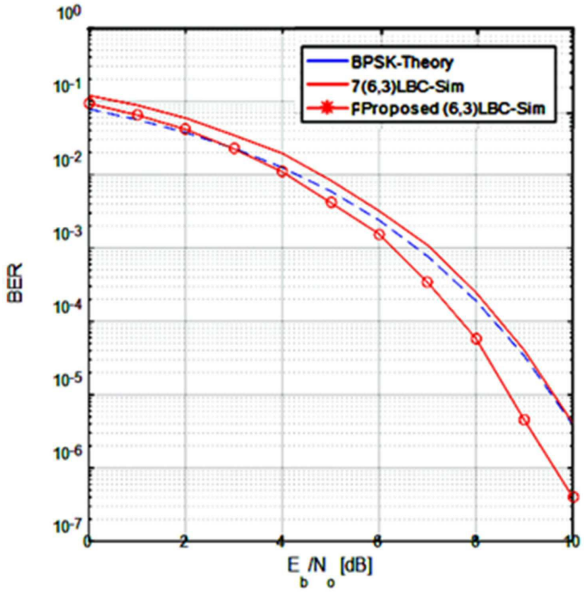
도면5


(예) s=(010) 일 경우
1) 우선 s≠0 이므로 e≠0
2) Lookup table에서 다음을 계산
Min{(q13^2+q14^2), (q21^2+q23^2), (q36^2X2), (q42^2+q47^2)}
3) 자승합의 최소값을 갖는 오류패턴 선택
4) 오류 정정
Syndrome
Error vector
[000]
[00000000]
[001]
[10100000]
[00011000]
[01000010]
[00000001]
[010]
[00110000]
[11000100]
[00000010]
[01000001]
[011]
[01000000]
[10010000]
[00101000]
[00000110]
[100]
[01100000]
[00001000]
[10000100]
[00010001]
복조기 입력
복조기 출력 단 소프트 값
수신 부호 벡터 (경판정)
s = rH^T
if s=0 then e=0;
else goto next step;
s에 해당하는 모든 오류 패턴 e_i에 대해
Min(z_i1^2 + z_i2^2) → e = e_i
410
420
430
440
450

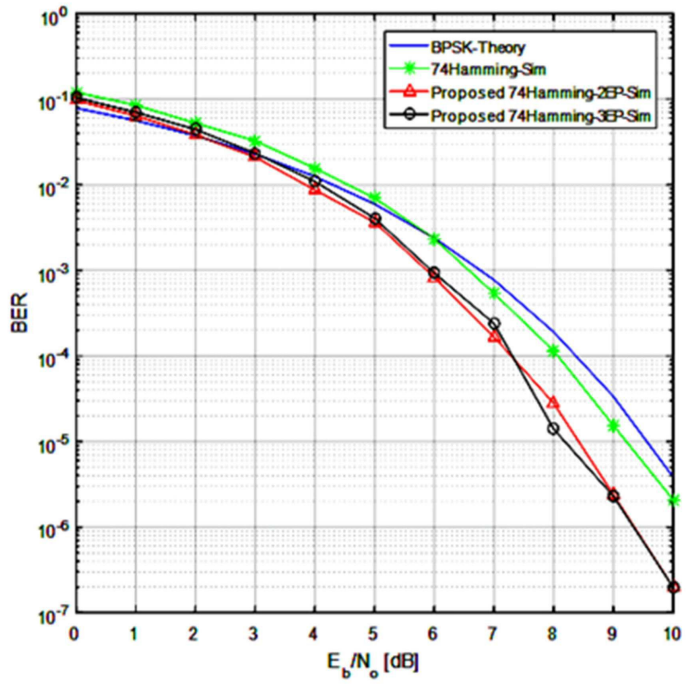
도면6


BPSK-Theory
74Hamming-Sim
Proposed 74Hamming-Sim
BER
Eb/No [dB]
Uncoded BPSK-Theory
1511Hamming-Sim
Proposed 1511Hamming-Sim
BER
Eb/No [dB]

도면7


BPSK-Theory
7(6,3)LBC-Sim
FProposed (6,3)LBC-Sim
BER
Eb/No [dB]

도면8


BPSK-Theory
74Hamming-Sim
Proposed 74Hamming-2EP-Sim
Proposed 74Hamming-3EP-Sim
BER
E_b/N_o [dB]