(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2015년12월07일
(11) 등록번호 10-1575123
(24) 등록일자 2015년12월01일

(51) 국제특허분류(Int. Cl.)
H04L 9/18 (2006.01)
(21) 출원번호 10-2014-0113158
(22) 출원일자 2014년08월28일
심사청구일자 2014년08월28일
(56) 선행기술조사문헌
US7254800 B1
KR1019880012030 A

(73) 특허권자
서강대학교산학협력단
서울특별시 마포구 백범로 35 (신수동, 서강대학교)
(72) 발명자
손원민
서울특별시 용산구 이촌로 104 (이촌동, 강변아파트) 가동 701호
조용기
서울특별시 강서구 양천로30길 123-9 101동 1409호 (마곡동, 마곡신안아파트)
(74) 대리인
김일환

전체 청구항 수 : 총 15 항 심사관 : 문형섭

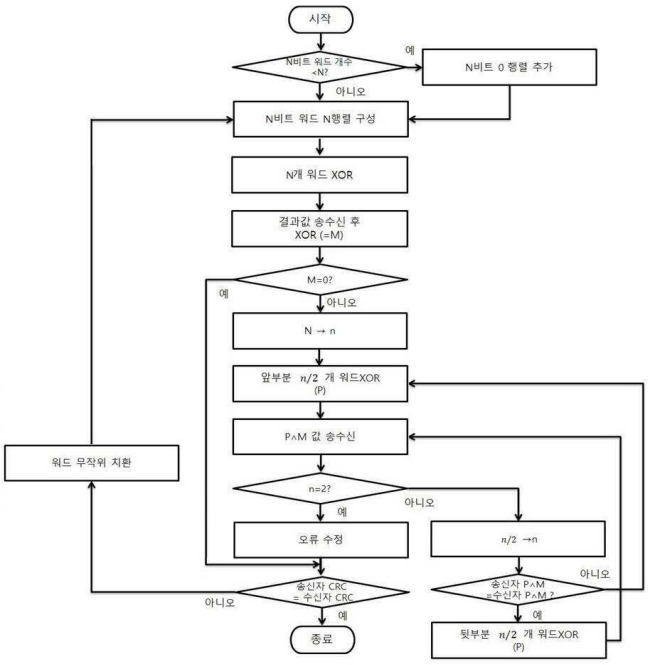
(54) 발명의 명칭 **워드 마스크를 이용한 고속 병렬 비트오류 보정방법**

(57) 요 약

본 발명은 워드 마스크를 이용한 고속 병렬 비트오류 보정방법에 관한 것으로, 통신에서 비트오류 보정방법에 있어서, (a) 송신자에서 생성된 N개의 비트를 갖는 데이터에서 워드 단위로 블록화하고 병렬화하여 n개의 행렬을 형성하는 단계; (b) 상기 n개의 행렬을 서로 배타적 논리합(XOR) 연산하여 패리티 워드를 생성하는 단계; (c) 상

(뒷면에 계속)

대 표 도 - 도1



기 생성된 패리티 워드와 수신자에서 산출된 패리티 워드를 배타적 논리합(XOR) 연산하여 마스크 워드를 생성한 후, 상기 송신자 및 수신자에서 동일한 상기 마스크 워드를 공유하는 단계; (d) 상기 마스크 워드를 상기 패리티 워드와 AND 연산하여 전송 패리티 워드를 생성하고, 상기 전송 패리티 워드를 이용하여 비트오류를 수정하는 단계를 포함한다.

이와 같은 본 발명은, 매우 긴 메시지를 연산 장치에서 빠르게 처리할 수 있는 워드를 이용하고, 각 워드를 병렬화한 패리티 계산으로 매우 빠르게 시행하는 방법으로 동시에 여러 개의 메시지에 대한 오류 검출 및 수정이 가능한 비트오류 수정방법을 제공한다.

이 발명을 지원한 국가연구개발사업

과제고유번호	2014-044-014-002
부처명	미래창조과학부
연구관리전문기관	정보통신기술진흥센터
연구사업명	방송통신인프라원천기술개발사업
연구과제명	양자암호 네트워킹 핵심기술 개발
기 여 율	1/1
주관기관	한국과학기술연구원
연구기간	2014.04.01 ~ 2015.02.28

명 세 서

청구범위

청구항 1

통신에서 비트오류 보정방법에 있어서,

(a) 송신자에서 생성된 N개의 비트를 갖는 데이터에서 워드 단위로 블록화하고 병렬화하여 n개의 행렬을 형성하는 단계;

(b) 상기 n개의 행렬을 서로 배타적 논리합(XOR) 연산하여 패리티 워드를 생성하는 단계;

(c) 상기 생성된 패리티 워드와 수신자에서 산출된 패리티 워드를 배타적 논리합(XOR) 연산하여 마스크 워드를 생성한 후, 상기 송신자 및 수신자에서 동일한 상기 마스크 워드를 공유하는 단계;

(d) 상기 마스크 워드를 상기 패리티 워드와 AND 연산하여 전송 패리티 워드를 생성하고, 상기 전송 패리티 워드를 이용하여 비트오류를 수정하는 단계를 포함하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 2

제1항에 있어서,

상기 (c) 단계는,

(c1) 상기 n개의 패리티 워드에서 절반의 개수로 A 행렬 및 B 행렬로 분리한 후, 어느 하나의 행렬을 상기 XOR 연산을 통해 상기 마스크 워드를 생성하는 단계;

(c2) 마스크 패리티 행렬을 통해 상기 A 행렬 및 B 행렬 중 오류가 나타나는 행렬을 선택하고, 선택된 행렬을 다시 절반의 개수로 분리한 후, XOR 연산을 통해 마스크 워드를 생성하는 단계;

(c3) 상기 (c1) 및 (c2) 과정을 반복하여 패리티 워드 상에서 오류의 위치를 특정하는 단계를 포함하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 3

제1항에 있어서,

상기 n개는 상기 N개 보다 크거나 같은 짝수인 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 4

제1항에 있어서,

상기 n개가 상기 N개 보다 크거나 같고 짝수가 아닌 경우,

패딩(padding) 방법을 이용하여, 상기 송신자 및 수신자에서 공유하는 임의의 더미 워드를 추가하여 상기 n개를 짝수로 변환하는 단계를 포함하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 5

제2항에 있어서,

상기 n개가 상기 N개 보다 크거나 같고 짝수가 아닌 경우,

상기 (c1) 단계에서, 상기 n개의 패리티 워드에서 절반의 개수는 n/2 개의 반올림된 수인 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 6

제1항에 있어서,

상기 (c) 단계에서, 상기 동일한 마스크 워드의 공유는,

상기 송신자 또는 수신자 어느 한쪽에서 상기 마스크 워드를 계산한 후, 상대방에게 전송하여 공유하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 7

제1항에 있어서,

상기 (c) 단계에서, 상기 동일한 마스크 워드의 공유는,

상기 송신자 및 수신자가 상기 각 패리티 워드 정보를 상호교환 한 후, 각각 상기 마스크 워드를 생성하여 공유하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 8

제1항에 있어서,

상기 마스크 워드에 1인 비트가 두 개 이상인 경우, 상기 마스크 워드에 존재하는 1의 개수만큼 서로 다른 서브 마스크 워드를 생성하는 단계를 더 포함하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 9

제1항에 있어서,

상기 (d) 단계에서,

상기 비트오류의 수정은,

NOT 연산을 통해 수정하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 10

제1항에 있어서,

상기 (d) 단계는,

오류 비트의 위치를 단일 워드까지 찾고, 해당 워드 비트내의 오류 비트를 수정하여 송신자와 수신자의 비트열을 동기화하는 단계를 포함하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 11

제1항에 있어서,

상기 (d) 단계는,

오류 비트의 위치를 2개의 워드까지 찾고, 상기 2개의 워드의 위치에 상기송신자와 수신자가 모두 00 혹은 11로 미리 공유된 비트로 변경하여 송신자와 수신자의 비트열을 동기화하는 단계를 포함하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 12

제1항에 있어서,

상기 (d) 단계 이후,

상기 워드 단위의 블록을 무작위로 재배치하여 상기 (a) 단계 이하를 반복하는 단계를 더 포함하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 13

제12항에 있어서,

상기 워드 단위의 블록을 무작위 재배치하는 방법은, 오류 검출에 사용하지 않은 다수의 워드를 상기 오류 검출에 사용한 다수의 워드와 치환하여 재배치하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 14

제12항에 있어서,

적어도 하나의 상기 워드 단위의 블록내에서 워드의 순서를 무작위로 재배치하는 순열을 적용하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

청구항 15

제1항에 있어서,

송신자와 수신자가 오류를 검출하고자 하는 워드가 같은지 확인하고, 색인 순으로 정렬하여 오류가 수정된 최초 메시지를 복원할 수 있도록 상기 각 워드에 색인 부여하는 단계를 더 포함하는 것을 특징으로 하는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 통신상에서 비트오류 보정방법에 관한 것으로, 보다 상세하게는 송신자가 수신자에게 비트열(bit string)를 전달할 때, 통신상의 문제로 수신자의 비트열에 오류가 생겨 송신자와 수신자의 비트가 다를 수 있는데, 전송한 비트 여러 개의 오류를 동시에 검사하고 수정함으로써 매우 빠른 오류 수정을 할 수 있고, 암호관련 통신에서는 통신에서 유출되는 정보를 최대한 적게 하는 것이 필요하여 마스크(Mask)를 이용하여 불필요한 정보의 유출 없이 오류를 수정할 수 있는 워드 마스크를 이용한 고속 병렬 비트오류 보정방법에 관한 것이다.

배 경 기 술

[0002] 장거리 통신에서 송신자가 수신자에게 비트로 이루어진 정보를 보낼 때, 통신 환경에 존재하는 잡음이나 외부 요인으로 인해서 송신자가 보낸 정보에 변조가 일어날 수 있다. 송신자가 수신자에게 자신이 전하고 싶은 비트

열을 정확하게 보내기 위해서는 송신자와 수신자 각자가 가지고 있는 비트열에 대한 정보를 공유하면서 송신자와 수신자가 가진 비트열이 같도록 해야 한다.

[0003] 일반적인 공개 통신에서는 송신자와 수신자가 각각 자신이 가진 비트열을 공개하여 서로 다른 부분을 수정하는 방법, 오류정정부호를 이용하는 방법 등 여러 가지 방법으로 오류의 수정이 가능하다. 하지만 비공개 정보를 전송하는 통신에서는 이와 같은 방식을 직접 사용할 수 없는 상황이 존재한다.

[0004] 예를 들어, 양자키분배(Quantum Key Distribution, 이하 QKD)와 같이 송신자가 전송하는 키정보를 제 3자가 알지 못하게 수신자에게 보내는 통신에서는 상호간에 정확한 키정보 공유가 필요함에도 불구하고, 상호간 주고받는 통신 비트열에 발생하는 변조와 같은 오류를 수정하는 과정에서 비트열을 공개해서는 안되는 경우가 생긴다. 이 경우 제 3자가 송신자와 수신자가 공유하는 비트열에 대한 정보 누수를 최소로 하면서 수신자의 비트 오류를 수정하는 것이 매우 중요해지며 반드시 해결되어야 할 필요가 있다.

[0005] 일반적으로 사용되는 패리티(Parity) 기반 오류 수정 방법은 송신자와 수신자가 각자의 비트열을 몇 개의 블록으로 나누어, 블록들의 패리티를 비교하는 방법으로 오류를 검출한다. 오류가 있는 블록을 찾아내면 해당 블록을 더 작은 블록으로 나누어 패리티를 비교하는 것으로 오류의 위치를 찾아내 오류를 수정할 수 있다.

[0006] 이 방법은 블록 내의 비트 하나하나에 대한 배타적 논리합(Exclusive or, 이하 XOR) 연산을 수행하여 패리티를 구하기 때문에 각각의 비트를 워드(word) 안에서 꺼내어 계산해야 한다. 즉 워드 안의 비트를 새로운 메모리 영역에 복사하여 계산한다. 이런 2중 처리 때문에 패리티 계산의 속도가 느려지게 된다.

[0007] 예를 들어, QKD에서의 후처리 과정인 reconciliation 과정에 필수적으로 사용되는 비트단위의 패리티 확인과정은 전체 작동시간의 상당부분을 차지할 만큼 많은 시간을 소모하게 하는 원인이 될 수 있어, 이에 대한 새로운 기술 개발이 반드시 해결되어야 하는 상황이다.

선행기술문헌

특허문헌

[0008] (특허문헌 0001) 대한민국 공개번호 제10-2014-0051736호(공개일자:2014년05월02일)

(특허문헌 0002) 대한민국 공개번호 제10-2009-0004667호(공개일자:2009년01월12일)

발명의 내용

해결하려는 과제

[0009] 상술한 문제를 해결하고자 하는 본 발명의 과제는 워드 단위 연산 수행을 통한 연산속도를 고속화하고, 적은 연산으로 더 빠른 오류 보정을 위해서 여러 개의 워드의 오류를 동시에 보정할 수 있는 비트오류 수정방법을 제공하고자 함이다.

[0010] 또한, 비트오류 수정과정 동안 유출되는 정보를 최소화할 수 있는 비트오류 수정방법을 제공하고자 함이다.

과제의 해결 수단

[0011] 상술한 과제를 해결하기 위한 본 발명의 특징은, 통신에서 비트오류 보정방법에 있어서, (a) 송신자에서 생성된 N개의 비트를 갖는 데이터에서 워드 단위로 블록화하고 병렬화하여 n개의 행렬을 형성하는 단계; (b) 상기 n개의 행렬을 서로 배타적 논리합(XOR) 연산하여 패리티 워드를 생성하는 단계; (c) 상기 생성된 패리티 워드와 수신자에서 산출된 패리티 워드를 배타적 논리합(XOR) 연산하여 마스크 워드를 생성한 후, 상기 송신자 및 수신자에서 동일한 상기 마스크 워드를 공유하는 단계; (d) 상기 마스크 워드를 상기 패리티 워드와 AND 연산하여 전송 패리티 워드를 생성하고, 상기 전송 패리티 워드를 이용하여 비트오류를 수정하는 단계를 포함한다.

[0012] 여기서, 상기 (c) 단계는, (c1) 상기 n개의 패리티 워드에서 절반의 개수로 A 행렬 및 B 행렬로 분리한 후, 어느 하나의 행렬을 상기 XOR 연산을 통해 상기 마스크 워드를 생성하는 단계; (c2) 마스크 패리티 행렬을 통해 상기 A 행렬 및 B 행렬 중 오류가 나타나는 행렬을 선택하고, 선택된 행렬을 다시 절반의 개수로 분리한 후, XOR 연산을 통해 마스크 워드를 생성하는 단계; (c3) 상기 (c1) 및 (c2) 과정을 반복하여 패리티 워드 상에서 오류의 위치를 특정하는 단계를 포함하는 것이 바람직하다.

[0013] 또한, 상기 n개는 상기 N개 보다 크거나 같은 짝수인 것이 바람직하고, 상기 n개가 상기 N개 보다 크거나 같고 짝수가 아닌 경우, 패딩(padding) 방법을 이용하여, 상기 송신자 및 수신자에서 공유하는 임의의 더미 워드를 추가하여 상기 n개를 짝수로 변환하는 단계를 포함하는 것이 바람직하다.

[0014] 또한, 바람직하게는 상기 n개가 상기 N개 보다 크거나 같고, 짝수가 아닌 경우, 상기 (c1) 단계에서, 상기 n개의 패리티 워드에서 절반의 개수는 n/2 개의 반올림된 수인 것일 수 있고, 상기 (c) 단계에서, 상기 동일한 마스크 워드의 공유는, 상기 송신자 또는 수신자 어느 한쪽에서 상기 마스크 워드를 계산한 후, 상대방에게 전송하여 공유하는 것일 수 있다.

[0015] 그리고, 상기 (c) 단계에서, 상기 동일한 마스크 워드의 공유는, 상기 송신자 및 수신자가 상기 각 패리티 워드 정보를 상호교환 한 후, 각각 상기 마스크 워드를 생성하여 공유하는 것이 바람직하고, 상기 마스크 워드에 1인 비트가 두 개 이상인 경우, 상기 마스크 워드에 존재하는 1의 개수만큼 서로 다른 서브 마스크 워드를 생성하는 단계를 더 포함하는 것이 바람직하다.

[0016] 한편, 상기 (d) 단계에서, 상기 비트오류의 수정은, NOT 연산을 통해 수정하는 것이 바람직하고, 상기 (d) 단계는, 오류 비트의 위치를 단일 워드까지 찾고, 해당 워드 비트 내의 오류 비트를 수정하여 송신자와 수신자의 비트열을 동기화하는 단계를 포함하는 것이 바람직하다.

[0017] 또한, 상기 (d) 단계는, 오류 비트의 위치를 2개의 워드까지 찾고, 상기 2개의 워드의 위치에 상기송신자와 수신자가 모두 00 혹은 11로 미리 공유된 비트로 변경하여 송신자와 수신자의 비트열을 동기화하는 단계를 포함하는 것이 바람직하고, 상기 (d) 단계 이후, 상기 워드 단위의 블록을 무작위로 재배치하여 상기 (a) 단계 이하를 반복하는 단계를 더 포함하는 것이 바람직하다.

[0018] 더하여, 바람직하게는 상기 워드 단위의 블록을 무작위 재배치하는 방법은, 오류 검출에 사용하지 않은 다수의 워드를 상기 오류 검출에 사용한 다수의 워드와 치환하여 재배치하는 것일 수 있고, 적어도 하나의 상기 워드 단위의 블록내에서 워드의 순서를 무작위로 재배치하는 순열을 적용하는 것일 수 있다.

[0019] 그리고, 송신자와 수신자가 오류를 검출하고자 하는 워드가 같은지 확인하고, 색인 순으로 정렬하여 오류가 수정된 최초 메시지를 복원할 수 있도록 상기 각 워드에 색인 부여하는 단계를 더 포함하는 것이 바람직하다.

발명의 효과

[0020] 이와 같은 본 발명은, 매우 긴 메시지를 연산 장치에서 빠르게 처리할 수 있는 워드를 이용하고, 각 워드를 병렬화한 패리티 계산으로 매우 빠르게 시행하는 방법으로 동시에 여러 개의 메시지에 대한 오류 검출 및 수정이 가능한 비트오류 수정방법을 제공한다.

[0021] 또한, 병렬로 여러 데이터를 처리하지만 통신시 유출되는 정보는 마스크 워드를 이용하여 최소화되므로 비밀 통신의 오류 검사 및 수정에도 매우 유용하다.

[0022] 또한 워드를 병렬로 처리하므로 병렬 워드 내에 존재하는 짝수 개의 워드의 같은 위치에 오류가 발생한 경우 이외에는 항상 오류를 찾을 수 있고, 워드의 길이가 길어짐에 따라 같은 위치에 오류가 발생할 확률은 더 적어지므로 기존의 비트열을 블록화하여 각각의 블록의 패리티를 비교하는 방법보다 오류를 찾을 확률이 더 높을 뿐만 아니라, 비트 단위의 무작위 치환이 아닌 워드 단위의 치환을 사용하므로 치환의 효율과 속도를 높일 수 있다.

도면의 간단한 설명

[0023] 도 1은 본 발명의 실시예에 따른 워드 마스크를 이용한 고속 병렬 오류 수정방법의 알고리즘 예시한 도면이고,

도 2는 n개 워드 병렬 패리티 행렬 계산을 나타낸 모식도이고,

도 3은 n개 워드에서 통신을 통한 마스크 행렬 계산을 나타낸 모식도이고,

도 4는 마스크를 이용하여 불필요한 패리티 유출 방지를 위한 특정 예시(8비트)한 도면이고,

도 5는 오류가 2개일 때 서브 마스크 생성 방법 예시(8비트)를 나타낸 도면이다.

발명을 실시하기 위한 구체적인 내용

[0024] 본 발명의 이점 및 특징, 그리고 그것을 달성하는 방법은 첨부되는 도면과 함께 상세하게 후술되어 있는 실시예들을 통해 설명될 것이다. 그러나 본 발명은 여기에서 설명되는 실시예들에 한정되지 않고 다른 형태로 구체화

될 수도 있다. 단지, 본 실시예들은 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 본 발명의 기술적 사상을 용이하게 실시할 수 있을 정도로 상세히 설명하기 위하여 제공되는 것이다.

[0025] 도면들에 있어서, 본 발명의 실시예들은 도시된 특정 형태로 제한되는 것이 아니며 명확성을 기하기 위하여 과장된 것이다. 또한 명세서 전체에 걸쳐서 동일한 참조번호로 표시된 부분들은 동일한 구성요소를 나타낸다.

[0026] 본 명세서에서 "및/또는"이란 표현은 전후에 나열된 구성요소들 중 적어도 하나를 포함하는 의미로 사용된다. 또한, 단수형은 문구에서 특별히 언급하지 않는 한 복수형도 포함한다. 또한, 명세서에서 사용되는 "포함한다" 또는 "포함하는"으로 언급된 구성요소, 단계, 동작 및 소자는 하나 이상의 다른 구성요소, 단계, 동작, 소자 및 장치의 존재 또는 추가를 의미한다.

[0027] 이하 본 발명의 바람직한 실시예를 도면을 참조하여 상세히 설명하기로 한다.

[0028] 도 1은 본 발명의 실시예에 따른 워드 마스크를 이용한 고속 병렬 비트오류 보정방법을 나타내는 흐름도이다. 도 1에 나타낸 바와 같이, 본 발명의 실시예에 따른 워드 마스크를 이용한 고속 병렬 비트오류 보정방법은, 통신에서 비트오류 보정방법에 있어서, (a) 송신자에서 생성된 N개의 비트를 갖는 데이터에서 워드 단위로 블록화하고 병렬화하여 n개의 행렬을 형성하는 단계; (b) 상기 n개의 행렬을 서로 배타적 논리합(XOR) 연산하여 패리티 워드를 생성하는 단계; (c) 상기 생성된 패리티 워드와 수신자에서 산출된 패리티 워드를 배타적 논리합(XOR) 연산하여 마스크 워드를 생성한 후, 상기 송신자 및 수신자에서 동일한 상기 마스크 워드를 공유하는 단계; (d) 상기 마스크 워드를 상기 패리티 워드와 AND 연산하여 전송 패리티 워드를 생성하고, 상기 전송 패리티 워드를 이용하여 비트오류를 수정하는 단계를 포함하여 구성된다.

[0029] 보다 구체적으로 살펴보면, 먼저 (a) 단계에서, 통신 오류 수정에 비트를 연산 장치의 처리 단위인 워드로 치환한다. 이때 워드 하나 당 비트 수는 연산 장치에 따르며 연산 장치의 처리 단위에 따라 정해진다. 비트열을 치환한 워드를 몇 개의 묶음 또는 블록으로 하여 워드의 행렬로 구성한다. 이를 워드의 병렬화라 한다.

[0030] 이때 병렬화된 워드의 개수는 워드의 비트 개수보다 크거나 같은 임의의 2의 거듭 제곱 꼴인 짝수가 해당 발명의 방법에서 적당하다. 하지만 짝수가 아닌 경우에도 몇 가지 방법을 이용하여 워드를 병렬화할 수 있다.

[0031] 첫 번째, 패딩(padding) 방법을 이용하여, 부족한 워드의 수만큼 상호 생성 방법을 공유한 임의의 더미 워드를 추가하여 워드의 개수를 짝수로 만들 수 있다.

[0032] 두 번째, 패리티 워드를 계산하는 워드의 개수를 조절할 수 있다. 일반적으로는 전 과정에서 사용한 워드 개수인 n의 절반 개수인 n/2개의 워드를 이용한 연산을 통해 오류 위치를 특정하지만 절반이 아닌 개수로 구성하여 계산하는 경우 워드의 개수가 짝수일 필요는 없다.

[0033] (b) 단계로서, 패리티 워드를 생성하는 방법은 다음과 같다. 병렬화된 워드에 대해 각 행렬의 워드와 워드 간의 배타적 논리합(XOR) 연산을 통해 각 워드 행렬의 x번째 원소들끼리의 XOR 연산을 x번째 원소로 갖는 패리티 워드를 얻을 수 있다.

[0034] (c) 단계에서는, 상기 병렬 처리한 패리티 워드를 상호 전달하여 두 패리티 워드를 XOR 연산하여 마스크 워드를 얻는다, 마스크 워드가 0이 아닌 경우 오류가 존재하므로 오류 수정 과정에 들어간다.

[0035] 위 과정에서 비교한 패리티 계산에 사용한 워드의 개수를 n이라 할 때 n을 반으로 나누어 앞 부분 n/2개의 워드의 패리티를 비교하는 것으로 오류가 앞 부분 n/2개의 워드에 포함되어 있는지, 뒷부분 n/2개의 워드에 포함되어 있는지 확인할 수 있다. 이 과정을 반복하여 오류의 위치를 특정한다.

[0036] 그리고, 패리티 정보를 교환할 때 송신자와 수신자의 비트열에 대한 불필요한 정보가 유출될 수 있으므로, 송신자와 수신자의 패리티 워드를 XOR 연산한 마스크 워드를 송신자와 수신자가 나누어 갖고, 마스크 워드를 패리티 워드에 AND 연산하여 오류 수정에 필요한 정보 외의 비트들을 제거한 후 전송한다.

[0037] 이때 마스크 워드에서 오류가 2개 이상이라고 판명되는 경우 서브 마스크 워드를 구성하여 오류를 수정할 수 있다. 서브 마스크 워드를 구성하는 방법은 다음과 같다.

[0038] 첫 번째 방법은, 마스크 워드에 1인 비트가 두 개 이상인 경우 다음과 같이 진행한다. 마스크 워드에 존재하는 1인 비트의 개수만큼 서브 마스크 워드를 생성한다. 서브 마스크 워드는 모두 다르며 마스크 워드에 1인 비트가 존재하는 위치 중 하나에 1인 비트를 갖고 다른 비트는 모두 0이다.

[0039] 두 번째 방법은, 패리티 워드 비교 과정에 있어서 송신자와 수신자의 패리티 워드와 마스크 워드의 XOR 연산 결과인 전송 패리티 워드를 XOR 연산한 결과의 워드와 해당 과정에서 사용한 마스크 워드와의 XOR 연산 값을 통해 서브 마스크 워드를 얻을 수 있다.

[0040] 이때 송신자와 수신자가 마스크 워드를 나누어 갖는 방법은 다음과 같다. 첫 번째 방법은 송신자 혹은 수신자가 자신의 패리티 워드를 상대방에서 보내어 마스크 워드를 계산한 후 상대방에게 전송하는 것이다. 두 번째 방법은 송신자와 수신자가 패리티 워드를 상호 교환한 후 각자 마스크 워드를 계산하는 방법이다.

[0041] 그리고, 발견한 오류를 수정하는 방법은 다음과 같다.

[0042] 첫 번째 방법은 패리티 워드 통신을 오류의 위치를 정확히 찾을 때까지 반복하여 검출된 오류를 NOT 연산을 통해 수정하는 것이다.

[0043] 두 번째 방법은 패리티 워드 통신을 통해 오류의 위치를 워드 두 개까지 좁힌 후 오류 비트일 가능성이 있는 비트 두 개를 송신자와 수신자가 모두 00, 11과 같이 사전에 미리 공유된 비트로 변경하여 송신자와 수신자의 비트열을 동기화하는 방법이다.

[0044] 또한, 더욱 정밀한 오류 검출을 위한 오류 분산 방법에는 다음과 같은 방법이 있다.

[0045] 첫째로, 오류 수정 과정 후 해당 오류 검출에서 사용하지 않은 워드와의 사용한 워드 몇 개를 치환하는 워드 단위 치환으로 효율적으로 오류를 분산시킨다. 둘째로, 오류 수정 과정 후 워드 블록 내에서 워드의 순서를 무작위로 재배치 또는 섞는 순열을 적용하여 워드 블록 내에서 수정되지 않은 오류가 효율적으로 전파되도록 한다. 셋째로, 워드 블록 여러 개에 대하여 워드의 순서를 무작위로 섞는 순열을 적용하여, 워드 블록 내에서 수정되지 않은 오류가 효율적으로 전파되도록 한다.

[0046] 더하여, 송신자와 수신자가 같은 워드간의 패리티 비교를 하고 있는지 확인하고, 오류 수정이 끝난 후 필요한 경우 최초의 워드 순서로 정렬하여 송신자가 보내고자 하는 메시지를 복원하기 위해서 워드에 색인을 부여한다.

[0047] 이하에서, 도 1 내지 도 4에서 예시된 실시예를 통해 보다 구체적으로 설명하기로 한다.

[0048] 본 발명에서는 먼저 일반적인 상황에서의 알고리즘에 대한 설명을 간단히 한 후 특정 예시를 통해 발명에 대한 이해를 돕는 방법을 사용한다. 일반적인 상황에서는 오류가 하나 존재하는 경우에 대한 프로토콜을 설명하며, 8비트 워드를 사용하는 특정 예시에서 오류가 두 개 이상 존재할 경우에 대한 프로토콜을 설명하도록 하겠다.

[0049] 먼저 통신을 통해 얻은 정보 비트열을 연산 장치가 쉽게 처리할 수 있는 크기의 구간으로 나누어 각 구간을 워드로 한다. 일반적으로 8비트 워드, 16비트 워드, 32비트 워드, 64비트 워드 등이 사용된다. 각각의 워드는 비트크기와 동일하거나 그 배수에 해당하는 블록으로 묶어 동시 연산 수행 준비를 한다. 편의상 본 발명의 실시예에서는 동일한 크기의 블록을 고려하도록 한다. 각 워드 블록에는 색인을 붙여 송신자와 수신자가 동일한 워드 블록에 대한 연산을 하고 있다는 사실을 확인할 수 있도록 한다.

[0050] 이 워드의 비트 크기를 n이라 하자. 계속되는 통신으로 복수의 워드가 생성되면 n개의 워드를 블록으로 묶어 행렬화 한다. 이 행렬은 워드를 단위로 보면 1차원 행렬이고, 비트를 단위로 보는 경우 $n \times n$의 2차원 행렬로 볼 수 있다. 생성된 워드의 개수가 n개보다 부족한 경우, 오류 수정 방법에 따라 n개보다 적은 워드의 행렬로 진행하거나 부족한 워드의 수만큼 모든 원소가 0 또는 1로 이루어진 더미 워드를 추가하여 길이가 n인 행렬을 만들어 사용할 수 있다. 이 더미 워드는 송신자와 수신자가 사전에 정의한 임의의 값을 가지기만 하면 된다.

[0051] 이후 행렬 안에 있는 각 워드들에 대해 XOR 연산을 시행한다. 도 2에 나타낸 바와 같이, 워드와 워드의 XOR 값은 워드이다. 여기서 얻는 워드는 각 워드의 x번째 자리의 비트들끼리 XOR 연산한 값을 x번째 자리의 비트로 갖는 패리티 워드라고 부를 수 있다.

[0052] 그리고 나서, 도 3에 나타낸 바와 같이, 송신자는 자신이 얻은 패리티 워드를 수신자에게 전달한다. 수신자는 자신이 계산한 패리티 워드와 송신자가 보낸 패리티 워드에 대해 XOR 연산을 행한다. 송신자와 수신자의 패리티 워드가 같다면 XOR 연산 후 얻을 워드는 워드 비트가 모두 0인 워드 즉, 0이다.

[0053] 만약 송신자와 수신자의 패리티 워드가 다르다면 XOR 연산 후 얻을 워드는 비트 행렬로 표현했을 때 송신자와 수신자의 패리티 워드가 다른 부분만 1의 원소를 갖는다. 패리티 워드가 1인 원소를 가지는 것은 송신자와 수신자가 서로 다른 패리티를 가지는 것이고, 통신 과정에 오류가 발생한 것을 의미한다.

[0054] 송신자와 수신자의 패리티 워드 사이에 XOR 연산을 통해 얻은 워드를 오류 수정 과정에서 정보를 보호하기 위한 마스크 워드로 사용하게 된다. 수신자는 송신자에게 자신이 계산한 패리티 워드를 전달하거나 마스크를 직접 전달하여 송신자와 수신자가 모두 마스크 워드를 가질 수 있게 한다.

[0055] XOR 연산후 얻은 값이 0이라면 순환 중복 검사(Cyclic Redundancy Check, 이하 CRC) 등의 방법을 이용하여 송신자와 수신자의 정보가 같은지 확인한다. 다른 경우 패리티 확인을 한 블록의 동일 위치의 비트들 안에 짝수개의 오류가 있었음을 의미하므로 추가 오류 수정이 필요함을 알 수 있다.

[0056] 추가 오류 수정을 위해 통신으로 얻은 정보 비트열 중 이번 연산에 사용하지 않은 워드와 이번 연산에 사용한 워드를 몇 개 치환한 후, 또는 이번 연산에 사용한 워드의 순서를 랜덤하게 변경한 후, 또다시 워드 단위 XOR을 통해 패리티 확인 작업을 반복하면 된다. XOR 연산 후 얻은 값이 0이 아니라면 오류 수정 과정에 들어가게 된다. 어느 위치에 오류가 있는지 확인하기 위해 n개의 워드 중 앞부분 n/2개의 워드만을 XOR 연산한다.

[0057] 이후 송신자는 이번 과정에서 얻은 패리티 워드와 마스크 워드 사이에 AND 연산을 시행한다. AND 연산을 통해 오류 수정이 필요하지 않은 패리티 정보는 0으로 치환되고 오류 수정에 필요한 패리티 정보만이 남는다. 패리티 워드와 마스크 워드를 AND한 값을 수신자에게 보내면, 수신자도 같은 연산을 행한 뒤 두 워드를 XOR 연산하여 자신의 워드와 송신자의 워드가 같은지 확인한다.

[0058] 만약 같은 경우, 이전 시행에서 발견한 오류는 뒤 n/2개 워드에 포함되어 있다. 다른 경우, 오류는 이번 시행에서 연산한 n/2개 워드에 포함되어 있다. 단계적(Cascade) 오류 정정 방법과 같이 계속해서 XOR 연산 워드의 값을 반으로 줄이면서 오류의 위치를 정확히 찾을 수 있다.

[0059] 그리고, 오류의 수정 방법은 몇 가지를 사용할 수 있다. 정확한 메시지를 전달해야 하는 경우, 수신자 측에서 오류의 위치를 정확히 찾아낸 후 해당 오류를 NOT 연산을 통해 반전시키면 된다. 통신량을 줄이면서 정보의 유출을 적게 하여 임의의 비트열을 전달하고자 하는 경우 특정 오류의 위치를 찾지 않고 오류일 가능성이 있는 워드를 두 개까지 좁힌 후 두 워드의 잠재적 오류 원소들을 00 혹은 11 등 사전에 합의한 대로 변환시키는 방법을 사용할 수 있다.

[0060] 오류 수정 후 CRC 체크 등을 통해 송신자와 수신자의 정보가 같은지 비교한 후, 같은 경우 오류 보정을 끝내고, 다른 경우 오류가 없을 때와 마찬가지로 워드 단위 치환을 시행한 후 계속해서 오류 보정 과정을 거친다.

[0061] 일 실시예로 8비트 워드를 통해 오류 보정 또는 수정 과정을 살펴보면 다음과 같다. 상기 실시예에서는 8비트 워드 8개를 이용하여 오류 보정 과정을 수행하며, 8비트 워드 8개에 오류 2개가 존재하는 경우에 대해 설명하겠다. 이 오류 2개가 두 워드의 같은 비트 위치에 존재하는 경우 패리티를 통한 오류의 확인이 불가능하나 치환 과정을 거쳐 확인하면 드러나게 되므로, 그렇지 않은 경우를 가정하겠다.

[0062] 송신자와 수신자가 각각 8비트 워드 8개에 대해 XOR 연산을 행한 후 송신자는 수신자에게 연산을 통해 얻은 패리티 워드를 보낸다. 수신자는 자신의 패리티 워드와 송신자의 패리티 워드에 대해 XOR 연산을 행하여 마스크 워드를 얻는다. 이 경우 오류가 두 개이므로 마스크 워드를 비트 행렬로 표현했을 때 두 개의 비트 1을 원소로 갖고 나머지 원소는 0인 행렬이 된다.(도 5 참조)

[0063] 도 5에 나타낸 바와 같이, 하나의 오류는 3번째 워드에 존재하고, 다른 하나의 오류는 5번째 워드에 존재한다. 이 경우 위에서 설명한 방법으로는 한 번에 두 개의 오류를 보정 할 수 없기 때문에, 필요 없는 정보의 유출이 생긴다.

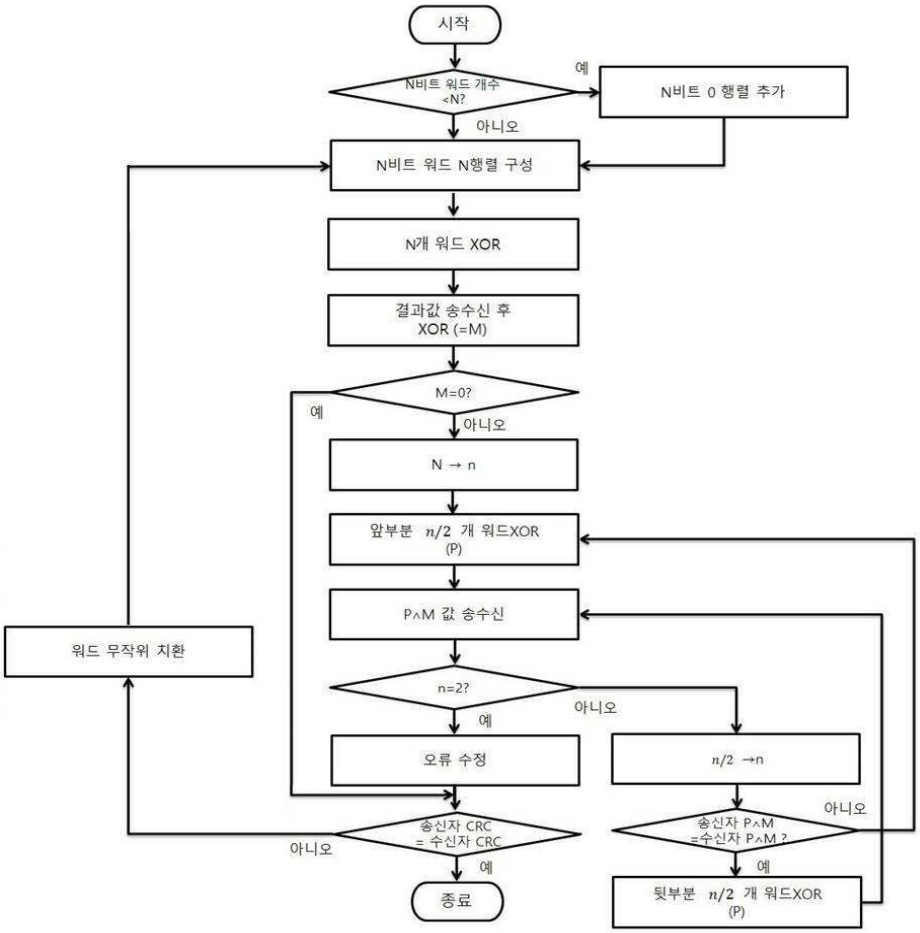
[0064] 따라서 마스크 행렬을 서브 마스크 행렬 두 개로 나누어 각 서브 마스크마다 위와 같이 XOR 연산을 수행하는 워드 수를 줄이면서 오류의 위치를 특정한다. 도 5에서 나타낸 서브 마스크 워드 1을 사용하는 경우 두 번째 패리티 워드 계산인 1행부터 4행까지 워드의 XOR 연산에서 송신자와 수신자가 서로 다른 값을 가지며 이후 1행과 2행의 XOR 연산 결과값이 같다는 것을 확인하면 오류의 위치가 3번째 혹은 4번째 워드라는 것을 확인할 수 있다.

[0065] 이후 송신자와 수신자가 3번째, 4번째 워드의 4번째 비트를 모두 0 혹은 1로 바꾸는 것으로 비트열의 오류가 수정된다. 서브 마스크 워드 2를 사용하는 경우 두 번째 패리티 워드가 같다는 것을 확인하여 다음 패리티 워드는 5번째와 6번째 워드의 XOR 연산으로 얻게 된다. 다음 패리티 워드가 다르므로 오류는 5번째 혹은 6번째 워드에 있으며 서브 마스크 워드를 통해 오류는 5번째 비트라는 것을 확인할 수 있으므로 마찬가지 방법으로 오류를 수정할 수 있다.

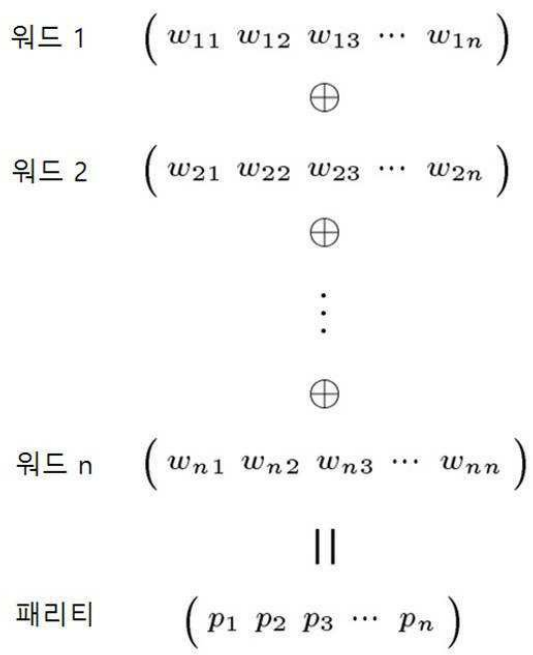
[0066] 이상의 설명에서 본 발명은 특정의 실시 예와 관련하여 도시 및 설명하였지만, 특허청구범위에 의해 나타난 발명의 사상 및 영역으로부터 벗어나지 않는 한도 내에서 다양한 개조 및 변화가 가능 하다는 것을 당 업계에서 통상의 지식을 가진 자라면 누구나 쉽게 알 수 있을 것이다.

도면

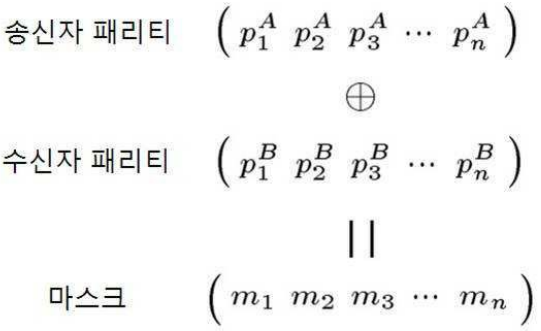
도면1



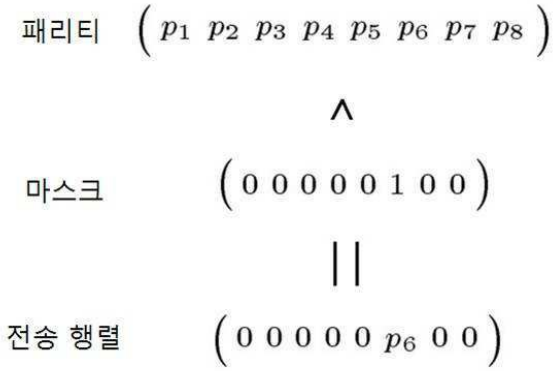
도면2


워드 1
$(w_{11}\ w_{12}\ w_{13}\ \cdots\ w_{1n})$
$\oplus$
워드 2
$(w_{21}\ w_{22}\ w_{23}\ \cdots\ w_{2n})$
$\oplus$
$\vdots$
$\oplus$
워드 n
$(w_{n1}\ w_{n2}\ w_{n3}\ \cdots\ w_{nn})$
||
패리티
$(p_1\ p_2\ p_3\ \cdots\ p_n)$

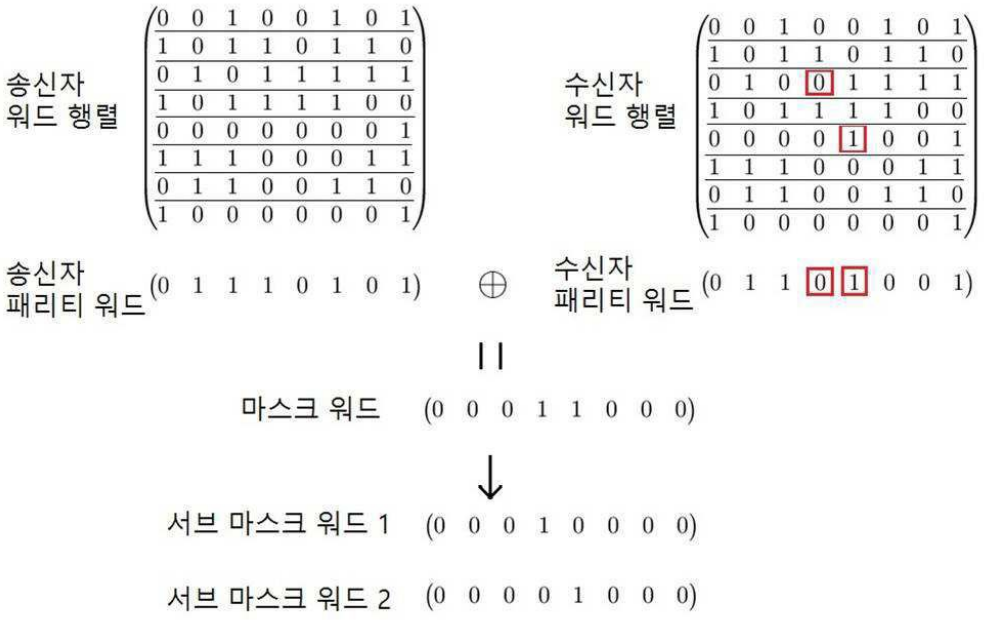
도면3


송신자 패리티
$(p_1^A\ p_2^A\ p_3^A\ \cdots\ p_n^A)$
$\oplus$
수신자 패리티
$(p_1^B\ p_2^B\ p_3^B\ \cdots\ p_n^B)$
||
마스크
$(m_1\ m_2\ m_3\ \cdots\ m_n)$

도면4


패리티
$(p_1\ p_2\ p_3\ p_4\ p_5\ p_6\ p_7\ p_8)$
∧
마스크
$(0\ 0\ 0\ 0\ 0\ 1\ 0\ 0)$
||
전송 행렬
$(0\ 0\ 0\ 0\ 0\ p_6\ 0\ 0)$

도면5


송신자
워드 행렬
수신자
워드 행렬
송신자
패리티 워드
(0 1 1 1 0 1 0 1)
수신자
패리티 워드
(0 1 1 0 1 0 0 1)
마스크 워드 (0 0 0 1 1 0 0 0)
서브 마스크 워드 1 (0 0 0 1 0 0 0 0)
서브 마스크 워드 2 (0 0 0 0 1 0 0 0)