(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2014년03월10일
(11) 등록번호 10-1370903
(24) 등록일자 2014년02월28일

(51) 국제특허분류(Int. Cl.)
H03M 13/11 (2006.01)
(21) 출원번호 10-2007-0080362
(22) 출원일자 2007년08월09일
심사청구일자 2012년07월09일
(65) 공개번호 10-2008-0084532
(43) 공개일자 2008년09월19일
(30) 우선권주장
60/895,416 2007년03월16일 미국(US)
(56) 선행기술조사문헌
US20040153938 A1*
US20050289437 A1*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
엘지전자 주식회사
서울특별시 영등포구 여의대로 128 (여의도동)
(72) 발명자
정지욱
경기도 안양시 동안구 흥안대로81번길 77, LG 제1연구단지 (호계동)
오민석
경기도 안양시 동안구 흥안대로81번길 77, LG 제1연구단지 (호계동)
(뒷면에 계속)
(74) 대리인
김용인, 박영복

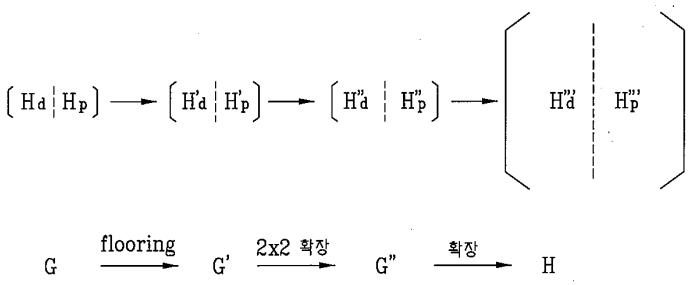
전체 청구항 수 : 총 9 항 심사관 : 성경아

(54) 발명의 명칭 **ＬＤＰＣ 코드를 이용한 부호화 및 복호화 방법**

(57) 요 약

본 발명은 저밀도 패리티 검사(LDPC: Low Density Parity Check) 코드를 이용하여 입력 데이터를 부호화하거나 부호화된 데이터를 복호화하는 방법에 관한 것이다. 본 발명의 일 양상에 따르면, 모델 행렬의 각 인덱스는 둘 이상의 인덱스들을 포함하는 인덱스 행렬로 확장된다. 상기 인덱스 행렬에 포함되는 각 인덱스는 특정 서브 행렬을 의미하고, 각 인덱스를 대응하는 서브 행렬로 대체함으로써 패리티 검사 행렬을 생성할 수 있다.

대 표 도 - 도3



(72) 발명자

조기형

경기도 안양시 동안구 흥안대로81번길 77, LG 제1 연구단지 (호계동)

강승현

경기도 안양시 동안구 흥안대로81번길 77, LG 제1 연구단지 (호계동)

석지애

경기도 안양시 동안구 흥안대로81번길 77, LG 제1 연구단지 (호계동)

이영섭

경기도 안양시 동안구 흥안대로81번길 77, LG 제1 연구단지 (호계동)

김소연

경기도 안양시 동안구 흥안대로81번길 77, LG 제1 연구단지 (호계동)

특허청구의 범위

청구항 1

LDPC 코드에 의한 부호화 또는 복호화를 수행하기 위해 패리티 검사 행렬을 생성하는 방법에 있어서,

각각의 인덱스가 $z_{max}\times z_{max}$ 차원의 영 행렬을 지시하는 '-1', $z_{max}\times z_{max}$ 차원의 단위 행렬을 지시하는 '0' 또는 상기 $z_{max}\times z_{max}$ 차원의 단위 행렬의 각 행 혹은 각 열을 x만큼 쉬프트하여 생성된 퍼뮤테이션 행렬을 지시하는 양의 정수 'x'인 다수의 인덱스들을 포함하여 이루어지는 제1 모델 행렬의 각 인덱스를 다음 식에 따라 z×z차원의 행렬을 지시하는 다른 인덱스 'shift(z)'로 대체하여 제2 모델 행렬을 생성:

$$shift(z)=floor(shift(z_{max})z/z_{max}),$$

여기서 "floor(i)"는 i로부터 음의 무한대 방향으로 가장 가까운 정수; 및

상기 제2 모델 행렬에 포함된 음이 아닌 정수 인덱스 'X' 각각을 'X'가 '2S+1'(여기서 S는 0 이상의 정수)인지 아니면 '2S'인지에 따라 2×2차원의 크로스 대각 행렬 $\begin{bmatrix} 0 & a_{1,2} \\ a_{2,1} & 0 \end{bmatrix}$ (여기서, $a_{1,2}+a_{2,1}=X$) 혹은 2×2차원의 대각 행렬 $\begin{bmatrix} a_{1,1} & 0 \\ 0 & a_{2,2} \end{bmatrix}$ (여기서, $a_{1,1}+a_{2,2}=X$)로 대체하고, 상기 제2 모델 행렬의 인덱스 '-1' 각각을 영 행렬들을 지시하는 인덱스들로 구성된 2×2차원의 행렬로 대체하여, 제3 모델 행렬을 생성; 및

상기 제3 모델 행렬의 각 인덱스를 대응하는 (z/2)×(z/2) 차원의 영 행렬, (z/2)×(z/2) 차원의 단위 행렬 혹은 (z/2)×(z/2) 차원의 퍼뮤테이션 행렬로 대체하여 패리티 검사 행렬을 생성하는 것을 포함하는,

패리티 검사 행렬 생성 방법.

청구항 2

제1항에 있어서,

'X'가 '2S+1'인 경우, 'X'는 $\begin{bmatrix} 0 & a_{1,2}=S+1 \\ a_{2,1}=S & 0 \end{bmatrix}$ 로 대체되고,

'X'가 '2S'인 경우, 'X'는 $\begin{bmatrix} a_{1,1}=S & 0 \\ 0 & a_{2,2}=S \end{bmatrix}$ 로 대체되는,

패리티 검사 행렬 생성 방법.

청구항 3

제1항에 있어서,

'X'가 '2S+1'인 경우, 'X'는 $\begin{bmatrix} 0 & a_{1,2}=S \\ a_{2,1}=S+1 & 0 \end{bmatrix}$ 로 대체되고,

'X'가 '2S'인 경우, 'X'는 $\begin{bmatrix} a_{1,1}=S & 0 \\ 0 & a_{2,2}=S \end{bmatrix}$ 로 대체되는,

패리티 검사 행렬 생성 방법.

청구항 4

제1항에 있어서,

'X'가 'S+1'인 경우, 'X'는 $\begin{bmatrix} a_{1,1}=S+1 & 0 \\ 0 & a_{2,2}=S \end{bmatrix}$ 로 대체되고,

'X'가 '2S'인 경우, 'X'는 $\begin{bmatrix} 0 & a_{1,2}=S \\ a_{2,1}=S & 0 \end{bmatrix}$ 로 대체되는,

패리티 검사 행렬 생성 방법.

청구항 5

제1항에 있어서,

'X'가 '2S+1'인 경우, 'X'는 $\begin{bmatrix} a_{1,1}=S & 0 \\ 0 & a_{2,2}=S+1 \end{bmatrix}$ 로 대체되고,

'X'가 '2S'인 경우, 'X'는 $\begin{bmatrix} 0 & a_{1,2}=S \\ a_{2,1}=S & 0 \end{bmatrix}$ 로 대체되는,

패리티 검사 행렬 생성 방법.

청구항 6

제1항에 있어서,

'X'가 '2S+1'인 경우, 'X'는 $\begin{bmatrix} a_{1,1}=S+1 & 0 \\ 0 & a_{2,2}=S \end{bmatrix}$ 로 대체되고,

'X'가 '2S'인 경우, 'X'는 $\begin{bmatrix} a_{1,1}=S & 0 \\ 0 & a_{2,2}=S \end{bmatrix}$ 로 대체되는,

패리티 검사 행렬 생성 방법.

청구항 7

제1항에 있어서,

'X'가 '2S+1'인 경우, 'X'는 $\begin{bmatrix} a_{1,1}=S & 0 \\ 0 & a_{2,2}=S+1 \end{bmatrix}$ 로 대체되고,

'X'가 '2S'인 경우, 'X'는 $\begin{bmatrix} a_{1,1}=S & 0 \\ 0 & a_{2,2}=S \end{bmatrix}$ 로 대체되는,

패리티 검사 행렬 생성 방법.

청구항 8

제1항에 있어서,

'X'가 '2S+1'인 경우, 'X'는 $\begin{bmatrix} 0 & a_{1,2}=S+1 \\ a_{2,1}=S & 0 \end{bmatrix}$ 로 대체되고,

'X'가 '2S'인 경우, 'X'는 $\begin{bmatrix} 0 & a_{1,2}=S \\ a_{2,1}=S & 0 \end{bmatrix}$ 로 대체되는,

패리티 검사 행렬 생성 방법.

청구항 9

제1항에 있어서,

'X'가 '2S+1'인 경우, 'X'는 $\begin{bmatrix} 0 & a_{1,2}=S \\ a_{2,1}=S+1 & 0 \end{bmatrix}$ 로 대체되고,

'X'가 '2S'인 경우, 'X'는 $\begin{bmatrix} 0 & a_{1,2}=S \\ a_{2,1}=S & 0 \end{bmatrix}$ 로 대체되는,

패리티 검사 행렬 생성 방법.

청구항 10

삭제

청구항 11

삭제

명 세 서

발명의 상세한 설명

기 술 분 야

[0001] 본 발명은 부호화 및 복호화 방법에 관한 것이다. 보다 구체적으로는, 저밀도 패리티 검사(LDPC: Low Density Parity Check) 코드를 이용하여 입력 데이터를 부호화하거나 부호화된 데이터를 복호화하는 방법에 관한 것이다.

배 경 기 술

[0002] 일반적으로 부호화(encoding)라 함은 송신측에서 송신된 데이터가 통신 채널을 통하여 전송되는 과정에서 발생되는 신호의 일그러짐, 손실 등에 의한 오류의 발생에도 불구하고 수신측에서 원래의 데이터를 복원할 수 있도록 하기 위하여 송신측에서 데이터 처리를 하는 과정을 의미한다. 복호화(decoding)은 부호화되어 송신된 신호를 수신측에서 원래의 데이터로 복원하는 과정이다.

[0003] 최근에 LDPC 코드를 이용한 부호화 방법이 부각되고 있다. LDPC 코드는 패리티 검사 행렬(parity check matrix) H의 원소(element)들의 대부분이 0이어서 저밀도(low density)인 선형 블록 부호(linear block code)로서 1962년 갤러거(Gallager)에 의해 제안되었다. LDPC 부호는 매우 복잡하여 제안 당시의 하드웨어 기술로는 구현이 불가능하였기 때문에 잊혀져 있다가 1995년에 재발견되어 성능이 매우 우수함이 입증된 이래로 최근에 그에 관한 연구가 활발히 진행되고 있는 상황이다. (참고문헌: [1] Robert G. Gallager, "Low-Density Parity-Check Codes", The MIT Press, September 15, 1963. [2] D.J.C.Mackay, Good error-correcting codes based on very sparse matrices, IEEE Trans. Inform. Theory, IT-45, pp.399-431(1999))

[0004] LDPC 코드의 패리티 검사 행렬은 1의 개수가 매우 적기 때문에 매우 큰 블록 크기에서도 반복 복호를 통하여 복호가 가능하여 블록 크기가 매우 커지면 터보 코드처럼 섀넌(Shannon)의 채널 용량 한계에 근접하는 성능을 보인다.

[0005] LDPC 코드는 (n-k)×n 패리티 검사 행렬 H에 의해 설명될 수 있다. 상기 패리티 검사 행렬 H에 대응하는 생성 행렬(generator matrix) G는 다음의 수학식1에 의해 구할 수 있다.

수학식 1

[0006] $H \cdot G = 0$

[0007] LDPC 코드를 이용한 부호화 및 복호화 방법에 있어서는 송신측에서 상기 패리티 검사 행렬 H와 수학식1의 관계에 있는 상기 생성 행렬 G를 이용하여 다음의 수학식2에 의해 입력 데이터를 부호화한다.

수학식 2

[0008] $c = G \cdot u$ (여기서, c는 코드워드(codeword)이고, u는 데이터 프레임임.)

[0009] 상기한 바와 같이, LDPC 코드를 이용한 부호화 방법에서는 상기 패리티 검사 행렬 H가 가장 중요한 요소라 할 수 있다. 상기 패리티 검사 행렬 H는 대략 1000×2000 정도의 크기를 갖기 때문에 부호화 및 복호화 과정에서 많은 연산이 요구되고, 구현이 매우 복잡하며, 많은 저장 공간을 요구한다.

[0010] 일반적으로 패리티 검사 행렬 H에 더 많은 무게(weight)를 부가하는 것은, 패리티 검사 방정식들(parity check equations)에 더 많은 변수를 부가하는 것이기 때문에 LDPC 코드에 의한 부호화 및 복호화 방법에서 더 좋은 성능을 발휘할 수 있다. 여기서, 무게란 패리티 검사 행렬에서 각 열(column) 또는 행(row)에 포함된 '1'의 개수를 의미한다. 그러나, 패리티 검사 행렬 H에 더 많은 무게를 부가하면 패리티 검사 행렬 전체에 4-싸이클(4-cycle)이나 6-싸이클(6-cycle)을 형성하는 경우가 많아져 이에 따라 LDPC 코드에 의한 부호화 및 복호화 과정의 복잡도를 증가시키고 성능을 저하시킬 수 있는 위험성도 따른다.

[0011] 4-싸이클은 패리티 검사 행렬 H의 두 행이 두 개의 지점에 동시에 1을 갖는 경우를 의미하고, 6-싸이클은 임의의 세 행 중에서 선택된 모든 조합 가능한 두 개의 행이 같은 지점에 1을 갖는 경우를 의미한다. 패리티 검사 행렬이 4-싸이클이나 6-싸이클을 많이 갖고 있다고 하는 것은 부호화 또는 복호화 성능이 저하될 가능성이 높음을 의미한다.

발명의 내용

해결 하고자하는 과제

[0012] 상기한 바와 같이, LDPC 코드를 이용한 부호화 및 복호화 방법에 있어서 패리티 검사 행렬 각 행 또는 열의 무게를 어떻게 배분할 것인지와 4-싸이클이나 6-싸이클과 등과 같은 싸이클을 어떻게 제어할 것인지는 부호화 및 복호화 성능에 지대한 영향을 미치게 된다.

[0013] 본 발명은 상기한 바와 같은 종래기술의 문제점을 해결하기 위해 안출된 것으로서, 본 발명의 목적은 LDPC 코드를 이용한 부호화 및 복호화 방법에서 그 성능을 향상시킬 수 있는 방안을 제공하는 것이다.

[0014] 본 발명의 다른 목적은 패리티 검사 행렬의 행 또는 열의 무게 및 싸이클 수를 용이하게 제어할 수 있는 방안을 제공하는 것이다.

과제 해결수단

[0015] LDPC 코드에 의한 구조화된 부호화 및 복호화 방법에 있어서, 부호화 및 복호화 시에 사용되는 패리티 검사 행렬은 각각 특정 서브 행렬을 지시하는 다수의 인덱스들 이루어지는 모델 행렬의 형태로 저장된다. 부호화 또는 복호화 수행 시에 상기 모델 행렬의 각 인덱스를 해당 인덱스가 지시하는 서브 행렬으로 대체함으로써 패리티 검사 행렬이 생성된다. 본 발명의 일 양상에 따르면, 모델 행렬의 각 인덱스는 둘 이상의 인덱스들을 포함하는 인덱스 행렬로 확장된다. 상기 인덱스 행렬에 포함되는 각 인덱스는 특정 서브 행렬을 의미하고, 각 인덱스를 대응하는 서브 행렬로 대체함으로써 패리티 검사 행렬을 생성할 수 있다.

효 과

[0016] 본 발명에 따르면 다음과 같은 효과가 있다.

[0017] 첫째, 모델 행렬로부터 패리티 검사 행렬을 생성하는 과정에서 각 서브 행렬 인덱스에 따라 모델 행렬을 확장함으로써 무게의 분배(weight distribution) 및 4-싸이클이나 6-싸이클 등과 같은 싸이클 수의 제어가 가능하다.

[0018] 둘째, 2×2 확장에 의해 두 개의 분리된 형태의 퍼뮤테이션 행렬을 사용함으로써 병렬적 부호화 및 복호화 관점에서 효율을 증대시킬 수 있다.

발명의 실시를 위한 구체적인 내용

[0019] 이하에서 첨부된 도면을 참조하여 설명되는 본 발명의 실시예들에 의해 본 발명의 구성, 작용 및 다른 특징들이 용이하게 이해될 수 있을 것이다.

[0020] 본 발명의 실시예들은 구조화된(structured) LDPC 부호화 및 복호화 방법을 전제로 하므로, 이하에서 구조화된 LDPC 부호화 및 복호화 방법에 대해 설명하도록 한다.

[0021] LDPC 코드를 사용하기 위해서는 패리티 검사 행렬을 사용한다. 전술한 바와 같이, 패리티 검사 행렬은 이진 행렬(binary matrix)로서 대부분의 원소(element)는 '0'이고, 일부는 '1'이다. 실제 부호화 또는 복호화에 사용되는 패리티 검사 행렬의 크기는 10^5 비트 이상이기 때문에 패리티 검사 행렬을 저장하기 위해 대용량의 메모리가 필요하다.

[0022] 구조화된 LDPC 부호화 기법에서는 패리티 검사 행렬은 모델 행렬(model matrix)의 형식으로 저장된다. 모델 행렬은 각각 특정 서브 행렬(sub-matrix)을 지시하는 다수의 인덱스들(indexes)로 이루어진다. 즉, 각 서브 행렬은 일정 크기(z×z)의 행렬로서 특정 인덱스에 의해 표현된다. 부호화 또는 복호화 수행 시에 상기 모델 행렬의 각 인덱스를 그 인덱스가 지시하는 서브 행렬로 대체함으로써 패리티 검사 행렬로 확장되어 사용된다.

[0023] 도 1은 모델 행렬의 일 예를 도시한 것이다. 도 1에서, 각 정수는 해당 서브 행렬의 인덱스를 의미한다. 예를 들어, 인덱스가 '-1'인 경우는 특정한 크기의 영 행렬(zero matrix)을 의미하며, 인덱스가 '0'인 경우는 특정한 크기의 단위 행렬(identity matrix)을 의미한다. '-1'과 '0'을 제외한 양의 정수인 인덱스는 해당 서브 행렬이 생성된 소정 규칙을 의미할 수 있다. 예를 들어, 양의 정수인 인덱스는 쉬프트 수(shift number)를 나타낸다. 즉, 각 서브 행렬이 단위 행렬의 각 행 또는 열을 일정 방향으로 쉬프트시켜 생성된 퍼뮤테이션 행렬이라고 할 경우, 상기 쉬프트 수를 해당 서브 행렬의 인덱스로 할 수 있다. 예를 들어, 서브 행렬을 '1'이라는 인덱스로 표현하는 경우, 해당 서브 행렬은 단위 행렬의 각 행 또는 열을 특정한 방향으로 한 칸(행 또는 열)씩 쉬프트함으로써 생성된 것이다. 모델 행렬 내에서 영 행렬인 서브 행렬을 표시하기 위해 아무런 인덱스를 사용하지 않을 수도 있다. 이하의 실시예들에 있어서는 영 행렬인 서브 행렬를 표시하기 위해서 인덱스를 사용하지 않는다.

[0024] 도 2는 상술한 인덱스, 즉 쉬프트 수(shift number)에 따른 행렬의 표현 방법을 설명하기 위한 도면이다. 특정한 패리티 검사 행렬을 4×4 크기의 행렬(즉, 서브 행렬)로 구조화하여 표현하는 경우, '3'이라는 인덱스를 갖는 서브 행렬은 4×4 크기의 단위 행렬의 각 열을 오른쪽으로 세 개의 열만큼 쉬프트시킴으로써 생성된 퍼뮤테이션 행렬이 된다.

[0025] 부호화된 구조화된 LDPC 기법에 따라 각 서브 행렬을 하나의 인덱스로 표현한 모델 행렬을 저장하고 부호화 또는 복호화 시에 저장된 모델 행렬을 원래의 패리티 검사 행렬로 확장하여 사용함으로써 패리티 검사 행렬을 저장하기 위한 메모리 용량을 절약할 수 있다.

[0026] 도 3은 본 발명에 따른 일 실시예를 설명하기 위한 도면이다. 도 3의 실시예는 제1 모델 행렬(G)을 토대로 플로어링(flooring) 연산 과정과 두 번의 확장(expansion) 과정을 거쳐 부호화 또는 복호화 시에 사용하기 위한 패리티 검사 행렬(H)을 생성하는 예이다. 도 3에서, 제1 모델 행렬을 표시하기 위한 기호로 'G'를 사용하였으나, 이는 생성 행렬(generation matrix)을 의미하는 'G'와는 구별되어야 한다.

[0027] 도 3을 참조하면, 상기 제1 모델 행렬(G)에 대해 플로어링 연산을 수행하여 제2 모델 행렬(G')을 생성한다. 상기 플로어링 연산은 가장 큰 차원(예를 들어, 12×12)의 제1 서브 행렬에 대한 제1 모델 행렬로부터 보다 작은 차원(예를 들어, 5×5)의 제2 서브 행렬에 대한 제2 모델 행렬을 생성하기 위한 연산 방법으로서 다음의 수학식 3에 의해 표현될 수 있다.

수학식 3

[0028] $$Shift(z)=floor(shift(z_{max})z/z_{max})$$

[0029] 여기서, $shift(z)$는 상기 제2 서브 행렬의 쉬프트 간격의 수, $floor(x)$는 x로부터 음의 무한대(minus infinity) 방향으로 가장 가까운 정수를 의미한다.

[0030] 다시 말해서, 상기 제1 모델 행렬의 각 원소가 $z_{max}\times z_{max}$ 차원의 제1 서브 행렬을 지시하는 인덱스(예를 들어, 기본 퍼뮤테이션 행렬의 각 행 또는 열을 일정 방향으로 쉬프트시킨 경우 그 쉬프트 수)에 의해 구성되는 경우, 상기 제1 모델 행렬로부터 각 원소가 z×z 차원의 제2 서브 행렬을 지시하는 인덱스로 구성되는 제2 모델 행렬을 생성할 필요성이 있는 경우가 있다. 이때, 수학식 3을 이용하여 상기 제1 모델 행렬의 각 원소를 상기 제2

모델 행렬의 각 원소로 대체할 수 있다.

[0031] 예를 들어, 12×12 차원의 단위 행렬의 각 행 또는 열을 특정 방향으로 '7' 만큼씩 쉬프트시켜 형성한 제1 서브 행렬을 5×5 차원의 단위 행렬에 대하여 매핑시켜 형성한 제2 서브 행렬은 수학식 3에 의하여 다음과 같이 구할 수 있다.

[0032] shift(5) = floor(shift(12)×5÷12) = floor(7×5÷12) = floor(2.92) = 2

[0033] 다시 말해서, 12×12 차원의 단위 행렬의 각 행 또는 열을 일정 방향으로 '7' 만큼씩 쉬프트시켜 형성한 제1 서브 행렬은 5×5 차원의 단위 행렬에 대하여 각 행 또는 열을 '2' 만큼씩 쉬프트시켜 형성한 제2 서브 행렬에 매핑된다.

[0034] 이와 같은 방법으로 상기 제1 모델 행렬의 각 원소 값들을 상기 제2 모델 행렬의 원소 값들로 대치시킴으로써 상기 제2 모델 행렬을 생성할 수 있다. 상기의 'floor' 연산은 이를 하드웨어 또는 소프트웨어로 구현할 때 복잡도를 매우 단순하게 할 수 있다. 상기 제1 모델 행렬로부터 상기 제2 모델 행렬을 생성하는 과정에서 상기 제1 모델 행렬의 각 인덱스에 모듈러(modulo) 연산을 수행하여 상기 제2 모델 행렬을 생성하는 것도 가능하다.

[0035] 도 3에서, 2×2 확장(expansion) 과정은 상기 제2 모델 행렬(G')의 각 원소(인덱스)를 2×2 차원의 인덱스 행렬로 확장하여 제3 모델 행렬(G'')을 생성하는 과정이다. 도 4는 2×2 확장 방법의 일 예를 설명하기 위한 도면이다. 도 4(a)는 상기 제2 모델 행렬의 특정 인덱스가 홀수인 경우의 2×2 확장 방법이고, 도 4(b)는 상기 제2 모델 행렬의 특정 인덱스가 짝수인 경우의 2×2 확장 방법을 나타낸 것이다. 상기 제2 모델 행렬(G')의 각 서브 행렬은 2×2 확장에 의해 상기 제3 모델 행렬(G'')에서 네 개의 서브 행렬들로 확장된다. 다만, 상기 제3 모델 행렬(G'')의 각 서브 행렬의 크기는 상기 제2 모델 행렬(G')의 각 서브 행렬의 1/2이 되기 때문에 상기 제2 모델 행렬(G')과 제3 모델 행렬(G'')의 전체 크기는 동일하게 된다. 예를 들어, 도 4에서, 크기가 1024인 서브 행렬은 크기가 512인 네 개의 서브 행렬들로 확장된다.

[0036] 도 4(a)에 도시된 바와 같이, 상기 제2 모델 행렬의 특정 원소, 즉 특정 인덱스가 홀수(2S+1, S는 0 이상의 정수)인 경우에, 해당 서브 행렬은 2×2 차원의 크로스 대각 구조(cross diagonal structure)의 형태로 확장된다. 즉, 상기 제2 모델 행렬의 각 서브 행렬은 네 개의 서브 행렬들로 확장된다. 이때, 네 개의 서브 행렬들 중에서 (1, 1) 및 (2, 2) 위치의 서브 행렬들은 영 행렬이 되고, (1, 2) 위치의 서브 행렬의 인덱스는 (S+1)이 되며, (2, 1) 위치의 서브 행렬의 인덱스는 S가 된다.

[0037] 도 4(b)에서, 상기 제2 모델 행렬의 특정 원소, 즉 특정 인덱스가 짝수(2S, S는 0 이상의 정수)인 경우에, 해당 서브 행렬은 2×2 차원의 대각 구조(diagonal structure)의 형태로 확장된다. 즉, 도 4(b)에 도시된 바와 같이, (1, 1) 및 (2, 2) 위치의 서브 행렬들의 인덱스는 S가 되고, (1, 2) 및 (2, 1) 위치의 서브 행렬들은 영 행렬이 된다. 상기 제2 모델 행렬의 서브 행렬의 인덱스가 '0'인 경우는 해당 서브 행렬이 단위 행렬을 의미하는 것으로서, 이 경우는 도 4(b)의 예에 따라 확장된다. 상기 제2 모델 행렬의 특정 서브 행렬이 영 행렬인 경우 2×2 확장 과정에 의해 1/2 크기의 네 개의 영 행렬로 확장된다.

[0038] 도 5는 도 4의 실시예에 따라 제2 모델 행렬(G')에 2×2 확장을 수행하여 제3 모델 행렬(G'')을 생성하는 예를 설명하기 위한 도면으로서, 4×5 개의 서브 행렬들을 갖는 제2 모델 행렬(G')이 8×10 개의 서브 행렬들을 갖는 제3 모델 행렬(G'')로 확장됨을 확인할 수 있다. 이때, 상기 제3 모델 행렬(G'')의 각 서브 행렬의 크기는 상기 제2 모델 행렬(G')의 각 서브 행렬 크기의 1/2이 된다. 도 5에서, '-1'은 (최대 쉬프트 크기-1)의 인덱스를 의미한다. 예를 들어, 서브 행렬의 크기가 1024인 경우 최대 쉬프트 크기는 1024가 되므로, '-1'인 인덱스는 1023과 동일한 의미이다.

[0039] 도 6a 및 도 6b는 본 발명의 다른 실시예에 따른 2×2 확장 방법의 예들을 도시한 것이다. 도 6a는 상기 제2 모델 행렬의 특정 서브 행렬의 인덱스가 홀수인지 짝수인지에 관계없이 대각 구조의 형태로 확장된 예이고, 도 6b는 크로스 대각 구조로 확장된 예이다.

[0040] 도 7a 내지 도 7g는 본 발명의 또 다른 실시예에 따른 2×2 확장 방법의 예들을 도시한 것이다. 도 7a 내지 도 7g의 실시예들은 상기 제2 모델 행렬의 특정 서브 행렬의 인덱스가 홀수(2S+1)인지 또는 짝수(2S)인지에 따라서 확장 방법을 달리하는 예들로서, 대각 구조 또는 크로스 대각 구조의 형태로 확장된다. 각각의 확장 방법에 대한 구체적인 설명은 도 4의 실시예를 참조할 수 있다.

[0041] 도 8a 및 도 8b는 본 발명의 일 실시예에 따라 2×2 확장을 반복해서 실시하는 예를 설명하기 위한 도면이다. 도 8a에서, 인덱스가 1023인 서브 행렬(a)은 도 4(a)의 예에 따라 (b)와 같이 2×2의 크로스 대각 구조로 확장

되고, (b)의 각 서브 행렬은 다시 2×2 확장되어 (c)와 같은 구조의 행렬이 될 수 있다. 도 8b에서, 인덱스가 1024인 서브 행렬(a)은 도 4(b)의 예에 따라 (b)와 같이 2×2의 대각 구조로 확장되고, (b)의 각 서브 행렬은 다시 2×2 확장되어 (c)와 같은 구조의 행렬이 될 수 있다. 다시 말해서, 2×2 확장을 두 번 반복한 것이다. 2×2 확장을 세 번 이상 할 수 있음은 자명하다. 도 8a 및 도 8b에서, (a)의 서브 행렬의 크기는 1024이고, (b)의 각 서브 행렬의 크기는 512이며, (c)의 각 서브 행렬의 크기는 256이다.

[0042] 다시 도 3을 참조하면, 2×2 확장에 의해 생성된 제3 모델 행렬(G'')은 패리티 검사 행렬로 확장된다. 즉, 상기 제3 모델 행렬의 각 원소를 해당 인덱스가 의미하는 서브 행렬로 대체함으로써 부호화 또는 복호화 시에 사용되는 패리티 검사 행렬을 생성한다. 보다 구체적으로, 상기 제3 모델 행렬의 '0'의 인덱스를 갖는 특정 원소는 z×z 차원의 단위 행렬로 대체되고, 양의 정수인 인덱스를 갖는 특정 원소는 상기 단위 행렬의 각 행 또는 열이 상기 양의 정수만큼 쉬프트되어 형성된 퍼뮤테이션 행렬로 대체된다. 영 행렬인 상기 제3 모델 행렬의 서브 행렬은 z×z 차원의 영 행렬로 대체된다.

[0043] 도 3에서, 상기 플로어링 과정과 2×2 확장 과정의 순서는 바뀔 수 있다. 다시 말해서, 제1 모델 행렬(G)을 상술한 바와 같은 2×2 확장 방법에 따라 확장하여 제2 모델 행렬을 생성하고, 생성된 상기 제2 모델 행렬의 각 원소, 즉 각 인덱스에 대해 플로어링 연산을 수행하여 제3 모델 행렬을 생성하는 과정을 거칠 수도 있다. 상기 제3 모델 행렬이 확장되어 패리티 검사 행렬이 된다.

[0044] 도 3에서, 상기 제3 모델 행렬이 확장되어 생성된 패리티 검사 행렬에 대해 부호화 또는 복호화를 수행하기 전에 행/열의 재배열 과정을 거칠 수 있다. 도 9는 행/열의 재배열 과정 및 패리티 검사 행렬을 통한 통신 시스템의 송신측에서의 부호화 과정을 설명하기 위한 절차 흐름도이다.

[0045] 도 9를 참조하면, 입력 데이터열의 부호화를 위해 소정 부분이 정형화된 구조를 갖는 최종 패리티 검사 행렬을 생성하기 위해 패리티 검사 행렬의 적어도 하나 이상의 행이나 열 또는 적어도 하나 이상의 행과 열을 재배열(permutation)한다[S91].

[0046] 일반적으로, 패리티 검사 행렬 H는 $H=[H_d | H_p]$(H_d는 (n-k)×k, H_p는 (n-k)×(n-k) 차원임)로 표현될 수 있다. H_d는 정보어 부분이고, H_p는 패리티 부분이다. 상기 k는 입력 데이터열의 길이(비트 단위)이고, 상기 n은 부호화된 코드워드(c)의 길이(비트 단위)를 의미한다.

[0047] 상기 패리티 부분 H_p는 이중 대각(dual diagonal) 구조를 갖는 것이 바람직하다. 상기 이중 대각 행렬은 그 차원에 관계없이 주 대각(main diagonal)과 상기 주 대각 바로 밑 또는 위의 대각이 1이고 나머지가 모두 0인 행렬을 의미한다. 따라서, 상기 재배열 과정(S91)에서 상기 패리티 검사 행렬의 패리티 부분 H_p가 이중 대각 구조를 갖도록 행 및/또는 열을 재배열하는 것이 바람직하다.

[0048] 한편, 상기 행/열의 재배열 과정은, 도 3에서 상기 제1 모델 행렬(G), 제2 모델 행렬(G') 및 제3 모델 행렬(G'') 중 어느 하나 또는 둘 이상의 행렬에 대해 수행되어 최종적으로 패리티 검사 행렬의 패리티 부분이 이중 대각 구조를 갖도록 하면 된다. 나아가, 상기 행 및/또는 열의 재배열 과정(S91)을 통해 최종 패리티 검사 행렬의의 패리티 부분의 첫 번째 열의 무게가 3이 되도록 하는 것이 바람직하다. 도 10은 도 5의 제3 모델 행렬(G'')에 대해 행/열 재배열 과정을 수행하여 패리티 부분이 이중 대각 구조가 되도록 한 예를 도시한 것이다.

[0049] 상기 최종 패리티 검사 행렬이 생성되면 이를 통해 입력 데이터열을 부호화한다[S93].

[0050] 상기한 바와 같이, LDPC 코드를 이용한 부호화 방법에 있어서는 생성 행렬(generator matrix)을 이용하여 입력 소스 데이터를 부호화할 수 있다. 즉, k 비트의 입력 소스 데이터 $s_{1\times k}$는 생성 행렬에 부호화되어 n 비트의 코드워드 $x_{1\times k}$이 된다. 코드워드 **x**는 $\mathbf{x}=[s\ p]=[s_0, s_1, \ldots, s_{k-1}, p_0, p_1, \ldots, p_{m-1}]$ 의 구성을 갖는다(여기서, $(p_0, p_1, \ldots, p_{m-1})$은 패리티 검사 비트(parity check bits)이고, $(s_0, s_1, \ldots, s_{k-1})$은 시스템 비트(systematic bits)이다.).

[0051] 그러나, 상기 생성 행렬을 이용한 부호화 방법은 매우 복잡하다. 따라서, 이러한 복잡도를 줄이기 위해 상기 생성 행렬에 의하지 않고, 패리티 검사 행렬 H를 이용해 직접 입력 소스 데이터를 부호화하는 것이 바람직하다. 즉, $\mathbf{x}=[s\ p]$이므로, $H \cdot x = 0$인 특성을 이용하면 $H \cdot x = H \cdot [s\ p] = 0$이 되고, 이 식으로부터 패리티 검사 비트 p를 얻을 수 있어, 결과적으로 코드워드 $\mathbf{x}=[s\ p]$를 구할 수 있다.

[0052] 다음으로, 상기 패리티 검사 행렬에 의해 부호화된 코드워드를 구성하는 심볼들의 순서를 재배열(de-

permutation)한다[S95]. 이때, 코드워드를 구성하는 심볼들의 순서는 상기 행/열 재배열 단계(S91)에서 열이 재배열되지 않은 경우의 최종 패리티 검사 행렬의 열들(columns)의 순서에 대응하도록 재배열된다. 즉, 코드워드의 심볼들을 재배열(de-permutation)하는 경우 그 순서는 상기 행/열 재배열 단계(S91)에서 이루어진 열의 재배열(permutation) 순서의 역순이 된다.

[0053] LDPC 부호화 방법에 있어서, k 개의 입력 데이터열에 대해 (n-k)×n의 패리티 검사 행렬에 의해 부호화를 수행하면 n 개의 심볼들로 구성되는 코드워드가 출력된다. 즉, 코드워드를 구성하는 심볼들의 개수(n)는 패리티 검사 행렬의 열들의 개수와 동일하고, 상기 코드워드를 구성하는 심볼들의 순서는 상기 패리티 검사 행렬의 열들의 순서에 대응한다.

[0054] 패리티 검사 행렬의 소정 부분(예를 들어, 패리티 부분)을 정형화된 구조로 만들기 위해 상기 행/열의 재배열 과정에 따라 열을 재배열할 경우, 이에 따라 부호화된 코드워드의 심볼들의 순서도 재배열된다. 따라서, 재배열된 코드워드 심볼들의 순서를 재배열되기 전의 패리티 검사 행렬의 열에 대응하도록 재배열할 필요가 있다. 코드워드의 순서를 재배열함으로써 ARQ 또는 HARQ 등과 같은 재전송 기법을 효율적으로 지원할 수 있다.

[0055] 상기 패리티 검사 행렬을 이용한 부호화가 송신측에서 수행된 경우 상기 재배열된 코드워드를 수신측으로 전송한다[S97]. 상기 코드워드의 전송 과정에서 레이트 매칭(rate matching), 인터리빙(interleaving), 변조(modulation) 등 통신 시스템에서 요구되는 데이터 처리 과정이 수행될 수 있다.

[0056] 본 문서에서 사용된 용어는 동일한 의미를 갖는 다른 용어들로 대체될 수 있다. 예를 들어, 모델 행렬은 '기본 행렬(base matrix)'이라는 용어에 의해 대체 가능하고, 서브 행렬은 '퍼뮤테이션 행렬'이라는 용어로 대체될 수 있다.

[0057] 본 발명은 본 발명의 정신 및 필수적 특징을 벗어나지 않는 범위에서 다른 특정한 형태로 구체화될 수 있음은 당업자에게 자명하다. 따라서, 상기의 상세한 설명은 모든 면에서 제한적으로 해석되어서는 아니되고 예시적인 것으로 고려되어야 한다. 본 발명의 범위는 첨부된 청구항의 합리적 해석에 의해 결정되어야 하고, 본 발명의 등가적 범위 내에서의 모든 변경은 본 발명의 범위에 포함된다.

도면의 간단한 설명

[0058] 도 1은 모델 행렬의 일 예를 도시한 것이다.

[0059] 도 2는 인덱스에 따른 행렬의 표현 방법을 설명하기 위한 도면이다.

[0060] 도 3은 본 발명에 따른 일 실시예를 설명하기 위한 도면이다.

[0061] 도 4는 2×2 확장 방법의 일 예를 설명하기 위한 도면이다.

[0062] 도 5는 도 4의 실시예에 따라 제2 모델 행렬(G')에 2×2 확장을 수행하여 제3 모델 행렬을 생성하는 예를 설명하기 위한 도면이다.

[0063] 도 6a 및 도 6b는 본 발명의 다른 실시예에 따른 2×2 확장 방법의 예들을 도시한 것이다.

[0064] 도 7a 내지 도 7g는 본 발명의 또 다른 실시예에 따른 2×2 확장 방법의 예들을 도시한 것이다.

[0065] 도 8a 및 도 8b는 본 발명의 일 실시예에 따라 2×2 확장을 반복해서 실시하는 예를 설명하기 위한 도면이다.

[0066] 도 9는 행/열의 재배열 과정 및 패리티 검사 행렬을 통한 통신 시스템의 송신측에서의 부호화 과정을 설명하기 위한 절차 흐름도이다.

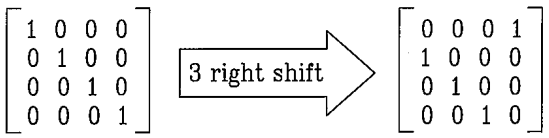
[0067] 도 10은 도 5의 제3 모델 행렬(G'')에 대해 행/열 재배열 과정을 수행하여 패리티 부분이 이중 대각 구조가 되도록 한 예를 도시한 것이다.

도면

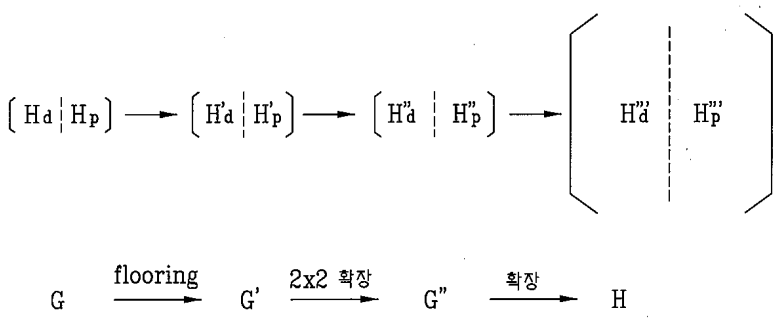
도면1

2	-1	19	-1	47	-1	48	-1	36	-1	82	-1	47	-1	15	-1	95	0	-1	-1	-1	-1	-1	-1
-1	69	-1	88	-1	33	-1	3	-1	16	-1	37	-1	40	-1	48	-1	0	0	-1	-1	-1	-1	-1
10	-1	86	-1	62	-1	28	-1	85	-1	16	-1	34	-1	73	-1	-1	-1	0	0	-1	-1	-1	-1
-1	28	-1	32	-1	81	-1	27	-1	88	-1	5	-1	56	-1	37	-1	-1	-1	0	0	-1	-1	-1
23	-1	29	-1	15	-1	30	-1	66	-1	24	-1	50	-1	62	-1	-1	-1	-1	-1	0	0	-1	-1
-1	30	-1	65	-1	54	-1	14	-1	0	-1	30	-1	74	-1	0	-1	-1	-1	-1	-1	0	0	-1
32	-1	0	-1	15	-1	56	-1	85	-1	5	-1	6	-1	52	-1	0	-1	-1	-1	-1	-1	0	0
-1	0	-1	47	-1	13	-1	61	-1	84	-1	55	-1	78	-1	41	95	-1	-1	-1	-1	-1	-1	0

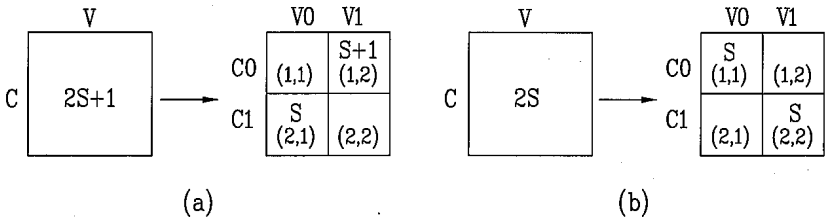
도면2



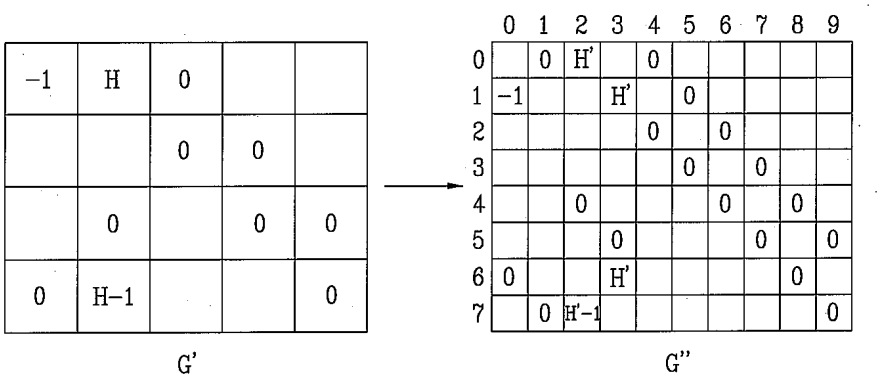
도면3



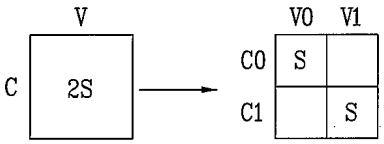
도면4



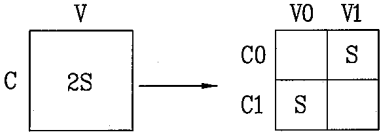
도면5



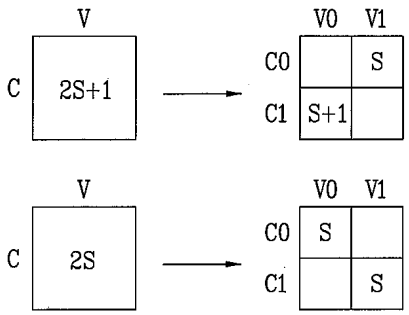
도면6a



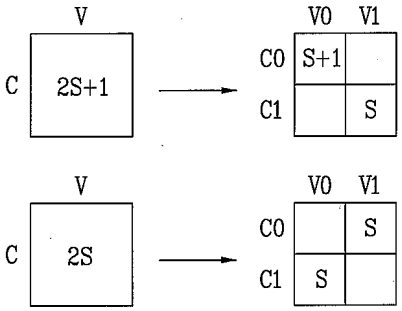
도면6b



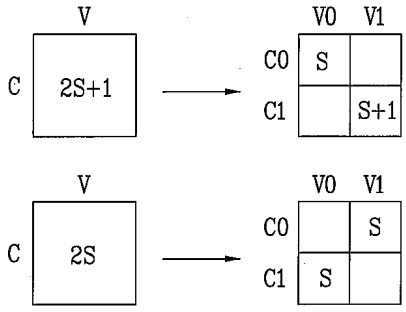
도면7a



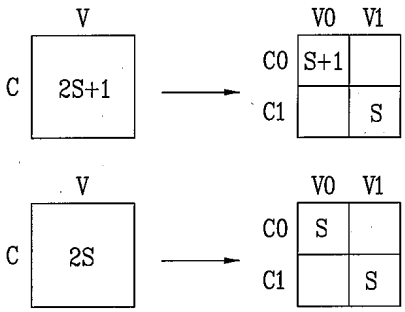
도면7b



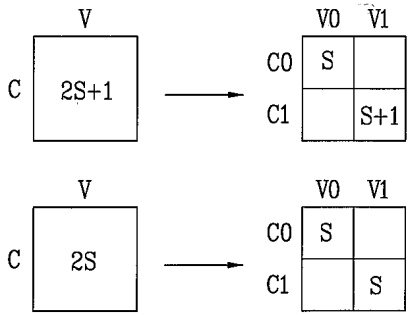
도면7c



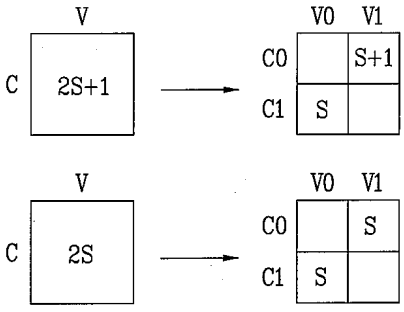
도면7d



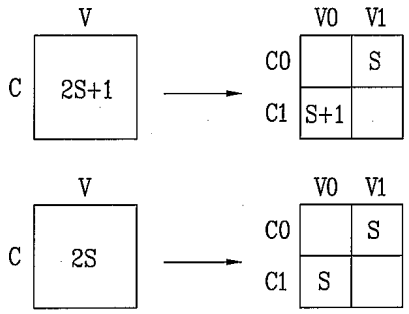
도면7e



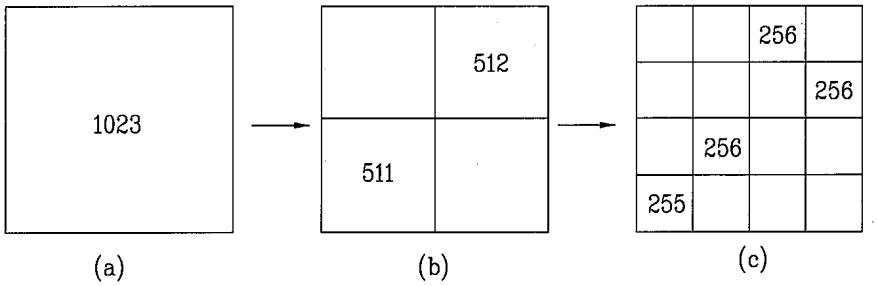
도면7f



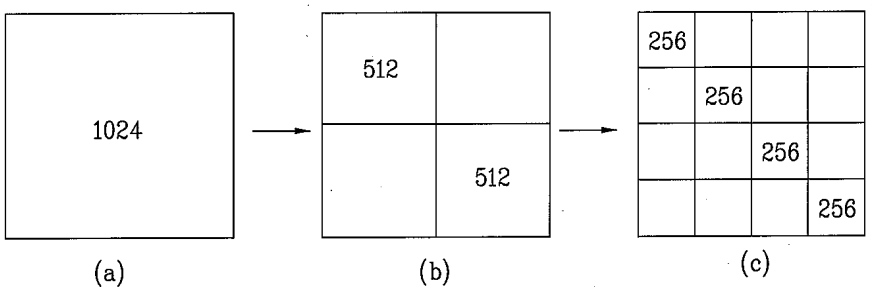
도면7g



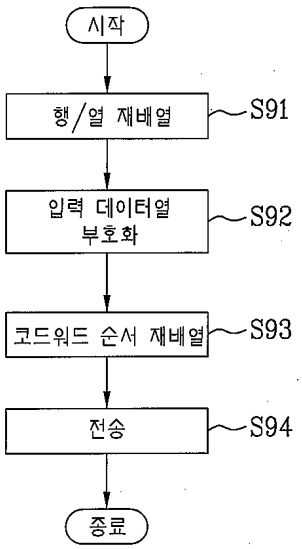
도면8a



도면8b



도면9



도면10

	0	2	3	5	7	9	1	4	6	8
1	-1		H'	0						
3				0	0					
5			0		0	0				
7		H'-1				0	0			
0		H'					0	0		
2								0	0	
4		0							0	0
6	0		H'							0