(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2008-0041488
(43) 공개일자 2008년05월13일

(51) Int. Cl.
H03M 13/27 (2006.01) *H04B 1/69* (2006.01)
(21) 출원번호 10-2006-0109627
(22) 출원일자 2006년11월07일
심사청구일자 2008년02월05일

(71) 출원인
삼성전자주식회사
경기도 수원시 영통구 매탄동 416
연세대학교 산학협력단
서울 서대문구 신촌동 134 연세대학교
(72) 발명자
김동호
서울특별시 서초구 서초1동 1436-1 현대아파트 21동 1206호
김영수
경기도 성남시 분당구 정자동 29 선경연립 111동 401호
(뒷면에 계속)
(74) 대리인
이건주

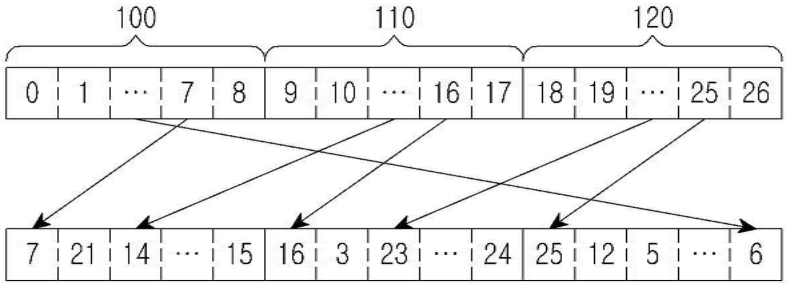
전체 청구항 수 : 총 5 항

(54) 병렬 인터리빙 장치 및 방법

(57) 요 약

본 발명은 병렬 인터리빙 방법에 있어서, 입력되는 정보 비트들을 미리 설정된 개수의 서브 블록으로 구분하는 과정과, 각 서브 블록으로 구분된 정보 비트들을 미리 설정된 제1 인터리빙 규칙에 의해 인터리빙 하는 과정을 포함한다.

대표도 - 도1a



(72) 발명자

유철우

서울특별시 서초구 서초2동 서초트라팰리스 C-1402

송홍엽

서울특별시 영등포구 여의도동 광장아파트 1동 306호

김대선

서울특별시 동대문구 청량리1동 32번지

오현영

경기도 안양시 동안구 호계2동 럭키아파트 108동 1303호

특허청구의 범위

청구항 1

병렬 인터리빙 방법에 있어서,

입력되는 정보 비트들을 미리 설정된 개수의 서브 블록으로 구분하는 과정과,

각 서브 블록으로 구분된 정보 비트들을 미리 설정된 제1 인터리빙 규칙에 의해 인터리빙 하는 과정을 포함하는 병렬 인터리빙 방법.

청구항 2

제1항에 있어서,

상기 서브 블록의 개수를 열(column)의 개수로 설정하고, 각 서브 블록당 비트 개수를 행(row)의 개수로 설정하여 행렬을 생성하는 과정을 더 포함하는 병렬 인터리빙 방법.

청구항 3

제1항에 있어서,

상기 서브 블록의 개수를 행(row)의 개수로 설정하고, 각 서브 블록당 비트 개수를 열(column)의 개수로 설정하여 행렬을 설정하는 과정을 더 포함하는 병렬 인터리빙 방법.

청구항 4

제2항 또는 제3항에 있어서,

상기 행렬은 라틴 방진을 만족하는 행과 열이 다수개로 이루어진 행렬임을 특징으로 하는 병렬 인터리빙 방법.

청구항 5

제1항에 있어서,

상기 제1 인터리빙 규칙은 하기 수학식 4와 같은 형태를 가짐을 특징으로 하는 병렬 인터리빙 방법.

수학식 4

$$i = M \times s_{jk} + \pi(j)$$

상기 수학식 4에서 M은 열의 개수에 해당하는 값이며, s_{jk}는 상기 행렬에서 행(j)과 열(k)이 교차하는 엘리먼트(element) 값을 의미하며, $\pi(j)$는 행의 개수와 동일한 길이를 가지는 제2 인터리빙 규칙에서 j번째에 해당하는 값을 의미함.

명 세 서

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

<3> 본 발명은 인터리빙(interleaving)에 관한 것으로서, 특히 병렬 처리가 가능한 인터리빙 방법에 관한 것이다.

<4> 차세대 통신 시스템은 패킷 서비스 통신 시스템(packet service communication system) 형태로 발전되어 왔으며, 패킷 서비스 통신 시스템은 버스트(burst)한 패킷 데이터(packet data)를 다수의 이동국들로 송신하는 시스템으로서, 대용량 데이터 송신에 적합하도록 설계되어 왔다. 특히, 차세대 통신 시스템에서는 제한된 주파수 자원을 사용하면서도 데이터 송신량을 증가시키기 위해 고차 변조(HOM: High Order Modulation) 방식을 사용하고 있다. 그러나, 상기 HOM 방식을 사용하기 위해서는 저차 변조(LOM: Low Order Modulation) 방식을 사용하는 경

우에 비해 동일한 성능을 획득하기 위해 더 높은 신호대 잡음비(SNR: Signal to Noise Ratio, 이하 'SNR'이라 칭함)가 요구된다. 이렇게 요구되는 SNR을 감소시키기 위해 성능이 우수한 순방향 오류 정정(FEC: Forward Error Correction) 부호들을 사용하는 것이 필요로 된다.

<5> 상기 순방향 오류 정정 부호들중 대표적인 부호들로는 터보 부호(turbo code)와, LDPC 부호 등이 있다. 상기 터보 부호는 채널 코딩 기술 중 하나로, 베로우(Berrou)가 1993년에 소개한 이후 지금까지도 많은 연구가 이루어지고 있다. 특히 복호시 지연 문제를 해결하기 위해 병렬 구조를 가지는 터보 부호가 관심있게 연구되고 있다.

<6> 병렬 구조의 터보 부호는 정보 블록(block)을 다수개의 서브 블록으로 나누어 부호화 및 복호화를 병렬적으로 수행할 수 있다. 따라서, 기존의 터보 부호에서 사용하던 인터리버 대신 병렬 처리가 가능한 새로운 인터리버가 필요하다.

<7> 베로우는 그의 논문 "Enhancement of Rel. 6 Turbo Code, 3GPP TSG RAN WG1#43, Seoul, Korea, November 7th-11th, 2005"에서 4개의 서브 블록을 가지는 병렬 구조의 인터리버를 제안하였다. 여기서, 정보 블록의 길이는 320 및 640인 경우를 가정하였다. 복호 알고리즘으로는 최대 로그 맵(Max Log MAP) 알고리즘을 사용하였고, 최대 반복 복호 횟수는 8회로 설정하였다. 이와 같이, 베로우가 제안한 병렬 구조의 인터리버는 이전에 제안된 인터리버들보다 뛰어난 성능을 가진다. 하지만, 최적화 과정에 따른 복잡도에 문제점이 있다.

<8> 하기 수학식 1은 4의 주기를 가지는 인터리빙 규칙을 나타낸 것이다.

수학식 1

$$p = \begin{cases} 0 & if \quad j = 0\,[\mathrm{mod}.4] \\ Q_1 & f \quad j = 1\,[\mathrm{mod}.4] \\ 4*P+Q_2 & f \quad j = 2\,[\mathrm{mod}.4] \\ 4*P+Q_3 & f \quad j = 3\,[\mathrm{mod}.4] \end{cases}$$

<9>

<10> 상기 수학식 1에서 P와 Q 값들은 실험을 통해 최적의 성능을 나타내는 값들로 결정된다. 즉, 정보 블록의 길이가320 및 640인 경우, 4개의 서브 블록을 가지는 인터리버를 설계하기 위해서는 P와 Q_1 , Q_2 및 Q_3 값을 결정하여야 한다. 상기 값들을 결정하기 위해, P는 정보 블록의 길이와 서로 소인 수여야 하고, Q_1 , Q_2 및 Q_3는 4의 배수여야 하는 제한 조건을 가진다. 이러한 제한 조건을 고려하더라도 모든 정보 블록의 길이에 대해 최적의 성능을 나타내는 P와 Q 값들을 결정하기 위해서는 많은 연산량을 필요로 한다.

<11> 베로우는 상기 논문에서 320과 640의 정보 블록 길이에 대해서만 하기 수학식 2와 같은 P와 Q 값들을 제시하였다.

수학식 2

$$P = 197, Q_1 = 8, Q_2 = 20, Q_3 = 12$$

$$P = 201, Q_1 = 24, Q_2 = 12, Q_3 = 4$$

<12>

<13> 하지만, 실제 시스템 구현에 있어서는 모든 정보 블록 길이에 대해 P와 Q 값들이 결정되어야 한다. 상기 P와 Q 값들을 결정하기 위해 모든 정보 블록 길이별 경우의 수를 조사하는 것은 효율적이지 못하다.

발명이 이루고자 하는 기술적 과제

<14> 본 발명은 상기와 같은 문제점을 해결 위해 창안된 것으로, 본 발명의 목적은 모든 정보 블록 길이에 대해 병렬

처리가 가능한 인터리빙 방법을 제공함에 있다.

<15> 본 발명의 다른 목적은 복호 과정에서 병렬 복호를 가능하게 하여 복호 수율을 증가시키는 병렬 인터리빙 방법을 제공함에 있다.

<16> 상기한 바와 같은 목적을 달성하기 위한 본 발명의 방법은; 병렬 인터리빙 방법에 있어서, 입력되는 정보 비트들을 미리 설정된 개수의 서브 블록으로 구분하는 과정과, 각 서브 블록으로 구분된 정보 비트들을 미리 설정된 제1 인터리빙 규칙에 의해 인터리빙 하는 과정을 포함한다.

발명의 구성 및 작용

<17> 이하, 본 발명의 바람직한 실시예를 첨부된 도면을 참조하여 상세히 설명한다. 하기의 설명에서는 본 발명의 동작을 이해하는데 필요한 부분만을 설명하며 그 이외의 배경 기술은 본 발명의 요지를 흩트리지 않도록 생략한다.

<18> 본 발명은 모든 정보 블록 길이에 대해 병렬 처리가 가능한 인터리빙 방법을 제안한다. 특히 짧은 길이를 가지는 인터리버를 이용하여 긴 길이를 가지는 정보 블록을 인터리빙 할 수 있는 병렬 구조 인터리빙 방법을 제안한다.

<19> 전체 정보 블록의 길이를 K, 서브 블록의 개수를 L, 각 서브 블록별 비트 수를 M이라고 설정한다. M=K/L이 되며, M은 (2^m-1) 의 배수가 되면 안 된다. 여기서, m은 길쌈부호화기의 시프트 레지스터의 개수이다. 만약, M이 (2^m-1)의 배수가 되면 각 서브 블록에 테일(tail) 비트를 추가하지 않고 테일 바이팅(tail-biting)하는 순환(circular) 부호화를 적용할 수 없게 된다.

<20> 본 발명에서는 작은 길이의 인터리버, 즉 M의 길이를 가지는 인터리버를 이용하여 병렬 처리가 가능한 긴 길이의 인터리버를 설계할 수 있게 된다. 한편, 서브 블록간의 인터리빙을 나타내는 행렬 S는 MxL로 표기한다. 행렬 S의 각 행들은 0부터 L-1까지의 순열이 되어야 한다.

<21> 예컨대, L=4인 서브 블록 4개를 가지는 병렬 구조 인터리빙을 가정하면, 복호기 내에 독립적으로 복호를 수행하는 4개의 프로세서(processor)가 메모리로부터 동시에 4개의 비트들을 독출(reading)한다. 이러한 과정은 각 프로세서가 M번째 비트를 독출하면 종료된다.

<22> 상기와 같이, 본 발명에서 새롭게 제안하는 인터리빙 규칙은 하기 수학식 3과 같은 형태로 나타낼 수 있다.

수학식 3

<23>
$$i = \prod(k * M + j) = M \times s_{jk} + \pi(j)$$

<24> 상기 수학식 3에서 k는 서브 블록의 인덱스 0부터 L-1까지의 값을 가진다. π는 길이 M을 가지는 인터리버의 인터리빙 규칙, Π는 본 발명에 따른 인터리버의 인터리빙 규칙, $s_{jk}(j=0,1,\ldots,M-1,\ k=0,1,\ldots,L-1)$는 행렬 S의 j행 k열의 원소를 나타낸다.

<25> 그러면, 도 1을 참조하여 본 발명에서 서브 블록을 분할하고, 생성된 S행렬을 이용하여 병렬 처리가 가능한 인터리빙 방안을 설명하기로 한다.

<26> 도 1a 및 1b는 본 발명의 실시예에 따른 인터리빙 수행 및 S 행렬을 나타낸 도면이다.

<27> 도 1a 및 1b를 참조하면, 총 정보 블록의 길이 K=27, 서브 블록의 개수 L=3(k=0,1,2), 각 서브 블록당 정보 비트의 수 M=9(j=0,1,...,8)를 가지며, 작은 길이의 인터리빙 규칙은 $\pi=(7,3,5,0,4,1,8,2,6)$라고 가정한다. 물론, 상기 작은 길이의 인터리빙 규칙은 상기 예 이외에도 다양하게 구성할 수 있다. 또한, S 행렬은 3x3 라틴(lantin) 방진이 반복되는 형태로 9x3 행렬을 이룬다. 도 1a에 표시된 숫자는 각 비트의 인덱스(index)를 나타내는 값이다. 상기 인덱스는 0부터 26까지의 값을 가진다.

<28> 각 서브 블록이 3개로 분할되었기 때문에 인터리빙 프로세서도 3개가 필요하며 모두 0부터 8까지 시점에서 인터

리빙을 수행하게 된다.

<29> 먼저, 0번째 시점에서, 제1 프로세서는 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 0번 서브 블록(100)의 8번째 비트(즉, 인덱스 7에 해당하는 비트)를 독출한다. 즉,

$$i = M \times s_{jk} + \pi(j) = 9 \times 0 + 7 = 7$$

이 된다.

<30> 이와 동시에, 제2 프로세서는 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 1번 서브 블록(110)의 8번째 비트(즉, 인덱스 16에 해당하는 비트)를 독출한다. 즉, $i = M \times s_{jk} + \pi(j) = 9 \times 1 + 7 = 16$ 이 된다.

<31> 이와 동시에, 제3 프로세서도 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 2번 서브 블록(120)의 8번째 비트(즉, 인덱스 25에 해당하는 비트)를 독출한다. 즉, $i = M \times s_{jk} + \pi(j) = 9 \times 2 + 7 = 25$ 가 된다.

<32> 다음으로, 1번째 시점에서, 제1 프로세서는 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 2번 서브 블록(120)의 4번째 비트(즉, 인덱스 21에 해당하는 비트)를 독출한다. 즉,

$$i = M \times s_{jk} + \pi(j) = 9 \times 2 + 3 = 21$$

이 된다.

<33> 이와 동시에, 제2 프로세서는 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 0번 서브 블록(100)의 4번째 비트(즉, 인덱스 3에 해당하는 비트)를 독출한다. 즉, $i = M \times s_{jk} + \pi(j) = 9 \times 0 + 3 = 3$ 를 만족한다.

<34> 이와 동시에, 제3 프로세서도 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 1번 서브 블록(110)의 4번째 비트(즉, 인덱스 12에 해당하는 비트)를 독출한다. 즉, $i = M \times s_{jk} + \pi(j) = 9 \times 1 + 3 = 12$ 를 만족한다.

<35> 다음으로, 2번째 시점에서, 제1 프로세서는 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 1번 서브 블록(110)의 6번째 비트(즉, 인덱스 14에 해당하는 비트)를 독출한다. 즉,

$$i = M \times s_{jk} + \pi(j) = 9 \times 1 + 5 = 14$$

가 된다.

<36> 이와 동시에, 제2 프로세서는 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 2번 서브 블록(120)의 6번째 비트(즉, 인덱스 23에 해당하는 비트)를 독출한다. 즉, $i = M \times s_{jk} + \pi(j) = 9 \times 2 + 5 = 23$ 를 만족한다.

<37> 이와 동시에, 제3 프로세서도 상기 수학식 3 및 작은 길이의 인터리빙 규칙을 이용하여 0번 서브 블록(100)의 6번째 비트(즉, 인덱스 5에 해당하는 비트)를 독출한다. 즉, $i = M \times s_{jk} + \pi(j) = 9 \times 0 + 5 = 5$ 를 만족한다.

<38> 상술한 바와 같은 절차로 S 행렬의 나머지 모든 행들에 대해 작은 길이의 인터리빙 규칙을 적용하여 인터리빙을 수행한다. 최종 인터리빙 된 결과는 (7, 21, 14, 0, 22, 10, 8, 20, 15, 16, 3, 23, 9, 4, 19, 17, 2, 24, 25, 12, 5, 18, 13, 1, 26, 11, 6)와 같다. 따라서, 동일 서브 블록에 속하는 비트들이 작은 길이의 인터리빙 규칙에 의해 서로 다른 서브 블록에 인터리빙 되는 확률이 높아짐으로써 성능이 열화되는 것을 방지할 수 있게 된다.

<39> 또한 S 행렬을 도 1b와 같이 행과 열 모두가 순열이 되는 라틴 방진의 형태가 되면, 인터리빙 후에 동일한 서브

블록에서 독출된 비트들이 최대한 균일하게 분포된다. 상기 S 행렬의 행을 결정하는 것에는 다양한 방법들이 존재할 수 있다. 첫번째로 매 행을 랜덤한 순열 형태로 생성할 수 있다. 두번째로, 도 1b에 도시한 바와 같이 라틴 방진이 반복되는 형태로 생성할 수 있다. 세번째로, 상기 두 방법을 결합하여 L-1개 보다 적은 수의 행들과 비교하여 라틴 방진의 형태를 유지하면서 랜덤하게 생성할 수 있다. 마지막 세번째 방안에 따른 S 행렬 생성을 도 2를 참조하여 설명하기로 한다.

<40> 도 2는 본 발명의 실시예에 따른 S 행렬을 생성하는 과정을 도시한 흐름도이다.

<41> 도 2를 참조하면, 먼저 201단계에서 S 행렬의 첫 행은 (0, 1,...,L-1)로 초기화 되고 203단계로 진행한다. 여기서, L은 도 1의 설명에서 기재된 바와 같이 서브 블록의 수를 의미한다.

<42> 상기 203단계에서 카운트(count) 값은 0으로 초기화 되고 205단계로 진행한다. 상기 205단계에서 S 행렬의 행을 (0, 1,...,L-1)와 같은 순열 형태로 랜덤하게 생성하고 207단계로 진행한다. 상기 207단계에서 생성된 행이 이전에 생성된 행들과 비교해 볼 때 라틴 방진 조건을 만족하는지 판단한다. 만약, 라틴 방진 조건을 만족하는 행이 생성된 경우 209단계로 진행하고, 만족하지 않는 행이 생성된 경우 205단계로 되돌아 간다.

<43> 상기 209단계에서 생성된 행은 S 행렬의 count 번째 행으로 저장되고 211단계로 진행한다. 상기 211단계에서 count 값은 1 증가하고 213단계로 진행한다. 상기 213단계에서 count 값이 M보다 작은 경우 205단계로 되돌아가고, count 값이 M 이상의 값을 가지는 경우 S 행렬 생성을 종료하게 된다.

<44> 본 발명의 상세한 설명에서는 구체적인 실시예에 관해 설명하였으나, 본 발명의 범위에서 벗어나지 않는 한도 내에서 여러 가지 변형이 가능함은 물론이다. 그러므로 본 발명의 범위는 설명된 실시예에 국한되지 않으며, 후술되는 특허청구의 범위뿐만 아니라 이 특허청구의 범위와 균등한 것들에 의해 정해져야 한다.

발명의 효과

<45> 상술한 바와 같이, 본 발명은 작은 길이의 인터리버를 가지고 긴 길이의 인터리버를 구현할 수 있다. 즉, 작은 길이의 인터리버는 기존에 사용되고 있던 인터리버로써 기존 인터리버를 이용하여 병렬 처리가 가능한 긴 길이의 인터리버를 구현할 수 있는 이점이 존재한다.

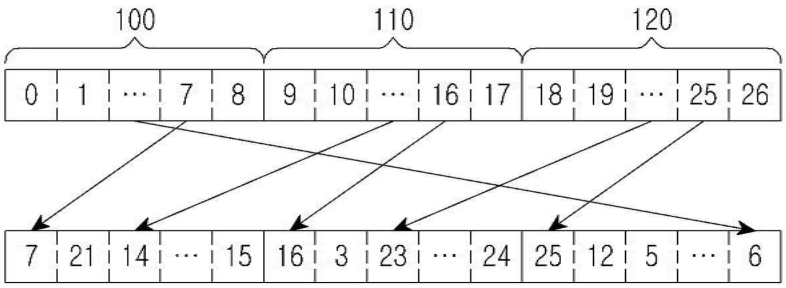
도면의 간단한 설명

<1> 도 1a 및 1b는 본 발명의 실시예에 따른 인터리빙 수행 및 S 행렬을 나타낸 도면

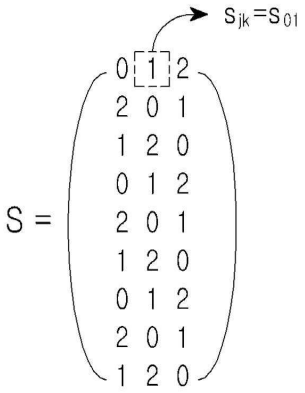
<2> 도 2는 본 발명의 실시예에 따른 S 행렬을 생성하는 과정을 도시한 흐름도

도면

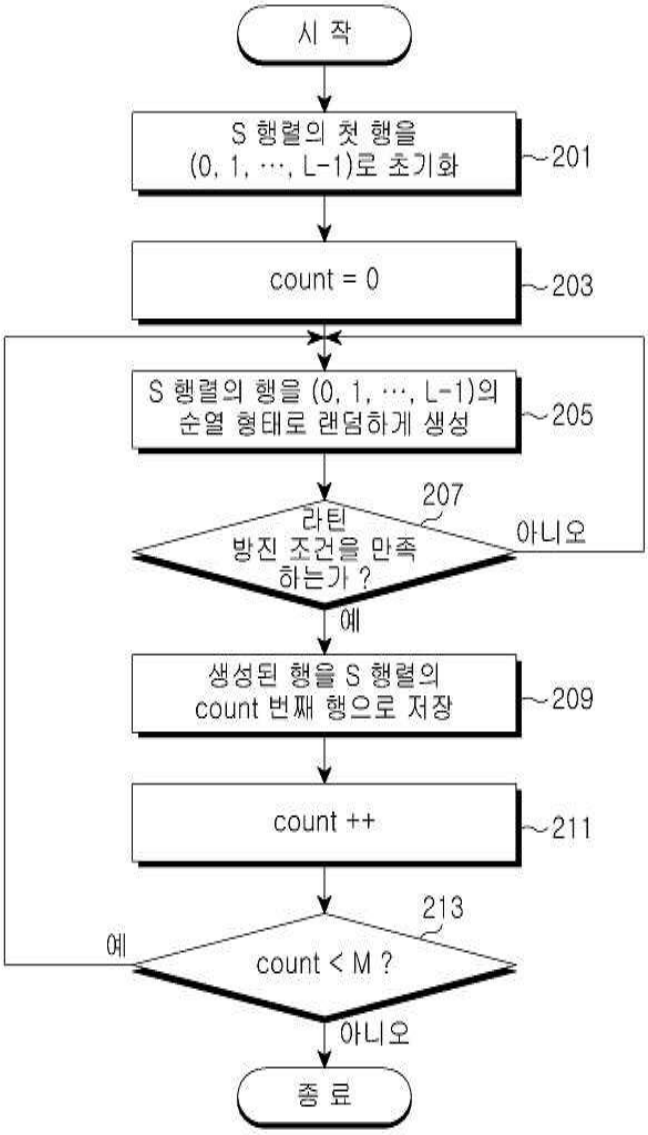
도면1a



도면1b


$S_{jk}=S_{01}$
S =
0 1 2
2 0 1
1 2 0
0 1 2
2 0 1
1 2 0
0 1 2
2 0 1
1 2 0

도면2


시 작
S 행렬의 첫 행을 (0, 1, ···, L-1)로 초기화
201
count = 0
203
S 행렬의 행을 (0, 1, ···, L-1)의 순열 형태로 랜덤하게 생성
205
207
라틴 방진 조건을 만족하는가 ?
아니오
예
생성된 행을 S 행렬의 count 번째 행으로 저장
209
count ++
211
213
count < M ?
예
아니오
종 료