(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 559 637**

(51) Int. CI.:

H03M 13/11 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **05.02.2010 E 10710590 (0)**

(97) Fecha y número de publicación de la concesión europea: **16.12.2015 EP 2395667**

(54) Título: **Codificación LDPC cuasi-cíclica**

(30) Prioridad:

06.02.2009 ES 200900343
20.01.2010 ES 201030066

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
15.02.2016

(73) Titular/es:

MARVELL HISPANIA S.L. (100.0%)
Plaza de Pablo Ruiz Picasso 1 Torre Picasso, 38th Floor
28020 Madrid, ES

(72) Inventor/es:

BLASCO CLARET, JORGE VICENTE;
IRANZO MOLINERO, SALVADOR y
BADENES CORELLA, AGUSTÍN

(74) Agente/Representante:

MILTENYI, Peter

DESCRIPCIÓN

Codificación LDPC cuasi-cíclica

La invención presentada en este punto se aplica al campo de la transmisión de datos, y más específicamente, a la comunicación de datos a través de medios con ruido, es decir, medios o canales de comunicación que pueden introducir errores en la comunicación.

Técnica anterior

En entornos de comunicación es común para el medio de comunicación o las señales externas introducir errores de señal. Dichos errores deben detectarse y, si es posible, corregirse en la recepción de modo que los datos corregidos puedan recuperarse. Existen varias maneras para incluir detección y corrección de errores en el estado de la técnica, siendo una de ellas la codificación y decodificación de los datos basándose en códigos de comprobación de paridad de baja densidad para corregir errores.

Los códigos de comprobación de paridad de baja densidad (LDPC) son códigos de corrección de errores que se usan al transmitir sobre canales de transmisión con ruido. Estos códigos introducen una cierta redundancia en el mensaje (se envía un mayor número de bits que en el mensaje original), pero de tal manera que en la recepción es posible detectar si hay errores en el mensaje recibido y corregirlos.

Un código de LDPC es un código cuya matriz de paridad no es muy densa, es decir que la mayoría de sus elementos son ceros. Este tipo de código se publicó por primera vez a principios de 1960, por Robert G. Gallagher "Low Density Parity Check Codes", M.I.T. Press, 1963, y se mostró que tienen características muy cercanas al límite de Shannon conocido (velocidad máxima teórica para transmisión de datos). Sin embargo, con la definición original de los códigos y la tecnología de ese momento, no fue posible una implementación conseguible de complejidad adecuada. Recientemente, gracias a la evolución de los circuitos integrados y a la invención de matrices estructuradas, estos códigos son de nuevo de gran interés.

En el estado de la técnica existen múltiples procedimientos para conseguir codificación y decodificación de errores. Algunos procedimientos son aquellos publicados en las patentes US 7.343.548B2 y US 7.203.897B2, ambas tituladas "Method and Apparatus for Encoding and Decoding Data", cada una de las cuales señala procedimientos para mejorar la protección cuando se enfrentan con errores en transmisión de datos. La invención puede relacionarse también con las normas IEEE802.16e y 802.11n, que presentan codificación y decodificación para reducir errores. En cualquier caso, las patentes y normas mencionadas usan la estructura de doble diagonal, que es conocida en el estado de la técnica, mientras la estructura presentada en este documento es nueva y permite una implementación con mejores características sin aumentar el nivel de complejidad (por lo tanto a coste inferior) de protección frente a errores al comunicar datos a través de medios con ruido. Se conoce en el estado de la técnica que tener columnas con un peso de Hamming igual a o menor que 2 en la matriz de paridad restringe las características de los LDPC. Sin embargo, por razones de complejidad de implementación del codificador, las matrices con una sección de doble diagonal $\mathbf{H}_{b1}$ se han usado en el estado de la técnica. La nueva estructura presentada en este documento, que añade una tercera diagonal a la sección $\mathbf{H}_{b1}$ de la matriz de modelo binario, permite que el número total de columnas en la matriz de paridad con un peso de Hamming menor que o igual a 2 sea inferior, y por lo tanto pueden conseguirse mejores características. Esta tercera diagonal se seleccionó de tal manera que el aumento en la complejidad de implementación del codificador es prácticamente insignificante.

El documento US 2008/0222486 A1 se refiere a procedimientos y aparatos para codificar y decodificar códigos de comprobación de paridad de baja densidad (LDPC). Este documento desvela un aparato y procedimiento novedosos para codificar datos usando un código de comprobación de paridad de baja densidad (LDPC) capaz de representación mediante un grafo bipartito. Para codificar los datos, puede generarse una cadena acumulada de una pluralidad de nodos variables de grado bajo. La cadena acumulada a continuación puede cerrarse para formar un bucle dos veces, una vez usando unos nodos variables de grado bajo y una vez usando un grado variable más alto que es más alto que el nodo variable de grado bajo, donde el nodo variable de grado más alto comprende un borde de cierre no de bucle. En una realización desvelada en este documento, la pluralidad de nodos variables de grado bajo pueden tener la misma permutación en cada borde.

Los documentos anteriormente presentados no interfieren con la novedad ni la superioridad inventiva de la presente invención. Aunque todos ellos están basados en la utilización de la tecnología de LDPC, que es conocida en el estado de la técnica, el procedimiento y dispositivo inventivos de este documento utilizan un tipo de código cuasi-cíclico (Código de Comprobación de Paridad de Baja Densidad Cuasi-Cíclico, o QC-LDPC), y aplica una matriz de paridad con una estructura diferente como el punto central de la invención.

A lo largo de todo este documento, se empleará una nomenclatura específica para diferenciar los elementos utilizados a lo largo de toda la descripción de la invención. Una letra mayúsculas en negrita (por ejemplo, **A**) indica que el elemento es una matriz; una letra minúscula en negrita (por ejemplo, **a**) indica que el elemento es un vector, mientras una letra minúscula no en negrita (por ejemplo, a) indica que el elemento es un valor escalar. Por otro lado, los elementos escalares que comprenden una matriz del tamaño *MxN* se indican en la forma *a(i,j)*, donde la tupla *(i,j)* es la posición de dicho elemento en la matriz, siendo $0 \leq i \leq M-1$ el número de fila y $0 \leq j \leq N-1$ el número de columna. Los

elementos que comprenden un vector de tamaño *M* se indican en la forma de *a(i)*, siendo (i) la posición del elemento en el vector (0≤i≤*M*-1).

También, se usará a lo largo de toda la invención la expresión “rotación cíclica”, que se definirá a continuación. Una rotación cíclica *z* en un vector **a**=|*a*(0),*a*(1),…,*a*(*M*-2), *a*(*M*-1)| consiste en rotar cíclicamente sus elementos hacia la derecha, obteniendo el vector |*a*((*M*-z)%*M*),…,*a*((*M*-z-1)%*M*)| como el resultado, siendo % el operador “módulo”. De la misma manera, una rotación cíclica *z* aplicada sobre una matriz **A**=|**a**(0),..., ,**a**(*N*-1)| opera en sus columnas, obteniendo la matriz |**a**((*N*-z)%*N*),…,**a**((*N*-z-1)%*N*)| como el resultado. Una rotación cíclica puede definirse también en la dirección opuesta (hacia la izquierda) de modo que una rotación cíclica *z* hacia la derecha es equivalente a una rotación cíclica *M*-*z* y *N*-*z* respectivamente para vector y matriz hacia la izquierda.

Descripción de la invención

Es por lo tanto el objeto de la presente invención proporcionar una codificación de datos mejorada que aplique una matriz de paridad a un bloque de datos y que use códigos de LDPC.

Este objeto se resuelve mediante la materia objeto de la reivindicación 1.

Se definen realizaciones preferidas mediante las reivindicaciones dependientes.

Para conseguir los objetivos y evitar las desventajas indicadas en las secciones anteriores, la invención consiste en un procedimiento y dispositivo para comunicar datos a través de un medio con ruido. En concreto, la invención presenta un procedimiento para codificar datos usados en transmisión, su dispositivo de codificación asociado, un procedimiento para decodificar y su dispositivo de decodificación asociado. Este grupo de invenciones componen un único concepto inventivo, que se describirá a continuación. Si el procedimiento o dispositivo se usan en transmisión, los equivalentes deben usarse también en recepción, y viceversa, de modo que los datos enviados puedan recuperarse.

El procedimiento para codificar datos se aplica en transmisión y genera bits de paridad en un bloque de datos de tal manera que se genera una palabra de código de N bits desde una palabra de K bits (N>K) que incluye protección frente a errores. Dicho procedimiento comprende múltiples etapas. En primer lugar se selecciona un factor *b*, que es un número natural entre 1 y *k* de manera que la división de N y K por el factor de b serán números naturales (n=N/b; k=K/b). A continuación se define una matriz de modelo binario $\boldsymbol{H_0}$ =*[***H**$_a$|**H**$_b$*]* de tamaño (n-k) x n como la combinación de una submatriz que corresponde a las posiciones de los bits de datos **H**$_a$ y una submatriz que corresponde a los bits de paridad **H**$_b$, donde dicha segunda submatriz **H**$_b$ =[**h**b_0|**H**b_1] está compuesta de un vector de columna de n-k posiciones **h**$_{b0}$ y una matriz **H**$_{b1}$ que tiene una estructura de triple diagonal, es decir, donde los elementos de las dos diagonales centrales $h_{b1}(i,i)$, $h_{b1}(i+1,i)$ $0 \le i \le n-k-2$ y la diagonal de la última fila $h_{b1}(n-k-1,0)$ son iguales a 1, donde n-k es el número de filas y columnas de la matriz **H**$_b$, y el resto de elementos son iguales a cero. Posteriormente, se genera la matriz compacta **H**$_1$ y a partir de ella, la matriz de paridad **H**. Desde allí, se toma un bloque de datos y se usa la matriz de paridad **H** en el bloque de datos para determinar los bits de paridad que corresponden a dicho bloque. Finalmente, los bits de paridad se transmiten juntos con el bloque de datos.

En una implementación del procedimiento, es posible eliminar uno o más elementos de la palabra de código antes de que se transmitan, reduciendo la redundancia en la transmisión sin dañar seriamente la capacidad de protección frente a errores. Esta técnica se denomina “perforar”. En este caso la palabra transmitida tendrá un número menor de bits que la palabra de código obtenida en el procedimiento inicial.

El dispositivo de codificación de datos comprende medios para almacenar la matriz compacta **H**; obtenida a partir de una matriz de modelo binario $\boldsymbol{H_0}$ = *[***H**$_a$/**H**$_b$*]* formada como la combinación de una submatriz que corresponde a la posición de los bits de datos **H**$_a$ y una submatriz que corresponde a los bits de paridad **H**$_b$, donde dicha segunda submatriz **H**$_b$ = [**h**b_0|**H**b_1] está compuesta de un vector de columna de n-k posiciones **h**$_{b0}$ y una matriz **H**$_{b1}$ que tiene una estructura de triple diagonal, es decir, donde los elementos de las dos diagonales centrales $h_{b1}(i,i)$, $h_{b1}(i+1,i)$ $0 \le i \le n-k-2$ y la diagonal de la última fila $h_{b1}(n-k-1,0)$ son iguales a 1, cuando n-k es el número de filas y columnas de la matriz **H**$_b$, y el resto de los elementos son iguales a cero; y de un microprocesador que toma el bloque de datos, usa la matriz compacta **H**$_1$ para generar la matriz de paridad **H**, aplica la matriz de paridad **H** al bloque de datos para obtener los bits de paridad que corresponden a dicho bloque y añade los bits de paridad al bloque de datos antes de que se transmitan.

En una implementación concreta de este dispositivo, uno o más elementos de la palabra de código se eliminan después de añadir los bits de paridad al bloque de datos pero antes de la transmisión aplicando la técnica de perforación. De esta manera la palabra transmitida tendrá un número menor de bits que la palabra de código generada originalmente.

Por otro lado, el procedimiento de decodificación de datos opera en la recepción y estima cuál es el bloque de datos recibido desde un vector de señal recibido desde el canal. Desde una palabra de código recibida de N bits (que puede tener errores debido al ruido del canal) se obtiene la palabra de datos de K bits que el transmisor desea enviar. Esta toma en primer lugar un vector de señal desde el canal y la matriz de modelo binario $\boldsymbol{H_0}$ = *[***H**$_a$/**H**$_b$*]* que es una combinación de una submatriz que corresponde a la posición de los bits de datos **H**$_a$, y una submatriz que

corresponde a los bits de paridad $\mathbf{H}_b$, donde dicha segunda submatriz $\mathbf{H}_b$ =[$\mathbf{h}b_0$|$\mathbf{H}b_1$] está compuesta de un vector de columna de n-k posiciones $\mathbf{h}_{b0}$ y una matriz $\mathbf{H}_{b1}$ que tiene una estructura de triple diagonal, es decir, donde los elementos de las dos diagonales centrales $\mathbf{h}_{b1}(i,i)$, $\mathbf{h}_{b1}$ $(i+1,i)$ $0 \leq i \leq n-k-2$ y la diagonal de la última fila $\mathbf{h}_{b1}$: (n-k-1,0) son iguales a 1, donde n-k es el número de filas y columnas de la matriz $\mathbf{H}_b$, y el resto de los elementos son iguales a cero. Posteriormente, se genera la matriz compacta $\mathbf{H}_1$ y a partir de ella, la matriz de paridad **H**, y finalmente el bloque de datos se estima a partir de la señal de vector recibida y la matriz de paridad **H**.

Si la técnica de perforación se usó en la transmisión, los datos perdidos deben recuperarse en la recepción. En este caso, se inserta un valor de indicador en las posiciones eliminadas en la transmisión por la técnica de perforación antes de hacer la estimación del bloque de datos a partir del vector de señal recibido y de la matriz de paridad.

El dispositivo de decodificación de datos comprende medios para almacenar la matriz compacta $\mathbf{H}_1$ que se obtiene a partir de la matriz de modelo binario $\mathbf{H}_0$ = *[*$\mathbf{H}_a$/$\mathbf{H}_b$*]* que es una combinación de una submatriz que corresponde a la posición de los bits de datos $\mathbf{H}_a$ y una submatriz que corresponde a los bits de paridad $\mathbf{H}_b$, donde dicha segunda submatriz $\mathbf{H}_b$ =[$\mathbf{h}b_0$|$\mathbf{H}b_1$] está compuesta de un vector de columna de n-k posiciones $\mathbf{h}_{b0}$ y una estructura de triple diagonal $\mathbf{H}_{b1}$, es decir, una en la que los elementos de las dos diagonales centrales $\mathbf{h}_{b1}(i, i)$, $\mathbf{h}_{b1}(i+1,i)$ $0 \leq i \leq n-k-2$ y la diagonal de la última fila $\mathbf{h}_{b1}$(n-k-1,0) son iguales a 1 donde n-k es el número de filas y columnas de la matriz $\mathbf{H}_b$, y el resto de los elementos son iguales a cero; un microprocesador que genera la matriz de paridad **H** a partir de la matriz compacta $\mathbf{H}_1$, aplica dicha matriz de paridad **H** al vector de señal recibido, y estima el bloque de datos recibido.

Si el dispositivo de transmisión usó la técnica de perforación, los datos perdidos deben recuperarse antes de que se haga la corrección de errores. Por lo tanto, en esta implementación y antes de aplicar la matriz de paridad **H** al vector de señal recibido, se inserta un valor de indicador en las posiciones eliminadas en la transmisión mediante la técnica de perforación.

En una implementación es posible usar una de las siguientes matrices compactas $\mathbf{H}_1$ para obtener palabras de código de 336 bits con una tasa de codificación de 1/2. La matriz:

13	-1	7	-1	-1	1	-1	1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	11	-1	-1	-1	4	-1	4	6	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	13	-1	-1	-1	-1	11	-1	10	-1	9	13	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	13	-1	-1	6	-1	10	-1	5	-1	-1	4	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	8	8	2	11	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	3	-1	-1	-1	-1	-1	-1	1	-1	4	1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	4	2	-1	2	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
2	-1	13	-1	4	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	11	-1	-1	-1	-1	-1	-1	6	4	11	12	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	-1	10	-1	-1	-1	-1	1	-1	13	13	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	-1	0	-1	-1	0	2	2	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
-1	-1	-1	-1	-1	-1	-1	-1	1	2	11	2	4	12	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

o de lo contrario esta matriz:

-1	-1	-1	6	-1	-1	9	6	-1	-1	2	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	0	-1	-1	-1	3	-1	12	1	-1	-1	3	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	9	11	-1	-1	13	-1	-1	2	12	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
1	-1	-1	11	-1	-1	7	-1	-1	-1	11	-1	-1	1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	-1	4	8	-1	-1	-1	-1	-1	2	5	4	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	3	0	-1	-1	8	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	0	6	-1	-1	-1	-1	5	13	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	9	-1	-1	-1	3	-1	-1	3	1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
9	0	13	-1	-1	12	-1	-1	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	5	-1	-1	1	4	-1	-1	5	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	-1	-1	8	-1	-1	8	-1	-1	9	0	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
10	11	-1	-1	-1	3	-1	-1	0	-1	-1	-1	4	8	-1	-1	-1	-1	-1	-1	-1	-1	-1

En otra implementación es posible usar una de las siguientes matrices compactas $\mathbf{H}_1$ para obtener palabras de código de 1920 bits con una tasa de codificación de 1/2. La matriz:

-1	52	-1	64	-1	-1	60	-1	-1	-1	-1	1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	79	-1	-1	79	-1	78	51	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	75	29	72	8	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	52	16	63	-1	-1	65	-1	-1	-1	-1	-1	40	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	24	-1	-1	47	1	39	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
52	-1	-1	-1	-1	-1	-1	-1	53	79	48	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	0	-1	-1	72	-1	67	57	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	7	-1	-1	-1	2	50	-1	-1	-1	-1	15	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
15	-1	19	-1	-1	-1	-1	-1	75	51	43	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
72	-1	-1	-1	38	-1	-1	-1	69	-1	62	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	19	-1	41	-1	-1	1	41	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
41	-1	17	-1	-1	-1	-1	-1	15	-1	30	-1	40	6	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

o de lo contrario esta matriz:

27	-1	-1	-1	55	19	-1	30	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	1	-1	70	-1	47	-1	62	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	41	-1	-1	-1	44	-1	-1	59	60	25	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
16	77	-1	-1	-1	5	-1	48	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	45	-1	27	-1	46	19	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	63	-1	-1	-1	55	-1	-1	-1	46	26	10	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	-1	42	-1	21	-1	58	-1	41	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	-1	78	0	-1	7	52	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	29	9	-1	-1	-1	37	-1	-1	-1	35	21	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	-1	22	72	-1	-1	47	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
35	-1	-1	-1	-1	13	-1	35	-1	70	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	46	28	-1	-1	-1	38	-1	-1	-1	8	-1	10	58	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

En otra implementación es posible usar una de las siguientes matrices compactas H_1 para obtener palabras de código de 8640 bits con una tasa de codificación de 1/2. La matriz:

-1	-1	-1	-1	-1	-1	297	106	328	-1	-1	99	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
290	0	312	-1	32	-1	120	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
183	57	-1	-1	187	68	-1	-1	-1	-1	260	-1	81	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	323	-1	-1	-1	137	354	-1	-1	162	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	228	-1	-1	-1	-1	224	-1	114	-1	245	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
113	98	-1	-1	120	23	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	138	-1	187	45	62	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	142	-1	-1	-1	347	67	-1	-1	-1	46	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
328	269	-1	66	156	96	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
212	184	-1	-1	102	-1	-1	-1	-1	120	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	-1	-1	-1	-1	-1	80	15	-1	329	153	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
207	70	-1	7	235	-1	-1	-1	-1	-1	-1	-1	81	185	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

o de lo contrario la matriz:

-1	34	-1	95	-1	279	-1	-1	-1	-1	248	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	0	-1	-1	-1	-1	134	356	275	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
51	-1	27	-1	-1	-1	-1	-1	22	152	-1	57	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	124	-1	290	-1	281	15	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	340	-1	99	336	-1	-1	1	-1	-1	-1	-1	33	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
163	-1	46	-1	-1	-1	-1	-1	-1	306	-1	86	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	185	-1	24	-1	-1	-1	94	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	223	-1	225	325	-1	-1	-1	-1	-1	297	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
46	-1	314	-1	-1	-1	59	-1	-1	67	-1	120	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	-1	121	-1	-1	-1	-1	161	-1	303	-1	264	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	303	-1	8	-1	185	-1	-1	138	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	-1	312	-1	-1	-1	100	-1	-1	144	-1	307	33	166	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

En otra implementación es posible usar la siguiente matriz compacta $\mathbf{H}_1$ para obtener palabras de código de 1440 bits con una tasa de codificación de 2/3.

49	-1	-1	21	31	-1	57	-1	-1	10	-1	29	2	-1	19	-1	-1	0	-1	-1	-1	-1	-1	-1
-1	7	22	-1	-1	37	-1	32	10	-1	26	-1	-1	59	-1	48	-1	0	0	-1	-1	-1	-1	-1
53	-1	-1	20	50	-1	-1	3	16	-1	49	-1	-1	28	14	-1	-1	-1	0	0	-1	-1	-1	-1
-1	58	23	-1	-1	15	54	-1	-1	5	-1	18	49	-1	-1	13	-1	-1	-1	0	0	-1	-1	-1
55	-1	-1	58	-1	9	-1	26	57	-1	41	-1	31	-1	21	-1	-1	-1	-1	-1	0	0	-1	-1
-1	10	49	-1	59	-1	7	-1	-1	30	-1	18	-1	48	-1	7	59	-1	-1	-1	-1	0	0	-1
48	-1	-1	50	18	-1	-1	11	52	-1	59	-1	-1	37	-1	10	0	-1	-1	-1	-1	-1	0	0
-1	24	16	-1	-1	0	53	-1	-1	41	-1	38	51	-1	59	-1	59	8	-1	-1	-1	1	-1	0

En otra implementación es posible usar la siguiente matriz compacta $\mathbf{H}_1$ para obtener palabras de código de 6480 bits con una tasa de codificación de 2/3:

78	-1	-1	167	237	-1	3	-1	266	-1	-1	1	002	153	-1	-1	212	-1	0	-1	-1	-1	-1	-1
1																							
-1	83	189	-1	-1	68	-1	178	-1	90	205	-1	-1	13	4	-1	-1	0	0	-1	-1	-1	-1	-1
-1	226	147	-1	46	-1	-1	76	-1	116	-1	211	-1	112	-1	118	-1	-1	0	0	-1	-1	-1	-1
92	-1	-1	214	-1	236	241	-1	157	-1	143	-1	214	-1	207	-1	-1	-1	-1	0	0	-1	-1	-1
144	-1	-1	258	264	-1	53	-1	114	-1	172	-1	-1	82	262	-1	62	-1	-1	-1	0	0	-1	-1
-1	153	120	-1	-1	199	-1	126	-1	61	-1	183	15	-1	-1	134	-1	-1	-1	-1	-1	0	0	-1
-1	100	-1	141	-1	36	-1	17	-1	156	-1	124	162	-1	-1	57	0	-1	-1	-1	-1	-1	0	0
196	-1	187	-1	73	-1	80	-1	139	-1	57	-1	-1	236	267	-1	62	256	-1	-1	-1	-1	-1	0

En otra implementación es posible usar la siguiente matriz compacta $\mathbf{H}_1$ para obtener palabras de código de 1152 bits con una tasa de codificación de 5/6:

-1	13	32	47	41	24	-1	25	22	40	1	31	8	15	20	15	42	30	13	3	-1	0	-1	-1
25	46	15	43	45	29	39	47	23	38	39	12	-1	21	-1	38	33	0	0	-1	39	0	0	-1
35	45	45	38	14	16	6	11	-1	18	7	41	35	17	32	45	41	-1	18	17	0	-1	0	0
9	32	6	22	26	31	9	8	22	32	40	4	18	40	36	-1	-1	23	31	41	39	20	-1	0

En otra implementación es posible usar la siguiente matriz compacta $\mathbf{H}_1$ para obtener palabras de código de 5184 bits con una tasa de codificación de 5/6:

-1	47	146	203	184	112	-1	116	103	181	3	140	38	68	91	70	191	138	62	14	-1	0	-1	-1
117	203	67	194	206	133	174	212	104	171	176	56	-1	96	-1	167	149	4	1	-1	177	0	0	-1
153	206	198	173	55	72	28	53	-1	82	34	186	161	80	144	204	187	-1	84	77	0	-1	0	0
44	147	27	83	118	130	41	38	100	146	183	19	85	180	163	-1	-1	106	140	185	177	94	-1	0

Una implementación en la que se usa la técnica de perforación se inicia con una palabra de código de 1152 bits y tasa de codificación de 5/6 y aplica el siguiente patrón de perforación:

$$pp_{1152}^{(16/18)} = [\underbrace{1\ 1\ \cdots\ 1}_{720}\ \underbrace{0\ 0\ \cdots\ 0}_{36}\ \underbrace{1\ 1\ \cdots\ 1}_{360}\ \underbrace{0\ 0\ \cdots\ 0}_{36}]$$

para obtener una palabra de código de 1080 bits y tasa de codificación de 16/18.

Otra implementación en la que se usa la técnica de perforación se inicia con una palabra de código con 5184 bits y una tasa de codificación de 5/6 y aplica el siguiente patrón de perforación

$$pp_{5184}^{(16/18)} = [\underbrace{1\ 1\ \cdots\ 1}_{3240}\ \underbrace{0\ 0\ \cdots\ 0}_{162}\ \underbrace{1\ 1\ \cdots\ 1}_{972}\ \underbrace{0\ 0\ \cdots\ 0}_{162}\ \underbrace{1\ 1\ \cdots\ 1}_{648}]$$

para obtener una palabra de código de 4860 bits y tasa de codificación de 16/18.

Otra implementación en la que se usa la técnica de perforación se inicia con una palabra de código con 1152 bits y una tasa de codificación de 5/6 y aplica el siguiente patrón de perforación

$$pp_{1152}^{(20/21)} = [\underbrace{1\ 1\ \cdots\ 1}_{720}\ \underbrace{0\ 0\ \cdots\ 0}_{48}\ \underbrace{1\ 1\ \cdots\ 1}_{240}\ \underbrace{0\ 0\ \cdots\ 0}_{96}\ \underbrace{1\ 1\ \cdots\ 1}_{48}]$$

para obtener una palabra de código de 1080 bits y tasa de codificación de 20/21.

Y por último, una implementación final en la que se usa la técnica de perforación se inicia con una palabra de código de 5184 bits y tasa de codificación de 5/6 y aplica el siguiente patrón de perforación

$$pp_{5184}^{(20/21)} = [\underbrace{0\ 0\ \cdots\ 0}_{216}\ \underbrace{1\ 1\ \cdots\ 1}_{4320}\ \underbrace{0\ 0\ \cdots\ 0}_{432}\ \underbrace{1\ 1\ \cdots\ 1}_{216}]$$

para obtener una palabra de código de 4536 bits y tasa de codificación de 20/21.

A continuación, para facilitar un mayor entendimiento de este documento descriptivo, se proporciona la descripción, a modo de ilustración pero no de limitación, de una implementación de ejemplo de la invención. Esta forma una parte integral del mismo documento.

Breve descripción de las figuras

La **Figura 1**.- muestra el diagrama de bloques del codificador en el transmisor.

La **Figura 2**.- muestra el diagrama de bloques del decodificador en el transmisor.

La **Figura 3**.- muestra el grafo de dos partes de la matriz **H** del ejemplo.

La **Figura 4**.- muestra el diagrama de flujo de la construcción de un código de LDPC estructurado.

Descripción de un ejemplo de la implementación de la invención

A continuación se presenta la descripción de un ejemplo de implementación de la invención, que hace referencia a la numeración adoptada en las figuras.

El problema que el procedimiento de la invención pretende resolver, desde un punto de vista teórico, consiste en el éxito al optimizar la corrección de errores en comunicación de datos usando implementaciones de hardware de bajo coste y códigos de LDPC.

Un código de LDPC es un código lineal que opera en bloques de datos. El código se define por su matriz de paridad **H**. En este ejemplo de implementación los códigos son códigos binarios, pero es posible generalizar la invención a códigos sobre cualquier campo de Galois *GF(q)*, donde *q* es ≥ 2.

En transmisión se tienen bloques de datos compuestos de K bits. Dicho bloque e datos se designa como **u** = [*u*(0),*u*(1),…,*u*(*K*-1)]. Después de aplicar el procedimiento de la invención, se genera una palabra de un código lineal **v** = [*v*(0),*v*(1),…,*v*(*N*-1)] con N bits (donde N<K). Dicho código se genera a través del producto **v=uG**, donde **G** es una matriz binaria ***KxN***, generadora del código de LDPC. El conjunto de posibles códigos generados se denomina conjunto **C**, y la tasa de codificación del código será ***R=K/N.***

Por lo tanto, es posible definir un código **C** de codificación R como el conjunto de vectores $\mathbf{v} \in \mathbf{C}$ generados por todos los posibles 2^K vectores binarios aplicándoles la matriz generadora **G**. Una definición equivalente sería que C es el espacio vectorial de tamaño ***N*** incluido en la base compuesta de las ***K*** filas de la matriz **G**. Otra forma alternativa de definir el código **C** es a través de su matriz de paridad **H**, que es la forma más usada en el estado de la técnica. Esta matriz, en tamaño (N-K)xN, tiene como filas la base del espacio dual **C**, y por lo tanto $\mathbf{GH}^T = \mathbf{0}$. Cualquier vector del código satisface

$$\mathbf{vH}^T = \mathbf{0}$$

(donde “T” es el operador de transposición).

En el momento que se utilizan estos códigos desde un punto de vista práctico, se prefiere considerar los códigos como códigos sistemáticos, es decir, aquellos en que los bits de la palabra de código están entre los bits de datos. Sin perder generalidad, este ejemplo se centra en el caso donde **v=[u|p]**, donde **p=[*p*(0), *p*(1), … *p*(*N*-*K*-1)]** es un vector compuesto de los bits de paridad, **u** el bloque de datos que se ha de transmitir y **v** la palabra de código realmente transmitida (después de incluir el código de LDPC).

Se muestra a continuación un ejemplo de implementación en el que puede observarse la relación entre la matriz de paridad y una palabra de código. En este ejemplo, el código tiene una tasa de codificación de *R*=1/2 y se define mediante la siguiente matriz de paridad:

$$H = \left[\begin{array}{ccccc|ccccc} 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \end{array}\right]$$

En esta matriz, la sección izquierda corresponde a los *K*=5 bits de datos, mientras la sección derecha corresponde a los *N*-*K*=5 bits de paridad. Aplicando la ecuación $\mathbf{vH}^T = \mathbf{0}$ a la matriz **H** se obtiene el siguiente sistema de ecuaciones:

$$\begin{cases} u(0)+u(1)+u(3)+p(0)+p(3)+p(4)=0 \\ u(0)+u(1)+u(2)+p(0)+p(2)+p(4)=0 \\ u(0)+u(1)+u(2)+u(4)+p(1)+p(3)=0 \\ u(2)+u(3)+u(4)+p(0)+p(1)+p(2)=0 \\ u(3)+u(4)+p(1)+p(2)+p(3)+p(4)=0 \end{cases}$$

Un código de LDPC puede representarse también en forma gráfica con un grafo de dos partes denominado un grafo de Tanner. En un grafo de Tanner los vértices o nodos se clasifican en dos grupos o conjuntos separados: los "nodos variables", que representan los bits de la palabra de código, y los "nodos de comprobación", que representan las relaciones de paridad. Entre ambos conjuntos de nodos se encuentran los posibles bordes que definen la ecuación de paridad. En el caso del código definido en el ejemplo anterior, este grafo correspondiente se representa en la Figura 3, pueden encontrarse 10 nodos (14) variables y 5 nodos (16) de comprobación, unidos por múltiples bordes (15). Cada nodo de comprobación está unido a través de bordes a 6 nodos variables, como se representa en el sistema de ecuaciones anterior. Puede observarse que el grafo tiene tantos nodos de control y variables como la matriz de paridad correspondiente tiene filas y columnas, y que se encuentra un borde entre el nodo de comprobación *i* y el nodo variable *j* cuando el elemento ***h(i,j)*** de la matriz, es decir, el localizado en la fila ***i*=0**,..., *N*-*K*-1 y en la columna *j*=0,...,*N*-1 no es cero.

Por otro lado, pueden definirse ciclos en códigos de LDPC, donde se define un ciclo de longitud 2c como la trayectoria de 2c bordes de longitud que procesan c nodos de comprobación y c nodos variables en el grafo de Tanner que representa el código antes de volver al mismo nodo de comienzo. Para optimizar las características del código, puede demostrarse que es de vital importancia que el número de ciclos cortos sea el mínimo posible. El ciclo de longitud mínima se denomina circunferencia. Es particularmente deseable que la circunferencia sea mayor que 4 para evitar reducir las características de un decodificador iterativo.

En la descripción general, R. Gallagher presentó códigos cuyas matrices de paridad se generaron aleatoriamente. En el estado de la técnica se conoce en general que para obtener buenas características, cerca del límite de Shannon, el tamaño del código debe ser relativamente grande, y como consecuencia de eso, la matriz de paridad debe ser grande. El problema es que las matrices que son grandes y se generan aleatoriamente producen dificultad en la implementación de tanto el codificador como el decodificador. Una manera de evitar esta dificultad es usar matrices con una estructura regular. A continuación se presentan las etapas necesarias para generar una estructura regular:

1. En primer lugar, se genera una matriz de modelo binario $\mathbf{H}_0$ de tamaño *(n-k)*x *n* donde *n*<*N*, *k*<*K* y *R*=*k*/*n*=*K*/*N*. Si el peso de Hamming de las columnas y filas de $\mathbf{H}_0$ es constante, el código generado se denomina LDPC regular. Sin embargo, pueden conseguirse mejores características si la matriz es irregular, es decir, si los pesos de las columnas siguen una distribución estadística dependiente de la tasa de codificación y del canal donde se hará finalmente la transmisión de datos.

2. Una vez que se ha obtenido la matriz de modelo binario $\mathbf{H}_0$, se genera la matriz compacta $\mathbf{H}_1$, sustituyendo cada elemento de $\mathbf{H}_0$ que es igual a "1" con un número entero positivo pseudo-aleatorio $0 \leq x < b$ (donde $b = N/n$) y cada elemento igual a "0" con el valor -1.

3. Para obtener la matriz de paridad **H**, los elementos positivos de $\mathbf{H}_1$ se sustituyen por una submatriz de identidad rotada cíclicamente el número de veces indicado por el valor del elemento positivo de $\mathbf{H}_1$ en cuestión, y los elementos iguales a -1 se sustituyen por una submatriz nula del mismo tamaño. El tamaño de estas submatrices será también *b* x *b*.

El resultado es una matriz de paridad **H** de tamaño (*N*-*K*) x *N* que define un código de LDPC con tasa de codificación *R*=*K*/*N*. El grado de densidad (dispersión) de la matriz dependerá del tamaño de *b*. En general, cuanto mayor es *b*, mejores son las características que se obtienen usando un decodificador iterativo.

Si los ciclos de la matriz generada (grafo de Tanner) resultan ser muy cortos, debería aplicarse la etapa 2 (o incluso la etapa 1 si fuera necesario) para el fin de mejorar estas propiedades.

Para facilitar la implementación del codificador, es necesario generar la matriz de modelo binario $\mathbf{H}_0$ en una forma específica. En primer lugar, dicha matriz se divide en dos partes $\mathbf{H}_0 = \mathbf{[H_a|H_b]}$ donde la submatriz $\mathbf{H}_a$ corresponde a las posiciones de los bits de datos y $\mathbf{H}_b$ a los bits de paridad. La primera submatriz se genera pseudo-aleatoriamente de la manera anteriormente descrita. Sin embargo, la segunda sección normalmente es determinística.

Esta segunda sección $\mathbf{H}_b$, de acuerdo con el estado de la técnica y pretendida para facilitar el diseño de un codificador eficaz, toma una de las siguientes dos formas; la primera forma es

$$\mathbf{H}_b = [\mathbf{h}_{b0} \mid \mathbf{H}_{b1}] = \left[\begin{array}{c|cccccc} h_b(0) & 1 & 0 & & & 0 \\ h_b(1) & 1 & 1 & \ddots & & \\ & 0 & 1 & \ddots & \ddots & \\ & & \ddots & \ddots & 1 & 0 \\ & & & \ddots & 1 & 1 \\ h_b(n-k-1) & 0 & & & 0 & 1 \end{array}\right]$$

donde la primera sección es un vector de columna pseudo-aleatorio que tiene un peso de Hamming mayor que 2 y $\mathbf{H}_{b1}$ es una matriz de doble diagonal cuyos elementos $h_{b1}(i,j)$ son iguales a “1” cuando $i=j,i=j+1$ e iguales a “0” en las posiciones restantes.

La segunda manera de generar $\mathbf{H}_b$ es totalmente doble diagonal

$$\mathbf{H}_b = \left[\begin{array}{cccccc} 1 & 0 & & & & 0 \\ 1 & 1 & \ddots & & & \\ 0 & 1 & \ddots & & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & \ddots & 1 & 1 & 0 \\ 0 & & & 0 & 1 & 1 \end{array}\right]$$

donde los elementos de la submatriz $h_b(i,j)$ son iguales a “1” cuando $i=j,i=j+1$ e iguales a cero en las posiciones restantes.

Una vez que se ha generado esta estructura de matriz base, se genera la matriz compacta $\mathbf{H}_1$, en la forma anteriormente descrita con la única excepción de que en la parte de doble diagonal de $\mathbf{H}_b$, los “1” se sustituyen por el mismo número entero positivo y los “0” por “-1”. También, la matriz de paridad final se obtiene cambiando los enteros positivos por identidades rotadas cíclicamente y los negativos por una submatriz nula. El procedimiento puede observarse gráficamente en la Figura 4, donde el bloque (17) genera la matriz de modelo binario $\mathbf{H}_0$, el bloque (18) genera la matriz compacta $\mathbf{H}_1$, el bloque (19) decide si los ciclos son lo suficientemente largos, y el procedimiento pasa a generar la matriz de paridad **H** con el bloque 20, o de los contrario si los ciclos son cortos, la matriz de modelo (21) o de lo contrario la matriz de base (22) se genera de nuevo.

El procedimiento y dispositivo de la invención juntos modifican la estructura de la matriz de paridad conocida en el estado de la técnica para facilitar la implementación final de la codificación y decodificación y mejorar las características. Para esto, la estructura propuesta consiste en que la sección del modelo binario $\mathbf{H}_0$ que corresponde a los bits de paridad tiene la siguiente forma:

$$\mathbf{H}_b = [\mathbf{h}_{b0} \mid \mathbf{H}_{b1}] = \left[\begin{array}{c|cccccc} h_{b0}(0) & 1 & 0 & & & 0 \\ h_{b0}(1) & 1 & 1 & \ddots & & \\ & 0 & 1 & \ddots & \ddots & \\ & & \ddots & \ddots & 1 & 0 \\ & & & \ddots & 1 & 1 \\ h_{b0}(n-k-1) & 1 & & & 0 & 1 \end{array}\right]$$

donde la estructura $\mathbf{H}_{b1}$ es triple diagonal, es decir, además de los elementos de las dos diagonales centrales $h_{b1}(l,i)$, $h_{b1}(i+1,i)$, el elemento de la diagonal de la última línea h_{b1} (n-k-1,0) es también igual a “1”. La matriz compacta se genera de la manera anteriormente descrita excepto que el elemento de la última fila igual a “1” se sustituye por un entero estrictamente positivo $w \geq 0$.

Una matriz de modelo binario $\mathbf{H}_0$ para r=1/2 con n=24 y k=12 puede tener la siguiente estructura:

1	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	1	0	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	1	0	1	0	0	1	0	0	1	1	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0
0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
0	1	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	1	1	0	0
0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1	1	0
0	0	1	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1

Una matriz compacta $\mathbf{H}_1$ obtenida a partir de la anterior por un tamaño de bloque de N=336 y teniendo por lo tanto un factor de expansión b=14 sería la siguiente:

13	-1	7	-1	-1	1	-1	1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	11	-1	-1	-1	4	-1	4	6	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	13	-1	-1	-1	-1	11	-1	10	-1	9	13	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	13	-1	-1	6	-1	10	-1	5	-1	-1	4	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	8	8	2	11	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	3	-1	-1	-1	-1	-1	-1	1	-1	4	1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	4	2	-1	2	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
2	-1	13	-1	4	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	11	-1	-1	-1	-1	-1	-1	6	4	11	12	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	-1	-1	10	-1	-1	-1	-1	1	-1	13	13	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	0	-1	-1	0	2	2	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	-1	-1	-1	-1	-1	-1	-1	1	2	11	2	4	12	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

o como se prefiera, puede usarse la siguiente matriz como alternativa:

-1	-1	-1	6	-1	-1	9	6	1	-1	2	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	0	-1	-1	-1	3	-1	12	1	-1	-1	3	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	9	11	-1	-1	13	-1	-1	2	12	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
1	-1	-1	11	-1	-1	7	-1	-1	-1	11	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	4	8	-1	-1	-1	-1	-1	2	5	4	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	3	0	-1	-1	8	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	-1	0	6	-1	-1	-1	-1	5	13	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	9	-1	-1	-1	3	-1	-1	3	1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
9	0	13	-1	-1	12	-1	-1	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	5	-1	-1	1	4	-1	-1	5	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	-1	8	-1	-1	8	-1	-1	9	9	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
10	11	-1	-1	-1	3	-1	-1	0	-1	-1	-1	4	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

para un tamaño de bloque diferente puede definirse una matriz compacta diferente que puede obtenerse a partir de la misma matriz de modelo binario o a partir de otra diferente.

Para obtener palabras de código de 1920 bits con una tasa de codificación de 1/2, puede usarse la siguiente matriz:

-1	52	-1	64	-1	-1	60	-1	-1	-1	-1	1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	79	-1	-1	79	-1	78	51	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	75	29	72	8	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	52	16	63	-1	-1	65	-1	-1	-1	-1	-1	40	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	24	-1	-1	47	1	39	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
52	-1	-1	-1	-1	-1	-1	-1	53	79	48	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
1	0	1	1	72	1	67	57	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	1
-1	7	-1	-1	-1	2	50	-1	-1	-1	-1	15	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
15	-1	19	-1	-1	-1	-1	-1	75	51	43	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
72	-1	-1	-1	38	-1	-1	-1	69	-1	62	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	19	-1	41	-1	-1	1	41	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
41	-1	17	-1	-1	-1	-1	-1	15	-1	30	-1	40	6	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

o de lo contrario, preferentemente, la matriz:

27	-1	-1	-1	55	19	-1	30	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	1	-1	70	-1	47	-1	62	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	41	-1	-1	-1	44	-1	-1	59	60	25	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
16	77	-1	-1	-1	5	-1	48	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	45	-1	27	-1	46	19	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	63	-1	-1	-1	55	-1	-1	-1	48	26	10	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	-1	42	-1	21	-1	58	-1	41	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	-1	78	0	-1	7	52	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	29	9	-1	-1	-1	37	-1	-1	1	35	21	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	-1	22	72	-1	-1	47	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
35	-1	-1	-1	-1	13	-1	35	-1	70	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	46	28	-1	-1	-1	38	-1	-1	-1	8	-1	10	58	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

Una matriz compacta para N=8640 bits con factor de expansión 360 obtenida a partir de una matriz de modelo binario diferente sería la siguiente:

-1	-1	-1	-1	1	-1	297	106	328	-1	-1	99	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
290	0	312	-1	32	-1	120	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
183	57	-1	-1	197	68	-1	-1	-1	-1	260	-1	81	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	323	-1	-1	-1	137	354	-1	-1	162	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	220	-1	-1	-1	-1	224	-1	114	-1	245	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
113	98	-1	-1	120	23	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	138	-1	187	45	62	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	142	-1	-1	-1	307	67	-1	-1	-1	46	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
329	265	-1	66	156	96	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
212	184	-1	-1	102	-1	-1	-1	-1	120	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	-1	-1	-1	-1	-1	80	15	-1	329	153	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
207	70	-1	7	235	-1	-1	-1	-1	-1	-1	-1	81	185	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

Otra alternativa preferente para obtener palabras de código de 8640 bits con una tasa de codificación de 1/2 es la matriz:

-1	34	-1	95	-1	279	-1	-1	-1	-1	248	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	0	-1	-1	-1	-1	134	356	275	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
51	-1	27	-1	-1	-1	-1	-1	22	152	-1	57	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	124	-1	290	-1	281	15	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	340	-1	99	336	-1	-1	1	-1	-1	-1	-1	33	-1	-1	-1	0	0	-1	-1	-1	-1	-1
163	-1	46	-1	-1	-1	-1	-1	-1	306	-1	86	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	185	-1	24	-1	-1	-1	94	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	223	-1	225	325	-1	-1	-1	-1	-1	297	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
46	-1	314	-1	-1	-1	59	-1	-1	67	-1	120	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	121	-1	-1	-1	-1	161	-1	303	-1	264	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	303	-1	8	-1	185	-1	-1	138	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
-1	-1	312	-1	-1	-1	100	-1	-1	144	-1	307	33	166	-1	-1	-1	-1	-1	-1	-1	-1	-1

La etapa desde la matriz compacta al modelo binario es unívoca, pero para la etapa opuesta, no lo es; es decir, pueden obtenerse diferentes matrices compactas a partir de una matriz de modelo binario. La matriz binaria se introduce como una etapa que facilita la descripción de la invención. Es posible desarrollar una matriz compacta directamente sin pasar a través de la matriz de modelo binario. En ese caso, para esa matriz compacta, puede obtenerse su matriz de modelo binario correspondiente que, si es de triple diagonal, estará de acuerdo con el procedimiento de la invención.

Los códigos de corrección de errores pueden “perforarse”, en el que la técnica para perforar consiste en eliminar elementos de la palabra de código de modo que no se transmitirán. En lugar de transmitir una palabra de código v=[$v(0),v(1),\ldots,v(N\text{-}1)$], se transmitirá una palabra w=[$w(0),w(1),\ldots,w(M\text{-}1)$] donde M<N. La perforación debe hacerse de una manera controlada, de modo que se transmita la redundancia necesaria para que el decodificador del receptor pueda evaluar los datos transmitidos. Esta perforación se aplica a tanto los bits de datos como los bits de paridad. Puede definirse un patrón de perforación como la secuencia de bits que se ha de transmitir como perforados, donde dicho patrón de bits pude ser periódico o aperiódico. En caso de que no haya regularidad, dicho patrón puede describirse con un vector pp de N posiciones, indicando con un “1” los bits a transmitirse y con un “0” los bits a eliminarse (perforarse). Gracias a la técnica de perforación, la comunicación de datos puede amplificarse, puesto que se envía menos redundancia. Si el peso de Hamming del patrón pp es M, la tasa de errores del sistema total es $R=K/M$.

Por ejemplo, si se tiene un código con R=5/6 y el tamaño de bloque N=5184 y se desea realizar una perforación para aumentar la tasa a R=16/18, puede usarse el siguiente patrón de perforación, que producirá un bloque con N=4860 bits:

$$pp_{5184}^{(16/18)} = [\underbrace{1\ 1\ \cdots\ 1}_{3240}\ \underbrace{0\ 0\ \cdots\ 0}_{\text{illegible}}\ \underbrace{1\ 1\ \cdots\ 1}_{972}\ \underbrace{0\ 0\ \cdots\ 0}_{162}\ \underbrace{1\ 1\ \cdots\ 1}_{\text{illegible}}]$$

Para conseguir la implementación de códigos de LDPC, se usa un dispositivo electrónico, ya sea un programa ejecutado en un microprocesador o una implementación de hardware FPGA o ASIC. Dicho dispositivo recibe un bloque de datos, calcula los bits de paridad, los concatena con los bits de información y los envía a la siguiente fase del transmisor para modularse adecuadamente y transmitirse a través del canal correspondiente. El cálculo de los bits de paridad puede hacerse por medio del producto de la matriz generadora ***G*** o la solución del sistema de ecuaciones anteriormente presentado.

El decodificador de códigos de LDPC normalmente está basado en un decodificador iterativo. Un posible decodificador, entre varias opciones de la técnica, consiste en un estimador que, tras recibir la palabra que corresponde al código transmitido r=v+z, donde Z es ruido de canal aditivo, realiza una estimación de ruido $\hat{\mathbf{z}}$ de manera que $(r-\hat{z})H^T=0$. En el caso donde el sistema hace uso de la técnica de perforación, antes del decodificador habrá una unidad que inserta un indicador en las posiciones perforadas. Este indicador sirve para dirigir al decodificador para estimar el valor apropiado en estas posiciones.

La Figura 1 muestra el diagrama de bloques de un codificador típico, donde (1) es el bloque de datos a transmitirse **u**=[$u(0),u(1),\ldots,u(K\text{-}1)$], (2) es una memoria que contiene la representación de la matriz de paridad **H** o la matriz generadora **G**, (3) es dicha matriz, (4) es el bloque que realiza el algoritmo de codificación, (5) es la palabra de código lineal obtenida desde la codificación **v** = [$v(0),v(1),\ldots,v(N\text{-}1)$], (6) es el bloque que hace la perforación, y (7) es la palabra obtenida después de la perforación **w** = [$w(0),w(1),\ldots,w(M\text{-}1)$].

La Figura 2 muestra el diagrama de bloques para un decodificador, donde (8) representa la señal recibida desde el canal **s**=[$s_n,s_1,\ldots,s_{M\text{-}1}$], que será similar a la palabra obtenida después de perforar pero después de verse afectada por el ruido de canal, (9) representa el bloque que realiza la “des-perforación”, obteniendo una palabra (10) **r**=[$r(()),r(1),\ldots,r(N\text{-}1)$] con el número de bits en el código de LDPC. (11) es la memoria que contiene cualquiera de la matriz

de paridad o la matriz generadora en el receptor y que la transmite (12) al bloque que realiza el algoritmo de decodificador (13). La salida de este bloque serán los datos reconstruidos (14) $\hat{u} = [\hat{u}(0), \hat{u}(1), \ldots, \hat{u}(K-1)]$.

REIVINDICACIONES

1. Un procedimiento aplicado a codificar datos durante transmisión de señal, en el que la codificación incluye generar bits de paridad en un bloque de datos de manera que desde una palabra de *K* bits se genera una palabra de código de *N* bits, comprendiendo el procedimiento:

seleccionar un factor *b*, donde $1 \leq b \leq K$, donde *n* y *k* son enteros positivos, y donde $n = N/b$ y $k = K/b$;

definir una matriz de modelo binario $H_0=[H_a|H_b]$ de tamaño (*n*-*k*)x *n* como una combinación de una primera submatriz H_a que corresponde a los bits de datos y una segunda submatriz H_b que corresponde a los bits de paridad, donde la segunda submatriz $H_b=[h_{b0}|H_{b1}]$ está compuesta de un vector de columna h_{b0} que tiene *n*-*k* posiciones y una triple diagonal estructurada H_{b1}, en el que los elementos de las dos diagonales centrales $h_{b1}(i,i)$, $h_{b1}(i+1,i)$, $0 \leq i \leq n-k-2$ y el primer elemento $h_{b1}(n-k-1,0)$ de la última fila son iguales a 1, donde *n*-*k* es el número de filas y columnas de la submatriz H_b, y de los elementos restantes de la triple diagonal estructurada H_{b1} son iguales a cero;

generar una matriz compacta H_1 desde la matriz de modelo binario H_0 sustituyendo cada uno de los 1 en la matriz de modelo binario H_0 con un número entero y los 0 en dicha matriz de modelo binario H_0 con -1, en el que los números enteros son mayores que o iguales a 0 y menores que *b*;

generar una matriz de paridad *H* de un código de comprobación de paridad de baja densidad cuasi-cíclico, LDPC, desde la matriz compacta H_1;

aplicar la matriz de paridad *H* al bloque de datos para determinar los bits de paridad para el bloque de datos para generar la palabra de código de LDPC; y

transmitir la palabra de código de LDPC mediante un transmisor a través de un canal desde un primer dispositivo de comunicación a un segundo dispositivo de comunicación.

2. El procedimiento de la reivindicación 1, en el que los enteros positivos usados para generar la matriz compacta son enteros positivos pseudo-aleatorios mayores que, o iguales a, 0 y menores que *b*.

3. El procedimiento de la reivindicación 1, en el que la generación de la matriz de paridad *H* comprende:

sustituir los elementos positivos de la matriz compacta H_1 con una sub-matriz de identidad rotada cíclicamente un número de veces indicado por el valor del elemento positivo; y

sustituir los elementos de la matriz compacta H_1 que son iguales a -1 con una sub-matriz nula.

4. El procedimiento de la reivindicación 1, en el que la matriz de paridad *H* define el código de LDPC con una tasa de codificación de R=K/N.

5. El procedimiento de la reivindicación 1, que comprende adicionalmente perforar la palabra de código mediante un dispositivo de perforación para eliminar los bits de la palabra de código antes de que se transmita la palabra de código, en el que la perforación incluye eliminar al menos un bit de datos y al menos un bit de paridad de la palabra de código.

6. El procedimiento de una cualquiera de las reivindicación 1 a 5, que comprende adicionalmente modular y transmitir la palabra de código con una fase del transmisor a través del canal, o

en el que la matriz compacta H_1 se genera mediante un procesador, o

en el que la matriz compacta H_1, se genera mediante uno de un circuito integrado y un campo de matriz de puertas programables.

7. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 336 bits con una tasa de codificación de ½ es:

13	-1	7	-1	-1	1	-1	1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
5	-1	-1	-1	11	-1	-1	-1	4	-1	4	6	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	13	-1	-1	-1	-1	11	-1	10	-1	9	13	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	13	-1	-1	6	-1	10	-1	5	-1	-1	4	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	8	8	2	11	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	3	-1	-1	-1	-1	-1	-1	1	-1	4	1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	4	2	-1	2	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
2	-1	13	-1	4	9	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	11	-1	-1	-1	-1	-1	-1	6	4	11	12	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	-1	10	-1	-1	-1	-1	1	-1	13	13	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	-1	0	-1	-1	0	2	2	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
-1	-1	-1	-1	-1	-1	-1	-1	1	2	11	2	4	12	-1	-1	-1	-1	-1	-1	-1	-1	-1

8. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 1920 bits con una tasa de codificacion de ½ es:

-1	52	-1	64	-1	-1	60	-1	-1	-1	-1	1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
10	-1	-1	-1	-1	79	-1	-1	79	-1	78	51	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
9	-1	-1	-1	-1	-1	-1	75	29	72	8	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	52	16	63	-1	-1	65	-1	-1	-1	-1	-1	40	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	24	-1	-1	47	1	39	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
52	-1	-1	-1	-1	-1	-1	-1	53	79	48	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	0	-1	-1	72	-1	67	57	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	7	-1	-1	-1	2	50	-1	-1	-1	-1	15	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
15	-1	19	-1	-1	-1	-1	-1	75	51	43	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
72	-1	-1	-1	38	-1	-1	-1	69	-1	62	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	19	-1	41	-1	-1	1	41	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
41	-1	17	-1	-1	-1	-1	-1	15	-1	30	-1	40	6	-1	-1	-1	-1	-1	-1	-1	-1	0

9. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 8640 bits con una tasa de codificacion de ½ es:

-1	-1	-1	-1	-1	-1	297	106	328	-1	-1	99	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
290	0	312	-1	32	-1	120	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
183	57	-1	-1	187	68	-1	-1	-1	-1	260	-1	81	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	323	-1	-1	-1	137	354	-1	-1	162	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	228	-1	-1	-1	-1	224	-1	114	-1	245	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
113	98	-1	-1	120	23	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	138	-1	187	45	62	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	-1	142	-1	-1	-1	347	67	-1	-1	-1	46	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
328	265	-1	66	156	96	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
212	184	-1	-1	102	-1	-1	-1	-1	120	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	-1	-1	-1	-1	-1	-1	80	15	-1	329	153	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
207	70	-1	7	235	-1	-1	-1	-1	-1	-1	-1	81	185	-1	-1	-1	-1	-1	-1	-1	-1	-1

10. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 1440 bits con una tasa de codificacion de 2/3 es:

49	-1	-1	21	31	-1	57	-1	-1	19	-1	29	2	-1	19	-1	-1	0	-1	-1	-1	-1	-1	-1
-1	7	22	-1	-1	37	-1	32	10	-1	26	-1	-1	59	-1	48	-1	0	0	-1	-1	-1	-1	-1
53	-1	-1	20	50	-1	-1	3	16	-1	49	-1	-1	28	14	-1	-1	-1	0	0	-1	-1	-1	-1
-1	58	23	-1	-1	15	54	-1	-1	5	-1	18	49	-1	-1	13	-1	-1	-1	0	0	-1	-1	-1
55	-1	-1	58	-1	9	-1	26	57	-1	41	-1	31	-1	21	-1	-1	-1	-1	-1	0	0	-1	-1
-1	10	49	-1	59	-1	7	-1	-1	30	-1	18	-1	48	-1	7	59	-1	-1	-1	-1	0	0	-1
48	-1	-1	50	18	-1	-1	11	52	-1	59	-1	-1	37	-1	10	0	-1	-1	-1	-1	-1	0	0
-1	24	16	-1	-1	0	53	-1	-1	41	-1	38	51	-1	58	-1	59	8	-1	-1	-1	-1	-1	0

11. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 6480 bits con una tasa de codificacion de 2/3 es:

78	-1	-1	167	237	-1	3	-1	266	-1	-1	102	153	-1	-1	212	-1	0	-1	-1	-1	-1	-1	-1
-1	83	189	-1	-1	68	-1	178	-1	90	205	-1	-1	13	4	-1	-1	0	0	-1	-1	-1	-1	-1
-1	226	147	-1	46	-1	-1	76	-1	116	-1	211	-1	112	-1	118	-1	-1	0	0	-1	-1	-1	-1
92	-1	-1	214	-1	236	241	-1	157	-1	143	-1	214	-1	207	-1	-1	-1	-1	0	0	-1	-1	-1
144	-1	-1	258	264	-1	53	-1	114	-1	172	-1	-1	82	262	-1	62	-1	-1	-1	0	0	-1	-1
-1	153	120	-1	-1	199	-1	126	-1	61	-1	183	15	-1	-1	134	-1	-1	-1	-1	-1	0	0	-1
-1	100	-1	141	-1	36	-1	17	-1	156	-1	124	162	-1	-1	57	0	-1	-1	-1	-1	-1	0	0
196	-1	187	-1	73	-1	80	-1	139	-1	57	-1	-1	236	267	-1	62	256	-1	-1	-1	-1	-1	0

12. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 1152 bits con una tasa de codificacion de 5/6 es:

-1	13	32	47	41	24	-1	25	22	40	1	31	8	15	20	15	42	30	13	3	-1	0	-1	-1
25	46	15	43	45	29	39	47	23	38	39	12	-1	21	-1	38	33	0	0	-1	39	0	0	-1
35	45	45	38	14	16	6	11	-1	18	7	41	35	17	32	45	41	-1	18	17	0	-1	0	0
9	32	6	22	26	31	9	8	22	32	40	4	18	40	36	-1	-1	23	31	41	39	20	-1	0

13. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 5184 bits con una tasa de codificacion de 5/6 es:

-1	47	146	203	184	112	-1	116	103	181	3	140	38	68	91	70	191	138	62	14	-1	0	-1	-1
117	203	67	194	206	133	174	212	104	171	176	56	-1	96	-1	167	149	4	1	-1	177	0	0	-1
153	206	198	173	55	72	28	53	-1	82	34	186	161	80	144	204	187	-1	84	77	0	-1	0	0
44	147	27	83	118	130	41	38	100	146	183	19	85	180	163	-1	-1	106	140	185	177	94	-1	0

14. El procedimiento de la reivindicación 1, en el que uno o más elementos de la palabra de código se eliminan antes de que se transmitan incluyendo aplicar una técnica de perforación de manera que la palabra de código transmitida tiene un número menor de bits que la palabra de código obtenida antes de la perforación.

15. El procedimiento de la reivindicación 14, en el que:

la palabra de código es una palabra de código de 1152 bits y una tasa de codificación correspondiente es 5/6; y

la técnica de perforación incluye el siguiente patrón de perforación:

$$pp_{1152}^{(16/18)} = [\underbrace{1\ 1\ \cdots\ 1}_{720}\ \underbrace{0\ 0\ \cdots\ 0}_{36}\ \underbrace{1\ 1\ \cdots\ 1}_{360}\ \underbrace{0\ 0\ \cdots\ 0}_{36}]$$

para obtener una palabra de código de 1080 bits y una tasa de codificación de 16/18.

16. El procedimiento de la reivindicación 14, en el que:

la palabra de código es una palabra de código de 5184 bits y una tasa de codificación correspondiente es 5/6; y

la técnica de perforación incluye el siguiente patrón de perforación:

$$pp_{5184}^{(16/18)} = [\underbrace{1\ 1\ \cdots\ 1}_{3240}\ \underbrace{0\ 0\ \cdots\ 0}_{162}\ \underbrace{1\ 1\ \cdots\ 1}_{972}\ \underbrace{0\ 0\ \cdots\ 0}_{162}\ \underbrace{1\ 1\ \cdots\ 1}_{648}]$$

para obtener una palabra de código de 4860 bits y una tasa de codificación de 16/18.

17. El procedimiento de la reivindicación 14, en el que:

la palabra de código es una palabra de código de 1152 bits y una tasa de codificación correspondiente es 5/6; y

la técnica de perforación incluye el siguiente patrón de perforación:

$$pp_{1152}^{(20/21)} = [\underbrace{1\ 1\ \cdots\ 1}_{720}\ \underbrace{0\ 0\ \cdots\ 0}_{48}\ \underbrace{1\ 1\ \cdots\ 1}_{240}\ \underbrace{0\ 0\ \cdots\ 0}_{96}\ \underbrace{1\ 1\ \cdots\ 1}_{48}]$$

para obtener una palabra de código de 1008 bits y una tasa de codificación de 20/21.

18. El procedimiento de la reivindicación 14, en el que:

la palabra de código es una palabra de código de 5184 bits y una tasa de codificación correspondiente es 5/6; y

la técnica de perforación incluye el siguiente patrón de perforación:

$$pp_{5184}^{(20/21)} = [\underbrace{0\ 0\ \cdots\ 0}_{216}\ \underbrace{1\ 1\ \cdots\ 1}_{4320}\ \underbrace{0\ 0\ \cdots\ 0}_{432}\ \underbrace{1\ 1\ \cdots\ 1}_{216}]$$

para obtener una palabra de código de 4536 bits y una tasa de codificación de 20/21.

19. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 336 bits con una tasa de codificación de ½ es:

-1	-1	-1	6	-1	-1	9	6	-1	-1	2	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	0	-1	-1	-1	3	-1	12	1	-1	-1	3	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	9	11	-1	-1	13	-1	-1	2	12	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
1	-1	-1	11	-1	-1	7	-1	-1	-1	11	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	-1	4	8	-1	-1	-1	-1	-1	2	5	4	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	3	0	-1	-1	8	-1	-1	1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	0	6	-1	-1	-1	-1	5	13	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	-1	-1	9	-1	-1	-1	3	-1	-1	3	1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
9	0	13	-1	-1	12	-1	-1	8	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	5	-1	-1	1	4	-1	-1	5	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	-1	-1	8	-1	-1	8	-1	-1	9	0	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0
10	11	-1	-1	-1	3	-1	-1	0	-1	-1	-1	4	8	-1	-1	-1	-1	-1	-1	-1	-1	0

20. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 1920 bits con una tasa de codificación de ½ es:

27	-1	-1	-1	55	19	-1	30	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	1	-1	70	-1	47	-1	62	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	41	-1	-1	-1	44	-1	-1	59	60	25	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
16	77	-1	-1	-1	5	-1	48	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	45	-1	27	-1	46	19	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	63	-1	-1	-1	55	-1	-1	-1	48	26	10	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	-1	42	-1	21	-1	58	-1	41	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	-1	-1	-1	78	0	-1	7	52	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
-1	29	9	-1	-1	-1	37	-1	-1	-1	35	21	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	-1	22	72	-1	-1	47	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
35	-1	-1	-1	-1	13	-1	35	-1	70	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	46	28	-1	-1	-1	38	-1	-1	-1	8	-1	10	58	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

21. El procedimiento de la reivindicación 1, en el que la matriz compacta H_1 usada para obtener palabras de código de 8640 bits con una tasa de codificación de ½ es:

-1	34	-1	95	-1	279	-1	-1	-1	-1	248	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	0	-1	-1	-1	-1	134	356	275	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
51	-1	27	-1	-1	-1	-1	-1	22	152	-1	57	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
-1	124	-1	290	-1	281	15	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	340	-1	99	336	-1	-1	1	-1	-1	-1	-1	33	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
163	-1	46	-1	-1	-1	-1	-1	-1	306	-1	86	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	185	-1	24	-1	-1	-1	94	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	223	-1	225	325	-1	-1	-1	-1	-1	297	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
46	-1	314	-1	-1	-1	59	-1	-1	67	-1	120	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	-1	121	-1	-1	-1	-1	161	-1	303	-1	264	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	303	-1	8	-1	185	-1	-1	138	-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
-1	-1	312	-1	-1	-1	100	-1	-1	144	-1	307	33	166	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

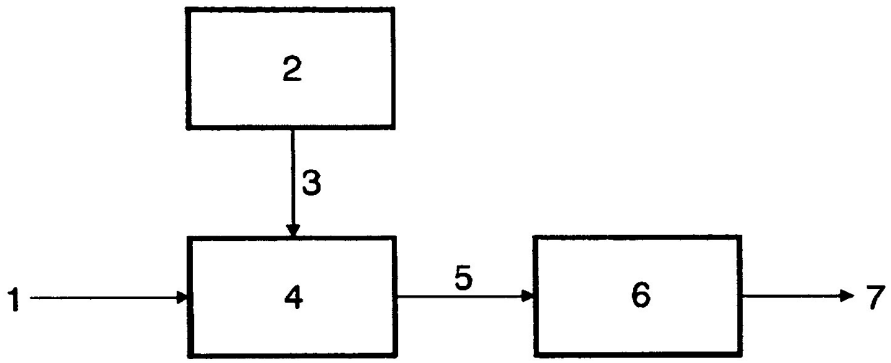

FIG. 1

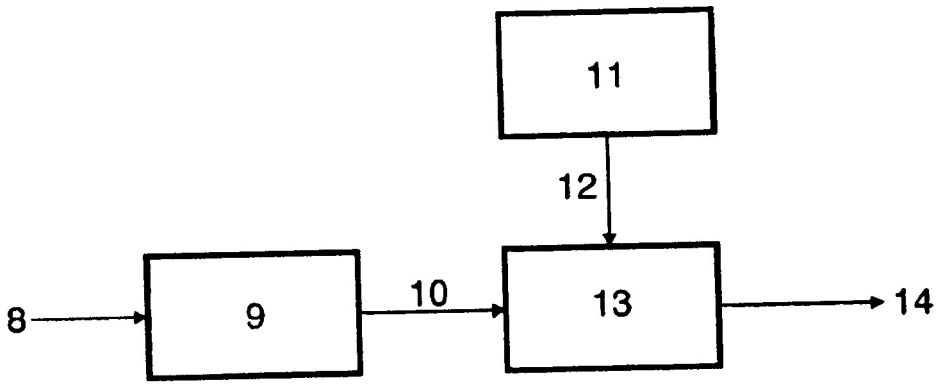

FIG. 2

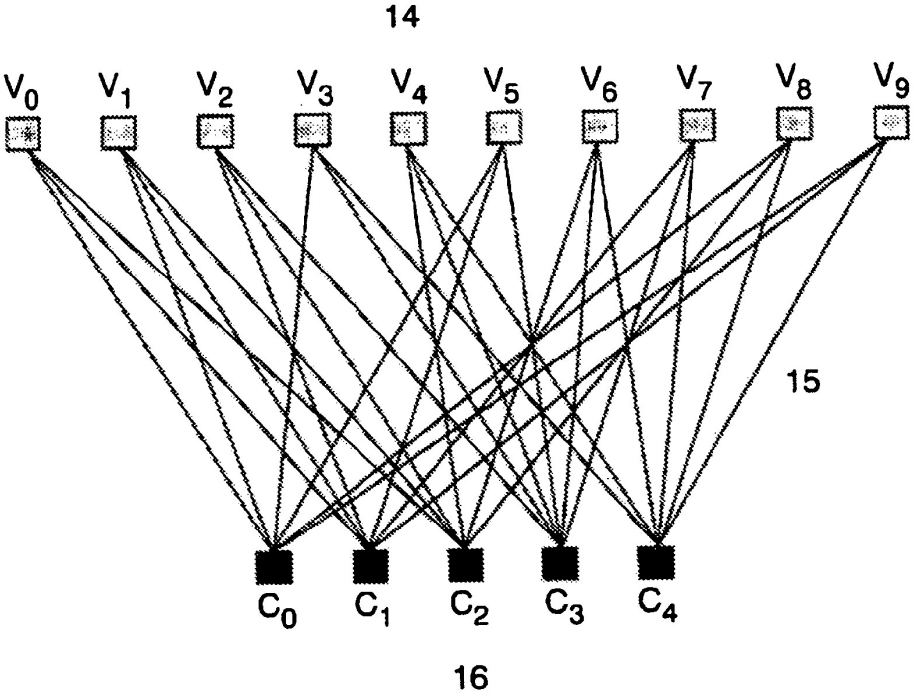

FIG. 3

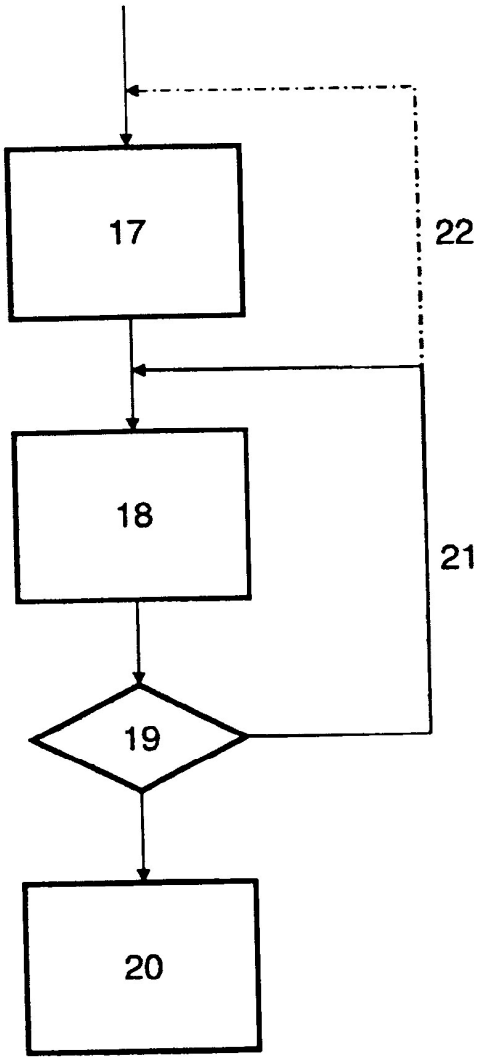

FIG. 4