

19



OFICINA ESPAÑOLA DE
PATENTES Y MARCAS

ESPAÑA



11 Número de publicación: **2 531 013**

51 Int. Cl.:

G10L 19/00

(2013.01)

12

TRADUCCIÓN DE PATENTE EUROPEA

T3

96 Fecha de presentación y número de la solicitud europea: **19.10.2010** **E 10768018 (3)**

97 Fecha y número de publicación de la concesión europea: **31.12.2014** **EP 2491552**

54 Título: **Codificador de audio, decodificador de audio, método para codificar información de audio, método para decodificar información de audio y programa de computación que usa la detección de un grupo de valores espectrales previamente decodificados**

30 Prioridad:

20.10.2009 US 253459 P

45 Fecha de publicación y mención en BOPI de la traducción de la patente:
10.03.2015

73 Titular/es:

**FRAUNHOFER-GESELLSCHAFT ZUR
FÖRDERUNG DER ANGEWANDTEN
FORSCHUNG E.V. (100.0%)
Hansastraße 27c
80686 München, DE**

72 Inventor/es:

**FUCHS, GUILLAUME;
SUBBARAMAN, VIGNESH;
RETTELBACH, NIKOLAUS;
MULTRUS, MARKUS;
GAYER, MARC;
WARMBOLD, PATRICK;
GRIEBEL, CHRISTIAN y
WEISS, OLIVER**

74 Agente/Representante:

PONTI SALES, Adelaida

ES 2 531 013 T3

Aviso: En el plazo de nueve meses a contar desde la fecha de publicación en el Boletín europeo de patentes, de la mención de concesión de la patente europea, cualquier persona podrá oponerse ante la Oficina Europea de Patentes a la patente concedida. La oposición deberá formularse por escrito y estar motivada; sólo se considerará como formulada una vez que se haya realizado el pago de la tasa de oposición (art. 99.1 del Convenio sobre concesión de Patentes Europeas).

DESCRIPCIÓN

Codificador de audio, decodificador de audio, método para codificar información de audio, método para decodificar información de audio y programa de computación que usa la detección de un grupo de valores espectrales previamente decodificados

Campo técnico

[0001] Las formas de realización de acuerdo con la invención se refieren a un decodificador de audio para proveer información de audio decodificada sobre la base de una información de audio codificada, un codificador de audio para proveer una información de audio codificada sobre la base de una información de audio entrante, un método para proveer información de audio decodificada sobre la base de una información de audio codificada, un método para proveer una información de audio codificada sobre la base de una información de audio entrante y un programa de computación.

[0002] Las formas de realización de acuerdo con la invención se refieren a una codificación espectral insonorizada mejorada, que se puede usar en un codificador o decodificador de audio, por ejemplo, como el llamado codificador de lenguaje y audio unificados (USAC, por sus siglas en inglés).

Antecedentes de la invención

[0003] A continuación se explicarán brevemente los antecedentes de la invención para facilitar la comprensión de la invención y sus ventajas. Durante la última década, se han hecho grandes esfuerzos para desarrollar la posibilidad de almacenar y distribuir en forma digital contenidos de audio con buena eficacia en la velocidad de transmisión de bits. Un logro importante con respecto a lo anterior es la definición de la Norma Internacional ISO/IEC 14496-3. La parte 3 de esta Norma se relaciona con la codificación y decodificación de contenidos de audio, y la subparte 4 de la parte 3 se relaciona con codificación general de audio. La ISO/IEC 14496, parte 3, subparte 4 define un concepto para la codificación y decodificación de contenidos de audio en general. Además, se han propuesto otras mejoras para aumentar la calidad y/o reducir la velocidad de transmisión de bits requerida.

[0004] De acuerdo con el concepto descrito en dicha Norma, una señal de audio de dominio temporal se convierte en una representación tiempo–frecuencia. La transformación del dominio temporal al dominio tiempo–frecuencia se lleva a cabo generalmente usando bloques transformadores, que también reciben el nombre de “marcos”, de muestras de dominio temporal. Se ha descubierto que es conveniente usar marcos superpuestos, que sufren un desplazamiento, por ejemplo, de medio marco, porque la superposición permite evitar (o por lo menos reducir) con eficiencia artefactos. Además, se ha descubierto que debería ejecutarse un sistema de ventanas para evitar los artefactos que se originan de este procesamiento de marcos temporalmente limitados.

[0005] Transformando una porción de ventana de la señal de audio entrante desde el dominio temporal al dominio tiempo–frecuencia, en muchos casos se obtiene una compactación de la energía, de manera que algunos de los valores espectrales comprenden una magnitud enormemente mayor que una pluralidad de otros valores espectrales. En consecuencia, en muchos casos, existe un número comparativamente pequeño de valores espectrales que poseen una magnitud considerablemente superior a una magnitud promedio de los valores espectrales. Un ejemplo típico de una transformación de dominio temporal a dominio tiempo–frecuencia que proporciona una compactación de energía es la denominada transformada discreta del coseno modificada (MDCT, por sus siglas en inglés).

[0006] Los valores espectrales generalmente están escalados y cuantificados de acuerdo con un modelo psicoacústico, de modo tal que los errores de cuantificación son comparativamente más pequeños para los valores espectrales psicoacústicamente más importantes, y son comparativamente mayores para los valores espectrales psicoacústicamente menos importantes. Los valores espectrales escalados y cuantificados son codificados para proveer una representación eficaz de la velocidad de transmisión de bits de aquellos.

[0007] Por ejemplo, la utilización de una codificación denominada de Huffman de coeficientes espectrales cuantificados está descrita en la Norma Internacional ISO/IEC 14496-3:2005(E), parte 3, subparte 4.

[0008] El artículo "Improved Quantization and Lossless Coding for Subband Audio Coding" de N. Meine et al. (ediciones preliminares de los trabajos presentados en la 118ª Convención AES, Barcelona, España, vol. 1-4, 31 de mayo de 2005, páginas 1-9) describe un algoritmo de codificación de fuente basado en el modelo de Markov clásico, que utiliza cuantización vectorial y codificación aritmética en conjunción con un contexto adaptado dinámicamente de índices espectrales previamente codificados. El núcleo del algoritmo es el mapeado numéricamente optimizado de un gran número de estados de origen a un pequeño número de diferentes tablas codificadas. Esto permite su aplicación a la codificación de audio.

[0009] Sin embargo, se ha descubierto que la calidad de la codificación de los valores espectrales tiene un importantísimo efecto en la velocidad de transmisión de bits requerida. Asimismo, se ha descubierto que la complejidad de un decodificador de audio, que generalmente está implementado como un dispositivo portátil para

uso del consumidor y que debería ser, por lo tanto, barato y de bajo consumo de energía, depende de la codificación que se haya usado para codificar los valores espectrales.

- 5 **[00010]** En vista de esta situación, es necesario desarrollar un concepto para la codificación y decodificación de un contenido de audio, que proporcione una mejor correlación entre la eficacia de la velocidad de transmisión de bits y la eficacia de recursos.

Descripción de la invención

- 10 **[00011]** Una forma de realización de acuerdo con la invención tal como se reivindica en la reivindicación 1 crea un decodificador de audio para proveer información de audio decodificada (o representación de audio decodificada) sobre la base de una información de audio codificada (o representación de audio codificada). El decodificador de audio comprende un decodificador aritmético para proveer una pluralidad de valores espectrales decodificados sobre la base de una representación codificada aritméticamente de los valores espectrales. El decodificador de audio también comprende un convertidor de dominio frecuencial a dominio temporal para proveer una representación de audio de dominio temporal que utiliza los valores espectrales decodificados, para obtener la información de audio decodificada. El decodificador aritmético tiene una configuración que le permite seleccionar una norma de mapeo que describe un mapeo de un valor de código en un código de símbolos de manera dependiente de un estado contextual. El decodificador aritmético tiene una configuración que le permite determinar el estado contextual actual de manera dependiente de una pluralidad de valores espectrales previamente decodificados. El decodificador aritmético tiene una configuración que le permite detectar un grupo de una pluralidad de valores espectrales previamente decodificados, que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, y determinar o modificar el estado contextual actual de manera dependiente de un resultado obtenido con la detección.

- 25 **[00012]** Esta forma de realización de acuerdo con la invención se basa en el hallazgo de que la presencia de un grupo de una pluralidad de valores espectrales (preferiblemente, pero no necesariamente, adyacentes) previamente decodificados, que satisfacen la condición predeterminada relacionada con sus magnitudes, da lugar a una determinación especialmente eficaz del estado contextual actual debido a que un grupo de valores espectrales (preferiblemente adyacentes) previamente decodificados es un rasgo característico de la representación espectral y puede usarse, por lo tanto, para facilitar la determinación del estado contextual actual. Mediante la detección de un grupo de una pluralidad de valores espectrales (preferiblemente adyacentes) previamente decodificados que comprenden, por ejemplo, una magnitud particularmente pequeña, es posible reconocer porciones de amplitud comparativamente baja dentro del espectro, y ajustar (determinar o modificar) el estado contextual actual en consecuencia, de tal manera que se pueden codificar y decodificar otros valores espectrales adicionales con buena eficacia de codificación (en términos de velocidad de transmisión de bits). Alternativamente, se pueden detectar grupos de una pluralidad de valores espectrales adyacentes previamente decodificados que comprenden una amplitud comparativamente grande, y el contexto puede ajustarse (determinarse o modificarse) adecuadamente para aumentar la eficacia de la codificación y decodificación. Por otra parte, se puede ejecutar en general la detección de grupos de una pluralidad de valores espectrales (preferiblemente adyacentes) previamente decodificados que satisfacen, considerados individualmente o en conjunto, la condición predeterminada, con menor esfuerzo computacional que una computación contextual en la que se combinan muchos valores espectrales previamente decodificados. En síntesis, la forma de realización de acuerdo con la invención que se acaba de analizar permite una computación contextual simplificada y permite ajustar el contexto a constelaciones de señales específicas en las que hay grupos de valores espectrales adyacentes comparativamente pequeños o grupos de valores espectrales adyacentes comparativamente grandes.

- 50 **[00013]** En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite determinar o modificar el estado contextual actual independientemente de los valores espectrales previamente decodificados en respuesta a la detección de que la condición predeterminada ha sido cumplida. En consecuencia, se obtiene un mecanismo especialmente eficaz desde el punto de vista computacional para la derivación de un valor que describe el contexto. Se ha descubierto que se puede conseguir una adaptación significativa del contexto si la detección de un grupo de una pluralidad de valores espectrales previamente decodificados, que satisfacen la condición predeterminada, da como resultado un mecanismo simple, que no requiere una combinación numérica de valores espectrales previamente decodificados complicada desde el punto de vista computacional. Por consiguiente, se reduce el esfuerzo computacional con respecto a otros métodos. Asimismo, se puede obtener una aceleración de la derivación contextual omitiendo pasos complicados de cálculo que dependen de la detección, porque este concepto resulta generalmente ineficiente cuando se lo aplica a una implementación de software ejecutada en un procesador.

- 60 **[00014]** En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados, que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes.

[00015] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados que, considerados individualmente o en conjunto, comprenden una magnitud que es más pequeña que una magnitud umbral predeterminada, y determinar el estado contextual actual dependiendo del resultado de la detección. Se ha descubierto que se puede usar un grupo de una pluralidad de valores espectrales adyacentes comparativamente bajos para seleccionar un contexto que está bien adaptado a esta situación. Si hay un grupo de valores espectrales adyacentes comparativamente pequeños, hay una enorme posibilidad de que el valor espectral que se decodificará a continuación también comprenda un valor comparativamente pequeño. En consecuencia, un ajuste del contexto proporciona una buena eficacia de codificación y puede ayudar a evitar cálculos contextuales que insuman demasiado tiempo.

[00016] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados, donde cada uno de los valores espectrales previamente decodificados es un valor cero, y determinar el estado contextual dependiendo del resultado de la detección. Se ha descubierto que, debido a efectos de enmascaramiento espectral o temporal, con frecuencia aparecen grupos de valores espectrales adyacentes que toman un valor cero. La forma de realización descrita proporciona un manejo eficaz de esta situación. Además, la presencia de un grupo de valores espectrales adyacentes, que se cuantifican a cero, hace muy probable que el valor espectral que se decodificará a continuación sea un valor cero o un valor espectral comparativamente grande, que produce el efecto de enmascaramiento.

[00017] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados, que comprenden un valor suma que es más pequeño que un valor umbral predeterminado, y determinar el estado contextual de manera dependiente de un resultado obtenido con la detección. Se ha descubierto que además de los grupos de valores espectrales adyacentes que son cero, también aquellos grupos de valores espectrales adyacentes que son casi cero en promedio (es decir, cuyo valor suma es más pequeño que un valor umbral predeterminado), constituyen un rasgo característico de una representación espectral (p. ej. una representación tiempo–frecuencia del contenido de audio) que se puede usar para la adaptación del contexto.

[00018] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite ajustar el estado contextual actual a un valor predeterminado en respuesta a la detección de la condición predeterminada. Se ha descubierto que esta reacción es muy fácil de implementar y además da como resultado una adaptación del contexto que aporta una buena eficacia de codificación.

[00019] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite omitir de manera selectiva un cálculo del estado contextual actual de manera dependiente de los valores numéricos de una pluralidad de valores espectrales previamente decodificados en respuesta a la detección de la condición predeterminada. En consecuencia, la computación del contexto se simplifica de manera considerable en respuesta a la detección de un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados que satisfacen la condición predeterminada. Al ahorrar esfuerzo computacional, se reduce también el consumo de energía del decodificador de señales de audio, que proporciona ventajas importantes en los dispositivos móviles.

[00020] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite ajustar el estado contextual actual a un valor que señala la detección de la condición predeterminada. Configurando el estado contextual a dicho valor, que puede encontrarse dentro de un intervalo predeterminado de valores, se puede controlar la evaluación posterior del estado contextual. Sin embargo, debe tenerse en cuenta que el valor al que se configura el estado contextual actual, puede depender también de otros criterios, aunque el valor puede hallarse en un intervalo de valores característico que señala la detección de la condición predeterminada.

[00021] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite mapear un código de símbolos a un valor espectral decodificado.

[00022] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite evaluar valores espectrales de una primera región de tiempo–frecuencia, para detectar un grupo de una pluralidad de valores espectrales que satisfacen, considerados individualmente o en conjunto, la condición predeterminada relacionada con sus magnitudes. El decodificador aritmético tiene una configuración que le permite obtener un valor numérico que representa el estado contextual, de manera dependiente de los valores espectrales de una segunda región de tiempo–frecuencia, que es diferente de la primera región de tiempo–frecuencia, si no se satisface la condición predeterminada. Se ha descubierto que es recomendable detectar un grupo de una pluralidad de valores espectrales que satisfacen la condición predeterminada relacionada con la magnitud dentro de una región que se diferencia de la región que habitualmente se usa para los cálculos del contexto. Esto se debe a que una extensión, por ejemplo una extensión de frecuencias, de regiones que comprenden valores espectrales comparativamente pequeños, o valores espectrales comparativamente grandes, generalmente es más grande que una dimensión de una región de valores espectrales que se tendrán en cuenta para un cálculo numérico de un valor numérico que representa el estado contextual. En consecuencia, es recomendable analizar regiones diferentes para detectar un

grupo de una pluralidad de valores espectrales que satisfacen la condición predeterminada, y para el cómputo numérico de un valor numérico que representa el estado contextual (donde el cálculo numérico solo puede esperarse en un segundo paso si la detección no aporta un bit.

[00023] En una forma de realización preferida, el decodificador aritmético tiene una configuración que le permite evaluar una o varias tablas *hash* (elección arbitraria de elementos o resumen criptográfico) para seleccionar una norma de mapeo dependiente del estado contextual. Se ha descubierto que la selección de la norma de mapeo puede ser controlada por el mecanismo de detección de una pluralidad de valores espectrales adyacentes que satisfacen la condición predeterminada.

[00024] Una forma de realización de acuerdo con la invención tal como se reivindica en la reivindicación 13 crea un codificador de audio para proveer una información de audio codificada, sobre la base de una información de audio entrante. El codificador de audio comprende un convertidor de dominio temporal a dominio frecuencial que compacta la energía para proveer una representación de audio de dominio frecuencial, sobre la base de una representación de dominio temporal de la información de audio entrante, de modo tal que la representación de audio de dominio frecuencial comprende un conjunto de valores espectrales. El codificador de audio también comprende un codificador aritmético que tiene una configuración para codificar un valor espectral, o una versión preprocesada de éste, que utiliza una palabra de código de longitud variable. El codificador aritmético tiene una configuración que le permite mapear un valor espectral o un valor de un plano de bits más significativos de un valor espectral a un valor de código. El codificador aritmético tiene una configuración que le permite seleccionar una norma de mapeo que describe un mapeo de un valor espectral o de un plano de bits más significativos de un valor espectral a un valor de código dependiente del estado contextual. El codificador aritmético tiene una configuración que le permite determinar el estado contextual actual de manera dependiente de una pluralidad de valores espectrales adyacentes previamente codificados. El codificador aritmético tiene una configuración que le permite detectar un grupo de una pluralidad de valores espectrales adyacentes previamente codificados, que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, y determinar el estado contextual actual de manera dependiente de un resultado obtenido con la detección.

[00025] Este codificador de señales de audio se base en los mismos hallazgos que el decodificador de señales de audio analizado más arriba. Se ha descubierto que el mecanismo para la adaptación del contexto, que ha demostrado ser eficaz para la decodificación de un contenido de audio, también podría aplicarse respecto del codificador, para permitir un sistema conveniente.

[00026] Una forma de realización de acuerdo con la invención tal como se reivindica en la reivindicación 16 crea un método para proveer información de audio decodificada sobre la base de información de audio codificada.

[00027] Aún otra forma de realización de acuerdo con la invención tal como se reivindica en la reivindicación 17 crea un método para proveer información de audio codificada sobre la base de una información de audio entrante.

[00028] Otra forma de realización de acuerdo con la invención tal como se reivindica en la reivindicación 18 crea un programa de computación para ejecutar uno de dichos métodos.

[00029] Los métodos y el programa de computación se basan en los mismos hallazgos que el decodificador de audio descripto más arriba y el codificador de audio descripto más arriba.

Breve descripción de las Figuras

[00030] A continuación se describirán las formas de realización de acuerdo con la presente invención con referencia a las figuras que se acompañan, en las cuales:

La Fig. 1A-B muestra un diagrama esquemático en bloque de un codificador de audio, de acuerdo con una forma de realización de la invención;

La Fig. 2A-B muestra un diagrama esquemático en bloque de un decodificador de audio, de acuerdo con una forma de realización de la invención;

La Fig. 3 muestra una representación de pseudocódigo de programa de un algoritmo "value_decode()" para decodificar un valor espectral;

La Fig. 4 muestra una representación esquemática de un contexto para un cálculo de estados;

La Fig. 5A muestra una representación de pseudocódigo de programa de un algoritmo "arith_map_context ()" para el mapeo de un contexto;

Las Fig. 5B y 5C muestran una representación de pseudocódigo de programa de un algoritmo "arith_get_context ()" para obtener un valor de estado contextual;

La Fig. 5D1-5D2 muestra una representación de pseudocódigo de programa de un algoritmo "get_pk(s)" para derivar un valor de índice de tablas de frecuencias acumulativas „pki" de una variable de estado;

La Fig. 5E muestra una representación de pseudocódigo de programa de un algoritmo "arith_get_pk(s)" para derivar un valor de índice de tablas de frecuencias acumulativas „pki" de un valor de estado;

La Fig. 5F muestra una representación de pseudocódigo de programa de un algoritmo "get_pk(unsigned long s)" para derivar un valor de índice de tablas de frecuencias acumulativas „pki" de un valor de estado;

La Fig. 5G1-5G2 muestra una representación de pseudocódigo de programa de un algoritmo "arith_decode ()" para decodificar aritméticamente un símbolo de una palabra de código de longitud variable;

La Fig. 5H muestra una representación de pseudocódigo de programa de un algoritmo "arith_update_context ()" para actualizar el contexto;

5 La Fig. 5I muestra una leyenda de definiciones y variables;

La Fig. 6A muestra una representación sintáctica de un bloque de datos crudos de codificación de lenguaje y audio unificados (USAC);

La Fig. 6B muestra una representación sintáctica de un elemento de canal simple;

La Fig. 6C muestra una representación sintáctica de un elemento de canal par;

10 La Fig. 6D muestra una representación sintáctica de una información de control "ics";

La Fig. 6E muestra una representación sintáctica de una corriente de canal de dominio frecuencial;

La Fig. 6F muestra una representación sintáctica de datos espectrales codificados aritméticamente;

La Fig. 6G muestra una representación sintáctica para decodificar un conjunto de valores espectrales;

La Fig. 6H muestra una leyenda de elementos y variables de datos;

15 La Fig. 7 muestra un diagrama esquemático en bloque de un codificador de audio, de acuerdo con otra forma de realización de la invención;

La Fig. 8 muestra un diagrama esquemático en bloque de un decodificador de audio, de acuerdo con otra forma de realización de la invención;

20 La Fig. 9 muestra una disposición para la comparación de una codificación insonorizada de acuerdo con una propuesta de trabajo 3 del estándar de propuestas del USAC con un esquema de codificación de acuerdo con la presente invención;

La Fig. 10A muestra una representación esquemática de un contexto para un cálculo de estados, tal como se usa de acuerdo con la propuesta de trabajo 4 del estándar de propuestas del USAC;

25 La Fig. 10B muestra una representación esquemática de un contexto para un cálculo de estados, tal como se usa en las formas de realización de acuerdo con la invención;

La Fig. 11A muestra un resumen de la tabla tal como se usa en el esquema de codificación aritmética de acuerdo con la propuesta de trabajo 4 del estándar de propuestas del USAC;

La Fig. 11B muestra un resumen de la tabla tal como se usa en el esquema de codificación aritmética de acuerdo con la presente invención;

30 La Fig. 12A muestra una representación gráfica de una demanda de memoria de solo lectura para los esquemas de codificación insonorizada de acuerdo con la presente invención y de acuerdo con la propuesta de trabajo 4 del estándar de propuestas del USAC;

La Fig. 12B muestra una representación gráfica de una demanda de memoria de solo lectura de datos de un decodificador USAC total de acuerdo con la presente invención y de acuerdo con el concepto de acuerdo con la propuesta de trabajo 4 del estándar de propuestas del USAC;

35 La Fig. 13A muestra una representación en forma de tabla de las velocidades promedio de transmisión de bits que son usadas por un codificador de codificación de lenguaje y audio unificados, empleando un codificador aritmético de acuerdo con la propuesta de trabajo 3 del estándar de propuestas del USAC y un decodificador aritmético de acuerdo con una forma de realización de la presente invención;

40 La Fig. 13B muestra una representación en forma de tabla de un control de depósito de bits para un codificador de codificación de lenguaje y audio unificados, empleando el codificador aritmético de acuerdo con la propuesta de trabajo 3 del estándar de propuestas del USAC y el codificador aritmético de acuerdo con una forma de realización de la invención;

45 La Fig. 14 muestra una representación en forma de tabla de las velocidades promedio de transmisión de bits para un codificador de tipo USAC de acuerdo con la propuesta de trabajo 3 del estándar de propuestas del USAC, y de acuerdo con una forma de realización de la presente invención;

La Fig. 15 muestra una representación en forma de tabla de las velocidades mínimas, máximas y promedio de transmisión de bits del USAC sobre la base de un marco;

La Fig. 16 muestra una representación en forma de tabla de los casos mejores y peores sobre la base de un marco;

50 Las Figs. 17-1 y 17-2 muestran una representación en forma de tabla de un contenido de una tabla "ari_s_hash[387]";

La Fig. 18 muestra una representación en forma de tabla de un contenido de una tabla "ari_gs_hash[225]";

Las Figs. 19-1 y 19-2 muestran una representación en forma de tabla de un contenido de una tabla "ari_cf_m[64][9]";

55 Las Figs. 20-1 y 20-2 muestran una representación en forma de tabla de un contenido de una tabla "ari_s_hash[387].

Descripción detallada de las realizaciones

1. Codificador de audio de acuerdo con la Fig. 7

60 **[00031]** La Fig. 7 muestra un diagrama esquemático en bloque de un codificador de audio, de acuerdo con una forma de realización de la invención. El codificador de audio 700 tiene una configuración que le permite recibir una información de audio entrante 710 y proveer, sobre esta base, una información de audio codificada 712. El codificador de audio comprende un convertidor de dominio temporal a dominio frecuencial que compacta la energía

65 720 que tiene una configuración que le permite proveer una representación de audio de dominio frecuencial 722 sobre la base de una representación de dominio temporal de la información de audio entrante 710, de modo tal que

la representación de audio de dominio frecuencial 722 comprende un conjunto de valores espectrales. El codificador de audio 700 también comprende un codificador aritmético 730 que tiene una configuración para codificar un valor espectral (tomado del conjunto de valores espectrales que forman la representación de audio de dominio frecuencial 722), o una versión preprocesada de aquel, que utiliza una palabra de código de longitud variable, para obtener la información de audio codificada 712 (que puede comprender, por ejemplo, una pluralidad de palabras de código de longitud variable).

[00032] El codificador aritmético 730 tiene una configuración que le permite mapear un valor espectral o un valor de un plano de bits más significativos de un valor espectral a un valor de código (es decir, a una palabra de código de longitud variable), de manera dependiente de un estado contextual. El codificador aritmético 730 tiene una configuración que le permite seleccionar una norma de mapeo que describe un mapeo de un valor espectral, o de un plano de bits más significativos de un valor espectral, a un valor de código, de manera dependiente de un estado contextual. El codificador aritmético tiene una configuración que le permite determinar el estado contextual actual de manera dependiente de una pluralidad de valores espectrales (preferiblemente, pero no necesariamente, adyacentes) previamente codificados. Con esta finalidad, el codificador aritmético tiene una configuración que le permite detectar un grupo de una pluralidad de valores espectrales adyacentes previamente codificados, que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, y determinar el estado contextual actual de manera dependiente de un resultado obtenido con la detección.

[00033] Según puede observarse, el mapeo de un valor espectral o de un plano de bits más significativos de un valor espectral a un valor de código puede realizarse mediante una codificación de valores espectrales 740 usando una norma de mapeo 742. Se puede configurar un rastreador de estados 750 para que rastree el estado contextual y puede comprender un grupo detector 752 para detectar un grupo de una pluralidad de valores espectrales adyacentes previamente codificados que satisfacen, considerados individualmente o en conjunto, la condición predeterminada relacionada con sus magnitudes. El rastreador de estados 750 también tiene, de preferencia, una configuración que le permite determinar el estado contextual actual dependiendo del resultado de dicha detección ejecutada por el detector de grupos 752. En consecuencia, el rastreador de estados 750 provee una información 754 que describe el estado contextual actual. Un selector de normas de mapeo 760 puede seleccionar una norma de mapeo, por ejemplo, una tabla de frecuencias acumulativas, que describe un mapeo de un valor espectral, o de un plano de bits más significativos de un valor espectral, a un valor de código. En consecuencia, el selector de normas de mapeo 760 provee la información de normas de mapeo 742 a la codificación espectral 740.

[00034] Para resumir lo anterior, el codificador de audio 700 realiza una codificación aritmética de una representación de audio de dominio frecuencial provista por el convertidor de dominio temporal a dominio frecuencial. La codificación aritmética depende del contexto, de modo tal que una norma de mapeo (p. ej., una tabla de frecuencias acumulativas) se selecciona de manera dependiente de los valores espectrales previamente codificados. En consecuencia, los valores espectrales adyacentes en tiempo y/o frecuencia (o por lo menos, dentro de un entorno predeterminado) entre sí y/o al valor espectral codificado actualmente (es decir, valores espectrales dentro de un entorno predeterminado del valor espectral codificado actualmente) se consideran en la codificación aritmética para ajustar la distribución de probabilidades evaluada por la codificación aritmética. Cuando se escoge una norma de mapeo apropiada, se realiza una detección para detectar si hay un grupo de una pluralidad de valores espectrales adyacentes previamente codificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes. El resultado de esta detección se aplica en la selección del estado contextual actual, es decir, en la selección de una norma de mapeo. Al detectar si hay un grupo de una pluralidad de valores espectrales que son particularmente pequeños o particularmente grandes, es posible reconocer rasgos especiales dentro de la representación de audio de dominio frecuencial, que puede ser una representación tiempo-frecuencia. Los rasgos especiales como, por ejemplo, un grupo de una pluralidad de valores espectrales particularmente pequeños o particularmente grandes, indican que debería usarse un estado contextual específico ya que este estado contextual específico puede proveer una eficacia de codificación particularmente buena. Por consiguiente, la detección del grupo de valores espectrales adyacentes que satisfacen la condición predeterminada, que normalmente se usa en combinación con una evaluación alternativa del contexto basada en una combinación de una pluralidad de valores espectrales previamente codificados, provee un mecanismo que permite una selección eficaz de un contexto apropiado si la información de audio entrante adopta algunos estados especiales (p. ej., comprende un intervalo grande de frecuencias enmascaradas).

[00035] En consecuencia, se puede obtener una codificación eficaz a la vez que el cálculo del contexto se mantiene lo suficientemente simple.

2. Decodificador de audio de acuerdo con la Fig. 8

[00036] La Fig. 8 muestra un diagrama esquemático en bloque de un decodificador de audio 800. El decodificador de audio 800 tiene una configuración que le permite recibir una información de audio codificada 810 y proveer, sobre esta base, información de audio decodificada 812. El decodificador de audio 800 comprende un decodificador aritmético 820 que tiene una configuración que le permite proveer una pluralidad de valores espectrales decodificados 822 sobre la base de una representación aritméticamente codificada 821 de los valores espectrales.

El decodificador de audio 800 también comprende un convertidor de dominio frecuencial a dominio temporal 830 que tiene una configuración que le permite recibir los valores espectrales decodificados 822 y proveer la representación de audio de dominio temporal 812, que puede constituir la información de audio decodificada, que utiliza los valores espectrales decodificados 822, para obtener información de audio decodificada 812.

[00037] El decodificador aritmético 820 comprende un determinador de valores espectrales 824 que tiene una configuración que le permite mapear un valor de código de la representación aritméticamente codificada 821 de valores espectrales a un código de símbolos que representa uno o varios de los valores espectrales decodificados, o por lo menos una porción (por ejemplo, un plano de bits más significativos) de uno o varios de los valores espectrales decodificados. El determinador de valores espectrales 824 puede tener una configuración que le permite realizar el mapeo de manera dependiente de una norma de mapeo, que puede ser descripta por una información de las normas de mapeo 828a.

[00038] El decodificador aritmético 820 tiene una configuración que le permite seleccionar una norma de mapeo (p. ej. una tabla de frecuencias acumulativas) que describe un mapeo de un valor de código (descripto por la representación aritméticamente codificada 821 de valores espectrales) a un código de símbolos (que describe uno o varios valores espectrales) de manera dependiente de un estado contextual (que puede ser descripto por la información del estado contextual 826a). El decodificador aritmético 820 tiene una configuración que le permite determinar el estado contextual actual de manera dependiente de una pluralidad de valores espectrales previamente decodificados 822. Con esta finalidad, se puede usar un rastreador de estados 826, que recibe una información que describe los valores espectrales previamente decodificados. El decodificador aritmético también tiene una configuración que le permite detectar un grupo de una pluralidad de valores espectrales (preferiblemente, pero no necesariamente, adyacentes) previamente decodificados, que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, y determinar el estado contextual actual (descripto, por ejemplo, por la información del estado contextual 826a) de manera dependiente de un resultado obtenido con la detección.

[00039] La detección del grupo de una pluralidad de valores espectrales adyacentes previamente decodificados que satisfacen la condición predeterminada relacionada con sus magnitudes puede ser realizada, por ejemplo, por un detector de grupos, que es parte del rastreador de estados 826. En consecuencia, se obtiene una información del estado contextual actual 826a. La selección de la norma de mapeo puede ser realizada por un selector de normas de mapeo 828, que deriva una información de las normas de mapeo 828a de la información del estado contextual actual 826a, y que provee la información de las normas de mapeo 828a al determinador de valores espectrales 824.

[00040] Respecto de la funcionalidad del decodificador de señales de audio 800, debe tenerse en cuenta que el decodificador aritmético 820 tiene una configuración que le permite seleccionar una norma de mapeo (p. ej. una tabla de frecuencias acumulativas) que se adapta bien, en promedio, al valor espectral que se ha de decodificar, ya que la norma de mapeo se selecciona de manera dependiente del estado contextual actual, que, a su vez, es determinado de manera dependiente de una pluralidad de valores espectrales previamente decodificados. En consecuencia, se pueden explotar las dependencias estadísticas entre valores espectrales adyacentes que se han de decodificar. Además, al detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, es posible adaptar la norma de mapeo a condiciones (o patrones) especiales de valores espectrales previamente decodificados. Por ejemplo, se puede seleccionar una norma de mapeo específica si se identifica un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados comparativamente pequeños, o si se identifica un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados comparativamente grandes. Se ha descubierto que la presencia de un grupo de valores espectrales comparativamente grandes o de un grupo de valores espectrales comparativamente pequeños puede ser considerada una significativa indicación de que debería usarse una norma de mapeo dedicada, adaptada específicamente a dicha condición. En consecuencia, se puede facilitar (o acelerar) una computación contextual mediante la explotación de la detección de dicho grupo de una pluralidad de valores espectrales. Asimismo, se pueden considerar características de un contenido de audio que no podrían ser consideradas tan fácilmente sin aplicar el concepto antes mencionado. Por ejemplo, se puede realizar la detección de un grupo de una pluralidad de valores espectrales que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, sobre la base de un conjunto diferente de valores espectrales, cuando se compara con el conjunto de valores espectrales utilizados para una computación contextual normal.

[00041] A continuación se describirán otros detalles adicionales.

3. Codificador de audio de acuerdo con la Fig. 1

[00042] A continuación se describirá un codificador de audio de acuerdo con una forma de realización de la invención. La Fig. 1 muestra un diagrama esquemático en bloque de dicho codificador de audio 100.

[00043] El codificador de audio 100 tiene una configuración que le permite recibir una información de audio entrante 110 y proveer, sobre esta base, una corriente de bits 112, que constituye una información de audio codificada. El

codificador de audio 100 comprende opcionalmente un preprocesador 120, que tiene una configuración que le permite recibir la información de audio entrante 110 y proveer, sobre esta base, una información de audio entrante preprocesada 110a. El codificador de audio 100 también comprende un transformador de señales de dominio temporal a dominio frecuencial que compacta la energía 130, que también recibe el nombre de convertidor de señales. El convertidor de señales 130 tiene una configuración que le permite recibir la información de audio entrante 110, 110a y proveer, sobre esta base, una información de audio de dominio frecuencial 132, que, de preferencia, adopta la forma de un conjunto de valores espectrales. Por ejemplo, el transformador de señales 130 puede tener una configuración que le permite recibir un marco de la información de audio entrante 110, 110a (p. ej. un bloque de muestras de dominio temporal) y proveer un conjunto de valores espectrales que representan el contenido de audio del respectivo marco de audio. Además, el transformador de señales 130 puede tener una configuración que le permite recibir una pluralidad de marcos de audio consecutivos, que se superponen o no, de la información de audio entrante 110, 110a y proveer, sobre esta base, una representación de audio de dominio tiempo-frecuencia, que comprende una secuencia de conjuntos consecutivos de valores espectrales, un conjunto de valores espectrales asociados con cada marco.

[00044] El transformador de señales de dominio temporal a dominio frecuencial que compacta la energía 130 puede comprender un banco de filtros que compacta la energía, que provee valores espectrales asociados con rangos de frecuencia diferentes, superpuestos o no superpuestos. Por ejemplo, el transformador de señales 130 puede comprender un transformador de MDCT de ventanas 130a, que tiene una configuración que le permite sistematizar en ventanas la información de audio entrante 110, 110a (o un marco de ésta) por medio de una ventana de transformación y llevar a cabo una transformación discreta del coseno modificada, de la información de audio entrante sistematizada en ventanas 110, 110a (o del marco sistematizado en ventanas de ésta). En consecuencia, la representación de audio de dominio frecuencial 132 puede comprender un conjunto, por ejemplo, 1024, de valores espectrales en forma de coeficientes de MDCT asociados con un marco de la información de audio entrante.

[00045] El codificador de audio 100 puede comprender además, como opción, un postprocesador espectral 140, que tiene una configuración que le permite recibir la representación de audio de dominio frecuencial 132 y proveer, sobre esta base, una representación de audio de dominio frecuencial postprocesada 142. El postprocesador espectral 140 puede tener, por ejemplo, una configuración que le permite realizar un perfilado de ruido temporal y/o una predicción a largo plazo y/o cualquier otro postprocesamiento espectral conocido en la técnica. El codificador de audio comprende además, opcionalmente, un escalador/cuantificador 150, que tiene una configuración que le permite recibir la representación de audio de dominio frecuencial 132 o la versión postprocesada 142 de ésta y proveer una representación de audio de dominio frecuencial escalada y cuantificada 152.

[00046] El codificador de audio 100 comprende además, opcionalmente, un procesador de modelos psicoacústicos 160, que tiene una configuración que le permite recibir la información de audio entrante 110 (o la versión postprocesada 110a de aquella) y proveer, sobre esta base, una información de control opcional, que puede ser usada para controlar el transformador de señales de dominio temporal a dominio frecuencial que compacta la energía 130, para controlar el postprocesador espectral opcional 140 y/o para controlar el escalador/cuantificador 150. Por ejemplo, el procesador de modelos psicoacústicos 160 puede tener una configuración que le permite analizar la información de audio entrante, determinar qué componentes de la información de audio entrante 110, 110a son especialmente importantes para la percepción humana del contenido de audio y qué componentes de la información de audio entrante 110, 110a son menos importantes para la percepción del contenido de audio. En consecuencia, el procesador de modelos psicoacústicos 160 puede proveer información de control, que es utilizada por el codificador de audio 100 para ajustar la escala de la representación de audio de dominio frecuencial 132, 142 mediante el escalador/cuantificador 150 y/o la resolución de cuantificación aplicada por el escalador/cuantificador 150. En consecuencia, las bandas de factores de escala importantes para la percepción (es decir, los grupos de valores espectrales adyacentes que son especialmente importantes para la percepción humana del contenido de audio) son graduados con un factor de escala enorme y cuantificados con resolución comparativamente alta, mientras que las bandas de factores de escala menos importantes para la percepción (es decir, los grupos de valores espectrales adyacentes) son graduados con un factor de escala comparativamente más pequeño y cuantificados con una resolución de cuantificación comparativamente menor. En consecuencia, los valores espectrales graduados de las frecuencias más importantes desde el punto de vista de la percepción generalmente son significativamente más altos que los valores espectrales de las frecuencias menos importantes desde el punto de vista de la percepción.

[00047] El codificador de audio también comprende un codificador aritmético 170, que tiene una configuración que le permite recibir la versión escalada y cuantificada 152 de la representación de audio de dominio frecuencial 132 (o, alternativamente, la versión postprocesada 142 de la representación de audio de dominio frecuencial 132, o incluso la representación misma de audio de dominio frecuencial 132) y proveer información de las palabras de código aritmético 172a sobre esta base, de tal modo que la información de las palabras de código aritmético representa la representación de audio de dominio frecuencial 152.

[00048] El codificador de audio 100 también comprende un formateador de carga útil de una corriente de bits 190, que tiene una configuración que le permite recibir la información de las palabras de código aritmético 172a. El formateador de carga útil de una corriente de bits 190 también tiene, por lo general, una configuración que le permite

recibir información adicional como, por ejemplo, información de factores de escala que describen cuáles factores de escala han sido aplicados por el escalador/cuantificador 150. Además, el formateador de carga útil de una corriente de bits 190 puede tener una configuración que le permite recibir otra información de control. El formateador de carga útil de una corriente de bits 190 tiene una configuración que le permite proveer la corriente de bits 112 sobre la base de la información recibida ensamblando la corriente de bits de acuerdo con una sintaxis deseada de corriente de bits, que se analizará más adelante.

[00049] A continuación se describirán detalles relacionados con el codificador aritmético 170. El codificador aritmético 170 tiene una configuración que le permite recibir una pluralidad de valores espectrales postprocesados y escalados y cuantificados de la representación de audio de dominio frecuencial 132. El codificador aritmético comprende un extractor de planos de bits más significativos 174, que tiene una configuración que le permite extraer un plano de bits más significativos *m* de un valor espectral. En este punto debe hacerse notar que el plano de bits más significativos puede comprender uno o incluso más bits (p. ej. dos o tres bits), que son los bits más significativos del valor espectral. Por consiguiente, el extractor de planos de bits más significativos 174 proporciona un valor del plano de bits más significativos 176 de un valor espectral.

[00050] El codificador aritmético 170 también comprende un primer determinador de palabras de código 180, que tiene una configuración que le permite determinar una palabra de código aritmético *acod_m* [*pki*][*m*] que representa el valor del plano de bits más significativos *m*. Opcionalmente, el determinador de palabras de código 180 también puede proveer una o varias palabras de código de escape (también denominadas aquí "ARITH_ESCAPE") que indican, por ejemplo, cuántos planos de bits menos significativos hay disponibles (y, en consecuencia, indican el peso numérico del plano de bits más significativos). El primer determinador de palabras de código 180 puede tener una configuración que le permite proveer la palabra de código asociada con un valor del plano de bits más significativos *m* usando una tabla escogida de frecuencias acumulativas que tiene (o que está en referencia con) un índice de tabla de frecuencias acumulativas *pki*.

[00051] Para determinar cuál tabla de frecuencias acumulativas debería escogerse, el codificador aritmético comprende, preferiblemente, un rastreador de estados 182, que tiene una configuración que le permite rastrear el estado del codificador aritmético, por ejemplo, mediante la observación de cuáles valores espectrales han sido codificados previamente. El rastreador de estados 182 proporciona, en consecuencia, una información de estado 184, por ejemplo, un valor de estado designado con "s" o "t". El codificador aritmético 170 también comprende un selector de tablas de frecuencias acumulativas 186, que tiene una configuración que le permite recibir la información de estado 184 y proveer una información 188 que describe la tabla de frecuencias acumulativas escogida al determinador de palabras de código 180. Por ejemplo, el selector de tablas de frecuencias acumulativas 186 puede proveer un índice de tablas de frecuencias acumulativas „*pki*“ que describe cuál tabla de frecuencias acumulativas escoger, de un conjunto de 64 tablas de frecuencias acumulativas, para que utilice el determinador de palabras de código. Alternativamente, el selector de tablas de frecuencias acumulativas 186 puede proporcionar la tabla completa de frecuencias acumulativas escogida al determinador de palabras de código. Por consiguiente, el determinador de palabras de código 180 puede usar la tabla de frecuencias acumulativas escogida para proporcionar la palabra de código *acod_m* [*pki*][*m*] del valor del plano de bits más significativos *m*, de tal modo que la palabra de código actual *acod_m* [*pki*][*m*] que codifica el valor del plano de bits más significativos *m* depende del valor de *m* y el índice de tablas de frecuencias acumulativas *pki* y, en consecuencia, de la información del estado actual 184. A continuación se describirán más detalles relacionados con el proceso de codificación y el formato de palabras de código obtenido.

[00052] El codificador aritmético 170 comprende además un extractor de planos de bits menos significativos 189a, que tiene una configuración que le permite extraer uno o varios planos de bits menos significativos de la representación de audio de dominio frecuencial escalada y cuantificada 152, si uno o varios de los valores espectrales a codificar exceden el intervalo de valores codificable usando solamente el plano de bits más significativos. Los planos de bits menos significativos pueden comprender uno o varios bits, según se desee. En consecuencia, el extractor de planos de bits menos significativos 189a provee una información de los planos de bits menos significativos 189b. El codificador aritmético 170 también comprende un segundo determinador de palabras de código 189c, que tiene una configuración que le permite recibir la información de los planos de bits menos significativos 189d y proveer, sobre esta base, 0, 1 o más palabras de código "acod_r" que representan el contenido de 0, 1 o más planos de bits menos significativos. El segundo determinador de palabras de código 189c puede tener una configuración que le permite aplicar un algoritmo de codificación aritmética o cualquier otro algoritmo de codificación para derivar las palabras de código del plano de bits menos significativos "acod_r" de la información de los planos de bits menos significativos 189b.

[00053] En este punto debe hacerse notar que la cantidad de planos de bits menos significativos puede variar según el valor de los valores espectrales escalados y cuantificados 152, de modo que podría no haber ningún plano de bits menos significativos en absoluto, si el valor espectral escalado y cuantificado a codificar es comparativamente pequeño, de modo que podría haber un plano de bits menos significativos si el valor espectral escalado y cuantificado actual a codificar pertenece a un intervalo medio y de modo que podría haber más de un plano de bits menos significativos si el valor espectral escalado y cuantificado a codificar alcanza un valor comparativamente grande.

[00054] Para resumir lo anterior, el codificador aritmético 170 tiene una configuración que le permite codificar valores espectrales escalados y cuantificados, que son descriptos por la información 152, mediante un proceso de codificación jerárquico. El plano de bits más significativos (que comprende, por ejemplo, uno, dos o tres bits por valor espectral) está codificado para obtener una palabra de código aritmético "acod_m[pki][m]" de un valor del plano de bits más significativos. Uno o varios planos de bits menos significativos (cada uno de los planos de bits menos significativos comprende, por ejemplo, uno, dos o tres bits) están codificados para obtener una o varias palabras de código "acod_r". Cuando se codifica el plano de bits más significativos, el valor **m** del plano de bits más significativos es mapeado a una palabra de código acod_m[pki][m]. Con esta finalidad, se dispone de 64 tablas de frecuencias acumulativas diferentes para codificar el valor **m** en forma dependiente de un estado del codificador aritmético 170, es decir, de manera dependiente de los valores espectrales previamente codificados. En consecuencia, se obtiene la palabra de código "acod_m[pki][m]". Además, se proporcionan una o varias palabras de código "acod_r" y se las incluye en la corriente de bits si se encuentran presentes uno o varios planos de bits menos significativos.

Descripción de la reiniciación

[00055] El codificador de audio 100 puede tener opcionalmente una configuración que le permite decidir si se puede obtener una mejora en la velocidad de transmisión de bits reiniciando el contexto, por ejemplo ajustando el índice de estados a un valor por defecto. En consecuencia, el codificador de audio 100 puede tener una configuración que le permite proveer una información de reiniciación (p. ej. llamada "arith_reset_flag") que indica si el contexto para la codificación aritmética está reiniciado, y que también indica si el contexto para la decodificación aritmética en un decodificador correspondiente debería ser reiniciado.

[00056] A continuación se describirán detalles relacionados con el formato de la corriente de bits y las tablas de frecuencias acumulativas aplicadas.

4. Decodificador de audio

[00057] A continuación, se describirá un decodificador de audio de acuerdo con una forma de realización de la invención. La Fig. 2 muestra un diagrama esquemático en bloque de dicho decodificador de audio 200.

[00058] El decodificador de audio 200 tiene una configuración que le permite recibir una corriente de bits 210, que representa una información de audio codificada y que puede ser idéntica a la corriente de bits 112 provista por el codificador de audio 100. El decodificador de audio 200 provee información de audio decodificada 212 sobre la base de la corriente de bits 210.

[00059] El decodificador de audio 200 comprende un desformateador de la carga útil de la corriente de bits opcional 220, que tiene una configuración que le permite recibir la corriente de bits 210 y extraer de la corriente de bits 210 una representación de audio de dominio frecuencial codificada 222. Por ejemplo, el desformateador de la carga útil de la corriente de bits 220 puede tener una configuración que le permite extraer de la corriente de bits 210 datos espectrales codificados aritméticamente como, por ejemplo, una palabra de código aritmético "acod_m[pki][m]" que representa el valor del plano de bits más significativo **m** de un valor espectral **a**, y una palabra de código "acod_r" que representa un contenido de un plano de bits menos significativo del valor espectral **a** de la representación de audio de dominio frecuencial. Por consiguiente, la representación de audio de dominio frecuencial codificada 222 constituye (o comprende) una representación aritméticamente codificada de valores espectrales. El desformateador de la carga útil de la corriente de bits 220 tiene además una configuración que le permite extraer de la corriente de bits información adicional de control, que no aparece ilustrada en la Fig. 2. Además, el desformateador de la carga útil de la corriente de bits tiene opcionalmente una configuración que le permite extraer de la corriente de bits 210 una información de reiniciación de estados 224, también denominada bandera de reiniciación aritmética o "arith_reset_flag".

[00060] El decodificador de audio 200 comprende un decodificador aritmético 230, también denominado "decodificador espectral insonorizado". El decodificador aritmético 230 tiene una configuración que le permite recibir la representación de audio de dominio frecuencial codificada 220 y, opcionalmente, la información de reiniciación de estados 224. El decodificador aritmético 230 también tiene una configuración que le permite proveer una representación de audio de dominio frecuencial decodificada 232, que puede comprender una representación decodificada de valores espectrales. Por ejemplo, la representación de audio de dominio frecuencial decodificada 232 puede comprender una representación decodificada de valores espectrales, que son descriptos por la representación de audio de dominio frecuencial codificada 220.

[00061] El decodificador de audio 200 también comprende un cuantificador/reescalador inverso opcional 240, que tiene una configuración que le permite recibir la representación de audio de dominio frecuencial decodificada 232 y proveer, sobre esta base, una representación de audio de dominio frecuencial inversamente cuantificada y reescalada 242.

[00062] El decodificador de audio 200 comprende además un preprocesador espectral opcional 250, que tiene una configuración que le permite recibir la representación de audio de dominio frecuencial inversamente cuantificada y reescalada 242 y proveer, sobre esta base, una versión preprocesada 252 de la representación de audio de dominio frecuencial inversamente cuantificada y reescalada 242. El decodificador de audio 200 también comprende un transformador de señales de dominio frecuencial a dominio temporal 260, también denominado “convertidor de señales”. El transformador de señales 260 tiene una configuración que le permite recibir la versión preprocesada 252 de la representación de audio de dominio frecuencial inversamente cuantificada y reescalada 242 (o, alternativamente, la representación de audio de dominio frecuencial inversamente cuantificada y reescalada 242 o la representación de audio de dominio frecuencial decodificada 232) y proveer, sobre esta base, una representación de dominio temporal 262 de la información de audio. El transformador de señales de dominio frecuencial a dominio temporal 260 puede comprender, por ejemplo, un transformador para ejecutar una transformada discreta del coseno modificada inversa (IMDCT) y una sistematización en ventanas apropiada (además de otras funcionalidades auxiliares como, por ejemplo, una superposición y adición).

[00063] El decodificador de audio 200 puede comprender además un postprocesador de dominio temporal opcional 270, que tiene una configuración que le permite recibir la representación de dominio temporal 262 de la información de audio y obtener la información de audio decodificada 212 usando un de dominio temporal. Sin embargo, si se omite el postprocesamiento, la representación de dominio temporal 262 puede ser idéntica a la información de audio decodificada 212.

[00064] En este punto debe hacerse notar que el cuantificador/reescalador inverso 240, el preprocesador espectral 250, el transformador de señales de dominio frecuencial a dominio temporal 260 y el postprocesador de dominio temporal 270 pueden ser controlados en forma dependiente de la información de control, que es extraída de la corriente de bits 210 por el desformateador de la carga útil de la corriente de bits 220.

[00065] Para resumir la forma de funcionamiento general del decodificador de audio 200, se puede obtener una representación de audio de dominio frecuencial decodificada 232, por ejemplo, un conjunto de valores espectrales asociados con un marco de audio de la información de audio codificada, sobre la base de la representación de dominio frecuencial codificada 222 usando el decodificador aritmético 230. A continuación, se cuantifica inversamente, reescala y preprocesa el conjunto, por ejemplo, de 1024 valores espectrales, que pueden ser los coeficientes de MDCT. En consecuencia, se obtiene un conjunto de valores espectrales cuantificado inversamente, reescalado y preprocesado espectralmente (p. ej., 1024 coeficientes de MDCT). Luego, se deriva una representación de dominio temporal de un marco de audio del conjunto de valores de dominio frecuencial cuantificado inversamente, reescalado y preprocesado espectralmente (p. ej. coeficientes de MDCT). En consecuencia, se obtiene una representación de dominio temporal de un marco de audio. La representación de dominio temporal de un determinado marco de audio puede combinarse con representaciones de dominio temporal de marcos de audio previos y/o posteriores. Por ejemplo, puede realizarse una superposición y adición entre representaciones de dominio temporal de marcos de audio posteriores para filtrar las transiciones entre las representaciones de dominio temporal de los marcos de audio adyacentes y para obtener una cancelación de las distorsiones (*aliasing*). En cuanto a los detalles relacionados con la reconstrucción de la información de audio decodificada 212 sobre la base de la representación de audio de dominio tiempo–frecuencia decodificada 232, se hace referencia, por ejemplo, a la Norma Internacional ISO/IEC 14496-3, parte 3, sub-parte 4 donde se proporciona un análisis detallado. Sin embargo, se pueden usar otros esquemas más elaborados de superposición y cancelación de las distorsiones (*aliasing*).

[00066] A continuación, se describirán algunos detalles relacionados con el decodificador aritmético 230. El decodificador aritmético 230 comprende un determinador de planos de bits más significativos 284, que tiene una configuración que le permite recibir la palabra de código aritmético `acod_m [pkj][m]` que describe el valor del plano de bits más significativos `m`. El determinador de planos de bits más significativos 284 puede tener una configuración que le permite usar una tabla de frecuencias acumulativas de un conjunto que comprende una pluralidad de 64 tablas de frecuencias acumulativas para derivar el valor del plano de bits más significativos `m` de la palabra de código aritmético “`acod_m [pkj][m]`”.

[00067] El determinador de planos de bits más significativos 284 tiene una configuración que le permite derivar valores 286 de un plano de bits más significativos de valores espectrales sobre la base de la palabra de código `acod_m`. El decodificador aritmético 230 comprende además un determinador de planos de bits menos significativos 288, que tiene una configuración que le permite recibir una o varias palabras de código “`acod_r`” que representan uno o varios planos de bits menos significativos de un valor espectral. En consecuencia, el determinador de planos de bits menos significativos 288 tiene una configuración que le permite proveer valores decodificados 290 de uno o varios planos de bits menos significativos. El decodificador de audio 200 también comprende un combinador de planos de bits 292, que tiene una configuración que le permite recibir los valores decodificados 286 del plano de bits más significativos de los valores espectrales y los valores decodificados 290 de uno o varios planos de bits menos significativos de los valores espectrales si dichos planos de bits menos significativos están disponibles para los valores espectrales actuales. En consecuencia, el combinador de planos de bits 292 provee valores espectrales decodificados, que forman parte de la representación de audio de dominio frecuencial decodificada 232. Naturalmente, el decodificador aritmético 230 tiene habitualmente una configuración que le permite proveer una

pluralidad de valores espectrales para obtener un conjunto completo de valores espectrales decodificados asociados con un marco actual del contenido de audio.

[00068] El decodificador aritmético 230 comprende además un selector de tablas de frecuencias acumulativas 296, que tiene una configuración que le permite seleccionar una de las 64 tablas de frecuencias acumulativas en forma dependiente de un índice de estados 298 que describe un estado del decodificador aritmético. El decodificador aritmético 230 comprende además un rastreador de estados 299, que tiene una configuración que le permite rastrear un estado del decodificador aritmético dependiente de los valores espectrales previamente decodificados. La información de estados puede reiniciarse opcionalmente a una información de estados por defecto en respuesta a la información de reiniciación de estados 224. En consecuencia, el selector de tablas de frecuencias acumulativas 296 tiene una configuración que le permite proveer un índice (p. ej. pki) de una tabla de frecuencias acumulativas escogida, o una tabla de frecuencias acumulativas escogida por sí misma, para aplicar en la decodificación del valor del plano de bits más significativos **m** dependiente de la palabra de código "acod_m".

[00069] Para resumir la forma de funcionamiento del decodificador de audio 200, el decodificador de audio 200 tiene una configuración que le permite recibir una representación de audio de dominio frecuencial eficientemente codificada por la velocidad de transmisión de bits 222 y obtener una representación de audio de dominio frecuencial decodificada sobre esta base. En el decodificador aritmético 230, que se emplea para obtener la representación de audio de dominio frecuencial decodificada 232 sobre la base de la representación de audio de dominio frecuencial codificada 222, se explota una probabilidad de distintas combinaciones de valores del plano de bits más significativos de valores espectrales adyacentes mediante la utilización de un decodificador aritmético 280, que tiene una configuración que le permite aplicar una tabla de frecuencias acumulativas. En otras palabras, se explotan las dependencias estadísticas entre valores espectrales seleccionando distintas tablas de frecuencias acumulativas de un conjunto que comprende 64 tablas de frecuencias acumulativas diferentes en forma dependiente de un índice de estados 298, que se obtiene observando los valores espectrales decodificados computados previamente.

5. Resumen sobre la herramienta de codificación espectral insonorizada

[00070] A continuación, se explicarán detalles sobre el algoritmo de codificación y decodificación, que es ejecutado, por ejemplo, por el codificador aritmético 170 y el decodificador aritmético 230.

[00071] Se pone el acento en la descripción del algoritmo de decodificación. Debe notarse, sin embargo, que un algoritmo de codificación correspondiente puede ejecutarse de conformidad con lo que se establece sobre el algoritmo de decodificación, en el que se invierten los mapeos.

[00072] Debe tenerse en cuenta que la decodificación, que se analizará a continuación, se usa para permitir lo que se llama "codificación espectral insonorizada" de valores espectrales normalmente postprocesados, escalados y cuantificados. La codificación espectral insonorizada se usa en un concepto de codificación/decodificación de audio para disminuir la redundancia del espectro cuantificado, que se obtiene, por ejemplo, por medio de un transformador de dominio temporal a dominio frecuencial compactador de la energía.

[00073] El esquema de codificación espectral insonorizada, que se usa en las formas de realización de la invención, se basa en una codificación aritmética en conjunción con un contexto adaptado dinámicamente. La codificación insonorizada es suministrada por (representaciones originales o codificadas de) valores espectrales cuantificados y utiliza tablas de frecuencias acumulativas dependientes del contexto, derivadas, por ejemplo, de una pluralidad de valores espectrales próximos previamente decodificados. En la presente se tiene en cuenta la proximidad tanto en tiempo como en frecuencia, tal como se ilustra en la Fig. 4. Las tablas de frecuencias acumulativas (que serán explicadas más adelante) son usadas entonces por el codificador aritmético para generar un código binario de longitud variable y por el decodificador aritmético para derivar valores decodificados desde un código binario de longitud variable.

[00074] Por ejemplo, el codificador aritmético 170 produce un código binario para un determinado conjunto de símbolos de manera dependiente de las respectivas probabilidades. El código binario es generado por el mapeo de un intervalo de probabilidades, donde se encuentra el conjunto de símbolos, a una palabra de código.

[00075] A continuación, se brindará otra breve reseña de la herramienta de codificación espectral insonorizada. La codificación espectral insonorizada se usa para disminuir aún más la redundancia del espectro cuantificado. El esquema de codificación espectral insonorizada se basa en una codificación aritmética en conjunción con un contexto adaptado dinámicamente. La codificación insonorizada es suministrada por los valores espectrales cuantificados y utiliza tablas de frecuencias acumulativas dependientes del contexto derivadas, por ejemplo, de siete valores espectrales próximos previamente decodificados.

[00076] En la presente se tiene en cuenta la proximidad tanto en tiempo como en frecuencia, tal como se ilustra en la Fig. 4. Las tablas de frecuencias acumulativas son usadas entonces por el codificador aritmético para generar un código binario de longitud variable.

[00077] El codificador aritmético produce un código binario para un determinado conjunto de símbolos y sus respectivas probabilidades. El código binario es generado por el mapeo de un intervalo de probabilidades, donde se encuentra el conjunto de símbolos, a una palabra de código.

6. Proceso de decodificación

6.1 Resumen sobre el proceso de decodificación

[00078] A continuación, se presenta un resumen del proceso de decodificación de un valor espectral con respecto a la Fig. 3, que muestra una representación de pseudocódigo de programa del proceso de decodificación de una pluralidad de valores espectrales.

[00079] El proceso de decodificación de una pluralidad de valores espectrales comprende una inicialización 310 de un contexto. La inicialización 310 del contexto comprende una derivación del contexto actual a partir de un contexto previo utilizando la función "arith_map_context (lg)". La derivación del contexto actual a partir de un contexto previo puede comprender una reiniciación del contexto. Tanto la reiniciación del contexto como la derivación del contexto actual a partir de un contexto previo serán discutidos a continuación.

[00080] La decodificación de una pluralidad de valores espectrales también comprende una iteración de una decodificación de valores espectrales 312 y una actualización contextual 314, actualización contextual que es ejecutada por una función "Arith_update_context(a,i,lg)" que se describirá a continuación. La decodificación de valores espectrales 312 y la actualización contextual 314 se repiten **lg** veces, donde **lg** indica la cantidad de valores espectrales que se han de decodificar (p. ej. para un marco de audio). La decodificación de valores espectrales 312 comprende un cálculo de valores contextuales 312a, una decodificación de planos de bits más significativos 312b, y una adición de planos de bits menos significativos 312c.

[00081] La computación de valores de estado 312a comprende la computación de un primer valor de estado **s** utilizando la función "arith_get_context(i, lg, arith_reset_flag, N/2)", función que retorna el primer valor de estado **s**. La computación de valores de estado 312a también comprende una computación de un valor de nivel "lev0" y de un valor de nivel "lev", valores de nivel "lev0", "lev" que se obtienen haciendo desplazar el primer valor de estado **s** hacia la derecha en 24 bits. La computación de valores de estado 312a también comprende una computación de un segundo valor de estado **t** de acuerdo con la fórmula que se muestra en la Fig. 3 con el número de referencia 312a.

[00082] La decodificación de los planos de bits más significativos 312b comprende una ejecución iterativa de un algoritmo de decodificación 312ba, donde una variable **j** es inicializada a 0 antes de una primera ejecución del algoritmo 312ba.

[00083] El algoritmo 312ba comprende una computación de un índice de estados "pki" (que también sirve como índice de tablas de frecuencias acumulativas) en forma dependiente del segundo valor de estado **t**, y también en forma dependiente de los valores de nivel "lev" y lev0, usando una función "arith_get_pk()", que se analizará a continuación. El algoritmo 312ba también comprende la selección de una tabla de frecuencias acumulativas en forma dependiente del índice de estados pki, donde una variable "cum_freq" puede ser establecida a una dirección de comienzo de una de 64 tablas de frecuencias acumulativas en forma dependiente del índice de estados pki. Asimismo, se puede inicializar una variable "cfl" a un tramo de la tabla de frecuencias acumulativas escogida, que es, por ejemplo, igual a la cantidad de símbolos del abecedario, es decir, la cantidad de valores diferentes que pueden decodificarse. Las longitudes de todas las tablas de frecuencias acumulativas desde "arith_cf_m[pki=0][9]" hasta "arith_cf_m[pki=63][9]" disponibles para la decodificación del valor del plano de bits más significativos **m** es 9, ya que se pueden decodificar ocho valores diferentes del plano de bits más significativos y un símbolo de escape. Seguidamente, se puede obtener un valor del plano de bits más significativos **m** ejecutando una función "arith_decode()", tomando en cuenta la tabla de frecuencias acumulativas escogida (descrita por la variable "cum_freq" y la variable "cfl"). Cuando se está derivando el valor del plano de bits más significativos **m**, se pueden evaluar los bits denominados "acod_m" de la corriente de bits 210 (véase, por ejemplo, la Fig. 6g).

[00084] El algoritmo 312ba también comprende la verificación de que el valor del plano de bits más significativos **m** es igual a un símbolo de escape "ARITH_ESCAPE", o no. Si el valor del plano de bits más significativos **m** no es igual al símbolo de escape aritmético, el algoritmo 312ba es abortado (condición "break") y, por lo tanto, se pasa por alto el resto de las instrucciones del algoritmo 312ba. En consecuencia, se continúa la ejecución del proceso estableciendo que el valor espectral **a** es igual al valor del plano de bits más significativos **m** (instrucción "a=m"). Por el contrario, si el valor decodificado del plano de bits más significativos **m** es idéntico al símbolo de escape aritmético "ARITH_ESCAPE", el valor de nivel "lev" aumenta en uno. Como ya se indicó, el algoritmo 312ba se repite entonces hasta que el valor decodificado del plano de bits más significativos **m** es diferente del símbolo de escape aritmético.

[00085] En cuanto se completa la decodificación de los planos de bits más significativos, es decir, cuando se ha decodificado un valor del plano de bits más significativos **m** diferente del símbolo de escape aritmético, se establece que la variable de valor espectral "a" sea igual al valor del plano de bits más significativos **m**. A continuación se obtienen los planos de bits menos significativos, por ejemplo, como se indica con el número de referencia 312c en la

Fig. 3. Para cada plano de bits menos significativos del valor espectral, se decodifica uno de dos valores binarios. Por ejemplo, se obtiene un valor del plano de bits menos significativos *r*. A continuación se actualiza la variable de valor espectral „a“ desplazando el contenido de la variable de valor espectral „a“ hacia la izquierda en 1 bit y agregando el valor del plano de bits menos significativos actualmente decodificado *r* como el bit menos significativo. Sin embargo, debe tenerse en cuenta que el concepto para obtener los valores de los planos de bits menos significativos no es de particular importancia para la presente invención. En algunas formas de realización, se puede incluso omitir la decodificación de cualquier plano de bits menos significativos. Alternativamente, se pueden usar diferentes algoritmos de decodificación con esta finalidad.

6.2 Orden de decodificación de acuerdo con la Fig. 4

[00086] A continuación, se describirá el orden de decodificación de los valores espectrales.

[00087] Los coeficientes espectrales se codifican y transmiten de manera insonorizada (p. ej. en la corriente de bits) comenzando con el coeficiente de menor frecuencia y avanzando progresivamente hasta el coeficiente de mayor frecuencia.

[00088] Los coeficientes de una codificación de audio avanzada (que se obtiene, por ejemplo, usando una transformada discreta del coseno modificada, como se discute en la ISO/IEC 14496, parte 3, subparte 4) son almacenados en una red denominada „x_ac_quant[g][win][sfb][bin]“, y el orden de transmisión de la palabra de código de codificación insonorizada (p. ej. acod_m, acod_r) es tal que cuando son decodificados en el orden recibido y almacenados en la red, „bin“ (el índice de frecuencias) es el índice de incrementación más rápida y „g“ es el índice de incrementación menos rápida.

[00089] Los coeficientes espectrales asociados con una frecuencia menor son codificados antes que los coeficientes espectrales asociados con una frecuencia más alta.

[00090] Los coeficientes de la excitación codificada por la transformada (tcx) son almacenados directamente en una red x_tcx_invquant[win][bin], y el orden de la transmisión de las palabras de código de codificación insonorizada es tal que cuando son decodificadas en el orden recibido y almacenadas en la red, „bin“ es el índice de incrementación más rápida y „win“ es el índice de incrementación menos rápida. En otras palabras, si los valores espectrales describen una excitación codificada por la transformada del filtro de predicción lineal de un codificador de lenguaje, los valores espectrales *a* están asociados a frecuencias adyacentes y crecientes de la excitación codificada por la transformada.

[00091] Los coeficientes espectrales asociados a una frecuencia menor son codificados antes que los coeficientes espectrales asociados con una frecuencia más alta.

[00092] Es notable que el decodificador de audio 200 puede tener una configuración que le permite aplicar la representación de audio de dominio frecuencial decodificada 232, provista por el decodificador aritmético 230, tanto para una generación „directa“ de una representación de señales de audio de dominio temporal que usa una transformación de señales de dominio frecuencial a dominio temporal como para una provisión „indirecta“ de una representación de señales de audio que emplea, además de un decodificador de dominio frecuencial a dominio temporal, un filtro de predicción lineal excitado por la salida del transformador de señales de dominio frecuencial a dominio temporal.

[00093] En otras palabras, el decodificador aritmético 200, cuya funcionalidad se analiza minuciosamente en la presente, es perfectamente adecuado para decodificar valores espectrales de una representación de dominio tiempo–frecuencia de un contenido de audio codificado en el dominio frecuencial y para proveer una representación de dominio tiempo–frecuencia de una señal de estímulo para un filtro de predicción lineal adaptado para decodificar una señal de lenguaje codificada en el dominio de predicción lineal. Por consiguiente, el decodificador aritmético es perfectamente adecuado para usar en un decodificador de audio que puede manejar tanto el contenido de audio codificado por el dominio frecuencial como el contenido de audio codificado por el dominio frecuencial predictivo lineal (modo de dominio de predicción lineal de excitaciones codificadas por la transformada).

6.3. Inicialización del contexto de acuerdo con las Figs. 5a y 5b

[00094] A continuación, se describirá la inicialización del contexto (también denominada „mapeo contextual“), que se realiza en una etapa 310.

[00095] La inicialización del contexto comprende un mapeo entre un contexto pasado y un contexto actual de acuerdo con el algoritmo „arith_map_context()“, que se ilustra en la Fig. 5a. Según puede observarse, el contexto actual es almacenado en una variable global q[2][n_context] que adopta la forma de una red que tiene una primera dimensión de dos y una segunda dimensión de n_context. Un contexto pasado es almacenado en una variable qs[n_context], que adopta la forma de una tabla que tiene una dimensión de n_context. La variable „previous_lg“ describe una serie de valores espectrales de un contexto pasado.

[00096] La variable "lg" describe una serie de coeficientes espectrales para decodificar en el marco. La variable "previous_lg" describe una cantidad previa de líneas espectrales de un marco previo.

[00097] Se puede ejecutar un mapeo del contexto de acuerdo con el algoritmo "arith_map_context()". En este punto debe hacerse notar que la función "arith_map_context()" ajusta las entradas $q[0][i]$ de la red de contextos actuales q a los valores $qs[i]$ de la red de contextos pasados qs , si la cantidad de valores espectrales asociados con el marco de audio actual (p. ej. codificados por el dominio frecuencial) es idéntico a la cantidad de valores espectrales asociados con el marco de audio previo para $i=0$ a $i=lg-1$.

[00098] Sin embargo, se realiza un mapeo más complicado si la cantidad de valores espectrales asociados al marco de audio actual es diferente de la cantidad de valores espectrales asociados al marco de audio previo. Sin embargo, los detalles relacionados con el mapeo en este caso no son de importancia especial para la idea clave de la presente invención, por lo que se hace referencia al pseudocódigo de programa de la Fig. 5a para los detalles.

6.4 Computación de los valores de estado de acuerdo con las Figs. 5b y 5c

[00099] A continuación, se describirá con más detalles la computación de los valores de estado 312a.

[000100] Debe tenerse en cuenta que el primer valor de estado s (ilustrado en la Fig. 3) puede obtenerse como valor de retorno de la función "arith_get_context(i, lg, arith_reset_flag, N/2)", del que se muestra una representación de pseudocódigo de programa en las Figs. 5b y 5c.

[000101] Con respecto a la computación del valor de estado, se hace referencia también a la Fig. 4, que muestra el contexto usado para una evaluación de estados. La Fig. 4 muestra una representación bidimensional de valores espectrales, tanto en función del tiempo como de la frecuencia. Una abscisa 410 describe el tiempo y una ordenada 412 describe la frecuencia. Según puede observarse en la Fig. 4, un valor espectral 420 a decodificar, está asociado a un índice de tiempo t_0 y un índice de frecuencia i . Según puede observarse, para el índice de tiempo t_0 , las tuplas que tienen índices de frecuencia $i-1$, $i-2$ e $i-3$ ya están decodificados para cuando se está por decodificar el valor espectral 420 que tiene el índice de frecuencia i . Según puede observarse en la Fig. 4, un valor espectral 430 que tiene un índice de tiempo t_0 y un índice de frecuencia $i-1$ ya está decodificado antes de que el valor espectral 420 sea decodificado, y el valor espectral 430 es considerado para el contexto que se usa para la decodificación del valor espectral 420. De modo similar, un valor espectral 434 que tiene un índice de tiempo t_0 y un índice de frecuencia $i-2$, ya está decodificado antes de que el valor espectral 420 sea decodificado, y el valor espectral 434 es considerado para el contexto que se usa para decodificar el valor espectral 420. De modo similar, un valor espectral 440 que tiene un índice de tiempo $t-1$ y un índice de frecuencia de $i-2$, un valor espectral 444 que tiene un índice de tiempo $t-1$ y un índice de frecuencia $i-1$, un valor espectral 448 que tiene un índice de tiempo $t-1$ y un índice de frecuencia i , un valor espectral 452 que tiene un índice de tiempo $t-1$ y un índice de frecuencia $i+1$, y un valor espectral 456 que tiene un índice de tiempo $t-1$ y un índice de frecuencia $i+2$, ya están decodificados antes de que el valor espectral 420 sea decodificado, y son considerados para la determinación del contexto, que se usa para decodificar el valor espectral 420. Los valores (coeficientes) espectrales ya decodificados en el momento en que el valor espectral 420 es decodificado y considerado para el contexto están ilustrados con cuadrados sombreados. Por el contrario, algunos otros valores espectrales ya decodificados (en el momento en que el valor espectral 420 es decodificado), que están representados por cuadrados con líneas entrecortadas, y otros valores espectrales, que todavía no están decodificados (en el momento en que el valor espectral 420 es decodificado) y que aparecen con círculos de líneas entrecortadas, no se utilizan para determinar el contexto para decodificar el valor espectral 420.

[000102] Sin embargo, debe tenerse en cuenta que algunos de estos valores espectrales, que no se usan para la computación "regular" (o "normal") del contexto para decodificar el valor espectral 420 pueden evaluarse, sin embargo, para una detección de una pluralidad de valores espectrales adyacentes previamente decodificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes.

[000103] Haciendo referencia ahora a las Figs. 5b y 5c, que muestran la funcionalidad de la función "arith_get_context()" en forma de un pseudocódigo de programa, se describirán algunos detalles adicionales relacionados con el cálculo del primer valor de contexto "s", que es ejecutado por la función "arith_get_context()".

[000104] Debe tenerse en cuenta que la función "arith_get_context()" recibe, como variables entrantes, un índice i del valor espectral que se ha de decodificar. El índice i normalmente es un índice de frecuencias. Una variable entrante lg describe un número (total) de coeficientes cuantificados esperados (para un marco de audio actual). Una variable N describe una cantidad de líneas de la transformación. Una bandera "arith_reset_flag" indica si el contexto debería ser reiniciado. La función "arith_get_context" provee, como valor de salida, una variable "t", que representa un índice de estados concatenados s y un nivel de planos de bits predicho lev0.

[000105] La función "arith_get_context()" utiliza variables con números enteros $a0$, $c0$, $c1$, $c2$, $c3$, $c4$, $c5$, $c6$, lev0, y "region".

[000106] La función "arith_get_context()" comprende como bloques funcionales principales, un primer procesamiento de reiniciación aritmética 510, una detección 512 de un grupo de una pluralidad de valores espectrales cero adyacentes previamente decodificados, un primer ajuste de variable 514, un segundo ajuste de variable 516, una adaptación de nivel 518, un ajuste de valores de región 520, una adaptación de nivel 522, una limitación de nivel 524, un procesamiento de reiniciación aritmética 526, un tercer ajuste de variable 528, un cuarto ajuste de variable 530, un quinto ajuste de variable 532, una adaptación de nivel 534, y una computación selectiva de valores de retorno 536.

[000107] En el primer procesamiento de reiniciación aritmética 510, se verifica que la bandera de reiniciación aritmética "arith_reset_flag" esté establecida, mientras que el índice del valor espectral que se ha de decodificar es igual a cero. En este caso, se retorna un valor contextual de cero, y se aborta la función.

[000108] En la detección 512 de un grupo de una pluralidad de valores espectrales cero previamente decodificados, que solamente se ejecuta si la bandera de reiniciación aritmética está inactiva y el índice i del valor espectral que se ha de decodificar es diferente de cero, una variable denominada "flag" (bandera) es inicializada a 1, indicada con el número de referencia 512a, y se determina una región de valor espectral que ha de ser evaluada, indicada con el número de referencia 512b. A continuación, la región de valores espectrales, que es determinada como se indica con el número de referencia 512b, es evaluada como se señala con el número de referencia 512c. Si resulta que hay una región suficiente de valores espectrales cero previamente decodificados, se retorna un valor contextual de 1, indicado con el número de referencia 512d. Por ejemplo, un límite de índice de frecuencia superior "lim_max" se ajusta a $i+6$, a menos que el índice i del valor espectral que se ha de decodificar sea cercano a un índice de frecuencia máximo $lg-1$, en cuyo caso se realiza un ajuste especial del límite de índice de frecuencia superior, indicado con el número de referencia 512b. Además, un límite de índice de frecuencia inferior "lim_min" se ajusta a -5 , a menos que el índice i del valor espectral que se ha de decodificar esté cercano a cero ($i+lim_min < 0$), en cuyo caso se realiza una computación especial del límite de índice de frecuencia inferior lim_min , indicado con el número de referencia 512b. Cuando se evalúa la región de valores espectrales determinada en el paso 512b, se realiza primero una evaluación para los índices de frecuencia negativos k entre el límite de índice de frecuencia inferior lim_min y cero. Para los índices de frecuencia k entre lim_min y cero, se verifica que por lo menos uno de los valores contextuales $q[0][k].c$ y $q[1][k].c$ es igual a cero. Sin embargo, si ambos valores contextuales $q[0][k].c$ y $q[1][k].c$ son diferentes de cero para cualquiera de los índices de frecuencia k entre lim_min y cero, se concluye que no existe un grupo suficiente de valores espectrales cero y la evaluación 512c se aborta. A continuación, se evalúan los valores contextuales $q[0][k].c$ para los índices de frecuencia entre cero y lim_max . Si resulta que alguno de los valores contextuales $q[0][k].c$ para cualquiera de los índices de frecuencia entre cero y lim_max es diferente de cero, se concluye que no existe un grupo suficiente de valores espectrales cero previamente decodificados, y la evaluación 512c se aborta. Sin embargo, si resulta que para cada índice de frecuencia k entre lim_min y cero, hay por lo menos un valor contextual $q[0][k].c$ o $q[1][k].c$ que es igual a cero y que hay un valor contextual cero $q[0][k].c$ para cada índice de frecuencia k entre cero y lim_max , se concluye que hay un grupo suficiente de valores espectrales cero previamente decodificados. En consecuencia, en este caso, se retorna un valor contextual de 1 para indicar esta condición, sin realizar cálculos adicionales. En otras palabras, se pasan por alto los cálculos 514, 516, 518, 520, 522, 524, 526, 528, 530, 532, 534, 536, si se identifica un grupo suficiente de una pluralidad de valores contextuales $q[0][k].c$, $q[1][k].c$ que tienen un valor de cero. En otras palabras, el valor contextual retornado, que describe el estado contextual (s), es determinado independientemente de los valores espectrales previamente decodificados en respuesta a la detección de que la condición predeterminada ha sido satisfecha.

[000109] Por el contrario, es decir, si no hay un grupo suficiente de valores contextuales $q[0][k].c$, $q[1][k].c$, que son cero, por lo menos algunos de los cálculos 514, 516, 518, 520, 522, 524, 526, 528, 530, 532, 534, 536 deben ejecutarse.

[000110] En el primer ajuste de variable 514, que se ejecuta en forma selectiva si (y sólo si) el índice i del valor espectral que se ha de decodificar es inferior a 1, la variable a_0 es inicializada para que tome el valor contextual $q[1][i-1]$, y la variable c_0 es inicializada para que tome el valor absoluto de la variable a_0 . La variable „lev0“ es inicializada para que tome el valor de cero. Luego, las variables „lev0“ y c_0 son incrementadas si la variable a_0 comprende un valor absoluto comparativamente grande, es decir, es inferior a -4 , o mayor o igual a 4 . El aumento de las variables „lev0“ y c_0 se ejecuta de manera iterativa, hasta que el valor de la variable a_0 llega a un rango entre -4 y 3 por medio de una operación de desplazamiento hacia la derecha (paso 514b).

[000111] Luego, las variables c_0 y „lev0“ son limitadas a los valores máximos de 7 y 3 , respectivamente (paso 514c).

[000112] Si el índice i del valor espectral que se ha de decodificar es igual a 1 y la bandera de reiniciación aritmética ("arith_reset_flag") está activa, se retorna un valor contextual, que se computa meramente sobre la base de las variables c_0 y lev0 (paso 514d). En consecuencia, solo se considera un único valor espectral decodificado previamente que tiene el mismo índice de tiempo que el valor espectral que se ha de decodificar y que tiene un índice de frecuencia que es menor, en 1 , que el índice de frecuencia i del valor espectral que se ha de decodificar, para la computación del contexto (paso 514d). Por el contrario, es decir, si no hay funcionalidad de reiniciación aritmética, la variable c_4 es inicializada (paso 514e).

[000113] En conclusión, en el primer ajuste de variable 514, las variables c_0 y „lev0“ son inicializadas en forma dependiente de un valor espectral decodificado previamente, decodificadas para el mismo marco como el valor espectral que ha de ser actualmente decodificado y para un bin espectral precedente $i-1$. La variable c_4 es inicializada en forma dependiente de un valor espectral decodificado previamente, decodificado para un marco de audio previo (que tiene índice de tiempo $t-1$) y que tiene una frecuencia que es inferior (p. ej., en un bin de frecuencia) a la frecuencia asociada con el valor espectral que ha de ser actualmente decodificado.

[000114] El segundo ajuste de variable 516 que se ejecuta en forma selectiva si (y solo si) el índice de frecuencia del valor espectral que ha de ser actualmente decodificado es superior a 1, comprende una inicialización de las variables c_1 y c_6 y una actualización de la variable lev0. La variable c_1 es actualizada en forma dependiente de un valor contextual $q[1][i-2].c$ asociado con un valor espectral decodificado previamente del marco de audio actual, una frecuencia del cual es más pequeña (p. ej. en dos bins de frecuencia) que una frecuencia de un valor espectral que se ha de decodificar actualmente. De modo similar, la variable c_6 es inicializada en forma dependiente de un valor contextual $q[0][i-2].c$, que describe un valor espectral decodificado previamente de un marco previo (que tiene índice de tiempo $t-1$), una frecuencia asociada del cual es más pequeña (p. ej. en dos bins de frecuencia) que una frecuencia asociada con el valor espectral que se ha de decodificar actualmente. Además, la variable de nivel „lev0“ se ajusta a un valor de nivel $q[1][i-2].l$ asociado con un valor espectral decodificado previamente del marco actual, una frecuencia asociada del cual es más pequeña (p. ej. en dos bins de frecuencia) que una frecuencia asociada con el valor espectral que se ha de decodificar actualmente, si $q[1][i-2].l$ es mayor que lev0.

[000115] La adaptación de niveles 518 y el ajuste de valores de región 520 se ejecutan en forma selectiva, si (y solo si) el índice i del valor espectral que se ha de decodificar es mayor que 2. En la adaptación de niveles 518, la variable de nivel „lev0“ aumenta a un valor de $q[1][i-3].l$, si el valor de nivel $q[1][i-3].l$ que está asociado a un valor espectral decodificado previamente del marco actual, una frecuencia asociada del cual es más pequeña (p. ej. en tres bins de frecuencia) que la frecuencia asociada con el valor espectral que se ha de decodificar actualmente, es mayor que el valor de nivel lev0.

[000116] En el ajuste de valores de región 520, se establece una variable de región „region“ en forma dependiente de una evaluación, en cuya región espectral, de una pluralidad de regiones espectrales, se dispondrá el valor espectral que se ha de decodificar actualmente. Por ejemplo, si resulta que el valor espectral que ha de ser actualmente decodificado está asociado a un bin de frecuencia (que tiene índice de bins de frecuencia i) que está en el primer (es decir, el más bajo) cuarto de los bins de frecuencia ($0 \leq i < N/4$), la variable de región „region“ se ajusta a cero. Por el contrario, si el valor espectral que se ha de decodificar actualmente está asociado a un bin de frecuencia que está en un segundo cuarto de los bins de frecuencia asociados al marco actual ($N/4 \leq i < N/2$), la variable de región se ajusta a un valor de 1. Por el contrario, es decir, si el valor espectral que se ha de decodificar actualmente está asociado a un bin de frecuencia que está en la segunda (es decir, la más alta) mitad de los bins de frecuencia ($N/2 \leq i < N$), la variable de región se ajusta a 2. Por consiguiente, se establece una variable de región en forma dependiente de una evaluación a cuya región de frecuencia se asocia el valor espectral que se ha de decodificar actualmente. Se pueden distinguir dos o más regiones de frecuencia.

[000117] Se ejecuta una adaptación de niveles adicional 522 si (y solo si) el valor espectral que se ha de decodificar actualmente comprende un índice espectral que es mayor que 3. En este caso, la variable de nivel „lev0“ es aumentada (ajustada al valor $q[1][i-4].l$) si el valor de nivel $q[1][i-4].l$, que está asociado a un valor espectral decodificado previamente del marco actual, que está asociado a una frecuencia que es más pequeña, por ejemplo, en cuatro bins de frecuencia, que una frecuencia asociada al valor espectral que se ha de decodificar actualmente es mayor que el nivel actual „lev0“ (paso 522). La variable de nivel „lev0“ está limitada a un valor máximo de 3 (paso 524).

[000118] Si se detecta una condición de reiniciación aritmética y el índice i del valor espectral que se ha de decodificar actualmente es mayor que 1, el valor de estado retorna en forma dependiente de las variables c_0 , c_1 , lev0, así como en forma dependiente de la variable de región „region“ (paso 526). En consecuencia, los valores espectrales previamente decodificados de cualquiera de los marcos previos no son tenidos en consideración si se da una condición de reiniciación aritmética.

[000119] En el tercer ajuste de variable 528, se ajusta la variable c_2 al valor contextual $q[0][i].c$, que está asociado a un valor espectral decodificado previamente del marco de audio previo (que tiene índice de tiempo $t-1$), valor espectral decodificado previamente que está asociado con la misma frecuencia que el valor espectral que se ha de decodificar actualmente.

[000120] En el cuarto ajuste de variable 530, se ajusta la variable c_3 al valor contextual $q[0][i+1].c$, que está asociado a un valor espectral decodificado previamente del marco de audio previo que tiene un índice de frecuencia $i+1$, a menos que el valor espectral que se ha de decodificar actualmente esté asociado con el índice de frecuencia máximo posible $lg-1$.

[000121] En el quinto ajuste de variable 532, se ajusta la variable *c5* al valor contextual *q[0][i+2].c*, que está asociado a un valor espectral decodificado previamente del marco de audio previo que tiene índice de frecuencia *i+2*, a menos que el índice de frecuencia *i* del valor espectral que se ha de decodificar actualmente esté demasiado cercano al valor de índice de frecuencia máximo (es decir, toma el valor de índice de frecuencia *lg-2* o *lg-1*).

[000122] Se ejecuta una adaptación adicional de la variable de nivel „lev0“ si el índice de frecuencia *i* es igual a cero (es decir, si el valor espectral que se ha de decodificar actualmente es el valor espectral más bajo). En este caso, la variable de nivel „lev0“ es aumentada de cero a 1, si la variable *c2* o *c3* toma un valor de 3, lo que indica que un valor espectral decodificado previamente de un marco de audio previo, que está asociado con la misma frecuencia o incluso una frecuencia más alta, cuando se compara con la frecuencia asociada al valor espectral que se ha de codificar actualmente, alcanza un valor comparativamente grande.

[000123] En la computación selectiva de valores de retorno 536, el valor de retorno es computado en forma dependiente de que el índice *i* de los valores espectrales que se han de decodificar actualmente tome el valor cero, 1, o un valor mayor. El valor de retorno es computado en forma dependiente de las variables *c2*, *c3*, *c5* y *lev0*, indicado con el número de referencia 536a, si el índice *i* toma el valor de cero. El valor de retorno es computado en forma dependiente de las variables *c0*, *c2*, *c3*, *c4*, *c5*, y „lev0“ indicado con el número de referencia 536b, si el índice *i* toma el valor de 1. El valor de retorno es computado en forma dependiente de la variable *c0*, *c2*, *c3*, *c4*, *c1*, *c5*, *c6*, „region“, y *lev0*, si el índice *i* toma un valor que es diferente de cero o 1 (número de referencia 536c).

[000124] Para resumir lo anterior, la computación de valores contextuales „arith_get_context()“ comprende una detección 512 de un grupo de una pluralidad de valores espectrales cero previamente decodificados (o por lo menos, valores espectrales suficientemente pequeños). Si se encuentra un grupo suficiente de valores espectrales cero previamente decodificados, se indica la presencia de un contexto especial ajustando el valor de retorno a 1. De lo contrario, se ejecuta la computación de valores contextuales. Se puede decir, en términos generales, que en la computación de valores contextuales, se evalúa el valor de índice *i* para decidir cuántos valores espectrales previamente decodificados deberían ser evaluados. Por ejemplo, una cantidad de valores espectrales previamente decodificados evaluados se reduce si un índice de frecuencia *i* del valor espectral que se ha de decodificar actualmente está cercano a un límite inferior (p. ej. cero), o cercano a un límite superior (p. ej. *lg-1*). Además, incluso si el índice de frecuencia *i* del valor espectral que se ha de decodificar actualmente está lo suficientemente alejado de un valor mínimo, las regiones espectrales diferentes se distinguen por el ajuste de valores de región 520. En consecuencia, se tienen en cuenta diferentes propiedades estadísticas de distintas regiones espectrales (p. ej. primero, región espectral de frecuencia baja; segundo, región espectral de frecuencia media y tercero, región espectral de frecuencia alta). El valor contextual, que se calcula como valor de retorno, depende de la variable de región „region“, de modo tal que el valor contextual retornado depende de que un valor espectral que se ha de decodificar actualmente esté en una primera región de frecuencia predeterminada o en una segunda región de frecuencia predeterminada (o en cualquier otra región de frecuencia predeterminada).

6.5 Selección de normas de mapeo

[000125] A continuación, se explicará la selección de una norma de mapeo, por ejemplo, una tabla de frecuencias acumulativas, que describe un mapeo de un valor de código a un código de símbolo. La selección de la norma de mapeo se hace en forma dependiente del estado contextual, que es descripto por el valor de estado *s* o *t*.

6.5.1 Selección de normas de mapeo usando el algoritmo de acuerdo con la Fig. 5d

[000126] A continuación, se describirá la selección de una norma de mapeo utilizando la función „get_pk“ de acuerdo con la Fig. 5d. Debe tenerse en cuenta que la función „get_pk“ puede ejecutarse para obtener el valor de „pki“ en el subalgoritmo 312ba del algoritmo de la Fig. 3. Por consiguiente, la función „get_pk“ puede tomar el lugar de la función „arith_get_pk“ en el algoritmo de la Fig. 3.

[000127] También debe observarse que una función „get_pk“ de acuerdo con la Fig. 5d puede evaluar la tabla „ari_s_hash[387]“ de acuerdo con las Figs. 17(1) y 17(2) y una tabla „ari_gs_hash[225]“ de acuerdo con la Fig. 18.

[000128] La función „get_pk“ recibe, como variable entrante, un valor de estado *s*, que se puede obtener mediante una combinación de la variable „t“ de acuerdo con la Fig. 3 y las variables „lev“, „lev0“ de acuerdo con la Fig. 3. La función „get_pk“ también tiene una configuración que le permite retornar, como valor de retorno, un valor de una variable „pki“, que designa una norma de mapeo o una tabla de frecuencias acumulativas. La función „get_pk“ tiene una configuración que le permite mapear el valor de estado *s* a un valor de índice de normas de mapeo „pki“.

[000129] La función „get_pk“ comprende una primera evaluación de tabla 540, y una segunda evaluación de tabla 544. La primera evaluación de tabla 540 comprende una inicialización de variables 541 en la que son inicializadas las variables *i_min*, *i_max*, e *i*, indicada con el número de referencia 541. La primera evaluación de tabla 540 también comprende una búsqueda de tablas iterativa 542, en el curso de la cual se determina si hay una entrada de la tabla „ari_s_hash“ que coincide con el valor de estado *s*. Si durante la búsqueda de tablas iterativa 542 dicha coincidencia es identificada, la función *get_pk* es abortada, donde un valor de retorno de la función es determinado

por la entrada de la tabla “ari_s_hash” que coincide con el valor de estado **s**, como ya se explicará con más detalles. Sin embargo, si no se produce coincidencia perfecta entre el valor de estado **s** y se encuentra una entrada de la tabla “ari_s_hash” en el transcurso de la búsqueda de tablas iterativa 542, se ejecuta una verificación de las entradas límite 543.

[000130] Volviendo ahora a los detalles de la primera evaluación de tabla 540, se puede ver que queda definido un intervalo de búsqueda por las variables $i_{\min}$ e $i_{\max}$. La búsqueda de tablas iterativa 542 es repetida mientras que el intervalo definido por las variables $i_{\min}$ e $i_{\max}$ sea suficientemente grande, lo que puede ser así si se cumple la condición $i_{\max} - i_{\min} > 1$. Consiguientemente, se establece la variable **i**, por lo menos de manera aproximada, para designar el punto medio del intervalo ($i = (i_{\min} + (i_{\max} - i_{\min})/2)$). Seguidamente, se ajusta una variable **j** a un valor que está determinado por la red “ari_s_hash” en una posición de red designada por la variable **i** (número de referencia 542). En este punto debe hacerse notar que cada entrada de la tabla “ari_s_hash” describe tanto un valor de estado, que está asociado a la entrada en la tabla, como un valor de índice de normas de mapeo que está asociado a la entrada en la tabla. El valor de estado, que está asociado a la entrada en la tabla, está descrito por los bits más significativos (bits 8-31) de la entrada en la tabla, mientras que los valores del índice de normas de mapeo son descritos por los bits inferiores (p. ej. bits 0-7) de dicha entrada en la tabla. El límite inferior $i_{\min}$ o el límite superior $i_{\max}$ están adaptados en forma dependiente de que el valor de estado **s** sea más pequeño que un valor de estado descrito por los 24 bits más significativos de la entrada “ari_s_hash[i]” de la tabla “ari_s_hash” referida por la variable **i**. Por ejemplo, si el valor de estado **s** es más pequeño que el valor de estado descrito por los 24 bits más significativos de la entrada “ari_s_hash[i]”, se ajusta el límite superior $i_{\max}$ del intervalo de tabla al valor **i**. En consecuencia, el intervalo de tabla para la próxima iteración de la búsqueda de tablas iterativa 542 queda restringido a la mitad inferior del intervalo de tabla (de $i_{\min}$ a $i_{\max}$) que se usó para la presente iteración de la búsqueda de tablas iterativa 542. Si, por el contrario, el valor de estado **s** es mayor que los valores de estado descritos por los 24 bits más significativos de la entrada en la tabla “ari_s_hash[i]”, entonces el límite inferior $i_{\min}$ del intervalo de tabla para la próxima iteración de la búsqueda de tablas iterativa 542 se ajusta al valor **i**, de modo tal que se usa la mitad superior del intervalo de tabla actual (entre $i_{\min}$ e $i_{\max}$) como intervalo de tabla para la próxima búsqueda de tablas iterativa. Sin embargo, si resulta que el valor de estado **s** es idéntico al valor de estado descrito por los 24 bits más significativos de la entrada en la tabla “ari_s_hash[i]”, el valor de índice de normas de mapeo descrito por los 8 bits menos significativos de la entrada en la tabla “ari_s_hash[i]” es retornado por la función “get_pk”, y la función es abortada.

[000131] La búsqueda de tablas iterativa 542 es repetida hasta que el intervalo de tabla definido por las variables $i_{\min}$ e $i_{\max}$ es lo suficientemente pequeño.

[000132] (Opcionalmente) se ejecuta una verificación de las entradas límite 543 para suplementar la búsqueda de tablas iterativa 542. Si la variable de índice **i** es igual a la variable de índice $i_{\max}$ después de completarse la búsqueda de tablas iterativa 542, se hace una verificación final sobre si el valor de estado **s** es igual a un valor de estado descrito por los 24 bits más significativos de una entrada de tabla “ari_s_hash[$i_{\min}$]”, y se retorna un valor de índice de normas de mapeo descrito por los 8 bits menos significativos de la entrada “ari_s_hash[$i_{\min}$]”, en este caso, como resultado de la función “get_pk”. Por el contrario, si la variable de índice **i** es diferente de la variable de índice $i_{\max}$, entonces se verifica si un valor de estado **s** es igual a un valor de estado descrito por los 24 bits más significativos de la entrada en la tabla “ari_s_hash[$i_{\max}$]”, y se retorna un valor de índice de normas de mapeo descrito por los 8 bits menos significativos de dicha entrada en la tabla “ari_s_hash[$i_{\max}$]” como valor de retorno de la función “get_pk” en este caso.

[000133] Sin embargo, debe tenerse en cuenta que la verificación de las entradas límite 543 puede considerarse opcional en su totalidad.

[000134] A continuación de la primera evaluación de tabla 540, se ejecuta la segunda evaluación de tabla 544, a menos que se haya producido un acierto directo “direct hit” durante la primera evaluación de tabla 540, donde el valor de estado **s** es idéntico a uno de los valores de estado descritos por las entradas de la tabla “ari_s_hash” (o, más precisamente, por sus 24 bits más significativos).

[000135] La segunda evaluación de tabla 544 comprende una inicialización de variable 545, en la que las variables de índice $i_{\min}$, **i** e $i_{\max}$ son inicializadas, como se indica con el número de referencia 545. La segunda evaluación de tabla 544 también comprende una búsqueda de tablas iterativa 546, en el curso de la cual se hace una búsqueda en la tabla “ari_gs_hash” para encontrar una entrada que representa un valor de estado idéntico al valor de estado **s**. Por último, la segunda búsqueda en la tabla 544 comprende una determinación de valor de retorno 547.

[000136] La búsqueda de tablas iterativa 546 se repite mientras que el intervalo de tabla definido por las variables de índice $i_{\min}$ e $i_{\max}$ es lo suficientemente grande (p. ej. de una longitud $i_{\max} - i_{\min} > 1$). En la iteración de la búsqueda de tablas iterativa 546, se establece la variable **i** en el centro del intervalo de tabla definido por $i_{\min}$ e $i_{\max}$ (paso 546a). Luego, se obtiene una entrada **j** de la tabla “ari_gs_hash” en una ubicación de la tabla determinada por la variable de índice **i** (546b). En otras palabras, la entrada en la tabla “ari_gs_hash[i]” es una entrada de tabla en el centro del intervalo de tabla actual definido por los índices de tabla $i_{\min}$ e $i_{\max}$. A continuación, se determina el intervalo de tabla para la próxima iteración de la búsqueda de tablas iterativa 546. Con

esta finalidad, el valor de índice i_max que describe el límite superior del intervalo de tabla es ajustado al valor i , si el valor de estado s es más pequeño que un valor de estado descripto por los 24 bits más significativos de la entrada en la tabla " $j=ari_gs_hash[i]$ " (546c). En otras palabras, se selecciona la mitad inferior del intervalo de tabla actual como nuevo intervalo de tabla para la próxima iteración de la búsqueda de tablas iterativa 546 (paso 546c). Por el contrario, si el valor de estado s es mayor que un valor de estado descripto por los 24 bits más significativos de la entrada en la tabla " $j=ari_gs_hash[i]$ ", el valor de índice i_min se ajusta al valor i . En consecuencia, se selecciona la mitad superior del intervalo de tabla actual como nuevo intervalo de tabla para la próxima iteración de la búsqueda de tablas iterativa 546 (paso 546d). Sin embargo, si resulta que el valor de estado s es idéntico a un valor de estado descripto por los 24 bits más altos de la entrada en la tabla " $j=ari_gs_hash[i]$ ", la variable de índice i_max se ajusta al valor $i+1$ o al valor 224 (si $i+1$ es mayor que 224), y la búsqueda de tablas iterativa 546 es abortada. Sin embargo, si el valor de estado s es diferente del valor de estado descripto por los 24 bits más significativos de " $j=ari_gs_hash[i]$ ", la búsqueda de tablas iterativa 546 es repetida con el intervalo de tabla recién ajustado definido por los valores de índice actualizados i_min e i_max , a menos que el intervalo de tabla sea demasiado pequeño ($i_max - i_min \leq 1$). Por consiguiente, el tamaño de intervalo del intervalo de tabla (definido por i_min e i_max) es reducido iterativamente hasta que se detecta un acierto directo "direct hit" ($s==(j>8)$) o el intervalo alcanza un tamaño mínimo permisible ($i_max - i_min \leq 1$). Finalmente, luego de abortar la búsqueda de tablas iterativa 546, se determina una entrada de tabla " $j=ari_gs_hash[i_max]$ " y se retorna un valor de índice de normas de mapeo, que está descripto por los 8 bits menos significativos de dicha entrada en la tabla " $j=ari_gs_hash[i_max]$ " como el valor de retorno de la función "get_pk". En consecuencia, el valor de índice de normas de mapeo es determinado en forma dependiente del límite superior i_max del intervalo de tabla (definido por i_min e i_max) luego de completarse o abortarse la búsqueda de tablas iterativa 546.

[000137] Las evaluaciones de tabla 540, 544 recién descriptas, que utilizan ambas la búsqueda de tablas iterativa 542, 546, permiten examinar las tablas " ari_s_hash " y " ari_gs_hash " en busca de la presencia de un estado significativo determinado con una eficacia computacional muy elevada. En particular, se puede mantener un número de operaciones de acceso a las tablas razonablemente pequeño, aún en el peor de los casos. Se ha descubierto que un ordenamiento numérico de la tabla " ari_s_hash " y " ari_gs_hash " permite la aceleración de la búsqueda de un valor hash apropiado. Además, se puede mantener un tamaño de tabla pequeño ya que no se requiere la inclusión de símbolos de escape en las tablas " ari_s_hash " y " ari_gs_hash ". Por consiguiente, se establece un mecanismo eficaz de hash de contextos aún cuando haya un gran número de estados diferentes. En una primera etapa (primera evaluación de tabla 540), se realiza una búsqueda de un acierto directo ($s==(j>8)$).

[000138] En la segunda etapa (segunda evaluación de tabla 544) se pueden mapear los rangos del valor de estado s a valores del índice de normas de mapeo. Por consiguiente, se puede ejecutar un manejo bien equilibrado de los estados particularmente significativos, para los cuales hay una entrada asociada en la tabla " ari_s_hash ", y los estados menos significativos, para los cuales hay un manejo basado en rangos. En consecuencia, la función "get_pk" constituye una implementación eficaz de una selección de normas de mapeo.

[000139] Para más detalle, se hace referencia al pseudocódigo de programa de la Fig. 5d, que representa la funcionalidad de la función "get_pk" en una representación de acuerdo con el lenguaje de programación C ampliamente difundido.

6.5.2 Selección de normas de mapeo usando el algoritmo de acuerdo con la Fig. 5e

[000140] A continuación, se describirá otro algoritmo para seleccionar la norma de mapeo tomando como referencia la Fig. 5e. Debe tenerse en cuenta que el algoritmo "arith_get_pk" de acuerdo con la Fig. 5e recibe, como variable entrante, un valor de estado s que describe un estado del contexto. La función "arith_get_pk" provee como valor de salida, o valor de retorno, un índice "pki" de un modelo de probabilidades, que puede ser un índice para seleccionar una norma de mapeo, (p. ej., una tabla de frecuencias acumulativas).

[000141] Debe tenerse en cuenta que la función "arith_get_pk" de acuerdo con la Fig. 5e puede adoptar la funcionalidad de la función "arith_get_pk" de la función "value_decode" de la Fig. 3.

[000142] También debe observarse que la función "arith_get_pk" puede, por ejemplo, evaluar la tabla ari_s_hash de acuerdo con la Fig. 20, y la tabla ari_gs_hash de acuerdo con la Fig. 18.

[000143] La función "arith_get_pk" de acuerdo con la Fig. 5e comprende una primera evaluación de tabla 550 y una segunda evaluación de tabla 560. En la primera evaluación de tabla 550, se hace un barrido lineal a través de la tabla ari_s_hash , para obtener una entrada $j=ari_s_hash[i]$ de dicha tabla. Si un valor de estado descripto por los 24 bits más significativos de una entrada de tabla $j=ari_s_hash[i]$ de la tabla ari_s_hash es igual al valor de estado s , se retorna un valor de índice de normas de mapeo "pki" descripto por los 8 bits menos significativos de dicha entrada de tabla identificada $j=ari_s_hash[i]$ y la función "arith_get_pk" es abortada. En consecuencia, se evalúa cada una de las 387 entradas de la tabla ari_s_hash en una secuencia ascendente a menos que se identifique un acierto directo "direct hit" (valor de estado s igual al valor de estado descripto por los 24 bits más significativos de una entrada de tabla j).

[000144] Si no se identifica un acierto directo dentro de la primera evaluación de tabla 550, se ejecuta una segunda evaluación de tabla 560. En el transcurso de la segunda evaluación de tabla, se ejecuta un barrido lineal con índices de entrada *i* que aumentan linealmente desde cero hasta un valor máximo de 224. Durante la segunda evaluación de tabla, se lee una entrada "ari_gs_hash[i]" de la tabla "ari_gs_hash" para la tabla *i*, y se evalúa la entrada en la tabla "j=ari_gs_hash[i]" para determinar si el valor de estado representado por los 24 bits más significativos de la entrada en la tabla *j* es mayor que el valor de estado *s*. Si este es el caso, se retorna un valor de índice de normas de mapeo descrito por los 8 bits menos significativos de dicha entrada en la tabla *j* como el valor de retorno de la función "arith_get_pk", y la ejecución de la función "arith_get_pk" es abortada. Sin embargo, si el valor de estado *s* no es menor que el valor de estado descrito por los 24 bits más significativos de la entrada de tabla actual j=ari_gs_hash[i], se continúa el barrido a través de las entradas de la tabla ari_gs_hash aumentando el índice de la tabla *i*. Sin embargo, si el valor de estado *s* es superior o igual a cualquiera de los valores de estado descritos por las entradas de la tabla ari_gs_hash, se retorna un valor de índice de normas de mapeo „pki" definido por los 8 bits menos significativos de la última entrada de la tabla ari_gs_hash como el valor de retorno de la función "arith_get_pk".

[000145] En síntesis, la función "arith_get_pk" de acuerdo con la Fig. 5e ejecuta una operación hash de dos pasos. En un primer paso, se ejecuta una búsqueda de un acierto directo, donde se determina si el valor de estado *s* es igual al valor de estado definido por cualquiera de las entradas de una primera tabla "ari_s_hash". Si se identifica un acierto directo en la primera evaluación de tabla 550, se obtiene un valor de retorno de la primera tabla "ari_s_hash" y la función "arith_get_pk" es abortada. Sin embargo, si no se identifica un acierto directo en la primera evaluación de tabla 550, se ejecuta la segunda evaluación de tabla 560. En la segunda evaluación de tabla, se ejecuta una evaluación basada en rangos. Las entradas siguientes de la segunda tabla "ari_gs_hash" definen rangos. Si resulta que el valor de estado *s* se encuentra dentro de dicho rango (que está indicado por el hecho de que el valor de estado descrito por los 24 bits más significativos de la entrada de tabla actual "j=ari_gs_hash[i]" es mayor que el valor de estado *s*, se retorna el valor de índice de normas de mapeo "pki" descrito por los 8 bits menos significativos de la entrada en la tabla j=ari_gs_hash[i].

6.5.3 Selección de normas de mapeo usando el algoritmo de acuerdo con la Fig. 5f

[000146] La función "get_pk" de acuerdo con la Fig. 5f es sustancialmente equivalente a la función "arith_get_pk" de acuerdo con la Fig. 5e. En consecuencia, se hace referencia al análisis anterior. Para más detalles, se hace referencia a la representación del pseudoprograma en la Fig. 5f.

[000147] Debe tenerse en cuenta que la función „get_pk" de acuerdo con la Fig. 5f puede tomar el lugar de la función "arith_get_pk" denominada la función "value_decode" de la Fig. 3.

6.6. Función "arith_decode()" de acuerdo con la Fig. 5g

[000148] A continuación, se analizará la funcionalidad de la función "arith_decode()" en detalle con respecto a la Fig. 5g. Debe tenerse en cuenta que la función "arith_decode()" utiliza la función ayudante "arith_first_symbol(void)", que da TRUE (VERDADERO) como retorno, si es el primer símbolo de la secuencia y FALSE (FALSO) en caso contrario. La función "arith_decode()" también usa la función ayudante "arith_get_next_bit(void)", que obtiene y provee el próximo bit de la corriente de bits.

[000149] Además, la función "arith_decode()" usa las variables globales "low", "high" y "value" (bajo, alto y valor). Asimismo, la función "arith_decode()" recibe, como una variable entrante, la variable "cum_freq[]", que señala hacia una primera entrada o elemento (que tiene índice de elementos o índice de entradas 0) de la tabla de frecuencias acumulativas escogida. Asimismo, la función "arith_decode()" usa la variable de entradas "cfl", que indica la longitud de la tabla de frecuencias acumulativas escogida designada por la variable "cum_freq[]".

[000150] La función "arith_decode()" comprende, como primer paso, una inicialización de variable 570a, que se ejecuta si la función ayudante "arith_first_symbol()" indica que el primer símbolo de una secuencia de símbolos está siendo decodificado. La inicialización de valores 550a inicializa la variable "value" (valor) de manera dependiente de una pluralidad, por ejemplo, de 20 bits, que se obtienen de la corriente de bits usando la función ayudante "arith_get_next_bit", de modo tal que la variable "value" (valor) toma el valor representado por dichos bits. Asimismo, la variable "low" (bajo) es inicializada para tomar el valor de 0, y la variable "high" (alto) es inicializada para tomar el valor de 1048575.

[000151] En un segundo paso 570b, se ajusta la variable rango "range" a un valor, que es mayor, en 1, que la diferencia entre los valores de las variables "high" y "low" (alto y bajo). La variable "cum" se establece a un valor que representa una posición relativa del valor de la variable "value" (valor) entre el valor de la variable "low" (bajo) y el valor de la variable "high" (alto). En consecuencia, la variable "cum" toma, por ejemplo, un valor entre 0 y 2^{16} dependiente del valor de la variable "value" (valor).

[000152] El indicador *p* es inicializado a un valor que es menor, en 1, que la dirección de comienzo de la tabla de frecuencias acumulativas escogida.

[000153] El algoritmo "arith_decode()" también comprende una búsqueda iterativa de tablas de frecuencias acumulativas 570c. La búsqueda iterativa de tablas de frecuencias acumulativas se repite hasta que la variable *cfl* es inferior o igual a 1. En la búsqueda iterativa de tablas de frecuencias acumulativas 570c, la variable de indicadores *q* se ajusta a un valor, que es igual a la suma del valor actual de la variable de indicadores *p* y la mitad del valor de la variable "*cfl*". Si el valor de la entrada **q* de la tabla de frecuencias acumulativas escogida, entrada que es señalada por la variable de indicadores *q*, es mayor que el valor de la variable "*cum*", la variable de indicadores *p* es ajustada al valor de la variable de indicadores *q*, y la variable "*cfl*" aumenta. Finalmente, la variable "*cfl*" se desplaza hacia la derecha en un bit, dividiéndose así de manera eficaz el valor de la variable "*cfl*" por 2 y despreciando la porción modular.

[000154] En consecuencia, la búsqueda iterativa de tablas de frecuencias acumulativas 570c compara efectivamente el valor de la variable "*cum*" con una pluralidad de entradas de la tabla de frecuencias acumulativas escogida, para identificar un intervalo dentro de la tabla de frecuencias acumulativas escogida, que está limitado por las entradas de la tabla de frecuencias acumulativas, de modo tal que el valor "*cum*" se encuentra dentro del intervalo identificado. En consecuencia, las entradas de la tabla de frecuencias acumulativas escogida definen intervalos, donde un respectivo valor de símbolo está asociado a cada uno de los intervalos de la tabla de frecuencias acumulativas escogida. Asimismo, los anchos de los intervalos entre dos valores adyacentes de la tabla de frecuencias acumulativas definen probabilidades de los símbolos asociados con dichos intervalos, de tal modo que la tabla de frecuencias acumulativas escogida define en su totalidad una distribución de probabilidades de los diferentes símbolos (o valores de símbolo). Más adelante se explicarán detalles sobre las tablas de frecuencias acumulativas disponibles con referencia a la Fig. 19.

[000155] Con respecto nuevamente a la Fig. 5g, el valor de símbolo se deriva del valor de la variable de indicadores *p*, donde se deriva el valor de símbolo, lo cual está indicado con el número de referencia 570d. Por consiguiente, se evalúa la diferencia entre el valor de la variable de indicadores *p* y la dirección de comienzo "*cum_freq*" para obtener el valor de símbolo, que está representado por la variable "*symbol*".

[000156] El algoritmo "arith_decode" también comprende una adaptación 570e de las variables "*high*" y "*low*" (alto y bajo). Si el valor de símbolo representado por la variable "*symbol*" (símbolo) es diferente de 0, la variable "*high*" (alto) es actualizada, lo cual está indicado con el número de referencia 570e. Asimismo, el valor de la variable "*low*" (bajo) es actualizado, indicado con el número de referencia 570e. La variable "*high*" (alto) se ajusta a un valor determinado por el valor de la variable "*low*" (bajo), la variable "*range*" (rango) y la entrada que tiene el índice "*symbol-1*" de la tabla de frecuencias acumulativas escogida. La variable "*low*" (bajo) aumenta, donde la magnitud del aumento queda determinado por la variable "*range*" (rango) y la entrada de la tabla de frecuencias acumulativas escogida que tiene el índice "*symbol*" (símbolo). En consecuencia, se ajusta la diferencia entre los valores de las variables "*low*" y "*high*" (bajo y alto) en forma dependiente de la diferencia numérica entre dos entradas adyacentes de la tabla de frecuencias acumulativas escogida.

[000157] En consecuencia, si se detecta un valor de símbolo que tiene una baja probabilidad, el intervalo entre los valores de las variables "*low*" y "*high*" (bajo y alto) se reduce hasta un ancho estrecho. Por el contrario, si el valor de símbolo detectado comprende una probabilidad relativamente grande, el ancho del intervalo entre los valores de las variables "*low*" y "*high*" (bajo y alto) se ajusta a un valor comparativamente grande. Una vez más, el ancho del intervalo entre los valores de las variables "*low*" y "*high*" (bajo y alto) depende del símbolo detectado y las entradas correspondientes de la tabla de frecuencias acumulativas.

[000158] El algoritmo "arith_decode()" también comprende una renormalización de intervalos 570f, en la que el intervalo determinado en el paso 570e es desplazado iterativamente y escalado hasta que se alcanza la condición "*break*" (rotura). En la renormalización de intervalos 570f, se ejecuta una operación selectiva de desplazamiento descendente 570fa. Si la variable "*high*" (alto) es inferior a 524286, no se hace nada, y la renormalización de intervalos continúa con una operación de incremento de tamaño de intervalo 570fb. Sin embargo, si la variable "*high*" (alto) no es inferior a 524286 y la variable "*low*" (bajo) es superior o igual a 524286, las variables "*values*", "*low*" y "*high*" (valores, bajo y alto) se reducen todas en 524286, de modo tal que un intervalo definido por las variables "*low*" y "*high*" (bajo y alto) es desplazado hacia abajo, y de modo tal que el valor de la variable "*value*" (valor) también es desplazado hacia abajo. Sin embargo, si resulta que el valor de la variable "*high*" (alto) no es inferior a 524286, y que la variable "*low*" (bajo) no es superior o igual a 524286, y que la variable "*low*" (bajo) es superior o igual a 262143 y que la variable "*high*" (alto) es inferior a 786429, las variables "*value*", "*low*" y "*high*" (valor, bajo y alto) se reducen todas en 262143, desplazando de esta manera hacia abajo el intervalo entre los valores de las variables "*high*" y "*low*" (alto y bajo) y también el valor de la variable "*value*" (valor). Sin embargo, si no se satisface ninguna de las condiciones anteriores, la renormalización de intervalos es abortada.

[000159] Sin embargo, si se satisface alguna de las condiciones antes mencionadas, que son evaluadas en el paso 570fa, se ejecuta la operación de incremento de intervalo 570fb. En la operación de incremento de intervalo 570fb, el valor de la variable "*low*" (bajo) se duplica. Asimismo, se duplica el valor de la variable "*high*" (alto), y el resultado de la duplicación aumenta en 1. Asimismo, se duplica el valor de la variable "*value*" (valor) (desplazada hacia la izquierda en un bit), y se usa un bit de la corriente de bits, que se obtiene por la función ayudante

“arith_get_next_bit”, como el bit menos significativo. En consecuencia, el tamaño del intervalo entre los valores de las variables “low” y “high” (bajo y alto) es duplicado de manera aproximada, y la precisión de la variable “value” (valor) es aumentada usando un nuevo bit de la corriente de bits. Como ya se indicó anteriormente, se repiten los pasos 570fa y 570fb hasta que se alcanza la condición “break” (rotura), es decir, hasta que el intervalo entre los valores de las variables “low” y “high” (bajo y alto) es lo suficientemente grande.

[000160] Respecto de la funcionalidad del algoritmo “arith_decode()”, debe tenerse en cuenta que el intervalo entre los valores de las variables “low” y “high” (bajo y alto) se reduce en el paso 570e en forma dependiente de dos entradas adyacentes de la tabla de frecuencias acumulativas referidas por la variable “cum_freq”. Si un intervalo entre dos valores adyacentes de la tabla de frecuencias acumulativas escogida es pequeño, es decir, si los valores adyacentes se encuentran comparativamente cercanos uno del otro, el intervalo entre los valores de las variables “low” y “high” (bajo y alto), que se obtiene en el paso 570e, será comparativamente pequeño. Por el contrario, si dos entradas adyacentes de la tabla de frecuencias acumulativas se encuentran más espaciadas, el intervalo entre los valores de las variables “low” y “high” (bajo y alto), que se obtiene en el paso 570e, será comparativamente grande.

[000161] En consecuencia, si el intervalo entre los valores de las variables “low” y “high” (bajo y alto), que se obtiene en el paso 570e, es comparativamente pequeño, se ejecutará una gran cantidad de pasos de renormalización de intervalos para reescalar el intervalo hasta un tamaño “suficiente” (de tal modo que no se satisface ninguna de las condiciones de la evaluación de condiciones 570fa). En consecuencia, se usará un número comparativamente grande de bits de la corriente de bits para aumentar la precisión de la variable “value” (valor). Si, por el contrario, el tamaño de intervalo obtenido en el paso 570e es comparativamente grande, sólo se necesitará un número más pequeño de repeticiones de los pasos de la normalización de intervalo 570fa y 570fb para renormalizar el intervalo entre los valores de las variables “low” y “high” (bajo y alto) hasta un tamaño “suficiente”. En consecuencia, se usará solamente un número comparativamente pequeño de bits de la corriente de bits para aumentar la precisión de la variable “value” (valor) y para preparar una decodificación de un próximo símbolo.

[000162] Para resumir lo anterior, si se decodifica un símbolo, que comprende una probabilidad comparativamente alta, y al que se asocia un intervalo grande por las entradas de la tabla de frecuencias acumulativas escogida, sólo se leerá una cantidad comparativamente pequeña de bits de la corriente de bits para permitir la decodificación de un símbolo subsiguiente. Por el contrario, si se decodifica un símbolo, que comprende una probabilidad comparativamente pequeña y al que se asocia un intervalo pequeño por las entradas de la tabla de frecuencias acumulativas escogida, se tomará un número comparativamente grande de bits de la corriente de bits para preparar una decodificación del próximo símbolo.

[000163] En consecuencia, las entradas de las tablas de frecuencias acumulativas reflejan las probabilidades de los diferentes símbolos y también reflejan una cantidad de bits necesaria para decodificar una secuencia de símbolos. Si se varía la tabla de frecuencias acumulativas en forma dependiente de un contexto, es decir, en forma dependiente de símbolos previamente decodificados (o valores espectrales), por ejemplo, seleccionando tablas de frecuencias acumulativas diferentes en forma dependiente del contexto, se pueden explotar dependencias estocásticas entre los diferentes símbolos, lo que permite una codificación eficaz de la velocidad de transmisión de bits particular de los símbolos subsiguientes (o adyacentes).

[000164] Para resumir lo anterior, la función “arith_decode()”, que ha sido descripta con referencia a la Fig. 5g, es pedida con la tabla de frecuencias acumulativas “arith_cf_m[*pk*]”, correspondiente al índice “*pk*” retornado por la función “arith_get_pk()” para determinar el valor del plano de bits más significativo *m* (que puede ser ajustado al valor de símbolo representado por la variable de retorno “symbol” (símbolo)).

6.7 Mecanismo de escape

[000165] A pesar de que el valor decodificado del plano de bits más significativo *m* (que es retornado como un valor de símbolo por la función “arith_decode ()”) es el símbolo de escape “ARITH_ESCAPE”, se decodifica un valor del plano de bits más significativo *m* adicional y la variable “lev” es incrementada en 1. En consecuencia, se obtiene una información acerca del significado numérico del valor del plano de bits más significativo *m* así como de la cantidad de planos de bits menos significativos que se han de decodificar.

[000166] Si se decodifica un símbolo de escape “ARITH_ESCAPE”, la variable de nivel “lev” es aumentada en 1. En consecuencia, el valor de estado que ingresa a la función “arith_get_pk” también se modifica en el hecho de que aumenta un valor representado por los bits superiores (bits 24 en adelante) para las próximas iteraciones del algoritmo 312ba.

6.8 Actualización contextual de acuerdo con la Fig. 5h

[000167] Una vez que se ha decodificado completamente el valor espectral (es decir, se han añadido todos los planos de bits menos significativos, las tablas de contexto *q* y *qs* son actualizadas pidiendo la función “arith_update_context(*a*,*i*,*lg*)”). A continuación, se explicarán detalles sobre la función “arith_update_context(*a*,*i*,*lg*)” con referencia a la Fig. 5h, que muestra una representación del pseudocódigo de programa de dicha función.

[000168] La función "arith_update_context()" recibe, como variable de ingreso, el coeficiente espectral cuantificado decodificado **a**, el índice **i** del valor espectral que se ha de decodificar (o del valor espectral decodificado) y la cantidad **lg** de valores espectrales (o coeficientes) asociados con el marco de audio actual.

[000169] En un paso 580, se copia el valor (o coeficiente) espectral cuantificado actualmente decodificado **a** en la tabla de contextos o red de contextos **q**. En consecuencia, la entrada $q[1][i]$ de la tabla de contextos **q** se ajusta a **a**. Asimismo, la variable "a0" es ajustada al valor de "a".

[000170] En un paso 582, se determina el valor de nivel $q[1][i].l$ de la tabla de contextos **q**. Por defecto, el valor de nivel $q[1][i].l$ de la tabla de contextos **q** se ajusta a cero. Sin embargo, si el valor absoluto del valor espectral codificado actualmente **a** es mayor que 4, se incrementa el valor de nivel $q[1][i].l$.

[000171] Con cada incremento, la variable "a" es desplazada hacia la derecha en un bit. El incremento del valor de nivel $q[1][i].l$ se repite hasta que el valor absoluto de la variable **a0** es inferior o igual a 4.

[000172] En un paso 584, se establece un valor contextual de 2 bits $q[1][i].c$ de la tabla de contextos **q**. El valor contextual de 2 bits $q[1][i].c$ es ajustado al valor de cero si el valor espectral actualmente decodificado **a** es igual a cero. Por el contrario, si el valor absoluto del valor espectral decodificado **a** es inferior o igual a 1, el valor contextual de 2 bits $q[1][i].c$ se ajusta a 1. Por el contrario, si el valor absoluto del valor espectral actualmente decodificado **a** es inferior o igual a 3, el valor contextual de 2 bits $q[1][i].c$ se ajusta a 2. Por el contrario, es decir, si el valor absoluto del valor espectral actualmente decodificado **a** es mayor que 3, el valor contextual de 2 bits $q[1][i].c$ se ajusta a 3. En consecuencia, el valor contextual de 2 bits $q[1][i].c$ se obtiene por medio de una cuantificación *grosso modo* del coeficiente espectral actualmente decodificado **a**.

[000173] En un paso posterior 586, que sólo es ejecutado si el índice **i** del valor espectral actualmente decodificado es igual a la cantidad **lg** de coeficientes (valores espectrales) en el marco, es decir, si el último valor espectral del marco ha sido decodificado) y el modo núcleo es un modo núcleo de dominio de predicción lineal (indicado por "core_mode==1"), se copian las entradas $q[1][j].c$ en la tabla de contextos $qs[k]$. Se ejecuta la acción de copiado indicada con el número de referencia 586, de modo tal que se tiene en cuenta la cantidad **lg** de valores espectrales en el marco actual para el copiado de las entradas $q[1][j].c$ a la tabla de contextos $qs[k]$. Además, la variable "previous_lg" toma el valor 1024.

[000174] Alternativamente, sin embargo, las entradas $q[1][j].c$ de la tabla de contextos **q** se copian en la tabla de contextos $qs[j]$ si el índice **i** del coeficiente espectral actualmente decodificado alcanza el valor de **lg** y el modo núcleo es un modo núcleo de dominio frecuencial (indicado por "core_mode==0").

[000175] En este caso, la variable "previous_lg" es ajustada al mínimo entre el valor de 1024 y la cantidad **lg** de valores espectrales en el marco.

6.9 Síntesis del proceso de decodificación

[000176] A continuación, se explicará brevemente el proceso de decodificación. Para los detalles, se hace referencia al análisis anterior y también a las Figs. 3, 4 y 5a a 5i.

[000177] Los coeficientes espectrales cuantificados **a** son codificados de manera insonorizada y transmitidos, comenzando del coeficiente de frecuencia menor y progresando hasta el coeficiente de frecuencia más elevado.

[000178] Los coeficientes de la codificación de audio avanzada (AAC, por sus siglas en inglés) son almacenados en la red "x_ac_quant[g][win][sfb][bin]", y el orden de transmisión de las palabras de código de codificación insonorizada es tal que cuando estas son decodificadas en el orden recibido y almacenadas en la red, "bin" es el índice de incrementación más rápida y **g** es el índice de incrementación menos rápida. El índice bin designa bins de frecuencias. El índice "sfb" designa bandas de factores de escala. El índice "win" designa ventanas. El índice "g" designa marcos de audio.

[000179] Los coeficientes de la excitación codificada por la transformada se almacenan directamente en una red "x_tcx_invquant[win][bin]", y el orden de la transmisión de las palabras de código de codificación insonorizada es tal que cuando son decodificadas en el orden recibido y almacenadas en la red, "bin" es el índice de incrementación más rápida y "win" es índice de incrementación menos rápida.

[000180] En primer lugar se hace un mapeo entre el contexto pasado guardado almacenado en la tabla o red de contextos "qs" y el contexto del marco actual **q** (almacenado en la tabla o red de contextos **q**). El contexto pasado "qs" es almacenado a 2 bits por línea de frecuencia (o por bin de frecuencia).

[000181] El mapeo entre el contexto pasado guardado almacenado en la tabla de contextos “qs” y el contexto del marco actual almacenado en la tabla de contextos “q” es ejecutado usando la función “arith_map_context()”, una representación de pseudocódigo de programa que se ilustra en la Fig. 5a.

5 **[000182]** El decodificador insonorizado da salida a coeficientes espectrales cuantificados identificados “a”.

[000183] Al principio, se calcula el estado del contexto sobre la base de los coeficientes espectrales previamente decodificados que rodean a los coeficientes espectrales cuantificados que se han de decodificar. El estado del contexto **s** corresponde a los 24 primeros bits del valor retornado por la función “arith_get_context()”. Los bits que se encuentran más allá del bit 24 del valor retornado corresponden al nivel de plano de bits predicho lev0. La variable „lev” es inicializada a lev0. En las Figs. 5b y 5c se muestra una representación del pseudocódigo de programa de la función “arith_get_context”.

15 **[000184]** Una vez conocidos el estado **s** y el nivel predicho „lev0”, se decodifica el plano discreto de 2 bits más significativos **m** usando la función “arith_decode()”, alimentada con la tabla de frecuencias acumulativas apropiada que corresponde al modelo de probabilidades correspondiente al estado contextual.

[000185] La correspondencia es realizada por la función “arith_get_pk()”.

20 **[000186]** En la Fig. 5e se muestra una representación de pseudocódigo de programa de la función “arith_get_pk()”.

[000187] En la Fig. 5f se muestra un pseudocódigo de programa de otra función “get_pk” que puede tomar el lugar de la función “arith_get_pk()”. En la Fig. 5d se muestra un pseudocódigo de programa de otra función “get_pk”, que puede tomar el lugar de la función “arith_get_pk()”.

25 **[000188]** El valor **m** es decodificado usando la función “arith_decode()” pedida con la tabla de frecuencias acumulativas, “arith_cf_m[**pk**]”, donde „**pk**” corresponde al índice retornado por la función “arith_get_pk()” (o, alternativamente, por la función “get_pk()”).

30 **[000189]** El codificador aritmético es una implementación de números enteros que usa el método de generación de marcas con escalamiento (véase, p. ej., K. Sayood “Introduction to Data Compression”, Tercera edición, 2006, Elsevier Inc.). El pseudocódigo C que se muestra en la Fig. 5g describe el algoritmo usado.

35 **[000190]** Cuando el valor decodificado **m** es el símbolo de escape, “ARITH_ESCAPE”, se decodifica otro valor **m** y se incrementa la variable „lev” en 1. Cuando el valor **m** ya no es el símbolo de escape, “ARITH_ESCAPE”, el resto de los planos de bits es decodificado desde el nivel más significativo al menos significativo, pidiendo „lev” veces la función “arith_decode()” con la tabla de frecuencias acumulativas “arith_cf_r[]”. Dicha tabla de frecuencias acumulativas “arith_cf_r[]” puede describir, por ejemplo, una distribución par de probabilidades.

40 **[000191]** Los planos de bits decodificados **r** permiten el refinamiento del valor previamente decodificado **m** de la siguiente manera:

```

a = m;
for (i=0; i<lev;i++) {
45   r = arith_decode (arith_cf_r,2);
   a = (a<<1) | (r&1);
}

```

50 **[000192]** Una vez que el coeficiente espectral cuantificado **a** ha sido completamente decodificado, las tablas de contextos **q**, o el contexto almacenado **qs**, es actualizado por la función “arith_update_context()”, para los próximos coeficientes espectrales cuantificados que se han de decodificar.

[000193] En la Fig. 5h se muestra una representación del pseudocódigo de programa de la función “arith_update_context()”.

55 **[000194]** Además, en la Fig. 5i se muestra una leyenda de las definiciones.

7. Tablas de mapeo

60 **[000195]** En una forma de realización de acuerdo con la invención, se usan tablas “ari_s_hash” y “ari_gs_hash” y “ari_cf_m” que son especialmente ventajosas para la ejecución de la función “get_pk”, que ha sido analizada con referencia a la Fig. 5d, o para la ejecución de la función “arith_get_pk”, que ha sido analizada con referencia a la 5e, o para la ejecución de la función “get_pk”, que ha sido analizada con referencia a la Fig. 5f, y para la ejecución de la función “arith_decode” que ha sido analizada con referencia a la Fig. 5g.

65 7.1. Tabla “ari_s_hash[387]” de acuerdo con la Fig. 17

[000196] En la tabla de la Fig. 17 se muestra un contenido de una implementación especialmente ventajosa de la tabla "ari_s_hash", que es utilizada por la función "get_pk" que fue descrita con referencia a la Fig. 5d. Debe tenerse en cuenta que la tabla de la Fig. 17 enumera las 387 entradas de la tabla "ari_s_hash[387]". También debe observarse que la representación de tabla de la Fig. 17 muestra los elementos en el orden de los índices de elementos, de modo tal que el primer valor "0x00000200" corresponde a una entrada de tabla "ari_s_hash[0]" que tiene índice de elementos (o índice de tablas) 0, de tal modo que el último valor "0x03D0713D" corresponde a una entrada de tabla "ari_s_hash[386]" que tiene índice de elementos o índice de tablas 386. En este punto debe hacerse notar asimismo que "0x" indica que las entradas de tabla de la tabla "ari_s_hash" están representadas en un formato hexadecimal. Por otra parte, las entradas de tabla de la tabla "ari_s_hash" de acuerdo con la Fig. 17 están distribuidas en un orden numérico para permitir la ejecución de la primera evaluación de tabla 540 de la función "get_pk".

[000197] Debería hacerse notar asimismo que los 24 bits más significativos de las entradas de tabla de la tabla "ari_s_hash" representan valores de estado, mientras que los 8 bits menos significativos representan valores del índice de normas de mapeo pki.

[000198] Por consiguiente, las entradas de la tabla "ari_s_hash" describen un mapeo de aciertos directos "direct hit" de un valor de estado a un valor de índice de normas de mapeo "pki".

7.2 Tabla "ari_gs_hash" de acuerdo con la Fig. 18

[000199] En la tabla de la Fig. 18 se muestra un contenido de una forma de realización especialmente ventajosa de la tabla "ari_gs_hash". En este punto debe hacerse notar que la tabla de la Fig. 18 enumera las entradas de la tabla "ari_gs_hash". Dichas entradas están referidas por un índice de entradas unidimensionales del tipo de números enteros (también denominado "índice de elementos" o "índice de red" o "índice de tablas"), que está designado, por ejemplo, con "i". Debe tenerse en cuenta que la tabla "ari_gs_hash" que comprende un total de 225 entradas es perfectamente adecuada para su uso por la segunda evaluación de tabla 544 de la función "get_pk" descrita en la Fig. 5d.

[000200] Debe tenerse en cuenta que las entradas de la tabla "ari_gs_hash" están enumeradas en orden ascendente del índice de la tabla i para valores de índice de tablas i entre cero y 224. El término "0x" indica que las entradas de tabla están descritas en un formato hexadecimal. En consecuencia, la primera entrada de tabla "0x00000401" corresponde a la entrada de tabla "ari_gs_hash[0]" que tiene índice de tabla 0 y la última entrada de tabla "0xffffffff3f" corresponde a la entrada de tabla "ari_gs_hash[224]" que tiene índice de tabla 224.

[000201] También debe observarse que las entradas de tabla están ordenadas en forma numéricamente ascendente, de tal modo que las entradas de tabla están perfectamente adecuadas para la segunda evaluación de tabla 544 de la función "get_pk". Los 24 bits más significativos de las entradas de tabla de la tabla "ari_gs_hash" describen límites entre rangos de valores de estado, y los 8 bits menos significativos de las entradas describen valores del índice de normas de mapeo "pki" asociados con los rangos de valores de estado definidos por los 24 bits más significativos.

7.3 Tabla "ari_cf_m" de acuerdo con la Fig. 19

[000202] La Fig. 19 muestra un conjunto de 64 tablas de frecuencias acumulativas "ari_cf_m[pki][9]", una de las cuales es seleccionada por un codificador de audio 100, 700, o un decodificador de audio 200, 800, por ejemplo, para la ejecución de la función "arith_decode", es decir, para la decodificación del valor del plano de bits más significativo. La tabla seleccionada de las 64 tablas de frecuencias acumulativas de la Fig. 19 toma la función de la tabla "cum_freq[]" en la ejecución de la función "arith_decode()".

[000203] Según puede observarse en la Fig. 19, cada línea representa una tabla de frecuencias acumulativas que tiene 9 entradas. Por ejemplo, una primera línea 1910 representa las 9 entradas de una tabla de frecuencias acumulativas para "pki=0". Una segunda línea 1912 representa las 9 entradas de una tabla de frecuencias acumulativas para "pki=1". Por último, una línea 64, 1964, representa las 9 entradas de una tabla de frecuencias acumulativas para "pki=63". Por consiguiente, la Fig. 19 representa efectivamente las 64 tablas de frecuencias acumulativas diferentes para "pki=0" hasta una "pki=63", donde cada una de las 64 tablas de frecuencias acumulativas está representada por una única línea y donde cada una de dichas tablas de frecuencias acumulativas comprende 9 entradas.

[000204] En una línea (p. ej. una línea 1910 o una línea 1912 o una línea 1964), el valor que se encuentra más a la izquierda describe una primera entrada de una tabla de frecuencias acumulativas y el valor que se encuentra más a la derecha describe la última entrada de una tabla de frecuencias acumulativas.

[000205] En consecuencia, cada línea 1910, 1912, 1964 de la representación de tablas de la Fig. 19 representa las entradas de una tabla de frecuencias acumulativas para ser usada por la función "arith_decode" de acuerdo con la

Fig. 5g. La variable de ingresos "cum_freq[]" de la función "arith_decode" describe cuál de las 64 tablas de frecuencias acumulativas (representadas por líneas individuales de 9 entradas) de la tabla "ari_cf_m" debería emplearse para la decodificación de los coeficientes espectrales actuales.

5 7.4 Tabla "ari_s_hash" de acuerdo con la Fig. 20

[000206] La Fig. 20 muestra una alternativa para la tabla "ari_s_hash", que puede usarse en combinación con la función alternativa "arith_get_pk()" o "get_pk()" de acuerdo con la Fig. 5e o 5f.

10 **[000207]** La tabla "ari_s_hash" de acuerdo con la Fig. 20 comprende 386 entradas, que están enumeradas en la Fig. 20 en un orden ascendente del índice de tabla. Por consiguiente, el primer valor de tabla "0x0090D52E" corresponde a la entrada en la tabla "ari_s_hash[0]" que tiene índice de tabla 0, y la última entrada de tabla "0x03D0513C" corresponde a la entrada en la tabla "ari_s_hash[386]" que tiene índice de tabla 386.

15 **[000208]** "0x" indica que las entradas de tabla están representadas en una forma hexadecimal. Los 24 bits más significativos de las entradas de la tabla "ari_s_hash" describen estados significativos, y los 8 bits menos significativos de las entradas de la tabla "ari_s_hash" describen valores del índice de normas de mapeo.

20 **[000209]** En consecuencia, las entradas de la tabla "ari_s_hash" describen un mapeo de estados significativos a valores del índice de normas de mapeo "pki".

8. Evaluación y ventajas del desempeño

25 **[000210]** Las formas de realización de acuerdo con la invención usan funciones (o algoritmos) actualizadas y un conjunto actualizado de tablas, tal como ya se analizó antes, para obtener una correlación mejorada entre la complejidad de la computación, los requerimientos de memoria y la eficacia de codificación.

30 **[000211]** En términos generales, las formas de realización de acuerdo con la invención crean una codificación espectral insonorizada mejorada.

35 **[000212]** La presente descripción describe formas de realización para CE sobre codificación espectral insonorizada mejorada de coeficientes espectrales. El esquema propuesto se basa en el esquema de codificación aritmética basado en el contexto "original", como se describe en la propuesta de trabajo 4 del estándar de propuestas del USAC, pero reduce significativamente los requerimientos de memoria (RAM, ROM), a la vez que se mantiene un desempeño de la codificación insonorizada. Se demostró que es posible una transcodificación sin pérdidas de la WD3 (es decir, de la salida de un codificador de audio que provee una corriente de bits de acuerdo con la propuesta de trabajo 3 del estándar de propuestas del USAC). El esquema descrito en la presente, en general, es escalable, lo que permite correlaciones alternativas adicionales entre los requerimientos de memoria y el desempeño de codificación. Las formas de realización de acuerdo con la invención tienen como objetivo reemplazar el esquema de codificación espectral insonorizada tal como se usa en la propuesta de trabajo 4 del estándar de propuestas del USAC.

45 **[000213]** El esquema de codificación aritmética descrito en la presente se basa en el esquema del modelo de referencia 0 (RM0) o la propuesta de trabajo 4 (WD4) del estándar de propuestas del USAC. Los coeficientes espectrales previos en frecuencia o en tiempo forman un modelo de contexto. Este contexto se usa para la selección de tablas de frecuencias acumulativas para el codificador (codificador o decodificador) aritmético. Comparado con la forma de realización de acuerdo con la WD4, el modelado de contexto es mejor y las tablas que contienen las probabilidades de símbolos fueron restringidas. La cantidad de modelos de probabilidades diferentes aumentó de 32 a 64.

50 **[000214]** Las formas de realización de acuerdo con la invención reducen el tamaño de las tablas (demanda de datos ROM) a 900 palabras de 32 bits o 3600 bytes de longitud. Por el contrario, las formas de realización de acuerdo con la WD4 del estándar de propuestas del USAC exigen 16894,5 palabras o 76578 bytes. La demanda estática RAM se reduce, en algunas formas de realización de acuerdo con la invención, de 666 palabras (2664 bytes) a 72 (288 bytes) por canal codificador de núcleo. Al mismo tiempo, preserva completamente el desempeño de la codificación e incluso puede alcanzar una ganancia de aproximadamente 1,04% a 1,39%, en comparación con la velocidad general de información en todos y cada uno de los 9 puntos operativos. Todas las corrientes de bits de la propuesta de trabajo 3 (WD3) pueden transcodificarse sin sufrir pérdidas y sin afectar las restricciones del depósito de bits.

60 **[000215]** El esquema propuesto de acuerdo con las formas de realización de la invención es escalable: son posibles correlaciones flexibles entre la demanda de memoria y el desempeño de la codificación. Aumentando el tamaño de las tablas a la codificación, se puede incrementar asimismo la ganancia.

65 **[000216]** A continuación, se presenta un breve análisis del concepto de codificación de acuerdo con la WD4 del estándar de propuestas del USAC para facilitar la comprensión de las ventajas del concepto descrito en la presente. En la WD4 del USAC, se usa un esquema de codificación aritmética basado en contextos para la

codificación insonorizada de coeficientes espectrales cuantificados. Como contexto, se usan los coeficientes espectrales decodificados, que son previos en frecuencia y tiempo. De acuerdo con la WD4, se usa como contexto un número máximo de 16 coeficientes espectrales, 12 de los cuales son previos en tiempo. Ambos, los coeficientes espectrales usados para el contexto y que se han de decodificar, están agrupados en 4 tuplas (es decir, cuatro coeficientes espectrales próximos en frecuencia, ver Fig. 10a). El contexto se reduce y se mapea en una tabla de frecuencias acumulativas, que entonces se usa para decodificar las próximas 4 tuplas de coeficientes espectrales.

[000217] Para el esquema de codificación insonorizada WD4 completo, se requiere una demanda de memoria (ROM) de 16894,5 palabras (67578 bytes). Además, se necesitan 666 palabras (2664 bytes) de ROM estática por canal de codificador de núcleo para almacenar los estados para el próximo marco.

[000218] La representación de tablas de la Fig. 11a describe las tablas como se utilizan en el esquema de codificación aritmética WD4 del USAC.

[000219] Se calcula que la demanda total de memoria de un decodificador completo de acuerdo con la WD4 del USAC es de 37000 palabras (148000 bytes) para la ROM de datos sin un código de programa y de 10000 a 17000 palabras para la RAM estática. Se puede ver claramente que las tablas de codificador insonorizado consumen aproximadamente 45% de la demanda total de ROM de datos. La tabla individual más grande consume de por sí 4096 palabras (16384 bytes).

[000220] Se ha descubierto que tanto el tamaño de la combinación de todas las tablas como las tablas individuales grandes exceden los tamaños de almacenamiento típicos provistos por los chips de puntos fijos para los dispositivos portátiles de bajo presupuesto, que están en un rango típico de 8-32 kByte (p. ej. ARM9e, TIC64xx, etc.). Esto significa que el conjunto de tablas podría no ser almacenado en la RAM de datos rápidos, que permite un acceso aleatorio rápido a los datos. Esto provoca que el proceso de decodificación completo se desacelere.

[000221] A continuación, se describirá brevemente el nuevo esquema propuesto.

[000222] Para resolver los problemas mencionados anteriormente, se propone un esquema de codificación insonorizada mejorado para reemplazar el esquema de la WD4 del estándar de propuestas del USAC. Como esquema de codificación aritmética basado en contextos, se basa en el esquema de la WD4 del estándar de propuestas del USAC, pero tiene las características de un esquema modificado para la derivación de tablas de frecuencias acumulativas del contexto. Más aún, se ejecuta una derivación de contexto y una codificación de símbolos según la granularidad de un único coeficiente espectral (opuesto a 4 tuplas, como en la WD4 del estándar de propuestas del USAC). En total, se usan 7 coeficientes espectrales para el contexto (por lo menos en algunos casos).

[000223] Mediante la reducción del mapeo, se selecciona uno de un total de 64 modelos de probabilidades o tablas de frecuencia acumulativas (en la WD4: 32).

[000224] La Fig. 10b muestra una representación gráfica de un contexto para el cálculo de estados, como se usa en el esquema propuesto (donde un contexto usado para la detección de la región cero no aparece ilustrado en la Fig. 10b).

[000225] A continuación, se presenta una breve reseña respecto de la reducción de la demanda de memoria, que se puede conseguir usando el esquema de codificación propuesto. El nuevo esquema propuesto exhibe una demanda total ROM de 900 palabras (3600 bytes) (ver la tabla de la Fig. 11b que describe las tablas como se usan en el esquema de codificación propuesto).

[000226] En comparación con la demanda ROM del esquema de codificación insonorizada en la WD4 del estándar de propuestas del USAC, la demanda ROM se reduce en 15994,5 palabras (64978 bytes) (ver también la Fig. 12a, que muestra una representación gráfica de la demanda ROM del esquema de codificación insonorizada propuesto y del esquema de codificación insonorizada en la WD4 del estándar de propuestas del USAC). Esto reduce la demanda ROM total de un decodificador USAC completo de aproximadamente 37000 palabras a aproximadamente 21000 palabras, o en más del 43% (ver la Fig. 12b, que muestra una representación gráfica de una demanda ROM total de datos del decodificador USAC de acuerdo con la WD4 del estándar de propuestas del USAC, de acuerdo, además, con la propuesta presentada aquí).

[000227] Todavía más, se reduce también la cantidad de información necesaria para la derivación contextual en el próximo marco (RAM estática). De acuerdo con la WD4, el conjunto de coeficientes completo (máximo 1152) con una resolución de 16 bits normalmente además de un índice de grupos por 4 tuplas de 10 bits de resolución necesarios para almacenar, lo que totaliza hasta 666 palabras (2664 bytes) por canal de codificador de núcleo (decodificador completo de la WD4 del USAC: aproximadamente 10000 a 17000 palabras).

[000228] El nuevo esquema, que se usa en las formas de realización de acuerdo con la invención, reduce la información persistente a sólo 2 bits por coeficiente espectral, lo que totaliza hasta 72 palabras (288 bytes) en total por canal de codificador de núcleo. La demanda de memoria estática puede reducirse en 594 palabras (2376 bytes).

[000229] A continuación, se describirán algunos detalles sobre un posible aumento de la eficacia de codificación. La eficacia de codificación de las formas de realización de acuerdo con la nueva propuesta se comparó con las corrientes de bits de acuerdo con la WD3 del estándar de propuestas del USAC como calidad de referencia. La comparación se realizó por medio de un transcodificador, basado en un decodificador de software de referencia. Respecto de los detalles relacionados con la comparación entre la codificación insonorizada de acuerdo con la WD3 del estándar de propuestas del USAC y el esquema de codificación propuesto, se hace referencia a la Fig. 9, que muestra una representación esquemática de una distribución de prueba.

[000230] Aunque la demanda de memoria se reduce drásticamente en las formas de realización de acuerdo con la invención cuando se compara con las formas de realización de acuerdo con la WD3 o WD4 del estándar de propuestas del USAC, la eficacia de codificación no sólo se mantiene sino que aumenta ligeramente. La eficacia de codificación aumenta en promedio en un 1,04% a 1,39%. Para ver detalles, se hace referencia a la tabla de la Fig. 13a, que muestra una representación en forma de tabla de las velocidades promedio de transmisión de bits producida por el Codificador USAC usando el codificador aritmético de la propuesta de trabajo y un codificador de audio (p. ej., Codificador de audio USAC) de acuerdo con una forma de realización de la invención.

[000231] Midiendo el nivel de carga del depósito de bits, se demostró que la codificación insonorizada propuesta puede transcodificar sin pérdidas la corriente de bits de la WD3 para cada punto operativo. Con respecto a los detalles, se hace referencia a la tabla de la Fig. 13b que muestra una representación en forma de tabla de un control de depósito de bits para un codificador de audio de acuerdo con la WD3 del USAC y un codificador de audio de acuerdo con una forma de realización de la invención.

[000232] Los detalles sobre las velocidades promedio de transmisión de bits por modo operativo, las velocidades mínimas, máximas y promedio de transmisión de bits sobre la base de un marco y un caso mejor y peor de desempeño sobre la base de un marco se pueden consultar en las tablas de las Figs. 14, 15, y 16, donde la tabla de la Fig. 14 muestra una representación en forma de tabla de las velocidades promedio de transmisión de bits para un codificador de audio de acuerdo con la WD3 del USAC y para un codificador de audio de acuerdo con una forma de realización de la invención, donde la tabla de la Fig. 15 muestra una representación en forma de tabla de las velocidades mínimas, máximas y promedio de transmisión de bits de un codificador de audio USAC sobre la base de un marco, y donde la tabla de la Fig. 16 muestra una representación en forma de tabla de casos mejores y peores sobre la base de un marco.

[000233] Además, debe tenerse en cuenta que las formas de realización de acuerdo con la presente invención proveen una buena escalabilidad. Adaptando el tamaño de tabla, se puede ajustar una correlación entre los requerimientos de memoria, la complejidad computacional y la eficacia de codificación de acuerdo con los requerimientos.

9. Sintaxis de la corriente de bits

9.1. Cargas útiles del codificador espectral insonorizado

[000234] A continuación, se explicarán algunos detalles sobre las cargas útiles del codificador espectral insonorizado. En algunas formas de realización, hay una pluralidad de diferentes modos de codificación, como por ejemplo, uno que recibe el nombre de dominio de predicción lineal, "modo de codificación" y un modo de codificación de "dominio frecuencial". En el modo de codificación llamado dominio de predicción lineal, se realiza un perfilado de ruidos sobre la base de un análisis de predicción lineal de la señal de audio, y se codifica una señal de perfilado de ruidos en el dominio frecuencial. En el modo de dominio frecuencial, se ejecuta un perfilado de ruidos sobre la base de un análisis psicoacústico y se codifica una versión de perfilado de ruidos del contenido de audio en el dominio frecuencial.

[000235] Los coeficientes espectrales tanto de una señal codificada por "dominio de predicción lineal" como de una señal codificada por "dominio frecuencial" son cuantificados por escala y luego codificados en forma insonorizada por una codificación aritmética que depende adaptativamente del contexto. Los coeficientes cuantificados se transmiten de la frecuencia más baja a la frecuencia más alta. Cada coeficiente cuantificado individual se divide en el plano discreto de 2 bits más significativos **m**, y el resto de los planos de bits menos significativos **r**. El valor **m** es codificado de acuerdo con el entorno de los coeficientes. El resto de los planos de bits menos significativos **r** son codificados por entropía, sin considerar el contexto. Los valores **m** y **r** forman los símbolos del codificador aritmético.

[000236] En la presente se describe un procedimiento de decodificación aritmética en detalle.

9.2. Elementos de sintaxis

[000237] A continuación, se describirá la sintaxis de corriente de bits de una corriente de bits que porta la información espectral aritméticamente codificada tomando como referencia las Figs. 6a a 6h.

[000238] La Fig. 6a muestra una representación sintáctica del denominado bloque de datos crudos del USAC ("usac_raw_data_block()").

[000239] El bloque de datos crudos del USAC comprende uno o varios elementos simples de canal ("single_channel_element()") y/o uno o varios elementos pares de canal ("channel_pair_element()").

[000240] Tomando como referencia ahora a la Fig. 6b, se describe la sintaxis de un elemento simple de canal. El elemento simple de canal comprende una corriente de canales de dominio de predicción lineal ("lpd_channel_stream()") o una corriente de canal de dominio frecuencial ("fd_channel_stream()") en forma dependiente del modo núcleo.

[000241] La Fig. 6c muestra una representación sintáctica de un elemento par de canal. Un elemento par de canal comprende información de modo núcleo ("core_mode0", "core_mode1"). Además, el elemento par de canal puede comprender una información de configuración "ics_info()". Adicionalmente, según la información de modo núcleo, el elemento par de canal comprende una corriente de canales de dominio de predicción lineal o una corriente de canales de dominio frecuencial asociada con uno primero de los canales, y el elemento par de canal también comprende una corriente de canales de dominio de predicción lineal o una corriente de canales de dominio frecuencial asociada con uno segundo de los canales.

[000242] La información de configuración "ics_info()", de la cual se ilustra una representación sintáctica en la Fig. 6d, comprende una pluralidad de diferentes ítems de información de configuración, que no son de particular relevancia para la presente invención.

[000243] Una corriente de canales de dominio frecuencial ("fd_channel_stream()"), de la cual se ilustra una representación sintáctica en la Fig. 6e, comprende una información de ganancia ("global_gain") y una información de configuración ("ics_info()"). Además, la corriente de canales de dominio frecuencial comprende datos de factor de escala ("scale_factor_data()"), que describen factores de escala usados para escalar los valores espectrales de diferentes bandas de factor de escala, y que son aplicados, por ejemplo, por el escalador 150 y el reescalador 240. La corriente de canales de dominio frecuencial también comprende datos espectrales codificados aritméticamente ("ac_spectral_data()"), que representan valores espectrales aritméticamente codificados.

[000244] Los datos espectrales codificados aritméticamente ("ac_spectral_data()"), de los cuales se muestra una representación sintáctica en la Fig. 6f, comprenden una bandera opcional de reiniciación aritmética ("arith_reset_flag"), que se usa para reiniciar de manera selectiva el contexto, como se describió anteriormente. Además, los datos espectrales codificados aritméticamente comprenden una pluralidad de bloques de datos aritméticos ("arith_data"), que portan los valores espectrales codificados aritméticamente. La estructura de los bloques de datos aritméticamente codificados depende de la cantidad de bandas de frecuencia (representas por la variable "num_bands") y también del estado de la bandera de reiniciación aritmética, como será analizado a continuación.

[000245] La estructura del bloque de datos aritméticamente codificados será descripta con referencia a la Fig. 6g, que muestra una representación sintáctica de dichos bloques de datos aritméticamente codificados. La representación de los datos dentro del bloque de datos aritméticamente codificados depende de la cantidad **lg** de valores espectrales que han de ser codificados, el status de la bandera de reiniciación aritmética y también del contexto, es decir, los valores espectrales previamente codificados.

[000246] El contexto para codificar el conjunto actual de valores espectrales es determinado de acuerdo con el algoritmo de determinación de contextos ilustrado con el número de referencia 660. Los detalles respecto del algoritmo de determinación de contextos han sido analizados más arriba con referencia a la Fig. 5a. El bloque de datos aritméticamente codificados comprende **lg** conjuntos de palabras de código, donde cada conjunto de palabras de código representa un valor espectral. Un conjunto de palabras de código comprende una palabra de código aritmético "acod_m [pkj][m]" que representa un valor del plano de bits más significativos **m** del valor espectral que usa entre 1 y 20 bits. Además, el conjunto de palabras de código comprende una o varias palabras de código "acod_r[r]" si el valor espectral requiere más planos de bits que el plano de bits más significativos para una correcta representación. La palabra de código "acod_r [r]" representa un plano de bits menos significativos que usa entre 1 y 20 bits.

[000247] Sin embargo, si son necesarios uno o varios planos de bits menos significativos (además del plano de bits más significativos) para una representación adecuada del valor espectral, esto es señalado por una o varias palabras de código de escape aritmético ("ARITH_ESCAPE"). Por consiguiente, se puede decir en general que para un valor espectral, se determina cuántos planos de bits (el plano de bits más significativos y, posiblemente, uno o varios planos adicionales de bits menos significativos) se necesitan. Si son necesarios uno o varios planos de bits menos significativos, esto es señalado por una o varias palabras de código de escape aritmético "acod_m [pkj][ARITH_ESCAPE]", que se codifican de acuerdo con una tabla de frecuencias acumulativas actualmente

escogida, de la cual la variable *pki* proporciona un índice de tablas de frecuencias acumulativas. Además, el contexto es adaptado, según puede observarse en los números de referencia 664, 662, si se incluyen una o varias palabras de código de escape aritmético en la corriente de bits. Luego de las una o varias palabras de código de escape aritmético, se incluye una palabra de código aritmético "acod_m [*pki*][*m*]" en la corriente de bits, indicado con el número de referencia 663, donde *pki* designa el índice de modelos de probabilidades actualmente válidos (tomando en cuenta la adaptación del contexto causada por la inclusión de las palabras de código de escape aritmético), y donde *m* designa el valor del plano de bits más significativo del valor espectral a codificar o decodificar.

[000248] Como se analizó anteriormente, la presencia de algún plano de bits menos significativos da como resultado la presencia de una o varias palabras de código "acod_r [*r*]", cada una de las cuales representa un bit del plano de bits menos significativos. Las una o varias palabras de código "acod_r [*r*]" están codificadas de acuerdo con una tabla de frecuencias acumulativas correspondiente, que es constante e independiente del contexto.

[000249] Además, debe tenerse en cuenta que el contexto es actualizado después de la codificación de cada valor espectral, indicado con el número de referencia 668, de modo tal que el contexto normalmente es diferente para codificar dos valores espectrales consecutivos.

[000250] La Fig. 6h muestra una leyenda de definiciones y elementos ayudantes que definen la sintaxis del bloque de datos aritméticamente codificados.

[000251] Para resumir lo anterior, se ha descrito un formato de corriente de bits, que puede ser provisto por el codificador de audio 100, y que puede ser evaluado por el decodificador de audio 200. La corriente de bits de los valores espectrales aritméticamente codificados está codificada de tal modo que se ajusta al algoritmo de decodificación analizado más arriba.

[000252] Además, debe observarse en general que la codificación es la operación inversa de la decodificación, de modo tal que se puede asumir en general que el codificador ejecuta una búsqueda tabular empleando las tablas ya analizadas, que es aproximadamente inversa a la búsqueda tabular ejecutada por el decodificador. En general, se puede decir que el experto en la técnica que conoce el algoritmo de decodificación y/o la sintaxis deseada de corriente de bits podrá diseñar sin problemas un codificador aritmético, que provee los datos definidos en la sintaxis de la corriente de bits y requerida por el decodificador aritmético.

10. Alternativas de implementación

[000253] Si bien se han descrito algunos aspectos del contexto de un aparato, está claro que estos aspectos también representan una descripción del método correspondiente, donde un bloque o dispositivo corresponde a un paso de método o una característica de un paso de método. De manera análoga, los aspectos descritos en el contexto de un paso de método también representan una descripción de un bloque o ítem o característica correspondiente de un aparato específico. Algunos de los pasos del método, o todos, pueden ser ejecutados (o usados) por un aparato de hardware, como por ejemplo, un microprocesador, una computadora programable o un circuito electrónico. En algunas formas de realización, uno o varios de los pasos más importantes del método pueden ser ejecutados por un aparato de ese tipo.

[000254] La señal de audio codificada de la invención puede almacenarse en un medio de almacenamiento digital o puede ser transmitida por un medio de transmisión tal como un medio de transmisión inalámbrico o un medio de transmisión cableado tal como Internet.

[000255] Según los requerimientos de la implementación, se pueden implementar las formas de realización de la invención en hardware o en software. La implementación puede ser ejecutada usando un medio de almacenamiento digital, por ejemplo un disco flexible, un DVD, un Blue-Ray, un CD, una memoria ROM, una PROM, una EPROM, una EEPROM o una FLASH, que tiene señales de control para su lectura electrónica almacenadas en aquel, que cooperan (o pueden cooperar) con un sistema de computadora programable para que se lleve a cabo el respectivo método. Por lo tanto, el medio de almacenamiento digital puede ser leído por una computadora.

[000256] Algunas formas de realización de acuerdo con la invención comprenden un portador de datos que tiene señales de control de lectura electrónica, que pueden cooperar con un sistema de computadora programable, para que se lleve a cabo uno de los métodos descritos en la presente.

[000257] En términos generales, las formas de realización de la presente invención pueden implementarse como un producto que consiste en un programa de computación con un código de programa, donde el código de programa resulta operativo para realizar uno de los métodos cuando se trabaja el producto que consiste en un programa de computación en una computadora. El código de programa puede almacenarse, por ejemplo, en un portador que puede ser leído por una máquina.

[000258] Otras formas de realización comprenden el programa de computación para ejecutar uno de los métodos descritos en la presente, almacenado en un portador que puede ser leído por una máquina.

[000259] En otras palabras, una forma de realización del método de la invención es, por lo tanto, un programa de computación que tiene un código de programa para ejecutar uno de los métodos descritos en la presente, cuando se trabaja el producto que consiste en un programa de computación en una computadora.

[000260] Otra forma de realización de los métodos de la invención es, por lo tanto, un portador de datos (o un medio de almacenamiento digital o un medio para que lea una computadora) que comprende, registrado en sí mismo, el programa de computación para ejecutar uno de los métodos descritos en la presente.

[000261] Otra forma de realización del método de la invención es, por lo tanto, una corriente de datos o una secuencia de señales que representa el programa de computación para ejecutar uno de los métodos descritos en la presente. La corriente de datos o la secuencia de señales pueden ser configuradas, por ejemplo, para ser transferidas mediante una conexión de comunicación de datos, por ejemplo vía Internet.

[000262] Otra forma de realización comprende un medio de procesamiento, por ejemplo una computadora, o un dispositivo lógico programable, configurado o adaptado para ejecutar uno de los métodos descritos en la presente.

[000263] Otra forma de realización comprende una computadora que tiene instalado en sí el programa de computación para ejecutar uno de los métodos descritos en la presente.

[000264] En algunas formas de realización, se puede usar un dispositivo lógico programable (por ejemplo una red de puertas programable por campo) para ejecutar algunas de las funcionalidades de los métodos descritos en la presente, o todas ellas. En algunas formas de realización, una red de puertas programable por campo puede cooperar con un microprocesador para ejecutar uno de los métodos descritos en la presente. En general, los métodos son ejecutados, preferiblemente, por algún aparato de hardware.

[000265] Las formas de realización antes descritas son meramente ilustrativas para los principios de la presente invención. Se entiende que las modificaciones y variaciones de las modalidades y los detalles descritos en la presente resultarán obvias a las personas expertas en la técnica. El ámbito de protección está definido por las reivindicaciones adjuntas.

11. Conclusión

[000266] A modo de conclusión, se puede hacer notar que las formas de realización de acuerdo con la invención crean un esquema mejorado de codificación espectral insonorizada. Las formas de realización de acuerdo con la nueva propuesta permiten una importante reducción de la demanda de memoria de 16894,5 palabras a 900 palabras (ROM) y de 666 palabras a 72 (RAM estática por canal de codificador de núcleo). Esto permite una reducción de la demanda de datos ROM del sistema completo en aproximadamente 43% en una forma de realización. Al mismo tiempo, el desempeño de la codificación no sólo se mantiene completamente, sino que incluso aumenta en promedio. Se ha demostrado que es posible una transcodificación sin pérdidas de la WD3 (o de una corriente de bits provista de acuerdo con la WD3 del estándar de propuestas del USAC). En consecuencia, se obtiene una forma de realización de acuerdo con la invención al adoptar la decodificación insonorizada descrita en la presente a la propuesta de trabajo del estándar de propuestas del USAC.

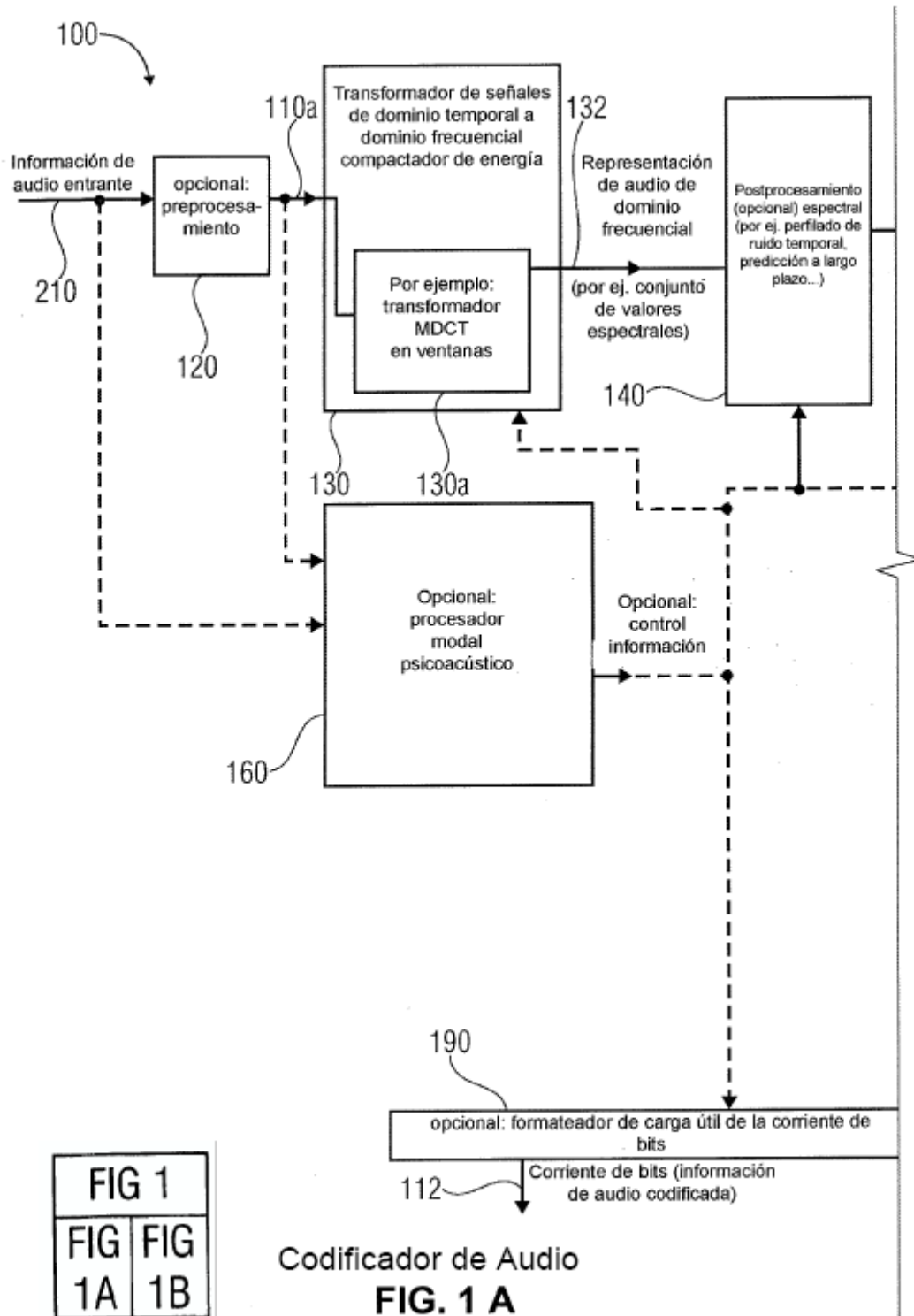
[000267] En síntesis, en una forma de realización, la nueva codificación insonorizada propuesta puede generar las modificaciones en la propuesta de trabajo MPEG del USAC con respecto a la sintaxis del elemento de la corriente de bits "arith_data()" ilustrado en la Fig. 6g, con respecto a las cargas útiles del codificador espectral insonorizado como se describió anteriormente e ilustrado en la Fig. 5h, con respecto a la codificación espectral insonorizada, como se describió anteriormente, con respecto al contexto para el cálculo de estados ilustrado en la Fig. 4, con respecto a las definiciones ilustradas en la Fig. 5i, con respecto al proceso de decodificación como se describió anteriormente con referencia a la Figs. 5a, 5b, 5c, 5e, 5g, 5h, y con respecto a las tablas ilustradas en las Figs. 17, 18, 20, y con respecto a la función "get_pk" ilustrada en la Fig. 5d. Alternativamente, sin embargo, se puede usar la tabla "ari_s_hash" de acuerdo con la Fig. 20 en lugar de la tabla "ari_s_hash" de la Fig. 17, y se puede usar la función "get_pk" de la Fig. 5f en lugar de la función "get_pk" de acuerdo con la Fig. 5d.

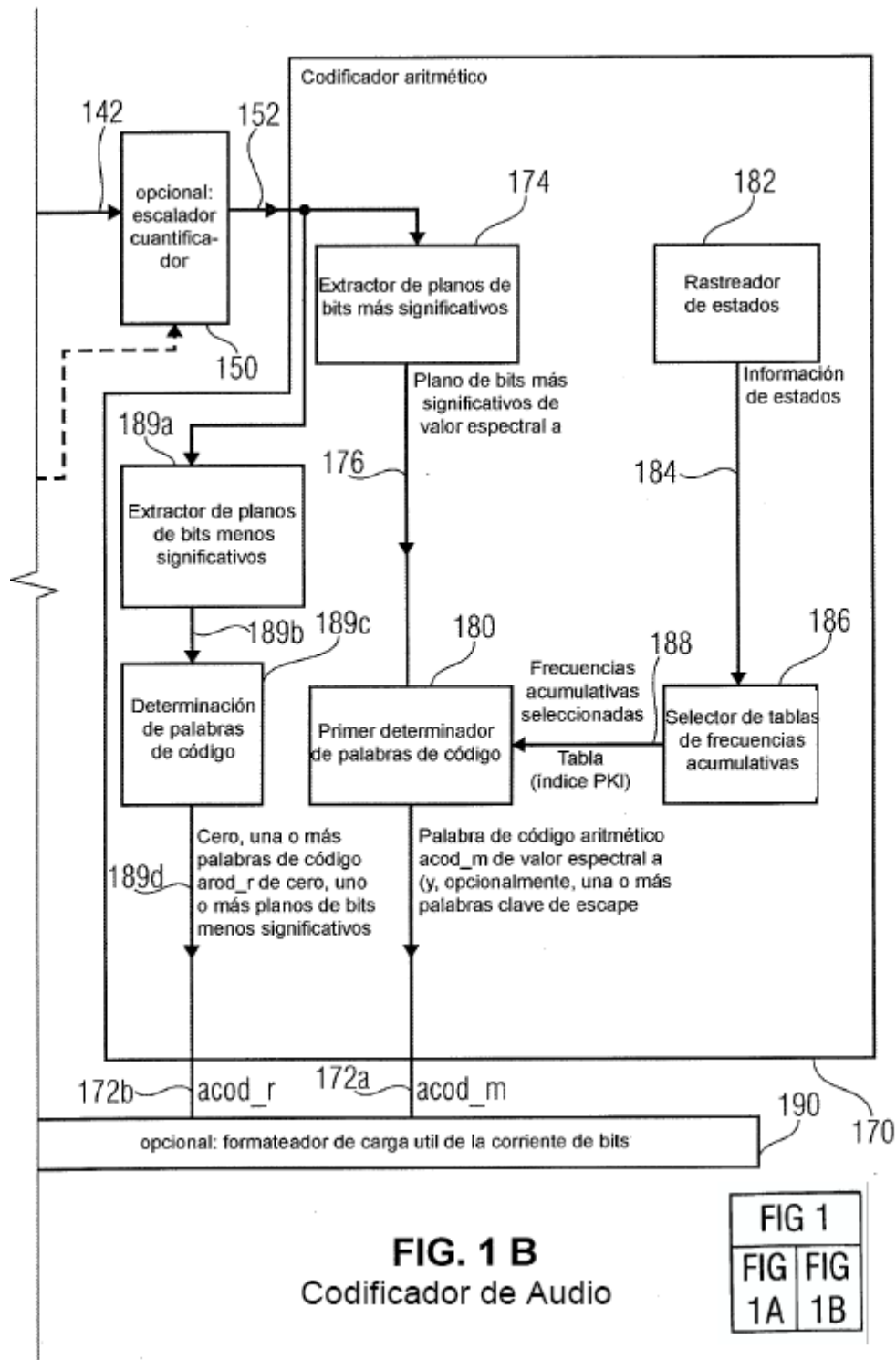
REIVINDICACIONES

1. Decodificador de audio (200;800) para proveer una información de audio decodificada (212;812) sobre la base de una información de audio codificada (210;810), que comprende:
5 un decodificador aritmético (230;820) para proveer una pluralidad de valores espectrales decodificados (232;822) sobre la base de una representación aritméticamente codificada (222;821) de los valores espectrales; y
un convertidor de dominio frecuencial a dominio temporal (260;830) para proveer una representación de audio de dominio temporal (262;812) que utiliza los valores espectrales decodificados (232;822), para obtener la información de audio decodificada (212;812);
10 en donde el decodificador aritmético (230;820) está configurado para seleccionar una norma de mapeo (297; cum_freq[]) que describe un mapeo de un valor de código (value) de la representación aritméticamente codificada a un código de símbolos (symbol) que representa uno o más de los valores espectrales decodificados, o por lo menos una porción de uno o más de los valores espectrales decodificados de manera dependiente de un estado contextual (s); y
15 en donde el decodificador aritmético (230;820) está configurado para determinar el estado contextual actual (s) de manera dependiente de una pluralidad de valores espectrales previamente decodificados,
en donde el decodificador aritmético está configurado para detectar un grupo de la pluralidad de valores espectrales previamente decodificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, y para determinar o modificar el estado contextual actual (s) de manera dependiente de un resultado obtenido con la detección.
20
2. El decodificador de audio (200; 800) según la reivindicación 1, en donde el decodificador aritmético está configurado para determinar o modificar el estado contextual actual (s) independientemente de los valores espectrales previamente decodificados en respuesta a la detección de que la condición predeterminada ha sido satisfecha.
25
3. El decodificador de audio (200; 800) según la reivindicación 1 o 2, en donde el decodificador aritmético está configurado para detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes.
30
4. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 3, en donde el decodificador aritmético (230) está configurado para detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados que, considerados individualmente o en conjunto, comprenden una magnitud que es más pequeña que una magnitud umbral predeterminada, y determinar o modificar el estado contextual actual (s) de manera dependiente de un resultado obtenido con la detección.
35
5. El decodificador de audio según una de las reivindicaciones 1 a 4, en donde el decodificador aritmético está configurado para detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados, donde cada uno de los valores espectrales previamente decodificados es un valor cero, y determinar o modificar el estado contextual (s) de manera dependiente de un resultado obtenido con la detección.
40
6. El decodificador de audio según una de las reivindicaciones 1 a 4, en donde el decodificador aritmético está configurado para detectar un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados, que comprenden un valor suma que es más pequeño que un valor umbral predeterminado, y determinar o modificar el estado actual (s) de manera dependiente de un resultado obtenido con la detección.
45
7. El decodificador de audio según una de las reivindicaciones 1 a 6, en donde el decodificador aritmético está configurado para ajustar el estado contextual actual (s) a un valor predeterminado en respuesta a la detección de que un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes.
50
8. El decodificador de audio según la reivindicación 7, en donde el decodificador aritmético (230) está configurado para omitir de manera selectiva un cálculo del estado contextual (s) en forma dependiente de valores numéricos de una pluralidad de valores espectrales previamente decodificados en respuesta a la detección de que un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes.
55
9. El decodificador de audio según una de las reivindicaciones 1 a 6, en donde el decodificador aritmético está configurado para establecer el estado contextual actual (s) dentro de un intervalo de valores que señala la detección de un grupo de una pluralidad de valores espectrales adyacentes previamente decodificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, en respuesta a la detección.
60
10. El decodificador según una de las reivindicaciones 1 a 9, en donde el decodificador aritmético está configurado para mapear un código de símbolos (symbol; **m**) a un valor espectral decodificado (a).
65

11. El decodificador de audio según una de las reivindicaciones 1 a 10, en donde el decodificador aritmético está configurado para evaluar valores espectrales previamente decodificados de una primera región de tiempo–frecuencia, para detectar un grupo de una pluralidad de valores espectrales que satisfacen, considerados individualmente o en conjunto, la condición predeterminada relacionada con sus magnitudes, y en donde el decodificador aritmético está configurado para obtener un valor numérico que representa el estado contextual (s) si la condición predeterminada no se satisface, de manera dependiente de los valores espectrales previamente decodificados de una segunda región de tiempo–frecuencia que es diferente de la primera región de tiempo–frecuencia.
12. El decodificador de audio según una de las reivindicaciones 1 a 11, en donde el decodificador aritmético está configurado para evaluar una o varias tablas hash (ari_s_hash, ari_gs_hash) para seleccionar una norma de mapeo (ari_cf_m[*pki*][9]) dependiente del estado contextual (s).
13. Codificador de audio (100;700) para proveer una información de audio codificada (112;712) sobre la base de una información de audio entrante (110;710), que comprende:
 un convertidor de dominio temporal a dominio frecuencial que compacta la energía (130;720) para proveer una representación de audio de dominio frecuencial (132;722) sobre la base de una representación de dominio temporal (110;710) de la información de audio entrante, de modo tal que la representación de audio de dominio frecuencial (132;722) comprende un conjunto de valores espectrales; y
 un codificador aritmético (170;730) configurado para codificar un valor espectral (**a**) o una versión preprocesada de aquel, usando una palabra de código de longitud variable (acod_m, acod_r), en donde el codificador aritmético (170) tiene una configuración que le permite mapear el valor espectral (**a**), o un valor (**m**) de un plano de bits más significativos del valor espectral (**a**) a un valor de código (acod_m),
 en donde el codificador aritmético está configurado para seleccionar una norma de mapeo que describe el mapeo de un valor espectral, o del plano de bits más significativos del valor espectral, al valor de código, de manera dependiente de un estado contextual (s); y
 en donde el codificador aritmético está configurado para determinar el estado contextual actual (s) de manera dependiente de una pluralidad de valores espectrales previamente codificados,
 en donde el codificador aritmético está configurado para detectar un grupo de una pluralidad de valores espectrales previamente codificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, y determinar o modificar el estado contextual actual (s) de manera dependiente de un resultado obtenido con la detección.
14. El codificador de audio (100; 700) según la reivindicación 13, en donde el codificador aritmético está configurado para determinar o modificar el estado contextual actual (s) de manera independiente de los valores espectrales previamente codificados en respuesta a la detección de que la condición predeterminada ha sido satisfecha.
15. El codificador de audio (100; 700) según la reivindicación 13 o 14, en donde el codificador aritmético está configurado para detectar un grupo de una pluralidad de valores espectrales adyacentes previamente codificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes.
16. Método para proveer una información de audio decodificada sobre la base de una información de audio codificada, que comprende:
 proveer una pluralidad de valores espectrales decodificados sobre la base de una representación aritméticamente codificada de los valores espectrales; y
 proveer una representación de audio de dominio temporal que utiliza los valores espectrales decodificados, para obtener la información de audio decodificada;
 en donde proveer la pluralidad de valores espectrales decodificados comprende seleccionar una norma de mapeo que describe un mapeo de un valor de código (acod_m; value) que representa un valor espectral, o un plano de bits más significativos del valor espectral, en una forma codificada a un código de símbolos (symbol) que representa el valor espectral, o el plano de bits más significativos del valor espectral, en una forma decodificada, de manera dependiente de un estado contextual; y
 en donde el estado contextual actual se determina de manera dependiente de una pluralidad de valores espectrales previamente decodificados,
 en donde se detecta un grupo de una pluralidad de valores espectrales previamente decodificados que satisfacen, considerados individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, y donde el estado contextual actual es determinado o modificado de manera dependiente de un resultado obtenido con la detección.
17. Método para proveer una información de audio codificada sobre la base de una información de audio entrante, que comprende:
 proveer una representación de audio de dominio frecuencial sobre la base de una representación de dominio temporal de la información de audio entrante usando una conversión de dominio temporal a dominio frecuencial

- que compacta la energía, de modo tal que la representación de audio de dominio frecuencial comprende un conjunto de valores espectrales; y
 codificar aritméticamente un valor espectral, o una versión preprocesada de aquél, que utiliza una palabra de código de longitud variable, en donde el valor espectral o un valor de un plano de bits más significativos del valor espectral es mapeado a un valor de código;
 5 en donde una norma de mapeo que describe el mapeo del valor espectral, o del plano de bits más significativos del valor espectral, al valor de código se selecciona de manera dependiente de un estado contextual; y
 en donde un estado contextual actual es determinado de manera dependiente de una pluralidad de valores espectrales adyacentes previamente codificados; y
 10 en donde se detecta un grupo de una pluralidad de valores espectrales previamente decodificados que satisfacen, individualmente o en conjunto, una condición predeterminada relacionada con sus magnitudes, y el estado contextual actual es determinado o modificado de manera dependiente de un resultado obtenido con la detección.
- 15 18. Programa de computación adaptado para ejecutar el método según la reivindicación 16 o la reivindicación 17 cuando el programa funciona en una computadora.





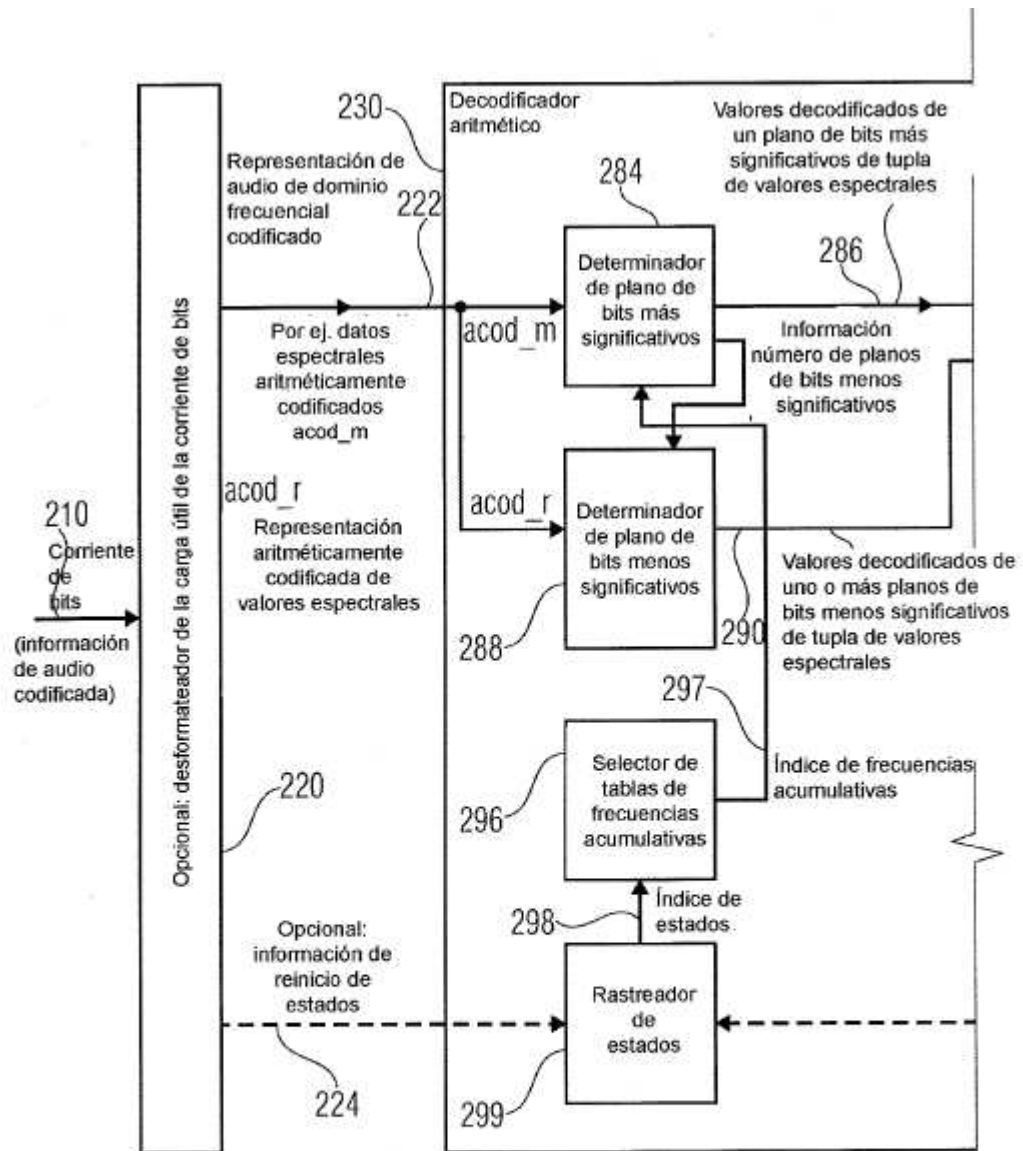
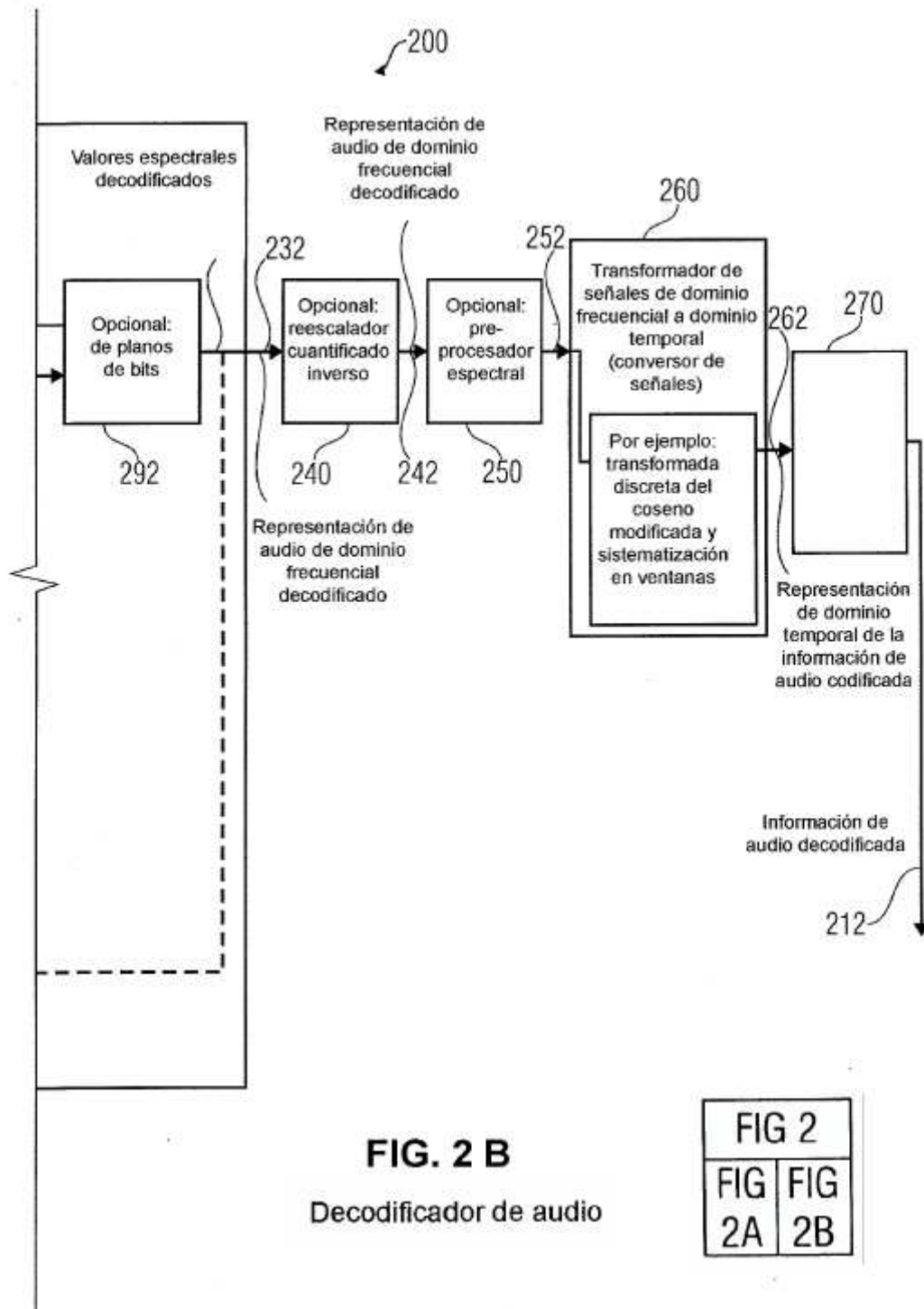


FIG 2	
FIG 2A	FIG 2B

FIG. 2 A
Decodificador de audio



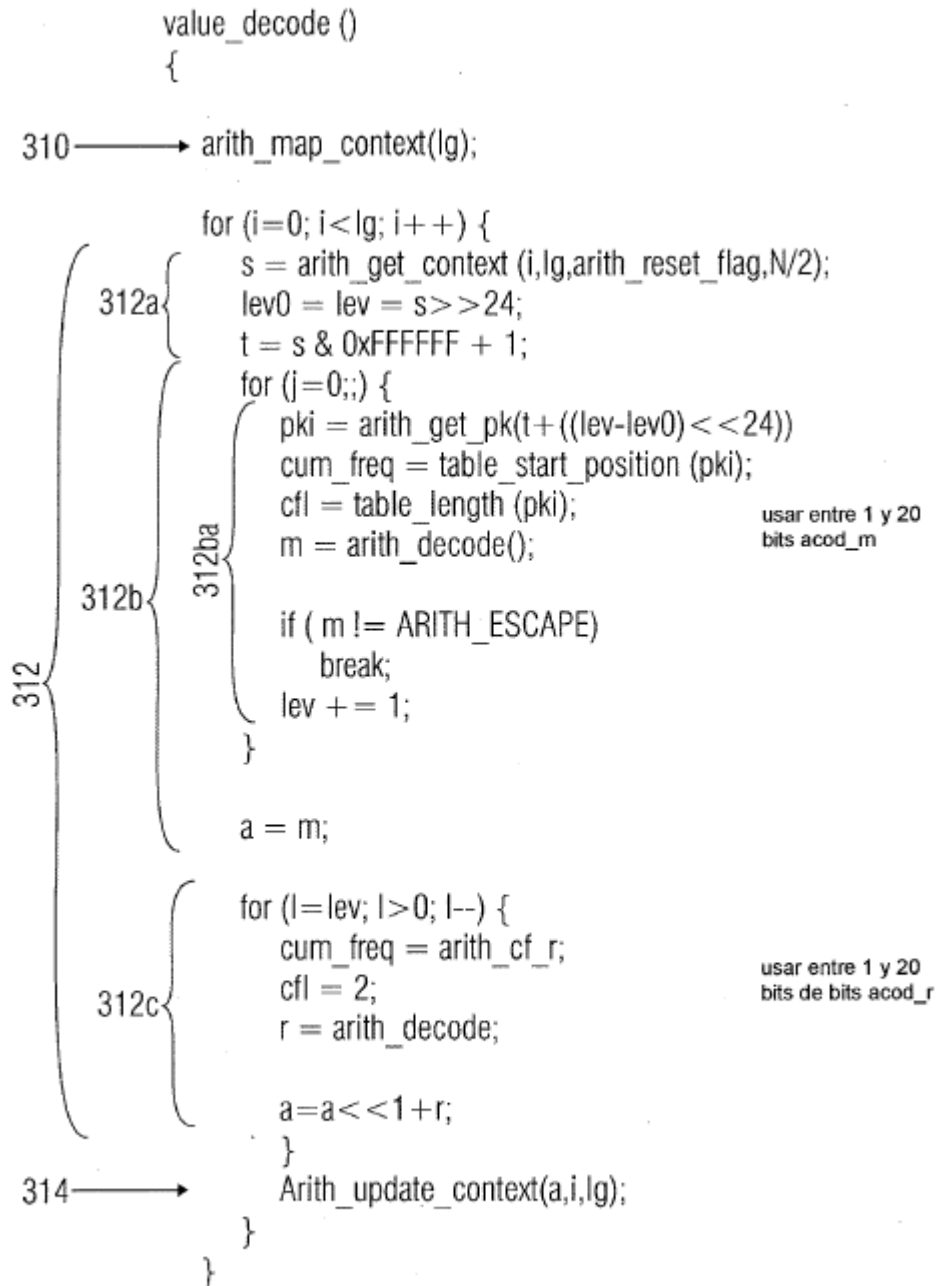
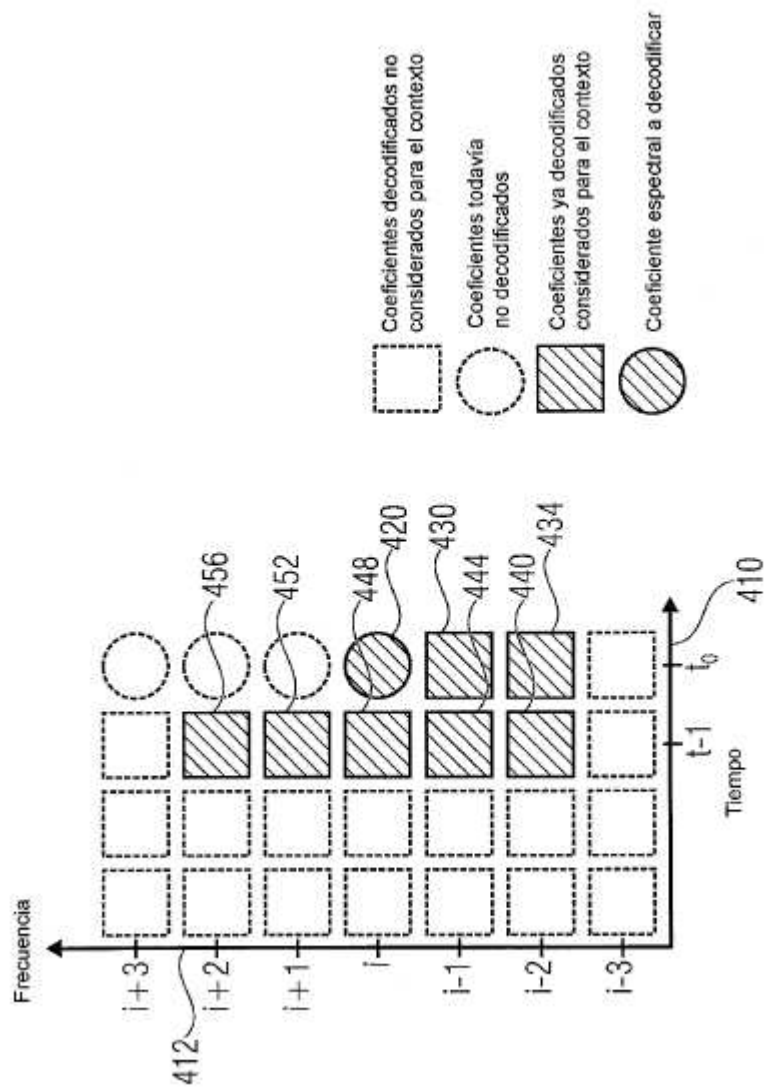


FIG. 3



```
/*Variables de entrada*/  
lg /*Número de coeficientes espectrales a decodificar en el marco*/  
previous_lg/Número previo de líneas espectrales del marco previo*/  
  
arith_map_context()  
{  
    v=w=0  
  
    ratio= ((float)previous_lg)/((float)lg);  
    for(j=0; j<lg; j++){  
        k= (int) ((float)) ((j)*ratio);  
        q[0][v++].c = qs[w+k];  
    }  
  
    previous_lg=lg;  
}
```

FIG. 5 A

```

/*Variables de entrada*/
i /*índice del valor espectral a decodificar en el vector*/
lg /*Número de coeficientes cuantificados esperados*/
N /*Número de líneas de la transformación*/
arith_reset_flag /*Bandera que indica si el contexto debe ser reiniciado*/
/*Valor de salida*/
t /*Índice de estados concatenados s y nivel de planos de bits predichos lev0*/

arith_get_context()
{
    int a0,c0,c1,c2,c3,c4,c5,c6,lev0,region;

    510 {
        if(arith_reset_flag && i==0)
            return(0);
        if((!arith_reset_flag) && (i!=0)){
            512a → int k;
                    int lim_min,lim_max;
                    int flag=1;
                    512b {
                        lim_max = i+6;
                        if((i+lim_max)>lg-1)
                            lim_max=lg-1-i;
                        lim_min = -5;
                        if((i+lim_min)<0)
                            lim_min=-i;
                    }
                    512c {
                        for(k=lim_min;k<0;k++)
                            if(q[0][k].c!=0 && q[1][k].c!=0)
                                flag=0; break;
                        for(k=<=lim_max;k++)
                            if(q[0][k].c!=0)
                                flag=0; break;
                    }
                    512d {
                        if(flag)
                            return(1);
                    }
                }
            if(i>0){
                514a {
                    a0=q[1][i-1];
                    c0=ABS(a0);
                    lev0=0;
                    while((a0<-4) || (a0>=4)){
                        514b {
                            a0>>=1;
                            lev0++;
                            c0=4+lev0;
                        }
                    }
                    514c {
                        if(c0>7)
                            c0=7;
                        if(lev0>3)
                            lev0=3;
                    }
                    514d {
                        if(arith_reset_flag && i==1)
                            return((2+c0)|(lev0<<24));
                    }
                    514e → c4=q[0][i-1].c;
                }
            }
        }
    }
}

```

<continúa en la fig. 5c>

FIG. 5 B

<continuación de la fig. 5b>

```

516 {
    if(i>1){
        c1=q[1][i-2].c;
        lev0=MAX(q[1][i-2].l,lev0);
        c6=q[0][i-2].c;
    }
    if(i>2){
518 → lev0=MAX(q[1][i-3].l,lev0);
520 {
        if(i<N/4)
            region=0;
        else if(i<N/2)
            region=1;
        else
            region=2;
    }

522 {
    if(i>3)
524 {
        lev0=MAX(q[1][i-4].l,lev0);
        if(lev0>3)
            lev0=3;
526 {
        if(arith_reset_flag)
            return((10+4*(8*c0+c1)+region)|(lev0<<24));

528 → c2=q[0][i].c;
530 {
        if(i<lg-1)
            c3=q[0][i+1].c;
        else
            c3=0;
532 {
        if(i<lg-2)
            c5=q[0][i+2].c;
        else
            c5=0;

534 {
        if(lev0==0)
            if((c2==3 || c3==3) && i==0)
                lev0=1;

536 {
536a → if(i==0)
        return((249+4*(4*c2+c3)+c5)|(lev0<<24));
536b → else if(i==1)
        return((313+4*(4*(4*(8*c0+c2)+c3)+c4)+c5)|(lev0<<24));
536c → else
        return((4212+4*(4*(4*(4*(4*(8*c0+c2)+c3)+c4)+c1)+c5)+c6)+region)
        |(lev0<<24));
    }

```

FIG. 5 C

```

unsigned long get_pk(unsigned long s)
{
    register unsigned long j;
    register long i,i_min,i_max;

    ari_get_pk_call_total++; ----- opcional

    541 {
        i_min=-1;
        i=i_min;
        i_max=386;
        while((i_max-i_min)>1){
            542a → i=i_min+((i_max-i_min)/2);
            542b → j=ari_s_hash[i];
                ari_get_pk_inc++; ----- opcional
                if(s<(j>>8))
                    i_max=i;
                else if(s>(j>>8))
                    i_min=i;
                else
                    return(j&0xFF);
        }
        if(i_max==i){
            j=ari_s_hash[i_min];
            ari_get_pk_inc++; ----- opcional
            if(s==(j>>8))
                return(j&0xFF);
        }
        543 {
            else{
                j=ari_s_hash[i_max];
                ari_get_pk_inc++; -----
                if(s==(j>>8))
                    return(j&0xFF);
            }
        }
    }
}

```

FIG. 5 D1

FIG 5D1	FIG
FIG 5D2	5D

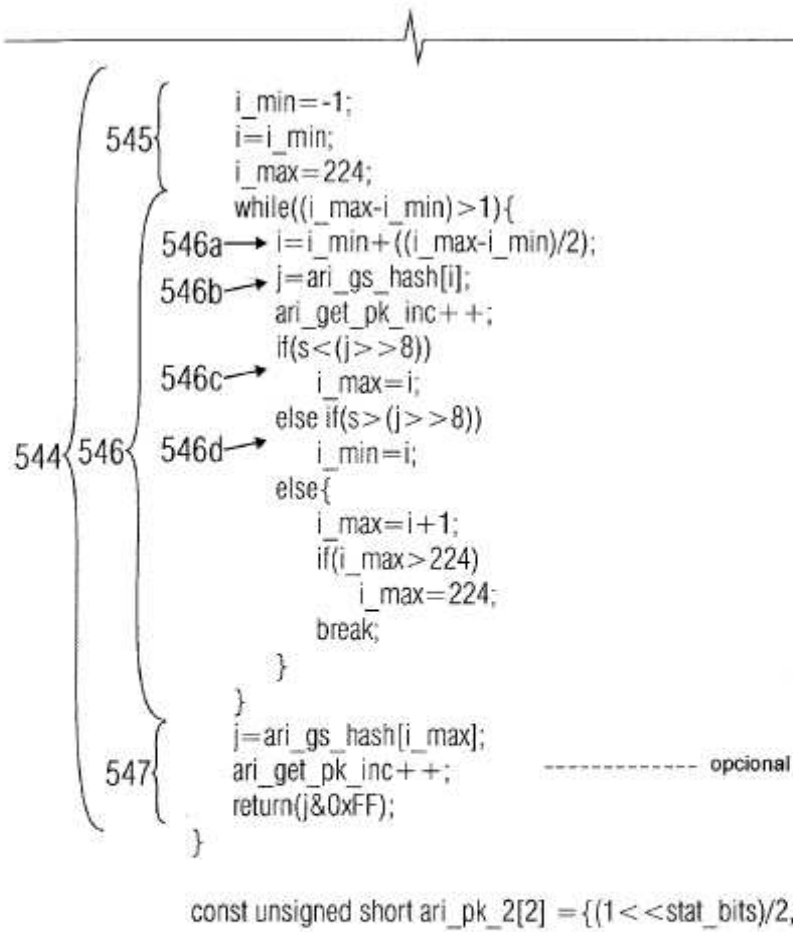


FIG. 5 D2

FIG 5D1	FIG
FIG 5D2	5D

```

/*Variable de entrada*/
s /*Estado del contexto*/
/*Valor de salida*/
pki /*índice del modelo de probabilidades*/

arith_get_pk(s)
{
    register unsigned long i,j;

    550 {
        for (i=0;i<387;i++)
        {
            j=ari_s_hash[i];
            if ( (j>>8)==s ) return j&255;
        }
        560 {
            for (i=0;i<225;i++)
            {
                j=ari_gs_hash[i];
                if ( s<(j>>8) ) return j&255;
            }
        }
        return j&255;
    }
}

```

FIG. 5E

```

unsigned long get_pk(unsigned long s)
{
    register unsigned longlong j;
    register unsigned long i;

    for (i=0;i<387;i++)
    {
        j=ari_s_hash[i];
        if ( (j>>8)==s )
            return j&0xFF;
    }
    for(i=0;i<225;i++){
        j=ari_gs_hash[i];
        if ( s<(j>>8) ) return j&0xFF;
    }
    return(j&0xFF);
}

```

FIG. 5F


```

/*Funciones auxiliares*/
bool arith_first_symbol(void);
/*Retomar VERDADERO si es el primer simbolo de la secuencia, de lo contrario FALSO*/
Ushort arith_get_next_bit(void);
/* Ir al bit siguiente de la corriente de bits */

/* Variables globales*/
low
high
value

/* Variables de entrada */
cum_freq[]; /* Tabla de frecuencias acumulativas*/
cfl; /* longitud de cum_freq[] */

arith_decode()
{
    if(arith_first_symbol())
    {
        value = 0;
        for (i=1; i<=20; i++)
        {
            value = (val<<1) | arith_get_next_bit();
        }
        low=0;
        high=1048575;
    }

    range = high-low+1;
    cum = (((int64) (value-low+1))<<16)-((int64) 1))/((int64) range);
    p = cum_freq-1;

    do
    {
        q=p+(cfl>>1);
        if ( *q > cum ) {p=q; cfl++; }
        cfl>>=1;
    }
    while ( cfl>1 );
}

```

570a {

570b {

570c {

FIG. 5 G1

FIG 5G1	FIG
FIG 5G2	5G

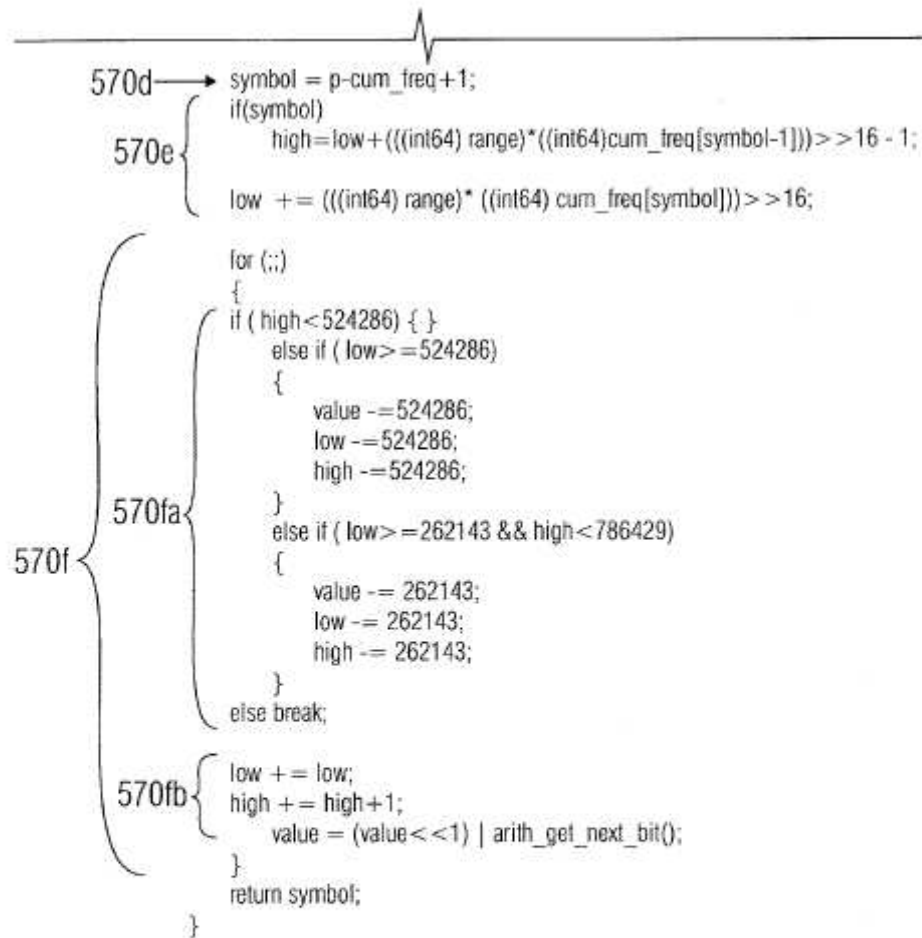
**FIG. 5 G2**

FIG 5G1	FIG
FIG 5G2	5G

```

/*Variables de entrada*/
a /*Coeficiente espectral cuantificado decodificado*/
i /*índice del coeficiente espectral cuantificado a decodificar*/
lg/*Número de coeficientes en el marco*/

arith_update_context()
{
    int a0;

580 → q[1][i]=a0=a;
    q[1][i].l=0;
582 { while(ABS(a0)>4){
        a0=a0>>1;
        q[1][i].l++;
    }
584 { if(a==0)
        q[1][i].c=0;
        else if(ABS(a)<=1)
            q[1][i].c=1;
        else if(ABS(a)<=3)
            q[1][i].c=2;
        else
            q[1][i].c=3;

586 { if(i==lg && core_mode==1){
            ratio = ((float) lg)/((float)1024);
            for(j=0; j<1024; j++){
                k = (int) ((float) j*ratio);
                qs[j] = q[1][k].c;
            }
            previous_lg = 1024;
        }
588 { if(i==lg/4 && core_mode==0){
            for(j=0; j<MIN(lg,1024; j++){
                qs[j] = q[1][i].c;
            }
            previous_lg = MIN(1024,lg);
        }
    }
}

```

FIG. 5 H

Definiciones

a	El coeficiente cuantificado a decodificar
m	El plano discreto de dos bits más significativos del coeficiente espectral cuantificado a decodificar
r	El plano discreto de dos bits más significativos del coeficiente espectral cuantificado a decodificar
lev	Nivel del resto de planos de bits. Corresponde al número de planos de bits menos significativos que el plano discreto de dos bits más significativo
$lev0$	Nivel de planos de bits predicho
$arith_s_hash[]$	Estados de mapeo de tabla hash del contexto a un índice de tablas de frecuencias acumulativas pki
$arith_gs_hash[]$	Grupo de mapeo de tabla hash de los estados de contexto a un índice de tablas de frecuencias acumulativas pki
$arith_cf_m[pki][9]$	Modelos de las frecuencias acumulativas para el plano discreto de dos bits más significativos m y el símbolo ARITH_ESCAPE
$arith_cf_r[]$	Frecuencias acumulativas para el símbolo de planos de bits menos significativos r
$previous_lg$	Número de coeficientes espectrales transmitidos previamente decodificados por el decodificador aritmético
N	Longitud de ventana. Para AAC se deduce de $window_sequence$ (ver sección 6.8.3.1) y para TCX $ll=2.lg$
$q[2] []$	El contexto actual para el coeficiente espectral a decodificar
$qs[]$	El contexto pasado almacenando para el próximo marco
$arith_reset_flag$	Bandera que indica si el contexto espectral insonorizado debe ser reiniciado

FIG. 5 I

```

usac_raw_data_block ()
{
    single_channel_element (); y/o
    channel_pair_element ();
}

```

FIG. 6A

Sintaxis de **single_channel_element()**

Sintaxis	Nº de bits	Mnemotécnica
<pre> single_channel_element() { core_mode if (core_mode == 1) { lpd_channel_stream(); } else { fd_channel_stream(); } } </pre>	1	uimsbf

FIG. 6B

Sintaxis de **channel_pair_element()**

Sintaxis	N° de bits	Mnemotécnica
channel_pair_element()		
{		
core_mode0	1	uimsbf
core_mode1	1	uimsbf
ics_info();		opcional: ics_info común para dos canales
if (core_mode0 == 1) {		
lpd_channel_stream();		
}		
else {		
fd_channel_stream();		
}		
if (core_mode1 == 1) {		
lpd_channel_stream();		
}		
else {		
fd_channel_stream();		
}		
}		

FIG. 6C

Sintaxis de **fd_channel_stream()**

Sintaxis	N° de bits	Mnemotécnica
fd_channel_stream() { global_gain; ics_info(); scale_factor_data (); ac_spectral_data (); }	8	uimsbf (a menos que esté incluido en elemento par de canales)

FIG. 6ESintaxis de **ac_spectral_data()**

Sintaxis	N° de bits	Mnemotécnica
ac_spectral_data() { arith_reset_flag for (win=0; win<num_windows; win++){ arith_data(num_bands, arith_reset_flag) } }	1	uimsbf

FIG. 6F

Sintaxis de **arith_data()**

Sintaxis	N° de bits	Mnemotécnica
<pre> Arith_data(lg, arith_reset_flag) { arith_map_context(lg); for (i=0; i<lg; i++) { s = arith_get_context (i,lg,arith_reset_flag,N/2); lev0 = lev = s > 24; t = s & 0xFFFFFFF + 1; for (j=0;;) { pki = arith_get_pk(t+((lev-lev0)<24)) acod_m[pki][m] if (m != ARITH_ESCAPE) break; lev += 1; } a=m; for (l=lev; l>0; l--) { acod_r[l] a=a<1+r; } Arith_update_context(a,i,lg); } } </pre>	1..20 1..20	Vlclbf vclbf

FIG. 6G

Definiciones	
<code>arith_data()</code>	Elemento de información para decodificar los datos del codificador espectral insonorizado
<code>arith_reset_flag</code>	Bandera que indica si el contexto espectral insonorizado debe reiniciarse
<code>acod_cf_m[pki][a]</code>	Palabra de código aritmética necesaria para la decodificación aritmética del plano discreto de dos bits más significativos del coeficiente espectral cuantificado
<code>arith_cf_r[]</code>	Palabra de código aritmética necesaria para la decodificación aritmética de los planos de bits residuales del coeficiente espectral cuantificado
Elementos auxiliares	
<code>a</code>	EL coeficiente espectral cuantificado a decodificar
<code>m</code>	El plano discreto de dos bits más significativos del coeficiente espectral cuantificado a decodificar
<code>r</code>	El plano discreto de dos bits más significativos del coeficiente espectral cuantificado a decodificar
<code>N</code>	Longitud de ventana. Para AAC se deduce de la <code>window_sequence</code> (ver sección 6.8.3.1) y para TCX $N = 2 \cdot lg$
<code>lg</code>	Número de coeficientes cuantificados a decodificar
<code>i</code>	Índice de los coeficientes cuantificados a decodificar dentro del marco
<code>pki</code>	Índice de la tabla de frecuencias acumulativas usadas por el decodificador aritmético para decodificar a
<code>arith_get_pk()</code>	Función que retorna el índice pki de la tabla de frecuencias acumulativas necesarias para decodificar la palabra de código <code>acod_ng[pki][a]</code>
<code>t</code>	Estado de contexto
<code>arith_get_context()</code>	Función que retorna al estado del contexto
<code>lev()</code>	Nivel predicho de planos de bits
<code>s</code>	Estado del contexto combinado con el nivel predicho de plano de bits <code>lev()</code>
<code>lev</code>	Nivel de planos de bits para decodificar más allá del plano discreto de dos bits más significativos
<code>ARITH_ESCAPE</code>	Símbolo de escape que indica planos de bits adicionales para decodificar más allá del nivel de planos de bits predicho <code>lev()</code>

FIG. 6H

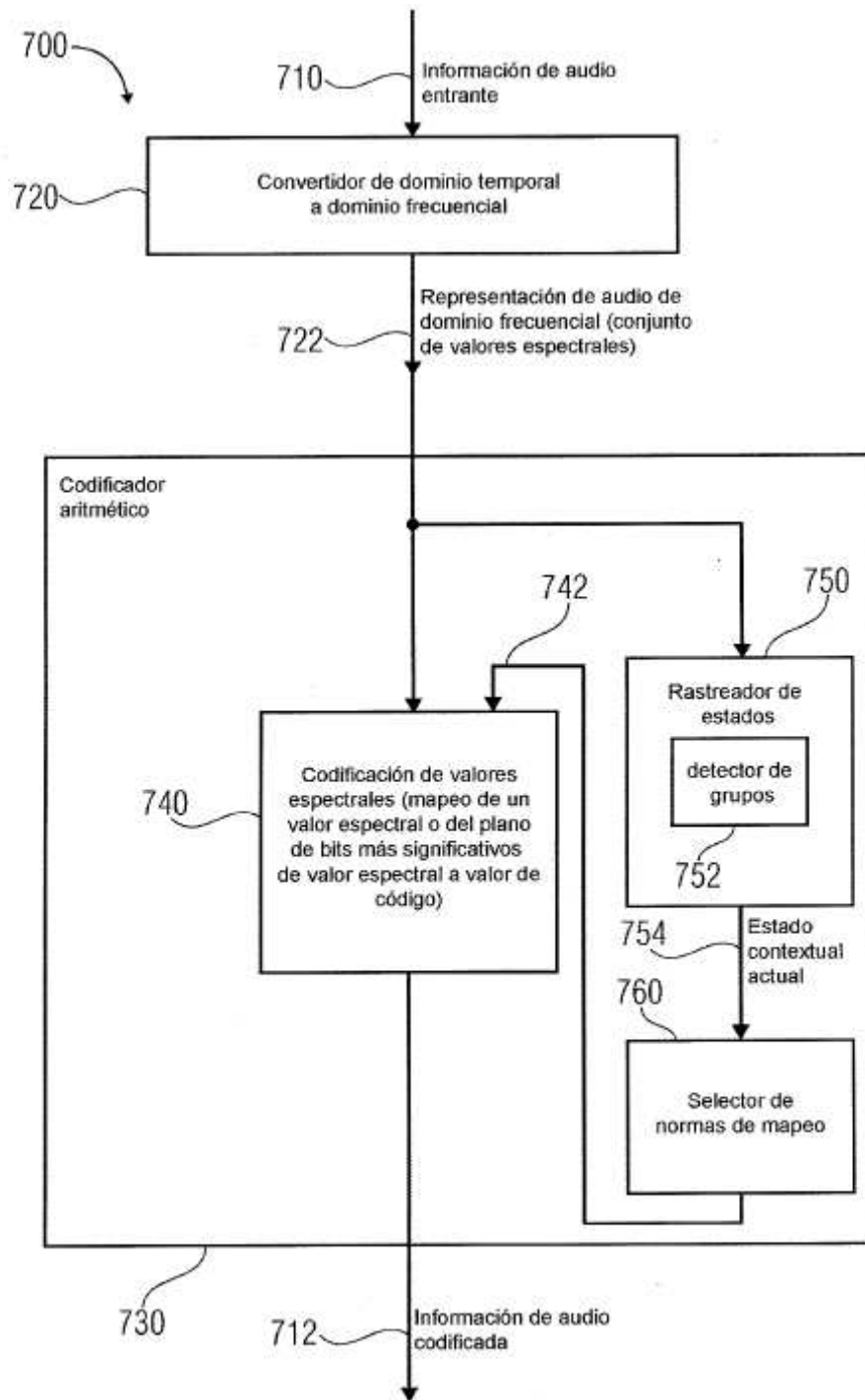


FIG. 7

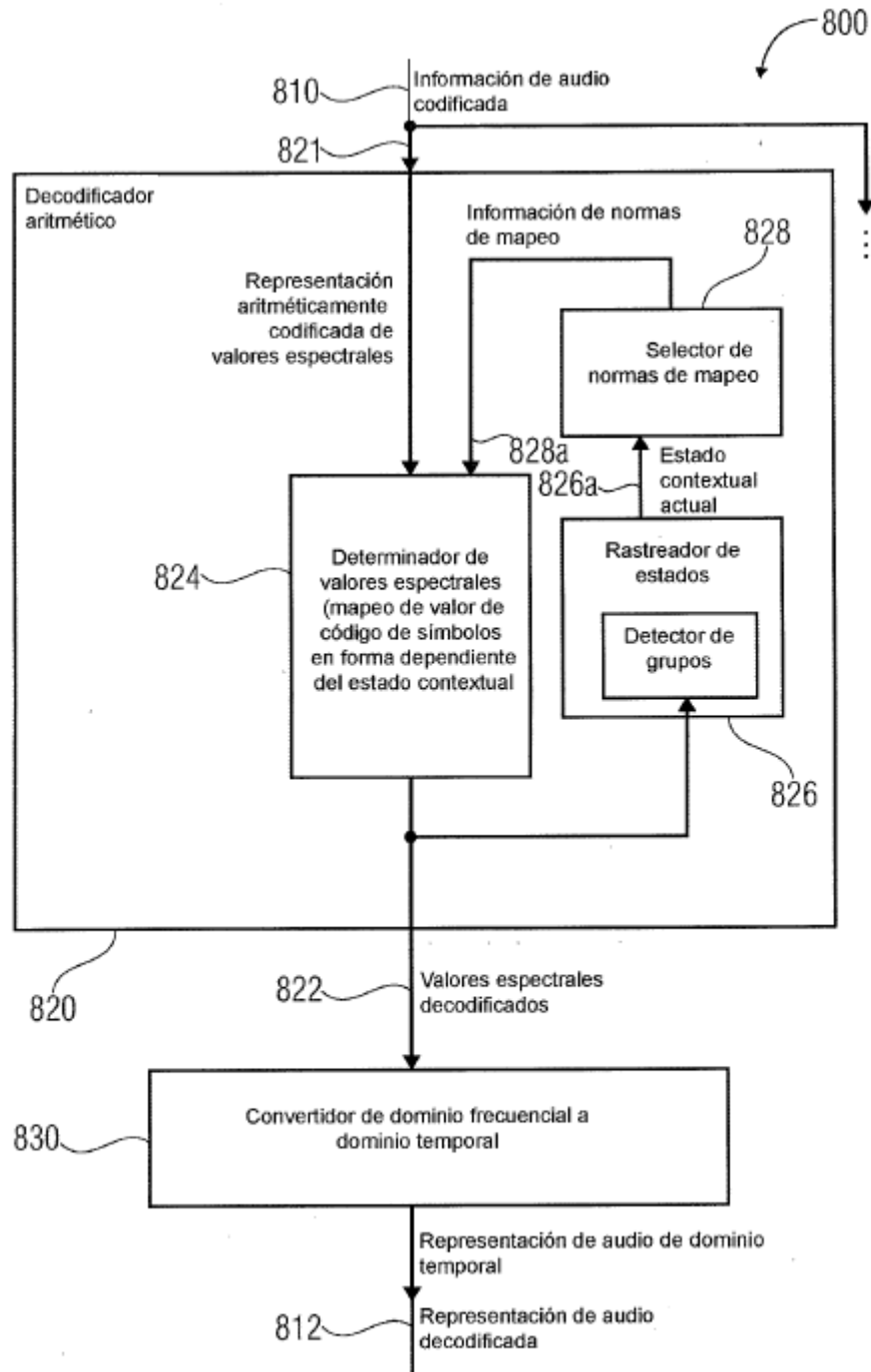


FIG. 8

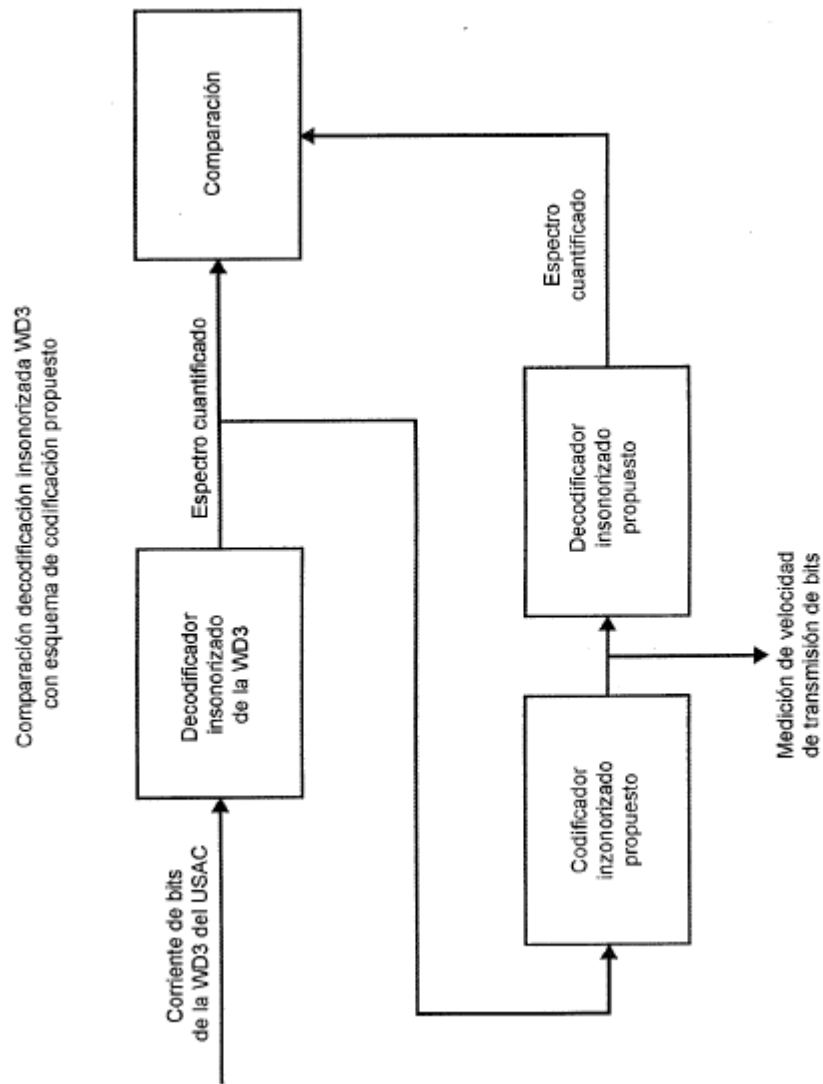


FIG. 9

contexto para cálculo de estado,
usado en USAC WD4

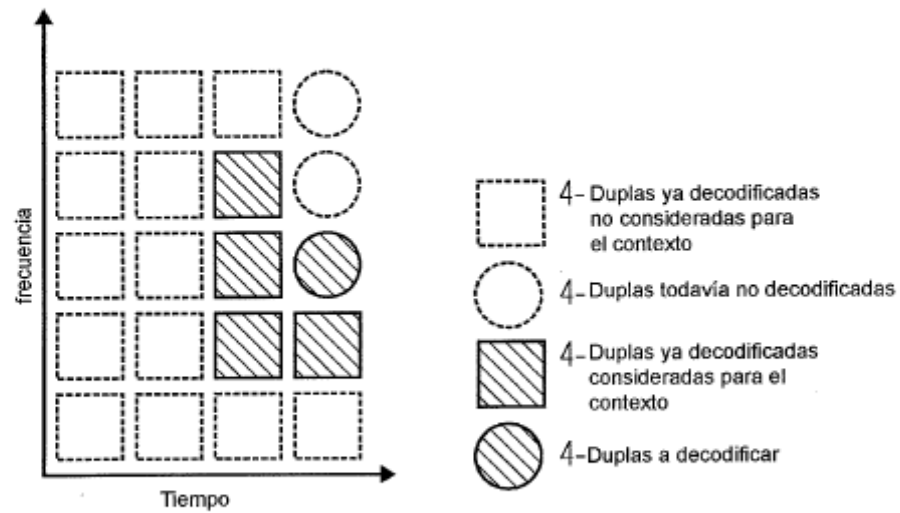


FIG. 10 A

contexto para cálculo de estado,
usado en el esquema propuesto

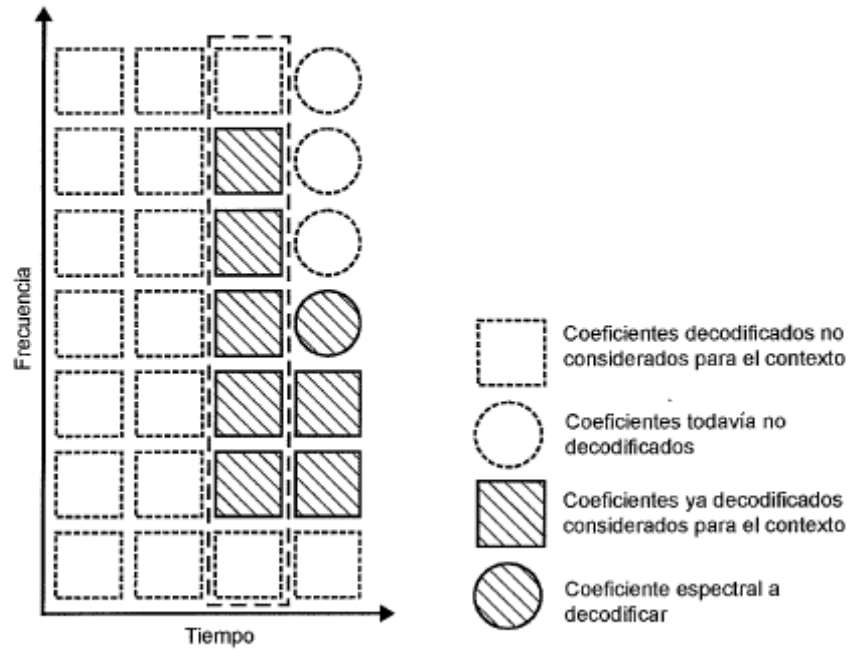


FIG. 10 B

Tablas usadas en el Esquema de Codificación Aritmética de la WD4 del USAC

Nombre de tabla	Descripción	Unidad de datos	Memoria (palabras de 32 bits)
arith_cf_ng_hash[128]	Contexto de mapeo de tabla Hash a un índice de modelos de probabilidades	palabra	128
arith_cf_ng[32][545]	Frecuencias acumulativas de grupos para cada modo de distribución de probabilidades, 1	1/2 palabra	8720
egroups[8][8][8][8] dgvector[4*4096]	Índice de grupos de 4 tuplas Índice de grupos de mapeo e índice de elementos a 4 tuplas	1/2 palabra 1/4 palabra	2048 4096
dgroups[544]	Índice de grupos de mapeo al cardinal del grupo y compensación en los vectores dg	palabra	544
arith_cf_ne[2701]	Frecuencias acumulativas del símbolo de índice de elementos	1/2 palabra	1350.5
arith_cf_r[16]	Frecuencias acumulativas de los planos de bits menos significativos	1/2 palabra	8
total			16894.5

FIG. 11 A

Tablas usadas en el Esquema de Codificación Propuesto

Nombre de tabla	Descripción	Unidad de datos	Memoria (palabras de 32 bits)
arith_s_hash[387]	Estados de mapeo de tabla Hash del contexto a una tabla de frecuencias acumulativas	palabra	387
arith_gs_hash[225]	Grupo de mapeo de tabla Hash de estados de contexto a una tabla de frecuencias acumulativas	palabra	225
arith_cf_m[64][9]	Modelos de las frecuencias acumulativas para el plano discreto de dos bits más significativos M y el símbolo ARITH_ESCAPE	1/2 palabra	288
total			900

FIG. 11 B

Esquema de Codificación insonorizada de la
demanda ROM como se propone y en la WD4

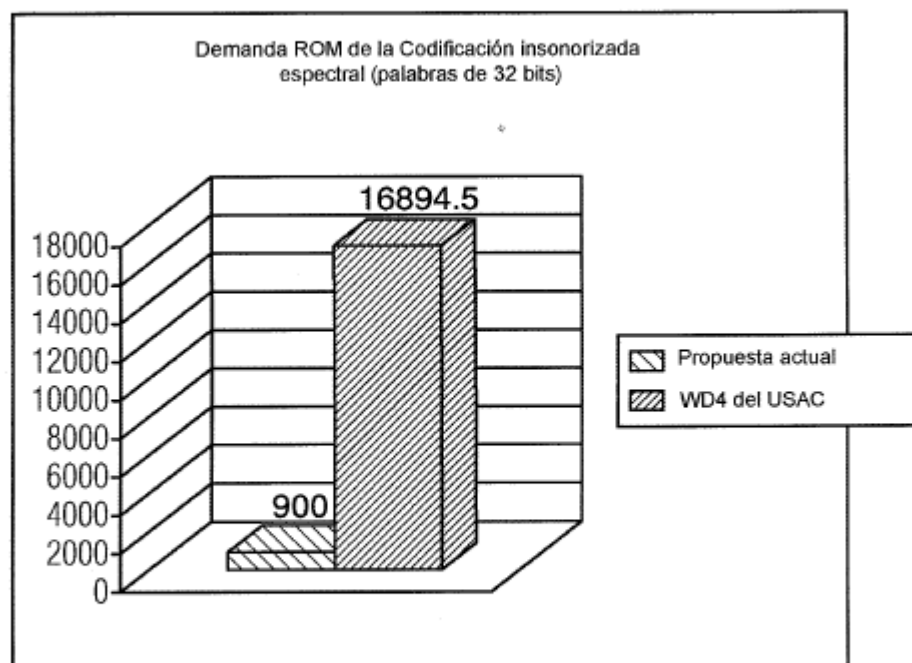


FIG. 12 A

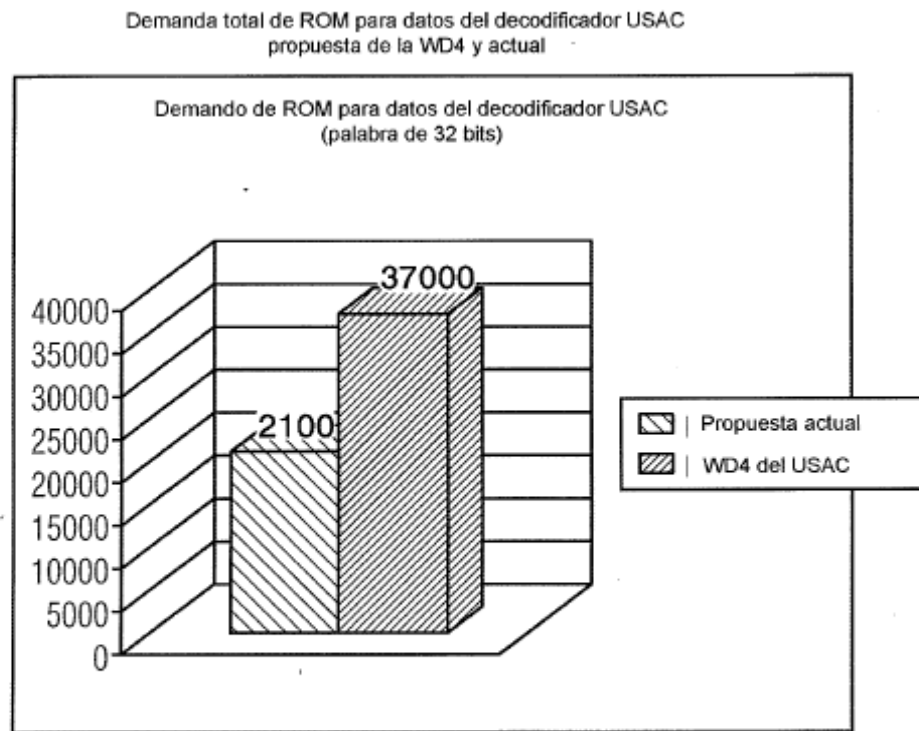


FIG. 12 B

Velocidades de transmisión de bits promedio para codificador USAC usando codificador aritmético WD y nueva propuesta

Modo operativo	WD (kbit/s)	Nueva propuesta (kbit/s)	Diferencia después de la transcodificación (kbit/s)	Diferencia después de la transcodificación (% de velocidad de transmisión de bits total)
Test 1, 64kbps estereo	64.00	63.34	-0.66	-1.04
Test 2, 32kbps estereo	32.00	31.66	-0.34	-1.05
Test 3, 24kbps estereo	24.00	23.73	-0.27	-1.11
Test 4, 20kbps estereo	20.00	19.78	-0.22	-1.11
Test 5, 16kbps estereo	16.00	15.82	-0.18	-1.10
Test 6, 24kbps mono	24.00	23.68	-0.32	-1.32
Test 7, 20kbps mono	20.00	19.72	-0.28	-1.39
Test 8, 16kbps mono	16.00	15.79	-0.21	-1.31
Test 9, 12kbps mono	12.00	11.86	-0.14	-1.19

FIG. 13 A

Control de reserva de bits para la WD3 del USAC y para la nueva propuesta

modo operativo	Control de reserva de bits					
	Nueva propuesta			WD		
	min	max	avg	min	max	avg
Test 1, 64kbps estereo	3653	9557	8137	2314	9557	7018
Test 2, 32kbps estereo	1808	4505	4196	581	4505	3530
Test 3, 24kbps estereo	1538	4704	4408	957	4704	3871
Test 4, 20kbps estereo	2367	4864	4600	712	4864	3854
Test 5, 16kbps estereo	2712	5006	4804	724	5006	4234
Test 6, 24kbps mono	2185	4704	4457	1002	4704	3927
Test 7, 20kbps mono	2599	4864	4630	1192	4864	3935
Test 8, 16kbps mono	2820	5006	4876	1434	5006	4450
Test 9, 12kbps mono	3529	5184	5081	2256	5184	4787

FIG. 13 B

Velocidades de transmisión de bits promedio para WD3 del USAC y para la nueva propuesta

Modo operativo	Velocidades de transmisión de bits promedio en KBIT/S					
	Nueva propuesta			WD		
	FD modo	wLPT modo	total	FD modo	wLPT modo	total
Test 1, 64kbps stereo	53.73	---	53.73	54.40	---	54.40
Test 2, 32kbps stereo	25.31	26.34	25.60	25.80	26.61	26.02
Test 3, 24kbps stereo	18.27	19.17	18.50	18.66	19.40	18.85
Test 4, 20kbps stereo	15.50	15.93	15.61	15.83	16.12	15.90
Test 5, 16kbps stereo	12.45	12.60	12.52	12.80	12.73	12.77
Test 6, 24kbps mono	19.94	19.51	19.73	20.41	19.42	20.15
Test 7, 20kbps mono	16.15	15.91	16.08	16.56	16.12	16.45
Test 8, 16kbps mono	13.02	12.59	12.81	13.45	12.73	13.09
Test 9, 12kbps mono	9.35	9.66	9.51	9.68	9.71	9.70

FIG. 14

Velocidades de transmisión de bits mínimas, máxima y promedio del USAC sobre una base de marcos

modo operativo	velocidad de transmisión de bits mínima (kbit/s)	Velocidad de transmisión de bits máxima (kbit/s)	Velocidad de transmisión de bits promedio (kbit/s)
Test 1, 64kbps stereo	15.26	101.79	63.34
Test 2, 32kbps stereo	13.13	48.61	31.66
Test 3, 24kbps stereo	11.69	36.58	23.73
Test 4, 20kbps stereo	3.09	30.94	19.78
Test 5, 16kbps stereo	4.02	26.47	15.82
Test 6, 24kbps mono	1.47	37.35	23.68
Test 7, 20kbps mono	1.38	31.13	19.72
Test 8, 16kbps mono	11.40	24.64	15.79
Test 9, 12kbps mono	8.72	18.91	11.86

FIG. 15

Caso mejor y caso peor sobre una base de marcos

Modos operativos	Caso mejor		Caso peor	
	(bit/s)	(%)	(bit/s)	(%)
Test 1, 64kbps stereo	-30.87	-33.06	6.14	9.07
Test 2, 32kbps stereo	-10.33	-28.63	2.17	6.77
Test 3, 24kbps stereo	-11.86	-30.75	1.85	7.71
Test 4, 20kbps stereo	-7.45	-30.27	1.67	8.36
Test 5, 16kbps stereo	-5.43	-27.89	1.50	9.42
Test 6, 24kbps mono	-17.06	-45.83	1.25	4.36
Test 7, 20kbps mono	-15.86	-41.46	0.88	3.38
Test 8, 16kbps mono	-4.75	-24.85	1.11	7.31
Test 9, 12kbps mono	-3.95	-26.33	0.82	6.99

FIG. 16

```

/*
Entropía:
fu mem.: 1.2792 bit (100.00 %)
no mem. : 1.6289 bit (127.34 %)
split:   : 1.2971 bit (101.40 %)
*/

```

FIG 17(1)

```

/* 1224 Entropía, aumento de entropía : 0.000384 */

```

```

/* Entropía final : 1.297556 */

```

```

/* Estados totales = 612; */
/* Estados significativos = 387; */
/* pseudo estados = 225; */
/* modelos de prueba 64; */

```

```

unsigned long long ari_get_pk_inc=0;
unsigned long long ari_get_pk_call_total=0;

```

```

static unsigned long ari_s_hash[387] = {
0x00000200,0x00000B01,0x00000C02,0x00000D03,0x00000F25,0x0000101C,0x0000110B,
0x00001327,
0x0000142F,0x00002B25,0x00002C22,0x00002D14,0x00002F2D,0x0000302B,0x0000312B,
0x00003330,
0x00003432,0x00003532,0x00004C32,0x00005031,0x00005131,0x0000FA02,0x0000FB01,
0x0000FC1C,
0x0000FE1C,0x0000FF1C,0x0001001E,0x00010A2E,0x00010B25,0x00010E25,0x00010F25,
0x00013938,
0x00013A04,0x00013B02,0x00013C01,0x0010393D,0x00107504,0x00107605,0x00107706,
0x0010790D,
0x00107A07,0x00107B08,0x0010850D,0x00108609,0x0010870A,0x00108B0E,0x0010B50B,
0x0010B60C,
0x0010B70D,0x0010B90B,0x0010BA1D,0x0010BB16,0x0010C615,0x0010C70C,0x0010F521,
0x0010F628,
0x0010F728,0x00110528,0x00117516,0x0011760E,0x0011770F,0x00117A12,0x00117B07,
0x0011870E,
0x0011B514,0x0011B615,0x0011B70C,0x0011B914,0x0011BA15,0x0011BB1D,0x0011C619,
0x0011C715,
0x00147516,0x00147610,0x00147711,0x0014791D,0x00147A0C,0x00147B0E,0x0014851B,
0x00148616,
0x00148707,0x0014890B,0x00148A1D,0x00148B16,0x0014950B,0x0014961D,0x0014B615,
0x0014B71D,
0x0014BA14,0x0014BB15,0x0014C614,0x0014C715,0x0015052D,0x0015750B,0x0015760C,
0x00157710,
0x0015790B,0x00157A1D,0x00157B16,0x0015850B,0x0015861D,0x0015870C,0x00158914,
0x00158A15,
0x00158B1D,0x0015B619,0x0015B714,0x0020751D,0x00207612,0x00207713,0x0020791D,
0x00207A0C,
0x00207B0E,0x0020850B,0x0020860C,0x00208710,0x00208A1D,0x00208B0C,0x0020B615,
0x0020B71D,
0x0021750B,0x0021760C,0x0021770E,0x0021790B,0x00217A1D,0x00217B12,0x00218514,
0x00218615,
0x0021870C,0x0021891C,0x00218A15,0x00218B1D,0x0021B619,0x0021B715,0x0021BA22,
0x0021BB19,
0x00247514,0x0024761D,0x0024770E,0x00247914,0x00247A15,0x00247B0C,0x00248514,
0x0024861D,
0x00248716,0x0024891C,0x00248A15,0x00248B1D,0x0024B619,0x0024B715,0x0025751C,
0x0025760B,

```

FIG	FIG 17(1)
17	FIG 17(2)


```

0x0025771B, 0x0025791C, 0x00257A0B, 0x00257B1B, 0x0025851C, 0x0025860B, 0x0025871B,
0x0025890B,
0x00258A1D, 0x00258B1B, 0x00258B618, 0x00258B722, 0x00258B1F, 0x00258B18, 0x00258C61F,
0x0025C718,
0x0025CA1F, 0x0025CB1F, 0x003A9D1C, 0x003A9E0B, 0x003A9F0B, 0x004FB125, 0x004FB21C,
0x004FB31C,
0x00907514, 0x00907615, 0x00907716, 0x00907919, 0x00907A15, 0x00907B1D, 0x00907D1C,
0x00907E1C,
0x00907F14, 0x00908522, 0x00908614, 0x0090871D, 0x00908918, 0x00908A19, 0x00908B14,
0x00908D24,
0x00909523, 0x0090961F, 0x0090992B, 0x00909B517, 0x00909B618, 0x00909B719, 0x00909B91F,
0x00909BA22,
0x00909BB19, 0x00909BD1C, 0x00909BE1C, 0x00909BF1C, 0x00909C52B, 0x00909C61F, 0x00909C718,
0x00909C917,
0x00909CA1F, 0x00909CB1F, 0x00909CD23, 0x00909D52E, 0x00909D62C, 0x00909D92C, 0x00909F52D,
0x00909F62D,
0x00909F72F, 0x00909F925, 0x00909FA2E, 0x00909FB2D, 0x00909FD1E, 0x00909FE1E, 0x0091052F,
0x0091062F,
0x00910928, 0x00910D25, 0x00917519, 0x00917615, 0x0091771D, 0x00917922, 0x00917A14,
0x00917B15,
0x00918619, 0x00918714, 0x00918A18, 0x00918B18, 0x00918B618, 0x00918B722, 0x00918BA18,
0x00918B18,
0x00947518, 0x00947614, 0x0094771D, 0x0094791F, 0x00947A19, 0x00947B14, 0x00948520,
0x00948619,
0x00948714, 0x00948A18, 0x00948B18, 0x00948B61F, 0x00948B718, 0x00948BA17, 0x00948BB1F,
0x00948C617,
0x0094C717, 0x00957520, 0x00957622, 0x00957714, 0x00957924, 0x00957A18, 0x00957B18,
0x00958524,
0x00958618, 0x00958718, 0x0095892B, 0x00958A17, 0x00958B1F, 0x00958B52B, 0x00958B617,
0x00958B71F,
0x00958B92B, 0x00958BA17, 0x00958BB17, 0x00958C52C, 0x00958C617, 0x00958C717, 0x00958C92C,
0x00958CA2B,
0x00958CB2C, 0x00A0751F, 0x00A07614, 0x00A07715, 0x00A0791F, 0x00A07A19, 0x00A07B14,
0x00A0851F,
0x00A08622, 0x00A08714, 0x00A08917, 0x00A08A18, 0x00A08B18, 0x00A08B52B, 0x00A08B61F,
0x00A08B718,
0x00A08BA17, 0x00A08BB1F, 0x00A08C617, 0x00A08C717, 0x00A17524, 0x00A17622, 0x00A17714,
0x00A17924,
0x00A17A18, 0x00A17B18, 0x00A1861F, 0x00A18718, 0x00A18A17, 0x00A18B17, 0x00A18B617,
0x00A18B71F,
0x00A18BA17, 0x00A18BB17, 0x00A47524, 0x00A47618, 0x00A47714, 0x00A4792B, 0x00A47A18,
0x00A47B18,
0x00A4852B, 0x00A4861F, 0x00A48718, 0x00A4892B, 0x00A48A17, 0x00A48B17, 0x00A48B52C,
0x00A48B617,
0x00A48B717, 0x00A48B92C, 0x00A48BA17, 0x00A48BB17, 0x00A48C52E, 0x00A48C62B, 0x00A48C72B,
0x00A48C92E,
0x00A48CA2C, 0x00A48CB2C, 0x00A5752C, 0x00A57617, 0x00A57718, 0x00A5792B, 0x00A57A17,
0x00A57B1F,
0x00A5852C, 0x00A58617, 0x00A5871F, 0x00A5892C, 0x00A58A17, 0x00A58B17, 0x00A58B52E,
0x00A58B62B,
0x00A58B717, 0x00A58B92E, 0x00A58BA2C, 0x00A58BB2C, 0x00A58C52E, 0x00A58C62C, 0x00A58C72C,
0x00A58C92E,
0x00A58CA2C, 0x00A58CB2C, 0x00BA9D2D, 0x00BA9E2E, 0x00BA9F2E, 0x00CDD938, 0x00CE2D38,
0x00CEB938,
0x00CF2D38, 0x00D02D38, 0x00D0313A, 0x00D0713A, 0x0110762F, 0x0110B632, 0x0110B731,
0x03D0113B,
0x03D0213C, 0x03D02D3D, 0x03D0313D, 0x03D0413C, 0x03D04D3D, 0x03D0513C, 0x03D05D3D,
0x03D06139,
0x03D0693D, 0x03D06D3D, 0x03D0713D
};

```

FIG 17(2)

FIG	FIG 17(1)
17	FIG 17(2)


```

static unsigned long ari_gs_hash[225] = {
0x00000401, 0x0001491A, 0x0001590B, 0x00017621, 0x0001891C, 0x0009492A, 0x000DFA38,
0x000F6D3D,
0x0010003B, 0x0010B21B, 0x0011321C, 0x00116B29, 0x0011B31D, 0x00126B1E, 0x00136623,
0x00146729,
0x00146F3B, 0x0015321F, 0x00156E27, 0x00163320, 0x00182725, 0x00186727, 0x00196323,
0x001C4721,
0x001E3F30, 0x001E433B, 0x00203F2A, 0x0020463B, 0x0020F322, 0x00216A2E, 0x00226723,
0x00245625,
0x00256724, 0x00286625, 0x002D3726, 0x002D573A, 0x00316627, 0x00326628, 0x00344729,
0x00366628,
0x003D4329, 0x00416A2A, 0x0042533A, 0x00916A2A, 0x00926B2B, 0x0093E72E, 0x00956B2C,
0x009D362D,
0x009D3B39, 0x009E4330, 0x00A2672E, 0x00AD372F, 0x01145630, 0x01146B27, 0x011C8231,
0x01226732,
0x012CC333, 0x01413B34, 0x019CA335, 0x019CB338, 0x01ACB736, 0x01AD823D, 0x01C37F37,
0x02156738,
0x0218AB3B, 0x021C9B35, 0x021E0738, 0x021FB73D, 0x0220E335, 0x02216B3C, 0x02217234,
0x0222B33C,
0x02239B3B, 0x0223B23A, 0x0224673B, 0x0238A739, 0x0240B23D, 0x024CBF38, 0x024CC23D,
0x024D8738,
0x0297AF3A, 0x02986727, 0x0298A33B, 0x0298A738, 0x029CAF3B, 0x029CC33A, 0x02A0AB35,
0x02A3E736,
0x02AC773B, 0x02B0B335, 0x02B3A73B, 0x02C0D73C, 0x02C1E735, 0x03108E3D, 0x03109737,
0x0311D639,
0x03147F3C, 0x0314B236, 0x0317A639, 0x0317D629, 0x0317DB33, 0x03187627, 0x0318AF3B,
0x0318F61A,
0x0319D739, 0x031C953B, 0x031D633C, 0x031FCF39, 0x0320873B, 0x0320963A, 0x03222639,
0x0323833C,
0x03239A27, 0x0323EA2F, 0x03242631, 0x03242B3B, 0x03249727, 0x0325AB39, 0x0327A73C,
0x0327C728,
0x03287727, 0x03287E3A, 0x03288737, 0x032BAA39, 0x032C7527, 0x032D2337, 0x032E9B39,
0x032EA23B,
0x032EBF3C, 0x032F7E39, 0x0330C63C, 0x0332B23B, 0x0332F230, 0x03339F3B, 0x0333EE27,
0x03348F30,
0x0336AB3C, 0x0338A73B, 0x033A7639, 0x033A7F1A, 0x033C793B, 0x033C9A34, 0x033CA33B,
0x033CA738,
0x033D0A3C, 0x033DB339, 0x033DFF3C, 0x033E9739, 0x0340CB3C, 0x0344573B, 0x0344AA3C,
0x0348263B,
0x034C7B3C, 0x034CBB3A, 0x034CD33C, 0x0390B73D, 0x0390E937, 0x0393653D, 0x0394B73B,
0x0394E33D,
0x0394FA38, 0x03950A3C, 0x0396CF3D, 0x03971A36, 0x0398673C, 0x0398E13B, 0x03994E39,
0x039C733B,
0x039D191A, 0x039D4536, 0x039E053C, 0x039E6E3D, 0x039E9D34, 0x039F8D39, 0x03A0C93B,
0x03A67939,
0x03A69D29, 0x03A6D637, 0x03A85A3C, 0x03AE5B3B, 0x03AEDB3D, 0x03AF2E3C, 0x03B0A13B,
0x03B2B139,
0x03B3123B, 0x03B36339, 0x03B3AD3C, 0x03B42E33, 0x03B4733B, 0x03B4F53C, 0x03B51F36,
0x03B59139,
0x03B5CB3C, 0x03B61737, 0x03B93A3C, 0x03B98F39, 0x03B9F53C, 0x03BA063B, 0x03BA2A3C,
0x03BB2739,
0x03BD3B3B, 0x03BDC939, 0x03BDF534, 0x03BF9A39, 0x03C1653B, 0x03C19E2A, 0x03C20527,
0x03C3633B,
0x03C3823C, 0x03C3A527, 0x03C45A3B, 0x03C4993C, 0x03C5B23B, 0x03C5D527, 0x03C9563B,
0x03C9A93C,
0x03CA063B, 0x03CB0E3C, 0x03CCB53B, 0x03CD1E3C, 0x03CED23D, 0x03CEDF3C, 0x03CFFA39,
0x40BC673E,
0xFFFFFFFFF
};

```

FIG. 18

```

1910→ unsigned short ari_cf_m[64][9] = {
1912→ {65535,65534,65532,65215, 321, 4, 2, 1, 0}, ←pki=0
    {65490,65339,64638,58133, 7463, 973, 270, 125, 0}, ←pki=1
    {65530,65509,65319,60216, 5308, 222, 30, 9, 0},
    {65534,65528,65470,62535, 3012, 67, 8, 2, 0},
    {65533,65524,65435,62110, 3434, 104, 14, 5, 0},
    {65535,65533,65499,62363, 3173, 37, 3, 1, 0},
    {65535,65534,65522,63164, 2371, 14, 2, 1, 0},
    {65535,65530,65448,59939, 5612, 88, 7, 2, 0},
    {65535,65533,65500,61498, 4044, 38, 3, 1, 0},
    {65535,65530,65444,59855, 5667, 92, 6, 1, 0},
    {65535,65532,65495,61386, 4140, 39, 3, 1, 0},
    {65522,65458,64905,55424,10056, 634, 88, 28, 0},
    {65532,65511,65238,57072, 8457, 297, 27, 6, 0},
    {65534,65522,65364,59096, 6461, 171, 15, 3, 0},
    {65535,65530,65426,59204, 6342, 109, 8, 2, 0},
    {65535,65533,65492,61008, 4512, 43, 3, 1, 0},
    {65535,65529,65417,58998, 6519, 118, 6, 1, 0},
    {65535,65533,65490,60856, 4679, 46, 4, 1, 0},
    {65535,65528,65384,58400, 7127, 149, 9, 1, 0},
    {65535,65532,65483,60544, 4984, 56, 4, 1, 0},
    {65517,65413,64537,53269,12264, 1002, 138, 38, 0},
    {65531,65503,65125,55553, 9985, 420, 37, 7, 0},
    {65534,65518,65303,57889, 7650, 235, 20, 3, 0},
    {65490,65288,63679,49500,15949, 1903, 301, 94, 0},
    {65522,65428,64429,51580,13957, 1113, 114, 22, 0},
    {65526,65447,64600,52808,12743, 937, 93, 17, 0},
    {63814,60228,53108,40709,26294,15412, 8961, 5729, 0},
    {65526,65486,65133,57227, 8244, 400, 58, 20, 0},
    {65500,65346,64297,52845,12477, 1283, 230, 70, 0},
    {65528,65486,65077,56652, 8871, 465, 56, 16, 0},
    {65464,65186,63581,50731,14351, 1992, 396, 128, 0},
    {65489,65278,63861,51225,14185, 1726, 302, 96, 0},
    {65485,65249,63632,50425,14933, 1943, 332, 96, 0},
    {65292,64495,61270,47805,17600, 4502, 1337, 542, 0},
    {65519,65421,64478,52517,12971, 1068, 129, 33, 0},
    {65470,65181,63344,49862,15299, 2233, 418, 132, 0},
    {65472,65197,63407,49933,15445, 2176, 396, 123, 0},

```

FIG 19(1)

FIG	FIG 19(1)
19	FIG 19(2)

```

{65376,64781,62057,48496,16676, 3614, 923, 340, 0},
{65259,64356,60836,47316,18158, 4979, 1517, 623, 0},
{64883,63190,58260,45006,21034, 8378, 3559, 1909, 0},
{65261,64180,60126,46710,18694, 5578, 1582, 531, 0},
{64933,63355,58991,46299,19470, 7245, 2989, 1449, 0},
{63999,61383,56309,44712,24964,14237, 9489, 7028, 0},
{65451,65091,62953,48747,16324, 2626, 522, 168, 0},
{65400,64870,62109,47037,18198, 3526, 794, 278, 0},
{65200,64074,59673,44322,20692, 6133, 1836, 739, 0},
{65376,64798,61822,46437,18673, 3881, 932, 368, 0},
{65151,63887,59083,43617,21491, 6768, 2081, 841, 0},
{64592,62314,56211,42184,24450,11142, 5265, 3075, 0},
{64908,62840,56205,41474,23652, 9844, 3388, 1379, 0},
{65021,63308,57341,42286,22972, 8709, 2895, 1232, 0},
{64790,62474,55461,40843,24327,10719, 3921, 1677, 0},
{64053,60476,52429,39583,26962,15208, 7592, 4166, 0},
{63317,58934,51305,40469,29263,19682,12661, 8553, 0},
{63871,59872,52031,39473,26093,15132, 7866, 4080, 0},
{63226,58553,50425,39191,28586,18779,11388, 7035, 0},
{62219,57006,49569,40492,32376,24784,18716,14447, 0},
{62905,58273,50651,39619,28123,18379,11633, 7478, 0},
{63420,59073,51922,41516,29863,20328,13529, 9237, 0},
{63582,59263,51165,37880,24026,13893, 7771, 4535, 0},
{63223,58418,49833,37279,25503,15421, 9122, 5802, 0},
{62322,56878,48746,39095,30723,22195,15849,11887, 0},
{61826,47222,47123,47015,46913,46806,13713, 6895, 0},
1964 → {60678,44085,44084,44083,44082,44081,16715, 9222, 0} ← pki=63
};

```

FIG 19(2)

FIG	FIG 19(1)
19	FIG 19(2)

```
static unsigned long ari_s hash[387] = {
0x0090D52E, 0x0090CB1F, 0x00A4CB2C, 0x00003330, 0x00107A07, 0x00907A15, 0x00207A0C,
0x00147A0C,
0x00247A15, 0x00A07A19, 0x00947A19, 0x00A47A18, 0x0010B70D, 0x0090B719, 0x0020B71D,
0x0014B71D,
0x00A0B718, 0x0094B718, 0x0024B715, 0x00A4B717, 0x0110B731, 0x0000FE1C, 0x0090FE1E,
0x00013B02,
0x00A5C92E, 0x0095C92C, 0x003A9E0B, 0x00000B01, 0x00BA9E2E, 0x0090992B, 0x0011B514,
0x00A5B52E,
0x0095B52B, 0x0090D62C, 0x0010850D, 0x0014851B, 0x00248514, 0x0020850B, 0x00908522,
0x00948520,
0x00A4852B, 0x00A0851F, 0x00003432, 0x00107B08, 0x00207B0E, 0x00907B1D, 0x00147B0E,
0x00247B0C,
0x00A07B14, 0x00947B14, 0x00A47B18, 0x00910928, 0x03D0713D, 0x00D0713A, 0x0000FF1C,
0x0090F52D,
0x0010F521, 0x00013C01, 0x03D05D3D, 0x00A5CA2C, 0x0025CA1F, 0x0095CA2B, 0x003A9F0B,
0x00000C02,
0x0021790B, 0x0025791C, 0x00917922, 0x0015790B, 0x00A17924, 0x00A5792B, 0x00957924,
0x00BA9F2E,
0x00CF2D38, 0x00000200, 0x0011B615, 0x0091B618, 0x0021B619, 0x0015B619, 0x00A1B617,
0x0095B617,
0x0025B618, 0x00A5B62B, 0x004FB125, 0x00108609, 0x00148616, 0x0020860C, 0x00908614,
0x0024861D,
0x00948619, 0x00A08622, 0x00A4861F, 0x0090CD23, 0x00003532, 0x00010A2E, 0x00002B25,
0x0010B90B,
0x0090B91F, 0x00A4B92C, 0x0001001E, 0x03D0213C, 0x0090F62D, 0x0010F628, 0x00A5CB2C,
0x0025CB1F,
0x0095CB2C, 0x00000D03, 0x00117A12, 0x00217A1D, 0x00917A14, 0x00257A0B, 0x00157A1D,
0x00A17A18,
0x00A57A17, 0x00957A18, 0x0011B70C, 0x0091B722, 0x0021B715, 0x0015B714, 0x00A1B71F,
0x0025B722,
0x0095B71F, 0x00A5B717, 0x004FB21C, 0x0010870A, 0x00148707, 0x00208710, 0x0090871D,
0x00248716,
0x00948714, 0x00A08714, 0x00A48718, 0x00907D1C, 0x00010B25, 0x00002C22, 0x0010BA1D,
0x0090BA22,
0x0014BA14, 0x00A0BA17, 0x0094BA17, 0x00A4BA17, 0x03D0693D, 0x0090F72F, 0x0010F728,
0x0025851C,
0x0015850B, 0x00218514, 0x00A5852C, 0x00958524, 0x00117B07, 0x00217B12, 0x00257B1B,
0x00917B15,
0x00157B16, 0x00A17B18, 0x00A57B1F, 0x00957B18, 0x0090D92C, 0x004FB31C, 0x03D0413C,
0x00907E1C,
0x0090C52B, 0x00A4C52E, 0x00002D14, 0x00D02D38, 0x03D02D3D, 0x0010BB16, 0x0090BB19,
0x0014BB15,
0x00A0BB1F, 0x0094BB1F, 0x00A4BB17, 0x0025860B, 0x0015861D, 0x00218615, 0x00918619,
0x00A58617,
0x00958618, 0x00A1861F, 0x00000F25, 0x00CEE938, 0x00004C32, 0x0011B914, 0x00A5B92E,
0x0095B92B,
0x0014890B, 0x0024891C, 0x00908918, 0x00A4892B, 0x00A08917, 0x00907F14, 0x0010C615,
0x0090C61F,
0x0014C614, 0x0094C617, 0x00A4C62B, 0x00A0C617, 0x00910D25, 0x00107504, 0x00907514,
0x00147516,
0x0020751D, 0x00247514, 0x00A0751F, 0x00947518, 0x00A47524, 0x0090F925, 0x0011870E,
0x0025871B,
0x0015870C, 0x0021870C, 0x00918714, 0x00A5871F, 0x00A18718, 0x00958718, 0x03D06139,
0x0000101C,
0x03D04D3D, 0x0011BA15, 0x0091BA18, 0x0021BA22, 0x00A1BA17, 0x0025BA1F, 0x00A5BA2C,
0x0095BA17,
0x00148A1D, 0x00208A1D, 0x00248A15, 0x00908A19, 0x00948A18, 0x00A08A18, 0x00A48A17,
0x0010393D,
```

FIG 20(1)

FIG	FIG 20(1)
20	FIG 20(2)

```

0x0010C70C, 0x0090C718, 0x0014C715, 0x0094C717, 0x00A0C717, 0x00A4C72B, 0x00010E25,
0x00002F2D,
0x00107605, 0x00907615, 0x00147610, 0x00207612, 0x0024761D, 0x00A07614, 0x00947614,
0x00A47618,
0x0110762F, 0x0090BD1C, 0x00CDD938, 0x0000FA02, 0x0090FA2E, 0x0000110B, 0x03D0113B,
0x00A5C52E,
0x0095C52C, 0x0011BB1D, 0x0091BB18, 0x0021BB19, 0x0014950B, 0x00A1BB17, 0x0025BB18,
0x00A5BB2C,
0x0095BB17, 0x00909523, 0x00108B0E, 0x00148B16, 0x00208B0C, 0x00248B1D, 0x00908B14,
0x00948B18,
0x00A08B18, 0x00A48B17, 0x00010F25, 0x0000302B, 0x00107706, 0x00907716, 0x00147711,
0x00207713,
0x0024770E, 0x00A07715, 0x0094771D, 0x00A47714, 0x0091052F, 0x00110528, 0x0090BE1C,
0x0015052D,
0x03D06D3D, 0x0000FB01, 0x0090FB2D, 0x0025890B, 0x00A5892C, 0x00158914, 0x0021891C,
0x0095892B,
0x0011C619, 0x0025C61F, 0x00A5C62C, 0x0095C617, 0x00117516, 0x0021750B, 0x0015750B,
0x00917519,
0x0025751C, 0x00A17524, 0x00957520, 0x00A5752C, 0x0014961D, 0x0090961F, 0x0090C917,
0x00A4C92E,
0x0000312B, 0x00D0313A, 0x03D0313D, 0x0090BF1C, 0x0091062F, 0x0010B50B, 0x0090B517,
0x00A0B52B,
0x00A4B52C, 0x0000FC1C, 0x00258A1D, 0x00A58A17, 0x00158A15, 0x00218A15, 0x00918A18,
0x00A18A17,
0x00958A17, 0x00013938, 0x00001327, 0x0011C715, 0x0025C718, 0x00A5C72C, 0x0095C717,
0x0011760E,
0x0021760C, 0x00917615, 0x0015760C, 0x0025760B, 0x00A17622, 0x00957622, 0x00A57617,
0x00005031,
0x00908D24, 0x0090CA1F, 0x00A4CA2C, 0x0010790D, 0x00907919, 0x0020791D, 0x0014791D,
0x00247914,
0x00A0791F, 0x0094791F, 0x00A4792B, 0x00CE2D38, 0x0010B60C, 0x0090B618, 0x0014B615,
0x0020B615,
0x00A0B61F, 0x0094B61F, 0x0024B619, 0x00A4B617, 0x0110B632, 0x00258B1B, 0x00158B1D,
0x00218B1D,
0x00A58B17, 0x00918B18, 0x00A18B17, 0x00958B1F, 0x0090FD1E, 0x00013A04, 0x0000142F,
0x003A9D1C,
0x00BA9D2D, 0x0011770F, 0x0021770E, 0x00157710, 0x0091771D, 0x0025771B, 0x00A17714,
0x00957714,
0x00A57718, 0x00005131, 0x03D0513C
};

```

FIG 20(2)

FIG	FIG 20(1)
20	FIG 20(2)