(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 536 957**

(51) Int. CI.:

G10L 19/00 (2013.01)
G10L 19/02 (2013.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **11.01.2011 E 11700401 (0)**

(97) Fecha y número de publicación de la concesión europea: **15.04.2015 EP 2517200**

(54) Título: **Codificador de audio, decodificador de audio, método para codificar e información de audio, método para decodificar una información de audio y programa de computación utilizando una modificación de una representación de un numero de un valor de contexto numérico previo**

(30) Prioridad:

12.01.2010 US 294357 P

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
01.06.2015

(73) Titular/es:

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

(72) Inventor/es:

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

(74) Agente/Representante:

PONTI SALES, Adelaida

DESCRIPCIÓN

Codificador de audio, decodificador de audio, método para codificar e información de audio, método para decodificar una información de audio y programa de computación utilizando una modificación de una representación de un numero de un valor de contexto numérico previo

Campo Técnico

[0001] Las realizaciones de acuerdo con la invención se relacionan con un decodificador de audio para producir información de audio decodificada sobre la base de una información de audio codificada, un codificador de audio para producir una información de audio codificada sobre la base de una información de audio de entrada, un método para producir una información de audio decodificada sobre la base de una información de audio codificada, un método para producir una información de audio codificada sobre la base de una información de audio de entrada y un programa de computación.

[0002] Las realizaciones de acuerdo con la invención se relacionan con una codificación espectral sin ruido mejorada, que se puede utilizar en un codificador o decodificador de audio o similar, un denominado codificador unificado de voz y audio (USAC).

Antecedentes de la invención

[0003] A continuación se explican brevemente los antecedentes de la invención a fin de facilitar la comprensión de la invención y las ventajas de la misma. Durante la última década, se invirtieron grandes esfuerzos en la creación de la posibilidad de guardar digitalmente y distribuir contenidos de audio con buena eficiencia de velocidad de transmisión de bits. Un logro importante en este sentido es la definición de la Norma Internacional ISO/IEC 14496–3. La parte 3 de esta Norma se relaciona con una codificación y decodificación de contenidos de audio, y la subparte 4 de la parte se relaciona con la codificación general de audio. ISO/IEC 14496 parte 3, subparte 4, define un concepto para la codificación y decodificación del contenido general de audio. Además, se han propuesto otras mejoras para mejorar la calidad y/o reducir la velocidad de transmisión de bits necesaria.

[0004] De acuerdo con el concepto descripto en dicha Norma, se convierte una señal de audio en el dominio del tiempo a una representación en el dominio de la frecuencia. La transformación del dominio del tiempo al dominio de la frecuencia se realiza por lo general empleando bloques de transformadas, que también se denominan “cuadros” de muestras en el dominio del tiempo. Se ha encontrado que es ventajoso utilizar cuadros traslapados, que están desplazados, por ejemplo, en medio cuadro, puesto que el traslapo permite evitar con eficiencia (o por lo menos reducir) las aberraciones. Además, se ha encontrado que se debería realizar una generación de ventanas a fin de evitar las aberraciones que se originan en este procesamiento de cuadros limitados temporalmente.

[0005] Al transformar una porción en ventana de la señal de audio de entrada del dominio del tiempo al dominio de la frecuencia, se obtiene una compactación de la energía en muchos casos, de tal manera que parte de los valores espectrales comprenden una magnitud significativamente mayor que una pluralidad de otros valores espectrales. En consecuencia, en muchos casos hay un número relativamente pequeño de valores espectrales que tienen una magnitud significativamente superior a una magnitud promedio de los valores espectrales. Un ejemplo típico de transformación del dominio del tiempo al dominio del tiempo–frecuencia que da lugar a una compactación de la energía es la llamada transformada de coseno discreta modificada (MDCT).

[0006] Los valores espectrales con frecuencia son escalados y cuantizados de acuerdo con un modelo psicoacústico, razón por la cual los errores de cuantización son comparativamente menores en lo que respecta a los valores espectrales más importantes y son comparativamente mayores en el caso de los valores espectrales menos importantes. Los valores espectrales escalados y cuantizados son codificados a fin de obtener una representación de los mismos más eficiente con respecto a la velocidad de transmisión de bits.

[0007] Por ejemplo, el uso de la denominada codificación de Huffman de coeficientes espectrales cuantizados ha sido descripto en la Norma Internacional ISO/IEC 14496–3:2005(E), parte 3, subparte 4.

[0008] Sin embargo, se ha encontrado que la calidad de la codificación de los valores espectrales tiene un impacto significativo sobre la velocidad de transmisión de bits requerida. Asimismo, se ha encontrado que la complejidad de un decodificador de audio, que con frecuencia está implementado en un dispositivo portátil del consumidor, y que, por lo tanto debería ser económico y de bajo consumo de energía, depende de la codificación utilizada para codificar los valores espectrales.

[0009] En vista de esta situación, existe la necesidad de un concepto de codificación y decodificación un contenido de audio, que da lugar a una compensación mejorada entre eficiencia de velocidad de transmisión de bits y eficiencia de recursos.

Descripción de la invención

[00010] Una realización de acuerdo con la invención, tal como se expone en la reivindicación independiente 1, da origen a un decodificador de audio para producir una información de audio decodificada sobre la base de una información de audio codificada. El decodificador de audio comprende un decodificador aritmético para producir una pluralidad de valores espectrales decodificados sobre la base de una representación aritméticamente codificada de los valores espectrales comprendidos en la información de audio codificada. El decodificador de audio comprende además un convertidor del dominio de la frecuencia al dominio del tiempo para producir una representación de audio en el dominio del tiempo utilizando los valores espectrales decodificados, a fin de obtener la información de audio decodificada. El decodificador aritmético está configurado para seleccionar una regla de mapeo que describe el mapeo de un valor de código de la representación codificada aritméticamente de los valores espectrales sobre un código de símbolo (código de símbolo que describe uno o más de los valores espectrales decodificados o por lo menos una porción de uno o más de los valores espectrales) dependiendo del estado del contexto descripto por un valor numérico actual del contexto. El decodificador aritmético está configurado para determinar el valor numérico del contexto actual dependiendo de un valor numérico de contexto previo y dependiendo de una pluralidad de valores espectrales previamente decodificados. El decodificador aritmético también está configurado para modificar una representación numérica del valor numérico de contexto previo, que describe un estado del contexto para la decodificación de dichos uno o más valores espectrales previamente decodificados, dependiendo de un valor de subregión de contexto describiendo una subregión de un contexto, para obtener una representación numérica de un valor actual del contexto que describe un estado del contexto para la decodificación de dichos uno o más valores espectrales previamente decodificados.

[00011] Esta realización de acuerdo con la invención se basa en el hallazgo de que es muy eficiente desde el punto de vista informático la modificación de una representación numérica de un valor del contexto previo dependiendo de un valor de subregión de contexto, papra obtener una representación numérica de un valor numérico de contexto actual, puesto que se puede evitar la repetición del cómputo total del valor de contexto actual. Por el contrario, se pueden aprovechar las correlaciones entre el valor numérico de contexto previo y el valor numérico del contexto actual para mantener suficientemente bajo el esfuerzo informático. Se ha descubierto que existe una gran variedad de posibilidades para la modificación de la representación numérica del valor numérico del contexto previo, incluyendo una combinación de reescalamiento de la representación numérica del valor de contexto previo, una suma del valor de subregión del contexto o un valor derivado de la misma (como, por ejemplo, una versión desplazada en bits de un valor de subregión del contexto) a una representación numérica del valor numérico del contexto previo o una representación numérica procesada del valor numérico de contexto previo, el reemplazo de una porción de la representación numérica (en lugar de la totalidad de la representación numérica) del valor numérico de contexto previo dependiendo del valor de subregión del contexto, etc. Por consiguiente, el mantenimiento de por lo menos una porción de una representación numérica del valor numérico del contexto previo (posiblemente, en una versión desplazada) permite reducir significativamente el esfuerzo informático del valor numérico de contexto.

[00012] En una realización preferida, el decodificador aritmético está configurado para suministrar la representación numérica del valor numérico del contexto actual de tal manera que porciones de la representación numérica que tienen diferentes ponderaciones numéricas están determinadas por diferentes valores de subregión de contexto. En consecuencia, una actualización iterativa del valor numérico de contexto, para derivar el valor numérico del contexto actual del valor numérico del contexto previo, se puede realizar con poco esfuerzo informático, aunque evitando una pérdida de información.

[00013] En una realización preferida, la representación numérica es una representación de números binarios de un único valor numérico de contexto actual. De preferencia, una primera subserie de bits de la representación de números binarios es determinada por un primer valor de subregión de contexto asociado a uno o más valores espectrales previamente decodificados, y una segunda subserie de bits de la representación de números binarios está determinada por un segundo valor de subregión de contexto asociado a uno o más valores espectrales previamente decodificados, donde los bits de la primera subserie de bits comprenden una ponderación numérica diferente de los bits de la segunda subserie de bits. Se ha descubierto que ese tipo de representación es muy adecuada para la derivación iterativa del valor numérico del contexto actual del valor numérico del contexto previo.

[00014] En una realización preferida, el decodificador aritmético está configurado para modificar una subserie con bits enmascarados de los bits de información de la representación numérica del valor numérico del contexto previo o de una versión desplazada en bits de la representación numérica del valor numérico del contexto previo, dependiendo de un valor de subregión de contexto que no ha sido tenida en cuenta para la derivación del valor numérico del contexto previo, para obtener la representación numérica del valor numérico del contexto actual. Al ejecutar un enmascaramiento por bits de la representación numérica del valor numérico del contexto previo, o mediante el desplazamiento en bits de la representación numérica del valor numérico del contexto previo, se puede lograr que porciones de un contexto que ya no son tan relevantes como antes se eliminen del valor numérico de contexto y, preferentemente, sean reemplazadas por otras porciones del contexto que son más relevantes en el contexto actual. Un enmascaramiento de bits de una subserie de bits de información de la representación numérica del valor numérico del contexto previo permite reemplazar una porción del valor numérico del contexto previo dependiendo de un valor de subregión de contexto lo que, a su vez, permite considerar una porción del contexto que

antes no había sido tenida en cuenta. Más aun, una operación de desplazamiento refleja el hecho de que hay cierto traslapo entre los valores espectrales previamente decodificados empleados para determinar el contexto anterior (es decir, el contexto utilizado para decodificar el tuplo anterior de valores espectrales) y los valores espectrales previamente decodificados utilizados para determinar el contexto actual (es decir, el contexto para la decodificación de los valores espectrales a decodificar ahora). Más aun, las operaciones de desplazamiento también reflejan el hecho de que la relación de frecuencia (por ejemplo, igual en frecuencia, un bit más grande en frecuencia, etc.) de los valores espectrales previamente decodificados con respecto a los valores espectrales decodificados usando el valor numérico del contexto previo es diferente de la relación de frecuencia de los valores espectrales previamente decodificados con respecto a los valores espectrales a decodificar usando el valor numérico del contexto actual.

[00015] En una realización preferida, el decodificador aritmético está configurado para desplazar en bits la representación numérica del valor numérico del contexto previo, de tal manera que las ponderaciones numéricas de las subseries de bits asociados a diferentes valores de subregión de contexto se modifiquen, para obtener la representación numérica del valor numérico del contexto actual. En consecuencia, el desplazamiento de la posición en la frecuencia entre el uno o más valores espectrales decodificados usando el valor numérico del contexto previo y el uno o más valores espectrales a decodificar usando el valor numérico del contexto actual se puede reflejar en el valor numérico de contexto de manera eficiente. Más aun, por lo general se puede ejecutar una operación de desplazamiento con bajo esfuerzo informático utilizando un microprocesador standard.

[00016] En una realización preferida, el decodificador aritmético está configurado para desplazar en bits la representación numérica del valor numérico del contexto previo, de tal manera que una subserie de bits, que están asociados a un valor de subregión de contexto, sea suprimida de la representación numérica, para obtener la representación numérica del valor numérico del contexto actual. En consecuencia, se puede obtener una doble funcionalidad mediante una única operación de desplazamiento, es decir la consideración de un cambio de la posición en la frecuencia y la consideración del hecho de que algunos valores espectrales (representados por un valor de subregión de contexto) que se han utilizado para obtener el valor numérico del contexto previo, ya no son necesarios para obtener el valor numérico del contexto actual.

[00017] En una realización preferida, el decodificador aritmético está configurado para modificar una primera subserie de bits de una representación de números binarios de un valor numérico de contexto previo o de una versión desplazada en bits de una representación de números binarios de un valor numérico de contexto previo, dependiendo de un valor de subregión de contexto, y para dejar inalteradas ciertas subseries de bits de la representación de números binarios del valor numérico del contexto previo o de una versión desplazada en bits de la representación de números binarios del valor numérico del contexto previo, para derivar la representación de números binarios del valor numérico del contexto actual de la representación de números binarios del valor numérico del contexto previo mediante la modificación selectiva de una o más subseries de bits asociadas a subregiones del contexto tenidas en cuenta para la decodificación de los valores espectrales previamente decodificados (decodificados usando el valor numérico del contexto previo) y no tenidas en cuenta para la decodificación de valores espectrales a decodificar usando el valor numérico del contexto actual. Este concepto ha demostrado ser particularmente eficiente.

[00018] En una realización preferida, el decodificador aritmético está configurado para suministrar la representación numérica del valor numérico del contexto actual de tal manera que una subserie de bits menos significativos de la representación numérica del valor numérico del contexto actual describa un valor de subregión de contexto, valor de subregión de contexto que se utiliza para una decodificación de valores espectrales respecto de la cual un estado del contexto está definido por el valor numérico del contexto actual, y el valor de subregión de contexto que no es utilizado para una decodificación de valores espectrales respecto de la cual un estado del contexto está definido por un valor numérico subsiguiente de contexto (por ej. un valor numérico de contexto derivado del valor numérico del contexto actual). Esta estrategia permite derivar el valor numérico del contexto actual del valor numérico del contexto previo (y derivar el valor numérico del contexto subsiguiente del valor numérico del contexto actual) usando una operación de desplazamiento, ya que los bits menos significativos de la representación numérica pueden ser fácilmente desplazados. Más aun, también se ha encontrado que es apropiado asignar una pequeña ponderación numérica a esos valores de subregión de contexto que son relevantes para el valor numérico del contexto previo, aunque ya no son relevantes para el valor numérico del contexto actual (o, de manera equivalente, cuáles son relevantes para el valor numérico del contexto actual, aunque ya no son relevantes para el valor numérico del contexto subsiguiente), puesto que esto da lugar a un mapeo eficiente del valor numérico de contexto (actual) sobre un valor índice de regla de mapeo.

[00019] En una realización preferida, el decodificador aritmético está configurado para evaluar por lo menos una tabla a fin de determinar si el valor numérico del contexto actual es idéntico a un valor de contexto de la tabla (por ejemplo, un valor de estado significativo) descripto por un elemento de la tabla o se halla dentro de un intervalo descripto por elementos de la tabla, y para derivar un valor índice de regla de mapeo que describe una regla de mapeo seleccionada dependiendo del resultado de una evaluación de dicha por lo menos una tabla. Se ha descubierto que un valor numérico de contexto (actual), que se construye y actualiza de la manera antes descripta, es muy adecuado para ese mapeo sobre un valor índice de regla de mapeo.

[00020] En una realización preferida, el decodificador aritmético está configurado para verificar si la suma de una pluralidad de valores de subregión de contexto es menor o igual a un valor umbral predeterminado de la suma, y para modificar selectivamente el valor numérico del contexto actual dependiendo de un resultado de la verificación. Se ha descubierto que dicha modificación selectiva adicional del valor numérico del contexto actual es muy adecuada para introducir eficientemente información significativa de contexto en el valor numérico del contexto actual sin ningún conflicto con respecto al concepto de actualización del valor numérico de contexto.

[00021] En una realización preferida, el decodificador aritmético está configurado para verificar si la suma de una pluralidad de valores de subregión de contexto, valores de subregión de contexto que están asociados a la misma porción temporal del contenido de audio que el uno o más valores espectrales a decodificar usando un estado del contexto definido por el valor numérico del contexto actual, y valores de subregión de contexto que están asociados frecuencias más bajas que el uno o más valores espectrales a decodificar usando el estado del contexto definido por el valor numérico del contexto actual, es menor o igual que un valor umbral predeterminado de la suma, y para modificar selectivamente el valor numérico del contexto actual dependiendo de un resultado de la verificación. Se ha descubierto que dicha verificación para identificar la presencia de una región de valores espectrales comparativamente bajos proporciona una valiosa información adicional.

[00022] En una realización preferida, el decodificador aritmético está configurado para sumar los valores absolutos de una primera pluralidad de valores espectrales previamente decodificados para obtener un primer valor de subregión de contexto asociado a la primera pluralidad de valores espectrales previamente decodificados, y para sumar los valores absolutos de una segunda pluralidad de valores espectrales previamente decodificados para obtener un segundo valor de subregión de contexto asociado a la segunda pluralidad de valores espectrales previamente decodificados. En consecuencia, se pueden obtener diferentes valores de subregión de contexto.

[00023] En una realización preferida, el decodificador automático está configurado para limitar los valores de subregión de contexto, de tal manera que los valores de subregión de contexto se puedan representar empleando una verdadera subserie de bits de información de la representación numérica del valor numérico del contexto previo. Se ha descubierto que una limitación de los valores de subregión de contexto no ejerce efecto adverso alguno de consideración en el contenido de información de los valores de subregión de contexto. Sin embargo, esta limitación trae aparejada la ventaja de que se puede mantener razonablemente bajo el número de bits necesario para representar el valor de subregión de contexto, lo que influye positivamente sobre la demanda de memoria. Además, la limitación de los valores de subregiones de contexto facilita la actualización iterativa del valor numérico de contexto.

[00024] Otra realización de acuerdo con la invención, tal como se expone en la reivindicación 16, genera un codificador de audio para la provisión de una información de audio codificada sobre la base de información de audio de entrada.

[00025] El codificador de audio se basa en los mismos hallazgos que el decodificador de audio. Además, el codificador de audio puede ser complementado por las funcionalidades indicadas con respecto al decodificador de audio.

[00026] Otra realización de acuerdo con la invención, tal como se expone en la reivindicación 17, genera un método para proporcionar una información de audio decodificada sobre la base de una información de audio codificada.

[00027] Otra realización de acuerdo con la invención, tal como se expone en la reivindicación 18, genera un método para proporcionar una información de audio codificada sobre la base de una información de audio de entrada.

[00028] Otra realización de acuerdo con la invención, tal como se expone en la reivindicación 19, se relaciona con un programa de computación para ejecutar uno de dichos métodos.

Breve Descripción de las Figuras

[00029] Seguidamente se describen las realizaciones de acuerdo con la presente invención tomando como referencia las figuras adjuntas, en las cuales:

La Fig 1 ilustra un diagrama esquemático de bloques de un codificador de audio, de acuerdo con una realización de la invención;

La Fig 2 ilustra un diagrama esquemático de bloques de un decodificador de audio, de acuerdo con una realización de la invención:

La Fig 3 ilustra una pseudo–representación de código de programa de un algoritmo “values_decode()” para decodificar valores espectrales;

La Fig 4 ilustra una representación esquemática de un contexto para un estado de cálculo;

La Fig 5a ilustra una pseudo–representación de código de programa de un algoritmo “arith_map_context()” para mapear un contexto;

La Fig 5b ilustra una pseudo–representación de código de programa de otro algoritmo “arith_map_context()” para mapear un contexto;

La Fig 5c ilustra una pseudo–representación de código de programa de un algoritmo “arith_get_context()” para obtener un valor de estado de contexto;

La Fig 5d ilustra una pseudo–representación de código de programa de otro algoritmo “arith_get_context()” para obtener un valor de estado de contexto;

La Fig 5e ilustra una pseudo–representación de código de programa de un algoritmo “arith_get_pk()” para derivar un valor índice de tabla de frecuencias cumulativas “pki” a partir de un valor de estado (o una variable de estado);

La Fig 5filustra una pseudo–representación de código de programa de otro algoritmo “arith_get_pk()”para derivar un valor índice de tabla de frecuencias cumulativas “pki” a partir de un valor de estado (o una variable de estado);

La Fig 5g ilustra una pseudo–representación de código de programa de un algoritmo “arith_decode()” para la decodificación aritmética de un símbolo a partir de una palabra código de longitud variable;

La Fig 5h ilustra una primera parte de una pseudo–representación de código de programa de otro algoritmo “arith_decode()”para la decodificación aritmética de un símbolo a partir de una palabra código de longitud variable;

La Fig 5iilustra una segunda parte de una pseudo–representación de código de programa de otro algoritmo “arith_decode()"para la decodificación aritmética de un símbolo a partir de una palabra código de longitud variable;

La Fig 5jilustra una pseudo–representación de código de programa de un algoritmo para derivar valores absolutos a,b de valores espectrales a partir de un valor común m;

La Fig 5k ilustra una pseudo–representación de código de programa de un algoritmo para ingresar los valores decodificados a,b en una matriz de valores espectrales decodificados;

La Fig 5lilustra una pseudo–representación de código de programa de un algoritmo “arith_update_context()” para obtener un valor de subregión de contexto sobre la base de los valores absolutos a,b de los valores espectrales decodificados;

La Fig 5m ilustra una pseudo–representación de código de programa de un algoritmo “arith_finish()” para cargar elementos de una matriz de valores espectrales decodificados y una matriz de valores de subregión de contexto;

La Fig 5n ilustra una pseudo–representación de código de programa de otro algoritmo para derivar los valores absolutos a,b de valores espectrales decodificados a partir de un valor común m;

La Fig 5o ilustra una pseudo–representación de código de programa de un algoritmo “arith_update_context()” para actualizar una matriz de valores espectrales decodificados y una matriz de valores de subregión de contexto;

La Fig 5p ilustra una pseudo–representación de código de programa de un algoritmo “arith_save_context()”para cargar elementos de una matriz de valores espectrales decodificados y elementos de una matriz de valores de subregión de contexto;

La Fig 5q ilustra una leyenda de definiciones;

La Fig 5r ilustra otra leyenda de definiciones;

La Fig 6a ilustra una representación de la sintaxis de un bloque de datos bruto de una codificación unificada de voz y audio (USAC);

La Fig 6b ilustra una representación de la sintaxis de un elemento de canal único;

La Fig 6c ilustra una representación de la sintaxis de un elemento de par de anales;

La Fig 6d ilustra una representación de la sintaxis de una información de control “ICS”;

La Fig 6e ilustra una representación de la sintaxis de un flujo de canales en el dominio de la frecuencia;

La Fig 6filustra una representación de la sintaxis de datos espectrales aritméticamente codificados;

La Fig 6g ilustra una representación de la sintaxis para decodificar una serie de valores espectrales;

La Fig 6h ilustra otra representación de la sintaxis para decodificar una serie de valores espectrales;

La Fig 6iilustra una leyenda de elementos de datos y variables;

La Fig 6jilustra otra leyenda de elementos de datos y variables;

La Fig 7 ilustra un diagrama esquemático de bloques de un codificador de audio, de acuerdo con el primer aspecto de la invención;

La Fig 8 ilustra un diagrama esquemático de bloques de un decodificador de audio, de acuerdo con el primer aspecto de la invención;

La Fig 9 ilustra una representación gráfica del mapeo de un valor numérico de contexto actual sobre un valor índice de regla de mapeo, de acuerdo con el primer aspecto de la invención;

La Fig 10 ilustra un diagrama esquemático de bloques de un codificador de audio, de acuerdo con un segundo aspecto de la invención;

La Fig 11 ilustra un diagrama esquemático de bloques de un decodificador de audio, de acuerdo con el segundo aspecto de la invención;

La Fig 12 ilustra un diagrama esquemático de bloques de un codificador de audio, de acuerdo con un tercer aspecto de la invención;

La Fig 13 ilustra un diagrama esquemático de bloques de un decodificador de audio, de acuerdo con el tercer aspecto de la invención;

La Fig 14a ilustra una representación esquemática de un contexto para un cálculo de de estado, como se utiliza de acuerdo con el borrador de trabajo 4 del Proyecto de Norma USAC;

La Fig 14b ilustra una reseña general de las tablas utilizadas en el esquema de codificación aritmética de acuerdo con el borrador de trabajo 4 del Proyecto de Norma USAC;

La Fig 15a ilustra una representación esquemática de un contexto para el cálculo de estado utilizado en las realizaciones de acuerdo con la invención;

La Fig 15b ilustra una reseña general de las tablas utilizadas en el esquema de codificación aritmética de acuerdo con la presente invención;

La Fig 16a ilustra una representación gráfica para el esquema de codificación sin ruido de acuerdo con la presente invención y de acuerdo con el borrador de trabajo 5 del Proyecto de Norma USAC y de acuerdo con la Codificación de Huffman AAC (codificación avanzada de audio);

La Fig 16b ilustra una representación gráfica de una demanda de datos de memoria de sólo lectura total del decodificador USAC de conformidad con la presente invención y de conformidad con el concepto de acuerdo con el borrador de trabajo 5 del Proyecto de Norma USAC;

La Fig 17 ilustra una representación esquemática de una disposición para una comparación de una codificación sin ruido de acuerdo con el borrador de trabajo 3 o el borrador de trabajo 5 del Proyecto de Norma USAC con un esquema de codificación de acuerdo con la presente invención;

La Fig 18 ilustra la representación de una tabla de las velocidades de transmisión de bits promedio producidas por un codificador aritmético USAC de acuerdo con el borrador de trabajo 3 del Proyecto de Norma USAC y de acuerdo con una realización de la presente invención;

La Fig 19 ilustra la representación de una tabla de los niveles mínimo y máximo de reserva de bits correspondientes a un decodificador aritmético de acuerdo con el borrador de trabajo 3 del Proyecto de Norma USAC y de acuerdo con una realización de la presente invención;

La Fig 20 ilustra la representación de una tabla de los números de complejidad promedio correspondientes a la decodificación de un flujo de bits de 32 kbits de acuerdo con el borrador de trabajo 3 del Proyecto de Norma USAC correspondiente a diferentes versiones del codificador aritmético;

Las Figs 21(1) y 21(2) ilustran la representación de una tabla que corresponde al contenido de una tabla "ari_lookup_m[600]";

Las Figs 22(1) a 22(4) ilustran la representación de una tabla que corresponde al contenido de una tabla "ari_hash_m[600]";

Las La Figs 23(1) a 23(7) ilustran la representación de una tabla que corresponde al contenido de una tabla "ari_cf_m[96][17]"; y

La Fig 24 ilustran la representación de una tabla que corresponde al contenido de una tabla "ari_cf_r[]".

Descripción Detallada de las Realizaciones

1. Codificador de audio de acuerdo con la Fig 7

[00030] Fig 7 ilustra un diagrama esquemático de bloques de un codificador de audio de acuerdo con una realización de la invención. El codificador de audio 700 está configurado para recibir una información de audio de entrada 710 para producir, sobre la base de la misma, una información de audio codificada 712. El codificador de audio comprende un convertidor compactador de energía del dominio del tiempo al dominio de la frecuencia 720 que está configurado para producir una representación de audio en el dominio de la frecuencia 722 sobre la base de una representación en el dominio del tiempo de la información de audio de entrada 710, de manera tal que la representación de audio en el dominio de la frecuencia 722 comprende una serie de valores espectrales. El codificador de audio 700 comprende asimismo un codificador aritmético 730 configurado para codificar un valor espectral (de la serie de valores espectrales que conforman la representación de audio en el dominio de la frecuencia 722), o una versión previamente procesada del mismo, utilizando una palabra código de longitud variable a fin de obtener la información de audio codificada 712 (que puede comprender, por ejemplo, una pluralidad de palabras código de longitud variable).

[00031] El codificador aritmético 730 está configurado para mapear un valor espectral, o un valor de un plano de bits más significativo de un valor espectral, sobre un valor de código (es decir, sobre una palabra código de longitud variable) dependiendo de un estado del contexto. El codificador aritmético está configurado para seleccionar una regla de mapeo que describe el mapeo de un valor espectral, o de un plano de bits más significativo de un valor espectral, sobre un valor de código, dependiendo de un estado (actual) del contexto. El codificador aritmético está configurado para determinar el estado actual del contexto, o un valor numérico de contexto actual que describe el estado actual del contexto, dependiendo de una pluralidad de valores espectrales previamente codificados (preferentemente, aunque no necesariamente adyacentes). Para este fin, el codificador aritmético está configurado para evaluar una tabla hash, los elementos de la cual definen tanto valores de estado significativos entre los valores numéricos de contexto, como los límites de los intervalos de valores numéricos de contexto, donde un valor índice de regla de mapeo está individualmente asociado a un valor numérico de contexto (actual) que es un valor de estado significativo, y donde un valor índice común de la regla de mapeo está asociado a diferentes valores numéricos de contexto (actuales) que yacen dentro de un intervalo limitado por límites de intervalo (donde los límites de los intervalos están definidos preferentemente por los elementos de la tabla hash).

[00032] Como se puede apreciar, se puede ejecutar el mapeo de un valor espectral (de la representación de audio en el dominio de la frecuencia 722), o de un plano de bits más significativo de un valor espectral, sobre un valor de código (de la información de audio codificada 712), mediante la codificación de un valor espectral 740 utilizando una regla de mapeo 742. Un rastreador de estado 750 puede estar configurado para rastrear el estado del contexto. El rastreador de estado 750 procura una información 754 que describe el estado actual del contexto. La información 754 que describe el estado actual del contexto puede tomar la forma, preferentemente, de un valor numérico de contexto actual. Un selector de reglas de mapeo 760 está configurado para seleccionar una regla de mapeo, por ejemplo, una tabla de frecuencias cumulativas, que describe el mapeo de un valor espectral, o de un plano de bits más significativo de un valor espectral, sobre un valor de código. En consecuencia, el selector de reglas de mapeo 760 otorga la información sobre reglas de mapeo 742 a la codificación de valores espectrales 740. La información sobre reglas de mapeo 742 puede asumir la forma de un valor índice de regla de mapeo o de una tabla de frecuencias cumulativas seleccionada de acuerdo con un valor índice de regla de mapeo. El selector de reglas de mapeo 760 comprende (o por lo menos evalúa) una tabla hash 752, los elementos de la cual definen tanto valores de estado significativos entre los valores numéricos de contexto como los límites y los intervalos de valores numéricos de contexto, donde un valor índice de regla de mapeo está asociado individualmente a un valor numérico de contexto que es un valor de estado significativo, y donde un valor índice común de regla de mapeo está asociado a diferentes valores numéricos de contexto que yacen dentro de un intervalo limitado por límites de intervalo. Se evalúa la tabla hash 762 para seleccionar la regla de mapeo, es decir para obtener la información sobre reglas de mapeo 742.

[00033] Para resumir lo expuesto, el codificador de audio 700 ejecuta una codificación aritmética de una representación de audio en el dominio de la frecuencia provista por el convertidor del dominio del tiempo al dominio de la frecuencia. La codificación aritmética está subordinada al contexto, por lo que se selecciona una regla de mapeo (por ej. una tabla de frecuencias cumulativas) dependiendo de valores espectrales previamente codificados. En consecuencia, se considera que los valores espectrales adyacentes en el tiempo y/o en la frecuencia (o, por lo tanto, dentro de un entorno predeterminado) entre sí y/o al valor espectral codificado en el momento (es decir, los valores espectrales dentro de un entorno predeterminado del valor espectral que se está codificando en el momento) en la codificación aritmética para ajustar la distribución de probabilidades evaluada por la codificación aritmética. Al seleccionar una regla de mapeo apropiada, se evalúan los valores numéricos actuales del contexto 754 provistos por un rastreador de estado 750. Como por lo general el número de reglas de mapeo diferentes es significativamente menor que el número de valores posibles de valores numéricos actuales de contexto 754, el selector de reglas de mapeo 760 asigna las mismas reglas de mapeo (descriptas, por ejemplo, por un valor índice de regla de mapeo) a un número relativamente grande de diferentes valores numéricos de contexto. Sin embargo, hay por lo general configuraciones espectrales específicas (representadas por valores numéricos específicos de contexto) con las cuales debe estar asociada una regla de mapeo específica a fin de obtener una eficiencia de codificación satisfactoria.

[00034] Se ha encontrado que se puede llevar a cabo la selección de una regla de mapeo dependiendo de un valor numérico de contexto actual con una eficiencia informática particularmente elevada si los elementos de una única tabla hash definen tanto valores de estado significativos como los límites de los intervalos de valores numéricos de contexto (actuales). Se ha encontrado que este mecanismo se adapta bien a los requerimientos de la selección de reglas de mapeo, puesto que hay muchos casos en que un único valor de estado significativo (o un valor numérico de contexto significativo) está inserto entre un intervalo de la izquierda de una pluralidad de valores de estado no significativos (con los cuales se asocia una regla de mapeo común) y el intervalo de la derecha de una pluralidad de valores de estado no significativos (con los cuales se asocia una regla de mapeo común). Además, el mecanismo de uso de una única tabla hash, los elementos de la cual definen tanto valores de estado significativos como los límites de los intervalos de valores numéricos de contexto (actuales) puede tratar con eficiencia diferentes casos, en los cuales, por ejemplo, hay dos intervalos adyacentes de valores de estado no significativos (también denominados valores numéricos de contexto no significativos) sin que haya un valor de estado significativo entre los mismos. Se obtiene una eficiencia informática especialmente alta debido a que se mantiene bajo un número de accesos de la tabla. Por ejemplo, una única búsqueda iterativa de tablas es suficiente, en la mayoría de las realizaciones, para averiguar si el valor numérico de contexto actual es igual a cualquiera de los valores de estado significativos, o en cuál de los intervalos de valores de estado no significativos se halla el valor numérico de contexto actual. Por consiguiente, se puede mantener bajo el número de accesos a las tablas que consumen tanto tiempo como energía. Por lo tanto, se puede considerar que el selector de reglas de mapeo 760, que utiliza la tabla hash 762, es un selector de mapeo particularmente eficiente en términos de complejidad informática, y que de todas maneras permite obtener una alta eficiencia de codificación (en términos de velocidad de transmisión de bits).

[00035] A continuación se describen detalles con respecto a la derivación de la información sobre reglas de mapeo 742 a partir del valor numérico de contexto actual 754.

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

[00036] La Fig. 8 ilustra un diagrama esquemático de bloques de un decodificador de audio 800. El decodificador de audio 800 está configurado para recibir una información de audio codificada 810 y para obtener, sobre la base de la misma, una información de audio decodificada 812. El decodificador de audio 800 comprende un decodificador aritmético 820 que está configurado para suministrar una pluralidad de valores espectrales 822 sobre la base de una representación codificada aritméticamente 821 de los valores espectrales. El decodificador de audio 800 comprende además un convertidor del dominio de la frecuencia al dominio del tiempo 830 que está configurado para recibir los valores espectrales decodificados 822 y para producir la representación de audio en el dominio del tiempo 812, que puede constituir la información de audio decodificada, empleando los valores espectrales decodificados 822, a fin de obtener una información de audio decodificada 812.

[00037] El decodificador aritmético 820 comprende un determinador de valores espectrales 824, que está configurado para mapear un valor de código de la representación codificada aritméticamente 821 de los valores espectrales sobre un código de símbolo que representa uno o más de los valores espectrales decodificados o por lo menos una porción (por ejemplo, un plano de bits más significativo) de uno o más de los valores espectrales decodificados. El determinador de valores espectrales 824 puede estar configurado para ejecutar un mapeo dependiendo de una regla de mapeo, que puede estar descripta por una información sobre reglas de mapeo 828a. La información sobre reglas de mapeo 828a puede asumir la forma, por ejemplo, de un valor índice de regla de mapeo, o de una tabla seleccionada de frecuencias cumulativas (seleccionada, por ejemplo, dependiendo de un valor índice de regla de mapeo).

[00038] El decodificador aritmético 820 está configurado para seleccionar una regla de mapeo (por ej. una tabla de frecuencias cumulativas) que describe el mapeo de valores código (descriptos por la representación codificada

aritméticamente 821 de los valores espectrales) sobre un código de símbolo (que describe uno o más valores espectrales, o un plano de bits más significativo del mismo) dependiendo de un estado del contexto (que puede estar definido por la información de estado del contexto 826a). El decodificador aritmético 820 está configurado para determinar el estado actual del contexto (descripto por el valor numérico de contexto actual) dependiendo de una pluralidad de valores espectrales previamente decodificados. Para este fin se puede emplear un rastreador de estado 826, que recibe una información que describe los valores espectrales previamente decodificados y que provee, sobre la base de la misma, un valor numérico de contexto actual 826a que describe el estado actual del contexto.

[00039] El decodificador aritmético también esta configurado para evaluar una tabla hash 829, los elementos de la cual definen tanto valores de estado significativos entre los valores numéricos de contexto como los límites de los intervalos de valores numéricos de contexto, para seleccionar la regla de mapeo, donde un valor índice de regla de mapeo está individualmente asociado a un valor de contexto significativo que es un valor de estado significativo, y donde un valor índice común de regla de mapeo está asociado a diferentes valores numéricos de contexto que se hallan dentro de un intervalo limitado por los límites de intervalo. La evaluación de la tabla hash 829 se puede realizar, por ejemplo, utilizando un analizador de tablas hash que puede ser parte del selector de reglas de mapeo 828. En consecuencia, se obtiene una información sobre reglas de mapeo 828a, por ejemplo, en forma de un valor índice de regla de mapeo sobre la base del valor numérico de contexto actual 826a que describe el estado actual del contexto. El selector de reglas de mapeo 828 puede determinar, por ejemplo, el valor índice de regla de mapeo 828a dependiendo de un resultado de la evaluación de la tabla hash 829. Por otro lado, la evaluación de la tabla hash 829 puede proporcionar directamente el valor índice de regla de mapeo.

[00040] Con respecto a la funcionalidad del decodificador de señales de audio 800, se debe tener en cuenta que el decodificador aritmético 820 está configurado para seleccionar una regla de mapeo (por ej. una tabla de frecuencias cumulativas) que, término medio, está adaptado favorablemente a los valores espectrales a decodificar, ya que la regla de mapeo se selecciona dependiendo del estado actual del contexto (descripto, por ejemplo, por el valor numérico de contexto actual), que a su vez se determina dependiendo de una pluralidad de valores espectrales previamente decodificados. En consecuencia, se pueden aprovecharlas dependencias estadísticas entre valores espectrales adyacentes a decodificar. Más aun, el decodificador aritmético 820 puede ser implementado con eficiencia con una buena compensación entre complejidad informática, tamaño de las tablas y eficiencia de codificación, utilizando el selector de reglas de mapeo 828. Mediante la evaluación de una (única) tabla hash 829, los elementos de la cual describen tanto valores de estado significativos como límites de intervalo de los intervalos de valores de estado no significativos, una única búsqueda iterativa de tablas puede ser suficiente para derivar la información sobre reglas de mapeo 828a del valor numérico de contexto actual 826a. En consecuencia, es posible mapear un número comparativamente grande de diferentes valores numéricos de contexto (actuales) posibles, sobre un número comparativamente más pequeño de valores índice de reglas de mapeo diferentes. Al usar la tabla hash 829, como se describiera anteriormente, es posible aprovechar el hallazgo de que, en muchos casos, hay un solo valor de estado significativo (valor de contexto significativo) entre el intervalo de la izquierda de valores de estado no significativos (valores de contexto no significativos) y un intervalo de la derecha de valores de estado no significativos (valores de contexto no significativos), donde un valor índice de regla de mapeo diferente está asociado al valor de estado significativo (valor de contexto significativo), en comparación con los valores de estado (valores de contexto) del intervalo de la izquierda y los valores de estado (valores de contexto) del intervalo de la derecha. Sin embargo, el uso de la tabla hash 829 también es muy adecuado para situaciones en las cuales dos intervalos de valores numéricos de estado están inmediatamente adyacentes, sin un valor de estado significativo interpuesto entre los mismos.

[00041] En conclusión, el selector de reglas de mapeo 828, que evalúa la tabla hash 829, aporta una eficiencia particularmente favorable al seleccionar una regla de mapeo (o cuando se genera un valor índice de regla de mapeo) dependiendo del estado actual del contexto (o dependiendo del valor numérico de contexto actual que describe el estado actual del contexto), puesto que el mecanismo de hasheo se adapta bien a las situaciones de contexto típicas de un decodificador de audio.

[00042] A continuación se describen más detalles.

3. Mecanismo de Hasheo de Valores de Contexto de acuerdo con la Fig. 9

[00043] A continuación se describe un mecanismo de hasheo de contextos, que puede ser implementado en el selector de reglas de mapeo 760 y/o en el selector de reglas de mapeo 828. Se puede utilizar la tabla hash 762 y/o la tabla hash 829 para implementar dicho mecanismo de hasheo de valores de contexto.

[00044] Tomando ahora como referencia la Fig. 9 que ilustra una situación de hashing de valor numérico de contexto actual, se pasa a describir otros detalles. En la representación gráfica de la Fig. 9, una abscisa 910 describe valores del valor numérico de contexto actual (es decir, los valores numéricos de contexto). Una ordenada 912 describe valores índice de regla de mapeo. Las marcas 914 describen valores índice de reglas de mapeo correspondientes a los valores numéricos de contexto no significativos (que describen estados no significativos). Las marcas 916 describen valores índice de reglas de mapeo correspondientes a los valores numéricos de contexto

"individuales" significativos (verdaderos) que describen estados significativos (verdaderos). Las marcas 916 describen valores índice de reglas de mapeo correspondientes a valores numéricos de contexto "incorrectos" que describen estados significativos "incorrectos", donde un estado significativo "incorrecto" es un estado significativo con el cual está asociado el mismo valor índice de regla de mapeo que el de uno de los intervalos adyacentes de valores numéricos de contexto no significativos.

[00045] Como se puede apreciar, un elemento de la tabla hash "ari_hash_m[i1]" describe un estado significativo individual (verdadero) que tiene un valor de contexto de c1. Como se puede apreciar, el valor índice de regla de mapeo mriv1 está asociado al estado significativo individual (verdadero) que tiene el valor numérico de contexto c1. En consecuencia, tanto el valor numérico de contexto c1 como el valor índice de regla de mapeo mriv1 pueden ser descriptos por el elemento de la tabla hash "ari_hash_m[i1]". Un intervalo 932 de valores numéricos de contexto está limitado por el valor numérico de contexto c1, donde el valor numérico de contexto c1 no pertenece al intervalo 932, por lo que el valor numérico de contexto más elevado del intervalo 932 es igual a c1 – 1. Un valor índice de regla de mapeo de mriv4 (que es diferente de mriv1) está asociado a los valores numéricos de contexto del intervalo 932. El valor índice de regla de mapeo mriv4 puede estar definido, por ejemplo, por el elemento de la tabla "ari_lookup_m[i1–1]" de una tabla adicional "ari_lookup_m".

[00046] Más aun, un valor índice de regla de mapeo mriv2 puede estar asociado a valores numéricos de contexto que se hallan dentro de un intervalo 934. Un límite inferior del intervalo 934 está determinado por el valor numérico de contexto actual c1, que es un valor numérico de contexto significativo, donde el valor numérico de contexto actual c1 no pertenece al intervalo 932. En consecuencia, el menor valor del intervalo 934 es igual a c1 + 1 (con la presunción de valores numéricos enteros de contexto). Otro límite del intervalo 934 está determinado por el valor numérico de contexto actual c2, donde el valor numérico de contexto actual c2 no pertenece al intervalo 934, por lo que el valor más grande del intervalo 934 es igual a c2 – 1. El valor numérico de contexto actual c2 es un valor numérico de contexto denominado "incorrecto", que está descripto en el elemento de la tabla hash "ari_hash_m[i2]". Por ejemplo, el valor índice de regla de mapeo mriv2 puede estar asociado al valor numérico de contexto actual c2, de manera tal que el valor numérico de contexto actual asociado al valor numérico de contexto significativo "incorrecto" c2 es igual al valor índice de regla de mapeo asociado al intervalo 934 limitado por el valor numérico de contexto actual c2. Por añadidura, un intervalo 936 de valores numéricos de contexto también está limitado por el valor numérico de contexto actual c2, donde el valor numérico de contexto actual c2 no pertenece al intervalo 936, por lo que el valor numérico de contexto más bajo del intervalo 936 es igual a c2 + 1. Un valor índice de regla de mapeo mriv3, que por lo general es diferente del valor índice de regla de mapeo mriv2, está asociado a los valores numéricos de contexto del intervalo 936.

[00047] Como se puede apreciar, el valor índice de regla de mapeo mriv4, que está asociado al intervalo 932 de valores numéricos de contexto, puede estar definido por un elemento "ari_lookup_m[i1–1]" de una tabla "ari_lookup_m", El índice de la regla de mapeo mriv2, que está asociado a los valores numéricos de contexto del intervalo 934, puede estar definido por un elemento de la tabla "ari_lookup_m[i1]" de la tabla "ari_lookup_m" y el valor índice de regla de mapeo mriv3 puede estar definido por un elemento de la tabla "ari_lookup_m[i2]" de la tabla "ari_lookup_m". En el ejemplo aquí presentado, el valor índice de la tabla hash i2 puede ser 1 mayor que el valor índice de la tabla hash i1.

[00048] Como se puede apreciar de la Fig. 9, el selector de reglas de mapeo 760 o el selector de reglas de mapeo 828 puede recibir un valor numérico de contexto actual 764, 826a y decidir, mediante la evaluación de los elementos de la tabla "ari_hash_m", si el valor numérico de contexto actual es un valor de estado significativo (independientemente de si es un valor de estado significativo "individual" o un valor de estado significativo "incorrecto") o si el valor numérico de contexto actual se halla dentro de uno de los intervalos 932, 934, 936, que están limitados por los valores de estado significativos ("individuales" o "incorrectos") c1, c2. Tanto la verificación de si el valor numérico de contexto actual es igual a un valor de estado significativo c1, c2 como la evaluación de en cuál de los intervalos 932, 934, 936 yace el valor numérico de contexto actual (en caso de que el valor numérico de contexto actual no sea igual a un valor de estado significativo) se pueden ejecutar utilizando una única búsqueda común de la tabla hash.

[00049] Más aun, se puede utilizar la evaluación de la tabla hash "ari_hash_m" para obtener un valor índice de la tabla hash (por ejemplo, i1–1, i1 o i2). Por consiguiente, el selector de reglas de mapeo 760, 828 puede estar configurado para obtener, mediante la evaluación de una única tabla hash 762, 829 (por ejemplo, la tabla hash "ari_hash_m"), un valor índice de la tabla hash (por ejemplo, i1–1, i1 o i2) que designa un valor de estado significativo (por ej., c1 o c2) y/o un intervalo (por ej., 932,934,936) y una información con respecto a si el valor numérico de contexto actual es un valor de contexto significativo (también denominado valor de estado significativo) o no.

[00050] Más aun, si se encuentra, en la evaluación de la tabla hash 762, 829, "ari_hash_m", que el valor numérico de contexto actual no es un valor de contexto "significativo" (o valor de estado "significativo"), se puede utilizar el valor índice de la tabla hash (por ejemplo, i1–1, i1 o i2) obtenido de la evaluación de la tabla hash ("ari_hash_m") para obtener un valor índice de regla de mapeo asociado a un intervalo 932, 934, 936 de valores numéricos de contexto. Por ejemplo, se puede utilizar el valor índice de la tabla hash (por ej., i1–1, i1 o i2) para designar un

elemento de una tabla de mapeo adicional (por ejemplo, “ari_lookup_m”), que describe los valores índice de regla de mapeo asociados al intervalo 932, 934, 936 dentro del cual se halla el valor numérico de contexto actual.

[00051] Por más detallas, se hace referencia a la siguiente explicación detallada del algoritmo “arith_get_pk” (donde hay diferentes opciones correspondientes a este algoritmo “arith_get_pk()”, ejemplos del cual están expuestos en las Figs. 5e y 5f).

[00052] Más aun, se debe tener en cuenta que el tamaño de los intervalos puede diferir de un caso al otro. En algunos casos, un intervalo de valores numéricos de contexto comprende un único valor numérico de contexto. Sin embargo, en muchos casos, un intervalo puede comprender una pluralidad de valores numéricos de contexto.

4. Codificador de audio De acuerdo con la Fig. 10

[00053] La Fig. 10 ilustra un diagrama esquemático de bloques de un codificador de audio 1000 de acuerdo con una realización de la invención. El codificador de audio 1000 de acuerdo con la Fig. 10 es similar al codificador de audio 700 de acuerdo con la Fig. 7, por lo que las señales y medios idénticos se designan con números de referencia idénticos en la Figs. 7 y 10.

[00054] El codificador de audio 1000 está configurado para recibir una información de audio de entrada 710 y para obtener, sobre la base de la misma, una información de audio codificada 712. El codificador de audio 1000 comprende un convertidor compactador de energía del dominio del tiempo al dominio de la frecuencia 720, que está configurado para producir una representación en el dominio de la frecuencia 722 sobre la base de una representación en el dominio del tiempo de la información de audio de entrada 710, de tal manera que la representación de audio en el dominio de la frecuencia 722 comprenda una serie de valores espectrales. El codificador de audio 1000 comprende asimismo un codificador aritmético 1030 configurado para codificar un valor espectral (de la serie de valores espectrales que conforman la representación de audio en el dominio de la frecuencia 722), o una versión previamente procesada del mismo, utilizando una palabra código de longitud variable para obtener la información de audio codificada 712 (que puede comprender, por ejemplo, una pluralidad de palabras código de longitud variable).

[00055] El codificador aritmético 1030 está configurado para mapear un valor espectral, o una pluralidad de valores espectrales, o un valor de un plano de bits más significativo de un valor espectral o de una pluralidad de valores espectrales, sobre un valor de código (es decir, sobre una palabra código de longitud variable) dependiendo de un estado del contexto. El codificador aritmético 1030 está configurado para seleccionar una regla de mapeo que describe el mapeo de un valor espectral, o de una pluralidad de valores espectrales, o de un plano de bits más significativo de un valor espectral o de una pluralidad de valores espectrales, sobre un valor de código dependiendo de un estado del contexto. El codificador aritmético está configurado para determinar el estado actual del contexto dependiendo de una pluralidad de valores espectrales previamente codificados (preferentemente, aunque no necesariamente adyacentes). Para este fin, el codificador aritmético está configurado para modificar una representación numérica de un valor numérico de contexto anterior, que describe un estado del contexto asociado a uno o más valores espectrales previamente codificados (por ejemplo, para seleccionar una regla de mapeo correspondiente), dependiendo de un valor de subregión de contexto, para obtener una representación numérica de un valor numérico de contexto actual que describe un estado del contexto asociado a uno o más valores espectrales a codificar (por ejemplo, para seleccionar una regla de mapeo correspondiente).

[00056] Como se puede apreciar, el mapeo de un valor espectral, o de una pluralidad de valores espectrales, o de un plano de bits más significativo de un valor espectral o de una pluralidad de valores espectrales, sobre un valor de código puede ser ejecutado mediante la codificación de un valor espectral 740 utilizando una regla de mapeo descripta por una información sobre reglas de mapeo 742. Un rastreador de estado 750 puede estar configurado para rastrear el estado del contexto. El rastreador de estado 750 puede estar configurado para modificar una representación numérica de un valor numérico de contexto anterior, que describe un estado del contexto asociado a una codificación de uno o más valores espectrales previamente codificados, dependiendo de un valor de subregión de contexto, para obtener una representación numérica de un valor numérico de contexto actual que describe un estado del contexto asociado a una codificación de uno o más valores espectrales a codificar. La modificación de la representación numérica del valor numérico de contexto anterior may, por ejemplo, puede ser ejecutada por un modificador de representaciones numéricas 1052, que recibe el valor numérico de contexto anterior y uno o más valores de subregión de contexto y proporciona el valor numérico de contexto actual. En consecuencia, el rastreador de estado 1050 proporciona una información 754 que describe el estado actual del contexto, por ejemplo, en forma de un valor numérico de contexto actual. Un selector de reglas de mapeo 1060 puede seleccionar una regla de mapeo, por ejemplo, una tabla de frecuencias cumulativas, que describe el mapeo de un valor espectral, o de una pluralidad de valores espectrales, o de un plano de bits más significativo de un valor espectral o de una pluralidad de valores espectrales, sobre un valor de código. En consecuencia, el selector de reglas de mapeo 1060 suministra la información sobre reglas de mapeo 742 a la codificación espectral 740.

[00057] Se debe tener en cuenta que, en algunas realizaciones, el rastreador de estado 1050 puede ser idéntico al rastreador de estado 750 o al rastreador de estado 826. También se debe tener presente que el selector de reglas

de mapeo 1060 puede, en algunas realizaciones, ser idéntico al selector de reglas de mapeo 760, o al selector de reglas de mapeo 828.

[00058] Como resumen de lo expuesto, el codificador de audio 1000 ejecuta una codificación aritmética de una representación de audio en el dominio de la frecuencia provista por el convertidor del dominio del tiempo al dominio de la frecuencia. La codificación aritmética está subordinada al contexto, por lo que se selecciona una regla de mapeo (por ej. una tabla de frecuencias cumulativas) dependiendo de valores espectrales previamente codificados. En consecuencia, se consideran los valores espectrales adyacentes en el tiempo y/o la frecuencia (o por lo menos dentro de un entorno predeterminado) entre sí y/o con el valor espectral que cuya codificación está en curso (es decir, los valores espectrales dentro de un entorno predeterminado del valor espectral actualmente codificado) en la codificación aritmética para ajustar la distribución de probabilidades evaluada por la codificación aritmética.

[00059] Al determinar el valor numérico de contexto actual, se modifica una representación numérica de un valor numérico de contexto anterior, que describe un estado del contexto asociado a uno o más valores espectrales previamente codificados, dependiendo de un valor de subregión de contexto, para obtener una representación numérica de un valor numérico de contexto actual que describe un estado del contexto asociado a uno o más valores espectrales a codificar. Esta técnica permite evitar una recomputación completa del valor numérico de contexto actual, recomputación completa que consume una considerable cantidad de recursos en las estrategias convencionales. Existe una gran variedad de posibilidades para la modificación de la representación numérica del valor numérico de contexto anterior, incluyendo una combinación de reescalamiento de una representación numérica del valor numérico de contexto anterior, una suma de un valor de subregión de contexto o un valor derivado del mismo a la representación numérica del valor numérico de contexto anterior o a una representación numérica procesada del valor numérico de contexto anterior, el reemplazo de una porción de la representación numérica (en lugar de la totalidad de la representación numérica) del valor numérico de contexto anterior dependiendo del valor de la subregión de contexto, y así sucesivamente. Por consiguiente, por lo general se obtiene la representación numérica del valor numérico de contexto actual sobre la base de la representación numérica del valor numérico de contexto anterior y también sobre la base de por lo menos un valor de subregión de contexto, donde típicamente se ejecuta una combinación de operaciones para combinar el valor numérico de contexto anterior con un valor de subregión de contexto, como por ejemplo, dos o más operaciones seleccionadas entre una operación de suma, una operación de resta, una operación de multiplicación, una operación de división, una operación Booleana Y, una operación Booleana O, una operación Booleana NAND, , una operación Booleana NOR, una operación Booleana de negación, una operación Booleana de complemento o una operación de desplazamiento. En consecuencia, por lo menos una porción de la representación numérica del valor numérico de contexto anterior se mantiene por lo general sin cambios (excepto por un desplazamiento opcional a una posición diferente) al derivar el valor numérico de contexto actual del valor numérico de contexto anterior. Por el contrario, se modifican otras porciones de la representación numérica del valor numérico de contexto anterior dependiendo de uno o más valores de subregión de contexto. Por consiguiente, se puede obtener el valor numérico de contexto actual con un esfuerzo informático comparativamente pequeño, evitando al mismo tiempo un recómputo completo del valor numérico de contexto actual.

[00060] De esa manera se puede obtener un valor numérico actual significativo, que es muy adecuado para ser utilizado por el selector de reglas de mapeo 1060.

[00061] En consecuencia, se puede obtener una codificación eficiente manteniendo suficientemente sencillo el cálculo de contextos.

5. Decodificador de audio de acuerdo con la Fig. 11

[00062] La Fig. 11 ilustra un diagrama esquemático de bloques de un decodificador de audio 1100. El decodificador de audio 1100 es similar al decodificador de audio 800 de acuerdo con la Fig. 8, por lo que los signos, medios y funcionalidades idénticos están indicados por números de referencia idénticos.

[00063] El decodificador de audio 1100 está configurado para recibir una información de audio codificada 810 y para proporcionar, sobre la base de la misma, una información de audio decodificada 812. El decodificador de audio 1100 comprende un decodificador aritmético 1120 que está configurado para suministrar una pluralidad de valores espectrales decodificados 822 sobre la base de una representación codificada aritméticamente 821 de los valores espectrales. El decodificador de audio 1100 comprende asimismo un convertidor del dominio de la frecuencia al dominio del tiempo 830 que está configurado para recibir los valores espectrales decodificados 822 y para proporcionar la representación de audio en el dominio del tiempo 812, que puede constituir la información de audio decodificada, usando los valores espectrales decodificados 822, a fin de obtener una información de audio decodificada 812.

[00064] El decodificador aritmético 1120 comprende un determinador de valores espectrales 824, que está configurado para mapear un valor de código de la representación codificada aritméticamente 821 de valores espectrales sobre un código de símbolo que representa uno o más de los valores espectrales decodificados o por lo menos una porción (por ejemplo, un plano de bits más significativo) de uno o más de los valores espectrales

decodificados. El determinador de valores espectrales 824 puede estar configurado para ejecutar el mapeo dependiendo de una regla de mapeo, que puede estar definida por una información sobre reglas de mapeo 828a. La información sobre reglas de mapeo 828a puede comprender, por ejemplo, un valor índice de regla de mapeo, o puede comprender una serie de elementos seleccionada de una tabla de frecuencias cumulativas.

[00065] El decodificador aritmético 1120 está configurado para seleccionar una regla de mapeo (por ej., una tabla de frecuencias cumulativas) que describe el mapeo de un valor de código (descripto por la representación codificada aritméticamente 821 de valores espectrales) sobre un código de símbolo (que describe uno o más valores espectrales) dependiendo de un estado del contexto, estado de contexto que puede estar representado por la información de estado del contexto 1126a. La información sobre el estado del contexto 1126a puede asumir la forma de un valor numérico de contexto actual. El decodificador aritmético 1120 está configurado para determinar el estado actual del contexto dependiendo de una pluralidad de valores espectrales previamente decodificados 822. Para este fin, se puede emplear un rastreador de estado 1126, que recibe una información que describe los valores espectrales previamente decodificados. El decodificador aritmético está configurado para modificar una representación numérica del valor numérico de contexto anterior, que describe un estado del contexto asociado a uno o más valores espectrales anteriormente decodificados, dependiendo de un valor de subregión de contexto, para obtener una representación numérica de un valor numérico de contexto actual que describe un estado del contexto asociado a uno o más valores espectrales a decodificar. Una modificación de la representación numérica del valor numérico de contexto anterior puede ser ejecutada, por ejemplo, por un modificador de representaciones numéricas 1127, que es parte del rastreador de estado 1126. En consecuencia, la información del estado actual del contexto 1126a se obtiene, por ejemplo, en forma de un valor numérico de contexto actual. La selección de la regla de mapeo puede ser ejecutada por un selector de reglas de mapeo 1128, que deriva una información sobre reglas de mapeo 828a de la información del estado actual del contexto 1126a, y que suministra la información sobre reglas de mapeo 828a al determinador de valores espectrales 824.

[00066] En lo que respecta a la funcionalidad del decodificador de señales de audio 1100, se debe tener en cuenta que el decodificador aritmético 1120 está configurado para seleccionar una regla de mapeo (por ej., una tabla de frecuencias cumulativas) que se adapta bien, término medio, al valor espectral a decodificar, ya que la regla de mapeo se selecciona dependiendo del estado actual del contexto, que a su vez se determina dependiendo de una pluralidad de valores espectrales previamente decodificados. En consecuencia, se pueden aprovechar las dependencias estadísticas entre valores espectrales adyacentes a decodificar.

[00067] Más aun, mediante la modificación de una representación numérica de un valor numérico de contexto anterior que describe un estado del contexto asociado a la decodificación de uno o más valores espectrales anteriormente decodificados, dependiendo de un valor de subregión de contexto, para obtener una representación numérica de un valor numérico de contexto actual que describe un estado del contexto asociado a la decodificación de uno o más valores espectrales a decodificar, es posible obtener una información importante acerca del estado actual del contexto, que es muy adecuada para mapear con respecto a un valor índice de regla de mapeo, con un esfuerzo de computación comparativamente bajo. Al mantener por lo menos una porción de una representación numérica del valor numérico de contexto anterior (posiblemente en una versión con desplazamiento de bits o escalada) en tanto que se actualiza otra porción de la representación numérica del valor numérico de contexto anterior dependiendo de los valores de subregión de contexto que no han sido tenidos en cuenta en el valor numérico de contexto anterior pero que serían tomados en cuenta en el valor numérico de contexto actual, se puede mantener razonablemente bajo el número de operaciones necesarias para derivar el valor numérico de contexto actual. Además, es posible aprovechar el hecho de que los contextos empleados para decodificar los valores espectrales son típicamente similares o correlacionados. Por ejemplo, el contexto para la decodificación de un primer valor espectral (o de una primera pluralidad de valores espectrales) depende de una primera serie de valores espectrales previamente decodificados. El contexto para la decodificación de un segundo valor espectral (o una segunda serie de valores espectrales), que es adyacente al primer valor espectral (o la primera serie de valores espectrales) puede comprender una segunda serie de valores espectrales previamente decodificados. Como se presume que el primer valor espectral y el segundo valor espectral son adyacentes (por ej., con respecto a las frecuencias asociadas), la primera serie de valores espectrales, que determina el contexto para la codificación del primer valor espectral, puede comprender cierto traslapo con la segunda serie de valores espectrales, que determina el contexto para la decodificación del segundo valor espectral. En consecuencia, se puede comprender fácilmente que el estado del contexto para la decodificación del segundo valor espectral comprende cierta correlación con el estado del contexto para la decodificación del primer valor espectral. La eficiencia informática de la derivación del contexto, es decir de la derivación del valor numérico de contexto actual, se puede lograr aprovechando esas correlaciones. Se ha encontrado que la correlación entre estados del contexto para la decodificación de valores espectrales adyacentes (por ej., entre el estado del contexto descripto por el valor numérico de contexto anterior y el estado del contexto descripto por el valor numérico de contexto actual) puede ser aprovechada con eficiencia modificando sólo las partes del valor numérico de contexto anterior que dependen de valores de subregión de contexto no considerados para la derivación del estado numérico del contexto anterior y la derivación del valor numérico de contexto actual del valor numérico de contexto anterior.

[00068] En conclusión, los conceptos aquí descriptos dan lugar a una eficiencia informática particularmente favorable al derivar el valor numérico de contexto actual.

[00069] A continuación se describen otros detalles.

6. Codificador de audio de acuerdo con la Fig. 12

[00070] La Fig. 12 ilustra un diagrama esquemático de bloques de un codificador de audio, de acuerdo con una realización de la invención. El codificador de audio 1200 de acuerdo con la Fig. 12 es similar al codificador de audio 700 de acuerdo con la Fig. 7, por lo que idénticos medios, signos y funcionalidades están indicados con idénticos números de referencia.

[00071] El codificador de audio 1200 está configurado para recibir una información de audio de entrada 710 y para proporcionar, sobre la base de la misma, una información de audio codificada 712. El codificador de audio 1200 comprende un convertidor compactador de energía del dominio del tiempo al dominio de la frecuencia 720 que está configurado para producir una representación de audio en el dominio de la frecuencia 722 sobre la base de una representación de audio en el dominio del tiempo de la información de audio de entrada 710, por lo que la representación de audio en el dominio de la frecuencia 722 comprende una serie de valores espectrales. El codificador de audio 1200 comprende asimismo un codificador aritmético 1230 configurado para codificar un valor espectral (de la serie de valores espectrales que conforman la representación de audio en el dominio de la frecuencia 722), o una pluralidad de valores espectrales, o una versión previamente procesada del mismo, utilizando una palabra código de longitud variable para obtener la información de audio codificada 712 (que puede comprender, por ejemplo, una pluralidad de palabras código de longitud variable.

[00072] El codificador aritmético 1230 está configurado para mapear un valor espectral, o una pluralidad de valores espectrales, o un valor de un plano de bits más significativo de un valor espectral o de una pluralidad de valores espectrales, sobre un valor de código (es decir, sobre una palabra código de longitud variable), dependiendo de un estado del contexto. El codificador aritmético 1230 está configurado para seleccionar una regla de mapeo que describe el mapeo de un valor espectral, o de una pluralidad de valores espectrales, o de un plano de bits más significativo de un valor espectral o de una pluralidad de valores espectrales, sobre un valor de código, dependiendo del estado del contexto. El codificador aritmético está configurado para determinar el estado actual del contexto dependiendo de una pluralidad de valores espectrales previamente codificados (preferentemente, aunque no necesariamente adyacentes). Para este fin, el codificador aritmético está configurado para obtener una pluralidad de valores de subregión de contexto sobre la base de valores espectrales previamente codificados, para almacenar dichos valores de subregión de contexto, y para derivar un valor numérico de contexto actual asociado a uno o más valores espectrales a codificar dependiendo de los valores de subregión de contexto almacenados. Más aun, el codificador aritmético está configurado para computar la norma de un vector formado por una pluralidad de valores espectrales previamente codificados, a fin de obtener un valor común de subregión de contexto asociado a la pluralidad de valores espectrales previamente codificados.

[00073] Como se puede apreciar, el mapeo de un valor espectral, o de una pluralidad de valores espectrales, o de un plano de bits más significativo de un valor espectral o de una pluralidad de valores espectrales, sobre un valor de código puede ser ejecutado mediante la codificación de un valor espectral 740 utilizando una regla de mapeo descripta por una información sobre reglas de mapeo 742. Un rastreador de estado 1250 puede estar configurado para rastrear el estado del contexto y puede comprender una calculadora de valores de subregión de contexto 1252, para calcular la norma de un vector formado por una pluralidad de valores espectrales previamente codificados, a fin de obtener valores comunes de subregión de contexto asociados a la pluralidad de valores espectrales previamente codificados. El rastreador de estado 1250 también está preferentemente configurado para determinar el estado actual del contexto dependiendo de un resultado de dicho cómputo de un valor de subregión de contexto ejecutado by la calculadora de valores de subregiones de contexto 1252. En consecuencia, el rastreador de estado 1250 proporciona una información 1254, que describe el estado actual del contexto. Un selector de reglas de mapeo 1260 puede seleccionar una regla de mapeo, por ejemplo, una tabla de frecuencias cumulativas, que describe el mapeo de un valor espectral, o de un plano de bits más significativo de un valor espectral, sobre un valor de código. En consecuencia, el selector de reglas de mapeo 1260 suministra la información sobre reglas de mapeo 742 para la codificación espectral 740.

[00074] Como resumen de lo expuesto, el codificador de audio 1200 ejecuta una codificación aritmética de una representación de audio en el dominio de la frecuencia provista por el convertidor del dominio del tiempo al dominio de la frecuencia 720. La codificación aritmética depende del contexto, por lo que se selecciona una regla de mapeo (por ej., una tabla de frecuencias cumulativas) dependiendo de 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 actualmente codificado (es decir, los valores espectrales dentro de un entorno predeterminado del valor espectral actualmente codificado) son tenidos en cuenta en la codificación aritmética para ajustar la distribución de probabilidades evaluada por la codificación aritmética.

[00075] Para proporcionar un valor numérico de contexto actual, se obtiene un valor de subregión de contexto asociado a una pluralidad de valores espectrales previamente codificados sobre la base del cómputo de una norma de un vector formado por una pluralidad de valores espectrales previamente codificados. El resultado de la

determinación del valor numérico de contexto actual se aplica en la selección del estado actual del contexto, es decir, en la selección de una regla de mapeo.

[00076] Mediante el cómputo de la norma de un vector formado por una pluralidad de valores espectrales previamente codificados, se puede obtener una información reveladora que describe una porción del contexto de dichos uno o más valores espectrales a codificar, donde la norma de un vector de valores espectrales previamente codificados puede estar típicamente representada con un número comparativamente pequeño de bits. Por consiguiente, la cantidad de información del contexto que se debe guardar para uso posterior en la derivación de un valor numérico de contexto actual, se puede mantener suficientemente baja mediante la aplicación del enfoque precedentemente descripto al cómputo de los valores de subregión de contexto. Se ha encontrado que la norma de un vector de valores espectrales previamente codificados comprende por lo general la información más significativa con respecto al estado del contexto. Por el contrario, se ha encontrado que el signo de dichos valores espectrales previamente codificados comprende por lo general un impacto subordinado sobre el estado del contexto, por lo que es razonable no tener en cuenta el signo de los valores espectrales anteriormente decodificados para reducir la cantidad de información que se ha de guardar para su uso posterior. Además, se ha encontrado que el cómputo de una norma de un vector de valores espectrales previamente codificados es una estrategia razonable para la derivación de un valor de subregión de contexto, ya que el efecto de promediado, que por lo general se obtiene mediante el cómputo de la norma, deja sustancialmente inalterada la información más importante acerca del estado del contexto. Para resumir, el cómputo del valor de subregión de contexto ejecutado por la calculadora de valores de subregión de contexto 1252 permite presentar una información compacta de subregiones de contexto para su almacenamiento y reutilización posterior, donde se preserva la información más relevante acerca del estado del contexto a pesar de la reducción de la cantidad de información.

[00077] En consecuencia, se puede obtener una codificación eficiente de la información de audio de entrada 710, manteniendo al mismo tiempo suficientemente bajos el esfuerzo informático y la cantidad de datos que ha de almacenar el codificador aritmético 1230.

7. Decodificador de audio de acuerdo con la Fig. 13

[00078] La Fig. 13 ilustra un diagrama esquemático de bloques de un decodificador de audio 1300. Como el decodificador de audio 1300 es similar al decodificador de audio 800 de acuerdo con la Fig. 8, y al decodificador de audio 1100 de acuerdo con la Fig. 11, los medios, señales y funcionalidades idénticos están indicados con números idénticos.

[00079] El decodificador de audio 1300 está configurado para recibir una información de audio codificada 810 y para proporcionar, sobre la base de la misma, una información de audio decodificada 812. El decodificador de audio 1300 comprende un decodificador aritmético 1320 que está configurado para suministrar una pluralidad de valores espectrales decodificados 822 sobre la base de una representación codificada aritméticamente 821 de los valores espectrales. El decodificador de audio 1300 comprende asimismo un convertidor del dominio de la frecuencia al dominio del tiempo 830 que está configurado para recibir los valores espectrales decodificados 822 y para producir la representación de audio en el dominio del tiempo 812, que puede constituir la información de audio decodificada, utilizando los valores espectrales decodificados 822, a fin de obtener una información de audio decodificada 812.

[00080] El decodificador aritmético 1320 comprende un determinador de valores espectrales 824 que está configurado para mapear un valor de código de la representación codificada aritméticamente 821 de valores espectrales sobre un código de símbolo que representa uno o más de los valores espectrales decodificados o por lo menos una porción (por ej. un plano de bits más significativo) de uno o más de los valores espectrales decodificados. El determinador de valores espectrales 824 puede estar configurado para ejecutar un mapeo dependiendo de una regla de mapeo, que está definida por una información sobre reglas de mapeo 828a. La información sobre reglas de mapeo 828a puede comprender, por ejemplo, un valor índice de regla de mapeo, o una serie seleccionada de elementos de una tabla de frecuencias cumulativas.

[00081] El decodificador aritmético 1320 está configurado para seleccionar una regla de mapeo (por ej., una tabla de frecuencias cumulativas) que describe el mapeo de un valor de código (descripto por la representación codificada aritméticamente 821 de valores espectrales) sobre un código de símbolo (que describe uno o más valores espectrales) dependiendo de un estado del contexto (que puede estar definido por la información de estado del contexto 1326a). El decodificador aritmético 1320 está configurado para determinar el estado actual del contexto dependiendo de una pluralidad de valores espectrales previamente decodificados 822. Para este fin, se puede utilizar un rastreador de estado 1326, que recibe una información que describe los valores espectrales previamente decodificados. El decodificador aritmético también está configurado para obtener una pluralidad de valores de subregión de contexto sobre la base de valores espectrales previamente decodificados y para almacenar dichos valores de subregión de contexto. El decodificador aritmético está configurado para derivar un valor numérico de contexto actual asociado a uno o más valores espectrales a decodificar dependiendo de los valores de subregión de contexto almacenados. El decodificador aritmético 1320 está configurado para calcular la norma de un vector formado por una pluralidad de valores espectrales previamente decodificados, a fin de obtener un valor común de subregiones de contexto asociado a la pluralidad de valores espectrales previamente decodificados.

[00082] El cómputo de la norma de un vector formado por una pluralidad de valores espectrales previamente codificados, a fin de obtener un valor común de subregiones de contexto asociado a la pluralidad de valores espectrales anteriormente decodificados, puede ser ejecutado, por ejemplo, por la calculadora de valores de subregiones de contexto 1327, que forma parte del rastreador de estado 1326. En consecuencia, se obtiene una información sobre el estado actual del contexto 1326a sobre la base de los valores de subregión de contexto, donde el rastreador de estado 1326 preferentemente proporciona un valor numérico de contexto actual asociado a uno o más valores espectrales a decodificar dependiendo de los valores de subregión de contexto guardados. La selección de la regla de mapeos puede ser ejecutada por un selector de reglas de mapeo 1328, que deriva una información sobre reglas de mapeo 828a de la información del estado actual del contexto 1326a, y que suministra la información sobre reglas de mapeo 828a al determinador de valores espectrales 824.

[00083] Con respecto a la funcionalidad del decodificador de señales de audio 1300, se debe tener en cuenta que el decodificador aritmético 1320 está configurado para seleccionar una regla de mapeo (por ej., una tabla de frecuencias cumulativas) que, en término medio, está bien adaptada al valor espectral a decodificar, ya que la regla de mapeo se selecciona dependiendo del estado actual del contexto que, a su vez, se determina dependiendo de una pluralidad de valores espectrales previamente decodificados. En consecuencia, se pueden aprovechar las dependencias estadísticas entre los valores espectrales a decodificar.

[00084] Sin embargo, se ha encontrado que es eficiente, en términos de uso de memoria, el almacenamiento de valores de subregión de contexto que se basan en el cómputo de una norma de un vector formado sobre una pluralidad de valores espectrales anteriormente decodificados, para su uso posterior en la determinación del valor numérico del contexto. También se ha descubierto que dichos valores de subregión de contexto de todas maneras comprenden la información de contexto más relevante. En consecuencia, el concepto utilizado por el rastreador de estado 1326 constituye un buen compromiso entre eficiencia de codificación, eficiencia informática y eficiencia de almacenamiento.

[00085] Más adelante se describen más detalles.

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

[00086] A continuación se describe un codificador de audio de acuerdo con una realización de la presente invención. La Fig. 1 ilustra un diagrama esquemático de bloques de ese tipo de codificador de audio 100.

[00087] El codificador de audio 100 está configurado para recibir una información de audio de entrada 110 y para producir, sobre la base de la misma, un flujo de bits 112, que constituye una información de audio codificada. El codificador de audio 100 comprende opcionalmente un preprocesador 120, que está configurado para recibir la información de audio de entrada 110 y para producir, sobre la base de la misma, una información de audio de entrada preprocesada 110a. El codificador de audio 100 comprende asimismo un transformador compactador de energía de señales del dominio del tiempo al dominio de la frecuencia 130, que también se denomina convertidor de señal. El convertidor de señal 130 está configurado para recibir la información de audio de entrada 110, 110a y para producir, sobre la base de la misma, una información de audio en el dominio de la frecuencia 132, que preferentemente asume la forma de una serie de valores espectrales. Por ejemplo, el transformador de señal 130 puede estar configurado para recibir un cuadro de la información de audio de entrada 110, 110a (por ej. un bloque de muestras en el dominio del tiempo) y para producir una serie de valores espectrales que representan el contenido de audio del respectivo cuadro de audio. Además, el transformador de señal 130 puede estar configurado para recibir una pluralidad de cuadros de audio traslapados o no traslapados de la información de audio de entrada 110, 110a y para producir, sobre la base de la misma, una representación de audio en el dominio del tiempo–frecuencia, que comprende una secuencia de series subsiguientes de valores espectrales, una serie de valores espectrales asociada a cada cuadro.

[00088] El transformador de señal compactador de energía del domino del tiempo al dominio de la frecuencia 130 puede comprender un banco de filtros compactador de energía, que proporciona valores espectrales asociados a rangos de frecuencia diferentes, traslapados o no traslapados. Por ejemplo, el transformador de señal 130 puede comprender un transformador MDCT de generación de ventanas 130a, que está configurado para colocar en ventanas la información de audio de entrada 110, 110a (o un cuadro de la misma) utilizando una ventana de transformada y para ejecutar una transformada de coseno discreta modificada de la información de audio de entrada colocada en ventana 110, 110a (o del cuadro de la misma colocado en ventana). En consecuencia, la representación de audio en el dominio de la frecuencia 132 puede comprender, por ejemplo, una serie de 1024 valores espectrales en forma de coeficientes MDCT asociados a un cuadro de la información de audio de entrada.

[00089] El codificador de audio 100 puede comprender además, opcionalmente, un posprocesador espectral 140, que está configurado para recibir la representación de audio en el dominio de la frecuencia 132 y para producir, sobre la base de la misma, una representación de audio en el dominio de la frecuencia post–procesada 142. El posprocesador espectral 140 puede estar configurado, por ejemplo, para ejecutar un modelado de ruido temporal y/o una predicción a largo plazo y/o cualquier otro post–procesamiento espectral conocido en la técnica. El codificador

de audio comprende además, opcionalmente, un escalador/cuantizador 150, que está configurado para recibir la representación de audio en el dominio de la frecuencia 132 o la versión post–procesada de la misma 142 y para producir una representación de audio escalada y cuantizada en el dominio de la frecuencia 152.

[00090] El codificador de audio 100 comprende además, opcionalmente, un procesador de modelos psicoacústicos 160, que está configurado para recibir la información de audio de entrada 110 (o la versión post–procesada 110a de la misma) y para producir, sobre la base de la misma, una información de control opcional, que se puede utilizar para el control del transformador compactador de energía de señal del dominio del tiempo al dominio de la frecuencia 130, para el control del post–procesador espectral opcional 140 y/o para el control del escalador/cuantizador opcional 150. Por ejemplo, el procesador de modelos psicoacústicos 160 puede estar configurado para analizar la información de audio de entrada, a fin de determinar qué componentes de la información de audio de entrada 110, 110a son particularmente importantes para la percepción humana del contenido de audio y qué componentes de la información de audio de entrada 110, 110a son menos importantes para la percepción del contenido de audio. En consecuencia, el procesador de modelos psicoacústicos 160 puede aportar información de control, que es utilizada por el codificador de audio 100 para ajustar el escalamiento de la representación de audio en el dominio de la frecuencia 132, 142 ejecutado por el escalador/cuantizador 150 y/o la resolución de cuantización aplicada por el escalador/cuantizador 150. En consecuencia, se escalan bandas de factores perceptualmente importantes (es decir, grupos de valores espectrales adyacentes que son de particular importancia para la percepción humana del contenido de audio) con un gran factor de escalamiento y se las cuantiza con una resolución comparativamente alta, en tanto que las bandas de factores de escala menos importantes para la percepción (es decir, grupos de valores espectrales adyacentes) se escalan con un factor de escalamiento comparativamente menor y se las cuantiza con una resolución de cuantización comparativamente menor. En consecuencia, los valores espectrales escalados de las frecuencias más importantes para la percepción son típicamente significativamente mayores que los valores espectrales de las frecuencias menos importantes para la percepción.

[00091] El codificador de audio comprende asimismo un codificador aritmético 170, que está configurado para recibir la versión escalada y cuantizada 152 de la representación de audio en el dominio de la frecuencia 132 (o, por otro lado, la versión post–procesada 142 de la representación de audio en el dominio de la frecuencia 132, o incluso la representación de audio en el dominio de la frecuencia 132 en sí) y para producir información aritmética de palabras clave 172a sobre la base de la misma, de manera tal que la información aritmética de palabras clave representa la representación de audio en el dominio de la frecuencia 152.

[00092] El codificador de audio 100 comprende asimismo un formateador de carga útil de flujo de bits 190, que está configurado para recibir la información aritmética de palabras código 172a. El formateador de carga útil de flujo de bits 190 también está configurado, por lo general, para recibir información adicional, como por ejemplo, información de factores de escalamiento que describen qué factores de escalamiento han sido aplicados por el escalador/cuantizador 150. Además, el formateador de carga útil de flujo de bits 190 puede estar configurado para recibir otro tipo de información de control. El formateador de carga útil de flujo de bits 190 está configurado para producir el flujo de bits 112 sobre la base de la información recibida armando el flujo de bits de conformidad con una sintaxis conveniente de flujo de bits, que se describe más adelante.

[00093] A continuación se describen detalles con respecto al codificador aritmético 170. El codificador aritmético 170 está configurado para recibir una pluralidad de valores espectrales post–procesados y escalados y cuantizados de la representación de audio en el dominio de la frecuencia 132. El codificador aritmético comprende un extractor de planos de bits más significativos 174, o incluso de dos valores espectrales, que está configurado para extraer un plano de bits más significativo m de un valor espectral. Cabe señalar en este caso que el plano de bits más significativo puede comprender uno o aún más bits (por 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 ofrece un valor de plano de bits más significativo 176 de un valor espectral.

[00094] Por otro lado, sin embargo, el extractor de planos de bits más significativo 174 puede producir un valor combinado de planos de bits más significativos m combinando el plano de bits más significativos de una pluralidad de valores espectrales (por ej., de los valores espectrales a y b). El plano de bits más significativo del valor espectral a está indicado con m. Por otro lado, el valor combinado de planos de bits más significativos de una pluralidad de valores espectrales a,b se designa con la letra m.

[00095] El codificador aritmético 170 comprende asimismo un primer determinador de palabras códigos 180, que está configurado para determinar una palabra código aritmética acod_m [pki][m] que representa el valor de plano de bits más significativo m. Opcionalmente, el determinador de palabras clave 180 puede producir asimismo una o más palabras clave de escape (en la presente también denominadas “ARITH_ESCAPE”) que indiquen, por ejemplo, cuántos planos de bits menos significativos hay disponibles (y, en consecuencia, que indiquen la ponderación numérica del plano de bits más significativo). El primer determinador de palabras clave 180 puede estar configurado para producir la palabra clave asociada a un valor de plano de bits más significativo m utilizando una tabla de frecuencias cumulativas seleccionada que tiene (o a la que hace referencia) un índice de tablas de frecuencias cumulativas pki.

[00096] Para determinar cuál de las tablas de frecuencias cumulativas se debe seleccionar, el codificador aritmético comprende preferentemente un rastreador de estado 182, que está configurado para rastrear el estado del codificador aritmético, por ejemplo, observando cuáles valores espectrales han sido codificados anteriormente. El rastreador de estado 182 otorga, en consecuencia, una información de estado 184, por ejemplo, un valor de estado designado con la letra “s” o “t” o “c”. El codificador aritmético 170 comprende asimismo un selector de tablas de frecuencias cumulativas 186, que está configurado para recibir la información de estado 184 y para suministrar una información 188 que describe la tabla de frecuencias cumulativas seleccionado al determinador de palabras clave 180. Por ejemplo, el selector de tablas de frecuencias cumulativas 186 puede suministrar un índice de tablas de frecuencias cumulativas pki“ que describe cuáles tablas de frecuencias cumulativas, de la serie de 96 tablas de frecuencias cumulativas, se ha de seleccionar para ser utilizada por el determinador de palabras clave. Por otro lado, el selector de tablas de frecuencias cumulativas 186 puede suministrar la totalidad de la tabla de frecuencias cumulativas seleccionada o una sub–tabla al determinador de palabras clave. Por consiguiente, el determinador de palabras clave 180 puede utilizar la tabla de frecuencias cumulativas o sub–tabla seleccionada para la provisión de la palabra clave acod_m[pki][m] del valor de plano de bits más significativo m, de tal manera que la palabra clave real acod_m[pki][m] que codifica el valor del plano de bits más significativo m está supeditado al valor de m y el índice de tablas de frecuencias cumulativas pki, y en consecuencia a la información de estado actual 184. Más adelante se describen más detalles con respecto al proceso de codificación y el formato de las palabras clave obtenidas.

[00097] Se debe notar, sin embargo, que en algunas realizaciones, el rastreador de estado 182 puede ser idéntico al rastreador de estado 750, el rastreador de estado 1050 o el rastreador de estado 1250, o asumir la funcionalidad del mismo. También se debe tener presente que el selector de tablas de frecuencias cumulativas 186, en algunas realizaciones puede ser idéntico al selector de reglas de mapeo 760, el selector de reglas de mapeo 1060, o el selector de reglas de mapeo 1260 o asumir la funcionalidad del mismo. Más aun, el primer determinador de palabras clave 180, en algunas realizaciones, puede ser idéntico o tomar la funcionalidad de la codificación de valores espectrales 740.

[00098] El codificador aritmético 170 comprende además un extractor de planos de bits menos significativos 189a, que está configurado para extraer uno o más planos de bits menos significativos de la representación de audio escalada y cuantizada en el dominio de la frecuencia 152, si uno o más de los valores espectrales a codificar exceden el rango de valores codificables utilizando sólo el plano de bits más significativo. Los planos de bits menos significativos pueden comprender uno o más bits, según convenga. En consecuencia, el extractor de planos de bits menos significativos 189a produce una información sobre planos de bits menos significativos 189b. El codificador aritmético 170 comprende asimismo un segundo determinador de palabras clave 189c, que está configurado para recibir la información sobre planos de bits menos significativos 189d y para producir, sobre la base de la misma, 0, 1 o más palabras clave “acod_r” que representan el contenido de 0, 1 o más planos de bits menos significativos. El segundo determinador de palabras clave 189c puede estar configurado para aplicar un algoritmo de codificación aritmética o cualquier otro algoritmo de codificación para derivar las palabras clave del plano de bits menos significativos “acod_r” de la información sobre planos de bits menos significativos 189b.

[00099] Aquí se debe tener en cuenta que el número de planos de bits menos significativos puede variar dependiendo del valor de los valores espectrales escalados y cuantizados 152, razón por la cual puede no haber ningún plano de bits menos significativos en absoluto, si el valor espectral escalado y cuantizado a codificar es comparativamente pequeño, por lo que puede haber un plano de bits menos significativos si el valor espectral actual escalado y cuantizado a codificar es de un rango intermedio y por lo que haber más de un plano de bits menos significativos si el valor espectral escalado y cuantizado a codificar es un valor comparativamente alto.

[000100] Como resumen de lo expuesto, el codificador aritmético 170 está configurado para codificar valores espectrales escalados y cuantizados, que están definidos por la información 152, utilizando un proceso de codificación jerárquica. El plano de bits más significativo (que comprende, por ejemplo, uno, dos o tres bits por valor espectral) de uno o más valores espectrales, es codificado para obtener una palabra clave aritmética “acod_m[pki][m]” de un valor de plano de bits más significativo m. Uno o más planos de bits menos significativos (donde cada uno de los planos de bits menos significativos comprende, por ejemplo, uno, dos o tres bits) de dicho uno o más valores espectrales son codificados para obtener una o más palabras clave “acod_r”. Al codificar el plano de bits más significativo, se mapea el valor m del plano de bits más significativo contra una palabra clave acod_m[pki][m]. Para este fin, hay 96 tablas de frecuencias cumulativas disponibles para la codificación del valor m dependiendo de un estado del codificador aritmético 170, es decir, dependiendo de valores espectrales previamente codificados. En consecuencia, se obtiene palabra clave “acod_m[pki][m]”. Además, se aporta e incluye una o más palabras clave “acod_r” en el flujo de bits si hay uno o más planos de bits menos significativos presentes.

Descripción de reinicio

[000101] El codificador de audio 100 puede estar opcionalmente configurado para decidir si se puede obtener una mejora de la velocidad de transmisión de bits mediante reinicio del contexto, por ejemplo fijando el índice de estado en un valor por defecto. En consecuencia, el codificador de audio 100 puede estar configurado para proporcionar una información sobre reinicio (por ej. denominada “arith_reset_flag”) que indica si se ha reiniciado el contexto para

la codificación aritmética, y que indica además si se debe reiniciar el contexto para la decodificación aritmética en un decodificador correspondiente.

[000102] Más adelante se describen detalles con respecto al formato del flujo de bits y las tablas de frecuencias cumulativas aplicadas.

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

[000103] A continuación se describe un decodificador de audio de acuerdo con una realización de la invención. La Fig. 2 ilustra un diagrama esquemático de bloques de dicho decodificador de audio 200.

[000104] El decodificador de audio 200 está configurado para recibir un flujo de bits 210, que representa una información de audio codificada y que puede ser idéntico al flujo de bits 112 provisto por el codificador de audio 100. El decodificador de audio 200 proporciona una información de audio decodificada 212 sobre la base del flujo de bits 210.

[000105] El decodificador de audio 200 comprende un deformateador de carga útil de flujo de bits opcional 220, que está configurado para recibir el flujo de bits 210 y para extraer del flujo de bits 210 una representación de audio codificada en el dominio de la frecuencia 222. Por ejemplo, el deformateador de carga útil de flujo de bits 220 puede estar configurado para extraer del flujo de bits 210 datos espectrales aritméticamente codificados, como por ejemplo, una palabra clave aritmética “acod_m [pki][m]” que representa el valor del plano de bits más significativo m de un valor espectral a, o de una pluralidad de valores espectrales a, b, y una palabra clave “acod_r” que representa un contenido de planos de bits menos significativos del valor espectral a, o de una pluralidad de valores espectrales a, b, de la representación de audio en el dominio de la frecuencia. Por consiguiente, la representación de audio codificada en el dominio de la frecuencia 222 constituye (o comprende) una representación aritméticamente codificada de valores espectrales. El deformateador de carga útil de flujo de bits 220 está configurado además para extraer del flujo de bits información de control adicional, que no se ilustra en la Fig. 2. Además, el deformateador de carga útil de flujo de bits está configurado opcionalmente para extraer del flujo de bits 210, una información de reinicio de estado 224, que también se denomina bandera de reinicio aritmético o “arith_reset_flag”.

[000106] El decodificador de audio 200 comprende un decodificador aritmético 230, que también se denomina “decodificador espectral sin ruido”. El decodificador aritmético 230 está configurado para recibir la representación de audio codificada en el dominio de la frecuencia 220 y, opcionalmente, la información de reinicio de estado 224. El decodificador aritmético 230 también está configurado para producir una representación de audio decodificada en el dominio de la frecuencia 232, que puede comprender una representación decodificada de valores espectrales. Por ejemplo, la representación decodificada de audio en el dominio de la frecuencia 232 puede comprender una representación decodificada de valores espectrales, que están definidos por la representación de audio codificada en el dominio de la frecuencia 220.

[000107] El decodificador de audio 200 comprende asimismo un cuantizador inverso /reescalador opcional 240, que está configurado para recibir la representación decodificada de audio en el dominio de la frecuencia 232 y para producir, sobre la base de la misma, una representación inversamente cuantizada y reescalada de audio en el dominio de la frecuencia 242.

[000108] El decodificador de audio 200 comprende además un preprocesador espectral opcional 250, que está configurado para recibir la representación inversamente cuantizada y reescalada de audio en el dominio de la frecuencia 242 y para producir, sobre la base de la misma, una versión pre–procesada 252 de la representación inversamente cuantizada y reescalada de audio en el dominio de la frecuencia 242. El decodificador de audio 200 comprende asimismo un transformador de señal del dominio de la frecuencia al dominio del tiempo 260, que también se denomina “convertidor de señal”. El transformador de señal 260 está configurado para recibir la versión pre–procesada 252 de la representación inversamente cuantizada y reescalada de audio en el dominio de la frecuencia 242 (o, por otro lado, la representación inversamente cuantizada y reescalada de audio en el dominio de la frecuencia 242 o la representación decodificada de audio en el dominio de la frecuencia 232) y para producir, sobre la base de la misma, una representación en el dominio del tiempo 262 de la información de audio. El transformador de señal del dominio de la frecuencia al dominio del tiempo 260 puede comprender, por ejemplo, un transformador para ejecutar una transformada inversa de coseno discreta modificada (IMDCT) y funcionalidades apropiadas de generación de ventanas (como así también otras auxiliares tales como, por ejemplo, un traslapo–y–suma.

[000109] El decodificador de audio 200 puede comprender además un post–procesador opcional en el dominio del tiempo 270, que está configurado para recibir la representación en el dominio del tiempo 262 de la información de audio y para obtener la información de audio decodificada 212 usando un post–procesamiento en el dominio del tiempo. Sin embargo, si se omite el post–procesamiento, la representación en el dominio del tiempo 262 puede ser idéntica a la información de audio decodificada 212.

[000110] Cabe señalar aquí que el cuantizador inverso/reescalador 240, el pre–procesador espectral 250, el transformador de señal del dominio de la frecuencia al dominio del tiempo 260 y el post–procesador en el dominio

del tiempo 270 pueden ser controlados dependiendo de la información de control que es extraída del flujo de bits 210 por el deformateador de carga útil de flujo de bits 220.

[000111] Para resumir la funcionalidad general del decodificador de audio 200, se puede obtener una representación decodificada de audio en el dominio de la frecuencia 232, por ejemplo, una serie de valores espectrales asociados a un cuadro de audio de la información de audio codificada, sobre la base de la representación codificada en el dominio de la frecuencia 222 utilizando el decodificador aritmético 230. Seguidamente, se cuantiza inversamente, se reescala y pre–procesa una serie, por ejemplo, de 1024 valores espectrales, que pueden ser coeficientes MDCT. En consecuencia, se obtiene una serie inversamente cuantizada, reescalada y pre–procesada espectralmente de valores espectrales (por ej., 1024 coeficientes MDCT). Posteriormente, se deriva una representación en el dominio del tiempo de un cuadro de audio de la serie inversamente cuantizada, reescalada y pre–procesada espectralmente de valores en el dominio de la frecuencia (por ej. coeficientes MDCT). En consecuencia, se obtiene una representación en el dominio del tiempo de un cuadro de audio. La representación en el dominio del tiempo de un cuadro de audio dado se puede combinar con representaciones en el dominio del tiempo de cuadros de audio anteriores y/o subsiguientes. Por ejemplo, se puede ejecutar un traslapo–y–suma entre las representaciones en el dominio del tiempo de los cuadros de audio posteriores a fin de alisar las transiciones entre las representaciones en el dominio del tiempo de los cuadros de audio adyacentes y a fin de obtener una cancelación del aliasing. Para obtener detalles con respecto a la reconstrucción de la información de audio decodificada 212 sobre la base de la representación de audio decodificada en el dominio del tiempo–frecuencia 232, se hace referencia, por ejemplo, a l Norma Internacional ISO/IEC 14496–3, parte 3, subparte 4, donde se presenta una descripción detallada. Sin embargo, se pueden utilizar otros esquemas más elaborados de traslapo y cancelación del aliasing.

[000112] A continuación se describen algunos detalles con respecto al decodificador aritmético 230. El decodificador aritmético 230 comprende un determinador de planos de bits más significativos 284, que está configurado para recibir la palabra clave aritmética acod_m [pki][m] que describe el valor de plano de bits más significativo m. El determinador de planos de bits más significativos 284 puede estar configurado para usar una tabla de frecuencias cumulativas de una serie que comprende una pluralidad de 96 tablas de frecuencias cumulativas para derivar el valor de plano de bits más significativo m de la palabra clave aritmética “acod_m [pki][m]”.

[000113] El determinador de planos de bits más significativos 284 está configurado para derivar valores 286 de un plano de bits más significativo de uno o más valores espectrales sobre la base de la palabra clave acod_m. El decodificador aritmético 230 comprende además a determinador de planos de bits menos significativos 288, que está configurado para recibir uno o más palabras clave “acod_r” que representa uno o más plano de bits menos significativos de un valor espectral. En consecuencia, el determinador de planos de bits menos significativos 288 está configurado para producir valores decodificados 290 de uno o más planos de bits menos significativos. El decodificador de audio 200 comprende asimismo un combinador de planos de bits 292, que está configurado para recibir los valores decodificados 286 del plano de bits más significativo de uno o más valores espectrales y los valores decodificados 290 de uno o más planos de bits menos significativos de los valores espectrales si se dispone de dicho plano de bits menos significativos para los valores espectrales actuales. En consecuencia, el combinador de planos de bits 292 produce valores espectrales decodificados, que son parte de la representación decodificada de audio en el dominio de la frecuencia 232. Naturalmente, el decodificador aritmético 230 está configurado por lo general para suministrar una pluralidad de valores espectrales a fin de obtener una serie completa de valores espectrales decodificados asociados a un cuadro actual del contenido de audio.

[000114] El decodificador aritmético 230 comprende además un selector de tablas de frecuencias cumulativas 296, que está configurado para seleccionar una de las tablas de frecuencias cumulativas 96 dependiendo de un índice de estado 298 que describe un estado del decodificador aritmético. El decodificador aritmético 230 comprende además un rastreador de estado 299, que está configurado para rastrear un estado del decodificador aritmético dependiendo de los valores espectrales previamente decodificados. La información de estado se puede reiniciar opcionalmente a una información de estado por defecto en respuesta a la información de reinicio de estado 224. En consecuencia, el selector de tablas de frecuencias cumulativas 296 está configurado para proveer un índice (por ej. pki) de una tabla de frecuencias cumulativas seleccionada o una tabla de frecuencias cumulativas seleccionada o una subtabla de la misma, para su aplicación en la decodificación del valor del plano de bits más significativo m dependiendo de la palabra clave “acod_m”.

[000115] Para resumir la funcionalidad del decodificador de audio 200, el decodificador de audio 200 está configurado para recibir una representación de audio codificada con eficiencia de velocidad de transmisión de bits en el dominio de la frecuencia 222 y para obtener una representación de audio decodificada en el dominio de la frecuencia sobre la base de la misma. En el decodificador aritmético 230, que se utiliza para obtener la representación decodificada de audio en el dominio de la frecuencia 232 sobre la base de la representación de audio codificada en el dominio de la frecuencia 222, se aprovecha la probabilidad de diferentes combinaciones de valores del plano de bits más significativo de valores espectrales adyacentes mediante el uso de un decodificador aritmético 280, que está configurado para aplicar una tabla de frecuencias cumulativas. En otras palabras, se aprovechan las dependencias estáticas entre valores espectrales mediante la selección de diferentes tablas de frecuencias cumulativas de una serie que comprende 96 tablas de frecuencias cumulativas diferentes dependiendo de un índice de estados 298, que se obtiene observando los valores espectrales decodificados anteriormente calculados.

[000116] Se debe tener en cuenta que el rastreador de estado 299 puede ser idéntico, o puede tomar la funcionalidad del rastreador de estado 826, el rastreador de estado 1126, o el rastreador de estado 1326. El selector de tablas de frecuencias cumulativas 296 puede ser idéntico, o puede tomar la funcionalidad del selector de reglas de mapeo 828, el selector de reglas de mapeo 1128, o el selector de reglas de mapeo 1328. El determinador de planos de bits más significativos 284 puede ser idéntico o puede tomar la funcionalidad del determinador de valores espectrales 824.

10. Reseña General de la Herramienta de Codificación Espectral sin Ruido

[000117] A continuación se explican detalles con respecto al algoritmo de codificación y decodificación que es ejecutado, por ejemplo, por el codificador aritmético 170 y el decodificador aritmético 230.

[000118] Nos enfocamos ahora en la descripción del algoritmo de decodificación. Se debe tener en cuenta, sin embargo, que se puede ejecutar un algoritmo de codificación correspondiente de acuerdo con los conceptos de algoritmo de decodificación, donde los mapeos entre los valores espectrales codificados y decodificados se invierten, y donde el cómputo del valor índice de regla de mapeo es sustancialmente idéntico. En un codificador, los valores espectrales codificados toman el lugar de los valores espectrales decodificados. Además, los valores espectrales a codificar toman el lugar de los valores espectrales a decodificar.

[000119] Se debe tener en cuenta que la decodificación, que se describe a continuación se utiliza con el propósito de dar lugar a la denominada “codificación espectral sin ruido” de valores espectrales escalados y cuantizados generalmente post–procesado. La codificación espectral sin ruido se utiliza en un concepto de codificación/decodificación de audio (o en cualquier otro concepto de codificación/decodificación) para reducir aun más la redundancia del espectro cuantizado que es utilizado, por ejemplo, por un transformador compactador de energía del dominio del tiempo al dominio de la frecuencia. El esquema de codificación espectral sin ruido, que se utiliza en las realizaciones de la invención, se basa en una codificación aritmética junto con un contexto adaptado dinámicamente.

[000120] En algunas realizaciones de acuerdo con la invención, el esquema de codificación espectral sin ruido se basa en 2–tuplos, es decir, que se combinan dos coeficientes espectrales cercanos. Cada 2–tuplo se divide en el signo, el plano de 2 bits más significativo y el resto de los planos de bits menos significativos. La codificación sin ruido correspondiente al plano de 2 bits más significativo m utiliza tablas de frecuencias cumulativas dependientes del contexto derivadas de cuatro 2–tuplos anteriormente decodificados. La codificación sin ruido es alimentada por los valores espectrales cuantizados y utiliza tablas de frecuencias cumulativas dependientes del contexto derivadas de cuatro 2–tuplos cercanos anteriormente decodificados. En este caso, se toma en cuenta la cercanía, tanto en tiempo como en frecuencia, como se ilustra en la Fig. 4. A continuación as tablas de frecuencias cumulativas (que se explica más adelante) son utilizadas por el codificador aritmético para generar un código binario de longitud variable (y por el decodificador aritmético para derivar valores decodificados de un código binario de longitud variable).

[000121] Por ejemplo, el codificador aritmético 170 produce un código binario correspondiente a una serie dada de símbolos y sus respectivas probabilidades (es decir, dependiendo de las probabilidades respectivas). El código binario se genera mapeando un intervalo de probabilidades, donde se halla la serie de símbolos, contra una palabra clave.

[000122] La codificación sin ruido del resto del plano de bits menos significativos r usa una sola tabla de frecuencias cumulativas. Las frecuencias cumulativas corresponden, por ejemplo, a una distribución uniforme de los símbolos que aparecen en los planos de bits menos significativos, es decir, se estima que existe la misma probabilidad de que aparezca un 0 o un 1 en los planos de bits menos significativos.

[000123] A continuación se presenta otra breve reseña general de la herramienta de codificación espectral sin ruido. La codificación espectral sin ruido se utiliza para reducir aun más la redundancia del espectro cuantizado. El esquema de codificación espectral sin ruido se basa en una codificación aritmética, conjuntamente con un contexto dinámicamente adaptado. La codificación sin ruido es alimentada por los valores espectrales cuantizados y utiliza tablas de frecuencias cumulativas dependientes del contexto derivadas, por ejemplo, de cuatro 2–tuplos de valores espectrales cercanos anteriormente decodificados. En este caso, se toma en cuenta la cercanía, tanto en tiempo como en frecuencia, como se ilustra en la Fig. 4. Las tablas de frecuencias cumulativas son utilizadas posteriormente por el codificador aritmético para generar un código binario de longitud variable.

[000124] El codificador aritmético produce un código binario correspondiente a una serie dada de símbolos y sus respectivas probabilidades. El código binario se genera mapeando un intervalo de probabilidades donde se halla la serie de símbolos, contra una palabra clave.

11. Proceso de Decodificación

11.1 Reseña General del Proceso de decodificación

[000125] A continuación se presenta una reseña general del proceso de de codificación de un valor espectral tomando como referencia la Fig. 3, que ilustra una pseudo–representación de código de programa del proceso de decodificación de una pluralidad de valores espectrales.

[000126] 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 anterior, utilizando la función “arith_map_context(N, arith_reset_flag)”. La derivación del contexto actual de un contexto anterior puede comprender selectivamente un reinicio del contexto. A continuación se describe tanto el reinicio del contexto como la derivación del contexto actual de un contexto anterior.

[000127] La decodificación de una pluralidad de valores espectrales comprende asimismo una iteración de una decodificación de valores espectrales 312 y una actualización del contexto 313, actualización del contexto 313 que es ejecutada por una función “arith_update_context(i, a,b)” que se describe más adelante. La decodificación de valores espectrales 312 y la actualización del contexto 312 se repiten lg/2 veces, donde lg/2 indica el número de 2–tuplos de valores espectrales a decodificar (por ej., correspondiente a un cuadro de audio), a menos que se detecte un llamado símbolo “ARITH_STOP”. Más aun, la decodificación de una serie de lg valores espectrales comprende asimismo una decodificación de signos 314 y un paso de terminación 315.

[000128] La decodificación 312 de un tuplo de valores espectrales comprende un cálculo del valor del contexto 312a, una decodificación de planos de bits más significativos 312b, una detección de símbolo aritmético de parada 312c, una adición de planos de bits menos significativos 312d y una actualización de matriz 312e.

[000129] El cómputo del valor de estado 312a comprende la evocación de la función “arith_get_context(c,i,N)” como se ilustra, por ejemplo, en la Fig. 5c o 5d. En consecuencia, se provee un valor numérico de contexto actual (estado) como valor de retorno de la evocación de función de la función “arith_get_context(c,i,N)”. Como se puede apreciar, el valor numérico de contexto anterior (también indicado con la letra "c"), que sirve como variable de entrada para la función “arith_get_context(c,i,N)”, se actualiza para obtener, como valor de retorno, el valor numérico de contexto actual c.

[000130] La decodificación del plano de bits más significativo 312b comprende una ejecución iterativa de un algoritmo de decodificación 312ba, y una derivación 312bb de valores a,b a partir del valor resultado m del algoritmo 312ba. En la preparación del algoritmo 312ba, se inicializa la variable lev en cero. El algoritmo 312ba se repite hasta que se alcanza una instrucción (o condición) de “interrupción”. El algoritmo 312ba comprende el cómputo de un índice de estado “pki” (que también sirve como índice de tablas de frecuencias cumulativas) dependiendo del valor numérico de contexto actual c, y también dependiendo del valor de nivel “esc_nb” usando una función “arith_get_pk()”, que se describe más adelante (y realizaciones de la cual se ilustran, por ejemplo, en las Figs. 5e y 5f). El algoritmo 312ba comprende asimismo la selección de una tabla de frecuencias cumulativas dependiendo del índice de estado “pki”, que es devuelto por la evocación de la función “arith_get_pk”, donde se puede establecer una variable “cum_freq” en una dirección de partida de una de las 96 tablas de frecuencias cumulativas (o subtablas) dependiendo del índice de estado “pki”. También se puede inicializar una variable “cfl” a una longitud de la tabla de frecuencias cumulativas seleccionada (o de una subtabla) que es igual, por ejemplo, a un número de símbolos del alfabeto, es decir, un número de valores diferentes a decodificar. La longitud de todas las tablas de frecuencias cumulativas (o subtablas) desde “ari_cf_m[pki=0][17]” a “ari_cf_m[pki=95][17]” disponible para la decodificación del valor de plano de bits más significativo m es 17, ya que se pueden decodificar 16 valores de planos de bits más significativos y un símbolo de escape (“ARITH_ESCAPE”).

[000131] Seguidamente, se puede obtener un valor m de plano de bits más significativo mediante la ejecución de una función “arith_decode()”, tomando en cuenta la tabla de frecuencias cumulativas seleccionada (descripta por la variable “cum_freq” y la variable “cfl”). Al derivar el valor de plano de bits más significativo m, se pueden evaluar los bits denominados “acod_m” del flujo de bits 210 (ver, por ejemplo, la Fig. 6g o la Fig. 6h).

[000132] El algoritmo 312ba comprende asimismo la verificación de si el valor del plano de bits más significativo m es igual a un símbolo de escape “ARITH_ESCAPE” o no. Si el valor m del plano de bits más significativo no es igual al símbolo aritmético de escape, el algoritmo 312ba es abortado (condición de “interrupción”) y luego se omite el resto de las instrucciones del 312ba. En consecuencia, la ejecución del proceso continúa con la fijación del valor a en el paso 312bb. Por el contrario, si el valor del plano de bits más significativo m es idéntico al símbolo aritmético de escape, o “ARITH_ESCAPE”, el valor de nivel “lev” se incrementa en uno. Se fija el valor de nivel “esc_nb” de manera que sea igual al valor de nivel “lev”, a menos que la variable “lev” sea superior a siete, en cuyo caso se fija la variable “esc_nb” para que sea igual a siete. Como ya se mencionara, seguidamente se repite el algoritmo 312ba hasta que el valor m de planos de bits más significativo decodificado sea diferente del símbolo aritmético de escape, donde se utiliza un contexto modificado (puesto que el parámetro de entrada de la función "arith_get_pk()" se adapta según el valor de la variable "esc_nb").

[000133] En cuanto se decodifica el plano de bits más significativo utilizando la ejecución única o la ejecución iterativa del algoritmo 312ba, es decir, cuando se ha decodificado un valor de plano de bits más significativo m diferente del símbolo aritmético de escape, se fija la variable de valor espectral “b” de modo que sea igual a una

pluralidad de (por ej. 2) bits más significativos del valor de plano de bits más significativo m, y la variable de valor espectral “a” se fija en los bits más bajos (por ej. 2) del valor de plano de bits más significativo m. Los detalles referentes a esta funcionalidad se pueden ver, por ejemplo, en el número de referencia 312bb.

[000134] Seguidamente, se verifica, en el paso 312c, si hay un símbolo aritmético de parada presente. Este es el caso si el valor m de plano de bits más significativo es igual a cero y la variable “lev” es mayor que cero. En consecuencia, una condición de parada aritmética es señalizada por una condición “inusual”, en la cual el valor m de plano de bits más significativo es igual a cero, en tanto que la variable “lev” indica que hay una ponderación numérica incrementada asociada al valor m del plano de bits más significativo. En otras palabras, se detecta una condición de parada aritmética si el flujo de bits indica que se debe transferir una ponderación numérica incrementada, superior a una ponderación numérica mínima, al valor m del plano de bits más significativo que es igual a cero, que constituye una condición que no aparece en una situación de codificación normal. En otras palabras, se señaliza una condición de parada aritmética si un símbolo de escape aritmético va seguido por un valor de plano de bits más significativo de 0.

[000135] Después de la evaluación de si hay una condición de parada aritmética, que se ejecuta en el paso 212c, se obtienen los planos de bits menos significativos, por ejemplo, como se ilustra con el número de referencia 212d en la Fig. 3. Por cada plano de bits menos significativo, se decodifican dos valores binarios. Uno de los valores binarios está asociado a la variable a (o el primer valor espectral de un tuplo de valores espectrales) y uno de los valores binarios está asociado a la variable b (o un segundo valor espectral de un tuplo de valores espectrales). Un número de planos de bits menos significativos es designado por la variable lev.

[000136] En la decodificación de dichos uno o más planos de bits menos significativos (si los hubiere) se ejecuta iterativamente un algoritmo 212da, donde el número de ejecuciones del algoritmo 212da está determinado por la variable “lev”. Cabe señalar aquí que la primera iteración del algoritmo 212da se ejecuta sobre la base de los valores de las variables a, b fijados en el paso 212bb. Otras iteraciones del algoritmo 212da se han de ejecutar sobre la base de los valores variables actualizados de la variable a, b.

[000137] Al comienzo de una iteración, se selecciona una tabla de frecuencias cumulativas. A continuación, se ejecuta una decodificación aritmética para obtener un valor de una variable r, donde el valor de la variable r describe una pluralidad de bits menos significativos, por ejemplo un bit menos significativo asociado a la variable a y un bit menos significativo asociado a la variable b. La función “ARITH_DECODE" se utiliza para obtener el valor r, donde la tabla de frecuencias cumulativas "arith_cf_r" es utilizada para la decodificación aritmética.

[000138] A continuación, se actualizan los valores de las variables a y b. Para ese fin, se desplaza la variable a hacia la izquierda un bit, y el bit menos significativo de la variable desplazada a se ajusta al valor definido por el bit menos significativo del el valor r. La variable b se desplaza un bit a la izquierda y se asigna al bit menos significativo de la variable b desplazada el valor definido por el bit 1 de la variable r, donde el bit 1 de la variable r tiene una ponderación numérica de 2 en la representación binaria de la variable r. Seguidamente se repite el algoritmo 412ba hasta que todos los bits menos significativos hayan sido decodificados.

[000139] Después de la decodificación de los planos de bits menos significativos, se actualiza una matriz “x_ac_dec” por el hecho de que los valores de las variables a,b se guardan en el elementos de dicha matriz que tienen los índices de matriz 2*i y 2*i+1.

[000140] A continuación, se actualiza el estado del contexto evocando la función “arith_update_context(i,a,b)”, detalles de la cual se explicarán más adelante tomando como referencia la Fig. 5g.

[000141] Con posterioridad a la actualización del estado del contexto, que se ejecuta en el paso 313, se repiten los algoritmos 312 y 313, hasta la variable de ejecución i alcance el valor de lg/2 o hasta que se detecte una condición de parada aritmética.

[000142] A continuación, se ejecuta un algoritmo de terminación “arith_finish()”como se puede apreciar por el número de referencia 315. Más adelante se describen los detalles del algoritmo de terminación “arith_finish()” con referencia a la Fig. 5m.

[000143] Con posterioridad al algoritmo de terminación 315, se decodifican los signos de los valores espectrales empleando el algoritmo 314. Como se puede apreciar, los signos de los valores espectrales que son diferentes de cero son codificados individualmente. En el algoritmo 314, se leen los signos correspondientes a todos los valores espectrales con índices i de entre i=0 y i=lg–1 que no son cero. Por cada valor espectral no cero que tiene un índice de un valor espectral i de entre i=0 e i=lg–1, se lee un valor (típicamente un solo bit) s del flujo de bits. Si el valor de s que se lee del flujo de bits es igual a 1, se invierte el signo de dicho valor espectral. Para ese fin, se accede a la matriz “x_ac_dec”, tanto para determinar si el valor espectral con el índice i es igual a cero como para actualizar el signo de los valores espectrales decodificados. Sin embargo, se debe tener en cuenta que los signos de las variables a, b quedan inalterados en la decodificación de signos 314.

[000144] Mediante la ejecución del algoritmo de terminación 315 antes de la decodificación de signos 314, es posible reiniciar todas las cajas necesarias después de un símbolo ARITH_STOP.

[000145] Cabe señalar aquí que el concepto de obtención de valores de los planos de bits menos significativos no es de particular importancia en algunas realizaciones de acuerdo con la presente invención. En algunas realizaciones, hasta se puede omitir la decodificación de algunos planos de bits menos significativos. Por otro lado, se puede utilizar diferentes algoritmos de decodificación para ese fin.

11.2 Orden de Decodificación de Acuerdo con la Fig. 4

[000146] A continuación se describe el orden de decodificación de los valores espectrales.

[000147] Los coeficientes espectrales cuantizados “x_ac_dec[]” son codificados sin ruido y transmitidos (por ej. en el flujo de bits) a partir del coeficiente de frecuencia más baja y progresando hasta el coeficiente de frecuencia más alta.

[000148] En consecuencia, los coeficientes espectrales cuantizados “x_ac_dec[]”son codificados sin ruido a partir del coeficiente de frecuencia más baja y progresando hasta el coeficiente de frecuencia más alta. Los coeficientes espectrales cuantizados son decodificados por grupos de dos coeficientes sucesivos (por ej. adyacentes en la frecuencia) a y b reunidos en lo que se denomina un 2–tuplo (a,b) (también designado con {a,b}). Cabe señalar aquí que los coeficientes espectrales cuantizados en ocasiones se denominan también “qdec”.

[000149] Los coeficientes decodificados “x_ac_dec[]” correspondientes a un modo en el dominio de la frecuencia (por ej., los coeficientes decodificados para una codificación de audio avanzada, por ejemplo obtenida mediante el uso de una transformada de coseno discreta modificada, de acuerdo con lo estipulado en ISO/IEC 14496, parte 3, subparte 4) y luego almacenados en una matriz “x_ac_quant[g][win][sfb][bin]”. El orden de transmisión de las palabras clave de la codificación sin ruido es tal que cuando son decodificadas en el orden recibido y almacenado en la matriz, el índice “bin” es el que se incrementa más rápidamente y “g” es el índice que se incrementa más lentamente. Dentro de una palabra clave, el orden de codificación es a,b.

[000150] Los coeficientes decodificados “x_ac_dec[]” correspondientes a la excitación codificada por transformadas (TCX) se guardan, por ejemplo, directamente en una matriz “x_tcx_invquant[win][bin]”, y el orden de la transmisión de la palabra clave de codificación sin ruido es tal que cuando se las decodifica en el orden recibido y aguardado en la matriz “bin” es el índice que se incrementa con más rapidez y “win” es el índice que se incrementa más lentamente. Dentro de una palabra clave, el orden de decodificación es a, b. En otras palabras, si los valores espectrales describen una excitación codificada por transformadas del filtro de predicción lineal de un codificador de voz, los valores espectrales a, b están asociados a frecuencias adyacentes y crecientes de la excitación codificada por transformadas. Los coeficientes espectrales asociados a una frecuencia menor son codificados y decodificados por lo general antes que un coeficiente espectral asociado a una frecuencia más alta.

[000151] Notablemente, el decodificador de audio 200 puede estar configurado para aplicar la representación decodificada en el dominio de la frecuencia 232, provista por el decodificador aritmético 230, tanto para una generación “directa” de una representación de señal de audio en el dominio del tiempo utilizando una transformada de señal del dominio de la frecuencia al dominio del tiempo como para una provisión “indirecta” de una representación de señal de audio en el dominio del tiempo utilizando tanto un decodificador del dominio de la frecuencia al dominio del tiempo como un filtro de predicción lineal excitado por la salida del transformador del dominio de la frecuencia al dominio del tiempo.

[000152] En otras palabras, el decodificador aritmético, la funcionalidad del cual se describe en forma detallada aquí, es muy adecuada para decodificar valores espectrales de una representación en el dominio del tiempo–frecuencia de un contenido de audio codificado en el dominio de la frecuencia, para la producción de una representación en el dominio del tiempo–frecuencia de una señal estímulo para un filtro de predicción lineal adaptado para decodificar (o sintetizar) una señal de voz codificada en el dominio de predicción lineal. Por consiguiente, el decodificador aritmético es muy adecuado para usar en un decodificador de audio con capacidad para tratar tanto contenido de audio codificado en el dominio de la frecuencia como contenido de audio codificado en el dominio de predicción lineal–frecuencia (modo de dominio de la predicción lineal excitada codificada por transformadas).

11.3 Inicialización del Contexto de acuerdo con las Figs. 5a y 5b

[000153] A continuación se describe la inicialización del contexto (también denominada “mapeo del contexto”), que se ejecuta en el paso 310.

[000154] La inicialización del contexto comprende un mapeo entre un contexto anterior y un contexto actual de acuerdo con el algoritmo “arith_map_context()”, un primer ejemplo del cual está expuesto en la Fig. 5a y un segundo ejemplo del cual está ilustrado en la Fig. 5b.

[000155] Como se puede apreciar, el contexto actual se almacena en una variable global "q[2][n_context]" que asume la forma de una matriz que tiene una primera dimensión de 2 y una segunda dimensión de “n_context”. Un contexto anterior se puede guardar opcional (aunque no necesariamente) en una variable “qs[n_context]” que toma la forma de una tabla con una dimensión “n_context” (en caso de utilizarla).

[000156] Tomando como referencia el ejemplo de algoritmo “arith_map_context” de la Fig. 5a, la variable N de entrada describe una longitud de una ventana actual y la variable de entrada “arith_reset_flag” indica si se debe reiniciar el contexto. Más aun, la variable global “previous_N” describe una longitud de una ventana anterior. Cabe señalar en este caso que por lo general un número de valores espectrales asociados a una ventana es, por lo menos aproximadamente igual a la mitad de la longitud de dicha ventana en términos de muestras en el dominio del tiempo. Más aun, se debe tener en cuenta que, en consecuencia, un número de 2–tuplos de valores espectrales es por lo menos aproximadamente igual a un cuarto de la longitud de dicha ventana en términos de muestras en el dominio del tiempo.

[000157] Tomando como referencia el ejemplo de la Fig. 5a, el mapeo del contexto se puede ejecutar de acuerdo con el algoritmo “arith_map_context()”. Cabe señalar aquí que la función ”arith_map_context()” establece los elementos “q[0][j]” de la matriz del contexto actual q en cero en el caso de j=0 a j=N/4–1, si la bandera “arith_reset_flag” está activa y, en consecuencia, indica que se debe reiniciar el contexto. De lo contrario, es decir, si la bandera “arith_reset_flag” está inactiva, los elementos “q[0][j]” de la matriz del contexto actual q derivan de los elementos “q[1][k]” de la matriz del contexto actual q. Se debe tener en cuenta que la función “arith_map_context()” de acuerdo con la Fig. 5a fija los elementos “q[0][j]" de la matriz del contexto actual q en los valores “q[1][k]” de la matriz del contexto actual q, si el número de valores espectrales asociado al cuadro de audio actual (por ej., codificado en el dominio de la frecuencia) es idéntico al número de valores espectrales asociado al cuadro de audio anterior en el caso de j=k=0 a j=k=N/4–1.

[000158] Se realiza un mapeo más complicado si el número de valores espectrales asociado al cuadro de audio actual es diferente del número de valores espectrales asociado al cuadro de audio precedente. Sin embargo, los detalles con respecto al mapeo en este caso no son especialmente relevantes para la idea clave de la presente invención, por lo que se hace referencia al seudo código de programa de la Fig. 5a para ver detalles.

[000159] Más aun, un valor de inicialización correspondiente al valor numérico de contexto actual c es devuelto por la función “arith_map_context()”. Este valor de inicialización es, por ejemplo, igual al valor del elemento “q[0][0]” desplazado a 12 bits a la izquierda. En consecuencia, el valor numérico de contexto (actual) c se inicializa correctamente para una actualización iterativa.

[000160] Más aun, la Fig. 5b ilustra otro ejemplo de algoritmo “arith_map_context()” que se puede utilizar como alternativa. Por detalles se hace referencia al pseudo código de programa de la Fig. 5b.

[000161] Como resumen de lo expuesto, la bandera “arith_reset_flag” determina si se debe reiniciar el contexto. Si la bandera es verdadera, se evoca un subalgoritmo de reinicio 500a del algoritmo “arith_map_context()”. Por otro lado, sin embargo, si la bandera “arith_reset_flag” está inactiva (lo que indica que no se debe ejecutar el reinicio del contexto), el proceso de decodificación comienza con una fase de inicialización en la que se actualiza el vector de elementos de contexto (o matriz) q copiando y mapeando los elementos del contexto del cuadro anterior guardado en q[1][] a q[0][]. Los elementos del contexto dentro de q se guardan en 4–bits por 2–tuplo. La copia y/o mapeo del elemento del contexto son ejecutados en un subalgoritmo 500b.

[000162] En el ejemplo de la Fig. 5b, el proceso de decodificación se inicia con una fase de inicialización en que se realiza un mapeo entre el contexto anterior guardado en qs y el contexto del cuadro actual q. El contexto anterior qs se almacena en 2–bits por línea de frecuencia.

11.4 Cómputo de Valores de Estado de acuerdo con las Figs. 5c y 5d

[000163] A continuación se describe el cómputo de los valores de estado 312a en forma más detallada.

[000164] Se describe un primer algoritmo ilustrativo con referencia a la Fig. 5c y se describe un segundo algoritmo ilustrativo con referencia a la Fig. 5d.

[000165] Se debe tener en cuenta que se puede tener el valor numérico de contexto actual c (expuesto en la Fig. 3) como valor de retorno de la función “arith_get_context(c,i,N)”, una representación de seudo código de programa del cual está expuesta en la Fig. 5c. Por otro lado, sin embargo, se puede obtener el valor numérico de contexto actual c como valor de retorno de la función “arith_get_context(c,i)”, una representación de seudo código de programa del cual está expuesta en la Fig. 5d.

[000166] Con respecto al cómputo del valor de estado, también se hace referencia a la Fig. 4, que ilustra el contexto utilizado para la evaluación de estado, es decir, para el cómputo de un valor numérico de contexto actual c. La Fig. 4

ilustra una representación bidimensional de los valores espectrales, tanto en el tiempo como en la frecuencia. Una abscisa 410 describe el tiempo y una ordenada 412 describe la frecuencia. Como se puede apreciar en la Fig. 4, un tuplo 420 de valores espectrales a decodificar (preferentemente usando el valor numérico de contexto actual), está asociado a un índice de tiempo t0 y un índice de frecuencia i. Como se puede apreciar, en el caso del índice de tiempo t0, los tuplos con índices de frecuencia i–1, i–2, e i–3 ya son decodificados en el momento en que se deben decodificar los valores espectrales del tuplo 120, que tiene el índice de frecuencia i. Como se puede apreciar en la Fig. 4, un valor espectral 430 que tiene un índice de tiempo t0 y un índice de frecuencia i–1 ya ha sido decodificado antes de la decodificación del tuplo 420 de valores espectrales, y se considera el tuplo 430 de valores espectrales para el contexto utilizado para la decodificación del tuplo 420 de valores espectrales. Del mismo modo, un tuplo 440 de valores espectrales con un índice de tiempo t0–1 y un índice de frecuencia de i–1, un tuplo 450 de valores espectrales con un índice de tiempo t0–1 y un índice de frecuencia de i, y un tuplo 460 de valores espectrales con un índice de tiempo t0–1 y un índice de frecuencia de i+1, ya han sido decodificados antes de la decodificación del tuplo 420 de valores espectrales, y se los considera para la determinación del contexto, que se utiliza para decodificar el tuplo 420 de valores espectrales. Los valores espectrales (coeficientes) ya decodificados en el momento de decodificar los valores espectrales del tuplo 420 y considerados para el contexto están indicados por un cuadrado sombreado. Por el contrario, otros valores espectrales ya decodificados (en el momento de la decodificación de los valores espectrales del tuplo 420) aunque no tenidos en cuenta para el contexto (para la decodificación de los valores espectrales del tuplo 420) están representados por cuadrados con líneas de guiones y otros valores espectrales (que aún no han sido decodificados en el momento de la decodificación de los valores espectrales del tuplo 420) están indicados por círculos con líneas de guiones. Los tuplos representados por cuadrados con líneas de guiones y los tuplos representados por círculos con líneas de guiones no se utilizan para determinar el contexto para la decodificación de los valores espectrales del tuplo 420.

[000167] Sin embargo, se debe tener en cuenta que algunos de estos valores espectrales, que o son utilizados para el cómputo “regular” o “normal” del contexto para decodificar los valores espectrales del tuplo 420 pueden ser evaluados, de todas maneras, para la detección de una pluralidad de valores espectrales adyacentes previamente decodificados que cumplan, individualmente o en su conjunto, con una condición predeterminada con respecto a sus magnitudes. Los detalles con respecto a este tema se describen más adelante.

[000168] Tomando ahora como referencia la Fig. 5c, se describen detalles del algoritmo “arith_get_context(c,i,N)”. La Fig. 5c ilustra la funcionalidad de dicha función “arith_get_context(c,i,N)” en forma de seudo código de programa que utiliza las convenciones del conocidísimo lenguaje C y/o el lenguaje C++. Por consiguiente, se describen algunos detalles más con respecto al cálculo del valor numérico de contexto actual “c” que es ejecutado por la función “arith_get_context(c,i,N)”.

[000169] Se debe tener en cuenta que la función “arith_get_context(c,i,N)” recibe, como variables de entrada, un “contexto de estado anterior” que puede ser descripto por un valor numérico de contexto anterior c. La función “arith_get_context(c,i,N)” también recibe, como variable de entrada, un índice i de un 2–tuplo de valores espectrales a decodificar. El índice i es por lo general un índice de frecuencia. Una variable de entrada N describe una longitud de ventana de una ventana, para la cual se decodifican los valores espectrales.

[000170] La función “arith_get_context(c,i,N)” produce, como valor de salida, una versión actualizada de la variable de entrada c, que describe un contexto de estado actualizado y que puede ser considerado valor numérico de contexto actual. Para resumir, la función “arith_get_context(c,i,N)” recibe un valor numérico de contexto anterior c como variable de entrada y produce una versión actualizada del mismo, que se considera como valor numérico de contexto actual. Además, la función “arith_get_context” considera las variables i, N, y también accede a la matriz “global” q[][].

[000171] En lo que respecta a los detalles de la función “arith_get_context(c,i,N)”, se debe tener en cuenta que la variable c, que en un principio representa el valor numérico de contexto anterior en forma binaria, se desplaza 4 bits a la derecha en el paso 504a. En consecuencia, se descartan los cuatro bits menos significativos del valor numérico de contexto anterior (representado por la variable de entrada c). Además, se reducen las ponderaciones numéricas de los demás bits de los valores numéricos de contexto anterior, por ejemplo, en un factor de 16.

[000172] Más aún, si el índice i del 2–tuplo es inferior a N/4–1, es decir, si no asume un valor máximo, se modifica el valor numérico de contexto actual por el hecho de que se agrega el valor del elemento q[0][i+1] a los bits 12 a 15 (es decir, a los bits que tienen una ponderación numérica de 2^{12}, 2^{13}, 2^{14}, y 2^{15}) del valor de contexto desplazado que se obtiene en el paso 504a. Para este fin, el elemento q[0][i+1] de la matriz q[][] (o, más precisamente, una representación binaria del valor representado por dicho elemento) se desplaza 12 bits a la izquierda. Luego se suma la versión desplazada del valor representado por el elemento q[0][i+1] al valor de contexto c, obtenido en el paso 504a, es decir, a una representación numérica desplazada en bits (desplazada 4 bits a la derecha) del valor numérico de contexto anterior. Cabe señalar aquí que el elemento q [0][i+1] de la matriz q[][] representa un valor de subregión asociado a una porción anterior del contenido de audio (por ej., una porción del contenido de audio que tiene el índice temporal t0–1, definido con referencia a la Fig. 4), y con una frecuencia mayor (por ej. una frecuencia con un índice de frecuencia i+1, como se define con referencia a la Fig. 4) que el tuplo de valores espectrales a codificar ahora (usando el valor numérico de contexto actual c presentado como salida por la función

"arith_get_context(c,i,N)"). En otras palabras, si el tuplo 420 de valores espectrales se debe decodificar usando el valor numérico de contexto actual, el elemento q[0][i+1] se puede basar en el tuplo 460 de valores espectrales previamente decodificados.

[000173] Se indica una adición selectiva del elemento q[0][i+1] de la matriz q[][] (desplazado 12 bits a la izquierda) con el número de referencia 504b. Como se puede apreciar, la suma del valor representado por el elemento q[0][i+1] sólo se ejecuta, naturalmente, si el índice de frecuencia i no designa un tuplo de valores espectrales con el índice de frecuencia más elevado i=N/4–1.

[000174] A continuación, en el paso 504c, se ejecuta una operación Y Booleana, en la cual se combina en Y el valor de la variable c con un valor hexadecimal de 0xFFF0 para obtener un valor actualizado de la variable c. Al ejecutar dicha operación Y, se fijan los cuatro bits menos significativos de la variable c efectivamente en cero.

[000175] En el paso 504d, se suma el valor del elemento q[1][i–1] al valor de la variable c, que se obtiene por el paso 504c, para actualizar así el valor de la variable c. Sin embargo, dicha actualización de la variable c en el paso 504d sólo es ejecutada si el índice de frecuencia i del 2–tuplo a decodificar es mayor que cero. Se debe tener en cuenta que el elemento q[1][i–1] es un valor de subregión de contexto basado en un tuplo de valores espectrales previamente decodificados de la porción actual del contenido de audio en el caso de frecuencias inferiores a las frecuencias de los valores espectrales a decodificar usando el valor numérico de contexto actual. Por ejemplo, el elemento q[1][i–1] de la matriz q[][] puede estar asociado al tuplo 430 que tiene el índice de tiempo t0 y el índice de frecuencia i–1, si se presume que el tuplo 420 de valores espectrales se debe decodificar usando el valor numérico de contexto actual devuelto por la presente ejecución de la función "arith_get_context(c,i,N)".

[000176] Para resumir, los bits 0, 1, 2, y 3 (es decir, una porción de cuatro bits menos significativos) del valor numérico de contexto anterior son descartados en el paso 504a desplazándolos de la representación en números binarios del valor numérico de contexto anterior. Más aun, se determinan los bits 12, 13, 14, y 15 de la variable desplazada c (es decir, del valor numérico del contexto anterior desplazado) de manera que tengan valores definidos por el valor de subregión de contexto q[0][i+1] en el paso 504b. A los bits 0, 1, 2, y 3 del valor numérico desplazado del contexto anterior (es decir, los bits 4, 5, 6, y 7 del valor numérico de contexto anterior original) se sobrescribe el valor de subregión de contexto q[1][i–1] en los pasos 504c y 504d.

[000177] En consecuencia, se puede decir que los bits 0 a 3 del valor numérico de contexto anterior representan el valor de subregión de contexto asociado al tuplo 432 de valores espectrales, los bits 4 a 7 del valor numérico de contexto anterior representan el valor de subregión de contexto asociado a un tuplo 434 de valores espectrales anteriormente decodificados, los bits 8 a 11 del valor numérico de contexto anterior representan el valor de subregión de contexto asociado al tuplo 440 de valores espectrales previamente decodificados y los bits 12 a 15 del valor numérico de contexto anterior representan un valor de subregión de contexto asociado al tuplo 450 de valores espectrales previamente decodificados. El valor numérico de contexto anterior, que es transmitido como entrada a la función "arith_get_context(c,i,N)", está asociado a la decodificación del tuplo 430 de valores espectrales.

[000178] El valor numérico de contexto actual, que se obtiene como variable de salida de la función "arith_get_context(c,i,N)", está asociado a la decodificación del tuplo 420 de valores espectrales. En consecuencia, los bits 0 a 3 de los valores numéricos de contexto actual describen el valor de subregión de contexto asociado al tuplo 430 de los valores espectrales, los bits 4 a 7 del valor numérico de contexto actual describen el valor de subregión de contexto asociado al tuplo 440 de valores espectrales, los bits 8 a11 del valor numérico de contexto actual describen el valor numérico de subregión asociado al tuplo 450 del valor espectral y los bits 12 a 15 del valor numérico de contexto actual describen el valor de subregión de contexto asociado al tuplo 460 de valores espectrales. Por consiguiente, se puede apreciar que una porción del valor numérico de contexto anterior, es decir los bits 8 a 15 del valor numérico de contexto anterior, también está incluida en el valor numérico de contexto actual, como bits 4 a 11 del valor numérico de contexto actual. Por el contrario, los bits 0 a 7 del valor numérico del contexto actual anterior se descartan al derivar la representación numérica del valor numérico de contexto actual de la representación numérica del valor numérico de contexto anterior.

[000179] En el paso 504e, la variable c, que representa el valor numérico de contexto actual es actualizada selectivamente si el índice de frecuencia i del 2–tuplo a decodificar es mayor que un número predeterminado, por ejemplo, 3. En este caso, es decir, si i es mayor que 3, se determina si la suma de los valores de subregión de contexto q[1][i–3], q[1][i–2], y q[1][i–1] es menor (o igual) a un valor predeterminado, por ejemplo de 5. Si se encuentra que la suma de dichos valores de subregión de contexto es menor que dicho valor predeterminado, se suma un valor hexadecimal de, por ejemplo, 0x10000, a la variable c. En consecuencia, se determina la variable c de tal manera que la variable c indique si hay una condición en la cual los valores de subregión de contexto q[1][i–3], q[1][i–2], y q[1][i–1] comprenden un valor de suma particularmente pequeño. Por ejemplo, el bit 16 del valor numérico de contexto actual puede actuar como bandera para indicar esa condición.

[000180] En conclusión, el valor de retorno de la función "arith_get_context(c,i,N)" se determina en los pasos 504a, 504b, 504c, 504d, y 504e, donde se deriva el valor numérico de contexto actual del valor numérico de contexto anterior en los pasos 504a, 504b, 504c, y 504d, y donde en el paso 504e se deriva una bandera que indica un

entorno de valores espectrales anteriormente decodificados que tienen, término medio, valores absolutos particularmente pequeños y se suman a la variable c. En consecuencia, el valor de la variable c obtenida en los 504a, 504b, 504c, 504d es devuelto, en el paso 504f, como valor de retorno de la función “arith_get_context(c,i,N)”, si no se cumple la condición evaluada en el paso 504e. Por el contrario, el valor de la variable c, derivado en los pasos 504a, 504b, 504c, y 504d, se incrementa en un valor hexadecimal de 0x10000 y el resultado de esta operación de incremento es devuelto en el paso 504e, si se cumple la condición evaluada en el paso 540e.

[000181] Como resumen de lo expuesto, se debe tener en cuenta que el decodificador sin ruido produce una salida de 2–tuplos de coeficientes espectrales cuantizados sin signos (como se describe más adelante en forma más detallada). En primer lugar se calcula el estado c del contexto sobre la base de los coeficientes espectrales anteriormente decodificados que “rodean” al 2–tuplo a decodificar. En una realización preferida, se actualiza en incrementos el estado (que está representado, por ejemplo, por un valor numérico de contexto) utilizando el estado del contexto del último 2–tuplo decodificado (lo que se denomina valor numérico de contexto anterior), considerando sólo dos nuevos 2–tuplos (por ejemplo, los 2–tuplos 430 y 460). El estado es codificado en 17–bits (por ej., usando una representación numérica de un valor numérico de contexto actual) y es devuelto por la función “arith_get_context()”. Por detalles, se hace referencia a la representación de código de programa de la Fig. 5c.

[000182] Más aun, se debe tener en cuenta que en la Fig. 5d se ilustra un seudo código de programa de una realización alternativa de una función “arith_get_context()”. La función “arith_get_context(c,i)” de acuerdo con la Fig. 5d es similar a la función “arith_get_context(c,i,N)” de acuerdo con la Fig. 5c. Sin embargo, la función “arith_get_context(c,i)” de acuerdo con la Fig. 5d no comprende un manejo o decodificación especial de tuplos de valores espectrales que comprenden un índice de frecuencia mínimo de i=0 o un índice de frecuencia máximo de i=N/4–1.

11.5 Selección de Reglas de Mapeo

[000183] A continuación se describe la selección de una regla de mapeo, por ejemplo, una tabla de frecuencias cumulativas que describe el mapeo de un valor de palabra clave sobre un código de símbolo. La selección de la regla de mapeo se realiza dependiendo de un estado del contexto, que está definido por el valor numérico de contexto actual c.

11.5.1 Selección de Reglas de Mapeo Utilizando el algoritmo de acuerdo con la Fig. 5e

[000184] A continuación se describe la selección de una regla de mapeo usando la función “arith_get_pk(c)”. Se debe tener en cuenta que la función “arith_get_pk()” es evocada al comienzo del subalgoritmo 312ba al decodificar un valor de código “acod_m” para otorgar un tuplo de valores espectrales. Se debe tener en cuenta que la función “arith_get_pk(c)” es evocada con diferentes argumentos en diferentes iteraciones del algoritmo 312b. Por ejemplo, en una primera iteración del algoritmo 312b, la función “arith_get_pk(c)” es evocada con un argumento igual al valor numérico de contexto actual c, provisto por la ejecución previa de la función “arith_get_context(c,i,N)” en el paso 312a. Por el contrario, en otras iteraciones del subalgoritmo 312ba, la función “arith_get_pk(c)” es evocada con un argumento que es la suma del valor numérico de contexto actual c provisto por la función “arith_get_context(c,i,N)” en el paso 312a, y una versión desplazada en bits del valor de la variable “esc_nb”, donde el valor de la variable “esc_nb” está desplazada 17 bits a la izquierda. Por consiguiente, el valor numérico de contexto actual c provisto por la función “arith_get_context(c,i,N)” se utiliza como valor de entrada de la función “arith_get_pk()” en la primera iteración del algoritmo 312ba, es decir, en la decodificación de valores espectrales comparativamente bajos. Por el contrario, al decodificar valores espectrales comparativamente altos, se modifica la variable de entrada de la función “arith_get_pk()” por tomar en cuenta el valor de la variable “esc_nb”, como se ilustra en la Fig. 3.

[000185] Tomando ahora como referencia la Fig. 5e, que ilustra una representación de seudo código de programa de una primera realización de la función “arith_get_pk(c)”, se debe tener en cuenta que la función “arith_get_pk()” recibe la variable c como valor de entrada, donde la variable c describe el estado del contexto, y donde la variable de entrada c de la función “arith_get_pk()” es igual al valor numérico de contexto actual provisto como variable de retorno por la función “arith_get_context()” por lo menos en algunos casos. Más aun, se debe tener en cuenta que la función “arith_get_pk()” provee, como variable de salida, la variable “pki”, que describe un índice de un modelo de probabilidades y que puede ser considerado valor índice de regla de mapeo.

[000186] Haciendo referencia a la Fig. 5e, se puede apreciar que la función “arith_get_pk()” comprende la inicialización de una variable 506a, donde la variable “i_min” se inicializa para tomar el valor de –1. De modo similar, se fija la variable i para que sea igual a la variable “i_min”, por lo que la variable i también se inicializa a un valor de –1. La variable “i_max” se inicializa para tomar un valor que es 1 menor que el número de elementos de la tabla “ari_lookup_m[]” (detalles de la cual se describen con referencia a la Figs. 21(1) y 21(2)). En consecuencia, las variables “i_min” y “i_max” definen un intervalo.

[000187] A continuación se realiza una búsqueda 506b para identificar un valor índice que designa un elemento de la tabla “ari_hash_m”, por lo que el valor de la variable de entrada c de la función “arith_get_pk()” yace dentro de un intervalo definido por dicho elemento y un elemento adyacente.

[000188] En la búsqueda 506b, se repite un subalgoritmo 506ba, en tanto que la diferencia entre las variables “i_max” y “i_min” es superior a 1. En el subalgoritmo 506ba, se fija la variable i de manera que sea igual a una media aritmética de los valores de las variables “i_min” e “i_max”. En consecuencia, la variable i designa un elemento de la tabla “ari_hash_m[]” en el centro de un intervalo de la tabla definido por los valores de las variables “i_min” e “i_max”. A continuación, se determina que la variable j ha de ser igual al valor del elemento “ari_hash_m[i]” de la tabla “ari_hash_m[]”. Por consiguiente, la variable j asume un valor definido por un elemento de la tabla “ari_hash_m[]”, elemento que se halla en el centro de un intervalo de la tabla definido por las variables “i_min” e “i_max”. A continuación, se actualiza el intervalo definido por las variables “i_min” e “i_max” si el valor de la variable de entrada c de la función “arith_get_pk()” es diferente de un valor de estado definido por los bits más altos del elemento de la tabla “j=ari_hash_m[i]” de la tabla “ari_hash_m[]”. Por ejemplo, los bits “más altos” (los bits 8 y superiores) de los elementos de la tabla “ari_hash_m[]”describen valores de estado significativos. En consecuencia, el valor “j>>8” describe un valor de estado significativo representado por el elemento “j=ari_hash_m[i]” de la tabla “ari_hash_m[]” designado por el valor índice de tabla hash i. En consecuencia, si el valor de la variable c es menor que el valor “j>>8”, esto significa que el valor de estado descripto por la variable c es menor que un valor de estado significativo descripto por el elemento “ari_hash_m[i]” de la tabla “ari_hash_m[]”. En este caso, el valor de la variable “i_max” se fija igual al valor de la variable i, lo que a su vez tiene el efecto de reducir el tamaño del intervalo definido por “i_min” y “i_max”, donde el nuevo intervalo es aproximadamente igual a la mitad inferior del intervalo anterior. Si se encuentra que la variable de entrada c de la función “arith_get_pk()” es mayor que el valor “j>>8”, lo que significa que el valor de contexto descripto por la variable c es mayor que un valor de estado significativo descripto por el elemento “ari_hash_m[i]” de la matriz “ari_hash_m[]”, el valor de la variable “i_min” se debe fijar igual al valor de la variable i. En consecuencia, se reduce el tamaño del intervalo definido por los valores de las variables “i_min” e “i_max” a aproximadamente la mitad del tamaño del intervalo anterior definido por los valores anteriores de las variables “i_min” e “i_max”. Para ser más precisos, el intervalo definido por el valor actualizado de la variable “i_min” y por el valor anterior (inalterado) de la variable “i_max” es aproximadamente igual a la mitad superior del intervalo anterior en caso de que el valor de la variable c sea mayor que el valor de estado significativo definido por el elemento “ari_hash_m[i]”.

[000189] Sin embargo, si se encuentra que el valor de contexto descripto por la variable de entrada c del algoritmo “arith_get_pk()” es igual al valor de estado significativo definido por el elemento “ari_hash_m[i]” (es decir, c==(j>>8)), un valor índice de regla de mapeo definido por los 8 bits más bajos del elemento “ari_hash_m[i]” es devuelto como valor de retorno de la función “arith_get_pk()” (instrucción “return (j&0xFF)”).

[000190] Como resumen de lo expuesto, se evalúa un elemento “ari_hash_m[i]”, los bits más altos del cual (los bits 8 y superiores) describen un valor de estado significativo, en cada iteración 506ba, y se compara el valor de contexto (o valor numérico de contexto actual) descripto por la variable de entrada c de la función “arith_get_pk()” con el valor de estado significativo descripto por dicho elemento de la tabla “ari_hash_m[i]”. Si el valor de contexto representado por la variable de entrada c es menor que el valor de estado significativo representado por el elemento de la tabla “ari_hash_m[i]”, el límite superior (descripto por el valor “i_max”) del intervalo de la tabla se reduce, y si el valor de contexto descripto por la variable de entrada c es mayor que el valor de estado significativo descripto por el elemento de la tabla “ari_hash_m[i]”, se incrementa el límite inferior (descripto por el valor de la variable “i_min”) del intervalo de la tabla. En ambos casos mencionados, se repite el subalgoritmo 506ba, a menos que el tamaño del intervalo (definido por la diferencia entre “i_max” y “i_min”) sea menor o igual a 1. Si, por el contrario, el valor de contexto descripto por la variable c es igual al valor de estado significativo descripto por el elemento de la tabla “ari_hash_m[i]”, se aborta la función “arith_get_pk()”, donde el valor de retorno está definido por los 8 bits más bajos del elemento de la tabla “ari_hash_m[i]”.

[000191] Sin embargo, si se da por terminada la búsqueda 506b porque el tamaño del intervalo alcanza su valor mínimo (“i_max – “i_min” es menor o igual a 1), el valor de retorno de la función “arith_get_pk()” está determinado por un elemento “ari_lookup_m[i_max]” de una tabla “ari_lookup_m[]”, lo que se puede ver en el número de referencia 506c. En consecuencia, los elementos de la tabla “ari_hash_m[]” definen tanto valores de estado significativos como los límites de los intervalos. En el subalgoritmo 506ba, los límites de los intervalos de búsqueda “i_min” e “i_max” se adaptan iterativamente de manera que el elemento “ari_hash_m[i]” de la tabla “ari_hash_m[]”, un índice de tabla hash que se halla, por lo menos aproximadamente, en el centro del intervalo de búsqueda definido por los valores límite de intervalo “i_min” e “i_max”, por lo menos es aproximado a un valor de contexto descripto por la variable de entrada c. Por lo tanto se obtiene que el valor de contexto descripto por la variable de entrada c se halla dentro de un intervalo definido por “ari_hash_m[i_min]” y “ari_hash_m[i_max]” una vez completadas las iteraciones del subalgoritmo 506ba, a menos que el valor de contexto descripto por la variable de entrada c sea igual a un valor de estado significativo descripto por un elemento de la tabla “ari_hash_m[]”.

[000192] Sin embargo, si la repetición iterativa del subalgoritmo 506ba se interrumpe porque el tamaño del intervalo (definido por “i_max – i_min”) alcanza o excede su valor mínimo, se presume que el valor de contexto descripto por la variable de entrada c no es un valor de estado significativo. En ese caso, de todas maneras se utiliza el índice “i_max”, que designa un límite superior del intervalo. El valor superior “i_max” del intervalo, que es alcanzado en la última iteración del subalgoritmo 506ba, se vuelve a utilizar como valor índice de la tabla para el acceso a la tabla “ari_lookup_m”. La tabla “ari_lookup_m[]” describe valores índice de regla de mapeo asociados a intervalos de una

pluralidad de valores numéricos de contexto adyacentes. Los intervalos a los cuales están asociados los valore índice de regla de mapeo descriptos por los elementos de la tabla “ari_lookup_m[]”, están definidos por los valores de estado significativos descriptos por los elementos de la tabla “ari_hash_m[]”. Los elementos de la tabla “ari_hash_m” definen tanto valores de estado significativos como límites de los intervalos de los intervalos de valores numéricos de contexto. En la ejecución del algoritmo 506b, se determina si el valor numérico de contexto actual descripto por la variable de entrada c es igual a un valor de estado significativo, y si éste no es el caso, en qué intervalo de valores numéricos de contexto (de una pluralidad de intervalos, los límites de los cuales están definidos por los valores de estado significativos) se encuentra el valor de contexto descripto por la variable de entrada c. De esa manera, el algoritmo 506b cumple una doble funcionalidad para determinar si la variable de entrada c describe un valor de estado significativo y, si ese no es el caso, para identificar un intervalo, limitado por valores de estado significativos, en el cual se halla el valor de contexto representado por la variable de entrada c. En consecuencia, el algoritmo 506e es particularmente eficiente y requiere sólo un número comparativamente pequeño de accesos de la tabla.

[000193] Como resumen de lo expuesto, el estado del contexto c determina la tabla de frecuencias cumulativas utilizada para decodificar el plano de 2 bits más significativos m. El mapeo de c al correspondiente índice de tabla de frecuencias cumulativas “pki” es ejecutado por la función “arith_get_pk()”. Una seudo representación de código de programa de dicha función “arith_get_pk()” ha sido explicada con referencia a la Fig. 5e.

[000194] Para resumir adicionalmente lo expuesto, se decodifica el valor m usando la función “arith_decode()” (lo que se describe más adelante con más detalle) evocada con la tabla de frecuencias cumulativas “arith_cf_m[pki][]”, donde “pki” corresponde al índice (también denominado valor índice de regla de mapeo) retornado por la función “arith_get_pk()”, que se describe con referencia a la fig 5e.

11.5.2 Selección de Reglas de Mapeo Utilizando el algoritmo de acuerdo con la Fig. 5f

[000195] A continuación se describe otra realización de un algoritmo de selección de reglas de mapeo “arith_get_pk()” con referencia a la Fig. 5f que ilustra una representación de seudo código de programa de un algoritmo que se puede utilizar en la decodificación de un tuplo de valores espectrales. El algoritmo de acuerdo con la Fig. 5f se puede considerar una versión optimizada (por ej., versión optimizada en la velocidad) del algoritmo, "get_pk()" o del algoritmo "arith_get_pk()".

[000196] El algoritmo “arith_get_pk()” de acuerdo con la Fig. 5f recibe, como variable de entrada, una variable c que describe el estado del contexto. La variable de entrada c puede representar, por ejemplo, un valor numérico de contexto actual.

[000197] El algoritmo “arith_get_pk()” produce, como variable de salida, una variable “pki”, que describe y un índice de distribución de probabilidades (o modelo de probabilidades) asociado a un estado del contexto descripto por la variable de entrada c. La variable “pki” puede ser, por ejemplo, un valor índice de regla de mapeo.

[000198] El algoritmo de acuerdo con la Fig 5f comprende una definición del contenido de la matriz “i_diff[]”. Como se puede apreciar, un primer elemento de la matriz “i_diff[]” (que tiene un índice de matriz 0) es igual a 299 y los demás elementos de la matriz (que tienen índices de matriz de 1 a 8) asumen los valores de 149, 74, 37, 18, 9, 4, 2, y 1. En consecuencia, el tamaño de paso para la selección de un valor índice de la tabla hash “i_min” se reduce con cada iteración, ya que los elementos de las matrices “i_diff[]” definen dichos tamaños de pasos. Para más detalles, se hace referencia a la explicación siguiente.

[000199] Sin embargo, en realidad se puede optar por diferentes tamaños de paso, por ej. diferentes contenidos de la matriz “i_diff[]”, donde el contenido de la matriz “i_diff[]” puede ser naturalmente adaptado a un tamaño de la tabla hash “ari_hash_m[i]”.

[000200] Se debe tener en cuenta que la variable “i_min” se inicializa tomando un valor de 0 justo al comienzo del algoritmo “arith_get_pk()”.

[000201] En un paso de inicialización 508a, se inicializa una variable s dependiendo de la variable de entrada c, donde una representación numérica de la variable c se desplaza 8 bits a la izquierda a fin de obtener la representación numérica de la variable s.

[000202] A continuación, se realiza una búsqueda de tablas 508b para identificar un valor índice de tabla hash “i_min” de un elemento de la tabla hash “ari_hash_m[]”, de tal manera que el valor de contexto descripto por el valor de contexto c se halle dentro de un intervalo que está limitado por el valor de contexto descripto por el elemento de la tabla hash “ari_hash_m[i_min]” y un valor de contexto descripto por otro elemento de la tabla hash “ari_hash_m” elemento adicional éste “ari_hash_m” que está adyacente (en términos de su valor índice de tabla hash) al elemento de la tabla hash “ari_hash_m[i_min]”. Por consiguiente, el algoritmo 508b da lugar a la determinación de un valor índice de la tabla hash “i_min” que designa un elemento “j=ari_hash_m[i_min]” de la tabla hash “ari_hash_m[]”, por lo

que el elemento de la tabla hash “ari_hash_m[i_min]” por lo menos se aproxima al valor de contexto descripto por la variable de entrada c.

[000203] La búsqueda de tablas 508b comprende una ejecución iterativa de un subalgoritmo 508ba, donde el subalgoritmo 508ba es ejecutado un número predeterminado de iteraciones, por ejemplo nueve. En el primer paso del subalgoritmo 508ba, se fija la variable i en un valor igual a la suma del valor de una variable “i_min” y un valor de un elemento de la taba “i_diff[k]”. Cabe señalar aquí que k es una variable de ejecución que se incrementa, a partir de un valor inicial de k=0, con cada iteración del subalgoritmo 508ba. La matriz “i_diff[]” define valores en incrementos predeterminados, donde los valores de incremento se reducen con el aumento del índice de la tabla k, es decir, con los números crecientes de iteraciones.

[000204] En un segundo paso del subalgoritmo 508ba, se copia un valor de un elemento de la tabla “ari_hash_m[]” en una variable j. Preferentemente, los bits más altos de los elementos de tabla de la tabla “ari_hash_m[]”describen un valor de estado significativo de un valor numérico de contexto y los bits más bajos (bits 0 a 7) de los elementos de la tabla “ari_hash_m[]”describen valores índice de regla de mapeo asociados a los respectivos valores de estado significativos.

[000205] En un tercer paso del subalgoritmo 508ba, se compara el valor de la variable S con el valor de la variable j, y se fija selectivamente la variable “i_min” en el valor “i+1” si el valor de la variable s es mayor que el valor de la variable j. A continuación se repite el primer paso, el segundo paso, y el tercer paso del subalgoritmo 508ba u número predeterminado de veces, por ejemplo nueve veces. Por consiguiente, en cada ejecución del subalgoritmo 508ba, el valor de la variable “i_min” se incrementa en i_diff[]+1, si el valor de contexto descripto por el índice de tabla hash válido actualmente i_min + i_diff[] es menor que el valor de contexto descripto por la variable de entrada c y sólo en ese caso. En consecuencia, el valor índice de tabla hash “i_min” se incrementa (iterativamente) en cada ejecución del subalgoritmo 508ba si (y sólo en ese caso) el valor de contexto descripto por la variable de entrada c y, en consecuencia, por la variable s, es mayor que el valor de contexto descripto por el elemento “ari_hash_m[i=i_min + diff[k]]”.

[000206] Más aun, se debe tener en cuenta que sólo se ejecuta una sola comparación, es decir la comparación de si el valor de la variable s es mayor que el valor de la variable j en cada ejecución del subalgoritmo 508ba. En consecuencia, el algoritmo 508ba es particularmente eficiente desde el punto de vista informático. Más aun, se debe tener en cuenta que hay diferentes resultados posibles con respecto al valor final de la variable “i_min”. Por ejemplo, es posible que el valor de la variable “i_min”, después de la última ejecución del subalgoritmo 512ba, sea tal que el valor de contexto descripto por el elemento de la tabla “ari_hash_m[i_min]” sea menor que el valor de contexto descripto por la variable de entrada c, y que el valor de contexto descripto por el elemento de la tabla “ari_hash_m[i_min +1]” sea mayor que el valor de contexto descripto por la variable de entrada c. Por otro lado, puede ocurrir que después de la última ejecución del subalgoritmo 508ba, el valor de contexto descripto por el elemento de la tabla hash “ari_hash_m[i_min –1]” sea menor que el valor de contexto descripto por la variable de entrada c, y que el valor de contexto descripto por el elemento “ari_hash_m[i_min]” sea mayor que el valor de contexto descripto por la variable de entrada c. Por otro lado, sin embargo, puede ocurrir que el valor de contexto descripto por el elemento de la tabla hash “ari_hash_m[i_min]” sea idéntico al valor de contexto descripto por la variable de entrada c.

[000207] Por esta razón, se ejecuta la provisión de un valor de retorno basado en decisiones 508c. Se fija la variable j para que tome el valor del elemento de la tabla hash “ari_hash_m[i_min]”. A continuación, se determina si el valor de contexto descripto por la variable de entrada c (y también por la variable s) es mayor que el valor de contexto descripto por el elemento “ari_hash_m[i_min]” (primer caso definido por la condición “s>j”), o si el valor de contexto descripto por la variable de entrada c es menor que el valor de contexto descripto por el elemento de la tabla hash “ari_hash_m[i_min]” (segundo caso definido por la condición “c<j>>8”), o si el valor de contexto descripto por la variable de entrada c es igual al valor de contexto descripto por el elemento “ari_hash_m[i_min]” (tercer caso).

[000208] En el primer caso, (s>j), se devuelve un elemento “ari_lookup_m[i_min +1]” de la tabla “ari_lookup_m[]” designado por el valor índice de tabla “i_min+1” como valor de salida de la función “arith_get_pk()”. En el segundo caso (c<(j>>8)), se devuelve un elemento “ari_lookup_m[i_min]” de la tabla “ari_lookup_m[]” designado por el valor índice de tabla “i_min” como valor de retorno de la función “arith_get_pk()”. En el tercer caso (es decir, si el valor de contexto descripto por la variable de entrada c es igual al valor de estado significativo descripto por el elemento de la tabla “ari_hash_m[i_min]”), se devuelve un valor índice de regla de mapeo descripto por los 8 bits más bajos del elemento de la tabla hash “ari_hash_m[i_min]” como valor de retorno de la función “arith_get_pk()”.

[000209] Como resumen de lo expuesto, se ejecuta una búsqueda de tablas particularmente sencilla en el paso 508b, donde la búsqueda de tablas otorga un valor de variable de una variable “i_min” sin distinguir si el valor de contexto descripto por la variable de entrada c es igual a un valor de estado significativo definido por uno de los elementos de estado de la tabla “ari_hash_m[]” o no. En el paso 508c, que se ejecuta con posterioridad a la búsqueda de tablas 508b, se evalúa la relación de magnitud entre el valor de contexto descripto por la variable de entrada c y un valor de estado significativo descripto por el elemento de tabla hash “ari_hash_m[i_min]” y se selecciona el valor de retorno de la función “arith_get_pk()”dependiendo de un resultado de dicha evaluación, donde

el valor de la variable “i_min”, que se determina en la evaluación de la tabla 508b, es tenido en cuenta para seleccionar un valor índice de regla de mapeo aun si el valor de contexto descripto por la variable de entrada c es diferente del valor de estado significativo descripto por el elemento de tabla hash “ari_hash_m[i_min]” .

[000210] Se debe notar asimismo que la comparación en el algoritmo se debe hacer preferentemente (o por otro lado) entre el índice de contexto (valor numérico de contexto) c y j=ari_hash_m[i]>>8. De hecho, cada elemento de la tabla "ari_hash_m[]" representa un índice de contexto, codificado más allá del 8º bit, y su correspondiente modelo de probabilidades codificada en los 8 primeros bits (bits menos significativos). En la presente implementación, estamos interesados principalmente en saber si el presente contexto c es mayor que ari_hash_m[i]>>8, que es equivalente a la detección de si s=c<<8 también es mayor que ari_hash_m[i].

[000211] Como resumen de lo expuesto, una vez calculado el estado del contexto (lo que se puede lograr, por ejemplo, empleando el algoritmo “arith_get_context(c,i,N)” de acuerdo con la Fig 5c o el algoritmo “arith_get_context(c,i)” de acuerdo con la Fig 5d, se decodifica plano de 2 bits más significativos usando el algoritmo “arith_decode” (que se describe más adelante) evocado con la tabla de frecuencias cumulativas apropiada correspondiente al modelo de probabilidades correspondiente al estado del contexto. La correspondencia es realizada por la función “arith_get_pk()”, por ejemplo, la función “arith_get_pk()” que ha sido descripta con referencia a la fig 5f.

11.6 Decodificación aritmética

11.6.1 Decodificación aritmética Usando el algoritmo de acuerdo con la Fig 5g

[000212] A continuación se describe en forma detallada la funcionalidad de la función “arith_decode()” con referencia a la fig 5g.

[000213] Se debe tener en cuenta que la función “arith_decode()” utiliza la función auxiliar “arith_first_symbol (void)”, que devuelve TRUE, si es el primer símbolo de la secuencia y FALSE de lo contrario. La función “arith_decode()” también utiliza la función auxiliar “arith_get_next_bit(void)”, que obtiene y proporciona el siguiente bit del flujo de bits.

[000214] Además, la función “arith_decode()” utiliza las variables globales “low”, “high” y “valor”. Asimismo, la función “arith_decode()” recibe, como variable de entrada, la variable “cum_freq[]”, que apunta a una primera anotación o elemento (que tiene el índice de elemento o índice de anotación 0) de la tabla de frecuencias cumulativas o subtabla de frecuencias cumulativas seleccionada. Más aun, la función “arith_decode()” utiliza la variable de entrada “cfl”, que indica la longitud de la tabla de frecuencias cumulativas o subtabla de frecuencias cumulativas seleccionada designada por la variable “cum_freq[]”.

[000215] La función “arith_decode()” comprende, como primer paso, una inicialización de variables 570a, que se ejecuta si la función auxiliar “arith_first_symbol()” indica que se está decodificando el primer símbolo de una secuencia de símbolos. La inicialización de valor 550a inicializa la variable “valor” dependiendo de una pluralidad de, por ejemplo, 16 bits, que se obtienen del flujo de bits usando la función auxiliar “arith_get_next_bit”, de manera tal que la variable “valor” asume el valor representado por dichos bits. Además, la variable “low” se inicializa para tomar el valor de 0, y la variable “high” se inicializa tomando el valor de 65535.

[000216] En un segundo paso 570b, se fija en un valor la variable “rango”, valor que es superior en 1 a la diferencia entre los valores de las variables “high” y “low”. La variable “cum” se fija en un valor que representa una posición relativa del valor de la variable “valor” entre el valor de la variable “low” y el valor de la variable “high”. En consecuencia, la variable “cum” toma, por ejemplo, un valor de entre 0 y 2^{16} dependiendo del valor de la variable “valor”.

[000217] El puntero p se inicializa en un valor menor en 1 que la dirección de inicio de la tabla de frecuencias cumulativas seleccionada.

[000218] El algoritmo “arith_decode()” comprende asimismo una búsqueda iterativa de tablas de frecuencias cumulativas 570c. La búsqueda iterativa de tablas de frecuencias cumulativas se repite hasta que la variable cfl sea menor o igual a 1. En la búsqueda iterativa de tablas de frecuencias cumulativas 570c, la variable puntero q se fija en un valor que es igual a la suma del valor actual de la variable puntero p y la mitad del valor de la variable “cfl”. Si el valor del elemento *q de la tabla de frecuencias cumulativas seleccionada, elemento que es señalado por la variable puntero q, es mayor que el valor de la variable “cum”, la variable puntero p se fija en el valor de la variable puntero q, y la variable “cfl” se incrementa. Por último, la variable “cfl” se desplaza un bit a la derecha, dividiendo así, efectivamente, el valor de la variable “cfl” por 2 y omitiendo la porción del módulo.

[000219] En consecuencia, la búsqueda iterativa de tablas de frecuencias cumulativas 570c compara efectivamente el valor de la variable “cum” con una pluralidad de elementos de la tabla de frecuencias cumulativas seleccionada a fin de identificar un intervalo dentro de la tabla de frecuencias cumulativas seleccionada, que está limitado por los elementos de la tabla de frecuencias cumulativas, por lo que el valor cum se halla dentro del intervalo identificado.

En consecuencia, los elementos de la tabla de frecuencias cumulativa seleccionada define intervalos, donde un valor símbolo respectivo está asociado a cada uno de los intervalos de la tabla de frecuencias cumulativas seleccionada. Además, los anchos de los intervalos entre dos valores adyacentes de la tabla de frecuencias cumulativas definen las probabilidades de los símbolos asociados a dichos intervalos, por lo que la tabla de frecuencias cumulativas seleccionada en su totalidad define una distribución de probabilidades de los símbolos diferentes (o valores de símbolo). Más adelante se describen los detalles con respecto a las tablas de frecuencias cumulativas con referencia a la Fig. 23.

[000220] Haciendo referencia, una vez más, a la Fig. 5g, se deriva el valor de símbolo del valor de la variable puntero p, donde el valor del símbolo se deriva de la manera ilustrada en el número de referencia 570d. Por consiguiente, se evalúa la diferencia entre el valor de la variable puntero p y la dirección de partida “cum_freq” a fin de obtener el valor del símbolo, que está representado por la variable “símbolo”.

[000221] El algoritmo “arith_decode” comprende asimismo una adaptación 570e de las variables “high” y “low”. Si el valor del símbolo representado por la variable “símbolo” es diferente de 0, la variable “high” se actualiza, como se ilustra en el número de referencia 570e. Asimismo, el valor de la variable “low” se actualiza, como aparece en el número de referencia 570e. La variable “high” se fija en un valor que está determinado por el valor de la variable “low”, la variable “rango” y el elemento que tiene el índice “symbol –1” de la tabla de frecuencias cumulativas seleccionada. La variable “low” se incrementa, donde la magnitud del aumento está determinada por la variable “rango” y el elemento de la tabla de frecuencias cumulativas seleccionada que tiene el índice “symbol” (símbolo). En consecuencia, la diferencia entre los valores de las variables “low” y “high” se ajusta dependiendo de la diferencia numérica entre dos elementos adyacentes de la tabla de frecuencias cumulativas seleccionada.

[000222] En consecuencia, si se detecta un valor de símbolo con baja probabilidad, se reduce el intervalo entre los valores de las variables “low” y “high” a 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” se fija en un valor comparativamente grande. Una vez más, el ancho del intervalo entre los valores de la variable “low” y “high” depende del símbolo detectado y de las correspondientes entradas de la tabla de frecuencias cumulativas.

[000223] El algoritmo “arith_decode()” comprende asimismo la renormalización de un intervalo 570f, en la cual el intervalo determinado en el paso 570e se desplaza y escala de manera iterativa hasta alcanzar la condición de “interrupción”. En la renormalización del intervalo 570f, se ejecuta una operación selectiva de desplazamiento descendente 570fa. Si la variable “high” es menor que 32768, no se hace nada y la renormalización de intervalos continúa con una operación aumento de tamaño del intervalo 570fb. Sin embargo, si la variable “high” no es menor que 32768 y la variable “low” es superior o igual a 32768, las variables “valores”, “low” y “high” se reducen en 32768, por lo que un intervalo definido por las variables “low” y “high” se desplaza en dirección descendente y por lo que el valor de la variable “valor” también se desplaza hacia abajo. Sin embargo, si se halla que el valor de la variable “high” no es menor que 32768, y que la variable “low” no es superior o igual a 32768, y que la variable “low” es mayor o igual a 16384 y que la variable “high” es menor que 49152, las variables “valor”, “low” y “high” se reducen en 16384, de esa manera desplazando hacia abajo el intervalo entre los valores de las variables “high” y “low” y también el valor de la variable “valor”. Sin embargo, si no se cumple ninguna de las condiciones mencionadas, se aborta la renormalización del intervalo.

[000224] Sin embargo, si se cumple alguna de las condiciones antes mencionadas, que son evaluadas en el paso 570fa, se ejecuta la operación de aumento de intervalo 570fb. En la operación de aumento de intervalo 570fb, se duplica el valor de la variable “low”. Además, el valor de la variable “high” se duplica y el resultado de la duplicación se incrementa en 1. Asimismo, se duplica el valor de la variable “valor” (desplazado un bit a la izquierda) y se utiliza un bit del flujo de bits, que es obtenido por la función auxiliar “arith_get_next_bit” como bit menos significativo. En consecuencia, el tamaño del intervalo entre los valores de las variables “low” y “high” es aproximadamente el doble y la precisión de la variable “valor” se incrementa utilizando un nuevo bit del flujo de bits. Como se mencionara anteriormente, los pasos 570fa y 570fb se repiten hasta alcanzar la condición de “interrupción”, es decir, hasta que el intervalo entre los valores de las variables “low” y “high” sea suficientemente grande.

[000225] Con respecto a la funcionalidad del algoritmo “arith_decode()”, se debe tener en cuenta que el intervalo entre los valores de las variables “low” y “high” se reduce en el paso 570e dependiendo de dos elementos adyacentes de la tabla de frecuencias cumulativas a los que hace referencia la variable “cum_freq”. Si un intervalo entre dos valores adyacentes de la tabla de frecuencias cumulativas seleccionada es pequeño, es decir, si los valores adyacentes están comparativamente cerca, el intervalo entre los valores de las variables “low” y “high”, que se obtienen en el paso 570e, ha de ser comparativamente pequeño. Por el contrario, si dos elementos adyacentes de la tabla de frecuencias cumulativas están más separados, el intervalo entre los valores de las variables “low” y “high”, que se obtienen en el paso 570e, es comparativamente mayor.

[000226] En consecuencia, si el intervalo entre los valores de las variables “low” y “high”, que se obtiene en el paso 570e, es comparativamente pequeño, un gran número de pasos de renormalización de intervalo se ejecuta para reescalar el intervalo a un tamaño “suficiente” (por lo que no se cumple ninguna de las condiciones de la evaluación

de condiciones 570fa). En consecuencia, se utiliza un número comparativamente grande de bits del flujo de bits para incrementar la precisión de la variable “valor”. Si, por el contrario, el tamaño del intervalo obtenido en el paso 570e es comparativamente grande, sólo se requiere un número más pequeño de repeticiones de los pasos de normalización de intervalos 570fa y 570fb para renormalizar el intervalo entre los valores de las variables “low” y “high” a un tamaño “suficiente”. En consecuencia, sólo se utiliza un número comparativamente pequeño de bits del flujo de bits para aumentar la precisión de la variable “valor” y para preparar la decodificación de un símbolo siguiente.

[000227] Como resumen de lo expuesto, si se decodifica un símbolo que comprende una probabilidad comparativamente alta, y al cual está asociado un gran intervalo de gran tamaño por los elementos de la tabla de frecuencias cumulativas seleccionada, sólo se lee un número comparativamente pequeño de bits del flujo de bits para dar lugar a la decodificación de un símbolo subsiguiente. Por el contrario, si se decodifica un símbolo que comprende una probabilidad comparativamente pequeño y al cual está asociado un pequeño intervalo por los elementos de la tabla de frecuencias cumulativas seleccionada, se toma un número comparativamente grande de bits del flujo de bits para preparar la decodificación del siguiente símbolo.

[000228] En consecuencia, los elementos de las tablas de frecuencias cumulativas reflejan las probabilidades de los diferentes símbolos y también reflejan el número de bits necesarios para decodificar una secuencia de símbolos. La variación de la tabla de frecuencias cumulativas dependiendo de un contexto, es decir, dependiendo de símbolos previamente decodificados (o valores espectrales), por ejemplo, mediante la selección de diferentes tablas de frecuencias cumulativas dependiendo del contexto, se pueden aprovechar las dependencias estocásticas entre los diferentes símbolos, lo que da lugar a una codificación específica eficiente en velocidad de transmisión de bits de los símbolos siguientes (o adyacentes).

[000229] Como resumen de lo expuesto, la función “arith_decode()”, que ha sido descripta con referencia a la Fig. 5g, es evocada con la tabla de frecuencias cumulativas “arith_cf_m[pki][]”, correspondiente al índice “pki” devuelto por la función “arith_get_pk()” para determinar el valor de plano de bits más significativo m (que se puede fijar en el valor del símbolo representado por la variable de retorno “símbolo”).

[000230] Como resumen de lo expuesto, el decodificador aritmético es una implementación de números enteros que utiliza el método de generación de etiquetas con escalamiento. Por más detalles, se hace referencia al texto “Introduction to Data Compression” de K. Sayood, Tercera Edición, 2006, Elsevier Inc.

[000231] El código de programa de computación de acuerdo con la Fig. 5g describe el algoritmo utilizado de acuerdo con una realización de la invención.

11.6.2 Decodificación aritmética que utiliza el algoritmo de acuerdo con la Figs. 5h y 5i

[000232] Las Fig. 5h y 5i ilustran una representación de pseudo código de programa de otra realización del algoritmo “arith_decode()”, que se puede utilizar como alternativa del algoritmo “arith_decode” descripto con referencia a la Fig. 5g.

[000233] Se debe tener en cuenta que ambos algoritmos de acuerdo con la Fig. 5g y las Figs. 5h y 5i se pueden utilizar en el algoritmo “values_decode()” de acuerdo con la Fig. 3.

[000234] Para resumir, se decodifica el valor m empleando la función “arith_decode()” evocada con la tabla de frecuencias cumulativas “arith_cf_m[pki][]” donde “pki” corresponde al índice retornado por la función “arith_get_pk()”. El codificador aritmético (o decodificador) es una implementación de números enteros que utiliza el método de generación de etiquetas con escalamiento. Por más detalles, se hace referencia al texto “Introduction to Data Compression” de K. Sayood, Tercera Edición, 2006, Elsevier Inc.. El código de programa de computación de acuerdo con la Fig. 5h y 5i describe el algoritmo utilizado.

11.7 Mecanismo de Escape

[000235] A continuación se describe brevemente el mecanismo de escape que se utiliza en el algoritmo de decodificación “values_decode()” de acuerdo con la Fig. 3.

[000236] Cuando el valor decodificado m (que se presenta como valor de retorno de la función “arith_decode()”) es el símbolo de escape “ARITH_ESCAPE”, las variables “lev” y “esc_nb” se incrementan en 1, y se decodifica otro valor m. En este caso, se evoca una vez más la función “arith_get_pk()” con el valor “c+ esc_nb<<17 como argumento de entrada, donde la variable “esc_nb” describe el número de símbolos de escape previamente decodificados por el mismo 2–tuplo y limitado a 7.

[000237] Para resumir, en caso de identificarse un símbolo de escape, se presume que el valor del plano de bits más significativo m comprende una ponderación numérica incrementada. Más aun, se repite la decodificación numérica actual, donde se utiliza un valor numérico de contexto actual modificado “c+ esc_nb<<17” como variable

de entrada a la función “arith_get_pk()”. En consecuencia, por lo general se obtiene un valor índice de regla de mapeo “pki” diferente en diferentes iteraciones del subalgoritmo 312ba.

11.8 Mecanismo de Parada Aritmética

[000238] A continuación se describe el mecanismo de parada aritmética. El mecanismo de parada aritmética da lugar a la reducción del número de bits necesarios en caso de que la porción de frecuencia más elevada sea cuantizada por completo a 0 en un codificador de audio.

[000239] En una realización, se puede implementar un mecanismo de parada aritmética de la siguiente manera: Una vez que el valor m no es el símbolo de escape, “ARITH_ESCAPE”, el decodificador verifica si el m sucesivo forma un símbolo “ARITH_ESCAPE”. Si la condición “esc_nb >0&&m==0” es verdadera, se detecta el símbolo “ARITH_STOP” y el proceso de decodificación finaliza. En ese caso, el decodificador salta directamente a la función “arith_finish()” que se describe a continuación. La condición significa que el resto del cuadro está compuesto por 0 valores.

11.9 Decodificación de planos de bits menos significativos

[000240] A continuación se describe la decodificación de uno o más planos de bits menos significativos. La decodificación del plano de bits menos significativos, se ejecuta, por ejemplo, en el paso 312d ilustrado en la Fig. 3. Por otro lado, sin embargo, se pueden utilizar los algoritmos ilustrados en las Fig. 5j y 5n.

11.9.1 Decodificación de planos de bits menos significativos de acuerdo con la Fig. 5j

[000241] Tomando ahora como referencia la Fig. 5j, se puede apreciar que los valores de las variables a y b se derivan del valor m. Por ejemplo, la representación numérica del valor m se desplaza 2 bits a la derecha para obtener la representación numérica de la variable b. Más aun, se obtiene el valor de la variable a restando una versión desplazada en bits del valor de la variable b, desplazada en bits 2 bits a la izquierda, del valor de la variable m.

[000242] A continuación se repite una decodificación aritmética de los valores de planos de bit menos significativo, donde el número de repeticiones está determinado por el valor de la variable “lev”. Se obtiene un valor de plano de bits menos significativos r empleando la función “arith_decode”, donde se utiliza una tabla de frecuencias cumulativas adaptada a la decodificación de planos de bit menos significativos (tabla de frecuencias cumulativas “arith_cf_r”). Un bit menos significativo (que tiene una ponderación numérica de 1) de la variable r describe un plano de bits menos significativos del valor espectral representado por la variable a, y un bit que tiene una ponderación numérica de 2 de la variable r describe un bit menos significativo del valor espectral representado por la variable b. En consecuencia, la variable a se actualiza desplazando la variable a 1 bit a la izquierda y sumando el bit con la ponderación numérica de 1 de la variable r como bit menos significativo. De modo similar, la variable b se actualiza desplazando la variable b un bit a la izquierda y sumando el bit con la ponderación numérica de 2 de la variable r.

[000243] En consecuencia, los dos bits transportadores de información más significativos de las variables a,b están determinados por el valor de plano de bits más significativos m, y dichos uno o más bits menos significativos (si los hay) de los valores a y b están determinados por los valores r de uno o más plano de bits menos significativos.

[000244] Como resumen de lo expuesto, si no se alcanza el símbolo “ARITH_STOP”, se decodifica el resto de los planos de bits, en caso de haberlos, correspondientes al presente 2–tuplo. El resto de los planos de bits son decodificados desde el nivel más significativo al menos significativo mediante la evocación de la función “arith_decode()” lev número de veces con la tabla de frecuencias cumulativas “arith_cf_r[]”. Los planos de bits decodificados 4 permiten la refinación del valor m antes decodificado de acuerdo con el algoritmo, un pseudo código de programa del cual está expuesto en la Fig. 5j.

11.9.2 Decodificación de la Banda de Bits Menos Significativos de acuerdo con la Fig. 5n

[000245] Por otro lado, sin embargo, también se puede utilizar el algoritmo cuya representación de pseudo código de programa está expuesta en la Fig. 5n para la decodificación de planos de bits menos significativos. En ese caso, si no se alcanza el símbolo “ARITH_STOP”, se decodifica el resto de los planos de bits, en caso de existir, correspondiente al 2–tuplo actual. El resto de los planos de bits es decodificado desde el nivel más significativo al menos significativo mediante la evocación de “lev" veces "arith_decode()” con la tabla de frecuencias cumulativas “arith_cf_r()”. Los planos de bits decodificados r dan lugar a la refinación del valor antes decodificado m de acuerdo con el algoritmo expuesto en la Fig. 5n.

11.10 Actualización del contexto

11.10.1 Actualización del contexto de acuerdo con las Figs. 5k, 5l, y 5m

[000246] A continuación se describen las operaciones utilizadas para completar la decodificación del tuplo de valores espectrales, haciendo referencia a la Figs. 5k y 5l. Más aun, se describe una operación que se utiliza para completar la decodificación de una serie de tuplos de valores espectrales asociados a una porción actual (por ejemplo, un cuadro actual) de un contenido de audio.

[000247] Tomando ahora como referencia la Fig. 5k, se puede apreciar que el elemento que tiene el índice de elemento 2*i de la matriz “x_ac_dec[]” se fija para que sea igual a a, y que el elemento con el índice de elemento “2*i+1” de la matriz “x_ac_dec[]” se fija igual a b después de la decodificación de los bits menos significativos 312d. En otras palabras, en el punto posterior a la decodificación de bits menos significativos 312d, se decodifica por completo el valor sin signos del 2–tuplo (a,b). Se lo guarda en el elemento (por ejemplo la matriz “x_ac_dec[]”) que contiene los coeficientes espectrales de acuerdo con el algoritmo expuesto en la Fig. 5k.

[000248] A continuación, el contexto “q” también se actualiza con respecto al siguiente 2–tuplo. Se debe tener en cuenta que esta actualización del contexto también debe ser ejecutada para el último 2–tuplo. Esta actualización del contexto es ejecutada por la función “arith_update_context()”, una representación de pseudo código de programa de la cual está expuesta en la Fig. 5l.

[000249] Tomando ahora como referencia la Fig. 5l, se puede apreciar que la función “arith_update_context(i,a,b)” recibe, como variables de entrada, coeficientes espectrales cuantizados sin signos decodificados (o valores espectrales) a, b del 2–tuplo. Además, la función “arith_update_context” también recibe, como variable de entrada, un índice i (por ejemplo, un índice de frecuencia) del coeficiente espectral cuantizado a decodificar. En otras palabras, la variable de entrada i puede ser, por ejemplo, un índice del tuplo de valores espectrales, los valores absolutos del cual están definidos por las variables de entrada a, b. Como se puede apreciar, el elemento “q[1][i]” de la matriz “q[][]” se puede fijar en un valor que es igual a a+b+1. Además, el valor del elemento “q[1][i]” de la matriz “q[][]” se puede limitar a un valor hexadecimal de “0xF”. Por consiguiente, se obtiene el elemento “q[1][i]” de la matriz “q[][]” mediante el cómputo de una suma de valores absolutos del tuplo actualmente decodificado {a,b} de valores espectrales que tienen el índice de frecuencia i y la suma de 1 al resultado de dicha suma.

[000250] Cabe señalar aquí que el elemento “q[1][i]” de la matriz “q[][]” puede ser considerada valor de subregión de contexto, puesto que describe una subregión del contexto que se utiliza para una decodificación subsiguiente de valores espectrales adicionales (o tuplos de valores espectrales).

[000251] Cabe señalar aquí que la sumatoria de los valores absolutos a y b de los dos valores espectrales actualmente decodificados (versiones firmadas de los cuales están almacenadas en los elementos “x_ac_dec[2*i]” y “x_ac_dec[2*i+1]” de la matriz “x_ac_dec[]”), se pueden considerar como cómputo de una norma (por ej. una norma L1) de los valores espectrales decodificados.

[000252] Se ha encontrado que valores de subregión de contexto (es decir, elementos de la matriz “q[][]”), que describe una norma de un vector formado por una pluralidad de valores espectrales anteriormente decodificados son particularmente importantes y eficientes en memoria. Se ha encontrado que ese tipo de norma, que se computa sobre la base de una pluralidad de valores espectrales anteriormente decodificados, comprende información significativa de contexto en forma compacta. Se ha encontrado que la firma de los valores espectrales por lo general no es particularmente relevante para la elección del contexto. También se ha encontrado que la formación de una normal a través de una pluralidad de valores espectrales anteriormente decodificados típicamente mantiene la información más importante, incluso aunque se descarten algunos detalles. Más aun, se ha encontrado que una limitación del valor numérico de contexto actual a un valor máximo por lo general no da lugar a una pérdida grave de información. Por el contrario, se ha encontrado que es más eficiente utilizar el mismo estado del contexto para valores espectrales significativos que son mayores que un valor umbral predeterminado. Por consiguiente, la limitación de los valores de subregión de contexto trae aparejada una mejora de la eficiencia de memoria. Más aun, se ha encontrado que la limitación de los valores de subregión de contexto a un cierto valor máximo da lugar a una actualización especialmente simple e informáticamente eficiente del valor numérico de contexto actual, que ha sido descripto, por ejemplo, con referencia a las Figs. 5c y 5d. Al limitar los valores de subregión de contexto a un valor comparativamente menor (por ej. a un valor de 15), se puede representar de manera eficiente el estado del contexto que se basa en una pluralidad de valores de subregión de contexto, lo que ya ha sido descripto con referencia a las Figs. 5c y 5d.

[000253] Más aun, se ha encontrado que una limitación de los valores de subregión de contexto a valores de entre 1 y 15, trae aparejado un compromiso particularmente favorable entre precisión y eficiencia de memoria, puesto que 4 bits son suficientes para almacenar ese valor de subregión de contexto.

[000254] Sin embargo, se debe tener en cuenta que, en algunas realizaciones adicionales, un valor de subregión de contexto se puede basar sólo en un único valor espectral decodificado. En ese caso, se puede omitir opcionalmente la formación de una norma.

[000255] El siguiente 2–tuplo del cuadro es decodificado una vez completada la función “arith_update_context” mediante el incremento de i en 1 y repitiendo el mismo proceso antes descripto, a partir de la función “arith_get_context()”.

[000256] Cuando se decodifican lg/2 2–tuplos dentro del cuadro, o cuando aparece el símbolo de parada de acuerdo con “ARITH_ESCAPE”, el proceso de decodificación de la amplitud espectral finaliza y comienza la decodificación de los signos.

[000257] Los detalles con respecto a la decodificación de los signos han sido descriptos con referencia a la Fig. 3, donde la decodificación de los signos está indicada por el número de referencia 314.

[000258] Una vez decodificados todos los coeficientes espectrales cuantizados sin signo, se agrega el signo correspondiente. Por cada valor cuantizado no nulo de “x_ac_dec” se lee un bit. Si el valor del bit leído es igual a 0, el valor cuantizado es positivo, no se hace nada y el valor con signo es igual al valor sin signo precedentemente decodificado. De lo contrario (es decir, si el valor del bit leído es igual a 1), el coeficiente decodificado (o valor espectral) es negativo y se toma el complemento de dos del valor sin signo. Los bits de signo se leen desde las frecuencias más bajas a las más altas. Para aportar detalles, se hace referencia a las Figs. 3 y a las explicaciones con respecto a la decodificación de signos 314.

[000259] La decodificación concluye mediante la evocación de la función “arith_finish()”. Se fija el resto de los coeficientes espectrales en 0. Se actualizan de modo correspondiente los respectivos estados del contexto.

[000260] Para ver detalles, se hace referencia a Fig. 5m, que ilustra una representación de pseudo código de programa de la función “arith_finish()”. Como se puede apreciar, la función “arith_finish()” recibe una variable de entrada lg que describe los coeficientes espectrales cuantizados decodificados. Preferentemente, la variable de entrada lg de la función “arith_finish” describe un número de coeficientes espectrales decodificados en realidad, dejando coeficientes espectrales de lado, a los cuales se ha asignado un valor de 0 en respuesta a la detección de un símbolo “ARITH_STOP”. Una variable de entrada N de la función “arith_finish” describe una longitud de ventana de una ventana actual (es decir, una ventana asociada a la porción actual del contenido de audio). Por lo general, un número de valores espectrales asociado a una ventana de longitud N es igual a N/2 y un número de 2–tuplos de valores espectrales asociados a una ventana de longitud de ventana N es igual a N/4.

[000261] La función “arith_finish” también recibe, como valor de entrada, un vector “x_ac_dec” de valores espectrales decodificados o por lo menos una referencia a dicho vector de valores espectrales decodificados.

[000262] La función “arith_finish” está configurada para fijar los elementos de la matriz (o vector) “x_ac_dec”, por los cuales no se han decodificado valores espectrales debido a la presencia de una condición de parada aritmética, en 0. Más aun, la función “arith_finish” fija valores de subregión de contexto “q[1][i]”, que están asociados a valores espectrales respecto de los cuales no se ha decodificado ningún valor debido a la presencia de una condición de parada aritmética, en un valor predeterminado de 1. El valor predeterminado de 1 corresponde a un tuplo de los valores espectrales donde ambos valores espectrales son iguales a 0.

[000263] En consecuencia, la función “arith_finish()” permite actualizar toda la matriz (o vector) “x_ac_dec[]” de valores espectrales y también toda la matriz de valores de subregión de contexto “q[1][i]”, aun en presencia de una condición de parada aritmética.

11.10.2 Actualización del contexto de acuerdo con las Figs. 5o y 5p

[000264] A continuación se describe otra realización de la actualización del contexto con referencia a las Figs. 5o y 5p. En el punto en que el valor sin signo del 2–tuplo (a,b) ya está completamente decodificado, el contexto q se actualiza luego respecto del siguiente 2–tuplo. La actualización también se ejecuta si el 2–tuplo presente es el último 2–tuplo. Ambas actualizaciones son realizadas por la función “arith_update_context()”, una representación de pseudo código de programa de la cual está expuesta en la Fig. 5o.

[000265] A continuación se decodifica el siguiente 2–tuplo del cuadro mediante el incremento de i en 1 y la evocación de la función arith_decode(). Si los lg/2 2–tuplos ya han sido decodificados con el cuadro, so si aparece el símbolo de parada “ARITH_STOP”, se evoca la función “arith_finish()”. El contexto se guarda y almacena en la matriz (o vector) “qs” para el cuadro siguiente. Se ilustra un pseudo código de programa de la función “arith_save_context()” en la Fig. 5p.

[000266] Una vez decodificada la totalidad de los coeficientes espectrales cuantizados sin signo, se agrega el signo. por cada valor no cuantizado de “qdec”, se lee un bit. Si el valor del bit leído es igual a 0, el valor cuantizado es positivo, no se hace nada y el valor con signo es igual al valor sin signo anteriormente decodificado. DE lo contrario, el coeficiente decodificado es negativo y se toma el complemento de dos del valor sin signo. Los bits con signo se leen desde las frecuencias bajas a las altas.

11.11 Síntesis del Proceso de decodificación

[000267] A continuación se resume brevemente el proceso de decodificación. Para ver detalles, se hace referencia a la explicación que antecede y también a las Figs. 3, 4, 5a, 5c, 5e, 5g, 5j, 5k, 5l, y 5m. Los coeficientes espectrales cuantizados “x_ac_dec[]” son decodificados sin ruido a partir del coeficiente de la menor frecuencia y progresando hasta el coeficiente de la frecuencia más alta. Son decodificados por grupos de dos coeficientes sucesivos a,b reunidos en lo que se denomina un 2–tuplo (a,b).

[000268] Los coeficientes decodificados “x_ac_dec[]” correspondientes al dominio de la frecuencia (es decir, correspondientes a un modo en el dominio de la frecuencia) son almacenados luego en la matriz “x_ac_quant[g][win][sfb][bin]”. El orden de transmisión de las palabras clave de codificación sin ruido es tal que cuando se las decodifica en el orden recibido y almacenado en la matriz, “bin” es el índice que se incrementa más rápidamente y “g” es el índice que se incrementa con mayor lentitud. Dentro de una palabra clave, se almacena el orden de decodificación es a, luego b. Los coeficientes decodificados “x_ac_dec[]” correspondientes a la “TCX” (es decir, para una decodificación de audio utilizando excitación codificada por transformadas) (por ejemplo, directamente) en la matriz “x_tcx_invquant[win][bin]” y el orden de la transmisión de palabras clave de codificación sin ruido es tal que cuando son decodificadas en el orden recibido y almacenado en la matriz, “bin” es el índice que se incrementa más rápidamente y “win” es el índice que se incrementa con mayor lentitud. Dentro de una palabra clave, el orden de decodificación es a, luego b.

[000269] En primer lugar, la bandera “arith_reset_flag” determina si el contexto debe ser reiniciado. Si la bandera es cierta, eso se considera en la función “arith_map_context”.

[000270] El proceso de decodificación comienza con una fase de inicialización en la cual el vector de elementos de contexto “q” se actualiza copiando y mapeando los elementos de contexto del cuadro anterior almacenado en “q[1][]” en “q[0][]”. Los elementos de contexto dentro de “q” se almacenan en 4 bits por 2–tuplo. Para ver detalles, se hace referencia al pseudo código de programa de la Fig. 5a.

[000271] El decodificador sin ruido da salida a 2–tuplos de coeficientes espectrales cuantizados sin signo. Al principio, se calcula el estado c del contexto sobre la base de los coeficientes espectrales anteriormente decodificados que rodean al 2–tuplo a decodificar. Por lo tanto, el estado se actualiza en incrementos utilizando el estado del contexto del último 2–tuplo decodificado tomando en cuenta sólo dos nuevos 2–tuplos. El estado se decodifica en 17 bits y es devuelto por la función “arith_get_context”. Una representación de pseudo código de programa de la función establecida “arith_get_context” está expuesta en la Fig. 5c.

[000272] El estado del contexto c determina la tabla de frecuencias cumulativas usada para decodificar el plano de 2 bits más significativos m. Al mapeo de c al correspondiente índice de tabla de frecuencias cumulativas “pki” es ejecutado por la función “arith_get_pk()”. Una representación de pseudo código de programa de la función “arith_get_pk()” está expuesta en la Fig. 5e.

[000273] El valor m es decodificado utilizando la función “arith_decode()” evocada con la tabla de frecuencias cumulativas, “arith_cf_m[pki][]”, donde “pki” corresponde al índice devuelto por “arith_get_pk()”. El codificador aritmético (y decodificador) es una implementación de números enteros que utiliza un método de generación de etiquetas con escalamiento. El pseudo código de programa de acuerdo con la Fig. 5g describe el algoritmo utilizado.

[000274] Cuando el valor decodificado m es el símbolo de escape “ARITH_ESCAPE”, las variables “lev” y “esc_nb” se incrementan 1 y se decodifica otro valor m. En este caso, la función “get_pk()” es evocada una vez más con el valor “c+ esc_nb<<17” como argumento de entrada, donde “esc_nb” es el número de símbolos de escape previamente decodificado para el mismo 2–tuplo y limitado a 7.

[000275] Una vez que el valor m ya no es el símbolo de escape “ARITH_ESCAPE”, el decodificador verifica si el m sucesivo forma un símbolo “ARITH_STOP”. Si la condición “(esc_nb>0&&m==0)” es cierta, se detecta el símbolo “ARITH_STOP” y el proceso de decodificación finaliza. El decodificador salta directamente a la decodificación de signos que se describe más adelante. La condición significa que el resto del cuadro está compuesto por valores 0.

[000276] Si no aparece el símbolo “ARITH_STOP”, luego se decodifica el resto de los planos de bits, en caso de haberlos, luego se decodifica el resto de los planos de bits, en caso de haberlos, correspondientes al presente 2–tuplo. El resto de los planos de bits son decodificados desde el nivel más significativo al menos significativo, evocando “arith_decode()" lev número de veces con la tabla de frecuencias cumulativas “arith_cf_r[]”. Los planos de bits decodificados r permiten la refinación del valor m antes decodificado de acuerdo con el algoritmo cuyo pseudo código de programa se ilustra en la Fig. 5j. En este punto, el valor sin signo del 2–tuplo (a,b) está completamente decodificado. Se lo guarda en el elemento que contiene los coeficientes espectrales de acuerdo con el algoritmo, una representación de pseudo código de programa del cual está ilustrada en la Fig. 5k.

[000277] El contexto “q” también se actualiza respecto del siguiente 2–tuplo. Se debe tener en cuenta que esta actualización del contexto también debe ser ejecutada para el último 2–tuplo. Esta actualización del contexto es

ejecutada por la función “arith_update_context()”, una representación de pseudo código de programa de la cual está expuesta en la Fig. 5l.

[000278] A continuación se decodifica el siguiente 2–tuplo del cuadro incrementándolo i en 1 y repitiendo el mismo proceso antes descripto a partir de la función “arith_get_context()”. Cuando se decodifican lg/2 2–tuplos dentro del cuadro, o cuando aparece el símbolo de parada “ARITH_STOP”, el proceso de decodificación de la amplitud espectral finaliza y comienza la decodificación de los signos.

[000279] La decodificación termina mediante la evocación de la función “arith_finish()”. El resto de los coeficientes espectrales se fijan en 0. Se actualizan correspondientemente los respectivos estados del contexto. Se ilustra una representación de pseudo código de programa de la función “arith_finish” en la Fig. 5m.

[000280] Una vez decodificados todos los coeficientes espectrales cuantizados sin signo, se agrega el signo correspondiente. Por cada valor cuantizado no nulo de “x_ac_dec”, se lee un bit. Si el valor del bit leído es igual a 0, el valor cuantizado es positivo, no se hace nada y el valor con signo es igual al valor sin signo anteriormente decodificado. De lo contrario, el coeficiente decodificado es negativo y se toma el complemento de dos del valor sin signo. Los bits con signo se leen desde las frecuencias bajas a las altas.

11.12 Leyendas

[000281] La Fig. 5q ilustra una leyenda de las definiciones que se relacionan con los algoritmos de acuerdo con las Figs. 5a, 5c, 5e, 5f, 5g, 5j, 5k, 5l, y 5m.

[000282] La Fig. 5r ilustra una leyenda de las definiciones que se relacionan con los algoritmos de acuerdo con las Figs. 5b, 5d, 5f, 5h, 5i, 5n, 5o, y 5p.

12. Tablas de mapeo

[000283] En una realización de acuerdo con la invención, se utilizan tablas “ari_lookup_m”, “ari_hash_m”, y “ari_cf_m” particularmente ventajosas para la ejecución de la función “arith_get_pk()” de acuerdo con la Fig. 5e o la Fig. 5f, y para la ejecución de la función “arith_decode()” que fuera explicada con referencia a la Figs. 5g, 5h y 5i. Sin embargo, se debe tener en cuenta que se pueden utilizar tablas diferentes en algunas realizaciones de acuerdo con la invención.

12.1 Tabla “ari_hash_m[600]” de acuerdo con la Fig. 22

[000284] En la tabla de la Fig. 22 se ilustra el contenido de una implementación particularmente ventajosa de la tabla “ari_hash_m”, que es utilizada por la función “arith_get_pk”, una primera realización de la cual fue descripta con referencia a la Fig. 5e, y una segunda realización de la cual fue descripta con referencia a la Fig. 5. Se debe tener en cuenta que la tabla de la Fig. 22 enumera los 600 elementos de la tabla (o matriz) “ari_hash_m[600]”. También se debe tener presente que la representación de tabla de la Fig. 22 ilustra los elementos en el orden de los índices de elementos, por lo que el primer valor “0x000000100UL” corresponde a un elemento de la tabla “ari_hash_m[0]” que tiene el índice de elemento (o índice de tabla) 0, y por lo que el último valor “0x7ffffffff4fUL” corresponde a un elemento de la tabla “ari_hash_m[599]” que tiene el índice de elemento o tabla 599. Se debe notar asimismo que “0x” indica que los elementos de tabla de la tabla “ari_hash_m[]” están representados en formato hexadecimal. Más aun, cabe señalar aquí que el sufijo “UL” indica que los elementos de tabla de la tabla “ari_hash_m[]”están representados en forma de valores enteros “largos” sin signo (con una precisión de 32 bits).

[000285] Por añadidura, se debe tener en cuenta que los elementos de tabla de la tabla “ari_hash_m[]” de acuerdo con la Fig. 22 están dispuestos en orden numérico a fin de permitir la ejecución de la búsqueda de tablas 506b, 508b, 510b de la función “arith_get_pk()”.

[000286] También se debe apreciar que los 24 bits más significativos de los elementos de tabla de la tabla “ari_hash_m” representan ciertos valores de estado significativos, en tanto que los 8 bits menos significativos representan valores índice de regla de mapeo “pki”. Por consiguiente, los elementos de la tabla “ari_hash_m[]” describen un mapeo de “coincidencia directa” de un valor de contexto con un valor índice de regla de mapeo “pki”. Sin embargo, los 24 bits más altos de los elementos de la tabla “ari_hash_m[]” representan, al mismo tiempo, los límites de intervalo de los intervalos de valores numéricos de contexto a los cuales está asociado el mismo valor índice de regla de mapeo. Ya se han descripto anteriormente los detalles referidos a este concepto.

12.2 Tabla “ari_lookup_m” de acuerdo con la Fig. 21

[000287] En la tabla de la Fig. 21 se ilustra el contenido de una realización particularmente ventajosa de la tabla “ari_lookup_m”. Cabe señalar aquí que la tabla de la Fig. 21 enumera los elementos de la tabla “ari_lookup_m”. Los elementos son tomados como referencia por un índice de elemento de número entero unidimensional (también denominado “índice de elemento” o “índice de matriz” o “índice de tabla”) al que se designa, por ejemplo, “i_max” o

"i_min". Se debe tener en cuenta que la tabla "ari_lookup_m", que comprende un total de 600 elementos, es muy adecuada para usar en la función "arith_get_pk" de acuerdo con la Fig. 5e o la Fig. 5f. También se debe tener presente que la tabla "ari_lookup_m" de acuerdo con la Fig. 21 está adaptada para cooperar con la tabla "ari_hash_m" de acuerdo con la Fig. 22.

[000288] Se debe tener en cuenta que los elementos de la tabla "ari_lookup_m[600]" están enumerados en orden ascendente del índice de tabla "i" (por ej. "i_min" o "i_max") entre 0 y 599. El término "0x" indica que los elementos de tabla están descriptos en formato hexadecimal. En consecuencia, el primer elemento de la tabla "0x02" corresponde al elemento de la tabla "ari_lookup_m[0]" que tiene el índice de tabla 0 y el último elemento de la tabla "0x5E" corresponde al elemento de la tabla "ari_lookup_m[599]" que tiene el índice de tabla 599.

[000289] También se debe tener presente que los elementos de la tabla "ari_lookup_m[]" están asociados a intervalos definidos por elementos adyacentes de la tabla "arith_hash_m[]". Por consiguiente, los elementos de la tabla "ari_lookup_m" describen valores índice de regla de mapeo asociados a intervalos de valores numéricos de contexto, donde los intervalos están definidos por los elementos de la tabla "arith_hash_m".

12.3. Tabla "ari_cf_m[96][17]" de acuerdo con la Fig. 23

[000290] La Fig. 23 ilustra una serie de 96 tablas de frecuencias cumulativas (o subtablas) "ari_cf_m[pki][17]", 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 de plano de bits más significativos. La tabla seleccionada de las 96 tablas de frecuencias cumulativas (o subtablas) ilustradas en la Fig. 23 asume la función de la tabla "cum_freq[]" en la ejecución de la función "arith_decode()".

[000291] Como se puede apreciar en la Fig. 23, cada subbloque representa una tabla de frecuencias cumulativas con 17 elementos. Por ejemplo, un primer subbloque 2310 representa los 17 elementos de una tabla de frecuencias cumulativas correspondiente a "pki=0". Un segundo subbloque 2312 representa los 17 elementos de una tabla de frecuencias cumulativas correspondiente a "pki=1". Por último, un 96º subbloque 2396 representa los 17 elementos de una tabla de frecuencias cumulativas correspondiente a "pki=95". Por consiguiente, la Fig. 23 representa, en efecto, 96 tablas de frecuencias cumulativas (o subtablas) diferentes correspondientes a "pki=0" a "pki=95", donde cada una de las 96 tablas de frecuencias cumulativas está representada por un subbloque (indicado entre llaves) y donde cada una de dichas tablas de frecuencias cumulativas comprende 17 elementos.

[000292] Dentro de un subbloque (por ej. un subbloque 2310 o 2312, o un subbloque 2396), un primer valor describe un primer elemento de una tabla de frecuencias cumulativas (que tiene un índice de matriz o índice de tabla de 0), y un último valor describe el último elemento de una tabla de frecuencias cumulativas (que tiene un índice de matriz o índice de tabla de 16).

[000293] En consecuencia, cada subbloque 2310, 2312, 2396 de la tabla representación de la Fig. 23 representa los elementos de una tabla de frecuencias cumulativas para utilizarse en la función "arith_decode" de acuerdo con la Fig. 5g, o de acuerdo con las Figs. 5h y 5i. La variable de entrada "cum_freq[]" de la función "arith_decode" describe cuál de las 96 tablas de frecuencias cumulativas (representada por subbloques individuales de 17 elementos de la tabla "arith_cf_m") se debe utilizar para la decodificación de los coeficientes espectrales actuales.

12.4 Tabla "ari_cf_r[]" de acuerdo con la Fig. 24

[000294] La Fig. 24 ilustra el contenido de la tabla "ari_cf_r[]".

[000295] Los cuatro elementos de dicha tabla están expuestos en la Fig. 24. Sin embargo, se debe tener en cuenta que la tabla "ari_cf_r" eventualmente puede ser diferente en otras realizaciones.

13. Evaluación de la Eficiencia y Ventajas

[000296] Las realizaciones de acuerdo con la invención son funciones actualizadas (o algoritmos) y una serie de tablas actualizada, de acuerdo con lo descripto en los párrafos anteriores, para obtener una compensación mejorada entre complejidad informática, requerimientos de memoria y eficiencia de codificación.

[000297] En términos generales, las realizaciones de acuerdo con la invención crean una codificación espectral sin ruido mejorada. Las realizaciones de acuerdo con la presente invención describen una mejora de la codificación espectral sin ruido en USAC (codificación unificada de voz y audio).

[000298] Las realizaciones de acuerdo con la invención dan origen a una propuesta actualizada a la CE sobre codificación espectral sin ruido mejorada de coeficientes espectrales, basada en los esquemas presentados en los documentos presentada en MPEG m16912 y m17002. Ambas propuestas fueron evaluadas, se eliminaron los inconvenientes potenciales y se combinaron los puntos fuertes.

[000299] Como en m16912 y m17002, la propuesta resultante se basa en el esquema de codificación aritmética basado en e contexto original como en el borrador de trabajo 5 de USAC (el proyecto de norma sobre codificación unificada de voz y audio), aunque puede reducir significativamente los requerimientos de memoria (memoria de acceso aleatorio (RAM) y memoria de sólo lectura (ROM) sin aumentar la complejidad informática, manteniendo al mismo tiempo la eficiencia de codificación. Además, se ha comprobado que es posible la transcodificación sin pérdidas de flujos de bits de acuerdo con el borrador de trabajo 3 del proyecto de Norma USAC y de acuerdo con el borrador de trabajo 5 del proyecto de Norma USAC. Las realizaciones de acuerdo con la invención apuntan a reemplazar el esquema de codificación espectral sin ruido empleado en el borrador de trabajo 5 del proyecto de Norma USAC.

[000300] El esquema de codificación aritmética aquí descripto se basa en el esquema del modelo de referencia 0 (RM0) o del borrador de trabajo 5 (WD) del Proyecto de Norma USAC. Los coeficientes espectrales en la frecuencia o el tiempo modelan un contexto. Este contexto es utilizado para a selección de tablas de frecuencias cumulativas para el codificador aritmético. En comparación con el borrador de trabajo 5 (WD), el modelado del contexto mejora aun más y las tablas que contienen probabilidades de símbolos son reestructuradas. El número de diferentes modelos de probabilidades se incrementó de 32 a 96.

[000301] Las realizaciones de acuerdo con la invención reducen los tamaños de las tablas (demanda de ROM de datos) a 1518 palabras de 32 bits o 6072 bytes de longitud (WD 5: 16, 894,5 palabras o 67.578 bytes). La demanda de RAM estática se reduce de 666 palabras (2.664 bytes) a 72 palabras (288 bytes) por canal codificador núcleo. Al mismo tiempo, conserva por completo la eficiencia de codificación e incluso puede obtener una ganancia de aproximadamente 1,29 a 1,95% en comparación con la velocidad de datos total en los 9 puntos operativos. Todos los flujos de bits del borrador de trabajo 3 y del borrador de trabajo 5odos los flujos de bits del borrador de trabajo 3 y del borrador de trabajo 5 se pueden transcodificar sin pérdidas y sin afectar las limitaciones de reserva de bits.

[000302] A continuación se presenta una breve explicación de los conceptos de codificación de acuerdo con el borrador de trabajo 5 del Proyecto de Norma USAC a fin de facilitar la comprensión de las ventajas del concepto descripto en la presente. A continuación se describen algunas realizaciones preferidas de acuerdo con la invención.

[000303] En el Borrador de trabajo de USAC 5, se utiliza un esquema de codificación aritmética basada en el contexto para la codificación sin ruido de coeficientes espectrales cuantizados. Como contexto se utilizan los coeficientes decodificados, que son anteriores en frecuencia y en tiempo. En el borrador de trabajo 5, se utiliza un número máximo de 16 coeficientes espectrales como contexto, 12 de los cuales son anteriores en el tiempo. Además, los coeficientes espectrales utilizados para el contexto y a decodificar, están agrupados en forma de 4–tuplos (es decir, 4 coeficientes espectrales cercanos en frecuencia; ver la Fig. 14a). El contexto se reduce y mapea sobre una tabla de frecuencias cumulativas, que luego es utilizada para decodificar el siguiente 4–tuplo de coeficientes espectrales.

[000304] Para el esquema de codificación sin ruido del borrador de trabajo 5 completo, se requiere una demanda de memoria (memoria de sólo lectura (ROM) de 16894,5 palabras (67578 bytes). Además, se requieren 666 palabras (2664 bytes) de RAM estática por canal codificador núcleo para almacenar los estados para el siguiente cuadro. La representación de tablas de la Fig. 14b describe las tablas utilizadas en el esquema de codificación aritmética USAC WD4.

[000305] Cabe señalar aquí que, en lo que respecta a la codificación sin ruido, los borradores de trabajo 4 y 5 del proyecto de norma USAC son iguales. Ambos utilizan el mismo codificador sin ruido.

[000306] Se estima que la demanda total de memoria del decodificador USAC WD5 completo es de 37000 palabras (148000 bytes) para la ROM de datos sin código de programa y de 10000 a 17000 palabras en el caso de la RAM estática. Se puede apreciar claramente que las tablas del codificador sin ruido consumen aproximadamente 45% de la demanda total de ROM de datos. La tabla individual más grande ya consume 4096 palabras (16384 bytes).

[000307] Se ha encontrado que tanto el tamaño de la combinación de todas las tablas como las tablas individuales exceden los tamaños de cache en los procesadores de punto fijo utilizados en los dispositivos portátiles del consumidor, que se encuentran en un rango típico de 8 a 32 Kbyte (por ej. ARM9e, TI C64XX, etc). Esto significa que es probable que no se pueda almacenar la serie de tablas en la RAM de datos rápidos, que permite un rápido acceso aleatorio a los datos. Esto hace que todo el proceso de decodificación se torne más lento.

[000308] Más aun, se ha encontrado que la tecnología de codificación de audio actual apropiada tal como HE–AAC ha demostrado ser implementable en la mayoría de los dispositivos móviles. HE–AAC utiliza un esquema de codificación por entropía de Huffman con un tamaño de tabla de 995 palabras. Para ver detalles, se hace referencia a ISO/IEC JTC1/SC29/WG11 N2005, MPEG98, febrero de 1998, San José, “Revised Report on Complexity of MPEG–2 AAC2”.

[000309] En la 90ª Reunión de MPEG, en los documentos presentados en MPEG m16912 y m17002, se presentaron dos propuestas que apuntaban a reducir la demanda de memoria y a mejorar la eficiencia de

codificación del esquema de codificación sin ruido. Mediante el análisis de ambos propuestas, se pudo llegar a las siguientes conclusiones.

- Es posible una significativa reducción de la demanda de memoria mediante la reducción de la dimensión de las palabras código. Como se ilustra en el documento presentado en MPEG m17002, al reducir la dimensión de 4–tuplos a 1–tuplos, se podría reducir la demanda de memoria de 16984,5 a 900 palabras sin comprometer la eficiencia de codificación y

- Se podría eliminar la redundancia adicional aplicando un libro de códigos de distribución de probabilidades no uniforme para la codificación LSB, en lugar de utilizar la distribución uniforme de probabilidades.

[000310] En el curso de estas evaluaciones, se identificó que pasando de un esquema de codificación de 4–tuplos a 1–tuplos se obtuvo un impacto significativo sobre la complejidad informática: una reducción de la dimensión de codificación aumenta en el mismo factor el número de símbolos a codificar. Esto significa que la reducción de 4–tuplos a 1–tuplos que las operaciones necesitan para determinar el contexto, acceder a las la tablas hash y decodificar el símbolo se deben ejecutar cuatro veces más que antes. Junto con el algoritmo más sofisticado para la determinación del contexto, esto llevó a un incremento de la complejidad informática en un factor de 2,5 o x.xxPCU.

[000311] A continuación se describe brevemente el nuevo esquema propuesto de acuerdo con las realizaciones de la presente invención.

[000312] Para superar el problema de las huellas de memoria y la complejidad informática, se propuso un esquema de codificación sin ruido mejorado para reemplazar el esquema del borrador de trabajo 5 (WD5). El foco principal del desarrollo fue puesto en la reducción de la demanda de memoria, manteniendo a la vez la eficiencia de compresión y sin aumentar la complejidad informática. Más específicamente, el objetivo consistió en obtener un buen (o incluso el mejor) compromiso en el espacio de complejidad multidimensional de la eficiencia de compresión, complejidad y demanda de memoria.

[000313] La nueva propuesta de esquema de codificación toma la principal característica del codificador sin ruido WD5, es decir la adaptación del contexto. El contexto se deriva utilizando coeficientes espectrales previamente decodificados, que provienen en WD5 tanto del cuadro anterior como el actual (donde se puede considerar que un cuadro es una porción del contenido de audio). Sin embargo, ahora se codifican los coeficientes espectrales combinando dos coeficientes entre sí para formar un 2–tuplo. Otra diferencia está en el hecho de que los coeficientes espectrales ahora se dividen en tres partes: el signo, los bits más significativos o los bits muy significativos (MSBs) y los bits menos significativos o los bits menos significativos de todos (LSBs). El signo se codifica de manera independiente de la magnitud que también se divide en dos partes, los bits más significativos (o bits más significativos) y el resto de los bits (o bits menos significativos), si existieran. Los 2–tuplos para los cuales la magnitud de los dos elementos es menor o igual a 3 son codificados directamente por la codificación de MSBs. De lo contrario, se transmite primero una palabra clave de escape para señalar cualquier plano de bit adicional. En la versión básica, la información faltante, los LSBs y el signo, son codificados en ambos casos utilizando una distribución uniforme de probabilidades. Por otro lado, se puede utilizar una distribución de probabilidades diferente.

[000314] La reducción del tamaño de las tablas es posible de todas maneras, ya que:

- sólo se necesita almacenar las probabilidades correspondientes a 17 símbolos: {[0;+3], [0;+3]}+ESC symbol;

- no hay necesidad de almacenar una tabla de agrupamientos (egroups, dgroups, dgvectors);

- se podría reducir el tamaño de la tabla hash con una apropiada formación de series.

[000315] A continuación se describen algunos detalles con respecto a la codificación por MSBs. Como ya se mencionara, una de las principales diferencias entre WD5 del Proyecto de Norma USAC, una propuesta presentada en la 90ª Reunión de MPEG y la presente propuesta es la dimensión de los símbolos. En WD5 del Proyecto de Norma USAC, se consideraron 4–tuplos para la generación del contexto y la codificación sin ruido. En una propuesta presentada en la 90ª Reunión de MPEG se utilizaron en su lugar 1–tuplos para reducir los requerimientos de ROM. En el curso del desarrollo, se encontró que 2–tuplos constituían el mejor compromiso para reducir la demanda de ROM, sin aumentar la complejidad informática. En lugar de considerar cuatro 4–tuplos para la innovación del contexto, ahora se consideran cuatro 2–tuplos. Como se ilustra en la Fig. 15a, tres 2–tuplos provienen del cuadro anterior (también denominado porción anterior del contenido de audio) y uno proviene del cuadro actual (también denominado porción actual del contenido de audio).

[000316] La reducción del tamaño de las tablas se debe a tres factores principales. En primer lugar, sólo se necesita almacenar las probabilidades correspondiente a 17 símbolos (es decir, {[0;+3], [0;+3]} + ESC symbol). Las tablas de agrupamientos (es decir, egroups, dgroups, y dgvectors) ya no son necesarias. Por último, se redujo el tamaño de la tabla hash ejecutando la serie de instrucciones apropiada.

[000317] Si bien se redujo la dimensión de cuatro a dos, la complejidad se mantuvo en el mismo rango que el WD5 del Proyecto de Norma USAC. Eso se logró simplificando tanto la generación del contexto como el acceso a las tablas hash.

[000318] Las diferentes simplificaciones y optimizaciones se realizaron de manera tal que no se afectó la eficiencia de codificación, y hasta se mejoró levemente. Eso se obtuvo principalmente aumentando el número de modelos de probabilidades de 32 a 96.

[000319] A continuación se describen algunos detalles con respecto a la codificación por LSBs. Los LSBs son codificados con una distribución de probabilidades uniforme en algunas realizaciones. En comparación con WD5 del Proyecto de Norma USAC, ahora se consideran los LSBs dentro de 2–tuplos en lugar de 4–tuplos.

[000320] En lo sucesivo se explican algunos detalles con respecto a la codificación de signos. El signo se codifica sin utilizar el codificador núcleo aritmético por razones de reducción de la complejidad. El signo se transmite en 1 bit solamente cuando la magnitud correspondiente no es nula. 0 significa un valor positivo y 1 representa un valor negativo.

[000321] A continuación se explican algunos detalles con respecto a la demanda de memoria. El nuevo esquema propuesto exhibe una demanda total de ROM de a lo sumo 1522,5 palabras nuevas (6090 bytes). Para ver detalles, se hace referencia a la tabla de la Fig. 15b, que describe las tablas utilizadas en el esquema de codificación propuesto. En comparación con la demanda de ROM del esquema de codificación sin ruido de WD 5 del Proyecto de Norma USAC, la demanda de ROM se reduce por lo menos 15462 palabras (61848 bytes). Ahora termina en el mismo orden de magnitud que el requerimiento de memoria necesario para el decodificador AAC Huffman de HE–AAC (995 palabras o 3980 bytes). Para ver detalles, se hace referencia a ISO/IEC JTC1/SC29/WG11 N2005, MPEG98, febrero de 1998, San José, “Revised Report on Complexity of MPEG–2 AAC2”, y también a la Fig. 16a. Esto reduce la demanda de ROM total del codificador sin ruido más de 92% y un decodificador USAC completo de aproximadamente 37000 palabras a aproximadamente 21500 palabras, o más de 41%. Para ver detalles, se hace referencia una vez más a las Figs. 16a y 16b, donde la Fig. 16a ilustra la demanda de ROM de un esquema de codificación sin ruido propuesto y el de un esquema de codificación sin ruido de acuerdo con WD4 del Proyecto de Norma USAC, y donde la Fig. 16b ilustra una demanda total de ROM del decodificador USAC de acuerdo con el esquema propuesto y de acuerdo con WD4 del Proyecto de Norma USAC.

[000322] Más adelante, también se reduce la cantidad de información necesaria para la derivación del contexto en el cuadro siguiente (ROM estática). En WD5 del Proyecto de Norma USAC, se debe almacenar la serie completa de coeficientes (un máximo de 1152 coeficientes) con una resolución típicamente de 16 bits adicionales a un índice de grupo por 4–tuplo de 10 bits de resolución, lo que suma hasta 666 palabras (2664 bytes) por canal del codificador de núcleo (decodificador USAC WD4 completo: aproximadamente 10000 a 17000 palabras). El nuevo esquema reduce la información persistente a sólo 2 bits por coeficiente espectral, lo que suma hasta 72 palabras (288 byte) en total por canal codificador de núcleo. La demanda de memoria estática se puede reducir en 594 palabras (2376 byte).

[000323] A continuación se describen algunos detalles con respecto al posible aumento de la eficiencia de codificación. Se comparó la eficiencia de codificación de las realizaciones de acuerdo con la nueva propuesta con los flujos de bits de calidad de referencia de acuerdo con el borrador de trabajo 3 (WD3) y WD5 del Proyecto de Norma USAC. La comparación se realizó por medio de un transcodificador, sobre la base de un decodificador de software de referencia. Por detalles con respecto a dicha comparación de la codificación sin ruido de acuerdo con WD3 o WD5 del Proyecto de Norma USAC y el esquema de codificación propuesto, se hace referencia a la Fig. 17, que ilustra una representación esquemática de una disposición de ensayo para una comparación de la codificación sin ruido de WD3/5 con el esquema de codificación propuesto.

[000324] Asimismo, se comparó la demanda de memoria de las realizaciones de acuerdo con la invención con realizaciones de acuerdo con el WD3 (o WD5) del Proyecto de Norma USAC.

[000325] La eficiencia de codificación no sólo se mantiene, sino que se incrementa ligeramente. Para más detalles, se hace referencia a la tabla de la Fig. 18, que ilustra una representación tabulada de las velocidades de transmisión de bits promedio por el codificador aritmético WD3 (o un codificador de audio USAC utilizando un codificador aritmético WD3) y un codificador de audio (por ej. el codificador de audio USAC) de acuerdo con una realización de la invención.

[000326] Se pueden encontrar detalles sobre las velocidades promedio de transmisión de bits por modo operativo en la tabla de la Fig. 18.

[000327] Más aun, la Fig. 19 ilustra una representación tabulada de los niveles de reserva de bits mínimo y máximo correspondientes al codificador aritmético WD3 (o un codificador de audio que utiliza el codificador aritmético WD3) y un codificador de audio de acuerdo con una realización de la presente invención.

[000328] A continuación se describen algunos detalles respecto de la complejidad informática. La reducción de la dimensionalidad de la codificación aritmética habitualmente lleva a un aumento de la complejidad informática. De hecho, la reducción de la dimensión en un factor de dos hace que el codificador aritmético evoque dos veces las rutinas.

[000329] Sin embargo, se ha encontrado que este aumento de la complejidad puede estar limitado por varias optimizaciones introducidas en el nuevo esquema de codificación propuesto de acuerdo con las realizaciones de la presente invención. La generación de contexto se simplificó considerablemente en algunas realizaciones de acuerdo con la invención. Por cada 2–tuplo, el contexto se puede actualizar en incrementos desde el último contexto generado. Las probabilidades son almacenadas ahora en 14 bits en lugar de 16 bits, lo que evita las operaciones de 64 bits durante el proceso de decodificación. Más aun, se optimizó considerablemente el mapeo de modelos de probabilidades en algunas realizaciones de acuerdo con la invención. El peor de los casos se redujo drásticamente y se limita a 10 iteraciones en lugar de 95.

[000330] Como resultado de ello, la complejidad informática del esquema de codificación sin ruido propuesto se mantuvo en el mismo rango que en WD 5. Se realizó un cálculo estimativo con “lápiz y papel” por diferentes versiones de la codificación sin ruido y esto está consignado en la tabla de la Fig. 20. Esto demuestra que el nuevo esquema de codificación es sólo aproximadamente 13% menos complejo que el codificador aritmético WD5.

[000331] Como resumen de lo expuesto, se puede apreciar que realizaciones de acuerdo con la presente invención ofrecen un compromiso particularmente ventajoso entre complejidad informática, demanda de memoria y eficiencia de codificación.

14. Sintaxis del flujo de bits

14.1 Cargas útiles del Codificador Espectral Sin Ruido

[000332] A continuación se describen algunos detalles con respecto a las cargas útiles del codificador espectral sin ruido. En algunas realizaciones, hay una pluralidad de modalidades de codificación diferentes tales como, por ejemplo, un modo de codificación denominado “en el dominio de predicción lineal” y un modo de codificación “en el dominio de la frecuencia”. En el modo de codificación en el dominio de predicción lineal se realiza un modelado de ruido sobre la base de un análisis de predicción lineal de la señal de audio y se codifica una señal de ruido modelado en el dominio de la frecuencia. En el modo de codificación en el dominio de la frecuencia se ejecuta el modelado de ruido sobre la base de un análisis psicoacústico y se codifica una versión con modelado de ruido del contenido de audio en el dominio de la frecuencia.

[000333] Los coeficientes espectrales tanto de la señal codificada en el “dominio de predicción lineal” como de la señal codificada en “el dominio de la frecuencia” son cuantizados y escalados y luego codificados sin ruido por una codificación aritmética dependiente adaptativamente del contexto. Los coeficientes cuantizados se agrupan entre sí en 2–tuplos antes de ser transmitidos desde la frecuencia más baja a la frecuencia más alta. CAda 2–tuplo se divide en un signo s, el plano de 2 bits más significativos m y uno o más planos de bits menos significativos restantes r (si los hubiera). El valor m se codifica de acuerdo con un contexto definido por los coeficientes espectrales cercanos. En otras palabras, m es codificado de acuerdo con la cercanía de los coeficientes. El resto de los planos de bits menos significativos r es codificado por entropía sin considerar el contexto. Por medio de m y r, se puede reconstruir la amplitud de estos coeficientes espectrales del lado del decodificador. En el caso de todos los símbolos no nulos, los signos s son codificados fuera del codificador aritmético usando 1 bit. En otras palabras, los valores m y r forman los símbolos del codificador aritmético. Por último, los signos s, son codificados fuera del codificador aritmético usando 1 bit por coeficiente cuantizado no nulo.

[000334] Se describe aquí un procedimiento de codificación aritmética detallado.

14.2 Elementos de Sintaxis

[000335] A continuación se describe la sintaxis del flujo de bits de un flujo de bits que acarrea la información espectral aritméticamente codificada haciendo referencia a las Figs. 6a a 6j.

[000336] La Fig. 6a ilustra una representación de la sintaxis del bloque de datos bruto denominado USAC (“usac_raw_data_block()”).

[000337] El bloque de datos bruto USAC comprende uno o más elementos de canal único (“single_channel_element()”) y/o uno o más elementos de pares de canales (“channel_pair_element()”).

[000338] Tomando ahora como referencia la Fig. 6b, se describe la sintaxis de un elemento de canal único. El elemento de canal único comprende un flujo de canales en el dominio de la predicción lineal (“lpd_channel_stream ()”) o un flujo de canales en el dominio de la frecuencia (“fd_channel_stream ()”) dependiendo del modo de núcleo.

[000339] La Fig. 6c ilustra una representación de la sintaxis de un elemento de par de canales. Un elemento de par de canales comprende información del modo del núcleo (“core_mode0”, “core_mode1”). Además, el elemento de par de canales puede comprender una información de configuración “ics_info()”. Además, dependiendo de la información del modo del núcleo, el elemento de par de canales comprende un flujo de canales en el dominio de la predicción lineal o un flujo de canales en el dominio de la frecuencia asociado al primero de los canales, y el elemento de par de canales comprende asimismo un flujo de canales en el dominio de la predicción lineal o un flujo de canales en el dominio de la frecuencia asociado al segundo de los canales.

[000340] La información de configuración “ics_info()”, una representación de la sintaxis de la cual está expuesta en la Fig. 6d, comprende una pluralidad de ítems diferentes de información de configuración, que no son de particular relevancia para la presente invención.

[000341] Un flujo de canales en el dominio de la frecuencia (“fd_channel_stream ()”), una representación de la sintaxis del cual está expuesta en la Fig. 6e, comprende una información de ganancia (“global_gain”) y una información de configuración (“ics_info ()”). Además, el flujo de canales en el dominio de la frecuencia comprende datos del factor de escalamiento (“scale_factor_data ()”), que describe factores de escala utilizados para el escalamiento de valores espectrales de diferentes bandas de factores de escalamiento y que son aplicados, por ejemplo, por el escalador 150 y el reescalador 240. El flujo de canales en el dominio de la frecuencia comprende asimismo datos espectrales aritméticamente codificados (“ac_spectral_data ()”), que representan valor valores espectrales aritméticamente codificados.

[000342] Los datos espectrales aritméticamente codificados (“ac_spectral_data()”), una representación de la sintaxis de los cuales está expuesta en la Fig. 6f, comprenden una bandera de reinicio aritmético (“arith_reset_flag”), que se utiliza para reiniciar selectivamente el contexto, como se describiera anteriormente. Además, los datos espectrales aritméticamente codificados comprenden una pluralidad de bloques de datos aritméticos (“arith_data”), que acarrean los datos espectrales aritméticamente codificados. La estructura de los bloques de datos espectrales aritméticamente codificados depende del número de bandas de frecuencia (representado por la variable “num_bands”) y también del estado de la bandera de reinicio aritmético, como se describe con.

[000343] A continuación se describe la estructura de los bloques de datos espectrales aritméticamente codificados tomando como referencia la Fig. 6g, que ilustra una representación de la sintaxis de dichos bloques de datos espectrales aritméticamente codificados. La representación de datos dentro del bloque de datos espectrales aritméticamente codificados depende del número lg de valores espectrales a codificar, del estado de la bandera de reinicio aritmético y también del contexto, es decir, los valores espectrales previamente codificados.

[000344] El contexto para la codificación de la serie actual (por ej., 2–tuplo) de valores espectrales se determina de acuerdo con el algoritmo de determinación del contexto indicado con el número de referencia 660. Los detalles con respecto al algoritmo de determinación del contexto han sido explicados anteriormente. Haciendo referencia a las Figs. 5a y 5b. El bloque de datos aritméticamente codificados comprende lg/2 series de palabras clave, donde cada serie de palabras clave representa una pluralidad (por ej., un 2–tuplo) de valores espectrales. Una serie de palabras clave comprende una palabra clave aritmética “acod_m[pki][m]” que representa un valor de plano de bits más significativo m del tuplo de valores espectrales que usa entre 1 y 20 bits. Además, la serie de palabras clave comprende una o más palabras clave “acod_r[r]” si el tuplo de valores espectrales requiere más planos de bits que el plano de bits más significativos para una representación correcta. La palabra clave “acod_r[r]” representa un plano de bits menos significativos que utiliza entre 1 y 14 bits.

[000345] Sin embargo, si es necesario uno o más planos de bits menos significativos (además del plano de bits más significativos) para una correcta representación de los valores espectrales, esto se indica utilizando una o más palabras clave 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 más planos de bits menos significativos adicionales) se necesitan. Si se necesita uno o más plano de bits menos significativos, esto está indicado por una o más palabras clave de escape aritmético “acod_m[pki][ARITH_ESCAPE]”, que son codificadas de acuerdo con una tabla de frecuencias cumulativas seleccionada actualmente, un índice de tablas de frecuencias cumulativas está dado por la variable “pki”. Además, el contexto se adapta, como se puede apreciar en los números de referencia 664, 662, si una o más palabras de escape aritmético están incluidas en el flujo de bits. Después de dichas una o más palabras clave de escape aritmético, se incluye una palabra clave aritmética “acod_m[pki][m]” en el flujo de bits, como se indica en el número de referencia 663, donde “pki” designa el índice modelo de probabilidades válido actualmente (tomando en cuenta la adaptación del contexto causada por la inclusión de las palabras clave de escape aritmético) y donde m designa el valor del plano de bits más significativos del valor espectral a codificar o decodificado (donde m es diferente de la palabra clave "ARITH_ESCAPE").

[000346] Como se señalara anteriormente, la presencia de algún plano de bits menos significativos da lugar a la presencia de una o más palabras clave “acod_r[r]”, cada una de las cuales representa 1 bit de un plano de bits menos significativos de un primer valor espectral y cada una de las cuales también representa 1 bit de un plano de bits menos significativos de un segundo valor espectral. La una o más palabras clave “acod_r[r]” son codificadas de acuerdo con una correspondiente tabla de frecuencias cumulativas, que por ejemplo puede ser constante e independiente del contexto. Sin embargo, son posibles diferentes mecanismos para la selección de la tabla de frecuencias cumulativas para la decodificación de dichas una o más palabras clave “acod_r[r]”.

[000347] Además, se debe tener en cuenta que el contexto se actualiza después de la codificación de cada tuplo de valores espectrales, como se indica con el número de referencia 668, por lo que el contexto es típicamente diferente en el caso de la codificación y decodificación de dos tuplos de valores espectrales subsiguientes.

[000348] La Fig. 6i ilustra una leyenda de definiciones y elementos de ayuda que definen la sintaxis del bloque de datos aritméticamente codificados.

[000349] Más aun, en la Fig. 6h se ilustra una sintaxis alternativa de los datos aritméticos “arith_data()”, con una correspondiente leyenda de definiciones y elementos de ayuda expuesta en la Fig. 6j.

[000350] Como resumen de lo expuesto, se ha descripto el formato de flujo de bits, que puede producir el codificador de audio 100 y que puede ser evaluado por el decodificador de audio 200. El flujo de bits de los valores espectrales aritméticamente codificados es codificado de tal manera que se ajuste al algoritmo de decodificación antes descripto.

[000351] Además, se debe notar generalmente que la codificación es la operación inversa al a decodificación, por lo que en general se puede presumir que el codificador ejecuta una búsqueda de tablas utilizando las tablas antes descriptas, que es aproximadamente inversa a la búsqueda de tablas realizada por el decodificador. En general, se puede afirmar que una persona con capacitación en la técnica que conoce el algoritmo de decodificación y/o la sintaxis del flujo de bits deseada ha de poder diseñar fácilmente un codificador aritmético, que presenta los datos definidos en la sintaxis del flujo de bits y necesarios para un decodificador aritmético.

[000352] Más aun, se debe tener en cuenta que los mecanismos para determinar el valor numérico del contexto actual y para derivar un valor índice de regla de mapeo puede ser idéntico en un codificador de audio y un decodificador de audio, puesto que, por lo general, es conveniente que el decodificador de audio utilice el mismo contexto que el codificador de audio, de tal manera que la decodificación se adapte a la codificación.

15. Alternativas de Implementación

[000353] Si bien se han descripto algunos aspectos en el contexto de un aparato, es claro que estos aspectos también representan una descripción del método correspondiente, donde un bloque o dispositivo correspondiente a un paso del método o una característica de un paso del método. De manera análoga, los aspectos descriptos en el contexto del paso de un método también representan una descripción de un correspondiente bloque o ítem o característica de un correspondiente aparato. Algunos o todos los pasos del método pueden ser ejecutados (o utilizando) un aparato de hardware, como por ejemplo un microprocesador, una computadora programable o un circuito electrónico. En algunas realizaciones, uno o más de los pasos más importantes del método pueden ser ejecutados por ese tipo de aparato.

[000354] La señal de audio codificada puede ser almacenada en un medio de almacenamiento digital o puede ser transmitida en un medio de transmisión tal como un medio de transmisión inalámbrico o un medio de transmisión conectado por cables tal como la Internet.

[000355] Dependiendo de ciertos requisitos de implementación, las realizaciones de la invención se pueden implementar en hardware o en software. La implementación se puede ejecutar empleando un medio de almacenamiento digital, por ejemplo un disco blando, un DVD, un Blue–Ray, un CD, una ROM, una PROM, una EPROM, una EEPROM o una memoria FLASH, que tiene almacenadas en la misma señales control legibles electrónicamente, que cooperan (o tienen capacidad para cooperar) con un sistema de computación programable de tal manera que se ejecute el método respectivo. Por lo tanto, el medio de almacenamiento digital puede ser legible por computadora.

[000356] Algunas implementaciones pueden comprender un transportador de datos que comprende señales de control legibles electrónicamente, con capacidad para cooperar con un sistema de computación programable de tal manera que se ejecute uno de los métodos descriptos en la presente.

[000357] En general, las realizaciones de la presente invención pueden ser implementadas en forma de producto programa de computación con un código de programa, donde el código de programa cumple la función de ejecutar uno de los métodos al ejecutarse el programa de computación en una computadora. El código de programa puede ser almacenado, por ejemplo, en un portador legible por una máquina.

[000358] Otras implementaciones comprenden el programa de computación para ejecutar uno de los métodos aquí descriptos, almacenado en un portador legible por una máquina.

[000359] En otras palabras, una realización del método de la invención consiste, por lo tanto, en un programa de computación que consta de un código de programa para realizar uno de los métodos aquí descriptos al ejecutarse el programa de computación en una computadora..

[000360] Otra implementación consiste, por lo tanto, en un portador de datos (o medio de almacenamiento digital, o medio legible por computadora) que comprende, grabado en el mismo, el programa de computación para ejecutar uno de los métodos aquí descriptos. El portador de datos, medio de almacenamiento digital, o medio legible por computadora son por lo general tangibles y no transitorios.

[000361] Otra implementación es, por lo tanto, un flujo de bits de datos o una secuencia de señales que representa el programa de computación para ejecutar uno de los métodos aquí descriptos. El flujo de datos o la secuencia de señales puede estar configurado, por ejemplo, para ser transferido a través de una conexión de comunicación de datos, por ejemplo por la Internet.

[000362] Otra implementación comprende un medio de procesamiento, por ejemplo una computadora, un dispositivo lógico programable, configurado o adaptado para ejecutar uno de los métodos aquí descriptos.

[000363] Otra implementación comprende una computadora en la que se ha instalado el programa de computación para ejecutar uno de los métodos aquí descriptos.

[000364] Otra implementación comprende un aparato o un sistema configurado para transferir (por ejemplo, por vía electrónica u óptica) un programa de computación para ejecutar uno de los métodos aquí descriptos a un receptor. El receptor puede ser, por ejemplo, una computadora, un dispositivo móvil, un dispositivo de memoria o similar. El aparato o sistema puede comprender, por ejemplo, un servidor de archivos para transferir el programa de computación al receptor.

[000365] En algunas implementaciones, se puede utilizar un dispositivo lógico programable (por ejemplo una matriz de puertas programables en el campo) para ejecutar algunas o todas las funcionalidades de los métodos aquí descriptos. En algunas realizaciones, una matriz de puertas programables en el campo puede cooperar con un microprocesador para ejecutar uno de los métodos aquí descriptos. Por lo general, los métodos son ejecutados preferentemente por cualquier aparato de hardware.

[000366] Las implementaciones y realizaciones precedentemente descriptas son meramente ilustrativas de los principios de la presente invención. Se entiende que las modificaciones y variaciones de las disposiciones y detalles aquí descriptos han de ser evidentes para las personas con capacitación en la técnica. Por lo tanto, sólo es intención limitarse al alcance de las siguientes reivindicaciones de patente y no a los detalles específicos presentados a manera de descripción y explicación de las realizaciones aquí presentadas.

16. Conclusiones

[000367] En conclusión, las implementaciones descritas comprenden uno o más de los siguientes aspectos, donde los aspectos pueden ser utilizados individualmente o en combinación.

a) Mecanismo de hasheo del estado del contexto

[000368] Los estados consignados en la tabla hash se consideran estados significativos y límites de grupos. Esto permite reducir significativamente el tamaño de las tablas necesarias.

b). Actualización del contexto en Incrementos

[000369] Algunas realizaciones de acuerdo con la invención comprenden una manera computacionalmente eficiente para actualizar el contexto y usan una actualización en incrementos del contexto en la cual se deriva un valor numérico de contexto actual de un valor numérico de contexto anterior.

c). Derivación del Contexto

[000370] El uso de la suma de dos valores espectrales absolutos es la asociación de un truncamiento. Es un tipo de cuantización por vectores de ganancia de los coeficientes espectrales (a diferencia de la cuantización por vectores de ganancia de forma convencional). Apunta a limitar el orden del contexto y a la vez transmitir la información más significativa de las inmediaciones.

[000371] Algunas tecnologías adicionales han sido descriptas en las solicitudes de patente previamente publicadas PCT EP2101/065725, PCT EP2010/065726, y PCT EP 2010/065727. Más aun, en algunas implementaciones, se utiliza un símbolo de parada. Por añadidura, en algunas implementaciones, sólo se tienen en cuenta los valores sin signo para el contexto.

[000372] Sin embargo, las solicitudes de patente internacional antes mencionadas no publicadas anteriormente describen aspectos que aún están en uso en algunas implementaciones.

[000373] Por ejemplo, en algunas implementaciones se utiliza una identificación de una región cero.

[000374] En consecuencia se fija una denominada “bandera de pequeño valor” (por ej., el bit 16 del valor numérico de contexto actual c).

[000375] En algunas implementaciones, se puede utilizar el cómputo de contexto dependiente de la región. Sin embargo, en otras implementaciones, se puede omitir el cómputo de contexto dependiente de la región para mantener razonablemente bajos la complejidad y el tamaño de las tablas.

[000376] Más aun, el hasheo del contexto utilizando una función de hash es un aspecto importante. El hasheo del contexto se puede basar en el concepto de dos tablas descripto en las solicitudes de patente internacional no publicadas previamente antes mencionadas. Sin embargo, en algunas implementaciones se pueden utilizar adaptaciones específicas del hasheo del contexto para aumentar la eficiencia informática. De todas maneras, in algunas implementaciones adicionales, se puede utilizar el hasheo del contexto que se describe en las solicitudes de patente internacional no publicadas previamente antes mencionadas.

[000377] Más aun, se debe tener en cuenta que el hasheo del contexto en incrementos es bastante sencillo y eficiente desde el punto de vista informático. Además, la independencia del contexto del signo de los valores que se utiliza en algunas implementaciones, contribuye a simplificar el contexto, manteniendo así la demanda de memoria razonablemente baja.

[000378] En algunas implementaciones, se utiliza una derivación del contexto que utiliza la suma de dos valores espectrales y una limitación del contexto. Estos dos aspectos se pueden combinar. Ambos apuntan a limitar el orden del contexto transportando la información más significativa de las inmediaciones.

[000379] En algunas implementaciones se utiliza una bandera de pequeño valor que es similar a la identificación de un grupo de una pluralidad de valores cero.

[000380] En algunas implementaciones, se utiliza un mecanismo de parada aritmética. El concepto es similar al uso de un símbolo "end–of–block" en JPEG, que tiene una función comparable. Sin embargo, en algunas implementaciones, el símbolo ("ARITH_STOP") no está explícitamente incluido en el codificador por entropía. En su lugar se utiliza una combinación de símbolos ya existentes, lo que no podía ocurrir anteriormente, es decir, “ESC+0”. En otras palabras, el decodificador de audio está configurado para detectar una combinación de símbolos existentes que normalmente no se utilizan para representar un valor numérico y para interpretar la aparición de dicha combinación de símbolos ya existentes como condición de parada aritmética.

[000381] Una implementación utiliza un mecanismo de hasheo de contexto por dos tablas.

[000382] Para resumir aún más, algunas implementaciones pueden comprender uno o más de los siguientes cuatro aspectos fundamentales.

- contexto extendido para detectar regiones cero o regiones de pequeña amplitud en las inmediaciones;
- hasheo del contexto;
- generación de estado del contexto: actualización en incrementos del estado del contexto y
- derivación del contexto: cuantización específica de los valores del contexto que incluye la sumatoria de las amplitudes y limitación.

[000383] Para concluir además, un aspecto de las implementaciones se basa en la actualización en incrementos del contexto. Las implementaciones comprenden un concepto eficiente para la actualización del contexto, que evita los cálculos extensos del borrador de trabajo (por ejemplo, del borrador de trabajo 5). Por el contrario, se utilizan sencillas operaciones de desplazamiento y operaciones de lógica en algunas implementaciones. La simple actualización del contexto facilita de manera significativa el cómputo del contexto.

[000384] En algunas implementaciones, el contexto es independiente del signo de los valores (por ej., los valores espectrales decodificados). Esta independencia del contexto del signo de los valores trae aparejada una complejidad reducida de la variable contexto. Este concepto se basa en el hallazgo de que la omisión del signo del contexto no causa una degradación severa de la eficiencia de codificación.

[000385] De acuerdo con una implementación, el contexto se deriva utilizando la suma de dos valores espectrales. En consecuencia, la demanda de memoria para el almacenamiento del contexto se reduce significativamente. En consecuencia, en algunos casos se puede considerar ventajoso el uso de un valor de contexto, que representa la suma de dos valores espectrales.

[000386] Además, la limitación del contexto causa una mejora significativa en algunos casos. Además de la derivación del contexto usando la suma de dos valores espectrales, los elementos de la matriz del contexto “q” se limitan a un valor máximo de “0xF” en algunas implementaciones, lo que a su vez da lugar a una limitación de la demanda de memoria. Esta limitación de los valores de la matriz del contexto “q” trae aparejadas algunas ventajas.

[000387] En algunas implementaciones se utiliza una llamada “bandera de pequeño valor”. Para obtener la variable de contexto c (que también se denomina valor numérico de contexto actual), se fija una bandera si los valores de algunos elementos “q[1][i–3]” a “q[1][i–1]” son muy pequeños. En consecuencia, el cómputo del contexto se puede ejecutar con gran eficiencia. Se puede obtener un valor de contexto especialmente significativo (por ej. el valor numérico del contexto actual).

[000388] En algunas implementaciones, se utiliza un mecanismo de parada aritmética. El mecanismo “ARITH_STOP” da lugar a una parada eficiente de la codificación o decodificación aritmética si quedan sólo valores cero. En consecuencia, la eficiencia de codificación se puede mejorar a costos moderados en términos de complejidad.

[000389] De acuerdo con una implementación, se utiliza un mecanismo de hasheo de dos tablas. El mapeo del contexto se ejecuta empleando un algoritmo de división de intervalos que evalúa la tabla “ari_hash_m” en combinación con una evaluación subsiguiente de búsqueda de tablas de la tabla “ari_lookup_m”. Este algoritmo es más eficiente que el algoritmo WD3.

[000390] A continuación se comentan otros detalles.

[000391] Cabe señalar aquí que la tablas “arith_hash_m[600]” y “arith_lookup_m[600]” son dos tablas distintas. La primera se utiliza para mapear un solo índice de contexto (por ej. el valor numérico de contexto) contra un índice de modelo de probabilidades (por ej., el valor índice de regla de mapeo) y la segunda se utiliza para mapear un grupo de contextos consecutivos, delimitados por los índices de contexto de “arith_hash_m[]”, en un solo modelo de probabilidades.

[000392] También se debe apreciar que la tabla “arith_cf_msb[96][16]” se puede utilizar como alternativa de la tabla “ari_cf_m[96][17]”, aunque las dimensiones sean ligeramente diferentes. “ari_cf_m[][]” y “ari_cf_msb[][]” se pueden referir a la misma tabla, ya que los 17os coeficientes de los modelos de probabilidades son siempre cero. En ocasiones no se toma en cuenta esto al contar el espacio requerido para almacenar las tablas.

[000393] Como resumen de lo expuesto, algunas implementaciones ofrecen una nueva codificación sin ruido propuesta (codificación o decodificación) que engendra modificaciones en el Borrador de trabajo de USAC MPEG (por ejemplo, en el Borrador de trabajo de USAC 5 MPEG). Dichas modificaciones se pueden ver en las figuras adjuntas y también en la descripción asociada.

[000394] Como comentario de cierre, se debe tener en cuenta que el prefijo "ari" y el prefijo "arith" en los nombres de las variables, matrices, funciones y demás se utilizan en forma intercambiable.

REIVINDICACIONES

1. Un decodificador de audio (200;800) para la provisión de una información de audio decodificada (212;812) sobre la base de una información de audio codificada (210;810), en donde el decodificador de audio comprende:

un decodificador aritmético (230;820) para la provisión de una pluralidad de valores espectrales decodificados (232;822) sobre la base de una representación aritméticamente codificada (222;821) de los valores espectrales comprendidos en la información de audio codificada; y

un convertidor del dominio de la frecuencia al dominio del tiempo (260;830) para la provisión de una representación de audio en el dominio del tiempo (262;812) utilizando los valores espectrales (232;822), para obtener la información de audio decodificada (212;812);

en donde el decodificador aritmético (230;820) está configurado para seleccionar una regla de mapeo (297;cum_freq[]) que describe un mapeo de un valor de código (value) de la representación aritméticamente codificada (222;821) de los valores espectrales sobre un código de símbolo (symbol) representando uno o más de los valores espectrales decodificados o por lo menos una porción de uno o más de los valores espectrales decodificados dependiendo de un estado del contexto descripto por un valor numérico de contexto actual (c); y

en donde el decodificador aritmético (230;820) está configurado para determinar el valor numérico de contexto actual (c) dependiendo de un valor numérico de contexto previo y dependiendo de una pluralidad de valores espectrales previamente decodificados,

en donde el decodificador aritmético está configurado para modificar una representación numérica del valor numérico de contexto previo, que describe un estado del contexto para la decodificación de uno o más valores espectrales previamente decodificados, dependiendo de un valor de subregión de contexto que describe una subregión de un contexto, para obtener una representación numérica de un valor numérico de contexto actual que describe un estado del contexto para la decodificación de uno o más valores espectrales a decodificar.

2. El decodificador de audio de acuerdo con la reivindicación 1, en donde el decodificador aritmético está configurado para proporcionar la representación numérica del valor numérico de contexto actual de tal manera que porciones de la representación numérica que tiene diferentes ponderaciones numéricas estén determinadas por diferentes valores de subregión de contexto (q[][]).

3. El decodificador de audio de acuerdo con la reivindicación 1 o 2, en donde la representación numérica es una representación de números binarios de un único valor numérico de contexto actual (c); y

en donde una primera subserie de bits de la representación de números binarios está determinada por un primer valor de subregión de contexto asociado a uno o más valores espectrales previamente decodificados; y

en donde una segunda subserie de bits de la representación de números binarios está determinada por un segundo valor de subregión de contexto asociado a uno o más valores espectrales previamente decodificados, en donde los bits de la primera subserie de bits comprenden una ponderación numérica diferente de la que tienen los bits de la segunda subserie de bits.

4. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 3, en donde el decodificador aritmético está configurado para modificar una subserie con bits enmascarados de los bits de información de la representación numérica de los valores numéricos de contexto previo o de una versión desplazada en bits de la representación numérica del valor numérico de contexto previo, dependiendo de un valor de subregión de contexto que no ha sido tenido en cuenta para la derivación del valor numérico de contexto previo, para obtener la representación numérica del valor numérico de contexto actual.

5. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 4, en donde el decodificador aritmético está configurado para el desplazamiento en bits de la representación numérica del valor numérico de contexto previo, de tal manera que se modifiquen las ponderaciones numéricas de las subseries de bits asociadas a diferentes valores de subregión de contexto, para obtener la representación numérica del valor numérico de contexto actual.

6. El decodificador de audio de acuerdo con la reivindicación 5, en donde el decodificador aritmético está configurado para el desplazamiento en bits de la representación numérica del valor numérico de contexto previo, de tal manera que una subserie de bits que están asociados a un valor de subregión de contexto, se suprima de la representación numérica, para obtener la representación numérica del valor numérico de contexto actual.

7. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 6, en donde el decodificador aritmético está configurado para modificar una primera subserie de bits de una representación de números binarios de un valor numérico de contexto previo, o de una versión desplazada en bits de una representación de números binarios de un valor numérico de contexto previo, dependiendo de un valor de subregión de contexto, y para dejar inalterada una segunda subserie de bits de la representación de números binarios del valor numérico de contexto previo, o de la versión desplazada en bits de la representación de números binarios del valor numérico de contexto previo,

para derivar la representación de números binarios del valor numérico de contexto actual de la representación de números binarios del valor numérico de contexto previo mediante la modificación selectiva de una o más subseries de bits asociadas a subregiones del contexto consideradas para la decodificación de los valores espectrales previamente decodificados y no consideradas para la decodificación de valores espectrales a decodificar usando el valor numérico de contexto actual.

8. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 7, en donde el decodificador aritmético está configurado para proporcionar la representación numérica del valor numérico de contexto actual de tal manera que una subserie de bits menos significativos de la representación numérica del valor numérico de contexto actual describa un valor de subregión de contexto, valor de subregión de contexto que se utiliza para una decodificación de valores espectrales para la cual un estado del contexto está definido por el valor numérico de contexto actual, pero donde dicho valor de subregión de contexto no es utilizado para una decodificación de valores espectrales para la cual el estado del contexto está definido por un valor numérico de contexto subsiguiente.

9. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 8, en donde el decodificador aritmético está configurado para evaluar por lo menos una tabla, a fin de determinar si el valor numérico de contexto actual es idéntico a un valor de contexto de la tabla descripto por un elemento de la tabla o se halla dentro de un intervalo descripto por elementos de la tabla, y para derivar un valor índice de regla de mapeo que describe una regla de mapeo seleccionada dependiendo de un resultado de una evaluación de dicha por lo menos una tabla.

10. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 9, en donde el decodificador aritmético está configurado para verificar si la suma de una pluralidad de valores de subregiones de contexto es menor o igual a un valor umbral predeterminado de la suma, y para modificar selectivamente el valor numérico de contexto actual dependiendo de un resultado de la verificación.

11. El decodificador de audio de acuerdo con la reivindicación 10, en donde el decodificador aritmético está configurado para verificar si la suma de una pluralidad de valores de subregión de contexto, valores de subregión de contexto que están asociados a una misma porción temporal del contenido de audio que el uno o más valores espectrales a decodificar usando un estado del contexto definido por el valor numérico de contexto actual, y donde dichos valores de subregiones de contexto están asociados a frecuencias más bajas que el uno o más valores espectrales a decodificar usando el estado del contexto definido por el valor numérico de contexto actual, es menor o igual a un valor umbral predeterminado de la suma, y para modificar selectivamente el valor numérico de contexto actual dependiendo de un resultado de la verificación.

12. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 11, en donde el decodificador aritmético está configurado para sumar los valores absolutos de una primera pluralidad de valores espectrales previamente decodificados para obtener un primer valor de subregión de contexto asociado a la primera pluralidad de valores espectrales previamente decodificados, y para sumar los valores absolutos de una segunda pluralidad de valores espectrales previamente decodificados para obtener un segundo valor de subregión de contexto asociado a la segunda pluralidad de valores espectrales previamente decodificados.

13. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 12, en donde el decodificador aritmético está configurado para limitar los valores de subregión de contexto, de manera que los valores de subregión de contexto se puedan representar empleando una verdadera subserie de bits de información de la representación numérica del valor numérico de contexto previo.

14. El decodificador de audio de acuerdo con una de las reivindicaciones 1 a 13, en donde el decodificador aritmético está configurado para actualizar la representación de números binarios c del valor numérico de contexto previo, para derivar el valor numérico de contexto actual c del valor numérico de contexto previo, utilizando el siguiente algoritmo:

```
c = c>>4;
if(i<i_max-1)
        c = c + (q[0][i+1]<<12);
c = (c&0xFFF0);
if(i>0)
```

c = c + (q[1][i-1]);

en donde c es una variable que representa, en una representación binaria, el valor numérico de contexto previo antes de la ejecución del algoritmo y que representa, en una representación binaria, el valor numérico de contexto actual después de la ejecución del algoritmo;

en donde " >>4" designa una operación de "desplazamiento 4 bits a la derecha";

en donde i es un índice de frecuencia de dicho uno o más valores espectrales a decodificar usando el valor numérico de contexto actual;
en donde i_max designa un número total de índices de frecuencia;

en donde q [0] [i+1] designa un valor de subregión de contexto asociado a uno o más valores espectrales previamente decodificados para frecuencias más elevadas que las frecuencias de uno o más valores espectrales a decodificar usando el valor numérico de contexto actual y correspondientes a una porción temporal anterior del contenido de audio;

en donde "<<12" designa una operación de "desplazamiento 12 bits a la izquierda";

en donde "&0xFFF0" designa una operación Y Booleana con un valor hexadecimal de "0xFFF0"; y

en donde q[1] [i-1] designa un valor de subregión de contexto asociado a uno o más valores espectrales previamente decodificados correspondientes a frecuencias más bajas que las frecuencias de uno o más valores espectrales a decodificar usando el valor numérico de contexto actual y correspondientes a una porción temporal actual del contenido de audio.

15. El decodificador de audio de acuerdo con la reivindicación 14, en donde el decodificador aritmético está configurado para modificar selectivamente la representación de números binarios c del valor numérico de contexto actual mediante el aumento de c en un valor hexadecimal de 0x10000, si

(q[1][i-3]+q[1][i-2]+q[1][i-1]) <5;

en donde q[1][i-3], q[1][i-2] y q[1][i-1] son valores de subregión de contexto, cada uno de los cuales está asociado a uno o más valores espectrales previamente decodificados para frecuencias más bajas que las frecuencias de uno o más valores espectrales a decodificar usando el valor numérico de contexto actual y correspondiente a la porción temporal actual del contenido de audio.

16. Un codificador de audio (100;700) para la provisión de una información de audio codificada sobre la base de una información de audio de entrada (110;710), en donde el codificador de audio comprende:

un convertidor compactador de energía del dominio del tiempo al dominio de la frecuencia (130;720) para la provisión de una representación de audio en el dominio de la frecuencia (132;722) sobre la base de una representación en el dominio del tiempo (110;710) de la información de audio de entrada, de manera que la representación de audio en el dominio de la frecuencia (132;722) comprende una serie de valores espectrales; y

un codificador aritmético (170;730) configurado para codificar un valor espectral (a) o una versión previamente procesada de los mismos, mediante el uso de una palabra código de longitud variable (acod_m, acod_r), en donde el codificador aritmético (170) está configurado para mapear uno o más valores espectrales (a, b), o un valor (m) de un plano de bits más significativos de uno o más valores espectrales (a, b), sobre un valor de código (acod_m),

en donde la información de audio codificada comprende una pluralidad de palabras código de longitud variable;

en donde el codificador aritmético está configurado para seleccionar una regla de mapeo que describe un mapeo de uno o más valores espectrales, o de un valor de un plano de bits más significativos de uno o más valores espectrales, sobre un valor de código dependiendo de un estado del contexto (s) descripto por un valor numérico de contexto actual (c); y

en donde el codificador aritmético está configurado para determinar el valor numérico de contexto actual (c) dependiendo de un valor numérico de contexto previo y dependiendo de una pluralidad de valores espectrales previamente codificados,

en donde el codificador aritmético está configurado para modificar una representación numérica (c) de un valor numérico de contexto previo, que describe un estado del contexto asociado a uno o más valores

espectrales previamente codificados, dependiendo de un valor de subregión de contexto que describe una subregión de un contexto, para obtener una representación numérica de un valor numérico de contexto actual que describe un estado para la codificación de uno o más valores espectrales a codificar.

17. Un método para proporcionar una información de audio decodificada sobre la base de una información de audio codificada, en donde el método comprende:

suministrar una pluralidad de valores espectrales decodificados sobre la base de una representación aritméticamente codificada de los valores espectrales comprendidos en la información de audio codificada; y

suministrar una representación de audio en el dominio del tiempo utilizando los valores espectrales, para obtener la información de audio decodificada;

en donde suministrar la pluralidad de valores espectrales decodificados comprende la selección de una regla de mapeo que describe un mapeo de un valor de código (acod_m; value) de la representación aritméticamente codificada (222; 821) de valores espectrales a un código de símbolo (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, dependiendo de un estado del contexto descripto por un valor numérico de contexto actual (c); y

en donde el valor numérico de contexto actual (c) se determina dependiendo de un valor numérico de contexto previo y dependiendo de una pluralidad de valores espectrales previamente decodificados,

en donde una representación numérica de un valor numérico de contexto previo, que describe un estado del contexto para la decodificación de uno o más valores espectrales previamente decodificados, se modifica dependiendo de un valor de subregión de contexto que describe una subregión de un contexto, para obtener una representación numérica de a valor numérico de contexto actual, que describe un estado del contexto para la decodificación de uno o más valores espectrales a decodificar.

18. Un método para proporcionar una información de audio codificada sobre la base de una información de audio de entrada, en donde el método comprende:

suministrar una representación de audio en el dominio de la frecuencia sobre la base de una representación en el dominio del tiempo de la información de audio de entrada usando una conversión compactadora de energía del dominio del tiempo al dominio de la frecuencia, de modo que la representación de audio en el dominio de la frecuencia comprende una serie de valores espectrales; y

codificar aritméticamente un valor espectral, o una versión previamente procesada del mismo, usando una palabra clave de longitud variable, en donde se mapea un valor espectral o un valor de un plano de bits más significativos de un valor espectral sobre un valor de código;

en donde una regla de mapeo que describe un mapeo de uno o más valores espectrales, o de un plano de bits más significativos de uno o más valores espectrales, sobre un valor de código es seleccionada según un estado del contexto descripto por un valor numérico de contexto actual (c); y

en donde el valor numérico de contexto actual (c) se determina dependiendo de un valor numérico de contexto previo y dependiendo de una pluralidad de valores espectrales previamente codificados;

en donde una representación numérica de un valor numérico de contexto previo, que describe un estado del contexto para la codificación de uno o más valores espectrales previamente codificados, se modifica dependiendo de un valor de subregión de contexto que describe una subregión de un contexto, para obtener una representación numérica de un valor numérico de contexto actual que describe un estado del contexto para la codificación de uno o más valores espectrales a codificar;

en donde la información de audio codificada comprende una pluralidad de palabras código de longitud variable.

19. Un programa de computación para poner en práctica el método de acuerdo con la reivindicación 17 o la reivindicación 18 al ejecutarse el programa de computación en una computadora.

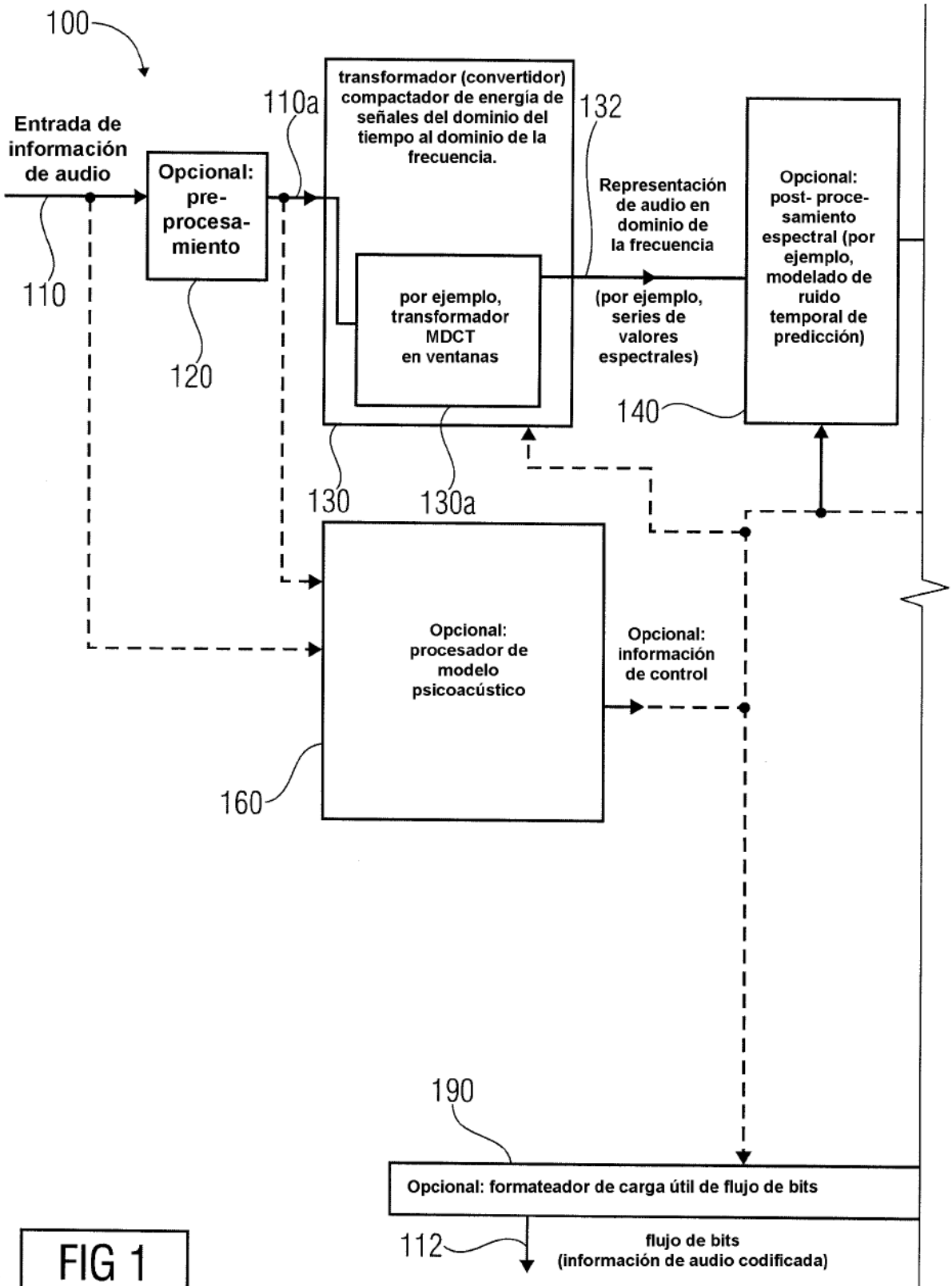


FIG 1A

FIG 1

FIG 1A	FIG 1B

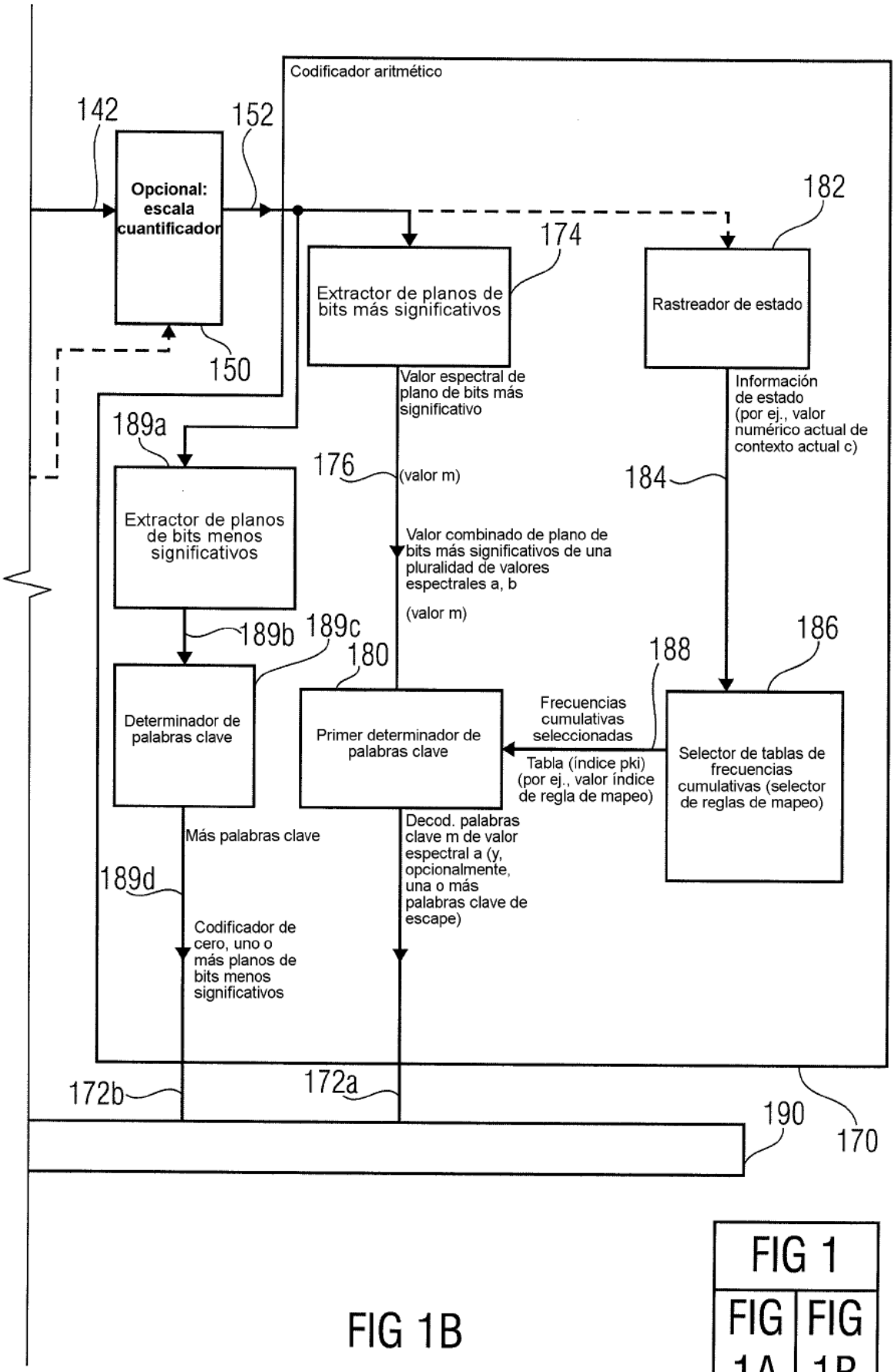


FIG 1B

FIG 1

FIG 1A	FIG 1B

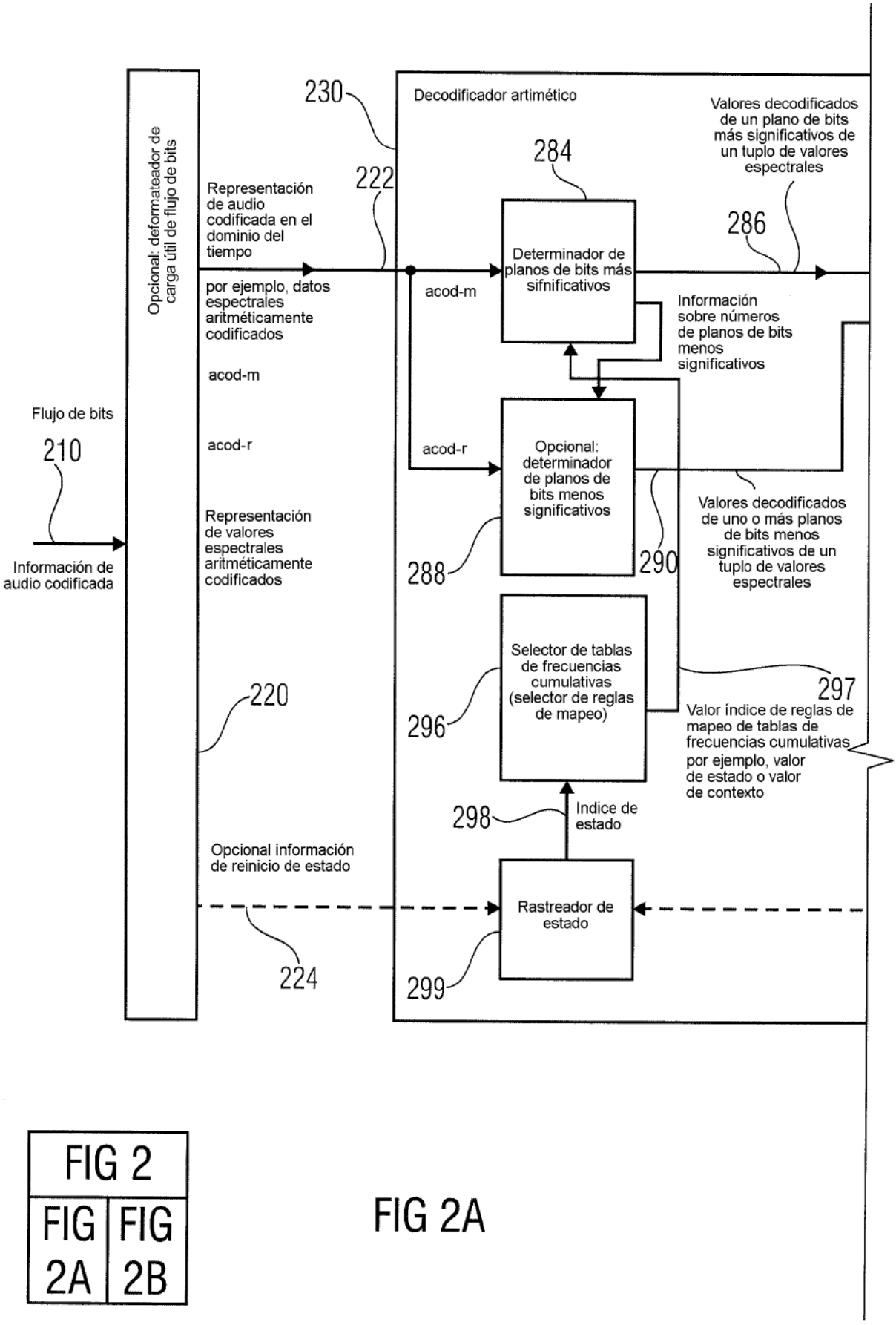


FIG 2A

FIG 2

FIG 2A	FIG 2B

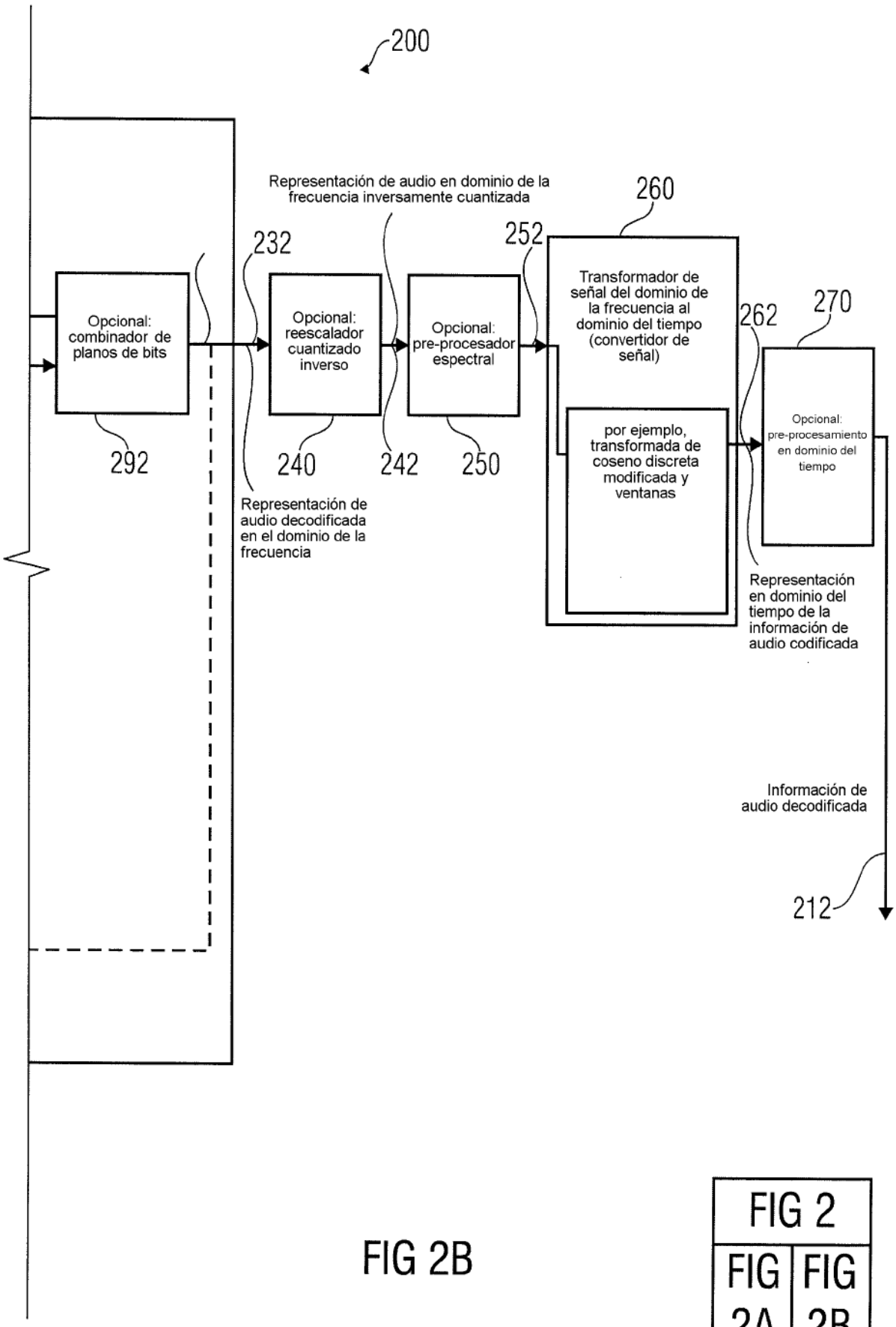


FIG 2B

FIG 2

FIG 2A	FIG 2B

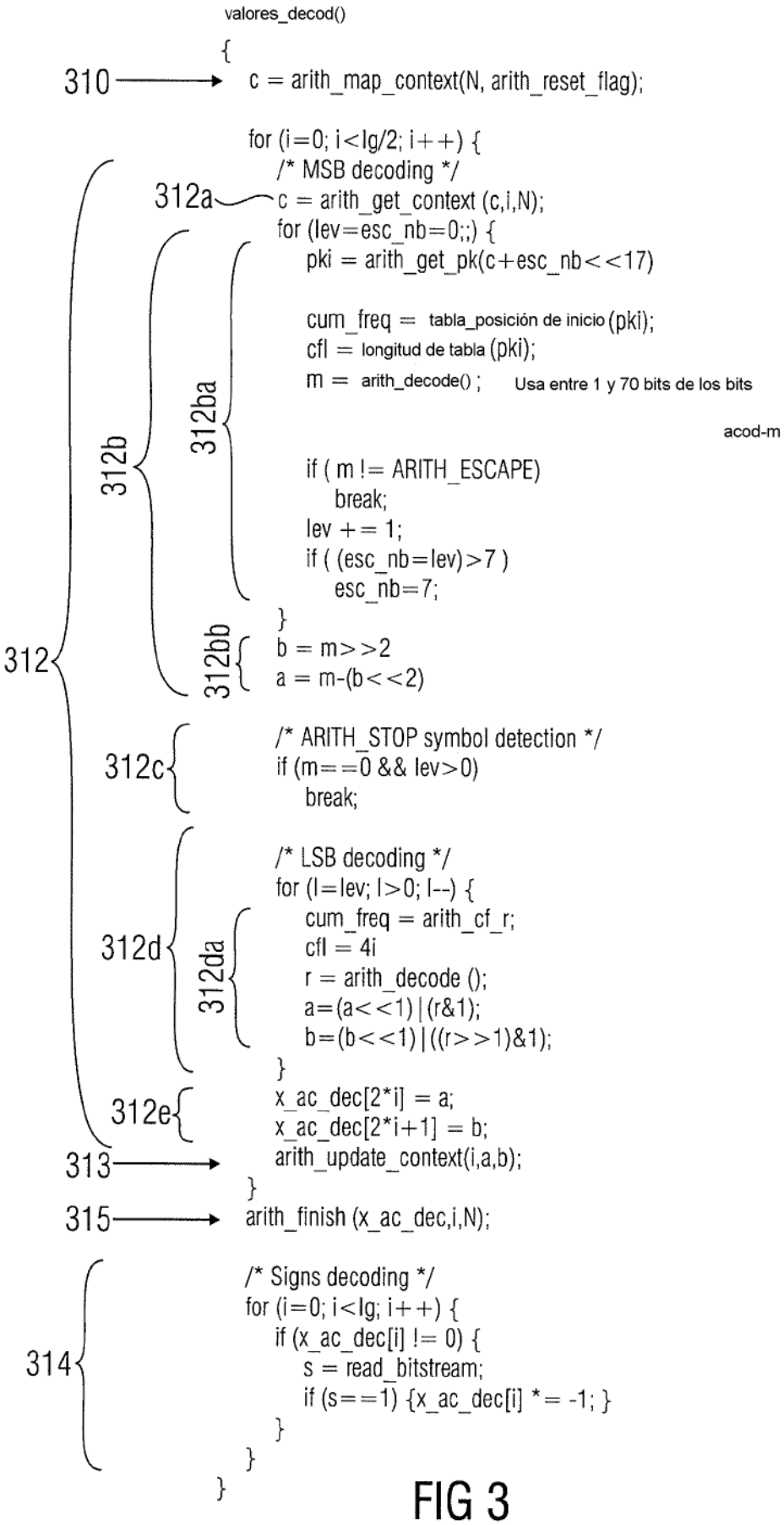

valores_decod()
{
310
c = arith_map_context(N, arith_reset_flag);
for (i=0; i<lg/2; i++) {
/* MSB decoding */
312a
c = arith_get_context (c,i,N);
for (lev=esc_nb=0;;) {
312b
312ba
pki = arith_get_pk(c+esc_nb<<17)
cum_freq = tabla_posición de inicio (pki);
cfl = longitud de tabla (pki);
m = arith_decode() ; Usa entre 1 y 70 bits de los bits
acod-m
if (m != ARITH_ESCAPE)
break;
lev += 1;
if ((esc_nb=lev)>7)
esc_nb=7;
}
312
312bb
b = m>>2
a = m-(b<<2)
/* ARITH_STOP symbol detection */
312c
if (m==0 && lev>0)
break;
/* LSB decoding */
312d
for (l=lev; l>0; l--) {
312da
cum_freq = arith_cf_r;
cfl = 4i
r = arith_decode ();
a=(a<<1)|(r&1);
b=(b<<1)|((r>>1)&1);
}
312e
x_ac_dec[2*i] = a;
x_ac_dec[2*i+1] = b;
313
arith_update_context(i,a,b);
}
315
arith_finish (x_ac_dec,i,N);
/* Signs decoding */
314
for (i=0; i<lg; i++) {
if (x_ac_dec[i] != 0) {
s = read_bitstream;
if (s==1) {x_ac_dec[i] *= -1; }
}
}
}

FIG 3

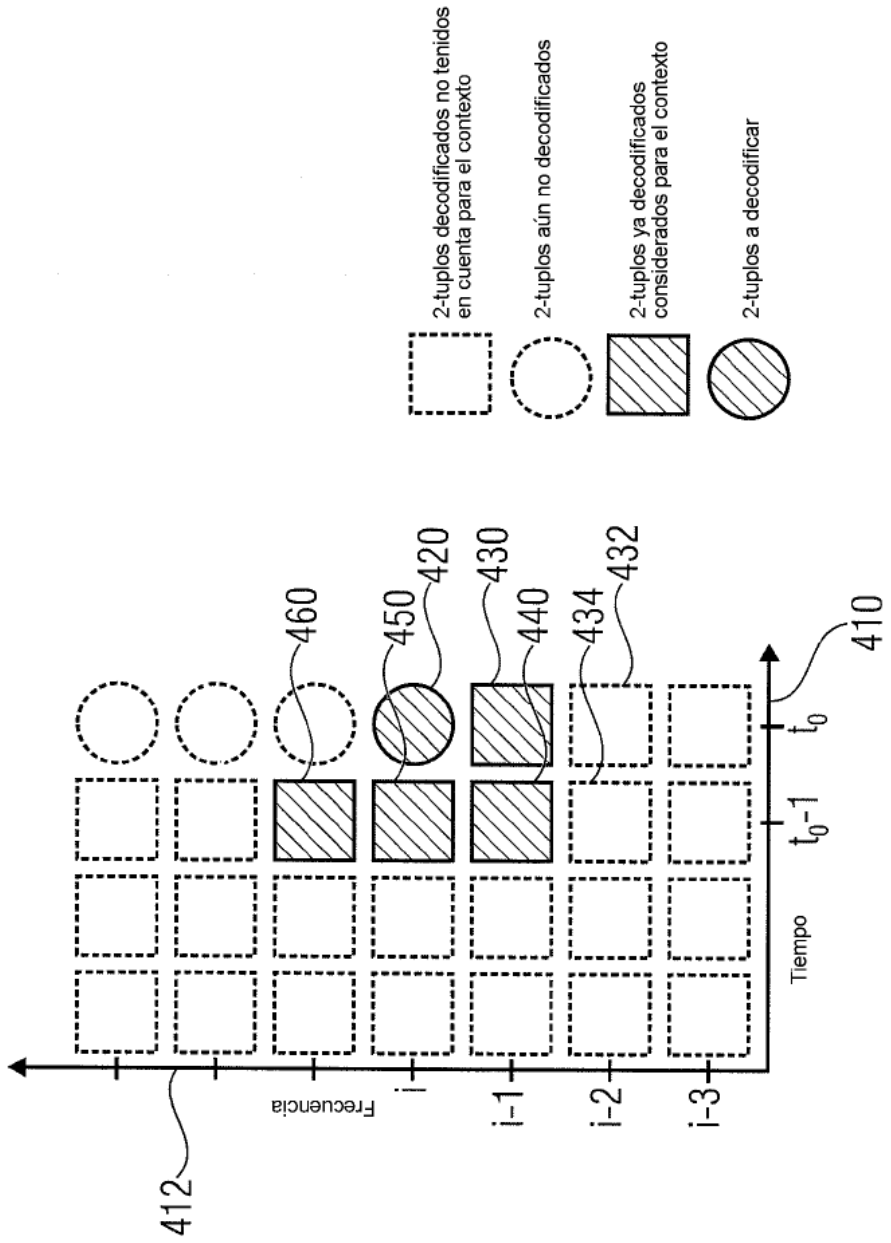

2-tuplos decodificados no tenidos en cuenta para el contexto
2-tuplos aún no decodificados
2-tuplos ya decodificados considerados para el contexto
2-tuplos a decodificar
460
450
420
430
440
434
432
410
412
Frecuencia
i
i-1
i-2
i-3
Tiempo
t_0-1
t_0
Contexto para el cálculo de estado

FIG 4

```
/* Variables de entrada */
N /* Longitud ventana actual */
arith_reset_flag /* Bandera reinicio codificador aritmético */

/* Variables globales */
previous_N /* Longitud ventana anterior */

c = arith_map_context(N,arith_reset_flag)
{
    si (arith_reset_flag) {
        por (j=0; j<N/4; j++) {
            q[0][j]=0;
        }
    } o {
        relación = ((float)previous_N) / ((float)N);
        por (j=0; j<N/4; j++) {
            k = (int) ((float) j * ratio);
            q[0][j] = q[1][k];
        }
    }

    previous_N=N;

    return(q[0][0]<<12);
}
```

500a

500b

FIG 5A

```
/* Variables de entrada */
lg /* Número de coeficientes espectrales a decodificar en el cuadro*/
arith_reset_flag /* Bandera reinicio codificador aritmético */
/*Global variables*/
previous_lg /* Número anterior de líneas espectrales del cuadro anterior*/

c=arith_map_context (lg,arith_reset_flag)
{
    v=w=0

    if(arith_reset_flag){
        for(j=0; j<lg/2; j++){
            q[0][v++]=0;
        }
      ◦ {
        ratio= ((float)previous_lg)/((float)lg);
        for(j=0; j<lg/2; j++){
            k = (int) ((float)) ((j)*ratio);
            q[0][v++] = qs[w+k];
        }
    }

    previous_lg=lg;

    return(q[0][0]<<12);
}
```

FIG 5B

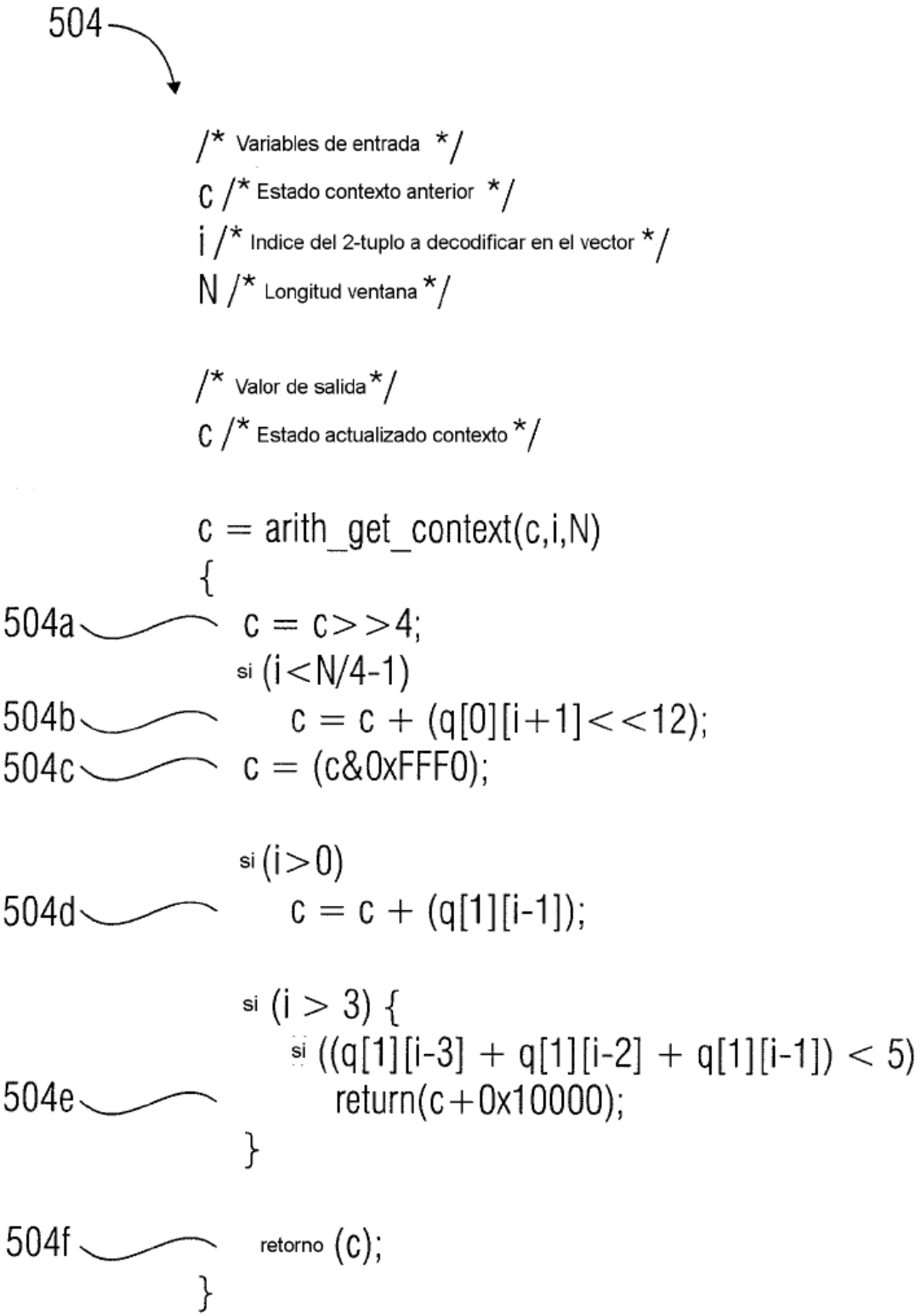

504
/* Variables de entrada */
c /* Estado contexto anterior */
i /* Indice del 2-tuplo a decodificar en el vector */
N /* Longitud ventana */
/* Valor de salida */
c /* Estado actualizado contexto */
c = arith_get_context(c,i,N)
{
504a
c = c>>4;
si (i<N/4-1)
504b
c = c + (q[0][i+1]<<12);
504c
c = (c&0xFFF0);
si (i>0)
504d
c = c + (q[1][i-1]);
si (i > 3) {
si ((q[1][i-3] + q[1][i-2] + q[1][i-1]) < 5)
504e
return(c+0x10000);
}
504f
retorno (c);
}

FIG 5C

```
/* Variables de entrada */
c /* Estado contexto anterior */
i /* Indice del 2-tuplo a decodificar en el vector */
/* Valor de salida */
c /* Estado actualizado contexto */


c=arith_get_context(c,i)
{
    c=c>>4;
    c=(c)+(q[0][i+1]<<12);
    c=(c&0xFFF0)+(q[1][i-1]);

    if(i > 3) {
        if((q[1][i-3] + q[1][i-2] + q[1][i-1]) < 5)
            return(c+0x10000);
    }

    return(c);
}
```

FIG 5D

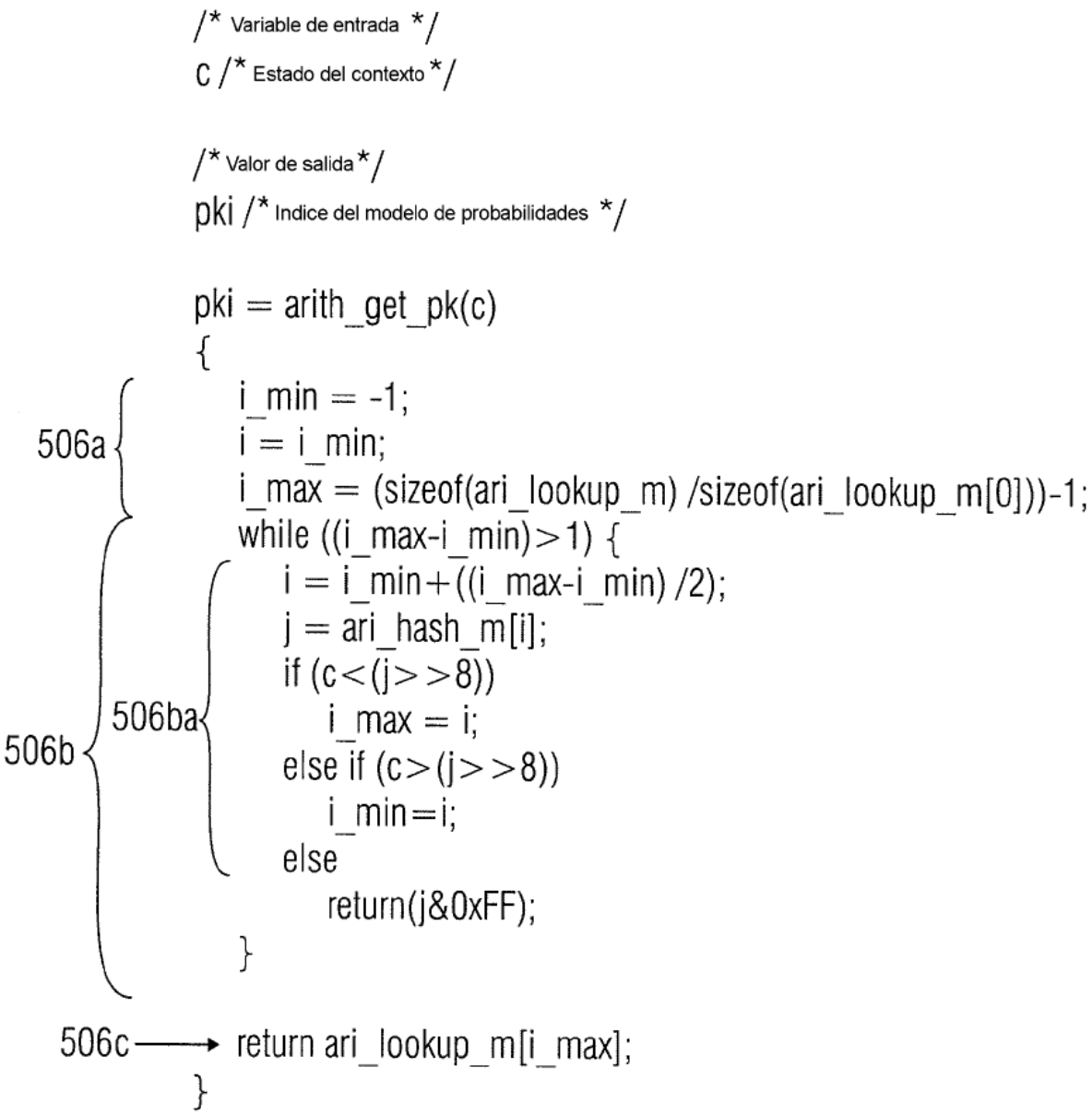

/* Variable de entrada */
c /* Estado del contexto */
/* Valor de salida */
pki /* Indice del modelo de probabilidades */
pki = arith_get_pk(c)
{
i_min = -1;
i = i_min;
i_max = (sizeof(ari_lookup_m) /sizeof(ari_lookup_m[0]))-1;
while ((i_max-i_min)>1) {
i = i_min+((i_max-i_min) /2);
j = ari_hash_m[i];
if (c<(j>>8))
i_max = i;
else if (c>(j>>8))
i_min=i;
else
return(j&0xFF);
}
return ari_lookup_m[i_max];
}
506a
506b
506ba
506c

FIG 5E

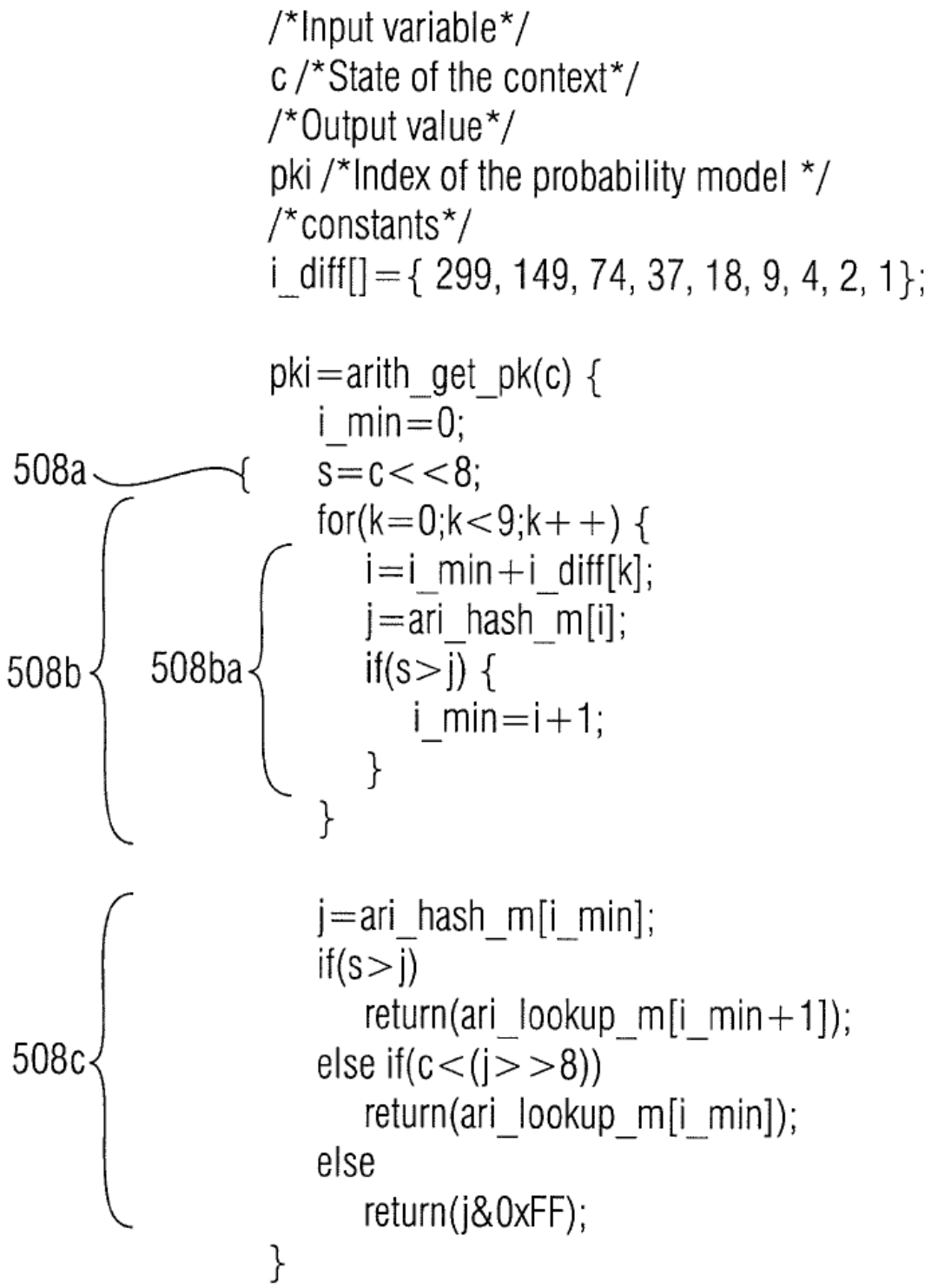

/*Input variable*/
c /*State of the context*/
/*Output value*/
pki /*Index of the probability model */
/*constants*/
i_diff[]={ 299, 149, 74, 37, 18, 9, 4, 2, 1};
pki=arith_get_pk(c) {
i_min=0;
s=c<<8;
for(k=0;k<9;k++) {
i=i_min+i_diff[k];
j=ari_hash_m[i];
if(s>j) {
i_min=i+1;
}
}
j=ari_hash_m[i_min];
if(s>j)
return(ari_lookup_m[i_min+1]);
else if(c<(j>>8))
return(ari_lookup_m[i_min]);
else
return(j&0xFF);
}
508a
508b
508ba
508c

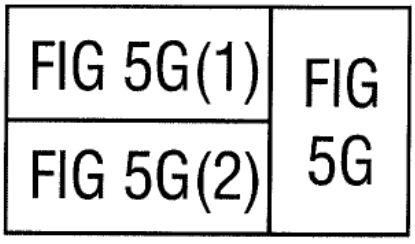
FIG 5F

```
/* funciones auxiliares */
bool arith_first_symbol(void);
    /* Return TRUE si es el primer símbolo de la secuencia,
       FALSE si no */
Ushort arith_get_next_bit(void);
    /* seguir con próximo bit del flujo de bits */

/* variables globales */
low
high
value

/* variables de entrada */
cum_freq[]; /* tabla de frecuencias cumulativas */
cfl; /* longitud de cum_freq[] */

symbol = arith_decode(cum_freq, cfl)
{
    if (arith_first_symbol()) {
        value = 0;
        for (i=1; i<=16; i++) {
            value = (val<<1) | arith_get_next_bit();
        }
        low  = 0;
        high = 65535;
    }

    range = high-low+1;
    cum =((((int) (value-low+1))<<14)-((int) 1))/range;
    p = cum_freq-1;
```

570a

570b

FIG 5G(1)



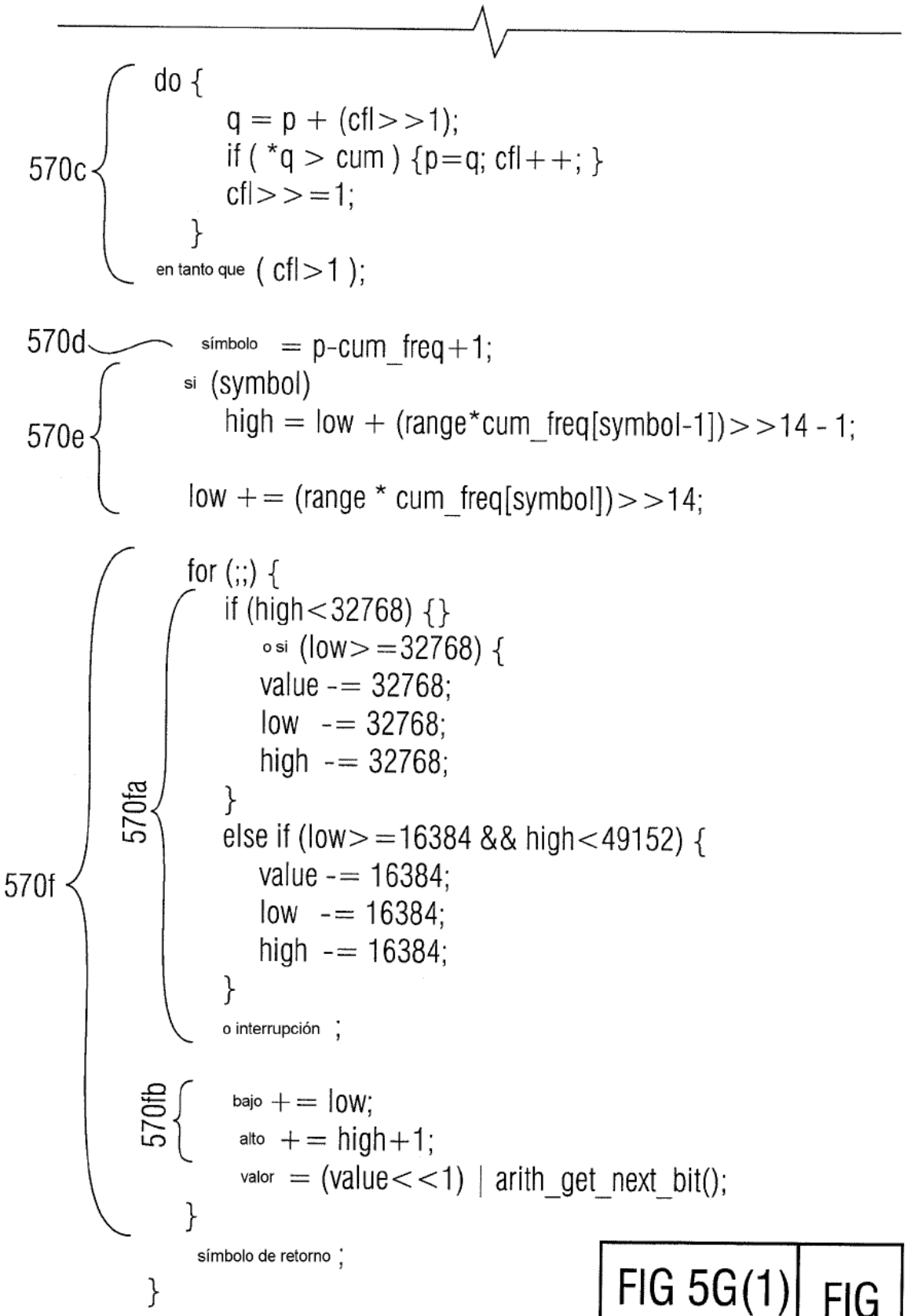


FIG 5G(2)

FIG 5G(1)	FIG 5G
FIG 5G(2)	

```
/*helper functions*/
bool arith_first_symbol(void);
    /* Return TRUE if it is the first symbol of the sequence,
    FALSE otherwise */
Ushort arith_get_next_bit(void);
    /* Get the next bit of the bitstream */

/* global variables */
low
high
value

/* input variables */
cum_freq[]; /* cumulative frequencies table */
cfl; /* length of cum_freq[] */

symbol = arith_decode(cum_freq, cfl)
{
    if (arith_first_symbol()) {
        value = 0;
        for (i=1; i<=16; i++) {
            value = (val<<1) | arith_get_next_bit();
        }
        low  = 0;
        high  = 65535;
    }

    range = high-low+1;
    cum =((((int) (value-low+1))<<14)-((int) 1));
    p = cum_freq-1;

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

< Continúa en la FIG 5I>

FIG 5H

< Continuación de la FIG 5H>

```
    while ( cfl>1 );

    symbol = p-cum_freq+1;
    if (symbol)
        high = low + (range*cum_freq[symbol-1])>>14 - 1;

    low += (range * cum_freq[symbol])>>14;

    for (;;) {
        if (high<32768) {}
        else if (low>=32768) {
            value -= 32768;
            low  -= 32768;
            high  -= 32768;
        }
        else if (low>=16384 && high<49152) {
            value -= 16384;
            low  -= 16384;
            high  -= 16384;
        }
        else break;

        low += low;
        high += high+1;
        value = (value<<1) | arith_get_next_bit();
    }
    return symbol;
}
```

FIG 5I

```
b = m>>2;
a = m-(b<<2);
for (j=0;j<lev;j++) {
    r = arith_decode(arith_cf_r,4);
    a = (a<<1) | (r&1);
    b = (b<<1) | ((r>>1)&1);
}
```

FIG 5J

```
x_ac_dec[2*i]   = a
x_ac_dec[2*i+1] = b;
```

FIG 5K

```
/*input variables*/
a,b /* Decoded unsigned quantized spectralcoefficients of the 2-tuple */
i /* Index of the quantized spectral coefficient to decode */

arith_update_context(i, a, b)
{
    q[1][i] = a+b+1;
    if (q[1][i]>0xF)
        q[1][i] = 0xF;
}
```

FIG 5L

```
/*input variables*/
offset /*number of decoded 2-tuple */
N /*Window length */
x_ac_dec  /*vector of decoded spectal coefficients*/

arith_finish(x_ac_dec,offset,N)
{
   for(i=offset ;i<N/4;i++) {
      x_ac_dec[2*i] = 0;
      x_ac_dec[2*i+1] = 0;
      q[1][i] = 1;
   }
}
```

FIG 5M

```
b= m>>2
a = m&0x03;
for(j=0;j<lev;j++){
   r = arith_decode(arith_cf_r,4);
   a = (a<<1) | (r&1);
   b = (b<<1) |((r>>1)&1);

}
```

FIG 5N

```
/*input variables*/
a,b /*Decoded unsigned quantized spectralcoefficients of the 2-tuple*/
i /*Index of the quantized spectral coefficient to decode*/

arith_update_context (){
   qdec[2*i]=a
   qdec[2*i+1]=b;
   q[1][i]=a+b+1;

   if(q[1][i]>0xF)
      q[1][i]=0xF;
}
```

FIG 5O

```
/*input variables*/
i /*Index of the quantized spectral coefficient to decode*/
lg /*number of coefficients in the frame*/

arith_save_context(i,lg){

    for(;i<N/4;i++){
        qdec[2*i]=0;
        qdec[2*i+1]=0;
        q[1][i]=1;
    }

    if(core_mode==1){
        ratio= ((float) lg)/((float)1024);
        for(j=0; j<512; j++){
            k = (int) ((float) j*ratio);
            qs[j] = q[1][k];
        }
        previous_lg = 512;
    }
    else{
        for(j=0; j<512; j++){
            qs[j] = q[1][j];
        }
        previous_lg = MIN(1024,lg);
    }
}
```

FIG 5P

Definiciones

a,b	2-tuplo para decodificar (el coeficiente espectral cuantizado de 2-tuplo a decodificar)
m	El plano de 2 bits más significativo del coeficiente espectral cuantizado a decodificar
r	El plano de 2 bits más significativo del coeficiente espectral cuantizado a decodificar
lev	Nivel de los planos de bits restantes.Corresponde al número de planos de bits menos significativos que el plano de 2 bits más significativo
arith_hash_m[]	Tabla hash que mapea estados del contexto a un índice de tablas de frecuencias cumulativas pki
arith_lookup_m[]	Tabla hash que mapea estados del contexto a un índice de tablas de frecuencias acumulativas pki
arith_cf_m[pki][17]	Modelos de frecuencias cumulativas correspondientes al plano m de 2bits más significativos y el símbolo ARITH_ESCAPE
arith_cf_r [lsbidx][]	Frecuencias acumulativas para los planos de bits menos significativos de símbolo r
arith_cf_r []	Frecuencias acumulativas para los planos de bits menos significativos de símbolo r
q[2][]	El contexto actual para el coeficiente espectral de decodificar
x_ace_dec[]	Los coeficientes espectrales cuantizados decodificados
arith_reset_flag	Bandera que indica si se debe resetear el contexto sin ruido espectral
ARITH_STOP	Símbolo de parada que consiste en la sucesión del símbolo ARITH_ESCAPE y m=0. Cuando aparece, el resto del cuadro se decodifica con valores de cero
N	Longitud de ventana. En el caso de AAC se deduce de la window_sequence (secuencia de ventanas) (ver la sección 6.8.3.1) y TCX N=2.lg.-
previous_N	Longitud de la ventana anterior

FIG 5Q

a,b	El coeficiente espectral cuantizado de 2-tuplo a decodificar
m	El plano de 2 bits más significativo del coeficiente espectral cuantizado a decodificar
r	El plano de 2 bits más significativo del coeficiente espectral cuantizado a decodificar
lev	Nivel de los planos de bits restantes. Corresponde al número de planos de bits menos significativos que el plano de 2 bits más significativo
arith_hash_m[]	
arith_lookup_m[]	Tabla hash de búsqueda que mapea un grupo de estados del contexto a un índice de tablas de frecuencias cumulativas pki
arith_cf_m[pki][17]	
arith_cf_r []	Modelos de frecuencias cumulativas correspondientes al plano de 2 bits más significativos y el símbolo ARITH_ESCAPE
previous_lg	Número de coeficientes espectrales transmitidos previamente decodificados por el decodificador aritmético
q[2][]	El contexto actual para el coeficiente espectral a decodificar
qs[]	El contexto anterior guardado para el cuadro siguiente
qdec[]	Los coeficientes espectrales cuantizados decodificados
arith_reset_flag	Bandera que indica si se debe resetear el contexto sin ruido espectral
ARITH_STOP	Símbolo de parada que consiste en la sucesión del símbolo ARITH_ESCAPE y m=0. Cuando aparece, el resto del cuadro se decodifica con valores de cero
N	Longitud de ventana. En el caso de AAC se deduce de la window_sequence (secuencia de ventanas) (ver la sección 6.8.3.1) y TCX N=2.lg.-

FIG 5R

```
Bloque de datos brutos USAC()
    {
        Elemento de canal único(); y/o
      Elemento - par de canales();
    }
```

FIG 6A

Sintaxis de canal único (single_channel_element)

Sintaxis	No. de bits	Mnemónica

single_channel_element()
{
core_mode
if (core_mode == 1) {
lpd_channel_stream();
}
else {
fd_channel_stream();
}
}
	1	**uimsbf**

FIG 6B

Sintaxis de elementos de par de canales (channel_pair_element)

Sintaxis	No. de bits	Mnemónica
channel_pair_element()		
{		
core_mode0	**1**	**uimsbf**
core_mode1	**1**	**uimsbf**
ics_info();		
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 ics_info()

Sintaxis	No. de bits	Mnemónica	
ics_info()			
{			
window_length;	**1**	**uimsbf**	
if(window_length !=0) {			
transform_length;	**1**	**uimsbf**	
}			
else {			
transform_length=0;			
}			
window_shape;	**1**	**uimsbf**	
if (window_length !=0 && transform_length !=0){			Opcional
max_sfb;	**4**	**uimsbf**	
scale_factor_grouping;	**7**	**uimsbf**	
}			
else {			
max_sfb;	**6**	**uimsbf**	
}			
}			

FIG 6D

Sintaxis de fd_channel_stream()

Sintaxis	No. de bits	Mnemónica
fd_channel_stream() {		
global_gain;	**8**	**uimsbf**
ics_info(); (a menos que esté incluido en el elemento de par de canales)		
scale_factor_data ();		
ac_spectral_data ();		
}		

FIG 6E

Sintaxis de ac_spectral_data()

Sintaxis	No. de bits	Mnemónica
ac_spectral_data() {		
arith_reset_flag	**1**	**uimsbf**
for (win=0; win<num_windows; win++){		
arith_data(num_bands, arith_reset_flag)		
}		
}		

FIG 6F

Sintaxis de arith_data()

Sintaxis	No. de bits	Mnemónica
arith_data(lg, arith_reset_flag)		
{		
c = arith_map_context(N, arith_reset_flag);		
for (i=0; i<lg/2; i++) {		
/* MSB decoding */		
c = arith_get_context (c,i,N);		
for (lev=esc_nb=0;;) {		
662 pki = arith_get_pk(c+esc_nb<<17)		
663 **acod_m**[pki][m]	1..20	vlclbf
if (m != ARITH_ESCAPE)		
break;		
664 lev += 1;		
if ((esc_nb=lev)>7)		
esc_nb=7;		
}		
b = m>>2;		
a = m – (b<<2);		
/* ARITH_STOP symbol detection */		
if (m==0 && lev>0)		
break;		
/* LSB decoding */		
for (l=lev; l>0; l--) {		
acod_r[r]	1..20	vlclbf
a=(a<<1)|(r&1);		
b=(b<<1)|((r>>1)&1);		
}		
x_ac_dec[2*i] = a;		
x_ac_dec[2*i+1] = b;		
668 arith_update_context(i,a,b);		
}		
arith_finish (x_ac_dec,lg,N);		
/* Signs decoding */		
for (i=0; i<lg; i++) {		
if (x_ac_dec[i] != 0) {		
s;	1	uimsbf
if (s==1) {x_ac_dec[i] *= -1; }		
}		
}		
}		

FIG 6G

Tabla SIntaxis de arith_data()

Sintaxis	No. de bits	Mnemónica
Arith_data(lg, arith_reset_flag){		
c=arith_map_context(lg, arith_reset_flag);		
for (i=0; i<lg/2; i++) {		
/*MSBs decoding*/		
c = arith_get_context (c,i);		
for (lev=esc_nb=0;;) {		
pki = arith_get_pk(c+esc_nb<<17)		
acod_m[pki][m]	1..20	vlclbf
if (m != ARITH_ESCAPE)		
break;		
lev += 1;		
if((esc_nb=lev)>7)		
esc_nb=7;		
}		
b=m>>2;		
a=m-(b<<2);		
/*ARITH_STOP symbol detection*/		
if(m==0 && lev>0)		
break;		
/*LSBs decoding*/		
for (l=lev; l>0; l--) {		
acod_r[r]	1..20	vlclbf
a=(a<<1)|(r&1);		
b=(b<<1)|((r>>1)&1);		
}		
arith_update_context(a,b,i);		
}		
arith_save_arith (l,lg);		
/*Signs decoding*/		
for (i=0; i<lg/2; i++) {		
if(a!=0){		
s;	1	uimsbf
If(s) a=-a;		
}		
if(b!=0){		
s;	1	uimsbf
if(s) b=-b;		
}		
}		
}		

FIG 6H

Definiciones

arith_data()	Elementos de datos para decodificar los datos del codificador espectral sin ruido
arith_reset_flag	Bandera que indica si se debe resetear el contexto espectral sin ruido
acod _m[pki][m]	Palabra código aritmético necesaria para la decodificación del plano de 2 bits más significativo *m* de los coeficientes espectrales cuantizados de un 2-tuplo
acod_r[lsbidx][]	
s	

Elementos de ayuda

a,b	2-tuplo que corresponde a los coeficientes espectrales cuantizados a decodificar
m	El plano de 2 bits más significativo de un coeficiente espectral cuantizado a decodificar
r	El plano de 2 bits menos significativo del 2-tuplo a decodificar
lg	Número de coeficientes cuantizados a decodificar
N	Longitud de ventana. En el caso de FD se deduce de la window_sequence (secuencia de ventanas) y TCX N=2.lg.
i	Indice de 2-tuplos a decodificar dentro del cuadro
pki	Indice de la tabla de frecuencias cumulativas utilizada por el decodificador aritmético para decodificar m
arith_get_pk ()	Función que devuelve el índice pki de la tabla de frecuencias cumulativas necesaria para decodificar la palabra código acod_m[pki][m]
c	Estado del contexto
lsbidx	Indice de las tablas de frecuencias cumulativas empleadas por el decodificador aritmético para decodificar r
lev	Nivel de los planos de bits a decodificar más allá del plano de 2 bits más significativo
ARITH_ESCAPE	Símbolo de escape que indica los planos de bits adicionales a decodificar más allá del nivel predicho del plano de bits lev0
esc_nb	Número de símbolo ARITH_ESCAPE ya decodificados por el presente 2-tuplo. EL valor se limita a 7
x_ac_dec[]	Elemento que contiene los coeficientes espectrales decodificados
arith_map_context()	Inicializa los contextos necesarios para decodificar el presente cuadro
arith_get_context()	Computa el estado del contexto para decodificar *m* símbolos del presente 2-tuplo
arith_update_context()	Actualiza el contexto para el siguiente 2-tuplo
arith_finish ()	Termina la decodificación sin ruido.

FIG 6I

Definiciones

arith_data()	Elementos de datos para decodificar los datos del codificador espectral sin ruido
arith_reset_flag	Bandera que indica si se debe resetear el contexto espectral sin ruido
acod _m[pki][m]	Palabra código aritmético necesaria para la decodificación del plano de 2 bits más significativo *m* de los coeficientes espectrales cuantizados de un 2-tuplo
arith_r[]	Palabra código aritmético necesaria para la decodificación de los planos de bits residuales *r* del coeficiente espectral cuantizado de un 2-tuplo
s	El signo codificado del coeficiente espectral cuantizado no nulo

Elementos de ayuda

a,b	2-tuplo que corresponde a los coeficientes espectrales cuantizados a decodificar
m	El plano de 2 bits más significativo de un coeficiente espectral cuantizado a decodificar
r	El plano de 2 bits menos significativo del 2-tuplo a decodificar
lg	Indice del 2-tuplo a decodificar dentro del cuadro
i	Indice de 2-tuplos a decodificar dentro del cuadro
pki	Indice de la tabla de frecuencias cumulativas utilizada por el decodificador aritmético para decodificar *m*
arith_get_pk ()	Función que devuelve el índice pki de la tabla de frecuencias cumulativas necesaria para decodificar la palabra código acod_m[pki][m]
c	Estado del contexto
lev	Nivel de los planos de bits a decodificar más allá del plano de 2 bits más significativo
ARITH_ESCAPE	Símbolo de escape que indica los planos de bits adicionales a decodificar más allá del nivel predicho del plano de bits lev0
esc_nb	Número de símbolo ARITH_ESCAPE ya decodificados en el presente 2-tuplo. El valor se limita a 7
arith_map_context()	Inicializa los contextos necesarios para decodificar el presente cuadro
arith_get_context()	Computa el estado del contexto para decodificar *m* símbolos del presente 2-tuplo
arith_update_context()	Actualiza el contexto para el siguiente 2-tuplo
arith_save_context()	Guarda el contexto para el siguiente cuadro a decodificar

FIG 6J

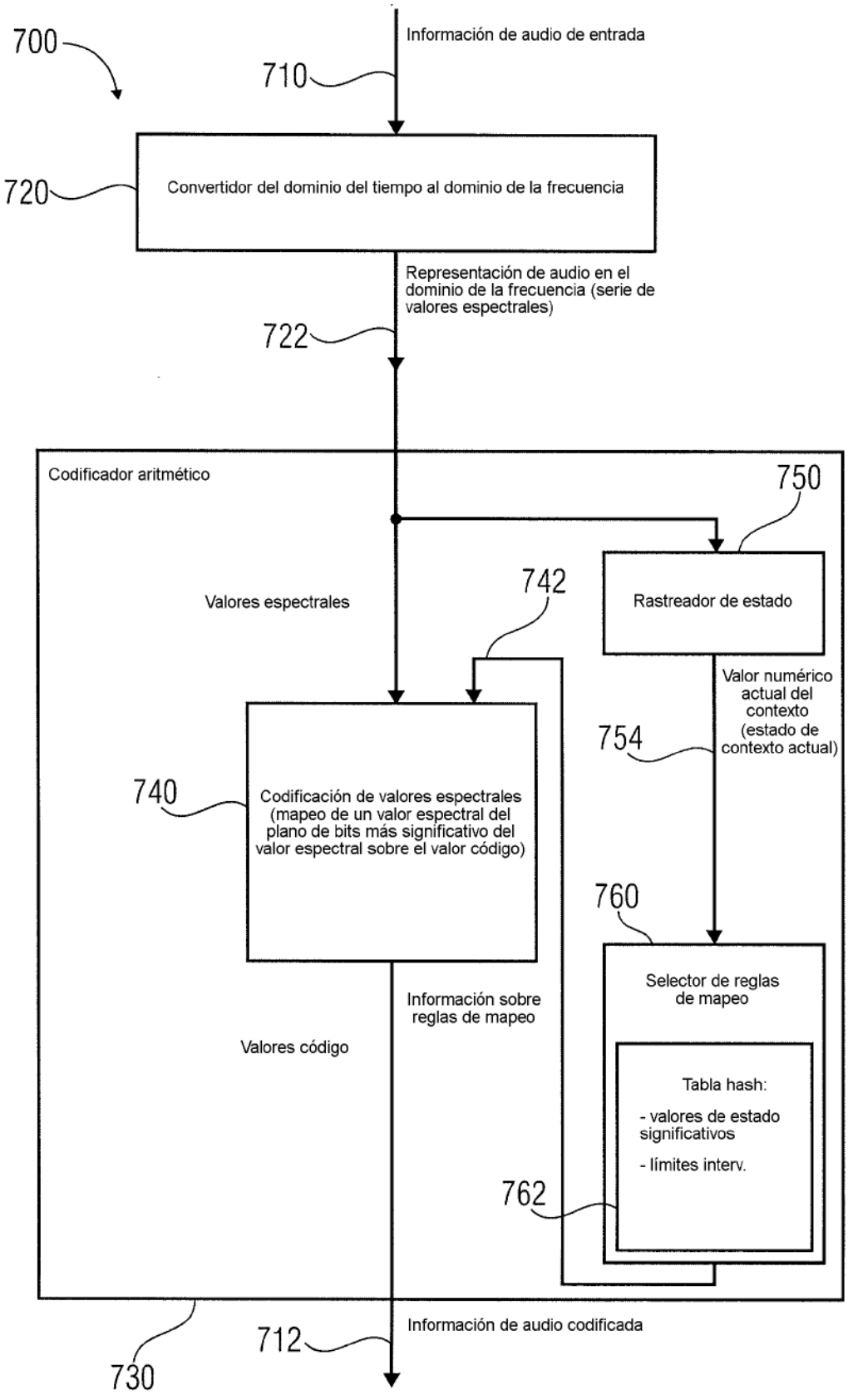

700
710
Información de audio de entrada
720
Convertidor del dominio del tiempo al dominio de la frecuencia
Representación de audio en el dominio de la frecuencia (serie de valores espectrales)
722
Codificador aritmético
750
Valores espectrales
742
Rastreador de estado
Valor numérico actual del contexto (estado de contexto actual)
754
740
Codificación de valores espectrales (mapeo de un valor espectral del plano de bits más significativo del valor espectral sobre el valor código)
760
Selector de reglas de mapeo
Información sobre reglas de mapeo
Valores código
Tabla hash:
- valores de estado significativos
- límites interv.
762
Información de audio codificada
712
730

FIG 7

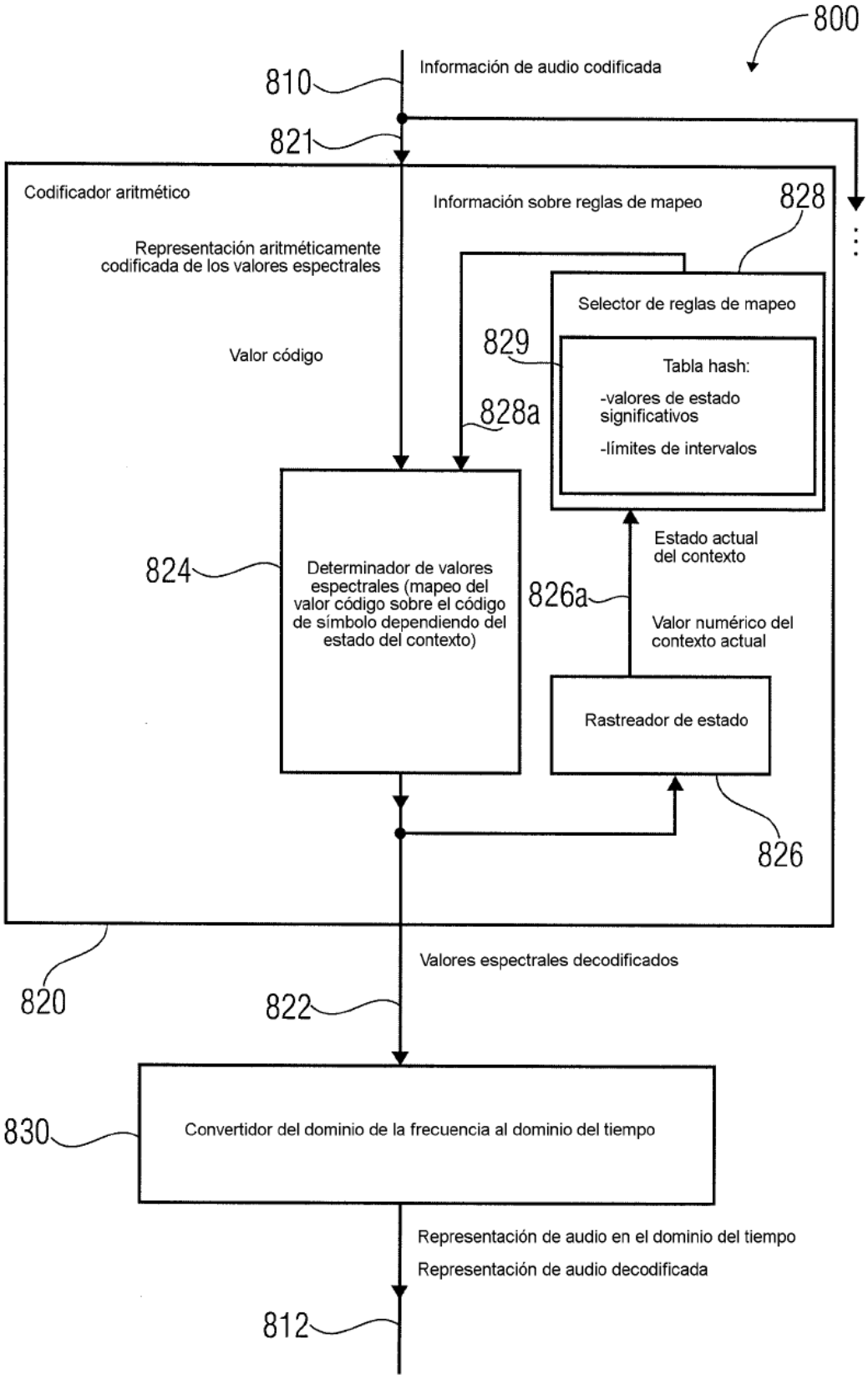

800
810
Información de audio codificada
821
Codificador aritmético
Información sobre reglas de mapeo
828
Representación aritméticamente codificada de los valores espectrales
Selector de reglas de mapeo
829
Valor código
Tabla hash:
-valores de estado significativos
-límites de intervalos
828a
824
Determinador de valores espectrales (mapeo del valor código sobre el código de símbolo dependiendo del estado del contexto)
Estado actual del contexto
826a
Valor numérico del contexto actual
Rastreador de estado
826
820
Valores espectrales decodificados
822
830
Convertidor del dominio de la frecuencia al dominio del tiempo
Representación de audio en el dominio del tiempo
Representación de audio decodificada
812

FIG 8

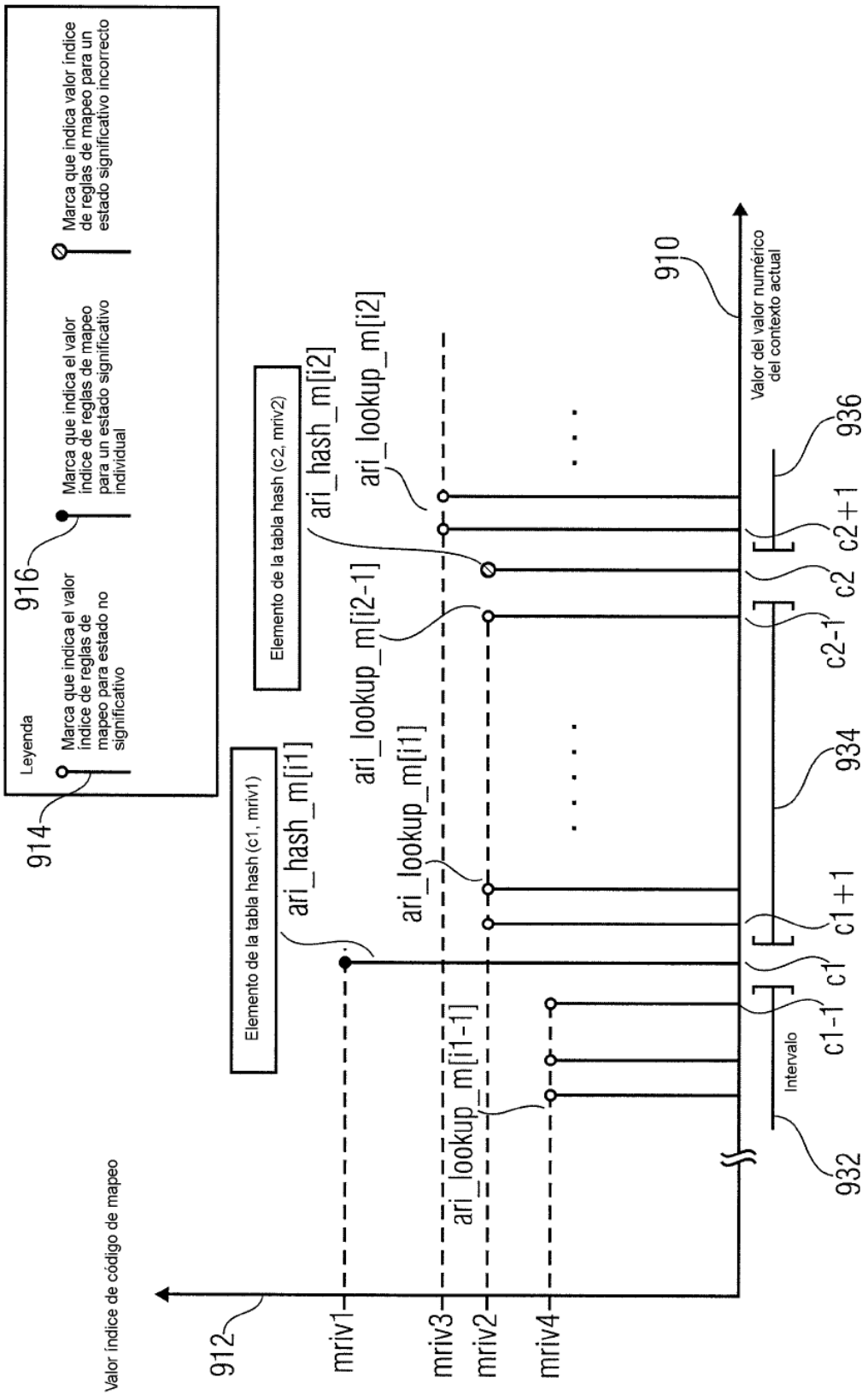

Leyenda
Marca que indica el valor índice de reglas de mapeo para estado no significativo
914
916
Marca que indica el valor índice de reglas de mapeo para un estado significativo individual
Marca que indica valor índice de reglas de mapeo para un estado significativo incorrecto
Elemento de la tabla hash (c1, mriv1)
Elemento de la tabla hash (c2, mriv2)
ari_hash_m[i1]
ari_lookup_m[i1]
ari_lookup_m[i1-1]
ari_hash_m[i2]
ari_lookup_m[i2]
ari_lookup_m[i2-1]
Valor índice de código de mapeo
912
mriv1
mriv3
mriv2
mriv4
910
Valor del valor numérico del contexto actual
c1-1
c1
c1+1
c2-1
c2
c2+1
Intervalo
932
934
936

FIG 9

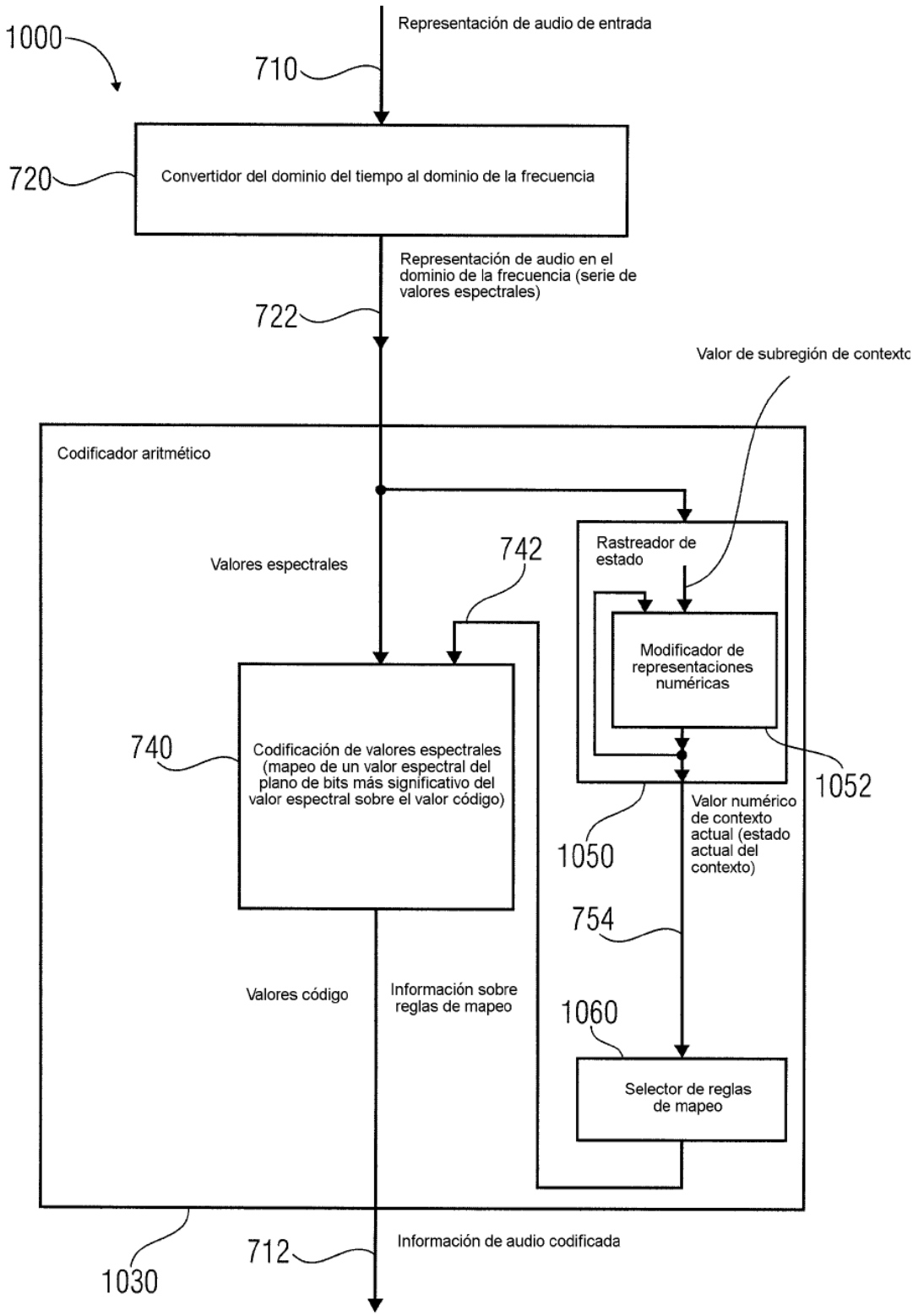

1000
Representación de audio de entrada
710
720
Convertidor del dominio del tiempo al dominio de la frecuencia
Representación de audio en el dominio de la frecuencia (serie de valores espectrales)
722
Valor de subregión de contexto
Codificador aritmético
Valores espectrales
742
Rastreador de estado
Modificador de representaciones numéricas
740
Codificación de valores espectrales (mapeo de un valor espectral del plano de bits más significativo del valor espectral sobre el valor código)
1052
1050
Valor numérico de contexto actual (estado actual del contexto)
754
Valores código
Información sobre reglas de mapeo
1060
Selector de reglas de mapeo
1030
712
Información de audio codificada

FIG 10

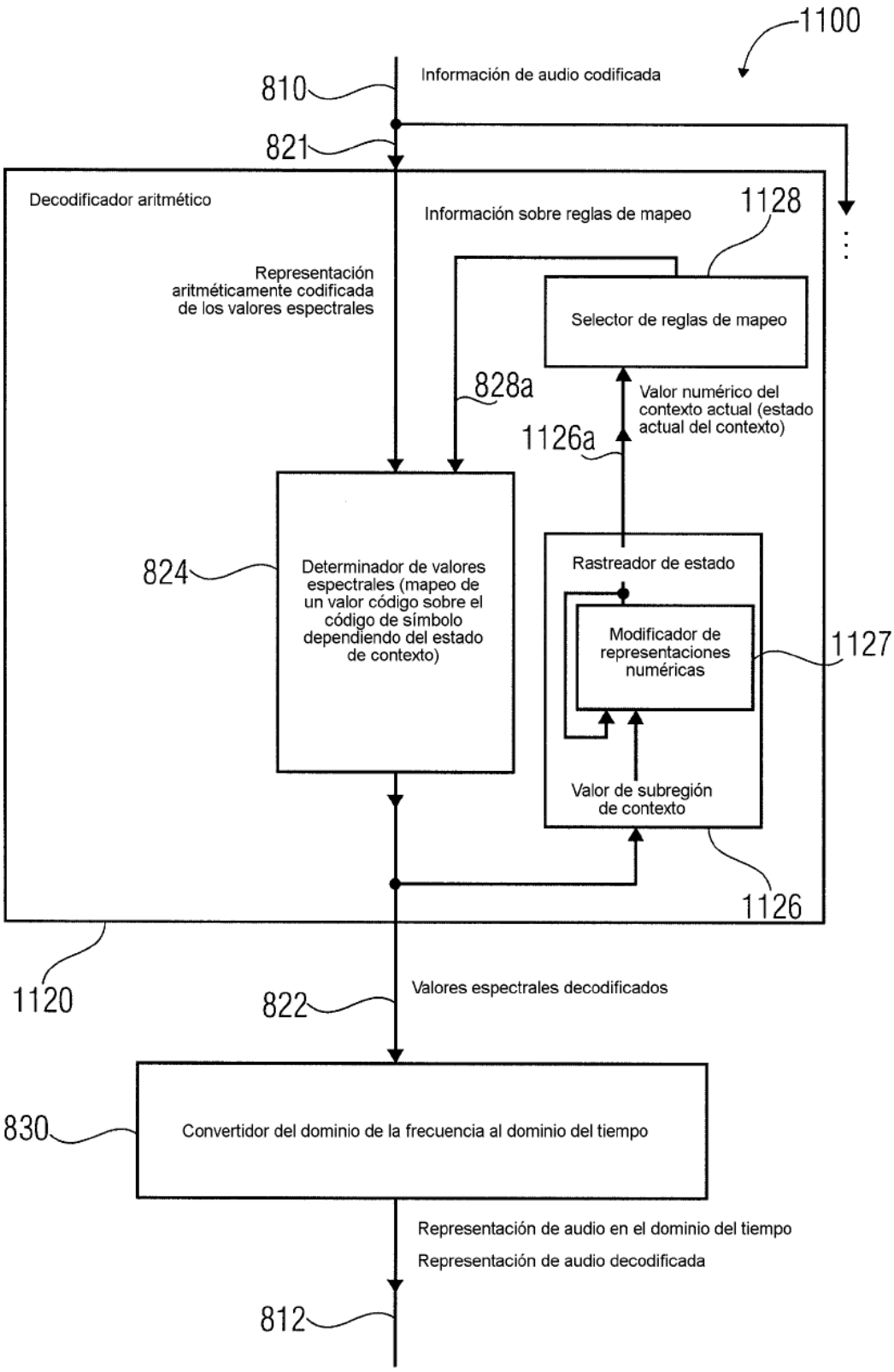

1100
810
Información de audio codificada
821
Decodificador aritmético
Información sobre reglas de mapeo
1128
Representación aritméticamente codificada de los valores espectrales
Selector de reglas de mapeo
828a
Valor numérico del contexto actual (estado actual del contexto)
1126a
824
Determinador de valores espectrales (mapeo de un valor código sobre el código de símbolo dependiendo del estado de contexto)
Rastreador de estado
Modificador de representaciones numéricas
1127
Valor de subregión de contexto
1126
1120
822
Valores espectrales decodificados
830
Convertidor del dominio de la frecuencia al dominio del tiempo
Representación de audio en el dominio del tiempo
Representación de audio decodificada
812

FIG 11

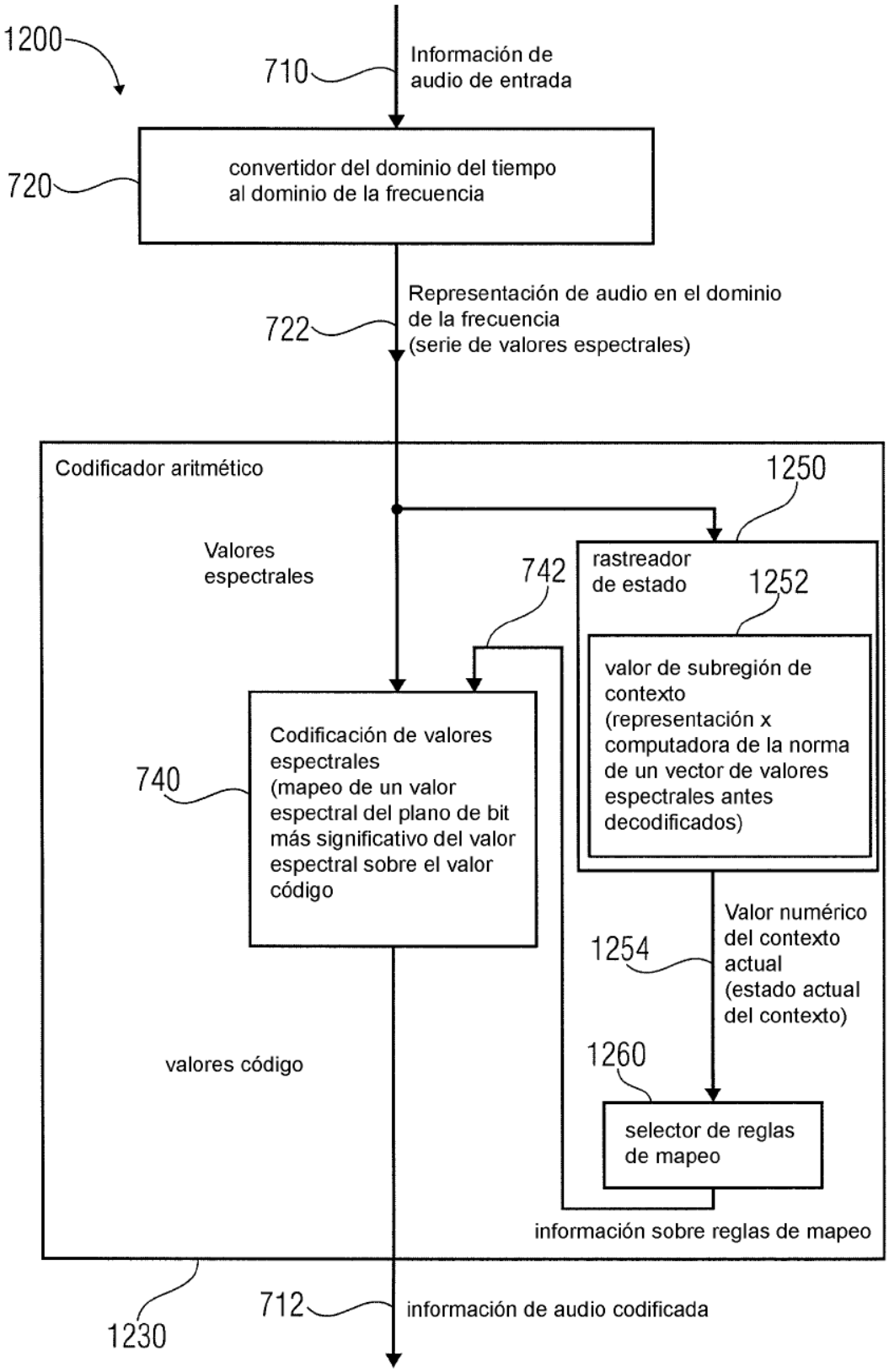

1200
710
Información de audio de entrada
720
convertidor del dominio del tiempo al dominio de la frecuencia
722
Representación de audio en el dominio de la frecuencia (serie de valores espectrales)
Codificador aritmético
1250
Valores espectrales
742
rastreador de estado
1252
valor de subregión de contexto (representación x computadora de la norma de un vector de valores espectrales antes decodificados)
740
Codificación de valores espectrales (mapeo de un valor espectral del plano de bit más significativo del valor espectral sobre el valor código
1254
Valor numérico del contexto actual (estado actual del contexto)
1260
valores código
selector de reglas de mapeo
información sobre reglas de mapeo
1230
712
información de audio codificada

FIG 12

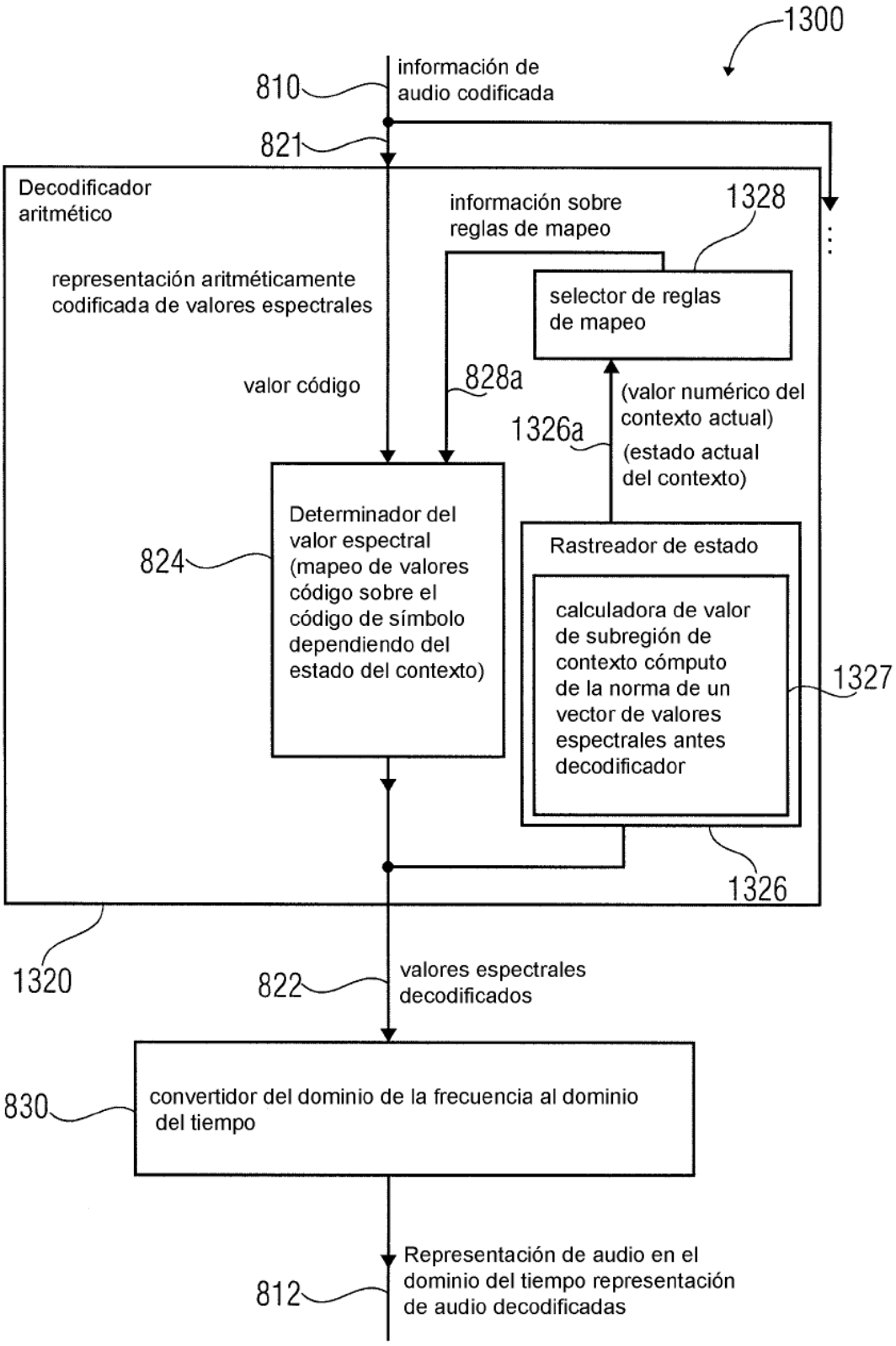

1300
información de audio codificada
810
821
Decodificador aritmético
información sobre reglas de mapeo
1328
representación aritméticamente codificada de valores espectrales
selector de reglas de mapeo
valor código
828a
(valor numérico del contexto actual)
1326a
(estado actual del contexto)
Determinador del valor espectral (mapeo de valores código sobre el código de símbolo dependiendo del estado del contexto)
824
Rastreador de estado
calculadora de valor de subregión de contexto cómputo de la norma de un vector de valores espectrales antes decodificador
1327
1326
1320
822
valores espectrales decodificados
830
convertidor del dominio de la frecuencia al dominio del tiempo
812
Representación de audio en el dominio del tiempo representación de audio decodificadas

FIG 13

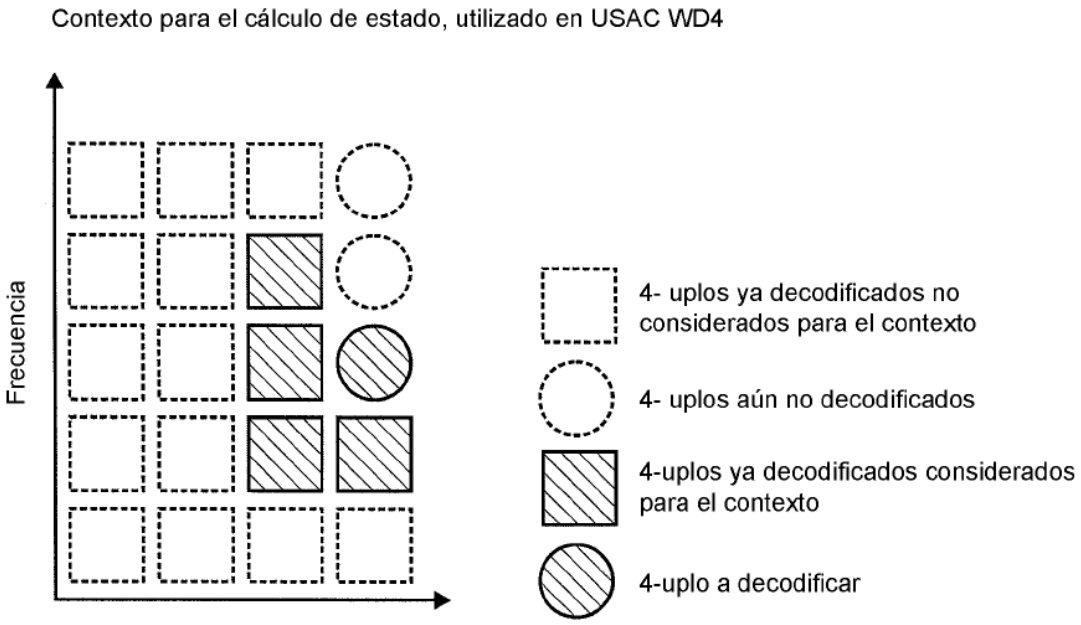

Contexto para el cálculo de estado, utilizado en USAC WD4
Frecuencia
4- uplos ya decodificados no considerados para el contexto
4- uplos aún no decodificados
4-uplos ya decodificados considerados para el contexto
4-uplo a decodificar

FIG 14A

Tablas utilizadas en el Esquema de Codificación Aritmética WD4 de USAC

Nombre de la Tabla	Descripción	Unidad de Datos	Memoria (palabras de 32 bits)
arith_cf_ng_hash[128]	Tabla hash que mapea el contexto a un índice modelo de probabilidades	palabra	128
arith_cf_ng[32][545]	Frecuencias cumulativas de grupos por cada modo de distribución de probabilidades, I	1/2 palabra	8720
egroups[8][8][8][8]	Índice grupal del 4 tuplo	1/2 palabra	2048
dgvectors[4*4096]	Índice de grupo del mapa e índice de elementos al 4-tuplo	1/4 palabra	4096
dgroups[544]	Índice de grupo del mapa al cardinal del grupo y desviación en dgvectors	palabra	544
arith_cf_ne[2701]	Frecuencias cumulativas del símbolo de índice de elementos	1/2 palabra	1350.5
arith_cf_r[16]	Frecuencias cumulativas de los planos de bits menos siginificativos	1/2 palabra	8
Total			**16894.5**

FIG 14B

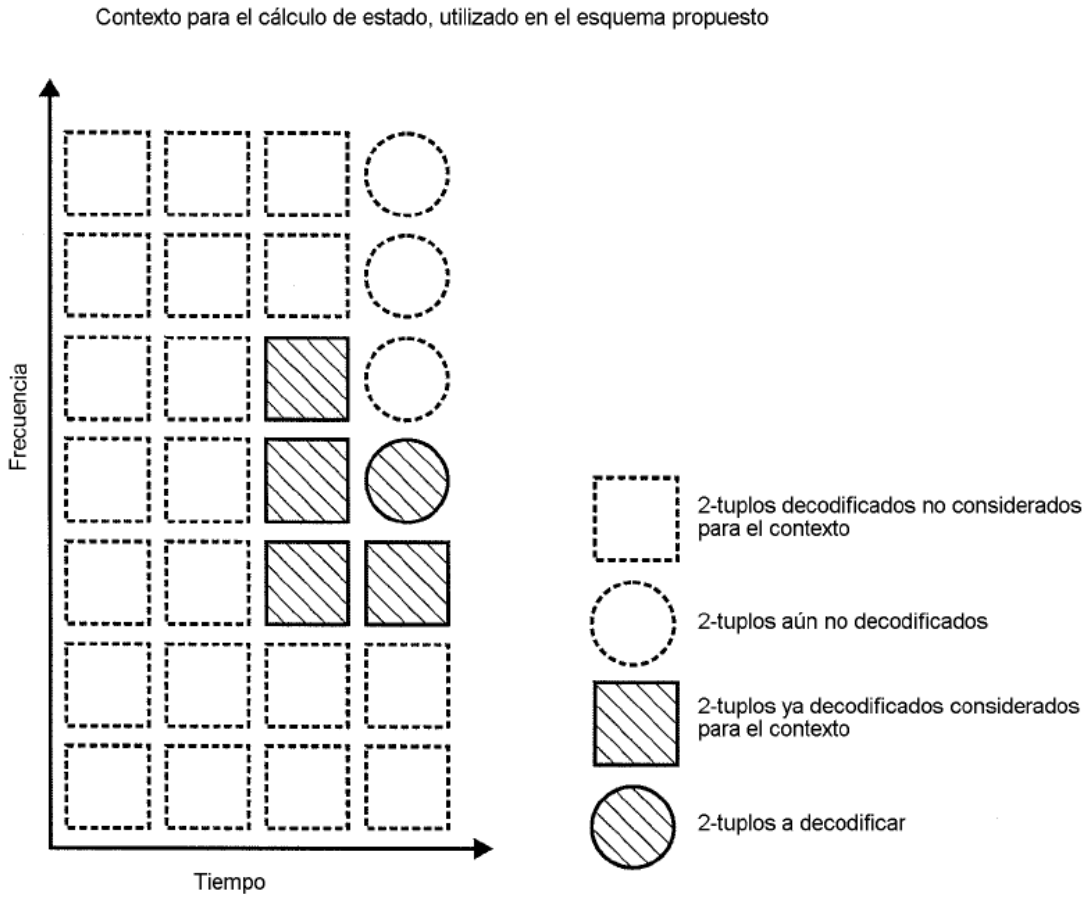

Contexto para el cálculo de estado, utilizado en el esquema propuesto
Frecuencia
Tiempo
2-tuplos decodificados no considerados para el contexto
2-tuplos aún no decodificados
2-tuplos ya decodificados considerados para el contexto
2-tuplos a decodificar

FIG 15A

Tablas utilizadas en el esquema de codificación propuesto

Nombre de la tabla	Descripción	Unidad de datos	Memoria (palabras de 32 bits)
arith_hash[600]	Tabla hash que mapea los estados del contexto a un grupo de estados	1 palabra	600
arith_lookup[600]	Mapeo de tablas de búsqueda	1/2 palabra	150
arith_cf_msb[96][16]	Grupos de estados contra una tabla de frecuencias cumulativas Modelos de frecuencias cumulativas correspondientes al plano de 2 bits más significativos m y el símbolo ARITH_ESCAPE	1/2 palabra	768
Total			**1518**

FIG 15B

Demanda ROM esquema de codificación sin ruido propuesto y en WD4

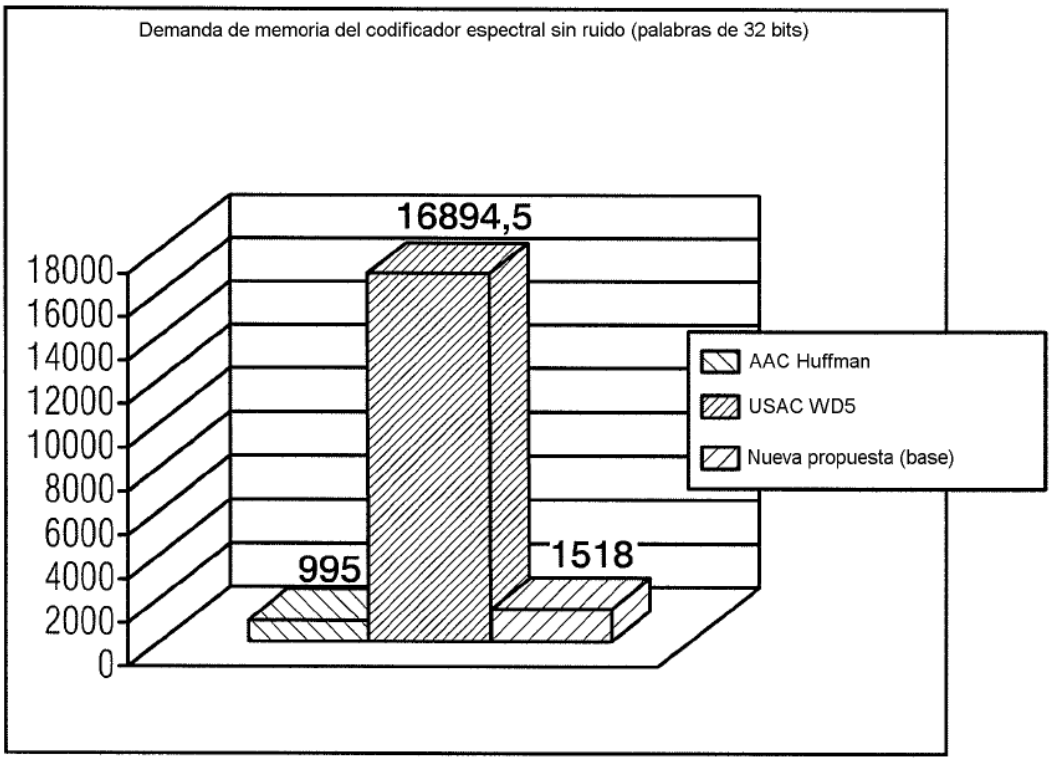


FIG 16A

Demanda total de ROM de datos decodificador USAC, WD4 y esquema propuesto

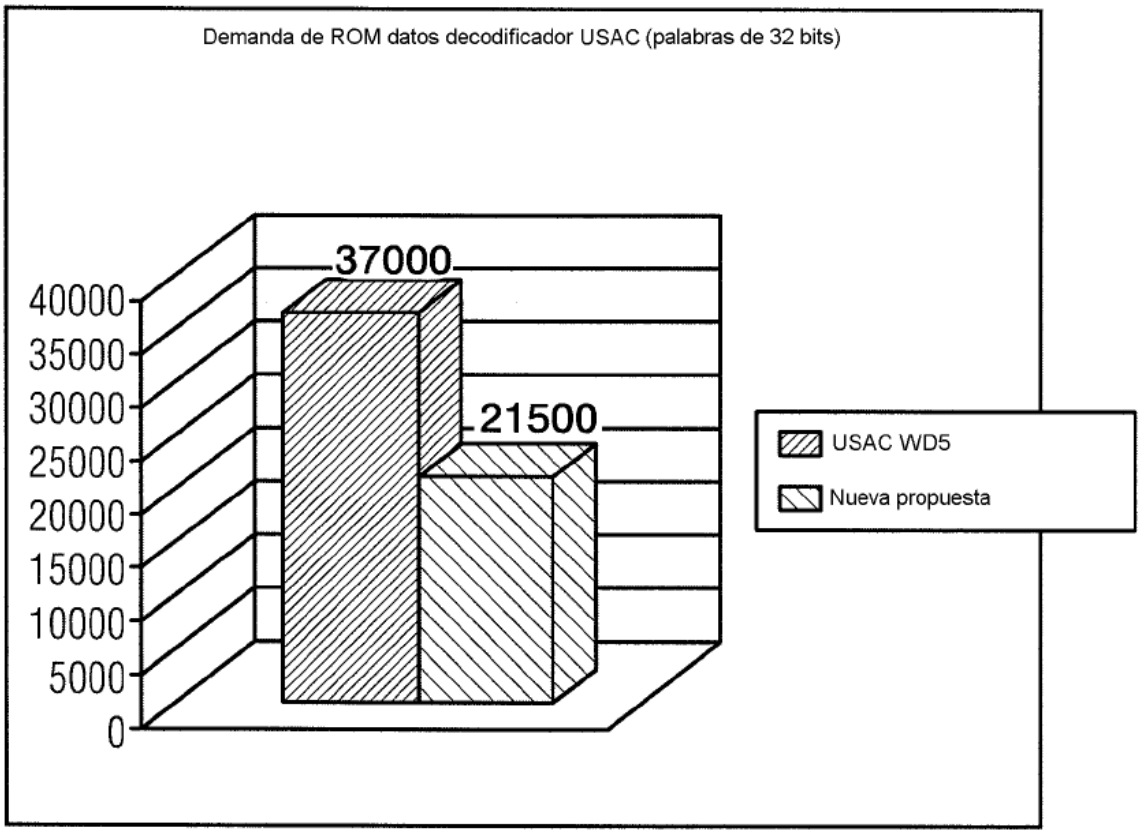


FIG 16B

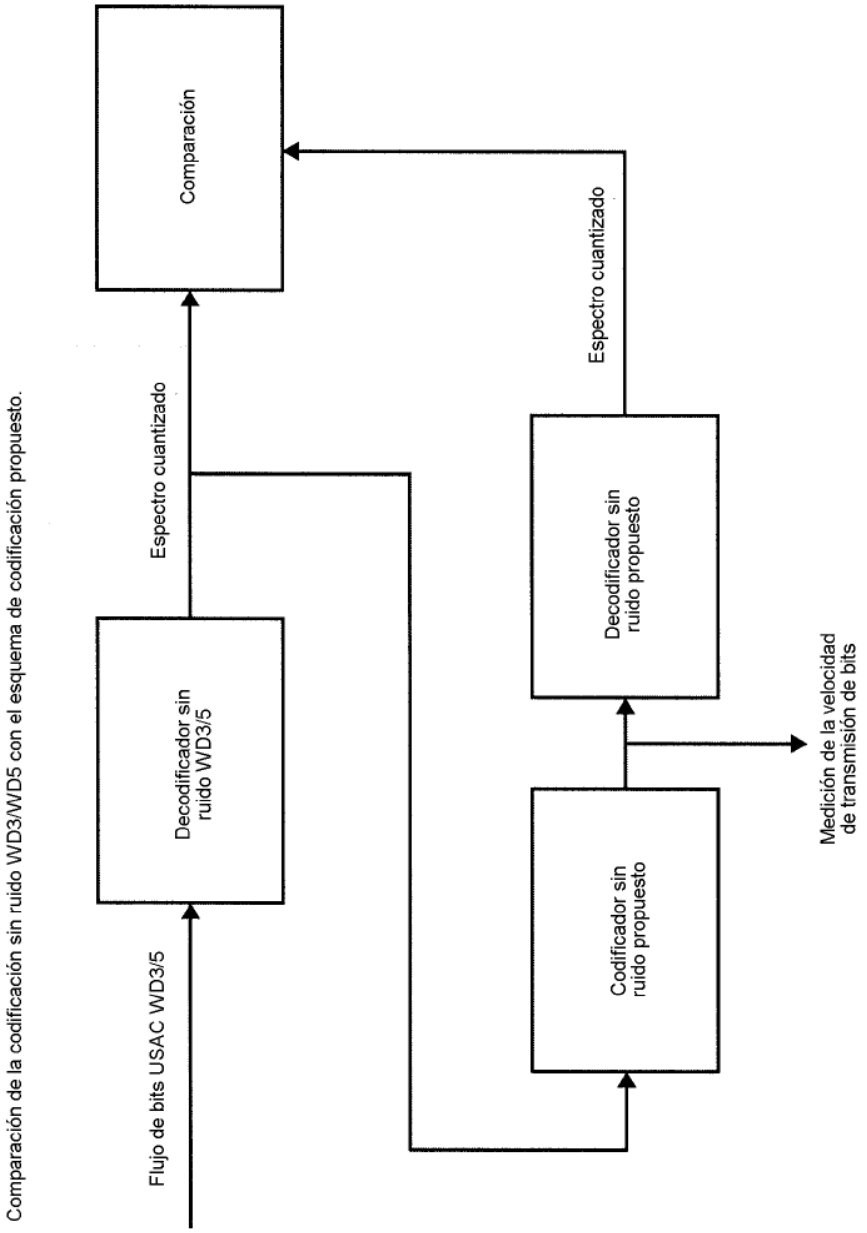

Comparación de la codificación sin ruido WD3/WD5 con el esquema de codificación propuesto.
Flujo de bits USAC WD3/5
Decodificador sin ruido WD3/5
Espectro cuantizado
Comparación
Codificador sin ruido propuesto
Decodificador sin ruido propuesto
Espectro cuantizado
Medición de la velocidad de transmisión de bits

FIG 17

Tabla: Velocidades de transmisión de bits promedio por el codificador aritmético WD3 y la nueva propuesta. Las velocidades de transmisión de bits netas no incluyen los bits por alineamiento de bytes o bits de relleno.

Modo operativo	WD3 (kbit/s)	Nueva propuesta (kbit/s)	Diferencia tras la transcodificación (kbit/s)	Diferencia tras la transcodificación (% de la velocidad de transmisión de bits total)
1, 64s	64.00	62.78	-1.22	-1.90
2, 32s	32.00	31.47	-0.53	-1.66
3, 24s	24.00	23.61	-0.39	-1.64
4, 20s	20.00	19.65	-0.35	-1.74
5, 16s	16.00	15.72	-0.28	-1.75
6, 24m	24.00	23.56	-0.44	-1.83
7, 20m	20.00	19.61	-0.39	-1.95
8, 16m	16.00	15.70	-0.30	-1.87
9, 12m	12.00	11.77	-0.23	-1.92

FIG 18

Tabla: niveles máximo y mínimo de reserva de bits en el caso del codificador aritmético WD3 y la propuesta.

Modo operativo	Control de reserva de bits					
	Nueva propuesta			WD3		
	Min.	Máx.	Promedio	Min.	Máx.	Promedio
1, 64kbps estéreo	3739	9557	8874	2314	9557	7018
2, 32kbps estéreo	2335	4505	4293	582	4505	3529
3, 24kbps estéreo	2184	4704	4472	957	4704	3871
4, 20kbps estéreo	2688	4864	4660	712	4864	3854
5, 16kbps estéreo	2965	5006	4859	724	5006	4234
6, 24kbps mono	2185	4704	4457	1002	4704	3927
7, 20kbps mono	2782	4864	4690	1192	4864	3935
8, 16kbps mono	2916	5006	4905	1434	5006	4450
9, 12kbps mono	3645	5184	5107	2256	5184	4787

FIG 19

Tabla: Números de complejidad promedio correspondientes a la decodificación del flujo de bits de 32 kbit/s de WD3 y a la versión diferente del codificador aritmético.

	WD3	Versión base
PCU (MHz)	0.953	0.823

FIG 20

```
static unsigned short ari_lookup_m[600] = {
0x02,0x01,0x03,0x38,0x3C,0x44,0x45,0x05,
0x07,0x37,0x41,0x08,0x0A,0x07,0x38,0x44,
0x0B,0x14,0x3E,0x3B,0x0C,0x14,0x3E,0x0C,
0x2B,0x5F,0x0F,0x42,0x3B,0x44,0x11,0x36,
0x42,0x41,0x13,0x2B,0x3E,0x3B,0x0C,0x14,
0x3E,0x14,0x2B,0x3E,0x15,0x07,0x3B,0x11,
0x25,0x42,0x41,0x16,0x36,0x3A,0x41,0x25,
0x36,0x3A,0x14,0x3E,0x36,0x42,0x36,0x3A,
0x2B,0x3E,0x42,0x38,0x41,0x17,0x19,0x42,
0x3B,0x44,0x19,0x1C,0x42,0x3B,0x44,0x1D,
0x2B,0x38,0x12,0x2B,0x1E,0x20,0x42,0x41,
0x22,0x36,0x37,0x41,0x24,0x36,0x42,0x41,
0x26,0x07,0x3A,0x2B,0x3E,0x01,0x3E,0x27,
0x07,0x37,0x44,0x3F,0x29,0x42,0x3B,0x44,
0x2A,0x07,0x37,0x41,0x29,0x07,0x38,0x2B,
0x3E,0x36,0x3E,0x39,0x42,0x34,0x07,0x3B,
0x26,0x36,0x42,0x41,0x36,0x42,0x3B,0x36,
0x3A,0x02,0x3A,0x03,0x38,0x07,0x37,0x07,
0x37,0x39,0x3A,0x3F,0x3B,0x41,0x44,0x57,
0x2C,0x07,0x3C,0x03,0x31,0x42,0x3C,0x22,
0x36,0x38,0x2A,0x07,0x27,0x2E,0x07,0x38,
0x44,0x2F,0x07,0x37,0x41,0x03,0x32,0x42,
0x3B,0x44,0x32,0x07,0x38,0x26,0x07,0x3A,
0x26,0x3E,0x02,0x2A,0x07,0x3B,0x33,0x03,
0x42,0x3B,0x44,0x1B,0x36,0x42,0x41,0x32,
0x07,0x3B,0x26,0x36,0x42,0x3B,0x36,0x3E,
0x39,0x42,0x39,0x42,0x3B,0x26,0x07,0x42,
0x41,0x36,0x42,0x37,0x41,0x36,0x42,0x3B,
0x07,0x3A,0x03,0x3B,0x07,0x42,0x3C,0x39,
0x42,0x3C,0x07,0x38,0x07,0x38,0x37,0x38,
0x3C,0x41,0x44,0x57,0x3B,0x3A,0x33,0x37,
0x33,0x03,0x02,0x33,0x07,0x3C,0x33,0x07,
0x3B,0x2A,0x34,0x42,0x41,0x34,0x07,0x38,
0x36,0x3E,0x26,0x07,0x3C,0x39,0x42,0x41,
0x33,0x36,0x42,0x3C,0x26,0x07,0x42,0x41,
0x07,0x38,0x07,0x3A,0x39,0x37,0x39,0x42,
0x3B,0x26,0x07,0x37,0x41,0x01,0x07,0x42,
0x3C,0x39,0x42,0x3B,0x03,0x3F,0x03,0x3B,
```

FIG 21(1)

```
0x39,0x42,0x3B,0x39,0x37,0x3C,0x39,0x37,
0x3C,0x03,0x38,0x3A,0x3B,0x3C,0x41,0x44,
0x57,0x03,0x33,0x03,0x26,0x37,0x34,0x42,
0x41,0x39,0x3F,0x34,0x42,0x39,0x37,0x39,
0x42,0x3B,0x26,0x07,0x37,0x41,0x07,0x3B,
0x07,0x3A,0x03,0x42,0x41,0x39,0x42,0x3C,
0x39,0x42,0x3C,0x07,0x3F,0x03,0x3B,0x03,
0x3B,0x39,0x37,0x3C,0x02,0x42,0x3C,0x03,
0x3D,0x40,0x3C,0x41,0x44,0x57,0x03,0x39,
0x3D,0x03,0x3B,0x39,0x42,0x41,0x39,0x3A,
0x03,0x3F,0x39,0x3F,0x02,0x3E,0x3F,0x02,
0x40,0x3C,0x41,0x44,0x27,0x03,0x03,0x03,
0x3D,0x3F,0x40,0x39,0x41,0x44,0x03,0x3D,
0x42,0x43,0x03,0x3C,0x44,0x03,0x3D,0x40,
0x3C,0x02,0x3C,0x44,0x3D,0x43,0x3C,0x02,
0x41,0x44,0x3D,0x43,0x3C,0x03,0x44,0x3D,
0x43,0x41,0x02,0x44,0x3D,0x41,0x44,0x43,
0x41,0x44,0x43,0x41,0x44,0x03,0x3C,0x44,
0x46,0x47,0x49,0x4B,0x50,0x4D,0x4F,0x0B,
0x56,0x0C,0x0C,0x2B,0x03,0x52,0x0D,0x10,
0x1A,0x56,0x56,0x57,0x58,0x58,0x0C,0x26,
0x01,0x01,0x02,0x3D,0x1A,0x1E,0x56,0x5A,
0x5B,0x5B,0x00,0x27,0x15,0x0C,0x39,0x39,
0x26,0x01,0x02,0x02,0x3D,0x57,0x57,0x27,
0x57,0x00,0x39,0x39,0x02,0x02,0x03,0x27,
0x27,0x02,0x27,0x02,0x02,0x3D,0x5E,0x5D,
0x5F,0x5E,0x5D,0x5D,0x5D,0x5D,0x5C,0x5C,
0x5F,0x5D,0x5D,0x5D,0x5E,0x5D,0x5D,0x5C,
0x5C,0x5C,0x5C,0x5F,0x5D,0x5D,0x5D,0x5E,
0x5D,0x5C,0x5C,0x5E,0x5C,0x5E,0x5C,0x5C,
0x5F,0x5D,0x5C,0x5D,0x5F,0x5D,0x5D,0x5F,
0x5E,0x5F,0x5D,0x5F,0x5D,0x5F,0x5F,0x5D,
0x5F,0x5D,0x5F,0x5D,0x5F,0x5F,0x5F,0x5F,
0x5F,0x5F,0x5F,0x5F,0x44,0x5F,0x5E,0x5D,
0x5D,0x5C,0x5C,0x5D,0x5C,0x5C,0x5C,0x5F,
0x5E,0x5D,0x5E,0x5D,0x5E,0x5D,0x5E,0x5F,
0x5E,0x5F,0x5E,0x5C,0x5E,0x5C,0x5E,0x5E,

};
```

FIG 21(2)

```
static unsigned long ari_hash_m[600] = {
0x00000100UL,0x00000302UL,0x0000053AUL,0x0000073BUL,0x00000A41UL,0x00000F44UL,
0x00111104UL,0x00111306UL,
0x00111542UL,0x0011173BUL,0x00111F44UL,0x00112209UL,0x0011242BUL,0x0011263EUL,
0x0011293CUL,0x00113105UL,
0x0011330CUL,0x0011352BUL,0x0011383AUL,0x00113F44UL,0x0011440CUL,0x0011462BUL,
0x00114F41UL,0x0011530CUL,
0x0012110DUL,0x0012120EUL,0x00121436UL,0x00121637UL,0x00121941UL,0x00122110UL,
0x00122312UL,0x00122507UL,
0x00122738UL,0x00123156UL,0x00123314UL,0x00123507UL,0x0012373AUL,0x00123F44UL,
0x00124212UL,0x0012452BUL,
0x00124F44UL,0x00125314UL,0x0012762BUL,0x00131121UL,0x00131323UL,0x00131542UL,
0x0013211DUL,0x00132216UL,
0x00132436UL,0x00132738UL,0x0013311DUL,0x00133329UL,0x00133507UL,0x00133838UL,
0x00134115UL,0x0013432BUL,
0x0013463EUL,0x00134F44UL,0x0013532BUL,0x0013FF3BUL,0x00142307UL,0x00143126UL,
0x00143407UL,0x00143E44UL,
0x00144307UL,0x0014FF3BUL,0x0015633EUL,0x0017863BUL,0x0021005AUL,0x00211218UL,
0x00211407UL,0x00211637UL,
0x00211941UL,0x0021211AUL,0x0021221BUL,0x00212436UL,0x00212637UL,0x00212941UL,
0x00213118UL,0x00213320UL,
0x00213507UL,0x00213F44UL,0x00214314UL,0x00220001UL,0x0022121FUL,0x00221407UL,
0x0022173BUL,0x00222121UL,
0x00222323UL,0x00222542UL,0x0022283BUL,0x0022311DUL,0x00223325UL,0x00223507UL,
0x00223737UL,0x00224115UL,
0x00224436UL,0x0022463EUL,0x00224F44UL,0x00225436UL,0x00225F44UL,0x0022622BUL,
0x0023112DUL,0x00231330UL,
0x00231542UL,0x0023183CUL,0x0023212DUL,0x00232228UL,0x00232407UL,0x00232637UL,
0x00232941UL,0x00233115UL,
0x0023332BUL,0x00233542UL,0x0023383BUL,0x0023412AUL,0x00234336UL,0x00234642UL,
0x00234F44UL,0x00235407UL,
0x00235F44UL,0x0023653EUL,0x0023FF3BUL,0x00241307UL,0x00241F44UL,0x00242336UL,
0x00242542UL,0x00242F44UL,
0x00243234UL,0x00243407UL,0x00243738UL,0x00244126UL,0x00244307UL,0x0024463AUL,
0x00244F44UL,0x00245442UL,
0x00245F44UL,0x00246236UL,0x0024FF3CUL,0x00252442UL,0x00252F44UL,0x00253303UL,
0x00253F44UL,0x00254303UL,
0x00254F44UL,0x00255303UL,0x0025FF41UL,0x0026733EUL,0x0027FF44UL,0x002AF203UL,
0x002FFF44UL,0x0031115BUL,
0x00311331UL,0x00311542UL,0x00312121UL,0x0031222DUL,0x00312436UL,0x00312637UL,
0x00312F44UL,0x00313335UL,
```

FIG 22(1)

0x00313507UL,0x00313F44UL,0x00314332UL,0x0031FF3CUL,0x0032112CUL,0x00321335UL,
0x00321542UL,0x0032183CUL,
0x0032212CUL,0x00322330UL,0x00322542UL,0x0032273BUL,0x0032312DUL,0x00323231UL,
0x00323407UL,0x00323637UL,
0x00323A41UL,0x0032412AUL,0x00324407UL,0x00324642UL,0x00324F44UL,0x00325336UL,
0x00325507UL,0x00325F44UL,
0x00326234UL,0x0032FF3CUL,0x00331127UL,0x00331332UL,0x00331542UL,0x0033212EUL,
0x00332334UL,0x00332407UL,
0x00332637UL,0x00332941UL,0x00333115UL,0x00333235UL,0x00333407UL,0x00333738UL,
0x0033412AUL,0x00334336UL,
0x00334637UL,0x00334F44UL,0x00335234UL,0x00335407UL,0x0033573AUL,0x00335F44UL,
0x00336407UL,0x0033FF3CUL,
0x00341307UL,0x00341F44UL,0x00342307UL,0x00342542UL,0x00342F44UL,0x00343234UL,
0x00343442UL,0x0034373BUL,
0x00344126UL,0x00344307UL,0x00344542UL,0x0034483BUL,0x00345126UL,0x00345307UL,
0x00345637UL,0x00345F44UL,
0x00346442UL,0x0034FF3CUL,0x00352442UL,0x00353139UL,0x00353303UL,0x00353637UL,
0x00353F44UL,0x00354303UL,
0x00354637UL,0x00354F44UL,0x00355442UL,0x00356102UL,0x00356303UL,0x0035FF41UL,
0x00366203UL,0x0037733DUL,
0x0039E43AUL,0x003BF641UL,0x003FFF44UL,0x00411227UL,0x00411334UL,0x0041212CUL,
0x00412407UL,0x0041312EUL,
0x00413334UL,0x0041FF3CUL,0x00421127UL,0x00421334UL,0x00421542UL,0x00422127UL,
0x00422334UL,0x00422542UL,
0x00422F44UL,0x00423232UL,0x00423407UL,0x0042373BUL,0x00424126UL,0x00424336UL,
0x00424542UL,0x00424F44UL,
0x00425407UL,0x0042FF3CUL,0x00431339UL,0x00431542UL,0x00432133UL,0x00432407UL,
0x0043273BUL,0x00432F44UL,
0x00433234UL,0x00433407UL,0x00433637UL,0x00433F44UL,0x00434234UL,0x00434442UL,
0x00434738UL,0x00435126UL,
0x00435542UL,0x00435F44UL,0x00436442UL,0x0043FF41UL,0x00441303UL,0x00442126UL,
0x00442307UL,0x00442542UL,
0x00442F44UL,0x00443239UL,0x00443442UL,0x0044373BUL,0x00443F44UL,0x00444234UL,
0x00444442UL,0x00444637UL,
0x00444F44UL,0x00445307UL,0x00445637UL,0x00445F44UL,0x00446537UL,0x0044FF41UL,
0x00452442UL,0x00452F44UL,
0x00453303UL,0x00453537UL,0x00453F44UL,0x00454303UL,0x00454638UL,0x00454F44UL,
0x00455303UL,0x00455638UL,
0x00455F44UL,0x00456437UL,0x0045FF41UL,0x00466203UL,0x00478438UL,0x0049FF44UL,
0x004CF741UL,0x004FFF44UL,

FIG 22(2)

0x00511226UL,0x00512127UL,0x00512339UL,0x00521127UL,0x00521339UL,0x00522127UL,
0x00522339UL,0x00522637UL,
0x00523126UL,0x00523403UL,0x00524126UL,0x00524307UL,0x00531126UL,0x00531307UL,
0x00532126UL,0x00532307UL,
0x00532537UL,0x00532F44UL,0x00533239UL,0x00533442UL,0x0053373BUL,0x00534126UL,
0x00534542UL,0x00534F44UL,
0x00535442UL,0x0053FF3CUL,0x00542303UL,0x00542638UL,0x00543102UL,0x00543307UL,
0x00543638UL,0x00543F44UL,
0x00544307UL,0x00544537UL,0x00545102UL,0x00545303UL,0x0054FF41UL,0x00552437UL,
0x00552F44UL,0x00553437UL,
0x00553F44UL,0x00554303UL,0x0055463BUL,0x00554F44UL,0x00555307UL,0x00555537UL,
0x00556102UL,0x00556342UL,
0x00566203UL,0x00577342UL,0x0059733AUL,0x005CF53CUL,0x005FFF44UL,0x00611239UL,
0x00621127UL,0x00623307UL,
0x00631126UL,0x00632442UL,0x00632F44UL,0x00633303UL,0x00633638UL,0x00634102UL,
0x00634303UL,0x0063FF41UL,
0x00642303UL,0x00642F44UL,0x00643303UL,0x00643F44UL,0x00644207UL,0x00655203UL,
0x00665F44UL,0x00666303UL,
0x00677203UL,0x0069A53BUL,0x006DF841UL,0x006FFF44UL,0x00711239UL,0x00721127UL,
0x00722239UL,0x00733239UL,
0x00744437UL,0x00755203UL,0x00776F44UL,0x00777437UL,0x007BF33FUL,0x007FFF44UL,
0x00822239UL,0x00833203UL,
0x00854540UL,0x00888102UL,0x00888538UL,0x008BF23DUL,0x008FFF44UL,0x00922239UL,
0x0094333DUL,0x0096533DUL,
0x00998F44UL,0x00999303UL,0x009BF53BUL,0x009FFF44UL,0x00A44203UL,0x00A6633FUL,
0x00AA9F44UL,0x00AAA303UL,
0x00ADF33DUL,0x00AFFF44UL,0x00B33203UL,0x00B55440UL,0x00BBAC44UL,0x00BBB53BUL,
0x00BFFF44UL,0x00C33203UL,
0x00C6423DUL,0x00CCBF44UL,0x00CCC303UL,0x00CFFF44UL,0x00D4423DUL,0x00DDD303UL,
0x00DFFF44UL,0x00E6453CUL,
0x00EEE342UL,0x00EFFF44UL,0x00F55343UL,0x00F7FF44UL,0x00FFEF44UL,0x00FFF33DUL,
0x00FFF744UL,0x01000145UL,
0x01011147UL,0x01111248UL,0x0111224AUL,0x0111324DUL,0x0112114CUL,0x0112214EUL,
0x01123108UL,0x01131150UL,
0x01132150UL,0x01133150UL,0x01141126UL,0x01144100UL,0x01211151UL,0x01212153UL,
0x01213108UL,0x01221154UL,
0x01222155UL,0x01223117UL,0x01224212UL,0x01231215UL,0x01232215UL,0x01233215UL,
0x01241126UL,0x01242239UL,
0x0124322BUL,0x01251103UL,0x01255100UL,0x01311159UL,0x01312155UL,0x01313117UL,
0x01315201UL,0x0132122CUL,

FIG 22(3)

```
0x0132222CUL,0x0132321DUL,0x01331157UL,0x01332158UL,0x01333150UL,0x01341126UL,
0x01342127UL,0x01343100UL,
0x01344100UL,0x01351103UL,0x01355100UL,0x01369100UL,0x0141115AUL,0x01421227UL,
0x01422227UL,0x01431226UL,
0x01432226UL,0x01441127UL,0x01442127UL,0x01443100UL,0x01444100UL,0x01458100UL,
0x01511157UL,0x01531239UL,
0x01555100UL,0x01611157UL,0x01631239UL,0x01777100UL,0x01D33102UL,0x01FFF203UL,
0x0200055DUL,0x0200085DUL,
0x02000F5FUL,0x0211155DUL,0x02111F44UL,0x02112F5FUL,0x02121F44UL,0x02122F44UL,
0x02133F5FUL,0x0217FF5FUL,
0x021FFF44UL,0x02211F44UL,0x02212F44UL,0x02221F44UL,0x0222245EUL,0x02222F5FUL,
0x02223F5FUL,0x02232F5FUL,
0x02233F5FUL,0x02243F5FUL,0x022BFF5FUL,0x022FFF44UL,0x02322F44UL,0x02323F5FUL,
0x02332F5FUL,0x0233355CUL,
0x02333F5FUL,0x02334F5FUL,0x0233FF5CUL,0x02343F5FUL,0x0234FF5CUL,0x02353F5FUL,
0x02364F5FUL,0x0239655CUL,
0x023FFF44UL,0x02433F5FUL,0x0246445CUL,0x024A955DUL,0x024FFF44UL,0x0257435EUL,
0x025AC55CUL,0x025FFF44UL,
0x0267565DUL,0x026FFF44UL,0x0278655DUL,0x027FFF44UL,0x0288875DUL,0x028CC95DUL,
0x028FFF44UL,0x029A965DUL,
0x029FFF44UL,0x02ABA65DUL,0x02AFFF44UL,0x02BAFF5FUL,0x02BFFF44UL,0x02CCC45DUL,
0x02CFFF44UL,0x02DFFF44UL,
0x02EEE65DUL,0x02EFFF44UL,0x02F7335EUL,0x02FF0044UL,0x02FFEF44UL,0x02FFFF44UL,
0x0400045EUL,0x04000F5FUL,
0x04111F5DUL,0x04234F5DUL,0x042DFF5CUL,0x04343F5DUL,0x043FFF5FUL,0x044FFF5FUL,
0x045ED75CUL,0x045FFF5FUL,
0x046DFF5FUL,0x046FFF5FUL,0x047B5C5CUL,0x047FFF5FUL,0x048B435EUL,0x048FFF5FUL,
0x0499FF5CUL,0x04EFFF5FUL,
0x04FDFE5DUL,0x04FFFF5FUL,0x0600075EUL,0x06DFFF5FUL,0x06FFFF5FUL,0x08BFFF5CUL,
0x08FFFF5CUL,0x7FFFFFFF4FUL,

};
```

FIG 22(4)

```
unsigned short ari_cf_m [96][17] = {
{ 7529, 5263, 5173, 5162, 2636, 825, 674, 653, 511, 281, 210, 195, 173, 130, 105,
96,
  0
},
{ 10351, 7392, 7203, 7178, 4327, 1620, 1279, 1230, 1008, 606, 436, 399, 362, 288, 233,
212,
  0
},
{ 12505, 9566, 9216, 9157, 6631, 3220, 2541, 2418, 2086, 1404, 1024, 922, 858, 721,
597, 535,
  0
},
{ 14710,12600,11956,11801,10114, 7056, 5723, 5381, 4870, 3724, 2870, 2566, 2437, 2122,
1809, 1609,
  0
},
{ 4186, 2623, 2608, 2607, 939, 109, 86, 84, 63, 23, 14, 13, 11, 8, 5, 4,
  0
},
{ 7310, 4598, 4547, 4544, 2079, 354, 264, 259, 204, 90, 42, 38, 34, 25, 14, 11,
  0
},
{ 9998, 6785, 6641, 6630, 4087, 1058, 770, 746, 621, 339, 166, 143, 131, 104, 67,
49,
  0
},
{ 13801,11125,10550,10472, 8498, 5030, 3694, 3509, 3101, 2141, 1247, 1038, 975, 821,
623, 440,
  0
},
{ 5784, 3557, 3523, 3521, 1359, 164, 122, 119, 85, 29, 16, 14, 12, 8, 5, 4,
  0
},
{ 7617, 4716, 4649, 4645, 2129, 361, 258, 252, 191, 80, 39, 34, 30, 22, 13, 10,
  0
},
{ 9553, 6223, 6064, 6052, 3524, 912, 643, 620, 501, 262, 132, 112, 102, 79, 52, 39,
  0
},
{ 8423, 5300, 5186, 5179, 2629, 539, 375, 362, 276, 119, 61, 53, 46, 33, 21, 17,
  0
},
{ 9570, 6162, 5952, 5936, 3574, 1016, 702, 670, 538, 282, 152, 129, 117, 90, 63,
50,
  0
},
```

2310

2312

FIG 23(1)

```
{  7355, 4678, 4630, 4628, 1834,  252,  190,  187,  124,   38,   21,   19,   16,   10,    6,    5,
   0
},
{  8916, 5717, 5627, 5622, 2647,  471,  340,  332,  242,   97,   46,   41,   36,   25,   15,   12,
   0
},
{ 10839, 7394, 7198, 7184, 4273, 1169,  832,  803,  643,  327,  167,  143,  130,  100,   66,
49,
   0
},
{  8036, 5132, 5045, 5040, 2207,  350,  250,  243,  167,   56,   31,   28,   23,   15,   10,    8,
   0
},
{  9663, 6256, 6110, 6101, 3064,  633,  441,  426,  311,  127,   65,   57,   49,   34,   22,   18,
   0
},
{ 11237, 7629, 7361, 7339, 4460, 1302,  898,  859,  675,  333,  178,  152,  136,  104,   73,
57,
   0
},
{ 10436, 6930, 6718, 6703, 3735,  902,  619,  595,  446,  189,   98,   85,   74,   51,   34,   28,
   0
},
{ 11678, 7933, 7574, 7541, 4826, 1560, 1057, 1002,  796,  406,  218,  184,  165,  124,   88,
70,
   0
},
{ 10597, 7265, 7057, 7039, 3904, 1071,  768,  738,  541,  227,  127,  111,   93,   60,   40,
34,
   0
},
{ 11298, 7892, 7596, 7566, 4416, 1276,  873,  832,  611,  261,  141,  122,  103,   70,   47,
40,
   0
},
{  6435, 4238, 4210, 4208, 1558,  216,  174,  172,  116,   39,   24,   22,   18,   11,    7,    6,
   0
},
{  8744, 5744, 5671, 5667, 2636,  496,  375,  368,  273,  113,   58,   52,   45,   32,   19,   15,
   0
},
{ 10899, 7632, 7457, 7444, 4488, 1270,  942,  914,  737,  383,  204,  177,  161,  122,   80,
62,
   0
},
```

FIG 23(2)

```
{ 7929, 5176, 5119, 5116, 2169, 307, 230, 226, 155, 51, 28, 25, 21, 13, 8, 7,
  0
},
{ 9520, 6304, 6204, 6199, 3044, 585, 424, 415, 304, 118, 59, 52, 45, 31, 19, 15,
  0
},
{ 11187, 7805, 7600, 7586, 4554, 1292, 928, 896, 703, 342, 178, 153, 138, 103, 68,
52,
  0
},
{ 10371, 7037, 6881, 6871, 3721, 860, 613, 596, 440, 176, 88, 77, 67, 46, 28, 23,
  0
},
{ 8562, 5804, 5738, 5734, 2472, 467, 374, 369, 233, 76, 48, 44, 34, 20, 13, 11,
  0
},
{ 10078, 6838, 6713, 6705, 3407, 782, 591, 578, 407, 164, 92, 83, 69, 46, 29, 24,
  0
},
{ 11694, 8342, 8108, 8089, 4999, 1559, 1143, 1105, 858, 424, 240, 210, 185, 135, 92,
73,
  0
},
{ 9115, 6191, 6084, 6077, 2924, 604, 455, 445, 304, 109, 64, 58, 47, 29, 20, 17,
  0
},
{ 10786, 7434, 7256, 7244, 3968, 1006, 734, 712, 517, 213, 118, 104, 88, 60, 40,
34,
  0
},
{ 12216, 8840, 8535, 8507, 5492, 1860, 1342, 1289, 1007, 505, 289, 249, 221, 163, 115,
94,
  0
},
{ 11473, 8085, 7843, 7825, 4602, 1304, 931, 898, 669, 283, 153, 135, 116, 80, 53,
44,
  0
},
{ 12650, 9279, 8908, 8872, 5924, 2160, 1539, 1470, 1162, 585, 329, 282, 251, 184, 130,
107,
  0
},
{ 12597, 9310, 8859, 8806, 6019, 2457, 1741, 1643, 1310, 684, 401, 343, 304, 225, 164,
137,
  0
},
```

FIG 23(3)

```
{ 11509, 8255, 8030, 8012, 4706, 1489, 1135, 1103, 793, 347, 212, 192, 154, 98, 68,
59,
  0
},
{ 11946, 8648, 8370, 8344, 5106, 1628, 1185, 1142, 842, 366, 206, 183, 153, 100, 68,
57,
  0
},
{ 13088, 9856, 9444, 9397, 6468, 2535, 1831, 1745, 1384, 700, 407, 352, 310, 224, 160,
132,
  0
},
{ 12262, 9021, 8683, 8647, 5519, 1929, 1405, 1342, 1024, 481, 280, 245, 210, 146, 104,
89,
  0
},
{ 13353,10225, 9742, 9678, 6908, 2945, 2133, 2017, 1619, 875, 524, 450, 401, 298, 218,
183,
  0
},
{ 10055, 6990, 6879, 6872, 3544, 882, 687, 675, 490, 209, 121, 112, 95, 63, 41, 36,
  0
},
{ 10647, 7427, 7278, 7270, 3911, 968, 723, 705, 513, 203, 109, 98, 83, 55, 34, 28,
  0
},
{ 11221, 8027, 7861, 7851, 4397, 1264, 981, 961, 689, 293, 174, 159, 132, 83, 54,
46,
  0
},
{ 11700, 8429, 8209, 8192, 4825, 1451, 1079, 1049, 767, 327, 186, 167, 140, 92, 61,
52,
  0
},
{ 12949, 9762, 9414, 9379, 6351, 2418, 1786, 1717, 1351, 686, 398, 345, 301, 216, 152,
125,
  0
},
{ 12208, 8979, 8703, 8682, 5437, 1780, 1303, 1261, 955, 422, 231, 204, 175, 120, 77,
63,
  0
},
{ 13277,10148, 9741, 9698, 6807, 2800, 2047, 1960, 1563, 808, 463, 398, 352, 255, 174,
140,
  0
},
```

FIG 23(4)

```
{ 12426, 9284, 8976, 8947, 5753, 2124, 1609, 1555, 1166,  548,  333,  298,  247,  162,  113,
99,
  0
},
{ 13492,10473,10042, 9988, 7177, 3133, 2347, 2244, 1797,  970,  594,  520,  457,  331,
241,  204,
  0
},
{ 12407, 9334, 9013, 8977, 5920, 2276, 1708, 1642, 1284,  642,  392,  348,  299,  211,  153,
132,
  0
},
{ 13579,10631,10182,10121, 7466, 3396, 2542, 2420, 1975, 1112,  691,  604,  542,  408,
300,  256,
  0
},
{ 15415,14163,13591,13335,12111, 9885, 8640, 8070, 7541, 6363, 5413, 4919, 4688, 4157,
3662, 3364,
  0
},
{ 15645,14640,14124,13838,12910,11112, 9969, 9338, 8909, 7923, 7060, 6507, 6315, 5841,
5365, 5003,
  0
},
{ 13803,11130,10691,10636, 7807, 3900, 3070, 2956, 2343, 1300,  866,  781,  667,  462,
337,  292,
  0
},
{ 15381,14112,13568,13348,12083, 9783, 8595, 8105, 7565, 6347, 5408, 4945, 4704, 4158,
3671, 3377,
  0
},
{ 15595,14555,14062,13813,12830,10944, 9820, 9259, 8813, 7769, 6872, 6351, 6149, 5655,
5160, 4798,
  0
},
{ 15855,15124,14719,14487,13812,12440,11517,10975,10628, 9802, 9036, 8493, 8312,
7869,7383, 6992,
  0
},
{ 15484,14324,13800,13578,12460,10325, 9149, 8632, 8118, 6954, 6030, 5548, 5320, 4797,
4296, 3961,
  0
},
{ 15407,14130,13600,13402,12107, 9730, 8527, 8059, 7475, 6205, 5245, 4803, 4563, 4022,
3557, 3260,
  0
},
```

FIG 23(5)

```
{ 15475,14322,13830,13621,12482,10384, 9270, 8779, 8266, 7125, 6200, 5730, 5493, 4962,
4469, 4140,
  0
},
{ 15596,14601,14153,13948,12948,11040, 9997, 9517, 9048, 7949, 7054, 6560, 6325, 5785,
5280, 4913,
  0
},
{ 15871,15158,14789,14602,13921,12540,11660,11198,10847,10009, 9224, 8719, 8532,
8078, 7594, 7190,
  0
},
{ 15474,14312,13843,13669,12397,10171, 9109, 8697, 8063, 6764, 5886, 5470, 5181, 4574,
4090, 3790,
  0
},
{ 15720,14858,14443,14249,13373,11680,10700,10226, 9779, 8747, 7881, 7386, 7156,
6611, 6097, 5712,
  0
},
{ 16133,15786,15587,15472,15104,14369,13854,13555,13314,12738,12214,11861,11701,
11314,10905,10579,
  0
},
{ 2594, 1645, 1633, 1631, 535, 108, 92, 89, 64, 34, 27, 25, 21, 15, 12, 11,
  0
},
{ 7130, 4521, 4444, 4431, 2088, 598, 489, 469, 378, 235, 185, 173, 153, 120, 100,
94,
  0
},
{ 1326, 780, 778, 777, 173, 17, 15, 14, 11, 7, 6, 5, 4, 3, 2, 1,
  0
},
{ 4974, 2832, 2814, 2812, 945, 102, 78, 76, 56, 26, 16, 15, 13, 9, 6, 5,
  0
},
{ 3593, 2051, 2043, 2042, 530, 36, 28, 27, 18, 8, 6, 5, 4, 3, 2, 1,
  0
},
{ 5521, 3055, 3028, 3026, 1052, 102, 72, 69, 49, 20, 12, 11, 9, 6, 4, 3,
  0
},
{ 4294, 2539, 2520, 2518, 812, 83, 61, 59, 41, 17, 12, 11, 9, 6, 4, 3,
  0
},
```

FIG 23(6)

```
{ 4456, 2667, 2653, 2652, 724, 67, 54, 53, 31, 11, 8, 7, 5, 3, 2, 1,
  0
},
{ 6684, 3922, 3872, 3869, 1425, 186, 133, 129, 86, 35, 21, 19, 15, 9, 6, 5,
  0
},
{ 5087, 3024, 2988, 2985, 965, 99, 71, 69, 43, 15, 10, 9, 7, 4, 3, 2,
  0
},
{ 7587, 4403, 4297, 4289, 1806, 279, 181, 171, 118, 49, 30, 27, 22, 15, 11, 10,
  0
},
{ 6242, 4036, 3962, 3957, 1604, 337, 256, 251, 154, 49, 31, 28, 20, 10, 6, 5,
  0
},
{ 3727, 2352, 2346, 2345, 568, 53, 46, 45, 27, 9, 7, 6, 5, 3, 2, 1,
  0
},
{ 6369, 3904, 3876, 3874, 1372, 186, 150, 147, 101, 42, 27, 25, 20, 13, 9, 8,
  0
},
{ 5012, 3060, 3046, 3045, 921, 76, 60, 59, 37, 13, 9, 8, 6, 4, 3, 2,
  0
},
{ 5471, 3590, 3569, 3568, 1118, 176, 153, 151, 83, 26, 19, 18, 13, 7, 5, 4,
  0
},
{ 5866, 3774, 3736, 3733, 1366, 204, 163, 159, 101, 37, 26, 24, 19, 11, 8, 7,
  0
},
{ 8648, 5538, 5437, 5427, 2527, 521, 397, 382, 269, 121, 81, 75, 60, 40, 30, 27,
  0
},
{ 7300, 5124, 5049, 5043, 2284, 682, 582, 574, 341, 124, 90, 86, 59, 29, 20, 18,
  0
},
{ 7183, 4913, 4817, 4808, 2167, 517, 404, 393, 252, 92, 63, 59, 43, 24, 17, 15,
  0
},
{ 4749, 3285, 3273, 3272, 938, 157, 141, 140, 81, 27, 21, 20, 15, 8, 6, 5,
  0
},
{ 6602, 4705, 4676, 4674, 1689, 388, 347, 344, 192, 63, 49, 47, 33, 16, 11, 10,
  0
},
```

FIG 23(7)

```
{ 6849, 4648, 4599, 4596, 1868, 367, 304, 299, 187, 68, 49, 46, 34, 19, 14, 13,
  0
},
{ 16383,16382,14098,13672,13671,13670,11058,10499, 8080, 5456, 4230, 3829, 3412,
2876, 2494, 2264,
  0
},
{ 16383,16382,14436,14062,14061,14060,11574,11038, 8597, 5905, 4621, 4204, 3720,
3106, 2674, 2430,
  0
},
{ 16383,16382,14635,14264,14263,14262,11898,11372, 8910, 6216, 4935, 4506, 3962,
3287, 2835, 2578,
  0
},
{ 16383,16382,14851,14380,14379,14378,12610,12032, 9670, 7609, 6543, 6085, 5335,
4590, 4112, 3825,
  0
}
};
```

2396

FIG 23(8)

```
unsigned short ari_cf_r [4] ={(3<<14)/4,(2<<14)/4,(1<<14)/4, 0};
```

FIG 24